Quadrocopter Flight Control

Project Proposal

By: Eric Backman

Advisor: Dr. Malinowski

Date: 12/8/12

Abstract

This project will use microcontroller Linux to control a quadrocopter. Most of the work will be integrating the BeagleBoard with the sensors, joystick and camera in order to create simple flight. The BeagleBoard will then send these commands to a slave microcontroller that will simulate the RC standard signals required by the quadrocopter flight controller. The quadrocopter will take wireless signals from a joystick initially to allow for manual control. Simple object avoidance will be implemented using infrared sensors. Eventually some simple image processing will be used on the camera images to allow for color coded landing. This should all be simple to accomplish by the end of the year and will allow for future senior projects.

Project Summary

Quadrocopters are ideal for small scale flying robots due to their ability to hover like a helicopter without the need to change the rotor blade pitch angle. This simplifies their design and control. Last year, Brad Bergerhouse, Nelson Gaske, and Austin Wenzel worked on an autonomous quadrocopter. They constructed the quadrocopter platform, installed a real time operating system on BeagleBoard, and started sensor implementation.

The primary goal for this year's project is to get the quadrocopter flying and responding to inputs from sensors using microcontroller Linux. The platform is already built but it needs controllers and sensors to function. A joystick will be connected to another Beagleboard or PC that will connect wirelessly to the onboard controller. This will allow a user to control the quadrocopter. Simple object avoidance will be programmed to prevent the quadrocopter from crashing into walls, as well as simple flight patterns to allow for semi-autonomous flight and navigation. Lastly, a camera will be integrated into the system to allow for better navigation.

System Block Diagram

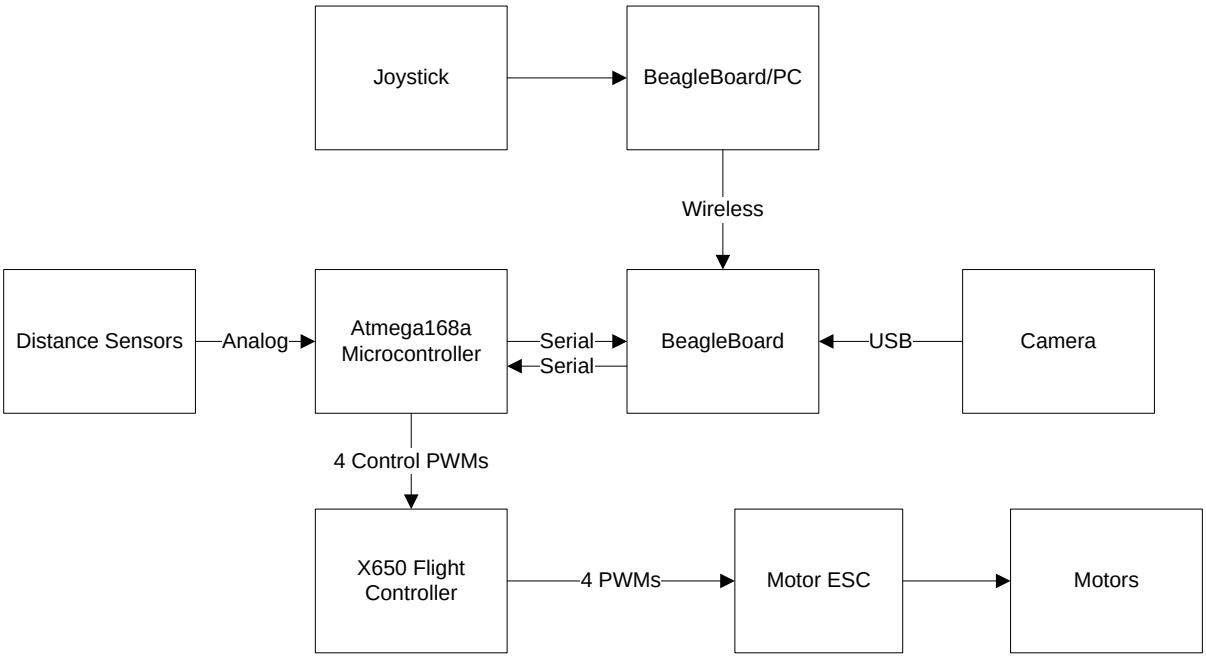

Figure 1. Quadrocopter System Block Diagram

The project is to replace the remote control portion of the remote control quadrocopter platform with my semi-autonomous control system. The quadrocopter already has a built in flight controller with accelerometers to maintain stability without user input. Controlling this requires me to simulate the RC signals to the flight controller. This will be done by using the Atmega's timers to create 50 Hz pulse width modulation (PWM) signals with different duty cycles corresponding to different commands.

The Atmega controller will use the built in analog to digital converters to take the inputs from the sensors and send them to the BeagleBoard through a serial connection. The BeagleBoard will take these inputs and use them to influence the command coming either from a controller or based on pre-

programmed autonomous decisions. It will generate a signal for yaw, pitch, roll and elevation and it will send these signals to the Atmega controller that will generate the required PWMs and give them to the built-in flight controller. The flight controller will take the commands and generate the PWMs for the four motors.

Overall System Requirements

- Quadrocopter can take off and land autonomously
- Quadrocopter will be able to fly for at least 10 minutes per charge
- Able to autonomously fly through corridors without crashing into walls
- Able to land on color coded pad using camera

Sensors

Distance sensors will be chosen and integrated into the system to provide object avoidance. These will generate an analog voltage that will be converted into a distance and implemented into the controls in BeagleBoard. The sensor array will be designed so that all sides are watched preventing any collisions.

The sensor array is as shown in Fig. 2 below, 6 sensors at 90 degree angles. This will allow simple object avoidance requiring minimal sensors and controls. This is the most simple arrangement that allows full coverage and give me the time to finish the whole project.

One of the main problems with sensors on a quadrocopter is how they handle the tilt during motion. Since the sensor will be pointed at a downward angle in the direction it is moving, a distance larger than the true horizontal distance will be recorded. This angle can be estimated using the control input but this will not be very accurate unless the angle can be measured in some way. This could be controlled by implementing an accelerometer and using that to find the angle but this is unnecessary if I have a large enough safety margin built in.

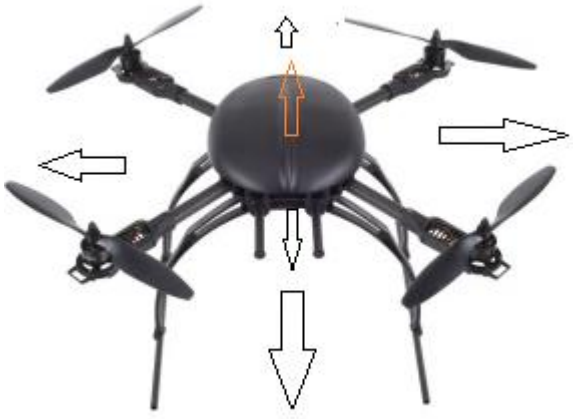

Fig. 2 Infrared Sensor Arrangement

Requirements:

- Sense distance to all objects within at least 1 meter with accuracy of 1 cm
- Distance sent to microprocessor from 6 sensors at least every millisecond

Controllers

BeagleBoard will start by receiving commands wirelessly but eventually will have navigation for autonomous flight. It will use these commands as well as the sensor inputs to create commands to be

sent to the PWM generating microcontroller. It will send these commands to the microcontroller via I2C. The microcontroller will generate the PWMs that are inputs to the flight controller. The flight controller comes with stability control and handles all the difficult controls associated with flying. It generates the PWMs that are sent directly to the motor ESCs which control the motors.

Microprocessor Requirements:

- Read the distance sensor voltage with accuracy equivalent to at least 1 cm accuracy of measured distance
- Send distance sensor data to BeagleBoard at least every 10 ms using serial port
- Create Pulse Width Modulation signal of 50 Hz that has at least 10 us resolution

BeagleBoard Requirements:

- Analyze distance sensor data to update command PWM settings every 20 ms
- Filter camera images to determine a certain color coded instruction
- Analyze images acquired from the camera to supply a command that updates PWM settings every 20 ms
- Capable of communicating using 802.11 WiFi infrastructure
- Lightweight underlying operating system of Linux family such as Angstrom or Microcontroller Linux

Camera

The camera will be placed at the bottom of the quadrocopter and allowing it to help in navigation. Simple digital signal processing will be implemented in the BeagleBoard to create navigation such as simple color following.

Camera Requirements:

- At least 1 Megapixel for accurate identification
- At least 10 frames per second

Control Modes

By the end of the project, I hope to implement 3 different control modes. There will be autonomous take-offs and landings. These will be activated wirelessly and should run until the quadrocopter is either hovering or completely powered down. Next, it will have wireless joystick control with object avoidance using the infrared sensors. Last, it will have a color coded landing sequence that will use the camera to hover over a certain color landing pad and land on it.

Testing Milestones

In order to successfully test features, keep it safe, and to stay on schedule, multiple control milestones will be implemented. The first run that will be achieved is a path that takes off, hovers for a second and then sets gently back down. This will prove stability and that the system works as a whole. It will also prove the ability to maintain a height using just the sensor pointing down.

The next attempt will be to have it lift off, fly in a certain direction, and then set down. This will prove stability in flight and will allow to me tune the control inputs for movement. Next, I will hook up the joystick wirelessly and use that to control the flight. This will allow me to control the object avoidance using the sensors. Finally, I hope to implement the camera to identify a landing pad. This path will take off, fly until it sees the landing pad, and will set down on the pad. This will allow me to test my

digital image processing algorithm as well as show everything works together. Once this is working, all my project goals will have been accomplished.

Completed Work

So far I have focused on the Atmega168 programming. Successfully implemented the analog to digital conversion and PWM signals that interact with the flight controller with 0.8% resolution. Also started looking into the serial communication between the Atmega168 and BeagleBoard.

Schedule

Tasks	January	February	March	April
Joystick				
Sensor Integration				
Mount Hardware				
Flight Testing				
Image Processing				
Finalizing Project				
Final Report				

Equipment

XAircraft X650
BeagleBoard xM
Atmel Atmega168a
6 Sharp GP2Y0A02YK0F Infrared Sensors
Li-Poly Battery
Battery Charger
5V Voltage Regulator
MAX 232

References

[1] Brad Bergerhouse, Nelson Gaske, Austin Wenzel. Aerial Collision Avoidance System. Senior Project, Electrical and Computer Engineering Department, Bradley University, May 2012, http://cegt201.bradley.edu/projects/proj2012/quadcptr/

[2] Introduction to Autonomous Mobile Robots, 2/ed., by R. Siegwart, I. R. Nourbakhsh, D. Scaramuzza, MIT Press, 2011, ISBN: 978-0262015356