A Reliable Control System for Autonomous Robotic Systems Using Sliding Autonomy and Data Visualization

Ethan Ito, Computer Science

Mentor: Dr. Daisy Tang

Kellogg Honors College Capstone Project

Problem

Autonomous robotic systems are widely used in today's applications. It is challenging to build a reliable system that gracefully handle various types of unforeseen situations. This project seeks to solve this issue with controlling robots through the use of sliding autonomy and data visualization. We applied this approach to a navigation task.

Approach

Sliding Autonomy

Sliding autonomy is a control process for robotics. The robot can switch between different levels of autonomy, creating a sliding scale. In this project there are four modes of operation: autonomous, semi-autonomous, teleoperation and peer to peer. The sliding autonomy is implemented with Player[1] to control the robot.

Data Visualization

To enhance situation awareness of the operator, data from three sources are visualized. On the main screen, we display the robot's location and it's path to the goal. The data from the robot's laser sensor is displayed in a separate window. The interface is implemented using Qt [2].

Figure 1

Figure 2

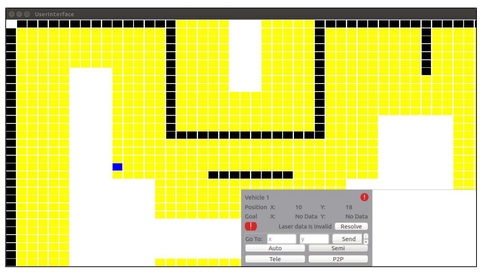

Figure 1: *iCreate robot* used for testing

Figure 2: User interface with invalid laser data alert

Testing

To validate the system, we have an iCreate robot travel to a position in a predetermined map. This task was then tested across three different scenarios: when there is no issue, when a new obstacle is introduced and when there is a laser sensor measurement error. The completion time, the solution quality and the operator workload are recorded for every test run. Each scenario was run multiple times and the average of the runs was calculated.

Performance Measurements

Completion Time

Definition: the time it takes for the robot to reach the goal measured in seconds.

Solution Quality

Definition: a value from 0 to 8 representing how well the robot completed the task.

Operator Workload

Definition: the workload that the operator encounters during execution using NASA Task Load Index.

Figure 3

Figure 4

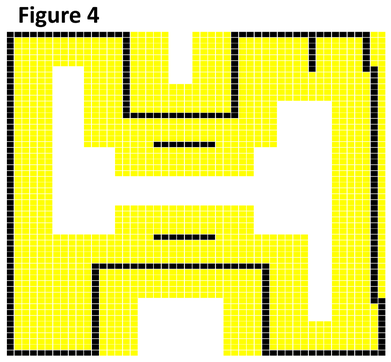

Figure 3: Laser Sensor Display

Figure 4: Static map used for testing.
Black : obstacles
Yellow : grown obstacles
White : empty space

Figure 5

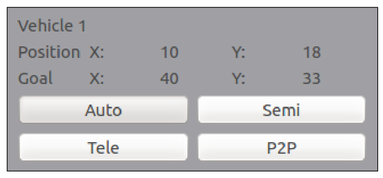

Figure 6

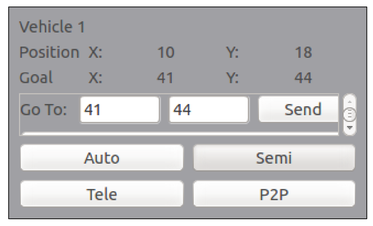

Figure 5: Robot Control Interface in autonomous mode

Figure 6: Robot Control Interface in semi-autonomous mode

Data and Analysis

	Average Completion Time	Average Solution Quality	Average Workload
Expected Results	Less than 2 Minutes	6	40-60
No Issue (Average of 3 runs)	1 Minute 8.6 Seconds	8	15
Introduced Object (Average of 3 runs)	1 Minute 2.6 Seconds	8	28
Laser Error (Average of 3 runs)	1 Minute 44.6 Seconds	8	26
Overall Average	1 Minute 28.67 Seconds	8	23

The longest completion time was 2 minutes and 2 seconds in a scenario where the robot encountered an unexpected obstacle. The operator teleoperated the robot around the obstacle to the goal. The shortest completion time was 1 minute and 9 seconds when there were no issues and the robot operated autonomously. The difference between the two time extremes is understandable when the middle ground between operator control and autonomous control is considered.

Discussion

The interface's design had an effect on the operator workload. The problem that arose was the robot's directional orientation. It was thought that the operator could use the laser sensor display. However it was found that the display did not make much visual sense for navigation when the operator was not used to the map or the interface. This leads to a higher operator workload when teleoperating the robot.

Conclusion and Future Work

The system preformed above expectations in all scenarios tested. It shows that the system is able to complete the task using the implemented interface and sliding autonomy. However improvements can be made to the interface which could result in a decrease in the average workload. There are many ways to improve the project such as optimizations in the code and the increase in the scope of the user interface. Other areas of interest would be quick communication encryption and decryption when handling time sensitive and critical data, and the management of robots in large numbers.

Acknowledgements

I would like to thank my mentor Dr. Daisy Tang, without her help and support this project would not have been possible. I also appreciate the assistance of Kwang Jun for the localization algorithm and supporting code.

References

[1] Qt Project : IDE and framework for user interface and robot control code.
[2] Player : Network Server for robot control