Sprint Report #3

Summary

Team Expeditus was unable to meet the expectations that it set for the conclusion of the first phase, ending with the conclusion of the third sprint. The team began this Sprint adapting to delays in receiving and testing UAV. The final contruction of the UAV was delayed beyond the end of this sprint due to ordering issues. Although the team had made progress on UAV construction, Simulation, and Landing approaches, the team acknowledges that we are currently behind schedule for Phase II. Workdays will be scheduled for team members to make up lost ground before the beginning of Sprint 4.

Team Work

- **Julian Brackins:** Worked on tasks relating to Autonomous Landing.
- **Jonathan Dixon:** Worked on tasks relating to Autonomous Landing.
- **Dylan Geyer:** Worked on tasks relating to assembly of UAV
- **Christopher Smith:** Worked on tasks relating to setting up simulation environment and artificial intelligence landing approach.
- **Steven Huerta:** Worked on tasks relating to setting up simulation environment and artificial intelligence landing approach.

Completed Backlog

Common Development Tasks

- **Setup Simulation Environment.**
  Software simulation of Pixhawk and Ardupilot (both make use of Mavlink which will be the handle used by our team to send commands and receive information from the flight controller). Simulations are successful in setup, however issues with communicating with Software-in-the-loop simulations.
  - Pixhawk simulation (Pix4 ROS SITL): is working. The simulation will successfully emulate the flight controller and set up a simulation in Gazebo using ROS framework. Unable to send instructions to flight controller. Can successfully control UAV with Joystick to provide manual commands. Attempted to setup ground control station software (QGroundControl) to connect to Pixhawk SITL simulation. Unsuccessful in attempts to relay waypoint missions or arm the simulated UAV.
  - Arducopter (Ardupilot SITL with simulated Quadrotor): This simulation is working and the UAV can be controlled with a controller. Unable to communicate missions or controls via Mavlink API. A ROS waypoint publisher was developed to supply the simulation with commands to navigate to designated waypoints via MavROS(ROS wrapper for Mavlink). The publisher is successful in reaching the simulated ardu, however, the simulated UAV is unresponsive to ARM attempts, as well as commands to take-off or to navigate to waypoints.

  Team members are hopeful that using the actual flight controller integration will be much smoother, as networking issues specific to simulated hardware are no longer a possible culprit, limiting the scope of troubleshooting.

- **Acquire parts needed for quadrotor**
  Frame, motors, and ESCs were received by Nov 13th. Waiting for remainder of order, including Flight Controller, GPS, power distribution board, and battery connector.

- **Build UAV**
  Construction began on the UAV. UAV frame is assembled. Motors and ESCs are attached.

As an owner, I want the UAV to autonomously land on the landing pad without damaging the craft

- **Modify/Rewrite implementation as necessary**
  The current setup of three points was found to insufficient to calculate the necessary transformation to unwarp the image. Team members have implemented a method to use a square, rather than a triangle.

As an owner, I want the UAV to autonomously land on the landing pad with the correct orientation.

- **Modify/Rewrite implementation as necessary**
  Changing the number of points from three to four causes some additional problem-solving. With three points, each can easily be assigned a color for clear distinction between points (Red, Green, Blue). The addition of a fourth causes some solving, as before only each of the RGB was used. Testing on simulated images, team members used Black. However, it is acknowledged that this will not be an available color. Once the color problem is solved, orientation will be regained.

Uncompleted Tasks

As a user, I want to communicate the waypoints to the UAV

- **Modify/Rewrite imlementation as necessary**
  - Waypoint Publisher: This is a test implementation to take a file of waypoints generated by a ground control station(GCS) and sequentially read them to supply the simulated flight controller with commands. This publisher is implemented with ROS, which will be the ultimate form of our landing command framework. The publisher is successful in sending messages, however, the UAV is not acting on the messages being passed. More troubleshooting will be needed.
  - Ground Control Station(GCS): There are some great GUI GCSs available. We have attempted to use QGroundControl to connect to the simulated flight controlled. This GCS is suggested by the developers of the flight controller simulations.

As an owner, I want the UAV to autonomously take-off from the landing pad.

- **Modify/Rewrite implementation as necessary**
  This is dependent on successful communication using Mavlink/MavROS. Once the communication problem has been solved, the team will be able to use the GCS to provide the command to take-off. Additional support will not be needed (ie, no need to code custom interface).

As an owner, I want the UAV to autonomously navigate through a set of waypoints.

- **Modify/Rewrite implementation as necessary**
  This is dependent on successful communication using Mavlink/MavROS. Once the communication problem has been solved, the team will be able to use the GCS to provide the command to navigation. Additional support will not be needed.

As an owner, I want the UAV to autonomously return to the location of the landing pad.

- **Modify/Rewrite implementation as necessary**
  This is dependent on successful communication using Mavlink/MavROS. Once the communication problem has been solved, the team will be able to use the GCS to provide the command to navigate back to the landing pad. Additional support will not be needed.

As an owner, I want the UAV to autonomously land on the landing pad without damaging the craft

- **Propose AI Landing Algorithm**
  Team members have discussed approaches, but were unable to dedicate a sufficient amount of time to create or begin an initial prototype.

Prototype

There is a prototype document for Sprint 3 (found here in the repository), where this same material will be covered in much greater detail. This is only a brief description.

- **SIMULATION: Ardu SITL**
  Software in the loop (SITL) was attempted again with many different simulators that work with ROS. One of them implemented there on version of a Mavlink and ROS interface and was not well documented on how to send commands besides direct motor. Another used Mavros and Mavlink for direct communication with the Pixhawk. However, it gave errors of invalid flight control unit and files did not hash properly so believed it was recieving invalid waypoint files that were generated by the mission planner. Hardware in the loop (HITL) was also attempted using the ardupilot with many different simulators. The same errors were also received with the file checksum integrety (MD5) and bot the SITL and HITL gave the same errors with all commands sent. Pixhawk was not attempted since it was received the week after the sprint, but will be attempted over break.

- **SIMULATION: Pix ROS SITL Setup**
  These are the instructions for setting up the ROS SITL simulation. This simulation has the advantage of completely modelling the Pixhawk flight controller, the FCU that will be used by the team. The simulation will start an instance of Gazebo where a representation of a UAV controlled by the FCU is implemented. The UAV may be manually flown by the use of a joystick (XBox controller needed).

- **SIMULATION: QGroundControl Setup**
  These are the instructions for setting up the QGroundControl ground control station software for Ubuntu 14.04. QGroundControl provides a GUI that can link to the UAV's flight controller unit and supply missions such as autonomous take-off, waypoint navigation, and landing.