


CS486C – Senior Capstone Design in Computer Science

Project Description

Project Title: Drone ground control interface for the UAV-RT system	
Sponsor Information: 	Michael Shafer , Associate Professor Dynamic and Active Systems Lab Mechanical Engineering Dept. Northern Arizona University michael.shafer@nau.edu 929-607-6158

Project Overview:

Very high frequency (VHF) radio tags are used throughout the ecology and biology field to monitor both large and small wildlife. The low cost and widespread adoption of VHF transmitters for wildlife localization likely ensures their use into the foreseeable future. Despite their ubiquity, the time required for localizing a tagged animal can significantly contribute to the overall expense of each localization, especially if manned aircraft are required for the search. This increased cost fundamentally limits what we know about the animals under observation.

Over the past three years, our group has worked to integrate a VHF receiver with an unmanned aerial vehicle (UAV), more commonly known as a “drone”. The two-fold benefit of high or low altitude capability and rapid mobility can significantly decrease the time to localize, as well as aid in initial tag detection. The project is currently sponsored by the National Science Foundation.

Our system uses a software-defined radio and onboard Linux computer to record radio data in flight. The current open-loop operational method relies upon user-designed flight plans to dictate the autonomous flight of the vehicle; this means that the drone is programmed to fly a particular pattern in advance, takes off, flies exactly that pattern, and returns to base for analysis of the data. This is executed regardless of what signals it detects along the way. We have been exploring a “closed-loop” approach that may be more efficient and goal-directed: the drone begins flying a pre-programmed search pattern to initially acquire a radio tag but then, once a tag is acquired, it may adjust its plan/pattern based on those signals, i.e., it could act to very quickly refine its search and localize the signal precisely, then return more quickly to allow searchers process its recorded data to provide audio and visual representations of the received signal to aid in localization.

The problem

The central goal of this capstone project will be to integrate two currently separate software portions of this project. The GUI that allows users to connect and control the software running on the UAV is currently programmed in Matlab, and is separate from the ground control station software that is used to plan, execute, and visualize flight missions. In an ideal world, users would not have to flip back and forth between these two applications. This project will leverage either QGroundControl or Mission Planner (both are open source ground control station (GCS) software). The team will integrate the radio system connection and control interface (currently in Matlab) into one of these GCS-software, e.g., as a custom extension module to the core project. The initial portion of this project will include a study of which GCS represents a tenable foundation from software integration and forward compatibility standpoints.

Specifically, the desired product is a new module that extends the chosen GCS base framework with the specialized UAV control software. Ideally, the product would work equally well for Windows and MacOS versions of the product, but if this is not feasible, a specific target for the prototype will be chosen. The integrated interface will:

1. Automatically and/or manually connect to the UAV companion computer (CC) via a wireless UDP connection.
2. Receive heartbeat and other status messages issued by the CC, visualizing them for the user.

3. Develop/improve command confirmation messages, and implement retransmission of commands as necessary
4. Send start and stop commands to the radio systems.
5. Display terminal information provided by the UAV CC.
6. Allow for the download of flight data via the FTP server hosted by the CC.
7. Catch and recover from errors.

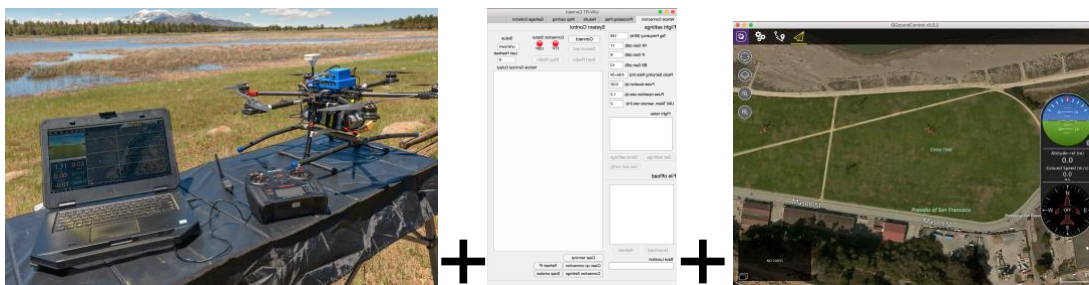
This initial list is subject to change after the initial design review and discussions between the client and the team.

If these goals are accomplished and thoroughly tested the following secondary objectives should be implemented:

8. Automatic data download upon mission completion.
9. Development of live data visualization capacity in the GCS. (Onboard CC data processing will be handled by the research group.) This objective is to receive messages, over the UDP link and plot the data on the live map as the vehicle is in flight.
10. Transition the communication from the UDP link to a 915 MHz radio telemetry link (which we already fly) using custom Mavlink messages to improve the range over which data can be communicated to the GCS.

Project website is available at: <https://uavrt.nau.edu/>

UAV-RT System + UAV-RT Connect Software + QGroundControl = Capstone Project



Knowledge, skills, and expertise required for this project:

Familiarity or ability to quickly learn:

- The basic concepts of drone control and flight programming
- Python, C++, C, Matlab,
- Git, Linux systems,
- UI Design, Creativity, and strong and clear documentation and technical writing skills

Equipment Requirements:

- There should be software or IDEs required other than a development platform and software/tools freely available online.
- A complete flight system will be provided to the team for testing purposes.

Software and other Deliverables:

1. The software applications as described above, deployed and tested successfully with real data. The product should be deployable on both Windows and Mac OSX. Code will be hosted and documented on the client's Github page.
2. A strong as-built report detailing the design and implementation of the product in a complete, clear and professional manner. This document should provide a strong basis for future development of the product. Should be written in language and with sufficient detail for non-C.S. (but technically proficient users) to understand the product's function and edit/extend the code.

3. Complete professionally-documented codebase, delivered both as a repository in GitHub, BitBucket, or some other version control repository; and as a physical archive on a USB drive
4. Must include a complete and clear User Manual for configuring and operating the software, written for the (non-coding) engineers and scientists that will deploy this product.
5. A professional video tutorial on software operation posted to the client's YouTube channel.