
# UAV - AGV Collaborative Robots for Fire-Fighting

## Final Report


- Team H:
  - Akshit Gandhi
  - Shubham Garg
  - Parv Parkhiya
  - Zhihao Zhu (Steve)
- Sponsors:
  - Oliver Kroemer
  - Sebastian Scherer
- Advisors:
  - John Dolan
  - Dimitrios Apostolopoulos


**Date:** December 11, 2019

## Abstract

Structural-fire has caused 2,640 civilian deaths and material loss of 9.7 Billion USD in the US alone in the year 2011 as per the National Fire Protection Association. Time is of the essence when it comes to tackling most of the fire incidents. The PhoeniX team proposes a cutting-edge, fully autonomous, heterogeneous, multi-agent robotic systems to collaboratively locate and extinguish the fire without any human intervention in an unknown environment. Our system comprises a UAV (Unmanned Aerial Vehicle) and an AGV (Automated Ground Vehicle) equipped with a thermal camera that uses image segmentation methods for detecting and localizing the fire.

Our system uses depth cameras to simultaneously create a real-time 3D map of the environment and localize itself in that map. The system uses state-of-the-art algorithms to explore the environment while avoiding collisions. The UAV has roughly 20 minutes of flight time, high payload capacity (1 Kg), and improved stability that can be attributed to its DJI Matrice M210 V2 platform [14]. Both vehicles carry extinguishing material which they can strategically deploy on the target fire. The UAV and the AGV share information of the fire location with each other to make smart decisions resulting in a timely & efficient response. The PhoeniX firefighting system attempts to push the technological boundaries to create a net positive impact on mankind.

This report outlines the progress of the PhoeniX team towards building a collaborative robotic system for fire-fighting.

# Contents


**1 Project description** **1**

**2 Use case** **1**

**3 System-level requirements** **3**

**4 Functional architecture** **6**

**5 System-level trade studies** **8**
5.1 System-level trade study . . . . . 8
5.2 Component level trade study . . . . . 8

**6 Cyberphysical architecture** **11**

**7 System Description and Evaluation** **13**
7.1 Subsystem descriptions . . . . . 13
7.1.1 Hardware Subsystems . . . . . 13
7.1.2 Software Subsystems . . . . . 14
7.2 Modeling, analysis, testing . . . . . 18
7.3 FVD performance evaluation . . . . . 21
7.3.1 Functional Evaluation . . . . . 21
7.3.2 Qualitative Performance Evaluation . . . . . 22
7.3.3 Quantitative Performance Evaluation . . . . . 22
7.4 Strong and Weak Points . . . . . 23

**8 Project Management** **24**
8.1 Schedule . . . . . 24
8.2 Budget . . . . . 25
8.3 Risk Management . . . . . 26

**9 Conclusion** **28**
9.1 Lessons Learned . . . . . 28
9.2 Future Work . . . . . 29

**10 References** **30**

**11 Appendices** **31**
11.1 Appendix A . . . . . 31

## 1. Project Description

On 8th October 1871, a small barn in Chicago caught fire because of unknown reasons and what followed was a conflagration lasting 3 days that killed up to 300 people and made 100,000 residents homeless. In the aftermath of this Great Chicago Fire, Chicago and many other cities updated and implemented better fire safety codes.

Table 1 shows the number of fire incidents in the US in the year 2011 [1].

Table 1: **Reported Fire Incidents**

| Fire Location | Number of Incidents |
|---|---|
| Outside (Forest) | 686,000 |
| Structure (Building) | 484,500 |
| Vehicle | 219,000 |
| Total | 1,389,500 |

Table 2 shows the damages caused by these fire incidents.

Table 2: **Damages due to these incidents**

| Damage | Structure | Outside | Vehicle | Total |
|---|---|---|---|---|
| Property (Billion USD) | 9.7 | 0.616 | 1.4 | 11.7 |
| Civilian Injuries | 15,635 | 675 | 1,190 | 17,500 |
| Civilian Deaths | 2,640 | 65 | 300 | 3,005 |

The PhoeniX team proposes an autonomous multiagent system with navigation, perception capabilities and mechanism to deploy fire extinguishing material. Our system can also act as the first responder for collecting information about surroundings (map) and location of fire & trapped people, which human firefighters can use to make better judgments.

## 2. Use case

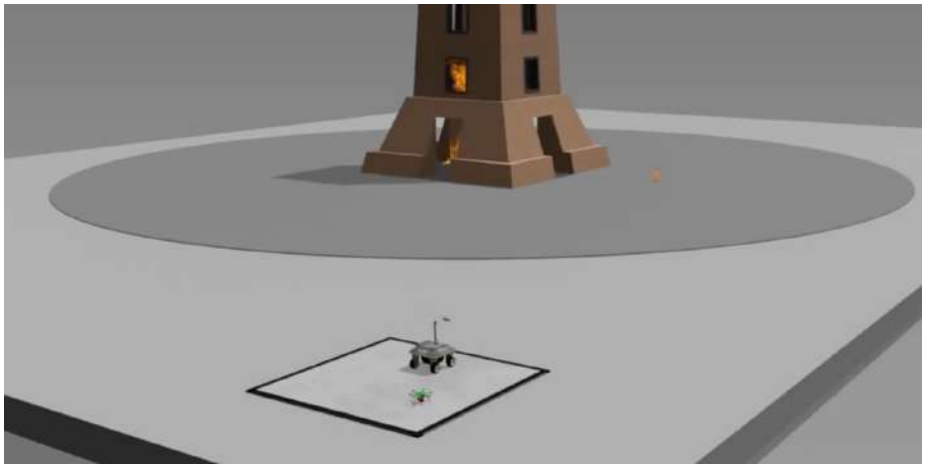

**Figure 1: UAV and AGV at base station**

North Oakland is a Pittsburgh community located near the world-famous Carnegie Mellon University. On the night of 11th Nov 2018, a three-story building caught fire. A few minutes after receiving the fire notification, firefighters reached an open area (the base station is shown in Figure 1) near the target building, and they put PhoeniX firefighting system on the ground and set off the initiating signal. The system becomes active and the UAV take-off from the station and the AGV also drives towards the building.

Both the robots coordinate and collaborate to optimally explore the surroundings by avoiding obstacles while creating a map of the environment. The UAV detects fire in the building at two locations: the ground floor and 1st floor. It shares that information with the other system.

**Figure 2: Robots moving towards fire locations**

The system divides the task of extinguishing the fire at those two locations as shown in Figure 2. The AGV is assigned the task of extinguishing the fire at the ground floor and UAV is assigned the first floor. As shown in Figure 3 the AGV uses a sweeping strategy to extinguish fire, whereas the UAV use some different mechanism to extinguish fire depending on the fire location.

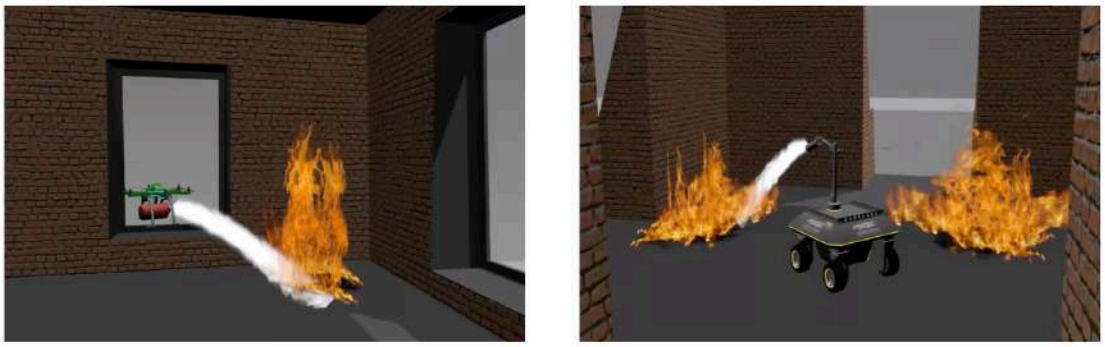

**Figure 3: Robots extinguishing fire**

Every robot monitors its fire extinguishing progress. The AGV reports that it has successfully extinguished the fire. When the UAV is out of the firefighting material, it requests help from the AGV. If the fire is reachable by the AGV, AGV comes and extinguishes the fire. After ensuring that there is no more fire in the building the UAV land back at the station along with the AGV driving back.

## 3. System-level requirements

Since all of our performance and non-functional requirements have been derived from the functional requirements and the objective tree, table 3 depicts a one-to-one mapping between the mandatory requirements. The naming convention is as follows:

- F.R. (Functional requirement)
- M.P. (Mandatory performance requirement)
- D.P. (Desirable performance requirement)
- M.N. (Mandatory non-functional requirement)
- D.N. (Desirable non-functional requirement)

Table 3: **Mandatory functional and corresponding performance requirements**

| Id | Requirement Description | |
|---|---|---|
| F.R.1 | Take-off and Land from base station | |
| | M.P.1 | Land within 5 m radius from center of base station for the UAV and 1 m for the AGV |
| F.R.2 | Plan Trajectory | |
| | M.P.2 | Explore 50 m x 60 m x 20 m environment with greater than 60% coverage (robot has seen and identified potential fire) in 10 minutes or less |
| F.R.3 | Create real-time map | |
| | Localize itself in the environment | |
| F.R.4 | M.P.3 | Accumulate less than 5 m drift for every 100 m of distance travelled |
| F.R.5 | Traverse desired trajectory | |
| | M.P.4 | Maximum error between desired and actual trajectory should be less than 1 m |
| F.R.6 | Avoid collision with obstacles and other UAV/AGV | |
| | M.P.5 | Keep 0.75 m minimum distance between system and obstacles |
| F.R.7 | Detect Fire | |
| | M.P.6 | Detect fire from a maximum 1.5 m away - in the line-of-sight of the UAV and AGV |
| F.R.8 | Localize and Monitor Fire | |
| | M.P.7 | Localize fire with less than 1 m error |
| F.R.9 | Deploy material strategically | |
| | M.P.8 | Carry 750 g of extinguishing material on the UAV and 1 Kg on the AGV |
| | M.P.9 | Deposit 40% deployed extinguishing material on the target area of minimum 0.5 m x 0.5 m at 1.5 m distance |
| F.R.10 | Coordinate between the different UAV and AGV | |
| | M.P.10 | Reliable communication within 25 m |

Table 4 shows mandatory non-functional requirements. A few non-functional requirements like size/form factor have been derived from the MBZ Challenge rule-book and thus they have remained consistent throughout the course of the project. They are subject to change only if any changes are made to the challenge. The rest of the requirements are added to make our system safe, modular, reliable and robust against environmental conditions (like wind).

Table 4: **Mandatory non-functional requirements**

| Requirement Id | Requirement Description |
|---|---|
| M.N.1 | Fit in the size of 1.2m x 1.2m x 0.5m (UAV) |
| M.N.2 | Fit in volume of 1.7m x 1.5 m x 2m (AGV) |
| M.N.3 | Feature kill switch for the AGV and manual control switch for the UAV |
| M.N.4 | Inter-operate with other MBZIRC team's systems by the means of functional modularity |

Table 5 shows the desirable non-functional and performance requirements. Our desirable requirements aim to make the system portable and easy to manufacture & have some additional safety features. Using propeller guards will not increase safety but will also reduce the wear and tear damage to the expensive carbon-fiber propellers. We also aim for the system to have a has parallel and distributed processing architecture to maximize the collaborative effort.

Our desirable requirements also aim to have intelligence built into the system such as docking when battery or extinguishing material falls below a certain threshold. Creation of a common map of the building based on the individual maps generated by the UAV and the AGV can help the firefighters to make better judgment and decisions. Since each UAV and AGV works independently, our system is easily scalable and capable of covering a large amount of area.

We are using a thermal camera for detecting fire in the environment. The same thermal technology can be utilized to detect and localize trapped humans in the different parts of the building. This information is crucial for firefighters to plan the rescue missions. Based on this we added a desirable requirement of notifying authorities about the trapped people inside the building.

Table 5: **Desirable non-functional and performance requirements**

| Type | Id | Requirement Description |
|---|---|---|
| Non-functional | D.N.1 | Perform non-overlapping tasks (mapping, extinguishing, etc) |
| | D.N.2 | Create a common global map by merging individual maps from different systems |
| | D.N.3 | Portable (weight, compact size/form factor within 0.5m x 0.5m x 0.25m) |
| | D.N.4 | Economical (system costs under $7000) |
| | D.N.5 | Feature user interface |
| | D.N.6 | Feature Prop Guards |
| Performance | D.P.1 | Dock to refill the extinguishing material when it is below 10% capacity within 5 minutes |
| | D.P.2 | Dock for battery recharge/battery replacement when it is below 20% capacity within 5 minutes -den |
| | D.P.3 | Detect humans trapped inside the building with 60% accuracy |
| | D.P.4 | Notify authorities about the location of people trapped inside to plan rescue mission within 45 seconds of human detection |

## 4. Functional architecture

The functional architecture of the PhoeniX firefighting system has been depicted in fig 4. It captures all the functionalities we derived from the functional requirements and the objectives tree. The operation of the system begins when an operator (person/system) triggers a start signal to the system which will be in the form of the approximate GPS location of the building on fire. Given this start signal the UAV takes off and the AGV drives off following a trajectory pre-computed by the AGV onboard and sent to the UAV. While they travel towards the fire location, the systems start to map the environment while avoiding any obstacles. This map will be used by the system to plan the path from point A to point B.

While the systems are exploring the environment, they will also keep on checking for any potential fire locations by using the fire detection subsystem. Once a system identifies a location with fire, it will inform other systems by adding the location of the fire in a shared database. If fire extinguishing tasks are pending in the database, an intelligent task assigner/scheduler will command an agent with enough extinguishing & battery resources to navigate to the fire location and extinguish the fire.

So, once the systems receive the coordinates of the fire location, they shall autonomously navigate in the environment while avoiding obstacles and now once they reach the proximity of the fire, they will orient themselves in an appropriate position to extinguish the fire. Now the system will deploy the extinguishing material using some strategy and update the database when they recognize that they have extinguished the fire. The system will stop once it runs out of the extinguisher material or battery.

Our functional architecture highlights the state-machine that we aim to implement. We have various modes such as navigation, exploration and mapping, fire extinguishing, collaboration, and scheduling.

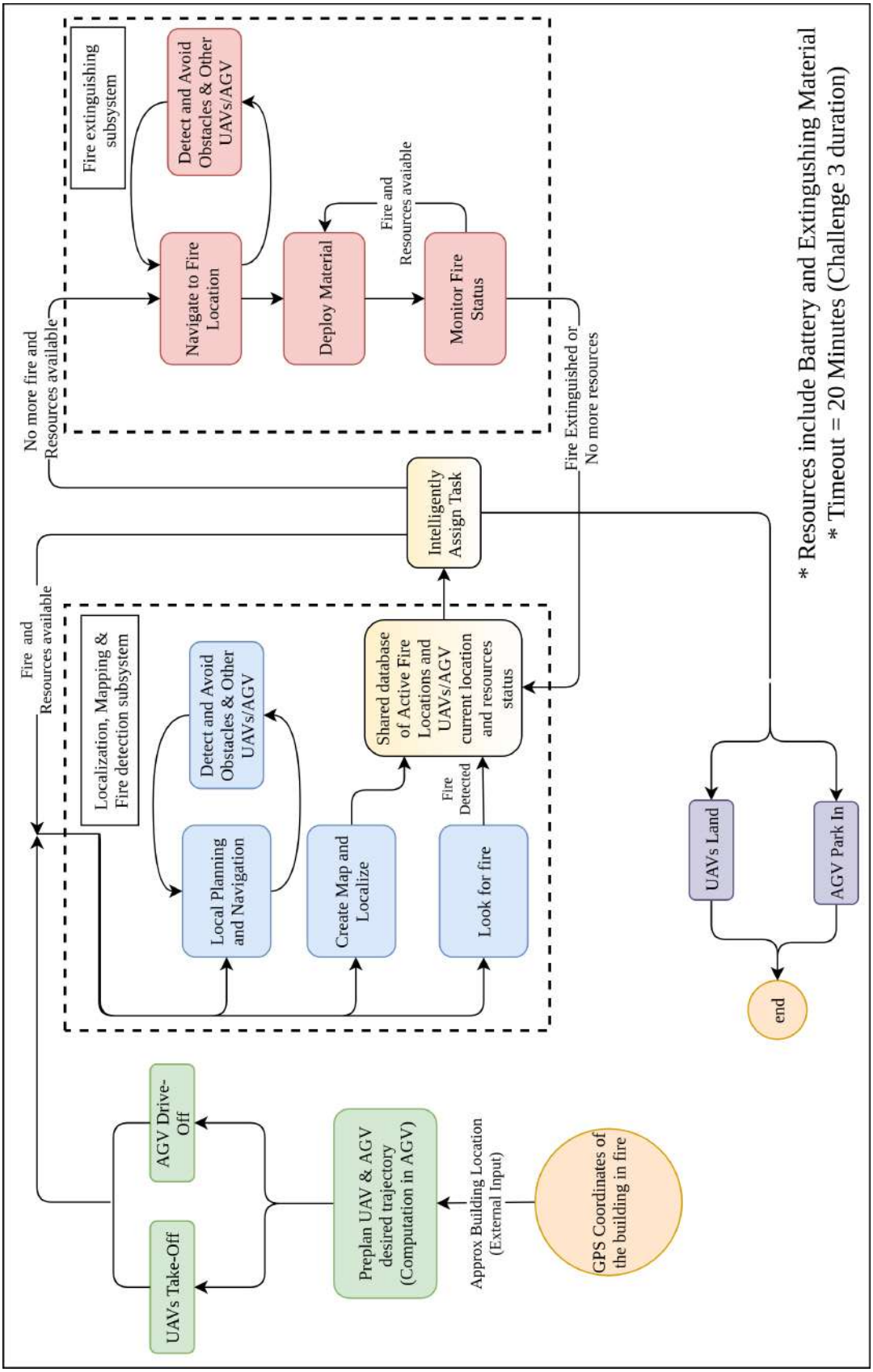


**Figure 4: Functional Architecture**

# 5. System-level trade studies

## 5.1. System-level trade study

We have done a system-level trade study between human firefighters, sensor-based traditional sprinkler system [6] and AGV + UAV collaborative firefighting. We ranked all attributes on a scale from 1 to 5, with 5 being the most favorable. Of all the attributes, response time and life risk are the most critical aspects of the fire extinguisher system. A sensor-based traditional sprinkler system has a response time between 7 secs to 33 secs. AGV+UAV can respond within a minute based on the base station's location while firefighters can take around four minutes to respond. Human firefighters have huge life risk but at the same time, they are currently much more reliable in extinguishing the fire as compared to the autonomous systems. Fire loss is minimum with human firefighters as they can extinguish the fire at large scale as they have a large extinguishing material carrying capacity but the UAV + AGV autonomous system has limited payload carrying capacity. AGV+UAV firefighting system can be placed at multiple remote locations which are not easily accessible to human firefighters. Similarly, the sensor-based sprinkler system can't be installed everywhere.

Table 6: **System-level Trade Study**

| **Criteria** | **Weight** | **Human Fire fighters** | **Sensor based fire sprinkler system** | **UAV + AGV** |
|---|---|---|---|---|
| Response Time | 20 | 3.5 | 4.5 | 4.0 |
| Life Risk | 20 | 2.5 | 4.0 | 4.5 |
| Fire loss | 15 | 4.5 | 2.5 | 4.0 |
| Robustness | 15 | 4.5 | 4.0 | 3.5 |
| Availability | 15 | 3.0 | 3.5 | 4.0 |
| Complexity | 10 | 4.5 | 4.0 | 3.5 |
| Setup Cost | 5 | 3.0 | 4.5 | 4.5 |
| Total | 100 | **3.6** | **3.8** | **4.0** |

## 5.2. Component level trade study

For the AGV+UAV firefighting system, we did four different component level trade studies. This spring semester, we did a trade study between our custom Hexacopter platform and DJI M200 V2 platform as shown in table 7. We realized that the DJI platform will be more reliable & stable as compared to our Hexacopter platform. DJI also provides better adaptive control which is useful in our case as our payload changes when our system extinguishes the fire. A major limitation of the DJI platform is that its payload capacity is less as compared to our custom Hexacopter platform. For that reason, we have changed our M.P.8 requirement where the UAV will now carry 750g instead of 1kg payload. Communication between the UAV and AGV is also crucial for our task. So, based on the trade study as done in table 8 we found WiFi [7] as the best reliable easy-to-use mode of communication.

Table 7: **Trade Study on UAV platform**

| **Criteria** | **Weight** | **Custom made Hexacopter platform** | **DJI Matrice 200 V2** |
|---|---|---|---|
| Payload Capacity | 20 | 4.5 | 4.0 |
| Reliability/Safety of structure | 20 | 2.5 | 4.0 |
| Power/Thrust | 15 | 4.5 | 4.0 |
| Stability | 15 | 3.5 | 5.0 |
| Ease of Maintenance | 15 | 3.0 | 5.0 |
| Cost | 10 | 3.5 | 3.5 |
| Control | 5 | 3.0 | 4.5 |
| Size | 5 | 3.0 | 4.5 |
| Total | 100 | **3.345** | **4.3125** |

Table 8: **Wireless Communication Trade Study**

| **Criteria** | **Weight** | **WiFi** | **Bluetooth** | **Cellular Technology** | **Zigbee** |
|---|---|---|---|---|---|
| Reliability | 10 | 4.4 | 3.1 | 4.2 | 4.2 |
| Power Consumption | 20 | 4 | 4.2 | 3.9 | 4.7 |
| Signal Penetration | 10 | 3.5 | 3 | 4 | 4 |
| Operating Range | 20 | 4 | 2.6 | 3.5 | 3.7 |
| Live Internet Connectivity | 5 | 3 | 0 | 2.5 | 0 |
| Additional HW Requirement | 5 | 3.8 | 4 | 3.2 | 4 |
| Ease of Integration | 25 | 4.5 | 3.4 | 3.2 | 3.6 |
| Location Dependency | 5 | 1.2 | 2 | 2 | 3 |
| Total | 100 | **4.2** | **2.44** | **3.345** | **3.03** |

Table 9: **Perception hardware Trade Study**

| **Criteria** | **Weight** | **SICK 2D LiDAR** | **3 front Stereo-cameras** |
|---|---|---|---|
| Size | 30 | 3.5 | 4.5 |
| Power consumption | 15 | 3.5 | 4.0 |
| Operating voltage | 15 | 3.5 | 4.5 |
| Control | 10 | 3.0 | 4.5 |
| Mounting | 10 | 3.0 | 3.5 |
| CPU usage | 10 | 4.5 | 3.0 |
| Total | 100 | **3.5** | **4.0** |

Table 10: **Single Board Computer Trade Study**

| **Criteria** | **Weight** | **Odroid XU 4** | **Snapdragon Flight** | **Nvidia Jetson** | **Intel Aero** |
|---|---|---|---|---|---|
| Cost | 5 | 1.25 | 0.25 | 0.25 | 1 |
| Power Consumption | 10 | 2.5 | 2 | 1.5 | 2 |
| Size | 5 | 1.25 | 1 | 0.5 | 1.25 |
| Peripherals | 5 | 1.25 | 1 | 2 | 1.25 |
| Storage | 5 | 1.25 | 1.25 | 1.25 | 1.25 |
| RAM | 10 | 1.5 | 2 | 2.5 | 1.5 |
| Speed | 10 | 1.5 | 2 | 2.5 | 1.5 |
| CPU | 10 | 1.5 | 2 | 2.5 | 1.5 |
| GPU | 20 | 2.5 | 4 | 5 | 2 |
| Total | 100 | **2.2** | **2.025** | **3.0** | **1.5375** |

In the last semester, we used SICK LiDAR for obstacle detection and it was supposed to be used for mapping in the Fall semester. But since we had to mount the UR5 arm also on the husky, we could not find a suitable location for LiDAR on the husky as it would now obstruct the UR5e arm work-space. Also, LiDAR is more power consuming device as compared to the stereo cameras as mentioned in table 9. So, we decided to use three front stereo cameras to create the map of the environment.

## 6. Cyberphysical architecture

Our Cyberphysical architecture is shown in figure 5 which maps the flow of data and energy between components and subsystems based on the results of our trade studies.

**UAV/AGV System:** Based on the trade studies, we have a finalized the combination of the stereo camera & IMU as mapping sensors on the UAV and three stereo camera (D435i) [10] added to the AGV for better localization. Different mobile robots communicate via a WiFi link. The Intel Tracking Camera T265 is also added on AGV for better localization. On the UAV we will use the stereo camera for window (opening) detection.

**Exploration Mode:** Path-Planning [16] subsystem will generate local trajectories avoiding the obstacles. Each mobile robot will use a classical computer vision algorithm for the fire detection. The system will use a path planning algorithm to create trajectories such that they cover a maximum possible area in minimum possible time. The exploration will also have a local planner that will use the obstacle detection hardware to generate strategies to avoid the obstacles. This mode uses the output of the localization module and the output of the thermal camera to detect the fire locations.

**Scheduler:** Each mobile robot will detect fire and update the shared database. Based on the fire location in the shared database, the scheduler will assign fire extinguishing tasks to different robots based on their locations from the fire. The scheduler will have heuristics such as proximity of the agent from fire location, battery status, extinguishing material status, etc.

**Extinguishing Mode:** Based on the global fire location, mobile robots will do visual servoing towards the assigned fire location, then monitor and extinguish the fire. The system also relies on a local planner to navigate around obstacles. The system utilizes the extinguishing hardware which currently is a water tank and a water pump. Based on the images from the thermal camera the agent decides when to engage the water pump.

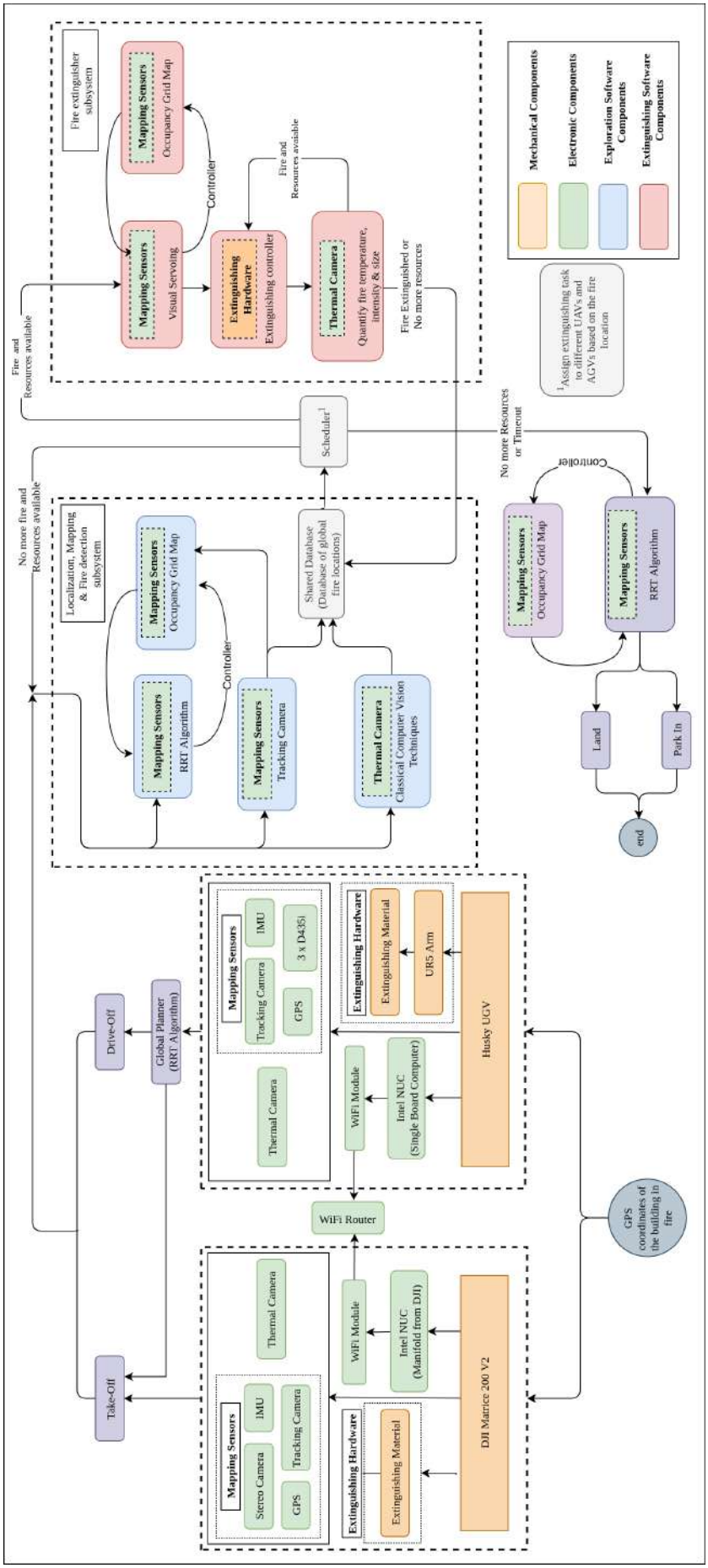


**Figure 5: CyberPhysical Architecture**

# 7. System Description and Evaluation

## 7.1. Subsystem descriptions

### 7.1.1. Hardware Subsystems

#### 7.1.1.1 UAV Subsystem

At the beginning of the Fall 2019 semester, our sponsors urged us to move to a new platform for the UAV because the previous custom drone platform was not stable and they could not get significant improvement in its performance during the summer break. Hence we shifted to the new DJI Matrice M210 V2 platform. Our UAV hardware subsystem consists of Realsense Depth Camera (D435i), a FLIR Boson 320 thermal camera, a fire extinguishing subsystem and the DJI Manifold 2-C onboard computer. Other miscellaneous components include the teensyduino micro-controller, relay, and a separate battery to operate the extinguishing subsystem. All the components of this hardware subsystem were changed from the Fall semester. Although there is a front-facing stereo pair on the DJI drone, the disparity map quality was not good enough and thus we attached an extra stereo camera. Currently, the PhoeniX UAV hardware subsystem looks as shown in figure 6.

**Figure 6: PhoeniX UAV**

We have an Intel Realsense D435i onboard for window detection  mapping, a Flir Boson 320 camera for thermal imaging and a Manifold 2C as an on-board computer figure 7.

**Figure 7: D435i Stereo Camera $(1^{st})$, Flir 320 thermal camera $(2^{nd})$, Manifold 2C $(3^{rd})$ and Camera mount 3D design $(4^{th})$**

We have custom designed our fire extinguishing subsystem which has two bottles mounted on the two legs of the landing gear (which can be seen in figure 6). To prevent the water from dripping/leaking we designed a cork type mechanism such that it prevents the dripping until the drone actuates the pump. We have also designed a custom mount for the Intel Realsense and the thermal camera, this mount was 3D printed and the design can be seen in figure 7 (the last picture).

#### 7.1.1.2 AGV Subsystem

The major task in the Fall 19 semester was to integrate the UR5e arm with the husky. We got a lot of help from Oliver's student (Kevin Zhang). The AGV hardware consists of the Clearpath Husky, 4 Intel Realsense Cameras ( x3 D435i, x1 T265), Ur5e arm and its control box, power supply inverter, 3DM-GX5-45 GNSS/INS (LORD sensor package), ethernet switch, NVIDIA Jetson TX2, FLIR Boson 320 thermal camera [12], extinguishing subsystem (including the Arduino Mega microcontroller, relay, and separate battery). The AGV subsystem is the most complicated hardware subsystem as it has a lot of components.

The AGV subsystem looks as shown in figure 8. This figure also shows the end effector for the UR5e arm which holds the extinguisher and the thermal camera.

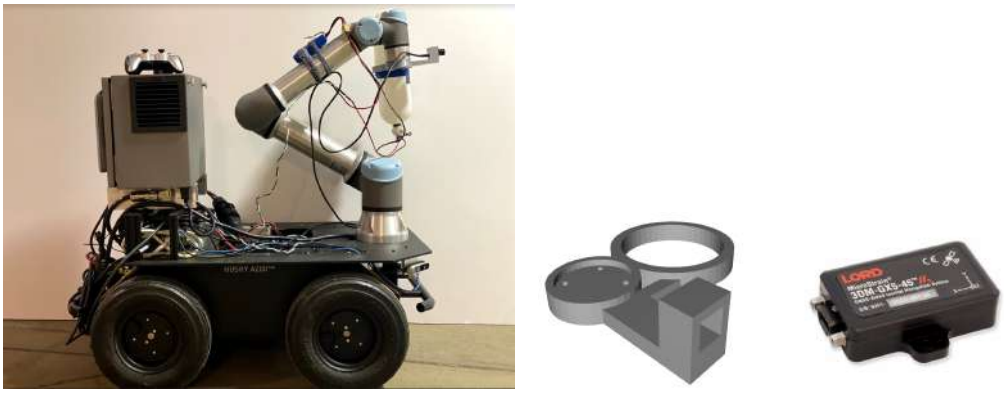

**Figure 8: PhoeniX AGV (left), Extinguisher and thermal camera mount (center), LORD sensor (right)**

### 7.1.2. Software Subsystems

#### 7.1.2.1 Behavior Tree framework

The AirLab uses a smarter version of a state-machine called a Behavior Tree. Behavior trees define how a set of actions and conditions should be used to accomplish a task. The tree is made up of execution nodes, control flow nodes, and decorator nodes. Action nodes and condition nodes are the two types of execution nodes. These nodes are where the state of the system is checked and actions are performed. A condition node returns either SUCCESS or FAILURE to indicate what the state of some part of the system is. For example, an "On Ground" condition node could indicate whether or not the robot is on the ground. These are shown as the oval-shaped nodes in the figure above. Green ones indicate SUCCESS, red ones indicate FAILURE. A sample behavior tree can be seen in figure 9.

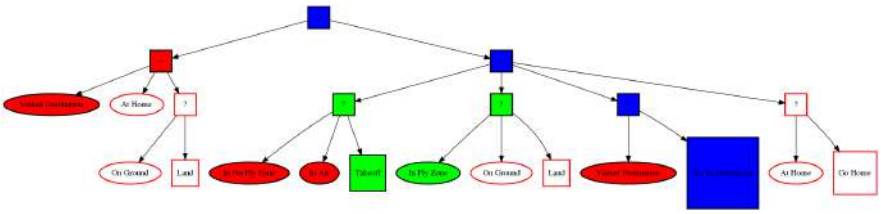


**Figure 9: Sample behavior tree**

The control flow nodes determine which condition nodes are checked and which action nodes are active or inactive. There are currently 3 types of condition nodes: fallback, sequence and parallel nodes. More details on the behavior tree framework can be found [2]. The UAV behavior tree has actions like takeoff, detect wall, align wall, detect window, enter the window, extinguish fire, land. The AGV behavior tree has actions like drive off, detect opening, enter the opening, extinguish the fire, receive fire location, go to the fire location, extinguish the fire.

#### 7.1.2.2 Simultaneous Localization and Mapping (SLAM) Subsystem

To efficiently plan a path towards the fire location, we need an occupancy grid-style map of our UAV and AGV. For the AGV, we have 3 Realsense depth cameras that provide us with point-clouds, we convert them into a laser scan and pass it to ROS GMapping package [15]. The GMapping package uses AMCL (Adaptive Monte-Carlo) to localize and simultaneously create a map of the environment. We have 3 ros nodes feeding data to the package at 2Hz each. Rather than having one node which combines all the point-cloud data from the 3 cameras, we have 3 separate nodes that prevent the system for failing if any of the cameras crash. The GMapping package also takes the sensor TF from the Odom frame and the vehicle odometry in the form of a ros message and as a TF transform. It publishes a cost map of the environment that is fed to our planning subsystem. A sample cost map can be seen in figure 10.

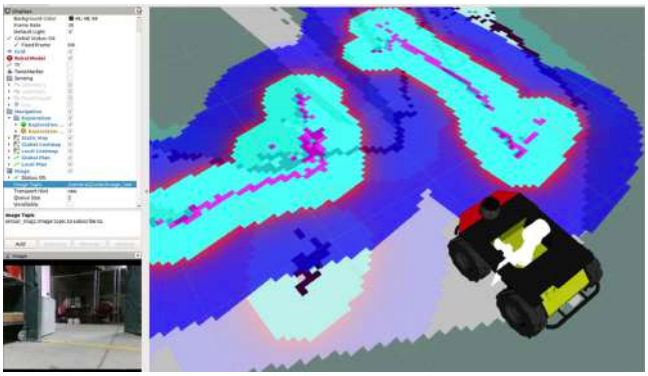

**Figure 10: Sample cost map from the GMapping package using point-clouds from 3 cameras on the AGV.**

For the UAV we currently don't have any mapping pipeline, and thus it lacks features like obstacle detection and avoidance. It also relies on the fact that the opening in the building is directly in front of it, which is like a prior in our case. Also for localization on the AGV we use the robot_localization package to fuse the IMU [13], wheel odometry and the tracking camera [9] data. For the UAV the localization is done using DJI's Visual Odometry pipeline using the downward-facing stereo cameras.

#### 7.1.2.3 Path Planning Subsystem

For the AGV we are using the DWAPlanner from ROS which takes the cost map from the GMapping package and runs the global and local planners. We tuned the parameters of the goal tolerance, robot size configuration, inflation around the obstacles and raytracing for obstacle path planning.

For the UAV we don't have a global planner, we just have a local planner to plan a trajectory through the opening. Given the coordinates of the window center, we first position the drone in front of the window at a distance of 2m from the center and then provide 7 waypoints to go inside the room (which is approximately 1.5m beyond the window center).

#### 7.1.2.4 Window/Opening detection

Our agents need to detect a door or a window to enter the building in order to extinguish the fire. At the beginning of the semester, we used the depth images to find an opening like a door or a window using classical vision techniques. But the results were not good as the depth images from the Realsense were not good in the outdoor environment.

Hence we moved towards point cloud processing where we use the line scanning method. For the UAV: we detect the dominant plane to identify the wall in the scene using the ICP method. Once the wall's plane is identified, the desired yaw change is computed for the UAV based on the orientation of the detected wall. Finally, in the point cloud, x-axis  y-axis are scanned using a sliding window mechanism to find the horizontal  vertical edges of the window. For the AGV, rising and falling edge laser scan values $(r, \theta)$ pair is converted into $(x, y)$ coordinates to get the centroid of the opening. This can be seen illustratively in figure 11.

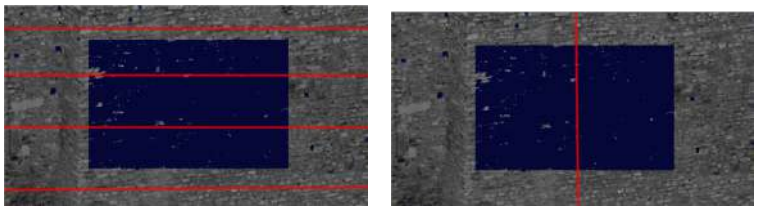

**Figure 11: Window scan across X axis (left), window scan across Y axis (right)**

#### 7.1.2.5 Collaboration Subsystem

Collaboration Subsystem will provide a communication link to transfer vital information such as active fire locations. This information can be extremely useful for high-level decision-making about what to do eg: explore, extinguish or return. The communication link would enable our system to fight the fire collaboratively.

To establish long-range communication between the systems, we are using a TP-Link AC1750 Wireless Dual Band Gigabit Router. This router helped us with our performance requirement of maintaining communication within 25m.

To send messages across agents, we used socket programming to establish bi-directional communication. Using a multi-threaded sender and receiver configuration of sockets, we were able to send data in the form of strings. We have a specific format in which the systems can send the data-casted as a string and the interpreter on the

other side can get the message and convert it into ROS messages/goals depending on what task we want to perform. Currently, we are just passing fire locations across agents.

#### 7.1.2.6 Fire Detection and Localization Subsystem

As the name suggest, the goal of this subsystem is to detect fire from the thermal image. The fire detection Subsystem looks for fire when the fire extinguishing task is active. We are using classical vision techniques (like morphological operations - erosion, dilation, and thresholding) to fetch the regions of high intensity in a thermal image and use prior knowledge of the shape of the fire region to make decisions. Figure 12 shows the hot water bag being segmented from the thermal images. We are able to identify fire regions within 1.5m line-of-sight (as it was our requirement).

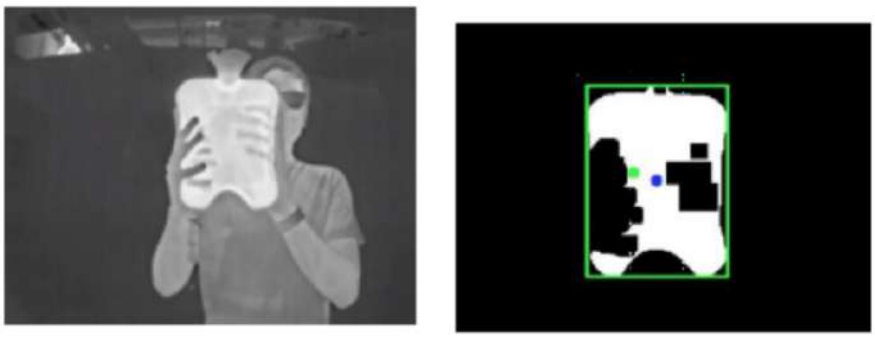

**Figure 12: Input image (left), Segmented output image (right)**

To localize the fire location, we take the current vehicle odometry and estimate the vehicles pose at a distance of 1m in front of it and pass on that pose as the fire location to the other agent.

#### 7.1.2.7 Fire Extinguishing Subsystem

This is a combination of the hardware and software subsystems. Previously in the spring semester, we used image-based visual servoing to orient the drone and the UR5e arm. This method relies on the fact that the fire is in the field of view of the camera. But this is not always the case. Thus we devised a simple exploration strategy to rotate the UR5e arm from -90 degrees to 90 degrees, similarly, the drone performs yaw from -90 degrees to 90 degrees to search for fire.

The extinguishing task is assigned after the agent has detected the fire. In the second step, control signals are sent to the micro-controller which in turn activates the extinguishing mechanism to start the pump. We have a master-slave communication architecture (using ROS serial) between the onboard computer and the micro-controller to engage and disengage the water pump.

## 7.2. Modeling, analysis, testing

1. Door/Window detection: One of the tasks in the competition is to enter a building through a window (for the UAV) and door (for the AGV). We initially tried a depth-image based window detection algorithm that didn't perform well in an outdoor environment and featureless environment. Moreover, it works for the door like opening where we have a boundary for all four sides. So, our idea is to write a generic opening detection algorithm that can be used for both windows and doors.
   We tried multiple algorithms to detect opening which can be used with both platforms. We tried a convex hull-based approach to find the opening. In this method, we need to remove all the dominant plane from the point cloud and just get the plan with the opening. Here, the major difficulty was to remove the dominant plane as it requires a lot of fine-tuning (for RANSAC) and also it might not work in all the environ- ments. Then we used a different approach inspired by the line scanning method as discussed in the above section.

2. Using Gazebo Simulation: We have built a Gazebo simulation environment, including AGV/UAV models, buildings, fire source, laser, obstacles (walls, closet), to test the functionality of the system. A snapshot of Gazebo simulation [3] is shown in Figure 13 (left). The simulation works as follows:

   (a) For the UAV, it first navigates towards the building. Once it detects a window, it flies into the room through the window. Then, it navigates inside the room and avoids obstacles along the way. When it detects the fire source, it flies towards it and positions itself right in front of the fire source by visual servoing. Once the UAV reaches a certain distance from the target fire source, it turns on the laser and points towards it. After all this is done, the UAV navigates the room and finds its way out via windows, and finally returns to the base station.

   (b) For the AGV, it basically follows the same procedures as UAV, the only difference is that AGV will only search for the first floor, while the UAV searches for higher floors. Through such a simulation, we can design our system's specifications accordingly, and fix the functionality of subsystems by testing each unit.

**Figure 13: Left: Gazebo Simulation for robot control, navigation and the fire detection, Right: Fire source simulation/modeling using massage bag filled with water as fire source.**

3. DJI front stereo disparity computation: DJI drone comes with a front-facing stereo camera but the disparity image coming out of this is not at all usable as shown in figure 14. We created our own node to take the raw images (240p) and compute the disparity and into the corresponding depth image that can be used for our previously demonstrated opening detection algorithm.

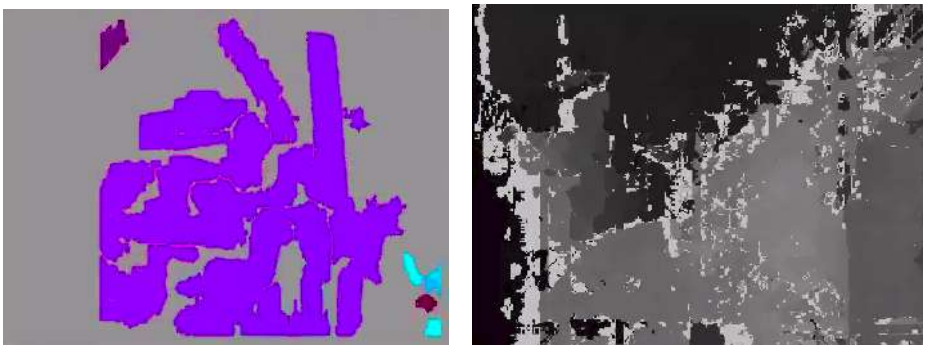

**Figure 14: DJI front camera disparity image**

4. Multi-camera calibration: Initially, we tried calibrating the multiple cameras simultaneously using Kalibr tool but the result was not satisfactory because the target (April tag) was not flat and we were moving target instead of the Robot (husky) to calibrate the cameras. So, we used husky-customization ROS package [17] to compute the camera transforms to the base-link by importing the STL files in the URDF settings of the robot model. Hence, we need not worry about any of the transforms between any link on the robot and the camera. Also, we are planning to mount the UR5 arm on the husky, having TF tree will make the transform between the camera and UR5 easier.
   Adding the camera joints/links in the existing robot model was challenging as I had to manually tweak the translation and rotation parameters to exactly align the camera mounts to their exact location on the robot base plate as shown in figure 15.

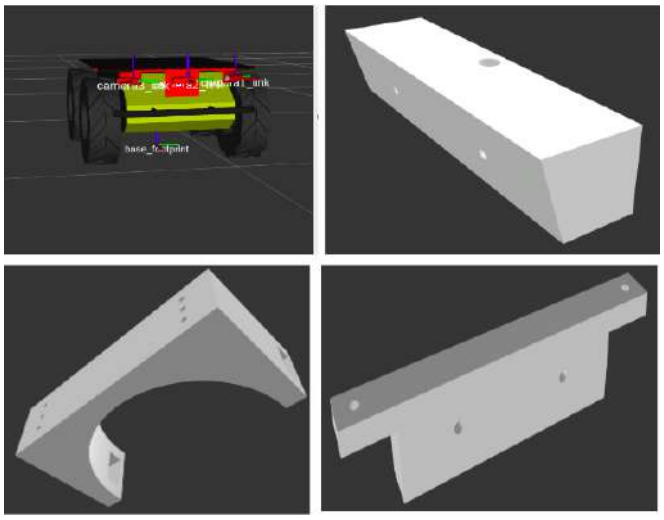

**Figure 15: Top Left: Husky URDF with cameras, Top Right: D435 Mount for drone, Bottom Left: Extinguisher mount for the UAV, Bottom Right: T265 tracking camera mount**

5. State estimation: We successfully fused the T265 tracking camera, wheel odometry & IMU (3DM-GX5-45 GNSS/INS) but we observed drift in the pose output for longer runs. Figure 16. shows the trajectory of the husky with the wheel odometry and with the EKF2 fused output (T265 tracking camera [4], wheel odometry & IMU). As can be seen clearly in the figure 16 that pose is drifting a lot with the wheel odometry alone because of wheel skidding and unevenness of the surface. Robot localization package [11] gives us the flexibility to select different parameters from $\left(X, Y, Z, \theta, \phi, \psi, \dot{X}, \dot{Y}, \dot{Z}, \dot{\theta}, \dot{\phi}, \dot{\psi}, \ddot{X}, \ddot{Y}, \ddot{Z}\right)$ for fusion. We tried changing this configuration for IMU to get the best results. In our testing, we found that tracking camera, IMU and wheel odometry fusion improves the localization for longer runs. There was around 0.35m improvement in the localization for the 20m trip.
   When measuring one pose variable with two sensors, a situation can arise in which both sensors under-report their covariances. This can lead to the filter rapidly jumping back and forth between each measurement as they arrive. In these cases, it often makes sense to only use an absolute pose from one sensor and its velocity from others. When the differential mode is enabled, all absolute pose data is converted to velocity data by differentiating the absolute pose measurements. These velocities are then integrated as usual. So, in our case differential mode is set got IMU and tracking camera while it is set false for the wheel odometry. Only accelerations (in x and y), linear velocities & angular velocities are fused from IMU and angular velocity from the T265 tracking camera.

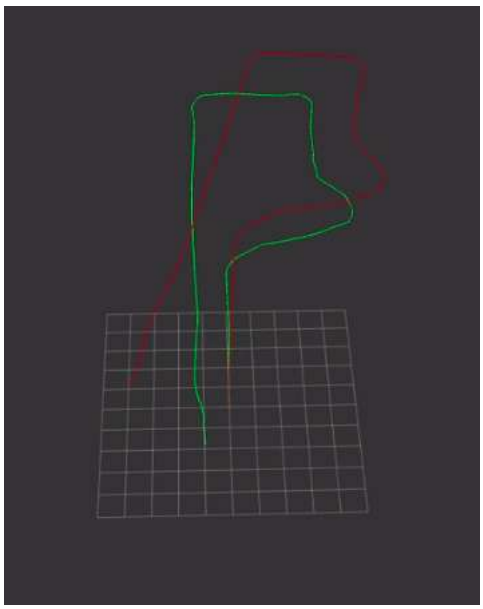

**Figure 16: Red line shows the trajectory with wheel odometry, green line show the trajectory with fused output**

6. Fire source simulation/modeling: In the MBZ challenge [5], a hot object will be used to simulate fire (if using thermal vision) and also there will be a red-colored fake flame. So, we are using massage bags filled with hot water as the simulated fire source. It is also shown in figure 13 (right). We adapted our fire detection algorithm according to the specification of such configuration of the simulated fire source (e.g. fire temperature and fire source volumes).

## 7.3. FVD performance evaluation

**Figure 17: UAV and AGV in action during FVD Encore**

As shown in figure 17, UAV-AGV collaborative firefighting robots were demonstrated at NSH-B level where an abstract building like structure was set up using tent and polystyrene sheets. Fires were represented using hot water bags. The window and door like opening were also created for showcasing the system's ability to autonomously enter the building. Performance evaluation of Fall Validation Demonstration(FVD) is done in the following three categories: Functional, Qualitative and Quantitative

### 7.3.1. Functional Evaluation:

- Given the start signal, AGV was able to drive towards the building. AGV was able to perform path-planning and reach inside the building and the UAV was able to detect window and fly through the window satisfying F.R. 1, F.R. 2 and F.R. 5
- After receiving the second fire location from the UAV and AGV were able to go near that fire demonstrating the AGV's and UAV's ability to localize fire with respect common map frame and satisfying F.R. 4 and F.R. 8
- The UAV and the AGV performed the mission without collision with any obstacle such as walls and window demonstrating its ability to avoid obstacle and satisfying F.R. 6
- The UAV and the AGV were able to do deploy water at multiple fires demonstrating the ability to detect fire and satisfying F.R. 7 and F.R. 9
- The UAV and the AGV shared the fire locations and the AGV deployed additional extinguishing material at the fire found by the UAV demonstrating their ability to collaborate through communication and satisfying F.R. 10
- Thus, the UAV-AGV system through the FVD, directly and indirectly, demonstrated all the mandatory functional requirements proposed in this project.

### 7.3.2. Qualitative Performance Evaluation

Some of the performance requirements from the coverage of the map to the minimum distance from the obstacle and fire are hard to measure during the FVD. These performance requirements are validated qualitatively or through subsystem testing.

- For all the demonstrations during FVD and FVD Encore, the system maintained a reasonable distance with obstacles such as walls and windows to avoid the collision.
- Apart from slight offset during one of the run, both the system deployed water quite accurately at the fire as shown in figure 18. In one of the run, because of hot water bags turning cold, the UAV didn't detect the fire and hence didn't deploy any water.

### 7.3.3. Quantitative Performance Evaluation

- Apart from the one particular run where hot water had cooled down, UAV-AGV successfully detected all the 3 fires (100%) placed inside the "building" at various distances up to 2 m away satisfying M.P. 6
- The UAV successfully carried 750 g extinguishing material and the AGV successfully carried 1 kg extinguishing material satisfying M.P. 8
- While we planned to validate M.P. 9 by collecting the water received at the fire using plastic bins (at least 40%), the collection of all the water from the fire was very hard. Hence, we decided to simply measure the amount of water deployed from the UAV. The UAV and the AGV deployed close to 100% of the extinguishing material they carried and the majority of which was sprayed directly on the fires.
- UAV-AGV were able to finish the joint collaborative mission in 2 minutes and 41 seconds satisfying the time aspect of M.P. 2

.

**Figure 18: AGV water deployment (left) and UAV water deployment (right)**

## 7.4. Strong and Weak points

**Strong Points:**

- Using the new DJI framework we were able to get even more stable flights, especially when the drone was inside the tent (which was used for the FVD). The drone has a pretty robust control framework.
- Both of the systems (UAV and AGV) don't rely on GPS and can work both indoors and outdoors using sensor fusion from multiple sensors, providing us with very accurate state-estimation.
- The extinguisher system for both the robots is very strong in terms of spraying the water at a very high distance (more than 2m).
- The AGV creates a map on the fly and thus we don't have any dependency on static maps.
- The UAV and the AGV can jointly perform autonomous missions using the behavior tree framework, which makes our system compatible with the other AirLab projects.
- The communication subsystem relying on the Dual Band 3 Antenna router is very strong and can help the robots communicate with each other within 40m of each other.
- Both the systems are anchored onto a common framework and thus the fire coordinates exchanged are pretty accurate, thanks to the common map frame and reliable state estimates on both the systems.
- The AGV just needs one RealSense camera to be active and can keep on functioning normally even if 2 other cameras crash or don't work.
- All the tune-able hyper-parameters are in launch files which makes it easy to experiment with different configurations on both the UAV and AGV.

**Weak Points:**

- Window detection code for the UAV works using the point-cloud from the Realsense Depth Camera, which has not been tested in the presence of sunlight and hence it needs more testing and refinement for an outdoor environment.
- The UAV cannot detect obstacles and do obstacle avoidance.
- The UAV and the AGV cannot compute how much water is left in the water tank.
- The UAV does not have any mapping capability yet, we need to integrate a voxel-based occupancy/3D mapping framework for the MBZIRC - 2020.
- The UAV cannot autonomously takeoff (it needs a manual intervention) and has some issues while landing, which needs debugging.
- Multiple USB devices on the Husky lead to RealSense nodes crashing, these crashes are not detected by the software which can prove fatal if all the 3 cameras go off.

# 8. Project Management

## 8.1. Schedule

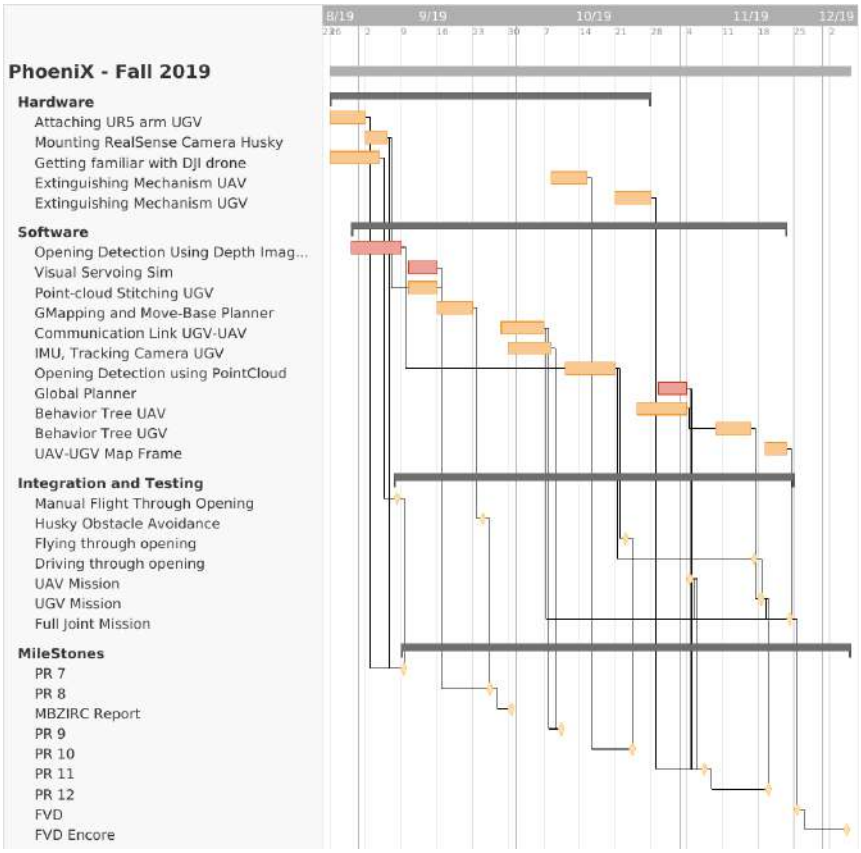


**Figure 19: Gantt Chart - Fall 2019**

The Gantt chart shown in figure 19 is the updated schedule that we followed in the fall semester. The spring semester was mainly spent on working on custom hex-copter. While some of the work from the spring semester carried forward to the fall semester, a lot of hardware-software had to be redone because of the UAV platform change and UR5 addition. Since many of the milestones from the spring semester are no longer relevant, only the fall semester schedule is shown.

The completion of the project with the new platform was a challenging task and required the timely completion of key milestones. Compare to the spring semester, our estimates for completions of various tasks were far better. The problems we faced while making the UAV autonomously enter through the window set us back for 2 weeks. Specifically, time spent on opening detection using depth images and visual servoing in simulation (shown in red) didn't work on real systems and we had move to pointcloud based detection and simple way-point based planner. Luckily we had kept Progress Review 12 as a buffer for dealing with such unexpected problems. Though, the crash of the UAV a week before FVD and subsequent time spent on repairing it left us with less time for final testing than we would have liked and a couple of sleepless nights could have been avoided. Overall while scheduling was not perfect, it was certainly significantly better than the spring semester and enabled successful completion of the project against odds.

## 8.2. Budget

Table 11: **List of Major Expenses broken down by category**

| Category | Description | Quantity | Total Price |
|---|---|---|---|
| Cameras | Stereo Cameras and Thermal Cameras | 4 | $2489.00 |
| Electronics | PCB, relays, microcontrollers, laser diodes and voltage converters | 60 | $582.93 |
| Custom Drone parts | Parts including screws, rubber dampers, mounts and misc replacement parts | 16 | $248.54 |
| Hardware | Screws, standoffs, pipes, pumps and plumbing supplies for extinguisher | 11 | $69.12 |
| Miscellaneous Consumables | Duct tapes, VHB, fire aerosol, hot water bag, water heater and safety glasses | 25 | $493.84 |
| DJI parts | Propellors, replacement arms | 7 | $223.00 |
| | | Total | $4106.43 |

Most of the expenses are done from our MBZIRC funds. We have spent around $20708 from MBZ funds for building two hexacopter UAV platforms and for buying a DJI Drone. From our $5000 MRSD budget, we have spent around $4220 (including shipping), i.e., 84.4% of our total budget. Also, we were given the husky and the UR5e arm from our sponsor Oliver Kroemer and we estimate the combined value to be at least $30,000. The team started working on the project from the winter break in 2018 and thus we ordered a lot of our big-ticket items like the ZED stereo camera (x2), thermal camera (x1), extinguisher bottles (x2) and other repair parts for the custom drone before the start of the Project Course - 1. The custom drone was built using MBZIRC funds and the repair supplies were ordered when the crashes happened in the first semester. During the PCB assignment, a lot of our electronic components were ordered along with the supplies for the SVD were ordered such as hot water bags, water heater, safety glasses, laser diodes, etc. Due to a failure of our SLAM subsystem, we had to order the T265 Tracking Camera in the Spring semester which was not initially accounted for in the budget. Approximately 62.56% of our budget was spent in the Spring Semester.

A brief breakdown of the purchase has been shown in Table 11. During the Fall semester, there were not many big-ticket items left to buy. We just purchased the components for a new extinguishing subsystem, DJI propellers, and miscellaneous repair components. There were some extra consumables like the T-REX tape, VHB, glues/epoxy and some more hot water bags purchased for the FVD. A detailed breakdown of the purchases can be found in Table 13 in Appendix A.

## 8.3. Risk Management

Table 12: **Risk Management**

| Risk Id | Risk | Category | L | C | Mitigation Strategy | Risk Owner |
|---|---|---|---|---|---|---|
| R.1 | Lack of availability of Test Site conforming MBZIRC Specifications | Schedule, Technical | 4 | 4 | Talk to sponsors to create a dummy test site of smaller scale by 27 September 2019. | Akshit |
| R.2 | Extra effort on repairing/maintaining the UAV and the AGV | Schedule, Cost | 5 | 5 | Maintain a contingency reserve especially for the UAV like motors, propellers and ESC | Zhihao |
| R.3 | Data/Code Corruption or changed configurations from DOE testing | Technical, Schedule | 3 | 4 | Ensure special version control of the key packages | Parv |
| R.4 | Software and hardware issues in porting to DJI platform | Schedule, Technical | 2 | 5 | Take help from other students at AirLab (John Kellar) | Parv |
| R.5 | Image based visual servoing does not perform up-to mark | Schedule, Technical | 4 | 5 | Move to a different approach for entering the window (for the drone) | Akshit |
| R.6 | State estimation of the robots in indoor environment is not accurate | Technical | 3 | 2 | 1. Ask Joshua about fixing the problem with robot localization package<br>2. Just use the tracking camera for indoors and perform no explicit sensor fusion | Akshit & Shubham |
| R.7 | ORB SLAM scale issue | Schedule, Technical | 3 | 5 | 1. Temporarily port to the Intel Realsense Tracking Camera for SVD<br>2. Use some other SLAM package which is compatible with ROS | Parv |

| Risk Id | Risk | Category | L | C | Mitigation Strategy | Risk Owner |
|---|---|---|---|---|---|---|
| R.8 | The UAV not able to carry the extinguishing payload due to the electrical or mechanical constraints | Technical | 4 | 5 | 1. Reduce the requirement for extinguishing payload<br>2. For electrical issues, route power directly from DJI battery<br>3. Add a separate battery for powering the extinguishing subsystem. | Akshit |
| R.9 | the Intel NUC USB buffer/processing not sufficient for multiple peripherals | Technical | 5 | 2 | Add extra compute in the form of a NVIDIA Jetson to control additional peripherals for the AGV | Akshit |

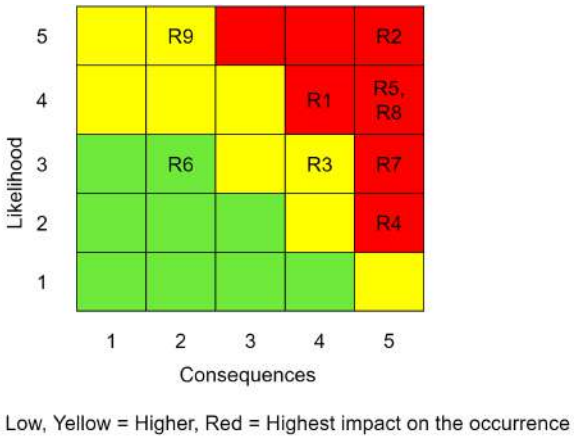


**Figure 20: Likelihood v/s consequence matrix**

Some of the risks that have been identified have been noted down in table 12. A likelihood v/s consequence matrix can be seen in figure 20. Overall, our risk management was successful, with most of the risks having been mitigated during the process of the project. Some of the successes include: assembling a tent indoors on the NSH-B Level helped us to navigate the weather uncertainty in the fall semester and also helped us to mimic the MBZIRC challenge at a small scale, changing the approach from image-based visual servoing to point cloud segmentation, powering the extinguishing subsystem on the drone using extra battery, etc.

We also had some failed risk management occasions where we had a serious crash of the drone just a few days before a progress review and we didn't have any replacement for the broken drone's arm. It had a very bad impact on our schedule. This is a failure case for our risk management where we didn't keep a contingency reserve of parts for the robots and it could have severely halted our progress.

# 9. Conclusion

## 9.1. Lessons Learned

Working on the project over the last year was an extremely challenging exercise. The biggest lesson is perhaps that making robots is hard. While mastering anything could be hard, robotics, in particular, is harder than what might seem on surface-level. At the core, robotics is the integration of complex electro-mechanical systems to perform complex algorithmic actions. All these constituting parts are a complex system in itself with their many assumptions for expected operation. Seemingly perfect working parts, when put together almost always, doesn't work right out of the box. The successful integration requires a thorough analysis of assumptions of its parts and rigorous testing for catching edge cases. The following is a non-exhausting list containing key lessons learned throughout the project.

- Continuous Integration: It's tempting to keep working on improving individual components and making them better. However, a more important thing is to start integration as soon as possible. We have spent a lot of time improving certain parts only to drop them when integration failed. The better approach would to first get the baseline component integrated with the system followed by improvement in its performance.
- Murphy's Law [8]: Anything that can go wrong, will go wrong. While the statement at first seems pessimistic, it's a golden rule in particular for "roboticists". Throughout the project, time and time again, we have seen system going haywire in very unexpected ways. While we can try to decrease the probability of accidents, things will go wrong and we must be prepared for it. Having various fail-safe mechanisms, emergency stop buttons, protective equipment, form pads, propeller guards, etc must be in place even if we are very confident about the particular simple test at hand.
- One thing at a time: Working on a big complex system becomes overwhelming at times and long debugging sessions without progress causes loss of perspective. In these difficult times, we must keep our focus on solving the particular problem at hand without worrying about the pile of work ahead of us.
- Having spare parts to save time in fixing systems: While It's not possible to have spare of every part of the system for monetary reasons, we should have spare parts whenever possible. In the spring semester, we faced multiple small delays when our 3D printed parts broke down near the SVD. Learning from our mistakes, we had 3D-printed all the parts twice. One of the spares was even used when one part, in particular, broke down just days before FVD.
- Plan is nothing, planning is everything: This particular idea was presented by one of the guest lecturers. However carefully, we plan the project, the plan inevitability will not pan out as expected. Planning and re-planning as we go along the project life-cycle are more important than sticking to the initial plan. Like the PID controller, the optimal approach is to constantly correct for the error.

## 9.2. Future Work

As we will be participating in the MBZIRC 2020 challenge [5] in the coming February, our future work will be first adapting our existing functionalities to the MBZIRC's specification, and in the meantime, developing necessary new modules that are needed to win the challenge.

First, to adapt our current system, there are several subsystems needed to be modified:

- the opening window/door detection algorithm needs to be ``re-calibrated''. Currently, we are relying on a scanning-based method for opening detection, where some hyper-parameters are only adapted to our current test environment and setting. So, we need to fine-tune these hyper-parameters for it to work in the MBZIRC settings.
- We also face a similar problem with the fire-detection algorithm. The fire source would be in the challenge will be a 110 degree Celsius hot metal patch of 5 cm x 5 cm which is very small, and thus it will require major tuning of our current algorithm.
- Water deployment mechanism needs to be re-designed. Currently, we are just using a relay to block the water from dripping, but once the relay is opened, there is no way to reserve the water, and we might have run out of the water before extinguishing all the fires. So, in the future, we need to design a mechanism to deploy the water only when it's needed and stop it from dripping after the first time robots deploy the water.

Secondly, we need to add new features to the robots. For instance,

- We need to develop a new fire exploration strategy. Currently, we just give the command to the robots for them to reach a desired location inside the room, and explore fire by moving the UR5e arm or by rotating the drone. This works well in a small-space room, but if the room is large that simply whirling around one point is not enough. So, we also need to develop a new indoor exploration system for the robot to explore as much as space as possible in order to find all the fires.
- We may also like to develop a strategy for the robots to navigate back to the base station after the fire extinguishing mission is done. At present, the AGV's planner can do this, but for the UAV we need a new planner that can plan trajectories around the obstacles.

## 10. References


[1] National Fire Protection Association (https://nfpa.org)

[2] Behavior tree: https://arxiv.org/pdf/1709.00084.pdf

[3] https://dev.px4.io/en/simulation/ros_interface.html

[4] https://www.intel.com/content/dam/support/us/en/documents/emerging-technologies/intel-realsense-technology/IntelRealSenseTrackingT265Datasheet.pdf
https://github.com/IntelRealSense/librealsense/blob/master/doc/t265.md

[5] MBZ-International Robotics Competition: https://www.mbzirc.com/

[6] Fire Sprinkler System https://en.wikipedia.org/wiki/Fire_sprinkler

[7] Wireless Communication: https://www.electronicproducts.com/Computer_Peripherals/Communication_Peripherals/Bluetooth_vs_Wi_Fi_vs_ZigBee.aspx

[8] Origin of Murphy's Law: https://www.murphys-laws.com/murphy/murphy-true.html

[9] T265 Tracking camera: https://www.intelrealsense.com/tracking-camera-t265/

[10] Intel D435i stereocamera: https://www.intelrealsense.com/depth-camera-d435/

[11] Robot localization: http://wiki.ros.org/robot_localization

[12] Flir Thermal camera: https://www.flir.com/products/boson/

[13] 3DM GX5 IMU: https://www.microstrain.com/sites/default/files/3dm-gx5-45_user_manual_8500-0010.pdf

[14] DJI Matrice 200 V2 platform: https://www.dji.com/matrice-200-series-v2

[15] GMapping: http://wiki.ros.org/gmapping

[16] ROS planner: http://wiki.ros.org/navigation

[17] Husky ROS package: https://github.com/husky/husky

# 11. Appendices

## 11.1. Appendix A

Table 13: **Bill of Materials**

| Quantity | Component Name | Unit Cost | Total Cost |
|---|---|---|---|
| 2 | ZED Stereo Camera | $449.00 | $898.00 |
| 2 | Washer Bottle with pump | $72.85 | $145.72 |
| 1 | Tarot X6 Landing gear connector (Aluminium) | $14.95 | $14.95 |
| 1 | Antenna for Nvidia Jetson TX2 | $8.59 | $8.59 |
| 1 | Thermal Camera | $1,242.00 | $1,242.00 |
| 1 | Thermal Camera USB C | $150.00 | $150.00 |
| 1 | Power converter | $25.60 | $25.60 |
| 4 | Tarot rubber damper | $7.50 | $30.00 |
| 4 | Tarot extended rubber damper | $9.90 | $39.60 |
| 8 | Hex Standoff | $4.39 | $35.12 |
| 24 | Resistors, Capacitors and wires | $12.29 | $12.29 |
| 2 | FQP30N06L | $1.22 | $2.44 |
| 2 | PC817 | $0.54 | $1.08 |
| 6 | POWERPOLE_POWER45A_DRILL | $0.88 | $5.28 |
| 2 | UWS-12/4.5-Q48N-C | $33.94 | $67.88 |
| 3 | UWS-5/10-Q48N-C | $41.23 | $123.69 |
| 6 | POWERPOLE_POWER45A_DRILL | $0.88 | $5.28 |
| 1 | Hot water bag | $12.99 | $12.99 |
| 1 | Laser diode | $8.95 | $8.95 |
| 1 | Water heater | $39.99 | $39.99 |
| 2 | Aluminum Motor Mounts (orange) | $25.90 | $51.80 |
| 1 | Aluminum Motor Mounts (black) | $25.90 | $25.90 |
| 1 | The Intel Tracking Camera | $199.00 | $199.00 |
| 2 | Laser diode (Dot) | $5.95 | $11.90 |
| 3 | Aluminum Motor Mounts (black) | $25.90 | $77.70 |
| 3 | 3M Tape | $9.24 | $27.72 |
| 10 | Safety Glasses | $9.95 | $99.50 |
| 1 | TP-Link AC1750 | $57.99 | $57.99 |
| 1 | Raspberry Pi 7 Inch Capacitive Touch Screen | $62.99 | $62.99 |
| 1 | Fire Aerosol spray | $21.90 | $21.90 |
| 1 | Water Pump (Pack of 2) | $13.99 | $13.99 |
| 1 | Pipe T-junction | $11.72 | $11.72 |
| 1 | Pipe | $8.29 | $8.29 |

| Quantity | Component Name | Unit Cost | Total Cost |
|---|---|---|---|
| 1 | Logitech USB Unifying Receiver | $12.48 | $12.48 |
| 1 | XT-30 Power distribution board | $10.00 | $10.00 |
| 1 | Teensyduino board | $22.40 | $22.40 |
| 1 | 24V to 19V voltage regulator | $25.60 | $25.60 |
| 6 | DJI Propellers | $19.00 | $114.00 |
| 1 | Relays | $5.79 | $5.79 |
| 2 | Hot water bag | $11.99 | $23.98 |
| 1 | T-Rex tape (Pack of 3 Roll) | $23.49 | $23.49 |
| 1 | Super 15187 , Clear- pack of 12 | $6.30 | $6.30 |
| 2 | Matrice 200 V2 Arm Carbon Tube Module (M2) | $65.00 | $130.00 |
| 1 | Matrice 210 V2 Arm Carbon Tube Module (M2) | $109.00 | $109.00 |
| 3 | Intex Mini Frame Pool | $28.69 | $86.07 |