Learning from failure:
Improving task execution with experience
Andrey Kurenkov
Carnegie Mellon Institute

Abstract
A machine learning approach that enables experience-based error detection and recovery during robotic task execution is developed. A task execution framework that uses classical task planning is used and integrated with persistent database storage of world state. Feature extraction is then done on the stored data to create classification trees that differentiate faulty from successful executions and impose additional constraints on future executions to avoid past errors.

Motivation
Tasks require a series of steps
A representation that is needed
Related

Even simple tasks require STRIPS[1] and more

Representation and execution of varied tasks
- Motivation
  - Tasks require a series of steps specified explicitly and formally to the robot
  - Even simple tasks require complex state machines to account for variations in state
- Need
  - A representation that is naturally adaptive and reuses the same actions smartly
- Related
  - STRIPS[1] and more advanced planners[2]

Problem
The goal of the Personal Robotics Lab is enabling robots to assist people with a wide range of everyday tasks. This research focused on the following two related problems associated with that goal:

- Multiple tasks were carried out in simulation with simulated uncertainty of certain parameters to generate a test database
- After being learned, the classification trees are used during execution to check whether the robot is ever in a state that predicts failure.
- In case of a failure state, such a case provides a goal state by finding a path to success in the pertinent tree.
- Trees with equivalent classification certainty are ranked by testing their effectiveness in simulation, Table 1 contains important trees.

Methods

- Intertwining of classical symbolic planning with execution
- Storage of state during both successful and failed executions with [3]
- Modification of subsequent planning and execution

Figure 1. The basic execution loop for execution with error handling

- Interweaving of classical symbolic planning with execution
- Storage of state during both successful and failed executions with [3]
- Modification of subsequent planning and execution

Table 1. Results of learning for a task with two variations

Results

- Implement on-line learning for constraint refinement and selection
- Perform experiments outside simulation
- Test refining real-world results with subsequent batch simulations

Figure 4. Task start state

Future Work

- Multiple tasks were carried out in simulation with simulated uncertainty of certain parameters to generate a test database
- After being learned, the classification trees are used during execution to check whether the robot is ever in a state that predicts failure.
- In case of a failure state, such a case provides a goal state by finding a path to success in the pertinent tree.
- Trees with equivalent classification certainty are ranked by testing their effectiveness in simulation, Table 1 contains important trees.

References


Figure 3. Classification trees for subactions

Python

- Action, predicate, and goal representation

Fast Downward

- Symbolic planning interleaved with execution

ROS

- Storage of execution logs and learned constraints

MongoDB

- Machine learning

1. Decompose actions into subactions
2. Use 1D continuous variables for features
3. Train classification tree at start and end of each subaction
4. Impose constraints during execution

Table 1. Results of learning for a task with two variations

Figure 4. Task start state

Figure 5. Failure due to book location

Variable varied

Classification Trees Learned

Effect of constraints

Robot heading
- At DriveSegway1_start:
  - if robot | locYaw=-0.025: Failure
  - if robot | locYaw=-0.025: Failure
- Robot correctly rotated to face the book

Book location on Y axis
- At RotateSegway1_start:
  - if robot | locY >= 0.165: Failure
  - if robot | locY <= 0.025: Failure
- Robot abstorted if the book was violating its position constraint

Figure 5. Failure due to book location

References