Using Kodu to Program Robots

Troi Williams and Professor David S. Touretzky

Abstract

Microsoft's Kodu Game Lab is a programming environment designed for young children. Kodu programmers create computer games by writing behavioral rules for animated characters. This research develops strategies for mapping Kodu's idealized perception and manipulation primitives onto real robots, which have significant physical constraints. Our target platform is the Calliope2SP robot.

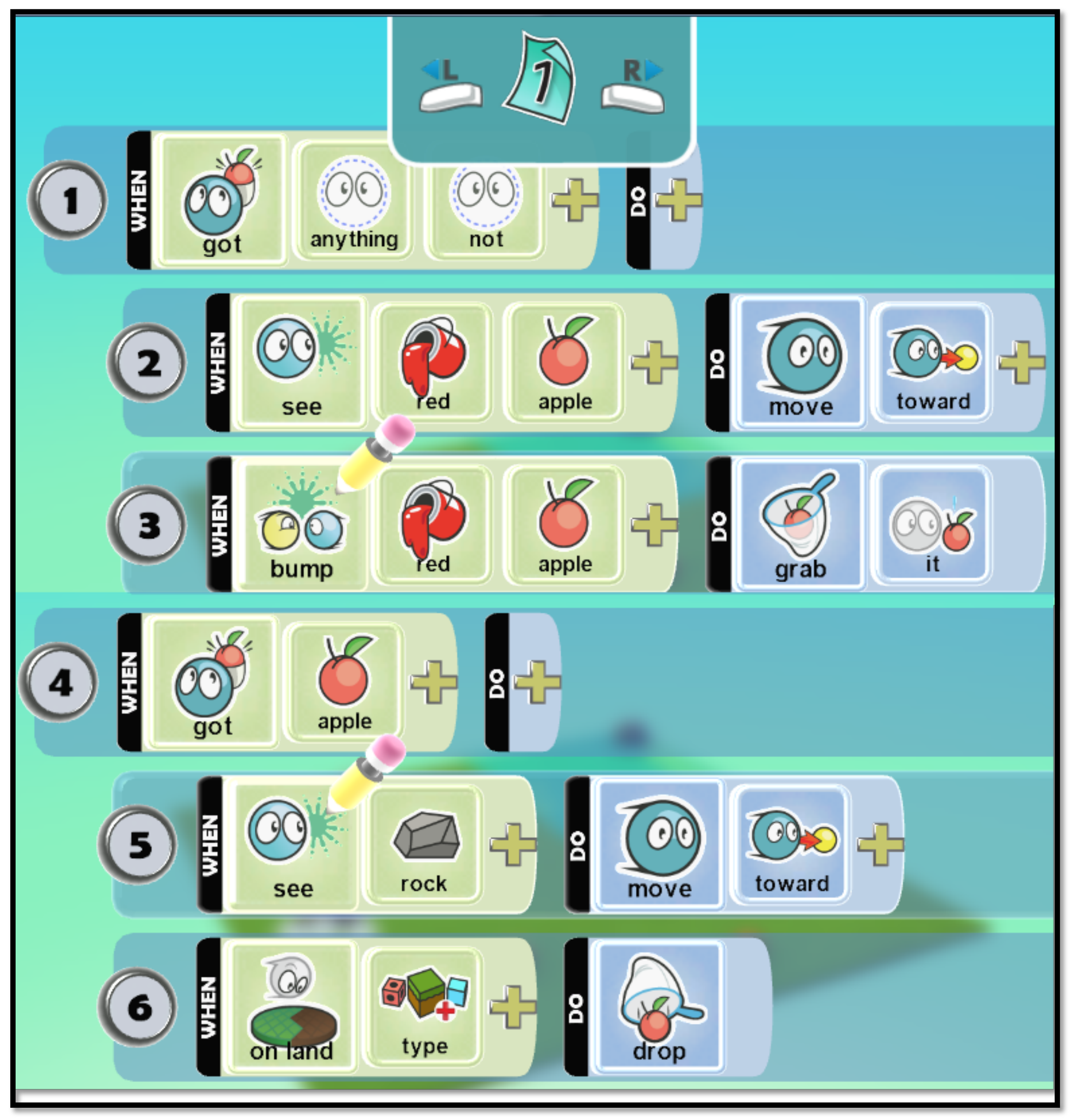
Problem Description

In Kodu's virtual environment, perception and manipulation are instantaneous, omnidirectional, and never fail. On a real robot, vision is slow, expensive, and limited to where the camera is pointed. Also, a robot can only manipulate an object within the gripper's reach, and because gripping is unreliable, the gripper may lose the object while the robot is moving. For these reasons we need strategies to reliably implement Kodu's primitives on a real robot such as the Calliope2SP.

Resources

Kodu Game Lab

- Uses rule-based programming to control Kodu characters.
- Makes programming perceptions and actions easy and intuitive.


L
1
R
1 WHEN got anything not DO
2 WHEN see red apple DO move toward
3 WHEN bump red apple DO grab it
4 WHEN got apple DO
5 WHEN see rock DO move toward
6 WHEN on land type DO drop

Tekkotsu

- Free, open source project developed at CMU.
- Enables high-level programming on multiple, predefined robotic platforms.

The Calliope2SP Robot

- Designed by RoPro Design, Inc. with CMU.
- Approx. 1 foot in diameter and 1.5 feet high.

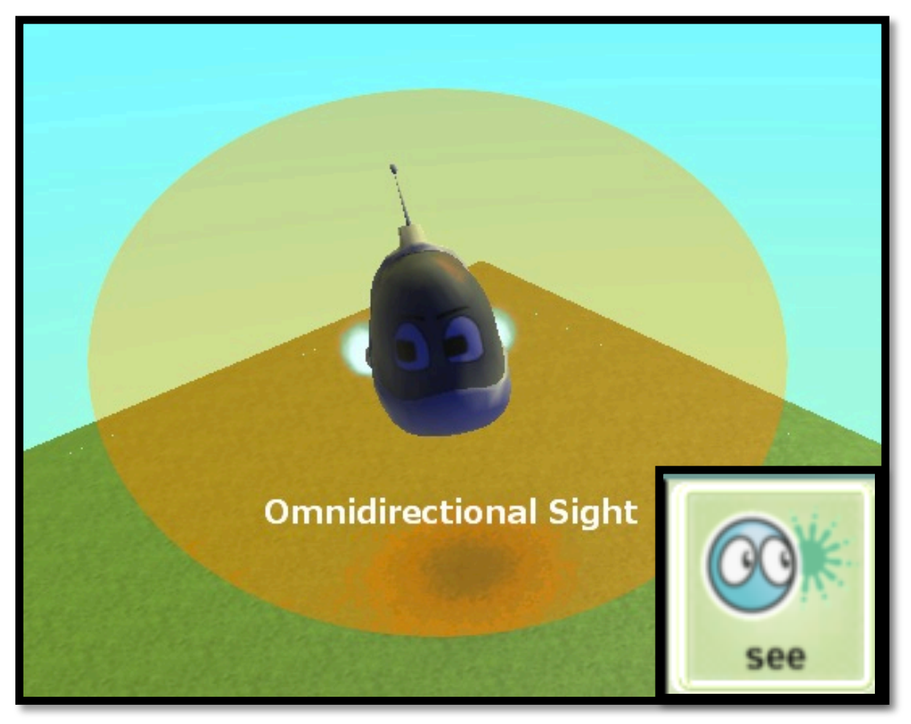
Pan/tilt with a Sony PlayStation Eye Camera

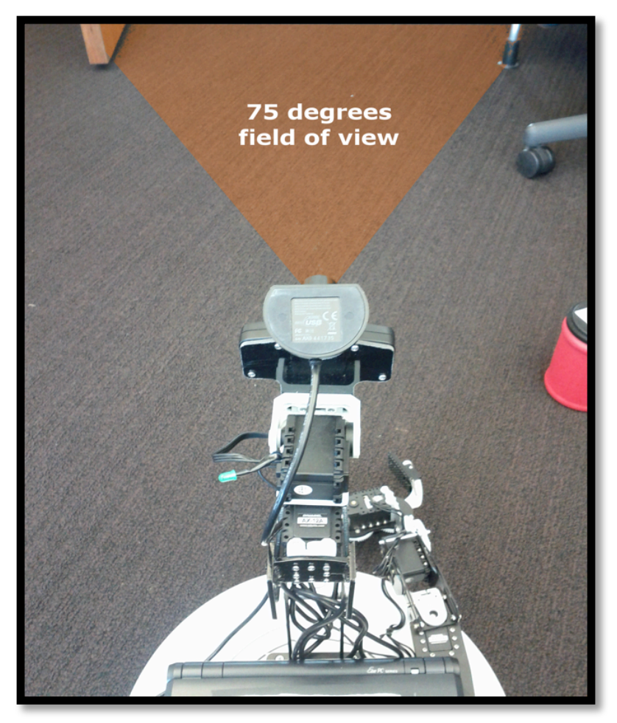
2-DOF arm plus gripper

Netbook running Ubuntu Linux

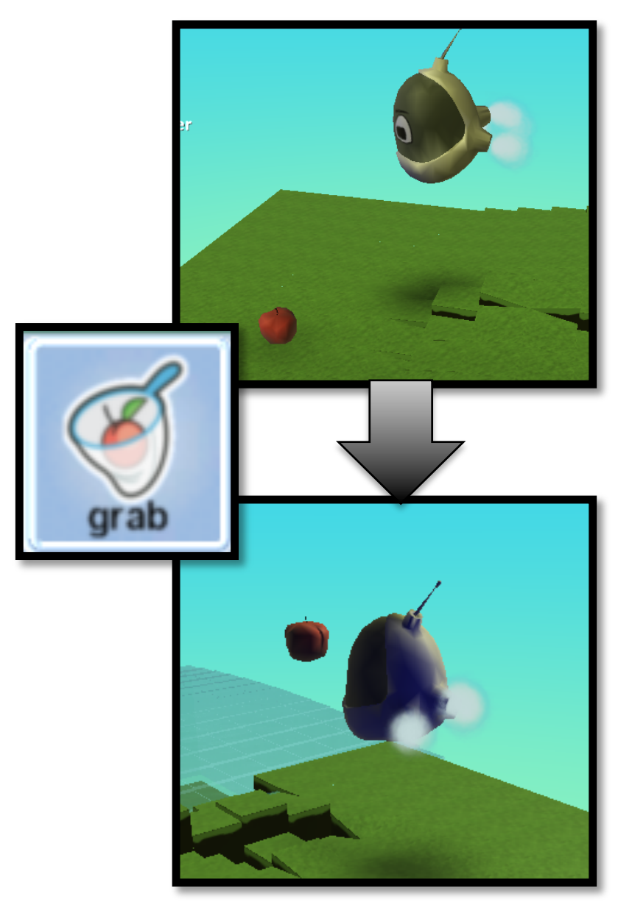
Methodology

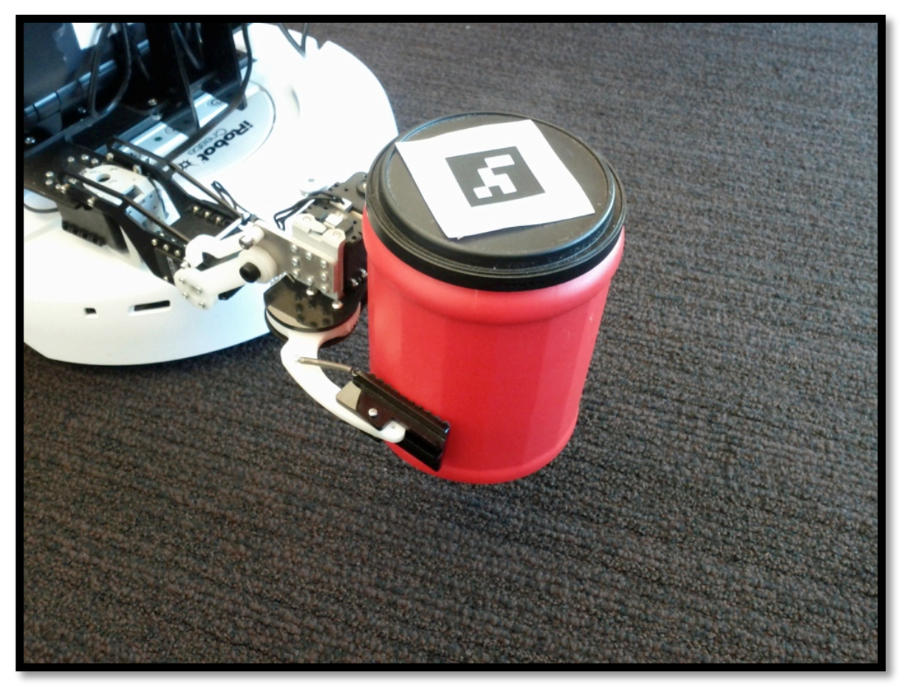
1) Analyzed how Kodu's rule interpreter works.
2) Wrote a detailed description of Kodu perception and manipulation primitives.
3) Examined the perception and manipulation differences between Kodu and the Calliope2SP.





Visual perception differences: Kodu characters can see everywhere (left) while the Calliope2SP is limited to a 75-degree field of view (right).





Object manipulation differences: Kodu characters have an abstract notion of grabbing and holding objects (left). The Calliope2SP has to physically touch an object to manipulate it (right).

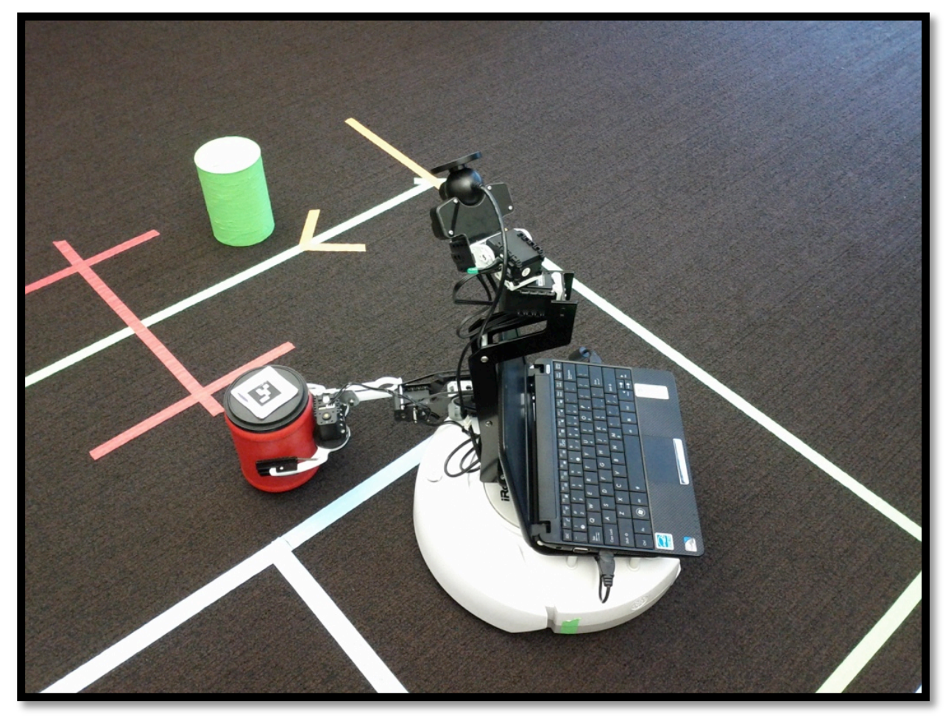
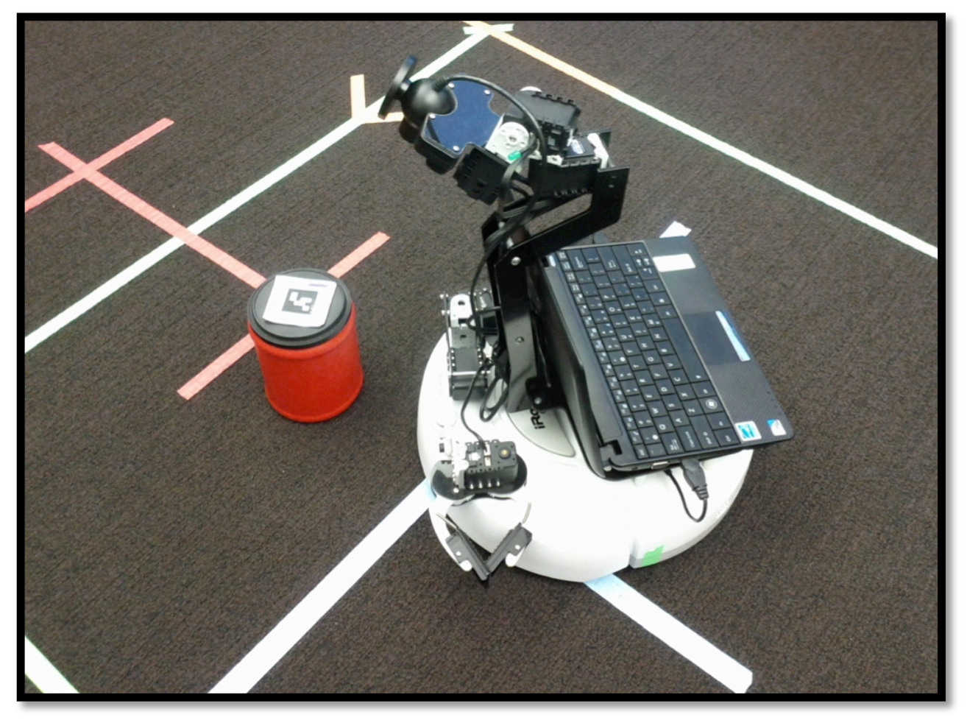
4) Developed strategies to reliably implement Kodu's idealized primitives on the Calliope2SP.

Proposed Strategies

Perceptual Multiplexing

Compiling a visual perception routine and dividing the robot's attention between:

- Detecting bump events
- Tracking distant objects
- Detecting landmarks for localization
- Monitoring a grasped object

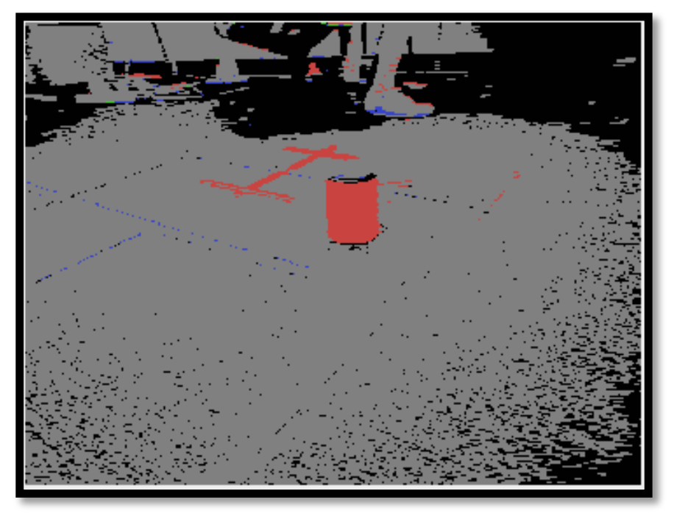
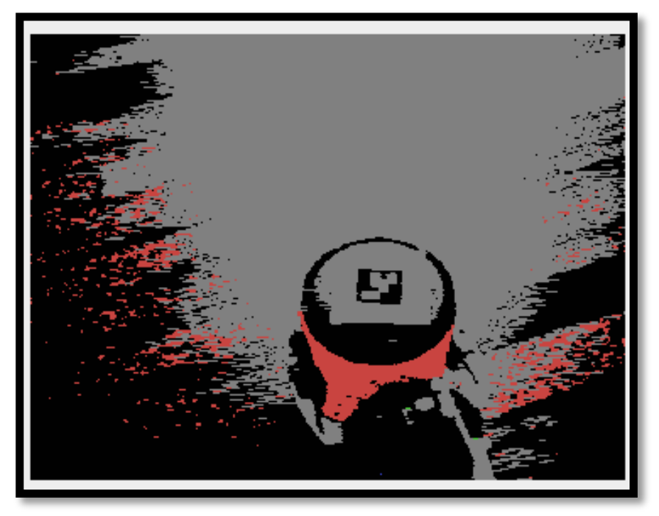



Adaptive Navigation

- Dead reckoning vs. visual homing.
- Knowing when to use each technique based on its capabilities and efficiency.

Fault-Tolerant Manipulation

- Acquiring and manipulating objects with the gripper.
- Detecting and recovering from failures.
- Knowing when to push in lieu of grasping an object.

Future Work

- Implement our proposed visual perception, adaptive navigation, and object manipulation strategies.
- Add other primitives that require inter-robot coordination such as *give.*

References

MacLaurin, Matthew B. "The Design of Kodu: A Tiny Visual Programming Language for Children on the Xbox 360." *POPL '11 Proceedings of the 38th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages.* ACM, 2011.

Touretzky, David S., and Ethan J. Tira-Thompson. "The Tekkotsu "Crew": Teaching Robot Programming at a High Level." *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10).* Association for the Advancement of Artificial Intelligence, 2010.