AFRL-AFOSR-VA-TR-2019-0124

Improving Human Interaction with Robot Teams

Katia Sycara
CARNEGIE MELLON UNIVERSITY

04/24/2019
Final Report

Air Force Research Laboratory
AF Office Of Scientific Research (AFOSR)/ RTA2
Arlington, Virginia 22203
Air Force Materiel Command

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
01-05-2019	Final Performance	01 Dec 2017 to 30 Nov 2018

4. TITLE AND SUBTITLE
Improving Human Interaction with Robot Teams

5a. CONTRACT NUMBER

5b. GRANT NUMBER
FA9550-18-1-0043

5c. PROGRAM ELEMENT NUMBER
61102F

6. AUTHOR(S)
Katia Sycara

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
CARNEGIE MELLON UNIVERSITY
5000 FORBES AVENUE
PITTSBURGH, PA 15213-3815 US

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AF Office of Scientific Research
875 N. Randolph St. Room 3112
Arlington, VA 22203

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/AFOSR RTA2

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AFRL-AFOSR-VA-TR-2019-0124

12. DISTRIBUTION/AVAILABILITY STATEMENT
A DISTRIBUTION UNLIMITED: PB Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The goal of the award was the procurement of robotic equipment, UAVs and UGVs that collaborate with themselves and a human operator to execute critical missions. Exploration and navigation in unknown environments is a challenging task for ground rovers. While AGVs are often equipped with a variety of sensors that can detect the surrounding environment, the information that these sensors can provide are limited to the immediate surrounding of the AGVs and do not provide sufficient information about the obstacles and eventual destination of different paths. In these cases, an AGV without additional information would have to proceed down each path, prioritized by the best heuristic, in the hopes that no obstacles or dead end block the path.

In time-critical environments such as disaster response and relief, this trial-and-error approach would not suffice. Ground operators need to know which paths are safe and absent of hazards to deliver the required aid to trapped survivors as soon as possible. To enhance the exploration and navigation capabilities of autonomous ground vehicles, we propose FalconEye a heterogeneous mapping solution that combines the sensor input from UAV's to detect the open routes and obstacles that AGV's sensors cannot. Using airborne HD cameras and ground LiDAR sensors, FalconEye creates and operates within a 3D map whose range far exceeds the map created by ground-only robotic systems.

We designed, implemented and demonstrated in field experiments a system of cooperating UAV and UGV under human supervisory control for search and rescue.

15. SUBJECT TERMS
Swarms, Human Robot Interaction, Human Factors

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			RIECKEN, RICHARD
Unclassified	Unclassified	Unclassified	UU		19b. TELEPHONE NUMBER *(Include area code)* 703-941-1100

Final Report for Award FA9550-18-1-0043-DURIP

Title: Effective Human Teaming and Interaction with Multiple Unmanned Vehicles

Equipment Award

Period of Performance: 12/1/2017-30/11/2018

PI: Katia Sycara

Organization: Carnegie Mellon University

Summary

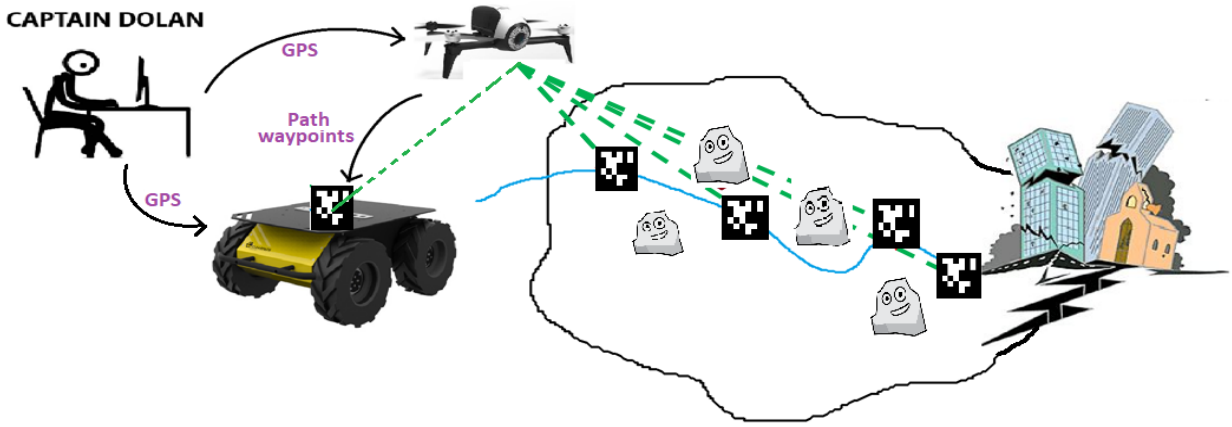
The goal of the award was the procurement of robotic equipment, UAVs and UGVs that collaborate with themselves and a human operator to execute critical missions. Exploration and navigation in unknown environments is a challenging task for ground rovers. While AGVs are often equipped with a variety of sensors that can detect the surrounding environment, the information that these sensors can provide are limited to the immediate surrounding of the AGVs and do not provide sufficient information about the obstacles and eventual destination of different paths. In these cases, an AGV without additional information would have to proceed down each path, prioritized by the best heuristic, in the hopes that no obstacles or dead end block the path.

In time-critical environments such as disaster response and relief, this trial-and-error approach would not suffice. Ground operators need to know which paths are safe and absent of hazards to deliver the required aid to trapped survivors as soon as possible. To enhance the exploration and navigation capabilities of autonomous ground vehicles, we propose FalconEye – a heterogeneous mapping solution that combines the sensor input from UAV's to detect the open routes and obstacles that AGV's sensors cannot. Using airborne HD cameras and ground LiDAR sensors, FalconEye creates and operates within a 3D map whose range far exceeds the map created by ground-only robotic systems.

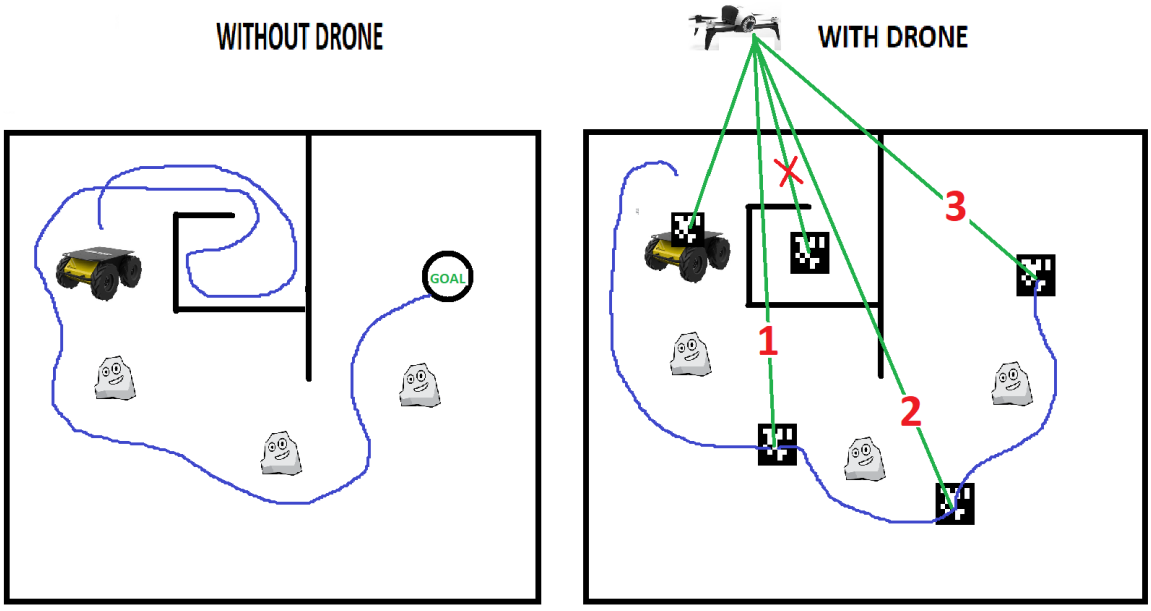
Use Case

In 2020, Pittsburgh is ravaged by a magnitude 3.0 earthquake and suffers heavy infrastructural damage to the buildings, pipelines, and electrical grid. Crumbled buildings, displaced streets and the resultant debris leave many locations inaccessible from the ground. Some of the buildings remain standing but rescue personnel are uncertain about the damage sustained by the infrastructure and the location of any safe access points. Satellite images lack the fidelity and the angle to show the areas rendered in accessible due to obstacles. Many people are trapped amidst the debris, time is of the essence, and disaster relief teams urgently need more information about the landscape of the disaster zones. The city of Pittsburgh dispatches Captain Dolan with FalconEye to explore and navigate the dispatches FalconEye, marking the center of the disaster zone as

the target destination. The UAVs take off and fly ahead of the AGV, gathering aerial information about the environment and stitching a 2D map. As the aerial map is formed, the UAV transmits this data to the AGV's, where the AGV localizes itself within this map and autonomously plans and follows a path forward that avoids the detected obstacles. By using both aerial and ground data, the AGV's are able to avoid significantly more ground obstructions and arrive at the target location faster than what would've been possible with just the AGV's.



Note: To simplify the problem, we will be using fiducial markers to represent valid nodes in the traversable path as well as the AGV. Below is a pictorial representation of how the system will behave with/without drone.

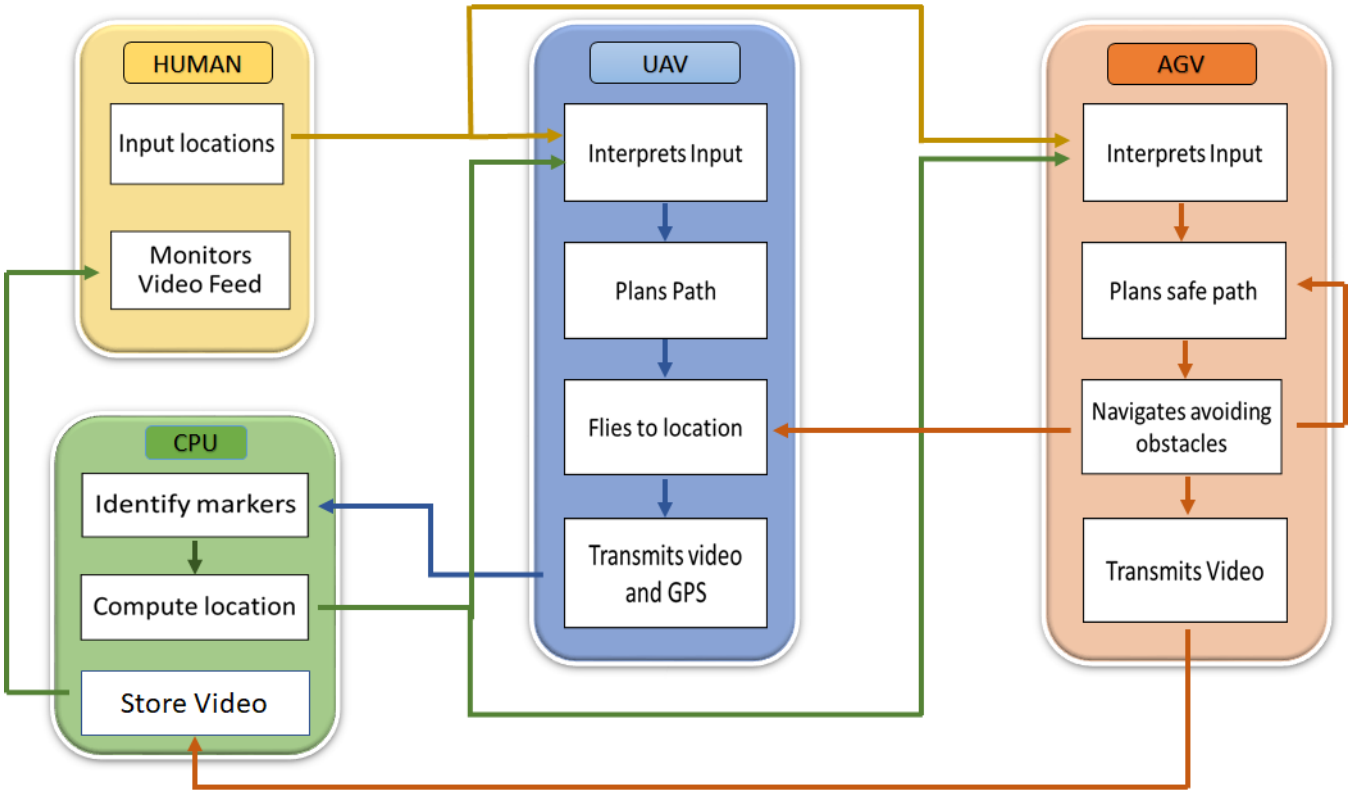


System Design

System Requirements

- The UAV should detect fiducial markers at a height of 5m (detection rate > 80%).
- The UAV should localize the fiducial markers with respect to one another accurate (error between fiducial locations < 30 cm).
- The UAV should fly to a target GPS location (within a 5m radius).
- The AGV should avoid ground obstacles (success rate > 80%).
- In a 50x50m field, the AGV should reach the target location quickly (< 15 min).
- In a 50x50m field, the AGV should reach the target location accurately (< 3 m radius).
- In a 50x50m field, the AGV should reach the target location successfully (>80%).

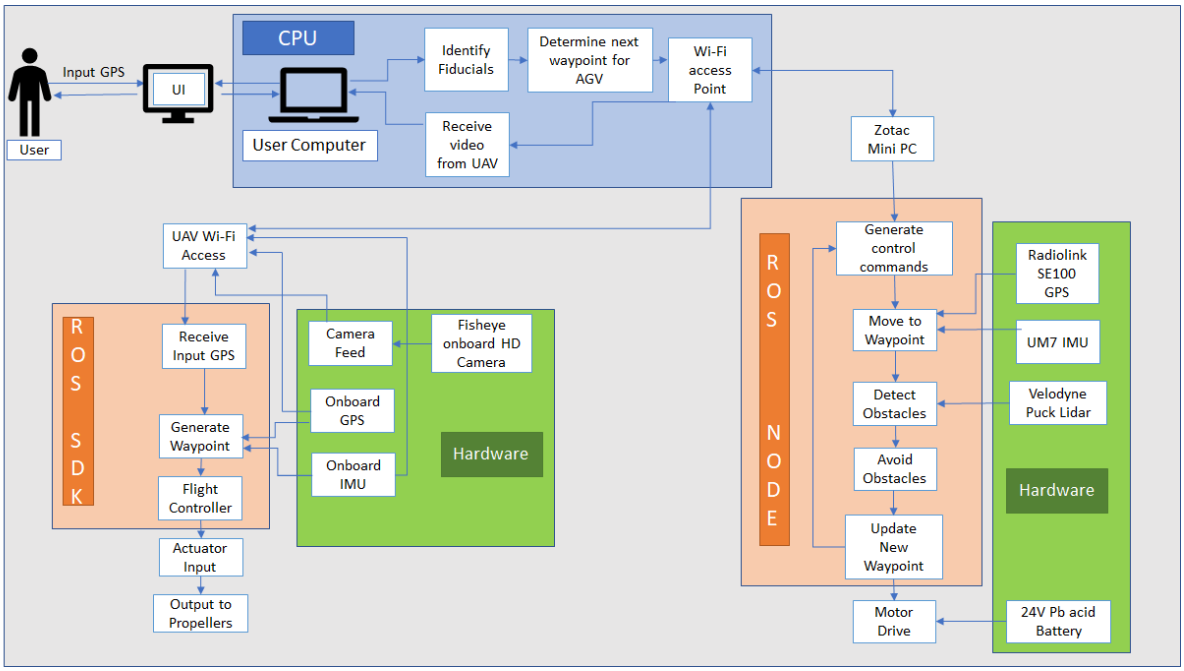
Functional Architecture



Our functional architecture can be divided into three major functions: the UAV perceives the environment, the CPU identifies the obstacles and plans the path for the AGV and the UAV, and the AGV navigates to the target destination. Each function can be further explored in more detail. The UAV needs to simultaneously capture a birds-eye video view for the CPU while also exploring the environment for the next fiducial marker. The CPU acts as the bridge between the AGV and the UAV. Using the video feed from the UAV, it generates an internal map of the

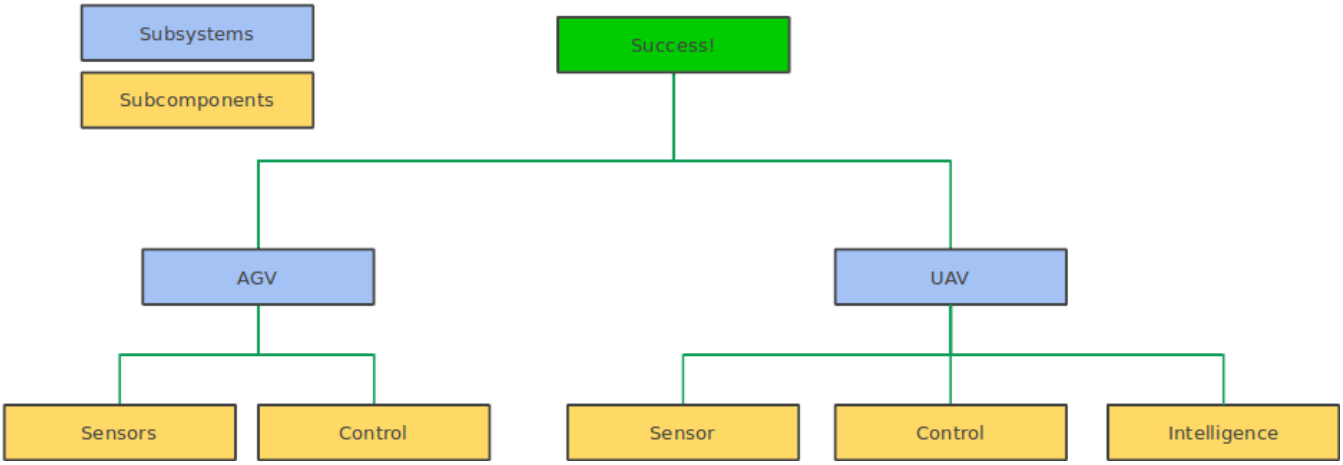
different fiducial nodes, determines which nodes comprise a traversable path, and forward this information to the AGV. It simultaneously provides the map back to the UAV so that the UAV can explore the most optimal areas. The AGV takes in the higher level commands from the CPU and travels to the target location. It uses its own onboard sensors to perform obstacle avoidance on static objects.

Cyberphysical Architecture



Our cyberphysical architecture shines some more light into the exact implementation of our system. At the highest level, we have 3 different nodes that represent the individual agents in our system. The UAV (Bebop 2 Drone) is represented by the ROS SDK, the CPU is the central command center of our system, and the AGV (Clearpath Husky) is represented by the ROS Node. The entire system is distributed over a ROS WiFi network where the AGV, the UAV, and the CPU are all on different computers. It is important to note that for both the AGV and the UAV, we do not have direct control and instead we interact via a ROS API. Similar to the functional architecture, the CPU acts as the main computational unit of our system. It generates an internal map with the video feed from the Bebop 2, localizes the UAV, the AGV, and the traversable paths, and provides the target locations to both the UAV and the AGV. The ROS SDK node converts the target location into roll/pitch/yaw commands and feeds the video to the CPU. The ROS node's main responsibility is lower-level path planning: specifically obstacle avoidance.

Subsystems



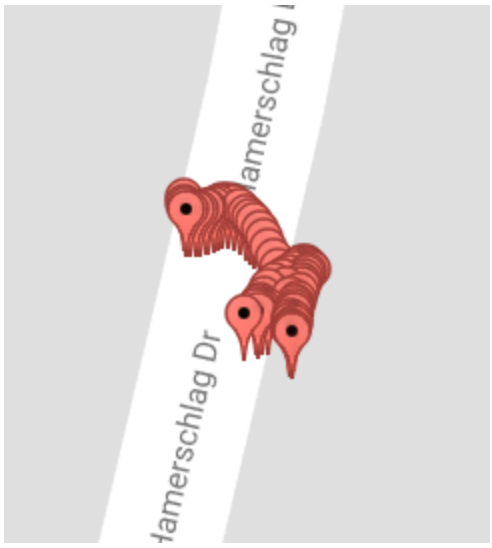
We have divided our entire system into 5 major subcomponents. For the AGV, we have sensing with the GPS and the Velodyne Puck and control for obstacle avoidance and navigation. For the UAV, we have sensing with the Bebop 2 Camera and GPS, control for GPS waypoint navigation, and intelligence for sensor fusion and an exploration algorithm to find the next fiducial marker.

AGV Sensors: The AGV is equipped with enough sensors such that it can navigate with or without AGV assistance (this is for comparative purposes). As a result, it combines GPS (SE100) data and IMU with point clouds from the Velodyne Puck to localize and avoid obstacles. The main considerations in the design of this subsystem are the types of sensors to be used. GPS was required because the CPU sends GPS coordinates as the target location. The Velodyne Puck was originally not part of our design choice as it comes with a non-trivial price tag and thus we were originally considering a stereo sensor such as the ZED camera. However, Captain Dolan has been able to procure two Pucks from Velodyne and has been gracious enough to lend us one.

Sensors and mount for the Husky.

UAV Sensors: The UAV sensors provide the aerial data feed necessary to help the AGV localize itself and detect traversable paths that are beyond its field of view. The choice of UAV sensors is dictated by the choice of UAV; as we have decided to use the Bebop 2, we have access to a 720p fisheye lens with auto stabilization and a GPS module. The Bebop 2 was chosen for 3 main reasons. First, the Bebop 2 has excellent developer support and ROS integration. It has an SDK that is available and because the official app was built with the SDK, it can be said that the developers have full control over the drone. Second, the Bebop 2 is cheap (~300) and thus we justified it as a prototyping platform that we can replace if it breaks. Third, the drone had already been ordered by our advisor and thus it was the fastest way for us to gain access to a development platform. The video feed is first fed through a filter that removes noise, then it is picked up by the April Tags ROS node. The tf output is passed through a low-pass filter to stabilize the reading before it is published as individual locations with respect to the home frame. The GPS reading from the drone is also passed through a low-pass filter and it is combined with the base link frame to get the GPS coordinates of all the fiducial markers.

Sensors on UAV

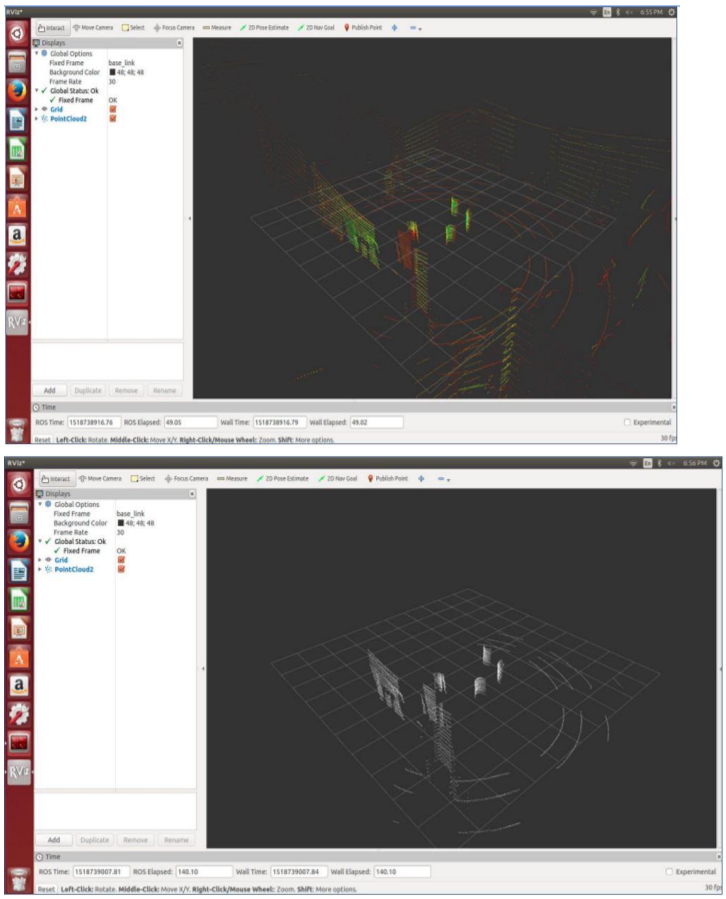


Drift of Bebop 2 GPS over 1 minute

AGV Control:

a) AGV Obstacle Detection

The AGV generates environment map using a LIDAR sensor. A LIDAR usually consists of a scanner and a laser. The laser emits a laser beam at a certain frequency and also receives the reflected laser beam from an object on the path of the laser. This can be used to estimate the distance between the LIDAR sensor and the object by estimating the time difference between the transmitted and received laser. This method is a very reliable method of estimating the distance up to a certain limit depending upon the LIDAR specifications. We have selected Velodyne VLP 16 sensor due to its excellent online support and reliability. As shown in the figure given below, first one depicts the raw point cloud which we get from the Velodyne, then using the PCL library, we filtered out the obstacles of our interest i.e, which are greater than a certain threshold height and which are in the proximity of the AGV, which is shown in the second figure.

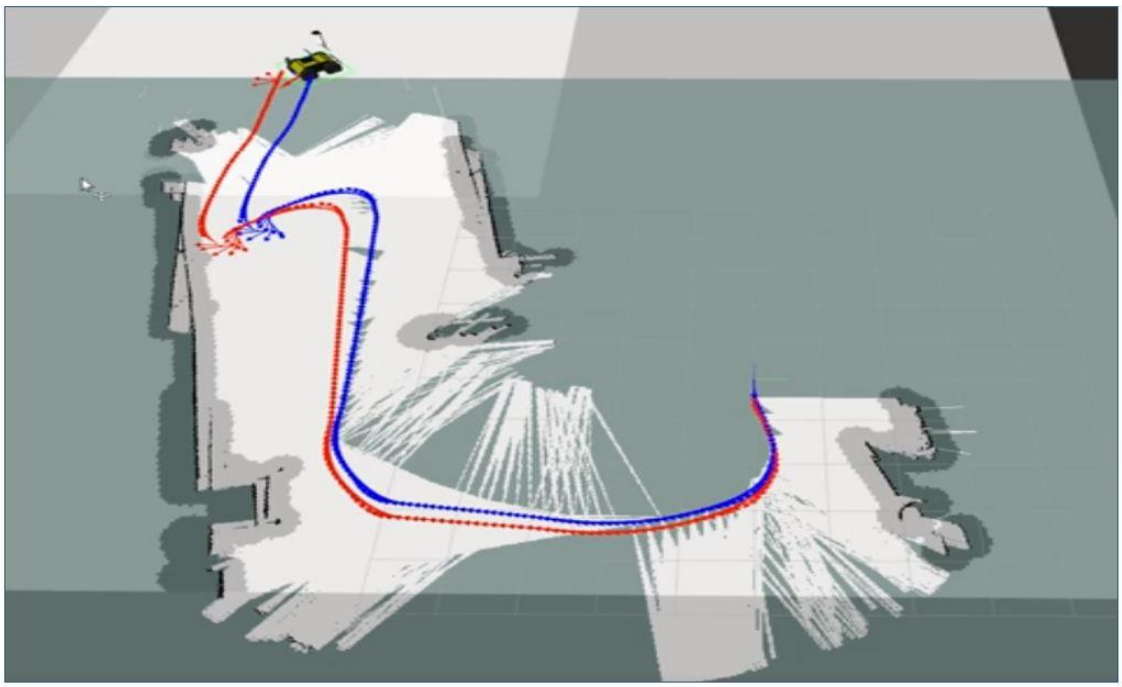


Velodyne raw and filtered outputs

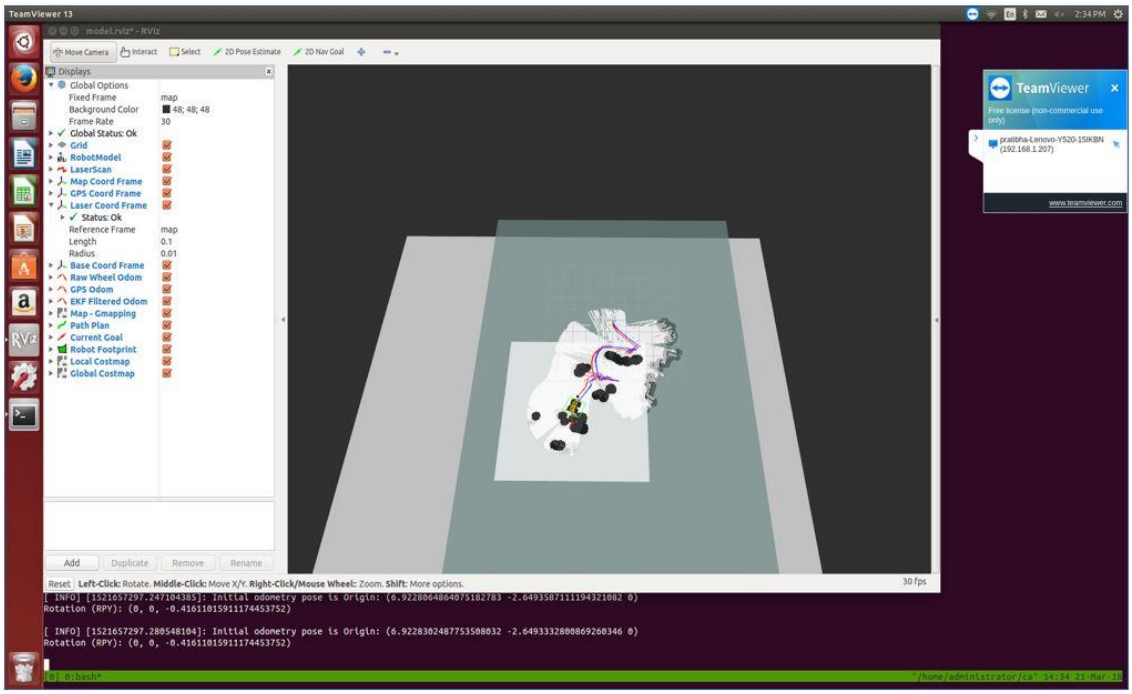
b) Path Planning and Obstacle Avoidance

We used ROS navigation stack for optimized localized-navigation. It uses Search based planning algorithm like A* and has proven to be very effective for motion planning. We integrated GPS and IMU data with this stack for global localization and planning. In addition to this augmenting this system with the input from the Velodyne, helped us in developing a complete obstacle detection and avoidance system. We were able to demonstrate that our system is able to detect and avoid, static as well as dynamic obstacles.

For the global level of path planning our final system was able to record the GPS location given by the UAV on the waypoint navigation text file. The husky navigation stack was reading that file sequentially and executing the path goals provided by the UAV. Two figures given below show the husky navigation with and without obstacle avoidance.



Husky Path Planning without Obsctacles



Husky Path Planning with Obstacle Avoidance

UAV Control:

UAV control is the lower level controller for the Bebop 2 drone. This controller is in charge of navigating the Bebop 2 toward a GPS location. For this, we implemented our own custom controller that first orients the drone into the correct heading and moves forward. Although simple in theory, we also added an exponentially decaying scale factor for the throttle such that the movements would be smoother. Additionally, we added a dynamically adjustable height controller to ensure that the drone flew at 10m. Although the odometry may drift over time, we correct this drift via true readings from the April Tag detections. The orientation and translation commands are shown in a diagram below.

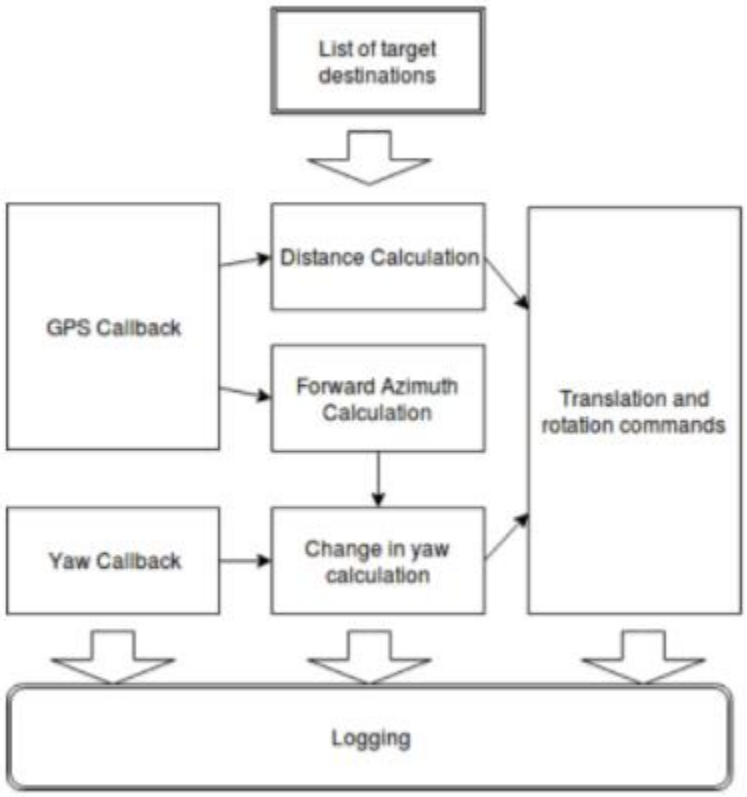


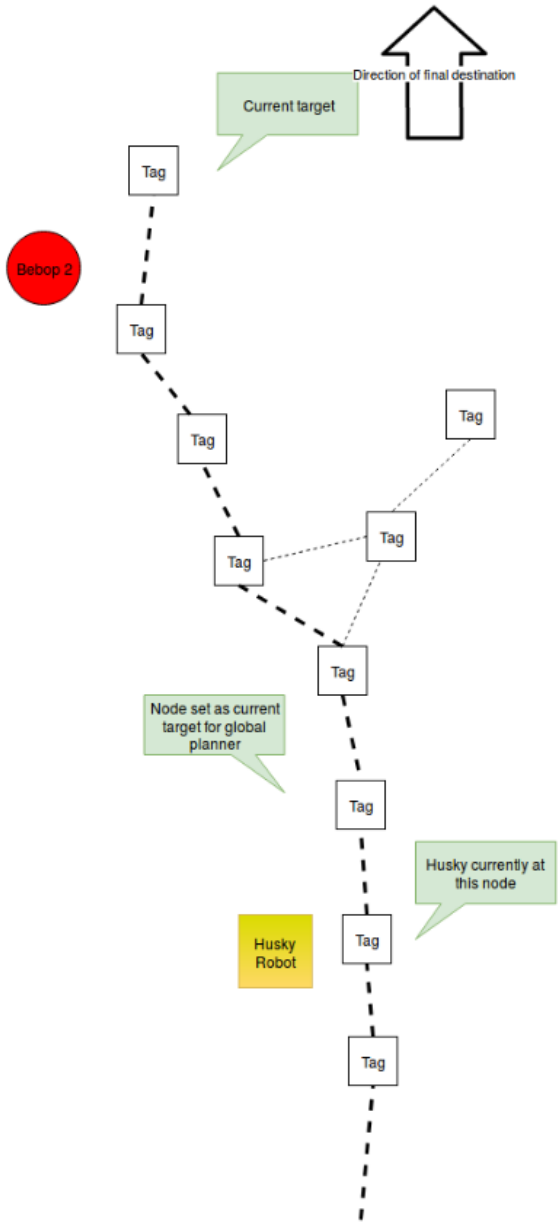
Figure 1: Schematic of controller

Fig. 1: Schematic of GPS controller

UAV Intelligence:

The UAV Intelligence subsystem the subsystem that is responsible for the exploration as well as the pathfinding for both the UAV and the AGV. The reason why it is considered to the UAV Intelligence is due to its implementation. Internally, we represent all the April Tags as vertices in a graph. These vertices are connected by an edge if the vertices are less than 3 meters apart. Vertices represent non-obstructed areas in the world and the edges represent valid paths between these areas. Both explorations for the UAV and global path planning for

the AGV are built on top of this graph representation. Exploration for the UAV is an iterative process whereby the closest **unexplored** April Tag vertex and the surrounding area is visited (marked as explored). The AGV is bounded by the vertices of the graph and uses an A-star algorithm to traverse through the environment. A high-level diagram of this process is shown below.



System Design

Heterogeneous Exploration of Unknown Environments

- In unknown environments, distant obstacles and paths that lead to dead ends are often hard to detect and plan around for AGV sensors.
- We propose a heterogeneous mapping solution that leverages the wide area coverage of UAV to enhance the mapping and localization capabilities of AGV.
- The UAV will fly ahead of the AGV, mapping the environment in front of the AGV, determining obstacles and viable routes, and transmitting this information to the AGV.
- The AGV will incorporate the information from the UAV with its own GPS and LiDAR sensors to localize more accurately and plan a more effective path forward.

Subsystem Details

1. AGV Subsystem

1. GPS based waypoint navigation:

This counts as the basis for the development of navigational capabilities of the AGV. We decided to work on GPS based navigation for the fall semester and extend the navigational capabilities of the AGV by including obstacle avoidance using a LiDAR in to the system. Currently, the AGV has been able to navigate to a given GPS location within a maximum inaccuracy of 3m. For the fall validation test, we gave three waypoints as the navigation goals and the Husky was able to reach all three of them within 3m of the given GPS locations. For the SVE, GPS navigation was removed on the AGV as a more complicated local planner was implemented.

For autonomous navigation, we developed our own GPS waypoint based controller in ROS. The controller reads the target locations one by one from the launch file as a ROS parameter, computes the distance between the current GPS location and target location, orients the AGV towards the target GPS location with the help of an Inertial Measurement Unit(IMU) and then actuates the motor until the distance between current and target location comes under some threshold value. This is repeated for all GPS coordinates saved in the launch file. After reaching the final location, the system enters idle mode.

The schematic of the controller is shown in Figure 1:

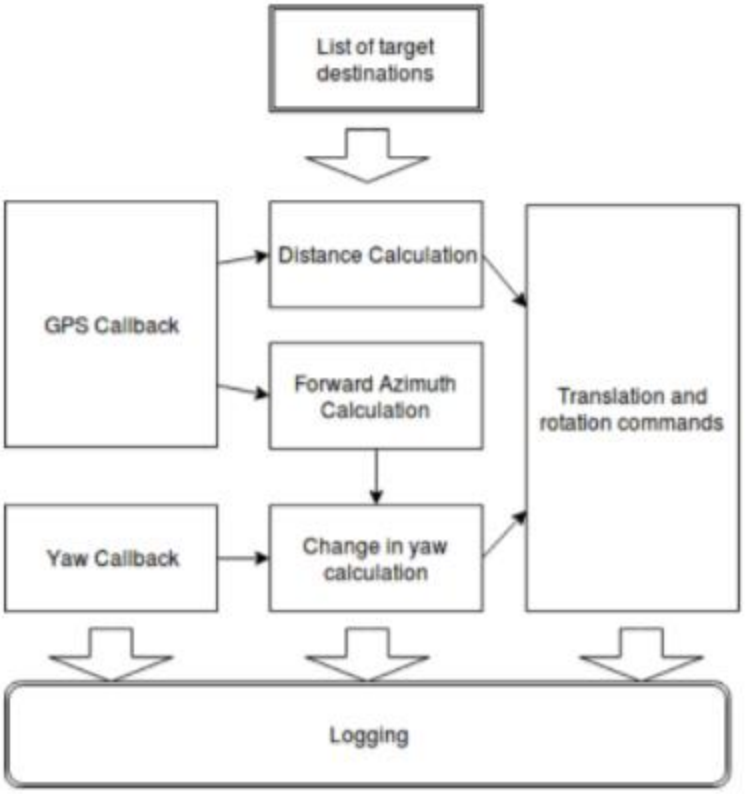


Figure 1: Schematic of controller

2. Mechanical Fabrication

We designed and fabricated specific mounts for the sensors and takeoff platform for the UAV. Figure 2 shows the CAD model for our system.

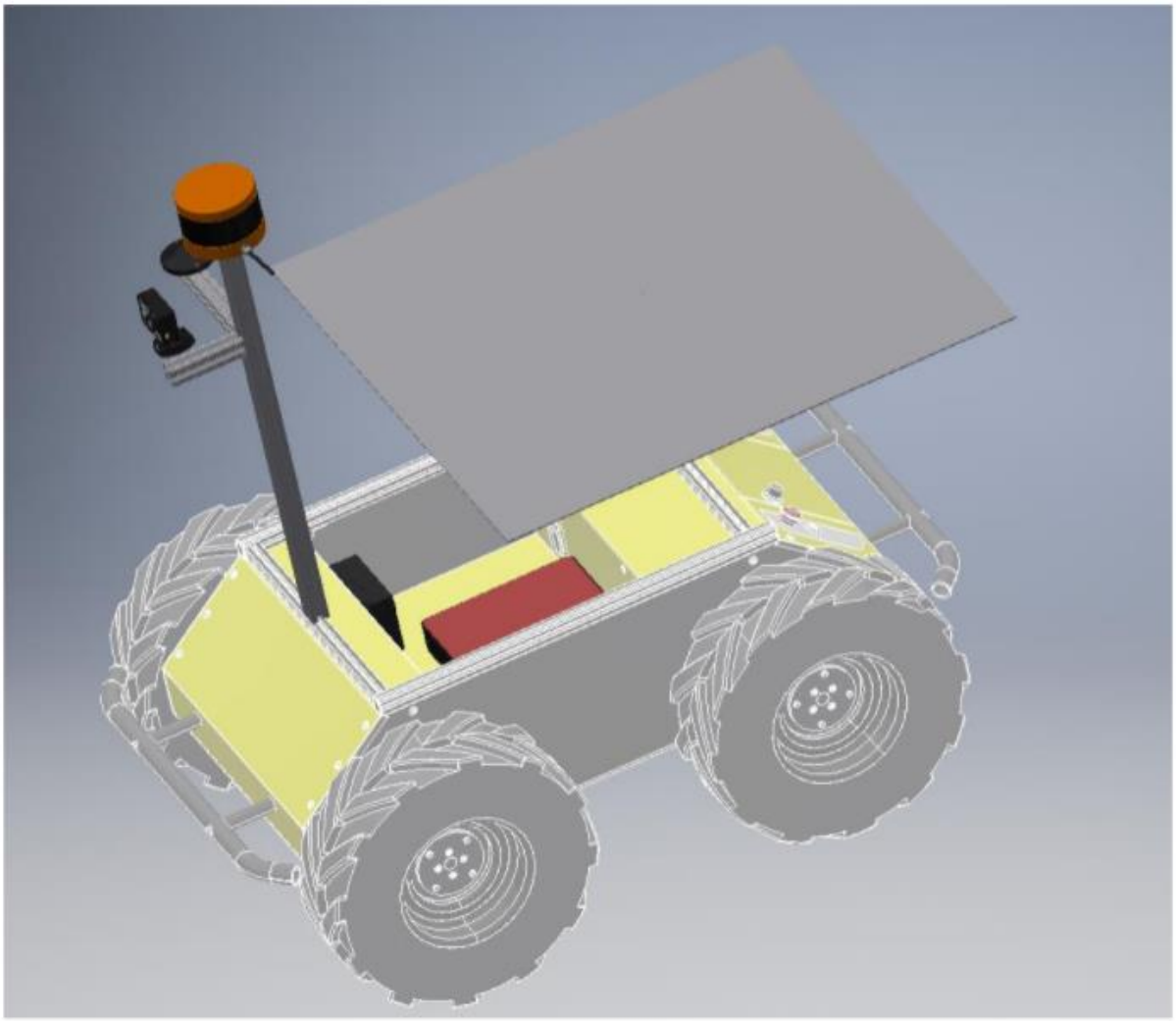
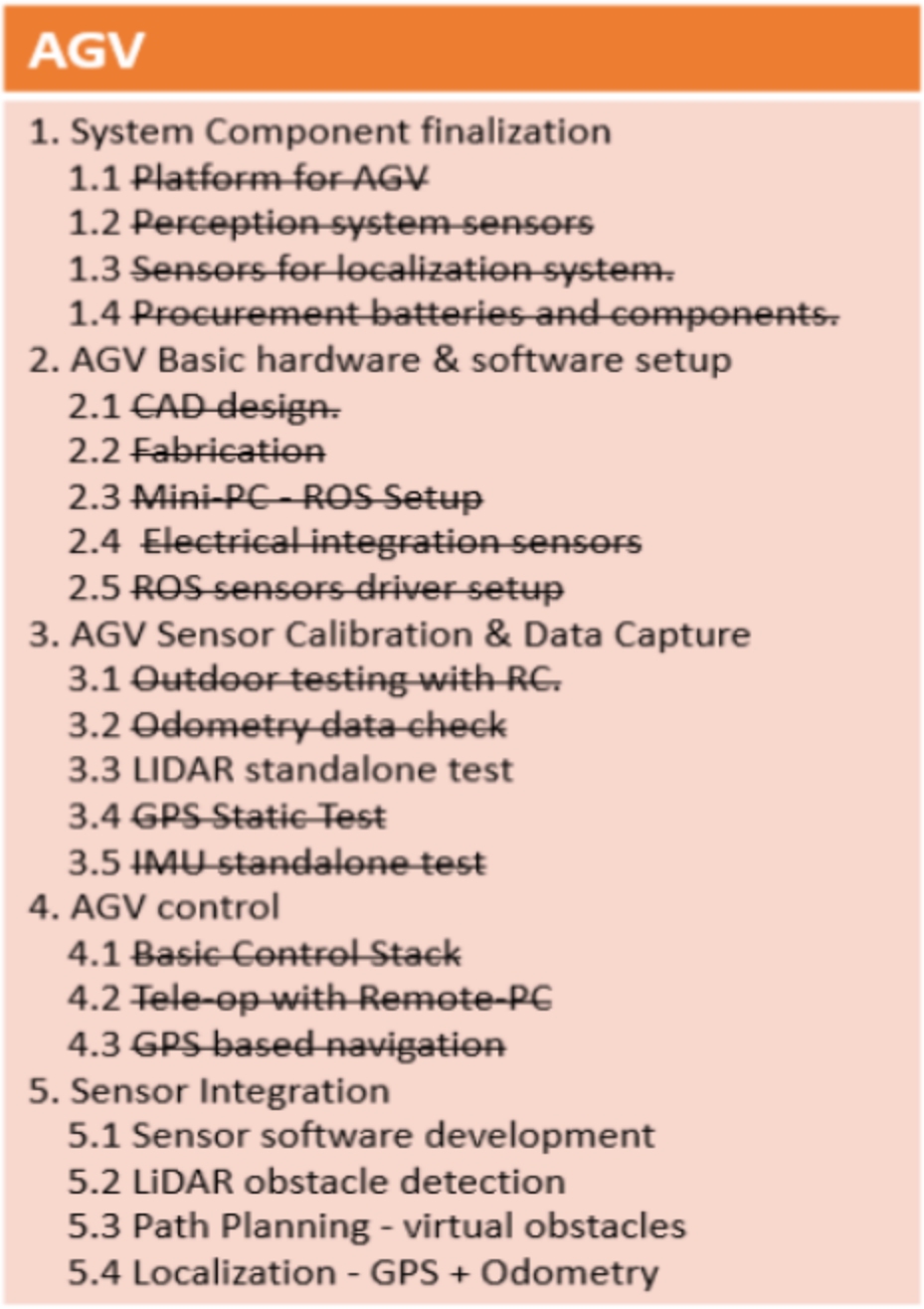


Fig. 2: Mechanical mounts CAD



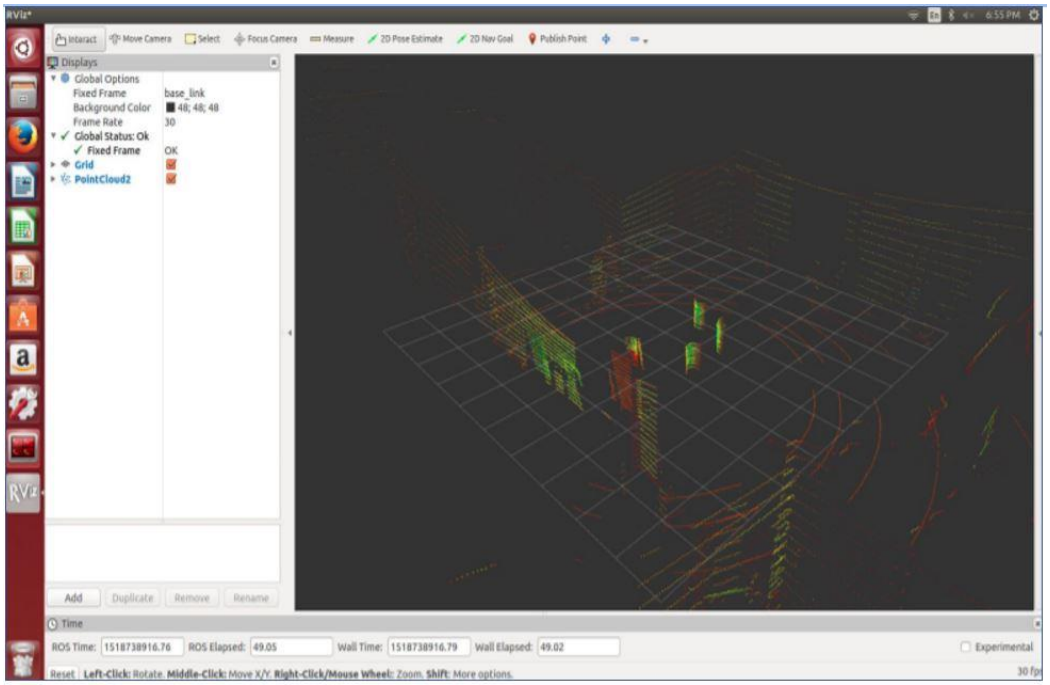
Overall AGV Status

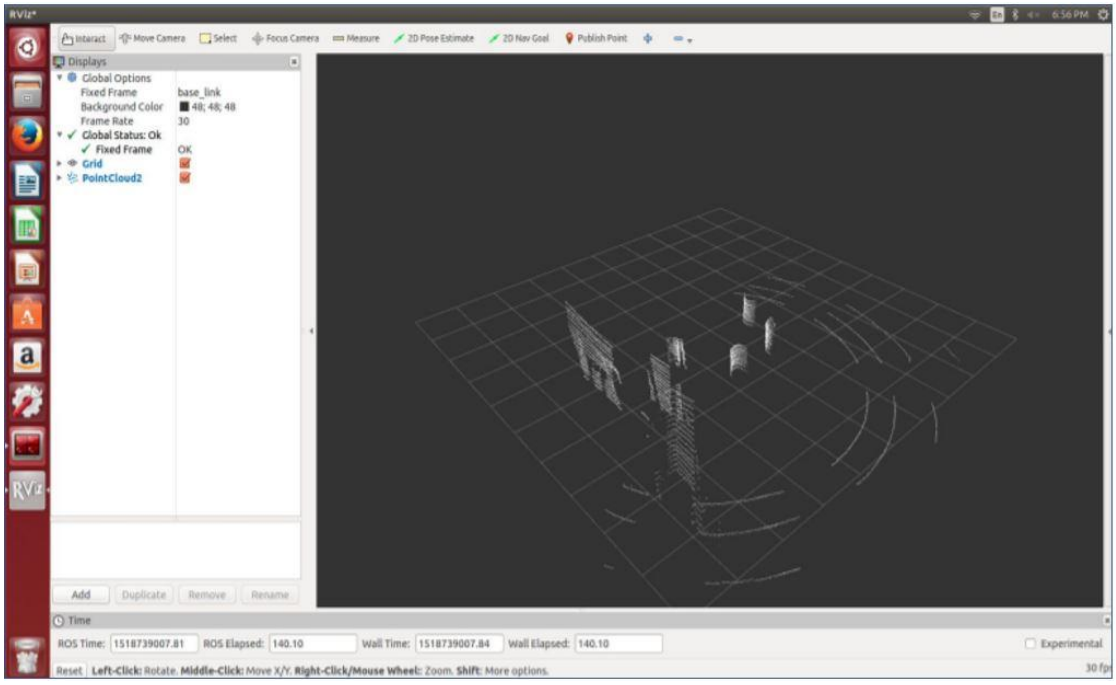
3. AGV Path Planning and Obstacle Avoidance

a) AGV Obstacle Detection

The AGV generates environment map using a LIDAR sensor. A LIDAR usually consists of a scanner and a laser. The laser emits a laser beam at a certain frequency and also receives the reflected laser beam from an object on the path of the laser. This can be used to estimate the distance between the LIDAR sensor and the object by estimating the time difference between the transmitted

and received laser. This method is a very reliable method of estimating the distance up to a certain limit depending upon the LIDAR specifications. We have selected Velodyne VLP 16 sensor due to its excellent online support and reliability. As shown in the figure given below, first one depicts the raw point cloud which we get from the Velodyne, then using the PCL library, we filtered out the obstacles of our interest i.e, which are greater than a certain threshold height and which are in the proximity of the AGV, which is shown in the second figure.

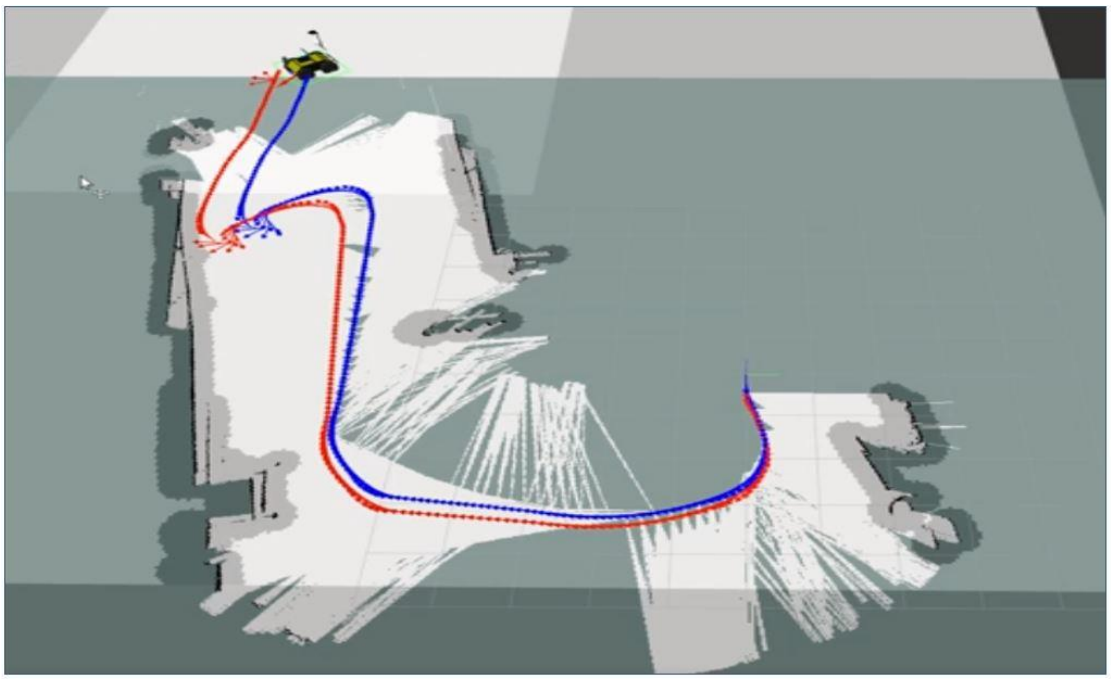


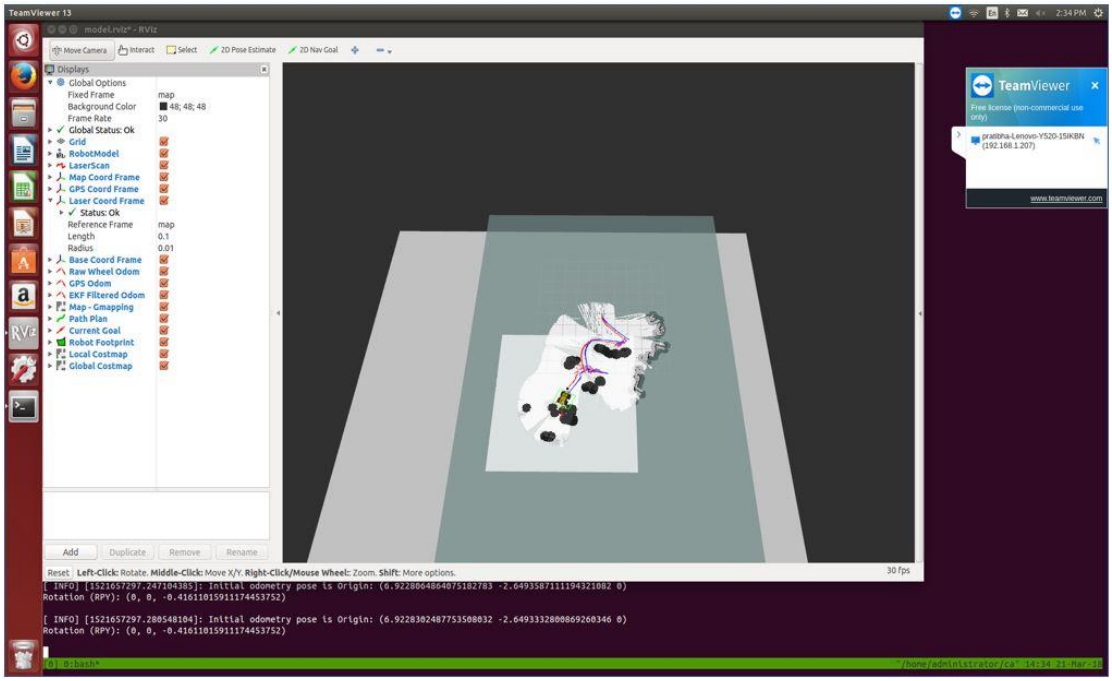


b) Path Planning and Obstacle Avoidance

We used ROS navigation stack for optimized localized-navigation. It uses Search based planning algorithm like A* and has proven to be very effective for motion planning. We integrated GPS and IMU data with this stack for global localization and planning. In addition to this augmenting this system with the input from the Velodyne, helped us in developing a complete obstacle detection and avoidance system. We were able to demonstrate that our system is able to detect and avoid, static as well as dynamic obstacles.

For the global level of path planning our final system was able to record the GPS location given by the UAV on the waypoint navigation text file. The husky navigation stack was reading that file sequentially and executing the path goals provided by the UAV. Two figures given below show the husky navigation with and without obstacle avoidance.





2. UAV Subsystem

1. GPS based waypoint navigation:

Similar to the AGV navigation, we developed our own controller for autonomous navigation of the UAV. Currently, the UAV has been able to navigate to a given GPS location within a maximum inaccuracy of 5m. For the fall validation test, we

gave three waypoints as the navigation goals and the bebop was able to reach all three of them within 5m of the given locations.

Figure 3 Shows the controller design for autonomous navigation. Figure 5 depicts the UAV performing autonomous navigation.

Fig. 3: Controller Design of UAV

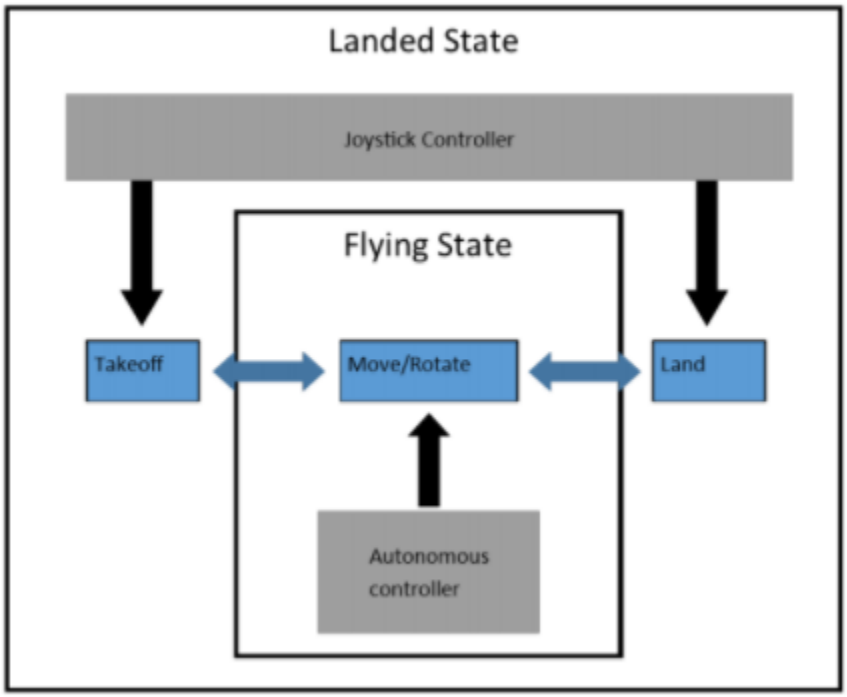
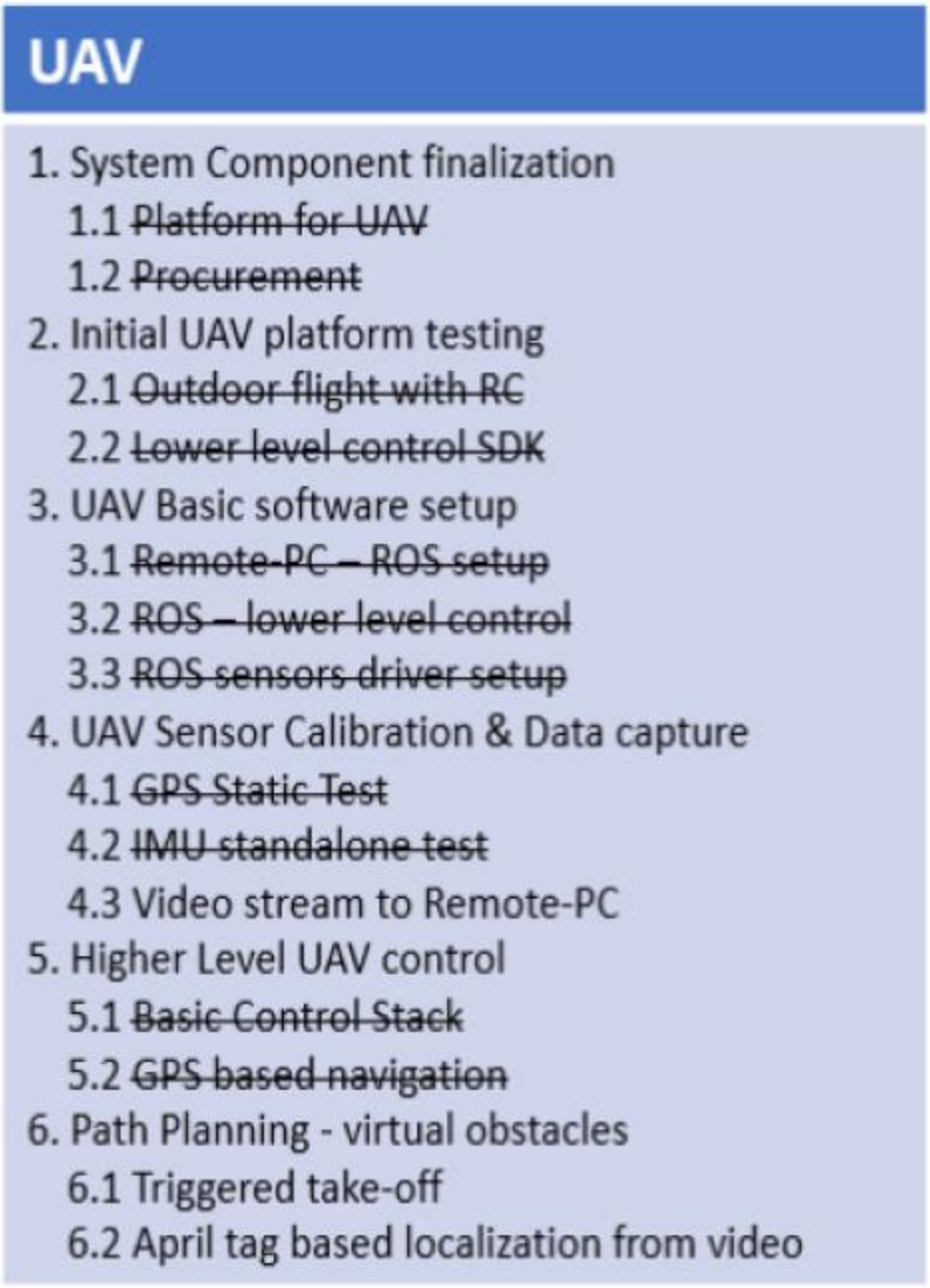


Fig. 4: UAV executing autonomous waypoint navigation.



Overall UAV Status

3. CPU subsystem

1. April tag detection:

To determine the traversable paths for the Husky it is essential we are able to recognize those paths through April Tags from UAV's camera feed. We are able to successfully detect all the april tags in the range of the drone's camera. The

fall validation testing was done with 10 april tags placed within a 2m radius circle and drone hovering at a height of 5m. The detection was shown by overlaying red circles over the april tags and also by displaying them in rviz.

Fig. 5: April Tags laid out for testing in a circle of radius 2m

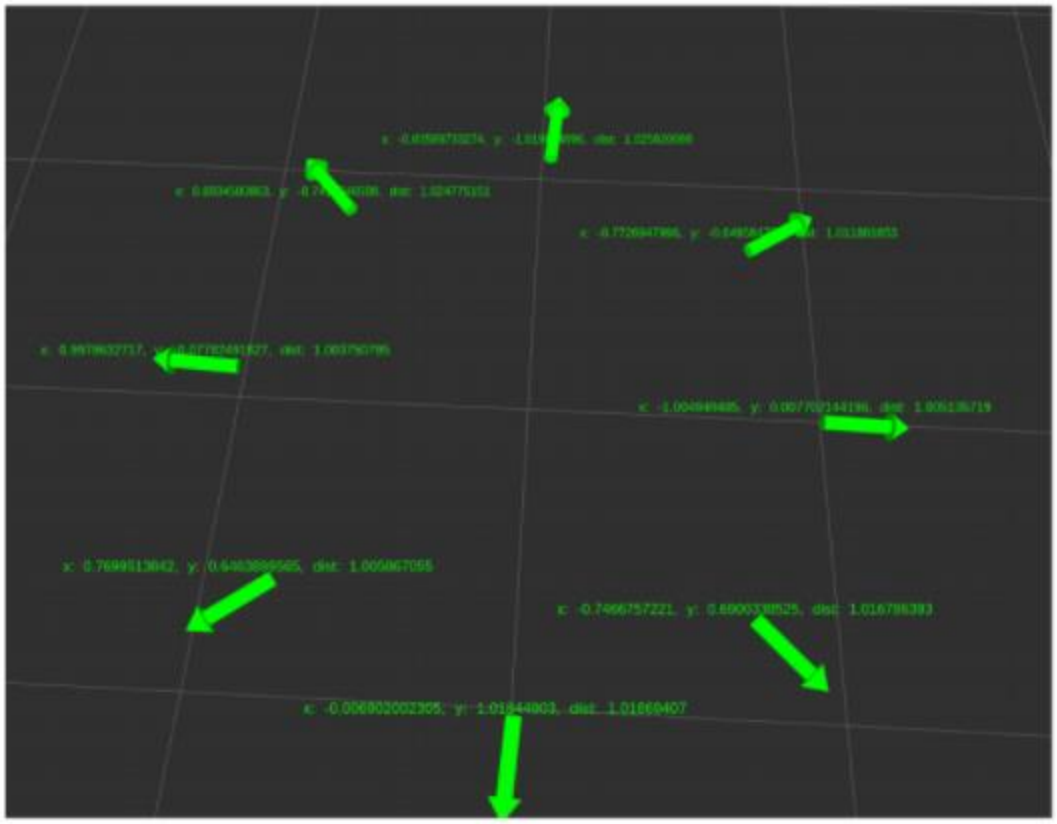


Fig. 6: RVIZ visualization

2. Localization of April Tags with respect to a home frame:

We are able to localize april tags with respect to each other with a maximum inaccuracy of 10cm. This is much better than our performance parameter which was 30cm. This test was done with the drone hovering at a height of 5m. The distance was shown by manual measurement between the tag centers in comparison with the computed distance in rviz.

4. Power Distribution Board

We have designed and developed a power distribution board for our system. The board provides Stable voltages to run two 5V devices (GPS and IMU), one 12V Wi Router and a 24V Mini-PC.We are getting nominal drift of 6% from the specied voltages that lies within the operating ranges of the devices we will be using. Since we already have stable DC output from the built-in DC-DC converters in AGV with less than 2% drift, we plan to use our power distribution board as a backup and for indoor testing.
Figures 7 and 8 show the schematic and photo of the PDB during Fall Validation Experiment.

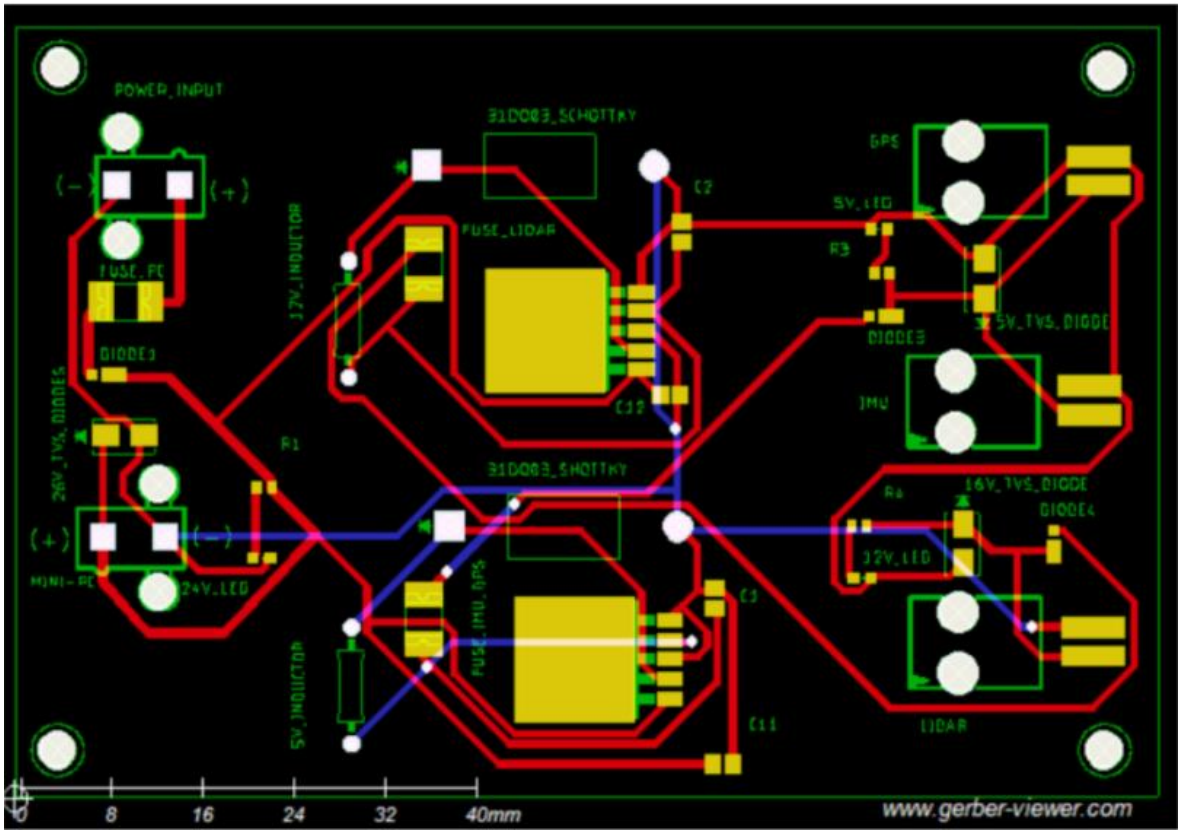


Fig. 7: PDB Layout

5. April Tag Graph

After detection of the April Tags, we use an internal graph representation as the map for the environment. Specifically, based on the locations of the April Tags, we have a graph of valid “nodes” that represent areas where there are no obstacles. These are key locations that the Husky can travel to. The edges between the nodes are paths that allow the Husky to move from one location to another. As the drone explores further, certain branches will expand more while others are dead ends that get pruned. Below is a visualization of a simple graph that is constructed in RVIZ with tf. Much more complex graphs are present for the SVE.

Figure 1: Panoramic view of April Tags

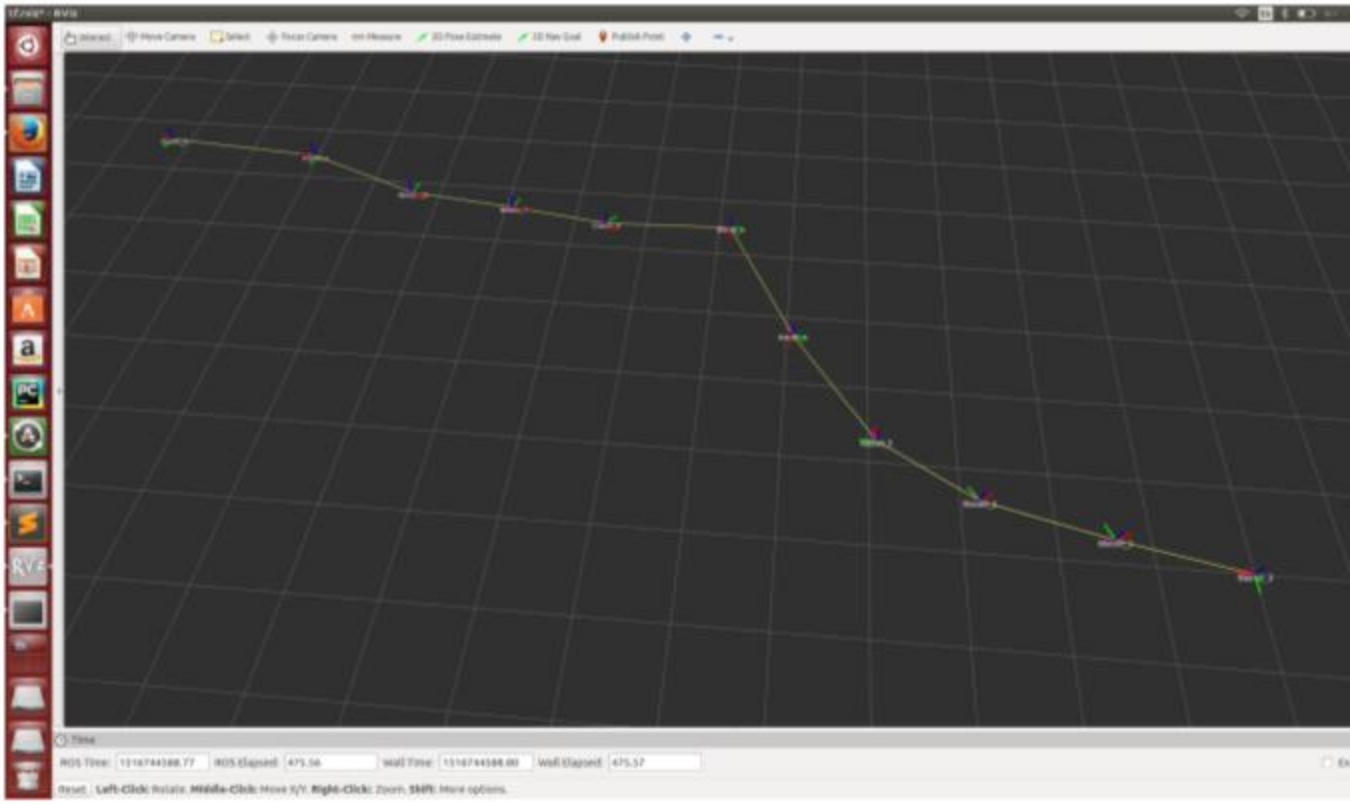


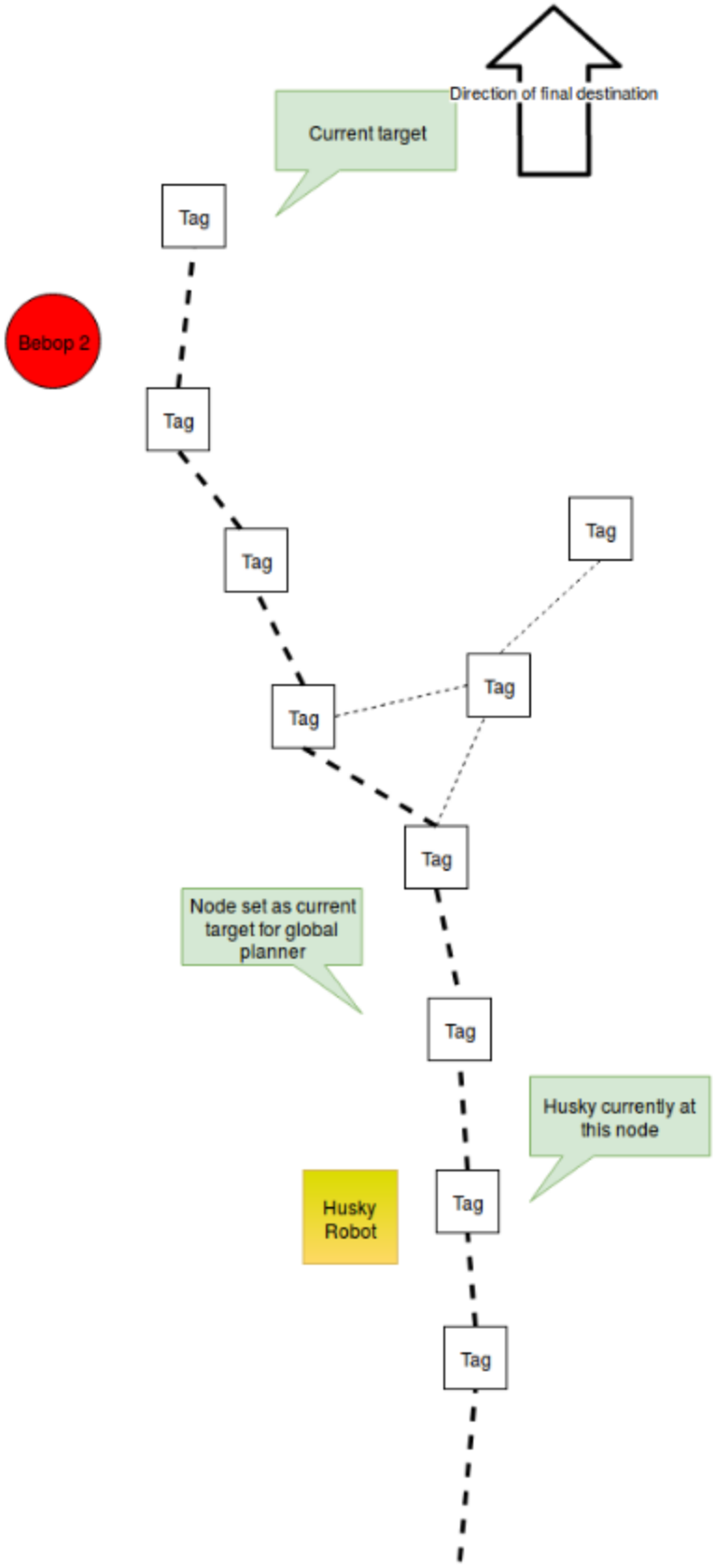
Figure 2: TF representation of April Tags

5. Exploration and path-finding algorithm

Based on the constructed graph, the UAV explores unseen areas of the map to find a valid path to the goal. It does so by finding the closest unexplored April Tag (node) to the target destination and exploring around that area. Based on our specific environment, we can guarantee that any successive April Tag will be within 3 meters to the previous one. At a 10m hovering height, our algorithm is extremely reliable at finding a path if a path exists.

The UAV is not constrained by ground obstacles and so it can directly fly over to any node that it wishes to explore. The same cannot be said for the Husky which must navigate along the vertices of the graph. Based on our graph, finding this path can be solved by a graph exploration algorithm such as A-star. This is exactly what we did. We mark the node closest to the Husky as the starting node, the node closest to the target as the goal node, and we run A-star on the graph with distance as the cost. A high-level depiction of the entire system algorithm is shown below.

The following figure shows a screenshot of the video where the UAV and the UGV collaborated for better path finding.



System Performance

FVE Performance

FVE Subsystem Performance

UAV Subsystem

1. We were able to successfully complete the GPS Waypoint Navigation of the UAV. We were well within the tolerance range of +/-5m.
2. We were able to detect 10 April tags from a height of 5m, with the accuracy of 100%.
3. We were able to localize two April tags with respect to each other. The actual distance between the April tags – 2m, Our reading of the distance 1.95m.

AGV Subsystem

1. We were able to successfully complete the GPS Waypoint Navigation of the UAV. We were well within the tolerance range of +/-5m.
2. We reduced the tolerance for FVE Encore to +/-3m and still, we were well within the range in that experiment too.

FVE Conclusions

Strengths:

- Robust GPS waypoint navigation capabilities of UAV and AGV.
- Stable April Tag Detection from a height of <=5m.
- Accurate April Tag Localization (much within the specified threshold).
- Easy to setup system.
- Great understanding and collaboration among team members.

Weakness:

- Low flying ceiling for the UAV.
- Standalone IMU integration
- Testing at night and in bad weather.

Areas of Improvements:

- Improvement in safety measures while demonstrating and testing subsystems.
- Better introductions to various subsystems to be tested.
- Increase UAV flying height with due considerations to the safe ceiling.
- Using better material for April Tags printing, to improve the detection.

- Replacing phone IMU with a better low-cost IMU.

SVE Performance

Our SVE metrics consisted of the following goals:

- AGV arrives within 3 m of final destination
- AGV navigates the path in less than 10 mins
- AGV avoids 100% of static obstacles.
- UAV-AGV system navigates faster than the AGV-alone system.

During the SVE, we have been able to achieve all of our targeted requirements. Thus, at a system-level, our design and implementation were successful with regards to our metrics. However, there are elements of our system that can be improved. They relate to the underlying sub-systems and we saw during the SVE that these areas could perform better.

Strong Points

1. Robust area exploration algorithm for the UAV. The UAV will always explore around the nodes to find the next April Tag and prune all dead ends.
2. Stable April Tag Detection from a height of <=5m due to the use of 12 x 18 sheets.
3. The goal point for the Husky can be given as a local location by clicking on the rviz GUI also as a global location in form of GPS latitude, longitude and orientation values.
4. The localization of the Husky robot using dual EKF fusing data from IMU, Odometry and GPS is accurate and stable.
5. Easy control of the Husky robot using Game Controller is very responsive and helpful in emergency situations.

Weak Points

1. Unreliable communication between the AGV and the UAV due to a software bug in the global path planner. Under certain circumstances, the internal graph representation may become disjointed. If this happens, then the software fails to find a path through the graph. We were aware of this issue before the SVE but only completed fixed for the SVE encore.
2. Exploration algorithm takes very long. Although we stated the robustness of the exploration algorithm, pruning dead ends and exploring ahead is

very slow. This comes down to a number of reasons: latency in the video feed, non-PID flight controls, inefficient pruning algorithm.

3. As the waypoints given to the Husky by Bebop are GPS values, system is heavily dependent on the accuracy of GPS signals. The performance of the system in terms of reaching the intermediate waypoints and final gps location is dependent on the weather and quality of signal received by the satellites.
4. The waypoints given by the bebop are only latitude and longitude values whereas the navigation stack on Husky requires a orientation/heading along with all the GPS values. We were running the Husky on a fixed orientation value for the SVE but we corrected this by calculating the heading value taking into account the previous GPS value for SVE encore.
5. Imu requires calibration based upon the environment it is being used in, which can be a limiting factor. The absolute Yaw values are heavily affected by the calibration of magnetometer in IMU.
6. Converting the location of April Tags to latitude/longitude using GPS on the drone and then following the GPS locations using the feedback from ground robot can add error from both the GPSs.
7. Testing at night and in bad weather – The optical sensor of the drone requires sufficient amount of light to operate and stabilize the drone. Hence testing at night is not an option for us. Also testing in bad weather could result in possible danger and should be avoided.

Refinement Areas

1. Improve exploration algorithm for the UAV
2. Increase UAV flying height with due considerations to safe ceiling.
3. Better localization for the AGV using RTK based GPS, which can increase the GPS accuracy and reduce the dependency on weather.
4. Instead of using the GPS values for waypoint following, bebop can provide the distance required to be travelled by Husky in terms of x and y, reducing dependency on one of the GPS.