James Irwin
Kyle Finch
Brooks Kindle
CptS 483 Robotics
Final Project

Link to video: http://youtu.be/5GO0mW9KMgk

Our Idea:

Our idea, in a nutshell, ~~was to win the best project competition~~ was to have a single ARDrone follow a predetermined path of a given color that could have any number of twists and turns in it until encountered the finish line, which was a different color than the line. Initially we would simulate our environment using gazebo with the drone taking off from the start of the line. Once we verified ~~that the ARDrone was as awesome as we expected it to be and~~ that it could follow said path accurately, we would then have the drone start where it could not see the line. After the drone could successfully locate the line and proceed to follow it with a reasonable amount of accuracy, then we would move on to the physical drone and test it with that.

What worked/what didn't

What worked:

Our final demo:

It's pretty cool, you should watch our video on it.

Our line centering algorithm:

To ensure that we could continue following the line, we had to make sure that the line was in our sights as often as possible. To perform this, we implemented two types of corrections for line centering, turn and bank. A turn consisted of an angular movement in the z direction and a bank consisted of an angular change in the y direction. Turn movements helped straighten out our drone after it had found the line or performed a turn, while bank movements helped ensure that we were flying directly over the line, rather than parallel to it.

State machine for logic processing:

We structured our solution around a state machine; the camera data that the drone is being fed determines its internal state, which in turn dictates how the robot responds to external stimuli. This allowed us to easily compartmentalize different logical actions.

What didn't:

Our initial image processing algorithm was incredibly inefficient and slow; so slow that the image feed from the bottom camera was collecting images faster than they were being processed, causing the drone to become laggy and unresponsive. However, we were able to find a solution that was both more efficient in terms of speed as well as a lot easier to understand.

Due to time constraints, we were unable to test our code on a physical ARDrone. As this was one of our stretch goals this is all right, however it would have been nice to at least get a physical drone to follow a some sort of a path.

Our initial line finding algorithm is very simplistic. The main goal of the project was line following, not search. However, the framework we designed allows us to easily modify the line searching functionality. The current search algorithm simply has the AR Drone fly forwards until a line is visible.

Next Project Steps

The next most obvious step for this project -- and probably the coolest -- would be to modify our code so that it worked on a physical drone rather than just in simulation.

It would be also very important to modify our project so that a drone could handle situations with multiple lines in its vision. At the moment we only assume one path exists in the field of vision, and introducing more paths would allow the drone to follow a single path when next to multiple others of the same color.

In addition, introducing a PID controller to help with our turns and banks would allow us to tackle sharper turns and greater offset corrections while minimizing error when performing smaller adjustments.

A more intelligent line searching algorithm would also be necessary. Our current line searching algorithm is one that simply moves forward until it finds the line; this would obviously not work in any kind of real world situation. To improve on this, we would have the AR Drone fly in a spiral search pattern, increasing altitude as it spiraled. The greater altitude would allow the camera to view more of the environment. Once a line had been spotted, we would center ourselves with the lines and return to a typical line following altitude. One potential issue with this solution is that as we increased altitude, we might see more than one line, so this algorithm would be dependent on our ability to handle multiple visible lines.