Motion Planning of Underwater Robots in Dynamic and Hostile Environment

Final Report

Shivaram Srikanth
Robotics Engineering
Worcester Polytechnic Institute
ssrikanth@wpi.edu

Shubham Malhotra
Robotics Engineering
Worcester Polytechnic Institute
smalhotra@wpi.edu

Venkatesh Mullur
Robotics Engineering
Worcester Polytechnic Institute
vmullur@wpi.edu

***Abstract*—This paper addresses the problem of robot navigation in a hostile environment that mimics the adversarial conditions posed by life under-water. A hybrid algorithm that fuses Rapidly Exploring Random Tree (RRT) to plan the initial path and an Artificial Potential Field (APF) algorithm to avoid dynamic obstacles, was used to solve the problem. The robot in this problem has to navigate to a goal position and then re-navigate back to its start location, in the presence of both static as well as non-deterministic dynamic agents.**

I. Introduction

One of the biggest motives of robotics is to supplant humans with robots in hostile or dangerous environments. This is perhaps why there is an emerging body of research that highlights the need to account for the presence of hostile agents in the robot's environment. This project addresses the problem of motion planning for a robot in a dynamic and hostile environment. The inspiration for the environment came from the kind of adversarial environment encountered under-water. Here, the locomotion of the dynamic obstacles are modeled after aquatic fauna, while the static obstacles represent weeds and rocks found under-water. The majority of the underwater domain remains unmapped, hence robots need to be capable of responding to new information about the environment. As such, if the environment contains structures or natural obstacles, the robot should preferably be able to both sense them and adjust the plan to avoid collisions. This renders traditional path-planning algorithms like A* and genetic algorithms unfit for solving the problem since they require the entire environment to be modeled. Hence, a requirement to combine a sampling-based algorithm with an algorithm that works in dynamic environments arises. This problem finds applications in Bathymetry, search and rescue operations and underwater photography among many others. The environment promises many challenges since aquatic life itself could pose as adversaries to the path of the robot. It may also have to maneuver its way around floating objects and objects on the sea bed. In this paper, the motion of the robot, from start to goal and then back to the start is planned using a combination of the RRT and the APF algorithms.

II. Background

There exists a plethora of path planning algorithms - some of which are summarized in [13]. Some of these algorithms have been in prevalence for decades and are reliable and robust enough to be used in mobile robots. But when it comes to motion planning in dynamic environments, different approaches have sprung up over time. Previously, motion planning in dynamic environments was addressed by adding a time dimension to the robot's configuration space, assuming bounded velocities and known trajectories of the obstacles. [14] solved the planar problem for a polygonal robot among many moving polygonal obstacles, by searching a visibility graph in the configuration-time space. [15] discretized the configuration- time space to a sequence of slices of the configuration space at successive time intervals. This method solved static planning problems at every slice, and put adjacent solutions together. [16] used cell decomposition to represent the configuration-time space, and joined empty cells to connect the start and goal together. In recent years, sampling based algorithms have come to the fore, for use in unpredictable environments. The most popular one is the Rapidly exploring Random Tree(RRT).

III. Goals

The focus of this project is on planning a suitable motion plan that executes the following goals:

- Robots reach the desired goal location
- Avoid dynamic and static obstacles
- Robot avoid any local minima
- Robots retrace path back to start position

The above-given goals are the high-level goals of this project. We hope to achieve and optimize the low-level sections involved in these goals.

IV. Methods and Materials

We started off with the implementation of the A-star algorithm in static obstacles where it could obtain a relatively short path, but it could not handle dynamic obstacles. To perform motion planning, we then implemented APF with dynamic obstacles, but the motion plan using the path generated by APF is not optimal and the convergence time increases as the dimensions increase. RRT being one of the most popularly used planners in AUVs (Autonomous underwater vehicles), is combined with APF to improve upon the time taken to reach the goal.

A. *Artificial Potential Field (APF):*

In the Artificial Potential Field method, the obstacle gives the robot a repulsive force and the target points give it gravity or an attractive force at the same time. The forces felt by the robot are inversely proportional to the distance between the robot and the entity. By continuously calculating the direction of the resultant force we calculate the velocity of the robot. Here, in APF the global planning is done using the attractive force between the target point and our robot and the local plan depends on the repulsive forces.
According to article [1] and a comparison study performed [3], we found out that APF is one of the efficient path planning algorithms in a dynamic environment, as the positions of robots keep on changing. To get a better idea of the performance, we also consider A* [12] and APF [3] when implementing a test scenario.
Performing the A* algorithm after every step is too costly computationally.
Implementation for APF:

- The obstacles in the environment are made to move at random using the moveobstacles( ) in the code.
- The combinedpotential( ) function returns the total force matrix which includes the attractive plus repulsive forces for each cell in the grid. The values are pre-calculated for each iteration depending on the configuration of the environment at that iteration.
- This force matrix, along with the current position is passed into the gradientplanner( ) function. This function calculates the next desired position of the robot based on the mean of the gradients in the neighborhood of the robot.
- A threshold is set, below which it is assumed the robot has reached the goal.
- The minimaChecker( ) function helps the robot escape a local minima if the robot is static with respect to any axis for a specified number of iterations

1) Implementation of RRT:: The idea of the RRT is to create a tree by random sampling in the search space where the node closest to the sampling point on this random tree is found and then link a specific step in the direction to the sampling point as the new node and perform collision detection. If isCollisionFreeVertex() is true, we add the Qnewnode to our random tree.

- First the path is generated using the RRT algorithm.
- Next, the forces are calculated. Here, two forces of attraction are applied by both the nearest node as well as the goal node. The repulsive force is added to this according to the position of the obstacles.
- If the robot deviates from the RRT path, recalculate the nearest node and plan the motion such that the attractive force is applied by the next nearest node and the goal.

B. *Simulation Tools*

A significant amount of time was utilised experimenting with different visualisation tools. From utilising Rviz for 3D path planning to A* algorithm using Pygame. However, because of the familiarisation of python and matplot lib over the duration of course. We decided to simulate the project on python.

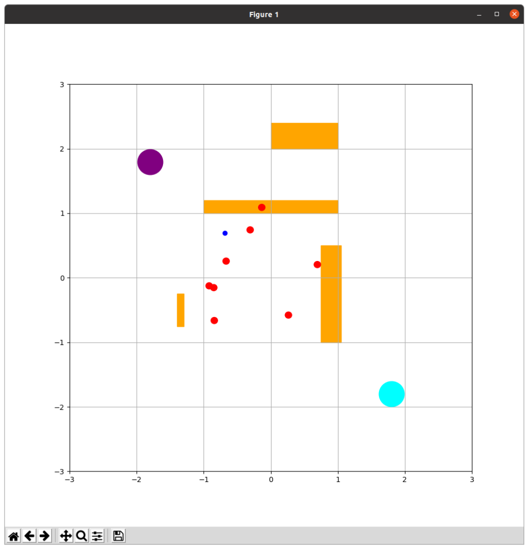

Fig. 1: Environment

C. *Final Environment*

The final environment for this project is a 500 by 500 unit grid where each grid is a traversable node. The dimensions are selected by scaling down a full model made in matplotlib representing the surface in 2D.

The two dimensional environment was selected to decrease complexity and focus more on the path planning aspect of the project. The purple point is a starting point, blue is the ending point and the small dark blue point is the robot. The multiple red points are our dynamic/ hostile enemy points. Their are traversing random paths thus making the path non-deterministic and finally orange blocks are the static obstacles representing the Flora under water.

D. *Implementation of Layered Algorithm: RRT-APF*

- First the path is generated using the RRT algorithm.
- Next, the forces are calculated. Here, two forces of attraction are applied by both the nearest node as well as the goal node. The repulsive force is added to this according to the position of the obstacles.
- If the robot deviates from the RRT path, recalculate the nearest node and plan the motion such that the attractive force is applied by the next nearest node and the goal.

E. *Collision avoidance and Path Update*

Collision avoidance is handled by the repulsive forces produced between the robot and obstacles due to the APF algorithm as well as the occupancy grid created in the RRT implementation.

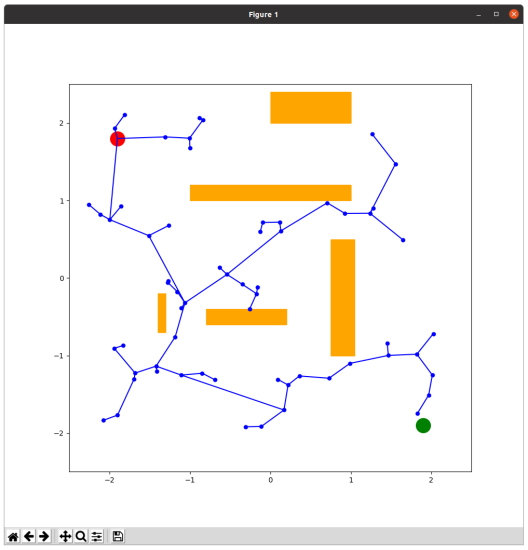

Fig. 2: RRT implementation as our global planner

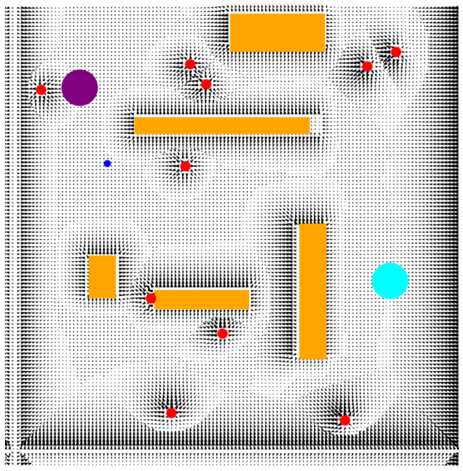
Fig. 3: Environment with the potential fields

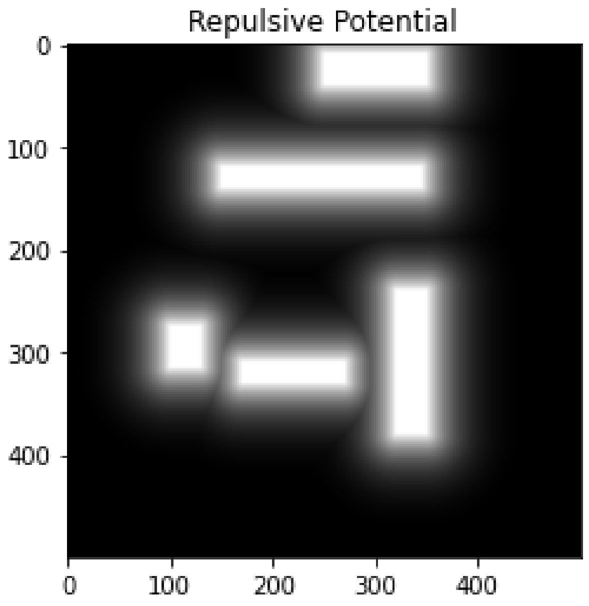

Fig. 4: Global Planner: APF Repulsion before fusion with RRT

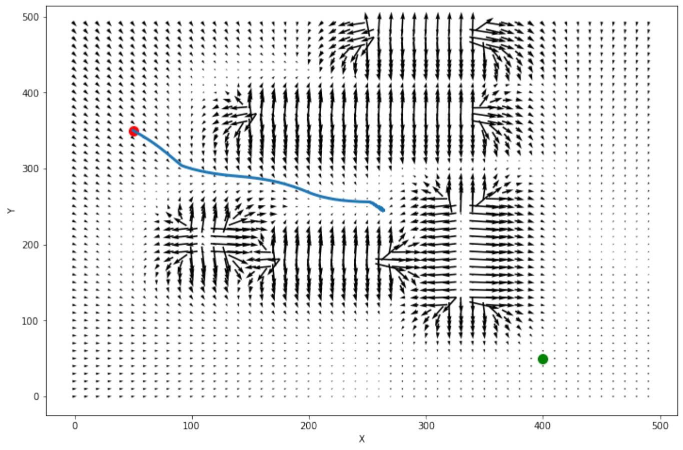

Fig. 5: The potential field visualization plot

F. Visualisation

A 2D visualisation of the environment was done using python's matplotlib tool.

V. Assumptions

Many assumptions were made during the project to aid in simplifying the implementation.

- A perception system is assumed to be present in the robot to focus purely on the motion planning aspects of the problem.
- The robot and the dynamic obstacles are assumed to be point robots to avoid the non-holonomic constraints that come with higher dimensional robots.
- The environment is visualised in two dimensions to reduce the complexity of the problem.

VI. Results

Implementing three major path planning algorithms which have been proven to make smart decisions, provides a good visualization of how the path will be chosen when moving from one point to another in the world. The final paths taken by A*, RRT, and APF algorithms when moving from a start node at position (-1.8, 1.8) to a goal position of (1.8, -1.8) in a 500 by 500 unit grid world.

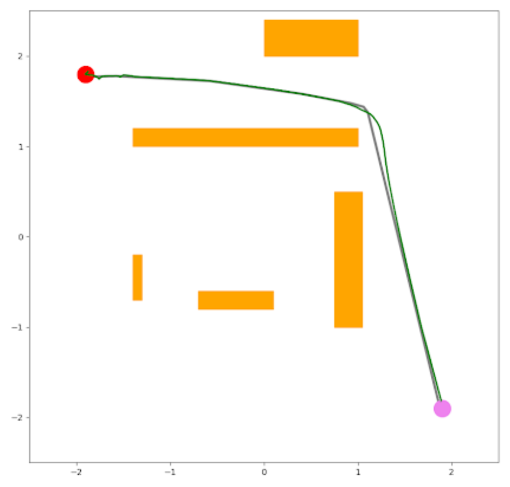

Fig. 6: Layered Planner Output Plots

A star path planning algorithm was considered initially for our problem statement but further research helped us determine derivatives of A star and other algorithms which are better suited for environments where obstacles might change position.

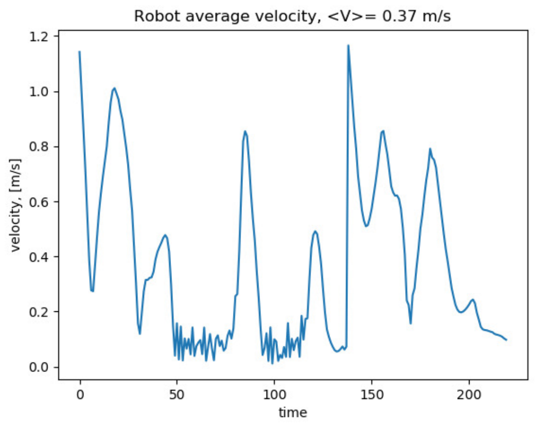

Fig. 7: Robots average velocity

Fig 7. and Fig 8 show the average velocity and the robots velocity calculated by the resultant of the attractive and repulsive forces.

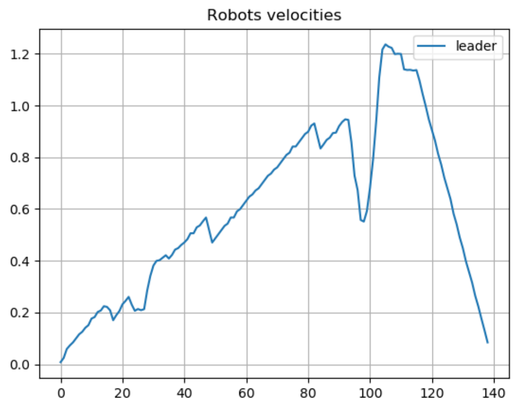

Fig. 8: Robots Velocity

VII. Challenges

- Collision avoidance for non-deterministic dynamic obstacles. This was a challenge that was overcome by tuning of the potential field parameters for both attractive and repulsive forces.
- Local minima caused by the Artificial potential field algorithm. The RRT implementations severely brought down the frequency of its occurence. Also, a function was implemented to help the robot escape local minima.
- The algorithm was hard to implement in 3D using matplotlib owing to the data being computationally heavy.

VIII. Future Scope

Motion Planning for Underwater Robots remains a challenging problem due to the dynamic and uncertain environment. Our project scales and simplifies the problem by considering the Robot to be a point robot (holonomic) and the dynamic environment to be 2D. Hence Future work on the problem can we work upon by relaxing the Assumptions that Means

- **Using Non-Holonomic Robot and 3D environment for motion planning: Constant-depth motions**
  In some AUV applications, it is reasonable to assume that the vehicle navigates at a constant depth. In the case of the Sparus II AUV it is still not capable of conducting lateral motion. Therefore, the vehicle is subject to a motion constraint along its y-axis. Differential motion constraints can be expressed as a set of differential equations in the general form qdot = f(q,u), where q and qdot are the system state and its first derivative, respectively, and u is the control input.
- **Deep Reinforcement Learning Using Raw Depth Images**
  Creates desirable motion plans for quadrotor navigation. Informed by a rough path to goal in partially unknown environments, using raw depth images from a front-facing camera. [11]

IX. Schedule

Task	Due Date
Literary survey and designing environment	15st February
Comparing different planning algorithms	10th March
Set up environment in Python dynamic obstacles	25th March
RRT algorithm implementation in Dynamic Environment	8th April
Implementing the APF algorithm in Dynamic Environment	18 th April
Implementation of Layered Planner	27th April
Code cleanup	29 th April
Final Presentation	29th April
Final Report	1st May

X. Conclusion

This project achieved the desired results of maneuvering through the hostile environment without being intercepted by the dynamic obstacles. It also finds its way back to the start location after reaching the goal. Future work on this project can include changing the global and local path planner. Also, a three dimensional system can be implemented along with implementations of the 3D kinematics of the robot.

References

[1] C. Ju, Q. Luo and X. Yan, "Path Planning Using Artificial Potential Field Method And A-star Fusion Algorithm," 2020 Global Reliability and Prognostics and Health Management (PHM-Shanghai), 2020, pp. 1-7, doi: 10.1109/PHM-Shanghai49105.2020.9280929.

[2] Agishev, R. (2019). Robot path planning, mapping and exploration algorithms (Version 0.0.1) [Computer software].

[3] Panda, M., Das, B., Subudhi, B. et al. A Comprehensive Review of Path Planning Algorithms for Autonomous Underwater Vehicles. Int. J. Autom. Comput. 17, 321–352 (2020).

[4] Buniyamin, N., Ngah, W. W., Sariff, N., Mohamad, Z. (2011). A simple local path planning algorithm for autonomous mobile robots. International journal of systems applications, Engineering development, 5(2), 151-159..

[5] https://github.com/addy1997

[6] Rosa, William & Bessa, Iury & Cordeiro, Lucas (2016). Application of Global Route-Planning Algorithms with Geodesy.

[7] https://medium.com/@rymshasiddiqui/path-planning-using-potential-field-algorithm-a30ad12bdb08

[8] Koenig, Sven, Maxim Likhachev, and David Furcy. "Lifelong planning A*." Artificial Intelligence 155.1-2 (2004): 93-146.

[9] Huiming Zhou (2020) PathPlanning [Source code]. https://github.com/zhm-real/PathPlanning.

[10] Hernández, J. D., Vidal, E., Moll, M., Palomeras, N., Carreras, M., Kavraki, L. E. (2019). Online motion planning for unexplored underwater environments using autonomous underwater vehicles. Journal of Field Robotics, 36(2), 370-396.

[11] E. Camci, D. Campolo and E. Kayacan, "Deep Reinforcement Learning for Motion Planning of Quadrotors Using Raw Depth Images," 2020 International Joint Conference on Neural Networks (IJCNN), 2020, pp. 1-7, doi: 10.1109/IJCNN48605.2020.9207490.

[12] Yan, Z.; Zhao, Y.; Chen, T.; Deng, C. 3D path planning for AUV based on circle searching. In Proceedings of the OCEANS 2012 MTS/IEEE: Harnessing the Power of the Ocean, Hampton Roads, VA, USA, 14–19 October 2012.

[13] Panda, Madhusmita, Bikramaditya Das, Bidyadhar Subudhi, and Bibhuti Bhusan Pati. "A comprehensive review of path planning algorithms for autonomous underwater vehicles." International Journal of Automation and Computing 17, no. 3 (2020): 321-352.

[14] Reif, John H. Complexity of the Generalized Mover's Problem. HARVARD UNIV CAMBRIDGE MA AIKEN COMPUTATION LAB, 1985.

[15] Lozano-Pérez, Tomás, and Michael A. Wesley. "An algorithm for planning collision-free paths among polyhedral obstacles." Communications of the ACM 22, no. 10 (1979): 560-570.

[16] Fujimura, Kikuo, and Hanan Samet. "A hierarchical strategy for path planning among moving obstacles (mobile robot)." IEEE transactions on robotics and Automation 5, no. 1 (1989): 61-69.