Individual Lab Report 10

Team: H

Name: Steve Zhu

Date: 11/07/2019

Team Member: Akshit Gandhi
Shubham Garg
Parv Parkhiya

1. Individual Progress:

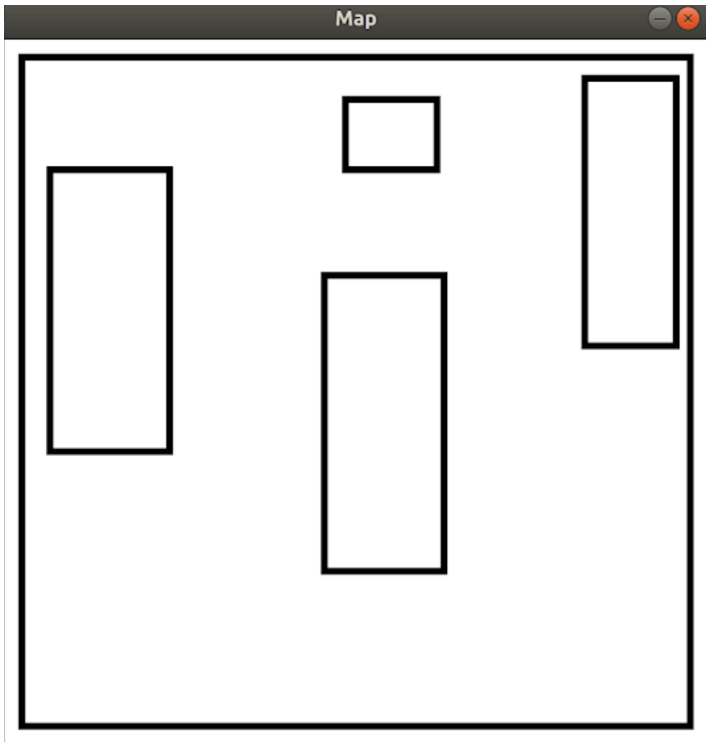
During the last two weeks, I mainly work on implementing the global planner for both UAV and UGV, as well as the water pump mounting.

1. Global Planner:

For the global planner, we assume that we mapped the environment abstractly: in the MBZ challenge's setting, as the whole surrounding will be pretty empty, and there will probably only be the base station and the destination building where the fire exist. So, we don't really need a complex global planner which frequently updates the map and the trajectory accordingly.

And in that setting, we know the GPS coordinates of the destination building as well the starting base station, we can simply use RRT or A* to generate the path.

But for the sake of making our robotic system seem to be more "pragmatic" in the real-life setting, we intentionally added several extra buildings besides of the starting and destination building as follows:



, where the left-most and right-most buildings are our "obstacle" buildings.

Our initial version of the global planner is based on RRT algorithm: basically, RRT is an algorithm designed to efficiently search nonconvex, high-dimensional spaces by randomly

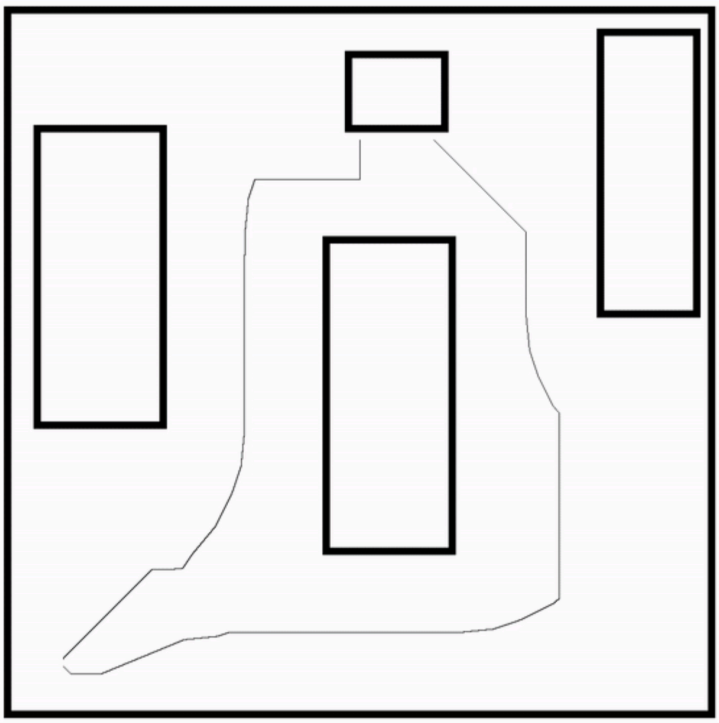
building a space-filling tree. The tree is constructed incrementally from samples drawn randomly from the search space and is inherently biased to grow towards large un-searched areas of the problem. While it seems to be working as it can indeed generate a reasonable and viable trajectory, the problem is that the generated trajectory has certain level of uncertainty: the path can be very long and winding sometimes.

In consideration of the whole robotic system, we find some problems if the trajectory generated might be too long: first, the battery of both Husky and UAV might be hugely consumed along the way if it takes a long distance for the robot to reach the destination. Secondly, if the generated path is too winding, as our whole Husky system carries large weight and the vertical center of weight is very high, around those very winding corners, the Husky might proceed very unstably.

Considering all the above, we finally decided to take the A* algorithm as our global planner. A* is a graph traversal and path search algorithm, and it is formulated in terms of weighted graphs: starting from a specific starting node of a graph, it aims to find a path to the given goal node having the smallest cost (least distance travelled, shortest time, etc.). It does this by maintaining a tree of paths originating at the start node and extending those paths one edge at a time until its termination criterion is satisfied.

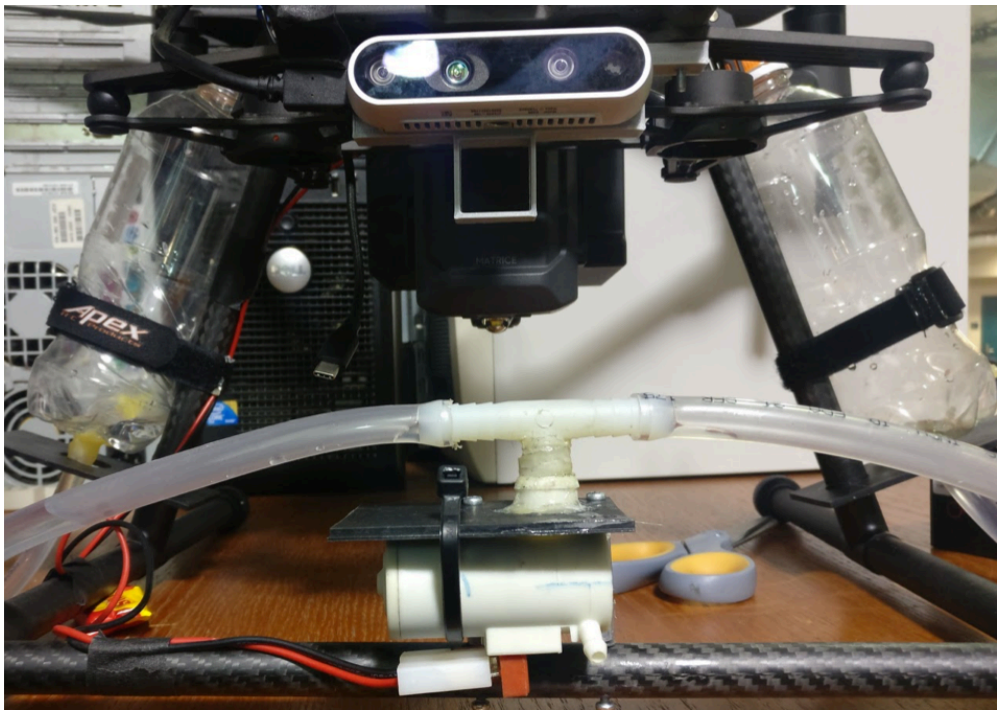
The A* algorithm can help find the possible optimal trajectory. Before applying the A* algorithm, we first generate the distance map for the given environment, where we used Euclidean distance, and the map generated looks like following:

Then, by applying the A* algorithm, we get a generated trajectory as following:



2. Water pump mounting:

We finished the water pump mounting for the UAV, as can be seem as below:



We have two bottles as water tank to reserve the water, and connects them through rubber pipe as one integral water tank, which is further connected with the water pump in front of the drone. The pump gets actuated once the thermal camera detects the fire (hot region in our test settings.).

2. Team Work

I worked with the team to implement the global planner where there are some hyper-paramerters that need to be tested and fine-tuned. And also when deployed on the UGV, initially there are some bugs where the planned trajectory can not be appropriately executed, and together fixed the bug. I also worked with the team to finalize the water pumping mechanism, including the mounting component design and building, as well as the later integration.

3. Challenge

The challenges we faced is first to understand how the behavior tree framework is organized, as none of the team members is familiar with that before. Then, after understanding it, we have to figure out how to integrate various ROS nodes by making them work compatibly with the behavior tree. Then, as we are using a new IMU sensor, which needs some fine-tuning on the fusion parameters, which takes us quite some time. Finally, we also have that water leakage issue on the extinguisher, and finally we fixed that by using a relay switch in front of the pump.

4. Future Work

1. Fix the power issue on UAV for controlling water pump
2. Full mission testing on UAV
3. Behavior tree for Husky missions
4. Full mission testing on Husky