Mobile Robot Fault Diagnostics using Local and Cooperative Methods

Student(s) Jonathan Lwowski (PhD), Shubham Sarpal (MSc)
Advisers: Patrick Benavidez and Mo Jamshidi (UTSA)

TECHLAV — DoD Center of Excellence in Autonomy
Testing, Evaluation, and Control of Heterogeneous Large-scale Systems of Autonomous Vehicles

Research Thrust Area: Thrust 2 Task T2-2

Introduction

With the advancement in the technology, autonomous mobile robots are widely used for performing various tasks in military industrial and healthcare environments. Due to failures in sensors, actuators and mechanical components, robots can fail to complete some tasks. ~~Failures~~ in these systems propagate through algorithms that are unaware of the presence of a fault. In multi-agent systems, these types of faults can be observed by additional agents. In this research, we are expanding upon an approach from the literature that uses two robots, "Faulty" and "Helper". "Faulty" is the robot which performs a task and "Helper" is the one that helps observes its operation for faults. Typical robotic agent failure scenarios will be developed and tested with the system. Fault diagnostics will be performed and compared using both local and cooperative methods. Results will be compared to the referenced system by using the Jason heterogeneous multi-agent software platform for detecting faults, AgentSpeak for programing the multi-agent communication, and the Robot Operating System (ROS) for hardware experiments on robotic agents.

Objectives

To fully develop the "Faulty" and "Helper" robot scenario, we need to complete the following objectives:

- Develop fault detection models for self-diagnosis of a single robot (aka "Faulty") for various types of faults
- Understand the types of faults and their impacts on the robot behavior through visual inspection and data-based techniques
- Expand the system to use a remote observer robot (aka "Helper")
- Re-train models based on observed data from a camera

For this poster, we have addressed the first two objectives

Technical Challenges / Gaps

Sensing Challenge: For a remote observer robot ("Helper") to be able to detect a fault in a faulty robot ("Faulty"), the sensors on the robot need to be able to capture the fault in sufficient detail. Cameras for instance need to be capable of recording at a rate high enough to capture the fault over more than one image frame.

Computation Challenge: The fault diagnosis algorithm should be able to run near real-time in order to make it useful in live detection of faults in the system. Training of the models should be completed with sufficient amounts of training data to capture types and levels of faults.

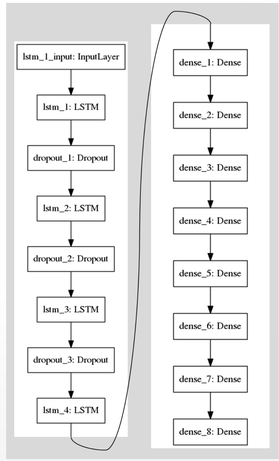
Methods

Training of Initial Fault Detection Models

Datasets were captured from the odometry sensors of a Kobuki Turtlebot 2 robot. Mechanical faults were induced on the left and right wheels in four different test configurations:

a) no fault
b) right wheel fault
c) left wheel fault
d) fault in both wheels

To induce the fault, a quarter turn of the robot's wheel was covered with electrical tape to modify the friction of the wheel. To train the models, we utilized Long Short Term Memory (LSTM) neural networks, which are a type of Recurrent Neural Networks.



Understanding of Visual Fault Indicators for Fault Detection

During acquisition of the datasets, the robot exhibited various behaviors due to the induced faults. The first behavior experienced was a rocking motion when the taped wheel would catch on the ground. The second behavior was a slightly altered path. In one case, the tape caused a catastrophic failure in one wheel when it became stuck in the drive housing. These behaviors can be observed with a camera on the "Helper" robot.

Expansion of System to use Remote Observer Robot

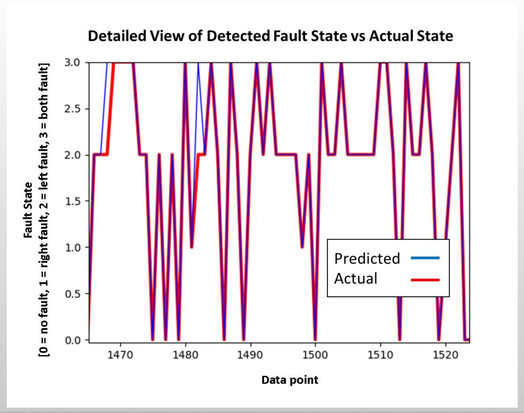
To provide for the remote observer robot "Helper" functionality, there needs to be a method to detect the state of the "Faulty" robot. To do this we have explored a couple methods. The first could use the method in the literature where two colored pillars are mounted on top of the robot. This method did not seem to work well based on the video the authors provided as color image processing suffers due to lighting conditions. Another method, which we have used extensively in the ACE Lab is the ar_track_alvar tags.

Results

Training of Initial Fault Detection Models

Long Short Term Memory (LSTM)

- 6 inputs from the Odometry sensor {X and Y position, Orientation in quaternions}
- 4 outputs {No Fault, Left Fault, Right Fault, Both Fault}
- 88% accuracy on ~19,000 testing datapoints



References

C. Olah. (2015, August) Understanding lstm networks. [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/#fn1

Conclusions / Future Work

In this work we have trained models for detection of a system faults, prepared two robots for the "Faulty" and "Helper" robot scenario, and developed an understanding of the physical indicators that the "Helper" robot can observe. For the cooperative approach, we are still working to integrate Jason, the extended version of AgentSpeak into our ROS system through the use of ROSJAVA and Rason, the ROS interface to Jason.

Acknowledgement

This research is supported by Air Force Research Laboratory and OSD for sponsoring this research under agreement number FA8750-15-2-0116