An Approach for Mapping Unknown Environments using Frontier-Led Swarms

Kumara Ritvik Oruganti
A James Clark School of Engineering
University of Maryland
College Park, Maryland 20742
Email: okritvik@umd.edu

Abstract—**Robots have become increasingly important in many aspects of life, from manufacturing to health care, logistics to entertainment. The basic building blocks for the robot or agent to function in an environment are Perception, Navigation and Control. To navigate in an environment, it is necessary for the robot to require a map in which they operate. This project proposes an implementation of a novel swarm-based algorithm for the exploration and coverage of unknown environments while maintaining the formation of the swarm and a frontier-based search for effective exploration of the unknown environment.**

***Index Terms*: Navigation, Control, Perception, Swarm, Frontier-Based Search**

1. Introduction

A common assumption for robotics applications is that the robot has prior information about the environment. The state-of-the-art planning algorithms like Dijkstra [10], A*, Randomly Exploring Rapid Trees (RRT) [11] and different flavors of RRT assume that the obstacle space is well known to the agent. Exploration is the process of selecting a location that maximizes the search area coverage. Conventional coverage algorithms like the Voronoi-based [9] approach, and the graph-based approach assumes independent agents. To tackle this problem of priori information of environment to the robot, this project proposes to implement a frontier-based exploration strategy using multiple robots motivated by the biological and physicomimetic emergent behaviours in bird flocks [12], ants [14], and repulsion between likely charged particles [15].

Frontiers are the boundary regions between known and unknown space. To gain knowledge of unknown space, the agents or robots should move near the frontiers and explore again [1]. This process should be repeated until there are no frontiers left for the robot to explore which completes the exploration of the environment. There are several approaches for the frontier-based search algorithm namely Wavefront Frontier Detection [7], Fast Frontier Detection etc. This project utilizes the concept of Reynolds' boids [8] to implement the search strategy. There are many other conventional algorithms for Simultaneous Localization and Mapping (SLAM) that updates the environment information while exploring and avoiding the obstacles simultaneously on platforms like TurtleBot and Robot Operating System (ROS).

2. Related Work

Titus Cieslewski et al. [2] proposed a novel approach of using a version control system to look up and update the map of an agent and reduce the bandwidth of inter-robot communication for the distributed networks. Alejandro Puente-Castro et al. [3] has proposed a swarm path planning using the reinforcement learning technique for the application of field prospecting regardless of the size of the field. Syed Irfan Ali Meerza et al. [13] proposed a dynamic obstacle avoidance and coverage technique using particle swarm optimization. Area coverage can be divided into two types, namely, static coverage [4] and dynamic coverage. Static coverage corresponds to the formation of a swarm in such a way that the sensors cover the whole environment. Practically, in many applications, this is not possible and hence, due to the limitations in the range of the sensors and communication bandwidth, the robots move and sample the environment at a reasonable resolution to cover the entire environment [5]. Collaborative exploration and coverage is another field where dynamic obstacle avoidance and situational awareness are required. Cheng et al. [6] proposed a dynamic coverage algorithm using a leader-follower that uses a flocking technique for the formation of the swarm. Dario Albani et al. used UAV swarm to distributively map the agricultural fields to monitor weed detection [16]. A novel multi-robot coverage path planning algorithm which is time efficient and results a balanced workload for the UAV swarm is proposed by Leighton Collins et al. for the assessment of disaster post-floods [17].

3. Problem Statement

3.1. Map Building

A 2D grid defined by G with a coverage grid cell location $G_{i,j}$ is said to be covered by the agent A_i when the position P of the agent A_i is $P_i = G^i_{i,j}$ and $G^i_{i,j} = 1$.

When the agent A_i receives the coverage grid cell from the neighbouring agent A_j, then the agent A_i is said to have covered the cell $G^i_{i,j}$ indirectly.

If there is an obstacle, then $G_{i,j} = -1$ and $G_{i,j} = 0$ if the area is unexplored.

3.2. Map Stitching

An agent $A_{i,j}$ starts at an initial position $P^i_{initial}$ and explores its own grid map. When it receives a grid map from another agent A_j, The maps are updated by taking the mathematical formulation of comparing each cell of the agent's grid map.

$$G^i_{i,j} = \begin{cases} 1, & \text{if } G^i_{i,j} = 0, G^j_{i,j} = 1 \\ 0, & \text{if } G^i_{i,j} = 0, G^j_{i,j} = 0 \\ -1, & \text{if } G^i_{i,j} = 0, G^j_{i,j} = -1 \end{cases}$$

Similarly, the Map of the agent A^j is updated by comparing the values with the grid G^i of the agent A_i

3.3. Agents within range

We can define set of agents

$$N_a = A^k | k \neq i, \quad \|P^k - P^i\| < R$$

3.4. Average position

The agent moves towards the average position of the nearby neighbours. The average position of the agents within the range of communication can be calculated as

$$\hat{P} = \frac{\Sigma_k P^k}{|N_a|}$$

The direction of the agent is towards the average heading of its neighbours. Similarly the average orientation can be calculated as

$$\hat{\theta} = \frac{\Sigma_k \theta^k}{|N_a|}$$

3.5. Velocity of the agent

The velocity of the agent A_i is given as

$$v^i_{flock(t+1)} = v^i_{flock(t)} + v_{c(t)} + v_{a(t)} + v_{s(t)}$$

where v_c, v_a, v_s are cohesion, alignment and separation velocity vectors. The alignment velocity is calculated as

$$v_a = \frac{\Sigma_k v^k_{flock}}{|N_a|}$$
$$N_a = A^k | k \neq i, \quad \|P^k - P^i\| < R_a$$

The cohesion velocity is calculated as

$$v_a = P^i - \frac{\Sigma_k P^k}{|N_a|}$$
$$N_a = A^k | k \neq i, \quad \|P^k - P^i\| < R_c$$

The agents move away from the other agents to avoid collision as in the repulsion in the likely charged particles. The separation velocity is calculated as

$$v_a = P^i - \frac{\Sigma_k P^k}{|N_a|}$$
$$N_a = A^k | k \neq i, \quad \|P^k - P^i\| < R_s$$

To limit the velocities, the velocity updation will be in the range of $[V_{min}, V_{max}]$.

3.6. Frontiers

The coverage problem is dealt with frontiers in an unknown environment by specifying the boundary conditions of the environment. This approach uses a distributed solution that has a coverage matrix in each agent that gets shared with its neighbours. The map is decomposed into cells with a resolution and those cells are initially set as unexplored. When the agent passes through that cell, the cell is set as explored and when there is an obstacle, the value of the cell is set with a value that denotes an obstacle. A Breadth-First-Search or connected components algorithm is then used to detect the frontier regions and explore nearby frontier centroids from the robot's present position. The flowchart is given in figure 1

Any cell that is in between the explored cell $G_{i,j} = 1$ and unexplored cell $G_{i,j} = 0$ is added to the list of cell F. Given the set of frontier cells F, we find the frontier regions F_r. The centroids of all the frontier regions F_r are calculated and a position p_r is chosen such that

$$p_r = min(\|\hat{F}^i_r - p^i\|)$$

where $\hat{F}^i_r$ is the centroid of the i^{th} frontier region

3.7. Obstacle avoidance

Obstacle avoidance is achieved using repulsion force between the robots by utilizing the LIDAR sensors on the robots. The cohesion and alignment velocities are updated by the dynamic distance-based repulsive scale.

$$\delta_i = \frac{D - p^i}{R_c}$$

where D is the obstacle distance.

$$v^{'}_a = v_a * max((1 - 2\delta_i), 0)$$
$$v^{'}_c = v_c * max((1 - 2\delta_i), 0)$$

Problem 1: *Coverage of an environment*
Given a group of robot agents and an unknown environment

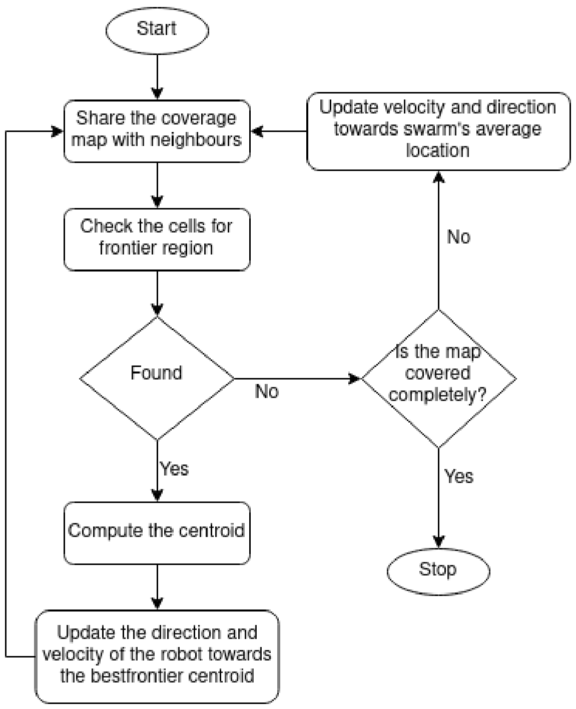
Start
Share the coverage map with neighbours
Update velocity and direction towards swarm's average location
Check the cells for frontier region
No
Found
No
Is the map covered completely?
Yes
Yes
Compute the centroid
Stop
Update the direction and velocity of the robot towards the bestfrontier centroid

Figure 1. An overview of the implementation

with boundaries, find an effective way of sharing the obstacle map asynchronously while finding frontier regions.
Problem 2: *Formation of the swarm*
Given the individual obstacle map (occupancy grid) of each robot, find neighbouring agents and compute the effective heading angle and velocity towards a common frontier centroid.

4. Progress

A 2D map of size (25,25) is created with randomized locations of obstacles. Three robot agents were placed in randomized locations. An occupancy grid for each robot is then created. The robot will search for the frontiers (initially it will be the centre of the occupancy grid) using the connected components algorithm and move towards the centre. The movement is one step at a time and the search region consists of 8 ways and the robot can move in all the 8 directions depending on the centroid of the closest frontier. When there is no obstacle in the vicinity (8 cells around the robot), the occupancy grid is updated with a value that corresponds to no obstacle region.
Robot collisions are avoided as the other agents know their heading action and the other robot's location in their vicinity. The robot will update its occupancy grid with obstacle if it finds an obstacle in its searching area (like a lidar). 10 iterations were performed on each robot agent and the obstacle grid is updated. There results are shown in the below figures.
Since there is a lot of learning in developing the code by integrating multiple concepts and randomizing the obstacles,

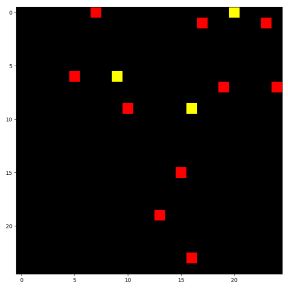

Figure 2. Randommized locations of obstacles and agents

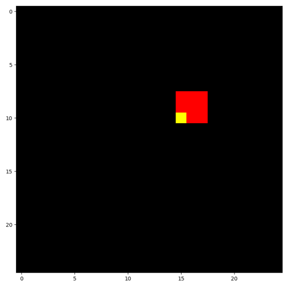

Figure 3. Agent 1 obstacle map in iteration 1

I decided to start the project by visualizing on a 2D map rather than simulating the algorithms on TurtleBot3 using Gazebo.

4.1. Results

In figure 2 The red regions indicate obstacles and the yellow regions indicate the locations of the robot. Figures 3, 4, 5 denote the initial search region in red and the updated robot location in yellow. Figures 6, 7, 8 denote the individual agent's coverage map where white regions denote obstacles. Agent 2 and 3 stopped their search as they reached their first frontier region. Since the robots were coming towards the closest frontier centroid, it is by default implementation that the rules of boids algorithm were followed by the agents.

4.2. To be done

- Update the next frontier region by comparing whether the agent reached the required frontier region.
- When the robots are nearby their vicinity, share the occupancy grid and share the information among themselves.
- Iterate the algorithm till the search is complete.

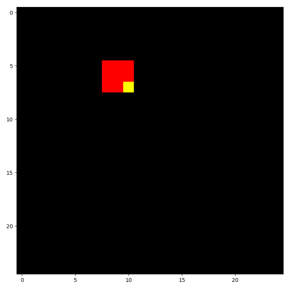
Figure 4. Agent 2 obstacle map in iteration 1

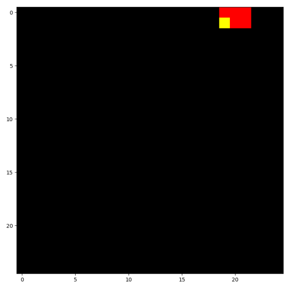
Figure 5. Agent 3 obstacle map in iteration 1

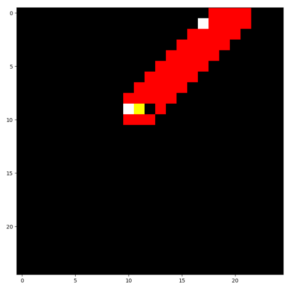
Figure 6. Agent 1 obstacle map in iteration 10

5. Goals

The goal of this project is to implement the above-proposed approach on a swarm of TurtleBot3 using ROS 2 framework in a Gazebo environment. Due to the development stage of ROS 2 and issues related to the ROBOTIS TurtleBot3 packages with ROS 2 Foxy and Humble versions, there can be a switch between using TurtleBot3 with ROS Noetic or ROS 2 Foxy and Humble.

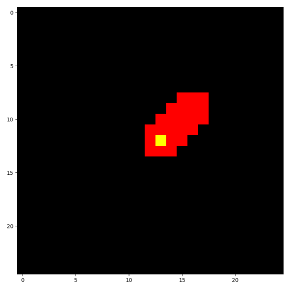
Figure 7. Agent 2 obstacle map in iteration 10

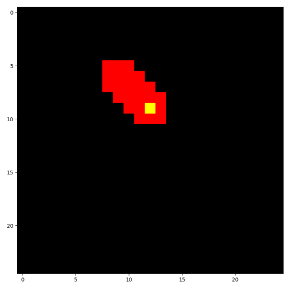
Figure 8. Agent 3 obstacle map in iteration 10

6. Fall back Goals

The implementation of obstacle avoidance can be more complex using the LIDAR data from the robots in Gazebo simulation environment. Hence, the assumption is that the robots will not collide with each other and there are no obstacles in the workspace. Another fallback goal is implementing the above methodology using a 2D visualization using OpenCV functions. Since there is a need to develop the algorithms, world map and visualization from scratch, an implementation of boids algorithm on a map that has no obstacles using a swarm of TurtleBot3 is proposed.

References

[1] V. P. Tran, M. A. Garratt, K. Kasmarik, and S. G. Anavatti, "Robust Multi-Robot Coverage of Unknown Environments using a Distributed Robot Swarm," CoRR, vol. abs/2111.14295, 2021, [Online]. Available: https://arxiv.org/abs/2111.14295

[2] T. Cieslewski, S. Lynen, M. Dymczyk, S. Magnenat and R. Siegwart, "Map API - scalable decentralized map building for robots," 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 2015, pp. 6241-6247, doi: 10.1109/ICRA.2015.7140075.

[3] Puente-Castro, A., Rivero, D., Pazos, A. et al. UAV swarm path planning with reinforcement learning for field prospecting. Appl Intell 52, 14101–14118 (2022). https://doi.org/10.1007/s10489-022-03254-4

[4] M. Zhong and C. G. Cassandras. Distributed coverage control and data collection with mobile sensor networks. IEEE Transactions on Automatic Control, 56(10):2445–2455, 2011.

[5] M. Masár. A biologically inspired swarm robot coordination algorithm for exploration and surveillance. In 2013 IEEE 17th International Conference on Intelligent Engineering Systems (INES), pages 271–275, 2013.

[6] K. Cheng, Y. Wang, and P. Dasgupta. Distributed area coverage using robot flocks. In 2009 World Congress on Nature Biologically Inspired Computing (NaBIC), pages 678–683, 2009.

[7] A. Topiwala, P. Inani, and A. Kathpal, "Frontier Based Exploration for Autonomous Robot," CoRR, vol. abs/1806.03581, 2018, [Online]. Available: http://arxiv.org/abs/1806.03581

[8] Reynolds, Craig W. "Flocks, herds and schools: A distributed behavioral model." Proceedings of the 14th annual conference on Computer graphics and interactive techniques. 1987.

[9] Bhattacharya, Priyadarshi, and Marina L. Gavrilova. "Roadmap-based path planning-using the voronoi diagram for a clearance-based shortest path." IEEE Robotics Automation Magazine 15.2 (2008): 58-66.

[10] Wang, Huijuan, Yuan Yu, and Quanbo Yuan. "Application of Dijkstra algorithm in robot path-planning." 2011 second international conference on mechanic automation and control engineering. IEEE, 2011.

[11] Karaman, Sertac, et al. "Anytime motion planning using the RRT." 2011 IEEE international conference on robotics and automation. IEEE, 2011.

[12] Virágh, Csaba, et al. "Flocking algorithm for autonomous flying robots." Bioinspiration biomimetics 9.2 (2014): 025012.

[13] Meerza, Syed Irfan Ali, Moinul Islam, and Md Mohiuddin Uzzal. "Optimal Path Planning Algorithm for Swarm of Robots Using Particle Swarm Optimization Technique." 2018 3rd International Conference on Information Technology, Information System and Electrical Engineering (ICITISEE). IEEE, 2018.

[14] Y. Zheng, L. Wang and P. Xi, "Improved Ant Colony Algorithm for Multi-Agent Path Planning in Dynamic Environment," 2018 International Conference on Sensing, Diagnostics, Prognostics, and Control (SDPC), Xi'an, China, 2018, pp. 732-737, doi: 10.1109/SDPC.2018.8664885.

[15] David Paez, Juan P. Romero, Brian Noriega, Gustavo A. Cardona, Juan M. Calderon, Distributed Particle Swarm Optimization for Multi-Robot System in Search and Rescue Operations, IFAC-PapersOnLine, Volume 54, Issue 4, 2021, Pages 1-6, ISSN 2405-8963, https://doi.org/10.1016/j.ifacol.2021.10.001.

[16] D. Albani, J. IJsselmuiden, R. Haken and V. Trianni, "Monitoring and mapping with robot swarms for agricultural applications," 2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Lecce, Italy, 2017, pp. 1-6, doi: 10.1109/AVSS.2017.8078478.

[17] D. Albani, J. IJsselmuiden, R. Haken and V. Trianni, "Monitoring and mapping with robot swarms for agricultural applications," 2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Lecce, Italy, 2017, pp. 1-6, doi: 10.1109/AVSS.2017.8078478.