Autonomous Quadrotors for the 2019 International Aerial Robotics Competition

Martin Deegan, Adam Dziedzic, Cheng Jiang, Ruth Moon, Daniel Pisarski, Ryan Wunderly
Michigan Autonomous Aerial Vehicles
University of Michigan
{mdeegan,dziedada,chengjia,ruthmoon,danpis,rywunder}@umich.edu

ABSTRACT

The 2019 International Aerial Robotics Competition (IARC) will be held in Atlanta, Georgia. Michigan Autonomous Aerial Vehicles (MAAV) has designed autonomous quadcopters to rise to the new mission. MAAV's quadcopters are designed to minimize overall structure while still carrying stat-of-the-art sensors and electronics. On board the quadcopter, highly sophisticated algorithms such as Unscented Kalman filter and deep neural networks are executed to allow the quadcopters to fly safely. MAAV's vehicle will compete in the 2019 IARC to demonstrate autonomy and attempt the new mission.

INTRODUCTION

The 2019 International Aerial Robotics Competition (IARC) will be held in Atlanta, Georgia. This document presents the MAAV system designed and fabricated for the IARC.

Problem Statement

Human-drone interaction is notably introduced in the IARC Mission 8 and proves to be crucial for its completion. Under the assistance of a human player via gesture or voice commands, helper drones will act semi-autonomously and aid the human player in retrieving objects from four password-protected bins guarded by sentry robots. In addition to autonomous flight and obstacle avoidance, helper robots must be capable of defending the human player from enemy attack and stream video to assist visual observation of the environment.

MAAV's Solution

Custom-built autonomous quadcopters have been tailored to the requirements of Mission eight. The quadcopters are designed to demonstrate full autonomy and function as a swarm. They also feature a man-machine interface to better serve and protect the human player. On board each quadcopter, there are advanced sensors including IMU, Lidar, RGBD cameras, and a tracking camera. Each quadcopter also features state-of-the-art computing devices such as NVIDIA Jetson TX 2 embedded GPU and Intel NUC single board computer, where highly sophisticated algorithms are executed to ensure safe autonomous flights while completing the mission. Some of these algorithms include Unscented Kalman Filter to fuse noisy and delayed sensor inputs as well as deep learning models to recognize objects in varied lighting conditions and backgrounds. There are two custom designed circuit boards that provide power to onboard electronics and contain a TI DK-TM4C TIVA embedded processor to interface with a controller, sensors, and motors. Figure 1 shows a diagram of MAAV system architecture.

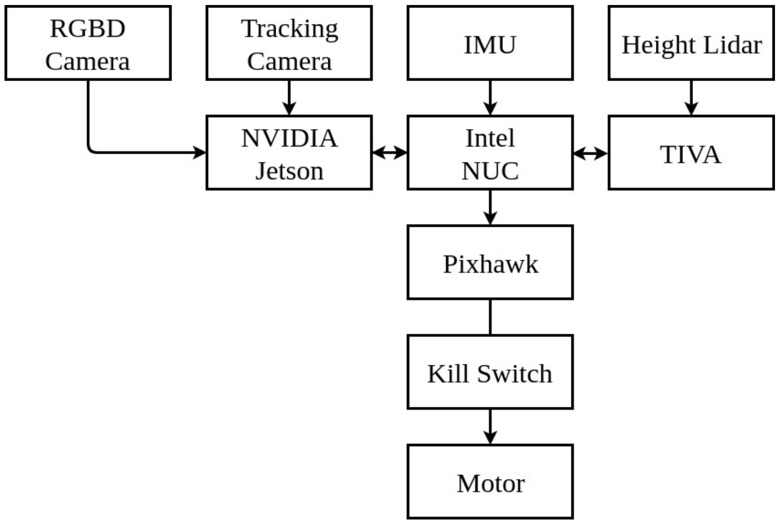


Figure 1: MAAV System Architecture Diagram

Yearly Milestones

MAAV has undergone several changes in preparation for our first year participating in Mission eight. The chassis has been completely re-designed to be lighter and smaller to increase flight time. New propguards have been built to protect against injury to the human player. Our codebase has been refactored and modulated to facilitate testing and design for future iterations. We have achieved autonomous flight and obstacle avoidance as per Mission 8 requirements.

VEHICLE DESIGN

MAAV's autonomous quadcopters are fully designed in house. This section describes the physical design of the vehicles, as well as the control and localization software that enables autonomy.

Physical System

The entire quadrotor design has been conceived using Solidworks 2018. The model has been designed and assembled to ensure proper placement of all components which allowed the team to predict the physical properties (i.e. moment of inertia, center of gravity) of the vehicle for use in simulations. Fusion 360 has also been used to generate the tool paths for machining custom parts, laser cutting, and 3D printing. All parts including the carbon fiber airframe, center body, PCBs, sensor mounts, and motor mounts have been custom designed and fabricated for this vehicle. An image of a prototype CAD model is shown in Figure 2.

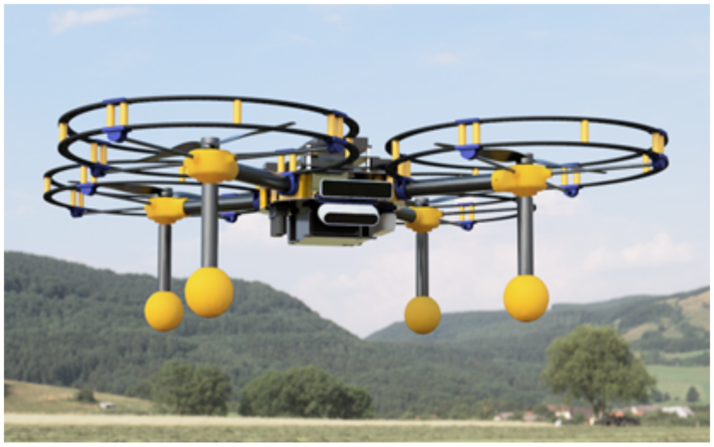


Figure 2: A prototype model in Solidworks.

The MAAV quadrotor weighs approximately 4 kg, spans 105 cm diagonally from outer prop guard to prop guard, has a height of 30 cm, and has a max vertical thrust of about 100N.

Structural Design

The overall structure of this vehicle is designed to its size while still carrying all the sensors and electronics necessary for autonomous flight and robust enough to survive falls and crashes. The vehicle has four motors on the end of carbon fiber arms that connect to a wide center body which houses the battery, sensors, and boards. The motors sit atop 3D printed mounts that are attached to a rigid carbon fiber tube. On the end of the tube, a foam ball is used to dampen the impact of any landings. The propellers are protected by an upper and lower ring of laser cut delrin, and a mesh netting (not pictured in Figure 2) is used to protect any objects from coming into contact with the propellers. 3D printed spacers and connecting components lock the eight-quarter circle delrin pieces together per propeller guard. 3D printed components are made using Zortrax Ultra-T, a strong, stiff, and impact resistant engineering grade material with all necessary properties for an aerial drone.

The sensors and sensitive components are located above the center body to ensure short connections between the electrical components while reducing wiring overall. The battery is located below the center body, in a lightweight aluminum case to have easy access at insertion or removal. This aluminum case also protects the battery from any impacts or sharp objects which could penetrate the battery upon a crash.

Propulsion and Lift Systems

The quadrotor is lifted by four 36 cm, two-blade carbon fiber propellers mounted on T-Motor MN4012 motors. These produce approximately 100 N of lift for a lift-to-weight ratio of 2.5, which is beneficial for agile flight. The quadrotor is equipped with a 6S 5700mA-hr lithium polymer (LiPo) battery manufactured by DJI which powers the motors and allows for a flight time of roughly 12 minutes under competition conditions.

LiPo batteries maintain a constant voltage for most of their charge and thus it is important to have a method for monitoring battery charge. MAAV monitors battery status on a custom circuit board to maintain safe flight conditions.

MAAV chooses the propeller size by testing on a thrust stand with the chosen motor and varying the propeller specifications like diameter and pitch. The propeller that could achieve 12 N of thrust, or enough to allow the drone to hover for 12 minutes from a full battery, while drawing the least amount of power was chosen. These tests also provide relationships between thrust and battery power to help the vehicle fly better when batteries are low on power. The propellers selected were T-Motor 14x4.8 carbon fiber propellers.

Circuit Design
The vehicle's electrical components are powered via a set of custom boards that are designed in-house with Autodesk Eagle. There are two main boards, the power distribution board and the signal board. The power distribution board takes power from the batteries at 22.2 V and sends it to the motors. It also steps the voltage down to allow it to be used by the sensors and the Intel NUC. The signal board holds the TIVA and takes input from the sensors and the kill switch to send them to the NUC and the TIVA. The NUC connects to the Pixhawk, a flight controller that sends signals to electronic speed controllers to determine how much power is sent to the motors.

Sensor Suite
The vehicle uses a plethora of sensors, as shown in Figure 2, to understand the environment and give feedback to the algorithms to determine its own location and the location of the ground robots.

VectorNav VN-100 AHRS: The VectorNav attitude and heading reference system (AHRS) returns the roll, pitch, and yaw angles as well as the roll, pitch, and yaw angular rates in the form of radians and radians per second. These values are already filtered and are used by the localization filter as odometry input.

Intel RealSense D435 Depth Camera: Two depth cameras provide a 3D dimensional point cloud of the arena. One camera mounted at the front of the vehicle, at a 30-degree downward angle, and the other camera is mounted facing directly downwards on the underside of the center body. The point cloud is used for localization and obstacle avoidance algorithms.

Intel RealSense T265 Tracking Camera: One tracking camera located on the front of the vehicle is used as a simultaneous localization and mapping device. This tracking camera creates a map of the explored environment while keeping track of its location relative to the overall map.

Pulsed Light Lidar-Lite Laser Module: One Lidar-Lite laser rangefinder will be mounted on the vehicle pointing directly downward to provide altitude feedback.

Figure 3: From left to right, VectorNav VN-100 AHRS, Intel Realsense D435 Depth Camera, Intel Realsense T265 Tracking Camera, and Pulsed Light Lidar-Lite Laser Module

Flight Control System

The flight control system onboard MAAV's quadcopter are consisted of two modules: navigation and control. The remainder of this section describes them in full detail.

Navigation / State Estimation System

All state estimation for the vehicle happens in 15 degrees of freedom (DoF) Unscented Kalman Filter (UKF). Our UKF is similar to [1], but our method was created independently before discovering their paper.

Our UKF estimates attitude (as a rotation matrix), position, velocity, and IMU biases (gyroscope and accelerometer) as well as provides a 15x15 covariance matrix which gives our uncertainty of these values. The filter loops at 200 Hz, the frequency at which it receives IMU measurements. The complex dynamics of the quadcopter are replaced with the integration of the IMU. There is a significant amount of noise in the IMU signal, mostly from the vibrations caused by the propellers. This accumulates large errors, up to a factor of 10 meters in a few seconds in our IMU integration. Therefore, we rely on additional measurements to correct our state.

The first measurement comes from our downward facing lidar. This allows us to partially observe altitude, vertical velocity, roll, and pitch. This measurement runs at 50 Hz. The Lidar-Lite often reads large spikes which fluctuate between 0m (minimum distance) and 10m (maximum distance). These can make the filter diverge immediately, therefore it is essential that we use a probabilistic outlier rejection scheme to discard these measurements.

Next, we use our downward facing depth camera to fit a plane to the flat gym floor. This allows us to directly observe altitude, roll, and pitch as we can compute these from the equation of the plane. This measurement serves as a more accurate version of our lidar measurement, but only runs at 30 Hz.

Third, we use the accelerometer and magnetometer to provide additional corrections to our attitude. By comparing the accelerometer reading to gravity and the magnetometer to the magnetic field, we can fully observe our attitude.

Finally, we use our Intel T265 tracking camera. This camera runs SLAM internally and provides a pose relative to its turn on location [2]. The loop closing is uncertain in the T265, but the drift in the visual odometry is extremely low; it is usually less than 1% of the total trajectory. Thus, regardless of its loop closing abilities (how much it can prevent drift), we use it as a global pose. This allows us to directly update position and attitude. The drift is accounted for in our occupancy mapping which is refreshed frequently enough to adjust the filled cells in time for us to avoid them.

It is important to note that velocity and IMU biases are not seen in any of these updates. These properties actually become fully observable under certain motions. Correlations in our covariance matrix that build up in the dynamics propagation allow us to fully observe velocity and our IMU biases using only these measurements.

a history of previous measurements is stored, and any delayed measurements are placed in the timeline and the filter is rerun on from that point, thus incorporating them correctly. The state is published along with the covariance at 200 Hz to be used in several other parts of the software stack including the controller, path planning, map building, and strategy.

Control Architecture

The flight control system architecture consists of a cascaded PID controller with an outer loop running on an Intel NUC and an inner loop provided by the Pixhawk 4 Mini flight controller running PX4 firmware. The inner loop tracks input commands for attitude and thrust and provides motor speed signals to the electronic speed controllers (ESCs). The outer loop control system outputs attitude and thrust setpoints to track global position commands set by the path planner. A cascaded controller architecture allows for quick responses to sudden changes without relying on new state information to be updated since this is a costly procedure. The inner loop running on the Pixhawk can quickly respond to sudden disturbances without needing to immediately rely on the outer loop which is much slower.

The z controller of the outer loop outputs a thrust based on the height of the vehicle while the x and y controllers output roll and pitch angles based on the current x and y position of the vehicle. Non-zero roll and pitch angles in response to a position error create lateral thrust components which move the vehicle in the desired direction. Deviation in altitude forces a response from the z controller which causes an increase of thrust to maintain the proper height. The maximum speed for x and y is roughly limited by not exceeding maximum roll and pitch angles. Setpoint yaw from the path planner is passed directly from the outer loop system to the inner loop system with only an adjustment into the local coordinate frame. All yaw control is provided by the inner loop controller; however, we limit the maximum yaw rate in the PX4 settings.

The controller runs in a 7-state finite state machine. This allows us to split the way feedback is handled depending on our current state as well as provide more safety as we know when the vehicle has spinning propellers. A list of controller states is listed below, and a controller state transition diagram is shown in figure 4.

- *STANDBY* - Motors off. Vehicle is safe to handle.
- *ARM* - Send arming commands to the Pixhawk until it arms. Once armed either switch to *TAKEOFF* or *ARMED* depending on which command was given.
- *DISARM* - Send disarming commands to the Pixhawk until it disarms. Once disarmed, switch to *STANDBY*.
- *ARMED* - Motors idle at a low throttle. The vehicle can now be commanded to takeoff or disarm.
- *TAKEOFF* - The vehicle raises throttle and tries to fly to the specified “takeoff altitude”. Once reached, the vehicle switches to *FLIGHT* mode.
- *LAND* - The vehicle descends directly down. A landing detector is started and continuously tries to detect touchdown. Once detected, switch to *DISARM*.
- *FLIGHT* - The main operating state of the vehicle. Position commands can now be issued to the vehicle and it can move laterally.

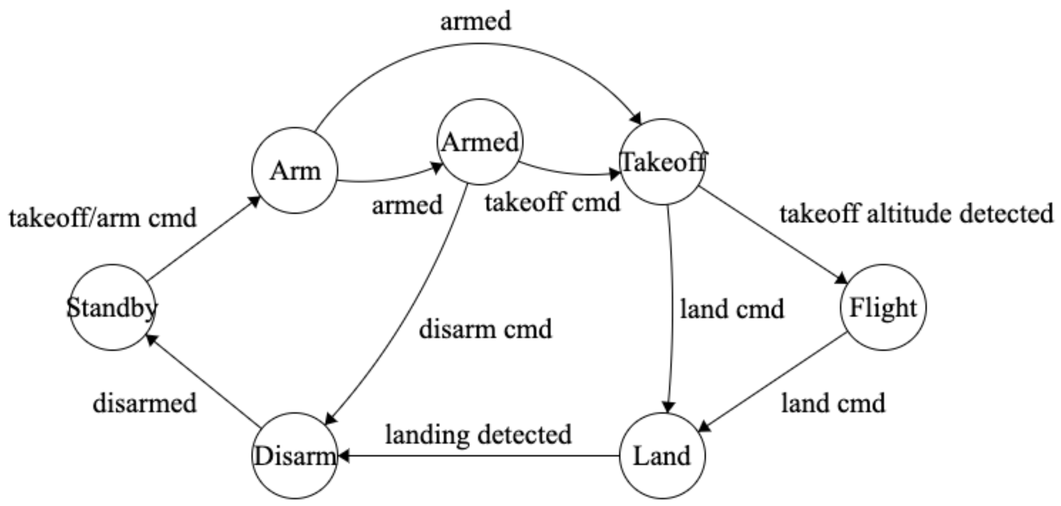


Figure 4 Controller state transitions.

Flight Termination System
A backup kill switch is implemented as a last resort. In the event of a complete computer meltdown that causes the quadrotor to enter an unresponsive and dangerous state, a human-operated kill switch disables all power to the motors. The source of the kill switch signal originates from a common RC controller supplied and operated by IARC judges. This standardization guarantees that the kill switch operates on a reliable frequency, separate from the communication frequencies used by the vehicle for data and video transmission. The signal from the kill switch receiver is a PWM signal that is processed by a microcontroller independent of the main system. We chose to use a microcontroller instead of the suggested design to give added flexibility to how the vehicle responds to the receiver's signal. The added complexity is justified because it allows us to add important features like noise immunity and fail-safe functionality without sacrificing response time.

MISSION SPECIFIC SOLUTION
MAAV's software is a highly sophisticated system tailored to the particular challenges posed by the eighth mission of IARC. This section describes perception and the behavior planning of MAAV software architecture.

QR Code Recognition
Retrieving the passcode from the QR fragments is a complex process. The first step is to capture images of the QR fragment while minimizing glare from the reflection of the iPad. This is accomplished by aligning many images taken from various positions and orientations, aligning all the images using correspondence RANSAC matching [3]. The reduced glare image can then be created using a weighted sum of these aligned images. The next step is to segment the QR code fragment from the rest of the image or reject the image because of video disruption or input noise. Next, QR fragments are aligned on the image space, i.e. the QR code edges are parallel or orthogonal to the image edges. This is accomplished by aligning detected lines on the QR fragments, where line detection can be done using canny edge detection followed by Hough line transform [3]. The image fragment is then resized so that each pixel corresponds to the smallest block on the QR fragment. Here, the representation of the QR code is efficient and the fragments

are ready to be combined into a complete QR code. They are placed based on their position on the screen, as well as the large QR code position markers on each fragment. The complete QR code is scanned using a third-party library, and the scan result is displayed to the human player on the player augmenting app.

Obstacle (Threat) Avoidance

For safety and reliability, the planner avoids all obstacles at any cost. A point cloud is created using a front facing Intel RealSense RGBD camera which is then used to check for the presence of an obstacle directly in front of the vehicle. In the case of an obstacle, it will immediately back up and re-compute the quadcopters' path to avoid the obstacle.

Quadcopter Behavior and Mission Strategy

The behavior planner supports a hierarchy of different behaviors. At the highest level, some supported primary behaviors include taking off, populating arena map, generating and following waypoints while avoiding obstacles, and safe landing. In addition, these behaviors may be overridden by voice commands through the player augmenting application.

The main task of the onboard planner is to continuously build and refine the map; and generate and refine waypoints. Here, the behavior planner can generate waypoints to support the secondary behaviors as itemized below:

- Get QR code: acquire the necessary images from QR fragments to retrieve the decoded passcode
- Heal: direct the vehicle to heal the human player with its healing laser
- Emergency: land the vehicle immediately.

Using the player augmenting app's voice commands, the player can instruct the behavior planner to perform a specific mode. The default behavior of the vehicles is creating maps. Note that the low-level path planning is beyond the responsibility of the behavior planner, which is accomplished by A* using an Octomap.

Object Detection

In order to populate the map used to make behavioral decisions, actors in the arena must be identified. Our visual detection system uses a pretrained YOLO network with its last two layers fine-tuned using a dataset annotated by us in Vatic. The system supports 3 classes: Human, Quadcopter, and bin. The YOLO network works by partitioning an image into regions and predicting bounding boxes of those regions [5]. Because it only takes one network evaluation to arrive at an accurate prediction, the system runs in real time.

Communications System

The communications system consists of a 5GHz WiFi channel. All WiFi communications are through a wireless protocol known as Zero Communications and Marshalling (ZCM). ZCM allows for low-latency multi-process communication.

Man-Machine Interface

A React native front-end application is on a tablet to accept player vocal command input. The tablet transcribes waveform recorded into words, then sends them over the network to the server upon user conformation. Natural language recognition and processing is performed on the server. If the user input matches a predetermined set of commands, then the command is sent to the vehicles over the network. The table also streams video from cameras on the quadcopters.

Multi-Robot Coordination
We are using AprilTags, a fiducial marker system developed at the University of Michigan, to identify and update the pose of friendly quadcopters [6]. AprilTags carry an identity much like a QR code, but they can also be used to compute the rotation and translation of the camera viewing the tag. AprilTags are placed at the 4 corners of each quadcopter. This means that if one of our quadcopters views another quadcopter, the viewing quadcopter knows the identity and the relative pose of the seen quadcopter. This data can be used to merge the maps generated separately by each quadcopter and can also be a helpful datapoint for localization.

RISK REDUCTION
To ensure that code and structure function properly, MAAV performs multiple different tests and simulations. Since the flying of an autonomous quadrotor is potentially dangerous, there are many safety features that are implemented.

Vehicle Design
MAAV takes safety with the highest priority and is continuously exploring ways to create safer vehicles during the design, manufacturing, and testing stages. This year, fully enclosed propeller guards were introduced due to the interaction of a human player on the field. This was a great challenge designing a solution that was safe, yet lightweight and durable.

EMI/RFI Solutions
Circuitry is prone to electromagnetic and radio frequency interference. Fortunately, our data and video streams are transmitted over UDP where the communication protocol checks to make sure all data is successfully sent. In the case of interference, checksums and other error checking procedures invalidate the flawed message.

Electromagnetic interference can also be problematic for an IMU. Magnetometers inside the IMU measure the magnetic field of the earth to determine the IMU's orientation. The Pixhawk relies on the magnetometer for yaw control and we use the magnetometer as an update to our Kalman filter. We have not found that interference causes problems for our magnetometers.

Shock/Vibration Isolation
MAAV tested the sensors under flight conditions and found that the cameras and IMU were sensitive to vibrations while the motors were running at high power settings. To counter this, the cameras have been mounted with fittings that incorporate a foam vibration dampening material. The foam dampening material is sandwiched between the 3D printed camera mount and the carbon fiber center body plate. Similarly, the IMU is mounted to the carbon fiber center body with the vibration dampening foam in the middle.

Safety
MAAV performs flight tests in at M-Air, a unique fully netted and enclosed aerial vehicle testing facility opened by the University of Michigan Robotics Department in 2018.

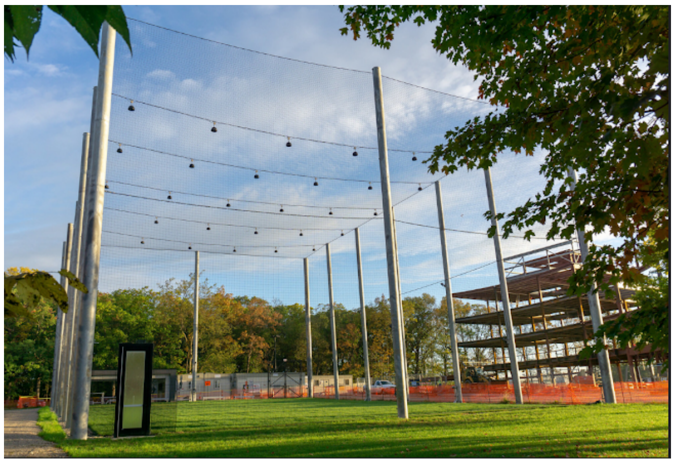
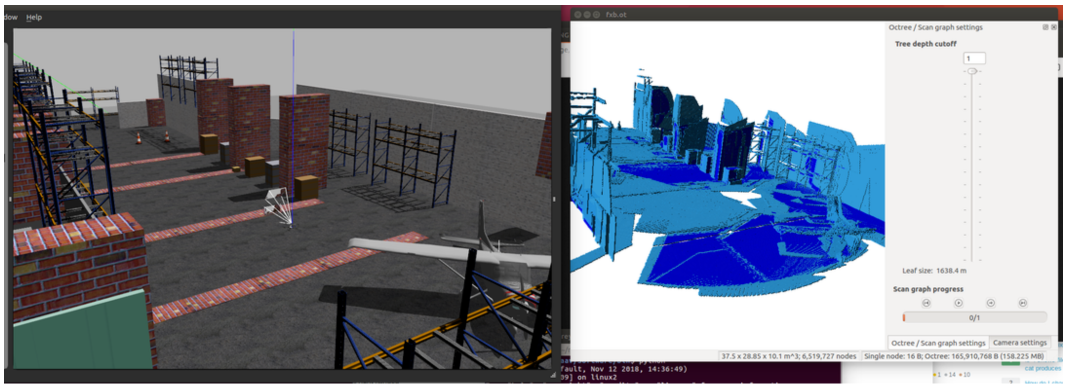


Figure 5. M-Air Outdoor Flight Facility

This 10,000 ft, four-story, netted scientific facility enables the study on autonomy and collaborative robotics in the “wild”. This space enables the evaluation of prototype hardware and algorithms, thereby encouraging risk-taking for rapid exploration of ideas.

While flying, all nearby members are equipped with safety glasses and stay 2 meters away from the vehicle at all times. In addition, the vehicle is always tethered with two ropes to prevent it from harming anyone or from crashing.

Modeling and Simulation
We use GazeboSim (gazebosim.org) as our 6 DoF simulator. We expand upon the software-in-the-loop simulation provided by the PX4 team. We added our own sensor suite, including IMU, LiDAR, depth cameras, and tracking camera. Additionally, we added easy access to ground truth information in order to ease the testing and tuning of our controls, path planning, and strategy. Finally, this ground truth can be used to assess our localization performance. A sample screenshot of the simulator is shown in figure 6.



37.5 x 28.85 x 10.1 m^3; 6,519,727 nodes | Single node: 16 B; Octree: 165,910,768 B (158.225 MB)

```
fault, Nov 12 2018, 14:36:49)
09] on linux2
```

Figure 6. Sample simulator screenshot

Physical Testing
Only after assessing that the software works in the simulator, we test our system in the real world. We use the perfect localization provided by the MAir facility to tune our controller and localization separately. Once we achieve acceptable performance, we test our system fully autonomously.

CONCLUSION
MAAV has designed and constructed two fully-autonomous quadrotors capable of swarm interaction as well as communication through a man-machine interface for their first year participating in IARC Mission 8. The vehicles are expected to autonomously avoid obstacles within the field and act accordingly to player commands. We expect to demonstrate autonomous flight and make significant process towards completing mission objectives.

ACKNOWEGEMENTS
MAAV would like to thank Prof. Ella Atkins, University of Michigan Department of Aerospace Engineering, for her continued support and guidance.

MAAV would also like to thank University of Michigan (U-M) College of Engineering, U-M Department of Aerospace Engineering, U-M Department of Computer Science and Engineering (CSE), U-M Department of Electrical and Computer Engineering (ECE), Northrop Grumman, Aptiv, Raytheon, Saleae, and Saturn Electronics for their generous donations.

REFERENCES
[1] Christoph Hertzberg, Rene Wagner, Udo Frese, and Lutz Schroder. Integrating generic sensorfusion algorithms with sound state representations through encapsulation of manifolds. *Infor-mation Fusion*, 14(1):57–77, 2013.
[2] Intel RealSense Tracking Camera T265. Revision 002. Intel Corporation. 2019.
[3] David A Forsyth and Jean Ponce. *Computer vision: a modern approach*. 2003.
[4] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. 2005.
[5] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. InProceedings of the IEEEconference on computer vision and pattern recognition, pages 7263–7271, 2017.
[6] John Wang and Edwin Olson. Apriltag 2: Efficient and robust fiducial detection. In2016IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 4193–4198. IEEE, 2016