Overview

The UAS MissionPlanner repository is forked from an old build of the diydrones MissionPlanner repo. It was initially designed to conform to the UI of the MissionPlanner and use Mavlink as the protocol to send messages back and forth to the plane. The current build, however, abandons some of the UI in place to load up a slightly modified OpenTLD application built by github user gnebehay, communicating with this third party process but leaving the UI to it. The 4 main components of the UAS part of the MissionPlanner are the UAS view, the Targeting HUD, the Servo Controller and the OpenTLD implementation, which will be discussed down below.

I have heard we will probably be migrating away from the MissionPlanner and the Mavlink protocol, which I recommend doing. The DIYDrones' MissionPlanner has an impressive amount of features, but most of them are not relevant to the project and only serve to bloat up the mission planner or complicate certain matters. It would be easier to start from the ground up, or from a simpler project, and build our core functionality on that, where our design benefits our functionality instead of sticking to a larger project's design.

GCSViews/UAS.cs

This file holds the UAS view class which handles most of the UI that we have implmented. It has multiple controls that use the Mavlink com. port to send messages to the plane. One notable thing is the recent addition of the internal TLDTracker class, which runs on a separate thread to run the OpenTLD application, read values from it and communicate with the camera's servos to move the camera towards the tracked object. This class uses the ServoController.cs class which will be discussed next.

ServoController.cs

The ServoController class receives and sends messages to the Arduino board connected to it. The protocol is briefly explained within the class. The gist of it is that it receives a message when the board is ready for a command, sends one of two commands (Absolute Angle or Relative Angle) to the board and then sends it the pan and tilt values separated by a'B'. The code is self-explanatory.

Controls/TargetingHUD.cs

The TargetingHUD was originally designed to be the UI element that allowed a user to pick a target visible on a camera. The initial plan was to have the coordinates of the target sent up to the plane's computer and have that run the TLD tracker and send back the new coordinates for the target to

visually show where the algorithm thinks the target is at any given moment. Once you enable the camera in the configuration menu, you can click and drag anywhere on the screen to send the coordinates up through Mavlink. There are 4 lines commented out which are supposed to update the target box based on information received by Mavlink, but as stated before this design has changed since.

OpenTLD

This is highly unmodified code. The only change made to the code was to expose the location of the target on the screen to outside processes. While this output was recorded into a file initially, the modification makes it put all that data into the command line as it gets it, so other processes can read it. This data is then used by the TLDTracker class mentioned above to determine where to move the camera.

PyMavlink

While not mentioned above, PyMavlink is a program used by the MissionPlanner to generate a C# class with all the different message types that it can identify. While this would be unnecessary information if we are moving away from Mavlink or the DIYDrones MissionPlanner, I'm including this information in case development on the MissionPlanner continues. PyMavlink is an implementation of the Mavlink protocol that has a generator that generates interfaces for Mavlink protocol messages in different languages. However, since I am unsure of which version of this program the DIYDrones MissionPlanner uses and since the messages that have been created for the UAS Mission Planner are not being used anymore, I am not including that code within this repository. However, adding new message types to the protocol is easy with the mavgen tool, as long as the right version is used.