VOLUME 5 FALL 2017

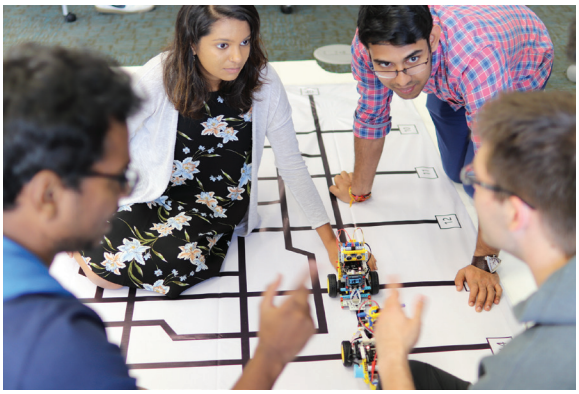
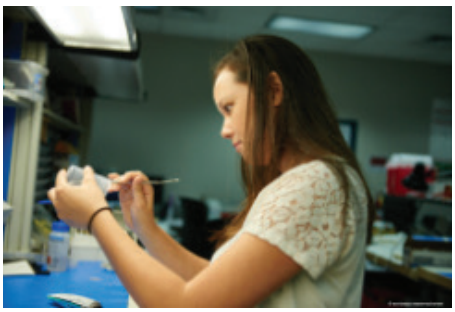
Carnegie Mellon University
The Robotics Institute

Summer Scholars (RISS)

Working Papers Journal





THE ROBOTICS INSTITUTE

Carnegie Mellon Robotics Institute Summer Scholars Working Papers Journal

RISS 2017 Team

Dr. John M. Dolan (Director)

Ms. Rachel Burcin (Co-director)

Ms. Mikana Maeda

Mr. Ziqi Guo

Ms. Xiaohuan Wang

Mr. Terrence Ting

Ms. Christine Downey

Cover Design & Document Layout

Julia Luo & Terrence Ting

We gratefully acknowledge the support of the National Science Foundation through the Research Experience for Undergraduates (REU) program (Grant # IIS 1659774).

Through the Summer Scholars Program, the Robotics Institute hosts the NSF Robotics & Intelligent Systems REU Site.

The Robotics Institute Summer Scholars Working Papers Journal is an annual publication of the Robotics Institute's Summer Scholars Program at Carnegie Mellon University.

TABLE OF CONTENTS

TABLE OF CONTENTS CONTINUED

TABLE OF CONTENTS CONTINUED

TABLE OF CONTENTS CONTINUED

ABOUT RISS

Carnegie Mellon's Robotics Institute Summer Scholars (RISS) program (http://riss.ri.cmu.edu/) is an eleven-week summer undergraduate research program that immerses a diverse cohort of scholars in cutting-edge robotics projects that drive innovative and have real-world impact.

Launched in 2006, RISS is among the best and most comprehensive robotics research programs for undergraduates in the world. The quality and breadth of research, high-level of institute and university engagement, extensive professional development curriculum, graduate school application counseling, and alumni network create transformative experiences and remarkable postprogram trajectories.

The Robotics Institute at Carnegie Mellon University is the largest university affiliated robotics research group in the world. It offers a remarkably diverse breadth of research with an extensive range of applications. With hundreds of active research projects, together with both graduate and undergraduate programs, the Institute is a global leader in robotics research, education, and innovation. The Institute has the nationally recognized research capacity, educational programming, and student development experience to provide, through the RISS program, high-quality research experiences and a developmentally appropriate professional development curriculum to a diverse cohort of undergraduate students.

RISS | ROBOTICS INSTITUTE SUMMER SCHOLARS

ABOUT RISS CONTINUED

The Robotics Institute Summer Scholars Program:

1. Immerses a highly motivated and diverse cohort of students (hereafter referred to as “scholars”) in a guided research process,

2. Challenges scholars to build an understanding of research philosophy that serves as a basis for creative problem-solving that transcends the summer research experience,

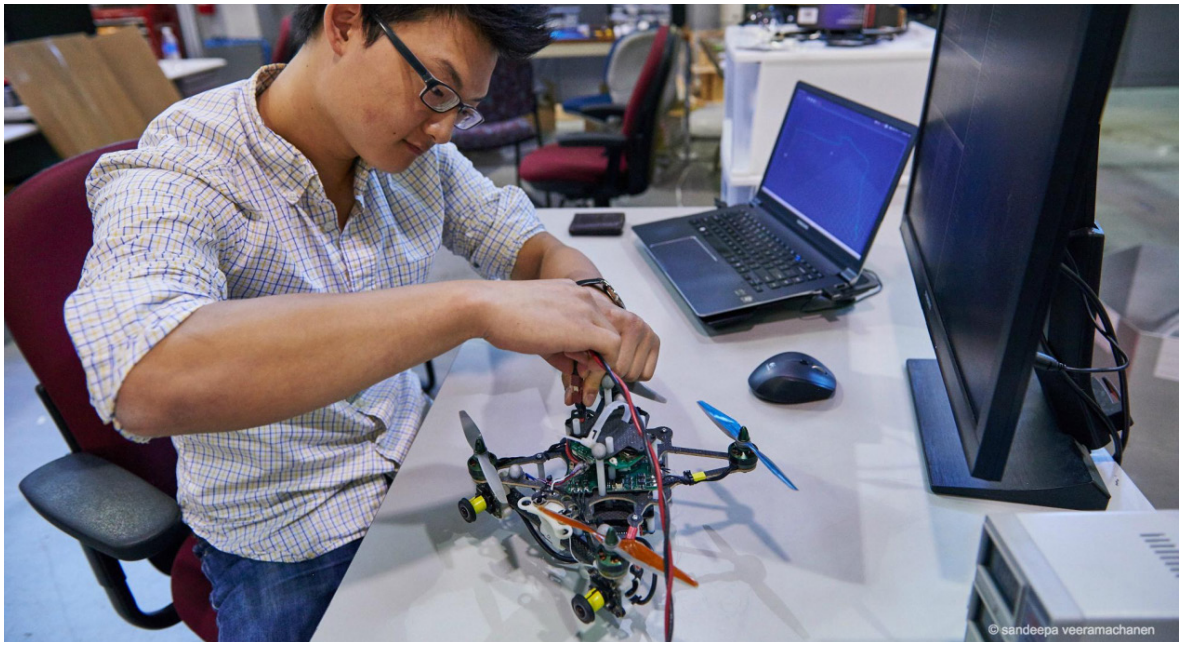
3. Introduces scholars to the highly interdisciplinary nature of robotics and the vast potential to impact and improve the world’s quality of life,

4. Provides professional development components that prepare students for successful transitions to graduate school and research careers,

5. Engages scholars in reflective service learning experience that imparts the value of broadening participation and engagement in STEM fields,

6. Inspires scholars to pursue careers in robotics and related fields, such as graduate programs, and equips them with new skills and knowledge, and

7. Helps scholars to build collaboration and lifelong connections with a dynamic global community of robotics researchers and entrepreneurs.



CONGRATULATIONS 2017 COHORT!

Jahdiel Alvarez	University of Puerto Rico, Mayaguez
Jonathan Alvarez	University of North Carolina at Chapel Hill
Gabriel Arpino	University of Toronto
Fay Blelloch	Swarthmore College
Eli Bronstein	University of California, Berkeley
Gerardo Consuelos	Instituto Tecnologico y de Estudios Superiores de Monterrey
Mihael Cudic	University of Florida

2017 COHORT CONTINUED

Mosam Dabhi	Sardar Vallabhbhai National Institute of Technology Surat
Kshitij Goel	Indian Institute of Technology, Kharagpur
Rachel Gologorsky	Harvard University
Deepthi Hegde	National Institute of Technology Karnataka, Surathkal
Conor Igoe	University College Dublin
Roman Kaufman	Carnegie Mellon University
Anjana K N	National Institute of Technology Karnataka, Surathkal
Kevin Chong-Loo	Instituto Tecnologico y de Estudios Superiores de Monterrey
Rebecca McManus	University of Vermont
Stephanie Milani	University of Maryland, Baltimore County
Aisha Mohamed	Carnegie Mellon University in Qatar
Daniele Moro	Boise State University
Kyle Morris	University of Manitoba
Ingrid Navarro	Instituto Tecnologico y de Estudios Superiores de Monterrey
Dat Nguyen	De Anza College
Kartik Patath	Visvesvaraya National Institute of Technology
Rohith Pillai	Carnegie Mellon University in Qatar
Vinitha Ranganeni	Carnegie Mellon University
David Russell	Clarkson University
Mayank Saxena	Delhi Technological University
Samuel Shaw	Tufts University
Yunfei Shi	The Hong Kong Polytechnic University
Griffin Tabor	Worcester Polytechnic Institute
Fatma Tlili	Carnegie Mellon University in Qatar
Tevon Walker	University of Alabama, Huntsville
Matthew Wilson	University of Utah
Chun Ming Jeff Zhang	Carnegie Mellon University
Tejas Zodage	Birla Institute of Technology and Science Pilani - Gao Campus

NOTE FROM THE 2017 COHORT

Dear Future Scholars,

The Carnegie Mellon Uiversity Robotics Institute Summer Scholar program offers a unique opportunity for talented undergraduates to interact with leading experts in robotics and contribute to real-world cutting-edge robotics research projects. The eleven-week full-time program truly allows students to dive deep and make a significant impact with their research. The guidance and mentorship throughout the summer allows hard-working undergraduates the invaluable opportunity to publish their research in the peer-reviewed RISS Working Papers Journal as well as to present it to the entire Robotics Institute community. This opportunity to present and to publish prepares one for graduate school, and many of the RISS graduates have refined their summer research into papers accepted by well-known research journals and have continued on to top PhD programs.

Indeed, the fantastic opportunity to present and publish is aligned with the RISS program's overall goal of fostering individual growth and providing undergraduates with the opportunity to shine. Throughout the summer, the RISS program organizers expend tremendous effort in helping their students achieve their full potential by providing professional mentoring and exposing them to cutting-edge research. Every week, students have the opportunity to visit each other's labs and learn about different research areas, to attend lectures by leading experts, and to improve their technical writing and poster design skills in both group workshops and individual appointments with professional staff. Students also have many opportunities to take on a leading role and create events for themselves. For example, we were both part of the RISS Summer Working Paper's Journal Board and took charge of the peer-review process, and our peers were involved with initiatives such as Robotics for Good, and took leading roles in organizing fun events such as a drone workshop, a good-natured robotics-build competition, and an industry night with cool startups.

A NOTE FROM THE 2017 COHORT

In addition, the RISS program fosters a strong sense of community. We worked together, learned together, and played together. We painted the fence and visited amusement parks, held spontaneous weekend get-togethers, and took headshots for each other. We had the time of our lives, and made lasting friendships.

In conclusion, the unparalleled opportunities to interact with world-class robotic researchers, to forge friendships with talented peers, to receive individual attention and guidance from mentors dedicated to our growth, and to actually get our hands dirty by doing cutting-edge research, make the Robotics Institute Summer Scholars program truly exceptional.

On behalf of the 2017 Cohort, we hope you apply, and wish you good luck!

Sincerely,

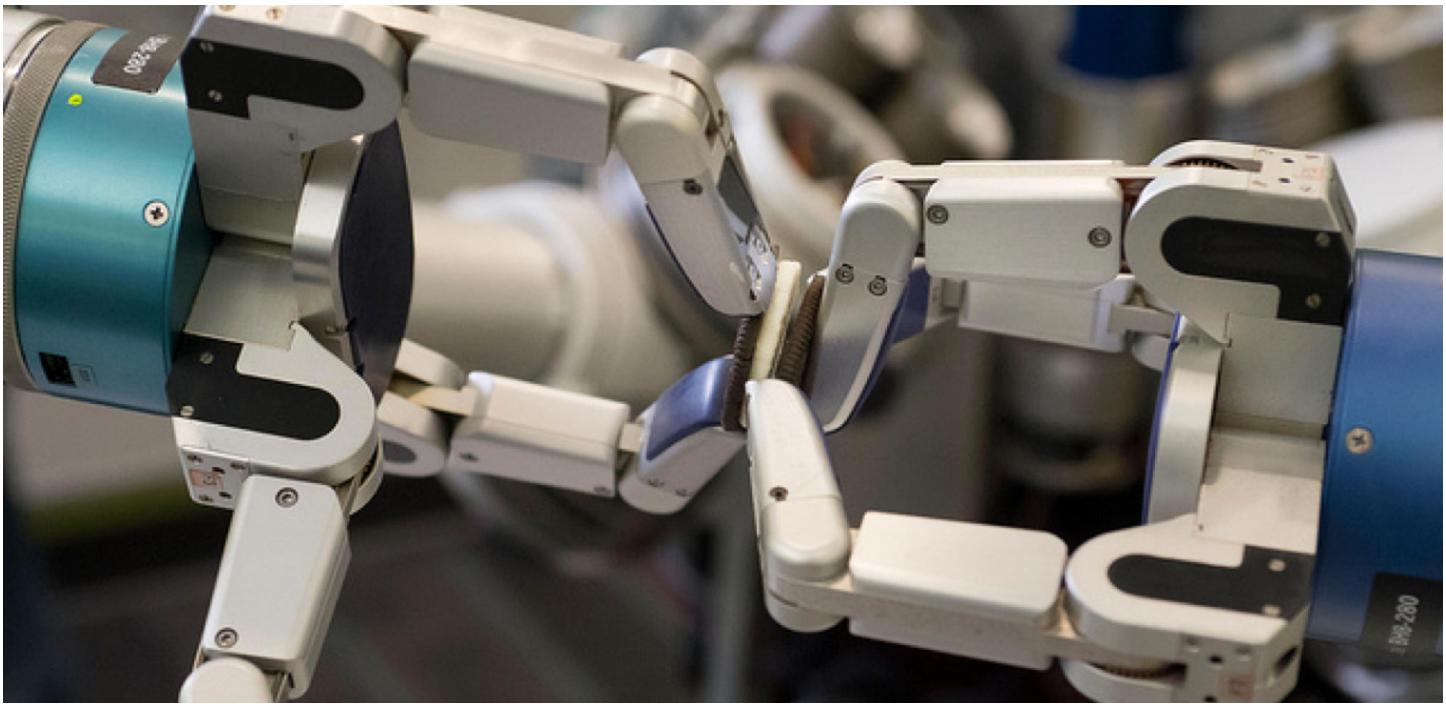
Rachel Gologorsky and Stephanie Milani

THANK YOU RISS MENTORS!

Carnegie Mellon University
The Robotics Institute

At the core of the Robotics Institute Summer Scholars' program are incredibly talented and dedicated faculty, graduate students, staff, and RISS alumni.

We are incredibly thankful for their support, participation, leadership, and vision that make this one of the best research experiences in robotics and intelligent systems in the world.



2017 MENTORS

Sankalp Arora
James Bern
David Bourne
Alexander Cebulla
Karen Chen
Howie Choset
Stelian Coros
Fernando De La Torre Frade
Ruta Desai
Vishnu Desaraju
John Dolan
Chiyu Dong
Geetesh Dubey
Artur Dubrawski
Martial Hebert
Ralph Hollis
Dong Huang
Isaac Isukapati
George Kantor
Azarakhsh Keipour
Jennifer King
Lu Li
Maxim Likhachev
Carmel Majidi
Christoph Mertz
Nathan Michael
James Kyle Miller
Omeed Mirbod
Jack Mostow
Katharina Muelling
Sasanka Nagavalli
Venkatraman Narayanan
Luis E. Navarro-Serment
Stephen Nuske
Jean Oh
Hamid Ossareh
William Paivine
Stanislav Panev
James Picard
Mabaran Rajaraman
Theresa Richards
Hadi Salman
Oren Salzman
Guillaume Sartoretti
Sebastian Scherer
Puneet Singhal
Stephen Smith
Alexander Spitzer
Arun Srivatsan
Siddhartha Srinivasa
Shuang Su
Katia Sycara
Matthew Travers
Jinhang Wang
Xuning Yang
Yihuan Zhang

THANK YOU TO OUR SPONSORS

NSF Award Number IIS 1659774
P.I. Dr. John Dolan

Fay Blelloch
Eli Bronstein
Rachel Gologorsky
Roman Kaufman
Daniele Moro
Vinitha Ranganeni
David Russell
Samuel Shaw
Griffin Tabor
Tevon Walker

Mosam Dabhi
Kshitij Goel
Deepthi Hegde
Kartik Patath
Tejas Zodage

Stephanie Milani

THANK YOU TO OUR PARTNERS

Carnegie Mellon University

Carnegie Mellon
SCHOOL OF COMPUTER SCIENCE

Carnegie Mellon University Qatar

Tecnológico de Monterrey

NREC
EXPERIENCE · EXPERTISE · EXCELLENCE

UBTECH EDUCATION

dji
THE FUTURE OF POSSIBLE

Leonard Gelfand Center
Service Learning & Outreach

CMU Office of the Assistant Vice Provost for Graduate Education

CMU Go Research Program

CMU Global Communication Center


MIAMI
HEAT


© sandeepa veeramachanen

RISS Working Papers

Congratulations to all the scholars for their hardwork and contributions!

Traffic Sign Inspection System Through Faster-RCNN Detection and Optical Flow Based Tracking

Jahdiel Alvarez[1] and Fatma Tlili[2]

***Abstract*— Traffic sign inspection and evaluation is mostly done manually by human inspectors. To automate this process, traffic sign detection and classification is an essential step. While most approaches to detect traffic signs deal with the shape and color of the signs, these approaches are sensitive to noise and difficult to find the right segmentation thresholds for. Other studies were more oriented to use deep learning methods, however these approaches remain computationally expensive. We present a traffic sign recognition system that combines the state-of-the-art object detection Faster-RCNN architecture and Optical Flow based tracking using the Lucas-Kanade method. Our approach consists of two main phases: the detection phase and the tracking phase. The detection phase is done by extracting the feature map from the image, which is followed by proposing the regions of interests using an RPN and finally classifying the signs using a regressor. The tracking phase consists of extracting feature points using different methods. The points are then tracked using optical flow, which allows the prediction of the bounding box of the traffic sign in the new frame. By combining the detector and the tracker we reduce the need for detection on every frame, thus making a more efficient framework.**

I. INTRODUCTION

In the United States, the Federal Department of Transportation (DOT) is responsible for ensuring safe, fast and efficient transportation systems. Their task consists of inspecting the city's transportation infrastructure, which includes monitoring the traffic signs state to assure that it meets federal standards. This monitoring is mostly conducted by human inspectors, which is an expensive and labor-intensive process. In order to automate the traffic sign inspection procedure, a traffic sign detection and classification system is required.

The state of the art traffic sign detection approaches are divided into two main directions: model-based approaches and learning-based approaches. The model-based approaches exploit the distinctive colors and shapes of traffic signs to be able to locate them in the image. While these approaches work well when the traffic signs are under good conditions, they become more sensitive when the traffic signs' state changes. For instance a change in the reflectivity or the illumination, a fade in the color, or an occlusion of the signs due to vegetation. In addition these approaches generally require many thresholds for the image segmentation which can be hard to set. The learning based approaches such as machine learning and deep learning methods became more prominent recently as they were able to detect the traffic signs with high accuracy. However these methods remain computationally expensive. In addition, both these approaches are able to detect traffic sings on individual frames, yet they aren't able to group the detections of a specific traffic sign over time and recognize it as a single traffic sign, which would be useful for the purpose of the inspection of the traffic sign.

We propose a traffic sign inspection system that uses time stamped images taken from smartphones to efficiently detect the traffic signs and track them overtime. These detections can then be tagged with GPS locations and either be displayed to inspectors or automatically evaluated. Our ultimate goal is the integration of this system with a larger transportation infrastructure project described in [16].

Our approach addresses the challenges stated earlier in two main steps. The first step is to detect and classify the traffic signs given a frame of a video using the Faster R-CNN neural network [15]. The second step is to track the detected signs for the next few frames using optical flow estimation.

By fine tuning Faster-RCNN we aim to achieve high accuracy in the detection of traffic signs. In comparison with other approaches that implement the detector and the classifier such as this Faster-RCNN only approach [21], our system is able to detect traffic signs with good accuracy. By applying tracking in the next consecutive set of frames we aim to reduce the total amount of computation needed per frame, as the tracking method is less expensive than the detection method. Our system also allows us to exploit the temporal information and recognize a traffic sign to be the same over a set of frames.

The outline of the rest of this paper is as follows. Section II explains the related work of the detector and the tracker respectively. Section III describes the data set that we used, section IV and V describe our approaches, section VI reports our experiments and results and section VII summarizes the system and emphasizes our future work.

II. RELATED WORK

A. Traffic Sign Datasets

As specified in [13], the traffic sign detection problem has been researched for about a decade, but most of the innovations in the field have been made with European datasets. The German Traffic Sign Detection Benchmark (GTSDB) [6] and the Belgium Traffic Sign Detection set (BTSD)[11] are the most influential datasets in the traffic sign detection field and in their respective challenges learning-based methods have obtained almost perfect mAP, but the problem cannot

This work was supported by Carnegie Mellon University's NAVLAB

[1]J. Alvarez is a Software Engineering student at University of Puerto Rico, Mayaguez.

[2]F. Tlili is a computer science student in Carnegie Mellon University, Qatar.

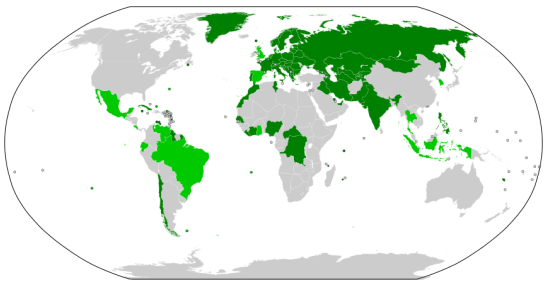


Fig. 1. Countries which have signed and/or ratified the Vienna Convention on Road Signs and Signals. Data Source: [14]

Fig. 2. US traffic signs: (left) Keep Right, (middle) speed limit, (right) turn right

Fig. 3. Vienna Convention signs: (left) Keep Right, (middle) speed limit, (right) turn right

be considered solved, just yet. The traffic signs in both, the GTSDB and the BTSD datasets are compliant with the Vienna Convention on Road Signs and Signals [14], which was an intent from the United Nations to have a unified transportation signage standard, but only 65 countries have ratified such accord, meaning that a significant number of countries do not abide by such standards (Fig. 1) resulting in very different traffic signs, as is the case for US road signs.

Similarity between corresponding traffic signs is important for a detection framework to work properly. Therefore one cannot assume learning-based methods, trained on traffic signs which come from the Vienna Convention standard, to work properly on US traffic signs. The differences can be evidently perceived from the corresponding sign samples in Fig. 2 and Fig. 3. Vienna Convention signs are much more noticeable than their US counterpart, therefore detectors would at least need to be fined-tuned in order to learn these new road signs.

Detection on US traffic signs are very scarce and most focus on speed limit signs. As far as we know this is the first detection framework which utilizes deep learning to detect multi-class US traffic signs. Our model is trained on the LISA-TS Extension Dataset [12], which at time of publication was the largest public US traffic sign dataset.

B. Faster R-CNN

Object Detection through deep learning is a continuously evolving area of research in the computer vision field. Since the development of Region-based Convolutional Neural Networks (R-CNN) [4] the mean average precision has increased drastically. In CVPR 2014, the R-CNN framework increased the mean average precision (mAP) from 40.9% to 53.3% in the PASCAL VOC 2012 [2] challenge. This architecture utilized a multi-stage training pipeline which used region proposal algorithms (e.g Selective Search [17], Edge Boxes [20]), a CNN for feature extraction, and multiple SVMs for classification and regression tasks.

As the field progress a more unified network, with shareable weights was developed in the form of Fast R-CNN [3]. Fast R-CNN introduced the concept of ROI Pooling layers which framed each region proposal as an input feature map and trained a fully connected layer network with it. This network was split into to two sibling layers, one which outputs a discrete probabilistic distribution over the number of classes in the training set and the second outputs bounding-box regression offsets, as specified in [3]. The training time with VGG16 network is 9x faster than R-CNN. The Faster R-CNN architecture proves to be much more efficient as such a pipeline provides an end-to-end training model, instead of multi-stage training and achieves a higher mAP (66% in PASCAL VOC 2012).

Once Fast R-CNN established a functioning end-to-end training network, the bottleneck for computation speed became the external region proposal algorithms. Such external proposal algorithms were the norm for R-CNN and Fast R-CNN, and did not provide any influence in the training process. Therefore the Faster R-CNN [15] architecture was developed to resolve this problem. In Faster R-CNN, Region Proposal Networks (RPN) were proposed based on the idea of learning the way in which regions are predicted and having a unified end-to-end network (Fig. 4). It Included the RPN into the Fast R-CNN model which improved the performance of the model in terms of computing speed. This improvement resulted in a testing time of 200ms per image, proving much faster than previous region-based convolutional models, while maintaining a superior mAP for renounced object detection challenges (MS COCO 2015 [9], ILSVRC 2015 [5], PASCAL VOC 2012).

C. Traffic Sign Tracking

While traffic sign detection is a topic that has been researched for about a decade, traffic sign tracking is still a fairly new topic and less effort has been oriented to solve this problem.

This paper [1] uses the most common approach in tracking the traffic signs. That is the Kalman Filter approach. KF takes a series of data measurement over time, possibly containing noise and inaccuracies, and it predicts the missing values with better precision as more data is being fed into the model.

In the case of using Kalman Filter to predict the traffic sign location in the mentioned paper, a linear model of evolution is assumed with respect to a given state vector. The state vector that measures the position, size and velocity of a track, coming from the detector, is fed into the model in every iteration. The KF model then predicts which detection

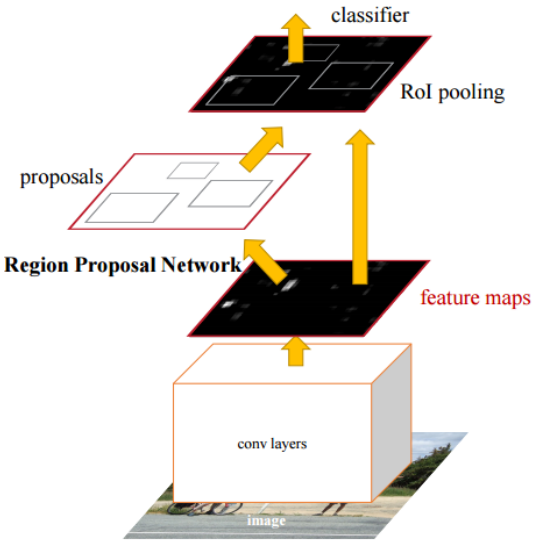


Fig. 4. Faster R-CNN architecture. Source: [15]

belongs to the measured state in the new frame by predicting the new location and the size of the sign. It then updates the measured state with the corrected version. This information is then sent to the detector to find the new candidate positions of the traffic sign in the next frame.

This paper [12] also combines the detection and the tracking of traffic signs. It tackles the tracking problem in a similar approach by using the Hungarian algorithm for the assignment and Kalman filter for tracking. The detector returns a set of detections for a given frame. Each of these detections is either assigned to an existing track by the Hungarian algorithm or a new track is created for it. If an existed track is not seen for 3 consecutive frames, it is considered a false positive by the detector. This paper aims to use tracking to suppress the false positives returned by the detector in every iteration. This means that the detector is applied to every single frame.

Another method of tracking that has been adopted is the one explained in this paper [10]. The tracking method is a variation of the condensation algorithm that exploits the colors in the traffic signs. The tracking is done by first randomly generating N bounding boxes that are in the neighborhood of the detected traffic sign at frame t taking into account the velocity of the movement of the traffic sign. The bounding box that has the most similarity with the detected traffic sign is then chosen to be the prediction of the traffic sign at time t+1.

In this paper, [18] object tracking is described to be divided to three main approaches: Point Tracking, Kernel Tracking and Silhouette Tracking. In point tracking objects are tracked by finding the correspondence of points across frames. Kernel tracking focuses more on estimating the motion of the object from one frame to the next. Silhouette tracking focuses on finding a good shape representation of the object and tracking it using the object model generated in from the previous frame.

US Traffic Sign Super Classes			
Warning	Stop	No Turn	Speed Limit
1232	1181	184	752

TABLE I
AMOUNT OF IMAGES PER SUPERCLASS IN THE LISA-TS EXTENSION DATASET

III. LISA-TS EXTENSION DATASET

The LISA-TS Extension Dataset [12] is a dataset composed of US traffic signs only, unlike most renounced traffic sign dataset which provide data of Vienna Convention standard complying traffic signs. It is called the Extension because it is an extension to the original LISA-TS [13] dataset, the original dataset did not have uniform image resolution and included grayscale images. There are 31 labeled classes, which are divided into four super classes: warning, stop, no turn, speed limit. Table I shows the distribution of traffic signs per superclass, revealing that the class with the most data is warning and the class with the least data is no turn signs. This might be expected given the rarity of both classes in US roads and in the dataset (Fig. 5), warning superclass captures much more classes than the no turn superclass.

All 3307 images in the LISA-TS Extension dataset were taken with a Point Grey FL3 color camera at a 1280x960 resolution around San Diego, California on the spring of 2014. As detailed in [13] the statistics of the dataset can provide important information on how the Faster R-CNN network we used trained on the dataset. As seen in Table II the traffic signs sizes can vary from 23-222 px, being approximately 50 px the average height of all signs, making the detection problem slightly harder given that the objects are relatively small. From the statistics we calculate that the average amount of signs per image is 1.11. The images in the dataset can be considered high resolution with a resolution of 1280x960 px.

The stop, no turn, and speed limit classes have a fixed or very similar shape, text, and color without much variation from within the class. This helps the network converge faster and improves accuracy given that it can learn exact features for a specific class since variations are minimal. In contrast as seen in Fig. 5 the warning class encapsulates a great variety of US road signs, many of which have little to no resemblance from each other (Fig. ADD NUM). This causes great difficulty for the network to predict warning signs with good accuracy. A more meaningful separation of the internal classes, at least of the warning superclass, could help achieve better results when training Convolutional Neural Networks (CNN) on this dataset.

IV. TRAFFIC SIGN DETECTION APPROACH

Traffic sign detection is a problem which has been tackled for a series of years and very high accuracy has been achieved on the benchmark datasets, but a focus in US traffic signs is scarce. Deep learning has become the state-of-the-art in image recognition tasks and a fundamental instrument in computer vision. Although, this is true from our research

LISA-TS Extension Set	
Number of classes:	**31**
Number of annotations:	**3672**
Number of images:	**3307**
Sign sizes, longest edge:	**23 -222 px**
Images sizes:	**1280x960 px**

TABLE II

SIGN STATISTICS IN THE LISA-TS EXTENSION DATASET

warning	addedLane	**speedLimit**	speedLimit15
	curveLeft		speedLimit25
	curveRight		speedLimit30
	dip		speedLimit35
	intersection		speedLimit40
	laneEnds		speedLimit45
	merge		speedLimit50
	pedestrianCrossing		speedLimit55
	roundAbout		speedLimit60
	signalAhead		speedLimit65
	slow		
	speedBumpsAhead		
	stopAhead		
	thruMergeLeft		
	thruMergeRight		
	turnLeft		
	turnRight		
	yieldAhead		
	warningUrdbl		
noTurn	noLeftAndUTurn		
	noUTurn		
	noLeftTurn		
	noRightTurn		

Fig. 5. Classes in each super class. Source: [13]

Fig. 6. Inner class differences in the Warning superclass. The difference between such signs increases the difficulty of the classification task.

there are very few implementations of using a deep learning framework to detect US traffic signs in images. In [8] a Fast R-CNN architecture is used but, the implementation only focuses on speed limit signs and not in a multi-class detection problem, as done in this paper. Our approach focuses on using deep learning's Faster R-CNN to detect four super classes composed of US traffic signs. An implementation as such is lacking in the researching field.

A. Model Description

In [7] an extensive analysis of the speed/accuracy trade-offs between convolutional object detectors was done. Some interesting results were obtained, most specifically experiments demonstrated that the Faster R-CNN meta-architecture produce higher accuracy from all the other meta-architectures (SSD, R-FCN) even though the training and test time are slower. Given our project accuracy is the most important attribute of our network, given that all processing should be done off-line, therefore Faster R-CNN was network of choice. Inside the the Faster R-CNN architecture we used ResNet-50 as the base convolutional network. In [7] ResNet-101 produced on average the second highest mean average precision (mAP), just below the Inception Resnet V2 model. ResNet-50 was selected for the fact that there was access to the Pascal VOC weights for that particular model and has the same framework as ResNet-101. Might be the case that using ResNet-101 or Inception Resnet V2 [7] models provide a higher mAP than the results registered by this paper. In our case, 300 region proposals were provided instead of the 2000 like in [15]. This change as stated in [7] does not affect mAP significantly.

B. Training the Model

The model was trained on the LISA-TS Extension Dataset.The dataset is composed of only 3307 images, therefore the dataset was split into two parts: a training set and a validation set. The training set was composed of 80% of all the images, those images were selected at random, and the images of the validation set were the remaining 20%. Our testing sets were composed of the LISA-TS Original Dataset and Robotic Institutes's NavLab's Stop Sign dataset; the results of the validation set are also presented.

When training our model we experimented on fine-tuning the network with two sets of weights. We experimented with learned Imagenet weights,for the ResNet-50 base convnet only,and with learned PASCAL VOC weights, which were for the whole network including the RPN and the classifier of our model.

None of the images were pre-processed and no data augmentation was done throughout the training and testing process. The training was done on the images as is, although experimenting on pre-processing techniques such as Contrast Limited Adaptive Histogram Equalization (CLAHE) might increase performance, although probably not by a significant amount. The deeper problems of this network are not encountered by the clarity of the signs in the images, but instead by the traffic sign sizes and the size of the receptive field on the last feature map of the convolutional network. In terms of data augmentation, this is definitely part of future works, because of the size of the Extension dataset. The 3307 images in the dataset make it to be considered an extremely small set. Due to the computational limitations, work with data augmentation proved inefficient.

C. Detection Experiments

The training was done on a single NVIDIA GTX 1070 GPU with 8GB of memory. The CPU also had 8GB of memory and an Intel i7 processor. The computational capacities were low therefore fine-tuning an already existing network was the best solution.

1) Fine-tuning and Transfer Learning: Fine-tuning on top of the Imagenet weights for ResNet-50 we trained for 5 epochs with a learning rate of 1e-4 and obtained a 29% precision on the validation set. We repeated the training but for this attempt the whole base convnet was frozen, this was experimented in order to minimize training time but the precision on the validation set decreased to 20%, reiterating

Frozen Layers in Base Convolutional Network	
No frozen layers	**29%**
1st conv layer	**39%**
1st conv layer and 2nd stage block	**39%**
1st conv, 2nd and 3rd stage blocks	**43%**
All frozen except last 4 conv layers	**31%**
All frozen	**20%**

TABLE III

PRECISION OBTAINED BY FREEZING CONV-BLOCKS IN RESNET-50

the fact that fine-tuning proves much more efficient than plain transfer learning as shown in [19].

Although plain transfer learning lowers performance, freezing some early layers might lower the computation time while still producing better results. Given that the early layers of convolutional networks focus on low level features these are very similar from network to network therefore freezing such layers should not have a negative impact in performance. In contrast, freezing the whole network means that the higher level features (at the end of the network) which find correlations between lower level features to predict the training classes are not being trained and these features the deeper they are the more class specific they become, therefore when used to predict new classes it has not seen before it lowers in performance. In our case, we experimented on how far we could freeze the layers without having an impact on precision to achieve faster processing time. We trained for 5 epochs each time using 1000 images per epoch (batch-size) with a learning rate of 1e-4 using the Adam optimizer.

As shown in Table III by freezing the the 1st convolutional layer the precision on the validation set increased from 29% to 39% precision. Next, freezing the 1st convolutional layer and the ResNet-50 stage two block precision maintained itself at 39%. Freezing the 1st convolutional layer, the stage two and stage three blocks resulted in a 4% increase to give a precision of 43%. Lastly, freezing all the layers except the last 4 convolutional layers decreased the precision by 12%, this because the weights at this stage are probably already to specialized on the original Imagenet classes (given they are Imagenet weights) and will hence need to be fine-tuned for the US traffic sign classes.

2) Images per Epoch: The amount of images per epoch we utilized on previous training experiments was 1k images per epoch. This was changed to 2232 images per epoch, iterating through approximately all the images in the training set for each epoch. We trained for 5 epochs, this time on top of the Pascal VOC weights, which were pre-trained on the whole network including the classifier and the RPN, and obtained 47% precision. Therefore 2232 images per epoch became the amount we used for the rest of the paper.

3) Faster R-CNN Input Image Size: In the dataset the sizes of the traffic sign are reasonably small, with an average height of approximately 50px. As shown in [7], modern convolutional object detectors do not perform well with small objects. Therefore, in our case this became a limiting factor

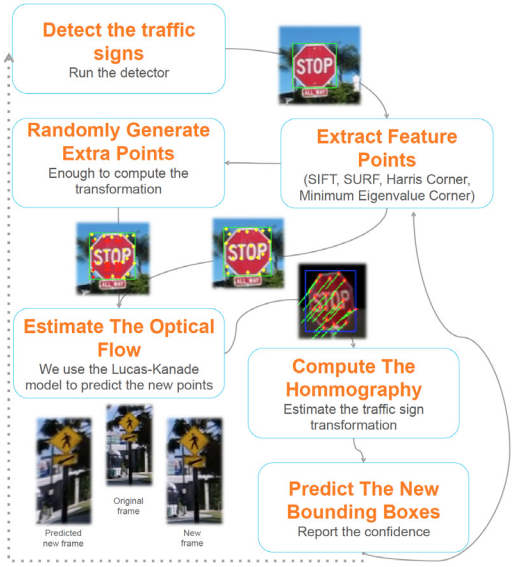


Fig. 7. Workflow for the tracking system

in the performance of the network when utilizing the Faster R-CNN standard parameters. In [15] the input images are resized to be 600x600px, but this parameter was chosen based on the Pascal VOC dataset, in which the object sizes are much more prominent in the images. For which they could get reduce the size quite drastically without affecting the receptive field in a significant matter. For this network and dataset the receptive field becomes to large at the last feature maps and more information from the surroundings of the traffic signs get into the traffic sign's feature because the signs are small in size.

We experimented by changing the images resizing parameters. From 600x600px to 900x900px was utilized and this change resulted in a 10% increase in performance, going from 47% to 57% performance. This parameters where changed again to 960px, which is the original height of the image, and the performance increased by 9%, going from the previous 57% to 66% precision.

4) Trained model: The trained model used ResNet-50 as the base convolutional network with the 1st convolutional layer, 2nd and 3rd stage blocks frozen. The input images are resized to 960x960px and an epoch was composed of 2232 images. Training for the 5th epoch weights produced 66% precision on the validation set and the 10th epoch weights gave 72% precision. Testing on the 40th epoch weights resulted in 75% precision and due to computational limitations we trained until the 53rd epoch and obtained the highest mean average precision with 80%.

V. TRAFFIC SIGN TRACKING APPROACH

Optical flow is a way of estimating the motion of every pixel of a frame with respect to the previous frame. This motion could be is caused by the movement of the object or the camera. In our case as the camera moves, the traffic signs position in the frame changes with respect to time. Therefore our approach consists of:

- Detecting the traffic signs in a certain frame using the detector described earlier.
- Extracting feature points from within the bounding box.
- Using Optical Flow Estimation to predict the new bounding box location of the new traffic signs.

The second and third steps are repeated over a set of frames until the tracking confidence drops below a threshold after which using the detector again becomes necessary.

A. Feature points extraction

After running Faster R-CNN to detect the traffic signs in a given frame, we use the generated bounding boxes to extract feature points (source points) from within these boxes by using the following different methods:

- Minimum eigenvalue corners
- Harris corners
- SIFT
- SURF

These feature points are also complemented by some randomly generated points from within the bounding boxes, so that the total number of source points is enough to compute the transformation.

B. Optical Flow Estimation

The second step is to compute the optical flow for the extracted feature points in order to predict their new position in the new frame.

By adopting the optical flow approach, we accept two main assumptions:

- Neighboring pixels have similar motion.
- The intensity of a pixel of the tracked object is constant between two consecutive frames.

$$f(x, y, t) = f(x + dx, y + dy, t + dt) (1)$$

Where $f(x, y, t)$ is the intensity of a the pixel with coordinates (x, y) at time t. From equation (1) we apply Taylor Series to obtain the following Optical Flow Equation:

$$f_x u + f_y v + f_t = 0 (2)$$

Where

$$fx = \sigma f/\sigma x \textit{ is the gradient with respect to } x$$

$$fy = \sigma f/\sigma y \textit{ is the gradient with respect to } y$$

$$ft = \sigma f/\sigma t \textit{ is the gradient with respect to time}$$

$$u = dx/dt \textit{ is the horizontal optical flow}$$

$$v = dy/dt \textit{ is the vertical optical flow}$$

There are many methods that could be used to solve for u and v. We use the Lukas-Kanade method for our purposes. The Lucas-Kanade method of estimating the Optical Flow uses the first assumption and states that all 9 pixels of the 3x3 box around a pixel of the image have the same optical flow. Therefore we end up with 9 equations to solve for the two optical flow variables u and v. Using the Least Square Fit we end up with the following solution for the Optical Flow Equation:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum_i {f_{x_i}}^2 & \sum_i f_{x_i} f_{y_i} \\ \sum_i f_{x_i} f_{y_i} & \sum_i {f_{y_i}}^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i f_{x_i} f_{t_i} \\ -\sum_i f_{y_i} f_{t_i} \end{bmatrix}$$

C. Bounding boxes prediction

Once we computed the optical flow vector for each feature point, we add to each source point its corresponding vector to get its destination point.

To estimate the new bounding boxes, we compute the homography H that transforms the source points to the destination points for each traffic sign separately. These transformations (H) are then applied to the bounding boxes to get the new bounding boxes in the new frame.

This procedure is repeated until the tracking confidence drops below a certain threshold. At that point the detection phase is repeated again.

Fig. 7 shows the workflow of our tracking system illustrated with example images. The bottom left part of the figure shows an example of applying the calculated homography to the image of the traffic sign (in the original frame) and it compares the prediction of the traffic sign in the new frame with the actual traffic sign in new frame.

TABLE IV
TRACKING PRECISION PER VIDEO

Video	Min EV(%)	Harris (%)	SIFT (%)	SURF (%)
1	84.72	87.56	90.48	81.64
2	81.63	85.97	93.02	84.21
3	88.56	88.98	89.69	87.43
4	92.05	91.49	92.27	84.12
5	87.10	86.84	89.25	87.91
6	91.07	90.57	91.32	87.69
7	87.14	83.92	88.08	73.91

VI. RESULTS

A. Detection Evaluation

For testing the network two datasets were utilized. The first was LISA-TS Original dataset and the second was NavLab's Stop Sign dataset. The mean average precisions at 0.4 IOU for both datasets were 0.924 for LISA-TS Original and 0.954 for NavLab's Stop Signs. Given that the LISA-TS Original set contained so few examples from the no turn signs, its average precision was not considered. Table V and Table VI shows the results for both testing sets.

TABLE V
DETECTION RESULTS FOR OUR NETWORK

Dataset	mAP@.40IOU	mAP@.50IOU
LISA-TS Original	**0.924**	0.919
NavLab's Stop Signs	**0.954**	0.887

Fig. 8. Multiple detections in one frame

TABLE VI
DETECTION RESULTS FOR PER CLASS

Dataset	AP Stop	AP Warning	AP No turn	AP Speed limit
LISA-TS Original	0.874	**0.943**	-	0.940
NavLab's Stop Signs	**0.887**	-	-	-

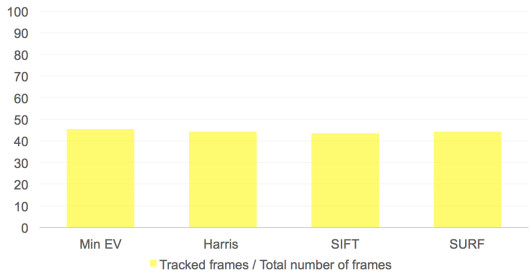


Fig. 9. Tracked frames / total number of frames

B. Tracking Evaluation

We analyzed the performance of the tracking module on 7 videos of the LISA dataset. The videos have a total of 301, 123, 494, 370, 463, 542 and 297 frames respectively with a resolution of 704x480. The tracker was able to process the frames at an average speed of 16 fps.

To evaluate the tracker, we consider the average precision metric defined as follows:

$$AP = \frac{TP}{TP + FP}$$

Where

AP *is average precision*

TP *is true positives*

FP *is false positives*

Table VI shows the average precision of the tracker on the videos. The SIFT feature points perform best in tracking the traffic signs compared to the other features. That is because SIFT feature points are invariant to uniform scaling, orientation and illumination changes and these aspects are specifically important in traffic sign tracking.

We also compute the total number of frames on which using the detector was necessary and compare it to the total number of frames on which only tracking was performed. Fig.7 is a chart that shows the percentage of frames on which tracking was performed. We can see that the total number of frames on which we use the detector drops by around half. This reduces the total amount of computation required and it makes our systems more efficient. However the images provided in the LISA dataset are at least 5 frames apart. Since the tracker assumes that the changes between two consecutive frames are minor, we can assume that if we were given all the frames of the videos the tracker's performance would have improved remarkably.

VII. CONCLUSION

The Faster R-CNN architecture works efficiently for the US traffic sign detection problem. We were able to achieve 91.9% on the LISA-TS Original dataset and 88.7% on NavLab's Stop Sign dataset. Our network had the first convolutional layer, the second, and third stage blocks frozen, meaning those weights were not updated in back-propagation. It resized input images into 960x960px dimensions, which is larger than the 600x600px standard resizing. Higher accuracy might be achieved by working through the limitations of the presented framework. Traffic signs on the dataset have an average height of 50px, which is small, therefore using a shallower network might improve performance, because the receptive field would be much smaller on the last feature map, meaning that less information from the outside will be inserted into the traffic signs representation. Also, more training iterations and a larger quantity of samples should improve the network.

We also implement a traffic sign tracking systems that uses the Lucas-Kanade method to estimate optical flow. The Tracking system allows to omit the use of the detector in almost half of the frames in an image sequence which results in a more efficient traffic sign detection system. Utilizing SIFT features it is able to track the traffic signs with high precision, mainly because of the scale invariance properties of SIFT. The tracker system in conjunction with the detection system are able to exploit the temporal information in an image sequence and are able to recognize the same sign in multiple consecutive frames. Exploring the use of optical flow estimation and object tracking to reduce the false positive rates in the detecting phase are part of the future works. Our traffic sign detection and tracking system is the first step towards building a fully automated traffic sign inspection system.

ACKNOWLEDGMENT

We want to thank Dr. Christoph Mertz for his exceptional mentoring and guidance through the RI Summer Scholar 2017 program and Research Assistant Jina Wang for helping us throughout our day-to-day research experience. Special

thanks, to the RI and RISS personnel for providing a state-of-the-art summer internship.

REFERENCES

[1] Mohammed Boumediene, Jean-Philippe Lauffenburger, Jérémie Daniel, and Christophe Cudel. Coupled detection, association and tracking for traffic sign recognition. In *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*, pages 1402–1407. IEEE, 2014.

[2] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[3] Ross Girshick. Fast r-cnn. In *International Conference on Computer Vision (ICCV)*, 2015.

[4] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition*, 2014.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[6] Sebastian Houben, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel. Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark. In *International Joint Conference on Neural Networks*, number 1288, 2013.

[7] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. *arXiv preprint arXiv:1611.10012*, 2016.

[8] Yuan Li, Andreas Møgelmose, and Mohan Manubhai Trivedi. Pushing the speed limit: High-accuracy us traffic sign recognition with convolutional neural networks. *IEEE Transactions on Intelligent Vehicles*, 1(2):167–176, 2016.

[9] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[10] Olac Fuentes Luis David Lopez. Color-based road sign detection and tracking. International Conference Image Analysis and Recognition, 2007.

[11] Markus Mathias, Radu Timofte, Rodrigo Benenson, and Luc Van Gool. Traffic sign recognitionhow far are we from the solution? In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pages 1–8. IEEE, 2013.

[12] Andreas Møgelmose, Dongran Liu, and Mohan M Trivedi. Traffic sign detection for us roads: Remaining challenges and a case for tracking. In *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, pages 1394–1399. IEEE, 2014.

[13] Andreas Møgelmose, Dongran Liu, and Mohan Manubhai Trivedi. Detection of us traffic signs. *IEEE Transactions on Intelligent Transportation Systems*, 16(6):3116–3125, 2015.

[14] United Nations. Vienna convention on road signs and signals. 1978.

[15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[16] K. Sharma L. Wander S. Varadharajan, S. Jose and C. Mertz. Vision for road inspection. In *IEEE Winter Conference on Applications of Computer Vision*, 2014.

[17] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[18] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *Acm computing surveys (CSUR)*, 38(4):13, 2006.

[19] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[20] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *European Conference on Computer Vision*, pages 391–405. Springer, 2014.

[21] Zhongrong Zuo, Kai Yu, Qiao Zhou, Xu Wang, and Ting Li. Traffic signs detection based on faster r-cnn. In *Distributed Computing Systems Workshops (ICDCSW), 2017 IEEE 37th International Conference on*, pages 286–288. IEEE, 2017.

A Virtual Reality Glove for Kinesthetic Feedback Using Electroadhesive Clutches

Jonathan Alvarez[1]

Abstract— **Haptic technology seeks to bring sensational feedback from the real world to technology through various methods of sensory emulation, whereby these sensations and forces are recreated artificially, in order to enhance the user experience and have it parallel a real world experience. Nowhere has this development been more imperative than in the very technology designed to fully parallel the real world: virtual reality. Current wearable haptic technology in virtual reality is limited to expensive exoskeletons or inelegant tactile glove designs. Glove designs, in particular, have the most potential for ubiquitous acceptance across virtual reality, yet there are two problems with current designs: (a) tactile feedback solely is ineffective in recreating true sensory feedback sensations; (b) current kinesthetic designs employing mechanical dampeners or actuators are far too bulky and expensive to provide a lasting solution. In this paper, the development and application of a new kinesthetic glove is proposed, using electroadhesive clutches, which are both lightweight and robust, as its primary means of restrictive feedback. Preliminary results have shown that the force achieved upon activation of the clutches is adequate in emulating true proprioceptive forces and sensations.**

I. INTRODUCTION

Humans navigate the world through interaction and feedback with the world around them. Our sensory systems allow us to feel the world, allowing us to execute tasks and objectives with unparalleled precision. With the advent of new technology, designers have sought to emulate these sensory feedback mechanisms with haptic technology, to provide a more realistic and natural ambiance to users. Haptics, in essence, aims to mimic the somatosensory system of human brains. We believe that providing this interactive and real-time feedback in a virtual environment can noticeably improve the user experience in a virtual environment.

Haptics, as a field, developed as a means to understand, describe, and study proprioception. There are traditionally two fields of haptic technology: one concerned with tactile feedback and the other with kinesthetic feedback.

Tactile feedback are those sensations mimicking the sense of touch, normally recreated electronically with pressure or vibration motors, which stimulate the skin to mimic touching an object. Nevertheless, tactile feedback is able to convey more information than a simple vibrational alert, in that it can vary in strength depending on the proximity to an object or based on how hard one is touching a virtual object.

Kinesthetic feedback aims to mimic proprioceptive senses, such as the weight of an object, object impedance, or joint angle. Kinesthetic devices are becoming increasingly common, especially in fields working with teleoperation, improving both user experience, and more importantly, user accuracy [1], in order to accurately recreate Newton's Third Law while interacting with the virtual world. The novelty of this idea is in the mechanism by which the users hand is restricted, electroadhesive clutches, which differs from the more common use of actuators to apply forces to the users fingers [2], [3], [4].

Almost all of the current research and designs that have been produced over the past several years have focused more on refining the various pneumatic-based kinesthetic haptic gloves [3] or integrating kinesthetics and tactile feedback into one glove [4]. However, little research has gone into changing the mechanism of actuation itself. While the current methods of actuation are not all equal, they do share common characteristics, mainly either pneumatics or small motorized components. Researchers at the University of California, San Diego, currently have the most novel method of kinesthetic feedback, with the integration of soft-robotic actuators into their exoskeleton, as elaborated upon further in the Related Works section [2]. Nevertheless, even with this innovative method of introducing proprioceptive feedback, with the need of an exoskeleton to hold their actuators, they have been unable to mimic the natural size and shape of the glove. The glove presented in this paper seeks to maintain the robustness and strength of current gloves, while not only introducing an entirely new method of restriction, but also keeping weight and overall components to minimum.

This glove takes advantage of a thin and powerful electroadhesive clutch, initially developed for improving gait in rehabilitative work [5]. A clutch itself is a device used for power transmission, whereby two moving objects are engaged or disengaged. Most clutches have been developed for the transportation industry, as a means of connecting the engine to the transmission. As opposed to the more well-known friction clutches common in many vehicles, the clutch used in this design is electroadhesive, and so the mechanism by which power is engaged or disengaged is through the phenomena of electrostatics. Electroadhesion describes the attraction of two oppositely charged surfaces to one another [6]. The specific mechanism by which the surfaces become attracted is outside of the scope of this paper, however, in essence, it occurs as charge builds up on one sheet of the clutch and thus the adjacent, grounded sheet becomes attracted to it.

A virtual environment was developed to test the glove upon completion of its design. It uses a near-IR camera (Leap

*This work was supported by the Robotics Institute for Summer Scholars Program

[1]J. Alvarez is a research intern with the Integrated Soft Materials Laboratory, Carnegie Mellon University, Pittsburgh, PA, 15213

Motion Controller; Leap Motion, Inc.) to track the motion of the glove. A simple virtual game was designed using Unity 5.6.1 (Unity Technologies) with several objects to pick up and interact with. The interface between the virtual environment was provided with an Arduino UNO microcontroller. A simple systems diagram of the components of this design and experiment is provided below [Fig. 1].

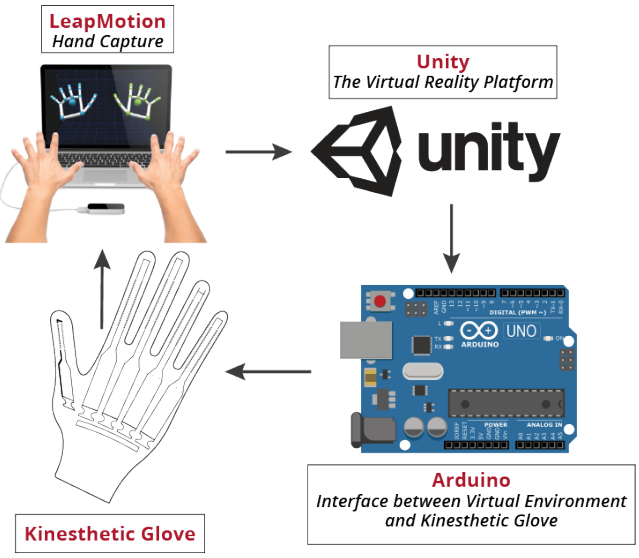


Fig. 1. Systems diagram schematic showing the main components of this project. The Leap Motion controller is used to capture hand gestures, which are fed into the virtual environment created in Unity. Scripts attached to the virtual objects call on the Arduino to activate or deactivate the clutches, according to the current position of the user's hand in the virtual environment

The following sections of this paper are organized as follows. Section II will elaborate upon and describe the design of some of the most innovative kinesthetic-feedback gloves to date. Section III describes the general approach and design of both the glove and the virtual environment used to test the glove. Section IV substantiates the glove's performance, while Section V will conclude the paper and point to future work.

II. Related Works

The integration of kinesthetic haptics into the field of virtual reality is a relatively recent development when compared to the development of the field itself. Accordingly, there is less literature on the subject than would be the case for a more traditional discipline. However, there have been several developments, specifically with respect to kinesthetic technologies for virtual reality, and they will briefly be mentioned below. The gloves mentioned below are by no means an exhaustive account of all haptic gloves conceived, as there do exist more gloves in existence, but these gloves are among the most advanced.

Current academic-sponsored research has only developed a handful of kinesthetic feedback glove, with one of the most inventive designs coming from researchers at the University of California, San Diego. Their glove is uses a soft exoskeleton, to be worn over the hand, with soft robotic actuators over each finger, and McKibben muscles for actuation [2]. The exoskeleton is worn above the hand, and upon activation, it becomes stiff, and the hand is restricted. Although this design provides reliable kinesthetic feedback, it suffers from a very extended and intensive fabrication time, as well as many overhead costs, mainly the Fluidic Control Board purchased which manages all the pneumatics. Furthermore, the glove is quite large and invasive, as it is an exoskeleton worn over the hand, and there are various valves and wires that must be run to and from the glove in order for its control and activation [2].

Although academic research is beginning to look into the realm of haptics and virtual reality, the overwhelming majority of current kinesthetic feedback gloves currently come from industry-sponsored or entirely private research. Of several designs currently on the market, *Dexmo* is considered top of the line. The brainchild of several researchers from the University of Cambridge in collaboration with Dexta Robotics, *Dexmo*, is an above-hand rigid exoskeleton which sits on top of the users hand to provide kinesthetic feedback upon use [3]. Its mechanism of actuation comes from several micro server units fitted with linear sliders which provide passive restriction, meaning no direct force is applied directly to the user. *Dexmo* is untethered, meaning it requires no desktop power supply. While *Dexmo* is inexpensive and reliable, it suffers from a rigid design which may be uncomfortable to the user. Moreover, it uses a shifting mechanism to actuate each force-feedback unit which introduces delays into the system which affect end users overall immersive experience [3].

There is an underlying issue in all of these gloves. While they all have specific advantages and disadvantages that have made them more or less popular with users, the overwhelming majority of all current haptic gloves are expensive both in fabrication methods and in monetary cost. Furthermore, none of the gloves are truly form fitting, which calls into question certain concerns with ergonomics that inevitably arise. Our design promises to not only be cheap and quick to fabricate, but it also is lightweight and unobtrusive, allowing the user to interact with the virtual environment just as they would the real world.

III. Approach

A. Design of Electroadhesive Clutches

When approaching the design of the clutches, the two most prominent concerns were size and strength. We knew we wanted the clutches to be as lightweight and unassuming as possible, while still having the resilience to hold under numerous cycles of actuation and providing a force in accord with those normally felt when grasping an object.

The design of the clutch was made to mimic, on a smaller scale, the original design prototyped by Stuart Diller [5] [Fig. 2]. The clutch itself is made of aluminum-sputtered Mylar®, a polyester film of biaxially-oriented polyethylene terephthalate, coated with LuxPrint®, (LuxPrint; Dupont Microcircuit Materials, Research Triangle Park, NC) a polymer-ceramic composite which acts as a dielectric. With a a high dielectric

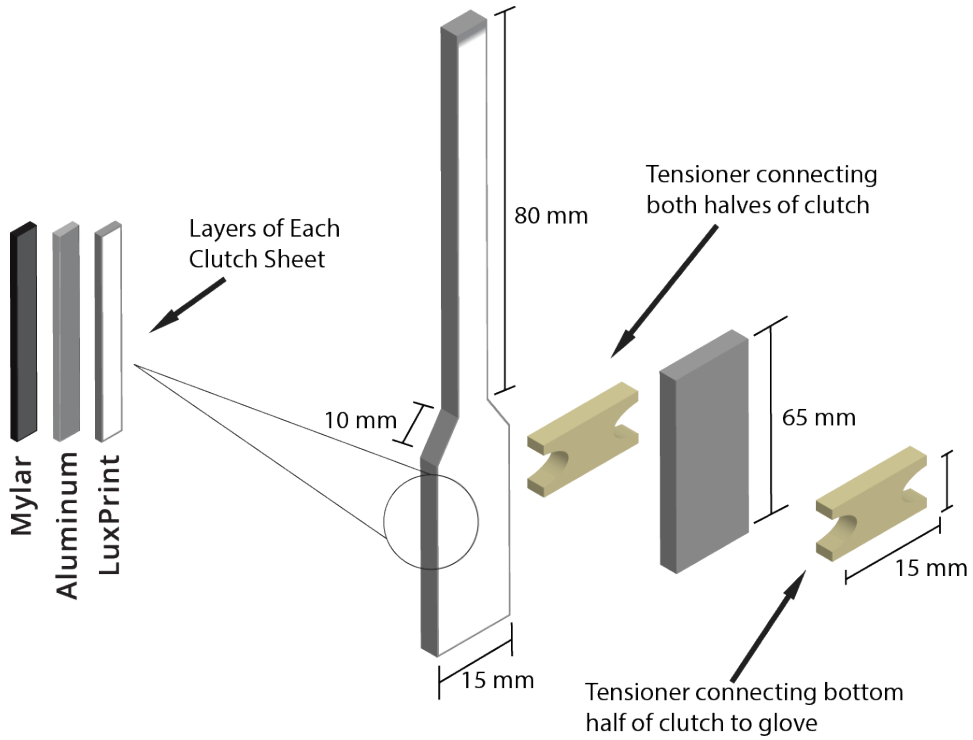


Fig. 2. The design of the electroadhesive clutches used in the kinesthetic feedback glove. Note the breakout section illuminating the face that each clutch is made up of three layers of material: Mylar, Aluminum, and LuxPrint, with LuxPrint being the inner layer and Mylar the outermost layer of each clutch.

constant and low affinity for moisture, LuxPrint provides an ideal insulating layer for the electroadhesive clutch. The idea behind the original design focuses on two sheets of the electroadhesive clutch attached by rubber bands, which act as tensioners, around slits in carbon fiber which is adhered to the individual sheets [5]. With this design, the sheets will slide past each other freely with resistance coming only from the inherent tensile strength of the rubber bands. Upon application of voltage, the clutches adhere through proton-electron interactions. Initial testing found that two sheets of the electroadhesive clutch 10 cm by 10 cm provided a tensile strength of approximately 200 N [5]. Further testing has shown that the tensile strength scales with size.

As mentioned previously, the kinesthetic glove clutches stay true to the original design with adaptations to allow for optimal activation and deactivation of the clutch with little to no residual space charge [7]. In order for ideal activation and deactivation cycles, the two sheets of the clutch must be in nearly constant contact, or at least separated by no more than a few micrometers; if the gap between the sheets is any larger, the clutches will not activate and adhere. These were the considerations taken in the design of each individual clutch for the gloves 4 fingers and thumb.

The clutch sheets were designed to maximize contact surface area, while still remaining thin enough to allow for separate clutches to be fastened per finger [Fig. 3], The top clutch spans the superior face of the palm and follows the fingers tendon, threaded through the glove to prevent lateral slippage, and secured to the tip of the finger with cyanoacrylate glue (Krazy Glue; Krazy Glue, Inc., High Point, NC). Because of the natural insulative properties of both the Mylar and LuxPrint, acetone was used to strip LuxPrint from the top and bottom of one side of each sheet, in order to provide access to the aluminum-sputtered side of the Mylar. Two 0.006 Natural Rubber tensioners were laser cut with a bench-top CO_2 laser (30 W VLS 3.50; Universal Laser Systems), and adhered to the base and top clutch sheets with 3M™ VHB™ tape, (3M; Maplewood, MN) primed with 3M Primer 94. These tensioners secure the bottom and top sheets of the clutch together, as seen in Fig. 2, while minimizing any force experienced upon closing the hand when the clutches were not active.

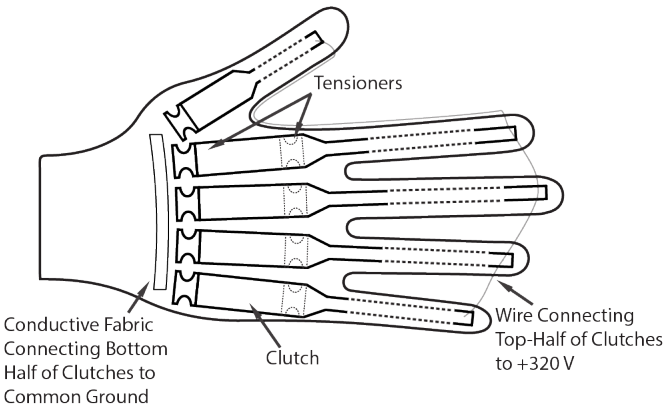


Fig. 3. Outline model of kinesthetic feedback glove. Dotted lines represent components of the glove that are not visible from a top view of the glove.

Using strips of conductive tape, the top electrodes are made electrically common, as are the bottom electrodes. The top electrodes are powered using a high-voltage power supply (Model PS375, Stanford Research Systems, Sunnyvale, CA) to 320V, which provides optimal electroadhesion according to original research [5]. The bottom electrodes, those in contact with the superior surface of the glove, are grounded. Although the voltage is quite high, each clutch only draws anywhere from 1-4 micro amps upon activation. Although the current drawn is very small, the user is still recommended to wear a nitrile glove as a base layer while using the clutch, so as to isolate them from the power supply. The thumb clutch is designed exactly the same, except the contact surface area has been shrunk to commensurate size. The four finger clutches are mounted on individual laser cut pieces of polylactide plastic which are mounted directly to the superior part of the glove using fabric adhesive (Fabric Fuse, Therm O Web, Inc., Wheeling, IL). The clutches are mounted here, rather than directly to the glove themselves, because the clutches adhere more easily and securely to the plastic than the fabric glove. The thumb clutch is mounted a similar way, however it has a slightly larger plastic base, mounted proximally to the thumbs knuckle.

B. Development of Virtual Environment

To assess the design and application of the glove a virtual environment was developed using a Leap Motion controller and the game engine Unity3D. The Leap Motion controller is a Near IR desktop tracking device which allows for reliable tracking of the users hands. Special consideration as for the material of the gloves was needed in order to ensure that the camera would still be able to dependably locate and track the glove. Unity3D was chosen as the platform for development of the virtual environment as opposed to other engines, such as Unreal, as the integration between Unity and Leap Motion is well-documented and there are numerous pre-fabricated assets for quick prototyping available for download through the Leap Motion website.

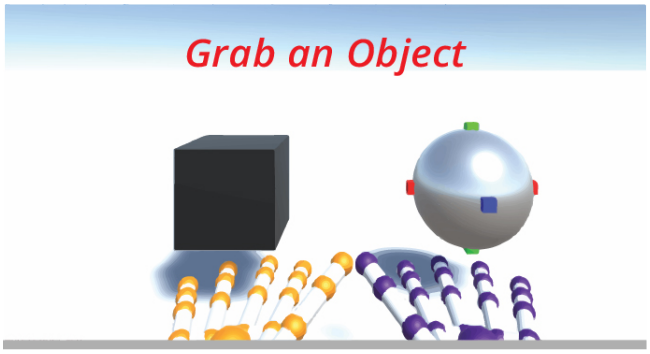


Fig. 4. Screenshot of the virtual environments and the objects with which the user can interact. The prefabricated hands provided by Leap Motion for use with Unity can be seen at the bottom of the screen.

The virtual world is designed to be very intuitive. There are several items laid out in front of the user upon beginning the game, and they are prompted to interact with them, able to pick them up, push them around, or throw them. Upon grabbing an object, defined as making contact with said object with the thumb and two fingers, the glove will activate, and the users hand will be restricted from closing any further, as the clutches activate. Following, upon releasing the item, the clutches will be disengaged and the user will feel no more force from the glove, until another grasping event is initiated. This intends to simulate the feeling of equal and opposite force when grabbing an object.

C. Integration of Glove and Virtual Environment

With the glove fabricated and the virtual environment developed, the final step was to interface between the two. This was accomplished using an Arduino UNO accompanied by a simple circuit with high-voltage photorelays (TLP222G, Tohsiba Corporation, Minato, Tokyo Japan) acting as the switches. The LeapMotion SDK comes with several prefabs and scripts that can be used and manipulated as needed for the game being created. One such script is called OnGraspBegin( ) and another, OnGraspEnd( ). As the names allude, both scripts are called once the virtual hand grasps or releases an item; where a grasp is registered once contact between the thumb and any two fingers is made between a virtual object. In order to interface with the Arduino UNO, a script developed by Darrenlloyd Gent (R3DStar UK) is called which sends, over the serial port, a call to the Arduino. The Arduino reads these calls and will then either switch on or off the power supply, which in turn would set its digital pins high or low. These pins were connected to a voltage divider to decrease the voltage to 1.25V, which was in turn connected to the anode of the photorelay. Upon application of 1.25V to the photorelays, their drains would connect together and the clutches would either activate or deactivate, depending on whether or not the virtual object was being grasped.

IV. Results

Preliminary testing has shown that the forces produced by each clutch are satisfactory in achieving perceptible proprioceptive forces on the hand's of users. As mentioned previously, the clutches scale with size, and with a contact area of approximately 9.75 cm^2, our clutches have an area about 10% that of the original 10 by 10 clutches. Therefore, in theory, each clutch should produce approximately 20N

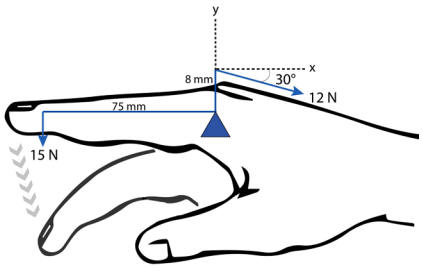


Fig. 5. A free body diagram describing the typical force produced by a single clutch upon activation, with the 4N force describing the force from the clutch upon activation. The forces can be modeled as a simple lever mechanism

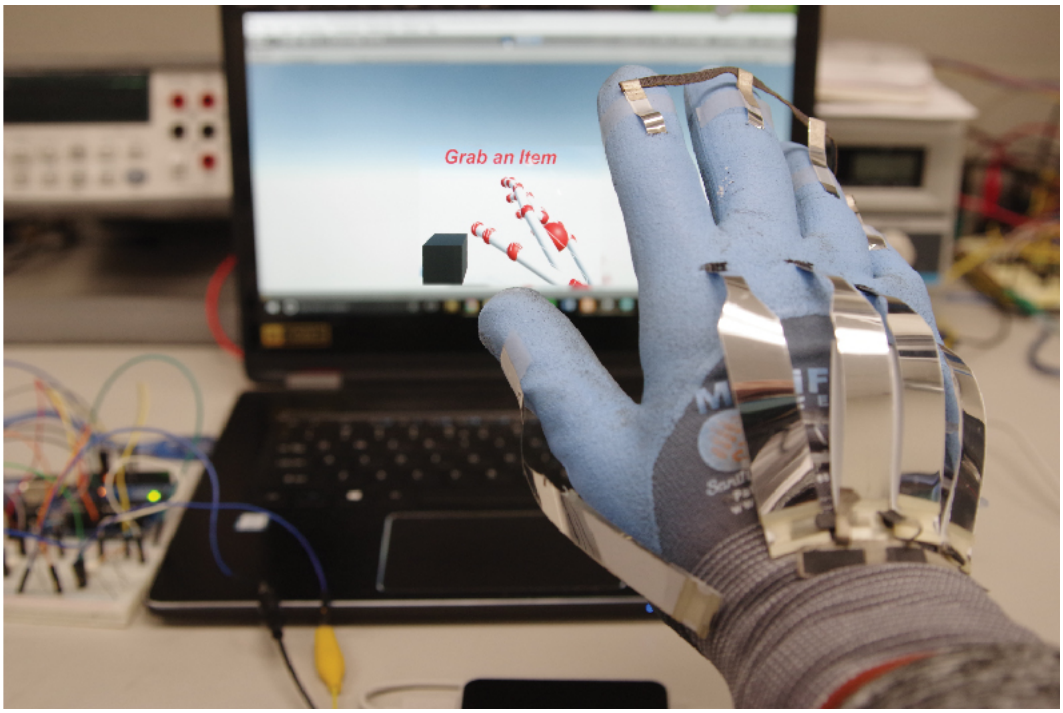


Fig. 6. Photograph of the assembled glove interacting with the virtual environment. The glove is being tracked by the Leap Motion controller seen at the bottom of the photograph, and the Arduino setup can be seen to the left of the computer screen.

of restrictive force before coming apart. In practice however, the actual force produced by each clutch is somewhat lower, (Fig. 5.) due to imperfect astriction on the part of the clutches; that is, the entire surface area of the clutch does not always come in contact.

V. Conclusions

In this research, a comprehensive development of a kinesthetic feedback glove for virtual reality was conducted and its implications proposed. The field of haptics has always been an integral part of user interface and interaction, and its integration into the field of virtual reality has been a natural progression. Whereas tactile haptics provide abrupt, short stimuli to the users skin, kinesthetics has the potential to bring a more natural feel into the virtual world. By utilizing a novel method of restricting the users fingers, with electroadhesive clutches, the glove is as robust and reliable as it is cheap and easy to produce.

The applications of this glove have the potential to span past just providing a more immersive virtual environment. They can be employed in various other novel fields, such as teleoperation and virtual rehabilitation. Furthermore, the clutches themselves have the potential to revolutionize the field of low-power clutches as their flexible and lightweight design could allow them to be subtly integrated into various applications.

Future work will seek to activate each clutch individually, so as to further mimic the true experience of grasping an object. Exploring the field of pseudo-haptics also has the potential to make this kinesthetic glove even more effective in providing to the user a sense of proprioceptive force [8]. Finally, a possible integration of tactile haptic technology in the fingertips of the gloves may also be able to give the user simple tactile cues, which, when combined with kinesthetics has the potential to be much more lifelike than any other haptic glove today.

ACKNOWLEDGMENT

This work has been supported by the Robotics Institute for Summer Scholars and the Integrated Soft Materials Laboratory at Carnegie Mellon University. Special thanks to Professor Carmel Majidi for all his support and insight over the course of this research. Moreover, I would like to thank Stuart Diller for his invaluable insight into the use of the electroadhesive clutches, and Yichu Jin for initially recognizing the possibility of integrating the clutches into a haptic feedback glove.

References

[1] C. S. Tzafestas, "Whole-hand kinesthetic feedback and haptic perception in dextrous virtual manipulation," in *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 33, no. 1, pp. 100-113, Jan. 2003. doi: 10.1109/TSMCA.2003.812600

[2] Jadhav S., Kannanda V., Kang B., Tolley M. T., Schulze J. P., (2017) "Soft robotic glove for kinesthetic haptic feedback in virtual reality environments," in *IS&T International. Symposium on Electronic Imaging: The Engineering Reality for Virtual Reality*, Springfield VA, 2017, pp. 19-24(6).

[3] X. Gu, Y. Zhang, W. Sun, Y. Bian, D. Zhou, and P. O. Kristensson. Dexmo: An inexpensive and lightweight mechanical exoskeleton for motion capture and force feedback in VR,. in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, Santa Clara, CA, 2016, pp. 19911995. doi: 10.1145/2858036.2858487

[4] M. W. Uddin, X. Zhang and D. Wang, "A Pneumatic-Driven Haptic Glove with Force and Tactile Feedback," *2016 International Conference on Virtual Reality and Visualization (ICVRV)*, Hangzhou, 2016, pp. 304-311. doi: 10.1109/ICVRV.2016.57

[5] S. Diller, C. Majidi and S. H. Collins, ”A lightweight, low-power electroadhesive clutch and spring for exoskeleton actuation,” *2016 IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, 2016, pp. 682-689. doi: 10.1109/ICRA.2016.7487194

[6] C. D. Shultz, M. A. Peshkin and J. E. Colgate, ”Surface haptics via electroadhesion: Expanding electrovibration with Johnsen and Rahbek,” *2015 IEEE World Haptics Conference (WHC)*, Evanston, IL, 2015, pp. 57-62. doi: 10.1109/WHC.2015.7177691

[7] G. Damamme and C. Le Gressus, ”Space charge characterization,” *Proceedings of Conference on Electrical Insulation and Dielectric Phenomena - CEIDP ’96*, Millbrae, CA, 1996, pp. 8-11 vol.1. doi: 10.1109/CEIDP.1996.564573

[8] C. Neupert, S. Matich, N. Scherping, M. Kupnik, R. Werthschtzky and C. Hatzfeld, ”Pseudo-Haptic Feedback in Teleoperation,” in *IEEE Transactions on Haptics*, vol. 9, no. 3, pp. 397-408, July-Sept. 1 2016. doi: 10.1109/TOH.2016.2557331

Anomaly Detection and Forecasting for Malfunctioning Radiation Portal Monitors

Fay Blelloch
Swarthmore College
fblello1@swarthmore.edu

***Abstract*—RPMs are the primary method for detecting radiation at a national scale. Prior to the 9/11 terrorist attacks, the use of RPMs was limited to secure US facilities. However, their use has expanded to all major borders and ports, and there are over 1400 RPMs currently in operation [1]. While their importance to homeland security has increased, their major growth presents challenges to effective and timely maintenance. Malfunctioning RPMs can produce false positives, which are costly in terms of time and expense, or even risk dangerous missing detection. Thus, the need to detect malfunctions is critical to effectively directing resources to a maintenance strategy. To meet this challenge, this project develops an anomaly detector that uses random forest classification and kernel density estimation to flag abnormalities that are likely the result of malfunctioning RPMs. All results were tested via cross-validation and flagged observations were compared with raw data. Our results showed that all flagged observations had low background gamma counts and high numbers of error codes, suggesting that our method was successful in detecting genuine malfunctions. This method also shows promise for malfunction prediction.**

***Keywords*—radiation portal monitors,randomForest, kernel density estimation, anomaly detection**

I. Introduction

Radiation Portal Monitors (RPMs) are devices used at borders and entrances to secure facilities in the United States to detect radiation threats entering the country by land, air, or sea. While these monitors were originally used almost exclusively to scan vehicles entering secure locations, the events of September 11, 2000 resulted in usage of these devices in many other locations. Over 1400 monitors are now used at locations such as port and border entrances, urban centers, nuclear facilities, and import and export terminals [1]. This exponential increase of RPMs in use creates some problems, as large amounts of monitors are more difficult to maintain across the board. Currently, there are two ways that RPMs are regularly checked for maintenance. Primarily, when port staff observe unusual behavior from a monitor, they call in maintainers from a separate organization to check the calibration. While this approach often catches malfunctioning RPMs, it can only catch the failures as they occur as there is no way to forecast and anticipate malfunction. A bigger problem with this method is that often RPMs can ”silently” malfunction, or stop working without actually making it clear in any output data. These problems will then go unchecked for long periods of time if there is no obvious problem with the monitor. A second method currently used for RPM maintenance involves customs workers monitoring summary statistics given off by the RPM data. Like the previous method, this form of maintenance is unable to forecast malfunctions. To be more successful in determining malfunction before it happens, as well as remove the difficulty to quantify RPM fleet health from maintenance records, a new method of malfunction detection must be developed.

Fig. 1. Example of an Radiation Portal Monitor (RPM)

This paper explores the development of an anomaly detection system that could eliminate the problems with current maintenance described earlier. This detection system can both identify RPMs that are behaving unusually and predict when an RPM may begin behaving unusually.The detection system developed relies on a classification method as well as a density estimation method to achieve these tasks. The idea is that the system marks anomalous RPM observations based off of how different from normal performance the observation is. More specifically, by using classification models to predict the probability of specific response variables occurring and then using the density of those probabilities, it is possible to see which observations are predicting the response variable unusually. These observations predicting the response variables unusually most likely have raw data that is unusual, a sign that the RPM may be malfunctioning at that point in time.

This data consists of two years of data from twenty-one different RPMs. Each RPM has data for background gamma counts, calibrations, and error codes, signals that there is an error somewhere in the monitor, such as data communication errors and neutron errors.The methodology went as follows,

first featurizing the data. Then all possible error codes were fit to many different classifiers to determine both which type of classifier is most successful in making error predictions and which errors are most successfully predicted. After a classifier, the random forest, was chosen, the scores received from the test set predictions from each of the four classifiers were put into a single vector for each observation. A kernel density estimation (KDE) was then built with all of the four-score vectors. New observation points were then run through the classifiers, received a four-score vector and compared to the KDE to determine the probability of that observation happening under the KDE. Probabilities over two standard deviations away from the mean probability were flagged as possible anomalies, as they were the least likely to be seen under the previously made KDE. These raw data for these flagged possible anomalies was then looked at further to determine if there was cause to investigate a specific RPM for possible malfunction. With further research, the detection system created in this paper could ensure that all RPMs are functioning at their full capability. This project provides the groundwork for a method that could eventually successfully ensure comprehensive maintenance across ports and borders.

II. Related Work

This project focuses on anomaly detection, a commonly used and researched aspect of machine learning. There are multiple types of anomaly detection, including intrusion detection, fraud detection data leakage protection and medical applications. [2]. This project's anomaly detection does not fall in any of those categories specifically, but rather simply determining when points are acting unusual. Anomaly detection methods fall into three groups: fully supervised, semi- supervised and unsupervised. Fully supervised detection involves training and testing data with correctly labeled anomalies, while semi-supervised uses training and testing data but rather than labeled points, just defines points as anomalies if they deviate from the model. Lastly, unsupervised methods use no training or testing sets and labels anomalies based solely on the properties of the data itself [2]. The data used in this project had no known anomalies, so a fully supervised method was out of the question. To best classify anomalies in this data, it was decided to use a semi-supervised method with training and testing data to locate anomalies in the testing data.

The methods used in this paper to analyze and determine anomalies semi-supervised were random forest classification and kernel density estimation. Random forest classification is a commonly used form of classification. In essence, the random forest algorithms, "add an additional layer of randomness to bagging" [3]. The model constructs trees using different samples of the data, making it so that even the model made with training data has been cross validated in a sense. Each node in the random forest is then "split using the best among a subset of predictors randomly chosen at that node... turns out to perform very well compared to many other classifiers, including discriminant analysis, support vector machines, and neural networks," [3]. One of the most compelling aspects of the random forest classifier is that the model prevents over fitting, an issue common in classification. With a featurized data set with over 90 features, it was important to consider using random forest because of this characteristic. By avoiding over-fitting, this anomaly detector can potentially be used successfully on new radiation portal monitors.

Kernel density estimation (KDE) is a non-parametric density estimator. Like histograms and other density estimators, KDE graphs the density of a set of observations and points. However, unlike the histogram, KDE is smooth, does not have end points, and relies on bandwidth rather than binwidth. [4]. KDE can find the density for multi-dimensional vectors. In this project the density estimator would be used on 4-dimensional data, so the KDE was important to use to achieve the correct and accurate density estimates.

III. Methods

A. Building the Classifiers

The data set used for this project contained daily time series data for two years (January 1, 2015 - December 31, 2016) for twenty-one different RPMs. Each observation included background gamma counts for each of the eight Radiation Sensing Panels (RSP) and error counts for fourteen different possible errors (scan, RSP communication, RSP gamma OOR, RSP neutron OOR, software neutron OOR, battery, datacom connection Closed, annunciator, vps, control box hardware, write socket, push socket, other). To begin building the classifiers, first the data was featurized in 20 day time windows for each RPM. The response variable was a Boolean of whether or not a specific error happened in the next 20 days. For each observation window, 92 features were extracted. The first feature was a Boolean of whether or not the target error occurred in the 20 days included in the time frame. For the background gamma counts and other variables that had high quantitative values, a simple moving average and standard deviation were used. For each of the 13 errors that were not used as the response variable, a moving sum was calculated for each time window. While there are many other possible featurizations of the data, this simple version was determined to be reasonable start.

Once featurized data was created for all 14 errors, multiple types of classifiers were built to determine which errors are most accurately predicted through classification as well as which types of models learned the data most successfully. Models tested on the errors were k nearest neighbors, classification and regression trees, linear discriminant analysis, generalized linear model, naive bayes, and random forest.

Each model was evaluated using an ROC curve and its subsequent AUC score. The majority of the models did not learn from the training data, having ROC curves that proved to give random or worse than random predictions. The only four errors that were able to be predicted well were push socket errors, vps errors, data communication connection closed errors and other errors. These four errors all had occurrences in over 20% of the time windows, with DataCom Connection Closed errors and Push Socket Errors occurring about 50% of the time. In terms of classifying, the tree models, random forest and CART, had the highest accuracy and ROC scores. The graphs on the

next page show the ROC curves for four of the models tested for the response variable of Push Socket Errors. It is clear from the curves that the Random Forest method learned and predicted the data most successfully, while the logistic model was least successful.

While there were multiple classifiers that resulted in respectable ROC curves, after the preliminary analysis, it was decided to use random forest models on the response variable errors listed earlier. While random forest models use an internal training and testing scheme with out-of-bag sampling, to further test and train the data, the twenty-one RPMs were broken into seven folds of three RPMs each to ensure that observations from the same RPM stayed together. Random forest models were then trained on six folds and tested on the last one fold. The average test results from each of the four error classifiers are summarized in the table below.

Error	OOB Error Avg	Test Set Error Avg	True Positive Rate Avg	False Positive Rate Avg
DataCom Connection Closed	4.20%	3.98%	0.98	0.058
Push Socket	10.47%	9.15%	0.96	0.13
VPS	10.09%	15.41%	0.28	0.02
Other	15.60%	19.51%	0.28	0.03

TABLE I. RANDOM FOREST MODEL SUMMARY FOR FOUR MOST COMMON ERRORS

As seen in the table, the models for data communication connection closed and push socket errors were very successful with high true positive rates and incredibly low false positive rates. VPS errors and other errors were predicted less successfully, with the models almost always predicting false. This could be explained by the fact that there were few errors, so the model for the most part learned to predict ”false”. However, because the ROC curves were still better than random with auc scores higher than 0.5, the models did learn something from the training data and therefore were kept for the kernel density estimation procedure.

B. Kernel Density Estimation

Each ”score” emitted for tested observations in the random forest classification included a probability for ”True” and ”False”. For each 20-day observation in the data set, the score for true for each error classifier was assembled into a vector. Because of the limited amount of data available, the same cross validation scheme had to be used for the kernel density estimation and the anomaly detector. The score vectors from the test sets were kept separate by fold and then a kernel density was found for 6 of the folds. The seventh was then tested and compared to the kernel density. The probability of the observations in the seventh fold, the test set, under the KDE was determined. This process was then repeated with every other possible combination of testing and training score vectors. All of the probabilities under the KDE were then ranked for all the observations. Because kernel density estimation relies heavily on the chosen bandwidth, the ”probability” given to each observation did not necessarily refer to the true probability, as different bandwidths reproduced different results. Instead, to determine possible anomalies, the mean of the probabilities was found. For simplicity in this preliminary study, any observation probabilities further than two standard deviations away were flagged to be examined further.

The graphs on the next page represent a preliminary exploration of the training and testing scheme for one fold of data with the KDE. For visual purposes, this preliminary exploration used only two dimensions, only two of the error scores. The density shapes represent the training data, the blue points are the testing data and the red points represent the testing data points that were eventually classified as anomalies with the four-dimensional KDE. Without looking at the red, marked anomalies, it is clear that there are observations in the testing set that could be potential anomalies. Later, marking the anomalies as red, it is clear that these points are furthest from the training data, therefore making sense as reasonable anomalies.

IV. RESULTS

Using the scheme to find possible anomalies described earlier using a standard deviation measure, twenty-nine possible observations were flagged. The raw data for the flagged observations were then observed. Nine of the twenty-one RPMs were represented in the flagged data. Twenty-seven points were from consecutive observations between May 2015 and August 2015 for each of eight different RPMs, while the last two were from the ninth RPM in December 2015. The twenty-seven points between May and August all had similar raw data: abnormally low background gamma counts, particularly on one radiation sensing monitor. Furthermore, the anomalous points had high amounts of error codes. About half of the points had errors that occurred less than 5% of the time in all the data. The table below summarizes key parts of the raw data of the six points that were the furthest away from the mean probability of the test points under the KDE. The first column is the value of the background gamma counts for one radiation sensing monitor, then the count of DataCom Connection Closed errors, Push Socket Errors, VPS Errors, and a sum of the other 11 errors. It is important to note that observation 2, 5, and 6 had errors that were highly unusual. To put the points into perspective, the median values for all observations are listed.

RSP 3 (median: 3898, minimum: 2882)	DataCom Connection Closed (2)	Push Socket (0)	VPS (0)	All Others(0)
3184	5	1	13	13
3162	5	3	9	13
3110	5	4	7	8
3113	4	3	7	7
3187	8	2	4	6
2900	9	3	14	2365

TABLE II. SUMMARY OF RAW DATA FOR 6 TOP ANOMALOUS POINTS

V. CONCLUSION

In conclusion, this method of anomaly detection and forecasting showed promise for future work. With the data given,

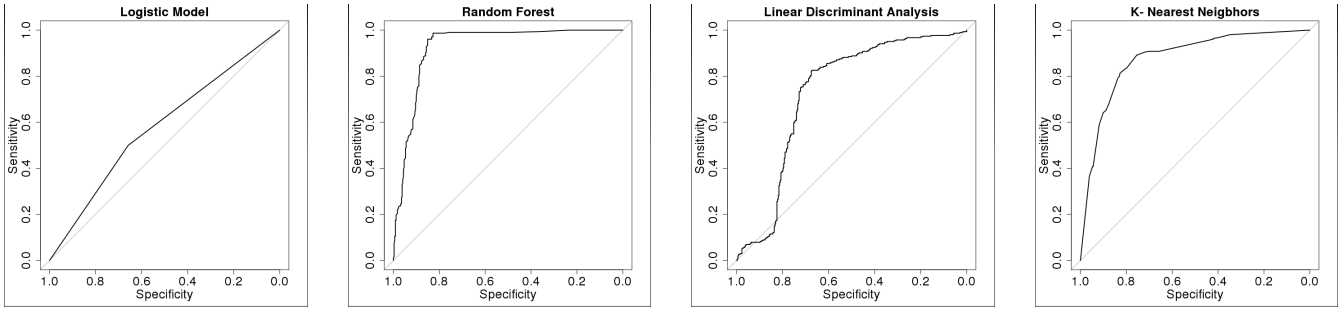


Fig. 2. ROC models.

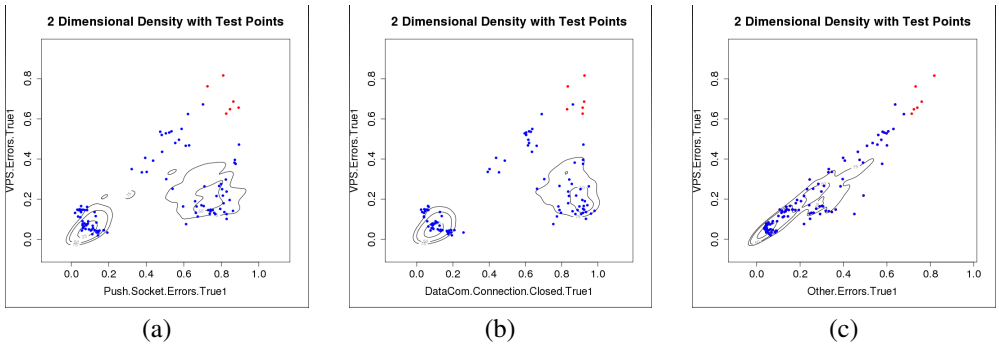


Fig. 3. Two Dimension KDE for 2 Error Types with Testing Data and Anomalous Points

there were no ground truths available to check the marked anomalies, so it is impossible to know for sure if the points flagged were correctly flagged. However, because they were all from the same month with highly irregular raw data, there is overwhelming evidence that the detector did work on the data. Because of the strong evidence that there was success in finding anomalous points, this preliminary method could be much improved to eventually help with radiation portal monitor fleet health.

A. *Limitations and Future Work*

As this project was just the beginning of an investigation into improving RPM maintenance, there is much future work to be done. One important step in future work is to include more RPM data, as only twenty-one out of over 1,400 were used. It is unknown where the RPMs in the data are from and whether they have something important like location in common. If this is the case, the classifiers as well as the kernel density estimation could be over fit to this small sample size when tested on completely new RPMs with fundamental differences. In addition, only one data featurization was tested and it was assumed that the features extracted were important to the model. This assumption of features limits the success of analyzing this project, so future work could include using different, more complex featurizations.

While none of the error codes, other than the four used, were successfully predicted with classification and used in the score vectors for the KDE, it is possible that combining the rare errors into one response would offer different results. Predicting whether any of the uncommon errors could provide more insight into the function of an RPM. This could be a future step in this project.

Finally, future work would involve truly testing the anomaly detector to see how successful it was. Since, the results presented in this paper solely delve into the possible anomalies it detected and the fact that the raw data associated with those observations were unusual points. As discussed earlier, there were other points in the data with similarly low gamma scores and high amounts of errors that were not detected as anomalies. To truly test the detection system, points that have known anomalies would be run through it to see if the detector could successfully flag them. Two types of anomalies to test for are drop outs and extra error codes. To test drop out anomalies, gamma photon counts could be severely reduced in one of the RSP panels to mimic an instance where one panel neglects to report its data. To test for extra error codes, the amount of different errors would be increased.

ACKNOWLEDGMENT

Special thanks to Kyle Miller for his mentorship, Artur Dubrawski for allowing me to join the Auton lab, Karen Chen for inspiring me to pursue this field, the NSF for their generous funding, and finally Rachel Burcin, John Dolan, Mikana Maeda, and the rest of RISS for a great summer!

REFERENCES

[1] G. Wolk. (2016) Radiation portal monitors. [Online]. Available: http://large.stanford.edu/courses/2016/ph241/wolk1/

[2] M. Goldstein and S. Uchida, “A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data,” April 2016.

[3] A. Liaw and M. Wiener, “Classification and regression by randomforest,” *R News*, vol. 2, no. 3, pp. 18–21, December 2002.

[4] T. Duong. (2001) An introduction to kernel density estimation. [Online]. Available: http://www.mvstat.net/tduong/research/seminars/seminar-2001-05/

ADDITIVE ASSEMBLY: NEW POSSIBILITIES FOR SYSTEMS INTEGRATION

Gerardo Consuelos
Rapid Manufacturing Laboratory
Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213
gconsuelosf@gmail.com

David Bourne*
Rapid Manufacturing Laboratory
Carnegie Mellon University
Pittsburgh, Pennsylvania, 15213
db@cs.cmu.edu

ABSTRACT

This article illustrates a procedure for additive assembly in PolyJet manufacturing. In the following sections we review PolyJet manufacturing, and develop a case study on a bi-stable gripper intended for human-robot collaborative assembly of an airplanes wing. The gripper is then redesigned for embedding conventional off-the-shelf parts and electromechanical connections. These modifications are intended to enhance the grippers mechanical and sensing capabilities. The objects are embedded during the printing process and the final result is integrated.

INTRODUCTION

Additive manufacturing (AM) is the process of creating, layer by layer a virtual object, usually designed in Computer Aided Design (CAD) software [1]. Development in AM technology started with the commercialization of Stereolitography (SLA) in 1987 [2]. The manufacturing process of SLA involves solidifying a photosensitive material using an Ultraviolet (UV) laser. Generally, the solidified material is then dragged layer by layer until the object is built [3]. In 2010, the American Society for Testing and Materials (ASTM) group "ASTM F42 - Additive Manufacturing" classified the current AM processes in seven categories: binder jetting, directed energy deposition, material extrusion, material jetting, powder bed fusion, sheet lamination and vat photopolymerization [4]. Being developed with both open and closed architectures, material extrusion, also known as FDM (a trademark by Stratasys) [5] has had a very strong growth since a key patent expired. The idea of additive manufacturing being an industry-level rapid prototyping method is shifting towards the inclusion of the public user [2]. Such shift has created a new space for public development of parts. Additionally, universities have made material extrusion machines available to student use. This has led to the development of a community of users experienced in additive manufacturing with material extrusion. Hence, it is common to associate the term 3D printing with a material extrusion process.

Material jetting, also known as Polyjet (a trademark by Stratasys) [5] is a process in which droplets of build material are deposited and solidified, layer by layer, in a similar manner as in an ink-jet printer. The technology itself was developed in Massachusetts Institute of Technology (MIT) in 1993. In contrast with the rest of the AM processes, Polyjet is among those that offer the finest resolution. Current machines can reach a layer accuracy of 16 microns. Another feature of Polyjet is the ability to print with different materials and colors during the same operation. This is often desirable in rapid prototyping because it approaches the end product. The materials vary in durometer scale values, hence rigid parts with flexible components may be obtained. As a consequence of the manufacturing process, Polyjet printers cannot build over-hanged bridges (In the way FDM printers do). Therefore, to produce those features, the printers build structures of support material. This material is usually soluble and/or easy to remove from the printed part.

In contrast with material extrusion, current material jetting machines are mostly produced for industrial and educational use; material jetting machines have not reached the end consumer. Hence, software used to process CAD files is provided by the manufacturer. This poses little degrees of freedom to the end

user in comparison with open source software. In this article, we use a Objet30 3D printer manufactured by Stratasys, Objet30 is supplied with Objet Studio software [6].

This paper explores how to design and assemble a complex electro-mechanical device using primarily 3D printing. The goal of the design is to achieve mechanical properties that improve traditional characteristics in strength, resistance to operational stresses while being essentially printed with intermediate assembly steps (e.g. additive assembly).

These are some design possibilities that can be obtained through additive assembly:

1. Metal stiffeners that reinforce 3D printed structures.
2. A more convenient placement for certain sensors.
3. Water-tight seals.
4. An internal network of struts for a better heat transfer.
5. Internal threaded fasteners.
6. A mechanical integration of electrical connections.

Prior Work

Fewer literature has been written regarding Polyjet than what has been written for SLA or FDM. Cazn, Morer, and Matey studied the influence of orientation, surface finish and post-processing in the mechanical properties of the end part [7]. The printed parts exhibited a highly anisotropic behavior. Through a statistical design of experiments (DOE) approach they made a series of tensile tests to each configuration. It was found that the parts with the highest Young's modulus were the parts in which the load axis of the part was printed parallel to the x-axis of an Objet Eden 330 printer. The opposite effect will occur when the load axis of the part is printed parallel to the z-axis of the printer. This effect occurs generally in layer-based AM machines. A reason for this, is that the technology itself creates material layer by layer, therefore there is a tendency towards delamination. In the case of Polyjet printers, a layer forms when liquid droplets get cured by the exposure of UV light, however when this layer adheres to the previous layer, the previous layer is already solidified, hence the addition process is different. Furthermore, it was found that parts with a glossy finish had a higher Young's modulus than the parts with a matte finish.

Regarding multi-material interfaces, Vu et al. designed a double cantilever beam test specimen to obtain information on sandwiched material [8]. Like most plastics, they found a high dependency in the rate of strain. Bass, Meisel and Williams performed tensile tests in specimens created by mixing two materials in equal quantities [9]. A design of experiments approach was developed to obtain information on material quality of both rigid and elastic parts. A factor of aging and type of storage was introduced in this work.

As current product development in additive manufacturing matures, new methodologies and procedures are created to optimize new designs. Additive assembly is the procedure of embedding objects inside a 3D printed part during the build operation. Prior literature has shown examples of embedding conventional off-the-shelf parts such as nuts and screws, additionally the embedding of custom made electrical connections has also been studied. However, so far, works have been written discussing the procedures of part embedding in FDM or SLA devices [10] & [11]. Aguilera et al. built a fully functional 3 phase direct current motor during one printing operation [12]. Their methodology involved pausing the 3D printer at a certain layer, and manually inserting each component. A total of five interruptions were necessary to complete the part. During these interruptions: ball bearings, magnets, electromagnets, and speed controllers were embedded. Aguilera et al. attempted using conductive inks as an alternative to embedding wire, however the motor's power consumption was high enough that current densities destroyed the connections. It is important to note that the software used was uPrint plus. uPrint plus gives the user no control over build support generation, in addition, the support removal procedure involved an ultrasonic bath; a process which requires the part to be submerged and heated for several hours, this process would have rendered the embedded electronics unusable. To solve this problem, it was determined by trial and error that the machine did not produce any support material if the part was built with a minimum slope angle of 45 degrees in each gap. Therefore, there was a need to redesign the part to complete a successful insertion.

In 2014, Stratasys released a set of recommendations for embedding hardware in FDM machines [10]. In the guideline, Insight software allowed to define regions of support deletion. Hence, there is no need to do a support removal operation in machines that use Insight software. Furthermore, it is possible to program an automatic pause within the software.

These procedures effectively provide standardization for embedding parts in FDM machines, however they are not adequate for Polyjet technology. The purpose of this article, is to provide a procedure for embedding operations in Polyjet printers.

Polyjet Manufacturing

Pre-Processing

The computer aided design model must be saved as an STL file format for compatibility with Objet Studio software [13]. Once imported to Objet Studio, the user has to define the build parameters. Among those parameters are: part orientation, position, and finish. An option allows the user to automatically position and orient the parts in a material efficient configuration. The user must select whether the finish is to be matte or glossy. A matte finish is rougher and may be chosen if part-painting is desirable. On the other hand a glossy finish is smoother. It is important for the purpose of this article to understand the build operation differences between a matte and a glossy finish. To

print a matte part, the 3D printer builds areas of support material every layer around the part. When this material is removed, the surfaces in contact with the material obtain a rough finish. If the glossy finish is selected, the 3D printer only builds support material in the regions where it is required for manufacturing. It is important to note that these surfaces will inevitably have a rougher finish. Regardless of the selected finish, the part is always built one millimeter higher than the build plate, therefore the bottom surface has always sacrificial support material and hence a rougher finish.

Printing Operation

The machine uses several mechanisms for motion and layer addition. This paper uses the coordinate system convention defined by ISO/ASTM [14]. The x-axis's translational motion is achieved through a conveyor belt attached to a step motor. A similar mechanism permits y-axis translational motion. The z-axis translation is obtained by Lifting or lowering the build surface. In Objet30 printers, the zeroth value of the z-axis is defined in the highest point of the z-axis translational mechanism. Then, as each layer is printed, the build surface is lowered. Additionally, an electric motor actuated roller is used every layer to create an even surface. A pre-heating operation is required before printing. As the x-axis sweeps through each layer, liquid droplets are deposited. In the next sweep operation, the machine cures the liquid by exposing it to its UV lamp.

Post-Processing

After the machine finishes its printing operation, several post-processing steps such as scraping, water-jetting, or immersing may be considered. Using a fine scraping tool, it is possible to manually remove the sacrificial support material from the part. However the support material is also soluble, and hence using a water-jet may enable the user a faster support removal. Stratasys designed a water-jet machine specifically for use in Objet-type printers [15]. Regardless of the advantages of using a water-jet procedure, sensor embedded parts may be harmed by water. In addition, even if a water jet is used thoroughly, a small amount of residue remains on the surface. To further enhance surface finish, Stratasys advises submerging the part in a caustic soda (sodium hydroxide) bath [16]. The bath must be done necessarily after using the water jet. This bath is a mixture of caustic soda and water, the exact compound ratio is given. A warning advises not to give a caustic soda bath to parts with wall thickness of less than 1 mm because deformation may occur. Furthermore, it is advised not to expose a part to a caustic soda bath for more than two hours. After the bath, the part must then be rinsed in water. A second water jet cleaning operation is recommended at this point.

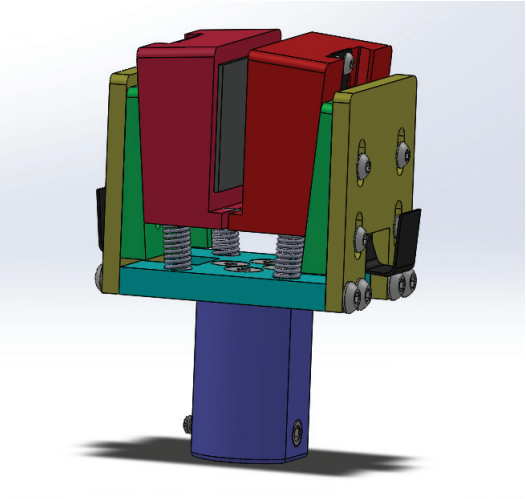


FIGURE 1. BI-STABLE GRIPPER.

Bi-stable Gripper

The gripper studied in this paper was introduced by Professor David Bourne in Carnegie Mellon University. This gripper is intended for a human-robot cooperative assembly of an airplane wing. The robot positions itself below the wing, and raises a rib carried by two grippers [17]. This rib is then attached to the wing by a human operator. Prior literature has examined the implementation of human-robot systematic cooperation. Often, tasks may be resolved efficiently if humans provide fine object manipulation while robots provide strength and accuracy.

The gripper consists of two moving grips. These grips possess a rubber strip. Each grip slides on a dovetail and is suspended on two coil springs. Two pins are slipped inside both grips as a means to provide a horizontal alignment constraint. A latching spring is attached to each grip by two screws. The latching spring is also suspended from the grips with another coil spring. Each dovetail is attached to an aluminum plate. Each aluminum plate has a tapered slope that gives each latching spring a place to attach upon. Both aluminum plates are connected together by an aluminum base that has blind holes that give placement to the springs that suspend each grip.

The mechanism works with a spring latch. Each grip has four coil springs for sliding purposes and a spring for the latching mechanical action. Such spring, latches with an aluminum plate in each side. A rib is inserted in between both ends of the end effector. Then, as the rib is pushed towards the gripper, both ends slide downwards in the dovetails. The rib is attached to the gripper by friction forces in the rubber sections. Once the gripper reaches certain depth, the rib will be latched by spring's geometry and its interface with the aluminum plates in the sides. To release the grip a human operator will press both latching springs. Another way of releasing this grip, is to have the robot

pull the gripper downwards after the rib is fixed by the operator.

In this work we will revisit the gripper in the purpose of adding and enhancing its electromechanical features through the use of additive assembly. Before this study, the last design iteration of the gripper was manufactured in an FDM 3D printer. The gripping part broke in the connection between the latching springs and the grip. The connections were directly screwed into the 3D printed part, thus creating a threaded path in the plastic. While this is a common structural connection in some plastics, 3D printed pieces are generally not given a 100% infill. If the part is not filled completely, subtracting material required for threading will provoke voids in the part and hence, damage the fill geometry. In these cases a threaded insert for plastic is sometimes pressed into the part. However tensile forces in the screw can pull or displace the insert from the grip. Therefore, we identified a necessity that could be addressed with an additive assembly approach.

It is of interest in a human-robot collaborative environment to possess failure-sensing capabilities. In that case, it was desired to add a slippage sensor to the gripper. Two design criteria had to be met. First, the sensor should not protrude from the gripper: It is undesirable (and perhaps in some cases hazardous) to have protrusions emerging from a mechanical body in a human-robot collaborative environment, especially if the robot has full translational motion. Second, no wires should hang around the grips: These wires may be caught by one of the moving parts involved in the operation. In this work we look for a solution in additive assembly that can satisfy these design constraints.

As a previous note, it is important to expose the changes made to the mechanical body of the gripper as an improvement of such. However as these changes were not developed through the use of additive assembly, they are not studied thoroughly in this work, but rather mentioned to give a better understanding of the integrated system.

Mechanical Modifications

The walls enclosing both pins were thin enough to break after a certain number of gripping operations. Additionally, the pins' surface was in contact with 3D printed FDM plastic. Hence, movement was impaired with high friction forces. This resulted in a discontinuous movement. In some cases the horizontal constraint was not satisfied and the grips were stuck in the dovetails. A solution to these was to provide the horizontal constraint by fixing both pins in one end and letting them slide horizontally inside linear bearings in the other end. The pins were threaded and fastened to an hexagonal nut in the fixed ends. This configuration was arranged at the sides of each grip. A chamfer was added to protect these structures from shear forces that may incur during operation.

We examined the possibility of embedding the hexagonal nuts and linear bearings inside the grips. However the faces of these parts were not parallel to the build plane. In order for a part to be introduced through additive assembly, its highest plane should be parallel to the build plane. While it could be argued that the entire part could be rotated for this purpose, that would eliminate the parallel relationship between other more critical embedded components and the build plane. Thus, another design solution was to place inserts for plastics in these structures and create fastening through screws. The tensile forces experimented by these screws were not significant enough to create displacement in contrast with the screws that tighten the latching springs.

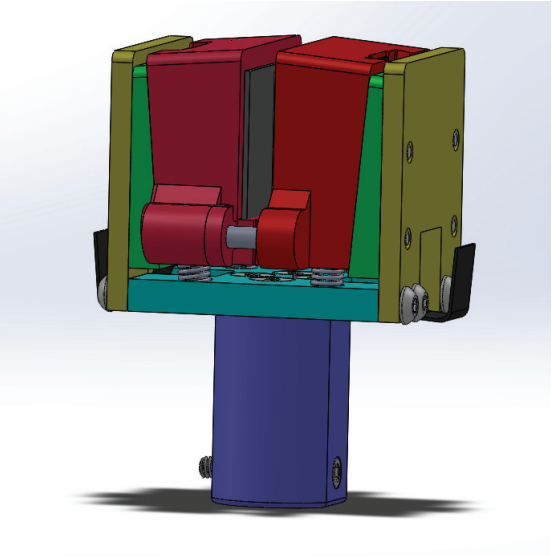


FIGURE 2. PREVIOUS MECHANICAL MODIFICATIONS.

Proposed Solution

In this case study, we propose two major design configurations implemented through additive assembly. The first one, was done to fasten the latching springs. The second major design proposal was done to give the gripper slippage sensing capabilities.

Latching spring fastening

Fig. 3 shows our procedure for embedding lock nuts. First, we replaced the threaded hole with a close fit hole of the same screw size. Then, we created a concentric hexagon and extruded that volume. A Boolean operation was used to subtract that volume from the grip. It may be argued that an hexagonal volume does not equals the geometric characteristics of a lock nut (which is rounded in one end). However it is fundamental to always consider the top layer of the additive assembly operation. Part embedding occurs at this layer. For insertion to be possible, the embedded part must be able to be slipped or pressed from the top

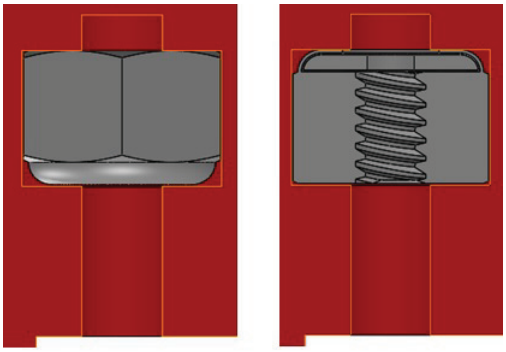


FIGURE 3. EMBEDDING PROCEDURE. AS IF PRINTED FROM BOTTOM TO TOP

layer into the part being printed. There has to be geometric compatibility between both parts. In fig. 3a we see that if the round surface of the lock nut is inserted into the bottom, then it is possible to geometrically match both parts. However, if the hexagonal flat surface is inserted into the bottom, then it is not possible to match both parts. While the bottom layer would be filled completely with the hexagonal flat surface, the top layer would only be filled partially with the round surface. This is shown in fig. 3b. While this problem would not be considerable in FDM manufacturing, it is of high importance in PolyJet manufacturing. An FDM printer would solve this problem by creating an overhang bridge. This is impossible in PolyJet machines [18]. We know that droplets will inevitable fall and adhere to the lock nut, thus creating a depression in the part's topology. Even though this might not have significant effects on this work, this event would be of paramount importance in PolyJet applications with larger gaps.

Slip sensor configuration

Fig. 4 shows an overview of the sensing application. We designed the sensing stage with a laser and a quadrant PIN photo diode. The selected configuration allows us to directly measure the shear strain exerted on the rubber strips. We propose a three layered "sandwich" structure. The top layer consists of a material that will provide a rough surface. This material must also be soft, in order to increase a ploughing, compliant effect. This layer would be in contact with the rib, and thus its purpose is to increase friction. The middle layer must be composed of a rigid material. From this layer, a structure will support a printed circuit board. This board will have the laser and connector terminals. The bottom layer must be composed of a highly elastic material. The working principle of this configuration lies on measuring the strain exerted on the bottom layer when the rib is gripped. The deformation suffered by the bottom layer will displace the

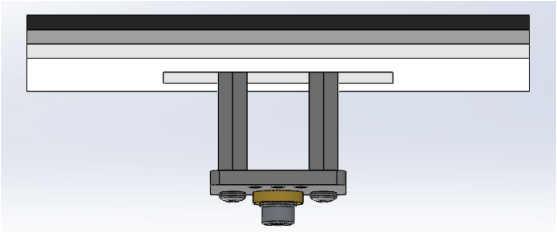


FIGURE 4. LASER "SANDWICH" CONFIGURATION.

medium layer. And thus, the laser will move accordingly. The necessity of the medium layer consisting of a rigid material arises with the purpose of minimizing ploughing in the bottom layer. If ploughing happens in the bottom layer, the laser would tilt. This is highly undesired for measuring purposes. Without ploughing, it will only suffer displacement in the same plane. This layered configuration will be held together by an adhesive and adhered to one grip. The other grip only requires any soft, and rough material for gripping purposes. The quadrant PIN photo diode is embedded by additive assembly in the dovetail part, directly facing the laser. For this purpose, we create a small window or hole in the latching spring.

Slippage is determined from anomalies in a measurement. Fig. 5 illustrates the expected behavior of the data obtained from the sensor during operation. As the rib is pressed to the right upon the sandwich structure, the laser is also displaced to the right. Once the latch is released, then power is removed from the sensor and we expect the signal to fall. However if the latch is not defeated and the signal falls, then it may be inferred that slippage has occurred. Furthermore, a rib may slip horizontally and still remain grasped. It is intended that the quadrant sensor determines this displacement. And thus, allow the operator to take the required action for safety purposes.

Fig. 6 shows an overview of our proposed solution for powering the laser. This approach relies on the transfer of electrical energy through embedded connections. Energy is sent to the printed circuit board through the connector terminals. Connected to each terminal block there is a wire. A total of three wires are used to power the chosen laser. Each wire is crimped on the other side with a ring terminal. These ring terminals are placed inside the grip and fastened with screw to a conductive female threaded hex standoff. These standoffs are embedded through additive assembly inside the grip that contains the laser. On the other side of the standoff there is a hex head conductive screw. This arrangement of components provides conductivity to a flat surface. This surfaces slides along the dovetail part during operation. We know by geometric constraint the exact location of the grips in the dovetails when an object is gripped. And because the mechanism itself is bi-stable, we are only concerned in measuring slippage at one point in space. Therefore, when the

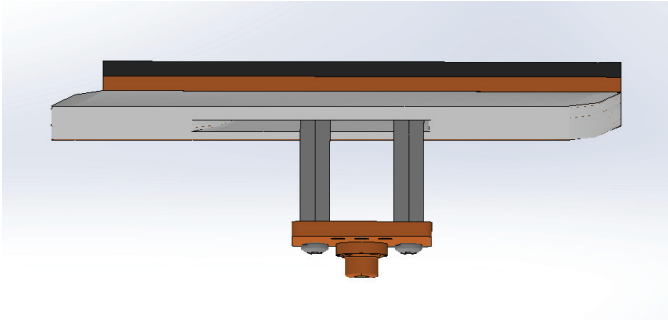


FIGURE 5. LASER EXPECTED BEHAVIOR AS THE RUBBER IS STRAINED TO THE RIGHT.

grips reach the end of the stroke, three spring loaded connectors make contact with the hex head conductive screws. The connectors are inserted in the dovetail part. The grip is then redesigned to give the spring loaded connectors a proper motion. This involves creating slopes in the grip's topology. The spring loaded connectors are sensitive to heat, and cannot be soldered with a soldering iron. For that reason they will be cold soldered to a wire with silver ink. The wires will be placed along slots that will be machined in the aluminum plate. Thus, they will reach an unobtrusive path to the main power supply.

Printing process

In the following section, we explain our procedure for additive assembly in PolyJet manufacturing.

Computer aided design

After redesigning the model with the proposed changes, further modifications were implemented to the CAD files. As with FDM, it is of high importance to avoid collision with the printing mechanism. In PolyJet manufacturing that mechanism makes physical contact with the part at every layer. If the embedded part is not placed at the right height, the roller may suffer from scraping. As a safety factor, we decided to increase the height of the material gaps created for embedding. The height was increased by one layer. Objet30 has a layer resolution in the z-axis of 28 micrometers, thus this was added to the design. During printing operation, it is expected that the printer's resolution does not match exactly the pause height. The reason being that the printer builds at discrete intervals of 28 micrometers and the embedded part may not exactly match this increase. This supports the idea that at least one layer of tolerance should be introduced. This quantity may be increased with highly irregular embedded part topologies.

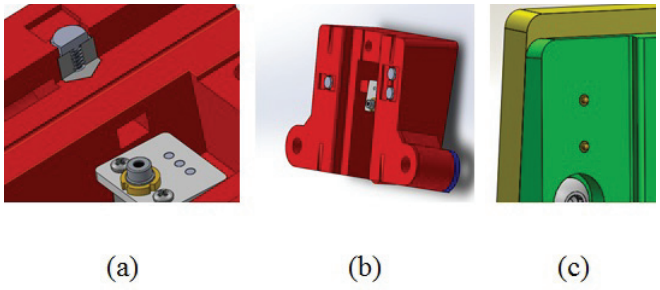


FIGURE 6. ELECTRICAL CONNECTIONS.

Pre-processing

It is undesirable to have high friction forces in a dovetail. Movement within the sliding mechanism should be smooth to reduce the work delivered by the actuators. It is in this interest that a glossy finish was selected in Objet Studio. The STL files were imported to the software and oriented in a way that satisfied the condition of parts being embedded parallel to the build plane. No further modifications were done to this configuration. A standard fill was chosen on default. The software automatically generates support material. It is important to note that it is not possible to pre-program a pause from Objet Studio.

Printing procedure

The printer's interface is shown on a computer. The interface displays information about current part height (in millimeters), current slice (layer) and time elapsed. Before printing the actual part, the mechanisms prints several layers of support material in the bottom of the part. After contacting Stratasys, it was known that the corresponding height of that material was one millimeter. This is a parameter of critical importance to the procedure. The pause height is then calculated adding the height of the material pre-built in the bottom of the part.

Estimating the pause time is not only dependant on the layer height but on the part's size in x and y directions, number of parts printed and number of required purge operations. These purge operations are undertaken by the printer to maintain a certain part quality. While the pause time could be estimated taking these factors into consideration, that was not attempted in this work. Nonetheless, the process was monitored remotely using TeamViewer software.

Once the operation progressed into the required layer or millimeter height, the printer was manually paused from the computer. Then, the mechanism finishes its current operation, makes a scanning operation and returns to its home position. The UV light is turned off and the door is unlocked. With a scraping tool, support material was removed from the gap where the part was to be embedded. Using a vacuum cleaner and scraping simultaneously was found helpful for the procedure. Furthermore, ex-

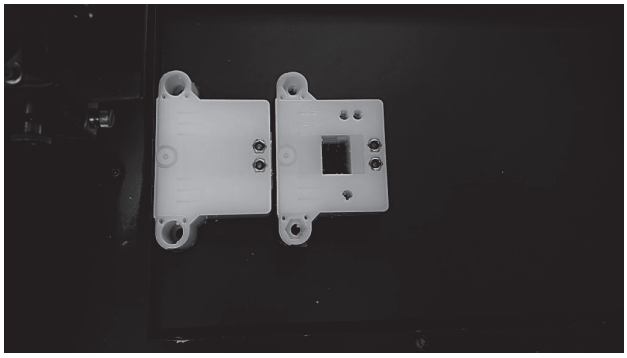


FIGURE 7. EMBEDDING LOCK NUTS.

tracting support material using a desoldering vacuum pump was found to be effective. After a considerable quantity of support material is removed, the embedded part is inserted into the 3D printed part. It is highly important to consider that the part is elevated by one millimeter of support material. If pressed too hard, then the support material is destroyed and the part loses its geometric fixation to printing mechanism. Hence, it is highly encouraged to press the part by hand and without the aid of any tool unless it is required for the procedure. Once the part is placed correctly embedded, the surface was briefly cleaned from excess of removed support material, and the printing operation was resumed.

Post processing

The parts were removed from the build tray using a flexible joint knife. Both grips were exposed to the conventional post-processing method discussed previously. However the dovetail part embedded with the quadrant PIN photo diode was not exposed to the water jet or caustic soda. Instead, it was manually scraped, and dried from excess humidity. Several attempts had to be done. In some cases, the scraping procedure deformed the geometry of the dovetail, and the mechanism suffered from severe stiffness. We found effective to lightly scrape manually the part, and let the last excess of support material to be removed by sliding the grip constantly into the dovetail. After a certain number of iterations, sliding motion was obtained.

Results

Fig. 8 shows the current assembled gripper. A conductivity test was done to examine a successful electrical connection during the grip operation.

Future Work

Although sliding motion was obtained, to reduce friction, design iterations are required to complete the end product. For this purpose, the quadrant PIN photo diode was not embedded into the dovetail part. After this is completed, we will adhere the rubber sandwich, and the latching springs. Finally, the integrated system will be tested.

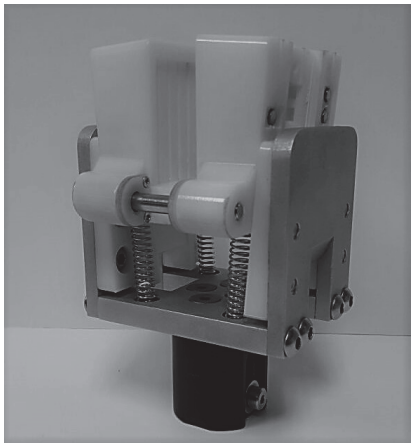


FIGURE 8. CURRENT ASSEMBLED GRIPPER.

Additionally, we are working on embedding an electrical interface to the dovetail part to transfer electrical energy to the quadrant PIN photo diode (which was not addressed in this work). We currently are attempting to embed spring loaded connectors into the dovetail part for this purpose.

Conclusion

As additive manufacturing technologies progress, it is important to explore the distinct implementations that can be developed. These implementations may allow us to rethink the way we see manufacturing and will define the future of additive manufacturing itself. This work for example, presents a need of software tools that allow pre-programmed pauses. Such necessity might result on the development of such. Currently, additive assembly is constrained by additive manufacturing being layer based, however objects with much more complex geometries may be embedded with layer-less additive manufacturing.

Implementation procedures with additive assembly will then be highly process dependant, hence new methods will be required for different AM technologies. Mass customization might one day require printers to have autonomous methods for embedding objects and delivering fully functional parts directly to the end user. The purpose of this work is to encourage the recognition of such methods and to inspire designers and engineers on the benefits of additive assembly.

ACKNOWLEDGMENT

The authors would like to thank Dr. John Dolan, Dr. Drew Bagnell and Rachel Burcin for the direction of the RISS program. Furthermore, we would like to thank Carnegie Mellon University for providing the infrastructure required to develop this project. Finally, we would like to thank Tecnolgico de Monterrey for providing sponsorship and making this work possible.

REFERENCES

[1] European Additive Manufacturing Group, 2015. Additive manufacturing technology.

[2] Wohlers, T. T., 2014. *Wohlers report 2014: 3D printing and additive manufacturing state of the industry annual worldwide progress report*. Wohlers Associates.

[3] Bartolo, P., 2011. *Stereolithography: materials, processes and applications*. Springer.

[4] ISO/ASTM, 2016. Standard terminology for additive manufacturing - general principles - terminology, May.

[5] Fischer, F., 2013. Fdm and polyjet 3d printing, Oct.

[6] Stratasys, 2015. Objet30, Jul.

[7] Cazn, A., Morer, P., and Matey, L., 2014. "Polyjet technology for product prototyping: Tensile strength and surface roughness properties". *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture,* ***228***(12), pp. 1664–1675.

[8] Vu, I., Bass, L., Meisel, N., Orler, B., Williams, C. B., and Dillard, D. A., 2015. Characterization of mutli-material interfaces in polyjet additive manufacturing, Oct.

[9] Bass, L. B., Meisel, N. A., and Williams, C. B., 2015. Exploring variability in material properties of multi-material jetting parts, Oct.

[10] Stratasys, 2015. Embedding hardware, Feb.

[11] Ota, H., Emaminejad, S., Gao, Y., Zhao, A., Wu, E., Challa, S., Chen, K., Fahad, H. M., Jha, A. K., Kiriya, D., Gao, W., Shiraki, H., Morioka, K., Ferguson, A. R., Healy, K. E., Davis, R. W., and Javey, A., 2016. "Application of 3d printing for smart objects with embedded electronic sensors and systems". *Advanced Materials Technologies,* ***1***(1), pp. 1600013–n/a. 1600013.

[12] Aguilera, E., Ramos, J., Espalin, D., Cedillos, F., Muse, D., Wicker, R., and MacDonald, E., 2013. 3d printing of electro mechanical systems, Oct.

[13] ISO/ASTM, 2016. Standard specification for additive manufacturing file format (amf) version 1.1, May.

[14] ISO/ASTM, 2016. Standard terminology for additive manufacturing-coordinate systems and test methodologies, May.

[15] Stratasys, 2015. Objet waterjet, Dec.

[16] Stratasys, 2015. Guide to basic post process applications - objet line of 3d printers, Oct.

[17] Bourne, D., Choset, H., Hu, H., Kantor, G., Niessl, C., Rubinstein, Z., Simmons, R., and Smith, S., 2015. "Mobile manufacturing of large structures". In 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 1565–1572.

[18] ISO/ASTM, 2015. Standard guidelines for design for additive manufacturing, Mar.

Using a Locally Connected Autoencoder to Identify Significant SNPs in *Sorghum bicolor*

Mihael Cudic
Field Robotics Center
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

***Abstract*—With advances in genome sequencing, we are now able to gather large amounts of DNA data in more cost effective and efficient manners. However, mapping the genome to known phenotypes has still proved difficult. Association mapping algorithms, such as Efficient Mixed-Model Association (EMMA), have become the standard in understanding the correlations between genotype and phenotype but suffer greatly from finding spurious associations. We propose an alternative genome wide assocation study (GWAS) that is averse to overfitting using a locally connected autoencoder (LCA). We demonstrate the LCA's ability to extract unsupervised features that directly correlate to measured phenotype values, thus, allowing us to isolate significant SNPs. All experiments were done using 345 *Sorghum bicolor* individuals with allele panels containing 232,303 SNPs. Five markers - S1_18096891, S2_2633689, S2_2793224, S6_61098274, S6_7123769 - were identified and thought to be associated with fresh weight and stalk height.**

***Index Terms*—Locally Connected Autoencoder, Genome Wide Association Studies**

I. Introduction

The genome encodes all information necessary to fully reconstruct the biological properties of an organism. Thus, understanding the role of each allele or sequence of alleles has tremendous scientific, medicinal and humanitarian significance. Huntington Disease, for example, is a heritable brain disorder that has been linked to multiple repetitions of cytosine, adenine, and guanine in humans' Huntingtin gene (*HTT*) found in chromosome 4 [1]. Likewise, *D8* and *Mpl1* mutations in maize (*Zea mays L.*) have be linked to dwarfism traits leading to reduced maize yield [2]. By isolating these significant genotype, further research can be done to target specific locations and augment genomes to potentially eradicate diseases or alter certain properties in organisms.

However, understanding the genome in its entirety is a fundamental challenge for computational biology as the sheer size of some genomes make it virtually impossible to perform any analysis. The human genome contains roughly 3 billion base pairs (bp) while the genome of some plants, like the *Paris japonica*, are 150 billion bp [3]. Fortunately, adjacent alleles tend to be correlated with one another and a significant portion of alleles are consistent for all individuals. Thus, by randomly sampling alleles throughout the genome and by keeping only those that have variation across a population of individuals, or Single Nucleotide Polymorphisms (SNPs), we are able to down sample the genome while retaining most of its information.

From these downsampled allele panels, genome wide association studies (GWAS) can then be used to associate alleles with measured phenotypes. Efficient Mixed-Model Association (EMMA) has become the standard GWAS algorithm due to its simplicity and numerous successes in finding significant SNPs [4]. Similarly, more recent attempts using Artificial Neural Networks (ANNs) have shown promising results due to its non-linear capabilities [5][6][7][8][9]. However, any regression task using a high dimensional input space has a tendency to overfit and find false positive associations. Therefore, to truly understand the features in an allele panel, an unsupervised approach is required.

To our knowledge, we propose the first successful implementation of an autoeconder for a high dimensional allele panel and demonstrate its usefulness in identifying SNPs. Unlike supervised learning systems, the autoencoder is forced to learn features in the allele space negating any potential for overfitting in the phenotype space. However, we still show that dimensions in the autoencoder are able to separate phenotypes into statistically significant clusters. By backpropagating the error gradients of correlated encoding dimensions, we can then identify significant SNPs [10].

Our analysis was done on *Sorghum bicolor* accessions provided by the Institute for Translational Genetics at Clemson University, South Carolina [11]. Sorghum has emerged as a leading candidate for bioenergy feedstock due to its ease of modification, diversity, and efficiency. If enough is known about Sorghum, different varieties can be crossbred or genetically modified to maximize biofeul production. The allele panels contain 345 accessions with 232,303 SNPs for each accession. Using our locally connected autoencoder (LCA), we show that we can isolate SNPs that directly correspond to various Sorghum phenotypes including fresh weight, stalk height, and lignin percentage. We identify five additional SNPs for further inspection.

II. Related Works

A. Genome Wide Association Studies

The simplest method to perform GWAS is through case-controlled studies that use statistical tests for independence by analyzing each individual allele's contribution to a phenotype.

However, treating each allele as independent leads to high false positive rates as it ignores population structure and genetic relatedness. To account for inbred populations, structured association models were proposed that require similar tests to be consistent throughout each subpopulation [12]. Likewise, principal component analysis (PCA) can also be used to correct for biases from predefined structure [13].

However, these solutions only partially account for genetic similarities and, therefore, more sophisticated models are necessary. By incorporating a kinship matrix describing pairwise genetic relatedness between all similar individuals as well as having several optimization parameters, Mixed Linear Models (MLM) are shown to be successful in associating quantitative phenotypes to genotypes given population structure [14]. Due to its computational efficiency and ability to account for internal structures, EMMA and other derivatives have been used for the foundation of most GWAS algorithms [4][15][16]. Specifically for high dimensional allele panels, Compressed Mixed Linear Models (CMLM) are used as they significantly reduce time complexity compared to other EMMA alternatives [17].

In addition, deep learning has been increasingly used for bioinformatics due to its notable success in various fields such as computer vision and game AIs [18] [19]. Most notably, DeepSea uses convolutions on DNA sequence of noncoding variants to predict traits with single nucleotide sensitivity [5]. An iteration of DeepSea, deepWAS, looks to perform explicit GWAS to associate SNPs to major depressive disorders. Similarly, DeepMotif uses convolutions to extract patterns in the DNA sequences that correlate to phenotype classes [8]. However, these methods don't take into account the internal structure of the genetic information and are heavily prone to overfitting.

B. Unsupervised

Although supervised methods are most frequently used, unsupervised approaches have seen success in extracting features from complex systems. Le et al. demonstrated that a deep autoecoder could separate features, such as a human or a cat, using 10 million images with no labels [20]. A similar application has been applied to genetics. Tan et al. used Denoising Autoencoders (DAs) to classify breast cancer tumors by extracting relevant features in the encoding layers [21]. Likewise, ADAGE shows a methodology for using autoencoders on gene expression data and demonstrates the ability to isolate high contributing inputs for further analysis [22]. Although there have been promising results using autoencoders for genetic based problems, all approaches that we know of have been applied to gene expression data and not allele panels.

III. Dataset

The *Sorghum bicolor* data used for GWAS consists of 345 individuals with 232,303 SNPs across 10 chromosomes. Although the vast majority of sampled SNPs have no known associations to phenotype, some SNPs come from previously mapped dwarfism genes (*DW1*) and maturity genes (*MA1*, *MA3*, and *MA6*) [11].All missing SNP values were imputed resulting in data falling into three classes: major allele, mixed, and minor allele. 95.62% of the data can be described either in the first or last class. However, the distribution of major and minor alleles is heavily skewed as seen in figure 1. Because it is hard to create generalizable associations given a few instances of an allele, we only used SNPs with a minor allele frequency greater than .1 leaving 125,980 SNPs for analysis.

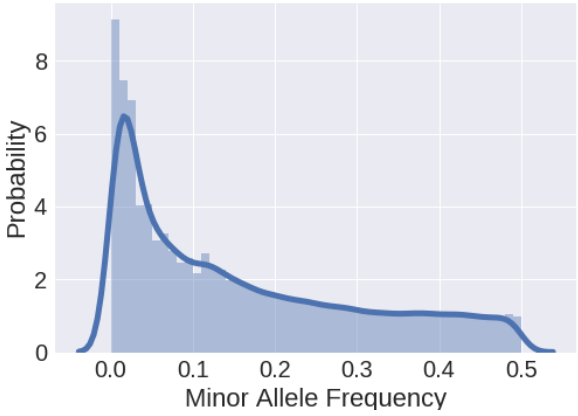


Fig. 1. The distribution of minor allele frequencies across all 232,303 SNPs. Only SNPs with a minor allele frequency above .1 were used for analysis to ensure more evenly distributed alleles.

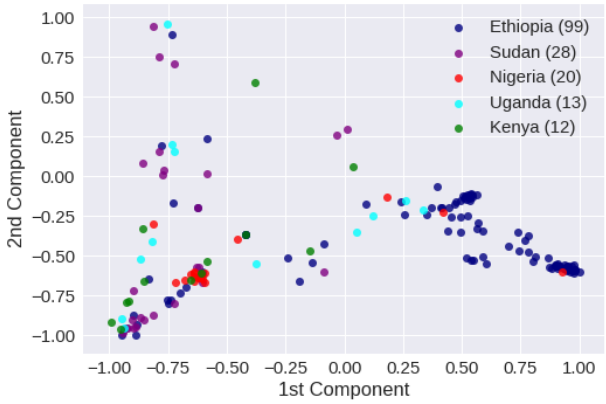


Fig. 2. The first and second normalized principal components for individuals originating from the 5 most reoccurring regions: Ethopia, Sudan, Nigeria, Uganda, Kenya. PCA was performed over 125,980 SNPs for all 345 individuals. The number of individuals from each region can be seen in the legend.

In addition, the Sorghum originated from 37 locations all over the world with only 87 individuals originating from unknown locations. However, roughly half of the individuals originated from only 5 locations: Ethiopia, Sudan, Nigeria, Uganda, and Kenya. Because the Sorghum was not sampled evenly throughout the world, there lies a high chance of evolutionary inbreeding and thus genetic relatedness. To visualize the significance of these sub-populations, we plotted the first and second principal components from individuals originating

from these 5 regions. As shown in figure 2, Ethopian and Nigerian plants seem to cluster the most with some clustering forming with Sudanese plants. However, PCA is only a linear dimensionality reduction technique and the principal components are skewed due to the varying distribution of origins. Thus, not all sub-population structures can be seen.

IV. Methodology

Although we were able to reduce the dimensionality of the allele panel from 232,303 SNPs to 125,980 SNPs, the input dimensionality was still significantly larger than the number of individuals. This low-sample high-dimensionality combination limits any significant analysis as it inflates the false positive rate and the possibility for overfitting. Thus, we wanted to further reduce the dimensionality by extracting features in the panel.

Autoencoders provide non-linear dimensionality reduction and have been successful in many instances [20]. However, the input dimensions were still too large to construct an autoencoder. Thus, we resorted to localized connections in the first and final layer as shown in figure 3. Similar architectures have been used before, but none have taken advantage of locally connected layers' ability to downsample and upsample with precision [20][23]. Unlike regular convolutional autoencoders that look to encode data with spatial correlation, each dimension of the allele panel must be treated as independent. In addition, this biologically inspired structure bodes well for genetic data as adjacent markers are more likely to be correlated than well separated markers. Thus, each locally connected node can hopefully represent the associating alleles more accurately. Overlap between locally connected nodes allow for more flexibility and information, providing a medium between reduced parameters and fully connected layers. Overlap is especially important in the upsampling layer as it allows output nodes to form more than one connection. Finally, the middle two layers of the system are fully connected to ensure the allele panels are encoding global information.

To prevent the LCA from learning a one-to-one mapping, the encoding layer must be less than 345 nodes. We did not know how many encoding nodes would generate the best separation among phenotypes, so we experimented with 256, 128, and 64 encoding nodes. Sigmoid and tanh activation functions were tested separately in the encoding layer to create discrete clusters for further analysis. In addition, we used leaky relu functions with an alpha of .05 throughout the LAE to ensure no nodes were lost. A sigmoid activation function was used for the output allele classification.

Inputs were encoded as a 0, 0.5, and 1 for major allele, mixed, and minor allele respectively. All 345 individuals were used for training and the LCA was trained using binary crossentropy across all 125,980 SNPs. However, errors were zeroed in an output node if a mixed SNP was present ensuring that all outputs were trained on binary data. Likewise, mixed outputs were not used to calculate classification scores. To prevent large errors caused from the high-dimensional output, a small step size of 5.0e-5 was used with a decay rate of .98 for every 1500 updates. Additionally, Adam optimizations were used with a batch size of 8 [24]. The system was stopped with 30 epochs of no improvement in allele decoding classification on the training set.

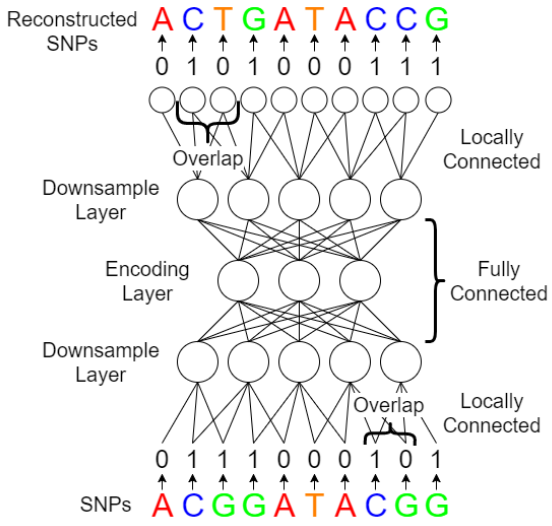
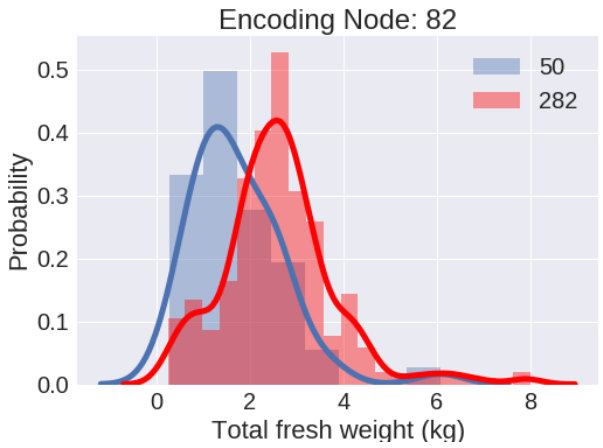


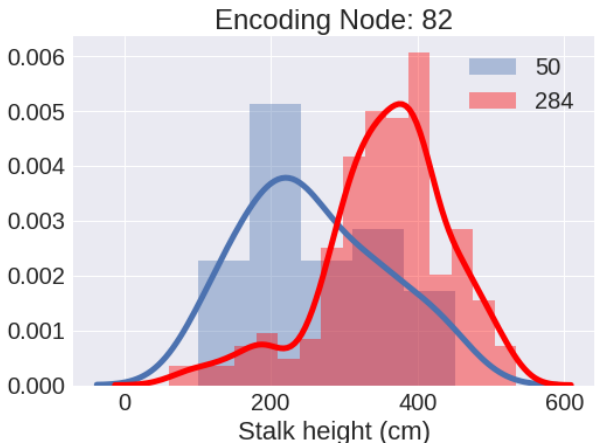
Fig. 3. A locally connected autoencoder was used to extract features from the allele panel. Only the first and last layers are locally connected to function as downsampling and upsampling layers respectively. Each downsample node corresponds to 40 core SNPs but has an additional overlap of 100 SNPs to the left and right of the core grouping resulting in a 3150 sized layer. The upsample layer follows an inverse structure. The encoding layer was tested using various dimension sizes - 64, 128, 256 - and activations - sigmoid or tanh. The outputs are sigmoid activations.

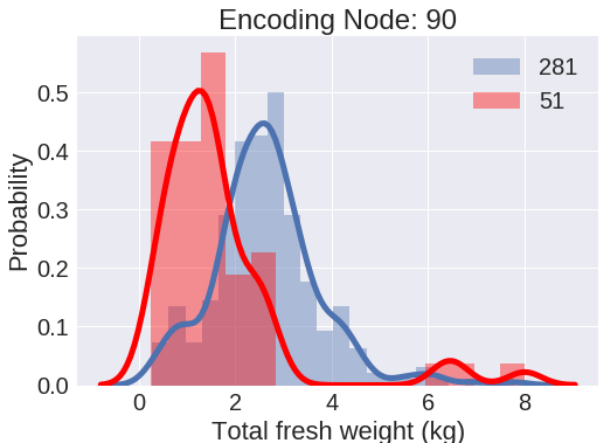
TABLE I
Performance of LCA with various encoding hyper-parameters

Encoding Nodes	Accuracy sigmoid	Accuracy tanh	p-test sigmoid	p-test tanh
64	84.05%	83.73%	4.10e-18	6.10e-17
128	87.56%	85.05%	6.88e-14	1.36e-17
256	89.48%	85.61%	1.03e-15	1.73e-17

Once fully trained, encoding dimensions were then binarized and saved for all individuals. Each encoding node was tested across all phenotypes to determine their ability to cluster into significantly separate distributions. Because we were interested in mean-based separation, we performed p-tests between clustered distributions to quantify the significance of the separation. All p-tests that scored below 5.0e-5 were then examined individually to ensure significant separation. Once the correlation between the encoding node and phenotype was confirmed, contribution of each allele was calculated by backpropogating a 1.0 error for the node in question as outline by Dimopoulos et al. [10]. Due to the skew in minor allele frequency, each calculated sensitivity was also multiplied by the minor allele frequency to generate an average allele contribution across all individuals.







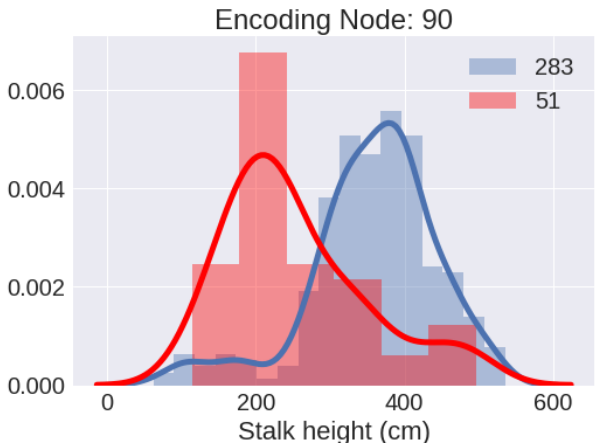


Fig. 4. Clusters created by encoding node 82 and 90 on total fresh plant weight data. The individuals per distribution can be seen in the legend. This phenotype separation was achieved through an unsupervised autoencoder demonstrating the systems ability to extract meaningful features.

Fig. 5. Clusters created by encoding node 82 and 90 on stalk height data. The individuals per distribution can be seen in the legend. This phenotype separation was achieved through an unsupervised autoencoder demonstrating the systems ability to extract meaningful features.

V. Results

Since we did not know the optimal hyperparameters for the encoding layer, we tested various activation functions and sizes. As shown in table I, more encoding nodes led to better reconstruction classification as each node encoded more resolved features. Additionally, sigmoid activations always performed better than tanh activations for classification receiving a maximum reconstruction accuracy of 89.48%. When analyzing activations across all encoding nodes, it seemed that sigmoid activations were more equally distributed than their tanh counterpart. Thus, sigmoid activations better optimize the use of each nodes.

However, our ultimate goal was not to encode allele panels with high accuracy but rather to extract meaningful features that can be expressed in the phenotype space. To quantify the usefulness of the features learned, p-test were performed on all cluster separations. The average p-test score over the 10 best separations for each LCA is shown in table I as well. Unlike classification, we can see that tanh activations generally perform best. This is likely due to skewed activations learned as equal subpopulations are rare to find in nature. Because all p-test scores were very low, addition visual analysis was done on the best 50 separations. As a result, a 256 encoding layer with tanh activations best separated phenotype data and thus will be used for the remainder of this study.

The most significant phenotype separations found using said structure can be seen in figure 4 and 5. Encoding dimensions 82 and 90 seemed to separate phenotypes measuring total fresh plant weight, stalk height and lignin percentage while encoding dimensions 30 seemed to correlate well with anthesis date and NFC percentage. Due to space constraints, only 4 distributions were shown. However, it is also imperative that the clusters are independent of population structure. Thus, Chi-squared tests were implemented based of the individuals origin. All 4 seperations had statically insignificant p-values showing that

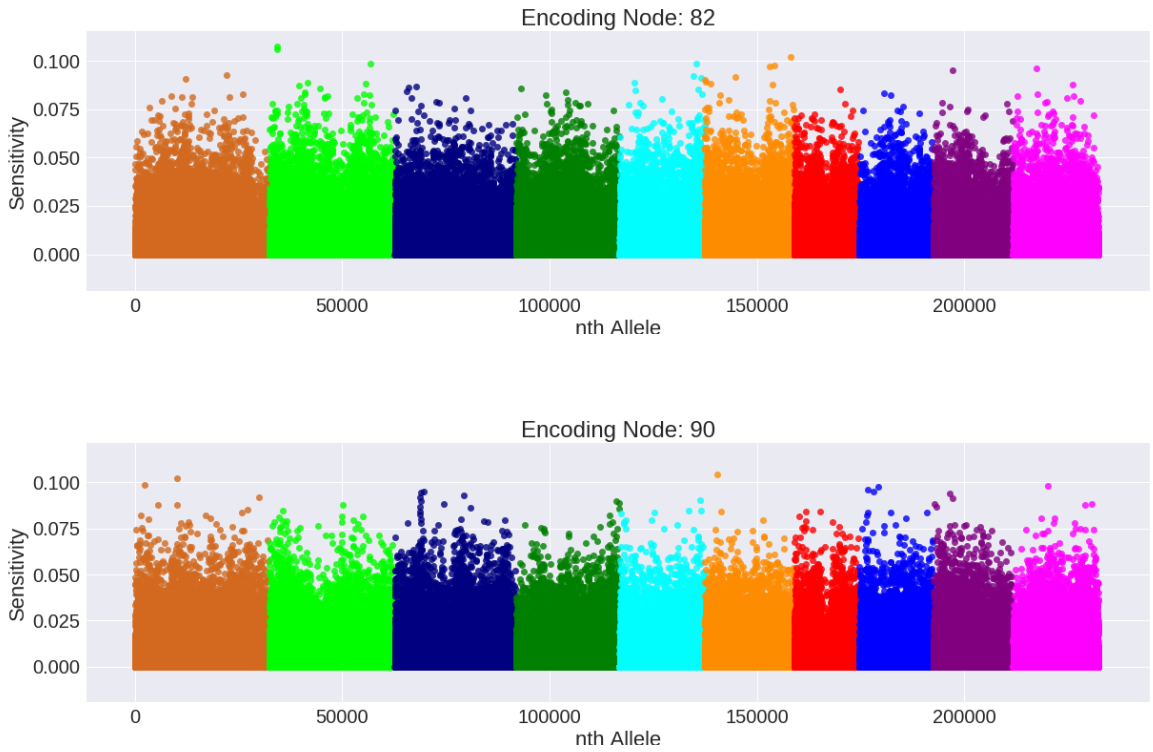


Fig. 6. Manhattan plots of the SNP's relative sensitivity to encoding node 82 and 90. Alleles with sensitivities above 0.1 were isolated as significant SNPs for further analysis. Although both plots highlight different regions, we can see a few SNPs have significantly higher sensitives than the rest: S1_18096891, S2_2633689, S2_2793224, S6_61098274, and S6_7123769. Note: each color corresponds to a new chromosome.

these features are true to the trait and not the region, validating the LCA's ability to take into account unknown population structures.

To ensure that we were extracting real features and not finding random separations, the LCA was rerun with new initializations but found the same phenotype separations. In addition, we generated a new randomized allele panel with the same dimensions and minor allele frequencies as the *Sorghum bicolor* panel. The LCA only learned the baseline reconstruction accuracy of 73.4% and was thus unable to find any statistically significant phenotype separations or SNPs. However, this experiment was also replicated using a CMLM and roughly 0.967% of SNPs were found to be associated with fresh weight using a .01 p-value cut off. These results are to be expected as the LCA cannot encode random data and relies on multiple corresponding alleles to extract features whereas MLMs look at a single SNP's correlation to quantitative phenotype. As a result, this validates the significance of the phenotype separation found as the LCA is able to avoid false positives.

All phenotypes analyzed are of interest for further analysis, but for the sake of this study we specifically looked to encoding nodes 82 and 90 as they relate to biofeul production. By backpropogating the gradient to each SNP and multiplying it by the minor allele frequency, we can perform a sensitivity analysis for each encoding node. Unlike EMMA where SNPs are given a p-value describing its significance, we prescribe relative sensitivities to each SNP as as shown in figure 6. Therefore, there is no intuitive cutoff for separating significant and non-significant SNPs. However, when looking specifically at these two nodes, only 5 SNPs had relative sensitivities greater than 0.10: S1_18096891, S2_2633689, S2_2793224, S6_61098274, and S6_7123769. Therefore, a cutoff of .10 was used for this study but SNPs slightly below might still be of interest.

VI. Conclusion

In this study, we propose a novel method to associate SNPs to phenotypes using LCAs. We show that LCAs are able to find redundancies in the allele panels and extract meaningful features that separate individuals in the phenotype space. This method avoids any possibility for overfitting and is thus averse to high false positive rates while still being able to account for sub-population structures. We conclude the analysis by identifying 5 markers that correspond to the fresh weight and stalk height of *Sorghum bicolor*. Further analysis needs to be done to verify the results.

However, to validate our methodology for GWAS, it must be applied to different allele panels to show reproducibility. In addition, one of the major challenges of this method is handling SNPs with highly skewed minor allele frequencies. Because such SNPs occur frequently, we filtered nearly half of

the SNPs in the original panel losing potential candidate genes for classification. Thus, an algorithm averse to these skewed distributions must be considered for further research.

REFERENCES

[1] M. E. MacDonald, C. M. Ambrose, M. P. Duyao, R. H. Myers, C. Lin, L. Srinidhi, G. Barnes, S. A. Taylor, M. James, N. Groot *et al.*, "A novel gene containing a trinucleotide repeat that is expanded and unstable on huntington's disease chromosomes," *Cell*, vol. 72, no. 6, pp. 971–983, 1993.

[2] N. P. Harberd and M. Freeling, "Genetics of dominant gibberellin-insensitive dwarfism in maize." *Genetics*, vol. 121, no. 4, pp. 827–838, 1989.

[3] J. Pellicer, M. F. Fay, and I. J. Leitch, "The largest eukaryotic genome of them all?" *Botanical Journal of the Linnean Society*, vol. 164, no. 1, pp. 10–15, 2010.

[4] H. M. Kang, N. A. Zaitlen, C. M. Wade, A. Kirby, D. Heckerman, M. J. Daly, and E. Eskin, "Efficient control of population structure in model organism association mapping," *Genetics*, vol. 178, no. 3, pp. 1709–1723, 2008.

[5] J. Zhou and O. G. Troyanskaya, "Predicting effects of noncoding variants with deep learning–based sequence model," *Nature methods*, vol. 12, no. 10, p. 931, 2015.

[6] D. Kleftogiannis, P. Kalnis, and V. B. Bajic, "Deep: a general computational framework for predicting enhancers," *Nucleic acids research*, vol. 43, no. 1, pp. e6–e6, 2014.

[7] B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey, "Predicting the sequence specificities of dna-and rna-binding proteins by deep learning," *Nature biotechnology*, vol. 33, no. 8, pp. 831–838, 2015.

[8] J. Lanchantin, R. Singh, Z. Lin, and Y. Qi, "Deep motif: Visualizing genomic sequence classifications," *arXiv preprint arXiv:1605.01133*, 2016.

[9] G. Eraslan, J. Arloth, J. Martins, S. Iurato, D. Czamara, E. B. Binder, F. J. Theis, and N. S. Mueller, "Deepwas: Directly integrating regulatory information into gwas using deep learning supports master regulator mef2c as risk factor for major depressive disorder," *bioRxiv*, p. 069096, 2016.

[10] I. Dimopoulos, J. Chronopoulos, A. Chronopoulou-Sereli, and S. Lek, "Neural network models to study relationships between lead concentration in grasses and permanent urban descriptors in athens city (greece)," *Ecological modelling*, vol. 120, no. 2, pp. 157–165, 1999.

[11] Z. W. Brenton, E. A. Cooper, M. T. Myers, R. E. Boyles, N. Shakoor, K. J. Zielinski, B. L. Rauh, W. C. Bridges, G. P. Morris, and S. Kresovich, "A genomic resource for the development, improvement, and exploitation of sorghum for bioenergy," *Genetics*, vol. 204, no. 1, pp. 21–33, 2016.

[12] J. K. Pritchard, M. Stephens, N. A. Rosenberg, and P. Donnelly, "Association mapping in structured populations," *The American Journal of Human Genetics*, vol. 67, no. 1, pp. 170–181, 2000.

[13] A. L. Price, N. J. Patterson, R. M. Plenge, M. E. Weinblatt, N. A. Shadick, and D. Reich, "Principal components analysis corrects for stratification in genome-wide association studies," *Nature genetics*, vol. 38, no. 8, p. 904, 2006.

[14] M. Malosetti, C. G. van der Linden, B. Vosman, and F. A. van Eeuwijk, "A mixed-model approach to association mapping using pedigree information with an illustration of resistance to phytophthora infestans in potato," *Genetics*, vol. 175, no. 2, pp. 879–889, 2007.

[15] X. Zhou and M. Stephens, "Genome-wide efficient mixed-model analysis for association studies," *Nature genetics*, vol. 44, no. 7, pp. 821–824, 2012.

[16] H. M. Kang, J. H. Sul, N. A. Zaitlen, S.-y. Kong, N. B. Freimer, C. Sabatti, E. Eskin *et al.*, "Variance component model to account for sample structure in genome-wide association studies," *Nature genetics*, vol. 42, no. 4, pp. 348–354, 2010.

[17] Z. Zhang, E. Ersoz, C.-Q. Lai, R. J. Todhunter, H. K. Tiwari, M. A. Gore, P. J. Bradbury, J. Yu, D. K. Arnett, J. M. Ordovas *et al.*, "Mixed linear model approach adapted for genome-wide association studies," *Nature genetics*, vol. 42, no. 4, pp. 355–360, 2010.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[19] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[20] Q. V. Le, "Building high-level features using large scale unsupervised learning," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 8595–8598.

[21] J. Tan, M. Ung, C. Cheng, and C. S. Greene, "Unsupervised feature construction and knowledge extraction from genome-wide assays of breast cancer with denoising autoencoders," in *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, vol. 20. NIH Public Access, 2015, p. 132.

[22] L. Chen, C. Cai, V. Chen, and X. Lu, "Learning a hierarchical representation of the yeast transcriptomic machinery using an autoencoder model," *BMC bioinformatics*, vol. 17, no. 1, p. S9, 2016.

[23] E. Hosseini-Asl, R. Keynton, and A. El-Baz, "Alzheimer's disease diagnostics by adaptation of 3d convolutional network," in *Image Processing (ICIP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 126–130.

[24] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

Evaluation of Explicit Experience-driven Predictive Control on a Computationally Constrained Platform

Mosam Dabhi, Vishnu R. Desaraju, and Nathan Michael

***Abstract*— This paper details experimental evaluation of Experience-driven Predictive Control (EPC) on a system with severely constrained computational resources. This work leverages an explicit EPC formulation to solve the nonlinear model predictive control (NMPC) problem with low computational overhead [1]. A pre-constructed controller database allows explicit EPC to avoid solving redundant optimization problems. We demonstrate the performance of explicit EPC through a set of experimental flight trials on a quadrotor equipped with a PX4 autopilot [2] and running the algorithm on a real-time operating system.**

I. INTRODUCTION

Quadrotors are employed in critical applications such as inspection and search and rescue missions that require stable and accurate control performance [3]. As quadrotors exhibit constrained nonlinear dynamics, nonlinear model predictive control (NMPC) is an ideal solution to achieve stable and accurate flight. However, NMPC is computationally expensive and thus infeasible on vehicles with limited onboard processing [4]. The Experience-driven Predictive Control algorithm (EPC) addresses this issue by creating an experience database that yields fast solutions to NMPC problems with uncertain dynamics [5]. However, to run EPC online, we require a host computer onboard the robot that is able to solve the NMPC problem, recast as a quadratic program. In this work, we transition the controller to an embedded system via the explicit EPC algorithm to enable application of the EPC algorithm without requiring a host computer (and thus permitting application on a larger class of systems).

Achieving real-time execution of the EPC algorithm on a computationally constrained embedded level platform is a challenge [6]. As the embedded level system is responsible for various tasks such as data acquisition of multiple sensors, navigation, state-estimation, control, data-fusion and synchronization, the system must be able to respond and calculate control commands within predictable time limits for stable behaviour [7]. The onboard embedded software system is developed under Real-time Operating System (RTOS) supplying a mixture of essential system functions for real-time operations such as: multitasking; threading; rapid context switching; and preemptive scheduling. Although any of the popular RTOS would be sufficient, such as VxWorks employed in Carnegie Mellon University [8], Linux in University of Southern California [9], and QNX in [7, 10], we use the POSIX based PX4 flight management unit as it is open-source and widely used in UAV research.

The authors are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA. {mdabhi,rajeswar,nmichael}@cs.cmu.edu

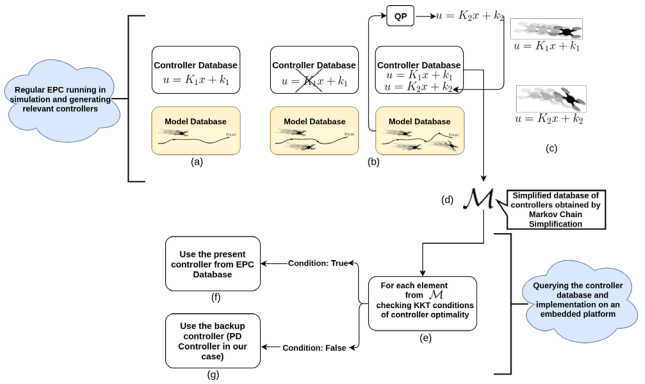


Fig. 1: Overview of the proposed approach that constructs a synthetic experience database consisting of parameterized feedback controllers (**a-c**), followed by database simplification and deployment of a simplified explicit controller database on an embedded platform (**d-g**).

The explicit EPC algorithm comprises of two parts: **1)** solving the nonlinear model predictive control (NMPC) problem with uncertain system dynamics online and creating a database of relevant controllers [5], and **2)** leveraging this database, which consists of reusable affine feedback controllers parameterized by system dynamics, to compute control commands rapidly that recover NMPC performance.

As the embedded level platform is severely constrained in its computational capabilities [11], we cannot execute the first component of the EPC algorithm online. However, we can generate the controller database of EPC for various scenarios using a simulation environment that is run offline to generate synthetic experiences. The database is then leveraged online for the second step at the embedded level. We refer to this synthetic experience-based approach as explicit EPC, and it permits implementation onboard the robot [1] as shown in Fig. 1.

In this work, we demonstrate the formulation and deployment of the explicit EPC algorithm to solve the NMPC problem on a computationally constrained system. To show that the explicit formulation gives comparable performance to the online version, we compare the controller performance running on the embedded micro-controller (without a host computer) with the results from running it on the host computer. Section II-A details the EPC approach that constructs an experience database consisting of parameterized feedback controllers and dynamic models via synthetic experience from a simulation environment [5]. Section II-B details a database simplification strategy using a Markov chain that

is implemented on an embedded platform [1]. The locally optimal controller is selected online by checking the KKT conditions for optimality [12]. Section III presents experimental results with a computationally constrained quadrotor that validate algorithm performance.

II. Approach

In this section, we detail the Explicit Experience-driven Predictive Control (EPC) approach [1]. This approach is divided into two parts.

1) Running a series of simulations offline using the EPC algorithm [5] to accumulate synthetic experiences and construct a database of relevant controllers.
2) Applying probabilistic database simplification and deploying the simplified explicit controller database on an embedded platform.

Figure 1 illustrates the algorithm applied to a quadrotor micro air vehicle (MAV). The first step simulates the MAV tracking a variety of trajectories. While the MAV operates away from the constraint boundaries (Fig. 1(**a**)), the EPC algorithm enables it to apply a controller in the database while the dynamics model continues to be updated. Figure 1**(b)** shows that a new controller is added to the experience database as the MAV transitions to a more aggressive flight and the updated dynamics model predicts that the system state is approaching a constraint boundary. Figure 1**(c)** shows that the MAV reuses controllers in the database based on updated state predicted by current estimate of its dynamics model. Through a Markov chain simplification strategy [1], we obtain a simplified database of controllers defined as a mapping $\mathcal{M}$ (Fig. 1(**d**)). In the second step, this offline-constructed database can be queried online on the embedded system for better control, equivalent to solving the EPC problem but requiring no online optimization. We query $\mathcal{M}$ (Figs. 1(**e**)-(**g**)) in each control iteration to identify and implement the appropriate controller by evaluating the KKT conditions (with PD controller as a backup/safety controller).

A. Receding-Horizon Control Formulation

To obtain synthetic experiences from simulation, we start with a nonlinear dynamics model $\dot{\mathbf{x}} = f(\mathbf{x},\mathbf{u})$ and linearize about the current state and control, $\mathbf{x}_0$ and $\mathbf{u}_0$, to obtain a linear approximation of the dynamics,

$$\bar{\mathbf{x}}_{k+1} = \mathbf{A}\bar{\mathbf{x}}_k + \mathbf{B}\bar{\mathbf{u}}_k \tag{1}$$

Taking the current state as the nominal state, $\bar{\mathbf{x}}^* = \mathbf{x}_0$ with given N reference states $\mathbf{r}_1$,...,$\mathbf{r}_N$, let $\bar{\mathbf{r}} = \mathbf{r} - \mathbf{x}^*$. We formulate this control problem as a quadratic,

$$\begin{aligned}\underset{\bar{\mathbf{u}}_k}{\operatorname{argmin}} \quad & \sum_{k=0}^{N-1} \frac{1}{2}(\bar{\mathbf{x}}_{k+1} - \bar{\mathbf{r}}_{k+1})^{\mathrm{T}}\mathbf{Q}(\bar{\mathbf{x}}_{k+1} - \bar{\mathbf{r}}_{k+1}) \\ & + \frac{1}{2}(\bar{\mathbf{u}}_k - \bar{\mathbf{u}}_{\hat{\mathbf{y}}})^{\mathrm{T}}\mathbf{R}(\bar{\mathbf{u}}_k - \bar{\mathbf{u}}_{\hat{\mathbf{y}}}) \\ \text{s.t.} \quad & \bar{\mathbf{x}}_{k+1} = \mathbf{A}\bar{\mathbf{x}}_k + \mathbf{B}\bar{\mathbf{u}}_k \\ & \mathbf{G}_{\mathbf{x}}\bar{\mathbf{x}}_{k+1} \le \mathbf{g}_{\mathbf{x}} \\ & \mathbf{G}_{\mathbf{u}}\bar{\mathbf{u}}_k \le \mathbf{g}_{\mathbf{u}} \\ & \forall \quad k = 0,\ldots,N-1 \end{aligned} \tag{2}$$

However, to solve this optimal control problem efficiently via an explicit formulation, we first simplify the notation by defining $\boldsymbol{x} = [\bar{\mathbf{x}}_1^{\mathbf{T}}, ..., \bar{\mathbf{x}}_N^{\mathbf{T}}]^{\mathbf{T}}$, $\boldsymbol{r} = [\bar{\mathbf{r}}_1^{\mathbf{T}}, ..., \bar{\mathbf{r}}_N^{\mathbf{T}}]^{\mathbf{T}}$, $\boldsymbol{u} = [\bar{\mathbf{u}}_0^{\mathbf{T}}, ..., \bar{\mathbf{u}}_{N-1}^{\mathbf{T}}]^{\mathbf{T}}$, $\boldsymbol{u}_{\hat{\mathbf{y}}} = [\bar{\mathbf{u}}_{\hat{\mathbf{y}}}^{\mathbf{T}}, ..., \bar{\mathbf{u}}_{\hat{\mathbf{y}}}^{\mathbf{T}}]^{\mathbf{T}}$

$$\boldsymbol{\mathcal{B}} = \begin{bmatrix} \mathbf{B} & 0 & \cdots & 0 \\ \mathbf{AB} & \mathbf{B} & \cdots & 0 \\ \vdots & \vdots & \ddots & \\ \mathbf{A}^{N-1}\mathbf{B} & \mathbf{A}^{N-2}\mathbf{B} & \ldots & \mathbf{B} \end{bmatrix}$$

$\boldsymbol{\mathcal{Q}} = \operatorname{diag}(\mathbf{Q},...,\mathbf{Q})$, $\boldsymbol{\mathcal{R}} = \operatorname{diag}(\mathbf{R},...,\mathbf{R})$, $\boldsymbol{\mathcal{G}}_{\mathrm{x}} = \operatorname{diag}(\mathbf{G}_{\mathbf{x}},...,\mathbf{G}_{\mathbf{x}})$, $\boldsymbol{\mathcal{G}}_{\mathrm{u}} = \operatorname{diag}(\mathbf{G}_{\mathbf{u}},...,\mathbf{G}_{\mathbf{u}})$, $g_{\mathbf{x}} = [\mathbf{g}_{\mathbf{x}}^{\mathbf{T}},...,\mathbf{g}_{\mathbf{x}}^{\mathbf{T}}]$.

Then, we can represent the QP entirely in terms of $\boldsymbol{u}$ by substituting the dynamics and dropping constant terms in the cost function,

$$\begin{aligned} \underset{\boldsymbol{u}}{\operatorname{argmin}} & \frac{1}{2}\boldsymbol{u}^{\mathbf{T}}\boldsymbol{\mathcal{H}}\boldsymbol{u} + \boldsymbol{h}^{\mathbf{T}}\boldsymbol{u} = 0 \\ & s.t. \quad \boldsymbol{\Gamma}\boldsymbol{u} \le \boldsymbol{\gamma} \end{aligned} \tag{3}$$

where $\boldsymbol{\mathcal{H}} = \boldsymbol{\mathcal{B}}^{\mathrm{T}}\boldsymbol{\mathcal{Q}}\boldsymbol{\mathcal{B}} + \boldsymbol{\mathcal{R}}$, $\boldsymbol{h} = -\boldsymbol{\mathcal{B}}^{\mathrm{T}}\boldsymbol{\mathcal{Q}}\boldsymbol{r} - \boldsymbol{\mathcal{R}}\boldsymbol{u}_{\hat{\mathbf{y}}}$,

$$\boldsymbol{\Gamma} = \begin{bmatrix} \boldsymbol{\mathcal{G}}_{\mathrm{x}}\boldsymbol{\mathcal{B}} \\ \boldsymbol{\mathcal{G}}_{\mathrm{u}} \end{bmatrix}, \quad \text{and} \quad \boldsymbol{\gamma} = \begin{bmatrix} g_{\mathbf{x}} \\ g_{\mathbf{u}} \end{bmatrix} \tag{4}$$

For calculating the first two Karush-Kuhn-Tucker (KKT) conditions of optimality (stationary and complementary slackness) for QP, we define $\boldsymbol{\lambda}$ as the vector of Lagrange multipliers and $\boldsymbol{\Lambda} = \operatorname{diag}(\boldsymbol{\lambda})$. The conditions can then be written as

$$\begin{aligned} \mathcal{H}\boldsymbol{u} + \boldsymbol{h} + \Gamma^{\mathbf{T}}\boldsymbol{\lambda} = 0 \\ \boldsymbol{\Lambda}(\boldsymbol{\Gamma}\boldsymbol{u} - \boldsymbol{\gamma}) = 0 \end{aligned} \tag{5}$$

By considering only active constraints (i.e., with $\boldsymbol{\lambda} > 0$) for a given solution, $\boldsymbol{u}$ and $\boldsymbol{\lambda}$ can be reconstructed by solving a linear system derived from (6) and (7), where subscript a indicates rows corresponding to the active constraints.

$$\begin{bmatrix} \boldsymbol{\mathcal{H}} & \boldsymbol{\Gamma}_a^{\mathbf{T}} \\ \boldsymbol{\Gamma}_a & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{\lambda}_a \end{bmatrix} = \begin{bmatrix} -\boldsymbol{h} \\ \boldsymbol{\gamma}_a \end{bmatrix}$$

Assuming the active constraints are linearly independent, the resulting control law, $\boldsymbol{u}$, is linear in the predicted state error, $\boldsymbol{r}$, and parameterized by the system dynamics

$$\boldsymbol{u} = \boldsymbol{\mathcal{E}}_5\boldsymbol{r} - \left(-\boldsymbol{\mathcal{E}}_4\boldsymbol{\mathcal{R}}\boldsymbol{u}_{\hat{\mathbf{y}}} + \boldsymbol{\mathcal{E}}_3 \begin{bmatrix} g_{\mathbf{x}}^{+} \\ -g_{\mathbf{x}}^{-} \\ g_{\mathbf{u}}^{+} \\ -g_{\mathbf{u}}^{-} \end{bmatrix}_a \right) \tag{6}$$

where $\boldsymbol{\mathcal{E}}_1 = \boldsymbol{\Gamma}_a\boldsymbol{\mathcal{H}}^{-1}$, $\boldsymbol{\mathcal{E}}_2 = -(\boldsymbol{\mathcal{E}}_1\boldsymbol{\Gamma}_a^{\mathrm{T}})^{-1}$, $\boldsymbol{\mathcal{E}}_3 = \boldsymbol{\mathcal{E}}_1^{\mathrm{T}}\boldsymbol{\mathcal{E}}_2$, $\boldsymbol{\mathcal{E}}_4 = \boldsymbol{\mathcal{H}}^{-1} + \boldsymbol{\mathcal{E}}_3\boldsymbol{\mathcal{E}}_1$, and $\boldsymbol{\mathcal{E}}_5 = \boldsymbol{\mathcal{E}}_4\boldsymbol{\mathcal{B}}^{\mathrm{T}}\boldsymbol{\mathcal{Q}}$. Active Lagrange multipliers are computed in a similar way and are required to check the KKT conditions to determine controller optimality.

$$\boldsymbol{\lambda} = -\boldsymbol{\mathcal{E}}_6\boldsymbol{r} + \left(-\boldsymbol{\mathcal{E}}_3^{\mathrm{T}}\boldsymbol{\mathcal{R}}\boldsymbol{u}_{\hat{\mathbf{y}}} + \boldsymbol{\mathcal{E}}_2 \begin{bmatrix} g_{\mathbf{x}}^{+} \\ -g_{\mathbf{x}}^{-} \\ g_{\mathbf{u}}^{+} \\ -g_{\mathbf{u}}^{-} \end{bmatrix}_a \right) \tag{7}$$

where $\mathcal{E}_6 = \mathcal{E}_3^{\mathrm{T}}\mathcal{B}^{\mathrm{T}}\mathcal{Q}$. To evaluate the KKT conditions, we just need the set of active constraints. Thus, instead of storing the controller gains and Lagrange multipliers, only the set of active constraints are stored. This is the prime reason that explicit EPC can be implemented in a memory-efficient way on severely computationally constrained platforms. The controller and KKT matrices can then be recalculated using (6)-(7).

Using this formulation, we develop the two parts of the explicit EPC algorithm described in Algs. 1 and 2. As described in Alg. 1, the controller database is initialized to be empty at the start of the simulation. We introduce a transition frequency matrix, $\mathbf{\Phi}$ which records the number of transitions between each pair of controllers added to $\mathcal{M}$ [1]. A reference sequence (line 3) is obtained from a randomly generated trajectory in simulation. $\mathbf{\Phi}$ is incremented as the system-dynamics are forward simulated using EPC with the current database of controllers (line 10). If an appropriate controller is not found in the database, EPC solves the QP (3), adding the resulting controller in the database (line 15) and consequently increasing the size of $\mathbf{\Phi}$. The final control input at the end of each iteration, $\mathbf{u}_k$ is applied to a dynamics model to get the updated state, $\mathbf{x}_{k+1}$ after Δt seconds (line 19). After the simulation ends and the database is complete, an order-1 Markov chain is defined which represents transitions between controllers based on empirical transition probabilities computed from $\mathbf{\Phi}$. Transitions of controllers are sorted from each state of Markov chain according to the probability of occurrence. Markov simplification reduces the number of transitions of each state to limit the query time per control iteration by retaining the highest probability transitions out of each state [1]. This yields a partial ordering, $\mathbf{\Omega}$ (line 22), which informs the online query process taking place at the embedded level described in Sect. II-B.

Algorithm 1 Generating synthetic experience from regular EPC in simulation

1: $\mathcal{M} \leftarrow \varnothing$, $\mathbf{\Phi} \leftarrow 0$, $k \leftarrow 0$, $\mathbf{x}_k \leftarrow \mathbf{x}_0$, $j \leftarrow 0$
2: **while** simulation is running **do**
3: $\quad \boldsymbol{r}_k \leftarrow$ reference sequence from random trajectory
4: $\quad \mathbf{A}_k$, $\mathbf{B}_k \leftarrow$ current linearized dynamics model
5: $\quad$ **for** each element $\mathbf{m}_i \in \mathcal{M}$ **do**
6: $\quad\quad$ Compute u, λ_a via (6), (7)
7: $\quad\quad$ **if** $\boldsymbol{x_k}$, $\boldsymbol{r_k}$ satisfy the KKT criteria (5) **then**
8: $\quad\quad\quad$ `importance`$_i \leftarrow$ current time, sort $\mathcal{M}$
9: $\quad\quad\quad$ `controller_found` $\leftarrow$ true
10: $\quad\quad\quad$ $\mathbf{\Phi}_{ij} \leftarrow \mathbf{\Phi}_{ij} + 1$, $j \leftarrow i$
11: $\quad\quad\quad$ **break**
12: $\quad\quad$ **end if**
13: $\quad$ **end for**
14: $\quad$ **if** `controller_found` is false **then**
15: $\quad\quad$ Solve QP (3) to generate new controller $\boldsymbol{u}$
16: $\quad\quad$ Add new element $\mathbf{m}_{new}$ to $\mathcal{M}$
17: $\quad\quad$ $i \leftarrow |\mathcal{M}|$, $\mathbf{\Phi}_{ij} \leftarrow 1$
18: $\quad$ **end if**
19: $\quad \mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \int_0^{\Delta t} f(\mathbf{x}, \mathbf{u}) d\tau$
20: $\quad k \leftarrow k + 1$
21: **end while**
22: Compute ordering $\mathbf{\Omega}$ based on transition frequencies $\mathbf{\Phi}$

Algorithm 2 Online database query for running explicit EPC on-board an embedded platform

1: **Inputs:** $\mathcal{M}$ and $\mathbf{\Omega}$ from Alg. 1, Current state ($\boldsymbol{x}$, $\boldsymbol{r}$)
2: **for** each element $\mathbf{m}_i \in \mathcal{M}$ ordered by $\mathbf{\Omega}$ **do**
3: $\quad$ Compute $\boldsymbol{u}$, $\boldsymbol{\lambda}_a$ via (6) (7)
4: $\quad$ **if** $\boldsymbol{x}$, $\boldsymbol{r}$ satisfy KKT criteria (5) **then**
5: $\quad\quad$ `solution_found` $\leftarrow$ true
6: $\quad\quad$ Apply affine control law (6) from $\mathbf{m}_i$
7: $\quad$ **end if**
8: **end for**
9: **if** `solution_found` is false **then**
10: $\quad$ Apply PD Control as a backup controller
11: **end if**

B. Running explicit EPC on an embedded platform

Algorithm 1 yields an explicit controller database with simplified ordering of controllers [1]. As presented in Alg. 2, $\mathcal{M}$ is queried in each control iteration to identify the appropriate controller. As each control iteration corresponds to a state transition in the Markov chain, we identify the next transition by iterating through $\mathcal{M}$ according to the order specified by $\mathbf{\Omega}$. This type of ordering based on Markov chain simplification helps to reduce the number of controllers that are evaluated online. As evaluating KKT conditions is more expensive than a simple lookup table query, this approach improves the controller efficiency thereby enabling explicit EPC on computationally constrained platform. If no controller in the database is suitable (line 9), we apply PD controller as a backup (line 10).

III. Results and Discussion

To assess the controller performance, we conduct tests on a severely computationally constrained platform. Specifically, we use a Pixhawk microcontroller with a 32-bit STM32F427 Cortex M4 core with FPU clocked at 168MHz, 256 KB of RAM, 2MB Flash Memory and a 32bit STM32F103 fail-safe co-processor [2]. To evaluate the performance of the explicit EPC algorithm on this severely computationally constrained platform, we seek to demonstrate the following results: **R1:** stable control performance, **R2:** real-time computation of control commands on-board, improved computational efficiency via the Markov chain based ordering and simplification, and **R3:** comparable explicit EPC controller performance running on the embedded micro-controller (without a host computer) as with the results from running it on the host computer.

Using the controller database generated from simulation via Alg. 1, we conduct a series of experimental trials on a quadrotor MAV equipped with the Pixhawk (Fig. 2) using Alg. 2.

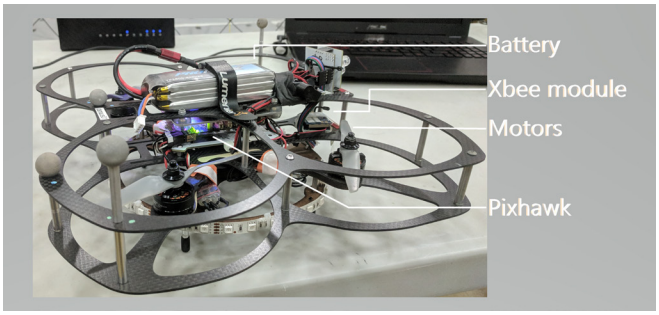


Fig. 2: Platform equipped with computationally constrained embedded autopilot (Pixhawk) used for experimental validation of explicit EPC.

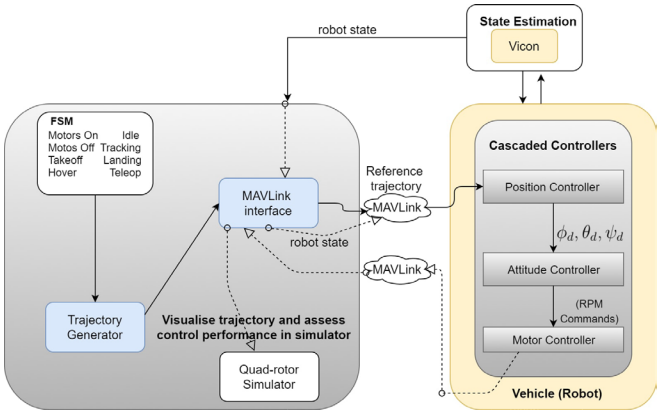


Fig. 3: Block diagram of the planning and control architecture that enables simulation and experimental evaluation of the ideas presented in this paper.

A. System Overview

A general motion manager architecture (described in Appendix **A** of [13]) is used for tracking trajectories. Robot state and reference states (trajectory references) are sent to the onboard microcontroller using the Mavlink [14] protocol via Xbee radios. The overall system setup is shown in Fig. 3.

B. Experimental Evaluation

The database for controlling the translational dynamics (outer loop [15]) of the MAV is computed offline and applied online at a rate of 100 Hz as described in Sect. II. Following a standard cascaded control formulation [15], we apply a PD control law for the inner loop to track the outer loop output.

Using Alg. 1, we generate the controller database with a horizon length of $N = 2$ control iterations and with constraints on x-velocity, y-velocity, and the three control inputs. This yields a database with only 19 entries, as opposed to $3^{10} \approx 59049$ total possible controllers. To reduce redundant online matrix operations, we precompute and cache some of the EPC matrices and hence the total memory usage for explicit EPC is 5.3 kB.

Figure 5 shows the response to step inputs along the x, y, and z axes, thereby demonstrating stable control performance **(R1)**. To evaluate online control performance, the vehicle is commanded to track a vertical circle trajectory, as shown in Fig. 4.

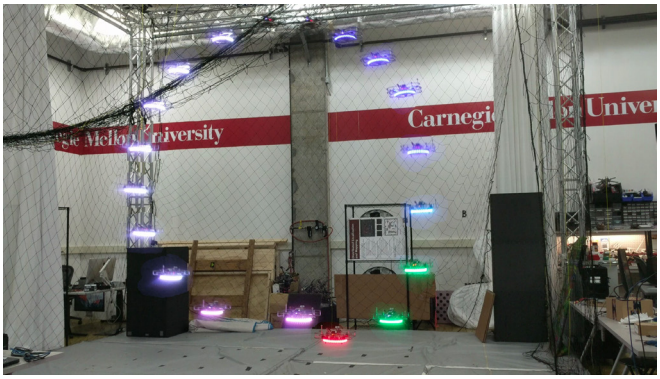


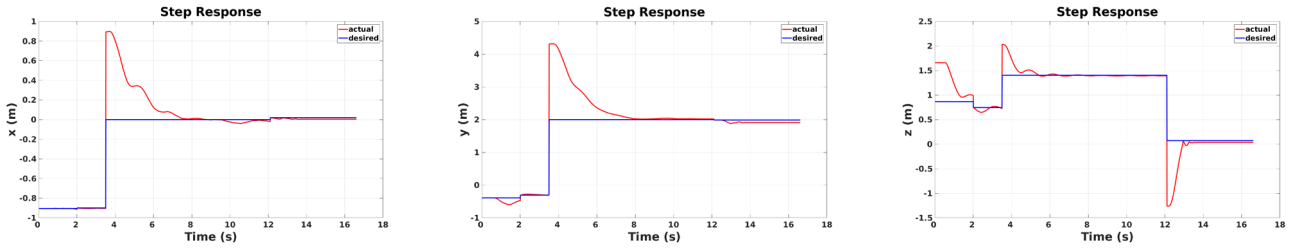
Fig. 4: Vertical circle trajectory used to assess trajectory tracking using explicit EPC (visualized as an overlay of images). LEDs equipped beneath the platform changes color depending on the x,y, and z coordinates of the platform in the arena.

The EPC update loop, including the database query step, requires approximately 1 ms to complete and thus reliably achieves the desired update rate to track the inner attitude control loop. Consequently, explicit EPC is sufficiently computationally efficient to be tractable on the severely computationally constrained autopilot **(R2)**. This real-time control performance is derived from Markov chain simplification. Each controller in the database has on average only three successors to evaluate and at most five successors **(R2)**. For larger controller database cases, without simplification, the evaluation of the database could cause the controller to block the update loop (outer loop in our case), thereby destabilizing the vehicle.

For **(R3)**, we compare the controller performance running on the embedded micro-controller (without a host computer) with that of the robot having a host computer. The same vertical circle trajectory tracking scenario is used as discussed above. Figure 6 show the tracking performance in x, y, and z, respectively. and demonstrate that the explicit formulation gives comparable performance to the online version **(R3)**. For fair comparison, similar testing scenarios are created for both the cases. As we used horizon length, $N = 2$, for explicit EPC formulation, same horizon length is used for regular EPC tests using host-computer. The small bias in Fig. 6 for regular EPC version may be due to communication delay between the host PC and the Pixhawk. Figure 6 highlight that the explicit EPC running onboard the embedded platform (without the host-computer) gives comparable performance to the regular EPC running using a host-computer **(R3)**.

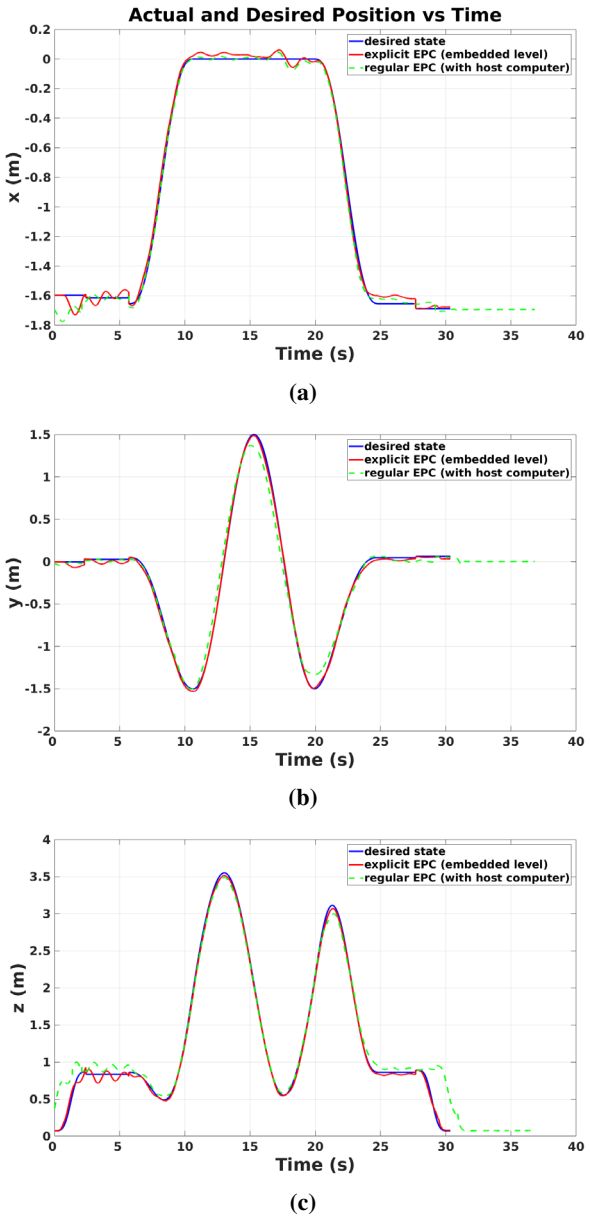
IV. Conclusion and Future Work

In this work, we present the experimental evaluation of the explicit Experience-driven Predictive Control algorithm on a severely computationally constrained platform. Explicit EPC is implemented as a two part algorithm where the first part constructs an experience database in simulation while the second part applies it online on the vehicle. We compare explicit EPC performance on a vehicle without a host computer with a vehicle running regular EPC with a host computer onboard. We demonstrate the viability



(a) **(b)** **(c)**

Fig. 5: Position control response to commanded step inputs in x, y, and z ((**a**), (**b**), and (**c**), respectively).



(a)

(b)

(c)

Fig. 6: Explicit EPC yielding comparable trajectory tracking performance in x, y, and z ((**a**), (**b**), and (**c**), respectively) with respect to regular EPC.

of the explicit EPC algorithm for real-time operation on computationally constrained platforms.

In the future, we intend to extend the present formulation to Robust Experience-driven Predictive Control using tightened constraints [16].

References

[1] V. R. Desaraju and N. Michael, "Efficient prioritization in explicit adaptive nmpc through reachable-space search," in *AIAA Guidance, Navigation, and Control Conference*, January 2018, [to appear].

[2] "PX4FMU Autopilot / Flight Management Unit." [Online]. Available: https://pixhawk.org/modules/px4fmu

[3] M. Turpin, N. Michael, and V. Kumar, "Trajectory design and control for aggressive formation flight with quadrotors," *Autonomous Robots*, vol. 33, no. 1-2, pp. 143–156, 2012.

[4] L. Grne and J. Pannek, *Nonlinear model predictive control: theory and algorithms*. Springer Publishing Company, Incorporated, 2013.

[5] V. R. Desaraju and N. Michael, "Leveraging experience for computationally efficient adaptive nonlinear model predictive control," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2017.

[6] R. Brockers, M. Hummenberger, S. Weiss, and L. Matthies, "Towards autonomous navigation of miniature uav," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 631–637.

[7] M. Dong, B. M. Chen, G. Cai, and K. Peng, "Development of a real-time onboard and ground station software system for a uav helicopter⋆," *Journal of Aerospace Computing, Information, and Communication*, vol. 4, 2007.

[8] F. Kendoul, "Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems," *Journal of Field Robotics*, vol. 29, no. 2, pp. 315–378, 2012.

[9] P. Campoy, J. F. Correa, I. Mondragón, C. Martínez, M. Olivares, L. Mejías, and J. Artieda, "Computer vision onboard uavs for civilian tasks," in *Unmanned Aircraft Systems*. Springer, 2008, pp. 105–135.

[10] S. E. de Nagy Köves Hrabar, "Vision-based 3d navigation for an autonomous helicopter," Ph.D. dissertation, University of Southern California Los Angeles, CA, 2006.

[11] "Pixhawk Autopilot system specifications," https://pixhawk.org/modules/pixhawk#specifications, accessed: 2017-10-18.

[12] H. W. Kuhn, "Nonlinear programming: a historical view," in *Traces and emergence of nonlinear programming*. Springer, 2014, pp. 393–414.

[13] V. R. Desaraju, "Safe, efficient, and robust predictive control of constrained nonlinear systems," Ph.D. dissertation, Carnegie Mellon University, April 2017.

[14] L. Meier, J. Camacho, B. Godbolt, J. Goppert, L. Heng, M. Lizarraga, *et al.*, "Mavlink: Micro air vehicle communication protocol," *Online]. Tillgänglig: http://qgroundcontrol. org/mavlink/start.[Hämtad 2014-05-22]*, 2013.

[15] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The grasp multiple micro-uav testbed," *IEEE Robotics & Automation Magazine*, vol. 17, no. 3, pp. 56–65, 2010.

[16] V. R. Desaraju, A. E. Spitzer, and N. Michael, "Experience-driven predictive control with robust constraint satisfaction under time-varying state uncertainty," in *Robotics: Science and Systems Conference (RSS)*, July 2017.

Experimental Evaluation of Motion Primitive-based Teleoperation of Mobile Robots

Kshitij Goel, Xuning Yang, and Nathan Michael

***Abstract*— This work experimentally evaluates the time efficiency of teleoperation of mobile robots using motion primitives. We evaluate forward arc motion primitives-based teleoperation with experimental studies on aerial and ground hardware in real world scenarios and present field results of the proposed teleoperation strategy via a user study and two-factor analysis of variance (ANOVA). These studies suggest that users prefer the motion primitives-based strategy for smoothness in operation and increased time efficiency during teleoperation tasks, over direct feedforward velocity-based teleoperation strategy.**

I. INTRODUCTION

In most strategies for teleoperation of mobile robots, user issued inputs via an external input device are directly executed leading to the possibility of infeasible motions due to possible communication delays [2]. This direct command architecture requires the human operator to be precise in issuing commands to the robot, hence making the teleoperation task tedious, fatiguing, and error prone [3]. In order to decrease workload on the user, we pursue teleoperation strategies that reduce user effort and and error and increase time efficiency while completing the teleoperation task. Hence, assistive teleoperation strategies are developed to assist the user while performing a teleoperation task and are often evaluated using statistical analysis methods via user studies. These user studies are evaluated based on various user efficiency metrics. Javdani et al. [4] show assistance to a human operator while teleoperating a high degree of freedom humanoid robot and evaluate the proposed strategy through a user study using time efficiency and input frequency metrics. Carlson et al. [5] show collaboration between human and wheelchair via adaptive assistance by developing an intelligent controller that compensates for invalid inputs, and evaluate the proposed teleoperation strategy via user study based on two metrics: execution time and deviation from safety. Similarly, You et al. [6] evaluated teleoperation strategy to operate a robot arm based on 2D inputs, based on execution time taken by the user. These examples suggest that time efficiency in completing a teleoperation task is indeed an important metric to evaluate teleoperation strategies.

Yang et al. [1] propose a motion primitives-based teleoperation strategy that enables dynamically feasible and safe trajectories amenable to naive user control of a mobile robot. This work evaluates this teleoperation approach in terms of time efficiency through experimental results on hardware for an aerial and a ground robot platform via a user study, analyzed using a two-factor Analysis of Variance (ANOVA) technique. For the aerial robot, motion primitives-based teleoperation and direct velocity-based teleoperation are compared. For the ground system, we find that the motion primitives-based teleoperation is equivalent to the direct teleoperation strategy, both theoretically and experimentally. User preference towards using these teleoperation strategies for both aerial and ground robots is also analyzed based on Likert-Scale post-trial responses.

Motion primitives-based teleoperation [1] is presented in brief in Sect. II followed by the overview of systems used in our user study in Sect. III. Experiment design and results are detailed in Sect. IV followed by conclusion in Sect. V.

The authors are affiliated with the Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh PA 15213, USA.

II. MOTION PRIMITIVES-BASED TELEOPERATION

In this section, motion primitives-based teleoperation [1] is briefly described. The action space for the robot is represented as a set of motion primitives called a Motion Primitive Library (MPL), which is constructed by densely discretizing the action space and generating motions associated with that action. An action is defined to be a set of input parameters that define a specific motion associated with the input parameters. For inputs of dimension $q \times 1$, if an action is denoted as $\mathbf{a} = a_1, ..., a_q$ then a corresponding motion primitive can be generated, represented as $\gamma(\mathbf{a})$. Any set of actions can generate a motion primitive library (MPL) $\Gamma = \gamma_i(\mathbf{a}_i)$, with size N. Furthermore, the set of MPL is called a motion primitive library collection, denoted by Γ_j, $j = 1, ..., M$. The robot is being controlled via a continuous input device, i.e. a joystick or gamepad and the action space of the robot is given by a set of values, obtained by densely discretizing the input values in each input dimension. Hence, a MPL will contain $q \times N$ motion primitives, which is treated as an indexed set of motion primitives parameterized by the action space.

It is assumed that motion primitive library can be generated for arbitrary mobile robots given their dynamical constraints as a function of the system-specific inputs. Smoothness and continuity issues arise in generating motion primitive types. It is assumed that sufficient information regarding actuator limitations and dynamic feasibility constraints are satisfied in design of the motion primitives.

A. Forward Arc Motion Primitives

Forward-arc motion primitives are sampled by propagating the dynamics of a unicycle model with a constant linear and angular velocity for a specified time, T. For an unicycle

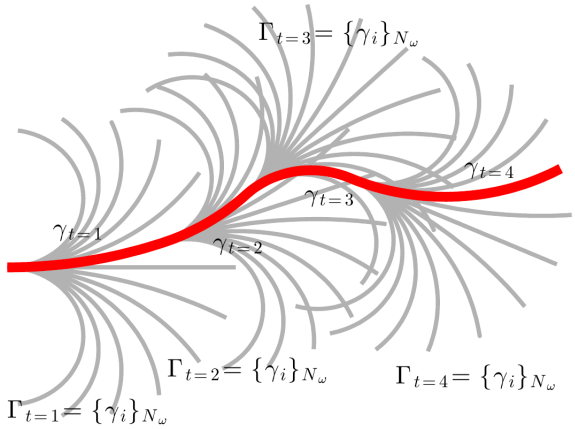


Fig. 1. Visualization of the forward arc motion primitives-based teleoperation [1]. The bold red line is the final path chosen by the user through the joystick, formed by combining the chosen motion primitives (one out of all the grey primitives at the four time instances).

model:

$$\dot{x} = v_s cos(\theta)$$

$$\dot{y} = v_s sin(\theta)$$

$$\dot{\theta} = \omega$$

where v_s is forward velocity and ω is angular velocity. We now show that for a ground robot, this primitive-based representation of motion is theoretically equivalent to kinematics of a differential drive robot. Euler integration to the above equations gives:

$$X_{t+T} = \begin{bmatrix} x_{t+T} \\ y_{t+T} \\ \theta_{t+T} \end{bmatrix} = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} + \begin{bmatrix} v_s cos(\theta) T \\ v_s sin(\theta) T \\ \omega T \end{bmatrix} \quad (1)$$

Now, $\theta = \theta_t + \omega T$. Substituting in (1):

$$\begin{bmatrix} x_{t+T} \\ y_{t+T} \\ \theta_{t+T} \end{bmatrix} = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} + \begin{bmatrix} v_s cos(\theta_t + \omega T) T \\ v_s sin(\theta_t + \omega T) T \\ \omega T \end{bmatrix} \quad (2)$$

For small T :

$$cos(\theta_t + \omega T) \simeq \frac{sin(\theta_t + \omega T) - sin(\theta_t)}{\omega T}$$

$$sin(\theta_t + \omega T) \simeq \frac{cos(\theta_t) - cos(\theta_t + \omega T)}{\omega T}$$

Substituting in (2) :

$$\begin{bmatrix} x_{t+T} \\ y_{t+T} \\ \theta_{t+T} \end{bmatrix} = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} + \begin{bmatrix} \frac{v_s}{\omega}(sin(\theta_t + \omega T) - sin(\theta_t)) \\ \frac{v_s}{\omega}(cos(\theta_t) - cos(\theta_t + \omega T)) \\ \omega T \end{bmatrix} \quad (3)$$

Equation (3) suggests that for ground robots, motion primitives-based teleoperation approach yields equivalent kinematics to a direct velocity-based teleoperation [8]. These

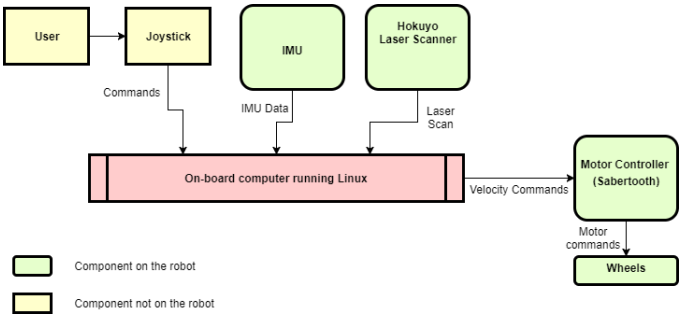


Fig. 2. Hardware overview of the ground robot test system. The user issues commands via the joystick to an on board computer, the computer takes feedback from an IMU and Hokuyo Laser Scanner and produces velocity commands for the motor controller. The motor controller finally converts velocity commands to RPM.

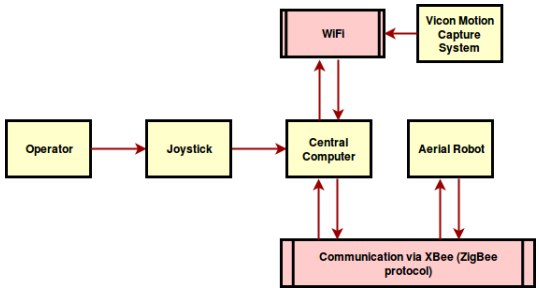


Fig. 3. Hardware overview of the aerial robot test system. The user issues commands via joystick to a central computer that receives odometry information via a motion capture system. Computation of the control action is done on the central computer itself and is sent over ZigBee to the aerial robot.

motion primitives can be extended for aerial vehicles by incorporating linear dynamics for the vertical velocity. Hence, for a quadrotor aerial vehicle:

$$\begin{bmatrix} x_{t+T} \\ y_{t+T} \\ z_{t+T} \\ \theta_{t+T} \end{bmatrix} = \begin{bmatrix} x_t \\ y_t \\ z_t \\ \theta_t \end{bmatrix} + \begin{bmatrix} \frac{v_s}{\omega}(sin(\theta_t + \omega T) - sin(\theta_t)) \\ \frac{v_s}{\omega}(cos(\theta_t) - cos(\theta_t + \omega T)) \\ v_z T \\ \omega T \end{bmatrix} \quad (4)$$

For ground vehicles, the heading of the vehicle is fixed to the yaw of the vehicle by nature. Although aerial platforms such as quadrotors can independently control heading from yaw, we maintain the use of a unicycle model by ensuring that the heading is equivalent to the yaw of the vehicle, as humans naturally optimize for curved trajectories in robot control with heading aligned to the zero yaw angle [9].

III. System Design

A. Hardware

The ground robot teleoperation system, shown schematically in Fig. 2, uses a laser scanner and an inertial measurement unit (IMU) for state estimation using an Unscented Kalman Filter (UKF) [10]. The user sends teleoperation commands through a Logitech gamepad F-100 Joystick and the commands are processed on board the vehicle to enable motor control. The aerial test platform, shown schematically in Fig. 3, uses a motion capture system is used through

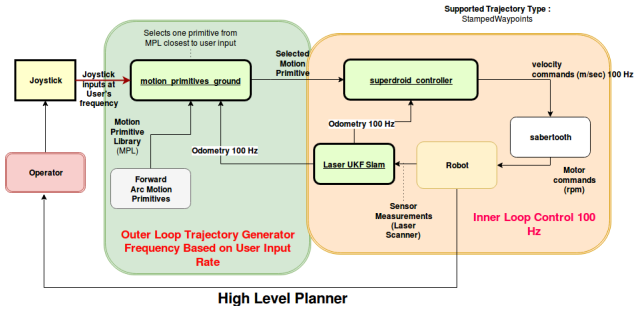


Fig. 4. Software system for the ground robot test platform consists of an inner loop PD controller running at 100 Hz, and an outer loop trajectory generator at the frequency of the user input.

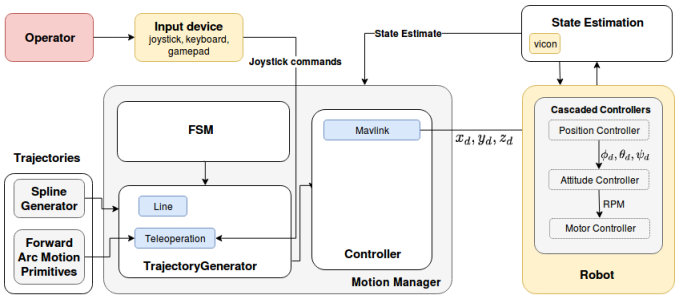


Fig. 5. Software system for the aerial robot test platform, consists of a finite state machine and a trajectory generator that is used to generate dynamically feasible reference commands for the cascaded PD controller running onboard the vehicle.

Vicon cameras for state estimation with the teleoperation commands being processed on a central computer.

B. Software

Software overview for both systems are shown in Figs. 4-5, where arrows correspond to the data flow using custom messages and services in Robot Operating System (ROS). Software runs on-board for the ground system and off-board for the aerial system.

IV. EXPERIMENTAL RESULTS & DISCUSSION

Evaluation of the efficiency of forward arc motion primitives-based teleoperation is carried out via a user study designed to access the following hypotheses.

- **H1**: Participants are able to perform the teleoperation task significantly faster using motion primitive based teleoperation as compared to other teleoperation strategies, when operating an aerial robot.
- **H2**: Participants are able to perform the teleoperation task in similar times using motion primitives based teleoperation as compared to direct velocity-based teleoperation strategies, when operating a ground robot.
- **H3**: Participants prefer motion primitive-based teleoperation above other teleoperation techniques, when operating an aerial robot.

Two-factor Analysis of Variance (ANOVA) is used to infer about first two hypotheses, considering above statements as null hypotheses, while we directly analyze averaged Likert scale responses from the users for **H3**.

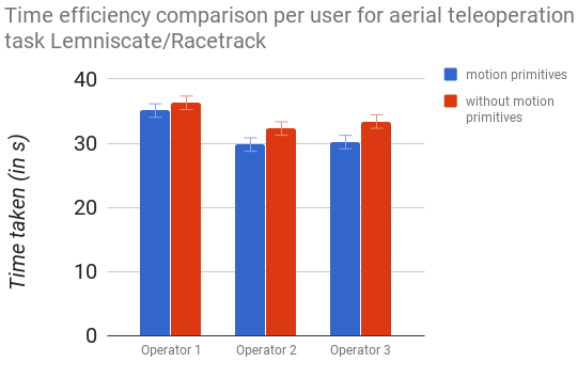


Fig. 6. Task completion times for aerial teleoperation task for three operators flying a lemniscate trajectory. Users are able to perform the teleoperation task faster using motion primitive-based teleoperation than using direct velocity-based teleoperation (without motion primitives).

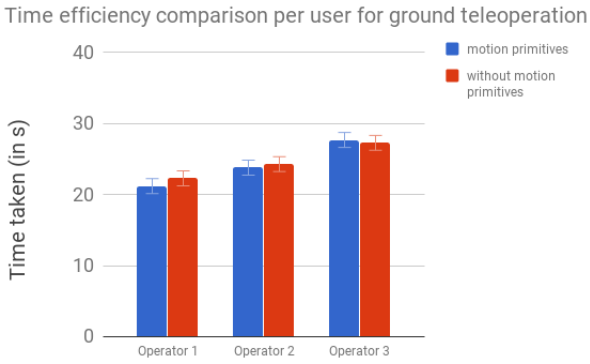


Fig. 7. Task completion times for ground robot teleoperation comparing motion primitives-based teleoperation and simple feed forward velocity teleoperation. Users achieve similar time efficiency using either of the teleoperation methods due to equivalence of motion primitive-based and direct velocity-based strategies in the ground teleoperation case.

A. Experiment Design

For aerial robot, users are asked to complete one lemniscate trajectory for two teleoperation methods: direct velocity-based and motion primitives-based teleoperation. At the start of the aerial teleoperation study, users are told they would be using two different teleoperation strategies, anonymized as ‘method 1’ and ‘method 2’. For the ground robot, simple feedforward velocity-based and motion primitives-based teleoperation are used and anonymized as ‘method 1’ and ‘method 2’. Users are not provided any additional information about the methods and conduct 5 trials each with these different teleoperation strategy. In order to familiarize with the joystick controls, users teleoperate the vehicle in simulation for both aerial and ground robots, prior to the recorded trials.

We recruit 3 participants (for both ground and aerial robots), all with experience in robotics, but none with prior exposure to both robots. To counterbalance individual differences of users, we choose a within-subjects design, where each user uses all the teleoperation methods. Post-trials,

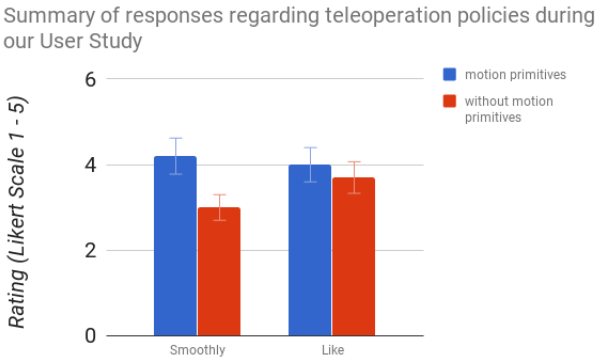


Fig. 8. Average responses from users for some specific questions from the study. Users prefer the motion primitives-based teleoperation over direct teleoperation for smoothness in completing the teleoperation task. The study also suggests that the users are more inclined to use the motion primitive-based teleoperation strategy for future teleoperation tasks.

users are given a short survey for each teleoperation strategy, consisting of the following 1-5 Likert Scale questions:

- Does the robot follow the commands *smoothly*? 1 corresponding to strongly disagree and 5 to strongly agree.
- If you are to teleoperate an aerial or a ground robot, would you *like* to use the teleoperation strategy? 1 corresponding to strongly disagree and 5 to strongly agree.

B. Aerial Robot Teleoperation Results

Figure 6 shows the average task completion times for the operators following a lemniscate trajectory. Trial times and total control input are assessed using a two-factor repeated measures ANOVA, first factor being the time it took to complete the desired teleoperation task. We found that using motion primitives method resulted in users accomplishing tasks more quickly, supporting **H1** ($F = 68.572, p-value = 0.0008$, $F_{crit} = 6.944$).

Figure 8 shows average responses from the users corresponding to the set of questions listed in Sect. IV-A, suggests users prefer motion primitive based teleoperation for smooth operation as well as likeness to use in a future teleoperation task. This proves the hypothesis **H3**.

C. Ground Robot Teleoperation Results

H2 holds true in terms of time efficiency ($F = 6.77, p = 0.12, F_{crit} = 19$). This is due to the fact that motion primitive-based teleoperation is equivalent in theory to a simple direct velocity-based teleoperation hence the observed similar performance times. This suggests motion primitive-based teleoperation is equivalent to direct velocity-based teleoperation experimentally.

V. CONCLUSION

In this work, motion primitives based-teleoperation evaluation for aerial and ground robots suggests that users are able to perform the teleoperation task significantly faster using motion primitives based teleoperation as compared to direct velocity-teleoperation for quadrotors, which is typically employed in industrial quadrotors. Moreover, we have shown that motion primitives-based teleoperation is theoretically and experimentally equivalent to direct velocity-based teleoperation in the case of ground robots. Moreover, it is also shown that users prefer motion primitive-based teleoperation strategy for smoothness in operation of teleoperation tasks.

REFERENCES

[1] Xuning Yang, Koushil Sreenath, Nathan Michael, "A Framework for Efficient Teleoperation via Online Adaptation", International Conference on Robotics and Automation (ICRA), 2017
[2] T. Fong and C. Thorpe, "Vehicle Teleoperation Interfaces", Autonomous Robots 11, 9-18, 2001
[3] T. Sheridan, "Telerobotics, Automation, and Human Supervisory Control", MIT Press, Cambridge, MA
[4] S. Javdani, S. S. Srinivasa, and J. A. Bagnell, "Shared Autonomy via Hindsight Optimization," in Proceedings of Robotics: Science and Systems, Rome, Italy, July 2015
[5] T. Carlson and Y. Demiris, "Human-Wheelchair Collaboration Through Prediction of Intention and Adaptive Assistance," in Proceedings of the IEEE International Conference on Robotics and Automation, Pasadena, CA, 2008, pp. 3926–3931.
[6] E. You and K. Hauser, "Assisted Teleoperation Strategies for Aggressively Controlling a Robot Arm with 2D Input", in Proceedings of Robotics: Science and Systems, 2011
[7] J. W. Crandall, and M. A. Goodrich, "Characterizing Efficiency of Human Robot Interaction: A Case Study of Shared-Control Teleoperation", in Proceedings of International Conference on Intelligent Robots and Systems, 2002.
[8] S. Lavalle, "Planning Algorithms", Cambridge University Press, Cambridge, U.K., 2006.
[9] N. Delson and H. West, "Robot Programming by Human Demonstration: The Use of Human Inconsistency in Improving 3D Robot Trajectories", in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Sep. 1994, pp. 1248-1255
[10] E. Nelson, "Environment Model Adaptation for Autonomous Exploration", The Robotics Institute, Carnegie Mellon University, 2015.

Designed for Flexibility: A New Highway Simulation Environment for Inverse Reinforcement Learning Algorithms

Rachel Gologorsky*

***Abstract*— Developing autonomous highway driving policies is a challenging yet necessary task for the development of truly autonomous vehicles. A currently popular approach is to use Inverse Reinforcement Learning (IRL) algorithms, which learn driving policies from human demonstration. Developing and evaluating IRL algorithms requires the use of customizable and highly controllable highway simulators. While highway simulators exist for general research and gaming purposes, they are unsuitable for IRL research because of their complexity and inflexibility. The lack of adaptable highway environments has caused researchers to individually implement custom "toy" highway environments specifically tailored to their research. However, this continued reimplementation is inefficient because the toy environments share many features and have similar requirements. In this report, I document the details of my highly customizable and open-source toy highway environment and thus contribute an understandable, extensible, and recyclable environment usable as the starting point many other research investigations.**

I. Introduction

A. Self-Driving Cars: The Road to IRL

Artificial intelligence in general, and its application to self-driving cars in particular, has enormous potential to transform society for the better. According to the National Highway Traffic Association and the Center for Disease Control, human error accounts for 94% of car accidents, which are the leading cause of death of those aged 15-29 years and cost the U.S. economy $870 billion in aggregate a year [1, 2, 3]. Developing self-driving car policies has thus become the focus of much current research effort.

In the past, rule-based and behavior cloning algorithms were popular approaches to developing self-driving cars [4, 5, 6]. However, the driving policies resulting from these algorithms are inherently limited because they are agnostic with respect to *why* humans take certain actions. As a result, they are unable to detect and adjust to unexpected situations.

The present paradigm resolves this issue by assuming that humans intuitively optimize a cost function of the state features (such as balancing the desire to increase speed with the desire to keep a safe distance from other vehicles). Given a human cost function, an autonomous vehicle can plan to minimize the cost of its actions, and thus mimic the internal human decision-making process instead of the outward driving behavior. Thus, much current research now focuses on developing inverse reinforcement learning (IRL) algorithms to recover increasingly sophisticated cost functions from observed human driving behavior, especially in the highway driving context.[1] Indeed, the highway environment has become a standard benchmark for evaluating algorithm performance in the inverse reinforcement learning field [7, 8, 9, 10, 11, 12, 13].

B. The Need for a Flexible Highway Simulator

Applying IRL algorithms requires the use of highway simulators to collect the human demonstration data and to simulate the driving policies resulting from the reconstructed cost function. While highway simulators exist for general research and gaming purposes, they are generally complex to use and difficult to tailor to specific research scenarios. Yet, in order to evaluate an IRL algorithm's ability to capture specific behaviors, researchers developing IRL algorithms require the ability to control the initial highway setting, specify the subsequent behavior of other cars on the road, and extract custom state features from the highway simulator. The lack of customizable highway simulators has caused IRL researchers to individually construct "toy" highway environments specifically tailored to their research projects. An example of the proliferation of researcher-specific toy environments is pictured in Figure 1 below.

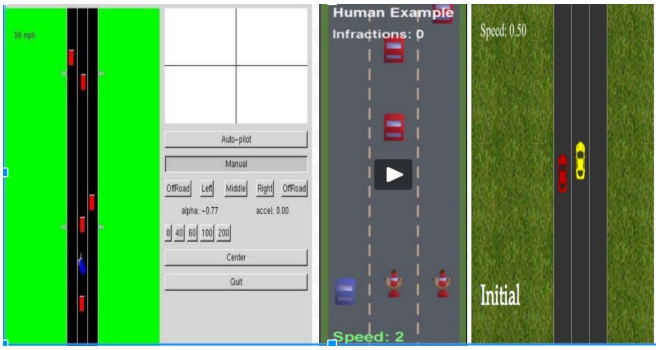


Fig. 1. Many Specific Environments. Left: Abbeel and Ng (2004), Center: Levine et al. (2011), Right: Sadigh et al. (2016) [2, 3, 4]

As a whole, these toy highway environments are similar in both their external characteristics (2d, bird's eye perspective) and in their requisite internal functionality that enables highway demonstration input and IRL algorithm application. Yet, despite the commonalities between the toy highway environments and the shared need, no well-documented and

* Rachel Gologorsky is a 2017 Robotics Institute Summer Scholar, and is a joint concentrator in Computer Science and Mathematics at Harvard. This work was supported by a National Science Foundation Undergraduate Research Grant.

[1]The field is called inverse reinforcement learning because the objective is to derive a cost function from the observed behavior - thus inverting the standard reinforcement learning problem of deriving an optimal behavior from a given cost function.

extensible toy highway environment usable as a common starting point for IRL research currently exists. Though this dearth of toy highway environments is understandable – the true research significance is ultimately in the algorithm developed, not the tool it was tested on – starting anew each time is inefficient. Thus, I built a toy highway environment that is flexible and adaptable to a variety of research investigations. By documenting my code in this technical report and making it open-source, I aim to tackle this inefficiency and contribute a toy highway environment usable as a convenient springboard for other inverse reinforcement learning research projects.

In Section II, I review the the specifics of the highway simulators currently available. Then, in Section III, I define the desired characteristics of a universal highway environment, and in Section IV, review the design and technical details of the highway environment. I conclude and outline possible enhancements to the simulation environment in Section V. Section VI contains an Appendix with more technical details.

II. Available Highway Simulators

A. Commercial Simulators

There are several commercial research driving simulators on the market, such as Carnetsoft's research driving simulator, the University of Iowa's National Advanced Simulator research products, and FAAC Inc.'s Research Simulator for Automotive Vehicles [14, 15, 16]. While such commercial products are quite realistic, the high fidelity afforded by these commercial research driving simulators comes at the expense of transparency and affordability. The cheapest option, Carnetsoft's software-only driving simulator package, costs approximately $3000. Ultimately, there is an inescapable trade off between realism, cost, and complexity, and in the case of IRL algorithm research, the cost and complexity of the commercial products outweighs their benefit of realism.

B. Open-Source Simulators

Another potential option is Udacity's recently released open-source self-driving car simulator [17]. This car simulator was built for students in Udacity's self-driving car nanodegree program. As a result, the car simulator is specifically tailored to the homework assignment of implementing deep learning to predict a steering wheel angle from still frames. The pre-built scenarios feature a single car driving around a track; thus, it would take significant effort to expand the almost undocumented code to include a highway scenario with multiple-car interactions. Moreover, familiarity with Unity's 3d game engine is generally restricted to the game developer community. The academic research community, the target audience in this case, is probably unfamiliar with Unity and so is likely to experience difficulty customizing and extending this highway environment.

C. Gaming Simulators and Open-AI Gym

There are several well-known gaming simulators, such as Grand Theft Auto V and TORCS ([18]), that have successfully been used as testbeds for reinforcement learning (RL) algorithms [19, 20, 21]. Similarly, Open AI Gym, a popular toolkit that interfaces between RL algorithms and a wide collection of environments (such as online games), also includes several driving environments. While these simulators are well-suited to the RL task of developing a policy that minimizes the cost of the agent car's actions (these simulators allow easy control over agent car and access to the cost of the agent car's actions via the game score or external indicators), they are unsuitable for IRL algorithm development because of the limited state information available and the inability to manipulate the behavior of the other cars in the simulator. However, the success of the Open AI Gym project is a source of inspiration, and I aim to achieve in the IRL field what Open AI Gym has accomplished in the RL field by providing a flexible, high-level interface to streamline the process of algorithm development and evaluation.

D. Simulators within Academia

Finally, Levine et al. and Sadigh et al. [8, 9] have made the code behind their research papers publicly available, including their toy highway implementations. However, their motive was to enable others to replicate their results, and so their highway environment code is undocumented. Moreover, their highway environments were specifically tailored to their research and thus are difficult to adapt to different situations such as different numbers of lanes or different car behaviors.

III. Requirements of a Universal Highway Simulation Environment

At a minimum, learning driving policies from human demonstration via IRL requires the ability to specify the starting scenario, input human driving demonstrations, and apply IRL algorithms to that driving data. A universal highway environment must not only be able to accomplish these tasks, but also must be flexible enough to allow modification to the underlying settings and methods by which those tasks are accomplished. A brief overview of the specific research scenarios of Abbeel and Ng, Levine et al., and Sadigh et al. (whose toy highway environments are pictured in Figure 1) will motivate the discussion of the flexibility required of a universal highway environment [7, 8, 9].

A. Specific Use Cases

(1) In their groundbreaking research paper introducing IRL in the driving context, Abbeel and Ng [7] tested the ability of their pilot IRL algorithm to recover a cost function that would mimic specific driving styles. Since this was the first foray into developing driving policies via an underlying cost function, they focused on a very simplified scenario. The IRL algorithm extracted a linear cost function from pre-determined discrete features (the distance to the car in front, to the nearest car length) in a circumscribed discrete state and action space. Specifically, the behavior of other cars on the road was fixed, as was the speed of the agent car (human-controlled vehicle). The five action inputs controlled the agent car's direction. From the picture of their toy highway environment, it appears that the human

actions were inputted via clicking (mouse/keyboard setup).

(2) To relax the constraint of predefining the pertinent state features, Levine et al. [8] developed an IRL algorithm that automatically determines relevant conjunctions of features from a large set of possible components. Specifically, they tested the ability of their algorithm to recover a driving policy that depended on the conjunction of current speed, lane, and "police car" proximity. Thus, in their toy highway environment, they defined more features and varied the appearance of the other cars on the road (though the cars' underlying behavior remained the same). They also displayed the user's current speed during the demonstration so that the user could adjust their speed to be consistent with the desired driving policy during driving demonstrations.

(3) To incorporate human-robot interaction into the IRL framework, Sadigh et al. [9] developed an IRL algorithm that chooses actions in order to distinguish between "distracted" and "attentive" drivers. In their work, a generic human driver cost function was obtained via demonstration with a wheel/pedal setup, and then the cost function weights were tweaked to model "distracted" and "attentive" human drivers. As a result, the toy highway environment had continuous features, a continuous state and action space, and the ability to specify different internal car behaviors (with the same outward appearance).

B. Flexibility Motivated by the Use Cases

From the above three research situations, we see that a universal highway environment should have:

- Customizable action inputs and input mechanism (e.g. to incorporate different discrete/continuous action spaces and keyboard/mouse and wheel/pedal setups).
- Customizable external car appearance (e.g. to incorporate "police cars" and other vehicle types).
- Customizable internal car behavior (e.g. to incorporate "attentive"/"distracted" drivers).
- Customizable features (e.g. to incorporate desired policy specifics).
- Customizable feature drawing options (e.g. to show features such as speed to the user or not).

Further, I have tried to anticipate future research directions by adding the following requirements.

- Customizable starting highway/traffic conditions - e.g. to model lane-changing dynamics in various traffic conditions, on small roads and multi-lane highways.
- Customizable physics - so that researchers can customize the "feel" of the driving simulator dynamics and adjust the complexity to be appropriate to the research scenario - in some cases, a simple point-mass model suffices (as in [9]); in others, a complex physics engine may be necessary (as in [18]). By default, I implemented the Kinematic Bicycle Model, which simulates realistic car dynamics quite well as long as the lateral acceleration is not too high [22].
- Customizable color scheme and drawing options - e.g. to accommodate red/green color blindness, to create dashed or solid lane markings (important to lane-change and lane-merge scenarios), or to simulate a potential difference between day and night driving policies. See Figure 2 for an illustration of this color scheme difference.
- Infinite length track - to enable flexibility in the length of the human driving demonstrations.
- Customizable recording options - to grant researchers flexibility in the content, data format, and visualization of their recorded data.

IV. System Design

I implemented my code in Python because it is a particularly flexible and readable coding language. Visually, I chose a 2d, bird's eye perspective of a vertically-oriented highway environment because it allows the user to intuitively orient themselves, infer all the information necessary to make rational decisions, and predict the future state while retaining the internal simplicity necessary for easy understandability and customizability. While 3d perspectives are more realistic, the benefit is more than offset by the additional complexity of programming 3d environments. Overall, I have consistently prioritized clarity in my code in order to align the environment with its purpose of being easily understandable and modifiable.

To make the highway environment flexible, I implemented it in a functional and highly modular style. The functional style allows concrete implementations to be changed without effecting the underlying logic, and the modular separation into classes allows for distracting implementation specifics to be encapsulated in different files (and easily referenced when necessary). To avoid duplication of effort, I have implemented common functionality such as collision detection, future state prediction, recording capabilities, and access to current state, we well as common features such as distance to road/lane boundaries, gap sizes to neighboring cars, and access to neighboring car state information.

Figure 2 highlights some of the physical flexibility of this environment, including the ability to specify different numbers of cars, highway lanes, colors, and types of cars. The red dot in the center indicates the human-controlled vehicle (agent car). The tree on the right is a reference point used as a visualization aid. The feature representation (displayed on the left, omitted on the right) is customizable as well.

V. System Architecture

A. File Summary

Abstract Car class - contains variables and functions common to all cars. Stores its highway as an instance variable. The AgentCar, MergingCar, OtherCar classes inherent common features and functionality from AbstractCar but customize the move function in order to implement specific behaviors. The AbstractCarMixin class contains helper functions specific to the Car class,

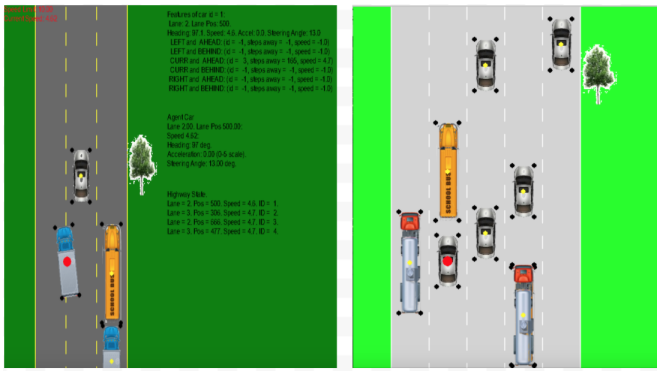


Fig. 2. One Universal Environment. Left: agent is a truck driving on a small 3 lane highway, night color scheme. Right: agent is a sedan driving on busy 5 lane highway during the daytime.

such as init functions, is_collision, and dist_to_lane_boundary.

Highway class - creates the highway. Stores a list a cars on the highway. Responsible for the majority of drawing functions, encapsulated in the HighwayMixin class. Adjustable parameters include the highway length, lane width, and the number of lanes; the lane and road color can also be easily customized at initialization. The drawing functions (implemented in a functional style) are encapsulated by the Highway Mixin class. These drawing functions are also used by the Playback class to play back a prior recorded human demonstration.

Simulator class - interfaces between a car's state and action inputs. By default, action inputs are continuous steering angle and acceleration/brake input. However, can be easily customized and discretized by changing the functions called to process the raw input. Note: Steering angle is measured in radians from the y-axis (clockwise from the y-axis is positive).

Playback class - contains key recording/playback functionality: writing to file, reading stored data from file, and visualizing the recorded demonstration. By default, the data - a list of the general highway parameters and the time series of all the cars on the highway - is pickled. The content and storage mechanism is easily modifiable by changing the write_recorded_data and get_recorded_data functions. The data is generated in the game_events file by the RECORD_HIGHWAY_EVENT, which fires with the frequency set in the constants file.

Dynamics.py - contains the physical dynamics model, by default Kinematic Bicycle Dynamics. Easily customizable by changing the next_step function in the dynamics file. Other models (such as the simpler point-mass or the more complex ones incorporating friction) can be easily substituted by changing the next_step function in the dynamics file. Note: the Kinematic Bicycle Dynamics Mode simulates real car dynamics surprisingly well as long as the lateral acceleration is not too high [22].

Constants - contains global constants. Collected into a single file so that they can be easily found and modified. These constants include the Pygame window's height and width dimensions, event timer frequencies (e.g. how often should the draw_all function be called or the highway recorded), and sundry others constants (e.g. common colors).

Helpers - contains general helper functions, which can be classified into 3 main categories: those that detect game events, deal with geometry, or are just generally useful. Note about geometry: the coordinate system used throughout is that of Pygame, i.e. the y-axis is orientated to be increasing from top to bottom.

Scenarios - defines some start scenarios. Currently, I utilize scenarios with a predetermined the number of highway lanes, the types of cars, and their speeds, headings, and positions. Over time, this file can be expanded into a library of different deterministic and randomly generated testing scenarios.

Game Events - handles game events such as Move_Car_Event and Record_Highway_Event timer events (frequency defined in the constants file), action inputs (passes them to the Simulator class), and other user-driven events such as pause, quit, and restart (defined in the helpers file).

Main file - contains two functions, play_game, and play_back. Play_game starts the game with the given input start scenario, and calling play_back with a specific file name plays back the demonstration that file captured. In both functions, a while loop iterates through the events generated by the game/playback until the game/playback is done. For readability, the events are identified and handled in the game events file.

B. Key Mechanism

A highway object is required for a car's initialization (other parameters are optional). At initialization, the car stores the highway as an instance variable and adds itself to its highway instance's car_list field. Thus, the highway has a list of all car objects on it, and every car has, through its highway instance, access to all the other car objects on the road. Since Python implements "Call by Reference," each car's highway instance always reflects the current highway object and the highway object's car list always reflects the current car objects.

There are several benefits to unrestricted information sharing. First, it allows all conceivable state features to be implemented because there is complete access to the highway and all the other cars' states. In addition, it simplifies the implementation of car behaviors that depend on neighboring car state and common functions such as checking

for collisions (simply scan the highway's car object list), projecting the highway state into the future, and playing back demonstrations. Since most toy research scenarios include < 20 cars, iterating through the highway's car list is in fact an efficient and simplifying design decision.

C. Visualization

In order to allow for the illusion of an infinite length track (so that human demonstration time is unlimited), all the objects on the highway move down rather than the agent car moving up. Hence the agent car's objective position never changes, though it advances relative to the other cars on the road. To aid in visualization, I added a simple Tree object by the highway to serve as a reference point for human drivers. In dashed lanes, I introduce a time parity parameter, "odd_timestep," in order to alternate between revealing "new" dashed lane separators and translating the "old" ones down.

VI. Conclusion/Future Directions

In this paper, I address the inefficiency of reimplementing similar toy highway environments and share the details of a versatile and customizable pilot highway simulation environment. This pilot environment, however, is just a start, and there are many possible avenues for further development and refinement. Ideas for future expansion include:

- An expanded car behavior library so that researchers could "plug and play" desired car behaviors into their research (e.g. to incorporate the effect of aggressive/timid cars on the driving policy, or to investigate human behavior in response to cars pre-programmed to merge into the agent car's lane in different ways).
- An Obstacle class to incorporate common car-obstacle interactions and model lane merge and highway exit behavior.
- Introduction of increasingly sophisticated models of future state prediction that incorporate human-robot interaction and reactions.
- A Graphical User Interface to ease scenario creation.
- Introduction of more precise polygonal (instead of rectangular) car bounding boxes into the car geometry.
- A library for different Physical dynamics models.
- An extended environment scope to intersections and other situations of research significance.

In the future, I hope that researchers will recognize the utility of a common starting point, and develop this highway environment into a high-level springboard for other research investigations.

VII. Appendix

A. Car Classes Derived from ***AbstractCar***

- The **OtherCar** class implements behavior of a general car on the road, which I programmed to to be: staying in its lane, adjusting its speed to keep its preferred distance to the car ahead, returning to its preferred speed when possible, and never crashing into the car in front.
- The **MergingCar** class roughly implements a car programmed to merge into the left lane; the merging car uses deterministic heuristics to guide its attempted lane merge.
- The **AgentCar** class implements behavior specific to the human-controlled vehicle - its behavior reflects that of the user controlling it. It is the only vehicle I allow to crash into other cars.

The AbstractCar move method is itself general, and has optional parameters that govern whether a car is allowed to collide with another car, whether it can go off-road, and whether its keeps a set distance from the car in front of it. Thus, the Abstract Car class provides a flexible starting point from which cars with different behaviors can be derived.

As the name suggests, the Abstract Car class is an abstract class meant to define common car behavior while leaving the specifics to its derived classes. However, the AbstractCar class has an init method meant to be called by the derived classes in order to initialize some common variables. The only required argument to this init function is the car's highway object, though optional arguments include a simulator object to interface with action inputs, and the desired initial x,y coordinates of the car's center, and its lane, lane position, speed, and role.

B. Explication of the Differences between Similar Car Instance Variables

x,y vs. lane, lane position.: The x and y position are closely related to, though not exactly equivalent to, the lane and lane position. The lane is a discrete number corresponding to the lane number (0-indexed), while the x-value is a float that stores the precise x-coordinate of the pixel representing the car center. The lane position is a float that stores the precise y-coordinate indicating how far the car center is from the beginning of the highway track (i.e. increases from bottom to top); in Pygame, however, the y-coordinate of the pixel position increases from the top down. Thus, the pixel y-value = highway length - the lane position. In the AbstractClass Mixin (which contains helper functions specific to cars), there are functions for converting x and y positions to lane and lane positions, and vice versa (when converting a lane number to a x value, the x value is calculated as the center point of the lane).

Role vs ID.: At each initialization, each car is given a unique "ID" number generated by a counter variable. This ID provides an efficient way to perform equality tests between car objects. In contrast, the role variable (a modifiable string parameter set at initialization) is used to identify special vehicles (such as the agent car) in functions.

C. Default Initialization of Optional Car Instance Variables

When the optional arguments are not provided, the vehicle settings (x, y, lane, lane position, and speed) are instantiated with functions such as init_place, which sets the lane and lane position, init_pixel_pos, which sets the x and y position based on the previously set lane and lane position, and init_speed, which initiates the speed. These init functions

grant flexibility over the parameters' default initialization. For example, I implemented the init_speed function to initialize a car's speed via a Gaussian distribution whose mean depends on the car's lane in order to incorporate the highway driving convention that the left lane tends to have faster vehicles.

There are some common variables that I initialize to have default values that could be analogously be customized or passed in as initialization parameters. For example, I initialize the car to have a vertical heading - $pi/2$ radians, i.e. 90 degrees counterclockwise from the x-axis, and set l_f and l_r (distance from the center of mass to the front and rear axles, respectively) with default value.

D. Selected Functions Highlighting Car-Highway Information Sharing

Get_closest_cars is a function in the highway class that, when given a lane and lane position, returns a list with information about the car state ahead and behind in the given and neighboring lanes. It relies on **get_closest_car**, which finds the closest car object on the highway in the specific direction to find the desired car information. I use this highway function in my car class to get the state features of the gap size and speed of the cars ahead and behind the agent car in its lane and in the neighboring lanes.

Crash_in_n_steps is a highway function that uses the dynamics model to simulate whether continuing the current trajectories would result in a car crash within the next n steps. This function is used by my MergingCar to inform its decision on how far to turn the steering wheel and how much to accelerate during a lane merge. A further refinement would incorporate reactions rather than simply projecting all current car inputs into the future.

VIII. Acknowledgements

I am indebted to Dr. John Dolan and Dr. Katharina Muelling for their guidance and mentorship this summer, and to the indefatigable Rachel Burcin for her hard work, boundless optimism, and commitment to making the RISS program truly stellar. I am honored and humbled to be part of such a distinguished and diverse cohort of young scientists, and I am very grateful to the National Science Foundation for funding my research. Finally, I would like to thank Professor Joseph Blitzstein and Professor Harry Lewis of Harvard University. Without their incredible support, this research experience would not have been possible.

References

[1] S. Singh, "Critical reasons for crashes investigated in the national motor vehicle crash causation survey," Tech. Rep., 2015.

[2] L. Blincoe, A. Seay, E. Zaloshnja, T. Miller, E. Romano, S. Luchter, R. Spicer, *et al.*, "The economic impact of motor vehicle crashes, 2000," Washington, DC, National Highway Traffic Safety Administration, Tech. Rep., 2002.

[3] C. for Disease Control and Prevention. (2016). Road traffic injuries and deaths a global problem, [Online]. Available: `https://www.cdc.gov/features/globalroadsafety/index.html`.

[4] D. Pomerleau, "Knowledge-based training of artificial neural networks for autonomous robot driving," in *Robot Learning*, Jan. 1993.

[5] A. Kesting, M. Treiber, and D. Helbing, "General lane-changing model mobil for car-following models," *Transportation Research Record: Journal of the Transportation Research Board*, no. 1999, pp. 86–94, 2007.

[6] C. Urmson, J. Anhalt, H. Bae, J. A. (Bagnell, C. R. Baker, R. E. Bittner, T. Brown, M. N. Clark, M. Darms, D. Demitrish, J. M. Dolan, D. Duggins, D. Ferguson, T. Galatali, C. M. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. Howard, S. Kolski, M. Likhachev, B. Litkouhi, A. Kelly, M. McNaughton, N. Miller, J. Nickolaou, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, V. Sadekar, B. Salesky, Y.-W. Seo, S. Singh, J. M. Snider, J. C. Struble, A. (Stentz, M. Taylor, W. (L. Whittaker, Z. Wolkowicki, W. Zhang, and J. Ziglar, "Autonomous driving in urban environments: Boss and the urban challenge," vol. 25, no. 8, pp. 425–466, Jun. 2008.

[7] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*, ACM, 2004, p. 1.

[8] S. Levine, Z. Popovic, and V. Koltun, "Feature construction for inverse reinforcement learning," in *Advances in Neural Information Processing Systems*, 2010, pp. 1342–1350.

[9] D. Sadigh, S. S. Sastry, S. A. Seshia, and A. Dragan, "Information gathering actions over human internal state," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, IEEE, 2016, pp. 66–73.

[10] E. Klein, M. Geist, B. Piot, and O. Pietquin, "Inverse reinforcement learning through structured classification," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., Curran Associates, Inc., 2012, pp. 1007–1015.

[11] J. Choi and K.-e. Kim, "Nonparametric bayesian inverse reinforcement learning for multiple reward functions," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., Curran Associates, Inc., 2012, pp. 305–313.

[12] U. Syed, M. Bowling, and R. E. Schapire, "Apprenticeship learning using linear programming," in *Proceedings of the 25th international conference on Machine learning*, ACM, 2008, pp. 1032–1039.

[13] E. Klein, B. Piot, M. Geist, and O. Pietquin, "A cascaded supervised learning approach to inverse reinforcement learning," in *Joint European Conference*

on Machine Learning and Knowledge Discovery in Databases, Springer, 2013, pp. 1–16.

[14] *Carnetsoft research driving simulator*. [Online]. Available: http://cs-driving-simulator.com/research-driving-simulator/.

[15] *University of iowa - the national advanced driving simulator - simulators - the nads minisim*. [Online]. Available: https://www.nads-sc.uiowa.edu/sim_minisim.php.

[16] *Faac incorporated research deskop driving simulator*. [Online]. Available: https://www.faac.com/research-simulators/research-desktop-driving-simulator/.

[17] *Udacity's open source self-driving car simulator*. [Online]. Available: https://www.udacity.com/self-driving-car.

[18] B. Wymann, C. Dimitrakakis, A. Sumner, E. Espié, C. Guionneau, and R. Coulom, *TORCS, the open racing car simulator, v1.3.5*, http://www.torcs.org, 2013.

[19] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[20] D. Loiacono, A. Prete, P. L. Lanzi, and L. Cardamone, "Learning to overtake in torcs using simple reinforcement learning," in *Evolutionary Computation (CEC), 2010 IEEE Congress on*, IEEE, 2010, pp. 1–8.

[21] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International Conference on Machine Learning*, 2016, pp. 1928–1937.

[22] P. Polack, F. Altché, B. D'Andréa-Novel, and A. De La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?" In *IEEE Intelligent Vehicles Symposium (IV)*, 2017.

Mining Discriminative Intervals for Affective State Estimation

Deepthi Hegde[1], Stanislav Panev[2] and Dong Huang[2]

***Abstract*— The recent advancement of machine learning and wearable sensors has propelled estimation of human affective status from streaming physiological signals. Previous machine learning solutions have focused on off-line analysis of datasets containing records of emotions stimulated in participants through short videos. Most of them perform predictions considering all the video segments to be equally important. In this paper, we emphasize the need to detect specific segments of the videos that trigger emotion as a first step towards better classification. We implement a multiple-instance learning approach to weight these segments accordingly during training of SVM. We use MAHNOB-HCI, a publicly available dataset for all our analyses. Our approach provides promising results.**

***keywords*- Affective computing, wearable sensors, physiological signals, multiple instance learning, leave-one-subject-out, leave-one-video-out.**

I. Introduction

Emotion is the backbone of every activity carried out by humans. It influences the way we think, behave, communicate and react to the surroundings around us. Emotion understanding spans a wide range of applications and has recently received a lot of attention in the field of Human Computer Interaction (HCI). It is an important step towards intelligent interaction and in enhancing user experience. For instance, televisions being able to empathize with and respond to users' according to their mental state. In online learning, engagement of the student can be improved by understanding the emotional status of the student and modifying the teaching methods accordingly [1]. Emotion estimation can also aid those with communication difficulties, like the autistic, to interact with their surroundings by natural means. Mental disorders like Schizophrenia and depression can be attended to and alleviated by early detection of symptoms.

One of the major drawbacks of computers today is that they lack an understanding of emotion. Coupling emotion with cognition and perception could greatly impact the performance of computers and bring us closer to building machines that are human-like. However, it is a hard task as emotion is an abstract quantity and is person-dependent and situational. Moreover, the lack of objective ways to evaluate emotions makes it difficult to annotate and obtain ground truth.

Emotion is an outcome of neurological processes in the brain and is manifested externally as facial expressions, voice modulations and muscle movements. The most popular modality for emotion recognition has been facial expressions. However, facial data raise two major issues. Firstly, continuous monitoring of emotions is not feasible and is invasive. Secondly, facial expressions are often deceiving. They may not represent the true emotional states of the participant, as masking emotions is rather easy. Thus, we need a more reliable and easy-to-capture way of obtaining data. Using wearable sensors helps eliminate these issues. However, this modality has received less attention.

Wearable sensors measure the internal changes that occur in the body when users are subjected to emotional stimuli. These sensors measure signals like heart rate, skin conductance, pupil dilation, skin temperature, etc. that are controlled by the sympathetic nervous system, which is part of the autonomous nervous system of the human body. These signals cannot be tampered with intentionally by the user and hence cannot be faked. There has been a widespread acceptance of wearable devices in day-to-day life lately. Most wearable devices monitor general health of the user by measuring parameters like heart rate, sleep patterns, level of exercise, etc. Because emotion is one of the primary contributors to overall health, estimating and monitoring it helps provide better awareness to the user.

In recent years, there has been rapid development of emotion recognition using physiological sensors and several databases have been collected. However, only a few of them are publicly available. The Database for Emotion Analysis using Physiological Signals (DEAP) [2] is one such publicly available dataset that contains physiological signals for 32 participants and face videos for 22 of those participants watching music videos. The signals that were recorded comprised electromyography (EMG), electroocologram (EOG), blood volume pulse (BVP) using plethysmograph (PPG), skin temperature and Galvanic skin response (GSR). The drawback of this dataset is that each video is 1 minute long and 40 videos are shown to the participants back to back with a small 5 minutes break in between. This time frame seems to be too less for significant emotion fluctuations. MAHNOB-HCI (Multimodal Database for Affect Recognition and Implicit Tagging) [3] is yet another publicly available multimodal database that is similar to DEAP containing both physiological signals and face video. MAHNOB-HCI makes use of movie videos to elicit emotions and consecutive videos with stimuli are well separated by a neutral video screening in between. We choose MAHNOB-HCI for our experiments. We describe the protocol of the dataset in Section II.

The previous approaches [3], [4] on the MAHNOB-HCI dataset have focused on statistical methods to find correlation between the features and labels, feature selection and feature

[1]Deepthi Hegde is a Robotics Institute Summer Scholar, Carnegie Mellon University, Pittsburgh, PA 15213, USA deepthimhegde@gmail.com

[2]S. Panev and D. Huang are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA {spanev, dghuang}@andew.cmu.edu

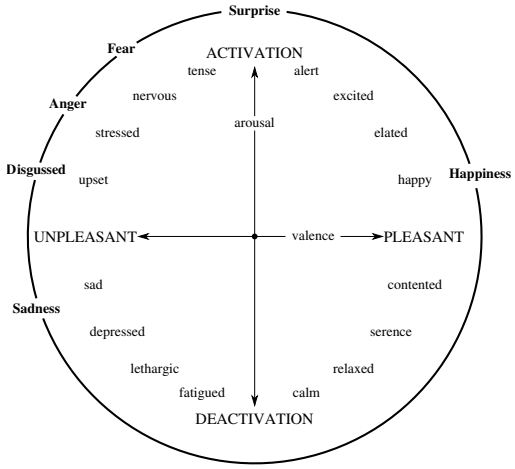


Fig. 1. Russell's Valence Arousal Scale [5]

and decision fusion techniques to improve the accuracy of affect estimation. There are however certain issues that need to be addressed before diving into decision making. Often, there are one or more segments of the video that induce the emotion indicated by the participant. In this paper, we emphasize the need to select those specific segments of the videos that trigger emotion as a first step towards good classification. We implement a Multiple-Instance Learning approach to find these segments prior to classification.

In Section II, we summarize the contents and the protocol used in MAHNOB-HCI. Section III describes all the features used for our experiments. In Section IV, we explain Multiple Instance Learning and our formulation to detect and weight the important segments of the video. In Section V, we present the results obtained by implementing a multiple instance learning approach.

II. Dataset description

MAHNOB-HCI is a multi-modal database with records of peripheral physiological signals from 27 participants who are shown 20 emotional movie clips. After each video containing stimulus is screened, a neutral clip is played hoping to bring the emotional status of the participant to neutral before the start of the next stimulus. The signals recorded in the database include electrocardiogram (ECG), electroencephalogram (EEG), GSR, respiration pattern, skin temperature and eye gaze.

The database uses Russells valence-arousal scale [5], which is the most commonly used measure of emotional states in affective computing. On this scale, each emotional state can be represented on a 2D orthogonal plane with arousal and valence as the horizontal and vertical axes respectively. Arousal can range from inactive (e.g., uninterested, bored) to active (e.g., alert, excited), whereas valence ranges from unpleasant (e.g., sad, stressed) to pleasant (e.g., happy, elated) as shown in Fig. 1

Self-assessment was recorded by asking participants to report the felt arousal and valence on nine point scales. Self assessment Manikins (SAM) were presented (see Fig. 3) and

TABLE I
MAHNOB-HCI Database Summary

Participants and Modalities	
Num. of participants	27, 11 male, 16 female
Recorded Signals	32-channel EEG (256 Hz), Peripheral physiological signals (256 Hz), Face and body video using 6 cameras (60 f/s), Eye gaze (60 Hz), Audio (44.1 kHz)
Emotional responses to videos	
Num. of videos	20
Selection method	Subset of online annotated videos
Self report	Emotional keyword, arousal, valence, dominance, predictability
Rating values	Discrete scale of 1-9

explained in order to smoothly carry out self-assessment. The distribution of labels across different values of valence and arousal are as show in Fig. 4

The summary of the contents of the dataset is presented in Table. I

Fig. 4 shows the distribution of self-assessment scores in the valence-arousal plane. It is evident that there are fewer instances of extreme valence and arousal scores, indicating that it is hard to stimulate these emotions in people.

III. Data preprocessing and features extraction

Physiological responses were initially recorded at a sampling rate of 1024 Hz and finally downsampled to 256 Hz. The eye gaze data was sampled at 60 Hz. We upsampled eye gaze data to 256 Hz in order to have matching window sizes for all the signals for feature extraction. Moreover, GSR signals were infiltrated by the frequency of the power supply. We removed those frequencies with the help of a Notch Filter.

In [3], the authors list the potential set of features for each of the sensor signals. We extract most of the features mentioned along with other features as listed in Table III. We extracted 576 ECG features, 216 EEG features, 39 GSR features, 4 features relating to skin temperature, 14 respiratory features and 35 features for eye gaze.

IV. Methodology

A. Weakly-supervised learning for faster and more accurate labeling

One of the biggest problems with self-reported affective status is the lack of accuracy and localization. Even with the addition of a more effective segment labeling system (as described in Section IV-B), it is still uncertain whether the self-reported labeling will be effective enough for the use of machine learning on the affective status. In order to remedy this accuracy and localization difficulty we use a Generalized Multi-Segment SVM (GMS-SVM) to improve

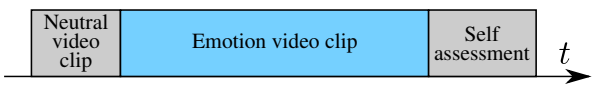


Fig. 2. Experimental protocol used in MAHNOB-HCI dataset.

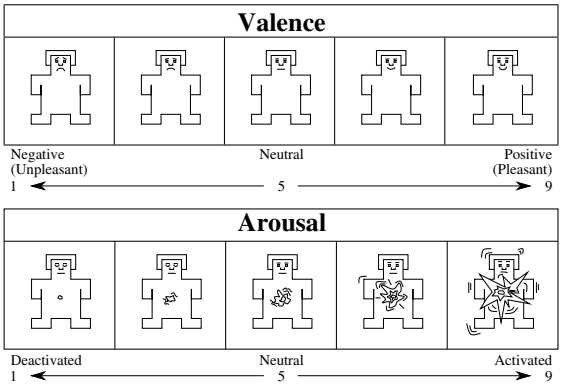


Fig. 3. Self-assessment manikin. [6]

the self-reported labeling accuracy by relabeling the data more discriminatively.

To date, most work in the area of time series has been supervised in nature, which typically requires a large amount of human annotation prior to event detection. However, labeling temporal affective status events is usually a labor-intensive, ambiguous process (e.g., where does the event start/end, what should the temporal granularity of the event be?) and is difficult to standardize across coders. Moreover, the subtle differences between two classes of temporal signals may not be visible over the entire observation period. For example, consider the task of classifying affective status. The differences between these two distinct classes do not appear continuously throughout the time series, but are rather associated with specific temporal patterns possibly occurring multiple times. This is a very challenging problem because the discriminative patterns are not known a priori. In this paper, we propose a multi-instance learning algorithm (for weakly-supervised learning) that facilitates faster, and often more accurate labeling of affective status events of interest with respect to current methods for labeling, such as human annotation.

We illustrate the use of this technique in the problem of finding temporal segments that provide higher discrimination

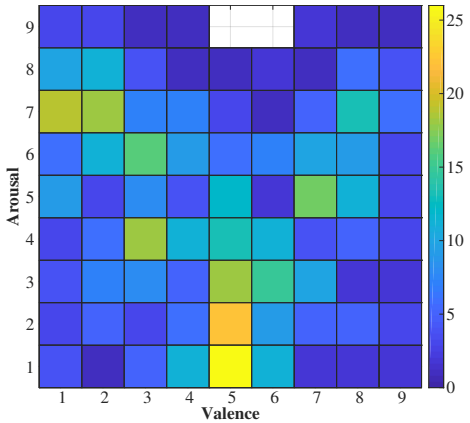


Fig. 4. Histogram showing the distribution of valence and arousal scores, assigned by the subjects from MAHNOB-HCI dataset.

TABLE II

FEATURES EXTRACTED FROM THE PHYSIOLOGICAL SIGNAL AND GAZE DATA

Sensor signal	Extracted features
ECG	*PQRST waveform segments:* Fiducial PQRST points of the ECG waveform, dynamic time warping to estimate the differences along time and amplitude axis of their positions, RMS, mean, standard deviation, skewness. kurtosis, correlation. *HRV (Time domain):* Mean, standard deviation, rMSSD, pNN50, RMS of RR interval. *Wavelet:* The Debauchies-3 level seven wavelet coefficients
EEG	theta, slow alpha, alpha, beta and gamma Spectral Power for each electrode; Spectral Power asymmetry between 14 pairs of electrodes in the four bands of alpha, beta, theta and gamma
GSR	Mean, variance, kurtosis, skewness, min, max, median, change (max-min), proportion of negative elements of the first derivative w.r.t. all the elements, FFT spectrum of the interval
Skin Temperature	Average, average of its derivative, spectral power in the bands ([0, 0.1]Hz,]0.1-0.2]Hz)
Respiration Pattern	Band energy ratio (difference between the logarithm of energy between the lower ([0.05, 0.25]Hz) and the higher ([0.25, 5]Hz) bands), range, mean of derivative (variation of the respiration signal), breathing rhythm (spectral centroid), breathing rate, average breathe depth, spectral power in the bands in [0, 2.4]Hz
Eye Gaze	*Pupil diameter:* Average, standard deviation, spectral power in the following bands -]0, 0.2]Hz,]0.2, 0.4]Hz,]0.4, 0.6]Hz and]0.6, 1]Hz. *Eye blinking:* Blinking rate, length of the longest blink, time spent with eyes closed. *Gaze coordinates:* Standard deviation, skewness, kurtosis, average fixation time, average scan path length, number of fixation zones, spectral power in the following bands]0, 0.2]Hz,]0.2, 0.4]Hz,]0.4, 0.6]Hz,]0.6, 0.8]Hz,]1, 2]Hz, average and the standard deviation of the standard deviation of gaze coordinates in each fixation zone.

between affective status; however, this technique is general and can be applied to any type of affective status event of interest.

1) Notations: Denote s the number of affective status. n the number of movie segments. In each movie segment, we take sensor signals in 30-seconds temporal intervals and use these temporal intervals as the smallest scale for affective status estimation. $\mathbf{x}_i \in \Re^{d\times 1}$ is a vectorized representation of the ith temporal interval. $\mathcal{N}_j$ is the set of temporal interval that belongs to the jth movie segment, (i $\in \mathcal{N}_j$).

We formulate the status estimation as pair-wise classifica-

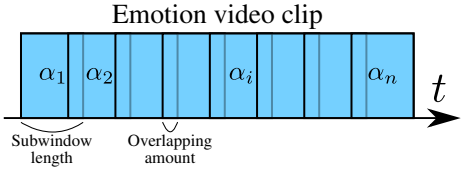


Fig. 5. Extracting features from fixed length sub-intervals (sub-windows) of each emotional video clip and assigning the weights α_i.

tion problems. For instance, to classify high and low valance, y_i is the binary label of $\mathbf{x}_i$ such that $y_i = +1$ if the ith temporal segment belongs to a high valence movie segment, otherwise $y_i = -1$.

The goal of GMS-SVM is jointly learning the classifier and selecting the most discriminative temporal segments in each movie segment.

2) Learning GMS-SVM: **Step 1: Learning a standard SVM classifier**

In the initial step, we use the the label of each movie segment as the labels of each temporal interval that belong to the movie segment.

The standard linear SVM solves

$$\min_{\mathbf{w},b} \frac{1}{2} \|\mathbf{w}\|^2 + \sum_i \|1 - y_i \left(\mathbf{w}^\top \mathbf{x}_i + b\right)\|_h \qquad (1)$$

where w and b are coefficients of the classifier, $\|\cdot\|_h$ is the hinge-loss that measures the classification error.

Step 2: Selecting the discriminative temporal segments Within each movie segment, we solve a set of positive weights, such that

$$\begin{aligned} \arg\max_{\alpha_i^{(k)}} & \sum_{i=1}^{N} y_i^{(k)} \left[\alpha_i^{(k)} f\left(\mathbf{x}_i^{(k)}\right)\right] \\ \text{s.t.} & \sum_{i=1}^{N_k} \alpha_i^{(k)} = 1 \\ & \alpha_i^{(k)} > 0 \end{aligned} \quad , \qquad (2)$$

where $\alpha_i^{(k)}$ are the weights assigned to the feature vectors $\mathbf{x}_i^{(k)}$ belonging to the same bag $\mathcal{B}_k$.

Step 3: Learning a weighted SVM classifier

Step 2-3 iterate until the changes to the classifier coefficients and weights are negligible.

3) Classification on test movie segment: We model the classifier to identify the video segments that stimulate emotion. These are segments with $\alpha_i^{(k)}$ greater than a certain threshold and are used to train the classifier. In the testing phase, we use the trained model to predict the response to the video by first predicting the label of every instance of the video (no segment weighting) and then taking a majority vote to determine the label of the bag as a whole.

B. MS-SVM as a labeling tool for physiological signals

The key difference between signals from two affective statuses is very subtle and too ambiguous for human labeling. One can only roughly label (1) the existence of a signal affective status event, and (2) the approximate time intervals related to affective status. This makes it very hard to train and adapt supervised classifiers for affective status.

In this project, before applying the standard event detection approaches, we propose to use MS-SVM as a labeling tool for exacting segments in the signals corresponding to fatigue. MS-SVM does not require a human to label exact locations of occurrences. Rather, it breaks each roughly labeled signal interval into a bag of shorter segments, and learns a classifier (Eq. 2) between the bags that contain affective status (the positive class) and the bags of non-affective status (the negative class). This learning process requires at least one segment from each affective status bag to produce a positive score. In other words, MS-SVM tries to select the key segments in each positive bag that discriminate this positive bag from the negative bags. Therefore, using MS-SVM as a labeling tool, we can locate more discriminative and finer-scale signal segments from the rough manual labels. After learning MS-SVM, the labeling is done as follows: given manually labeled signal intervals, break each interval into a bag of shorter segments, and select the segments that produce positive scores on the MS-SVM classifier. There are two main advantages of MS-SVM labeling over the exact manual labeling: (1) MS-SVM is more practical for it only requires roughly manual input, e.g., a signal interval contains an affective status event or not; (2) MS-SVM is less prone to human bias for it is based on an objective metric: the MS-SVM classifier score.

In the following sections, the MS-SVM labeled signal segments will be used as positive training samples for affective status detectors.

TABLE III
AVERAGE CLASSIFICATION ACCURACY OF DIFFERENT SVM TECHNIQUES ON AROUSAL

	5-CV	**LOSO**	**LOVO**
Linear SVM	67.31%	32.64%	30.47%
Kernel SVM	50.12%	44.48%	49.32%
MS-SVM	73.64%	58.43%	49.38%

V. EXPERIMENTAL RESULTS

We use multiple instance learning approach for training the model. We first initialize all weights in every bag($\alpha_i^{(k)}$) to one and run an SVM. The decision values from the SVM output are used in the optimization objective of MS-SVM to find the updated weights which represent the importance of the segments in the video. An SVM classifier is then modeled on instances with weights greater than a threshold. Testing is performed on the trained model to obtain the labels of all instances in the test bag. Majority voting strategy is used to determine the label of the bag.

We compared our approach against Linear and Kernel SVM. We test the accuracies of 5-fold cross validation, Leave-One-Subject-Out (training on *n*-1 subjects and testing on one) and Leave-One-Video-Out (training on *n*-1 videos and testing on one) for all 3 methods. Our results show that multiple instance SVMs in specific and the method of extracting discriminating segments of the video in general is promising. Table III shows the results for arousal and Table IV shows the results for valence.

VI. CONCLUSIONS

In this work, we aimed at finding specific segments of the movies that strongly correspond to the emotions felt by the participant in order to improve the accuracy of affective state estimation using physiological signals. We present a

TABLE IV

AVERAGE CLASSIFICATION ACCURACY OF DIFFERENT SVM TECHNIQUES ON VALENCE

	5-CV	**LOSO**	**LOVO**
Linear SVM	62.96%	39.53%	16.47%
Kernel SVM	53.83%	50.38%	53.88%
MS-SVM	70.73%	47.09%	61.26%

multiple instance learning approach to find these segments as a precursor to classification. We believe that future work along this direction may lead to improved classification.

ACKNOWLEDGEMENT

We thank FICCI (Federation of Indian Chambers of Commerce and Industry) for funding this work. Special thanks to Dr. John Dolan, Ms. Rachel Burcin and the entire RISS team for the constant support.

REFERENCES

[1] S. Jerritta, M. Murugappan, R. Nagarajan, and K. Wan, "Physiological signals based human emotion Recognition: a review," in *2011 IEEE 7th International Colloquium on Signal Processing and its Applications*, Mar. 2011, pp. 410–415.

[2] S. Koelstra, C. Muhl, M. Soleymani, J. S. Lee, A. Yazdani, T. Ebrahimi, T. Pun, A. Nijholt, and I. Patras, "DEAP: A Database for Emotion Analysis ;Using Physiological Signals," *IEEE Transactions on Affective Computing*, vol. 3, no. 1, pp. 18–31, Jan. 2012.

[3] M. Soleymani, J. Lichtenauer, T. Pun, and M. Pantic, "A Multimodal Database for Affect Recognition and Implicit Tagging," *IEEE Transactions on Affective Computing*, vol. 3, no. 1, pp. 42–55, Jan. 2012.

[4] C. Godin, F. Prost-Boucle, A. Campagne, S. Charbonnier, S. Bonnet, and A. Vidal, "Selection of the most relevant physiological features for classifying emotion," 01 2015.

[5] J. Russell, "A circumplex model of affect," *Journal of personality and social psychology*, vol. 39, no. 6, pp. 1161–1178, 1980.

[6] M. M. Bradley and P. J. Lang, "Measuring emotion: The self-assessment manikin and the semantic differential," *Journal of Behavior Therapy and Experimental Psychiatry*, vol. 25, no. 1, pp. 49 – 59, 1994.

Predicting Bus Dwell Times in Dense Urban Transit Networks Using Bayesian Hierarchical Modeling

Conor Igoe[1], Eli Bronstein[2], Isaac Isukapati[3], Stephen Smith[4]

***Abstract*— Public transit service quality must improve as urban societies seek to reduce their carbon footprint and utilize more energy-efficient technologies. The current smart transportation frontier involves Transit Signal Priority (TSP) systems, but these fail to make strategic traffic signal control decisions that simultaneously accommodate bus schedules and minimize delays for non-transit vehicles. An accurate characterization of bus behavior can empower the development of TSP systems that maximize public transit efficiency and reliability while still working to optimize overall vehicular flow. Bus behavior is principally characterized by frequent stops with uncertain dwell times, but currently there exists no well-studied model of such behavior appropriate for adaptive traffic systems. To this end, we develop a lightweight, robust, real-time Bayesian hierarchical model for bus dwell time prediction, consisting of an Onboarding (passenger) Model and a Dwell Time Model. We implement and test these models separately, achieving a median absolute error of 1.96 passengers for the Onboarding Model and 3.9 seconds for the Dwell Time Model. These results lay the foundation for a complete hierarchical dwell time prediction model that can be used to inform an adaptive TSP system in real-time.**

I. INTRODUCTION

Road traffic is a major contributor to the carbon footprint of a city, with recent studies suggesting that 27% of U.S. greenhouse gas emissions are attributable to the transportation sector[1]. As urban societies strive to reduce their negative environmental impact and rely on less energy-intensive technologies, it is critically important that the quality of public transit service improve. Such developments would both drive public adoption of more energy-efficient modes of transit and increase the overall efficiency of one of the major sources of urban transport related emissions[2].

The current frontier of smart transportation systems seeks to improve public transit, specifically buses, through signal control strategies. Typically, these strategies adopt one of two approaches. One approach takes advantage of adaptive traffic signal control systems to focus on overall vehicular flow, ignoring the specific scheduling objectives of buses. The other approach involves transit signal priority (TSP) methodologies that respond to driver-initiated preempts

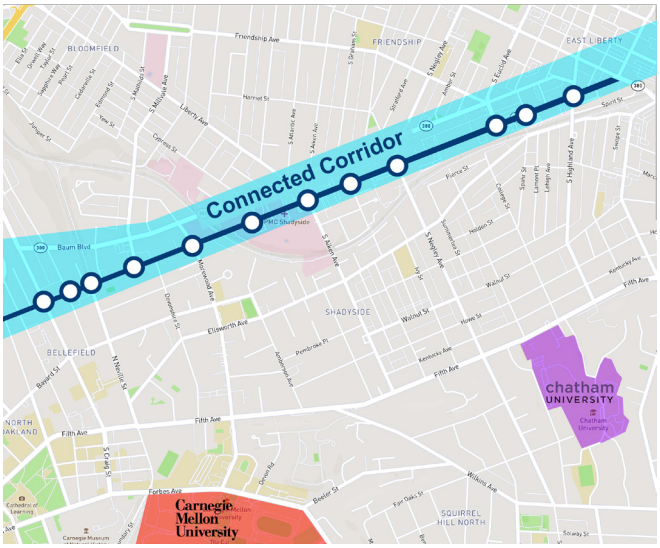


Fig. 1: Map of the bus stops considered in this study. 12 bus stop locations are shown, 6 of which have an inbound and an outbound stop.

by either prematurely creating a green light condition or extending the green for the approaching bus. However, due to their reflexive nature and inability to forecast, current TSP methods serve only to shorten overall transit duration, and are not effective at ensuring bus schedule reliability or service quality. Furthermore, TSP systems result in myopic traffic signal schedules that fail to be strategic in determining the best control decisions to implement for the benefit of all traffic.

These shortcomings serve as a clear mandate for innovation in technologies aimed at improving public transit quality. Our hypothesis is that as the number of connected vehicles continues to grow, adaptive traffic control systems ought to capitalize on the availability of real-time data streams in order to enable high quality service. Due to the dependence on traffic flow forecasting, a solution involving adaptive traffic signal control is not possible without an understanding of bus behavior and its effect on surrounding traffic. Such behavior is primarily characterized by frequent stops with uncertain dwell times. Moreover, the presence of buses stopping on urban streets typically restricts or blocks other traffic on the road, resulting in increased overall wait times and delays throughout the system. In this regard, a more accurate characterization of bus dwell times can empower the development of TSP systems that maximize public transit

*C. Igoe and E. Bronstein are co-authors.
[1]C. Igoe is with the School of Electrical & Electronic Engineering, University College Dublin, Ireland. conor.igoe@ucdconnect.ie
[2]E. Bronstein is with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, USA. ebronstein@berkeley.edu
[3]I. Isukapati is with the Robotics Institute, Carnegie Mellon University, USA. isaack@cs.cmu.edu
[4]S. Smith is with the Robotics Institute, Carnegie Mellon University, USA. sfs@cs.cmu.edu

efficiency and reliability while still working to optimize overall vehicular flow.

In this research, we explore the efficacy of Bayesian random effect modeling techniques for bus dwell time prediction. We develop a Bayesian hierarchical model that is computationally efficient in delivering real-time predictions, robust to the stochasticity of bus dwell times, generalizable to various urban network scenarios due to its low data dependency, and capable of addressing uncertainty by delivering predictions with corresponding confidences. We aim to eventually field test the efficacy of this model in the connected vehicle test-bed of the Pittsburgh area.

In this paper, we first describe and analyze the bus dwell time dataset used throughout the research. We then explore parametric distribution fits in order to analytically describe the dwell time data. Next, we formalize our Bayesian hierarchical model, detail its implementation, and assess its accuracy on historical data. Finally, we discuss the implications of our data analysis and modeling, and suggest avenues for future work.

II. APPLICATION CONTEXT

Uncertainty in dwell times serves as an obstacle in many smart transportation settings:

- As discussed, TSPs aim to improve the quality of bus service but currently have limited broader impact due to their inability to forecast and make strategic scheduling decisions for all traffic. This limitation is a direct result of dwell time uncertainty.
- Bus scheduling remains unreliable due to the difficulty of making accurate timing plans. The compounding uncertainty in dwell times of stops along a route amounts to a high degree of uncertainty in overall route duration, which gives rise to timetables of limited value.
- Reporting accurate real-time bus information (e.g. the Expected Time of Arrival of the next bus at a particular stop) is made difficult due to the uncertain dwell time at each preceding stop.

In general, a lightweight model of bus dwell times enables further research of traffic phenomena that depend on bus behavior.

III. TRANSIT DATA

A. Overview

The specific dataset used in this research was provided by the Port Authority of Allegheny County (PAAC) for weekdays from September 2012 to August 2014. The dataset includes records for 18 bus stops along Centre Avenue on the ”Connected Corridor” indicated in Figure 1, a region of particular interest due to its high volume of public transit activity. These records were obtained using On-Board Units (OBUs) that record a variety of data for the majority of Port Authority buses travelling along the Connected Corridor.

Table IV in the appendix describes each of the columns in the dataset, where each row corresponds to an individual bus passing one of the 18 bus stops. Note in particular that the dataset includes events where a bus passes one of the 18 bus stops without stopping. Relevant artifacts pertaining to our dataset include:

- No weekend data was available.
- No data was available for August 2013 and November 2013.
- A small fraction of the data was marked as invalid bus stops.

B. Longitudinal Analysis

It is clear that bus dwell times change throughout the day due to shifting ridership patterns. For example, more passengers are expected during morning and evening rush hours, which could increase dwell times. It is important to understand how the dwell time distributions change throughout the day, as well as how much similarity there is in the dwell times over a short period of time. To conduct this analysis, we focused on the Negley Avenue at Centre Avenue stop (direction 1) for morning rush hour (6:00-9:59 am) in spring 2014, considering the days of the week separately. For each test, we divided the data into 15 minute intervals starting 5 minutes apart (e.g. 6:00-6:15 am and 6:05-6:20 am). We then assessed the similarity of the dwell time distributions for each of these intervals using max-deviation tests[3]. A max-deviation test between intervals A and B involves computing all of the percentiles (0-100) of A and B and checking how many of these have an error percentage within a specified threshold (e.g. 5%). The number of percentiles within the error threshold is the max-deviation score. For the morning rush hour, there are 46 individual 15 minute intervals. We computed the max-deviation scores of each of these intervals with all of the other intervals, and visualized the scores as a 46x46 heatmap.

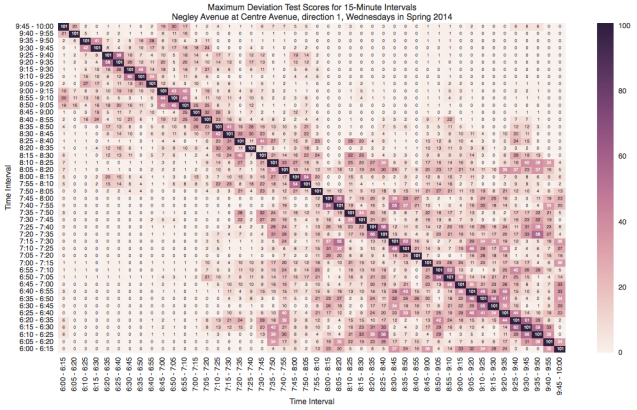


Fig. 2: Heatmap of maximum deviation test scores for 15-minute intervals spaced 5 minutes apart throughout the morning rush hour (6:00-9:59 am) for Wednesdays in spring 2014.

Figure 2 indicates that dwell time distributions that are temporally closer to each other are more similar, which is demonstrated by darker regions near the main diagonal and lighter regions farther from it. Intervals near the end of morning rush hour, approximately after 8 am, exhibit slightly more similarity than those near the beginning of the

time period, though not by a large amount. Intervals that are directly 5 minutes apart (one step above and below the main diagonal) are the most similar, with max-deviation scores that generally lie in the 30s-50s range. These scores are not very high, considering there is an overlap of 10 minutes (approximately $\frac{2}{3}$ of the interval, assuming the bus stops are uniformly distributed in time) between adjacent intervals. Intervals that are 15 minutes apart (e.g. 6:00-6:15 am and 6:15-6:30 am) have significantly lower max-deviation scores, which usually fall below 25. Such low scores indicate the data is highly stochastic, and dwell time distributions can change significantly over a short period of time.

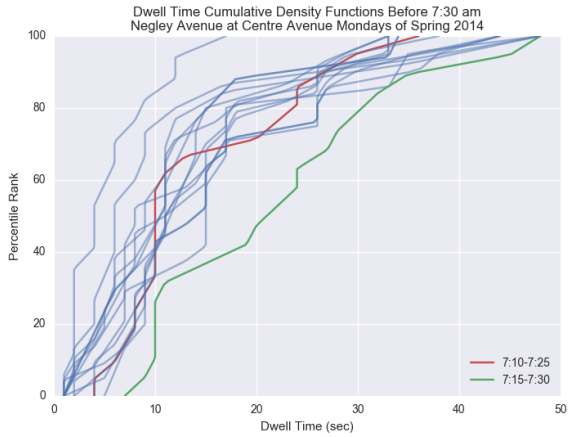


Fig. 3: Dwell time cumulative density functions for Negley Avenue at Centre Avenue, direction 1, Mondays of spring 2014. The 7:10-7:25 am data is in red, the 7:15-7:30 am is in green, and all of the intervals before the 7:10-7:25 am interval are in blue.

An interesting phenomenon displayed in the heatmaps is an abrupt change in the max-deviation scores that occurs in a span of 5 minutes. To visualize this change, consider Figure 3, which shows a cumulative density function (CDF) of the 7:10-7:25 am data in red, the 7:15-7:30 am data in green, and all of the intervals before 7:10 am in blue. The 7:10-7:25 am CDF is clearly much more similar to all of the previous CDFs than the 7:15-7:30 am CDF is, which indicates a sudden change in dwell times in a span of just 5 minutes (7:25-7:30 am).

To visualize how the dwell time distributions change throughout the morning rush hour, consider Figure 4, which displays the CDF plots of the 15 minute intervals, grouped by hour. Several observations can be made from these plots. Shorter dwell times are more common than longer ones, particularly earlier in the day. As morning rush hour progresses, longer dwell times become more common and the higher percentiles increase dramatically. In addition, the variance of each CDF increases as time goes on. See Table I for max-deviation scores of the CDFs for a particular hour for each day of the week. In general, this deviation is larger later in the morning. Moreover, for most of the hour ranges, the CDFs are relatively similar for smaller percentiles (e.g. below

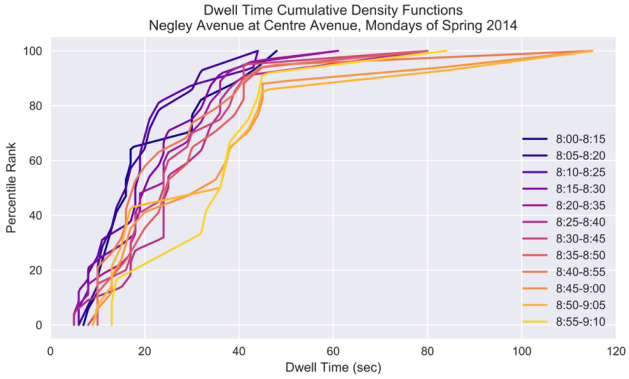


Fig. 4: Dwell time cumulative density functions for 15-minute intervals spaced 5 minutes apart, from 8 to 9 am. As the time progresses, the duration and variance of dwell times increase.

the 70th percentile). However, the larger percentiles exhibit very unstable CDFs due to the highly stochastic nature of the dwell times.

The longitudinal analysis suggests that the early morning is characterized by short dwell times, most likely due to a small number of passengers. Moreover, the dwell time distributions are quite stable early in the morning, but become highly unstable as the day goes on. Due to the large variance and highly stochastic nature of dwell times, the potential for large prediction errors is increased, necessitating a robust approach to dwell time prediction.

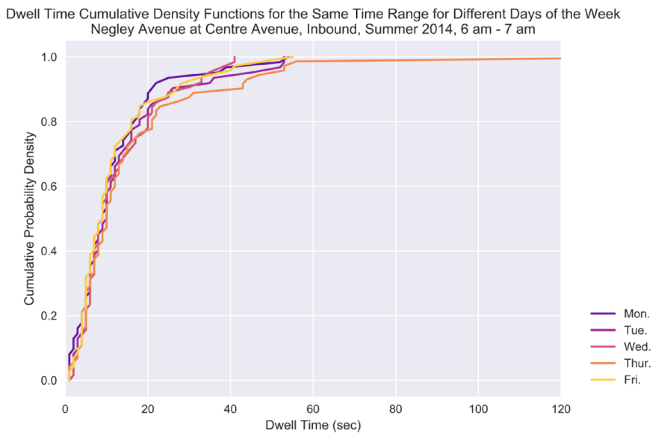
	Monday	Tuesday	Wednesday	Thursday	Friday
6-7 am	11	7	7	9.5	11
7-8 am	11	7.5	11	11.5	10
8-9 am	20	10	13.5	6	17
9-10 am	16	9.5	4	11	12

TABLE I: Largest dwell time percentile deviation of all the CDFs of 15 minute intervals starting 5 minutes apart, in a particular hour interval for each day of the week. This data is for Negley Avenue at Centre Avenue, direction 1, spring 2014.

C. *Cross-sectional Analysis*

We also analyze the similarity of bus dwell time distributions for different days of the week. Figure 5 shows the CDFs of bus dwell times for the inbound bus stop Negley Avenue at Centre Avenue. Each CDF is calculated for one specific day of the week (i.e., all Mondays in summer 2014, or all Tuesdays in summer 2014, etc.) using data from a particular time range spanning 1 hour during the morning rush period in summer 2014. The main observations from these plots include:

- The 50th percentile increases as the morning progresses, for all days.
- The variance in dwell times increases as the morning progresses, for all days.



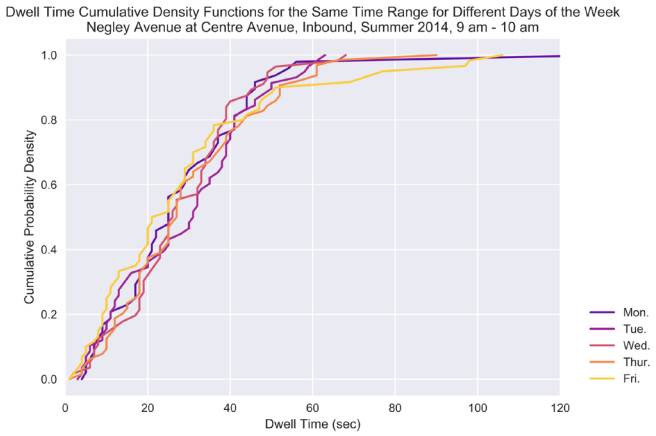


Fig. 5: Dwell time CDFs for each weekday for 6-7 am (upper) and 9-10 am (lower), at Negley Avenue at Centre Avenue during summer 2014.

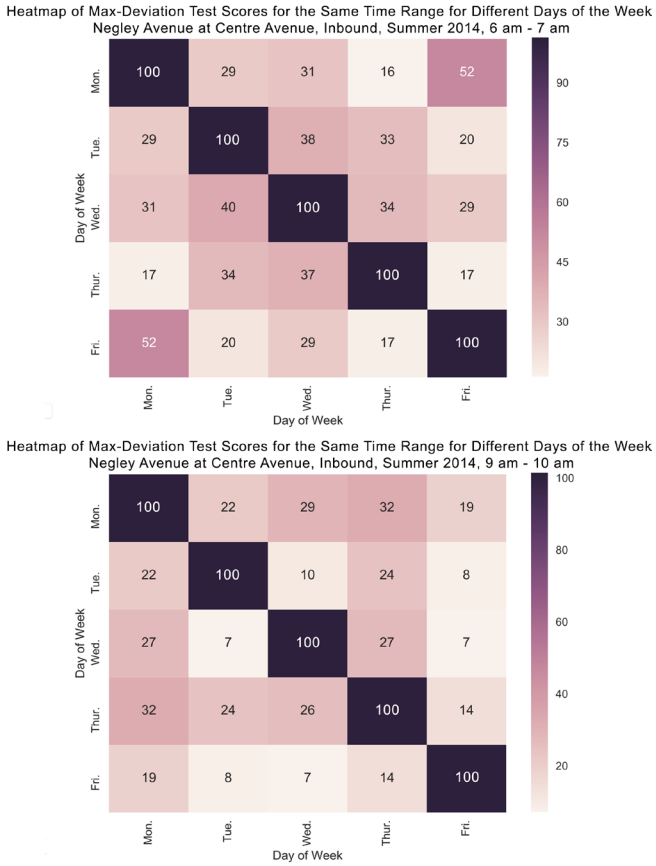


Fig. 6: Max-deviation scores using a 5% threshold for each weekday for 6-7 am (upper) and 9-10 am (lower), at the Negley Avenue at Centre Avenue stop (direction 1) during summer 2014.

- The general shape of the distribution prevails cross-sectionally across days for all specific hour ranges, with moderate cross-sectional deviation.

In other words, it is generally the case that as the morning progresses, dwell times become longer and more unpredictable for each day of the week.

Figure 6 shows the resulting scores for computing the max-deviation test between the CDFs shown in Figure 5 for an error threshold of 5%. The main observations from these plots include:

- No test resulted in a particularly high score, suggesting that the days of the week are statistically significantly different.
- The highest scores were in the high 40s and low 50s, and correspond to Monday-Thursday compared to Monday-Thursday for the 7-8 am range.
- Friday seems to differ the most from the other days of the week, except when compared to Monday 6-7 am.

D. Distribution Fitting

We analyze the representation of dwell times using specific analytic distributions in order to ease computational requirements. If such an analytic distribution is found, computing the posterior dwell time distribution using Bayesian updates can be done efficiently using Markov chain Monte Carlo (MCMC) sampling. For this analysis, we fit 6 distributions commonly used in survival analysis to the data and assessed their goodness of fit. The same aggregation approach as described in the ”Sparsity and Aggregation” subsection (in the Appendix) was employed throughout model fitting, focusing on spring and summer of 2014. Due to the discretization of the dwell time into seconds, we computed the Kernel Density Estimation (KDE) of the dwell times using bandwidths of 0.5, 0.1, and 0.2 to obtain a continuous distribution. This allowed max-deviation tests to yield more meaningful results that were not skewed by the discrete nature of the empirical data.

Figure 7 shows a sample of the distribution fits for the indicated time ranges for the Non-central F, Burr, Weibull, Beta, Log-normal, and Fisk (Log-logistic) distributions. We computed the max-deviation scores between each analytic distribution fit and each of the 3 KDEs, reporting the highest score and which bandwidth it corresponds to.

Figure 8 shows the distribution of the max-deviation scores for each fitted distribution and each KDE bandwidth, and Table II contains the means of these distributions, which

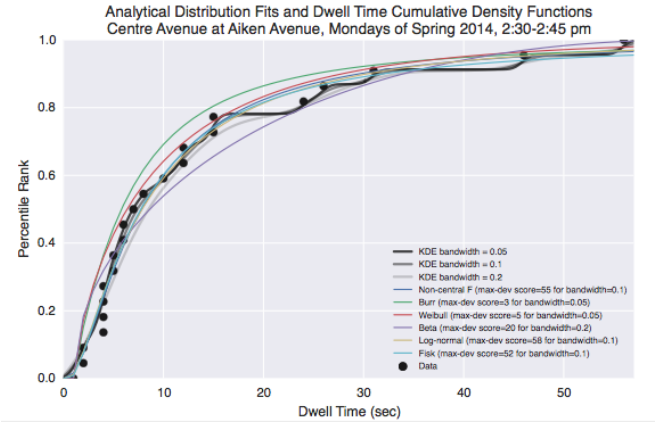


Fig. 7: Dwell time cumulative density function KDEs with bandwidth values of 0.05, 0.1, and 0.2, along with the analytical distribution fit CDFs and the empirical data. The best max-deviation score and the KDE bandwidth for which it was computed is shown for each distribution fit.

	Non-central F	Fisk (Log-logistic)	Burr	Beta	Log-normal	Weibull
bw=0.05	24.19	26.05	23.06	**20.05**	17.98	**12.35**
bw=0.1	25.79	28.17	**23.15**	19.76	18.70	11.83
bw=0.2	**28.49**	**30.00**	22.92	19.44	**19.31**	11.28

TABLE II: Mean max-deviation score between each fitted analytic distribution and the dwell time distribution Kernel Density Estimation (KDE), for each tested KDE bandwidth. Bolded values indicate the highest max-deviation scores for each distribution. The data is for the Negley Avenue at Centre Avenue bus stop, direction 1, spring 2014.

can be used to rank them by goodness of fit. Despite these low scores, which are to some capacity impacted by the discrete dwell time data, it is worthwhile to explore analytic distributions for the prediction model.

E. Covariate Analysis

In order to develop a dwell time regression model with covariates, several relationships were explored between covariate data and dwell time, such as the number of onboarding passengers, number of alighting passengers, and load of the bus. Figure 9 shows the relationship between onboardings and dwell time for the inbound Centre Avenue at Aiken Avenue stop in spring of 2013. We found a clear positive correlation between all aforementioned covariates and dwell time that weakens for higher covariate values. Each of these relations is non-linear, suggesting that kernelization functions would be worthwhile to use in future work.

Figure 10 offers another perspective on the relationship between the number of onboarding passengers and dwell time. Here, Kernel Density Estimation (KDE) was used to determine estimates of dwell time distributions for a particular number of observed onboardings. Note the near-monotonic increase in median dwell times for increasing covariate values, another illustration of the positive correlation

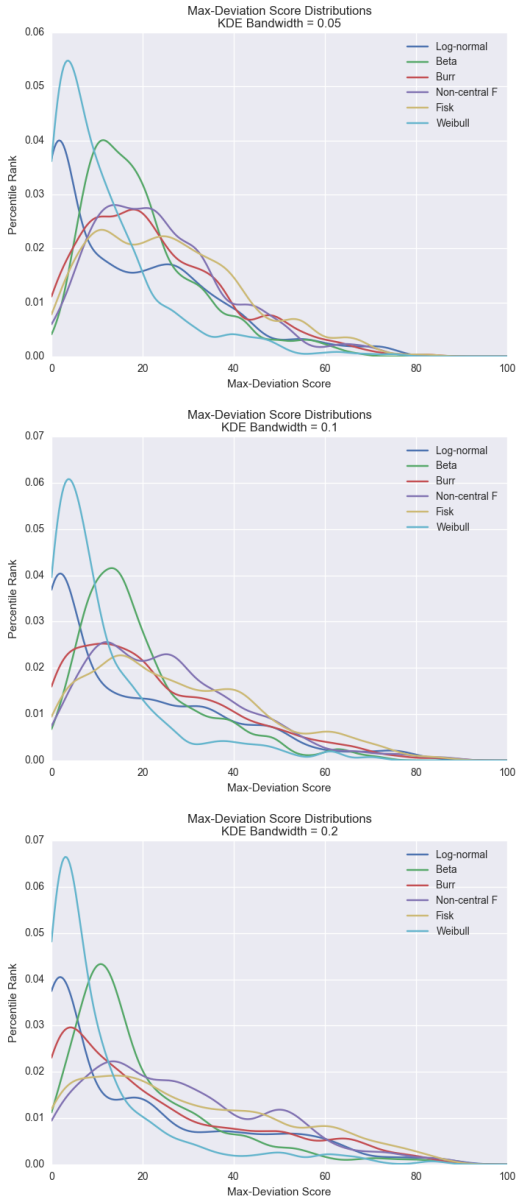


Fig. 8: Plots of the max-deviation score distributions between the fitted analytic distributions and Kernel Density Estimation (KDE) distributions of the dwell time data. These plots were smoothed using KDE with bandwidth of 0.2 to visually highlight the general shape.

concluded from the scatter plots.

More importantly, note that for low covariate values, the mass of the KDE is densely centered over the median, whereas for high covariate values, there is significantly more variance in the distribution. This indicates that predicting long dwell times using onboarding as a covariate will pose a greater challenge.

We also analyzed the relationship between the number of onboardings and the headway, which is defined as the difference between the arrival of a given bus and the departure of the previous bus. Figure 11 displays a positive

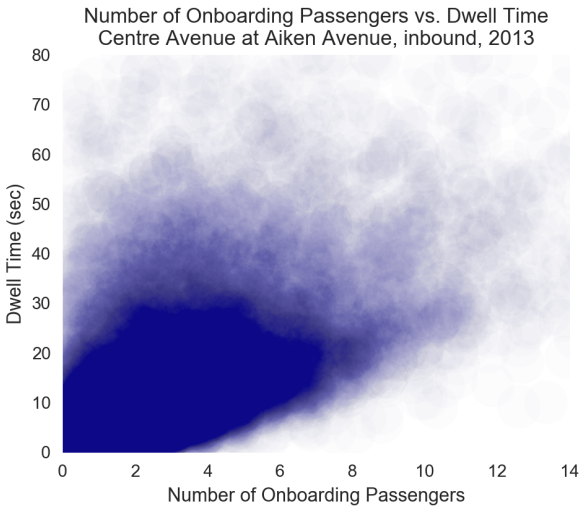


Fig. 9: Scatter plot of the number of onboarding passengers and the dwell time for Centre Avenue at Aiken Avenue in 2013..

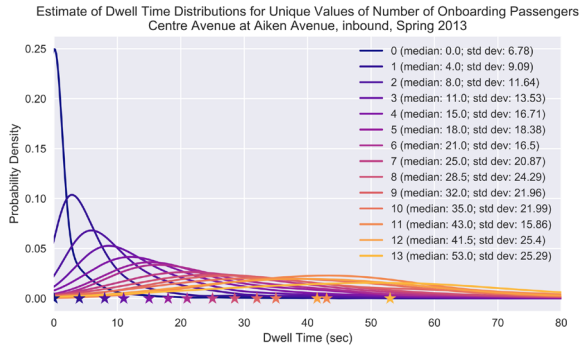


Fig. 10: Kernel Density Estimates of the dwell time probability density function for different numbers of onboarding passengers. Both the dwell time duration and variance are positively correlated with the number of onboardings.

correlation between the headway and onboardings, particularly for a headway of under 12 minutes. The plot may be less informative for longer headways because buses generally pass Negley Avenue at Centre Avenue frequently. There is clearly a large variance in the number of onboardings, which can be seen from the extent of the lighter regions of the plot.

IV. BAYESIAN HIERARCHICAL MODEL

A. *Model Overview*

To predict bus dwell times, we developed the framework of a Bayesian hierarchical model consisting of an Onboarding (passenger) Model, Alighting (passenger) Model, and a Dwell Time Model. Previous research on this topic has primarily focused on exploring the factors affecting bus dwell time using fixed effect regression models. However, this approach is not well-suited for real-time prediction for several reasons. First, dwell times are strictly positive values, but linear regression does not generally adhere to this constraint. Second, linear regression is mainly used

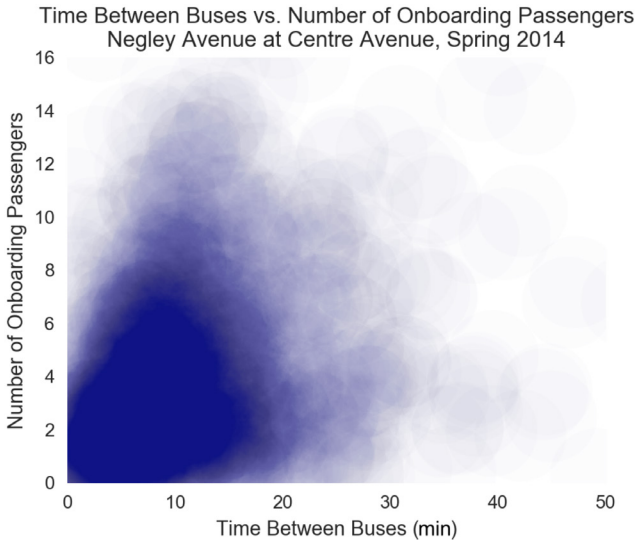


Fig. 11: Scatter plot of headway (time between departure of previous bus and arrival of approaching bus in seconds) and the number of onboarding passengers for Negley Avenue at Centre Avenue in spring 2014.

to explore the relationship between the dependent variable and the covariates, not necessarily to predict the dependent variable in real-time. Third, fixed effect modeling treats the dependent variable as a random variable and the covariates as fixed values, which is an inaccurate representation in the context of predicting bus dwell times using the number of onboarding and alighting passengers as covariates[4]. There is uncertainty regarding the onboardings and alightings for two reasons: passenger count measurements can be noisy due to sensor error, particularly for large numbers of passengers, and the exact value of onboardings and alightings for an approaching bus must be predicted, not measured, to deliver a dwell time prediction. We chose the Bayesian random effect modeling framework to address these issues, as well as to fulfill several important requirements of a reliable, real-time predictive model. First, this framework offers more robust predictions in the face of highly stochastic dwell times, which often have a large variance. Second, the Bayesian framework effectively addresses uncertainty by delivering a confidence in the prediction through the posterior predictive distribution, rather than simply supplying a point estimate. Third, a large amount of data is not necessary to bootstrap the system. The prior distribution can be computed from a small amount of historical data, after which the model can begin making predictions and updating the posterior distribution in a rolling fashion. Fourth, the model is computationally efficient because analytical posterior distributions are simply described by their parameters, and non-analytical distributions can be sampled efficiently using MCMC methods. Computational efficiency is crucial in order to predict bus dwell times in real-time, allowing a TSP to adaptively update the signal scheduling plan.

The pipeline of the hierarchical model begins with the Onboarding and Alighting Models and ends with the Dwell

Time Model. As a bus approaches a bus stop, the headway is estimated by using the bus's current geographic position and speed. The Onboarding Model utilizes the headway as a covariate to obtain a posterior predictive distribution of onboardings. The Alighting Model similarly produces a posterior predictive distribution of alightings, though the details of its implementation are left for future work. The Dwell Time Model then uses the onboarding and alighting distributions as covariates and samples the posterior predictive dwell time distribution using MCMC methods. The model can now deliver various point estimates of this distribution, along with corresponding confidences. Once the bus completes its dwell, the observed onboardings, alightings, and dwell time can be used to update the Onboarding, Alighting, and Dwell Time Models.

B. Onboarding Model

As described earlier, the dwell time is positively correlated with the number of onboarding passengers. To predict the dwell time of an approaching bus, we must first predict the number of onboardings, which is positively correlated with the headway, or the time between the departure of the previous bus and the arrival of the approaching bus. Thus, we develop a Bayesian model for the arrival rate of passengers at a bus stop, using the predicted arrival rate and estimated headway to deliver an onboarding prediction.

We model the arrival rate of passengers at a bus stop using a Poisson distribution with parameter λ. For the purpose of Bayesian updates, we wish to derive $p(\lambda|y)$, the posterior for λ. Given an observed number of onboardings y, assuming the observations are independent,

$$p(y|\lambda) = \Pi_{i=1}^{n} \frac{\lambda^{y_i} e^{-\lambda}}{y_i!} \propto \lambda^{ny} e^{-n\lambda}$$

This is the kernel of a Gamma distribution. Therefore, if $\lambda \sim Ga(\alpha, \beta)$, then

$$p(\lambda|y) \propto p(y|\lambda)p(\lambda)$$
$$p(\lambda|y) \propto \lambda^{n\bar{y}} e^{-n\lambda} \lambda^{\alpha-1} e^{-\beta\lambda} = \lambda^{\alpha+n\bar{y}-1} e^{-(\beta+n)\lambda}$$
$$p(\lambda|y) \sim Ga(\alpha + n\bar{y}, \beta + n)$$

where β is the number of previous observations and α is the sum of previous arrival rates.

We consider a non-informative prior such as Jeffreys' prior, so $p(\lambda) \propto J(\lambda)^{\frac{1}{2}}$ where $J(\lambda)$ is the Fisher information, which is the negative expectation of the second derivative of the log likelihood.

$$\log\, p(y|\lambda) = -\log\,(y!) + y\log\,(\lambda) - \lambda(\log \text{ likelihood})$$

The second derivative of the above function is equal to $\frac{-y}{\lambda^2}$.

$$J(\lambda) = -E[\frac{-y}{\lambda^2}|\lambda] = \frac{1}{\lambda}$$
$$J(\lambda)^{\frac{1}{2}} = \frac{1}{\sqrt{\lambda}}$$

The previous equation can be treated as $Ga(\frac{1}{2}, 0)$. Note that this is an improper Gamma, but it is acceptable for the purpose of Bayesian updates.

In order obtain a posterior arrival rate distribution via a Bayesian update, we maintain a list of observed arrival rates, which are defined by the number of onboardings divided by the headway. Once a new observation (headway and onboardings) is made, we compute the arrival rate and append it to the list. We then calculate a new value for alpha by summing the recent β arrival rate observations, where β is an integer that should be empirically found to maximize prediction accuracy. To make an onboarding prediction for an approaching bus, we multiply a point estimate of the posterior arrival rate distribution by the headway, which can be estimated using the bus location and speed. Various options for the arrival rate point estimate are possible, such as mean, median, mode, etc.

C. Dwell Time Model

The following describes a Bayesian parametric model for bus dwell times using two covariates: the number of onboarding and alighting passengers. Due to the goodness of fit and clear interpretability of the Log-Normal distribution, we model a bus dwell time y as a random variable following a Log-Normal distribution with parameters (μ, τ^2):

$$f(y|\mu, \tau^2) = \frac{\tau}{y\sqrt{2\pi}} \exp\left\{-\frac{\sigma^2}{2}(\log(y) - \mu)^2\right\}$$

where $\sigma = 1/\tau$. At any given time, our belief of the current distribution of bus dwell times is described by our belief of the two parameters μ and τ. In a real-time system with access to dwell time observations, we continuously update our beliefs of the parameter distributions to obtain new beliefs given our most recent observation. Bayes' Theorem offers a natural way to achieve such an update scheme. As we are only considering one observed dwell time d during any Bayesian update, the likelihood function is

$$L(\mu, \tau|d) = f(y = d|\mu, \tau^2)$$

To construct the regression model, we introduce the 2 covariates through μ and write $\mu = f(x)'\beta$, where x is a 2×1 vector of covariates, β is a 2×1 vector of regression coefficients, and f is a kernelization function. The first elements of x and β correspond to the number of onboarding passengers and regression coefficient for observed dwell time d, respectively; the second elements correspond to the number of alighting passengers. We use a power kernelization $f(x) = x^a$ to account for the non-linear relationship between dwell time and covariates, where a is a hyperparameter value that lies between 0 and 1 to ensure a monotonically decreasing positive slope in $f(x)$ that was observed in the data.

Before obtaining any posterior distributions to use as priors in our real-time system, we use a Normal prior for each of the 2 regression coefficients in β and a Half-Normal prior

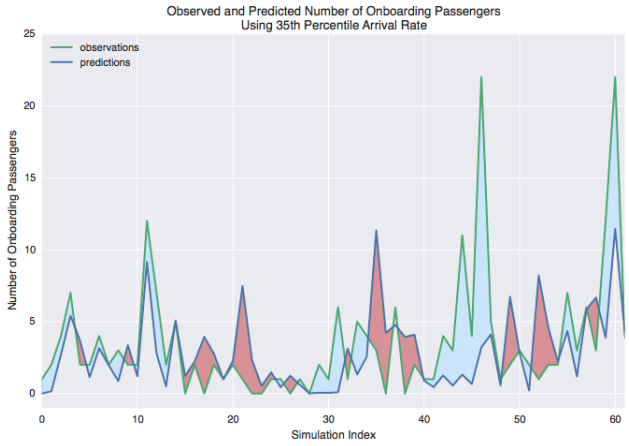


Fig. 12: Observed and predicted onboardings in simulation. The previous 8 observations were used to inform the predictive onboarding distribution, and the 35th percentile was used for prediction. The data is for Negley Avenue at Centre Avenue, the first Monday of spring 2014, starting at 6 am.

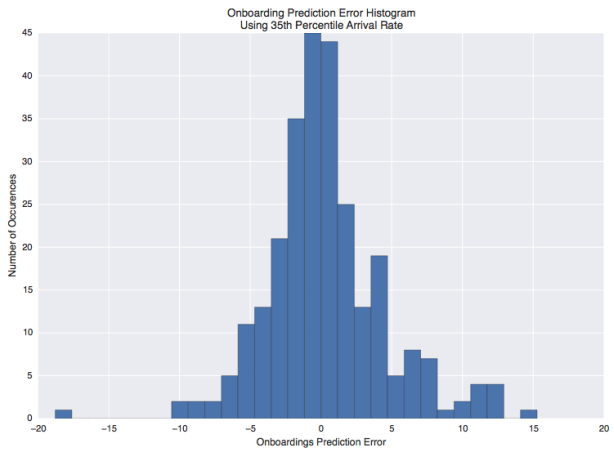


Fig. 13: Histogram of the Onboarding Model prediction error. The previous 8 observations were used to inform the predictive onboarding distribution, and the 35th percentile was used for prediction. The data is for Negley Avenue at Centre Avenue, the first Monday of spring 2014, starting at 6 am.

for the model parameter τ with appropriate hyperparameters as determined through independent experimental analysis. Once a set of posterior distributions are obtained, we use the most recent posterior distributions as priors in our next Bayesian update.

Closed forms for the posterior distribution of β are generally not available, necessitating the use of numerical integration or MCMC methods. In this paper, we utilize the Metropolis Hastings algorithm to obtain MCMC samples of the posterior distribution over β.

To make a dwell time prediction for an approaching bus, we estimate a set of covariates using the Onboarding and Alighting Models, and use our posterior distributions of β and τ to determine the posterior predictive distribution of y.

The next time we observe a dwell, we repeat the process outlined above, using our most recent posterior distributions of β and τ as priors for the next Bayesian update.

V. RESULTS

A. Experimental Setup

We simulate a real-time prediction scenario for an individual bus stop by stepping through the historical data chronologically. The Onboarding and Dwell Time Models are implemented and assessed separately. Both models assume error-free covariate prediction, meaning that the Onboarding Model uses the correct headway and the Dwell Time Model uses the correct number of onboardings and alightings.

B. Onboarding Model

The Onboarding Model was tested on the Negley Avenue at Centre Avenue bus stop for Mondays of spring 2014, which contained 1193 observations with a mean onboarding passengers value of 3.65. We tried different values of β, the number of previous observations to use for the posterior arrival rate distribution, ranging from 3 to 100. We also

	35th Percentile	Median	Mean	60th Percentile
$\beta = 3$	2.90, 1.99	3.25, 2.18	3.58, 2.32	3.66, 2.39
$\beta = 5$	**2.84**, 1.97	**3.19**, 2.51	**3.42**, 2.35	**3.55**, 2.47
$\beta = 8$	2.99, **1.96**	3.30, **2.12**	3.49, **2.20**	3.63, **2.38**

TABLE III: Mean and median (format: mean, median) absolute value errors for onboarding passenger predictions using various predictive summary statistics of the arrival rate and values for β (the number of previous observations used for prediction). Bolded values indicate the best value for β and the underlined values indicate the best summary statistic for this value of β.

experimented with different point estimates of the arrival rate distribution, such as the mean, median, 35th percentile, and 60th percentile. From Table III we observe that predicting the 35th percentile for the arrival rate minimized the mean absolute error, with β=5 achieving the lowest error of 2.84 onboarding passengers. We similarly see that predicting the 35th percentile for the arrival rate minimized the median absolute error, with β=8 achieving the lowest error of 1.96 onboardings. Figure 13 shows the histogram of prediction errors with β=8, which has a standard deviation of 4.22. Although the prediction error is sizeable compared to the mean number of onboardings, an error of approximately 2 passengers should not significantly impact the dwell time prediction. Figure 12 illustrates that the predicted onboardings generally follow the pattern of observed onboardings, though poor predictions are more common for particularly large onboarding values.

C. Dwell Time Model

The Dwell Time Model was tested on the Centre Avenue at Aiken Avenue bus stop for a typical day in winter of 2013, which contained 184 observations with a mean

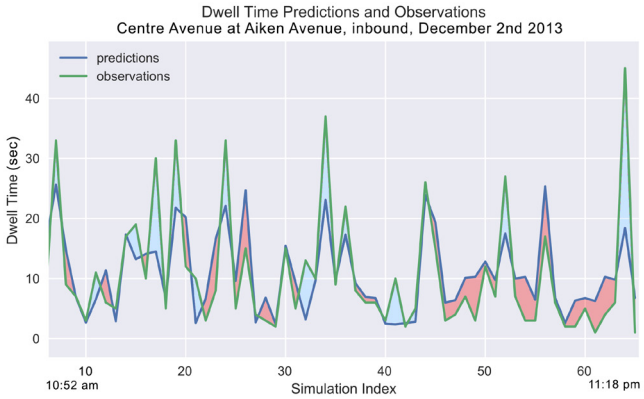


Fig. 14: Observed and predicted dwell time in simulation for Centre Avenue at Aiken Avenue on December 2nd, 2013, between 10:52 and 11:18 am.

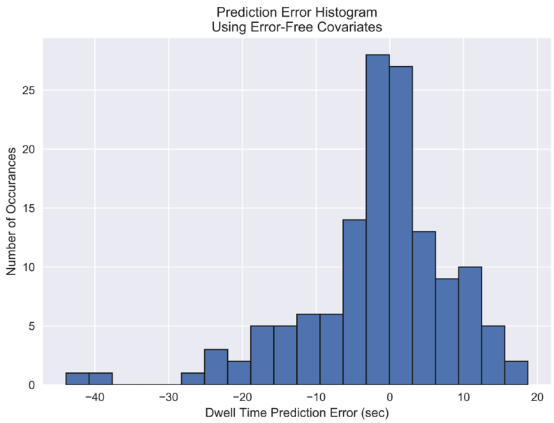


Fig. 15: Histogram of the Dwell Time Model prediction error using the ground truth for the number of onboarding and alighting passengers.

dwell time of 10.9 seconds. Hyperparameter $a = 0.4$ was experimentally found to optimally reduce overprediction bias for this data set. Figure 14 shows a comparison of predictions and observations and Figure 15 shows the distribution of dwell time prediction errors. Note in particular that longer dwell time predictions are typically more inaccurate, a direct result of the high degree of variability in dwell times for large numbers of onboarding passengers.

Although a mean absolute error of 7.12 seconds may seem high, it is important to note that this statistic is overly biased by the comparatively large errors arising from long dwell time predictions. The median absolute error of 3.86 seconds is thus a fairer metric of performance in this setting.

VI. DISCUSSION

A. Limitations

The dataset itself had particular limitations, such as the absence of weekends and the discrete representation of the dwell time in integer seconds. Moreover, we focused our statistical analysis on the popular bus stops in the connected vehicle corridor of the Pittsburgh area, so the results must be tested in other areas as well. In terms of the hierarchical model, the Onboarding Model used error-free headway estimation, which is not an unrealistic assumption because the velocity and location of a bus can be used to accurately approximate its arrival time. The model also assumed that the onboarding observations are independent, but this fact is unlikely to introduce significant modeling error. Since the the Dwell Time Model was tested in isolation, it utilized the correct number of onboardings and alightings. Once the onboarding and Alighting Model predictions are used instead, the dwell time prediction error will inevitably increase.

B. Future Work

Further research directions involve improving the performance of the implemented models, completing the hierarchical model, and conducting field tests. We aim to increase the accuracy of the Onboarding and Dwell Time Models, particularly by testing other parametric distributions and kernelizations for the latter. Developing a model for alightings passengers will then allow for integration of the complete Bayesian hierarchical model. This model can be tested on historical data, and eventually a field test can be conducted to assess the model's generalizability and real-time capabilities. Finally, the development of a classification model to predict whether a bus will dwell can lead to a much more complete characterization of bus behavior.

ACKNOWLEDGMENT

Thank you to Dr. Isukapati, Prof. Smith, Rachel Burcin, Prof. Dolan, Cormac O'Meadhra, and the entirety of the RISS team. This work was supported by the National Science Foundation.

REFERENCES

[1] "Inventory of U.S. Greenhouse Gas Emissions and Sinks: 1990-2015", U.S. Environmental Protection Agency, Washington, DC, Rep. 430-R-16-002, Apr. 15, 2017.

[2] S. Smith, G. Barlow, X.-F. Xie, and Z. Rubinstein, "SURTRAC: Scalable Urban Traffic Control," in Transportation Research Board. 92nd Annual Meeting Compendium of Papers, January 2013.

[3] I. K. Isukapati and G. F. List. "Synthesizing route travel time distributions considering spatial dependencies," in Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on (pp. 2143-2149), November 2016.

[4] I. K. Isukapati, H. Rudova, G. Barlow, and S. Smith. "Analysis of Trends in Transit Bus Dwell Time Data," in Transportation Research Board, January, 2017.

ANAME	Bus Stop Name (String)
QSTOPA	Bus Stop ID (String)
DIR	Direction of trip along route (Integer, where 0 indicates inbound (toward downtown) and 1 indicates outbound)
ROUTE	Route Code (String)
TRIPA	Trip Number (String of Hour+Minute, for a trip beginning at hour:minute)
CAP	Capacity of Bus (Integer)
VEHNOA	Vehicle Number (Integer 3xxx-5xxx)
ARR	Arrival Time and Date (String "YYYY-MM-HHThh:mm:ss")
DEP	Departure Time and Date (String "YYYY-MM-HHThh:mm:ss")
DWTIME	Length of Dwell in Seconds (Integer)
ON	Number of Onboarding Passengers (Integer)
OFF	Number of Alighting Passengers (Integer)
LOAD	Number of Passengers on Bus (Integer)
SRTIME	Scheduled Run Time from Previous Record to Current Record
ARTIME	Actual Travel Time from Previous Record to Current Record

TABLE IV: Dataset field names and descriptions.

VII. APPENDIX

A. Data Processing

The original data contained several unnecessary and unintuitive fields, requiring cleaning to make the dataset easier to manage and analyze. The primary change regarded the arrival and departure datetimes of each record. The original dataset included individual fields for the arrival and departure hour, minute, and second, as well as a field for the day, month, and year of the bus run. The hour field contained integer values between 3 and 26, where 24, 25, and 26 referred to midnight, 1 am, and 2 am of the day after the bus run started, respectively. We condensed these multiple values into an arrival field and a departure field, both of which are complete datestrings (YYYY-MM-HHThh:mm:ss) referring to when the bus arrived at and departed from the bus stop, respectively. Moreover, the original dataset recorded the dwell time in minutes up to two decimal places, which resulted in a loss of precision. Therefore, we converted the dwell time to seconds by taking the difference in the departure and arrival datetimes. Furthermore, the original dataset contained a vehicle number field corresponding to the bus model, and a table mapping bus model to capacity (number of passengers). However, the mapping did not include all bus models in the dataset. This information was incorporated as a bus capacity field for bus models in the table, while the other bus model capacities remained blank.

B. Data Reformatting

The original data was organized by month, which resulted in large CSV files that would make file-reading a bottleneck in the data analysis process when accessing specific days or weeks rather than entire months. Thus, we grouped the data by date and by the bus stop code, since it was common to analyze individual bus stops. This organization made it simple to create a filtering system to return the filenames corresponding to specific bus stops, years, seasons, months, and days of the week.

Spring	Jan 7 - May 14 (92 weekdays)
Summer	May 15 - Aug 31 (77 weekdays)
Fall	Sep 1 - Dec 14 (75 weekdays)
Winter	Dec 15 - Jan 6 (17 weekdays)

TABLE V: Season date.

However, bus stop names and bus stop codes became a consistency issue because neither of these fields was a unique identifier for a particular bus stop, and there was no one-to-one mapping between these two fields. A particular bus stop code could refer to multiple bus stop names, which usually, but not always, correspond to the same physical bus stop. This occurred when a normal stop was distinguished from an end-of-line (EOL) stop, when bus stop names were changed, and in certain cases in which "at" and "opposite" where grouped together (e.g. "STREET A AT STREET B" and "STREET A OPP STREET B" may have the same code even though these are distinct bus stops). A particular bus stop name could also refer to multiple bus stop codes. The majority of these cases consisted of two codes, one for each direction of the bus route. To standardize the process of filtering by bus stop, we chose to specify only the bus stop name and used the following protocol, which relies on a mapping from bus stop name to code based on their co-ocurrences in the data. When searching for bus stop name "name1," identify the codes "codeA", "codeB", "codeC" that "name1" maps to. For each of these codes, determine the names it maps to, and for each of these names decide if it refers to the same physical bus stop as "name1." When comparing "name2" to "name1," the names were deemed as the same bus stop unless it was an "at" vs. "opposite" case. In nearly all other situations, two names that are mapped to by the same code refer to the same stop.

This reformatting of the data allows for efficient and simple identification of exactly which files contain the data of interest for a particular query, which can specify bus stops, years, seasons, months, and days of the week.

C. Sparsity and Aggregation

In order to make meaningful statistical analyses from the distribution of bus dwell times, a minimum number of observations should be made for a given interval of time. To this end, we decided to aggregate data over the day of the week for a particular time range of interest (e.g. the first 15 minutes of morning rush hour on Friday). This choice was motivated by the idea that bus dwell times depends on passengers' schedules, which are often dictated by the day of the week. Data analysis generally focused on particular seasons, which are defined in Table V.

For example, Figure 16 shows a heatmap of data counts for the inbound bus stop Centre Avenue at Aiken Avenue. Each segment of the heatmap displays the number of observations made for a particular 15 minute interval, for a particular day of the week during 2014, aggregated over the entire season. These counts indicate that aggregating over days of the week

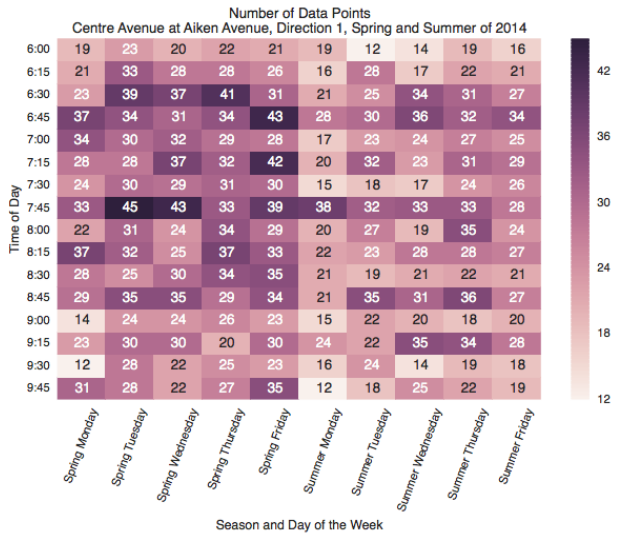


Fig. 16: Number of data points for each 15-minute interval at Centre Avenue at Aiken Avenue in the morning rush hour (6:00-9:59 am), aggregated by the day of the week for spring and summer of 2014.

yields a number of observations that will suffice for statistical analyses.

D. Short Dwell Times

The dataset consists of an observation for a bus passing a bus stop, whether it dwelled or not. [4] provides the proportion of zero to nonzero dwell times for each bus stop, showing that there is large variability in the popularity of bus stops. More popular bus stops, such as those in the connected-vehicle corridor, proportionally have many more nonzero dwells than less popular stops, such as those in the uptown area. Thus, we divided the overall problem into two subproblems: predicting whether the bus will stop, and predicting the dwell time given that the bus will stop.

Figure 17 show the distribution of dwell time counts for two stops in the corridor: Centre Avenue at Morewood Avenue and Negley Avenue at Centre Avenue. The distribution for Negley Avenue at Centre Avenue aligns with our initial expectation that the most common dwell time is short (e.g. 6 seconds) and that longer and shorter dwells times are less common. However, the large frequency of 1-second dwell times was unexpected. Moreover, the distribution for Centre Avenue at Morewood Avenue is surprising because it suggests that the shorter the dwell time, the more commonly it occurs. Even if the majority of stops involve one passenger onboarding or alighting, it seems unlikely that one second is enough time for this individual to pass through the doors. The distribution of dwell times could be a result of error in the automatic passenger counting (APC) sensors and the dwell time sensors, which would imply that nonzero dwell times are not necessarily legitimate bus stops and zero dwell times are not necessarily a bus simply passing the stop. To explore this possibility, we investigated the relation between short dwell times (0-5 seconds) and the number of onboardings

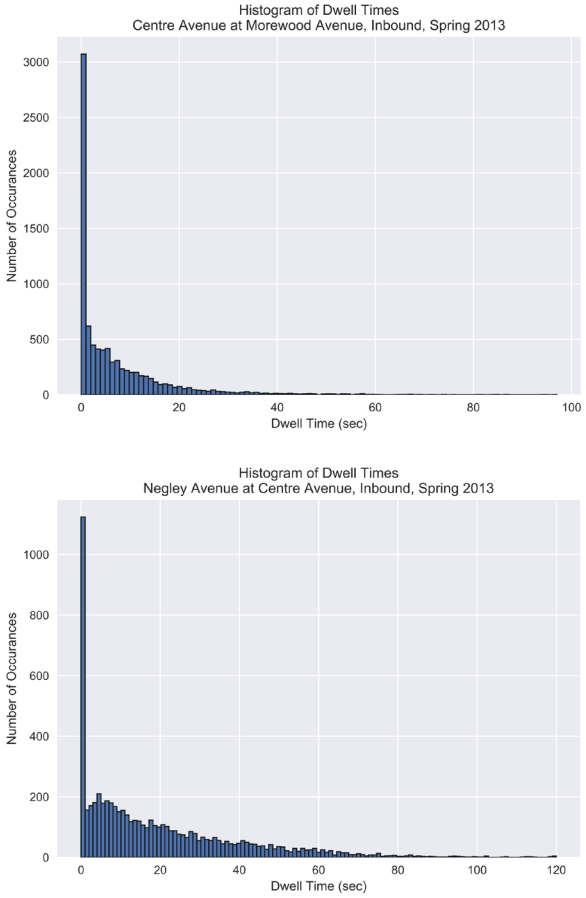


Fig. 17: Distribution of dwell times for bus stops Centre Avenue at Morewood Avenue (upper) and Negley Avenue at Centre Avenue (lower).

and alightings. Figure 18 display the proportions of zero-on-zero-off (zero onboardings and alightings), nonzero-on-zero-off (nonzero onboardings, zero alightings), zero-on-nonzero-off (zero onboardings, nonzero alightings), and nonzero-on-nonzero-off (nonzero onboardings and alightings) for each dwell time between 0 and 5 seconds at the Centre Avenue at Graham Street stop, in both directions. As expected, the zero dwell times are majority zero-on-zero-off, with a small error in the direction 1 plot. If the dwell time sensor erroneously recorded a zero dwell time as nonzero, a larger proportion of the nonzero dwell times would be zero-on-zero-off. The same behavior would be expected due to an APC sensor error that incorrectly recorded nonzero onboardings and alightings. If the dwell time sensor instead recorded a nonzero dwell time as a zero dwell time, a larger proportion of the zero dwell times would indicate people onboarding and/or alighting because a large majority of nonzero dwell times involve passengers getting on or off the bus. Since the data does not show these expected proportions, it is unlikely that the zero and nonzero dwell times are being incorrectly separated due to one sensor providing inaccurate readings. It is possible that both the dwell time and APC sensors had errors for the

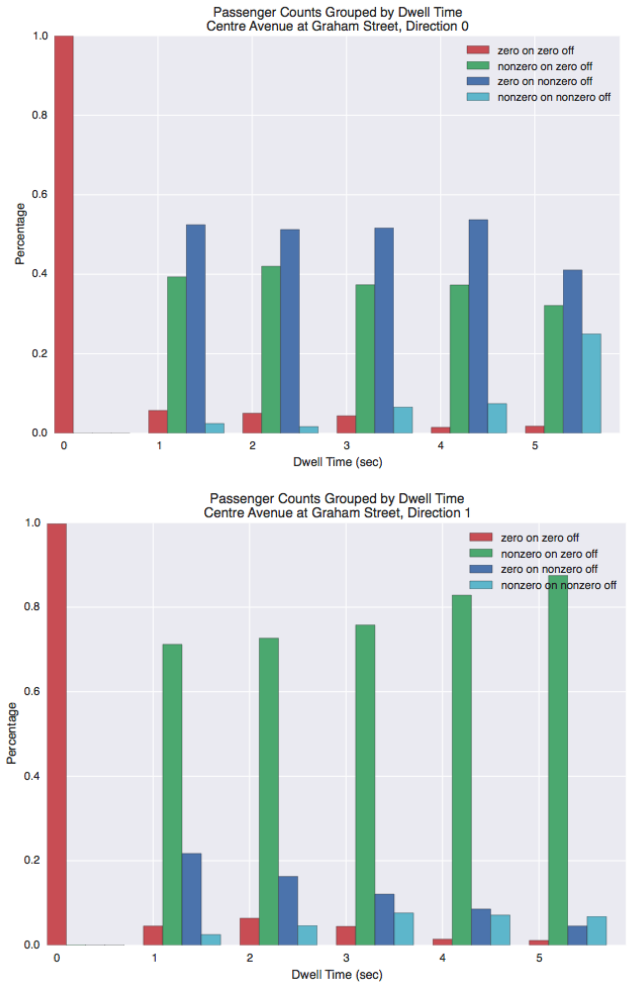


Fig. 18: Percentage of passengers grouped into four cases: zero onboarding and zero alighting, nonzero onboarding and zero alighting, zero onboarding and nonzero alighting, and nonzero onboarding and nonzero alighting. ”On” refers to onboarding and ”off” refers to alighting. The data is for the Centre Avenue at Graham Street bus stop, directions 0 (upper) and 1 (lower).

same data point, causing some of the zero dwell times to be recorded as nonzero dwell times in the nonzero-on-zero-off, zero-on-nonzero-off, or nonzero-on-nonzero-off categories, or vice versa. However, the probably of this occurring is low because the errors of the two sensors are independent.

Another explanation for the larger frequency of the shortest dwell times is rounding of the dwell time sensor measurements, which do not include milliseconds. Thus, a dwell time of 1.4 seconds could be rounded to 1 second, or in a worse scenario, a dwell time of 1.9 seconds could be truncated to 1 second. This lack of precision could explain the unexpected dwell time distributions. Regardless of the source of the error, whether it is noisy sensors or lack of precision, the frequency and magnitude of the errors would be small and only relevant for very short dwell times, which would not have an effect on control decisions made by a traffic signal controller.

Adaptive sky mapping for High-throughput plant phenotyping: Light Interception measurement

Anjana K N, Dr. Stephen Nuske, Robotics Institute, Carnegie Mellon University

***Abstract*—Light Intercepted by a plant is what decides it's productivity. We propose a high-throughput plant phenotyping system that accomplishes Light interception (LI) estimation of *Sorghum bicolor* using an industrial grade monocular camera attached with a fisheye lens. Our algorithm adapts to the changing sky and background conditions which allows the system to image at any time of the day. We were able to image a field with around 0.2 million plants belonging to 3000 different species. Comparison with harvest data shows that there is a positive correlation between leaf weight at harvest and LI.**

***Index Terms*—Computer Vision, Leaf Area Index, Light interception, Phenotyping, Segmentation**

I. Introduction

Plant phenotyping(process of identifying and measuring the physical characteristics of plant) has long been a bottleneck in advancing crop yields. The need for rapid improvement in plant performance is intensified by the increasing demand and depleting arable land and water availability and hence improved efficiency and accuracy in plant phenotyping is a critical factor. The process of phenotyping with conventional approaches is very time-intensive and predominantly an imprecise, manual process [1]. They also require an additional effort to estimate field plant density and optimal sample size. Hence, these direct methods can only be applied over small areas and for a limited number of measurements during the crop cycle [2].

We propose a high-throughput plant phenotyping technique that uses sensor networks and automated computer algorithms to extract phenotypic traits for large genetic mapping populations using nondestructive and noninvasive sampling method. With the availability of advanced DNA sequencing technologies, modern techniques of crop improvement which rely both on quantification of plant traits and DNA sequencing to identify genes and breeds of interest, could be dramatically sped up with the proposed technique. Current Light Interception measuring techniques have been proven to work efficiently with minimum human intervention only for plants grown in growth chambers or greenhouse [3].

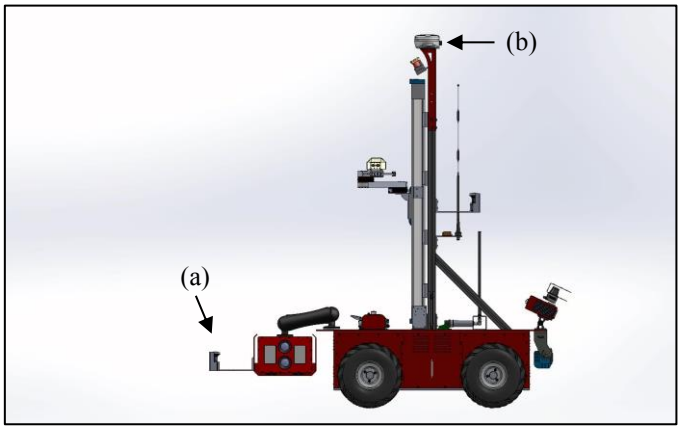


Fig. 1. Side view of the Robotanist (a) Upward facing camera (b) GPS device

We developed a system that measures Canopy density based Light Interception capacity of Sorghum Bicolor, a biofuel, by imaging thousands of plants with minimal time and effort. This system is robust to noisy backgrounds, thus can be used to collect data anytime of the day as opposed to the state-of-the-art image based phenotyping techniques which need good illumination conditions.

The rest of this paper is organized as follows: Section II describes related work on this topic and the current state of the art. Section III describes the hardware platform used. Section IV explains the challenges and our approach. Results and validation are presented in Section V, Conclusion and Future work is discussed in Section VI.

II. Related Work

Current approaches for plant phenotyping are laborious and is performed by highly skilled plant scientists, which is most of the time limited to indoor phenotyping, (i.e. measuring the plants in labs or greenhouse). Not each and every plant is measured and the results are usually approximated which brings

*This work was supported by the ARPA-E TERRA Program.

Anjana K N was with National Institute of Technology Karnataka, Surathkal, 575025, India .She is now with The Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh PA, USA. anjanak@andrew.cmu.edu.

Dr. Stephen Nuske is with The Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh PA, USA. nuske@andrew.cmu.edu

down the accuracy and thus cannot be used in breed selection and yield estimation.

Whereas, outdoor phenotyping not only offers researchers the opportunity to study a greater volume of crops, but also enables them to study plant traits that are difficult to test indoors, such as disease, pest resistance leaf area index and light interception. Plants phenotypes also tend to differ significantly when planted outdoors, and it is therefore necessary to grow them under field conditions in order to accurately determine important commercial measures such as crop yield. State-of-the-art outdoor phenotyping systems consist of sensor suites attached to gantries, drones or over-row tractor. These systems can only view phenotypes visible from the top of the canopy, neglecting information below and within the vegetative layers.

A. Plant Canopy Analyser (LAI-2200)

LI-COR LAI-2000 (LI-COR, Inc., Nebraska, USA) is one of the most widely used optical instruments for in situ LAI estimation. Leaf Area Index (LAI) is the ratio of foliage area to ground area. The LAI-2200 computes LAI from measurements made above the canopy and below the canopy, which are used to determine canopy light interception at 5 angles which is then fit to a well-established models to compute leaf area index and canopy gap fraction. This is achieved with the assumption that foliage absorbs all the radiation in the blue waveband seen by the sensor which is not true in direct sunlight, as reflectance off foliage causes a much greater overestimation of the gap fraction and underestimation of leaf area index.

This approach also requires inputs like Solar position, sky radiation properties, and sky brightness distribution for light scattering corrections, making it unusable under cloudy conditions[4].Though it requires no calibration, it can measure small plots and isolated plants and is not suitable for large scale field crops[5].

B. 3D based phenotyping methods

Correspondence-based triangulation methods, silhouette-based carving, time-of-flight cameras, light detection and ranging laser scanning are the currently used 3D strategies for phenotyping [4]. Plant motion during data acquisition is a major challenge for all 3D based measuring methods. Time delays due to scanning or sequential image acquisition lead to notable geometric distortions, especially for outdoor measurements with wind. The data then cannot be described by a static model and all current approaches doing so fail one way or another. They require multiple images surrounding the subject for a single plant, including a variety of angles from side on to above. Representations they produce may also be unsuitable for direct use in some situations.

III. Hardware Configuration

Robotanist, the hardware platform developed by Prof. George Kantor and his team which consists of a wheeled-skid steer, electrically powered ground vehicle that can travel at speed up to 2 m/s for more than 8 hours per charge was used for data collection.It was equipped with LiDAR, GPS, RGB Monocular and stereo cameras along with the computing power necessary to run the algorithms in real time. The system collected data from fields located in Clemson, SC and Florence, SC in July, August and September of 2016 demonstrating that the platform is capable of navigating and phenotyping a wide array of sorghum plants.

An Industrial grade monocular camera by Point Grey Technologies which can capture images at a frame rate of up to 20Hz was used. The fisheye lens was attached to the camera to get a wider view of the canopy and it was positioned in such a way that it could see plants from the row to its left and the row to its right as shown in Fig.2.

TABLE I
HARDWARE SPECIFICATIONS

Parts	Model	Features
Monocular camera	Point Grey Flea3 GigE	Opto-isolated GPIO; an on-camera frame buffer; non-volatile user data storage; new trigger modes 1000Mb/s Gigabit Ethernet
Fisheye lens	Fujinon FE185C05 7HA-1	185 degree wide angle lens; 1.8mm focal length; 5 megapixel C-Mount lens
GPS Device	Novatel smartt6-L GPS antenna	solar rechargeable battery Detects upto 9 satellites Gives GPS co-ordinates for every image with less than 1 meter accuracy

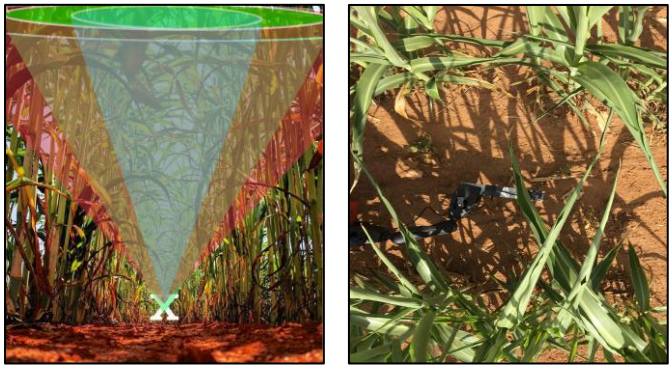


Fig. 2. Position of upward facing camera (a) Front view (X stands for camera location, conical shapes indicate different degrees of view from the camera center) (b) Top view

IV. Algorithm Overview

In order for the imaging system to be able to collect data at any time of the day, one of the challenges to be overcome is the effect of various natural light conditions such as sunny condition with a shadow, sunny with specular reflection and cloudy condition on crop images. The images differ a lot with respect to background sky conditions from row to row and from time to time. This often makes the segmentation of leaf from background parts rather difficult. Fig.3. gives an overview of

the algorithm. Algorithm involves collecting data from various sensors, syncing them with respect to their timestamps and processing them to extract useful data and storing them as csv files with timestamps and geotags for further analysis.

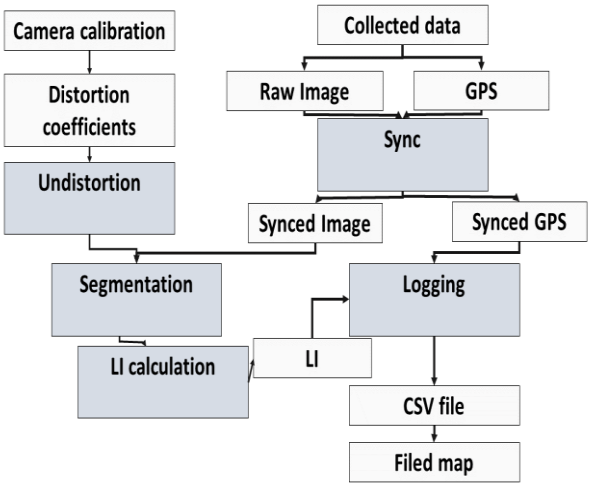


Fig. 3. Algorithm overview

A. *Camera Calibration*

The monocular camera along with fisheye lens was calibrated in order to produce an estimate of the extrinsic and intrinsic camera parameters. These parameters can be used to correct for lens distortion, measure the size of an object in world units, or determine the location of the camera in the scene and is used for rectification of radial distortion in images. It is accomplished by using standard computer vision algorithms included in the Open Computer Vision Library (OpenCV) [6] and OCam calib toolbox of MATLAB [7].

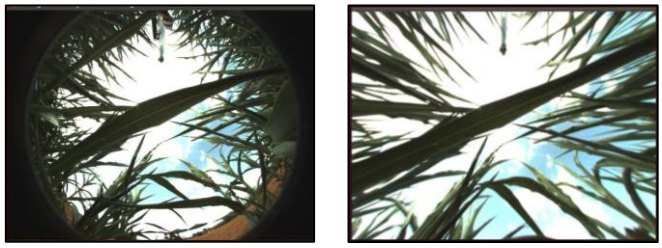


Fig. 4. Raw image and rectified image

B. *Adaptive sky mapping*

The algorithm presented here can be run both online and offline. Given the raw image with GPS data and timestamps, it takes about 0.5 seconds to process one frame (50-80 plants).

Segmentation is partitioning of image pixels into meaningful groups like foreground and background, in our case into leaf and non-leaf pixels, for further analysis.

A key task in the image-based phenotyping is the segmentation of all individual leaves in images. Varying image characteristics caused by differing levels of foliation and fluctuating lighting conditions present a unique challenge for consistent segmentation in order to reliably measure the transparency of plant canopy. Small leaves surrounded by bright sky, small openings among dense canopy and mixed pixels also resent difficulties for segmentation. Image is converted to Hue-Saturation-Value (HSV) space from RGB and then segmented to leaf and non-leaf as HSV is one of the many color spaces that separate color from intensity.

The novelty of this technique is that it learns the background and illumination conditions (i.e. bright, cloudy, and partially cloudy) for every row and segments the image into leaf and non-leaf pixels based on the background conditions. The way it is done is:

1) First image of every row of sorghum plant is picked up and converted to HSV space.
2) The HSV image is then shown to the user who selects the sky region by drawing a rectangle around it.
3) The pixels from inside the rectangles are stored in a map called Sky_map.
4) All the pixels from other images of that row are then compared with the Sky_map to see if the pixel belongs to sky or leaf.
5) To improve the performance of segmentation, a global threshold range of all the shades of cloud and sky are predefined. After comparing with the sky map and segmenting the pixels as sky, the remaining pixels are again checked with this predefined threshold range to see if they belong to any shades of sky or clouds.

This ensures that no sky is wrongly segmented as leaf pixel.

C. *LI Calculation*

As per Beer and Lambert's law, which relates the attenuation of light to the properties of the material through which the light is traveling, radiation intercepted by a canopy depends on the incident irradiance and the structure of the canopy. Light interception is therefore calculated as a function of Gap fraction, i.e. ratio of leaf pixels to total number of pixels. We calculated Gap fraction at three separate degrees of view with respect to the camera center as shown in Fig. 2(a).
Fig 5. shows raw image and the Light Interception calculated for the image.

Fig. 5. Raw image and processed image

D. *Data logging*

The huge amount of plant image data collected in the field are stored in network attached storage units like QNAPs at Carnegie Mellon University, USA. Image data is converted to from converted to csv files with time stamps, GPS coordinates and LI value. It is then used for comparing with other characteristics like Vegetation Index, Leaf Necrosis, Leaf area, Stalk width.

V. Results and Validation

The maps below show the Light Interception map of the field integrated with Google Earth and the LI maps at 90 and 120 days. We can see how plants at one end of the field have higher LI capacity as compared to the species on the other end.

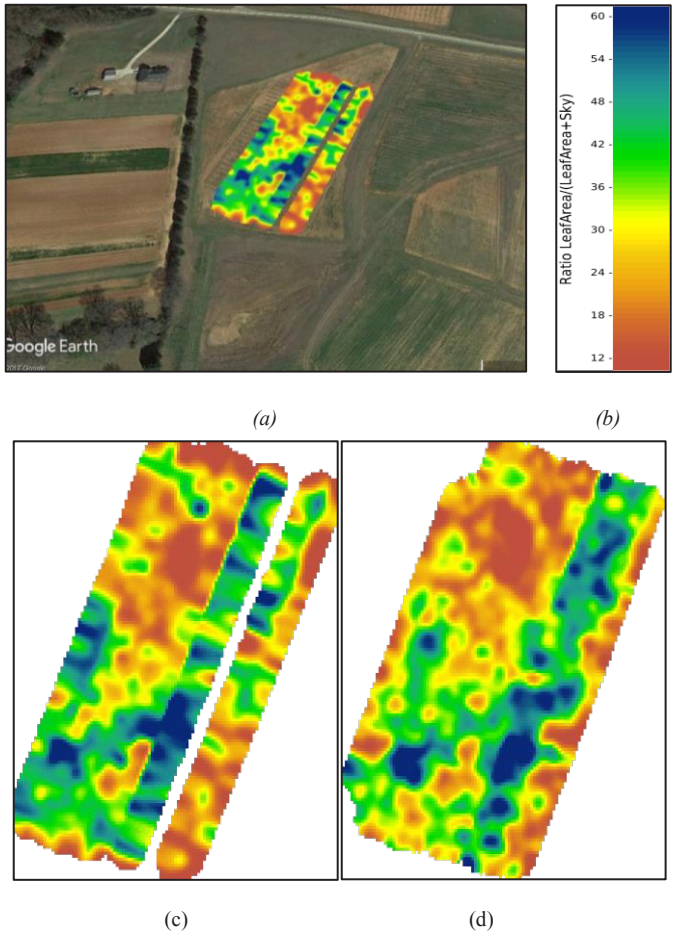


Fig. 6 (a). Light Interception at 60 days mapped using kml file (b) Color bar (c) LI at 90 days (d) LI at 120 days

Ground truth for sky mapping was done by manually segmenting the image pixels into Leaf and non-leaf which showed the algorithm output as 96% accurate. As there is no accepted ground truth/benchmark for Light Interception modeling, LI results were correlated with Leaf weight at harvest time. Fig.7. shows the correlation between calculated Light Interception and Harvest data (Total Fresh weight of the leaves at harvest). Light interception data was collected at 60, 90 ad 120 days after planting. Correlation of Light Interception at 90 and 120 days as shown in Fig. 8

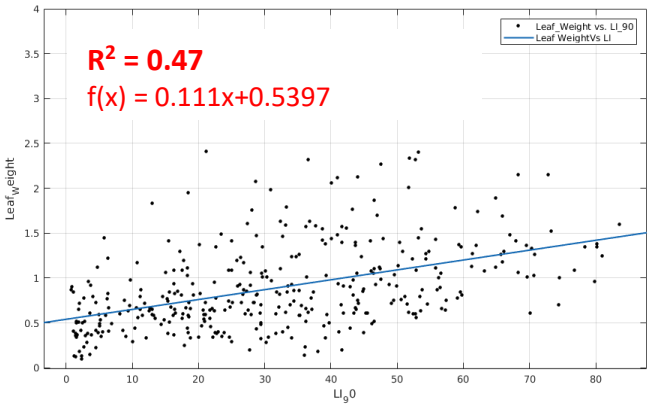


Fig. 7. Correlation map of Light Interception at 120 days (x-axis) with the Total leaf weight at harvest(y-axis)

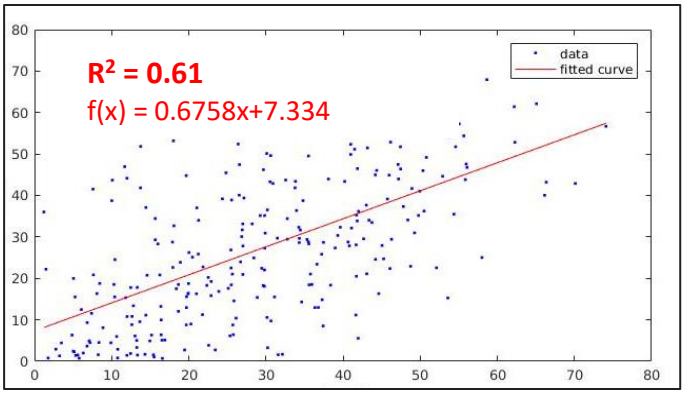


Fig. 8. Correlation of Light Interception at 90(y-axis) and 120 days (x-axis)

VI. Conclusion

The development and deployment of this novel system has yielded the following contributions for the field of agricultural robotics: A custom high-resolution, high through-put, lighting invariant imaging system for measuring Sorghum phenotypes from beneath the plant canopy and a light interception measurement algorithm that processes at 2 frames (80-100 plants) per second.

This work is faster than the present state-of-the-art plant phenotyping techniques listed in Section 2, and can be deployed a large scale crop field.

VII. Acknowledgement

This research was supported by the Advanced Research Projects Agency-Energy. Authors would like to thank the following people for assistance and helping in data collection: George Kantor, Omeed Mirbod, Jay Patravali.

REFERENCES

[1] Duursma, R. A., Falster, D. S., Valladares, F., Sterck, F. J., Pearcy, R. W., Lusk, C. H., Sendall, K. M., Nordenstahl, M., Houter, N. C., Atwell, B. J., Kelly, N., Kelly, J. W. G., Liberloo, M., Tissue, D. T., Medlyn, B. E. and Ellsworth, D. S. (2012), Light interception efficiency explained by two simple variables: a test using a diversity of small- to medium-sized woody plants. New Phytologist, 193: 397–408. doi:10.1111/j.1469-8137.2011.03943.

[2] S.Garrgues, N.V Shabano, K Swanson, J.T Morisette, F>Baret, R.B>Myneni "Intercomparison and sensitivity analysis of Leaf Area Index retrievals from LAI-2000, AccuPAR, and digital hemispherical photography over croplands"

[3] A review of Imaging techniques for plant phenotyping Sensors 2014,14,20078-20111;doi:10.3390/s141120078

[4] DanielaStroppianaa, MircoBoschettia, RobertoConfalonieric, StefanoBocchib. , Pietro AlessandroBrivioa "Evaluation of LAI-2000 for leaf area index monitoring in paddy rice"

[5] Michael P. Pound, Andrew P. French, Erik H. Murchie, Tony P. Pridmore ,"Automated Recovery of Three-Dimensional Models of Plant Shoots from Multiple Color Images".

[6] Opencv.org. (2015). ABOUT | OpenCV. [online] Available at: http://opencv.org/about.html [Accessed 25 Mar. 2015].

[7] Scaramuzza, D., Martinelli, A. and Siegwart, R., (2006). "A Flexible Technique for Accurate Omnidirectional Camera Calibration and Structure from Motion", Proceedings of IEEE International Conference of Vision Systems (ICVS'06), New York, January 5-7, 2006.

A Particle Filter Approach to Tracking Multiple Vehicles Over Time

Roman Kaufman
Carnegie Institute of Technology
Carnegie Mellon University
Pittsburgh, Pennsylvania
Email: rkaufman@andrew.cmu.edu

***Abstract*—Attempting to identify and track objects in real time across frames is a well-known problem with many different approaches. This paper assumes a working segmentation algorithm that can determine the objects heading and location in a given frame. Still, given the heading and location of seen objects frame to frame isn't all that needs to be done, especially for occluded vehicles that one would like to remember. This paper attempts to track and 'remember' target vehicles by using a particle filter approach (as initialized in the first frame). The particles are sampled from a three dimensional gaussian distribution, with the dimensions being x position, y position, and heading of the given object. The association step is completed by a prediction step and comparison of the proximity of each target's predicted location with the given frame's objects.**

Some basic examples of the proposed algorithm's function are reviewed, and weaknesses of the algorithm and future improvements are considered.

Keywords*—*Particle Filter, Multiple-Object Tracking, Data-Association

I. Introduction

Within the field of autonomous vehicle control, tracking multiple objects is an integral problem. The difference between safety and danger for a passenger often lies not only on how much information a vehicle can receive from its sensors, but also on its ability to process the real-world story that information tells. Thus, reliably determining, from the sensor data given to a vehicle (in this case 2D lidar data, or x-y coordinates coupled with a heading angle) the location, speed, and heading (or orientation) of the vehicles around the user vehicle is of paramount importance.

Following and identifying objects over time is a problem that is generally divided into two steps. First, one must be able to determine, at a point in time, certain characteristics of the objects one wishes to track. This stage is called segmentation. Second, when given the targets one has been tracking for the time up to a given point, one must know when to associate a time step's object with a target, when to initialize a new target, and when to leave a target un-associated.

This paper deals with the second stage, and assumes the segmentation step has been solved and can be used as a black box function for this algorithm. Though there are many approaches to this problem [1], this paper uses a particle filter for each vehicle.

In II, we give a high level overview of how a particle filter can be used when attempting to track multiple objects over time, and some of the main obstacles to overcome. In III, the proposed algorithm is reviewed in detail and some examples of its use are shown. Finally, in IV, future developments are reviewed.

II. Particle Filters Used for Tracking

This section reviews how particle filters generally work for single target detection and the additional complexity introduced with using them for multiple object detection.

When tracking a single object, a particle filter functions with N particles sampled over a normal distribution, so that there are more particles closer to the mean, where the mean is the initial state of the object. Thus, each particle represents a possible belief as to what the current state might be, with the mean representing the best guess.

Every frame/update step for the input data, the particle filter does three key steps in order to keep on tracking. Prediction, weight assignment, and then resampling. The prediction occurs by using the previous frame's state data. Then, for each particle in the newly predicted particle set, the probability that that particle ought to be associated with this frame's measurements is the weight given to the particle (so the particles closest to representing the current frame's object have higher weight). With the weights then normalized, the re-sampling step occurs, whereby the particles are picked with replacement, with the probability of picking particle i being equal to the weight of particle i. The resampled set then becomes our updated particle set, and we go to the next frame [2].

In general, particle filters can become inefficient when the state space has a high number of dimensions. However, in this paper there are only three properties, and thus dimensions, for each particle: x position, y position, and heading.

When considering the problem for multiple objects, the issue becomes how to associate a given frame's objects with the targets being tracked. Several approaches [2], [3] use Joint Probabilistic Data Association Filters for this step; with further explanations offered in both.

For this paper, however, the association step is completed by a simple assignment heuristic that matched a target to an object if the distance between them summed with their differences in heading were below some threshold, otherwise a new object as initialized.

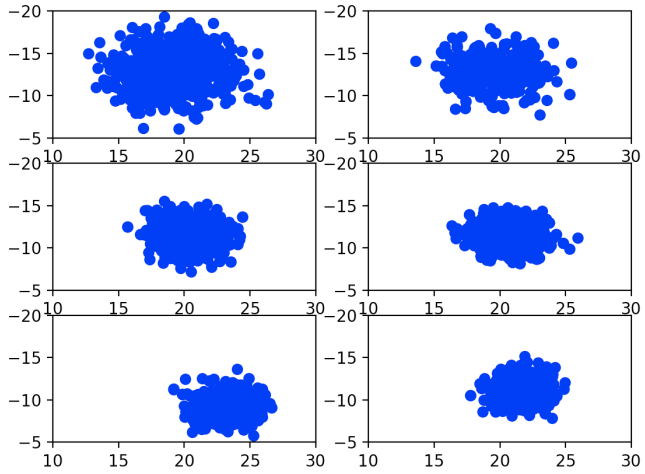


Fig. 1. A single object being tracked - note how, as the frames progress, the particle cloud converges.

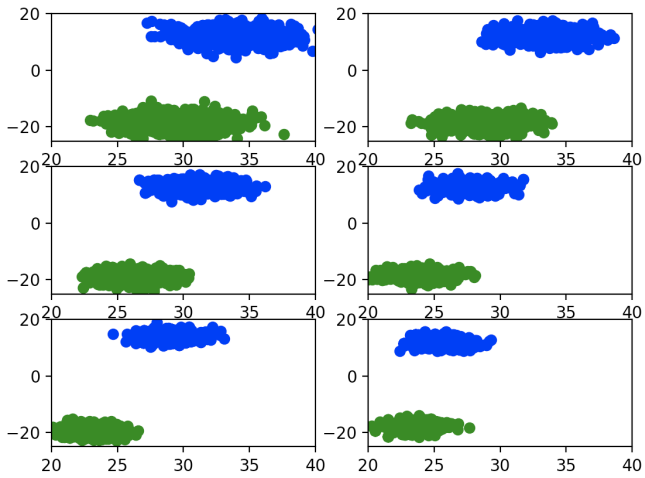


Fig. 2. Demonstrates proposed algorithm when working with two vehicles moving left - note the elongation of particles is due to the uneven scaling of x and y axes.

III. Method

In this approach, the following algorithm was used:

1: **for all** targets, t_i **do**
2: **for** object, o_j in current frame objects **do**
3: $k \leftarrow 0$
4: **while** $k <$ number of particles **do**
5: Weight for kth particle, $w_k \leftarrow$ p(o_j came from p_k)
6: **end while**
7: Normalize weights, so they sum to one
8: Redraw particles from t_i's particle set by their calculated weight, with replacement
9: Calculate covariance matrix and arithmetic mean of newly drawn particles
10: **if** distance(arithmetic mean of redrawn particle set, arithmetic mean of t_i's initial particle set) $<$ closestSeenDistance **then**
11: closestSeenDistance $\leftarrow$ distance above
12: bestCandidate $\leftarrow o_j$
13: **end if**
14: **end for**
15: **end for**
16: remainingObjects $\leftarrow$ current frame objects
17: **for** target, t_i in sorted order of closest bestCandidates **do**
18: **if** t_i's bestCandidate is in remainingObjects **then**
19: assign t_i to bestCandidate object
20: **else if** t_i's bestCandidate is not in remainingObjects **and** remainingObjects is not empty **then**
21: assign t_i to best candidate remaining, if candidate close enough in orientation and position
22: **else**
23: toggle t_i as no longer identified
24: **end if**
25: **end for**
26: **for all** r_i left in remainingObjects **do**
27: initialize new target from r_i
28: **end for**
29: **for all** t_i in targets **do**
30: Update t_i's particle set to expected values for next frame
31: **end for**

Throughout the above algorithm, the x position, y position, and heading of a given target vehicle were assumed to be independent variables - so all covariance matrices are diagonal.

On line 5, the weight assigned to particle p_k was the probability density function of a multivariate distribution,

$$w_k = \frac{1}{\sqrt{(2\pi)^k \det \Sigma}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right) \quad (1)$$

Where k, the dimensions that the input, x, takes, was 3; Σ, the covariance matrix, was diagonal accounting for a standard deviation of 3 meters in both x and y position and 5 degrees in heading; μ, the mean, was taken as the particle- p_k's - x, y position and heading; and x, the argument, was taken as the x position, y position, and heading of the object in question, o_j.

In actually testing this algorithm several incrementally more difficult scenarios were chosen, all of which came from real segmented data taken in the Pittsburgh area. The first example in which interesting bugs arose can be seen in figure 3, where the blue and green targets are jumping great distances in a single frame (each frame is taken every 15 milliseconds). This hopping illustrates a key weakness of the assignment process chosen - a single incorrect assignment can throw off the calculated velocity of a given target and force its next predicted step to be well off target - literally.

IV. Conclusion

In terms of future optimizations, it would be interesting to pursue how doing away with the assumption of independence for x position, y position, and heading would alter the results (if at all). However, actually determining the variance that ought to exist between x-y coordinates and heading would be a problem in itself.

Additionally, a future work could compare how a linear assignment approach for data association (like the hungarian algorithm) might compare to the JPDAF approach (as mentioned earlier and used in [2], [3]).

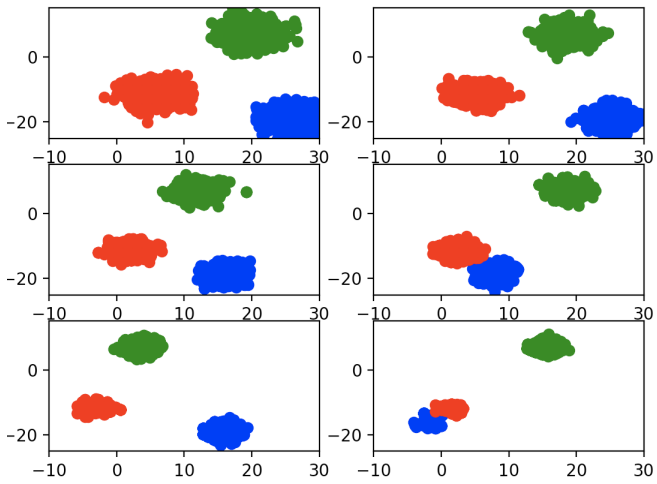


Fig. 3. Three vehicles being tracked - note the back-and-forth motion of the blue and green targets in frames 4 through 6.

ACKNOWLEDGMENT

The author would like to thank Chiyu Dong and Dr. John Dolan from the Autonomous Driving Collaborative Research Lab, for their priceless guidance and help in introducing the relevant material to the author.

REFERENCES

[1] L. Fan, Z. Wang, B. Cail, C. Tao, Z. Zhang, Y. Wang, S. Li, F. Huang, S. Fu, and F. Zhang, "A survey on multiple object tracking algorithm," in *Information and Automation (ICIA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1855–1862.

[2] A. Almeida, J. Almeida, and R. Araújo, "Real-time tracking of moving objects using particle filters," in *Industrial Electronics, 2005. ISIE 2005. Proceedings of the IEEE International Symposium on*, vol. 4. IEEE, 2005, pp. 1327–1332.

[3] M. Jaward, L. Mihaylova, N. Canagarajah, and D. Bull, "Multiple object tracking using particle filters," in *Aerospace Conference, 2006 IEEE*. IEEE, 2006, pp. 8–pp.

Wheelchair Pose Estimation Using Monocular Images

Kevin Brian Kwan Chong-Loo, *ITESM, Monterrey*

***Abstract*—About 3.3 million people in the United States use wheelchairs; automating services for them can help to improve the quality of life. This can be achieved in many ways, such as allowing the right amount of time where an automatic door is opened in a regular hallway or elevator, or increasing their accessibility to public transportation, hospitals, and even crosswalks. In these applications, it is essential to determine the wheelchair's location and orientation. Work to date has produced algorithms which are capable of detecting the area of an image where a wheelchair is present. However, determining the exact position of the wheelchair in the world plane still remains an issue. In this paper, we describe an approach that focuses on detecting the exact points where a wheelchair makes contact with the ground, which under certain assumptions allows estimating its pose. These points are obtained by detecting the wheels through fitting ellipses in its Canny edge counterpart. The method proposed focuses on the basic structure of a wheelchair, and includes a contour fragmentation step that improves the ability to fit the ellipses that represent wheels. Furthermore, it does not require a training data set, it can work for various types of wheelchairs, and it is easy to replicate.**

***Keywords*—Wheelchair, Pose Estimation, Fragmentation of Contours, Fit Ellipse Algorithm.**

I. Introduction

Detecting wheelchairs and where they are in the world plane is a difficult task. The wheels can be of different styles and sizes; they come in various colors; and their appearance changes significantly depending on whether a person is seated on them or not. Machine learning approaches have been able to detect a region where a wheelchair is most likely present. However, to detect the exact location of the wheelchair in the world plane, very specific points where the wheelchair touches the ground have to be identified. Unlike stereo pairs, monocular cameras cannot estimate depth directly, therefore they have to infer it by using prior knowledge of the scene. To estimate depth more accurately, it is assumed that the wheelchairs are moving on a planar surface, and that the extrinsic calibration of the camera is available. Then, the distance from the camera can be estimated by calculating the intersection of rays coming from the camera with the ground plane. In this study, the use of monocular cameras is desirable since it allows the deployment of less expensive systems.

A. Related Work

There have been previous approaches related to pose estimation of objects. In [1], they are able to estimate the pose of an object in monocular images even if the object is occluded. Similar work is presented in [2] where they only use a single RGB image to estimate the 6D pose of objects. Nevertheless, for both mechanisms, the type of object must be previously known and its parts must be labeled, thus generating a model to compare it to what is obtained from an image. In the case of a wheelchair, the target object is not always the same and it can differ depending on the type and color of the wheelchair.

The idea of using wheels to determine the pose of an object is presented in [3] where they demonstrate a method to estimate the pose of cars by focusing on the rims. By identifying the highest and lowest point of the ellipses formed by the edges of the rims, they are able to approximate the pose of the car. However, it is much simpler for an edge detector to work on the wheels of a car than of a wheelchair. On the one hand, in most cases the rims of the car are not occluded and are usually of a different color than the tire, which makes it easier for the edge filter to isolate the contour of the rim. On the other hand, the edges of wheels on the wheelchair can be connected to ground edges or other parts of the wheelchair, which makes it difficult to isolate and detect the edges as elliptical figures. This particular issue is specifically addressed by the approach proposed here.

Although object pose estimation has been the focus of much research work, only a few approaches have addressed the problems presented by wheelchairs. In particular, the work described in [4] considers an approach for wheelchair detection using HOG, SVMs, and Cascaded Decision Trees. In this research, the authors were able to detect the wheelchairs and its pose with more than 90 percent accuracy. A training set of around 4,000 labeled images of wheelchairs was used in this case. They also presented an estimation of the orientation of the wheelchair in 8 different directions (front, backward, left, right, left-front, right-front, left-backward, right-backward). Nonetheless, this approach is not able to determine the exact location of the wheelchair in the world plane since it is not able to estimate the exact points where the object makes contact with the ground.

The closest work related to our method is [5], where the wheels of the wheelchair are pinpointed using edges obtained from a Canny edge detector. They are able to estimate the location of a wheelchair in an image by fitting ellipses to the wheels and looking for the person's arm and head. However, their approach needed a previously calibrated environment which required prior knowledge of the size of the wheel, the color of the background, and the wheels could not be occluded. It was only tested on pose facing the left or right where the wheel had a circular shape and not an ellipse.

In our approach, the location and orientation of the wheelchair are determined by considering the basic geometry of how a wheelchair is structured and its key points i.e. the

wheels, since they are the objects that most likely contain the points where it makes contact with the ground. They are detected by applying a fitting ellipse algorithm to each of the edges that form a contour. The fitting ellipse algorithm is able to detect ellipses by approximating a model of an ellipse to a set of continuous points. Nevertheless, the consecutive points tend to be combined among each other (Figure 1). In other words, the wheel's contour is merged with contours belonging to another feature of the target object. Consequently, the estimated ellipse would try to be fitted to both the wheel contour and the contour of another part of the wheelchair. Our main contribution is to present a mechanism that is able to give the an ellipse fitting algorithm a better chance of detecting the wheels correctly.

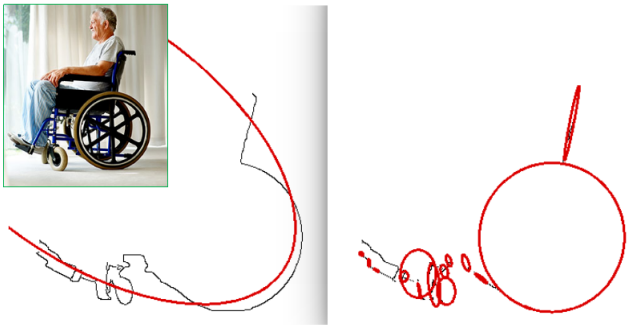


Figure 1: These images show the fit ellipse algorithm proposed by Fitzgibbon [6] for a set of points. The small image on the top left corner is the original image. The left image shows an ellipse estimated by the whole contour given by a Canny edge filter. The right image shows the implementation of the developed algorithm where the contour is fragmented, and ellipses are estimated to each segment.

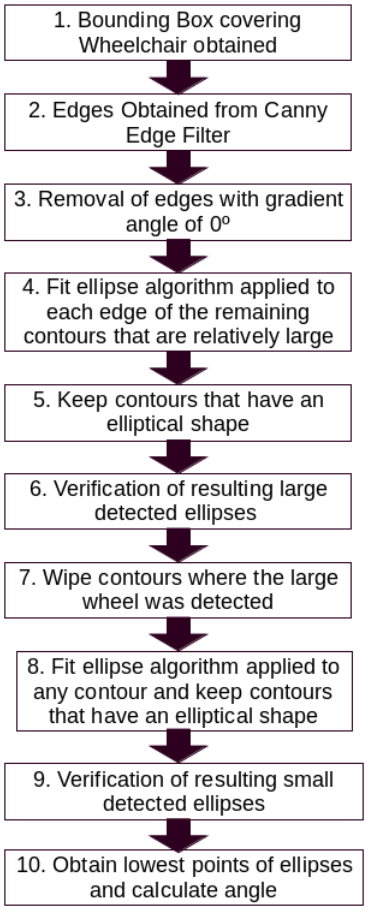


Figure 2: Algorithm flowchart.

II. Methodology

The algorithm we developed for pose estimation focuses on the structure of the wheelchair and specifically its wheels. To determine the location and orientation of the target, the points where the object touches the ground have to be detected. In other words, these spots, are most likely the lowest points in each wheel. This algorithm works for the basic structure of a wheelchair composed by two large and two small wheels. The application of this method assumes that the wheelchair is standing on a planar surface. Our algorithm has the steps presented in Figure 1.

Step 1 is receiving the input to the algorithm, which is a bounding box that encloses the detected wheelchair which can be obtained from a visual recognition technique e.g. a deep learning algorithm. In Step 2, to blur the image, a Gaussian filter is applied to the region that encloses the wheelchair. The grayscale version of the blurred image is generated and then the edges are identified through a Canny filter. Subsequently, the contours of the wheelchair are obtained by merging the edges that are adjacent to each other.

As seen in Figure 1, the contours belonging to the wheels are not completely isolated, which means that the fit ellipse algorithm won't find the appropriate ellipse that represents the wheel. Instead, it will try to approximate an ellipse with contours of a wheel and of some other elements that are part of the ground or any other section of the wheelchair. Therefore, Step 3 was implemented, where the gradient angle was acquired for each of the contours using Sobel Filters.

$$\mathbf{G_x} = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \begin{bmatrix} A_{i-1,i-1} & A_{i,i-1} & A_{i+1,i-1} \\ A_{i-1,i} & A_{i,i} & A_{i+1,i} \\ A_{i-1,i+1} & A_{i,i+1} & A_{i+1,i+1} \end{bmatrix}$$

$$\mathbf{G_y} = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \begin{bmatrix} A_{i-1,i-1} & A_{i,i-1} & A_{i+1,i-1} \\ A_{i-1,i} & A_{i,i} & A_{i+1,i} \\ A_{i-1,i+1} & A_{i,i+1} & A_{i+1,i+1} \end{bmatrix}$$

$$\theta_{\mathbf{G}} = arctan(\frac{\mathbf{G_y}}{\mathbf{G_x}}) \quad (1)$$

The previous equations show the Sobel operators or kernels that were used to obtain the gradient angle of each of the

edges. This is achieved by doing a convolution between each kernel and the image. By examining at the gradient angles in the contours, it was deduced that the contours of the wheels were combined most likely with other contours in the edges with 0^o gradient angle. The edge segments with this angle are the ones that resemble straight horizontal lines. In most cases, wheel contours tend to intersect with contours from the ground, thus the ellipse approximation algorithm would try to be fitted to both ground and wheel edges. Therefore, the contours were split exactly at the points that had 0^o of gradient angle. This facilitated the isolation of round contours that in the end would reassemble the structure of the wheel by applying the fit ellipse algorithm which is done in Step 4. The implementation of this method gave better results in estimating ellipses that actually belonged to only the wheels. The fragmentation of contours at 0^o can be seen seen in Figure 3.

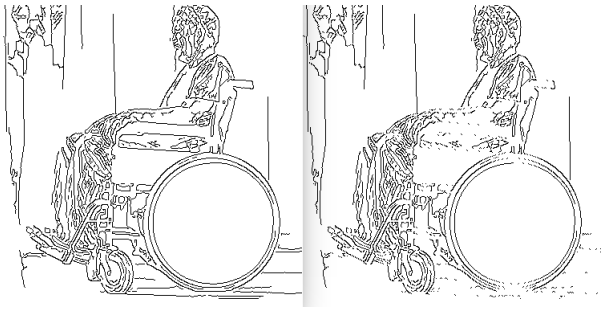


Figure 3: These images show the application of the 3rd step in the algorithm. The left image shows all the contours and the right image shows the cropped contours.

Nevertheless, the fit ellipse algorithm is not able to determine if the contour belongs to an elliptical shape: it only tries to approximate an ellipse that best matches the given points. Therefore in Step 5, a verification is made by checking if the points that form the estimated ellipse match with the points of the contour, as seen in Figure 4.

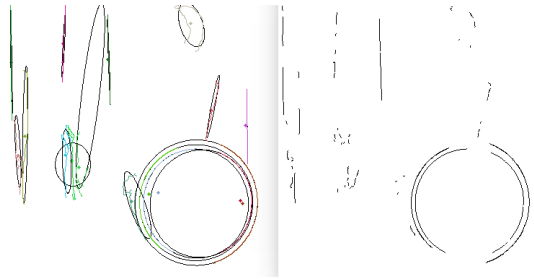


Figure 4: The 4th and 5th step in the algorithm are presented in these two images. On the left image, each contour has its own estimated ellipse based on its points. And on the right side, the remaining points are the ones of contours that matched with the points of its approximated ellipse.

In Step 6, the remaining contours are the ones that best resemble an elliptical region of the image. In this case, the big wheel is searched for first because it's the easiest one to detect and it covers a larger part of the image. In this way, the biggest ellipse corresponds to the big wheel. In this same step, the remaining contours are fitted to an ellipse again. Fitzgibbon [6] proposes a fitting ellipse algorithm that acquires the parameters of the ellipse which include width, size, center, and rotation angle. These values can be used to discard the estimated ellipses if they are small or resemble straight lines. Also, ellipses that tend to have close centers are merged. This evaluation can be seen in Figure 5.

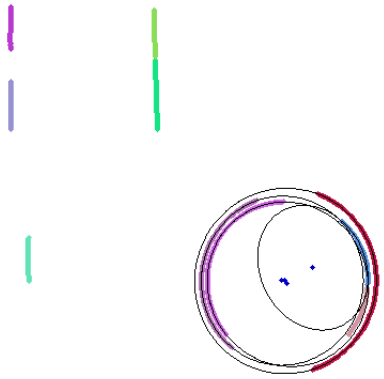


Figure 5: The 6th step is presented in this image where the ellipses are drawn in contours that are large enough proportionally to the image and are not straight lines. Remaining ellipses are combined by keeping the one that covers other ellipses and if their centers are close to each other.

After detecting the big wheel successfully, in Step 7, the contours near the big wheel are cleared so that the search is focused on a small wheel following a similar approach in Step 8. However, in Step 9, the small wheel is identified by looking for the lowest ellipse found which is smaller than the big wheel detected before.

Subsequently, having successfully detected both a large and a small wheel, the lowest points of each wheel are determined by using their respectively fitted ellipses. In Step 10, with those two points, the angle can be determined by computing the inverse tangent which will give an estimated angle corresponding to the orientation of the wheelchair, as seen in Figure 6.

III. Results

Figure 7 shows a few sample results of the implemented algorithm, which was tested using 356 images of wheelchairs from ImageNet [7]. The test images selected had the following characteristics:

- The wheelchairs are varied in size, color and shape. However the structure of having a large and a small wheel is common among all the images.
- A bounding box area is given beforehand that encloses the wheelchair.

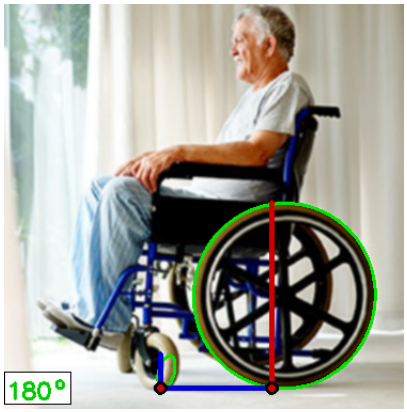


Figure 6: This is the result of the implemented algorithm where one big and one small wheel are detected and the orientation is estimated. Also, the ground points are marked at the bottom of the detected wheels.

Figure 7: Other test cases where pose estimation is obtained.

- At least one large and one small wheel are visible.
- Images of wheelchairs not included were the ones facing toward or away from the camera, where the wheelchairs don't seem to have an elliptical shape.

Table I shows the results of the test implemented. Table II shows the results of detecting wheels without splitting contours at 0°. The success rate refers to the frequency of each of the scenarios in the set of images. The algorithm implemented estimated the ground points by looking at the lowest points of the fitted ellipses. The error for each wheel's ground point is the Euclidean distance between the pixel location estimated by our algorithm with the ground truth. The ground truth was obtained by manually pinpointing the ground points of wheels in the wheelchair. The error in the angle follows the same calculation where it is the error between the angle obtained in the ground truth and the implemented algorithm.

For case 1, two pixel points in the image are classified as the location where the wheelchair touches the ground and both

Case	Description	Success Rate	Big Wheel Error (Pixels)		Small Wheel Error (Pixels)		Angle Error (Degrees)	
			Mean	Std Dev	Mean	Std Dev	Mean	Std Dev
1	2 Same Side Ground Points	**28.93%**	12.01	8.21	11.03	9.88	4.17	3.95
2	2 Opposite Ground Points	**3.65%**	13.16	8.40	8.64	6.86	-	-
3	1 Big Wheel Ground Point	**39.05%**	15.32	11.19	-	-	-	-
4	1 Small Wheel Ground Point	**9.27%**	-	-	7.51	6.24	-	-
5	No Ground Points Detected	**19.10%**	-	-	-	-	-	-

Table I: Results with tested images by splitting contours at 0°.

Case	Description	Success Rate	Big Wheel Error (Pixels)		Small Wheel Error (Pixels)		Angle Error (Degrees)	
			Mean	Std Dev	Mean	Std Dev	Mean	Std Dev
1	2 Same Side Ground Points	**14.04%**	13.32	10.65	9.65	7.42	4.81	3.99
2	2 Opposite Ground Points	**1.69%**	13.06	6.26	4.33	3.89	-	-
3	1 Big Wheel Ground Point	**42.41%**	13.91	11.61	-	-	-	-
4	1 Small Wheel Ground Point	**8.43%**	-	-	6.08	3.78	-	-
5	No Ground Points Detected	**33.43%**	-	-	-	-	-	-

Table II: Results with tested images **without** splitting contours at 0°.

are from wheels on the same side. In this scenario, each point is identified as the point belonging to the large or small wheel. By comparing Table I and Table II, it is shown that the success rate for this case increases considerably by splitting contours at 0°. Case 2, detects a large and a small wheel along with its ground points but they are wheels that are on different sides (e.g. left large wheel and small right wheel or right large wheel and small left wheel). Even if the pose is not estimated correctly, two ground points are still obtained with an average error of about 13 pixels.

In case 3, only one ground point is detected correctly and it belongs to the big wheel. This scenario is the one that contains the most occurrence due to the fact that the large wheel is one of the main features that is recognizable in the target object. Even if only one ground point is detected correctly, the parameters of the ellipse that was fitted to the wheel's contour are useful. The fitted ellipse has its own values of width and height. Depending on the proportion of the width with the height, the pose can be obtained with only the big wheel detected. If these two values are almost equal, it would imply that the wheelchair is most likely facing left or right (0° or 180°) since the wheels tend to look circular in these orientations. The narrower the ellipse is, it is close to 90° or 270° which are the poses of facing toward or away from the camera. Nevertheless, the eccentricity of the ellipse fitted to the contour still presents ambiguity to the side in

which the wheelchair is oriented. Nonetheless, the fact that the ground point is obtained, the orientation could be estimated by determining if that ground point is closer to the left or right boundary of the image that encloses the wheelchair. This same process can be applied to case 2 by using the large wheel. The increase of success rate between Table I and Table II for this case is most likely due to the fact that images that resulted in case 1 scenario turned into case 3 by not having the small wheels detected.

Case 4 is limited in information since it only determines one ground point from a small wheel. The orientation would be difficult to determine, but it still gives information of a location point in the image where the wheelchair touches the ground. Case 5 is the one that gives no relevant information. During the tests, the most frequent causes of case 5 were blurry images where the edges wouldn't convey an ellipse. Also, images with shadows hindered the Canny edge detector in differentiating wheels, specially with dark wheels. Another cause is that only parts of the elliptical shape of the wheel were detected but not the complete wheel. It can be observed with the two tables that by applying the mechanism of cropping contours at 0^o, the rate in which ground points are not located decreases.

For the execution time, the algorithm was tested on a laptop with an Intel Core i7 processor with 4Gb of RAM. On average, it took 0.954 seconds to estimate the pose of the wheelchair for images of 450x340 pixels on average. In this case, the whole wheelchair with its pedestrian is analyzed. If possible locations of the wheels are previously estimated, it would narrow the area of focus and improve the execution time.

IV. Discussion

The algorithm that we applied is a basic approach to determining the orientation and location of a wheelchair in a monocular image. By using the mechanism of splitting contours at 0^o, we increased the chance of locating the wheels correctly by fitting ellipses more accurately. This algorithm can be easily replicated and it doesn't require any training and labeled data beforehand which can help in reducing the workload when implementing this algorithm. It only requires a single monocular image and is able to work on different wheelchairs that have similar structures containing large and small wheels. Based on the results obtained, 28.93% of the images gave a very good approximation of the orientation and ground points of the wheelchair with an average error of 4.17 degrees and errors in its ground points of approximately 12 pixels. The case of detecting the big wheel has the highest percentage of 39.05%. Even though this case, case 2 and case 4 don't give the orientation of the wheelchair, it still gives an estimated pixel location of the points where a wheelchair makes contact with the ground with an average error of 15 pixels.

There were problems with the contours when the wheel's edges are not detected specially for the small wheels. Another issue that arose were shadows, since dark wheels are not easily differentiated with an edge detector or even for a human eye. Also, the orientation is not easily determined in cases where the wheelchair is facing toward or away from the camera which implies that the wheels don't resemble an elliptical shape, but rather seem like straight lines or rectangular shapes.

V. Future Work

The next step is to improve our algorithm by using the cases that at least detect a ground point from the wheels. As mentioned before, by having the ground point of the big wheel and the approximated parameters of the fitted ellipse for the case 2 and 3, the orientation can be estimated. This can be achieved by using eccentricity of the fitted ellipse. If the ellipse is narrower, the orientation tends to be closer to either facing toward or away from the camera. As it is wider, closer to its height, it would most likely be oriented to the left or right side. To distinguish between the left side or right side, the ground point can be of a lot of use. If the ground point of the big wheel is closer to the left of the bounding box, it would imply that the wheelchair is facing to the right, and vice versa.

Acknowledgments

We would like to thank Carnegie Mellon University for the opportunity to work in research as part of the Robotics Institute of Summer Scholars Program. We thank Professor Luis Ricardo Salgado and Tomasa Valdivieso for their support during this Summer Program. We also like to thank Instituto Tecnológico y de Estudios Superiores de Monterrey, Campus Monterrey for sponsoring the research stay of the students that worked on this project.

References

[1] A. Crivellaro, M. R. Pascal, Y. V. Vincent, Y. Kwang, "A Novel Representation of Parts for Accurate 3D Object Detection and Tracking in Monocular Images", IEEE International Conference on Computer Vision (ICCV), 2015.

[2] E. Brachmann, F. Michel, A. Krull, M. Y. Yang, S. Gumhold, C. Rother, "Uncertainty-Driven 6D Pose Estimation of Objects and Scenes from a Single RGB Image", CVPR 2016.

[3] K. Rammath, S. N. Sinha, R. Szeliski, E. Hsiao, "Car Make and Model Recognition using 3D Curve Alignment", Applications of Computer Vision (WACV), IEEE Winter Conference, 2014.

[4] H. Chun-Rong, P. Chung, L. Kuo-Wei, T. Sheng-Chieh, "Wheelchair Detection Using Cascaded Decision Tree", IEEE Transactions on Information Technology in Biomedicine (Volume: 14, Issue: 2, March 2010).

[5] A. Myles, N. Vitoria, M. Shah, "Wheelchair Detection in a Calibrated Environment", 2002.

[6] A. Fitzgibbon, R. B. Fisher, "A Buyer's Guide to Conic Fitting", Proc.5th British Machine Vision Conference, Birmingham, pp. 513-522, 1995.

[7] O. Russakovsky*, J. Deng*, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, L. Fei-Fei, (* = equal contribution), "ImageNet Large Scale Visual Recognition Challenge", IJCV, 2015.

Contact Bridge Inspection: Using Drones to Improve Infrastructure

Rebecca McManus
Dept. of Electrical Engineering
University of Vermont
rsmcmanu@uvm.edu

Sebastian Scherer
The Robotics Institute, Department of Computer Science
Carnegie Mellon University
basti@andrew.cmu.edu

***Abstract*—Bridge inspection in the state of Pennsylvania, and the united states at large, is both expensive and infrequent. The time and money spent on inspection could be greatly reduced simply by incorporating an unmanned air vehicle (UAV) into the process. Specifically, contact bridge inspection, which is a technique that requires an inspector to climb the structure of a bridge to use an ultrasonic sensor to detect the thickness of the bridge's steel gussets, can be made cheaper and faster if completed by a UAV. This is done by outfitting the UAV with a sensor which allows it to collect readings, with the challenge being how to modify the UAV in such a way that it can make contact with the bridge without crashing. This paper suggests a design for a lightweight manipulator that can be attached to a UAV and used to collect gusset plate thickness data without compromising the stability of the UAV.**

I. Introduction

THE state of Pennsylvania owns 25000 public bridges. In order to conserve resources, a given bridge is only inspected once every two years, and inspectors rank the damage out of ten to gauge how much repair is needed [1]. This low frequency of inspection reduces the state government's ability to perform preventative maintenance on bridges that could save money in the long run. This paper proposes a new solution to bridge inspection by a novel manipulation strategy aboard an unmanned aerial vehicle (UAV).

A UAV can be used to achieve many different varieties of inspection, including contact bridge inspection. This type of inspection entails an engineer climbing a bridge and using an ultrasonic sensor to detect the thickness of the gusset plates of a steel bridge [2]. Through the use of a light weight unmanned aerial manipulator (UAM), this task can be completed by a UAV.

By using a drone to complete inspection tasks, the process can be done in a way that is safer, cheaper, and more frequent than the current standard. While not highly common, there are documented instances of bridge inspectors being injured and/or killed on the job. These instances are particularly common during processes that require close proximity to the underside and support beams of a bridge [3][4]. The introduction of a UAV into the process of inspection would allow workers to avoid potential injury. Furthermore, By introducing automation through drones, economic resources can be made more optimal in allocation every year by Pittsburgh authorities to other public works projects that improve quality of life in Pittsburgh and beyond.

The UAM is a light, 3 degree of freedom (DoF) articulated robotic manipulator that is designed to compensate for errors in roll and pitch, as shown in figs 1 and 2. This design choice is key for any UAM, as it is unrealistic to assume that a UAV will remain perfectly level throughout its flight. The servos chosen to actuate the roll and pitch joints were selected specifically for their speed and torque ratings, to allow the manipulator to maneuver smoothly through the air. Moreover, the materials from which the arm is built were selected specifically for their low density and strength [5][6]. In order to get an estimate for the manipulators weight, calculations were preformed based on the density of the proposed materials and component dimensions. This was done to ensure that the final manipulator weight would not exceed the payload constraints of the DJI Matrice 100, the UAV upon which the manipulator will be mounted [7].

For overall lightness of the manipulator, special consideration was taken during the process of selecting the servo motors which actuate the joints. Heavier servos were placed at the base of the manipulator, where more torque is required. Meanwhile a smaller servo with a lower torque rating, but higher no load velocity, was placed at the elbow of the arm. The thoughtful actuator placement is meant to distribute weight favorably throughout the manipulator. Finally, rather than adding redundant motors for added maneuverability, the end effector, the part of the manipulator that interacts with the surface of the bridge, was outfitted with a passive 2 DoF to allow it to flush to the surface of the bridge without adding excessive weight.

II. Related Works

Currently, contact inspection is preformed by technicians who get readings from an ultrasonic thickness sensor [2]. Because this requires that the sensor make contact with the surface of the bridge, to collect data a technician must climb the bridge or else use a cherry picker to access hard to reach areas. This can be a slow and dangerous process easily improved by making use of a UAV equipped with an ultrasonic sensor.

Several labs that specialize in UAV application have begun to explore the possibilities of UAMs. One such example is the aerial manipulator designed by T. Bartelds et al [8]. This particular model was designed primarily for the purpose of

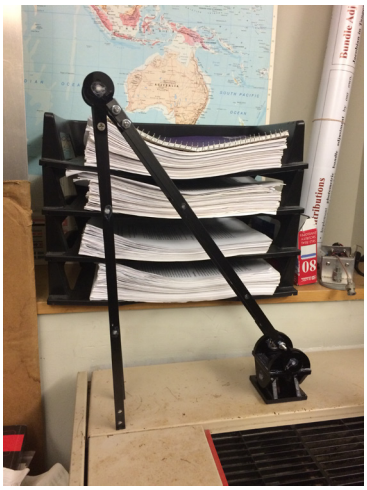


Fig. 1. Physical model of the manipulator

Fig. 2. Physical model of the manipulator with joints labeled

cushioning the impact between the UAV upon which it is mounted and a wall. The task is managed using an elastic in combination with a locking mechanism. The problem with this approach is that without the locking mechanism, the kinetic energy absorbed by the elastic results in oscillations that slow the process down and threaten to destabilize the UAV. When the locking mechanism is in place, the issue lies in the fact that after a single impact, the UAV must land so that the mechanism can be reset. In either event, time is wasted from one impact to the next. To avoid this, this manipulator is articulated, to allow for shock absorbency without undesirable oscillation. Rather than incorporating a locking mechanism that requires the drone to land and be reset manually, the manipulators articulated design allows it to cushion impacts repeatedly without stopping.

Another recent example is a UAM being developed is AEROARMS [9][10]. AEROARMS are 4 to 5 DoF UAV mounted manipulators used for complex aerial maneuvers. Unlike the manipulator designed, AEROARMS are designed to work in pairs, as well as to operate in tandem with other UAVs also outfitted with a pair of AEROARMS manipulators. Depending on the model, a single manipulator weighs between 1.3 and 2.5 kg, which requires that a pair be mounted on a hexrotor. The complexity of this design is unnecessary for the task of bridge inspection, and the weight is excessive. As a result, this particular model is incompatible with the objectives of this project. Given that weight is the largest concern in drone flying, adding more than what is necessary is inadvisable. To allow for longer flight times with fewer battery changes, the manipulator is designed to be as light as possible.

Some UAMs are parallel manipulators, rather than serial manipulators, such as the Dexterous Aerial Manipulator designed by Mina Kamel et al [11]. A parallel manipulator is made of multiple actuated manipulators that connect to a single end effector, which allows for high rigidity and precision of movement [12]. The desirable qualities of a parallel manipulator carry over to this particular UAM, however this design also has its disadvantages. One such issue is the manipulator's position on the vehicle. In order to properly perform inspection, the end effector must have the ability to make contact with parts of the bridge that lay directly above it. In this case, the manipulator is place underneath the UAV, preventing any possibility of overhead maneuvers. the manipulator is designed and positioned to allow the end effector to reach areas both in front of the UAV and above it, improving its effectiveness as a tool for inspection.

The Aerobi project is a final example of a manipulator designed to be mounted onto a UAV [9][13]. In this case, there are a few different ideas utilized specifically for the purpose of bridge inspection. One model simply uses a phenomenon called the 'ceiling effect'. This allows a UAV with a sensor mounted on the top of it to cling to the underside of the bridge and complete contact inspection that way. There are a few problems with this technique. One issue is that there is too much time between each instance of contact due to the UAV needing to align itself properly with the surface it must cling to. Another glaring issue is that this particular technique only works with flat, horizontal surfaces. The UAV is unable to inspect support beams or areas of the bridge's underside that lie at an angle. This leaves a large portion of the bridge unattended to. Aerobi has also produced several different articulated manipulators to complete inspection tasks, but these also have their disadvantages. These manipulators have between 3 and 7 DoF. The models that have 5-7 DoF have actuated end effectors, which adds both excess weight and complexity to the system. Meanwhile, the Aerobi UAM that has only 3 DoF lacks the passive 2 DoF that is included in the manipulator designed and analyzed in this paper.

III. Approach

A. Mechanical Design

The manipulator was designed with certain constraints in mind to suite its environment. Given that the structure will be mounted on a drone, the materials used to create it were kept as light as possible.

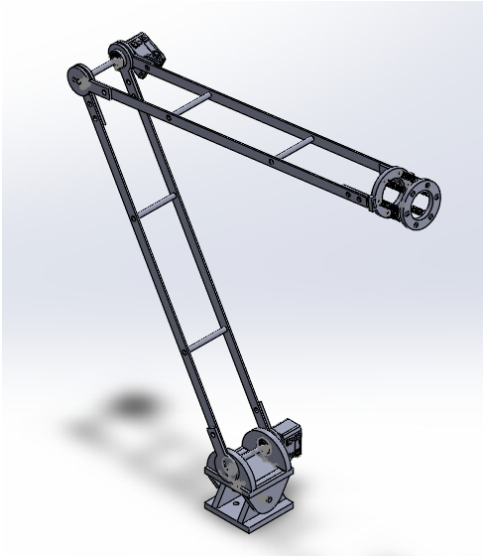


Fig. 3. Solidworks model of the manipulator

The links of the manipulator are made of thin, solid carbon fiber bars to provide rigidity and strength without adding excessive weight. The joints are made from solid aluminum rod in addition to laser cut acrylic, and each contains two thin section bearings which allow for reduced friction. The base and joint connector pieces were custom designed and made using acrylic. To aid in the process of material selection and allocation, the approximate volume of each manipulator part was calculated based on dimensions specified in the model in fig 3. These values were then multiplied by the density of a range of different materials, with the stipulation that the final weight of the arm be less than 1 kg, the maximum payload of a DJI Matrice 100. This estimation yielded a final theoretical weight of about 400 g.

The roll and pitch DoF are each actuated by an off the shelf Dynamixel XM430-W210-T, and the elbow joint is actuated by an off the shelf Dynamixel XL320. The XM430s were chosen for their superior torque rating, at a max stall torque of 3.7 N*m. The disadvantage to these particular motors is their weight. For this reason, a different, lighter motor was chosen to control the elbow joint. Though the XL320 has a lower torque rating, of .39 N*m, it is significantly lighter and faster than the other actuator model used in the manipulator.

The manipulator has 3 (DoF) that allow it to maneuver, as well as to compensate for disturbances. Error correction is imperative to keep the end effector in the right spatial position, as a UAV will roll and pitch unpredictably. The end effector of the manipulator designed not to need motors, instead using a passive 2 DoF, shown in fig 4. This allows the end effector to be flexible and easily flush to a surface without the need for motor redundancy.

B. Electrical Design

The board used for this project is an OpenCM9.04, also produced by ROBOTIS. Both the board and the servo motors chosen to control the manipulator have the advantage of ROS compatibility. The servos themselves are specialized for position or torque control, meaning that the manipulator can potentially utilize elements of force as well as position control.

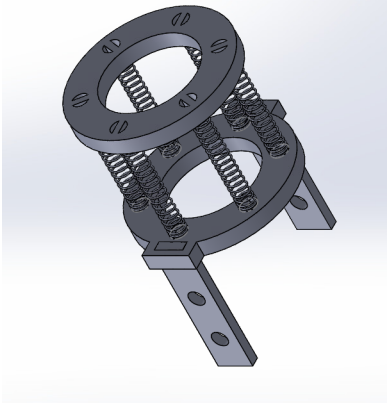


Fig. 4. The sensor is mounted in this structure, which is rendered flexible by a series of light duty springs. This allows the end effector to bend based on the angle at which it touches the surface of the bridge.

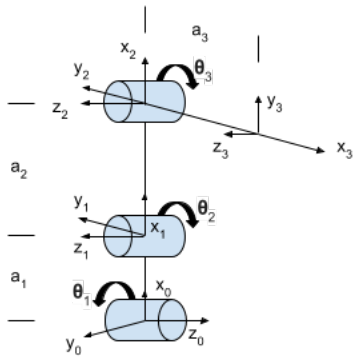


Fig. 5. This isometric model view is labeled to conform to the Denavit Hartenberg frame convention. This choice of convention simplifies the process of analysis. It should be noted that the end effector frame x_3, y_3, z_3 has been placed on top of the x_2, y_2, z_2 frame. This was done to simplify the final mathematical figuring while still conforming to the frame conventions.

C. Analysis

The analytical model of the manipulator shown in fig 5 was developed based on the Denavit Hartenberg modeling conventions [14][15]. This convention is used to create reference frames in each joint as well in the end effector in order to assist in the process of kinematic analysis.

The mathematical analysis is based on the geometry of the manipulator, where forward kinematics can be used to find the end effector position based on the angle of each joint, and inverse kinematics can be used to find the joint angles based on the final x, y, z position of the end effector.

The forward and inverse kinematics are based on fig 5 as well as the side and front view models in fig 6. Fig 7 shows the variables being solved for.

Table 1 contains the Denavit Hartenberg parameters that can be used to determine the homogeneous transformation matrices needed for the forward kinematics of the system.

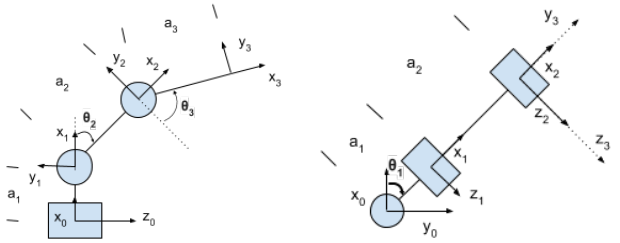


Fig. 6. In order to find the inverse kinematic equations for the manipulator, a model must be examined from several directions to get a full definition for the movement of the joints. In this case, the manipulator is articulated using revolute joints, meaning that variables being calculated for are the angles θ_1, θ_2, and θ_3. The solutions to these equations can be used to find the joint angles that will result in a desired end effector position. Note that here the x_3, y_3, z_3 is distinguished from the x_2, y_2, z_2 as a dotted line.

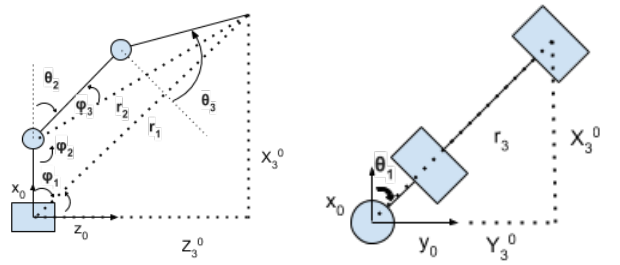


Fig. 7. A combination of trigonometry and algebra must be used to find the inverse kinematic equations for the manipulator. As is shown in the figure, the variables ϕ_n and a_n denote the angles and link lengths that are used to find θ_n. X_3^0 and Z_3^0 on the left denote the final [x,z] position of the end effector in relation to the base frame (x_0, y_0, z_0). $X_3^0 and Y_3^0$

These parameters can be plugged into 1 in order to calculate each transformation:

$$^{i-1}A_i(q_i) = Trans(0,0,d_i)Rot(z_{i-1},\theta_i)Trans(a_i,0,0)Rot(x_i,\alpha_i) \quad (1)$$

where $^{i-1}A_i(q_i)$ is the transformation between the $(i-1)$ coordinate frame and the ith frame. These equations result in the

Table 1: Denavit Hartenberg Parameters

a_i	α_i	θ_i	d_i
a_1	$\pi/2$	θ_1	0
a_2	0	θ_2	0
a_3	0	θ_3	0

$$^{i-1}A_i(q_i) = \begin{bmatrix} c(\theta_i) & -s(\theta_i)c(\alpha_i) & s(\theta_i)s(\alpha_i) & a_i c(\theta_i) \\ s(\theta_i) & c(\theta_i)c(\alpha_i) & -c(\theta_i)s(\alpha_i) & a_i s(\theta_i) \\ 0 & s(\alpha_i) & c(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

IV. Results

Using the models in fig 7, the following system of equations, equations 3 - 11, were derived.

$$r_1 = \sqrt[2]{(X_3^0)^2 + (Z_3^0)^2} \quad (3)$$

$$\phi_1 = \pi/2 - tan^{-1}(X_3^0/Z_3^0) \quad (4)$$

$$r_2^2 = a_1^2 + r_1^2 - 2 * a_1 * r_1 * cos(\phi_1) \quad (5)$$

$$\phi_2 = cos^{-1}(\frac{r_1^2 - (a_1^2 + r_2^2)}{-2 * a_1 * r_2}) \quad (6)$$

$$\theta_3 = cos^{-1}(\frac{r_2^2 - (a_2^2 + a_3^2)}{-2 * a_2 * a_3}) - \pi/2 \quad (7)$$

$$\phi_3 = cos^{-1}(\frac{a_3^2 - (a_2^2 + r_2^2)}{-2 * a_2 * r_2}) \quad (8)$$

$$\theta_3 = 180 - (\phi_2 + \phi_3) \quad (9)$$

$$r_3 = \sqrt[2]{(Y_3^0)^2 + (X_3^0)^2} \quad (10)$$

$$\theta_1 = \pi/2 - tan^{-1}(X_3^0/Y_3^0) \quad (11)$$

These equations are the inverse kinematics of the system, which use a desired end effector position to determine the joint angle values.

The homogeneous transformation matrix between the base frame [X_0, Y_0, Z_0] and the end effector frame, [X_3, Y_3, Z_3] is the following the result of multiplying the homogeneous transformation matrices between each frame, A_1, A_2, and A_3, multiplied together.

V. Conclusion

The next steps for this project is developing the dynamical model for the manipulator. Given that it is a 3 DoF, articulated, spatial manipulator, the dynamics of the system are complex. Recently, neural networks have been used to develop the dynamics of more complicated manipulators, and that option seems viable for this project.

Once the dynamical model is complete, a control system can be developed. Potentially, this could be based in position and force control. Using the dense, three dimensional point cloud of the environment already collected, the control could calculate vector normals to the surface of the bridge. From there, traditional position control techniques can position the end effector to take readings. Once the end effector makes contact with the bridge, torque sensors in the manipulator's joints will detect differences in the force on the system and the manipulator can move to accommodate the change.

REFERENCES

[1] C. of Pennsylvania, "Bridge information," Available at http://www.penndot.gov/ProjectAndPrograms/Bridges/Pages/default.aspx (2017/06/23).

[2] M. P. . S. Division, "Bridge gusset plate ultrasonic inspection," Princeton Junction, NJ, Tech. Rep., 2009.

[3] B. Mayo, "Bridge inspector fatally injured on interstate 279 on north side," Available at http://www.wtae.com/article/bridge-inspector-fatally-injured-on-interstate-279-on-north-side/7461776 (2017/07/19).

[4] M. Hasch, "Worker dies from injuries in east ohio street bridge accident," Available at http://triblive.com/news/adminpage/4224098-74/police-bridge-worker (2017/07/19).

[5] A. Composites, "Mechanical properties of carbon fiber composite materials, fiber / epoxy resin (120c cure)," Livermore, CA, Tech. Rep., 2014.

[6] "Densities of solids," Available at http://www.engineeringtoolbox.com/density-solids-d_1265.html (2017/07/10).

[7] "Matrice 100 specs," Available at https://www.dji.com/matrice100/info (2017/06/13).

[8] S. H. S. S. T. Bartelds, A. Capra and M. Fumagalli, "Compliant aerial manipulators: Toward a new generation of aerial robotic workers," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 477–483, 2016.

[9] A. F. Marco Tognon, "Position tracking control for an aerial robot passively tethered to an independently moving platform," *Rapport LAAS n 17107*, 2017.

[10] A.Ollero, "Workshop on autonomous structural monitoring and maintenance using aerial robots," *2017 International Conference on Robotics and Automation (ICRA)*, 2017.

[11] K. A. M. Kamel and R. Siegwart, "Design and modeling of dexterous aerial manipulator," *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4870–4876, 2016.

[12] E. by Serdar Kk, *Serial and Parallel Robot Manipulators Kinematics, Dynamics, Control and Optimization*. Rijeka, Croatia: InTech, 2012.

[13] "Aerial robotic system for in-depth bridge inspection by contact: About," Available at http://www.aerobi.eu/about (2017/06/29).

[14] M. M. Tadej Bajd, Matja Mihelj, *Introduction to Robots*, ser. SpringerBriefs in Applied Sciences and Technology. Springer Netherlands, 2013.

[15] P. Corke, "Denavit-hartenberg notation for common robots," 2014.

Generating Hard Positive Examples via Adversary for Occluded Traffic Sign Detection

Stephanie Milani
Department of Computer Science
University of Maryland, Baltimore County
Baltimore, MD 21250, USA
smilani@andrew.cmu.edu

Christoph Mertz
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
cmertz@andrew.cmu.edu

***Abstract*— Detecting occluded traffic signs is a challenging problem due to the rarity of some occlusions and deformations. To address this issue, we propose using an adversary to generate hard positive examples of occlusions to make it more difficult for a detector to detect the sign. We detail our framework and the Keras implementation of connecting the adversarial network with the Faster R-CNN pipeline. We conclude by stating the steps necessary to complete the implementation and suggest areas for future work.**

I. INTRODUCTION

Well-maintained traffic signs are crucial for helping both human and non-human drivers make safe and informed decisions. The task of maintaining these signs primarily falls upon federal and local maintenance departments who must monitor sign conditions for issues of damage, defacement, theft, and occlusion. Currently, signs are monitored by: (1) inspectors and (2) citizens who report their observations. The first method is tedious and suffers from unreliability due to the subjective judgment of inspectors. The second method is unideal because reports by citizens are typically only made about severe problems, not about ”mild” graffiti or a few stickers on a stop sign. Previously, a road infrastructure inventory and assessment system, was developed by the Navlab group to provide a low cost and consistent quality solution to monitor infrastructure distress on a continuous basis with minimal human intervention [28]. Although the system detected most of the stop signs, a significant portion of those that it missed were in poor condition or obstructed by vegetation, graffiti, or other vehicles.

Ample research has been conducted on the problem of traffic sign detection; however, detecting occluded traffic signs is still a challenging problem. One reason is the rarity of some occlusions and deformations results in a low probability of representation in large datasets, meaning that the classifier will not be able to robustly model invariances to these rare events. The standard solution to this problem is to gather more data; however, this method is tedious and does not guarantee that the more rare occlusions and deformations will be represented. A less tedious solution that solves the representation problem is to generate realistic images depicting all possible occlusions that might be encountered. The problem with generating all possible occlusions is that it is infeasible due to the huge space of possible occlusions and suboptimal because selecting hard examples is better than showing all possible examples [21], [25]. A better solution, then, is to create hard positive examples of occlusions without actually generating the images, or pixels, themselves.

Wang et al. [30] used an adversary to create hard examples of occlusion by spatially blocking some feature maps. The goal of the adversary is to make it more difficult for a detector, like Fast Region-based Convolutional Network (Fast R-CNN), to detect the object; the goal of Fast R-CNN is to adapt to the challenges of the adversary to better classify these adversarial examples. While this work showed improved detector performance on the PASCAL VOC 2007, PASCAL VOC 2012 [7] and MS COCO [15] datasets, it was not applied to the task of traffic sign detection.

In this work, we seek to enhance the robustness of our inspection system to detect occluded traffic signs. To that end, we propose a novel approach to traffic sign detection: the use of an adversary to create hard examples for the detector. For our detector, we select the third generation of R-CNN: Faster Region-Based Convolutional Network (Faster R-CNN).

The primary contributions of this paper are:

- We combine the adversarial network proposed by Wang et al. [30] with Faster R-CNN, creating adversarial Faster R-CNN (A-Faster R-CNN).
- We discuss the steps taken to implement the network.
- We propose applying A-Faster R-CNN to the task of traffic sign detection using the LISA dataset [18]. Specifically, we target the task of stop sign detection. We focus on stop signs because they are one of the most common types of road signs and issues with these signs are a severe safety issue.
- Finally, we propose ways to extend the method.

The remainder of the paper is organized as follows. Section II reviews the related work in object detection, traffic sign detection, and adversarial learning. In Section III, we describe the framework of the network. Then, in Section IV, we discuss the steps taken to implement the network. In Section V, we discuss the experimental setup for applying the network to detecting occlude traffic signs. Finally, we discuss future work in Section VI.

II. RELATED WORKS

A. Object Detection

In recent years, significant progress has been made in object detection. These achievements were made possible by the successful application of deep ConvNets [14] to ImageNet classification [6]. One of the detectors in the forefront of the movement, R-CNN [9], achieved impressive results on ImageNet and PASCAL VOC detection by using region proposals. This method, popularized by the selective search algorithm [27], has been more recently improved, resulting in other region-based convolutional neural networks, such as SPPnet [11], Fast R-CNN [8], and Faster R-CNN.

Since then, Faster R-CNN has been successfully used in a plethora of tasks, including, but not limited to, a fruit detection system for fruit yield estimation and automated harvesting [20]. To our knowledge, Faster R-CNN was applied to the task of traffic sign detection only once before [32]. Unfortunately, the method obtained relatively poor results. However, the results could be attributed to any number of factors - including the choice to classify traffic signal lights as traffic signs, which may have confused the detector. Applying Faster R-CNN and any of its variations to traffic sign detection is an area still in need of exploration.

B. Traffic Sign Detection

Many of the previous traffic sign detection methods benchmark their systems using European signs [16], [31], a large number of which are standardized by the Vienna Convention on Road Signs and Signals [5]. Although the countries that follow this standard may have other types of signs, those signs tend to be similar. Traffic signs in the United States are not as uniform. In the U.S., the Manual on Uniform Traffic Control Devices (MUTCD) [4] attempts to provide uniformity of traffic devices by specifying the standards by which traffic signs are designed, installed, and used. What makes U.S. traffic signs particularly complicated is that each state can choose to adopt the MUTCD in its entirety, adopt the MUTCD but add a State Supplement, or adopt a State MUTCD that must substantially conform with the national MUTCD. Due to the level of complexity resulting from the variations between signs throughout the country, developing a traffic sign recognition system for signs in the U.S. that is robust to inter-class variations is of particular importance that is not strictly limited to detecting traffic signs that have been covered in some way.

C. Adversarial Learning

To simulate adversarial learning, a discriminative model is trained under various scenarios that are formulated by an intelligent adversary [10]. Recent work in adversarial learning has explored a number of directions, including, but not limited to, using augmented data to improve the robustness of neural networks [2]. Using adversarial learning to improve the robustness of a network with respect to occluded traffic signs has yet to be explored.

III. FRAMEWORK

In this section, we describe the details of our framework. We first briefly go over the base detector, Faster R-CNN. After that, we describe the adversarial network we use to generate different types of occlusions. Finally, we discuss how we combined the two networks into one framework.

A. Overview of Faster R-CNN

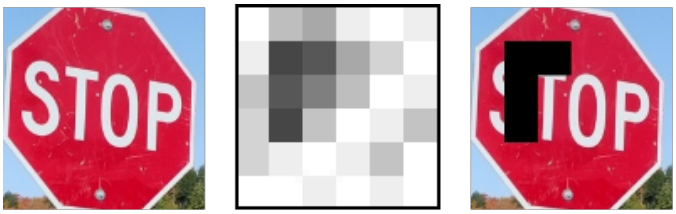
For our framework, we build upon Faster R-CNN [19] as the base detector for traffic sign detection. Faster R-CNN is a unified network for object detection composed of two modules: (1) a Regional Proposal Network (RPN), which is a deep fully convolutional network that proposes regions, and (2) the Fast R-CNN detector that uses the regions proposed by the RPN [19].

Given an input of an entire image, the convolutional network of Faster R-CNN outputs convolutional feature maps. These feature maps are then shared with the RPN. As input, the RPN takes an image of any size. It then outputs a set of rectangular object proposals, called bounding boxes. Each of these bounding boxes has an objectness score. Because the RPN shares full-image convolutional features with the detector at test time, the method alleviates the computational burdens of selective search, making the region proposals almost entirely cost-free. After that, each bounding box that is likely to be an object is passed into Fast R-CNN to generate a classfication and to refine the bounding boxes.

B. Adversarial Spatial Dropout for Occlusion Detection

To improve the robustness of our system, we use the Adversarial Spatial Dropout Network (ASDN), the adversarial network proposed by Wang et al. [30], to generate occlusions on deep features of foreground objects - namely, traffic signs. Rather than generating entirely new images, we restrict the space of new positive generation to adding occlusions to the existing examples from the dataset. The network accomplishes the task of generating occlusions by taking the region-based convolutional features obtained for each foreground object proposal after the region-of-interest (RoI) pooling layer in the standard Faster R-CNN pipeline and generating a mask to indicate which parts of the feature to dropout. Dropout is a technique typically used to deal with overfitting in deep neural networks by randomly removing units, as well as their connections, from the neural network during training [24]. In the case of the ASDN, dropout is not randomly applied; instead, it is intentionally applied through

Fig. 1: Example of mask. Black regions are occluded when passed into Faster R-CNN pipeline.



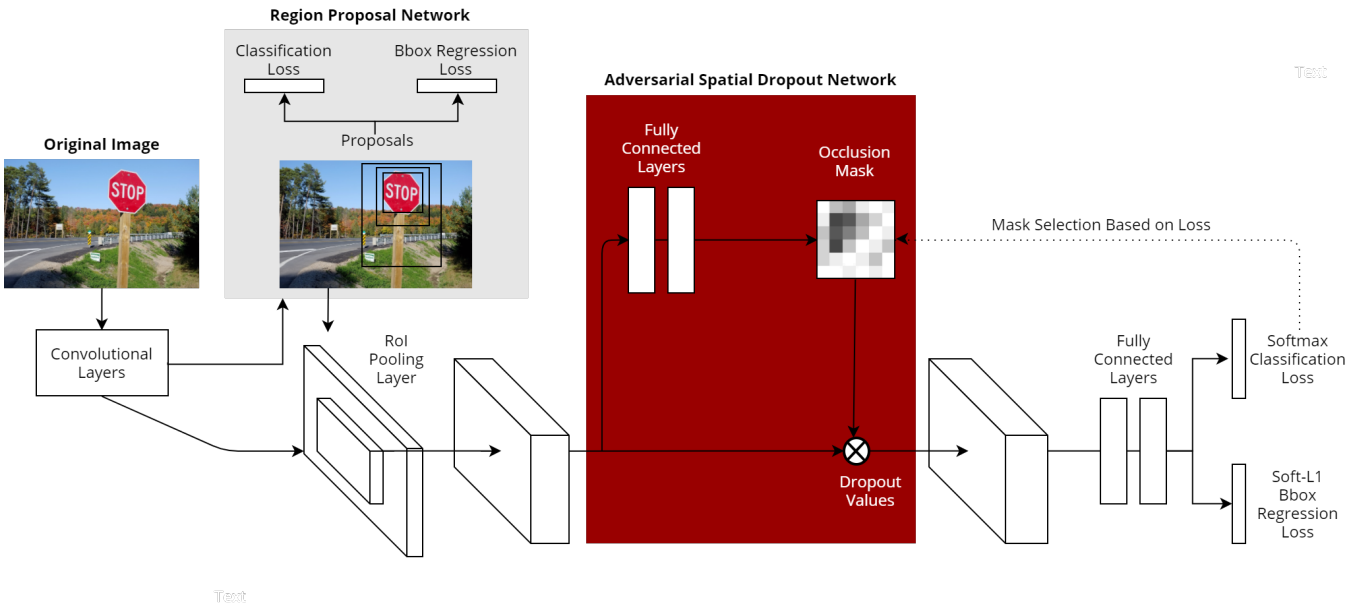


Fig. 2: Our network architecture of ASDN and how it combines with Faster R-CNN.

the generated dropout masks. The network learns to generate the dropout masks that cause the detector to have high loss by using the window with the highest loss from the RPN to generate a spatial mask for each positive region proposal. See Figure 1 for an example of a dropout mask.

IV. Implementation

We combine the Faster R-CNN architecture with the ASDN to create A-Faster R-CNN. Figure 2 depicts the ASDN architecture integrated with the Faster R-CNN approach. In this section, we describe the details of our implementation. For our implementation, we are working in Keras with a TensorFlow [1] back end and using the standard Faster R-CNN architecture with VGG [22]. To implement the pipeline, we are using existing Keras code [1] for the detector, Faster R-CNN. We are basing the Keras version of the ASDN off of the original Caffe implementation [2]. In this section, we detail the changes and additions that are made to the existing code to create A-Faster R-CNN.

A. Convolutional Network and Region Proposal Network

Not many changes need to be made to the part of the network before the ASDN. The convolutional network consists of various convolutional and max pooling layers. The main change we make is we separate the RoI pooling layer from the layers to enable its outputs to be used by both the adversarial network and the classifier at the end of the pipeline. The output of the RoI pooling layer, a fixed size feature vector, is passed to both the ASDN and to the fully connected layers. No changes are made to the region proposal network.

[1]https://github.com/yhenon/keras-frcnn
[2]https://github.com/xiaolonw/adversarial-frcnn

B. Adversarial Spatial Dropout Network

To implement the Adversarial Spatial Dropout Network, we separate the network into two parts: the convolutional layers and the masking layer. For the first part of the ASDN, the convolutional layers, we create five convolutional layers. These layers generate the mask prediction as output. The second part of the ASDN, the data layer, is what creates the mask. Since this type of layer does not exist yet in Keras, we create our own using the custom layer functionality.

The input for the custom ASDN data layer is the mask prediction generated by the convolutional layers of the ASDN and the labels. Since it is critical that the input tensors are the proper shape, we set the shape of the ASDN to be the same shape as output of the RoI pooling layer. The output for this layer is the dropout mask, which is then passed to the classifier. Currently, our layer outputs a dropout mask of all ones. Future work for this part of the network is to make this layer generate the actual mask.

C. Classifier

The classifier tower after the ASDN requires a few changes to connect the ASDN to the Faster R-CNN pipeline. We update this part of the network so that it takes as input both the output of the data layer of the ASDN, the dropout mask, and the output of the RoI pooling layer. Before entering the fully connected layers, we use a multiplication layer to combine the dropout mask and the output of the RoI pooling layer to create the masked layer. In the future, we will compute which masks cause greater classification loss for Faster R-CNN and use those hard examples as ground truth to train the ASDN as described in the next section.

V. Experimental Setup

In this section, we describe the experimental setup that we will use to apply the network to the task of traffic sign detection. First, we will describe the pre-training and training processes we will use. After that, we will provide details about the data we will use for training the network. Finally, we present the data we will use to test the network.

A. Pre-Training

As suggested by Wang et al. [30], we will use stage-wise training [19] to pre-train the ASDN for creating occlusions prior to using it to improve Faster R-CNN. First, we will train the Faster R-CNN detector without the ASDN to give the detector a sense of the signs in the database. We train the detector without the ASDN because join training from the beginning will cause the adversarial network to learn more quickly than the detector and will easily overpower it. Second, we will fix all the layers in the detector to train the ASDN model for occlusion generation. Although this method of training is typically not recommended for generative adversarial networks, our model does not run the same risk of collapsing as traditional generative adversarial networks.

B. Training

When training A-Faster R-CNN, we will jointly optimize the pre-trained ASDN and Faster R-CNN model in each iteration of training. We will use the approach detailed by Wang et al. [30] for training both the ASDN and the Faster R-CNN detector.

C. Training with the LISA Dataset

We will apply A-Faster R-CNN to the challenge of traffic sign recognition by training it using the LISA dataset. The LISA dataset contains 6610 color and greyscale images of 47 types of United States traffic signs [18]. The images, collected in the state of California, vary in size from 640x480 to 1024x522 pixels. The database contains 7855 annotations; each annotation includes the sign type, position, size, whether or not the sign was on a side road, and whether or not the sign was occluded.

Fig. 3: Example of a stop sign obstructed by vegetation from the LISA dataset.

(a) Obstruction by vegetation

(b) Obstruction by damage

(c) Obstruction by vandalism

Fig. 4: Examples from the Navlab dataset of stop sign occlusion by vegetation, damage, and vandalism.

We analyzed the dataset to see the distribution of occluded signs. Of the 7855 annotated signs, 1821 are stop signs (23.2 %). At a count of 304, occluded signs make up a small portion of the total number of signs (3.9%). Of the total number of occluded signs, 124 are stop signs (40.8%), which makes them the most common type of occluded sign. The most common causes of occlusion are from other infrastructure (53.2%) and vegetation (42.7%). Figure 2 shows an example of a stop sign obstructed by vegetation from the LISA dataset.

We plan to extract the occluded signs and only use the unobstructed data for training. We will use the occluded signs as a means of validating and testing the network on real-world examples.

D. Testing with the Navlab Dataset

We will use the Navlab stop sign dataset to test the network. Derived from a larger dataset of around 20 million street images collected in and around Pittsburgh, the Navlab stop sign dataset consists of 1132 images of stop signs [17]. Of the total number of signs, 784 are undamaged (69.3%), 90 are undecided (8.0%), 60 are three- or four-way (5.3%), and 290 are damaged (25.6%). Of the ones that are damaged, the most common causes are vandalism (15.1%) and displacement (8.1%). Figure 4 shows some examples of occluded stop signs from the dataset. These images will be used for testing the detector to see if the addition of the adversarial network for generating occlusions will improve the ability of the network to detect real examples of occluded stop signs from a different dataset than the LISA.

The dataset also includes 1755 negative examples of stop signs. These examples focus on images that share properties with stop sign, such as white lettering, red coloring, or similar shape. These examples will be used to test the network to understand issues of overgeneralization in both the standard Faster R-CNN pipeline and the new A-Faster R-CNN network.

VI. Future Work

In the future, this network should be fully implemented in Keras. The work that remains includes connecting all of

the layers together properly and the other implementation details mentioned in Section IV. After the network is fully implemented, we will train the detector as stated using the LISA dataset and test the method using the Navlab dataset. This method could then be trained and tested on the German Traffic Sign Recognition Benchmark (GTSRB) [23] to benchmark it to the standard and to see how well it generalizes.

After that, this method could also be extended in a number of ways. One extension is adding new adversarial networks that target remaining issues with traffic sign detection, such as sign discoloration, differences in lighting, and warped signs.

Another area for exploration is the incorporation of top-down methods, such as utilizing a preexisting inventory of traffic signs to predict whether a sign ought to be in the scene or not. The detector could also use previous information about the world, such as where specific types of signs typically occur, to make predictions about where signs ought to be.

Finally, this method could be extended to real-time traffic sign detection. Previous work in traffic sign detection systems, mobile or otherwise, has focused on minimizing the computational burden of the system to enable it to run efficiently and effectively in real-time [3], [26], [29]. Although our detection system performs its detection offline, alleviating the computational burden of online detection, we selected a network that balances both speed and accuracy [12] in the hopes that future improvements to this work could lead to the development of a real-time traffic detection system for road infrastructure monitoring.

ACKNOWLEDGMENT

This work was generously supported by Traffic21. Stephanie would like to thank Nicholay Topin and Dat Nguyen for their advice, assistance with software, and support. She would like to thank Jina Wang and the rest of the Navlab group for great discussions and support.

REFERENCES

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp. G. Irving, M. Isard. R. Jozefowicz, Y. Jia, L. Kaiser, M. Kudlur, J. Levenberg. D. Mané, M. Schuster, R. Monga, S. Moore, D. Murray, C. Olah, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems", 2015. Software available from tensorflow.org.

[2] S. W. Akhtar, S. Rehman, M. Akhtar, M. A. Khan, F. Riaz, Q. Chaudry, and R. Young, "Improving the Robustness of Neural Networks Using K-Support Norm Based Adversarial Training" in *IEEE Access*, 2016.

[3] S. Ardianto, C. Chen, and H. Hang, "Real-time traffic sign recognition using color segmentation and SVM" in *IWSSIP*, 2017.

[4] *California manual on uniform traffic control devices for streets and highways*, State of California, Dept. Transportation, Sacramento, CA, 2006.

[5] *Convention on road signs and signals of 1968*, United Nations Economic Commission for Europe, Geneva, Switzerland, 2006.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li., and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database", in *CVPR*, 2009.

[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge", in *IJCV*, 2010.

[8] R. Girshick, "Fast R-CNN" in *ICCV*, 2015.

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation" in *CVPR*, 2014.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Networks" in *NIPS*, 2014.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition in *CVPR*, 2015.

[12] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy, "Speed/accuracy trade-offs for modern convolutional object detectors" in *CoRR*, 2015.

[13] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding" in *arXiv preprint*, 2014.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks" in *NIPS*, 2012.

[15] T. Lin, M. Maire, S. Belongie, L. D. Bourdev, R. Girschick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context" in *CoRR*, 2014.

[16] H. Luo, Y. Yang, B. Tong, F. Wu, and B. Fan, "Traffic sign recognition using a multi-task convolutional neural network" in *IEEE Transactions on Intelligent Transportation Systems*, 2017.

[17] C. Mertz, J. Kozar, J. Wang, J. Doyle, C. Kaffine, A. Kelkar, N. Poddatur, L. Chan, and K. Amladi, "Smartphone based traffic sign inventory and assessment" in *Technologies for Safe and Efficient Transportation*, 2016.

[18] A. Møgelmose, M. Trevedi, and T. Moeslund, "Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey" in *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, 2012.

[19] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks" in *CVPR*, 2015.

[20] I. Sa, Z. Ge, F. Dayoub, B. Upcroft, T. Perez, and C. McCool, "DeepFruits: A Fruit Detection System Using Deep Neural Networks", in *Sensors (Basel, Switzerland)*, vol. 16, no. 8, 2016.

[21] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining" in *CVPR*, 2017.

[22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition" in *CoRR*, 2014.

[23] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition", in *Neural Networks*, 2012.

[24] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting" in *Journal of Machine Learning Research*, vol. 15, 2014.

[25] M. Takáč, A. Bijral, P. Richtárik, and N. Srebo, "Mini-batch primal and dual methods for svms" in *Proceedings of the 30th International Conference on Machine Learning*, vol. 28, 2013.

[26] B. Trăsnea, G. Măceşanu, S. Grigorescu, and T. Cociaş, "Smartphone based mass traffic sign recognition for real-time navigation maps enhancements" in *IEEE*, 2017.

[27] J. R. R. Uijilings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition", in *International Journal of Computer Vision*, 2013.

[28] S. Varadharajan, S. Jose, K. Sharma, L. Wander, and C. Mertz, "Vision for road inspection", in *Proceedings of 2014 Winter Conference on Applications of Computer Vision*, 2014.

[29] D. Wang, X. Hou, J. Xu, S. Yue, and C. Liu, "Traffic sign detection using a cascade method with fast feature extraction and saliency test" in *IEEE Transactions on Intelligent Transportation Systems*, 2017.

[30] X. Wang, A. Shrivastava, and A. Gupta, "A-Fast-RCNN: Hard positive generation via adversary for object detection" in *CVPR*, 2017.

[31] F. Zaklouta and B. Stanciulescu, "Real-time traffic sign recognition in three stages" in *Robotics and Autonomous Systems*, vol. 62, no. 1, 2014.

[32] Z. Zuo, K. Yu, Q. Zhou, X. Wang, and T. Li, "Traffic signs detection based on faster r-cnn" in *ICDCSW*, 2017.

Autonomous Coarse localization in Industry

Aisha Mohamed
Computer Science Department
Carnegie Mellon University
Doha, Qatar
aishahas@andrew.cmu.edu

Mabaran Rajaraman
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA, US

Abstract*—Global robot localization is essential to the function of autonomous robot systems. Current approaches to global localization start the localization from scratch knowing nothing about the object. It extract low-level features from sensor data and use template matching to estimate the robot's pose. This process of matching low-level features is very expensive in terms of computation and memory. Additionally, attempting to match detailed low-level features within a large feature space can lead to inaccurate estimations. In this work, we propose a system that first finds an initial coarse pose estimation based on high-level features extracted from CAD models. Variation in appearance of an object across progressive viewpoints is an abstract indicator of its features/shape. This variation also encodes high-level information of the object like symmetry, feature locations and their transition patterns. Our system uses the variation in appearance to build progressive viewpoints and localize the robot in that map. The system proposes actions and moves for the robot to perform to collect sensor data, which it uses to update its estimation. (Results) When the robot localizes itself in a specific domain, it then performs fine localization in the narrowed feature space. By moving from coarse to fine localization, this process increases accuracy and efficiency of pose estimation.

Keywords*—localization, viewpoint maps, pose estimation.

I. Introduction

Robot localization is the problem of finding the position and direction of a robot with respect to the environment. It is an essential problem in building autonomous robot systems. Autonomy of a robot requires a minimum knowledge of its location w.r.t the environment[1]. There are two instances of the robot localization problem. The first instance is the position tracking problem where a robot has an initial estimation of its pose and is required to keep track of it's pose while navigating the environment. The second instance is the global localization problem which doesn't assume an initial pose and tries to localize itself from scratch using the sensory information[2]. Global localization is especially important for autonomous systems as it enables the robot to autonomously localize itself at initialization and relocalize itself whenever it loses its location. The current system robots perform tasks that require more than knowing the absolute position of the robot like navigation or automated assembly where it needs to know the location of a robot w.r.t. to a specific object. This adds to the definition of the global localization problem that it finds the pose of a robot with respect to an object. In this paper we are addressing Robot localization w.r.t an object.

Most of the current approaches to global localization do low level feature extraction on the sensory data and then compare it to a-priori information[4]. The system finds the similarities between features of the sensory data and features in the pre-built map. While this approach results in a fine localization of the robot, It has some inherent weaknesses. The performance of this approach depends on the feature representation and the accuracy and efficiency of the feature extraction and matching. It is very hard for a robot to find a best-fit for the location in a multi-dimensional map and a huge search space. Some of the features may be repeated in the environment and some may be too small to be detected before the algorithm converges. As a result,, It only provides an estimation of the location with high uncertainty. Accurate approximation of the location has a very high computational and memory cost[5] necessary for feature extraction and matching in multi-dimensional pose-space representation.

In this paper we present a system for staggering the localization. The idea of the system is starting with coarse localization of objects to narrow down the search space and then use fine localization to estimate the pose in the narrowed search space. It assumes the presence of a CAD model of the observed object which is available for most of the man-made objects.[3]. It extracts the high-level features that describes the outer appearance of the object like shape and symmetry from the CAD models. It uses CAD models as a-priori information rather than low-level features maps. It then builds progressive viewpoint maps to show the variation in appearance across the viewpoints and use it to coarsely localize the robot in a wide domain with high certainty. It represents the belief of the robot by a probabilistic distribution over all the possible poses to allow the robot to confirm or abandon its belief of the location. It then proposes moves or actions and update the robot belief about its pose after each move using temporal patterns. This allows the localization system to have an initial coarse localization before fine localization which increases the accuracy and reduces the cost of fine localization.

Our main contributions are:

1) Given a CAD model of the object, we build a rendered 4D image of the object and extract its high level features across progressive viewpoints. We cluster viewpoints based on these features and build a viewpoint map of the object. The map shows the appearance variation across the viewpoints and indicates the shape and symmetry property of the object.
2) We coarsely localize the observer (robot) in one of the views by providing a Gaussian probabilistic distribution over all the views.
3) We suggest moves and actions and use the sensory data collected to update the probabilistic distribution.

In the rest of the paper, we will discuss the related work in section II. We will describe coarse localization using high level feature extraction in section III and probabilistic model of the localization in section IV. We will show results in section IIV.

II. Related Work

The main approach to localization uses a probabilistic distribution to present the robot's location. It assumes there is noise in the sensors, and in the affectors. It takes these into consideration be representing the location as a belief. It starts with an initial estimation of the location. After each move, action or observation it updates the belief using a conditional distribution on the change that happened. In most of these systems, a map of the environment is present and a

the location is a specific location in the map. The difference between localization systems is in the representation of the map, in the representation of the vectors and the choice of the feature vectors to be extracted from the map.

III. Coarse localization from CAD models

Our system suggests that CAD models encode information about the object's shape that can be used to extract the appearance variation across the object. The CAD models carry enough information to build a rendered image that is very similar to the real image of the object in terms of appearance and high-level geometric features. Our system extracts the high level features from the rendered image from different positions, clusters the positions with similar features in one cluster and use it to build a viewpoint map that indicates the appearance variation across different viewpoints. Figure 1 shows the viewpoint map of the sphere and the cube when rotated around one axis. The pose of the sphere is exactly the same along all the viewpoints as it's a symmetric shape along all axis. The view point map show all the sphere views to be in one domain as all the views have the same features. However, The cube views change by rotation around one axis. The similar views are clustered in the same domain in the viewpoint map.

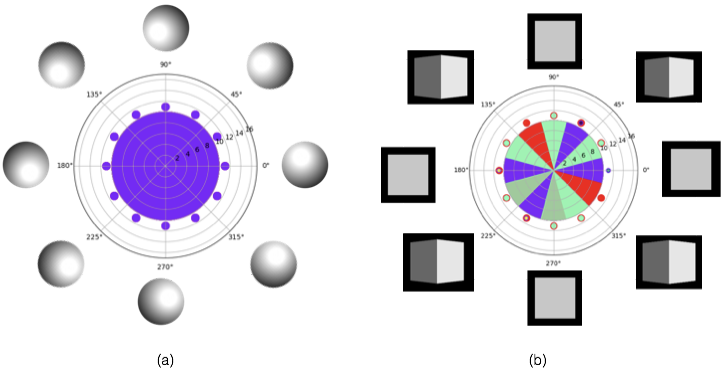


Fig. 1. Viewpoint map of (a) a sphere rotated around one axis using GMM. (b) a cube rotated around the one axis using GMM with the corresponding pose of the object at each view.

The view point map can then be used to convey the location of the observer with respect to the object. To illustrate how this is possible, assume the observer robot is facing the cube at figure 1(b) at angle 230. The robot will take a picture of the cube and will match it's features with one or more of the viewpoint map domains. In this case, the robot will be mapped to all the four views in the purple domain. The robot will then move right and left to confirm its pose. It will move to the right and capture another image of the cube. It will map it to one of the domains in the map which will be any of the views adjacent to a purple view. It knows that it was initially at view 230 and now at view 220 because it is the only view in the purple domain that is adjacent to a bright green view. This asymmetry in the shape helped the robot discover it's coarse location (wide domain in the map). The basic idea of this step is using Pattern of neighbors to resolve the conflicts when you have more than one domain mapped to the same pose.

The example above assumed hard association of poses to zones in the map. It assumes that the robot at can be matched to a specific domain with absolute certainty. The real world breaks this assumption as the domain matching is a coarse estimation with high uncertainty in the edges of the zone and in similar zones. The information that the robot has from the CAD models is too high level, the sensor data is subject to errors and inaccuracies and the feature extraction and matching is an estimation with possible uncertainty. This inacuracy introduces a probabilistic model to represent the localization problem that takes uncertainties into account by representing the location as the robot's belief of its location. The belief is a probability distribution over all the possible locations. The localization problem then is the problem of optimizing the estimation of the belief to be as close as possible to the right belief which has a peak at the right location and zero everywhere else. In our system, we use Gaussian mixture clustering model to allow for soft association of the domains.

The features extracted from the CAD model and the first image are used to provide the initial estimation of the location. This first estimation should provide a right lead to the robot even if it is not very specific. The correctness and certainty of this estimation is more important than the how specific it is. It should guide the robot to look in the most probable location and the robot will fine that initial estimation using the sensor data. To achieve that goal of correctness, the system will use only the high level features to provide a very coarse localization in a wide domain with high certainty rather than a narrow domain with low certainty. The use of high level features provides a faster computation for a coarse estimate (wide domains) with low certainty in a trade off with a fine estimate (narrow domain with high uncertainty.

View point maps

Building the viewpoint map is done in three steps. First, the system creates a rendered image of the object from the CAD models. Then, It takes screenshots of the rendered image from each view generated in the first step. The second step is extracting the high level features from these views. The last step is clustering the viewpoints.

A. Rendered Images

We rendered the shapes from the CAD models then we choose the views that we will extract features from. We tried 2 different view generation techniques to choose the different views of the object. The first technique rotates the object around one axis in rings. This enables detection of features along one axis from different angles. It is significance is in indicating symmetry along one axis. The other view generation technique rotates the image around three axises. This enables capturing all the features of the object from all the angles. For example, a sphere would have an identical

pose from all views while a cone would have an identical view along the y axis but would lock different from the z axis. The first technique would show both the sphere and the cone to have identical pose and features from all the views. the second technique would find the difference in the views along different axes. Figure 2(b) shows the 2D rendered image of a cone.. The system then takes screenshots of the object from different views and feeds them to the feature extraction step.

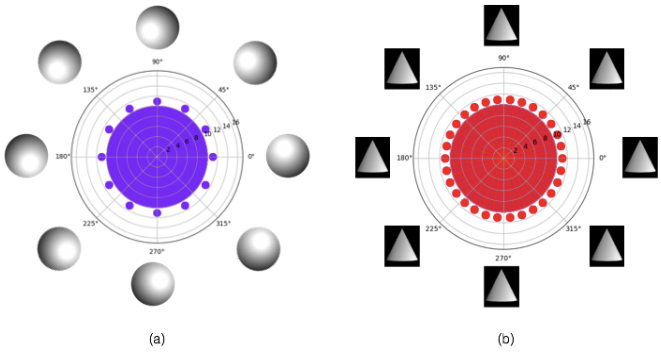


Fig. 2. Viewpoint map of (a) a sphere rotated around one axis using GMM. (b) a cone rotated around the one axis using GMM with the corresponding pose of the object at each view.

B. *High level feature extraction*

The second step is the feature extraction from the different views. The system needs to extract features that are clear enough to rely on differentiating different locations. The problem is that at a far distance, finer features are less visible and not very dependable to use as landmarks. The system uses high level features that identify the shape of the object which is clear from a far distance. The system extracts the perimeter, area of the object. It then finds the largest contour of the shape and finds its area, perimeter and the minimum bounding box. It also finds the convex hull of the shape. These features are used to create the feature vector for each view. The feature view contains perimeter of the object, perimeter of the convex hull of the object, aspect ratio of the object, filled Object to convex hull area ratio, area of the workpiece, area of the convex hull of the object.

C. *Clustering the view points*

The main goal of the view point map is to learn appearance of the object from the CAD model. After extracting the features, the system clusters the features' vectors representing the views. It then uses the clusters to build a viewpoint map of the object. We tried two different clustering algorithms to find the model that fits the data the best. The first clustering model is Gaussian Mixture Model (GMM). This model performed well as it doesn't provide hard association of views to clusters. It rather provides a probabilistic distribution that indicates the certainty of the clustering. The second option was Mean Shift model. It doesn't take the number of clusters as input but rather finds it on its own. The cluster size doesn't need to be even but the radius of each cluster needs to be the same. This advantage deals with models with no expected number of clusters.

IV. Gaussian model for localization

The localization problem is the problem of estimating the location given the prior information and the previous estimations. At the initialization phase, we need an initial estimation. After each observation, the estimation is updated to integrate the information learned from the observation. The noise in the observation leads to uncertainty in the estimation. The robot's certainty of its location may be very weak in the beginning of the estimation but it increases with more sensor data being integrated. To account for this uncertainty, the robot considers all the possible locations by representing the estimation as a probability distribution over all the possible locations. The robot's belief (estimation) of it is location is the probability it is in location y for all locations y.

Localization can be redefined to a bayesian estimation problem where for robot X, We need to calculate the probability it is at location y given the new observation for all possible y's.

Let Y be the set of all possible locations, we need to find the robot's belief of its estimation which is

P(robot at y | obs.) for all y belong to Y

We need to find an initial belief and a way to update the belief by action or by observation. As we have the CAD models, we can use the information provided in the CAD models to find an initial estimation. We will use the Gaussian distribution that resulted from the GMM clustering in the first step to as an initial estimate of our position in the viewpoint map.

After each observation, we will have to find

*P(y | obs.) = P(obs | y) * P(y) / P(obs) for all y belong to Y*

V. Results

The first experiment that I did was building the viewpoint maps using 2D views. I tried running it with both GMM and Mean-Shift algorithms. Table 1 shows the number of clusters in each shape. Figure 3 shows the viewpoint maps for the shapes using GMM clustering. The view point maps for the figures symmetric around the y axis (sphere, cone and cylinder) are one cluster because all of the viewpoints look exactly the same. For the figures that are not symmetric, we can see the similar views in the same cluster.

Figure 4 shows the viewpoint maps for the shapes using Mean-Shift clustering.

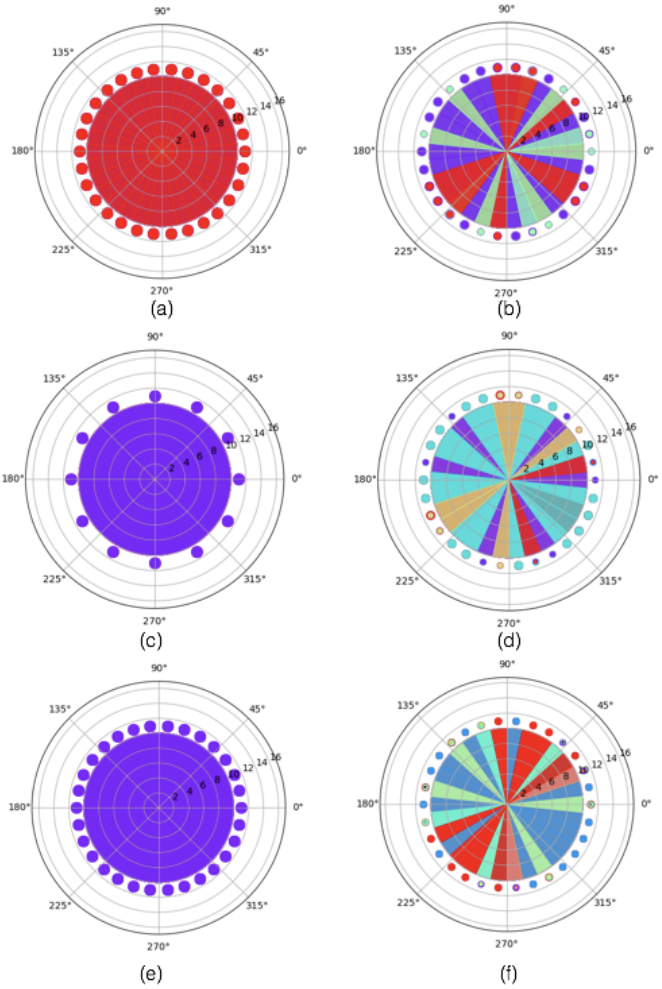


Fig. 3. Viewpoint maps using GMM of (a) a sphere. (b) a cube. © a cone. (d) a roof. (e) cylinder. (f) polygon

Conclusion and Future Work

This paper presented a system for stagerizing localization. Our system extracts high level features from CAD models and builds a viewpoint map of the object. The robot uses high level features from sensor data to coarsely localize itself in a domain in the map and then finds a fine estimation of its location in the specific domain. Coarse estimation results in a faster computation, memory efficiency and more accurate results than fine estimation. Also the viewpoint map built by the robot makes it easier for the human to understand how the system works.

While the importance of the extracted features differs from a domain to the other, the current systems extracts all the features and uses all of them for clustering with equal weights. One of the Future areas to explore is weighting the features in the domains.

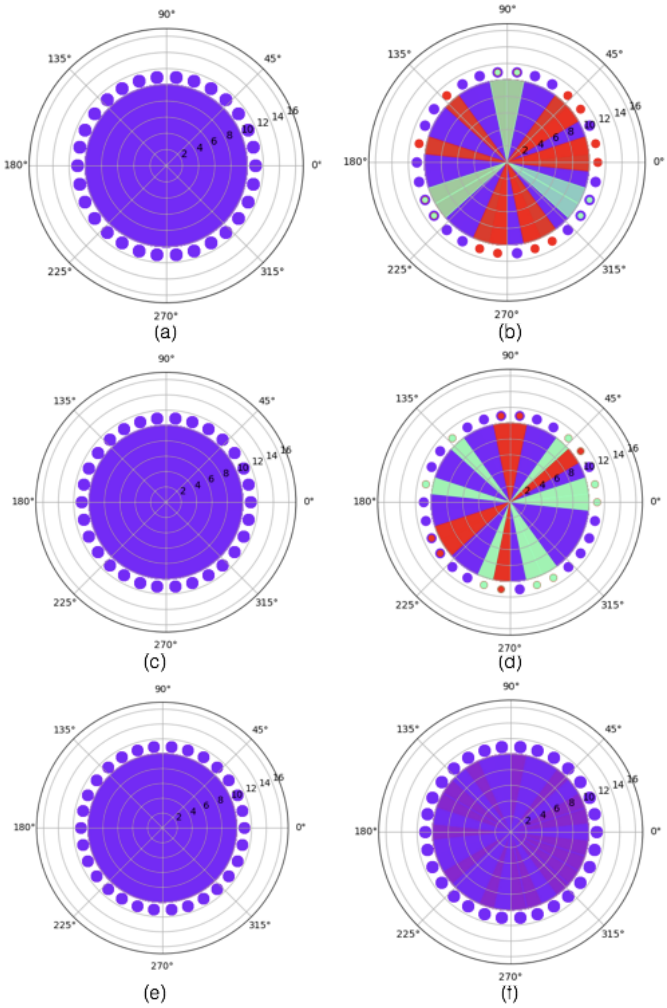


Fig. 4. Viewpoint maps using Mean-Shift of (a) a sphere. (b) a cube. © a cone. (d) a roof. (e) cylinder. (f) polygon

Acknowledgment

This research was done as part of the Robotics Institute Summer Scholars program 2017 (RISS 2017). I was sponsored by Carnegie Mellon University-Qatar to participate in the program.I was working in the Rapid Manufacturing Lab under professor David Bourne and Mabaran Rajaraman.

References

[1] Fox, D., Burgard, W., Dellaert, F., & Thrun, S. (1999). Monte carlo localization: Efficient position estimation for mobile robots. AAAI/IAAI, 1999(343-349), 2-2.

[2] Jensfelt, P., & Kristensen, S. (2001). Active global localization for a mobile robot using multiple hypothesis tracking. IEEE Transactions on Robotics and Automation, 17(5), 748-760.

[3] Lim, J. J., Pirsiavash, H., & Torralba, A. (2013). Parsing ikea objects: Fine pose estimation. In Proceedings of the IEEE International Conference on Computer Vision (pp. 2992-2999).

[4] Negenborn, R. (2003). Robot localization and Kalman filters. Utrecht Univ., Utrecht, Netherlands, Master's thesis INF/SCR-0309.

[5] Park, S., & Roh, K. S. (2016). Coarse-to-Fine Localization for a Mobile Robot Based on Place Learning With a 2-D Range Scan. IEEE Transactions on Robotics, 32(3), 528-544.

Inverse Kinematics of Soft Robots using Neural Networks

Daniele Moro[1], James M. Bern[2], and Stelian Coros[3]

***Abstract*— We present a method to solve inverse kinematics of tendon-activated soft robots that is accurate, generalizable, and model-free. This solution attempts to replace existing soft robot control methods that are unable to execute complex actions. We leverage a feed forward neural network that activates tendons given the desired location of feature points. The neural network is trained by sampling tendon activations and recording the corresponding feature point locations through the use of a finite element model simulation. We highlight the versatility of our system with three examples: a mesh controlled by specifying desired locations of feature points, a mesh controlled by specifying the overall shape, and a three-dimensional hand. We demonstrate that our solution shows promise while discussing the challenges associated with using a neural network to control tendon-activated soft robots.**

I. INTRODUCTION

Soft robotics is an emerging subfield of robotics which focuses on robots with a soft body without predefined points of articulation seen in traditional rigid robots. Soft robots have many advantages, such as an increased degree of safety because of their soft parts, an ability to move in unstructured environments, and an ability to handle fragile objects such as fruits for the agriculture industry. This makes soft robots promising in their future interaction with humans [1] [2].

Currently, soft robots have been used to make efficient jumping robots [3], simple octopus arms [4], and locomotive soft robots [5]. To achieve these results, most robots have been controlled through very simple methods, such as step-by-step instructions for activating certain tendons or pneumatics. More complex controls have been developed, but each has significant drawbacks which inhibit the versatility of soft robots [5] [6] . This is due to the fundamental problem that controlling soft robots is much less straightforward than controlling rigid robots.

In order to provide a generalized control of soft robots, we focus on inverse kinematics - finding the correct tendon or pneumatic activations to move a soft robot to a specific desired point or shape [7]. Solving this problem is more difficult than solving inverse kinematics for rigid robots because rigid robots exhibit high mechanical impedance. The inherently soft materials in a soft robot make the control problem much harder by introducing deformations, lag times, and vibrations. As Lipson et. all puts it, controlling a soft robot is "like controlling a marionette with rubber bands rather than with strings"[2]. Furthermore, a small change in a tendon contraction may result in a large displacement of a robots extremities, highlighting that solving an inverse kinematics problem on a soft robot requires the utmost accuracy. The focus of our work is to solve inverse kinematics on tendon-activated soft robots using a neural network that takes input from a user-defined goal, while considering how this solution could be used on physically realized soft robots.

To solve the given problem, we propose a real time control interface for tendon driven soft robots which leverages a feed forward neural network in order to translate user-specified target poses into corresponding real world signals. This work is general in the sense that network architecture and the overall training method is agnostic to the actual robots being controlled. We require only a finite element mesh, the boundary of which specifies the shape of the robot, and the actual layout corresponding to the real world tendon routing through the robot skin. This solution highlights high accuracy, reasonable speed, and versatile user control of the desired goal.

II. RELATED WORK

Currently, many soft robots have been controlled manually or with very simple control systems, such as flowcharts. Calisti et. all developed an octopus-inspired soft robot that was able to perform a myriad of actions in an unstructured environment. Using a flexible silicone arm driven via cables, the team was able to effectively demonstrate movements such as crawling, pushing, and grasping. This soft robot was controlled with a simple flowchart consisting of sequential actions such as activating a cable, waiting a certain period of time, and activating another cable. The controlling algorithm was exceptionally simple and performed the demonstrated simplistic tasks [4]. However, if Calisti et. all desired to make the octopus move in a much more intricate way, such as general manipulation, this control strategy would break down because it is not generalizable. Shepherd et. all created a soft locomotive robot inspired by animals lacking hard internal skeletons. This quadrupedal robot was able to crawl and undulate, allowing a navigation of various obstacles. To achieve their results, the robot was programmed with a simple sequential script of instructions to control solenoid valves. Although this control method functioned decently for simple movements, more complex, articulate, and generalizable tasks such as careful placement of its end effectors to climb uneven terrain requires a much more intelligent control strategy. The paper mentions that future

*This research was supported by National Science Foundation and the Robotics Institute Summer Scholars Program

[1]D. Moro is with the Computer Science Department, Boise State University, 1910 W University Dr, Boise, ID - 83725, USA. moro.daniele@gmail.com

[2]James M. Bern is with the Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA - 15213 USA. jamesmbern@gmail.com

[3]Stelian Coros is with the Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA - 15213, USA. scoros@gmail.com

work may include “the development of motion control systems [that]... may require the use of nonlinear models and may require neural-net-like learning methods ”[5]. More complex controlling mechanism are desperately needed as soft robots evolve beyond simple movements and into more generalizable actions, a gap which a neural network solution has the ability to address.

Bern et. all proposes a more complex control strategy to the inverse kinematics problem as related to tendon-activated soft robotics. They utilize a global optimization routine based on a cost function derived from current positions and corresponding desired positions. Using Newtons Method, the cost function is minimized to compute the optimal tendon contraction that reduces the distance between the current and desired robot locations. However, this approach requires a highly realistic model of a tendon-activated soft robot built with a finite element simulation. Unfortunately, it is extremely challenging to model friction, clumping, and other properties of stuffed structures. Therefore, a model-based approach may perform accurately in simulation, but poorly on a physical robot. A neural network solution may solve this problem because it has no need to use a model of the physical robot. A neural network can be trained using real world data, in the process learning how to account for factors such as friction, clumping, and more. In this way, a neural network may control physical soft robots more accurately by being model-free. [6]

Neural networks can be used to control rigid robots, as shown by Hesselroth et. all. The team created a rigid pneumatic arm controlled by a neural network which was able to move to a specified location with a high accuracy of about 3 mm. These results demonstrate that a neural network may be an effective solution for accurately solving inverse kinematic for rigid robotic arms. However, these results do not address how a neural network may be applied to structures lacking a high mechanical impedance, specifically soft robots. [8]

Giorelli et. all used a neural network to activate tendons on an octopus-inspired conical manipulator to move the tentacle to a given location. The team used a simple feed-forward neural network with one hidden layer composed of 34 nodes. With an average relative error with respect to the total length of the manipulator of 1.36%, they highlighted how a neural network can be effectively used to control a soft manipulator in simulation. However, only one feature point of the robot is inputted to the neural network: the tip of the manipulator. Furthermore, their work considers only one shape: a conically shaped manipulator. Without applications to a variety of shapes, the mathematical and practical solutions found are not easily generalizable [9]. Our work fills these gaps by implementing the use of multiple desired feature points located on the robot to determine desired shape and location and a method to control the entire desired shape of a soft robot. This drastically increases the complexity of the problem because the neural network needs to provide a solution to combinations of feature points so that it is as close to every point as possible, while not favoring certain points over others. Furthermore, the simulated model that we implement has a myriad of significantly different shapes and features, demonstrating that a neural network solution to the inverse kinematics problem has potential to be generalized to countless shapes, sizes, and tendon positions.

III. METHODS

In this section we present the simulation model that our solutions use, the structure of the neural network, the structure of the baseline nearest neighbor solution, and the ways in which we control desired mesh shape.

A. FEMSim

These solutions were created using a finite element mesh simulation (FEMSim) developed by Bern et. all [6]. FEMSim models a soft robot through an energy-based model. The overall energy of the mesh of a tendon-activated soft robot is calculated by accounting for the energy of each individual component. Newton’s method is then used to statically solve for a low energy state.

B. Neural Network

We use a neural network to solve the inverse kinematics problem for a tendon-activated soft robot in simulation. We implement the neural network solution in two stages: training and prediction.

1) Training: To train the neural network to predict tendon activations U given feature point locations X, we teach the network to map X to U. Therefore, the training stage involves providing the network hundreds of X to U mappings that we know are accurate. We sample the tendon activation space by iterating through every possible combination of tendon activations with a certain step size. From this sampling, the tendon activation(s) and corresponding feature point location(s) are printed to two files. 90% of the data is written to a training file, while 10% is written to a testing file. We use the training data to train the network, while the data in the testing file is used to evaluate the network and test for problems such as overfitting. Collection of this data is done in the C++ codebase of FEMSim.

The neural network is a simple feed-forward network consisting of 5 layers, each of which contains 30 nodes. The neural network trained for 300 epochs, or learning steps, through all the data in the training file. This solution was implemented using Python and Keras, a high-level API used to interface with TensorFlow. This neural network program reads the training and testing files in order to train the network and then evaluate the resultant loss, a measure of the inaccuracies of the prediction.

2) Prediction: When the main program (written in C++) is running the FEMSim simulation, we predict tendon activations given the desired feature point locations in real time using file input/output to communicate with an instance of the neural network program (written in Python). This neural network program instance is started at the beginning of the main program in order to give the neural network program time to import dependencies such as TensorFlow and Numpy.

If the neural network program is started every time the main program desires a new prediction, the time taken to compute the tendon activations is significantly increased. When the main program communicates to the neural network program that it requires a tendon activation prediction, the main program sends the location(s) of the desired feature point(s) as input. The neural network outputs the predicted tendon activations, which are read by the main program and implemented on the simulation.

C. Nearest Neighbor

We compare the neural network solution to a baseline nearest neighbor solution. The desired feature point locations are inputted to the nearest neighbor function, which then finds the distance from the inputted desired locations to every point stored in the sample space. We select the points in the sample space nearest the desired location, and we activate the tendon activations corresponding to the selected point.

This solution requires a dense sampling to work accurately. Sampling of the space takes s^t samples, where s is the number of sample activations per tendon, and t is the number of tendons. Even when $s = 10$ and $t = 4$, 10,000 samples are required. This demonstrates how sampling of the tendon activation space increases exponentially with density and number of tendons.

A neural network is an excellent candidate for solving this inverse kinematics problem because it does not require a dense sampling to work accurately. We show how a neural network performs well when $s = 5$ and $t = 4$, only 625 samples.

D. Controlling Desired Mesh Shape

Inside the simulation, a user may drag around desired feature points, and the simulated soft robot moves in real time to place the feature point as close as possible to the user-defined desired feature point locations by activating its tendons. By dragging multiple desired points, the user can instruct the soft robot to assume a myriad of possible shapes that better resemble the desired shape that the user intended as compared to simple one desired point.

The neural network solution can easily account for multiple feature points by training the data on select multiple feature points. However, using this system, the user cannot spontaneously add new feature points because this would change the number of inputs to the neural network: i. e. If the neural network expects 4 inputs of x positions and y positions of two feature points, a user cannot include another feature point because this would change the number of inputs to 6. This problem can be addressed by re-training the network to the number of feature points desired by the user. However, this solution is cumbersome and computationally expensive. Instead, we attempted to solve the problem of multiple dynamic feature points with two solutions: masking and extracting feature points after statically solving the mesh.

A straightforward approach to creating a neural network that accepts a dynamic number of inputs is masking. This means that after each feature point input to the network, a third input is created. This input is either a one or zero, one if the feature point should be considered, and zero if it should not be considered. In this way, unwanted feature points are masked, while feature points that the user want to be considered are considered. Unfortunately, this is difficult to create in practice because the neural network has difficulty understanding the meaning behind the masking input.

Another approach to creating a neural network that accepts a dynamic number of inputs is to extract select feature points after statically solving a user-defined pose. In order to run predictions, the user drags any point on the mesh to create a desired shape. While the user is dragging the mesh, we run static solves in real time to force the user to create feasible mesh shapes. We then extract the position of select feature points and input them into the neural network. The neural network predictions are then used to activate the meshs tendons.

In simulation, we can bypass the neural network by simply reading the tendon activation whenever the user drags the mesh into the desired shape. However, the neural network is necessary when this work will be eventually applied to a physical tendon-activated soft robot. The model itself can be used to shape the robot to a desired form, but the corresponding tendons activations may be inaccurate because the model in simulation is not a perfect rendition of a physical tendon-activated soft robot. By training the neural network with real world data, it is able to understand the mapping from X to U compensates for variables such a friction and clumping, which the model ignores. Therefore, we statically solve the mesh so that the user can easily create a desired shape that is likely to be feasible. The feature points are extracted so that the neural network can provide an accurate mapping from X to U.

E. Two or Three Dimensions

This approach functions well in both two and three dimensions. The inputs are simple collections of positions, and the energy function of the simulation remains the same. The only major difference is that the feature point positions are composed of x, y, and z positions rather than solely x and y inputs with two dimensions.

A two dimensional shape is discretized through user control of a boundary and largest possible triangle size.

In order to take a three-dimensional shape and place it into the simulation, we use a program called TetGen to tetrahedralize a .stl file. A separate Python program parses the tetrahedralized output and rewrites it in the format required by the FEMSim program.

IV. RESULTS

We present three examples to demonstrate the accuracy and versatility of our neural network solution. The first example highlights how directly controlling desired feature points performs more accurately than a nearest neighbor approach, while achieving the user-specified goal with reasonable accuracy. The second example suggests that controlling a mesh by extracting feature points from another user-controlled mesh

is a viable and effective method to allow dynamic feature points. This gives the user further control over the desired shape of the soft robot. Finally, we demonstrate that the neural network solution can be applied to three-dimensional models with relative ease and high accuracy. This emphasizes the versatility and utility of the solution.

A. *Controlling desired feature points*

The simplest example involves allowing the user to directly control feature point locations. We demonstrate the use of a neural network to control tendon activations to arrive at user-defined desired points while comparing this solution to a baseline nearest neighbor solution.

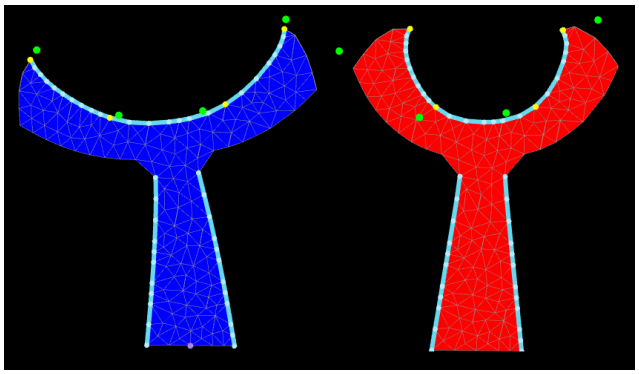


Fig. 1. A neural network controlling tendon activations on the blue mesh is able to approximate user-defined desired feature point locations with higher accuracy that a nearest neighbor solution on the red mesh.

Fig. 1 illustrates this example. The blue mesh is controlled by a neural network, while the red mesh is controlled by a nearest neighbor solution. The yellow dots on both the blue and red meshes are the current feature point locations for each mesh respectively. The green dots are the desired feature points. Both the neural network and the nearest neighbor solution control the light blue tendons in order to place the yellow feature points on their own meshes as close as feasibly possible to the green desired points, which the user may drag around.

As we illustrate in Fig. 1, the blue mesh controlled by a neural network performs considerably better than the nearest neighbor solution. This is especially apparent when considering the difference in location of the upper left part of the T-shaped mesh. Furthermore, the neural network smoothly moves around the simulation, while the nearest neighbor solution snaps from one position to another. Smooth motion is important when a soft robot intends to grab something or move from one location to another continuously.

Furthermore, when the density of the sampling is reduced, the neural network behaves exponentially more accurately than the nearest neighbor. This is because the neural network is able to interpolate between the points, while a nearest neighbor approach does not have this functionality. The ability to function accurately without a dense sampling is important when the tendons are numerous enough that a dense sampling would run in an infeasible time.

B. *Controlling desired shape*

To give the user more control in how they desire the final mesh to be shaped, we allow the user to control the shape of the entire desired mesh shape rather than a select few feature points.

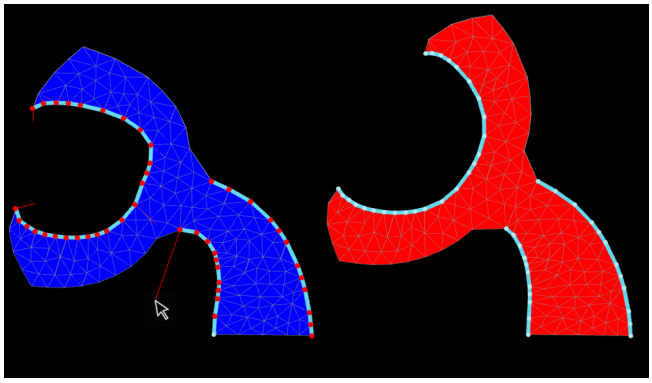


Fig. 2. The user is able to control a desired mesh shape on the blue mesh. A neural network reasonably approximates the desired shape by extracting feature points and activating tendons on the red mesh

As shown in Fig. 2, the user is able to click and drag and point on the blue mesh and place it in a desired position. While the user interacts with the simulation to create a desired pose, the blue mesh is statically solved every frame so that the user is softly forced to create feasible shapes. The red dots on the blue mesh represent the feature points we extract and then input into the neural network. The neural network outputs tendon activations that are applied on the light blue tendons seen on the red mesh.

We show that this solution behaves accurately and in real time in Fig. 2. The mesh controlled by the neural network is able to successfully approximate the desired blue mesh to a reasonable degree. Although the blue mesh is positioned in a way to is infeasible for the red mesh to attain simply through a contraction of tendons, the neural network finds the most reasonably close solution.

This same solution may be used for a physically realized tendon-activated soft robot, except that the tendon contractions predicted by the neural network are activated on a physically realized robot instead of in simulation. If the neural network is trained on real-world data, it would be able to control the realized robot with high accuracy and without the need of a model.

C. *Controlling a hand in three dimensions*

The neural network solution can be used to control countless shapes and sizes of soft robots. It can also be used to control a three-dimensional mesh of a hand, as seen in Fig. 3.

Fig. 3 demonstrates how the hand shown on the left is controlled by the user and the hand shown on the right is controlled by the neural network. The user clicks and drags the yellow points on the tips of the fingers on the hand shown on the left. The feature points are extracted from the hand shown on the left and fed into the neural network,

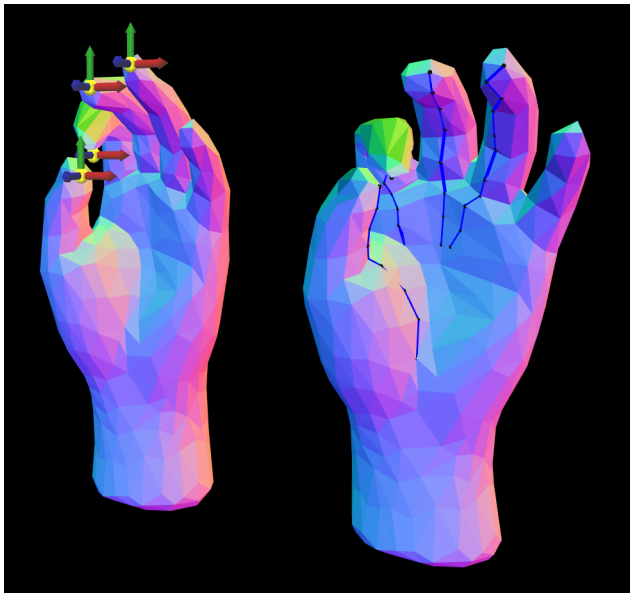


Fig. 3. The user is able to control a three-dimensional desired mesh shape on the hand shown on the left. A neural network reasonably approximates the desired shape by extracting feature points and activating blue tendons on the hand shown on the right.

which outputs tendon contractions. The blue tendons on the hand shown on the right are contracted to approach the user defined shape.

We observe that the neural network solution performs accurately and in real time to control the soft hand. This solution could be used to control a physically realized soft hand to grab objects, or to control a locomotive three-dimensional soft robot such as an octopus to move in fluid and complex ways.

V. DISCUSSION

A. Difference between a user controlled mesh and a neural network controlled mesh

The fundamental difference between the mesh controlled by the user and the mesh controlled by the neural network is that the user drags points on the mesh to create a desired shape, while the neural network contracts tendons. By virtue of the physical properties of the mesh, translations of points produce fundamentally different shapes than contractions of tendons. For example, given that a user desires a mesh in the shape of a long bar with tendons on both long ends of the bar to contract. If the user simply attempts to drag the ends of the bar together, the bar will bend in the middle in order to deform as little as possible. This would indicate to the user that their desired contraction is impossible, even though the neural-network controlled-bar could easily contract by simply reducing the length of both tendons equal amounts.

We cannot resolve the problem easily, and we will need to address this problem depending on the purpose behind a user's intent to create a certain pose. For example, if a user intends to use a hand-shaped mesh to grasp a ball, the shape of the fingers is irrelevant given that the neural-network controlled mesh moves the tips of the fingers to the surface of the ball. In this example, we would place the feature points on the tips of the fingers because the shape of the rest of the finger is agnostic to achieving the intended action.

B. Training the neural network on real-world data

The entirety of the work we demonstrate in this paper concerns controlling a soft robot in simulation. However, our neural network solution has significant promise to perform accurately and quickly on physically realized tendon-activated soft robots. If we train a neural network with shape to tendon mappings obtained through a physically realized tendon-activated soft robot, we predict that the neural network would be able to predict new tendon activations in a way that would compensate for real-world variables such as clumping, friction, and imperfectly flexible materials. This is because the neural network solution is fundamentally a model-free approach.

Therefore, a neural network could control a physically realized tendon-activated soft robot much more accurately and quickly than most model-based approaches.

C. Redundancy

An essential problem in controlling tendon-activated soft robots is the potential for multiple solutions to a given desired goal. In our tests, we notice that the neural network is able to manage such redundancies by simply finding a solution that minimizes the deformation of the soft robot to achieve the required objective.

VI. CONCLUSION

A myriad of interesting soft robots have been created, but most have been controlled through simple and inflexible methods because soft robots are fundamentally difficult to control. We presented a neural network based approach to solving inverse kinematics for tendon-activated soft robots. Our method filled the gap of oversimplified and limited strategies with a generalizable and real time approach. We demonstrated how this neural network solution is effectively used on three examples: controlling desired feature points, controlling desired mesh shape, and controlling a hand in three dimensions. We discussed the difference between a user controlled mesh and a neural network controlled mesh and explained how this training the neural network on real-world data is a promising next step.

Future work may include building a physically realized tendon-activated soft robot, training a neural network with real-world data, and testing performance. We plan to build a variety of three-dimensional soft robots to test the neural networks ability to control complex soft robot actions.

ACKNOWLEDGMENT

We would like to thank Dr. John Dolan, Mrs. Rachel Burcin, and the Robotics Institute Summer Scholar Program for their help. Thank you to the National Science Foundation for funding this research

References

[1] H. Lipson, "Challenges and opportunities for design, simulation, and fabrication of soft robots," *Soft Robotics*, vol. 1, no. 1, pp. 21–27, 2014.

[2] J. Rossiter and H. Hauser, "Soft roboticsthe next industrial revolution," *IEEE Robot. Autom. Mag*, vol. 23, pp. 17–20, 2016.

[3] S. Li, R. Katzschmann, and D. Rus, "A soft cube capable of controllable continuous jumping," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pp. 1712–1717, IEEE, 2015.

[4] M. Calisti, M. Giorelli, G. Levy, B. Mazzolai, B. Hochner, C. Laschi, and P. Dario, "An octopus-bioinspired solution to movement and manipulation for soft robots," *Bioinspiration & biomimetics*, vol. 6, no. 3, p. 036002, 2011.

[5] R. F. Shepherd, F. Ilievski, W. Choi, S. A. Morin, A. A. Stokes, A. D. Mazzeo, X. Chen, M. Wang, and G. M. Whitesides, "Multigait soft robot," *Proceedings of the National Academy of Sciences*, vol. 108, no. 51, pp. 20400–20403, 2011.

[6] J. Bern, G. Kumagai, and S. Coros, "Fabrication, modeling, and control of plush robots," *Submitted*, 2017.

[7] L. Josuet, B. Carlos, L. Hsien, H. Te-Sheng, W. Chun-Sheng, *et al.*, "An improved inverse kinematics solution of 6r-dof robot manipulators with euclidean wrist using dual quaternions," in *Automatic Control Conference (CACS), 2016 International*, pp. 77–82, IEEE, 2016.

[8] T. Hesselroth, K. Sarkar, P. P. Van Der Smagt, and K. Schulten, "Neural network control of a pneumatic robot arm," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 24, no. 1, pp. 28–38, 1994.

[9] M. Giorelli, F. Renda, G. Ferri, and C. Laschi, "A feed-forward neural network learning the inverse kinetics of a soft cable-driven manipulator moving in three-dimensional space," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pp. 5033–5039, IEEE, 2013.

Full Stack Swarm Architecture

Kyle Morris, Gabriel Arpino, Sasanka Nagavalli, *Student Member, IEEE*,
Katia Sycara, *Fellow, IEEE*

***Abstract*— Robot swarms are homogeneous multi-robot systems that form collective behaviour from decentralized local interactions. Swarms are a favorable choice for solving various problems in robotics as they are robust and fault tolerant in nature, the individual swarm agents themselves being cost effective alternatives to the solution. Developing swarm technology aids in large scale data collection and environmental exploration, package delivery, warehouse management, military reconnaissance, and search and rescue. With the growing interest in developing swarm systems, much researcher and developer time is spent migrating, integrating, and coordinating these various swarm solutions that become deprecated and isolated from the community in frequent cycle. Furthermore, despite the aforementioned swarm architectures provided insights into specific design considerations, there still lacks a generalized architecture that outlines a full swarm pipeline, and is modular enough in design to be readily interchanged with new components as both research and industry advance. We present an architecture for developing full stack swarm systems. Such a system must allow for easy design, deployment, interaction, and evaluation. Using our proposed architecture, we then implemented a framework titled: CMUSWARM, on the ROS platform using Gazebo with irobot create for simulation. We then conduct an evaluation of our architecture by comparing two simple swarm control-laws within the framework.**

I. Introduction

Robot swarms are homogeneous multi-robot systems that form collective behaviour from decentralized local interactions. Swarms are robust and fault tolerant in nature, with individual swarm agents being inexpensive. As individual mobile robots become more robust to real-world conditions [1][6], there is growing interest in swarm robotics [2][5][7]. Developing swarm technology aids the data science industry by allowing for large-scale data collection and environmental exploration [9]. Swarms also show promise in object transportation [10][8], military reconnaissance[15], and search and rescue [13][14].

Multi-robot and swarm algorithms have been developed over the years addressing problems such as navigation [16][19], exploration[20], and coverage [18]. Middleware platforms such as ROS, and Player [30] have an associated community of developers and their software that enables these algorithms to be deployed on a variety of robots [12]; however there is a need for separate bench-marking components in order to evaluate performance. Simulation tools often come prepackaged with performance evaluation tools, and have accelerated the community by allowing multi-robot and swarm systems to be visualized without the complexity and cost of real world experimentation. Platforms include Argos [21], SwarmSimX [22], Menge [23], MATLAB MRSim [26], and Stage [30]. Without middleware such as ROS, these simulators do not allow for deploying swarm algorithms on real robots. The TeleKyb framework merges the ROS middleware with the SwarmSimX simulator using MATLAB/Simulink to provide both the ability to operate a swarm, and to evaluate swarm performance [25]; however this framework is narrow in focus on particular UAVs, and has quickly become deprecated as of ROS fuerte without generalizing the design to other swarms. Recent development of abstraction layers attempt to generalize from specific swarm frameworks to more broad architectures of swarm systems. Current swarm architectures support interaction with a swarm; but are constrained to centralized algorithms [24]. Other designs support both decentralized and centralized swarms but are designed primarily around one aspect of a swarm system such as communication [27], and don't enable interaction with the swarm, or evaluation of performance [31].

This highlights a significant obstacle in the swarm robotics community: the lack of a generalized design pattern (architecture) for creating these swarm systems. Much developer and researcher time is thus spent following a trail of deprecated software and migrating between separate esoteric components in hopes of creating hybrid software for their particular use-case. Furthermore, when a framework is showing great potential in the community [25], the sufficient components in the given framework are not abstracted away to highlight necessary components for any swarm system. As a result, such contributions become isolated for one particular application and deprecated in short time.

We present an architecture that provides a generalized solution for developing full stack swarm systems. A full stack swarm system must allow a developer to **a) design** of a swarm behaviour library by integrating existing centralized or decentralized algorithms, **b) deploy** swarm algorithms on various real robots **c) operate** the swarm through human control and **d) evaluate** algorithm performance through various metrics. Using our proposed architecture, we then implemented a framework titled: CMUSWARM, on the ROS platform using Gazebo with irobot create for simulation. In section II, we outline the architecture design, and then describe our implementation on the ROS platform in section III. Section IV demonstrates the usage of our Framework by conducting an experimental evaluation of two decentralized swarm algorithms We then discuss the various decisions of our work and future research direction.

II. Architecture Design

In this section we cover the architecture components and design decisions. We design by keeping closely in mind the use cases of operators and developers in the swarm robotics community. An operator will be primarily concerned with sending behaviour requests to a swarm and configuring such behaviours, whereas a developer is most concerned with integrating new algorithms and components seamlessly. As such, modularity and simplicity are the fundamental design philosophy.

A. High Level Overview

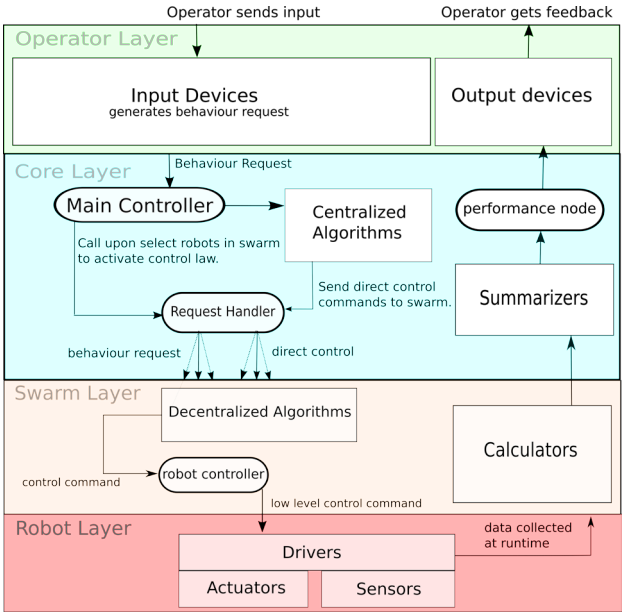


Fig. 1. High level overview of the architecture

The architecture consists of four layers (operator layer, core layer, swarm layer, and robot layer). Both inter-layer and intra-layer information is communicated asynchronously through nodes using a distributed publisher subscriber pattern [4]. This pattern favors the decentralized nature of a swarm in that it provides scalable communication with low coupling. It is important to note that while the swarm algorithms may be decentralized, they are deployed from a centralized core layer. Each layer is also assumed to have a local database in which any node in the layer may access at runtime to gather configuration parameters. The following sections will further outline each layer, it's configuration, and design considerations.

B. Operator Layer

The operator layer allows for interaction with the system. An external input device must act as an adapter to the core layer by generating a behaviour request. A behaviour request is a message consisting of a behaviour name, which corresponds to the algorithm to deploy in the swarm system, along with a list of robot ids, that identify the robots in the swarm to control with this request. Further behaviour specific information may be appended to the request through key-value pairs at run-time. An example of an input device is a joystick. Moving the joystick left may translate to a behaviour request named move_left, a list of all robot ids in the swarm, and additional velocity information with magnitude depending on the joysticks angle of offset. This message will be sent to core layer and received by the main controller.

C. Core Layer

The core layer receives behaviour requests and determines how to toggle the requests upon the swarm.

1) Main Controller: The main controller is a node that first receives the behaviour request. The behaviour name is looked up in a local database for an associated record. If the behaviour name is valid, then static information will be appended to the request from its record. In the example of a joystick generating a move_left_bc request, this will be looked up by the main controller. The barrier certificates portion of the request requires additional information such as robot collision radius, which will be appended from the database to the behaviour request. Each behaviour record in the database must also specify if it is centralized or not. In the case of a centralized behaviour, the updated behaviour request will be remapped to a centralized algorithm node on the topic behaviour_name/control_request. A decentralized behavior request will be delegated to the request handler directly, to be further spread out to the swarm.

2) Request Handler: The request handler receives behaviour requests from the main controller and publishes the request to each robot listed in the request. Alternatively, a centralized node will compute an explicit command for each robot. The request handler will distribute each command to a corresponding robot in the swarm. The request handler also ensures that the swarm robots have disabled active behaviours prior to toggling a new one. In the event that multiple behaviours are requested rapidly, the request handler will schedule them across some time interval.

D. Swarm Layer

The swarm layer is decentralized and thus refers to each robot separately in the swarm. This layer contains a node for each supported algorithm (ie behaviour) that the robot can perform. When each robot in the swarm performs the same desired algorithm, the global behaviour emerges. Each robot falls into its own topic namespace, and as such receives commands to its local node on /{robot_id}/{behaviour_name}/control_requests. Each algorithm node in this layer will publish the computed velocities to a robot controller, which further delegates to the robot layer. In the case of a centralized algorithm, a direct command is propagated to this robot controller.

E. Robot Layer

The robot layer contains the drivers and other on-board software for operating the swarm robot. In the case of a simulation, the robot layer may redirect output commands

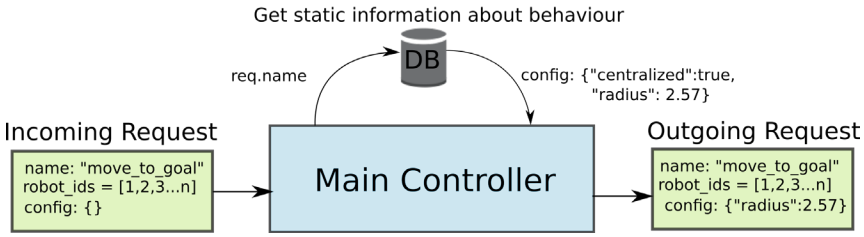


Fig. 2. The process of gaining information about incoming messages to the main controller.

to a separate program which will then simulate the robot's trajectory.

F. Performance Evaluation

This performance component is spread across the robot, swarm, and core layers of the architecture using a series of what we denote as calculators, and summarizers. To outline how performance metrics are calculated by propagating information through the system, we will use the example of a work summarizer.

1) Calculators: Each robot may contain a collection of calculators, located in the swarm layer. Each calculator reads information gathered during a trial. For example, a work summarizer may use odometry to track the distance a robot has travelled. The calculator will then publish results to a corresponding summarizer. It is important to note that in a real-world evaluation, the onboard sensors of a robot will be inaccurate, and thus the calculators would publish inaccurate data. We acknowledge this; and made this decision choice despite it. During a real-world evaluation, if there exists some supervisory process monitoring each robot in the swarm, it may act as the calculator and communicate whatever precise information it has captured. There exists inaccuracy in any real-world sensor, thus we are not concerned with this matter in our performance evaluation. Furthermore, tasking each robot with providing performance calculations allows for effective debugging of a real-world system, as inconsistencies may be found between measured, and predicted data.

2) Summarizers: Summarizer nodes are centralized in the core layer and receive calculations from each active robot in the swarm. The summarizer then evaluates this information and organizes it. In the case of a work summarizer, it would collect each robots indivdiual distance information, and compute the sum and average to give insights about the whole swarms travel. Each summarizer must have access to termination conditions stored in the core layers local database. These termination conditions ensure the summarizer will either halt after a maximum execution time, or when some other termination condition is met.

3) Performance Node: A single performance node subscribes to each summarizer, and will save performance summaries in a readable format. This may be in a database, excel spreadsheet, etc. Furthermore, results can be mapped to an output device and shown to the operator.

III. Framework Implementation

We present a framework titled: CMUSWARM, which implements the aforementioned architecture. The framework was implemented on the ROS platform and tested with both indigo and kinetic. ROS provides the publisher and subscriber pattern required along with a parameter server that acts as the database for the system. Gazebo was used as a simulation platform with iRobot create sdf modelled robots, however the framework may be used on real robots supported by ROS compatible drivers.

Three performance metrics have been implemented.
work: track the overall distance travelled by all robots in swarm
coverage: using the gridmap_ros package [28], this will track how much of the map was covered by the swarm. Each robot covers the area created by a disk with the radius of the robot.
collisions: tracks the number of collisions a robot has with nearby obstacles, assuming it can read in the environmental information.
Experimental trials are run by having a supervisory process fork a roslaunch instance of the framework. The launch file is provided with arguments generated dynamically by the supervisory program, which specifies where robots and obstacles are located, as well as the termination conditions and any other arguments required for an instance of the framework. Once the framework is launched, a mock_publisher node will sent a behaviour request to activate the algorithm in question. The performance is then evaluated by the work, coverage, and collision calculators until their corresponding summarizers terminate (either by termination condition, or by max_execution time). The results are saved and the supervisory monitor will kill the child framework instance and create a new one.

A. Algorithms and Dynamics

In order to demonstrate and verify the functionality of the framework and implementation, a number of swarm control algorithms were implemented and some benchmarked on the machine. They are also used as a ground-truth comparison for any future work we do using CMUswarm. The algorithms currently implemented are:

- Move to goal
- Flocking
- Rendezvous
- Formation
- Voronoi static coverage

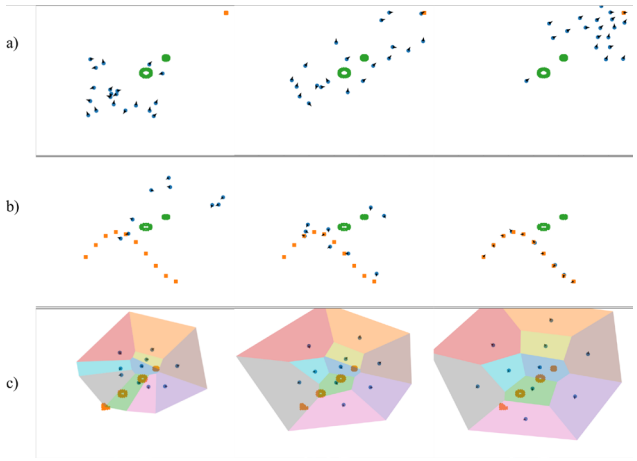


Fig. 3. Our independent script implementation of common control laws outside of the Framework. a) The move_to_goal behaviour with barrier certificates, b) formation control with barrier certificates c) voronoi static coverage

The details and guarantees for the algorithms are detailed below.

The robot model used for all algorithms was the following:

$$\begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{\theta}_i \end{bmatrix} = \begin{bmatrix} cos(\theta_i) & 0 \\ sin(\theta_i) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_{i,v} \\ u_{i,w} \end{bmatrix}$$

where v and w are the control linear and angular velocities, respectively. Following the definition in [3], we represent the ith robot by its position vector $\boldsymbol{p}_i \in \mathbb{R}^2$ and heading vector in the direction of θ_i, $\boldsymbol{b}_i \in \mathbb{R}^2 : \|\boldsymbol{b}_i\|_2 = 1$.

Remark: The framework is not restricted to this robot model, each algorithm is free to utilize any desired robot model.

Let S represent the set of robots used in simulation. We calculate the change in heading and update u_v and u_w depending on the desired velocity pointed towards the target, $\dot{\boldsymbol{p}}$:

$$u_{i,v} = K_v * (\boldsymbol{b}_i)^T \dot{\boldsymbol{p}}$$

$$u_{i,w} = K_w * w$$

The algorithms differ in their calculation of $\dot{\boldsymbol{p}}$ and w. The velocity relations are discretized into difference equations, outputting a $\boldsymbol{dv}$ vector from an input of $\boldsymbol{p}_k$, $k \geq 0$.

The robot dynamics are updated according to $\boldsymbol{v_{k+1}}$:

$$\boldsymbol{v_{k+1}} = \boldsymbol{v_k} + \boldsymbol{dv}$$

$$d\theta = atan2(\frac{\boldsymbol{v_{k+1}}[1]}{\boldsymbol{v_{k+1}}[0]}) - \theta_k$$

$$w = atan2(\frac{\sin(d\theta)}{\cos(d\theta)})$$

$$\boldsymbol{b} = \begin{bmatrix} \cos(\theta_k) \\ \sin(\theta_k) \end{bmatrix}$$

The linear and angular velocities are capped at a maximum:

$$u_v = \max(-U_{v_{max}}, \min(U_{v_{max}}, u_v))$$

$$u_w = \max(-U_{w_{max}}, \min(U_{w_{max}}, u_w))$$

The positions of the robots are then updated to follow the dynamics:

$$\boldsymbol{p_{k+1}} = \boldsymbol{p_k} + \begin{bmatrix} u_v \cos\theta dt \\ u_v \sin\theta dt \end{bmatrix}$$

$$\theta_{k+1} = wraptopi(\theta_k + u_w dt)$$

where the $atan2$ function is a four-quadrant inverse tangent, and $wraptopi$ is a function used to wrap angles in radians to $[-\pi, \pi]$.

It is important to note that every algorithm performed inter-robot and obstacle collision avoidance in the form of barrier certificates [11], a obstacle avoidance method chosen due to its available collision avoidance guarantees on holonomic agents and its minimally invasive nature which allows for obstacle avoidance only when absolutely necessary. It is achieved using a dynamic programming approach outlined in [11]. Every algorithm presented therefore possesses an underlying control in the form of barrier certificates that only intervenes when it is absolutely necessary in order to avoid a collision, and it does so while minimizing the difference between the avoidance trajectory and the original intended trajectory. Due to our non-holonomic robot model, the barrier certificates algorithm still allowed for collisions to happen in simulation, though less than in the no obstacle avoidance case (see Figure 4).

1) Move to goal: The move to goal algorithm involves a simple proportional controller that leads the robot towards the desired position. The algorithm consists of a discrete loop of the following differential equation:

$$\dot{\boldsymbol{p}} = -K_i(\boldsymbol{p} - \boldsymbol{E})$$

where d is the distance of the robot to the goal position E, $K_i > 0$. The solution

$$\boldsymbol{p}(t) = \boldsymbol{p}(0)e^{-K_i t} + \boldsymbol{E}$$

guarantees convergence to the desired point $\boldsymbol{E}$ since K_i is positive definite. This is approximated using a difference equation:

$$\boldsymbol{dv} = -K_i(\boldsymbol{p_{i,k}} - \boldsymbol{E})$$

The dynamics are then updated according to the robot model. An independent script implementation of this algorithm is seen in Figure 3 a).

2) Flocking: The flocking algorithm is inspired by the known boids algorithm [17]e and [3], using the three steers of separation, cohesion, and alignment. Specifically, the following control laws were utilized for each steer:

Remark: The distance vector from agent i to agent j is noted as $\boldsymbol{d} = \boldsymbol{p}_{ij} = \boldsymbol{p}_j - \boldsymbol{p}_i$.

Separation

$$\boldsymbol{dv} = -\frac{\boldsymbol{d}}{\|\boldsymbol{d}\|_{\boldsymbol{2}}}$$

$$d\theta = atan2(\frac{\boldsymbol{dv}[1]}{\boldsymbol{dv}[0]}) - \theta_i$$

Alignment

$$\boldsymbol{dv} = \boldsymbol{0}$$

$$d\theta = \theta_j - \theta_i$$

Cohesion

$$\boldsymbol{dv} = \boldsymbol{d}$$

$$d\theta = atan2(\frac{\boldsymbol{dv}[1]}{\boldsymbol{dv}[0]}) - \theta_i$$

The robot positions are then updated according to the previously stated dynamics, and each control law is activated once $\boldsymbol{d}$ reaches certain physical thresholds of separation, alignment, and cohesion defined by the user, termed $repulsion\_radius, alignment\_radius, attraction\_radius$ in our implementation.

3) Rendezvous: This control law is described in [3] and controls the robots such that they meet at a common point. Define $\boldsymbol{d}$ as before: $\boldsymbol{d_{ij}} = \boldsymbol{p}_{ij} = \boldsymbol{p}_j - \boldsymbol{p}_i$. The following update is provided to $\boldsymbol{v}$ in order to produce the rendezvous behaviour:

$$\boldsymbol{dv_i} = \boldsymbol{dv_i} + \boldsymbol{d_{ij}}, \forall j \neq i \in S$$

$\boldsymbol{v_i}$ is then normalized by $\|S\|$ and the dynamics follow from above.

4) Formation: In the formation control algorithm, each agent is provided with a target position E_i, and they follow a simple proportional controller towards this goal. Recall obstacle avoidance and inter-robot collision avoidance are handled using barrier certificates.

$$\boldsymbol{dv} = -K_i(\boldsymbol{p_{i,k}} - \boldsymbol{E_i})$$

And the dynamics follow. An independent script implementation of this algorithm is displayed in Figure 3 b).

5) Voronoi Static Coverage: The voronoi static coverage algorithm addresses the multi-robot problem of maximizing static coverage over a certain area. The solution implemented follows that of [Lloyd's algorithm] where a controller moves the robots to the centroid of their respective voronoi regions (denoted as $\boldsymbol{C_i}$). The voronoi regions were constructed for each robot, and the following controller was then applied:

$$\boldsymbol{dv} = -K_i(\boldsymbol{C_i} - \boldsymbol{p_{i,k}})$$

followed by the dynamics update detailed above. The voronoi static coverage method outlined by Lloyd's Algorithm guarantees convergence to the optimal static coverage configuration for holonomic robots. An independent script implementation of this algorithm is displayed in Figure 3 c).

The algorithms were successfully implemented in our framework and they cover different important areas of multi-robot algorithms: coverage, formation, and navigation, all with obstacle avoidance. It is shown in simulation that the framework is well-adept for rapid porting and automated benchmarking of such multi-robot algorithms. Figure 3 highlights a simple independent prototype implementation of three of the above algorithms. We initially wrote the controllers independently from the framework, as seen in Figure 3, and we later integrated them into the framework to evaluate the ease of integration.

IV. Experimental Evaluation

We evaluate the performance component of the framework using gazebo_ros on the ROS indigo and kinetic. Two navigation control laws, namely move to goal, and move_to_goal_bc are compared across four scenarios using 4, 8, 16, and 32 robots for 20 trials each totalling 640 trials. The purpose was to highlight the use of the framework for evaluating swarm algorithms.

A. Setup

The swarm is homogeneous and each robot represented as an irobot create sdf file in gazebo. Obstacles are static and modelled as 1x1x1 boxes. A swarm robot has a sensing radius of 3 meters, as read from gazebo model positions, and is assumed to have perfect sensing ability to obtain local information. The robots max velocity is 4 meters/sec, and maximum angular velocity is pi/4 radians. Four environment scenarios are used, each a 20x20 meter world generated in gazebo. The swarm robots are spawned in randomly generated locations with some specified region, and must navigate to a standard goal region within 1 minute.

Scenario 1: Empty World: This world is empty, with robots spawning in the upper plane defined as $R1 = (x, y)$ such that $x \in [1, 6], y \in [1, 19]$. This is a minimal Scenario to serve as easy comparison.

Scenario 2: Empty Dense World This world is also empty, with a smaller spawning region for robots defined as $R2 = (x, y)$ such that $x \in [1, 6], y \in [1, 6]$. The goal region G is a disk of radius 4 centered at $(17, 17)$. This scenario explores algorithm performance when swarm robots begin within close proximity of each other. All other scenarios use

the larger radius except this one. The reason, is that if an algorithm struggles when initialized in a dense environment, it wont matter what other obstacles are elsewhere, thus replicating this density on any other world would be extraneous.

Scenario 3: Uniform World. Robots are spawned in the region $R3 = (x, y)$ such that $x \in [1, 6], y \in [1, 19]$. Obstacles are spawned across the region $O = (x, y)$ such that $x \in [6, 16], y \in [1, 18]$. The goal region G is a disk of radius 4 centered at $(17, 17)$. This scenario evaluates basic collision avoidance ability.

Scenario 4: Concave World. This world uses a fixed map with no randomly generated obstacles. Robots are spawned in the region $R4 = (x, y)$ such that $x \in [1, 6], y \in [1, 19]$. The goal region G is a disk of radius 4 centered at $(18, 18)$. This scenario evaluates complex collision avoidance in a non-convex environment.

Additional environments have been provided with the framework, such as a corridor environment with a long passageway. We dont evaluate proportional control laws on such an environment, as the effort would be wasted. A planning algorithm would be needed instead.

B. Comparison Results

The benchmarking capabilities of the framework allowed for the sample comparison of $move\_to\_goal$ and $move\_to\_goal\_bc$ algorithms (proportional controllers with and without obstacle avoidance, respectively). Sample data is displayed for the Concave Map and the Uniform Map configurations (see Figures 4 and 5). The data confirm our predictions that control algorithms that did not involve Barrier Certificates (no avoidance) yielded a larger number of robot collisions in simulation, both for the Concave and Uniform Map scenarios. This test was performed in order to evaluate the benchmarking capabilities of the framework, confirmed by the output of logically coherent results.

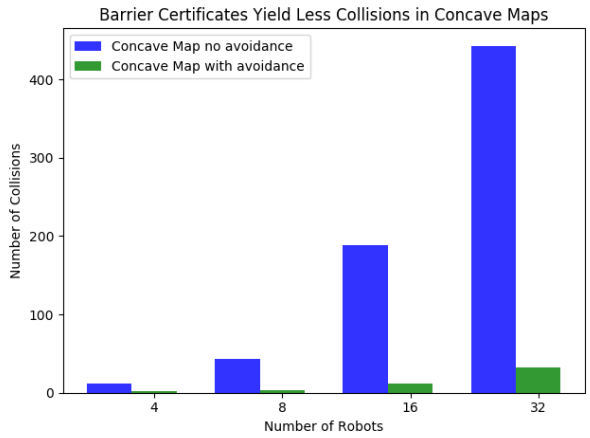


Fig. 4. Data on collisions in Concave Map.

V. Discussion and Conclusion

We presented an architecture that provides a design pattern for developing full stack swarm systems. This entails 4 primary layers: design (through building a swarm behaviour library), deployment (on various systems), interaction (through operator control) and evaluation(through benchmarking) of the swarm. Using our proposed architecture, we then implemented a framework titled: CMUSWARM, on the ROS platform using Gazebo with irobot create for simulation. We then conduct an evaluation of our architecture by comparing two simple swarm control-laws within the CMUSwarm framework. Our work generalizes existing swarm architectures by focusing not on one particular subsystem, or existing framework; but rather the challenge of designing a full stack system that is platform independent. Such design patterns are required to allow for rapid integration and evaluation of new methods and technologies. There remains much additional work we will focus on in the future that will better highlight the usage of our proposed architecture. Using our implemented framework CMUSWARM, we wish to address the following issues: **a)** Providing a wider span of algorithms, input devices, and evaluation data for the framework. **b)** Extending the architecture to support heterogeneous swarms **c)** Performing human-swarm interaction experiments using CMUSWARM **d)** Alternative extension of our framework using a DDS vendor for publisher-subscriber architecture on the ROS2 platform.

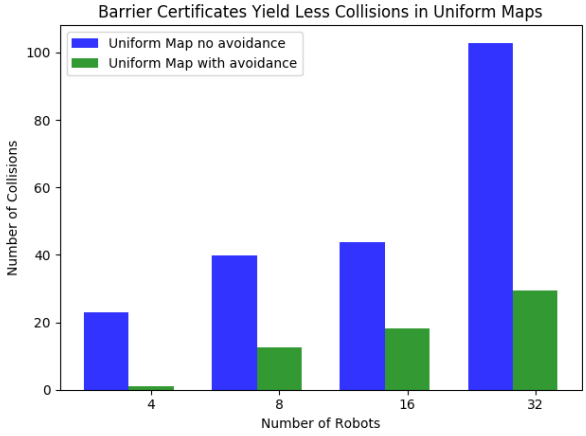


Fig. 5. Data on collisions in Uniform Map.

Our work in this paper presents the architecture and brief overview, but the effectiveness of our design will best be demonstrated through the proposed future work. We hope that our proposed architecture will be used as a standard for developing further swarm systems. By using a common overarching design, the swarm research community will become less decentralized and able to focus on moving forward.

References

[1] R. Gerndt, D. Seifert, J. Baltes, S. Sadeghnejad, and S. Behnke, "Humanoid robots in soccer: Robots versus humans in robocup 2050," *IEEE Robotics and Automation Magazine*, vol. 22, no. 3, pp. 147–154, 9 2015.

[2] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, "Swarm robotics: a review from the swarm engineering perspective," *Swarm*

Intelligence, vol. 7, no. 1, pp. 1–41, Mar 2013. [Online]. Available: https://doi.org/10.1007/s11721-012-0075-2

[3] S. Nagavalli, "Behavior composition in human interaction with robotic swarms," PhD Thesis Proposal, Carnegie Mellon University, 2017.

[4] K. Birman and T. Joseph, "Exploiting virtual synchrony in distributed systems," *SIGOPS Oper. Syst. Rev.*, vol. 21, no. 5, pp. 123–138, Nov. 1987. [Online]. Available: http://doi.acm.org/10.1145/37499.37515

[5] E. Şahin, *Swarm Robotics: From Sources of Inspiration to Domains of Application*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 10–20. [Online]. Available: https://doi.org/10.1007/978-3-540-30552-1_2

[6] J. Bjerknes and A. Winfield, "On fault tolerance and scalability of swarm robotic systems," *Distributed autonomous robotic systems*, pp. 431–444, 2013.

[7] W. Truszkowski, M. Hinchey, J. Rash, and C. Rouff, "Nasa's swarm missions: The challenge of building autonomous software," *IT professional*, vol. 6, no. 5, pp. 47–52, 2004.

[8] M. H. M. Alkilabi, A. Narayan, and E. Tuci, "Cooperative object transport with a swarm of e-puck robots: robustness and scalability of evolved collective strategies," *Swarm Intelligence*, pp. 1–25.

[9] M. Duarte, J. Gomes, V. Costa, T. Rodrigues, F. Silva, V. Lobo, M. M. Marques, S. M. Oliveira, and A. L. Christensen, "Application of swarm robotics systems to marine environmental monitoring," in *OCEANS 2016 - Shanghai*, April 2016, pp. 1–8.

[10] M. Rubenstein, A. Cabrera, J. Werfel, G. Habibi, J. McLurkin, and R. Nagpal, "Collective transport of complex objects by simple robots: theory and experiments," in *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2013, pp. 47–54.

[11] L. Wang, A. D. Ames, and M. Egerstedt, "Safety barrier certificates for collisions-free multirobot systems," *IEEE Trans. Robotics*, vol. 33, no. 3, pp. 661–674, 2017. [Online]. Available: https://doi.org/10.1109/TRO.2017.2659727

[12] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.

[13] M. Bakhshipour, M. J. Ghadi, and F. Namdari, "Swarm robotics search & rescue: A novel artificial intelligence-inspired optimization approach," *Applied Soft Computing*, 2017.

[14] A. Kolling, P. Walker, N. Chakraborty, K. Sycara, and M. Lewis, "Human interaction with robot swarms: A survey," *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 1, pp. 9–26, 2016.

[15] R. Bogue, "The role of robots in the battlefields of the future," *Industrial Robot: An International Journal*, vol. 43, no. 4, pp. 354–359, 2016. [Online]. Available: https://doi.org/10.1108/IR-03-2016-0104

[16] L. He, J. Pan, W. Wang, and D. Manocha, "Proxemic group behaviors using reciprocal multi-agent navigation," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 292–297.

[17] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '87. New York, NY, USA: ACM, 1987, pp. 25–34. [Online]. Available: http://doi.acm.org/10.1145/37401.37406

[18] G. M. Atin, D. M. Stipanovi, P. G. Voulgaris, and M. Karkoub, "Swarm-based dynamic coverage control," in *53rd IEEE Conference on Decision and Control*, Dec 2014, pp. 6963–6968.

[19] W. Luo, N. Chakraborty, and K. Sycara, "Distributed dynamic priority assignment and motion planning for multiple mobile robots with kinodynamic constraints," in *American Control Conference (ACC)*, July 2016.

[20] A. Viseras, T. Wiedemann, C. Manss, L. Magel, J. Mueller, D. Shutin, and L. Merino, "Decentralized multi-agent exploration with online-learning of gaussian processes," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 4222–4229.

[21] C. Pinciroli, V. Trianni, R. O'Grady, G. Pini, A. Brutschy, M. Brambilla, N. Mathews, E. Ferrante, G. Di Caro, F. Ducatelle, M. Birattari, L. M. Gambardella, and M. Dorigo, "ARGoS: a modular, parallel, multi-engine simulator for multi-robot systems," *Swarm Intelligence*, vol. 6, no. 4, pp. 271–295, 2012.

[22] J. Lächele, A. Franchi, H. Bülthoff, and P. Robuffo Giordano, "Swarmsimx: Real-time simulation environment for multi-robot systems," *Simulation, modeling, and programming for autonomous robots*, pp. 375–387, 2012.

[23] S. Curtis, A. Best, and D. Manocha, "Menge: A modular framework for simulating crowd movement," *Collective Dynamics*, vol. 1, no. 0, pp. 1–40, 2016.

[24] F. Aznar, M. Sempere, F. Mora, P. Arques, J. Puchol, M. Pujol, and R. Rizo, "Agents for swarm robotics: Architecture and implementation," in *Highlights in Practical Applications of Agents and Multiagent Systems*. Springer, 2011, pp. 117–124.

[25] V. Grabe, M. Riedel, H. H. Bulthoff, P. R. Giordano, and A. Franchi, "The telekyb framework for a modular and extendible ros-based quadrotor control," in *Mobile Robots (ECMR), 2013 European Conference on*. IEEE, 2013, pp. 19–25.

[26] MATLAB. (2012) Mrsim - multi-robot simulator. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/38409-mrsim-multi-robot-simulator–v1-0-

[27] J. Vain, T. Tammet, A. Kuusik, and E. Reilent, "Software architecture for swarm mobile robots," in *Electronics Conference, 2008. BEC 2008. 11th International Biennial Baltic*. IEEE, 2008, pp. 231–234.

[28] P. Fankhauser and M. Hutter, "A Universal Grid Map Library: Implementation and Use Case for Rough Terrain Navigation," in *Robot Operating System (ROS) The Complete Reference (Volume 1)*, A. Koubaa, Ed. Springer, 2016, ch. 5. [Online]. Available: http://www.springer.com/de/book/9783319260525

[29] C. Pinciroli, A. Lee-Brown, and G. Beltrame, "Buzz: An extensible programming language for self-organizing heterogeneous robot swarms," *arXiv preprint arXiv:1507.05946*, 2015.

[30] B. Gerkey, R. T. Vaughan, and A. Howard, "The player/stage project: Tools for multi-robot and distributed sensor systems."

[31] K. Hawick, H. James, J. Story, and R. Shepherd, "An architecture for swarm robots," *Computer Science Division, School of Informatics University of Wales, North Wales, UK*, 2002.

A Faster RCNN-Based Wheelchair Recognition System

Ingrid Navarro
Instituto Tecnológico de Monterrey
ingrid.navarro.an@gmail.com

Luis E. Navarro-Serment
Robotics Institute
Carnegie Mellon University
lenscmu@ri.cmu.edu

***Abstract*—Wheelchairs are one of the most important auxiliary instruments for people with mobility impairments. Accurate detection and tracking of these devices could bring a number of improvements in automated services that aim to assist, monitor and provide better accessibility to allow wheelchair users to participate in community life. In this paper, we present a Deep Learning-based wheelchair detection and tracking system to address some of the limitations of previous approaches, which include detecting different types of wheelchairs in cluttered environments and from different viewing angles. We explore region-based Convolutional Neural Networks (R-CNN), in particular Faster R-CNN, as it has become one of the top performers for object detection tasks. We evaluate the performance of different training techniques using two Faster R-CNN frameworks and different backbone network structures. Furthermore, we present how we empirically addressed some of the preceding limitations by applying specific data augmentation techniques and constraints to our model. We demonstrate that using a region-based implementation outperforms previous approaches in terms of overall robustness, accuracy and flexibility.**

***Index Terms*—Object Detection, Deep Learning, Computer Vision**

I. Introduction

People with mobility impairments are particularly vulnerable to deficiencies in services that are meant to provide accessibility to them. Even with increasing number of initiatives for disability inclusion, there are still multiple barriers that can make it extremely difficult for them to participate in community life. Wheelchair users face some of these barriers in their daily lives, particularly related to transportation, accessibility to facilities and health assistance. With the increase of adaptive and assistive technologies, providing an accurate wheelchair detection and tracking system could bring a number of automated services to assist and monitor people who suffer from any kind of mobility limitation. For instance, assistive and adaptive robots, improvement of accessibility guidelines for buildings, transportation and handling of wheelchair users and traffic-light time adjustments by detecting people in wheelchairs crossing the roads. Also, by detecting a wheelchair, one can further monitor and analyze the behavior of its user, which could be useful in healthcare applications.

Several approaches focused on automated wheelchair control systems, obstacle detection and smart navigation [1, 2, 3] have been explored in recent years with the increase of object detection and scene understanding techniques. Nevertheless, wheelchair detection and pose estimation are topics that remain relatively unexplored in the field of computer vision. Current approaches that attempt to detect this particular object, present certain limitations. Some of these limitation include camera pre-calibration assumptions [4], detection of very specific kind of wheelchairs [5], difficulties differentiating between walkers and wheelchairs when they are in motion and use of costly sensor equipment, which may be an obstacle for widespread use [6].

The aforementioned limitations motivated the exploration of state-of-the-art visual recognition techniques using deep learning since these techniques are capable of producing end-to-end robust solutions. Such approaches like Fully Convolutional Networks (FCN) for Semantic Segmentation [7], Faster R-CNN [8] and Mask R-CNN [9], currently outperform traditional computer vision and feature extraction techniques in terms of accuracy, scalability and adaptability. By exploiting these advantages, we aim to develop a framework to accurately detect and track wheelchairs in cluttered environments. In contrast

to previous approaches [4, 5, 6], this implementation also intends to detect different types of wheelchairs (e.g.. motorized wheelchairs, transport wheelchairs, recliners) and from different viewpoints.

Wheelchair detection poses a nontrivial problem for a number of reasons. Feature extraction, for instance, can become a difficult task as wheelchairs of different types may or may not have evident features in common. Also, alternating the viewing angle of a camera drastically changes the appearance of a wheelchair, which makes feature extraction a more difficult task. Similarly, occlusions provoked by wheelchair users also pose an important challenge, especially during the data labeling task. Finally, wheelchairs share certain features with other common objects like vehicles, bicycles and strollers. Although we are not attempting to overcome all of these complications, we will explain how did we empirically address some of them.

Our long term objective is to estimate pose detection by further segmenting regions once the object of interest is detected. Nevertheless, pose estimation will be beyond the scope of this paper, as we will focus more on evaluating the performance of the wheelchair detection system.

II. Related Work

This section gives an insight into the previous work on wheelchair detection systems. We analyze the advantages and highlight important aspects that need to be improved for each approach. We thus explain our motivation to address these aspects by exploring state-of-the-art object recognition techniques.

Regarding wheelchair recognition systems, Myles *et. al.* [4] presented a method for detection with pre-calibration assumptions. Such assumptions include knowing the approximate width of the wheels, calibration details with respect to the ground and the pitch of the camera. Furthermore, their method requires to expose the face of the person sitting on the wheelchair and the entire wheelchair to appear on the image. This implementation uses Hough Transform to first locate the face of a person sitting on a wheelchair and then locate the wheels. Pose estimation is then constructed by using 2D ellipse projection. An important disadvantage of their implementation is that it requires extensive pre-calibration and would fail to detect a wheelchair if the conditions of the environment are not ideal and moreover if the face of its user is not exposed.

Huang *et. al.* [5] designed a single-camera wheelchair detector based on a decision tree structure combined with boosted-cascade classifiers. This implementation attempts to detect wheelchairs and their discrete moving directions without the need for pre-calibration as in [4]. It was designed to perform detection in cluttered environments using early confidence to reject non-wheelchair image regions using invariant local descriptors such as Histogram of Oriented Gradients (HOG) [11] and Contrast Context Histogram [12]. Although this implementation addresses the limitations found in [4], there are certain disadvantages of using Cascaded Decision Trees (CDT), such as design complexity and instability due to perturbations.

In [6], Beyer *et. al* used a laser scanning technique to perform wheelchair/walker detection. It consists of preprocessing laser data by cutting out a resampled window around every laser point and computing detection locations in a coordinate system. Then, they use a CNN to predict relative locations and a Non Maximum Suppression (NMS) scheme to transform these predictions into detections. This approach presents a relatively fast and robust solution for wheelchair detection using deep learning. Nevertheless, the authors highlighted that walker detection presents a high confusion with wheelchairs suggesting that their training was rather biased towards wheelchairs. Additionally, the use of laser sensors entails high costs, which in general is an impediment for widespread use.

The foregoing limitations motivated the exploration of a robust and flexible deep learning framework for visual recognition. For this reason, we based our work on the current top performing object detection approaches. Particularly the following:

Fully Convolutional Networks for Semantic Segmentation: J. Long *et. al.* [7], trained fully convolutional networks end-to-end to perform pixelwise prediction from a supervised pre-training. Their approach is capable of taking an input of arbitrary size and perform learning and inference to produce correspondingly-sized dense-predictions for per-pixel size like semantic segmentation.

Faster RCNN: Ren *et. al.* [8], designed an object detection system consisting of two stages. The first stage is a Region Proposal Network (RPN) that proposes candidate regions with$k$-objectness scores, where k is the number of classes. The second stage is essentially the Fast RCNN detector [10] that uses the RPN proposed regions to extract features using RoI Pooling from each candidate region and performs classification and bounding-box regression.

Mask RCNN: In [9] K. He *et. al.*, proposed extending Faster RCNN by adding a branch to predict a mask in parallel to performing classification and bounding-box regression. In this approach encoding the spatial structure of masks needs to be addressed by pixel-to-pixel correspondence. This correspondence requires RoI

features to be faithfully aligned to the spatial layout and therefore predict pixel-accurate masks.

As mentioned in the previous section, we ultimately aim to reproduce Mask RCNN approach to create an accurate and robust wheelchair detection system. An important reason is to provide a framework that not only detects and tracks wheelchairs by performing bounding box regression and classification. We aim to further estimate wheelchair pose by creating a mask after predicting candidate regions where the object of interest is located. Mask RCNN provides a flexible solution to perform strict instance segmentation given a Faster RCNN framework.

III. Approach

In this section we explain how we implemented our detection system using two different training methodologies for the Faster R-CNN approach. Each methodology was developed using a different framework. We thus evaluate their performance to determine the advantages that each of these implementations offer and are more convenient to further reproduce the Mask R-CNN. As explained in [5] and [6] the RPN generator and the Fast R-CNN detector that compose Faster RCNN share convolutional features that could be used to propose regions and perform region-based detection. This observation allows to merge both networks, thus providing a more effective solution to reduce proposal computation time as compared to previous region proposal methods [13], [14], [15].

There are several schemes to combine and train these modules. In this paper, we explored two training methodologies as explained in [5]:

a) ***4-step alternating training:***
: For this training method we used a simple architecture we designed based on a MATLAB implementation for object detection and tracking. This simple architecture was used because the 4-step training can be computationally expensive in terms of time and memory usage.

The general objective of this method (Figure 1) is to learn shared features in an alternating manner. In the first step, we trained the RPN to generate region proposals using a pre-trained model from the Computer Vision Toolbox provided by MATLAB. In the second step, the detection network was trained separately using the proposed regions generated on the first step and the same pre-trained model from the previous step. In the third step, the RPN was re-trained using the detector from the previous step to initialize the training but we only fine-tuned the layers corresponding to this network. At this point the convolutional layers are shared, whereas in the first and second steps the layers were separated. In the final step, we only fine-tuned the layers unique to the Fast RCNN detector using the regions from step three. This scheme adopts the multi-task classification and regression loss defined in [10].

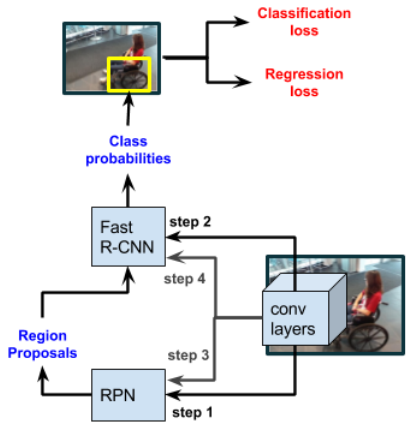


Fig. 1: Alternating training. Convolutional layers are not shared until step 3.

b) ***Approximate joint training***
: For this training methodology (Figure 2) we used the ResNet50 [16] network structure implemented in Keras. In this solver, both the RPN generator and the Fast RCNN detector are merged into one network during training. Each forward pass iteration generates region proposals treated as a pre-computed proposal for the detector. This approach is simpler to implement; however, it may generate less accurate results - as explained in [5] - but reduces training time significantly. On each iteration, this scheme generates classification and regression losses from the detector and the RPN. At the end of the epoch, the average of each loss is obtained. and the total loss is calculated by adding up each of the four losses.

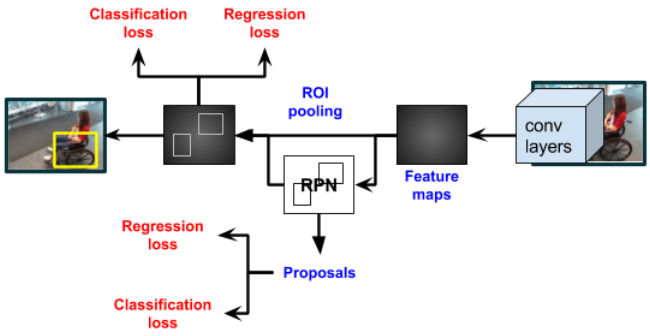


Fig. 2: Approximate Joint Training

IV. Implementation Details

Using the methodologies described in the previous section, we train and test two different backbone architectures: ResNet50 for the approximate joint training, and a simple architecture designed for the 4-step alternating training. We report the mean average precision metric (mAP) and the average detection time per image for each of the methods evaluated. We perform the experiments using a 386-core GeForce GT 750M and a 1050-core GeForce GT 760M. Using the second GPU we reduced training time about $\tilde{5}0\%$.

Our dataset consists of a total of 800 manually labeled images including standard wheelchairs, electric wheelchairs, recliners and transport wheelchairs. The dataset included images with empty wheelchairs and wheelchairs carrying users, in both cluttered and uncluttered environments. In particular, we avoided images that presented significant truncation and images where a walker or a particular object significantly occluded the wheelchair.

We adopt image-centric training as described in [10]. We resized the images such that their shorter side was 320 pixels for the 4-step training method and 450 pixels for the approximate joint training method, whereas in [8] they set the shorter side to 600 pixels. This step was carried out differently as compared to the original approach in order to address memory allocation issues. We also tested this implementation with consecutive video frames by performing detection every 10 frames on as system with a rate of 30fps.

RPN proposals threshold was fixed at 0.7 to reduce overlapping redundancy using a NMS scheme as in the original implementation, then we empirically applied an additional reduction with a threshold at 0.5. Next, we calculated detection using mAP@[0.50, 0.70, 0.85] IoU for positive Regions of Interest (RoI). The remaining hyperparameters were left unchanged following the original approaches [8] and [10].

V. Experiments

We analyzed test results when we trained using the original dataset to understand what features did the networks learn and to further compare how performance could change by adding or removing data as proposed in [17], with the purpose of separating objects into groups that share significant features. To demonstrate this, we created an additional dataset starting from the original one but excluding all kinds of electrical wheelchairs as shown in Figure 3 (a). This new dataset consisted of 470 images of standard wheelchairs as illustrated in Figure 3 (b).

We also compared performance using data augmentation techniques as explored in [17] and [18]. By empirical analysis, we found that augmenting the training data in favour of certain appearance factors is not always helpful to improve performance and may even be detrimental. In particular, adding more images where wheelchairs presented occlusion and applying random truncations and flippings produced poor results. For this reason, we applied certain constraints to our dataset such as avoiding these augmentation techniques. However, by trying other methods for data augmentation we found that random rotations helped improve detection of wheelchairs when presented from different viewing points.

a) Electric wheelchairs

b) Standard wheelchairs

Fig. 3: Wheelchair bin separation.

VI. Results

Table 1 shows mAP results from the 4-step alternating training using the MATLAB Faster R-CNN framework. Table 2 shows mAP results from the approximate joint training method using Keras Faster R-CNN framework. In both Table 1 and 2, we include results from the experiments with the original dataset as well as the ones using wheelchair-bin separation and data augmentation. We also show the average detection time for each experiment. In both tables we refer to the average detection time as DT.

Plots shown in Figures 4 and 5 show mini-batch loss and accuracy from the different experiments that were carried out using the 4-step alternating training method. Plots shown in Figures 6 and 7 exhibit total loss and classifier accuracy from the experiments using the approximate joint training method. In each of the plots, we refer to the complete dataset as CD, to the standard wheelchair as SW, and to the data augmentation as DA.

Final total loss for CD was 20.0% achieved during epoch 9 with classifier accuracy of 94.4%. For CD-

DA final loss was 28.5% achieved during epoch 16 a 93.4% accuracy. After this point, the total loss started oscillating. For SW and SW-DA, the final losses were 16.2% at epoch 11 without data augmentation and 22.2% at epoch 19 with data augmentation and with accuracies of 95.2% and 94.5%, respectively.

We observe that mAP results from the approximate joint training method are significantly better as compared to the results from the 4-step alternating training method. The reason for this is that the network architecture used for the approximate joint method [16] is significantly more complex than the network from the 4-step method, thus higher overall performance was achieved. However, the original approach [8] explains that in general the approximate method tends to give slightly less accurate results as it does not consider the derivative with respect to the proposal boxes' coordinates, which are responses of the network and are considered in the 4-step method. In future work, we could focus on adapting a more complex network architecture for this framework to achieve better results.

Regarding the use of data augmentation and bin-separation methods, we observe how in general applying these techniques helps improve performance of the detection system (Tables 1 and 2). In particular, we found that augmenting our data helped improve detection from different viewing points.

Finally, in Figure 8 we observe some selected samples of wheelchair detection results using the Faster R-CNN framework. These include different types of wheelchairs from different viewing angles and in presence of clutter.

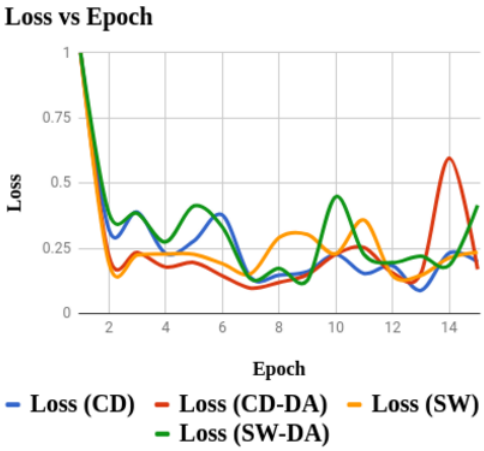


Fig. 4: Loss during step 4 using the 4-step alternating training method.

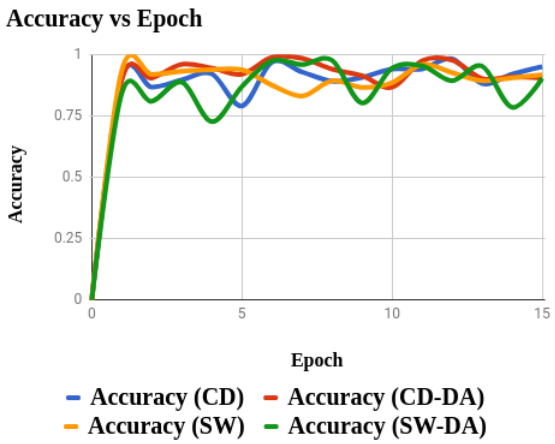


Fig. 5: Accuracy during step 4 using the 4-step alternating training method.

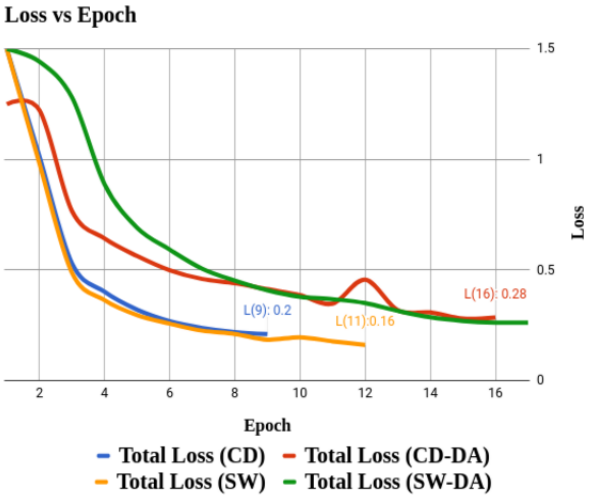


Fig. 6: Total loss using approximate joint training.

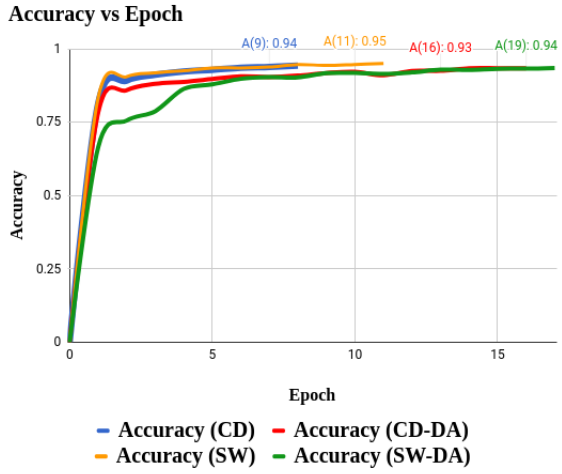


Fig. 7: Classifier accuracy using approximate joint training.

No Data Augmentation				
	mAP@0.50	mAP@0.70	mAP@0.85	DT(s)
Standard Wheelchair Dataset	41.5%	15.2%	8.4%	0.98
Complete Wheelchair Dataset	56.9%	**26.8%**	6.5%	1.24
Data Augmentation				
	mAP@0.50	mAP@0.70	mAP@0.85	DT(s)
Standard Wheelchair Dataset	47.8%	22.6%	**11.8%**	**0.93**
Complete Wheelchair Dataset	**59.6%**	20.4%	6.7%	1.38

TABLE I: mAP@[0.50, 0.70, 0.85] test results from the 4-step alternating training method and average detection time per image (DT).

No Data Augmentation				
	mAP@0.50	mAP@0.70	mAP@0.85	DT(s)
Standard Wheelchair Dataset	92.2%	78.7%	**19.9%**	**0.95**
Complete Wheelchair Dataset	87.5%	75.5%	15.2%	1.03
Data Augmentation				
	mAP@0.50	mAP@0.70	mAP@0.85	DT(s)
Standard Wheelchair Dataset	**92.7%**	75.2%	14.1%	0.98
Complete Wheelchair Dataset	91.6%	**82.8%**	14.6%	1.16

TABLE II: mAP@[0.50, 0.70, 0.85] results from the approximate joint training method and average detection time per image (DT)

VII. Discussion and Future Work

In this paper, we present a flexible and robust deep learning-based wheelchair detection and tracking approach using two different Faster R-CNN frameworks. We address some important challenges and limitations of previous wheelchair detection approaches. However, we observe that adding more data does not always solve the weak points of detection systems. An example of this, is that our implementation often fails to recognize wheelchairs when they are facing the camera or when they are significantly occluded by people or objects in general. There are several novel approaches like [19] and [20] focused on occlusion handling. Such techniques could be further explored to address the

Our long term purpose is to reproduce Mask RCNN approach to increase wheelchair detection robustness and accuracy. As mentioned in previous sections, we aim to provide a framework that will be able to estimate pose using segmentation techniques, all together with performing detection and tracking. Furthermore, semantic masks could reduce the complexity of wheelchair detection by providing a more accurate definition of a wheelchair instance.

VIII. Acknowledgements

I would like to thank Instituto Tecnológico de Monterrey, Santander and EDUCAFIN for supporting this work. In addition, I would like to extend my gratitude to the organizers of the RISS program.

References

[1] A. Myles, N. V. Lobo and M. Shah. *Wheelchair Detection in a Calibrated Environment*. Melbourne, Australia: Asian Conference on Computer Vision, 2002.

[2] C.R Huang, P.C. Chung, K.W. Lin and S.C. Tseng. *Wheelchair Detection Using Cascaded Decision Tree*. IEEE: Transactions on Information Technology in Biomedicine, 2010.

[3] L Beyer, A. Herman, and B. Leibe. *DROW: Real-Time Deep Learning-Based Wheelchair Detection in 2D Range Data*. IEEE: Robotics and Automation, 2017.

[4] F. Utaminigrun and T. Kurniawan. *A laser-vision based obstacle detection and distance estimation for smart wheelchair navigation*. Indonesia, 2014.

[5] J. Pineau, A. Moghaddam, H. Yuen, P. Archambault, F. Routhier, F. Michaud and P. Boissy. *Automatic Detection and Classification of Unsafe Events During Power Wheelchair Use*. IEEE: Journal of Transactional Engineering in Health and Medicine, 2014.

[6] L. Liao, Y. Wu, Y. Xiang, X. Yab, J. Shi, J. Bai X. Zhang, M. Du, R. Yang and K. Jenkins. Boissy. *Control System of Powered Wheelchairs Based on Tongue Motion Detection*. IEEE: 15th International Conference on Cognitive Informatics and Cognitive Computing, 2016.

[7] J Long, E. Shellhamer, T. Darrell *Fully Convolutional Networks for Semantic Segmentation*. UC Berkeley, CA, 2015.

[8] S. Ren, K. He, R. Girshick, J. Sun *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. NIPS, 2016.

[9] K. He, G. Gkioxari, P. Dollar, R. Gishick *Mask RCNN*. 2017.

[10] R. Girshick *Fast RCNN*. ICCV, 2015.

[11] N. Dalal and B. Triggs. *Histograms of Oriented Gradients for Human Detection*. IEEE: Computer Society Conference on Computer Vision, 2005.

[12] C. Ancuty and P. Bekaert. *SIFT-CCH: Increasing the SIFT Distinctness by Color Co-Occurrence Histogram*. International Symposium on Image and Signal Processing and Analysis, 2007.

[13] J. Yang, Z. Gan, K. Li, C. Hou *Graph-Based Segmentation for RGB-D Data Using 3D Geometry Enhanced Superpixels*. IEEE: Transactions on cybernetics, 2015.

[14] J.R. Uijlings, K.E. van de Sande, T. Gevers, A.W. Smeulders *Selective search for object recognition*. Journal of Computer Vision (IJCV), 2013.

[15] C.L. Zitnick, P. Dollar *Edge boxes: Locating object proposals from edges*. European Conference on Computer Vision (ECCV), 2014.

[16] K. He, X. Zhang, S. Ren and J. Sun. *Deep Residual Learning for Image Recognition*. 2015.

[17] B. Pepik, R. Benenson, T. Ritshel, B. Schiel *What is Holding Back Convnets for Detection*. 2015.

[18] J. Wang and L. Perez, The Effectiveness of Data Augmentation in Image Classification using Deep Learning, 2017.

[18] J. Wang and L. Perez. *The Effectiveness of DAta Augmentation in Image Classification using Deep Learning*. 2017.

[19] J. Marín, D. Vázquez, A. López, J. Amores and L. Kuncheva. *Occlussion Handling via Random Subspace Classifiers for Human Detection.*. IEEE: Transactions on Cybernetics, 2014.

[20] Y. Cai, H. Wang, X. Chen and L. Chen. *Mutilevel framework to handle objec occlusions for real-time tracking*. IET: Journals and Magazines, 2016.

Fig. 8: Selected examples of object detection results using our Faster R-CNN wheelchair detection system.

YOLOv2 and Hierarchical Softmax Tree for US Traffic Sign Detection System

Dat Nguyen
Robotics Institue Summer Scholar,
Carnegie Mellon University

Abstract

With the emergence of self-driving cars, transportation inspection becomes more critical for maintaining a good and safe infrastructure. The inspection process, particularly traffic sign inventory tracking, usually requires manual and intensive human labor. In previous works, traditional computer vision approaches were proposed to automate this process. However, these methods usually do not scale well when there are more labels in the dataset. In this paper, we present a traffic sign detection system based on YOLOv2, a fully Convolutional Neural Networks (CNNs). We treat the localization and a classification as regression task to detect over 50 different US traffic signs. Therefore, the network can run at 40 fps on NVIDA GTX 1070. In addition, we construct a hierarchical soft-max tree, which embeds prior knowledge of relationships between traffic signs.

1. Introduction

Traffic Sign Detection (TSD) system is an application of object detection. Its task is to recognize and localize traffic signs on the road. In the context of infrastructure inventory, TSD system is useful for governments and cities to automatically keep track of their traffic sign inventory by reducing the amount of manual labor work and passive maintenance [3]. Furthermore, with the emergence of autonomous vehicles, TSD plays an critical role in safety operation. Given the importance of TSD, we want to design a reliable and robust system for Traffic Sign Detection.

Recent advances in Machine Learning suggest that deep learning network have potentials to achieve high accuracy at pattern recognition task such as image classification, speech recognition, and object detection [6]. We introduce a new TSD system based on You Only Look Once version 2 (YOLOv2) by Redmon et al. [10], a fully convolutional network. The advantages of this model is its fast interference time and high accuracy.

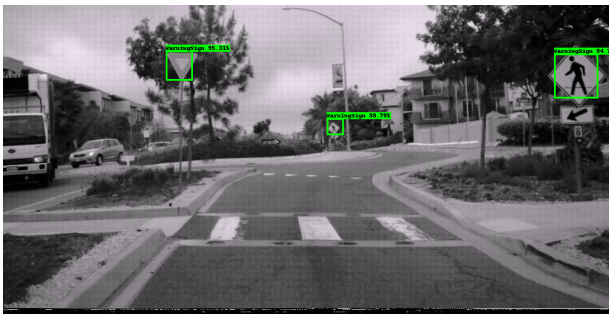


Figure 1. Example of YOLOv2 prediction on LISA Dataset

In this paper, we will provide following contributions:

- Construct YOLOv2 for Traffic Sign Detection Task.
- Integrate hierarchical soft-max tree to embed prior knowledge of the traffic sign dataset into the network.
- Experiment and compare accuracy of MobileNet [5] and DenseNet [4] as feature extractor on YOLOv2.

In the next section, we describe related work and why we decided to choose YOLOv2. Then, we explain the network architecture, loss function and how we integrated hierarchical soft-max tree. Finally, we provide our experiment results and future directions.

2. Related Work

Traffic Sign Recognition System Traditional computer vision approaches such as Histogram of Oriented Gradients (HOG) and Support Vector Machine Classifier (SVM) has previously been applied to Traffic Sign Detection [9]. Although there are certain successes, these still require to use hand-drafted features, which could create a challenge when there are a large number of labels in the dataset. For example, recent ImageNet Detection Challenge contains over 1000 different categories.

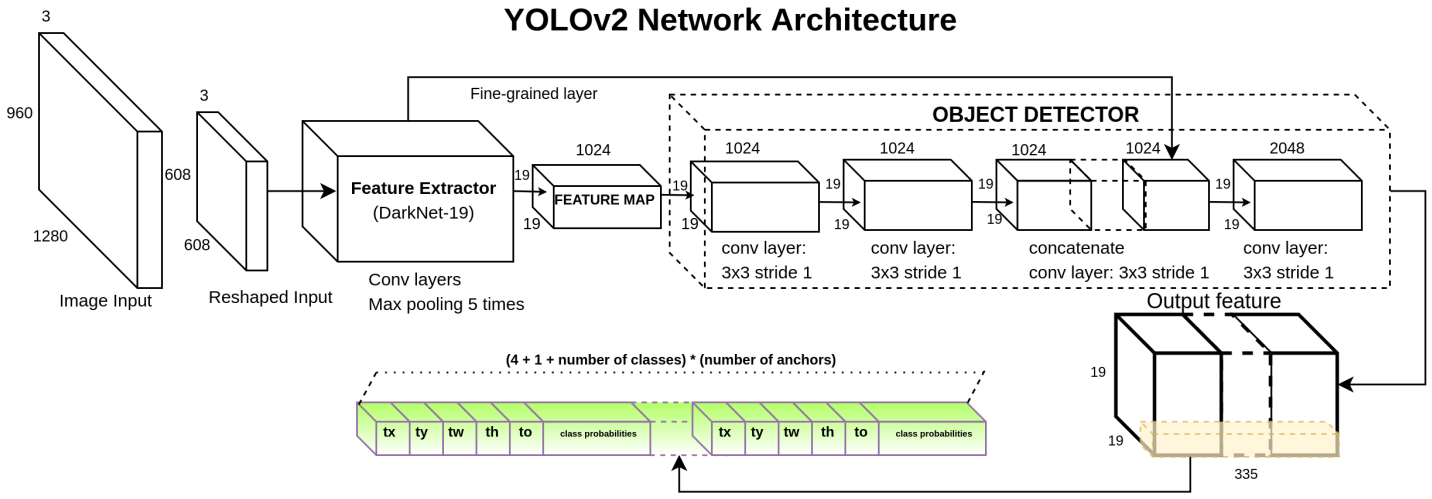


Figure 2. YOLOv2 network consists of feature extractor and object detector.Its feature output contains information

Deep Learning Detection Model An emerging and promising approach is to apply machine learning, specifically deep learning, in order to let the computer automatically decide which features to use. A survey of current state-of-the-art object detection models is summarized in Table 1. We decided to use YOLOv2 due to its fast interference and accuracy.

LISA Dataset. In order to train deep model, data plays a critical role. The more data, the better for the model to generalize. To the best of our knowledge, LISA is the only open-sourced Traffic Sign Detection dataset available in the US. There are a few important findings in this dataset, which might affect performance of the network. First, this dataset uses different cameras and vehicles with various angles, lighting conditions and color spaces. This is beneficial because it potentially forces the network to generalize and make better prediction in adversarial cases. LISA contains frames in time series, which is an unique advantage for tracking. The video frames provide temporal information could help the network learn easier.

Detection Frameworks	Train	mAP	FPS
Faster-RCNN [11]	2007+2012	73.2	5
SSD500 [8]	2007+2012	76.8	19
SSD500 [8]	COCO Trainval	46.5	19
YOLOv2 544x544	2007+2012	**78.6**	40
YOLOv2 608x608	COCO Trainval	48.1	40

Table 1. Accuracy/Speed Comparisons on state-of-the-art object detection models.
Mean Average Precision (mAP) were measured on Pascal VOC 2007 Test Dataset [3] and COCO Test set [7]. .

3. Network Architecture

In this section, we describe the YOLOv2 architecture, how anchors and grid cells improve numerical stability for the network, and YOLOv2 loss function.

3.1. YOLOv2

YOLOv2 is a fully convolutional network. The model includes a feature extractor and object detector. The feature extractor's task is to extract important features from the image to feed into the object detector. Then, the object detector analyzes this feature map to output the prediction. We described the network architecture in Figure 3. The term "You Only Look Once" refers to how the network produces its prediction. Instead of having a parallel region proposal network like Faster-RCNN, YOLOv2 goes through the feature map only once to predict the objects.

Feature Extractor. YOLOv2 uses DarkNet-19 [10] as default feature extractor. It is an improved version of VGG-19, a simple yet effective network for extracting important features [12]. DarkNet-19 uses Leaky Rectified Unit (LeakyRELU) as default activation function with $\alpha = 0.01$ and Batch Normalization in each convolutional layer. This would help the network to avoid "dying RELU" problem from bad initialization.

Object Detector. Given the feature map, the object detector produces prediction using three blocks of (3x3) convolutional layers with 1024 filters and a fine-grained layer from feature extractor. Note that the object detector does not produce bounding boxes for objects directly. Instead, it produces thousand of bounding boxes with certain probability and confidence through an *output feature map*.

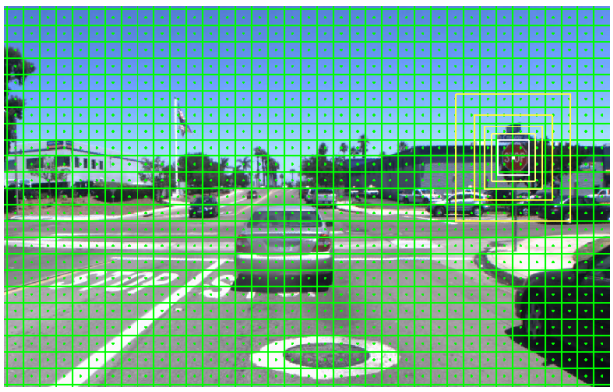


Figure 3. Anchor and Grid cell map Visualization

These later can be filtered out in post-processing step using Non-Max Suppression. For example, one would like to extract only bounding boxes with object confidence over 90% and having class probability greater than 95%.

Output Feature Map contains the predictions of YOLOv2. One might imagine it as a three dimensional grid cell map. Each cell is a vector with length of $number\_of\_anchors * (number\_of\_classes + 5)$. This allows the network to predict thousand of bounding boxes very fast. For example, we feed into the network an image with size of (960 x 960), we use 5 anchors and 31 classes. For each prediction, feature extractor produces a feature map size $(30, 30, 5 * (4 + 1 + 31))$. The total number of predictions is $30 * 30 * 5 = 4500$ bounding boxes. In each bounding box prediction, it contains a relative coordinate, object confidence and class probability.

3.2. Anchors and Grid Cells

Anchors and grid cells are used in YOLOv2 as reference values. It improves the numerical stability for predictions. Instead of directly predicting the bounding box's coordinate, the network predicts four values t_x, t_y, t_w, t_h for each cell in its output feature map, visualized in Figure 2. In order to convert the predictions to actual bounding boxes $b_{x_c}, b_{y_c}, b_w, b_h$ we use the following equation:

$$\begin{aligned} b_{x_c} &= \sigma(t_x) + c_x \\ b_{y_c} &= \sigma(t_y) + c_y \\ b_w &= \exp(t_w) * p_x \\ b_h &= \exp(t_y) * p_y \end{aligned} \tag{1}$$

where as (x_c, yc, w, h) are centroid and size of the bounding box. (c_x, cy) are the offset of current cell from the top left corner. (p_x, py) are the anchor size.

3.3. Loss Function

We set up the loss function as a weighted sum of localization loss, object confidence loss and classification loss. We treat the detection and classification as regression problem. Also, we only calculate classification loss for predictions having $IOU_{pred}^{ground_truth}$ greater than 0.5. We still penalize predictions of incorrect localization and object confidence. We formally describe the loss function as following:

$$\begin{aligned} Loss &= L_{localization} + L_{objectness} + L_{classification} \\ &= \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}[(x_i - \hat{x_i})^2 + (y_i - \hat{x_i})^2] \\ &+ \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i}])^2] \\ &+ \lambda_{obj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (C_i - \hat{C}_i)^2 \\ &+ \lambda_{no\_obj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{no\_obj} (C_i - \hat{C}_i)^2 \\ &+ \lambda_{cls} \sum_{i=0}^{S^2} \mathbb{1}_{ij}^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 \end{aligned} \tag{2}$$

where as $\{\lambda_{coord}, \lambda_{obj}, \lambda_{no\_obj}, \lambda_{cls}\}$ are weight terms. We reused author's original weight values as $\{5.0, 5.0, 0.5, 1.0\}$ [10]. $\{x, y, w, h, c, p(c)\}$ is the ground truth of bounding box's centroid (x, y), its width and height, object confidence, and class probability. Similarly, $\{\hat{x}, \hat{y}, \hat{w}, \hat{h}, \hat{c}, \hat{p(c)}\}$ are prediction values from network. $\mathbb{1}_{ij}^{obj}$ indicates if there is an object in cell i and j^{th} bounding box predictor. S and B represent number of grid cells and number of anchors. In this work, we use 5 anchors ($B = 5$) and S is ranging from 14 to 30 (variable image size from 448 to 960).

3.4. Hierarchical Soft-max Tree

One challenge of classification problem is when number of classes is large, which makes it harder for the network to learn and converge if we apply soft-max on the whole feature vector. Secondly, this implicitly assumes that each

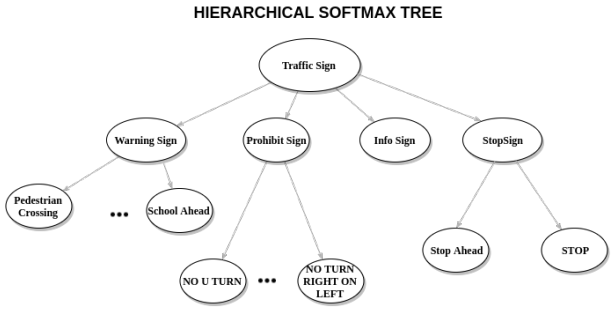


Figure 4. Each label has a relationship through their parent

label is mutually exclusive.

In this work, we introduce a hierarchical soft-max tree. We make an assumption that each class in the dataset has inherent relationships [2]. For example, a speed limit 45 and speed limit 60 can be inferred as Speed Limit sign. Also, Speed Limit Sign and Stop Sign are also a Traffic Sign. We argue this technique allows the network to use prior attributes of abstract classes to learn better. More importantly, we reduce the length of the soft-max vector by only applying soft-max on classes having the same parent. Our loss function remains almost the same. The difference is that we need to build a soft-max tree so that during training, the network knows how to calculate the soft-max properly.

4. Experiment on LISA Dataset

In this section, we will validate the model and hierarchical soft-max tree on both LISA and LISA extension dataset. We specify our training hyper-parameters and how we trained the network. Then, we evaluate our experiment result and run-time performance.

4.1. Training

Transfer Learning. We used a pretrained YOLOv2 608x608 model on COCO [10] to perform transfer learning. We replaced the last layer of the network to match the size of labels and anchors in our dataset. The number of features on the output layer is calculated as described in Figure 3. Due to good weight initialization, we lower the learning rate to 0.0001. Our implementation was on Keras/Tensorflow.

Input Resolution. In order to improve the accuracy of model, we train YOLOv2 on multiple spatial inputs ranging from 448 to 960. Higher image resolution can result in better mAP, however it would dramatically slow down the network [5]. In case of traffic sign detection, it is important to keep the input resolution high, otherwise we might lose important features due to max-pooling layers.

LISA Soft-Max Tree. We combined two datasets through Hierarchical Softmax Tree. We created five abstract classes: *Warning Sign*, *Prohibit Sign*, *No Turn Sign*, *Information Sign*, and *Speed Sign*.

For our experiment, we split the LISA dataset and LISA extension dataset into training and testing set at ratio of 0.2. During training, we shuffled the data randomly in order to avoid over-fitting and performed data augmentation as suggested [10].

Optimizer	Adam
Batch Size	8
Learning Rate	0.00001

Table 2. Hyper-parameters for training YOLOv2 on LISA Dataset

4.2. Evaluation

We measure detections using standard procedures as suggested in PASCAL VOC [3]. We set the IoU threshold as 0.5 to compute the mean average precision (mAP). In this work, we only evaluated the detection on all classes, not a specific type. The test dataset is from the split dataset setup previously before training. The network does not see this data during training.

For interference evaluation, we measure a single forward pass. Our measurement was tested on CPU Intel Core i7 4790, 32GB RAM, and GPU NVIDIA GTX 1080.

Experiment Result (IoU = 0.5) - LISA Test Dataset			
Network	Resolution	mAP	Foward Pass
DarkNet-19 YOLOv2	608x608	76.8	36 ms
MobileNet YOLOv2	960x960	82.8	20ms
DenseNet YOLOv2	960x960	90.8	55ms

5. Conclusion and Future Work

In this work, we have discussed YOLOv2 network architecture and how it is trained on LISA, a US Traffic Sign Dataset. During the training, we also experimented by swapping different feature extractors, particularly DenseNet and MobileNet, to compare the accuracy and interference.

In the future, we would like to train the network on larger dataset with various different scenarios. In order to improve high speed during run time, it is important to find a way to remove batch normalization. Super input resolution and shallow network are also promising to improve the detection [1]. Finally, we might also consider to use Recurrent Neural Network (RNN) on top of our current detection model for accurate tracking.

6. Acknowledgment:

Special thanks to Dr. Christoph Mertz, research assistant Jina Wang for guidance and mentorship. Also, thanks to our colleague at NAVLAB for great feedback and suggestions. Finally, special thanks to Robotics Institute Summer Research Program for providing us a wonderful opportunity to work on state-of-the-art problems.

References

[1] K. Ashraf, B. Wu, F. N. Iandola, M. W. Moskewicz, and K. Keutzer. Shallow networks for high-accuracy road object-detection. In *VEHITS*, 2017. 4

[2] J. Deng, N. Ding, Y. Jia, A. Frome, K. Murphy, S. Bengio, Y. Li, H. Neven, and H. Adam. *Large-Scale Object Classification Using Label Relation Graphs*, pages 48–64. Springer International Publishing, Cham, 2014. 4

[3] M. Everingham, ·. Luc, V. Gool, C. K. I. Williams, J. Winn, A. Zisserman, M. Everingham, L. V. Gool, K. Leuven, B. C.

K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision manuscript*. 1, 2, 4

[4] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten. Densely Connected Convolutional Networks. aug 2016. 1

[5] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. nov 2016. 1, 4

[6] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015. 1

[7] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 2

[8] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single Shot MultiBox Detector. dec 2015. 2

[9] A. Møgelmose, M. M. Trivedi, and T. B. Moeslund. Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey. *IEEE Transactions on Intelligent Transportation Systems*, 13(4):1484–1497, dec 2012. 1

[10] J. Redmon and A. Farhadi. YOLO9000: better, faster, stronger. *CoRR*, abs/1612.08242, 2016. 1, 2, 3, 4

[11] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. pages 91–99, 2015. 2

[12] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2

Visual system for a Modular 6-Legged robot

Kartik Patath, Hadi Salman, Howie Choset

***Abstract*— Search and rescue has always been a critical question to be answered in times of natural disasters. The destroyed areas are difficult and sometimes impossible for humans to access. Biologically inspired legged robots can be quite effective at search and rescue operations, especially in confined areas where humans cannot reach. These robots can reduce risk levels of human rescuers. This creates an opening to new research area in the field of search and rescue. In most of the scenarios the legged robots work using compliance, for motion in unknown environments. We propose a vision system for a 6-legged robot which enables it to better perceive the environment, helping the robot to navigate in an unknown environment.**

I. Introduction

Search and rescue takes many forms, each with its own unique risks and dangers to victim and responder. Urban search and rescue has become familiar to many of us as recent natural and man-made disasters have made the news all over the world. In some situation such as wilderness search and rescue there will often be only one victim with a large geographic search region. There are also numerous recent mine collapses in the United States and China [1]. These rescue operations are mostly performed by humans. There are significant dangers rescuers face during search and rescue tasks in a uniquely hazardous operating environment, such as mines for example. In case of destroyed buildings, there are many voids that are simply too small for people and dogs to enter, thus limiting the search to no more than a few feet from the perimeter. Robots on the other hand can bypass the danger and expedite the search for victims immediately after a collapse. This advanced technology can be useful to the rescue workers in ways such as: (1) reducing personal risk to workers and dogs by entering unstable structures, (2) allowing to penetrate ordinarily inaccessible voids, (3) increasing the efficiency by searching areas with multiple sensors, to provide a complete search in three dimensions, and (4) extending the reach of rescuers to go places that were otherwise inaccessible.

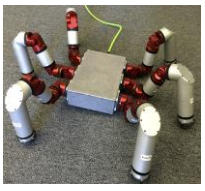
In this paper, we consider a platform for doing search and rescue known as the Snake Monster. It is basically a 6-Legged robot. The Snake Monster is reconfigurable robot with reliable field-tested hardware. By using parallel position and force control, the Snake Monster can actively and passively conform its footsteps to traverse unmodeled terrain with relative ease and minimal computation. Using the mechanical spring compliance built into each joint, this robot conforms to uneven terrain rather than explicitly planning and optimizing footholds and step locations.

So far, the snake monster had no vision system, and used compliance to understand the environment. Compliance aims towards either process improvement (active) or human safety (passive), more about compliance can be found here [2]. During a compliant movement, the robot interacts with the environment and changes its behavior depending on whatever forces it senses. Tasks such as stair climbing, crawling, obstacle avoidance add much more complexities in coming up with a compliance algorithm. Even though we come up with a suitable algorithm, it will be having many constraints that need to be kept in mind. Adding a vision system to the current mechanical structure, eases many of the constraints. Hence, we come up with a vision system and some methods that can be implemented using such a system to improve the robot's capabilities. We demonstrate that the vision system is useful for better perceiving the environment compared to simple compliance system.

II. Modeling

A. Mechanical and system architecture

The six legs have a reach of 12 inches (30 cm), and are connected to a rectangular body, with the whole robot weighing 18 pounds (8 kg). The robot can move with an alternating tripod gait, with three legs in the air always. The robot can also operate in quadruped configuration. The robot has series elastic actuators (SEA) placed at each of its joints. More information about SEA can be found in [3]. Fig 1-b shows a series elastic module that has been developed in the Biorobotics lab and used as a building block of the snake monster. The modules use ethernet communicates through ethernet. Each of these SEA modules are embedded with Inertial measuring unit (IMU) and a torque sensing unit. These provide a base level component for the compliance model.



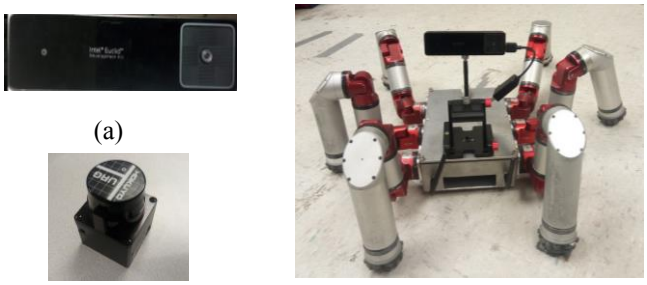
(a)

(b)

Fig. 1. (a) Snake Monster without a vision system, simply based on torque sensing (b) a series elastic module

B. Vision Architecture

The vision system of the robot consists of Intel Euclid, which is the integration of a depth camera and a motion camera. Along with an Intel-Atom™ x7-Z8700 Quad core CPU to produce an all-in-one compute and depth camera device in a compact and sleek package. We use this sensor to reduce the computation burden on the on-board processor. Unlike the traditional cameras which just pass the sensor data to the onboard computer, here the sensor itself is a unit of perception. Hence, the vision system perceives the environment and sends high level commands to the on-board computer. These high-level commands will be used for the functioning of the robot. We can observe the modularity in this system, which makes this model more versatile and reusable with similar sensors.



(a) (b) (c)

Fig. 2. (a) Intel Euclid camera (b) Hokuyo URG Laser, 2D-LIDAR (c) Then Snake Monster along with the vision system mounted on it.

III. Methods

In this section, we list the various vision algorithms and tools that we tested and used on the Snake Monster to allow it to perceive its environment. We will talk about a SLAM algorithm (LSD slam), a mapping algorithm (Hector mapping), and a tracking algorithm (person recognition and tracking)

A. LSD SLAM

In this section, we briefly describe the LSD (Large Scale Direct) algorithm. For more details, the reader is encouraged to refer to the original LSD paper [4]. LSD is a direct featureless monocular SLAM algorithm that allows to build large-scale consistent maps of the environment. The algorithm uses highly accurate pose estimation based on direct image alignment, and reconstructs the 3D environment in real-time as a pose graph of keyframes. The global map is represented as a pose graph consisting of keyframes as vertices with 3D similarity transforms as edges, and this allows for incorporating multi scale environment nature. This algorithm runs in real-time on a CPU.

The algorithm consists of three main parts which are tracking, depth estimation, and map optimization.

For tracking, the rigid body pose is estimated with respect to the current keyframe. This part continuously runs at the highest frequency of the pipeline. The depth map estimation part basically uses the tracked frames from part one to either replace the current frame if some conditions are met, or to refine this current frame. The third part which is map optimization consists of including the keyframe that has been replaced in the previous part (depth map estimation), and thus means that it is not going to be refined anymore, into the global map. This parts also contains loop closure and scale drift detection which is done by estimating a similarity transform using a method called direct-sim(3)-image alignment (more details in the original paper [4]).

For initialization, the first keyframe is initialized with random depth map and large variance. LSD best initializes if a sufficient translational camera movement is detected in the first few seconds.

As of most state of the art SLAM algorithms, the map is represented as a pose graph of keyframes that captures the dependencies between all the keyframes that are added to the map. Each keyframe consists basically of an image, an inverse depth map, and variance of the inverse depth map. The edges of the pose graph represent similarity transforms between the keyframes.

B. Hector mapping

Hector mapping is a SLAM approach which does not use the odometric information, it works well on platforms which do not exhibit much of roll, pitch and yaw. It leverages the high update rate of the Laser to construct the occupancy grids. It constructs good resolution occupancy grids for environments which are smaller in area and have a lot of boundaries. The draw backs of this system are it cannot account for loop closure. It is sufficiently accurate for many real-world scenarios, more can be found in[5].

C. Person recognition and tracking

In this section, we describe our implemented method of person recognition and tracking. Person tracking enables the robot to recognize and follow a specific person. This gives the robot an ability to differentiate to obey the commands of this person, and enables it to differentiate this person from other people. We fuse the person tracking and color detection algorithms to track a specific person among a crowd of people. We provide the information about this person's face and the color of the costume that he is wearing. First the robot detects a person's face and after detecting the face it uses a bounding box to track the person. We extract the geometric center of this bounding box by computing centroid by taking the mean over the box and use that to stabilize the position of the Snake Monster. After tracking a person's face and constructing a bounding box, the robot tries to match the color of the person's costume, with the current color in the bounding box. If both the color and a face are detected, then the robot tracks this person, otherwise, it stays in its place.

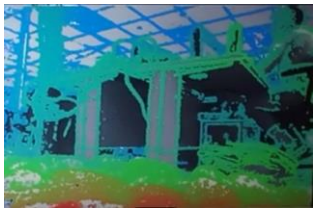
IV. Results

In this section, we describe the results that we got from applying the above-mentioned vision algorithms on different data sets collected using the Snake Monster.

A. LSD SLAM

Using the monocular sequence of images generated by the on board monocular camera, pointcloud of a rocky terrain was generated. The reason for using a monocular camera was based on an effort to make the vision system cost effective. The LSD algorithm was tuned such that it stays invariant to high camera noise. The camera captures images at 30Hz. The information was recorded in a bag file and played back at half the recording speed. The results shown in Fig. 2. (a) are the results for offline pointcloud computation. The reason for choosing offline computation was due to the high computation time demand, by LSD algorithm. As the size of the map increases with time, the algorithm starts slowing down. As a result, the time taken to construct the newer regions in the map increases. This delay in the map construction sometimes gives you an error of dropping frames during to the computational overburden. This leads to either less informative regions or corrupted maps (maps with too many) outliers.

(a)



(b)

(c)

Fig. 3. (a) LSD generated pointcloud (b) Disparity map of the current scene with respect to previous scene (c) The real image of the rocky terrain.

B. Person Tracking

We modified the person tracking algorithm given written by Intel so that it tracks only people with yellow costumes. As you see in the following figure, the algorithm correctly tracks the person who is wearing a yellow coat but doesn't track the same person if he is earing another color. For a demonstration, we made the robot follow a person who wears a yellow coat.

(a)

(b)

Fig. 5. (a) Person with yellow coat is tracked (b) Same person without coat not tracked

V. Future Work

What we have included in this paper so far is basically a bunch of vision based algorithms that enables any robot of perceiving its environment, and performing intelligent tasks that wouldn't have been possible otherwise (person tracking). Yet, a lot of additions and improvements can be done on the Snake Monster platform in the presence of all the sensor that we mentioned previously (Intel Euclid, 2D lidar, monocular camera, IMU). One extension of our work is basically combining the IMU readings with the 2D map built by the 2D-lidar, which will allow us to avoid drift and allows for loop closures. Another possible extension of our work is using the Euclid to reconstruct the terrain in front of the Snake Monster, thus allowing it to avoid obstacles during its navigation. Furthermore, we are planning to enhance the intelligence of the Snake Monster by allowing it to recognize hand gestures of a person, and doing actions accordingly.

Acknowledgment

I humbly acknowledge Hadi Salman, Puneet Singhal and Matthew Travers for their support in the project.

References

[1] https://en.wikipedia.org/wiki/Sago_Mine_disaster
[2] A. Albu-Schaffer et al., "Soft robotics," in IEEE Robotics & Automation Magazine, vol. 15, no. 3, pp. 20-30, September 2008. G. A. Pratt and M. M. Williamson, "Series elastic actuators," Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots, Pittsburgh, PA, 1995, pp. 399-406 vol.1.Ref 3.
[3] Engel J., Schps T., Cremers D. (2014) LSDSLAM: Large-Scale Direct Monocular SLAM. In: Fleet D., Pajdla T., Schiele B., Tuytelaars T. (eds) Computer Vision ECCV 2014. ECCV 2014.
[4] Stefan Kohlbrecher, Johannes Meyer, Thorsten Graber, Karen Petersen, Oskar von Stryk, and Uwe Klingauf, "Hector Open Source Modules for Autonomous Mapping and Navigation with Rescue Robots".

Student Behavior Analysis using Affective State Estimation in RoboTutor

Rohith Krishnan Pillai[1] and Mayank Saxena[2]

Abstract— **Intelligent tutor systems have been around for many years with many that cater to different levels of education, languages, and demographics. Recently there has also been a significant amount of research in trying to make such tutor systems reactive to students to allow for a more intuitive interface with the tutor and also more importantly to be able to keep the student engaged in the activity for longer periods. Most prominently using affective state estimation or emotion detection as a way to discern the students engagement levels. Current research mostly focuses on making automated affective state detectors to be used in conjunction with the tutors, whereby implementing their intervention policy to keep students engaged. We discuss an alternative computer vision based method of perception for the intervention policy, which is not depended on the affective state of the student but on the anticipated behavior of the student. In this paper we report the preliminary work for such a system, by analyzing the best affective state predictors for behaviors such as quiting an activity etc., and also outline the development of an android application that detect affective states in order to predict the most probable student behavior, in the context of RoboTutor, an android math, reading and writing tutor for use by children age 6-12, who have little or no prior schooling nor access to technology.**

I. Introduction

Over the last decade there has been a significant increase in the usage of intelligent tutor systems, mobile applications and massive open online courses (MOOCs) which has opened up access to education to a wider audience both geographically and temporally as compared to traditional means. However, these predominantly provide content related to higher education, while fewer cater to children who are non-English speaking and have no little or no access to education. RoboTutor is an Android application that teaches basic arithmetic, reading and writing to children specifically in this context, who have little to no prior schooling and/or access to technology. RoboTutor is currently being used by children between the ages of 6 to 12 in two sites in Tanzania for testing effectiveness and stability.

Similar to traditional teaching methods, intelligent tutor systems (ITS) such as RoboTutor are only effective when student is engaged in the content being taught to them. A tutor that can keep the student engaged on a task for a longer period of time is more likely to be able to successfully teach a knowledge component, as compared to a student who is disengaged by the tutor. This is one of the reasons why traditional classroom teaching is more effective than any state-of-the-art ITS. The teacher in a classroom is able to gauge the engagement levels of the students and dynamically adapt the content or it's presentation in real-time. Similarly, predicting engagement and disengagement of the students is a task that can add great value to an ITS and can make it more efficient at helping students learn. Affective state estimation and emotion detection from facial action units is a technique that has been used to analyze the engagement and disengagement levels of students in many ITS. [1] In our research we look into the use of affective state estimation for the prediction of the student behavior while using the application such as answering correctly vs. incorrectly, quiting activities in between them, and 'good' vs.'bad' performance (where 'good' and 'bad' are criteria based solely on the score of the student on a particular activity).

A. Research contributions

There have been numerous studies trying to create automated affective state estimation using computer vision built for use in intelligent tutor systems cite which have been used to understand the effectiveness of the intelligent tutor system on the student. Many of the work done in this field has established which affective states and facial action units needs to be studied for an educational setting [1], [8], and more frequently on how to build such automated systems on affective state estimation for analysis of engagement. However there has been very little work on how to use these automated affective state detectors to analyze and predict student behavior on a tutor. In addition, none of these systems have used the data collected for the prediction of student behavior during tutor use. For instance, the behavior action of 'quiting' in-between an activity. Hence, in this paper we discuss the work done in relating the student affective states/emotions with the student behavior while using the tutor in order to create a system that can predict in real-time what a student's next action could be to the tutor so that the appropriate action could be taken.

Furthermore, the challenges faced by deploying similar emotion detection for RoboTutor are numerous due to the drastically different environments and conditions in which the android application is meant to be used, as opposed to most other studies. For instance, most of the previous studies have been conducted on adults and college students, indoors and in controlled, well-lit environments, and can use multiple sensors in addition to a camera, such as Microsoft Kinect and other wearable devices. Due to the intended context of RoboTutor which is to be used by children with no prior

[1]Rohith Krishnan Pillai is with the School of Computer Science, Carnegie Mellon University in Qatar, Ar-Rayyan, Qatar. E-mail: rohithkp@andrew.cmu.edu

[2]Mayank Saxena is with the Department of Computer Science, Delhi Technological University, Delhi, India. E-mail:mayank26saxena@gmail.com

access to education in remote areas, these scenarios are not applicable. RoboTutor is built for use by children which leads to even more challenges for affective state estimation using computer vision techniques. These challenges include the detection of smaller faces, and the different range and peak intensity of emotions between adult and children's faces. RoboTutor as a tablet application also has the limitation that the affective state detection system has to be able to use a simple front facing camera found in the tablet, and without the use of any additional wearable devices. The technological divide that exists between the developed and developing nations also create many human-computer interaction challenges to the mix. This means that the sparse use of technology also a necessity to not discourage the use of the tutor due to the lack of or little prior exposure to technology of the children. Hence, even though heart rate has been shown to be a good predictor of affective states [14], we opt for using computer vision due to it's non-intrusive nature. Therefore, in this paper we discuss our results in using affective state estimation using computer vision to analyze the behavior of children in order to better understand the link between emotional engagement of the student and the behavioral actions exhibited by them. In particular our goals are to answer these questions: Can we apply a novel approach of affective state estimation on children belonging to different demographics using existing open source tools? Given the additional information of the students affective state, what can we infer about the student, the different activities and the subject categories? Is it possible to predict the students next action based on their affective states? For example, can we make a real-time system to detect when the student is about to tap the back button in-between the activity? These questions if answered can lead to better performing ITS in a wide variety of education levels.

II. Background and Related Work

Most of the work done previously in this field has focused on detecting basic emotions such as happiness, sadness, contempt, fear, anger and disgust. None of these emotions are relevant in the educational setting. In an educational environment, emotional states such as frustration, confusion, delight, boredom, surprise and neutral are more relevant and more informative [5], [8]. We acknowledge that the emotional states mentioned above are not an exhaustive set of all the emotional states exhibited in a given learning environment, but these are the most common ones which are seen while kids interact with RoboTutor as well as through prior research. There have been studies which suggest that emotional states like boredom and frustration are negatively correlated with learning [4].

The most prominent work done in the field is done by Sidney D'Mello [4], [5]. In more recent years, a mathematical correlation has been identified to exist between specific facial action units and affective states. For example - Brow lowering (AU4) and eyelid tightening (AU7) were correlated with confusion, while Lid Tightener (AU7), lip corner puller (AU12), lips part (AU25) and jaw drop (AU26) are associated with delight. [5], [6]

More recently, work has is also being done on integrating real-time facial expression analysis into tutor systems. [3], [5]. Integrating behavior tracking into ITS is a necessary step toward meaningful real-time interventions. There has also been recent research that may lead to robust sensor-free affect detection [11].

As discussed before, one of the most important behavioral action which is accompanied with learning is engagement. Many authors [12] agree on four types of engagement which are observed during learning -

1) Behavioral engagement - Can be assessed by observing persistence and effort
2) Emotional engagement - Can be assessed by detecting supportive emotions (e.g., interest, curiosity) and self-efficacy
3) Cognitive engagement - Is demonstrated when the student shows a sophisticated approach to the activity, for example by using deep rather than superficial learning strategies.
4) Agentic engagement - Occurs when the student attempts to actively enrich the experience, instead of merely acting as a passive recipient.

Engagement can be measured from different behavioral expressions: eye-gaze movements, facial features, gestures, and so on. Nakono and Ishii [15] in their work showed that considering the users engagement has a positive effect on the users experience during their interaction with the robot. Rich et al. [2] in their work proposed a framework to detect and track user engagement using eye-gaze tracking, speech and gesture tracking. [9] says that eye-gaze is one of the most useful indicator to evaluate a students attention and engagement. It is also very helpful in figuring out instances of mind wandering and zoning out.

[8] identified the facial action units which are accompanied during the experience of educationally relevant emotions including confusion, frustration, surprise etc. It argued that once the affective states of the person interacting with the tutor can be detected, then educational software in general can revise their pedagogical strategy by taking into consideration both the affective state of the learner as well as their cognitive state.

Similar to some related work [7], we detected emotional engagement with a video-based approach using computer vision techniques. We used OpenFace [13], a state of the art, open source facial behavior analysis toolkit to find facial action units related to educationally relevant emotional states.

III. Methods

In order to create an application that can automatically detect the affective states of the student actively using RoboTutor and provide the tutor with important clues and hints about the most possible behaviour, we first need to be able to create a method that would allow us to use the emotions detected online to predict the in-app behavior of the student. Such a task, can be carried out by the use of

machine learning techniques, but requires annotated data, to create a model of the student. A very important precursor task that needs to be carried out includes the need to be able find possible features that could be decent predictors for the set of in-app behaviors that we are interested in. Here we will outline the pipeline that we created for the purpose of being able to identify the correlations between the affective states and the students in-app behavior to figure out if such predictors, correlations and patterns exists and if so then what they can also say about the efficiency of the tutor. We also detail the secondary process of using the model learned to create an android application with the intermediary task of being able to automatically detect student affective states. An overall outline of the analyses pipeline for the project is given in Figure 1.

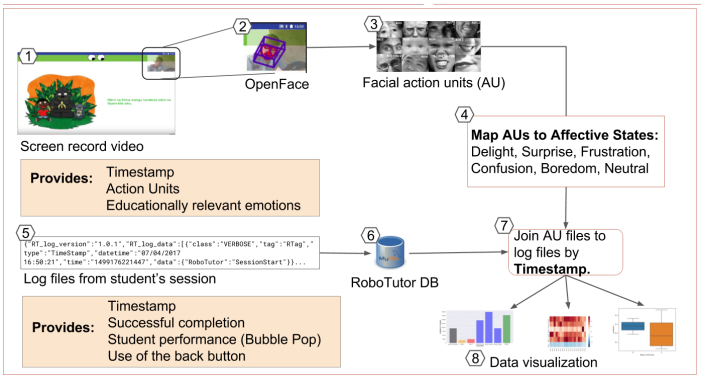


Fig. 1: Pipeline for student behavior and affective state analysis

A. Data Collection

RoboTutor is an application that is made for children between the ages of 6-12, and contains tutors for basic arithmetic, reading and writing in both Swahili and English languages. Currently RoboTutor is being field tested in 2 test sites in Tanzania, with 2 android tablets. To be able to do affective state estimation on the student we require video of the students using the application with at least a reasonably direct angle facing the camera. Since the application runs on android tablets, and given the fact that all most all the given tablets in our 2 testing sites in Tanzania contained front-facing cameras, the video recorded from these cameras were used for our research. However, the data collected in the testing sites for RoboTutor uses a proprietary screen capture application that provides both the screen video and a smaller scaled down inlet of the front facing camera. These video files all had both the screen video and the front facing camera video, as they served the purpose for helping in debugging and tracking student actions, demographics etc. However, due to the nature of the android operating system locking resources such as camera and microphone access to a unique process, having another service or application record video alongside the screen capture is not possible, and hence the videos used for the emotion detection had to be extracted from the screen capture video. Similarly, the resource locking by the android operating system also impairs the use of sound from the videos as one of the 2 tablets, deploys the speech recognition module of RoboTutor for the reading tutor and hence all the videos were made mute and cannot include a speech based analysis either.

TABLE I: Encoding for Facial Action Units to Emotions

Emotion	Pattern	Action Units Description
Frustration	1,2 1->2	Inner and Outer Brow Raised Together Presence of an inner brow raise will trigger an outer brow raise and vice versa
Confusion	4,7 4->7	Brow lowered with tighened lids Tightened lids will lead to lowered brows
Surprise	1 2 5 26	Inner brow raise, Outer brow raise Upper lid raiser Jaw drop
Delight	7 12 25 26	Lid tightener Lip corner puller Lips part Jaw drop
Neutral	-	Absence of all action units

There are 2 kinds of data that was collected for the analysis. The first one is the video files as explained previously and the second were the log files from the running of the android tutor. The log files keep a track of all the components that were used in each activity, the correctness of the student inputs, and also various behaviors that can be interpreted from it. These log files are automatically generated upon use of RoboTutor application. In our case, log files provide information regarding the students actions, performance during the sessions, and overall in-app behavior of the students. We used video log data of children interacting, playing and learning with RoboTutor and we used the corresponding log files for the associated video to calculate their performance and for predicting their behavioral action. In total, we had 17 log files in JSON format and their corresponding video log files ranging from 20 minutes to 30 minutes, of children playing with RoboTutor. Since we dont have the a method to do facial recognition yet integrated fully into the field testing versions of RoboTutor, we cannot be sure about the student who was using the tutor, and so we assume that all 17 sessions are unique student sessions.

B. Data Processing

There was a lot of preprocessing of data which was involved in order to get it into a format through which we could query and calculate the required metrics. Some of the steps involved in the preprocessing of data included parsing the log files, inserting the data into the database and finally finding out instances of good vs bad performance, instances of children transferring knowledge from one activity to another. For instance, if the child came across the word ”hujambo” in the reading task, could the child identify the same word in the bubble pop activity or not.

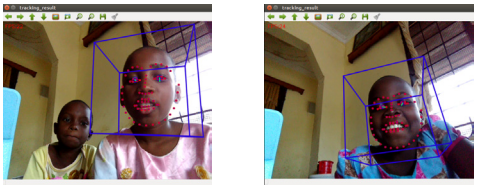


Fig. 2: Children playing with RoboTutor

The log file contains the information regarding the time of the activity, the name of the activity, the successful completion of the activity, performance with reference to correctness in the activity, and the use of the back button. Important information regarding the session such as the session unique identifiers and data features such as the completion etc. are extracted into a more easier to read format of a csv file and uploaded into the RoboTutor database. The database has multiple tables generated from the log files which are uploaded to it, and the in particular we use the data from the derived view-table that provides the list of activity change sequence for each of the sessions to make the final table which contains the data features and the activity at each timestep.

The next step in our pipeline was the task of doing video data processing in order to retrieve affective states from the student videos that we have. Since the front-facing video is the only thing that we are interested in from the original video we crop the screen capture footage for each session into the roughly only the front-facing camera video inlet on the video. We then feed each of these cropped video files with just the front-facing camera video to a state-of-the-art, open source tool for facial behavior analysis. We decided to use an existing open source tool instead of creating our custom tool to do the same due to the quick time for deployment. In addition, we used OpenFace instead of other softwares such as CERT was that OpenFace was highly customizable, and also contained additional facial expression analysis tools, such as gaze tracking, head pose estimation etc, that could be used for further analysis in the future. In particular, we used the Feature Extraction module of OpenFace [13] to find the presence or absence and the intensity of the facial action units.

Fig. 3: Distribution of Action Units.

Facial action units (AU) are small changes in the facial feature that are expressed such as lip corner puller (AU 12), inner brow raising (AU 1), and cheek raiser (AU 6) that can together be used to decompose an emotion. OpenFace is able to recognize a subset of the 64 AUs, specifically: 1, 2, 4, 5, 6, 7, 9, 10, 12, 14, 15, 17, 20, 23, 25, 26, 28, and 45. Hence using OpenFace on our cropped video data provided us with the frame by frame breakdown for the the different action units that were detected, along with their confidence levels. However, we are interested mapping these AU back to the affective states that matter to students in a educational setting, and these are different from the normal set of 5 affective states that are returned by most other computer vision based emotion detection software like Microsofts vision API etc, that only provide a value for the emotions of anger, contempt, disgust, fear happiness, neutral, sadness, surprise. Prior research has informed us that such emotions are not very relevant to an educational environment where learning is being measured. Instead the affective states of delight, surprise, confusion, frustration, neutral, boredom are more commonly seen and their relations to learning also more widely studied and accepted.

Mapping the AUs that we could detect through the OpenFace interface to the 6 educationally relevant affective states that we are interested in happened through looking up existing literature to query the correlation between the different AUs and these emotions. These mapping found from literature can be found on Table 1. We use the correlation factors to guide us in weighting the AUs to create a linear weighted average of the AUs for each of these 6 affective states at each timestep and session file. This creates the numerical values for the emotions that we normalize across a timestep, to create the normalized affective state value and the maximum likelihood affective state for that particular timestep. Finally the actions/behaviors file that were queried from the RoboTutor database are joined with the affective state estimate file by timestamp interval matching.

Once we had this information regarding all the videos, we had enough data to do our analysis on and to visualize our data. To begin with, we found out the number of instances of occurrences of the 6 affective states, and their distribution across RoboTutor. We also aggregated these affective states over different components of RoboTutor as well to gain a more detailed view. These aggregations included the distribution of emotions over different activities, distribution of emotions over each session and lastly the distribution of affective states in a good performance vs a bad performance. Some of our results are given in the results section.

C. Android application for affective state estimation

We also worked simultaneously on the problem of detecting these very 6 affective states directly from a set of 68 landmark units, in a real-time android application. To do this we relied on the dlib facial feature recognition library which is popularly used in face detection tasks to return the 68 landmarks on the student's face. We ran the dlib library on the extended Cohn-Kanade (CK+) database phase 1 [10] in order to retrieve the training data for the classifier. We then also ran OpenFace on the CK+ phase 1 dataset to retrieve the

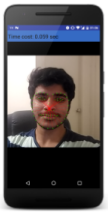


Fig. 4: Android app running affective state estimation.

AUs for the data, and subsequently mapped these AUs using our weights assigned to the affective state using the Table 1 mappings, to obtain the maximum likelihood affective state for each of the images. Each instance in the training data contained the 68 landmark points as well many pairwise vectors between landmarks, with affective state as a label. The training data contained approximately 1000 instances as we also tried to use images from the CK+ dataset that were not in peak emotion either. The training accuracy was as high as 97-98% using a multi-layer perceptron neural network, 18 layers deep. We used Googles tensorflow to create the model that was trained on the training data set and ported to our android application in order to use the model on the live frames obtained from the camera to publish the affective state detected. Figure 4 shows a screen-shot of the application built, with the 68 landmarks being detected and displayed on screen, overlaid on the live video feed.

IV. Results

The data visualization of the data that we processed gives rise to some interesting findings in that shows promise in answering the questions that we asked, about the possibilities of a relation between the student's affective state and the in-app behavior exhibited.

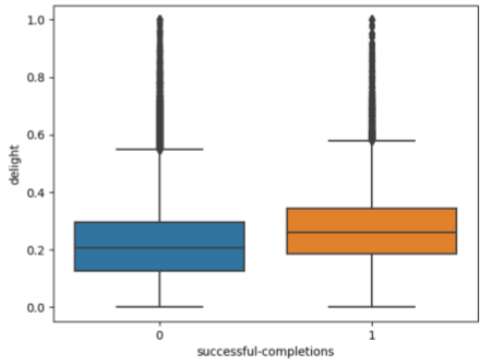


Fig. 5: Delight in successful completion vs incomplete.

The Figure 5 shows that the median delight that students expressed across all activities was much higher than when the student chose to not complete the activity. The spread for both these cases are almost similar, with multiple outliers due to the large dataset. The difference in delight between the two groups is statistically significant as the Wilcoxon's rank test, returned a p-value < 0.05 at a confidence value of 95%. The Wilcoxon test was used instead of the t-test due the skewed distribution of the data. Figure 6 suggests tries

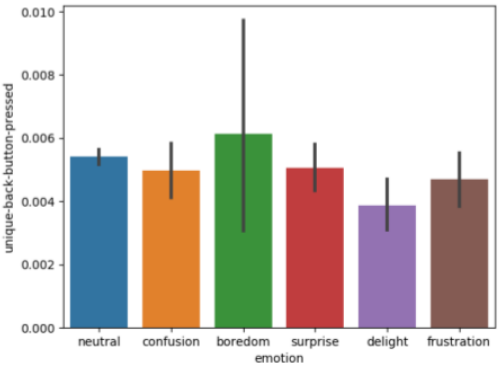


Fig. 6: Affective states plotted against instances of student prematurely exiting the activity.

to look at the distribution of the affective states exhibited by the students when the students hit the back button while using the application. In this case the highest value seems to be deceivingly of the emotion boredom, however, on significance testing it is revealed that only the emotions of neutral/flow, surprise and delight are statistically significant. Of these certainly delight is lower than all other emotions in terms of quiting an activity midway, which is consistent with the results from Figure 5. The p-value for surprise is 0.038 at a confidence value of 95%.

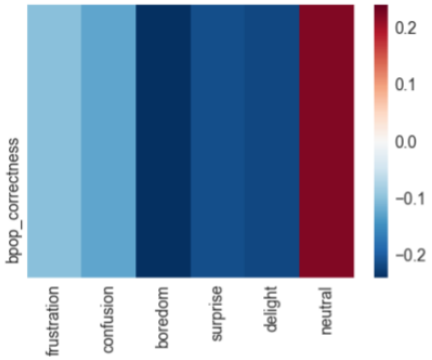


Fig. 7: Heatmap of correlation between correctness and emotions.

We can also see that from Figure 7 the only positive correlation of correctness in the bubble pop activity in the RoboTutor application is the emotion, neutral/flow. This is in stark comparison the the emotion of boredom which seems to be the emotion that least correlates with the student answering correctly on the tasks. Other emotions are also mostly negatively correlated with correctness and performance in the bubble pop activity.

V. Discussion

The main goals of our research included figuring out if there are any interesting links between student behavior

and their affective states in an intelligent tutor system like RoboTutor. We are especially interested in understanding if these affective states be used to predict students behavior in real-time. The results that we have presented in Figures 5-7 show that there is in fact a link between the affective states and the students in-app behaviors such as performance on activity tasks and completion of activity. We can also infer from the results the following. The very first observation is that successful completions of the activities are much more likely when the students are in the affective state of delight. This also prove that the our methods are consistent with the expectation, as we would expect that students be happier when they feel like they have accomplished an objective versus when they don't. This also shows us that RoboTutor is working as expected and validates it. Most important take away from this observation is the fact that delight could be a predictor of completion of activity. Similarly, the second observation is that surprise is good indicator of the students hitting the back button during an activity. Although neutral is higher, it is also a more uniformly distributed emotion across the activities, but surprise seems to be high enough to distinguish itself from the emotion delight, and clearly shows the links between itself and the quitting an activity. Also from Figure 7 note that correctness in different activities can also be predicted using a certain affective state, such as neutral/flow. This makes sense as neutral/flow is most notably seen in students when they are actively learning, and processing the information presented, and so students would be more likely to get tasks right in this context. All these showcase the fact the we can definitely see links and possible predictors for different behaviors on RoboTutor by carrying out the methodology described in this paper, which could be used to create the real-time system for finding optimal timings for the intervention policy to be used to keep the student engaged in the app. We believe that this methodology and technology can be extended to more platforms and that this will help in all tutors become more personalized. This will help decrease the dropout rate in MOOC's as well as once the application will be able to predict that user is about to quit, it could change its strategy and display content which engages the user more.

This research can further be extended to help design better suited games for different audiences. For example, our results show that students aged between 6 and 12 are most 'delighted' while playing the bubble pop activity. Where as, the child is least 'delighted' while using the story read and story write activities. This is one of the measures which indicate that the child is more engaged while using the bubble pop activity and is more inclined to continue using the tutor as compared to the story read and story write activities. This idea can be used to develop more personalized tutors and improve engagement levels.

VI. Limitations and Future Work

One of the limitations we faced during this research was the poor quality and low resolution of the RoboTutor video logs. This proved to be a big roadblock since the tool we were using to gather the facial action units data, OpenFace, could not work with all the videos. This lead to data loss and we had to remove few of the videos from our dataset. Since the videos were of poor quality, OpenFace detected the action units with a low confidence on several occasions. We conducted our research based on the following two assumptions -

1) There is only one child who is interacting with RoboTutor at a given point of time.
2) The child is answering all the questions based on his/her own knowledge only.

These limitations can easily be overcome in the future by improving camera quality and recording methods.

There is a lot of scope for future work in this research. We propose a framework which could detect affective states such as frustration, confusion etc in real time using computer vision techniques and could prompt the tutor to change its strategy in case of such a situation. This could help improve the experience of the student tremendously and will also lead to improved learning and better results. This framework could be implemented in the form of an API or a background service, similar to the android application that we have created, which could be work along with every tutor and communicate with the tutor so that the tutor can dynamically change its strategy based on inputs about the learner's emotional affective states.

We are already working on completing development of the Android application, as described in the methods which could work as a background service and communicate with RoboTutor and prompt it to change its content based on the feedback it gets from the background service. Another interesting application of this research would be in creating a back-end analysis tool using the such a method to analyze the students' performance and spot problems with tutors themselves, in which multiple students seem to show similar affective states.

VII. Concluding Remarks

From this research, we hope that automatic detection of affective states in tutor environments will lead to a more engaging and personalized experience for the students. To that end we presented our methods and results for analyzing the relations between the affective states that students exhibit and their in-app behavior, and also described our work in detecting affective states in students aged between 6 to 12 in the context of their interaction with RoboTutor. We showed that the behavior actions of students can be predicted based on their affective state which opens doors to the possibilities of real-time student behavior prediction for intelligent tutor systems like RoboTutor which would allow for improved timings for triggering an effective intervention policy. Future improvements to video quality and further work in development of a framework will further increase the effectiveness of real time detection of affective states and consequent prediction of behavior action in computerized learning environments.

REFERENCES

[1] Bethany McDaniel Sidney DMello Patrick Chipman Barry Gholson Art Graesser, Amy Witherspoon. Detection of emotions during learning with autotutor. 2006.

[2] Aaron Holroyd Candace L. Sidner Charles Rich, Brett Ponsler. Recognizing engagement in human-robot interaction. 2010.

[3] Ivon Arroyo Beverly Park Woolf Winslow Burleson David G. Cooper, Kasia Muldner. Ranking feature sets for emotion models used in classroom based intelligent tutoring systems. proceedings of the 18th international conference on user modeling, adaptation, and personalization, 135146. 2010.

[4] Craig S.D. Graesser A.C. D'Mello, S.K. Multimethod assessment of affective experience and expression during deep learning. international journal of learning technology. 2009.

[5] Graesser A.C. D'Mello, S.K. Multimodal semiautomated affect detection from conversational cues, gross body language, and facial features. user modeling and user-adapted interaction. 2010.

[6] Lehman B. Pekrun R. Graesser A.C. D'Mello, S.K. Confusion can be beneficial for learning. learning and instruction. 2011.

[7] Y.C. Lin A. Foster J. R. Movellan J. Whitehill, Z. Serpell. The faces of engagement: Automatic recognition of student engagement from facial expressions. Jan. 2014.

[8] DMello S. King B. Chipman P. Tapp K. Graesser A. McDaniel, B. Facial features for affective state detection in learning environments. 2007.

[9] Bixler R. Wang X. DMello S. Mills, C. Automatic gaze-based detection of mind wandering during narrative film comprehension. 2013.

[10] T. Kanade J. Saragih Z. Ambadar P. Lucey, J.F. Cohn and I. Matthews. 'the extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression' in the proceedings of ieee workshop on cvpr for human communicative behavior analysis, san francisco, usa. 2010.

[11] Sujith M. Gowda Jessica Kalka Angela Z. Wagner Aatish Salvi Vincent Aleven Gail W. Kusbit Jaclyn Ocumpaugh Lisa Rossi Ryan SJD Baker, Sujith M. Gowda. Towards sensor-free affect detection in cognitive tutor algebra. proceedings of the 5th international conference on educational data mining, 126133. 2012.

[12] C. Wylie S. L. Christenson, A. L. Reschly. Handbook of research in student engagement. 2012.

[13] Louis-Philippe Morency Tadas Baltrusaitis, Peter Robinson. Openface: an open source facial behavior analysis toolkit. 2016.

[14] M. T. Valderas, J. Bolea, P. Laguna, M. Vallverd, and R. Bailn. Human emotion recognition using heart rate variability analysis with spectral bands based on respiration. In *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 6134–6137, Aug 2015.

[15] Ryo Ishii Yukiko I. Nakano. Estimating user's engagement from eye-gaze behaviors in human-agent conversations. 2010.

Focusing Footstep Planning for Humanoids Using Homotopy-Class Guidance

Vinitha Ranganeni, Oren Salzman, Maxim Likhachev
The Robotics Institute, Carnegie Mellon University
{vrangane, osalzman}@andrew.cmu.edu, maxim@cs.cmu.edu

***Abstract*— Planning the motion for humanoid robots is a computationally-complex task due to the high dimensionality of the system. Thus, a common approach is to first plan in the low-dimensional space induced by the robot's feet—a task referred to as *footstep planning*. This low-dimensional plan is then used to guide the full motion of the robot. One approach that has proven successful in footstep planning is using search-based planners such as A* and its many variants. To do so, these search-based planners have to be endowed with effective heuristics to efficiently guide them through the search space. However, designing effective heuristics is a time-consuming task that requires the user to have exceptional domain knowledge. Thus, our goal is to be able to effectively plan the footstep motions taken by a humanoid robot while obviating the burden on the user to carefully design local-minima free heuristics. To this end, we propose to use user-defined homotopy classes in the workspace that are intuitive to define. These homotopy classes are used to automatically generate heuristic functions that efficiently guide the search algorithm. We conceptually explain our algorithm to dynamically generate homotopic-based heuristics and how it is integrated into our footstep planner.**

I. Introduction

Humanoid robots have superior navigational abilities, in human-structured environments than wheeled mobile robots [1]. However, planning for humanoid robots using standard motion-planning algorithms is a computationally-complex task due to the high dimensionality of the system. Thus, a common approach to efficiently compute paths is to guide the high-dimensional search using low-dimensional footstep motions [2].

One approach that has been successful in footstep planning is using search-based planners such as A* [3], [4] and its anytime variants [1], [5]. These planners still require effective heuristics to perform well in low-dimensional spaces. There are two inherent challenges in designing good heuristic functions. First, if the heuristic function were to have a local minima, the search-based planner would spend a large amount of time expanding in that depression region. Second, we want the heuristics to guide the search algorithm away from regions that are difficult to plan in. These two challenges make constructing local-minima free heuristics, that can intelligently reason about areas of the environment to avoid, a time consuming and tedious task. Designing such heuristics also requires the user to have excellent domain knowledge. Multi-Heuristic A* (MHA*) [6] is a recently-proposed method that attempts to leverage information from multiple heuristics to bypass local minima but still requires a user to construct heuristics that are independently useful in various parts of the state-space.

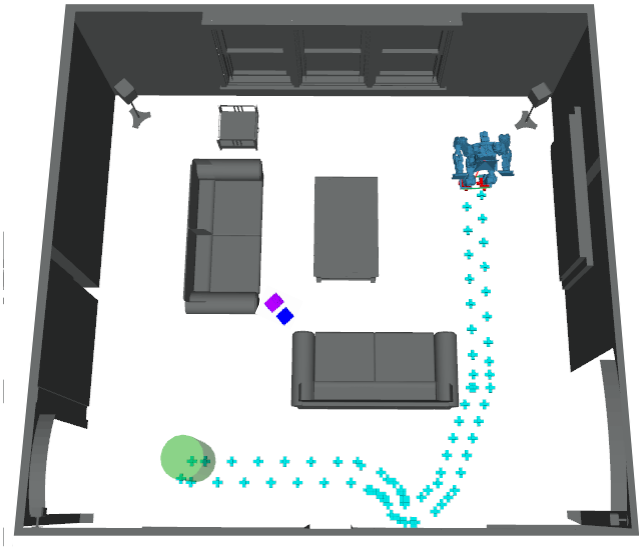


Fig. 1: We want to plan the footstep motions of the humanoid robot from its start position, right corner of the environment, to the goal region, represented by the green cylinder, while avoiding the obstacles. When we use the 2D Dijkstra baseline heuristic to guide the planner, the search gets trapped in a local minima between the couches. The purple and blue footsteps are in the region where the planner continually expanded till it timed out. When the planner is given one homotopic-based heuristic that guides the search around the obstacles, it finds a feasible path in 4.214s shown by the cyan markers.

In this work we effectively plan footstep motions for a humanoid robot while eliminating the need for the user to manually fabricate heuristics. Our key insight is that we can use user-defined homotopy classes in the workspace to dynamically generate heuristic functions that efficiently guide the search algorithm in the configuration space. By using the homotopic-based heuristic functions with the MHA* [6] framework for footstep planning, we can quickly find feasible paths while maintaining all the theoretical guarantees of MHA* [6].

The remainder of this paper is structured in the following way. In Sec.II we discuss previous contributions for footstep planning for humanoid robots using search-based planning techniques. In Sec.III we define homotopy classes on a 2-dimensional plane and explain the MHA* algorithm. Finally, in Sec.IV we outline our algorithm for dynamically generating homotopic-based heuristics and how it is integrated into our footstep planner.

II. Related Work

In this section we describe related work on using search-based planning algorithms for footstep planning and dynamic heuristic generation for these algorithms. In Sec. II-A we describe commonly used search-based planning algorithms and a few of its anytime variants. We then explain, in Sec. II-B, how these algorithms have been used in footstep planning, previous work on dynamically generating heuristics and how homotopy classes have been used in the context of motion planning.

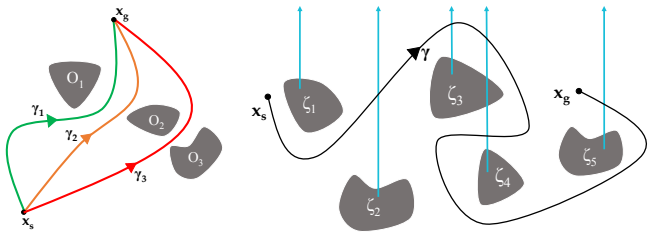
A. Search-Based Planning

Search-based planning is a motion planning method which uses graph search methods over a discretized representation of the environment. Most of these algorithms use a heuristic function that has special domain knowledge to steer the search towards the goal. A* [7] is a commonly used search-based planning algorithm that chooses the path that minimizes $f(n) = g(n) + h(n)$ where n is the current node on the path, $g(n)$ is the cost of the path from the start node to n, and $h(n)$ is a heuristic that estimates the cost of the path from n to the goal. However, the search can be guided toward a local minima if the heuristic function is poorly constructed.

In order to reduce the dependency on the heuristic functions, anytime variants of A* have been introduced. Unlike A* [7] which executes a single search to the goal, R* [8] is a randomized version of A* that tries to execute a series of randomly chosen short-range Weighted A* (WA*) searches [9], where $h_w(n) = \epsilon h(n)$, $\epsilon > 1$, towards to goal state and reconstruct the solution from the paths generated by these individual searches. ARA* [10] runs a series of WA* searches but reuses search efforts from previous executions. D* Lite [11] repeatedly determines the shortest paths between the current vertex and the goal vertex as the edge costs of the graph change while the robot moves.

B. Footstep Planning Using Search-Based Planning

Several approaches for footstep planning have been proposed over the last few years. Chestnutt *et al.* were the first to propose using A* to plan around and over planar obstacles [3], [4]. However, the heuristic functions used in this approach overestimate the cost in some situations; they sacrificed optimality guarantees for execution speeds. It is very difficult to hand craft good heuristics that can intelligently reason about the environment while maintaining guarantees on optimality. In order to compensate for the flaws in trivial heuristics, dynamic and anytime variations of A* were proposed for footstep planning. Garimort *et al.* used D* lite with various euclidean distance heuristics to dynamically replan footsteps when necessary [1]. Hornung *et al.* used an anytime search-based planners, ARA* and R*, with simple euclidean and 2D Dijkstra heuristics [5]. All these approaches have one thing in common: they all use simple heuristics. The dynamic and anytime variants of A* improve the path while searching and do not as heavily depend on the heuristic function as A* does, therefore, using simple heuristics does not impede the performance of the algorithm.



(a) (b)

Fig. 2: (a) γ_1 and γ_2 are in the same homotopy class, however, γ_3 is in a different homotopy class because of obstacle O_2. (b) The signature for this curve is $t_2 t_3 t_4 \bar{t_4} \bar{t_5}$. The homotopy invariant of curve γ is $t_2 t_3 \bar{t_5}$.

We want to be able to dynamically generate heuristics to guide the search to their fullest potential. In the past few years, there have been several approaches to learning heuristic functions [12]–[15]. The downside to this approach, as with all learning approaches, is that it requires a large amount of training data and a good feature set for training. These few learned heuristic functions are also limited to a specific set of environments.

Yi *et al.* proposed a Homotopy-Aware RRT* algorithm [16] which restricts the RRT* search to homotopic path constraints that were translated from human intent. Homotopy classes have also been frequently used to model the motion of a robot tethered to a fixed base point [17]–[19]. We, instead, propose to collect desirable homotopy classes from a user to generate our heuristic functions. To the extent of our knowledge, we are the first to use homotopy classes to dynamically generate heuristic functions.

III. Algorithmic Background

In this section we describe the relevant algorithmic background necessary to understand our approach. We start in Sec. III-A by formally defining the notion of homotopy classes and how to efficiently identify if two curves are in the same homotopy class—a procedure that will be used in our homotopy-based shortest-path algorithm (HBSP). We then describe Dijkstra's algorithm in Sec. III-B—an algorithm that is the basis for HBSP. Finally, in Sec. III-C we provide background on MHA* which we will use to compute footstep plans by incorporating the heuristic computed using HBSP.

A. Homotopy classes of curves

Informally, two continuous functions are called *homotopic* if one can be "continuously deformed" into the other (see Fig.2a). In general, uniquely identifying the homotopy class of a curve is non-trivial, however if both curves are embedded in the plane, a straightforward characteristic exists to identifying and compute the homotopy class of a curve [20].

Specifically, let $\mathcal{W} \subset \mathbb{R}^2$ be a subset of the plane (in our work this will be a two-dimensional projection of the three-dimensional workspace where the robot moves) and let $\mathcal{O} = \{\mathcal{O}_1, ..., \mathcal{O}_m\}$ be a set of obstacles (in our work, these will be projections of the three-dimensional workspace obstacles).

In order to identify if two curves $\gamma_1, \gamma_2 \in \mathcal{W} \setminus \mathcal{O}$ that share the same endpoints are homotopic we use the notion of *h-signature* (see [17]–[19]). The h-signature uniquely identifies the homotopy class of a curve. That is, γ_1 and γ_2 have identical reduced words if and only if they are homotopic.

In order to define the h-signature, we choose a point $p_k \in \mathcal{O}_k$ in each obstacle such that no two points share the same x-coordinate. We then extend a vertical ray or "beam" b_k towards $y = +\infty$ from p_k. Finally, we associate a letter t_k with beam b_k.

Now, given γ, let $b_{k_1}, \ldots, b_{k_m}$ be the sequence of m beams crossed when tracing γ from start to end. The *signature* of γ, denoted by $s(\gamma)$, is a sequence of m letters. If γ is intersected by the beam b_k, by crossing it from left to right (right to left), then the i'th letter is t_k ($\bar{t_k}$, respectively). The reduced word, denoted by $r(s(\gamma))$, is constructed by eliminating a pair of consecutive letters in the form of $t_k\bar{t_k}$ or $\bar{t_k}t_k$. The reduced word $r(s(\gamma))$ is a *homotopy invariant* for curves with fixed endpoints. It will be denoted as $h(\gamma) = r(s(\gamma))$ and called the *h-signature* of γ (See Fig.2b).

As we will see, our search algorithm HBSP will incrementally construct paths. After they are fully constructed, they should be in the same homotopy class as a given reference path. Thus it will be useful to understand how the h-signature of a curve γ which is a concatenation of two curves γ_1, γ_2 can be easily constructed. This reduced signature of $\gamma = \gamma_1 \cdot \gamma_2$ is simply the reduced signature of the concatenation of two curves' signatures $h(\gamma) = r(s(\gamma_1) \cdot s(\gamma_2))$.

B. Single-Source Shortest Path—Dijkstra)

Dijkstra's algorithm computes the costs of the shortest paths from the source node x_s to all other nodes in the graph $\mathcal{G}$. The algorithm starts at x_s and iteratively updates the shortest paths and costs via the current node. The node with the lowest cost is popped from the priority queue and all of its successors are explored. If the successor v has not been explored, it is added to the priority queue with its respective cost. Otherwise, the cost of the node is updated if the currently calculated cost is less than the assigned cost. This process repeats until all the nodes are expanded (i.e. the priority queue is empty).

Algorithm 1 Dijkstra's Algorithm

```
1:  function Dijkstra(G, x_s)
2:      create priority queue Q
3:      dist[x_g] ← 0
4:      Q.add_with_priority(x_s, dist[x_s])
5:      while Q ≠ ∅ do
6:          u ← Q.extract_min()
7:          V ← succ(u)
8:          for each v ∈ V do
9:              alt ← dist[u] + length(u, v)
10:             if dist[v] = ∅ then
11:                 dist[v] ← alt
12:                 Q.add_with_priority(v, dist[v])
13:             else if alt < dist[v] then
14:                 dist[v] ← alt
15:                 Q.decrease_priority(v, dist[v])
16:     return
```

C. Multi-Heuristic A* (MHA*)

The performance of heuristic search-based planners, such as A*, depend heavily on the quality of the heuristic function. For many domains, it is difficult to produce a single heuristic function that captures all the complexities of the environment. Furthermore, it is difficult to produce an admissible heuristic (namely, one that provides lower bounds on the cost-to-goal), which is a necessary condition for providing completeness guarantees.

One approach to cope with these challenges is by using *multiple* heuristic functions. MHA* [6], [21] is one such approach that takes in a single admissible heuristic called the *anchor* heuristic, as well as multiple (possibly) inadmissible heuristics. It then simultaneously runs multiple A*-like searches, one associated with each heuristic, which allows to automatically combine the guiding powers of the different heuristics in different stages of the search.

Aine *et al.* [6], [21] describe two variants of MHA*, Independent Multi-Heuristic A* (IMHA*) and Shared Multi-Heuristic A* (SMHA*). In IMHA* each individual search runs independently on the other searches while in SMHA*, the best path to reach each state in the search space is shared among all searches. This allows each individual search to benefit from progress made by other searches. This also allows SMHA* to combine partial paths found by different searches which, in practice, makes SMHA* more powerful than IMHA*. Therefore in this work we will use SMHA*. and for brevity will refer to SMHA* as MHA*.

IV. Homotopy-Based Footstep Planning

Our footstep planner is comprised of the Homotopic-Based Shortest Path Algorithm (HBSP), which generates the heuristic functions from a set of user-defined homotopy classes, and MHA*, which uses the heuristics to find a feasible path. We first convey the intuition and performance of our footstep planner on the simple example shown in Fig.1.

In Fig.1 there is humanoid that has to navigate from its current position to the goal region denoted by the green

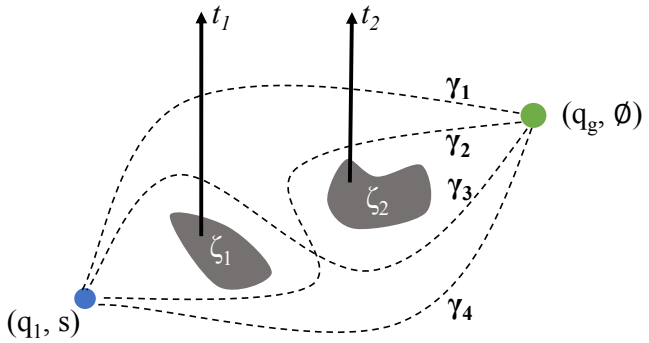


Fig. 3: There are four possible paths from vertex $(q_g, \emptyset)$ to (q_1, s) where s can be any *h-signature* of the given paths $\gamma_1, ..., \gamma_4$. When we construct our augmented graph $\mathcal{G}$ we add the following vertices: $(q_1, r_2 r_1), (q_1, r_2), (q_1, r_1), (q_1, \emptyset)$. These vertices correspond to the signatures of $\gamma_1, ..., \gamma_4$ respectively.

cylinder. The baseline heuristic is a 2D Dijkstra search starting from the robot's current position. This heuristic tries to guide the search through the narrow passage between the couches, as it is the shortest path to the goal region, however, it gets trapped in a local minima.

HBSP takes guidance from the user to determine which homotopy classes the heuristic functions should guide the search through. The provided homotopic classes guide the search around the obstacles instead of going through the narrow passage. The path the footstep planner produces with this homotopic-based heuristic is not the shortest path, however, it is a feasible path to the goal that was quickly produced using human intuition.

A. Homotopic-Based Shortest Path

Our algorithm, HBSP, is a modified version of Dijkstra's algorithm that allows us to find the shortest paths for a set of homotopy classes. This algorithm uses a homotopy-augmented graph $\mathcal{G} = (\mathcal{X}, \mathcal{E})$. In $\mathcal{G}$, every vertex $x = (q, \tilde{s})$, $x \in X$, where q is the configuration of the robot, $\tilde{s}$ is the current signature and every edge $e \in E$ represents the euclidean distance between the two vertices. Our augmented graph can have vertices that have the same configuration q but different signatures $\tilde{s}$, as shown in Fig.3.

First, we create a goal set $V_G = \{(q_s, \bar{s}) | s \in S\}$ where q_s is the start vertex. Then we run a single backward Dijkstra's search from the goal vertex $(q_g, \emptyset)$ till we reach every vertex in V_G. During the search we incrementally add edges and prune away those that do not belong in any of the homotopy classes of the reference paths. We do this by concatenating the signatures of the path from $(q_g, \emptyset)$ to the current vertex u and the path from u to the successor v. We remove the vertex v from the successor list if $\tilde{s_u} \cdot \tilde{s_v}$ is not a *suffix* of the *h-signatures* of the desired homotopy classes (Lines 6-7). The *suffix* set $\mathbb{S}$ defines the set signatures for vertices that belong in the desired homotopy classes (Line 4). For example, $\mathbb{S}$ of γ from Fig.2b is $\{t_2 t_3 \bar{t_5}, t_2 t_3, t_2, \emptyset\}$. If $\tilde{s_u} \cdot \tilde{s_v} \in \mathbb{S}$ then

Algorithm 2 HBSP Successor Function

```
1: function succ(u, S)                          ▷ u = (q_u, s̃_u)
2:     V ← neighbors(u)                         ▷ v = (q_v, s̃_v)
3:     if S ≠ ∅ then
4:         𝕊 ← suffixes(S)
5:         for each v ∈ V do
6:             if s̃_u · s̃_v ∉ 𝕊 then
7:                 V.remove(v)
8:             else
9:                 v = (q_v, s̃_u · s̃_v)
10:    return V
```

we set the signature for vertex v to $\tilde{s_u} \cdot \tilde{s_v}$, the *h-signature* of the path taken to reach that vertex (Line 9).

B. Footstep Planning using HBSP

Our footstep planner performs a search from the start vertex $(q_s, \emptyset)$ to every vertex in $\{(q_g, s) | s \in S\}$. HBSP computes the heuristic function for each of the homotopy classes from the user-defined reference paths. Alg.3 is our homotopic-based heuristic function $h_s(x)$ where $s \in S$ and $x \in \mathcal{X}$. $h_s(x)$ returns the under-estimated path cost from vertex x to the goal vertex (q_g, s) (Line 9). For example, let $\tilde{s} = t_1$, $s = t_1 t_2$ and $\bar{s} = \bar{t_2}\bar{t_1}$, then $h(x) = c(q, \bar{s} \cdot \tilde{s}) = c(q, \bar{t_2}\bar{t_1} t_1) = c(x, \bar{t_2})$. Here, our desired *h-signature* is $s = t_1 t_2$ and the *h-signature* of our current vertex is $\tilde{s} = t_1$. The *h-signature* of $\bar{s} \cdot \tilde{s}$ is the signature of the remaining path to the goal. Therefore, $c(q, \bar{s} \cdot \tilde{s})$ is the cost of the remaining path.

If x has not been explored (i.e. no cost was computed for that state-signature pair) we continue HBSP (Line 8). By keeping track of the state of the Q and passing it back to HBSP along with the $\mathcal{G}$, x and S we can call HBSP on demand. This allows us to only expand vertices which are necessary for the footstep planner to find the shortest feasible path within in given set of homotopy classes without over-expanding vertices. When HBSP reaches the vertex $(q, \bar{s} \cdot \tilde{s})$ the search is terminated and the newly calculated cost for that vertex is returned (Line 9). Fig.4 gives an overview of how HBSP is integrated with MHA* for footstep planning.

Algorithm 3 Homotopic-Based Heuristic Function

```
1: function plan(S, q_s, q_g, 𝒢)
2:     Q ← HBSP(∅, 𝒢, q_g, q_s, S)
3:     Run SMHA* with heuristics {h_s | s ∈ S}
4:
5: function h_s(x)                              ▷ x = (q, s̃)
6:     if c(q, s̄s̃) = ∅ then
7:         S = {s̄ · s̃}
8:         Q ← HBSP(Q, 𝒢, q, ∅, S)
9:     return c(q, s̄s̃)
```

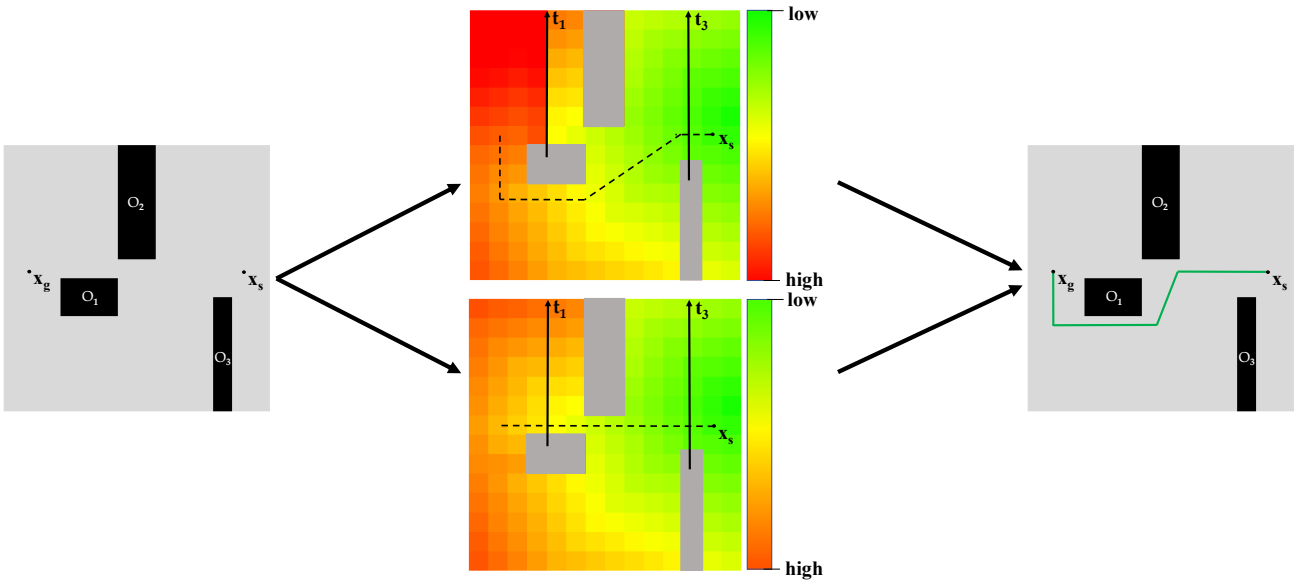


Fig. 4: (*left*) We start with a simple 2D environment with obstacles O_1, O_2, O_3, start position x_s and goal position x_g. (*center: HBSP*) We provide the footstep planner two homotopy classes whose signatures are $\bar{t_3}$ (*top*) and $\bar{t_3}\bar{t_1}$ (*bottom*). We display a color-map of heuristic values from low (green) to high (red). (*right: MHA**) It is not feasible for the robot to go through the narrow passage so the planner plans around the obstacles.

References

[1] J. Garimort, A. Hornung, and M. Bennewitz, "Humanoid navigation with dynamic footstep plans," in *IEEE International Conference on Robotics and Automation*, 2011, pp. 3982–3987.

[2] K. Gochev, B. J. Cohen, J. Butzke, A. Safonova, and M. Likhachev, "Path planning with adaptive dimensionality," in *SOCS Symposium on Combinatorial Search*, 2011.

[3] J. E. Chestnutt, M. Lau, G. K. M. Cheung, J. Kuffner, J. K. Hodgins, and T. Kanade, "Footstep planning for the honda ASIMO humanoid," in *IEEE International Conference on Robotics and Automation*, 2005, pp. 629–634.

[4] J. E. Chestnutt, K. Nishiwaki, J. Kuffner, and S. Kagami, "An adaptive action model for legged navigation planning," in *IEEE-RAS International Conference on Humanoid Robots*, 2007, pp. 196–202.

[5] A. Hornung, A. Dornbush, M. Likhachev, and M. Bennewitz, "Anytime search-based footstep planning with suboptimality bounds," in *IEEE-RAS International Conference on Humanoid Robots*, 2012, pp. 674–679.

[6] S. Aine, S. Swaminathan, V. Narayanan, V. Hwang, and M. Likhachev, "Multi-heuristic A," in *Robotics: Science and Systems*, 2014.

[7] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[8] M. Likhachev and A. Stentz, "R* search," in *AAAI Conference on Artificial Intelligence*, 2008, pp. 344–350.

[9] I. Pohl, "Heuristic search viewed as path finding in a graph," *Artificial Intelligence*, vol. 1, no. 3, pp. 193–204, 1970.

[10] M. Likhachev, G. J. Gordon, and S. Thrun, "Ara*: Anytime a* with provable bounds on sub-optimality," in *Conference on Neural Information Processing Systems*, 2003, pp. 767–774.

[11] S. Koenig and M. Likhachev, "D*lite," in *AAAI Conference on Artificial Intelligence*, 2002, pp. 476–483.

[12] J. ús Virseda, D. Borrajo, and V. Alcázar, "Learning heuristic functions for cost-based planning," *Planning and Learning*, p. 6, 2013.

[13] S. J. Arfaee, S. Zilles, and R. C. Holte, "Learning heuristic functions for large state spaces," *Artificial Intelligence*, vol. 175, no. 16-17, pp. 2075–2098, 2011.

[14] J. T. Thayer, A. J. Dionne, and W. Ruml, "Learning inadmissible heuristics during search," in *International Conference on Automated Planning and Scheduling*, 2011.

[15] M. Bhardwaj, S. Choudhury, and S. Scherer, "Learning heuristic search via imitation," *CoRR*, vol. abs/1707.03034, 2017.

[16] D. Yi, M. A. Goodrich, and K. D. Seppi, "Homotopy-aware rrt*: Toward human-robot topological path-planning," in *ACM/IEEE International Conference on Human-Robot Interaction*, 2016, pp. 279–286.

[17] S. Bhattacharya, M. Likhachev, and V. Kumar, "Topological constraints in search-based robot path planning," *Autonomous Robots*, vol. 33, no. 3, pp. 273–290, 2012.

[18] D. Grigoriev and A. Slissenko, "Polytime algorithm for the shortest path in a homotopy class amidst semi-algebraic obstacles in the plane," in *International symposium on Symbolic and algebraic computation*, 1998, pp. 17–24.

[19] O. Salzman and D. Halperin, "Optimal motion planning for a tethered robot: Efficient preprocessing for fast shortest paths queries," in *IEEE International Conference on Robotics and Automation*, 2015, pp. 4161–4166.

[20] M. A. Armstrong, *Basic topology*. Springer Science & Business Media, 2013.

[21] S. Aine, S. Swaminathan, V. Narayanan, V. Hwang, and M. Likhachev, "Multi-heuristic A*," *The International Journal of Robotics Research*, vol. 35, no. 1-3, pp. 224–243, 2016.

Pose Estimation for Soft Robots Using RFID Sensing

David Russell
Department of Computer Science
Clarkson University
russeldj@clarkson.edu

James Bern, Stelian Coros
The Robotics Institute, Department of Computer Science
Carnegie Mellon University

***Abstract*— Soft robots are easily deformable and can therefore conform to the world around them. This fact makes them particularly well-suited to perform tasks in unstructured environments, such as grasping amorphous objects or traversing uncertain terrain. As with many robotic applications, feedback about the pose of the robot can dramatically help in these situations. However, because soft robots can have an essentially infinite number of degrees of freedom, this information is very challenging to obtain. Vision-based methods are limited by robot self-occlusion and piezoresistive approaches are dependent on a high quality model of the physical properties of the robot. We implement a generic and easily-adaptable system for estimating the pose of plush robots using data gathered from radio frequency identification (RFID) tags. Our method relies on obtaining the angle between linear arrays of tags and the direction of an antenna. This measurement is derived by using the multiple signal classification (MUSIC) algorithm using the phase of the radio signals from each tag. We map the linear sets of tags onto a finite element mesh that represents the robot. Then we deform this mesh by imposing an objective that rotates these sets to be consistent with the sensed angle. This simulation can be used to to inform control strategies. We then validate this approach using spoofed data and with preliminary real-world results.**

I. Introduction

Soft robots are an emerging class of robotic systems that are designed to experience large deformations during their normal use. This compliance can be leveraged to perform complex tasks in unstructured environments. Notable among these are grasping delicate objects of unknown orientation and locomotion on irregular terrain [1], [2]. Additionally, soft robots can interact with non-expert users much more safely than traditional robots, which means they show promise in assistive robotics and other related fields.

In past work, we implement a novel class of soft ”plush” robots that are tendon-actuated and are constructed in a fashion similar to many soft toys [3]. We seek to make soft robot design, fabrication, and control more accessible to non-experts so they may perform a diverse set of tasks in these arenas. Currently, there is a lack of high-level design tools for soft robots; to create physically-realizable robots, one must have a large body of knowledge and prior low-level CAD experience. Additionally, the fabrication techniques usually involve casting polymers, which is a skill that few non-experts have and is also poorly suited to one-off creations.

Our design tools only require high-level user input and the results can be fabricated with simple tools and accessible materials. This generalized framework is currently limited by the lack of sensing strategies that are applicable across a variety of robot morphologies. To direct physically-realizable motion strategies, we must inform our control simulation with real-world state data. Unfortunately, there is no simple approach for this when working with soft robots. In contrast to traditional robots, it is not possible to simply use encoders at every joint, because the robot is expressly designed to be able to deform at points all throughout its body, leading to an incredibly high number of degrees of freedom.

While a large body of work explores applications of marker-based vision tracking for both hard and some soft robots [4], we conclude that this approach is unsuited to our work. A major advantage of our plush robots is that they can be made in a wide variety of shapes. Many of these shapes we envision are likely to experience self-occlusion, when one part of the robot blocks the line of sight from another part to the camera.

A common alternative to vision-based tracking is embedding piezoresistive elements in the soft robot [5]. These wire-like structures can sense the aggregate bend along their length. However, in complexly-shaped robots or ones for which only a coarse model exists, it is challenging to determine good placement for these cables and fabricate the resultant designs [5].

We propose a pose estimation technique based on small and low-cost passive radio frequency identification (RFID) tags. Because many objects—notably many soft robots—are transparent to frequencies in the radio spectrum, RFID tags are a robust sensing modality. Additionally, they can be hidden from view and are easy to incorporate into soft robots. We use an approach based on the phase of the received signal and the well-studied multiple signal classification (MUSIC) algorithm [6], [7]. MUSIC allows us to compute the angle between the axis of a linear patch of tags and the incidence vector from the reader. The value for each angle is constantly updated as new data is acquired.

To utilize this information, we construct a 2D finite element mesh (FEM) that represents the real-world robot. This means that the outline of the robot is filled with a collection of triangles, which intersect at ”nodes.” We then formulate two objectives; one which seeks to minimize the deformation of each triangle and another which drives the the simulated angles toward the sensed ones, by deforming the mesh. These are both posed as energy functions, where undesirable states result in higher energy values. We then perform an energy minimization optimization using Newton’s method on the set of nodal positions. This results in the estimated pose of the robot.

II. Background

Soft robotic control could benefit from having an accurate estimate of the robot's current configuration. It is challenging to perform pose estimation on soft robots for exactly the same reason they can deal well with unstructured environments; they are compliant and have a large number of deformation modes [8]. Many traditional, hard robots have linear or rotational encoders at every major joint [9]. From the joint angles measured by these encoders a model of the robot can be easily reconstructed. Most traditional robotic fabrication materials, such as metal or hard plastic, have a Young's moduli of around $10^9 - 10^{12}$ Pa [1], whereas those used in soft robotics have moduli of $10^5 - 10^9$ Pa [1]. This means that soft robots will deform orders of magnitude more under the same load. In fact, the primary actuation modes of soft robots often involve bending large portions of the robot, rather than rotating at a few pre-defined points. Taken together, these considerations mean that more robust and general pose estimation techniques must be developed.

A common paradigm in estimating the pose of soft bodies is distributed deformation sensing. Systems such as FlexSense [10] utilize piezoresistive sensing to this end. This could in principal be combined with other systems, such as Defsense [5], which presents a computational framework for determining where to place the piezoresistive elemements. These sensors aggregate the bending over wire-like structures as a resistive value. This approach requires high resolution models of the common deformation modes and high-fidelity fabrications. As mentioned by Bächer et al. [5], small changes in the routing of piezoresistive wires can have large impact on the resultant resistivity. In practice, this limits their use to geometrically simple robots or those that are made of highly homogeneous materials, which can be modeled with high fidelity [8]. Additionally, the data received is rather low dimensionality in that it maps all of the deformations across one wire to a single value. Mapping the measured bending of piezoresisitvie elements to deformations of the overall soft body becomes especially challenging for complex robots and may necessitate numerous wires. Moreover, the information is solely local, so in large systems with high degrees of freedom, it is easy to globally propagate errors. This is because information about the relative position of two features can only be derived from the relative position of every intermediate part.

Another common approach to estimating the pose of objects is computer vision [11], [12]. However, this is most successful when trying to localize an object which is only subject to rigid body rotations and translation. If one knows the shape of the object a priori, then it is easy to extrapolate the location of regions that are not seen, with adequate knowledge of the other surfaces. When dealing with any object which is not rigid, one must be able to collect information about the vast majority of points on the surface or begin to use heuristics to predict where these unseen regions are. In a system with complex geometries, it is challenging to ensure that all faces are seen or to develop quality algorithms for extrapolating the position of unseen features. Since our system would likely be used in unstructured environments, with robots of arbitrary geometries, we see occlusion as a frequent and significant problem.

Our current work is most closely related to that of Yamashita et. al [4], where they implement a stuffed animal that moves and responds to humans. A Kinect RGBD camera is pointed at the area in which the user is going to be, and the robot can do such things as follow them with it's gaze. However, it is important to note that it is the human, not the robot that is tracked, meaning that the approach is highly targeted to this one use case. The robot from [4] does have some proprioceptive sensing capabilities in the form of force sensors on each of its tendons. In the example they present, the robot can sense a person pushing on its limb. However, in the demonstration, they show that their particular architecture is only suited to sensing small deformations. It also relies on a high number of tendons to form a model of the direction of the force. This is in contrast to the current paradigm of highly underacted systems. Additionally, it appears that in would be challenging to implement their method on systems with higher degrees of freedom.

We see the lack of generality of the current soft robotic pose estimation approaches as a impediment to rapid iteration both in the research and non-expert communities. We do not seek to replace highly targeted solutions for projects which have the resources to implement them effectively, but rather propose a flexible sensing framework which is agnostic to robot form and function. While our method is validated on plush robots, we see it as valuable for other macro-scale soft robots which may be challenging to model with high fidelity.

III. Methods

We present a method for doing pose estimation of plush robots using RFID tags. The phase data received from linear-arrays of real-world tags is fed into the MUSIC algorithm to estimate the angle of arrival of local patches of the robot. These angle of arrival estimates are incorporated as additional soft objectives into a finite element simulation of the robot. The energy-minimal configuration of the simulated robot is found via optimization, and is our output estimate of the real-world robot's pose.

A. Plush Robots

In past work [3], we propose a class of plush soft robots which are fabricated in a manner similar to that of traditional stuffed animals. These soft robots consist of a rugged acrylic felt skin and are stuffed with Polyfill batting. To fabricate them, one only needs a sewing machine and either a laser cutter or a paper template and scissors. They are actuated by tendons which are contracted by self-contained motor assemblies. We see robots of this class as well-suited to a variety of tasks in the broader soft robotic domain.

B. RFID Hardware

To estimate the pose of the robot, we distribute passive RFID tags across the surface of the robot. An RFID tag

consists of an antenna and a small integrated circuit, encapsulated in a flexible, planar covering. The passive nature of these tags means that all the energy they need can be extracted from radio waves or a certain frequency range. When exposed to such a signal, the integrated circuit collects power from the antenna, and broadcasts a signal in return, which is at the same frequency as the power signal, but is modulated to encode information. To interface with tags, one uses a reader equipped with antennas that transmit the initial signal and receive the modulated one. The data includes which tags have been read and the respective phase angle, power, and frequency of the received signal.

For these experiments, we utilize two different sorts of tags. The first is the Alien Square RFID White Wet Inlay. These tags are 2.5cm square and are less than 1mm thick. As such, they are as deformable as paper. These tags have strong read fidelity and their form-factor makes them easy to incorporate into soft robots. We also use Confidex Silverline Blade RFID tags because they are especially robust around metal objects. They are 2.5cm by 6cm and are 2mm thick. Since the are made out of foam, they are quite easy to deform without damaging them. To obtain data from the tags, we use a RFMAX S9028PCR Indoor RFID Antenna powered by a Impinj R420 Revolution RFID reader. The data is transmitted over Ethernet to any computer or an Ethernet switch.

C. MUSIC Algorithm

One of the most challenging parts of using radio signals is the fact that they are reflected off of many surfaces such as walls and floors. This confounds many approaches that utilize only the received signal strength or phase of individual tags. This is because all the information is aggregated between the different signal propagation paths. For example, a tag with a wall behind it will have a stronger received signal strength than one without. This is because some of the signal which wouldn't have otherwise been read, is reflected toward the reader. The phase of the signal now also becomes a composition of the phases of the two transmission paths. Fortunately, if one has a linear array of tags, the well-studied MUSIC algorithm can be used to disambiguate the direct signal transmission path from the reflected ones [7], [6]. This is done by looking at the difference in phase between neighboring tags. This difference arises because the length of the propagation path varies for each tag. All of this is done using the phase of the received signal from each of the tags. The algorithm return an angle from the oriented axis of the array to the incident vector from the reader, as seen in Fig. 1. It is important to note that this measurement is solely related to the coordinates of the array and not theoretically relative to the reader orientation. These measured angles form the fundamental priors of our system. We utilize the MUSIC implementation from the MATLAB 2016b Phased Array Toolbox to calculate these angles.

D. Finite Element Discretization

An integral part of this pose estimation approach is a finite element mesh (FEM) discretization of the robot. We model the boundary of the robot and then fill the space inside it with triangles. The coordinates of each triangular point, or "node," are optimized to minimize a deformation energy function.

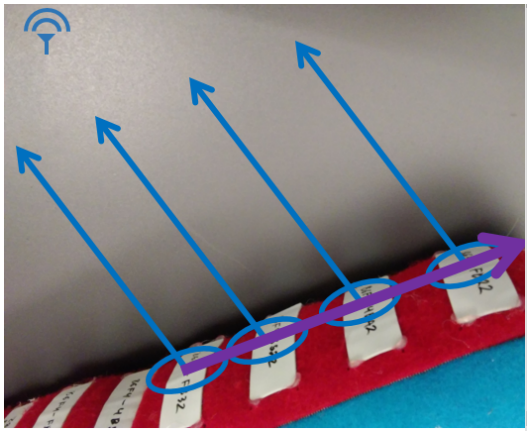


Fig. 1. An example of a linear array of RFID tags on a plush robot, with the incident vectors in blue and the oriented array in purple. The output from the MUSIC algorithm is the angle between these two vectors.

This process is done using Newton's method with the nodal positions as input. The generality of Newton's method means that it is easy to add additional energy penalties, as long as they are a function of the nodal positions and are twice-differentiable. To this end, we create an objective that is related to the state of the mesh relative to the sensed angles.

E. Angular Objective

We introduce the concept of oriented arrays, or collections of tags which, in simulation, are roughly linear. Additionally, we add a simulated reader which is currently constrained in world space. From each simulated array, we can determine the real-world angle between its oriented direction and the incident vector from the simulated reader.

We formulate our objective to minimize the deviation between the sensed angle and the simulated one. The angular energy function is then:

$$E_{angular}(\vec{x}) = \sum_{i=0}^{n} \alpha e^{-kt}(\theta_{sensed} - \theta_{simulated}(\vec{x}))^2$$

For each array on the mesh, the energy value is related both to how much that array deviates form the sensed value and how recently the sensed value has been updated. This time period is represented by the variable t while the initial weight of the objective and how quickly it is reduced are determined by the constants α and k, respectively. The variable that changes during each iteration of Newton's method is the simulated angle, which is the angle between the oriented array and the vector to the simulated reader. Since the array is constrained to the FEM, changing the vector of nodal positions, $\vec{x}$, changes this angle. An example of the two inputs to the minimization can be seen in Fig. 2.

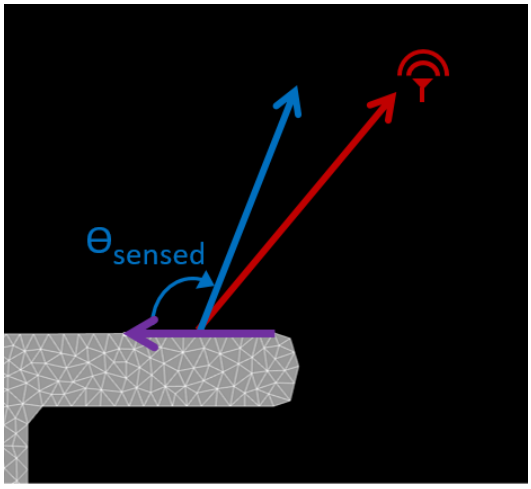


Fig. 2. We deform the finite element mesh so that the vector to the simulated antenna, in red, is as close in direction as possible to the vector projected at the sensed angle. In this case, the array seen would be forced to rotate clockwise, thereby deforming the underlying mesh and bending the arm downward.

F. Resolving MUSIC Ambiguity

Unfortunately, there is an inherent loss of information between the real world and the angle MUSIC returns. Because the algorithm is only looking at phase differences between elements in a linear array, the measurement provides no data about the rotation around the axis of the array. In 3D space, this means that for each sensed angle, there is a cone of possible reader locations. In a 2D model, there are two possibilities, mirrored across the array. In either dimensionality, there is an unambiguous edge case at $\theta = 0$ and $\theta = \pi$.

To provide a meaningful input to our optimization, it is important to remove this ambiguity. If the deformation per discrete timestep is low, it is unlikely in any given timestep, the side will switch. Given this assumption, we assign the sign of the angle, based on whichever projection is closer, after the angle is updated, but before the optimization occurs. If this method is inaccurate, it will likely be at points where the angle is near parallel to the array. Therefore, switching the sign will only have a small impact. Hopefully, the other arrays which are unambiguous will pull the system in the correct direction. In effect, we are using the FEM to interpolate between unambiguous arrays.

To address the ambiguity in the base-case, or first timestep, we impose the limitation that before the system is used, the robot be placed in a pose that resembles the simulated one. While this might appear onerous, both the real world and simulated robot will have approximately the same rest pose. Even if the user would like to start the program with the physical robot not in its rest pose, we have implemented an intuitive click-and-drag interface for posing the simulated robot. Then, the only consideration becomes orienting the rotation of the robot relative to the reader.

G. Spoofing Data Collection

To test the theoretical underpinnings of this method, we develop simulated data sets. Rather than creating a signal for each array by hand, we simulate the data collection step in our pipeline. This is done by creating ”broadcast” model with the same form-factor as the ”reconstruction” model. This model has arrays in identical locations to includes duplicated placement of all arrays. Into the broadcast simulation, we introduce a point that represents the reader.

We can deform this broadcast model in much the same way that a physical robot would get deformed. We take the angle between the oriented array vector and the incident vector from the broadcast reader to the center of the broadcast array. This quantity is the so-called spoofed data.

H. Real-world Data Collection

To collect real data, we first fabricate unactuated versions of the robots described in subsection A. and more extensively in [3]. RFID tags are then affixed to the surface of the felt with an adhesive. We deform the soft object in front of the antenna at a range of approximately 1-2 meters. The data is recorded by the system described in subsection B. A ground-truth video recording is captured in an orientation that is close to orthogonal to the deformations.

IV. Results

In this section, we validate our approach with both spoofed data and an real-world prototypes. The spoofed data shows that using angle of arrival coupled with a finite element mesh is a robust solution for reconstruction. Physical tests also demonstrate that the quality of real-world data is sufficient for fair-quality reconstruction.

A. Simulation

The reconstructions utilizing the spoofed data were quite promising. Using this approach, we were able to get high-quality reconstructions of the gripper robot, as seen in Fig. 3. Even when our data was subjected to simulated noise, the reconstruction provided a reasonable macro-scale approximation for the broadcast mesh. The reconstruction was robust, even as level of noise surpassed the levels we expect from the real world.

Experiments also indicated that the relative weights of the angular and FEM objectives was important to the quality of the reconstruction. If the weight of the FEM objective is too low, then the simulated robot can assume physically unrealizable configurations. If the weight is too high, there are two distinct problems. One is that the simulated robot will never come close to the real world, because the FEM objective will pull it toward the rest pose, where the FEM energy is minimized. Additionally, it leads to ”puckering,” where only small sections of the FEM, around where the arrays are constrained to it, are deformed. This problem is less important than the previous one and could be addressed in part by constraining the array to nodes other than the endpoints.

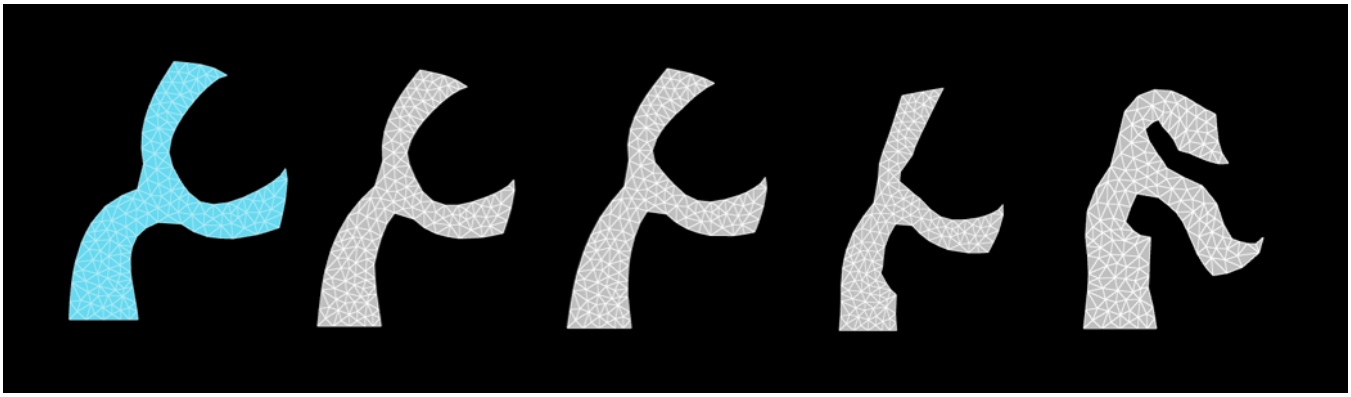


Fig. 3. This image shows a set of gray reconstruction meshes, all derived from the blue broadcast mesh. The leftmost gray pose if constructed with accurate data from the broadcast mesh. The next three have spoofed angles that were changed by a Gaussian-distributed amount. The standard deviations of this noise, from left to right, is 10°, 20°, and 30°. This shows the method is reasonably robust to inaccurate measurements.

B. Real-World

To test the validity of the real-world reconstructions, we compared a time-series of deformations to a video ground truth. The object we reconstructed was a soft bar measuring 45cm x 9cm x 9 cm. Four arrays of tags were placed on one face of the bar, with two running in parallel at each end. We rotated and deformed this bar through a representative space of motions.

During most of the trials, the simulation matched the real world fairly well. A frame from a reconstruction can be seen in Fig. 4. However, there where some times where the simulation would suddenly deviate from the real world. Looking at the data, this would be caused by large spikes in the angle of arrival computed by the MUSIC algorithm. However, these problems were usually very brief.

A second trial was done with a T-shaped object fabricated in the same way. The quality of these reconstructions were markedly worse. There seemed to especially be problems when one array would move behind another array, relative to the reader. This result was especially true with the Silverline tags. From this we can conclude that the that the Alien Square tags may be best for dense tag applications.

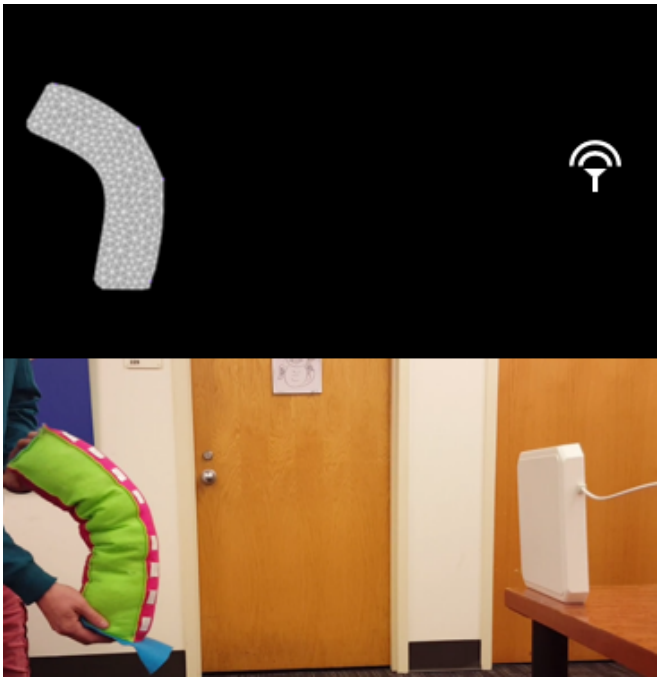


Fig. 4. This is an example of the simulation versus the real world.

V. Discussion

In this work, we summarize an approach for sensing the pose of a soft object or robot. Our main contribution is the fact that this system is scalable to a wide variety of robot morphologies, with minimal high-level changes. For this approach, we require only a reasonably accurate finite element discretization of the robot. We present preliminary results of this technique, both in simulation and when estimating the pose of an un-actuated version of our plush robots. When the reconstructed results are compared to a ground-truth, the results are promising.

However, there are still limitations with our current method. The quality of the angle of arrival data for robots with more tags is quite noisy and limits the usefulness of the resultant simulation. There are several modifications to our current approach that we believe could mitigate this problem. The first is simply a more thorough exploration of the hardware space. In this work, we only test two different RFID tags. There was a marked difference in the quality of these two tags in some cases, so it is not unreasonable to believe that there is a better tag for this application on the market.

A factor that also contributed to our poor-quality data was the fact that our reader changed radio frequency every .2 seconds to comply with FCC regulations. This reduces the amount of data that can be sent to MUSIC for each calculation, because the algorithm requires the frequency of each read to be the same. In simple experiments, we have seen that more data usually results in a more accurate output. Since it is possible to obtain readers for the Chinese market which dont have this limitation, we believe that this could be another viable exploration.

On the software side, it appears that a well-designed filtering approach could also help reduce the variance in the angle of arrival data. A Kalman filter seems to be a reasonable choice though simpler filters or a weighted rolling average could also be tried.

In the future, we hope to implement this framework in three dimensions. This would allow us to test a broader variety of soft robots, including ones made of more traditional materials. We also want to explore the possibilities of sensor fusion, with some form of vision or depth sensing.

VI. Acknowledgements

We would like to thank Haojian Jin, Dr. Jason I. Hong and Dr. Swarun Kumar for openly sharing their experience about RFID tag localization. We are also deeply grateful to Dr. Ioannis Gkioulekas for providing insights into resolving some of the challenging ambiguities inherent to this sort of sensing. Finally, we would like to acknowledge the National Science Foundation for their financial support of this work.

References

[1] D. Rus and M. T. Tolley, "Design, fabrication and control of soft robots," *Nature*, vol. 521, no. 7553, pp. 467–475, 2015.

[2] D. Trivedi, C. D. Rahn, W. M. Kier, and I. D. Walker, "Soft robotics: Biological inspiration, state of the art, and future research," *Appl. Bionics Biomechanics*, vol. 5, no. 3, pp. 99–117, Jul. 2008. [Online]. Available: http://dx.doi.org/10.1080/11762320802557865

[3] J. M. Bern, K.-H. Chang, and S. Coros, "Interactive design of animated plushies," in *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4. ACM, 2017.

[4] Y. Yamashita, T. Ishikawa, H. Mitake, Y. Takase, F. Kato, I. Susa, S. Hasegawa, and M. Sato, "Stuffed toys alive!: cuddly robots from fantasy world," in *ACM SIGGRAPH 2012 Emerging Technologies*. ACM, 2012, p. 20.

[5] M. Bächer, B. Hepp, F. Pece, P. G. Kry, B. Bickel, B. Thomaszewski, and O. Hilliges, "Defsense: Computational design of customized deformable input devices," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ser. CHI '16. New York, NY, USA: ACM, 2016, pp. 3806–3816. [Online]. Available: http://doi.acm.org/10.1145/2858036.2858354

[6] A. Belouchrani and M. G. Amin, "Time-frequency music," *IEEE Signal Processing Letters*, vol. 6, no. 5, pp. 109–110, May 1999.

[7] R. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Transactions on Antennas and Propagation*, vol. 34, no. 3, pp. 276–280, Mar 1986.

[8] H. Lipson, "Challenges and opportunities for design, simulation, and fabrication of soft robots," *Soft Robotics*, vol. 1, no. 1, pp. 21–27, 2014.

[9] G. G. Rigatos, "Particle filtering for state estimation in nonlinear industrial systems," *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 11, pp. 3885–3900, Nov 2009.

[10] C. Rendl, D. Kim, S. Fanello, P. Parzer, C. Rhemann, J. Taylor, M. Zirkl, G. Scheipl, T. Rothländer, M. Haller, and S. Izadi, "Flexsense: A transparent self-sensing deformable surface," in *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '14. New York, NY, USA: ACM, 2014, pp. 129–138. [Online]. Available: http://doi.acm.org/10.1145/2642918.2647405

[11] R. Plankers and P. Fua, "Articulated soft objects for multiview shape and motion capture," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 9, pp. 1182–1187, Sept 2003.

[12] X. Lu, W. Xu, and X. Li, "A soft biomimetic tongue: model reconstruction and motion tracking," pp. 97 970Y–97 970Y–8, 2016. [Online]. Available: http://dx.doi.org/10.1117/12.2218318

Central Pattern Generator with Inertial Feedback for Stabilized Locomotion in Unstructured Terrain

Samuel Shaw[1], Guillaume Sartoretti[2], Matthew Travers[3] and Howie Choset[4]

Abstract— **Replicating the agility of legged animals is essential to developing legged robots that can navigate the unstructured terrain presented by exploration and rescue situations. In this sense, biologically-inspired central pattern generators (CPG) have been widely studied. Although current CPG models provide universal strategies for open loop gait generation, few provide concrete methodologies for incorporating sensory feedback. In this contribution, fill this gap by incorporating a generalized inertial stabilization approach within a CPG framework that provides strict control of body orientation during locomotion. In trials of blind locomotion using a modular, series-elastic hexapod robot, we show that strict body control increases a robot's locomotive ability while traversing unstructured terrain; body-control both increases the speed of locomotion and the steepness of terrain a robot can traverse, while filtering out minor disturbances to produce a stable base.**

I. Introduction

Legged locomotion of robots proves superior to that of wheels and other types of locomotion in unstructured environments due to its high degree of adaptability; as such, this technology continues to be the most viable in search and rescue situations. However, the aforementioned adaptability comes at a cost, and is the reason why the locomotion of legged robots is so complex. Optimally modelling the basic coordination of limbs through their degrees-of-freedom to produce and adapt stable gaits is a on-going area of research – both in the context of robotics and biology.

Luckily, nature provides an effective solution to the complex problem of legged locomotion; legged animals of varying sophistication subconsciously navigate extreme terrain with ease. Biological Central Pattern Generators (CPGs) are neural networks located in the spinal cord that produce signals to drive the rhythmic motions required for locomotion [1] [2]. Although CPGs do not rely on sensory inputs to produce locomotive outputs, these outputs are adapted based on proprioceptively-sensed environmental information [3] [4]. In this way, sensory input can adapt the gait produced by the CPG in real time by inhibiting or extenuating certain characteristics [5] [6].

In robotics applications, using a CPG-based model provides a versatile framework for gait generation and control. Mathematically modeling of biological CPGs provides a decentralized, dynamic design for locomotion controllers [7]. In the context of legged locomotion, one can select the step height, step length, gait speed, and the gait itself by adjusting the model parameters. Unfortunately, CPG-inspired modelling generally lacks a generic methodology for integrating sensory feedback to adapt the locomotion model [2].

In this contribution, we present methods for manipulating these parameters for the improved blind locomotion of a hexapod robot. We use sensing capabilities provided by Inertial Measurement Units (IMU) within the body to develop a strict body control stabilization strategy; our robust control strategy allows a blind, legged robot encounter and react to rough, unstructured terrain without a-priori knowledge of the environment. That is, we use basic on-board sensing to replicate the feedback that a biological system's peripheral nervous capabilities would provide to the CPG. Our results show that the implementation of such a control strategy using only rudimentary on-board sensing can greatly enhance a robot's locomotive ability on basic terrains and extend its ability to allow it to traverse more complicated and extreme terrains.

This contribution is structured as follows: Section II provides background on the CPG model and describes how we extend the model to integrate inertial feedback. Section III explains our metric for determining error in end effector position and how we use it to compute joint offsets to achieve stability. Section IV presents our specific implementation and experimental setup. Section V evaluates the method's performance. Finally, Section VI summarizes our contribution and offers suggestions for future work.

G. Sartoretti[2], M. Travers[3] and H. Choset[4] are with the Robotics Institute at Carnegie Mellon University, Pittsburgh, PA 15213, USA. S. Shaw[1] is with Tufts University, Medford, MA 02155. {gsartore}@andrew.cmu.edu, {mtravers@andrew,choset@cs}.cmu.edu, {samuel.shaw}@tufts.edu

II. CPG Modeling

We model the CPG as a dynamical system of linked Hopf oscillators in the joint space for a n-legged robot with at least two joints per leg; $x = [x_1, ..., x_n]$ represents the angles of the proximal, lateral joints and $y = [y_1, ..., y_n]$ represents the angles of distal, vertical joints [2]:

$$\begin{cases} \dot{x_i} = \gamma(1 - \frac{x_i^2}{a^2} - \frac{y_i^2}{b^2}) \cdot x_i - \omega \frac{a}{b} \cdot y_i \\ \dot{y_i} = \gamma(1 - \frac{x_i^2}{a^2} - \frac{y_i^2}{b^2}) \cdot y_i - \omega \frac{b}{a} \cdot x_i + \lambda \sum K_{ij} y_j, \end{cases} \tag{1}$$

where a and b define the major and minor axes of the limit ellipse (i.e. the maximum magnitude attained by x and y, respectively), ω defines the angular speed of the gait cycle, γ defines the forcing to the limit cycle, λ defines the coupling strength, and K, the coupling matrix [2], defines the gait by setting the phase relationship between legs.

In order to extend the CPG in Eq.(1) to incorporate inertial feedback for stabilization, we consider the offsets $c_{x,i}$ and $c_{y,i}$ as the center of the elliptic limit cycle to achieve the following modified CPG-model equations:

$$\begin{cases} \dot{x_i} = \gamma(1 - \frac{(x_i - c_{x,i})^2}{a^2} - \frac{(y_i - c_{y,i})^2}{b^2}) \cdot (x_i - c_{x,i}) \\ \qquad\qquad - \omega \frac{a}{b} \cdot (y_i - c_{y,i}) \\ \dot{y_i} = \gamma(1 - \frac{(x_i - c_{x,i})^2}{a^2} - \frac{(y_i - c_{y,i})^2}{b^2}) \cdot (y_i - c_{y,i}) \\ \qquad -\omega \frac{b}{a} \cdot (x_i - c_{x,i}) + \lambda \sum K_{ij} (y_i - c_{y,i}). \end{cases} \tag{2}$$

By applying the offsets as presented in Eq.(2), the operating window of each limb can be adjusted to achieve stability while preserving step height and speed, as described in the next section.

III. Inertial Stabilization

In this section, we present our method for stabilization in the null space of the lateral joints by adapting $c_{y,i}$ using inertial feedback and incorporating this correction into the CPG-model equations. We first present our method for computing the end effector error, before exploring an approach to obtain CPG offsets from this error.

A. Computing CPG Offsets Using End Effector Error

We first define the pose of the body, $P \in SO(3)$, as the orientation of the body in the world frame. $R \in SO(3)$ is then defined at the rotation which positions the end effectors such that the body observes a desired orientation while retaining its heading in the reference frame of the world. Specifically, R is obtained as follows:

$$\begin{cases} z_y = \dfrac{\begin{pmatrix} -i & 1 & 0 \end{pmatrix} \cdot \left(P \cdot \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}^T \right)}{\| \begin{pmatrix} -i & 1 & 0 \end{pmatrix} \cdot \left(P \cdot \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}^T \right) \|_2} = e^{i\theta_z} \in \mathbb{C} \\ R = R_z(\theta_z) = \begin{pmatrix} \cos(\theta_z) & -\sin(\theta_z) & 0 \\ \sin(\theta_z) & \cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{pmatrix} \in SO(3). \end{cases} \tag{3}$$

By applying forward kinematics, we obtain the end effector positions, $r \in \mathbb{R}^{3 \times n}$, with n the number of legs, in Cartesian space in the reference frame of the robot:

$$r = \begin{bmatrix} x_1 & \cdots & x_n \\ y_1 & \cdots & y_n \\ z_1 & \cdots & z_n \end{bmatrix}. \tag{4}$$

The corrected end effector positions, $\tilde{r}$, are then obtained by first shifting the end effector positions into the world frame before further applying the rotation R to achieve the desired body orientation:

$$\tilde{r} = R \cdot (P \cdot r). \tag{5}$$

The z-components of $\tilde{r}$ are then adjusted to ensure that the body height, defined as the vertical distance between the geometric center of the body and the estimated ground plane, remains constant. We define e to be the error in end effector position of each leg in the reference frame of the body, which reads

$$e = P^T \cdot (\tilde{r} - r). \tag{6}$$

Using the well-known kinematics equation,

$$\dot{x} = J\dot{\theta}, \tag{7}$$

expanded as follows,

$$\begin{bmatrix} \dot{x_1} & \cdots & \dot{x_n} \\ \dot{y_1} & \cdots & \dot{y_n} \\ \dot{z_1} & \cdots & \dot{z_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \theta_1} & \cdots & \frac{\partial x}{\partial \theta_m} \\ \frac{\partial y}{\partial \theta_1} & \cdots & \frac{\partial y}{\partial \theta_m} \\ \frac{\partial z}{\partial \theta_1} & \cdots & \frac{\partial z}{\partial \theta_m} \end{bmatrix} \cdot \begin{bmatrix} \dot{\theta_{1,1}} & \cdots & \dot{\theta_{1,n}} \\ \vdots & & \vdots \\ \dot{\theta_{m,1}} & \cdots & \dot{\theta_{m,n}} \end{bmatrix}, \tag{8}$$

where $\dot{x}$ is the end effector velocity, J is the linear Jacobian, $\dot{\theta}$ is the angular speed of the joints, n represents the number of legs present on the robot, and m represents the number of joints composing each leg, we can estimate the error in joint angles:

$$\Delta\theta_e \approx J^{-1}e, \tag{9}$$

where e is the error in end effector position given by Eq.(6).

To ensure that joint angle correction does not diverge, we add a constant dissipation to the origin:

$$\Delta\theta_e = \Delta\theta_e - \alpha \cdot \theta_e, \tag{10}$$

where $\alpha \in \mathbb{R}$ governs the dissipative term.

B. Adaptation of CPG Parameters

By integrating the error in joint angle, denoted $\Delta\theta_e$, with respect to time as shown in Eq.(11), we obtain joint offsets that correspond to the desired corrections in the orientation and height of the robot's body:

$$\theta_e = \int_0^t \Delta\theta_e dt. \tag{11}$$

As mentioned in Section II, an offset on the proximal, lateral joint is not needed for stabilization and not desired to ensure that the heading is maintained for locomotion. Thus, it follows that the first row of $\Delta\theta_e$ should be zero. The second row gives offsets for the distal, vertical joints, which are used to adjust c_y to obtain stability as shown in Eq.(12).

$$c_{y,i} \equiv \theta_{e_{2,i}} \tag{12}$$

Furthermore, to ensure smooth transition, we translate the y_i by $\Delta\theta_{e_{2,i}}$. Therefore, the final form of the CPG motion equations reads

$$\begin{cases} \dot{x_i} &= \gamma(1 - \frac{(x_i - c_{x,i})^2}{a^2} - \frac{(y_i - c_{y,i})^2}{b^2}) \cdot (x_i - c_{x,i}) \\ & \quad - \omega\frac{a}{b} \cdot (y_i - c_{y,i}) \\ \dot{y_i} &= \gamma(1 - \frac{(x_i - c_{x,i})^2}{a^2} - \frac{(y_i - c_{y,i})^2}{b^2}) \cdot (y_i - c_{y,i}) \\ & \quad -\omega\frac{b}{a} \cdot (x_i - c_{x,i}) + \lambda \sum K_{ij}(y_i - c_{y,i}) \\ & \quad +\Delta\theta_{e_{2,i}}. \end{cases} \tag{13}$$

IV. Experiments

In this section, we begin by describing the specifications of the hexapod robot used for our implementation. Then, we detail how we appropriately initialize CPG-model parameters, and how we develop joint angles for each leg from the CPG-model by distributing tasks among the joints of each leg. Finally, we present our experimental results that compare open-loop to stabilized locomotion.

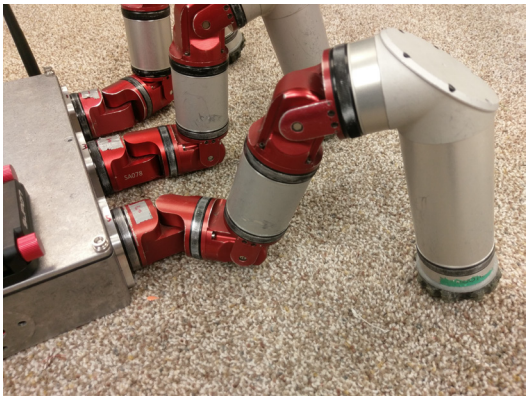


Fig. 1: Hexapod leg joint configuration.

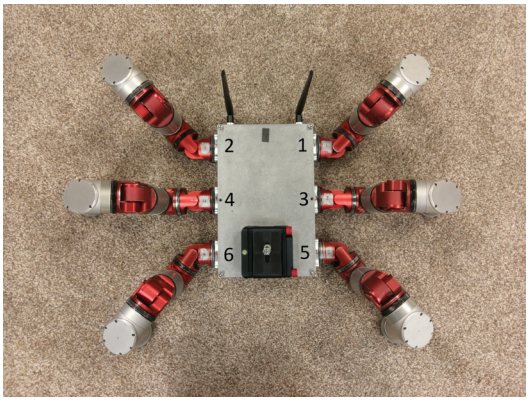


Fig. 2: Hexapod leg numbering convention.

A. Robot Specifications

We use a modular, series-elastic hexapod robot as an experimental platform [8]. Each leg of the hexapod, shown in Fig. 2, consists of three joints: a proximal joint aligned with the yaw axis, and intermediate and distal joints aligned with the roll axis (when the proximal joints align the legs such that they are parallel to the pitch axis).

The robot's body is a rectangular prism with length, width, and height dimensions of 27cm, 17cm, and 7cm, respectively. Legs may extend from the body approximately 44cm horizontally or 32cm vertically.

The robot is blind, meaning that no on-board vision system is used for the purpose for step planning; on-board sensing is provided by the joint modules themselves. Each of the eighteen modules contains an IMU and series-elastic element [9] that provides torque-sensing capabilities.

B. CPG Implementation

Since a hexapod robot is used, an alternating tripod gait is chosen for its prevalence as a locomotive gait

in insects, and for its static stability; three legs always remain on the ground, forming a constant, large support polygon. In this sense, the stability of the robot in unstructured terrain relies only on the integrity of environment footholds. To implement this gait within the CPG framework, we define the coupling matrix, K:

$$K = \begin{bmatrix} 0 & -1 & -1 & 1 & 1 & -1 \\ -1 & 0 & 1 & -1 & -1 & 1 \\ 0 & 1 & 0 & -1 & -1 & 1 \\ 1 & -1 & -1 & 0 & 1 & -1 \\ 1 & -1 & -1 & 1 & 0 & -1 \\ -1 & 1 & 1 & -1 & -1 & 0 \end{bmatrix}. \quad (14)$$

With knowledge of the gait's behavior, we define the estimated ground plane based on the positions of the lowest tripod, which is given by either $\tilde{r}_{3,(1,4,5)}$ or $\tilde{r}_{3,(2,3,6)}$.

We initialize constant offsets, c_x and c_y on the center of the CPG limit cycle on a leg-by-leg basis:

$$\begin{cases} c_x = \begin{bmatrix} \frac{\pi}{4} & \frac{\pi}{4} & 0 & 0 & -\frac{\pi}{4} & -\frac{\pi}{4} \end{bmatrix} \\ c_y = \begin{bmatrix} \frac{\pi}{16} & \frac{\pi}{16} & \frac{\pi}{16} & \frac{\pi}{16} & \frac{\pi}{16} & \frac{\pi}{16} \end{bmatrix} \end{cases}. \quad (15)$$

It should be noted that c_x remains as initialized; it defines only the spread of the legs, and does not need to be adjusted in order to achieve stability.

The remaining CPG parameters are initialized as outlined below:

$$\begin{cases} \gamma = 40 \\ \lambda = \frac{1}{4} \\ a = \frac{\pi}{18} \\ b = \frac{\pi}{6} \end{cases}. \quad (16)$$

Note that since the values of x and y are used directly to assign joint angles to the proximal and intermediate joints, a and b are in radians.

Following convergence, the joint angles are set based on the CPG according to Eq.(17):

$$\begin{cases} \theta_{1,i} = x_i \\ \theta_{2,i} = \max(y_i, c_{y,i}) \\ \theta_{3,i} = f(\theta_{1,i}, \theta_{2,i}) \end{cases}. \quad (17)$$

As shown above, the x-outputs from the CPG determine $\theta_{1,i}$, the angles of the proximal joints. Similarly, the y-outputs describe $\theta_{2,i}$, the angles of the intermediate joints, with one exception: assuming the legs are on the ground when $y_i < c_{y,i}$, we set $\theta_{2,i} = c_{y,i}$ to ensure a constant body height throughout the stance phase.

Inverse kinematics are used to determine $\theta_{3,i}$, the angle of the distal joint to ensure that as the proximal and intermediate joints move, the end effector remains at a constant, desired distance from the body. Given this architecture, we allow the intermediate joint to stabilize while the distal joint ensures an ideal, linear end effector trajectory. In this sense, we distribute constraints among each leg's degree-of-freedoms to ensure that tasks do not interact adversarially.

C. Experimental Results

Two metrics were used to measure the efficacy of our body stabilization strategy: locomotive speed and error in body orientation. Experiments were conducted on a twelve-degree slope, featuring randomly-placed and -oriented rocks that varied in elevation from approximately 5cm to 20cm. A twelve-degree slope provided a suitable comparison between methods, as its steepness is appropriate for examining both stability and locomotive speed (i.e., it is neither so steep that the open-loop controller will be unable to progress nor so shallow that body stability is not longer as essential).

We used internal sensing provided by an IMU in each of the six proximal joints to determine the error in orientation, E:

$$\begin{aligned} E &= \cos^{-1}\left(\frac{z \cdot (P \cdot z)}{\|P \cdot z\|}\right) \\ z &= \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \end{aligned}. \quad (18)$$

An over-head camera was used to capture the robot's movement up the slope with respect to time; this information was used to determine the speed, and thus overall locomotive performance, during each of the trials.

Figures 3 and 4 present our experimental results. Specifically, Figure 3 documents the average speed of the hexapod during each of the twelve trials. Six trials serve as the control group and incorporate no sensory feedback (red); another six employ the described inertial stability strategy (blue). Additionally, the mean speeds with overlaid standard deviations for both the stabilized and control sets are shown (gray). Figure 4 shows the average error in body orientation across trials where stabilization was enabled (blue). The mean across all trials is shown (gray).

V. Discussion

As evidenced by Figures 3 and 4, the inertial-based body control for stabilization greatly improves locomotive performance in terms of both speed and body orientation error. It is important to point out the obvious

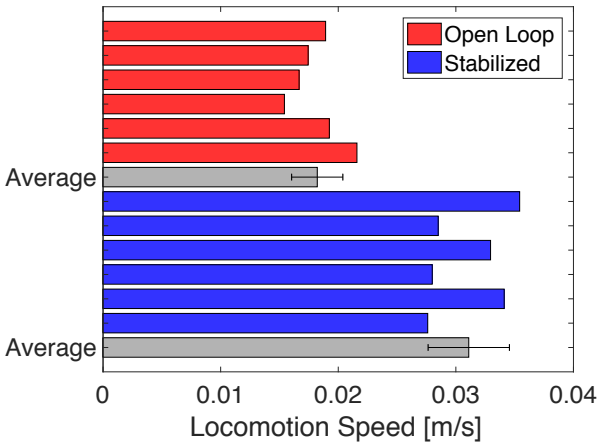


Fig. 3: Performance of stabilized trials (blue) and open-loop trials (red); average values are shown in gray with overlaid standard deviations. Locomotive speed greatly improves when robot body orientation is strictly controlled.

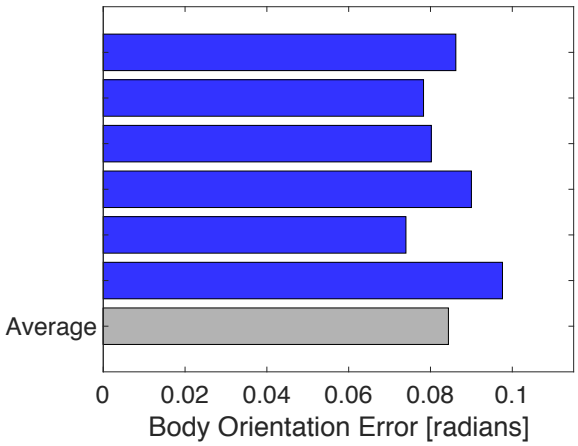


Fig. 4: Body orientation error for each stabilized trial (blue), and average across all trials (gray) (shown in radians, $\sim 5°$). Error in orientation is minimal given the twelve-degree test incline and unstructured nature.

trade-off between speed and stability; for a given navigable terrain, as speed increases, stability inherently diminishes. The speed at which significant degradation in stability occurs largely depends on the roughness of terrain. In our trials, we prioritized stability more than climbing speed.

Locomotive performance increases for several reasons when a body stabilization strategy is employed. By orienting the body such that the lateral, proximal joints responsible for propelling the body forward – do not have to overcome a component of the gravitational force, we are able to increase the robots overall speed.

Additionally, this body orientation naturally positions the body's center of gravity (COG) more centrally within the support polygon formed by grounded legs. This positioning of the COG ensures that the normal force exerted on the end effector of each grounded leg remains significant, minimizing foot slip. Leg slip is problematic for a variety of reasons. Uneven leg slip can reorient the robot, changing its heading. Redirecting the robot – especially if its legs continue to slip – can prove especially difficult. In the context of stability, leg slip can introduce rapid changes in body position which not only disrupts locomotion, but would also be very detrimental to an on-board vision system.

In this sense, keeping the COG well within the support polygon is also vital for navigating more extreme terrain. In fact, with stabilization, the hexapod traversed unstructured terrain of up to 25 degrees – unimaginable for an open-loop locomotion model. Additionally, this body orientation prevents the robot from tipping while stepping onto or off of an obstacle; with a body stabilization strategy, if the robot is able to step onto an object, it will be able to stabilize.

VI. Conclusion

In this paper, we present a method integrating an inertial stabilization approach within a CPG. We highlight the beneficial adaptability a CPG locomotive controller presents and detail how we used internal feedback to

modulate CPG parameters and achieve dynamic stability. Using a CPG framework generalizes the method, so that it may serve as a fundamental tool for future research on legged locomotion or a potential building block for vision-based applications.

We then discuss our specific CPG implementation using a high-DOF, modular, series-elastic hexapod robot. The results of this controller's performance indicate that strict body control improves blind locomotion, by smoothing out sudden perturbations and increasing locomotive speed. Additionally, we conclude that stabilization can allow a robot to traverse steeper and more challenging terrains, where open-loop locomotion would otherwise fail.

Future work will explore the addition of a vision system. Up to this point, we have examined the body's stability both visually and numerically, through examining the orientation given by IMU data. Developing a frame-based metric for determining to what extent frames are useable for vision application will be critical not only in further improving the stabilization strategy but also to provide greater application for it.

VII. Acknowledgements

The authors thank Carnegie Mellon University undergraduates Katie Lam and Naixin Fan for their participation in the project. Samuel Shaw would like to thank the National Science Foundation for funding his work on this project as part of the 2017 Carnegie Mellon University Robotics Institute Summer Scholar (RISS) program.

References

[1] Philip Holmes, Robert J Full, Dan Koditschek, and John Guckenheimer. The dynamics of legged locomotion: Models, analyses, and challenges. *Siam Review*, 48(2):207–304, 2006.

[2] Ludovic Righetti and Auke Jan Ijspeert. Pattern generators with sensory feedback for the control of quadruped locomotion. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 819–824. IEEE, 2008.

[3] Alain Frigon and Serge Rossignol. Experiments and models of sensorimotor interactions during locomotion. *Biological cybernetics*, 95(6):607, 2006.

[4] Patrick J Whelan. Control of locomotion in the decerebrate cat. *Progress in neurobiology*, 49(5):481–515, 1996.

[5] Milan R Dimitrijevic, Yuri Gerasimenko, and Michaela M Pinter. Evidence for a spinal central pattern generator in humans. *Annals of the New York Academy of Sciences*, 860(1):360–376, 1998.

[6] Marilyn MacKay-Lyons. Central pattern generation of locomotion: a review of the evidence. *Physical therapy*, 82(1):69–83, 2002.

[7] Auke Jan Ijspeert. Central pattern generators for locomotion control in animals and robots: a review. *Neural networks*, 21(4):642–653, 2008.

[8] Matt Travers, Alex Ansari, and Howie Choset. A dynamical systems approach to obstacle navigation for a series-elastic hexapod robot. In *Decision and Control (CDC), 2016 IEEE 55th Conference on*, pages 5152–5157. IEEE, 2016.

[9] Gill A Pratt and Matthew M Williamson. Series elastic actuators. In *Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*, volume 1, pages 399–406. IEEE, 1995.

Distributed Learning for the Decentralized Control of Articulated Mobile Robots

Yunfei Shi
Department of Electrical Engineering
The Hong Kong Polytechnic University
Hong Kong
yunfei.shi@connect.polyu.hk

Guillaume Sartoretti
The Robotics Institute
Carnegie Mellon University
United States
gsartore@andrew.cmu.edu

William Paivine
The Robotics Institute
Carnegie Mellon University
United States
wjp@andrew.cmu.edu

Matthew Travers
The Robotics Institute
Carnegie Mellon University
United States
mtravers@andrew.cmu.edu

Howie Choset
The Robotics Institute
Carnegie Mellon University
United States
choset@cs.cmu.edu

***Abstract*—The locomotion of snake robots continues to be of interest to researchers because of the high degree of freedom, making it appropriate for a variety of terrains and applications. Current state-of-the-art approach to control a snake robot includes shape-based compliant control and its decentralized extension. Of these, decentralized shape-based compliant control shows promise for operating a snake robot in unstructured environments because it allows control over individual sections of the robot. However, even with this level of control, the snake cannot universally adapt to random obstacles, sometimes resulting in the robot entering a cyclic pattern preventing the steady progression. Recent studies on deep reinforcement learning shows promising performance of agents in unknown environments. A possible solution is to substitute the admittance controller with a reinforcement learning agent. In particular, we found that the state-of-the-art distributed learning framework for reinforcement learning employs ideas conceptually related to decentralized control; agents explore the environment independently and update the common policy asynchronously. However, no previous work focusing on the potential relationship between these two concepts exists. This work explicitly explores the potential relationship between the underlying concepts in decentralized control for articulated locomotion and distributed learning. We use the architecture of the asynchronous advantage actor-critic (A3C) algorithm to create a distributed framework for learning a decentralized control policy on the robot. We show that an individual agent in distributed learning framework, specifically in A3C algorithm, can be assigned to an independent portion of the robot, making distributed learning on single platform possible and efficient. Our experimental results show that the steady progression of a trained agent through randomized, highly cluttered environments is 40% better than that of a compliant controller.**

I. Introduction

The locomotion of snake robots continues to be of interest to researchers because of the high degree of freedom, making it appropriate for a variety of terrains and applications [1]. Recent researchers have shown that the locomotion of snake robots through unstructured environments can be improved by applying decentralized control [2], [3] which refers to an approach that divides the robot's body into distributed portions, coordinated later using central pattern generators [4]. New state-of-the-art in deep reinforcement learning – the asynchronous framework – employs a conceptually related idea to that of having distributed agents learning in the same environment to asynchronously update a global policy [6]. In this paper, we present a learning approach that uses a reinforcement learning algorithm with a distributed framework to learn a decentralized control policy. This work shows that each distributed portion of the robot can be regarded as an individual worker in a distributed learning framework.

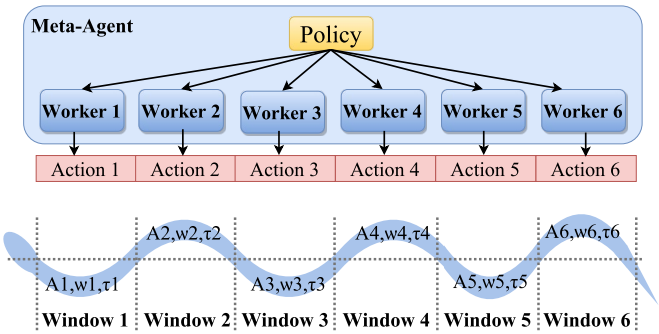


Fig. 1. Analogy between the A3C meta-agent containing multiple workers (top), and a snake robot separated into independent control windows (bottom) [5].

Current learning-based control approaches in robotics are often at robot-level [7], [8], [9]. The actions chosen using these approaches affect all degrees-of-freedom of the robot simultaneously. The action space is large due to the high degrees of freedom. Hence, a great amount of training episodes are needed to obtain a near-optimal policy. Distributed learning is distinguishing because it increases the training time efficiency by having multiple agents learn the optimal policy in a distributed manner. The Asynchronous Advantage Actor-Critique

(A3C) algorithm applies the asynchronous framework. A set of workers interact with the environment independently and receive a reward, updating the global policy asynchronously. This method is proven to converge faster than conventional approaches [10], [6].

The paper is outlined as follows: In Section II, we summarize the general shape-based controller for locomotion, whose feedback controller we seek to replace by a trained agent. Section III outlines the reinforcement learning problem structure, as well as the agent structure used in this work. We describe the training procedure of our agent in Section IV. We then validate the learned policy experimentally on a snake robot and discuss the results in Section V. Concluding remarks and future works are finally presented in Section VI.

II. Background -Decentralized Shape-Based Control for Locomotion

In this section, we review the result from serpenoid locomotion based on low-dimensional *shape-based* compliant control. We also overview the work which highlights the benefit of decentralized control of a snake robot in locomotion. The considered decentralized control is shared at different parts of the robot, making it possible to learn a common policy distributively in a time-efficient manner.

A. Serpenoid Curves

Studies in snake robots reveal that two sinusoidal waves propagating in two planes can be used to parameterize and control the gait of an N-joint snake robot [11], [1], [2]. Specifically, one wave controls joints that rotate in the dorsal plane of the snake, while the other wave controls the lateral joints:

$$\begin{cases} \theta_i^{lat}(t) &= \phi^{lat} + A^{lat}\sin(\omega_S^{lat} s_i^{lat} - \omega_T^{lat} t) \\ \theta_i^{dor}(t) &= \phi^{dor} + A^{dor}\sin(\omega_S^{dor} s_i^{dor} - \omega_T^{dor} t + \sigma), \end{cases} \tag{1}$$

where θ_i^{lat} and θ_i^{dor} are the commanded joint angles on the lateral and the dorsal plane, ϕ^{lat} and ϕ^{dor} the angular offset, A^{lat} and A^{dor} the amplitude of the curvatures, σ the phase shift between the two planes sine waves. ω_T^{lat} and ω_T^{dor} are the temporal frequencies, while ω_S^{lat} and ω_S^{dor} refers to the spatial frequencies of the waves, which determines the number of waves on the snake robot's body. $s_i^{lat} \in \{0, 2\cdot l_s, \ldots, N\cdot l_s\}$ and $s_i^{dor} \in \{l_s, 3\cdot l_s, \ldots, (N-1)\cdot l_s\}$ are the distances from the head of the module i, where l_s is the length of one module.

These parameters are used to define different open-loop joint trajectories for locomotion, i.e., gaits [12]. In this work, we consider the planar slithering gait, for which the corresponding open-loop serpenoid curve reads:

$$\theta_i(t) = \theta_0 + A\sin\left(\omega_S\, s_i - \omega_T\, t\right), \tag{2}$$

where the commanded joint angles are in the lateral plane while the dorsal joint angles are set to be zero.

B. Shape-based Compliant Control

Shape-based control is generally used to reduce the dimensionality of robots with high degree-of-freedom using shape functions [3]. The robot is controlled using parameters in Eq.(1) rather than at joint level. In this paper, we consider the shape function $h(A, \omega_S)$to be simply a planar slithering gait function $\theta(t)$, Eq.(2), where the lateral amplitude and lateral spatial frequency are the two controlling parameters while other parameters remain fixed.

$$\begin{array}{rcl} h: & \Sigma = \mathbb{R}^2 \mapsto \mathbb{R}^N \\ h_i(A(t), \omega_S(t)) & = & \theta_0 + A(t)\sin\left(\omega_S(t)\, s_i - \omega_T\, t\right). \end{array} \tag{3}$$

In the shape-based compliant controller, which is regarded as the state-of-the-art [3],the output control parameters ($A(t)$ and $\omega_S(t)$) are set using an admittance controller [13] which enable the parameters to change joint-level external torques. Therefore, the snake robot can adapt to the various environment surroundings by complying the shape. The state-space function of the admittance controller of $\beta = \big(A(t), \omega_S(t)\big)^T$ is:

$$M\,\ddot{\beta}(t) + B\,\dot{\beta}(t) + K\,(\beta(t) - \beta_0) = F(t), \tag{4}$$

where the $M, B, K \in \mathbb{R}^{2\times 2}$ refer to the effective mass, damping and spring constant matrices of the system respectively, and $F(t)$ is a mapping of the external torques $\mu_{ext}(t)$ from the joint space to the shape space, which is detailed below.

The shape function maps the external torque detected at each module of the snake (joint level) to shape space:

$$F(t) = J(h)|_{(A(t),\omega_S(t))} \cdot \mu_{ext}(t), \tag{5}$$

where the $J(h) \in \mathbb{R}^{2\times N}$ refers to the Jacobian matrix:

$$J(h) = \begin{pmatrix} \frac{\partial h_1(A,\omega_S)}{\partial A} & \cdots & \frac{\partial h_N(A,\omega_S)}{\partial A} \\ \frac{\partial h_1(A,\omega_S)}{\partial \omega_S} & \cdots & \frac{\partial h_N(A,\omega_S)}{\partial \omega_S} \end{pmatrix} \tag{6}$$

In this paper, we present a method which substitutes the admittance controller with the learning agent mentioned above and we use shape function to reduce dimensionality.

C. Decentralized Control

Decentralized control is proposed based on shape-based compliant control [2]. It creates different groups of joints on the snake by activating the windows along the snake backbone, that is, the joint group in one window is isolated from joints in other windows. The decentralized architecture allows the different portions of the snake to react to the environment independently which also has advantages in an unknown, changing environment [2].

The motion parameters for each window are defined using the sigmoid function:

$$\beta(s,t) = \sum_{j=1}^{W} \beta_{s,j}(t)\left[\frac{1}{1+e^{m(s_{j,s}-s)}} + \frac{1}{1+e^{m(s-s_{j,e})}}\right] \tag{7}$$

where m controls the steepness of the windows, W is the number of windows, and $\beta_{s,j}(t)$ are the values of the serpenoid parameters in window j at position s along the backbone of

the snake. It is worth mentioning that the windows on the snake are not fixed. It can move along the snake backbone with the same velocity as the moving serpenoid curve, which is refereed to sliding-window. In our case, we consider the decentralized control with sliding windows with the agents allocated to different windows.

III. Policy Representation

In this section, we present the detailed structure of the learning agent. We choose A3C to distributively learn the common stochastic policy [6]. The worker agents then choose the action based on the common policy they learned. Intuitively, the meta-agent in the distributed learning framework can be regarded as the robot, while the worker agents can represent the separate parts of the robot where the worker agents directly interact with the surroundings and update the finding to the meta-agent to update the common policy.

A. State Space

The state s is a 7-vector of shape-based quantities:

$$s = \langle \mu(t),\, \beta(s,t)^T,\, F(s,t)^T,\, \beta_0^T \rangle. \tag{8}$$

where $\beta(s,t)$ represents the current shape parameters of a window on the snake (i.e., amplitude and spatial frequency), β_0 is their nominal values to which the shape parameters will converge to in absence of external forces. $F(t)$ refers to the external torque for the same window in shape space as shown in Eq.(5). $\mu(t) \in [0;1]$ is the *modular time* we defined to feature the information about the cyclic nature of the serpenoid gait. It is defined as $\mu(t) = \frac{t \mod T_s}{T_s}$, with $T_s = \frac{2\pi}{\omega_T}$, the period of the serpenoid curve in Eq.(1).

B. Action Space

In order to reduce the dimensionality of the action space, instead of allowing continuous increments in shape parameters as in the admittance controller, we discretize the possible increments of these shape parameters. In this case, the action space is defined as a vector with two parameters $a = \langle a_A, a_\omega \rangle$, with $a_A \in \{0, \pm\Delta_A\}$ and $a_\omega \in \{0, \pm\Delta_\omega\}$.The shape paremeters are updated by applying the selected increment combination.

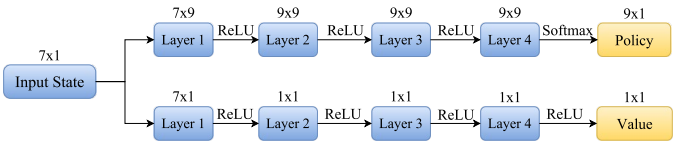


Fig. 2. Policy (Actor) and Value (Critic) network used in this paper to approximate their respective functions. Each network is composed of 4 fully-connected layers, with ReLU's as the activation function, except for the output of the last policy layer which uses a softmax function to estimate the stochastic policy.

C. Actor-Critic Network

We design the Actor-Critic networks as two neural networks with weights Ψ_A, Ψ_C both using four fully connected layers to approximate the stochastic policy and the value function respectively. As shown in Figure 2, where the activation function of each layer is a rectifier linear unit (ReLU) for introducing nonlinearity, with exception to the final layer of the actor network which uses softmax function to estimate the stochastic policy.

We set six worker-agents corresponding to each of the six windows mentioned in Section II-C on the snake to explore the environment concurrently. Each worker calculates the gradient of weights in the networks during each episode and optimizes the weight using a gradient descent method. The global network is updated at the end of each episode.

To update the policy, we select an entropy-based loss function [10]:

$$f_\pi(\Psi_A) = \log\left[\pi(a_t \mid s_t; \Psi_A)\right](R_t - V(s_t; \Psi_A)) + \kappa \cdot H\left(\pi(s_t; \Psi_A)\right) \tag{9}$$

$$f_v(\Psi_C) = \left(R_t - V(s_t; \Psi_C)\right)^2, \tag{10}$$

where $(R_t - V(s_t; \Psi_A))$ represents the advantage which estimates the effects of actions on reward using discounted reward R_t and $\kappa \cdot H\left(\pi(s_t; \Psi_A)\right)$ represent the entropy for managing exploration with the help of factor κ. An increase of κ will result in a more exploitative policy. We update the stochastic policy using policy gradient.

The value loss function is set as Eq. 10 to depict the temporal difference error. Both loss functions are calculated at the end of each episode and optimized using ADAM optimizer [14].

D. Shared Rewards

We set the shared reward based on the instantaneous forward progression as shown in Eq. 11, while the worker-agents receive individual state information and can select different action based on the common policy. Our hypothesis is that shared reward could allow the joint actions of all agents to be valued simultaneously, in order to better train them for a collaborative locomotion task.

The simultaneous progression for generating rewards is calculated using current position and the initial position.

$$r_t = \tanh\left(\lambda_r \cdot \|X(t) - X_{0,i}\|_2\right), \tag{11}$$

where λ_r a scaling factor, $X(t)$ and $X_{0,i}$ refers to the current position of the robot and initial position at the beginning of each episode, respectively. The $\tanh$ function is applied to normalized the reward.

The global structure of the considered A3C meta-agent is illustrated in Figure 3.

IV. Learning

In order to learn the policy Ψ on robot hardware with a reasonable amount of trials, we use the idea replaying experience from the database to train the learning agent.

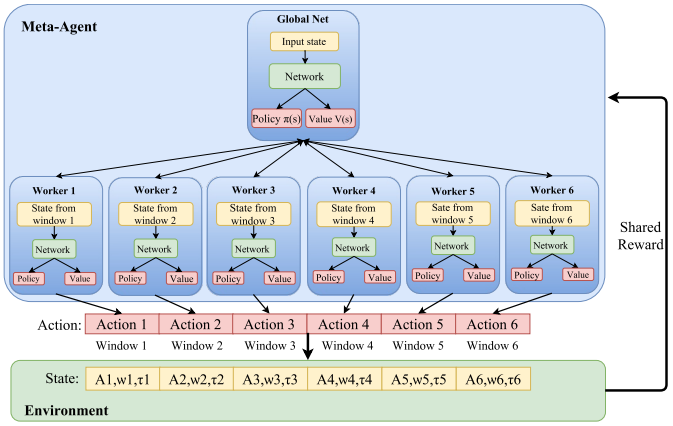


Fig. 3. Structure of the A3C Meta-Agent, with global weights for the Actor-Critic network. Each internal agent (worker) updates its local weights based on shared rewards during local interactions with the environment, and regularly pushes its gradients to the global networks. Each worker draws its state from the current Rewards are calculated in response to a vector of the 6 agents' action vector, after it has been enacted on the robot.

This has advantages of stabilizing and improving the learned policy [15].

Instead of learning from scratch, we propose to bootstrap using recorded actions of the current state-of-art compliant controller in order to sample the state-action space effectively in the more relevant region of efficient locomotion. In addition, this approach ensures that the learning agent can further improve the policy based on the compliant controller rather than converging to the near-optimal policy first. Therefore, the approach is time-efficient.

We collected data for 310 trials of the compliant controller in the peg array (Figure 4). The diagonal terms of three 2×2 diagonal matrices (i.e., a total of 6 parameters) M, B, and K, controlling the mass, damping, and spring constant of the system in the compliant controller are drawn at random from a uniform distribution over $[\frac{1}{2}; 5]$.

Each trial takes approximately 15 seconds and for each time step, the state and action are recorded according to Section III-A and III-B. The reward is calculated with position data provided from the overhead Optitrack system using Eq.(11). Eight OptiTrack beads are placed on the snake backbone. We store the collected data in a single experiment database.

During the training, we set six low-level worker-agents corresponding to the six windows on the snake. Each agent asynchronously selected a random run and a random starting point in that run. The agent then learns based on state-action and reward collected 89 time steps which match the length of half of a gait cycle as an episode before starting a new episode. The choice of episode length we selected provides agents with enough temporally-correlated experiences. After $50,000$ episode drawn by worker-agents, we obtained the learned policy.

We set the off-line learning parameters as:

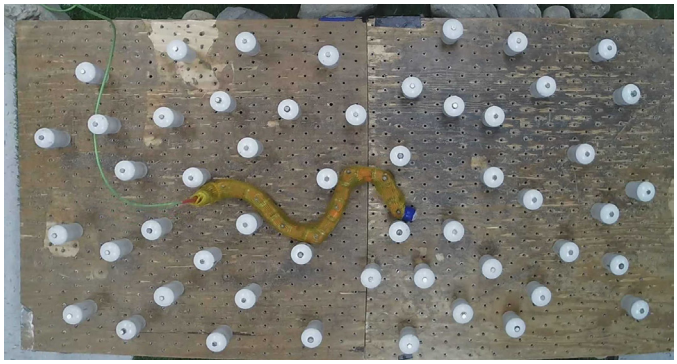


Fig. 4. Snake robot wearing a polyester sleeve traversing an unstructured peg array.

$$\Delta_A = 0.005 \qquad \Delta_\omega = 0.012 \qquad \lambda_r = 100$$
$$\gamma = 0.995 \qquad \alpha_A = \alpha_C = 0.0001 \qquad \kappa = 0.01,$$

where γ is the reward discount factor, and α_A, α_C are the learning rates of the ADAM optimization step for the Actor, Critic networks respectively.

V. Experimental Validation

To test the performance of the learned controller, we implement the learned controller to adapt the shape parameters of a snake robot in a decentralized manner to be consistent with the decentralized compliant controller in Section II-C. The experiments are run in a randomized, highly cluttered peg array to compare the average forward progression of the robot when shape parameters are updated by the trained agent or the compliant controller Eq.(4). .

A. Experimental Setup

Two decentralized controllers using 1) the learned policy, and 2) the compliant controller Eq.(4) to adapt shape parameters are implemented and tested on a snake robot with 16 identical series elastic modules [16]. Only 8 modules in the lateral plane are considered in this situation since the neighboring modules are torsionally rotated 90 degrees. The snake is covered with a polyester sleeve to reduce the friction with the environment and better locomote. Position data is also collected using the overhead OptiTrack System (NaturalPoint Inc., 2011).

B. Experimental Results

We calculated the gait cycle needed to traverse one meter and its variance over 10 trials according to [2] to compare the performance of both controllers. This evaluation metric discards the potential effect of running gaits at different temporal frequencies. This metric removes the potential variations due to running the gaits with differing temporal frequencies, and therefore only compare the kinematic ability of the two controllers. Moreover, the variance also indicates the repeatability of the controller.

We use the same peg array distribution for both controllers and the start position of the snake is identical for each pair of runs to ensure the same initial condition. We also set the temporal frequency and time step size to be the same for both the learning-based controller and compliant controller.

For the compliant controller, the following empirically optimal parameters are selected:

$$M = \begin{bmatrix} 1.5 & 0 \\ 0 & 2 \end{bmatrix} \qquad B = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix} \qquad K = \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\beta_0 = \begin{bmatrix} \frac{\pi}{4} \\ 3\pi \end{bmatrix} \qquad \omega_T^{lat} = 1.8 \qquad dt = \frac{\pi}{160}$$

The average time to traverse of the learning-based controller is $2.43\,[cycles/m]$ over 10 trials with variance of 0.092, while the compliant controller achieves $3.41\,[cycles/m]$ on average with a variance of 0.42. The experimental results of the compliant controller closely matches or are slightly better than previous results in [2], [3] where they show the result significantly increase when using the admittance controller. As summarized in Figure 5, the learned policy outperforms the compliant controller by more than 40% while the variance is decreased, implying that the learning based controller is more stable and more repeatable.

Figure 6 shows how the snake with a learning-based controller adapts the shape parameters to adapt to the local configuration only relying on local force sensing.

VI. Conclusion

In this paper, we consider the problem that the current state-of-the-art control method of shape-based control cannot universally adapt to the unknown environment and elucidate the potential relationship between decentralized control and distributed learning.

To this end, we regard the problem as a Markov Decision Process (MDP) and use A3C to solve this problem for the distributed learning framework which closely matches the decentralized control architecture of the underlying locomotion controller.

We train a meta agent with six lower-level worker-agents using the data collected from the state-of-the-art controller to sample the states and actions in the region close to efficient locomotion. The learning-based controller which follows the learned policy is then implemented on the snake. The learning-based controller outperforms the compliant controller by more than 40% in terms of steady progression through randomized, highly cluttered and unstructured environment.

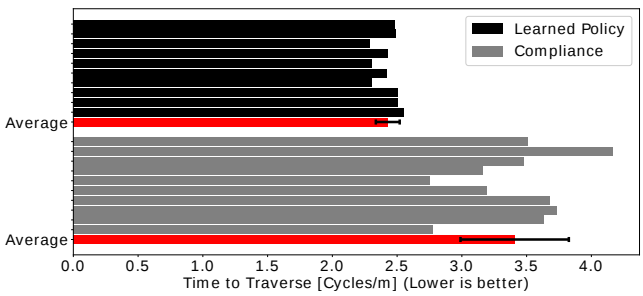


Fig. 5. Experimental results, comparing the performance of both controllers during the 10 trial runs in terms of serpenoid gait cycles needed to progress one meter. Lower values are better (i.e., larger forward progression per gait cycle). The learning-based controller (in black) can be seen to outperform the state-of-the-art compliant controller [2] by more than 40%.

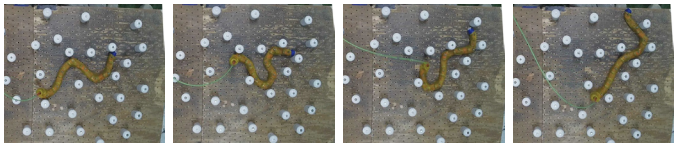


Fig. 6. Example frames sampled from one of the trials, showing the behavior of the learning-based controller in the peg array. Note how the amplitude and frequency of the waves in each of the windows of the snake change to fit the peg array.

Apart from the improvement in the steady progression of the highly articulated robot, the proposed approach also shows that an individual agent in a distributed learning framework, specifically in the A3C algorithm, can be assigned to an independent portion (e.g., groups of joints, limbs, etc.) of the robot, making distributed learning on a single platform possible and efficient.

In future works, we will consider applying this approach online in a more challenging environment, for example, on rocks. Experience can be gathered by all worker-agents corresponding to independent portions of the robot's body to learn the common policy. We are also interested in applying this approach to other platforms. Specifically, a distributed learning approach can be applied to a legged robot by assigning the low-level worker-agents to the limbs of the legged robot.

References

[1] M. Tesch, K. Lipkin, I. Brown, R. Hatton, A. Peck, J. Rembisz, and H. Choset, "Parameterized and scripted gaits for modular snake robots," *Advanced Robotics*, vol. 23, no. 9, pp. 1131–1158, 2009.

[2] J. Whitman, F. Ruscelli, M. Travers, and H. Choset, "Shape-based compliant control with variable coordination centralization on a snake robot," in *CDC 2016*, December 2016.

[3] M. Travers, C. Gong, and H. Choset, "Shape-constrained whole-body adaptivity," in *International Symposium on Safety, Security, and Rescue Robotics*, 2015.

[4] L. Righetti and A. J. Ijspeert, "Pattern generators with sensory feedback for the control of quadruped locomotion," *IEEE International Conference on Robotics and Automation*, pp. 819–824, 2008.

[5] G. Sartoretti, Y. Shi, W. Paivine, M. Travers, and H. Choset, "Distributed learning for the decentralized control of articulated mobile robots," in *Submitted to 2017 International Conference on Robot Learning, CORL 2017, Mountain Views, California, USA*, 2017.

[6] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International Conference on Machine Learning*, 2016, pp. 1928–1937.

[7] X. B. Peng, G. Berseth, and M. V. D. Panne, "Terrain-Adaptive Locomotion Skills Using Deep Reinforcement Learning," vol. 35, no. 4, pp. 1–12, 2016.

[8] R. Yamashina, M. Kuroda, and T. Yabuta, "Caterpillar robot locomotion based on Q-Learning using objective/subjective reward," *2011 IEEE/SICE International Symposium on System Integration, SII 2011*, pp. 1311–1316, 2011.

[9] K. Ito and F. Matsuno, "A Study of Reinforcement Learning for the Robot with Many," no. May, pp. 3392–3397, 2002.

[10] M. Babaeizadeh, I. Frosio, S. Tyree, J. Clemons, and J. Kautz, "Reinforcement learning through asynchronous advantage actor-critic on a gpu," 2016.

[11] S. Hirose, *Biologically Inspired Robots: Serpentile Locomotors and Manipulators*. Oxford University Press, 1993.
[12] D. Rollinson and H. Choset, "Gait-based compliant control for snake robots," in *IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 5138–5143.
[13] C. Ott, R. Mukherjee, and Y. Nakamura, "Unified impedance and admittance control," in *IEEE International Conference on Robotics and Automation*, 2010, pp. 554–561.
[14] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980
[15] T. de Bruin, J. Kober, K. Tuyls, and R. Babuska, "The importance of experience replay database composition in deep reinforcement learning," *Deep Reinforcement*, pp. 1–9, 2015. [Online]. Available: https://pdfs.semanticscholar.org/091d/1712b97c0353b9a79e8563e7f174a8fd5450.pdf{%}5Cnrll.berkeley.edu/deeprlworkshop/papers/database{_}composition.pdf
[16] D. Rollinson, Y. Bilgen, B. Brown, F. Enner, S. Ford, C. Layton, J. Rembisz, M. Schwerin, A. Willig, P. Velagapudi, and H. Choset, "Design and architecture of a series elastic snake robot," in *IEEE International Conference on Intelligent Robots and Systems*, 2014, pp. 4630–4636.

Unsupervised Techniques to Augment Fully Supervised Classification of Parkinson's Tremor

Griffin Tabor Alexander Cebulla Dong Huang

***Abstract*—Training a deep neural network requires incredibly large amounts of labeled data. Performance improvements traditionally require doubling or tripling the amount of samples. Unlabeled data on the other hand is cheap and easy to collect. We propose a novel way of initializing a deep neural network. The key is training the network using unsupervised methods so that every labeled sample you have can be used to fine tune the network instead of getting stuck in local minimums. Our experiments show this technique out performs identically designed fully supervised networks.**

I. INTRODUCTION

A. Background

Parkinsons disease affects 10 million people worldwide. Parkinsons Disease, while incurable, causes symptoms that can be combated by specific medication. However, evaluating and adjusting various treatments can be difficult. Currently patients with Parkinsons disease monitor tremors 2 ways. The first is they visit a doctor every 3-6 months and get a UPDRS score after going through a series of exercises to try to induce tremor. The second is the patient describes and changes in their condition. Both of these have obvious downsides. Doctors are trained and experienced but dont have appointments frequent enough to actually understand how severe a patient's day-to-day symptoms are. Patients on the other hand dont have the same expertise that medical professionals do and inherently experience changes so slowly they dont notice or remember much of interest. We present a simple wearable device that can be worn on a patient's wrist to track tremors long term. This sensor uses 3 DOF accelerometers to collect movement data. Our wearable sensor is designed to be non-intrusive and objective. The final goal would be to produce a report detailing a patients symptoms over several months. Doctors would then be able to use this report in adjusting medication to better treat the symptoms.

B. Challenges

One of the main issues is that there are several types of PD tremor and not every patient will experience every type. Fundamentally it comes down to trying to detect specific subtle differences in accelerometer readings without knowing what a patient is doing.

C. Related Work

Several studies have been conducted on automatically measuring motor symptoms of PD. One of the earliest works used what was effectively a stylus and tablet to record pen movements. They found patients with PD traditionally had fluctuations between 4 and 6 Hz.[1] Doctors use the Unified Parkinsons Disease Rating Scale to quantify the severity of different types of tremors. The authors of [1] attempted to learn these precise scores using an array of accelerometers. They extracted features from the sensors in the time and frequency domains to train an SVM.

Other papers looked into adding more complicated tasks to their study. [2] and [3] added daily living tasks to the more traditional UPDRS motor tasks. Participants wore accelerometers and went through the UPDRS tasks, drank from a cup and laid down. However they hard coded the type of actions and run separate algorithms for detecting tremor in each action. Deciding which action is happening with a rudimentary decision tree, no arm movement meant to check for resting tremor for example. The second paper the order was random as was the duration. This paper ran two differently trained hidden markof models in parallel and selected the score with the highest certainty. They both relied on different models for detecting kinetic tremor and resting tremor.

D. Solution

We propose training a single deep convolutional classifier to detect all types of PD tremor. The classifier will be initialized using the weights of a autoencoder pretrained on months worth of unlabeled data. This initialization should allow for a better general classifier. Brief overview of the pipeline can be seen in Figure 1.

II. DATA COLLECTION

For collecting data we had 6 participants come into the lab and go through the UPDRS exercises, while wearing our sensors. The participants were monitored by 3 cameras to ensure their hands were always in view. After completing the exercises we also had them do a series of simulated living tasks, making a sandwich, typing on a keyboard etc. Using the camera feeds we were able to label the raw accelerometer data with binary labels (tremor/no tremor). This will be referred to as the lab data throughout the paper. The different patients lab data is broken down in Table 1.

We then had 3 of these patients take the sensors home and wear them for a month. Using a phone app every hour patients were asked the severity of the tremor during the previous 5 minutes. Their response is matched up with the timestamps recorded by the wearable sensor and is used to label the data. In other words this portion of the dataset consists of 1 hour of fully unlabeled data for every 5 minutes of weakly labeled data. This will be referred to as the home data throughout the

*Funded by the National Science Foundation

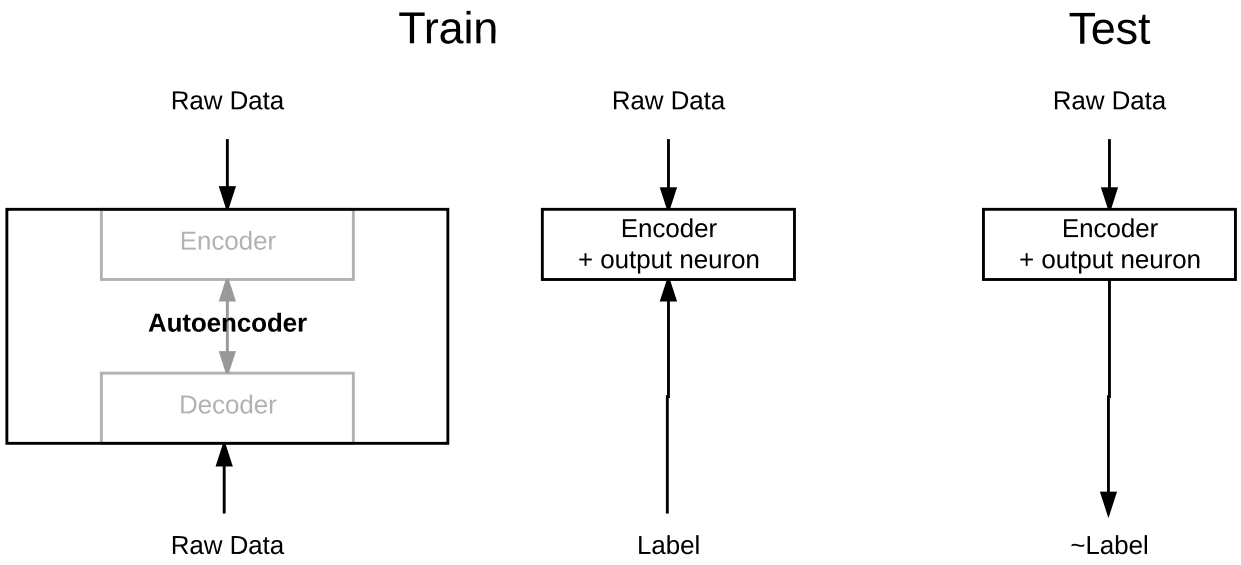


Fig. 1: Pipeline

paper.A breakdown of the home data collected can be seen in table 2.

For the purpose of this paper we will ignore the weakly labeled portions and use the home data exclusively as unlabeled data. The timestamps of the first labeled portion and the last labeled portion of every day were used to extract only the portions of the day subjects were awake to lower computation time. The wearable sensors sampled the 3 axis accelerometers at 50HZ. The original signal was fed through a high pass filter and then broken into 3 second windows with 50 percent overlap. So a single instance is 150X3. We had a medical expert go over the recordings and labeled the UPDRS tasks and provided scores for each patient. Their scores, shown in table 3, show how different patients can be and how different hands can tremor differently on the same patient. Important details to pick out:

- Patients 1 and 3 have primarily tremor while at rest
- Patient 4 only has tremor when not at rest.
- Patient 5 LH tremor is barely noticeable
- Patient 6 has little to no tremor in this portion of the experiment

These scores line up with the labels we assigned. Patient 5's left hand is labeled as only tremoring 1% of the time. Our labels are over 50 % tremor for the first three patients and as little as 15 % tremor for patients 5 and 6.

III. Methodology

The biggest struggle with training neural networks is getting better results without collecting more labeled data. The benefits of course are that a neural network can learn complex non linear representations that other techniques cant. Fully supervised networks traditionally require hundreds of thousands of samples to properly learn how to predict on new unique data.

TABLE I: Lab Data Breakdown

Participant	LH %Tremor	RH %Tremor	Time (min)
1	85.0	43.2	74.9
2	75.0	71.8	55.8
3	61.0	42.2	55.2
4	36.5	41.4	88.1
5	1.1	34.4	91.3
6	21.4	6.7	96.1

TABLE II: Home Data Breakdown

Participant	Time Awake	Time user labeled
1	411h 33 minutes	28h 35 minutes
3	377h 43 minutes	37h 5 minutes
4	456h 31 minutes	14h 25 minutes

Unsupervised networks can be used to find clusters or patterns in data. There is no guarantee however that the patterns found have statistical correlation to the actual label. We used a type of unsupervised technique known as a convolutional autoencoder to encode the full time series signal down to 32 dimensions and then try to reproduce the original signal. In theory this type of network has to learn what is the minimum information required to describe/reproduce the original signal. Our original tests looked into taking this lower dimensionality set as a list of features for training a radial based SVM. To avoid overfitting the autoencoder was trained using only the in home data and the SVM was trained using the lab data with LOSO cross validation. For comparison we trained the same structured neural network,with an additional output neuron, as a fully supervised classifier also using LOSO. With these two techniques as a baseline we looked into how we could give the fully supervised network a better general understanding of the data. Up until this point all the networks were initialized

TABLE III: UPDRS Scores
Scale: 0: Normal; 1: Slight; 2: Mild; 3: Moderate; 4: Severe.

	P1		P2		P3		P4		P5		P6	
	LH	RH	LH	RH	LH	RH	LH	RH	LH	RH	LH	RH
Resting tremor	2	2	1	2	3	3	0	0	1	1	0	0
Postural tremor	2	1	1	2	2	3	2	2	0	1	0	0
Kinetic tremor	1	1	1	2	1	1	2	2	1	1	0	0

with random weights. Our new approach was to initialize the classifier with the values from the autoencoder.We took the autoencoder that had the best results trained on home data and used it to initialize the classifier which was then trained using LOSO cross validation.

IV. Experiments

A. Results

The goal of our technique is to improve on the performance offered by a randomly initialized classifier. We used both accuracy scores and F1 scores when evaluating the success of each experiment.

All neural networks were trained with randomly shuffled data.Training the autoencoder was done with the awake data from all 3 of participants in the at home study. The SVM was trained using 5 patients and predicted on the one left out for each participant. We trained the deep classifiers with the same structure on 5 patients and tested on the last participant, this was repeated for a total of 5 times to find the network with the best results on that patient. Then repeated for each patient. The procedure was the same for both the randomly initialized and the autoencoder initialized. While this approach does skew the results it allowed all the different approaches to be evaluated on an even playing field.

Figure 2 shows the results of our method compared to our two initial experiments and a dummy classifier randomly guessing tremor/no tremor. You can see that none of our approaches had as much success identifying tremor on patients 5 and 6 as on the others. These lower intensity tremors are harder to detect without allowing for false positives. A single network to predict multiple types of tremor shows good result.

B. Conclusion

We found 3 noteworthy conclusions. First, an autoencoder being given access to enough unlabeled data can approach the performance of a identically designed classifier in selecting what information is important. Second, pretraining a classifier on unlabeled data as an autoencoder is a cheap way to get noticeably improved results. That patients that were hardest to detect tremor on were the ones that had the biggest improvement when using the autoencoder initialized classifier. Third, a single technique can classify multiple types of tremor.

References

[1] S. L. Smith and K. Shannon, Vector-based analysis of motor activities in patients with parkinsons disease, in EUROMICRO 97.New Frontiers of Information Technology. Short Contributions., Proceedings of the 23rd Euromicro Conference, IEEE, 1997, pp. 5055.

[2] S. Patel, K. Lorincz, R. Hughes, N. Huggins, J. Growdon, D. Standaert, M. Akay, J. Dy, M. Welsh, and P. Bonato, Monitoring motor fluctuations in patients with parkinsons disease using wearable sensors, IEEE transactions oninformation technology in biomedicine, vol. 13, no. 6, pp. 864873, 2009.

[3] L. Shargel, A. H. Mutnick, P. F. Souney, and L. N. Swanson, Comprehensive Pharmacy Review, Seventh. Lippincott Williams & Wilkins, 2008, ch. 46, ISBN : 978-1582557120.

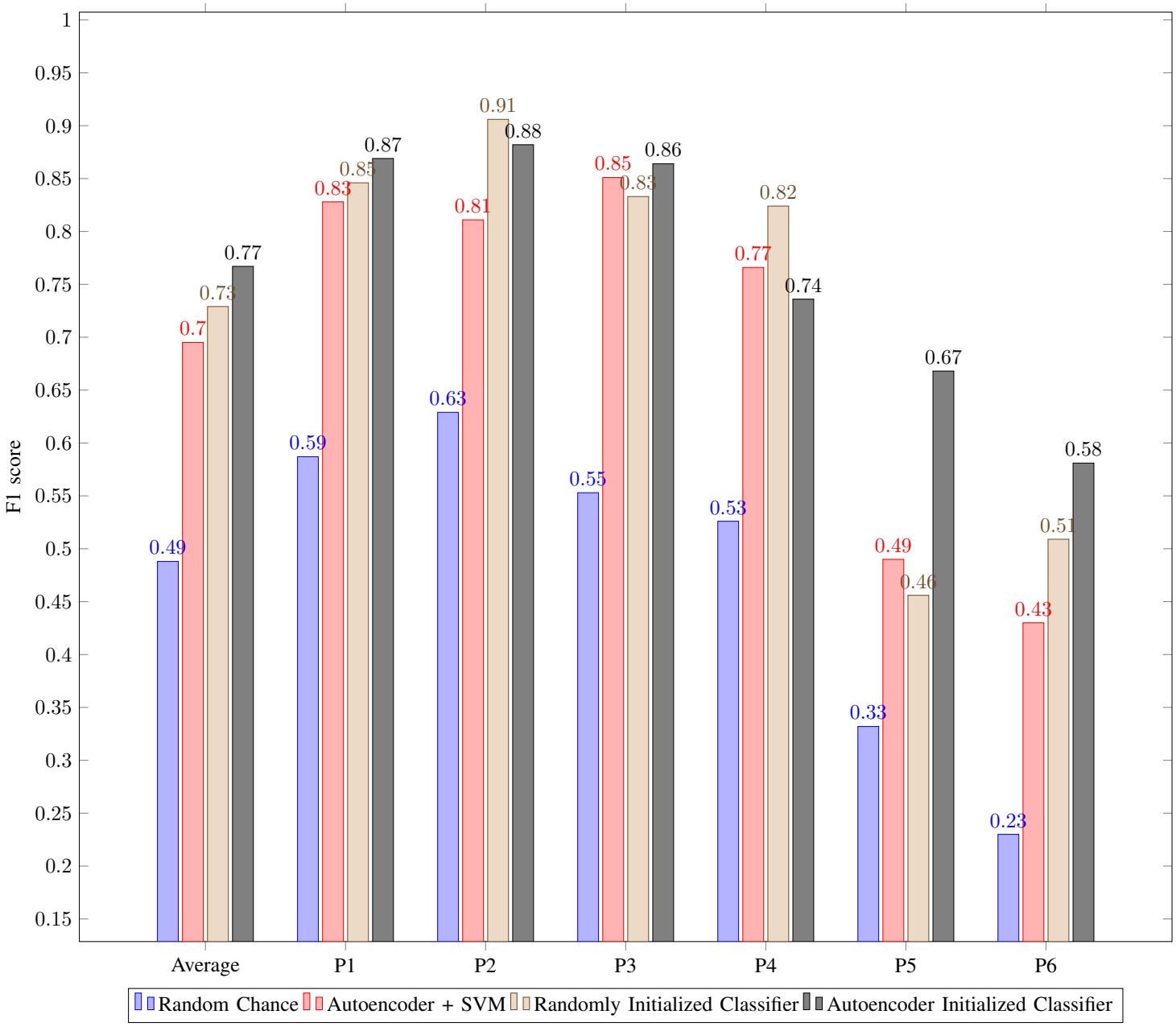


Fig. 2: Results

Active Occlusion Handling in Vision-Based Yield Estimation Systems

Tevon Walker, Carnegie Mellon University

Abstract— **In this paper, we present a novel approach to occlusion handling in computer vision-based crop yield estimation systems. Yield estimation is an essential part of crop management; however, the traditional method of estimation, manual inspection of small sample sets, is quite tedious and is not highly accurate. These characteristics make this method non-ideal; ergo, an automated and more practical means of yield estimation is needed. There has been much exploration in the utilization of artificial vision to produce high-resolution crop yield estimates. High-performance cameras can be used to capture images of crops, and then software can analyze these images to quantify the crop of interest. These systems produce consistent and accurate counts, but a major problem that plagues them is occlusion. Occlusions, in agriculture, are foliage or crops that obstruct the view of the crop. Because there is an object blocking the camera's view, the system misses the crop and the accuracy of the yield estimation suffers. We have done experimentation using a leaf blower and an image stitching algorithm that improves crop yield estimation despite heavily-occluded wine grapes, making our vision system more performant.**

***Index Terms*—Agricultural Robotics, Computer Vision, Occlusion Handling, Yield Estimation, Precision Viticulture**

I. Introduction

The estimation of crop yield is an essential task in crop management. When managers have information about their yields, they can make informed decisions about the operation of their farm. Having an insight to the size of the yield allows managers to improve the quality of their crop, predict the labor force needed for harvesting the crop, and increase overall operational efficiency of their farm. The traditional approach for performing yield estimation, however, is not optimal. This method entails using historical data, weather analytics, and manual hand-counting to generate yield information. This practice is very laborious, time-inefficient, destructive, and does not typically fruit highly accurate information [1]. Additionally, the number of semi-skilled laborers available to perform this imprecise estimation is declining. Approximately 50% of the migrant workers in the United States come from Mexico, and due to an increase of border security and stricter screening processes, the amount of potential farm workers has decreased, threatening the labor workforce [2]. Contrastingly, while the number of people available to work on farms has shrunken, the market size and demand for food has grown [3]. Due to these factors, a completely new technique is needed in order to sustain crop production for an increasing demand.

Computer vision systems have the potential to revolutionize agriculture. Artificial vision based robotics have facilitated the automation of crop yield estimation, fruit picking, crop disease detection, and many more crucial farm operations [4,5,6]. Intelligent machines can eliminate the tedious, impractical processes of human inspection, manipulation, and harvest. These machines are not perfect, however. A major problem that most computer vision systems suffer from, especially those in the context of agriculture, is that of occlusion. Occlusions degrade the performance of computer vision systems by obstructing the view of the object of interest. There have been many cases where people exploiting vision systems had their fruit yield estimates skewed by occlusions. Despite the crops in these experiments being different, the underlying issue is the same: the leaves and fruits cause the vision systems to miss the fruits, thereby tainting the fruit count. We present an original and unexplored modus of occlusion handling for wine grape vineyards.

(a) A section of a concord grape vineyard in Portland, NY. Note that not many berries are visible

Tevon Walker is an undergraduate student in Electrical and Computer Engineering at the University of Alabama, Huntsville. Huntsville, Alabama, 35899 USA (tjw0018@uah.edu). He worked as a research intern at the Robotics Institute at Carnegie Mellon University. Pittsburgh, Pennsylvania, 15213 USA.

(b) A closer look at the Portland vineyard. After some manipulation of the vine, berries become visible

Fig. 1. A key example showing the natural structure of a grape vine occluding the view of berries

II. Related Work

A. Computer vision in agriculture

Vision-based yield estimation has been used in many studies on a variety of crops. The work done proves the potential of cameras automating yield estimation in agriculture. Even though there are various works in vision-based yield estimations, occlusions cause the fruit counts to be less accurate. Dorj et al deploy a vision system to count the number of tangerines on fruit trees with RGB images; however, when comparing the estimate to the ground truth, the estimate is only 59% of the actual number of fruit [7]. Dorj et al attribute this error to occlusions [7]. Roy and Isler and Wang et al develop and experiment with automated apple counters, and both works state that a difficult challenge in these vision systems is dealing with occlusion [8,9]. Nuske et al perform yield estimations with artificial vision systems to quantify grapes and state that they cannot count all grapes due to occlusion [4]. One vision system uses sensor fusion to combine thermal images with visible-light images using fuzzy-logic [10]. Despite the utilization of multispectral cameras, the occluded oranges are still missed, and this method does not appropriately defeat occlusion [10].

B. Occlusion handling

To approach the occlusion problem in yield estimation, we began by researching occlusion handling in general. We found that these generalized solutions for occlusion handling were inadequate. What makes these approaches inappropriate for our application is the fact that they are all passive techniques that require some portion of the object to be visible. The generalized methods propose algorithms that operate on partially occluded objects where only a fraction of the object is visible. Once part of the object is seen, an inference of the remaining part of the object is made to reconstruct the hidden portion of the object. Gao presents a segmentation-aware approach where an object detection system refines its performance by learning and inferring visibility patterns of objects [12]. This approach assumes the object of interest is visible to begin with. Shu et al develops a method of tracking where a part-based model is used to identify partially occluded people [13]. Again, the assumption is made that the object of interest has some portion visible to the camera. General occlusion handling techniques simply cannot mitigate the problem in the context of yield estimation. Typically, fruit in agricultural fields are hidden by partial and complete occlusions, and what is needed is a method of actively handling the occlusions to see more grapes. Kapach suggests an active vision method by changing the point of view of the camera system in order to avoid occlusion [11]. This technique involves additional actuation, planning, and inference of three-dimensional properties of the scene. Furthermore, it does not address the problem of completely occluded fruits; regardless of the point of view, some fruits cannot be seen without manipulation of the leaves.

III. Approach

Our solution presents a means of actively manipulating the occluding structures whether they be partial or complete. Moreover, the cost and complexity of our counter-occlusion solution is relatively low when compared to our overall setup. The main goal of this paper is to explain and highlight our contributions to this occlusion problem:

- Active occlusion handling with a canopy-manipulating method
- Image stitching algorithm for improving view of occluded object(s)

A. Vehicle-mounted imaging system

We use a field utility vehicle to carry our stereo camera rig. The camera rig is mounted to the utility vehicle, and as we drive through the vineyard rows, images are taken of the grape vines. Our camera setup consists of two Point Grey 9.1 megapixel cameras. High-intensity flash lamps fire in sync with stereo cameras to illuminate the scene. A GPS is mounted on the vehicle to register each stereo image with a geographic location within the vineyard. More details of the system are given [14]. Images are captured at 5 Hz while the camera/vehicle rig is driven at about 0.89 m/s. The stereo images are then combined with GPS information and saved onto an external solid state drive. The images are then post-processed to extract grape count, size, color, and location information. The algorithm for the detection of grapes is detailed in Nuske et al [4].

(a) The stereo camera mounted on the back of a field utility vehicle

(b) Sampled image of merlot grapes in Paso Robles, CA

(c) Detection algorithm placing yellow dots on detected berries

Fig. 2. (a) shows the mechanism for grape counting. The vehicle is driven through vineyard rows as images are taken. (b) is a typical image that the system produces. (c) Nuske et al's detection algorithm detecting and marking grape berries [4].

B. Vehicle with leaf blower

To address the problem of occlusions, we attach an off-the-shelf leaf blower to the utility vehicle below the camera. The blower is turned on before beginning the imaging of a row. While the leaves are rapidly shuffling around, the stereo camera system captures images of the scene. Finally, our image stitching algorithm tracks individual regions within the stream of images and determines which regions reveal the highest number of wine grapes and stitches these regions into one composite image. While in the field, we noticed two different types of occlusion structures. One occlusion type was when the leaves lied directly atop the grapes. Leaves on the vine would hang down over the grapes, often causing complete occlusions. The other type of occlusion was when a branch would hang out from the vine into the vineyard row. The branch had leaves that would block the view of the leaves. This occlusion type was also successful in blocking the view of the grapes, and sometimes the entire image would be blocked out by the branches and leaves. Because there were two prevalent structures of occlusion, we needed to attach PVC elbows onto the end of the leaf blower. We used a 45° elbow for occlusion type A and a 90° elbow for occlusion type B.

(a) Occlusion type A

(b) Occlusion type B

Fig. 3. The above images give examples of the two occlusion structures we witnessed in both the New York and California vineyards and the blower configurations used to manipulate the vines.

C. Image stitching algorithm

The stitching algorithm works by creating two two-dimensional grids of cells that lay over each sequential image. The cells in these grids represent a region of pixels, and each cell contains the count of detected grapes in that region of pixels within the image. Grid one, the temporary grid, tracks the berry count of each cell as images come in from the camera. As more grapes appear in the images due to the leaf blower removing occlusions, some of the cells in the temporary grid increase in count value. If a cell has a new maximum for the grape berry count, then it replaces the corresponding cell in grid two, the final grid. As cells are replaced in the final grid, the corresponding regions of pixels

that represent the replaced cell are inserted into the final stitched image.

Algorithm 1 Image Stitching with Maximum Cell Count

```
1: procedure COUNTSTITCH(points,curImg,gridX,gridY)
2:     finalCount ← 0
3:     tempCount ← 0
4:     deltaX ← ImageWidth/gridX
5:     deltaY ← ImageHeight/gridY
6:     for i ← 0; i < deltaX do
7:         for j ← 0; j < deltaY do
8:             for k ← 0; k < points.size() do
9:                 if points[k] isInCell(tempCount[i][j]) then
10:                tempCount[i*(1/deltaX)][j*(1/deltaY)] += 1
11:                end if
12:            end for
13:        end for
14:    end for
15:    for i ← 0; i < gridX do
16:        for j ← 0; j < gridY do
17:            if tempCount[i][j] > finalCount[i][j] then
18:                finalCount[i][j] = tempCount[i][j]
19:                imageSubRegion = curImg.subRegion(i,j)
20:                imageSubRegion.copyTo(finalImg(i,j))
21:            end if
22:        end for
23:    end for
24: end procedure
```

Fig. 4. The (left) image shows a vine with heavy occlusion. The grid cells overlaying the images represent the regions that our algorithm creates for analysis. After blowing the leaves and using the stitching algorithm, a mosaic (right) is created of the selected image regions with the highest berry count. Red circles show where our algorithm reveals more grapes, thus improving the yield estimate

IV. Experimentation and Results

In order to assess the performance of our solution, we noted the number of grapes from each of the scenarios below and observed how using the blower changed the berry counts. We performed these experiments in the Paso Robles merlot and Portland concord vineyards. We wanted consistent imagery while driving the vehicle between the vines. So, instead of imaging an entire row, we chose a subset of vines within a row with significant occlusion and performed imaging.

The method of our experimentation is as follows:

- Perform imaging without blower
- Perform imaging with blower
- Perform imaging with all leaves removed from vine (no blower)

With this design of our experiment, we can gauge how using the blower affects the berry count. After performing image collection with and without the blower, we removed all of the leaves from the vine. After the vine was free of all leaves, we imaged the vine to obtain the true number of grapes that the camera system could capture without occlusion. For the imagery data represented below, we used the blower in the occlusion type A configuration. The following bar graphs show the effect of using the blower:

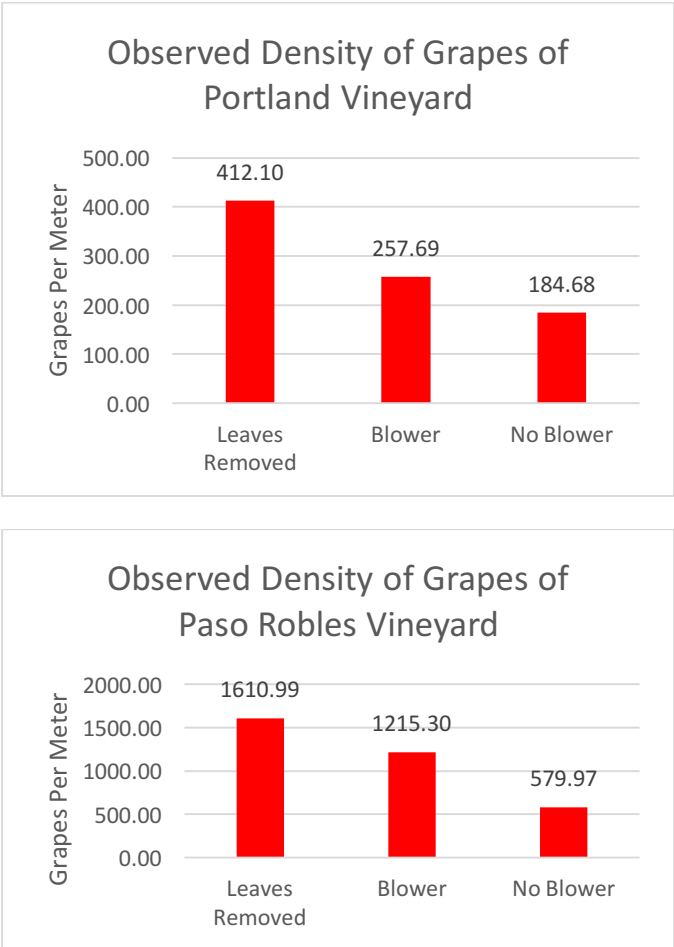


Fig. 5. Two bar graphs representing the effects of using the blower. The leftmost bar is the grapes per meter count when all leaves are removed from the vine. The center bar depicts the grapes per meter when using the blower. The rightmost bar shows the grapes per meter when the blower is not used.

V. Conclusion and Discussion

We have demonstrated a simple, yet effective technique for handling occlusions in vision-based yield estimation systems. The experiments proved the concept of manipulating leaves to create more visibility of grapes. When using our blower technique for counter-occlusion, our method resulted in an increase in the number of grapes detected by 109.8% in the Paso Robles vineyard and 39.5% in the Portland vineyard. This is a low-cost and low-complexity solution when compared to the rest of the equipment for the camera setup. Our technique is also non-destructive; when the blower is in use, even at full speed, the leaves and grapes do not separate from the vine. This means viticulturists would not have to remove nutrient-rich leaves to see more grapes. The stitching algorithm works very well in the case where the camera is not moving. Because the grid cells are static and localize grape count changes in one particular region, the algorithm does not work when the camera is moving. Further work is needed to adapt the algorithm to be able to track berry counts when the grid cells move across each image. Even though the stitching algorithm is currently restricted to imaging with little to no camera movement, the graphs above illustrate how much the count improves just by using the blower alone. A stitching algorithm that is aware of where to track changes as the camera is driven through vineyard rows is a great topic for future work. Additionally, our method should be tested on other fruits to assess the performance across other crop types. Another topic of exploration for this research is alternative manipulation methods. Perhaps differing the blower configurations, blower types, or using multiple blowers would reveal more fruit.

VI. Acknowledgements

I would like to express my appreciation of Dr. Stephen Nuske for the opportunity to contribute to this amazing project. His mentoring and guidance was conducive to me succeeding in this work. Omeed Mirbod and Abhisesh Silwal were instrumental in the data collection for this research, thank you. Finally, I owe a huge *thank you* to Rachel Burcin and CMU RISS for a summer of learning, discovery, and growth.

References

[1] Clingeleffer P., Dunn, G.M., Krstic, M. and Martin, S. Crop development, crop estimation and crop control to secure quality and production of major wine grape varieties: A national approach. *Grape and Wine Research and Development Corporation*, 2001.

[2] Zahniser S., Hertz T., Dixon P., Rimeer M. The Potential Impact of Changes in Immigration Policy on U.S. Agriculture and the Market for Hired Farm Labor: A Simulation Analysis. *ERR-135, U.S. Department of Agriculture, Economic Research Service*, May 2012.

[3] Okrent A.M, Alston J.M. Demand for Food in the United States: A Review of Literature, Evaluation of Previous Estimates, and Presentation of New Estimates of Demand. *Giannini Foundation: Agricultural and Resource Economics*, 2011.

[4] Nuske S., Wilshusen K., Achar S., Yoder L., Narasimhan S., Singh S. Automated Visual Yield Estimation in Vineyards. *Journal of Field Robotics*, Vol. 31, Issue 5, Pages 837–860, 2014.

[5] Davidson J.R., Silwal A., Hohimer C.J., Karkee M., Mo C., Zhang Q. Proof-of-Concept of a Robotic Apple Harvester. *International Conference on Intelligent Robots and Systems*, 2016.

[6] Mahlein AK. Plant Disease Detection by Imagining Sensors – Parallels and Specific Demands for Precision Agriculture and Plant Phenotyping. *The American Phytopathological Society*, Vol. 100, Pages 241-251, 2016.

[7] Dorj U., Lee M., Han S. A Counting Algorithm for Tangerine Yield Estimation. *ISA 2013, ASTL* Vol. 21, Pages *279- 282,* 2013.

[8] Roy P., Isler V. Vision-Based Counting and Yield Estimation. *International Symposium on Experimental Robotics*, 2016.

[9] Wang Q., Nuske S., Bergerman M., Singh S. Automated Crop Yield Estimation for Apple Orchards. *International Symposium on Experimental Robotics*, 2012.

[10] Bulanon D.M., Burks T.F., Alchanatis V. Visible and Thermal Images for Fruit Detection. *Biosystems Engineering,* Vol. 103, Issue 1, Pages 12-22, 2009

[11] Kapach K., Barnea E., Mairon R., Edan Y., Ben-Shahr O. Computer Vision for Fruit Harvesting Robots – State of the Art and Challenges Ahead. *Int. J. Computational Vision and Robotics,* Vol. 3, Issue 1/2 2012.

[12] Gao T., Packer B., Koller D. A Segmentation-aware Object Detection Model with Occlusion Handling. *IEEE Conference on Computer Vision and Pattern Recognition,* Pages 1361-1368, 2011

[13] Shu G., Dehghan A., Oreifej O., Hand E., Shah M. Part-based Multiple-Person Tracking with Partial Occluding Handling. *Computer Vision and Pattern Recognition,* 2012.

[14] Mirbod O., Yoder L., Nuske S., Automated Measurement of Berry Size in Images. *International Federation of Automatic Control*, Vol. 49, Issue 16, Pages 79-84, 2016.

Go, Look, and Tell: Natural Language Communication with a Dynamically Stable Mobile Robot

Matthew Wilson[1], Jean Oh[2], Ralph Hollis[2]

***Abstract*— Robots will become much more useful when they can effectively communicate with humans and better understand the world. If a user can access the growing capabilities of mobile robots through a natural speech interface, it would enable new uses, in more personal and interactive tasks.**

In this paper, we describe initial formulation and results for a natural speech interface on the ballbot, a dynamically stable mobile robot that balances on a single spherical wheel. We develop an end-to-end system for perceiving the world, responding to questions, and receiving natural language commands to navigate in the world.

I. Introduction

Robots will become much more useful when they can effectively communicate with humans and better understand the world. In contrast to the domain specific tasks service robots are limited to now, such as vacuuming, with a speech interface they could receive commands interactively from users and accomplish a wide variety of tasks, issued dynamically. With these advances, service robots could achieve more general purpose use—an analogous step up from a calculator to a general purpose computer.

Few robots today offer an intuitive interface for users to interact with them; completing tasks almost always involves explicit programming. This either limits the benefits of robots to expert users, or else only enables a trivial set of functionality. Today, general utility human-sized mobile service robots have great capability to move about and perceive their world. However, this general utility is often unavailable to users because there are no comprehensive natural interfaces.

A natural option for giving commands and receiving information in a more human-centric way is a speech interface. Using speech has two large benefits:

1) It is intuitive. Untrained users can quickly learn to use unfamiliar systems.
2) It requires little cognitive load, i.e. can be used without much active effort.

Non-expert users could interact with robots to complete tasks and gain world information, without having to be trained and without having to think much about it. These properties could lead to wide-spread and frequent daily use of household service robots.

[1]Matthew Wilson is an undergraduate student at the University of Utah and conducted this research as part of the Robotics Institute Summer Scholars at Carnegie Mellon University, Pittsburgh, PA 15213 USA matthew.b.wilson@utah.edu

[2]Jean Oh and Ralph Hollis are with the Robotics Institute of Carnegie Mellon University, Pittsburgh, PA 15213 USA (jeanoh, rhollis)@cs.cmu.edu

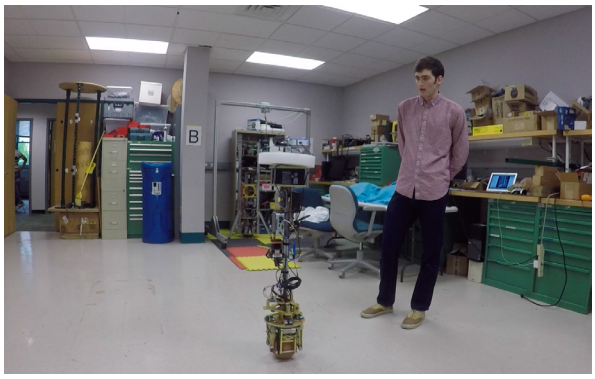


Fig. 1: Giving a speech command to the ballbot

This paper focuses on the system integration of a speech interface, mobility, and world perception, on a dynamically stable mobile robot to allow natural language commands and communication of world observations. The system we develop enables the user to interact with the robot in a natural way to accomplish tasks and gain world information. This is meant to serve as an initial formulation of a more extended system, that with probabilistic modeling and more machine learned components, could be used to generate a widely useful, general purpose service robot.

II. Platform

Critical in a robot that communicates and remains in close contact to humans is ability to move safely in human environments. The system this paper describes is developed on the ballbot, a omnidirectionally compliant human-sized robot.

A. Background on the ballbot

The ballbot, introduced in [1], is a robot that balances on a ball driven by two sets of parallel rollers. The drive system is termed Inverse Mouseball (IMB) drive, as inversely to old computer mouses which used a free-rolling mouseball to move roller encoders and determine mouse displacement, the IMB uses four rollers to actuate the ball and keep the ballbot balanced. During operation, the ballbot uses an Inertial Measurement Unit (IMU) to sense the lean angle and motors to drive the rollers and actively keep its center of gravity over the point of contact with the ground.

The original ballbot is an omnidirectional robot, equipped with arms, and with dimensions approximating that of humans (see Table I). The ballbot used in this paper, called Shmoobot, is a slightly smaller version of the original ballbot and does not have any manipulators.

TABLE I: Comparison of human, original ballbot, and Shmoobot dimensions

	Human (average male)	Original ballbot	Shmoobot (mini ballbot used in this paper)
Height (m)	1.7	1.7	1.2
Shoulder width /diameter (m)	0.45	0.40	0.28

The core advantages of the ballbot in human spaces are **dynamic stability** and **omnidirectionality**. Dynamic stability refers to the property of ballbot that it must actively work to keep balance, similar to humans which are also considered dynamically stable; in contrast to this, statically stable robots, such as the PR2, rest on a wide base with several supporting wheels. For basic service robotic tasks, that have been the focus of much prior work, dynamic stability adds an unnecessary control challenge. But for interacting with humans in more complex ways, dynamic stability is largely beneficial. It enables direct control over a robot's center of gravity. By being able to shift its weight and by having omnidirectional mobility, the original ballbot, equipped with arms, can do tasks such as:

- **Help people out of chairs [2]**. This is infeasible for most robots with static bases, as they would tip over.
- **Lead someone by hand [3]**. This is possible with other robotcs, but not in a natural way, as they cannot move in any direction or react to large forces.

For the components this paper is specifically interested in—of purely mobility, perception, and speech interface—the advantages are:

- **Safety.** While it is moving through an environment, ballbot can easily be pushed away with only a finger [1]. Other robots are susceptible to tipping or else can hurt a person by running into them.
- **Moving in most [2] spaces a humans can**. The ballbot has a zero turn radius; it can rotate in place and navigate through dense environments [4], [5].
- **Small base and large height.** It can move in tight, cluttered, human spaces and yet can interact at human height. Other robots would be susceptible to tipping.
- **Moving in crowds of people.** Some work has been done in [3]. Other robots cannot move as compliantly through an area, and if they are stationary, become a static obstacle.

Ballbots offer many of the advantages that a fully humanoid robot would, and they are reliable. They are a versatile platform for physical Human Robot Interaction (pHRI) and play a key component in the formulation of this system.

III. Related work

There have been many progressions in interpreting natural language in use for commanding robots. Previous work in controlling robots by natural language processing has been done on stationary manipulator robots [6], and on mobile robots such as assistive wheel chairs [7] and military vehicles [8]. Work has also been done on more interactive and robust interactions, such as enabling the robot to ask for clarification of commands [9].

There has less focus, on the sub-field of general utility mobile robots for domestic or service industry use. The technology is still in many ways far from developing a fully capable domestic robot. One recent paper [10] demonstrates the use of a PR2 to do various tasks that are specified by the user. The robot can achieve quite a lot from voice control, but still relies on a controlled and specific environment and does not generalize well to new spaces and objects that it has not seen yet.

The two major ways in which this paper differ from other works are that it is an interactive household setting featuring mobility to other locations and it is done on the dynamically stable ballbot.

[1] https://youtu.be/8BtDuzu2WeI?t=1m55s

[2] The ballbot can handle slopes and bumpy terrain, but is currently limited to approximately level spaces (no stairs, etc.).

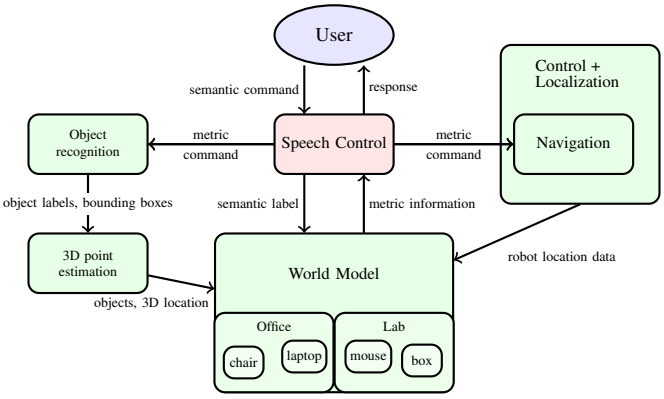


Fig. 2: Interaction of system components

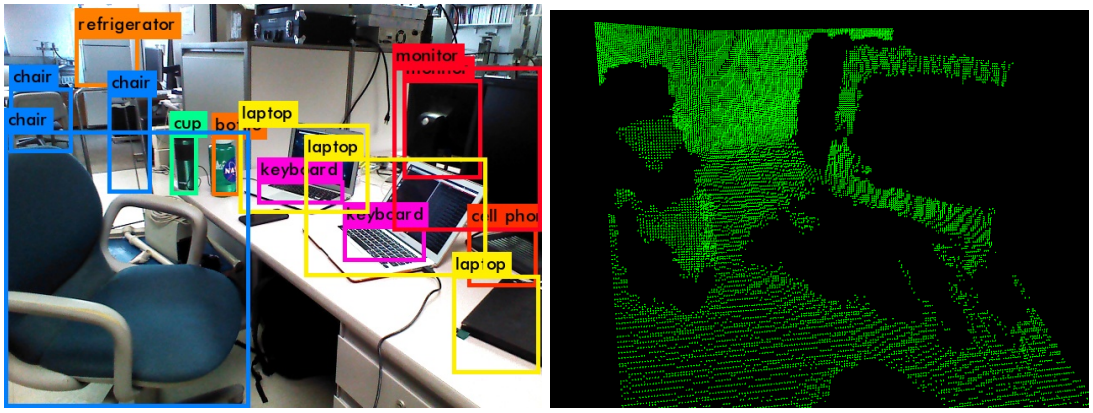
IV. Approach

To develop this system, we integrated and developed interfaces for perception and navigation components, using a world model to maintain relationships of metric and semantic labels, so that they could be accessed by a central natural speech interface. The structure of interaction can be seen in Figure 2. The user interacts directly with the speech interface, using natural language, with semantic commands and queries such as "go the office". The command verbs (ex. go) get mapped to a discrete set of robot actions in the speech control, and the semantic information (ex. office) in these interactions is resolved into metric information by the world model (ex. coords=(3,3)). Thus, the user can give natural commands that get executed with respect to discrete objects in the world.

The details of the components are discussed below.

A. Natural language processing

The speech interface receives processed text from an on-board Amazon Echo Dot. Our system maps this processed text to a set of **actions** or **queries**, along with **semantic**



(a) Labels and bounding boxes of detected objects. Each bounding box can be used to segment the point cloud data and calculate the centroid of the objects

(b) Point cloud segmentation of "laptop" bounding box. The method incorrectly includes point data from the water bottle to calculate the centroid of the laptop.

Fig. 3: Object detection and point cloud segmentation

parameters. Actions are things that the robot can do—specifically: move and observe. Queries are questions that the robot can respond to. Semantic parameters are natural labels for concepts in the world and are resolved by the world model into metric data that the speech control can use to execute actions to detect objects, move in the environment, or respond to user queries.

B. Perception

To provide the user with relevant information about the environment, the system detects objects and generates estimates of 3D location of objects for the user. The system uses the YOLOv2 object detector [11] to detect a discrete set of the objects from the MSCOCO dataset [12]. The output of the detector is a set of labels, associated probabilities of the labels, and bounding boxes in a 2D image. The bounding boxes are then used to segment point cloud data from the RGB-D camera to obtain the matching points for each object. The corresponding points from the bounding box are used to calculate an estimated centroid for each object. This allows the object detections to be grounded in global 3D space. The advantage of this approach over those used in the past, is that it does not require full 3D models of all objects. The disadvantage, illustrated in Figure 3b, is that the segmentation based on bounding boxes is imperfect and points from other objects or the background can be included in centroid estimation.

C. Navigation

To move in the world, this work interfaces with the ballbot navigation stack, developed in [4], [5], by providing world coordinates to generate a global path plan from the ballbot's current location.

D. World model

For the robot to be able to reason and create more intelligent relations of the world, we developed a hierarchical world model, including **locations**, **surfaces**, and **objects**. For example, objects such as a laptop would be on the surface of the table, inside the location of the office. This is a natural representation for humans and supports easy commanding and querying. A user can tell the robot to go to lab and look what is on a table in that location.

The representations of these objects, namely the semantic label and information associated with underlying metric data, are saved in a database structure. This representation allows querying by either the metric data (such as location, or radius of locations) or by semantic label (such as bottle, office, etc).

The world model is key to integrating the perception and speech components to the control of the robot system. It resolves the semantic labels in the speech interface to metric data that can be used to command the robot.

V. Experiment

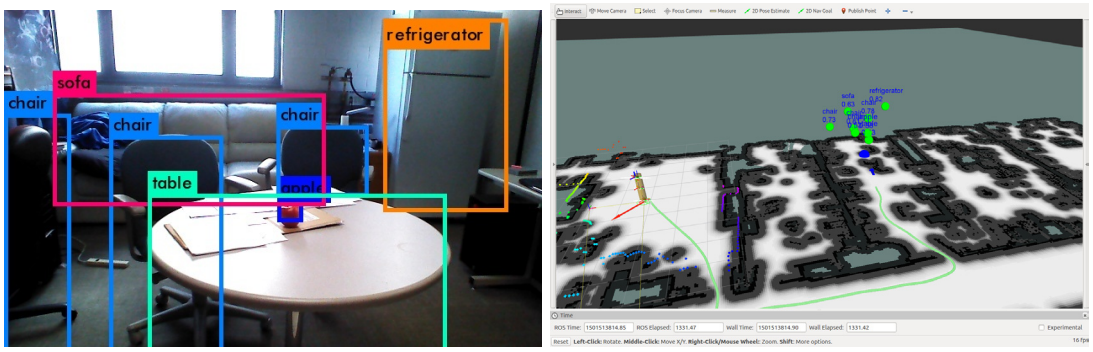
We develop an indoor service scenario as a proof of concept demo and showcase of the system. This scenario consists of a user commanding the robot, in spoken English, to go to another room, look what is on a table there, and come back to tell the user what it saw. The flow of each of these commands through the system is described below.

A. "Go to the office"

The first command is spoken to the robot and enters the system through the speech interface. The speech interface maps the phrase "go to", to the navigation action. It then passes the semantic parameter "office" to the world model to resolve into metric coordinates. Because this may be a long running goal, it is added to a command queue and further commands can be given to the robot to execute. However, if no "office" is found, the system will give immediate feedback to the user, by speaking, "I don't know where the office is".

B. "Look what is on the table"

The second command follows a similar flow. The phrase "look what" is mapped to an observe action. It then takes the phrase "on the table" as a semantic parameter which similarly



(a) Detected objects from YOLO object detector [11]. (b) 3D point estimates for objects, obtained from point cloud data in bounding box regions

Fig. 4: Detected objects grounded in 3D world and placed in world model

gets resolved by the world model to metric coordinates. With these coordinates, the robot turns its body in that direction and takes a picture to detect objects on the table and place them into the world model. Similarly, it will give immediate feedback if no surface, such as table, is known.

C. *"Come back to the lab. That's all."*

Just as in the first command, this is interpreted as a navigate action, and the semantic parameter "lab" is resolved into the coordinates of the lab in the world model—in this case, the starting place of the robot.

"That's all" is one of the phrases to command the ballbot to execute a series of queued commands. Upon hearing this command, the robot will ask for confirmation by speaking back the queued commands. If the user confirms the actions, the robot will then execute them.

D. *"What have you seen?"*

Finally, when the robot returns to the lab, the user can ask what the robot has observed. The phrase "What have you seen" is mapped to a query with no parameters. This pulls from the world model all the objects that have been detected. These are spoken to the user (in a phrase something like "I have seen 2 monitors, a laptop, etc"), and images of objects and their corresponding 3D locations on a map can be displayed on an offline computer screen. The query can also take semantic parameters, allowing the user to search by object label or room. For example, "Did you see a laptop?" or "Have you seen a laptop?" will cause the robot to respond with, "I have seen 1 laptop", and show an image and map location of the detected laptop.

VI. Conclusion

We have developed an end-to-end system for a dynamically stable mobile robot to receive natural language commands and communicate world observations. This system, unlike previous work, is done in a household environment with a dynamically stable mobile robot, allowing adaptive and safe interactions with humans.

This is an initial formulation of the system, and with the framework outlined, there are many potential extensions of this work.

VII. Future Work

While the demonstrations were simple, in that the surfaces and locations were pre-loaded into the world model, the system could allow more intelligent sources of input. Further methods for determining room location and location of table as well as probabilistic representations and methods for interpreting speech commands could be implemented, as in [6], [13], [14], [15], [16].

A. *Language grounding and spatial reasoning*

The system described in this work is limited to discrete, deterministic mappings between symbols in commands and objects in the real world. With a way to handle ambiguities arising from multiple objects of the same label and of representing the spatial relation of objects in the world, a more natural interface could be developed. The user could specify actions with phrases, such as "the table to the right of the door", using learned spatial relations as in [8].

More intricate label grounding could also be used in navigation commands. With probabilistic grounding of symbols and learned spatial relations applied to a costmap as in [8], the robot could be commanded "go to the left of the person, to the lab".

B. *Multi-hypothesis probabilistic world model*

Another important extension to improve the robustness and adaptiveness of this system, is in probabilistic world modeling. The object detector and location pipeline produced some uncertainties in both the labels and positions of objects which were largely ignored in this system. This data, could be used to maintain a multi-hypothesis state of the system as in [17]. This would be more adaptive to movement of objects and time variance. Furthermore, it would be necessary in integrating continuous real-time object data.

Acknowledgments

This work was supported by an NSF REU award and by NSF grant IIS-1547143. Thank you to Michael Shomin and Roberto Shu for their work on the ballbots and help in conducting experiments.

References

[1] T. Lauwers, G. Kantor, and R. Hollis, "A dynamically stable single-wheeled mobile robot with inverse mouse-ball drive," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2006.

[2] M. Shomin, J. Forlizzi, and R. Hollis, "Sit-to-stand assistance with a balancing mobile robot," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[3] M. Shomin, "Navigation and physical interaction with balancing robots," Ph.D. dissertation, Carnegie Mellon University.

[4] M. Shomin and R. Hollis, "Differentially flat trajectory generation for a dynamically stable mobile robot," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[5] ——, "Fast, dynamic trajectory planning for a dynamically stable mobile robot," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014.

[6] R. Paul, J. Arkin, N. Roy, and T. Howard, "Efficient grounding of abstract spatial concepts for natural language interaction with robot manipulators," in *Proc. Robotics: Science and Systems*, 2016.

[7] M. Walter, S. Hemachandra, B. Hamberg, S. Tellex, and S. Teller, "A framework for learning semantic maps from grounded natural language descriptions," *International Journal of Robotics Research*, vol. 31, 2004.

[8] J. Oh, A. Suppe, F. Duvallet, A. Boularias, J. Vinokurov, L. Navarro-Serment, O. Romero, R. Dean, C. Lebiere, M. Hebert, and A. Stentz, "Toward Mobile Robots Reasoning Like Humans," in *Proc. AAAI Conf. on Artificial Intelligence*, 2015, pp. 1371–1379.

[9] S. Tellex, P. Thaker, R. Deits, D. Simeonov, T. Kollar, and N. Roy, "Toward information theoretic human-robot dialog," in *Proc. Robotics: Science and Systems (RSS)*, 2014.

[10] D. Misra, J. Sung, K. Lee, and A. Saxena, "Tell me dave: Context-sensitive grounding of natural language to manipulation instructions," in *Proc. Robotics: Science and Systems (RSS)*, 2014.

[11] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," *arXiv preprint arXiv:1612.08242*, 2016.

[12] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollr, "Microsoft coco: Common objects in context," 2014.

[13] T. Kollar, S. Telle1, M. R. Walter, A. Huang, A. Bachrach, S. Hemachandra, E. Brunskill, A. Banerjee, D. Roy, S. Teller, and N. Roy, "Generalized grounding graphs: A probabilistic framework for understanding grounded language," *Artificial Intelligence Research*, 2013.

[14] T. M. Howard, S. Tellex, and N. Roy, "A natural language planner interface for mobile manipulators," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[15] I. Chung, O. Propp, M. R. Walter, and T. M. Howard, "On the performance of hierarchical distributed correspondence graphs for efficient symbol grounding of robot instructions," in *Proc. IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[16] A. Boteanu, T. Howard, J. Arkin, and H. Kress-Gazit, "A model for verifiable grounding and execution of complex natural language instructions," in *Proc. IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[17] J. Elfring, S. van den Dries, R. van de Molengraft, and M. Steinbuch, "A model for verifiable grounding and execution of complex natural language instructions," in *Proc. IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[18] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS: an open-source Robot Operating System," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA) Workshop on Open Source Robotics*, Kobe, Japan, May 2009.

Appendix

A. Technical implementation

The ballbot uses the Robot Operating System (ROS)[18]. All message passing and communication between components happens over the ROS network. For the world model, custom ROS messages were defined for the different models, such as a semantic location or semantic object. These messages were stored in a MongoDB database. The speech interface used an Amazon Echo Dot and a custom Alexa skill was used to interact with an http Flask-Ask server running on an on-board system. Most of the code was written in Python.

Optimization of the use of Motion Profiling in the First Robotics Competition with a New Velocity Profile Generator

Chun Ming Jeffrey Zhang

***Abstract*—Motion planning is an important aspect of automation and is defined by delineating a motion task into small, navigable motions. For example, programming a robotic arm that moves on a specific 3-dimensional path would require knowledge of the power output of several motors at each specific point in time. Implementing motion planning has two major steps: 1. Generating the needed profiles of controller outputs as a function of time 2. Setting the controllers to the specific outputs at the correct time with precision and accuracy. This paper will examine the use of motion planning in the First Robotics Competition and will focus on the second portion. Through using a velocity generating tool called the "SmoothPathPlanner" and the newly developed "Motion Profiling" library in the motor control system, we successfully implemented motion planning on a robot. This paper will summarize the procedures for such implementation and analyze the effects of several parameters such as PID values and time steps on the quality of the motion outcome.**

***Index Terms*—Motion Profiling, Motion Planning, Velocity Profile.**

I. Introduction

This paper focuses on the implementation of motion planning in the First Robotics Competition (FRC). Traditionally, motion planning in the FRC is mostly used for smooth straight-line driving, as many teams use a trapezoidal motion profile to speed up and slow down a robot near the endpoints of a path. With the newly introduced "Motion Profile" mode of the motor controller SRX Talon by Cross the Roads Electronics, much flexibility is given to users who want to execute an auto-controlled curve path. In most cases, teams implementing the "Motion Profile" mode use spline interpolation to generate a curve from several waypoints with software such as Excel and Matlab, then calculating the power output (as function of time) of the left and right wheels needed to execute the path. This method is feasible, but require that the user generate the motion profile externally.

A trapezoidal velocity profile with a constant acceleration, constant velocity, and a constant deceleration phase is typically used for point-to-point motion in many robotics applications. The main disadvantage of using a trapezoidal velocity profile is the jerk induced on the vehicle. Hence, numerous research have developed s-shape motion profiles to minimize vibration and jerk [1] [2]. Curved motion profiles build on this foundation and generally uses polynomial fit curves, but there is a lack of general-purpose tools easy for FRC teams to generate motion profiles.

Our work is focused on using a tool called the "SmoothPathPlanner" [3], developed by KHEngineering and available on GitHub, to create a real-time motion planning using a tool that has potential for wide adoption. This paper will present the problems that we encounter, and the optimization of using this tool to implement auto-controlled curved paths. The first part of the paper will detail the procedures to implement this method, and the second part of the paper will present the field experiment results of the effects of several parameters such as PID values and time steps on the quality of the motion outcome.

II. Approach

A. Overview

The tool that we use to generate the velocity profiles is called the "SmoothPathPlanner", developed by KHEngineering and available on GitHub. We first step up our competition robot for motion profile by changing the motor controllers to "motion profile" mode, then creating a control loop for the motor controllers to internally process the velocity profile. To execute the motion profile, we create a command that generates our velocity profile using given waypoints and pushes the profile's sampling points onto the talon. The command also determines whether the velocity profile points have been updated enough, at which point the motors are set to neutral.

To test and calibrate our motion profile command, we first calibrate the robot motors' PID values so that the velocity response is optimized. Another problem needed is to address is the robot's inability to drive straight when given a straight-line motion profile. After calibration, we test the motion profile on the robot with a simple straight-line profile, then find the ratio between the programmed distances of the waypoints and the actual distance run by the robot. This ratio will be used to calibrate the waypoint coordinates in the curved path.

In the end, we will study the effects of the different parameters in the "SmoothPathPlanner" on the quality of the curve path, to accurately recreate the intended drive path.

All code used in this paper is written in Java.

B. *Setting up & calibration*

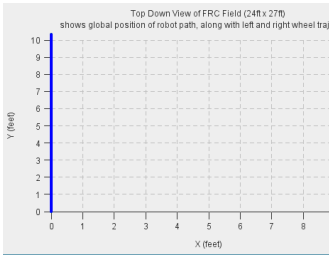
By using instructions provided by Cross the Road Electronics (CTRE), we established the programming framework whereby a generated velocity profile could be loaded onto the motor controller (talon) buffers as points that include the velocity (in RPS), position (in wheel rotations) and duration (in ms) of each sampling point in the velocity profile. We also adopted the motion profile code instructions to fit our "Command-Based" programming framework, which is a method to organize code so that different commands (high-level robotic operations) can run concurrently with different activation mechanisms (such as a click of a joystick button).

There are two calibration steps needed before testing to use motion profile on the robot. The first step is to tune the PID (proportional, integral, derivative) values for smooth velocity transitions. This can be done by changing the motor controller mode to velocity mode, then set the velocity of the motors as a continuous function of time. For an untuned system, the robot will typically exhibit high-frequency jittery motion, as the default kP value is usually too large, which results in proportional oscillation where the motors never settle on the target velocity. With properly tuned PID values, the robot should drive smoothly and react to the change in velocity when given a set velocity, with minimum jittery motion.

The second calibration step is necessary because a typical robot inevitably tends to turn at a wide (or small) angle even when motors on both sides are prompted to run at the same output due to mechanical imperfections. This can be a problem, depending on the magnitude of the turn, as the motion profile-executed path will be swayed in one direction. There are two common ways to address this issue: 1. create a PID control loop so that the rate of rotation (monitored by a gyro) matches the intended turn rate (in the case of driving straight, keep the bearing constant) 2. multiply the motor output of one side by a predetermined constant to minimize the discrepancy. In our case, we choose option two, for two reasons: 1. it is very difficult to modify the velocity values of the velocity profile points once they are sent into the motor controller's internal processor 2. there can be multiple variables besides mechanical imperfection, as we shall see soon, that can lead to distorted paths of the robot, and these variables should be addressed before using PID to correct for the heading.

C. *Generating velocity profile*

Next, we generate our velocity profiles using the "SmoothPathPlanner" tool. For our first experiment, we use a simple straight-line path. To use the tool requires specifying several parameters, including the total time intended to finish the path, the time step (duration) of each sampling point, and the track width (distance between the left and right wheels). The tool can also generate graphs of the simulated trajectory and velocity profiles for left and right wheels.



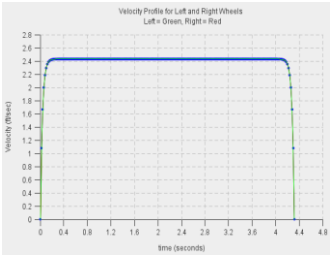


Fig. 1. Simulated path of a simple straight-line run.
Fig. 2. Simulated velocity profile of the same path. Each blue dot represents a sampling point.

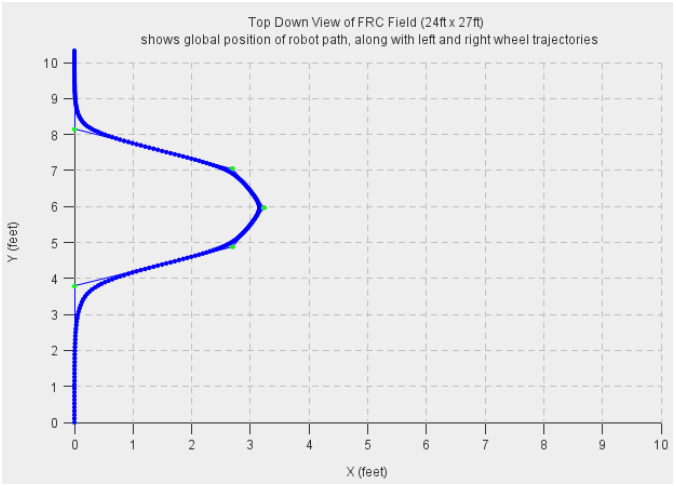


Fig. 3. Simulated path of the robot of an example curved path. Note that the path does not necessarily touch all waypoints for the smoothness of the path. This feature can also be adjusted in the tool.

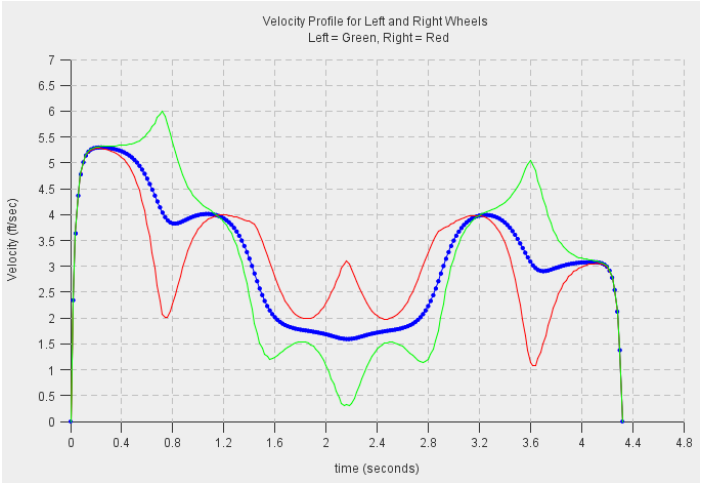


Fig. 4. Simulated velocity profiles of the path in figure 3. Blue line represents the robot-center velocity, while left and right wheel velocities are in green and red. Each blue dot represents a sampling point.

D. *Straight Line Path Testing*

We will first test a straight-line path, which is the simplest path possible, to study the reliability of our implementation as well as the effects of different input parameters, including time step and total time.

We field-test the motion profile under different combination of the input parameters and iterate through different length

paths (by half-foot intervals). By acquiring the actual distance traveled by the robot with a quadrature encoder, we can plot the relationship between the actual distance traveled and the input distances.

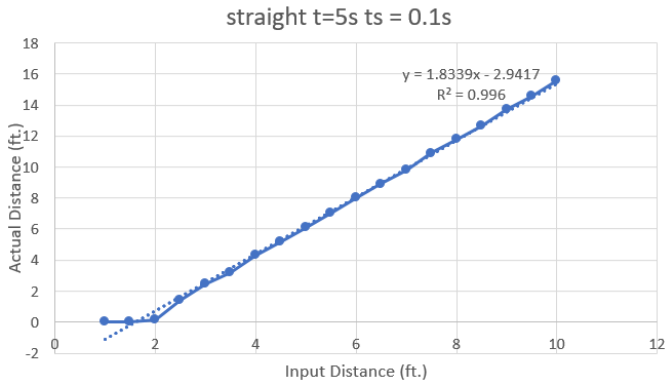


Fig. 5. An example of our measurement result. The plot shows the actual distance traveled versus input distance in feet when total time is 5 seconds and time step is 0.1 second. Inputs less than 2.5 feet don't result in movement as the velocities are too low for the robot to overcome friction.

E. Curved Path Testing

Next, we use a simple curve path to test the robustness of our tool. Our priority is to investigate the effects of the input parameters, now including track width, on the fidelity of the actual path run by the robot. Since there is a consistent scaled difference between the actual distance traveled and the input distance, we need to map the intended coordinates of the waypoints to the input values that would yield those intended distances on the actual path. Therefore, our waypoints are as following,

```
{
{0,0},
{0, calibrate(4.0)},
{calibrate(2.0), calibrate(6.0)},
{calibrate(3.0),calibrate(8.0)},
{calibrate(2.0),calibrate(10.0)},
{0,calibrate(12.0)},
{0,calibrate(16.0)}}
```

whereby calibrate() is the transfer function from real distances to input distances. The position profile and the velocity of this curve path is shown in Figure 3 and 4. We use this particular path for testing because it has a peak-point at calibrate(3.0),calibrate(8.0), which is easily noticeable during field-testing. To gather data about the quality of the curves, we videotape each run and use the video to determine the x and y errors of the actual path at the peak point and the end point.

III. RESULTS

A. Straight-Line Path

To test the total time and time step variables, we run paths in 3s, 5s, and 7s, each tested with time steps of 0.02s, 0.05s, and 0.1s (except for total time=5s, which is tested for more time steps). For each combination of total time and time step, we test paths from 1 feet to 10 feet in half-foot intervals.

The first question that we ask is whether time step and total time effect the ratio between input distance and actual distances. The following graph presents our findings.

Fig. 6. The slope of the input distance-actual distance trend line for each combination of time step and total time. Different lines represent the slope of the distance-response under different total times.

As shown from the graph, there is not enough evidence to suggest a relationship between total time and slope, since other factors such as changing battery power can account for the slight deviations. Nonetheless, it seems that time step and slope are generally negatively related. The outlier, occurring at time step = 0.4s and total time = 5s, cannot be alone explained by anomaly because of the data point's high R^2 value and the fact that battery voltage discrepancies are monitored and kept low.

Between different total times, there are differences between the minimum distance input for the robot to have any movement, and it is expected because a longer total time generally results in a slower velocity profile, which can be too low to overcome static friction when the input distance is low.

All trend lines have a high R^2 value that is greater than 0.987, which suggest high fitness of all the trend lines. There is a negative correlation between total time and the R^2 value, and this can be affirmed by the weak linearity (within the linear portion) of the graph when total time = 10s and time step = 0.02s, which is not shown in Fig. 6.

With enough information on the effects of total time and time step on a straight-line path, we conclude that the transfer function between real distances and input distances are approximate for all three total times with time step lower than or equal to 0.1s.

B. Curved Path

Similarly, we test the effects of total time, time step, and this time with the addition of track width. We tested total times of 5s, 7s and 10s, time steps of 0.01s, 0.02s and 0.05s, and track widths of 2 feet and 1 foot. We measured the x and y error of two points on the actual path with respect to the intended path. The two points that we select are the peak of the curve and the end point, since they are the easiest to recognize.

One pattern that is easily discernable is that the curvature of the path is directly related to the track width. Although the actual track width of our robot is 2 feet, with the track width initially set to 2 feet, the curve portion of the path is accentuated as the robot take much sharper turns than intended, twisting and distorting the general shape of the path,

causing the robot to stop while facing an angle relative to the straight line (and last) portion of the path. By halving the track width, the general shape of the path drastically improved, across different time steps and total times.

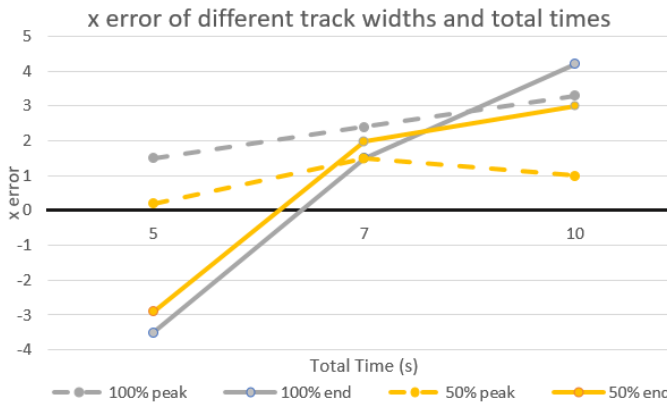


Fig. 7. the x (horizontal) error of the peak point and the end point under different configurations. Horizontal axis represents the total time, and the vertical axis represents the error relative to the correct positions. Dashed lines represent the peak points, and solid lines represent end points. Grey lines are run with 100% track width (2 feet) while yellow lines are of 50% track width (1 foot).

As this graph shows, 50% track width (yellow lines) generally yields less x error, which is a rough indication of the path curvature's fidelity. This improvement is especially apparent at the peak point. This confirms our prediction that the trackwidth influences the intensity of turning, since robots with a larger track width need a greater difference in velocity between left and right wheels to turn with a same angle, compared to another robot with a smaller track width.

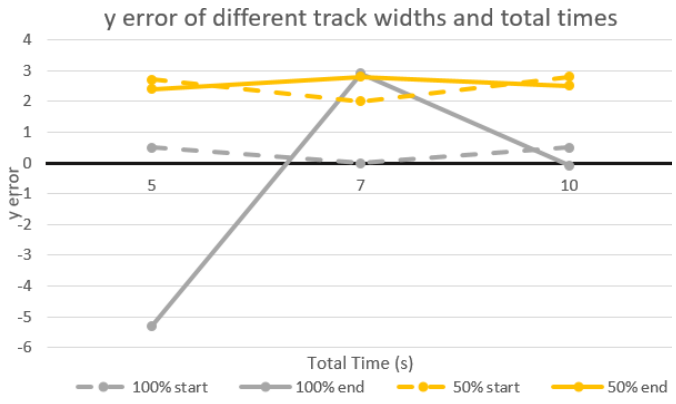


Fig. 8. the y (vertical) error of the peak point and the end point under different configurations. Refer to the previous graph for legend.

This graph can be harder to decipher without the footage of the runs. It can be tempting to conclude from the tow dashed lines that 100% track width yields more accurate y positions. However, the footage shows that it is so only because the turns at 100% track width are sharp, almost at a right-angle, causing the y-position of the peak point to be pushed lower. As consistency as the y error under 50% track width is, we have reason to believe that other factors, possibly the offset value of our transfer function, is causing such discrepancies.

Changes in time steps seem to have unpredictable effects on x and y error alike. To find out why, we can check the velocity profiles generated by the tool:

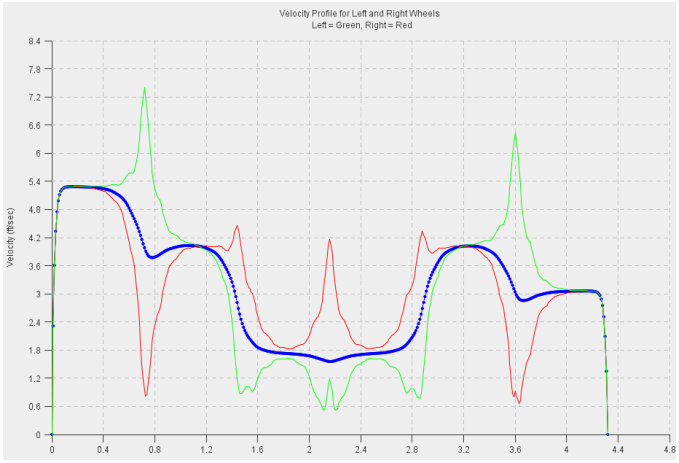


Fig. 9. the velocity profile of left (green) and right (red) wheels at time step = 0.01, total time = 5s, track width = 1 foot.

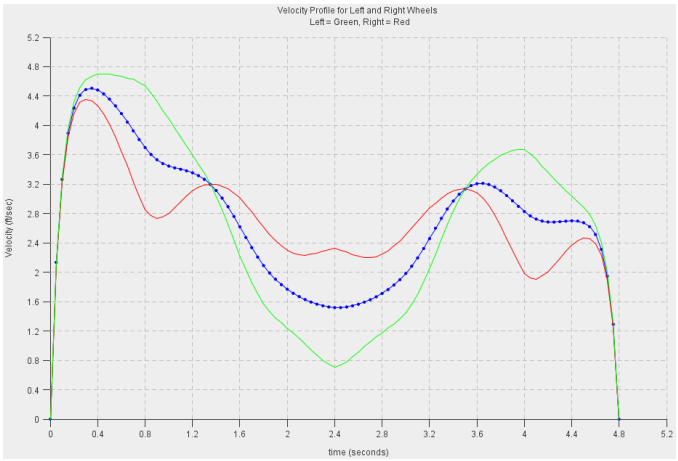


Fig. 10. the velocity profile of left (green) and right (red) wheels at time step = 0.05, total time = 5s, track width = 1 foot.

As seen from this comparison between time steps of 0.01s and 0.05s, we can clearly see that a smaller time step causes a much bumpier (albeit in higher resolution) velocity profile. While such bumpy velocity profile is not necessarily undesirable, and might even be in place to improve the path's fidelity, sharp changes in velocity require very good velocity response from the motors, which in turn requires an extremely well-tuned PID system. We suspect that because of the imperfection of our tuning, our motors are not able to actuate the fast-changing velocity profile created by the tool, thereby causing unpredictable path changes when time step is set to a low value.

IV. Discussion

Figure 7 and Figure 8 tells us that total time is generally negatively correlated to the fidelity of the path shape, but does not influence the static y-errors of the path. We try to confirm this by generating a 3-second path, but that caused the program to generate polygon-like velocity profiles, a sign that

the algorithm has reached a minimum time limit for this specific path. Therefore, we suggest lowering total time right before reaching this breaking-point.

Since track width effects the fidelity of the shape of the curves, we suggest tuning that immediately after tuning total time, so that it is easier to tune time step when the general shape of the curve is correct. While doing so, keep the time step at a large value to ensure that the velocity profile is not bumpy, causing unpredictable effects on the path.

The velocity mode PID values can have a profound effect on implementing motion profile on the robot. In our experiment, we lacked a tool to fine-tune our PID values, since the WPILib "SmartDashboard" does not refresh frequently enough to view the motor velocities with a high enough resolution to tune PID values. As a result, our PID values were only as good as achieving no jittery motion, and cannot be tuned to a higher precision. For reference, our kD value is about 20 times greater than the kP value.

There are several limitations to our experiment. First, not enough data was gathered, especially with curved-path tests. This makes it more difficult to make any general statements about the effects of different parameters on the quality of the path. Second, the data-gathering process of the curved-path tests portion was inadequate—we are relying on camera footage view from a bystander's perspective to determine the coordinates of the robot at the peak point and end point. That can be improved by installing an overhanging camera and having a grid projected onto the test field to accurately measure the coordinates. Third, the effect of battery voltage and the straight-line profile should be studied, since battery voltages can vary slightly across different tests.

Computer vision-guided motion profile will be the one of the most promising potentials of using this tool. Although still needing further study and optimization, once proven to be reliable in reproducing curved paths, this tool will make real-time automatic curved path-planning possible on FRC robots. By using the OpenCV-powered GRIP software, we have been able to measure the area and hence the distance of the robot to a standard FRC competition reflective tape (used as guides for robots to perform CV-guided tasks). As we have also tested, we can accurately calculate the angle of the reflective tape relative to the robot front. This information allows us to write a procedure to generate waypoints that direct our robot to reach the reflective tape.

Fig. 10. picture of a reflective tape that is facing the robot at an angle.

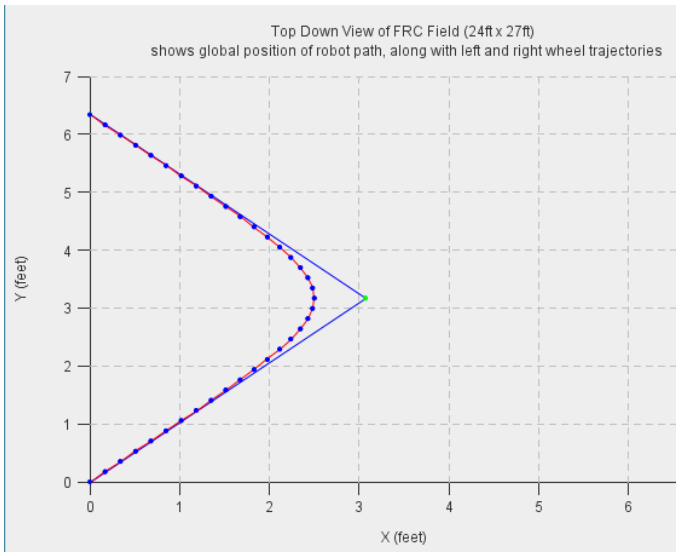


Fig. 11. the motion profile generated to make the robot approach the reflective tape in Fig. 10 perpendicularly. Information such as angle facing the robot and distance are calculated solely based on the image.

Acknowledgement

I would like to thank Dr. George A. Kantor, Girls of Steel team programming mentor Joseph Jackson, and Girls of Steel team student mentor Austin Frownfelter for providing directions and resources to my research.

V. Bibliography

[1] H. Z. Li, Z. M. Gong, W. Lin, and T. Lippa, "Motion profile planning for reduced jerk and vibration residuals," *SIMTech technical reports,* vol. 8, p. Number 1, 2007.

[2] Kim Doang Nguyen, Teck-Chew Ng, I-Ming Chen, "On Algorithms for Planning S-Curve Motion Profiles," *International Journal of Advanced Robotic Systems ,* vol. 5, no. 1, 2008.

[3] KHEngineering," SmoothPathPlanner," (2014), GitHub repository, https://github.com/KHEngineering/SmoothPathPlanner.

Hand-Held Stiffness Measuring Device (HHSMD) for Tumor Detection

Tejas Ajay Zodage, Arun Srivatsan.

***Abstract*— Getting a quantitative stiffness vs position data of an organ will be useful for treatment of diseases which involves tumor formation. For preliminary analysis of such tumors near surface, doctors use palpation techniques which are very subjective. In this paper we present the "Handheld Stiffness Measurement Device (HHSMD)" which gives a map of stiffness of an organ. We are using a Force-Torque sensor and a Position sensor. Measuring the position and stiffness simultaneously and independently makes the device robust as compared to existing devices. The device was tested on silicon specimens embedded with 18-20 times stiffer bodies (compared to silicon) to simulate a real tissue with tumors. We tested the same silicon specimen on a Cartesian bot and compared the results. The accuracy in the measurement of stiffness is ___. The use of this device in real life can help the doctors to treat diseases like breast cancer, prostate cancer, spasm, spasticity, swelling, tumors, hematomas etc.**

I. Introduction

Tumor is an abnormal growth of cells inside body. A tumor may get pressed against a vital organ and can cause serious damage [1]. Tumors are also found in diseases like prostate cancer, breast cancer [2],[3]. 161,360 cases of prostate cancer and 255,180 cases of breast cancer are estimated in US in 2017 [4]. If detected and treated in early stages, high survival rates can be obtained [5].

Information about the shape and size of the tumor is necessary for the treatment of these diseases. To detect tumor, there exist different methods like MRI, CT scan etc. But these methods take time of around 20-90 minutes, need expertise to handle the machines and are least affordable in lower income settings [6]. It's not convenient to use these methods for preliminary analysis of tumors near surface of a body.

For the preliminary tests on a tumor closer to the surface of an organ, doctors use palpation techniques. In palpation technique, doctor uses tip of the finger to press against an organ and gauges a qualitative estimate of stiffness [7], [8]. The palpation techniques are subjective, and the accuracy of its results depend upon skills and experience of the doctor. There may be difference between opinions of two individuals.

Many researchers have worked towards developing devices and sensors for tissue stiffness measurement. Fischer *et al.* [9] was the first one to devise tissue compliance meter for soft tissue consistency [10]. Jari P A Arokoski *et al.* used the force transducer. They use the principle that the force with which tissue resist constant deformation is a measure of stiffness constant [10]. Hakan Oflaf *et al.* [11] have used a similar approach of finding the force under constant deformation. Chieu Van Nguyen *et al.* [7] used the tactile films with polymers and nanoparticles. These films emit light proportional to the local stress. A. Eklund *et al.* and V. Jalkanen *et al.* used the principle that resonance frequency of a Piezo Crystal changes when it is kept in contact with a soft material, depending on the stiffness of the material [11], [12].

For quantitative stiffness measurement, we have developed a prototype of a handheld stiffness measuring device (HHSMD). This device gives a map of stiffness at a point on the organ vs the position of that point on the organ. The device consists of a force torque sensor (FT sensor) and a position sensor. It has an indentation probe attached to the FT sensor. Markers (For position and orientation detection) are fixed to the container containing the FT sensor in scalene triangle format. The organ deforms when indented with the probe. And the reaction force by the body is measured by the FT sensor. The position of the point of indentation as well as the deformation of body at that point is recorded by the position sensor. Throughout the indentation, the data of FT sensor and position sensor are recorded simultaneously. The slope of the force vs deformation is the local stiffness. Which will be interpolated for remaining points

Jari P A Arokoski *et al.* and Hakan Oflaf *et al.* measure force for a constant deformation of the organ [9], [11]. They use a constant length indenter to do this. In this method, prior knowledge of stiffness range of the organ is needed to select the appropriate indenter rod length. It means that doctor can't use the same predetermined deformation for two parts with large difference in stiffness (e.g. cheek and wrist). Our device uses position sensor to detect deformation. Hence it is possible to get a stiffness vs position map which is not attempted by any of the device mentioned above. Our device doesn't require a constant deformation since the force and deformation are independently measured. This makes our device becomes robust as compared to other existing devices.

This device was developed to gauge the effectiveness of a medical treatments of diseases involving tumors. Fischer *et al.* suggested that such devices will help in treatment of diseases like muscle spasm, spasticity, swelling, tumors, hematomas [8].

*Resrach supported by Biorobotics Lab, Robotics Institute Carnegie Mellon University, Pittsburgh.

Tejas Zodage is with the Birla Institute of Technology & Science, Pilani – Goa India (+91 7066890296; e-mail: tzodge@gmail.com).

Arun Srivatsan is with the Robotics Institute, Carnegie Mellon University, Pittsburgh.

II. PROCEDURE

A. Device and System Specification

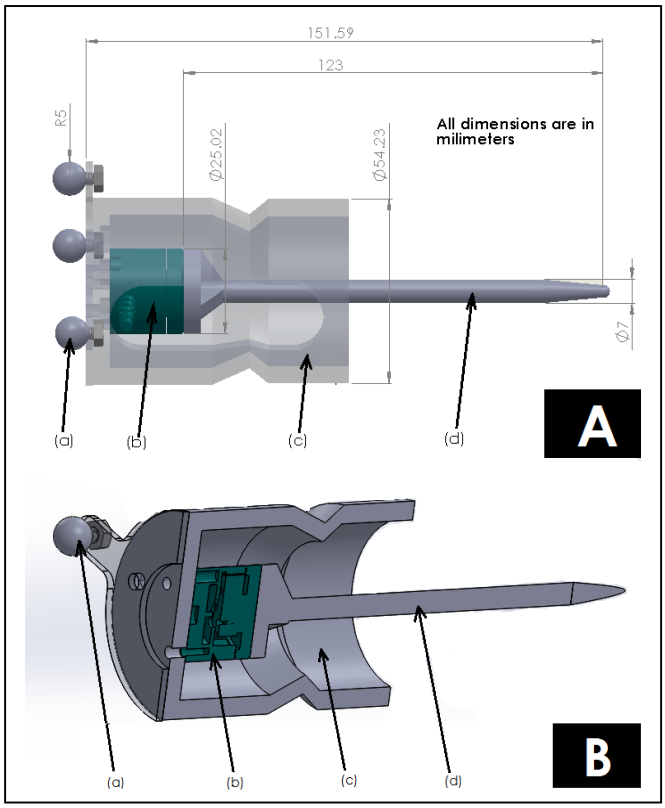


Figure 1. (A) CAD model of handheld stiffness measuring device. (B) Cross sectional view of the handheld stiffness measuring device. (a) IR reflector markers to detect position of the device. (b) Force-torque sensor. (c) 3D printed container. (d) 3D printed Indentation probe.

For force sensing we used ATI nano 25 E transducer and for position sensing we used Optitrack flex 13 cameras. We 3D printed and assembled the device as shown in figure 1. It consists of a probe having spherical end to indent on the The FT sensor is fitted in a hollow plastic container. Three IR reflector beads are attached to the top of the motor mounting in a scalene triangle format. The device is held by placing hand on the container and the probe's tip is pressed against the specimen.

We took the data of the Optitrack system in a windows XP system. It was streamed to a ROS-Kinetic system through Ethernet (IP Multicast). We used pre written ROS packages with some modifications suggested on online forum and motive 1.7 software provided by optitrack. The data of the force torque sensor was directly transmitted to the ROS system using the ROS interface. We are processing this data in MATLAB 2016a.

B. Assumptions for the experiments

For the experiments we are assuming that

- Cameras are calibrated for a level of precision 'Excellent' or 'Exceptional' as standards of the motive software.
- After calibration, camera positions are not changed.
- The test object does not permanently deform when indented.
- The test specimen does not change its position with respect to the platform.

C. Experiments

We embedded a rigid plastic body of random shape in a silicon specimen of dimensions 10cm x 10cm x 2cm. This body is placed on a ground plane set up for the Optitrack cameras.

The probe of the device was indented at 68 points on the silicon object keeping the distribution of points approximately uniform. The data of both FT sensor and Optical sensor is streamed simultaneously. The biasing of FT sensor was done before every experiment. An average force data was collected keeping the probe touched to the organ. This was termed as threshold data. The readings of optical sensor and FT sensor were recorded when the force measured by the FT sensor was above the threshold force. For each indentation, we calculated the slope of force vs displacement of the indentation point. Which was the local stiffness. This data was used to interpolate the data of points in between two indentation points.

To find the ground truth, experiment on the same silicon object were then conducted using a Foxconn robotic arm attached with FT sensor. The Foxconn robot has a precision of 0.05mm. The robotic arm probed the organ in a raster scan pattern at 100 points.

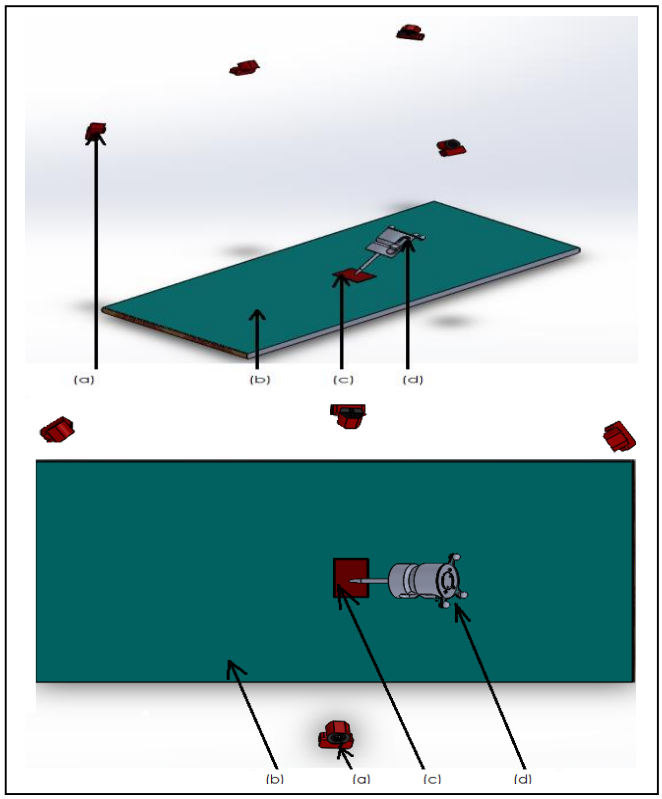


Figure 2 Experimental setup (a) IR imaging camera. (b) Fixed platform for placing the specimen (c) Silicon specimen with embedded bodies. (d) Hand held stiffness measuring device.

III. Results

The stiffness map obtained by using Cartesian bot and optitrack sensors are shown in fig 3. The map gives an idea of the shape of the tumor.

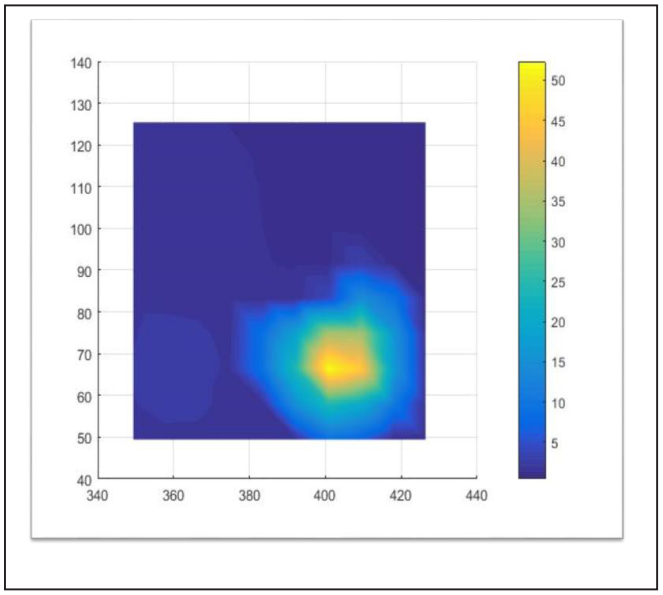


Fig 3. Stiffness map obtained by Foxcon robot (Ground truth).

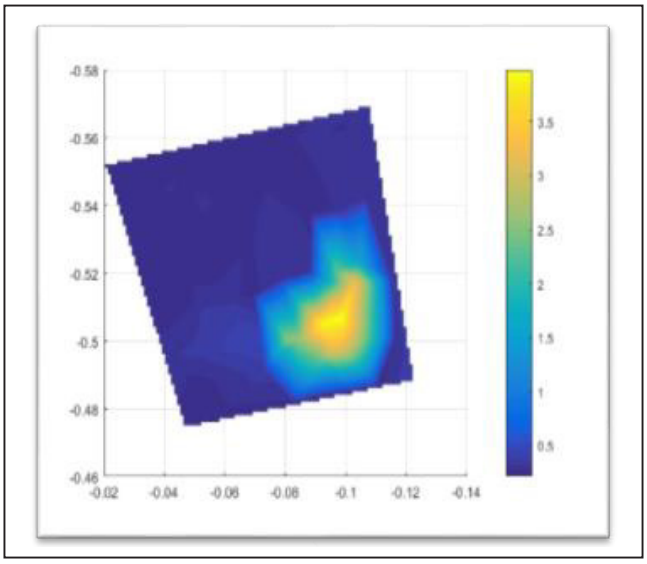


Fig 4. Stiffness map obtained by HHSMD

IV. Future Work

The device will be tested on real tissues under the supervision of a doctor in a real scenario to understand the practical problems it may face.

The upcoming versions can be accommodated with more accurate and smaller sensors. We made the prototype bigger for convenient replacement of components. The size of the device can be optimized for convenient use.

The device currently gives only a stiffness map. Since our FT sensor gives data of forces and torques in all dimensions, there is a possibility of getting a 3D model and position of the tumor using the stiffness data. This will be make the tumor visualization easy and treatment will be more effective.

We are assuming that the body remains stationary and doesn't deform permanently. The assumption of stationary body can lead to errors since there might be unavoidable motions e.g. movement of chest while breathing. Algorithms can be developed to account the slight movement and shape change of the organ.

V. Conclusion

It was observed that the optical sensor gives false readings at some points. These readings were noted and then were replaced by interpolation of the two adjacent point readings. The tip of the device slipped when we probed against the vein like structures in other test samples. Stiffness at such points show unrealistic values such as zero of infinity. So currently, the vein like structure is the limitation of the device.

The tumor's shape was visible in the stiffness map. It was observed that shape will be closer to the real shape if probing is dense near the tumor.

The experimental results were observed qualitatively so far. A quantitative error analysis couldn't be done due to time constrains.

Acknowledgment

I would like to thank Dr Howie Choset and Robotics Institute Carnegie Mellon University, Pittsburgh for providing the opportunity to work on this project. I would also like to thank FICCI the scholarship provider for the RI summer scholars. I express my gratitude towards the RISS program organizing team for organizing this program and providing us the opportunity to explore into robotics.

References

[1] "What You Need to Know About Brain Tumors", *NIH Publication No.09-1558, Revised February 2009, Printed May 2009.*

[2] "Understanding Prostate Cancer, A guide for people with cancer, their families and friends" *First published July 1998. This edition April 2016. © Cancer Council Australia. ISBN 978 1 925136 71 5*

[3] American Cancer Society. "Cancer Facts & Figures 2017" *Atlanta: American Cancer Society; 2017.*

[4] American Cancer Society "Cancer Treatment & Survivorship Facts & Figures 2016-2017" *Atlanta: American Cancer Society; 2016*

[5] Hannah C. McLane, MD, MA, MPH, Aaron L. Berkowitz, MD, PhD, Bryan N. Patenaude, MA, Erica D. McKenzie, BSc, Emma Wolper, BA, Sarah Wahlster, MD, Günther Fink, PhD and Farrah J. Mateen, MD, PhD "Availability, accessibility, and affordability of neurodiagnostic tests in 37 countries"

[6] Nguyen, Chieu Van, and Ravi F. Saraf. "Tactile Imaging of an Imbedded Palpable Structure for Breast Cancer Screening." *ACS Applied Materials & Interfaces 6.18 (2014): 16368–16374. PMC. Web. 1 Aug. 2017.*

[7] Ninghuan Wang, Gregory J. Gerling, Member, IEEE, Reba Moyer Childress, and Marcus L. Marti "Quantifying Palpation Techniques in Relation to Performance in a Clinical Prostate Exam" *IEEE*

TRANSACTIONS ON INFORMATION TECHNOLOGY IN BIOMEDICINE, VOL. 14, NO. 4, JULY 2010

[8] Fischer A A 1987b "Tissue compliance meter for objective documentation of soft tissue consistency and pathology" *Arch. Phys. Med. Rehabil. 68 122-5.*

[9] Jari P A Arokoski, Jarkko Surakka, Tuula Ojala, Pertti Kolari and Jukka S Jurvelin "Feasibility of the use of a novel soft tissue stiffness meter", *2005 Physiol. Meas. 26 215*

[10] Hakan Oflaz, "A new medical device to measure a stiffness of soft materials" Acta of bioengineering and biomechanics" *Wroclaw University of Technology 16(1):125-31 · April 2014*

[11] V.Jalkanen B.M.Andersson O.A.Lindahl "Stiffness of a small tissue phantom measured by a tactile resonance sensor" *P.D. Bamidis and N. Pallikarakis (Eds.): MEDICON 2010, IFMBE Proceedings 29, pp. 395–398, 2010.*

[12] [A. Eklund, A.Bergh O.A., Lindahl, "A catheter tactile sensor for measuring hardness of soft tissue: measurement in a silicone model and in an in vitro human prostate model", *Medical & Biological Engineering & Computing 1999, Vol. 37*


CARNEGIE MELLON UNIVERSITY
PITTSBURGH PENNSYLVANIA 1900