Fall 2023

Final Project Proposal

Software Development for Robotics

November 29, 2023

Students:
Sameer Arjun S (119385876)

Manav B Nagda (119133545)

Ishaan S Parikh (119135891)

Instructors:
Tommy Chang

Course code:
ENPM808X

Contents

List of Figures

1 Overview

Acme Robotics aims to revolutionize fire response systems within warehouse facilities susceptible to fire incidents by introducing "Project Hydra" a swarm of turtle bots designed to effectively contain and address fire emergencies. This innovative solution utilizes a fleet of 20 turtlebots equipped with waterjet spraying capabilities to encircle and contain fire outbreaks within warehouse environments.

The Hydra project employs robots which are designed to respond autonomously to fire emergencies within the facility. The project targets a unique feature to surround the affected machine/object as the most optimal formation to prevent any fire hazards. In the event of a fire outbreak, the system receives input specifying the affected machine and later the robots approach the destination and form the optimal shape of surroundings in shapes of square, circle or triangle.

2 System architecture

The Hydra project system architecture involves a central control unit that receives fire alerts and coordinates the turtlebot swarm's response. Each turtlebot is equipped with ROS-enabled software for autonomous navigation and waterjet spraying mechanisms. The communication network employs ROS Humble, facilitating seamless interaction between the turtlebots and the central control unit.

3 Design and development

The development adheres to Agile methodologies, dividing the project into sprints focusing on key functionalities and testing at each iteration. The programming language of choice is C++ for its flexibility and compatibility with ROS humble.

In the development of the Hydra project employing ROS 2 and C++, the integration of a heuristic mechanism preventing robots from following similar paths, coupled with the implementation of the Rapidly-exploring Random Tree (RRT) algorithm for path planning, stands as a pivotal strategy. This entails creating dedicated C++ classes for RRT-based path planning within ROS 2 nodes, alongside designing and incorporating a heuristic function considering past paths or covered areas. As the system generates paths using RRT, the heuristic guides path selection, ensuring diverse trajectories for individual turtlebots.

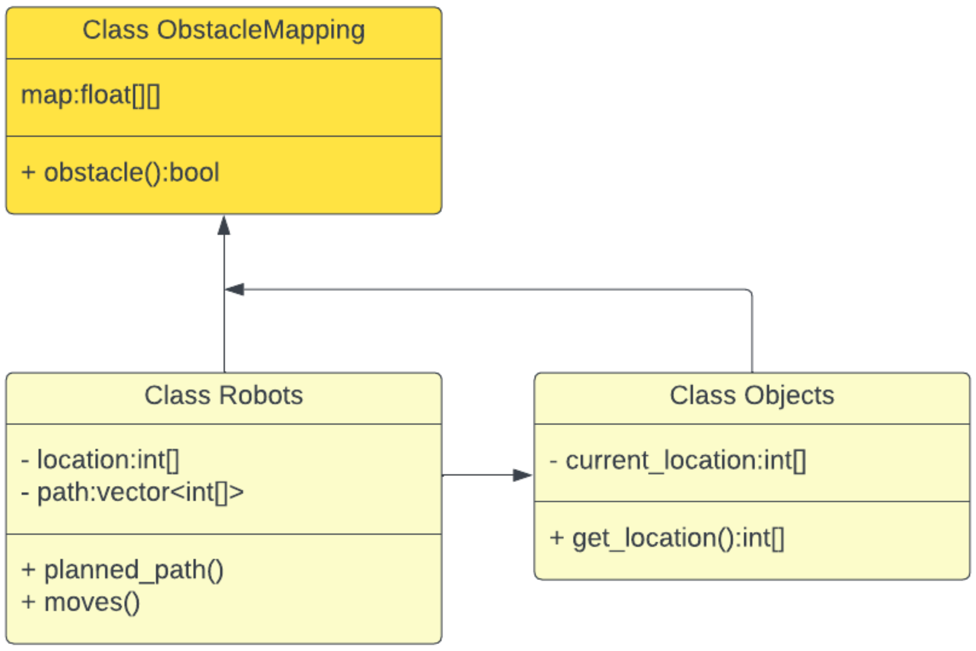

Figure 1: Class Diagram

4 Potential risks and mitigation

The primary technical risks involve sensor integration, swarm coordination algorithms, and real-time fire detection.

- Sensor Integration Issues: Integrating sensors for fire detection and localization may present compatibility or accuracy challenges. This can be resolved by thorough testing and calibration of sensors in simulated environments using Gazebo.
- Swarm Coordination Complexity: Coordinating a swarm of 20 turtlebots simultaneously might lead to communication or coordination issues. This can be resolved by gradually scaling up the number of turtlebots in simulation, testing and optimizing communication protocols and coordination algorithms.
- Safety Concerns: Autonomous systems in emergency situations might pose safety risks to humans or the environment. This can be resolved by implementing fail-safes and emergency stop protocols through extensive risk assessment and scenario-based simulations to ensure the system's safe operation in various fire scenarios.
- Sensor Data Integration:

  Risk: Integrating data from multiple sensors for fire detection might lead to inconsistencies or compatibility problems.
  Mitigation: Develop standardized data interfaces and conduct extensive testing to ensure proper integration and compatibility between different sensor data sources.
- Real-time Processing Demands:

Risk: The need for real-time processing of sensor data and path planning might exceed computational resources, causing delays or system failures.
Mitigation: Optimize algorithms for efficiency, consider parallel processing where applicable, and conduct stress tests to determine system limits.

- Communication and Coordination Challenges:

  Risk: Ensuring seamless communication and coordination among a large number of turtlebots may result in network congestion or message delays.
  Mitigation: Implement decentralized communication protocols, prioritize critical messages, and conduct simulations to assess communication efficiency at scale.

5 Final deliverables

- The final swarm robotics module for project Hydra
- Optimal central communication between Turtlebots using ROS
- Implementation of efficient path planning to target region using RRT* algorithm
- Agile Iterative Process documentation and code coverage with TravisCI and Coveralls
- Memory leaks check and profiling using Valgrind
- Code documentation using Deoxygen tool