Cellular Network Unmanned Aerial Vehicle

Computer Engineering Project Proposal

Team:

Nic McDonald
nic.mcdonald@utah.edu

Grant Ayers
ayers@cs.utah.edu

Project Website:

http://pisco.flux.utah.edu/uav

Department of Electrical and Computer Engineering
University of Utah

Table of Contents

Abstract

Unmanned Aerial Vehicles (UAVs) are used for many purposes when employing a pilot is impossible, dangerous, or too expensive. For small-scale UAVs the communication link is typically line-of-sight, which greatly limits the distance of such vehicles. We present a communication system that utilizes the existing cellular network to remotely pilot a small-scale UAV. This system enables a vehicle with sufficient power to travel anywhere in the world where cell coverage is available.

Introduction

Unmanned Aerial Vehicles are used in a variety of applications spanning from large-scale military grade hunter-killer surveillance vehicles to small-scale personal and industrial vehicles. However, almost all remote control consumer vehicles have very short-range communication systems. These systems are typically line-of-sight links and have a range of a few hundred meters. Overcoming this limitation is the purpose and intent of our project.

A true UAV should be able to travel long distances and be tolerant of communication interruptions. Unfortunately, this type of technology is largely inaccessible to non-military grade products. Furthermore, FCC regulations limit the amount of power that can be transmitted over open frequencies. However, several large-scale ubiquitous data networks already exist, including the mobile phone infrastructure. These types of networks allow for relatively fast data communication, are accessible to the public, and require little additional radio equipment.

The goal of our project is to develop a communications link through an existing mobile network and use it to link a ground station to a remote control helicopter. We will design the hardware and software required to control the helicopter and enable it to fly via remote commands. This system will be fault-tolerant when the latency of the link is too high, or when the link disconnects during flight. On-board logic will enable the helicopter to back-trace its path or land safely if communication cannot be reestablished. Additionally, several sensors will be employed to monitor the state of the helicopter and provide this data to the ground station. These sensors include proximity detectors, accelerometers, a compass, a GPS module, and a camera.

A high-level system diagram of our intended project is shown in Appendix I.

Hardware Implementation

Align T-Rex 600 ESP

We have chosen the Align T-Rex 600 ESP (shown in Figure 1) electric helicopter as our platform because we have received sponsorship from L3 Communications. The T-Rex 600 helicopter is a relatively large RC helicopter with a length of 45.67 inches and a main rotor diameter of 53.15 inches. Without additional equipment placed upon it, the T-Rex 600 has a flying weight of 7.3 pounds. It is driven by a brushless electric motor that is powered by a 7.4v 1900mAh lithium polymer battery. Typical flight time is approximately 8 minutes when flying lightly.

The T-Rex 600 has a payload of approximately 3-5 pounds. A larger payload contributes to a shorter flight time and higher vertical instability. We estimate that our system will add 2-3 pounds to the weight of the vehicle. Due to this additional weight our system will need to utilize altitude and distance-to-ground stabilization methods. This will be measured using proximity sensors and GPS, which will be discussed in later sections.

Figure 1 - Align T-Rex 600 ESP

VIA EPIA PX x86

The VIA Technologies EPIA PX x86 embedded board is the heart of our on-board processing system. We call this piece of our system the Central Processing Unit (CPU). We chose this board because it implements a generic x86 system with all the common I/O ports of a small computer. Our only I/O requirement for this board is USB. We need USB to connect communicate with the cellular network transceiver, webcam, and Sensor & Servo Processing Unit (SSPU). Some additional features of this board that we will utilize are Serial ATA for the hard drive, DDR2 533MHz memory, and VGA for development and debugging. This board is extremely small but still utilizes an efficient processor and chipset. Its dimensions are 10cm by 7.2cm. A full system with a memory module and 2.5" SATA hard drive weights about 1 pound. The VIA EPIA PX is shown in Appendix II as item 1.

Cellular Network Transceiver

The cellular network transceiver will allow us to connect to the internet remotely. It is the crux of our project but its specific implementation is irrelevant. We decided to create a system in which any cellular USB modem and manufacturer-supplied software can be used. There are numerous cellular networks around the globe. Our project will be a generic implementation such that any modern cellular network can be used. A cellular network USB modem is shown in Appendix II as item 2.

Nearly every manufacturer of USB cellular network modems provides drivers and software for Windows operating systems only. This was our main driving constraint in deciding to use the x86 board described earlier. The Windows operating systems will allow us to use the manufacturer-supplied software to connect to the internet using the USB transceiver. This also allows us to focus our efforts on the main projects goals instead of writing vendor specific drivers.

Logitech Webcam

The camera we chose to use for our project is the Logitech C200 webcam. Since we were already forced into using a Windows operating system, a Logitech webcam works well because it implements the DirectShow multimedia framework and API. Using DirectShow in the Windows environment is very simple and requires only basic system calls. The Logitech C200 webcam has many settings which will allow us to adjust the color, brightness, etc. This camera can be seen in Appendix II as item 3.

ATmega 1280

We have chosen to use the ATmega 1280 microcontroller for the Sensor & Servo Processing Unit (SSPU). The ATmega 1280 is an 8-bit microcontroller with 128kB of flash memory and 8kB of SRAM. It works well for our system because of its numerous I/O ports. It has 54 digital I/O pins (14 of which provide pulse width modulation (PWM) output), 16 analog input pins, 4 UARTs, I2C, and SPI. The ATmega 1280 implemented in an Arduino Mega board is shown in Appendix II as item 4.

The SSPU has two main tasks: translating positional and directional commands into servo positions; and reporting all sensor data to the CPU. As stated earlier, the CPU will do all the main processing. The SSPU is basically just a hardware interface module. It will control all measurement and positional hardware.

Flight Control Servos

The servos on the T-Rex 600 helicopter accept pulse width modulation (PWM) signals for controlling the motor position. PWM for servos is quite different than general PWM. Servo PWM sets the position using the signal high time rather than duty cycle. The difference is that duty cycle is a ratio between high and low whereas servo PWM just measures the high time and sets the position accordingly. Most servos input PWM ranges from 1ms to 2ms. For example, on a 180° servo, 1ms maps to 0°, 1.5ms maps to 90°, and 2ms maps to 180°. Each time the signal is sent the servo controller moves the motor to the proper position. When desired, the same signal is sent at 25-40Hz to hold a position. An example of servo PWM is shown in Figure 2. The servos found in our system are shown in Appendix II as item 6.

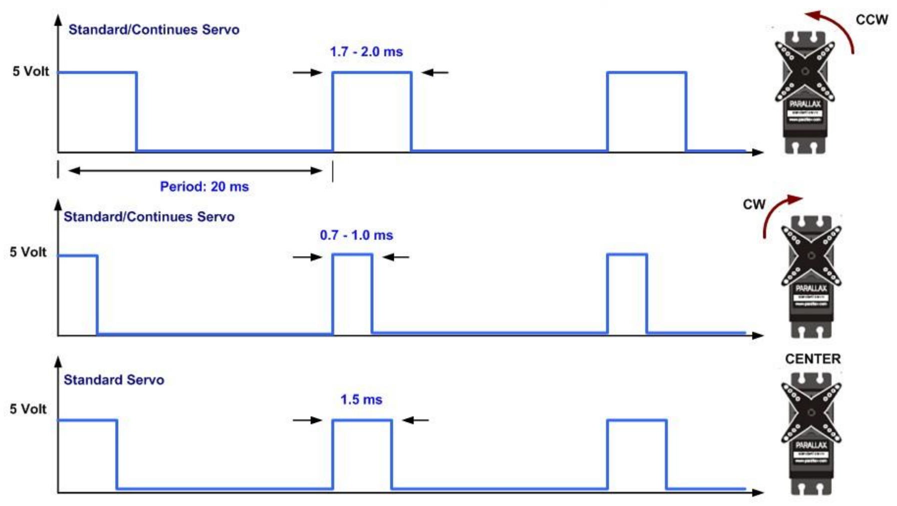


Figure 2 - Servo Pulse Width Modulation (PWM)

Helicommand 3A Stabilization System

RC Helicopters are extremely hard to control, especially when wind is present. For this reason, we will retrofit the helicopter with a Helicommand 3A stabilization system. This stabilization kit is fitted between the SSPU and the servos. It acts as a filter that stabilizes the helicopter when the controls are neutral. Controls are defined as neutral when their inputs signals are 1.5ms PWM (90° position). The stabilization system utilizes accelerometers to adjust the helicopter's tilt and keep it level. It also has a setting for position hold in which it utilizes a CCD camera that is pointed at the ground. It is able to do this when the helicopter's distance from the ground is less than 9 meters. The stabilization system is shown in Appendix II as item 5.

Locosys LS20031 GPS Receiver

We chose to use the Locosys LS20031 GPS receiver because it has a standard serial interface and has an update rate of 5Hz. It can be configured to output several different standard GPS protocols. This module is shown in Appendix II as item 7.

Any normal GPS system has inaccuracy that our system will need to be able to handle. GPS is generally accurate within about 10-20 feet. GPS data is often more accurate the faster the vehicle is moving. Because our vehicle is relatively slow, we will need to guarantee that we only rely on the GPS data for statistical reference.

GPS data includes global position and altitude, heading, velocity, and time. Heading is only valid when the vehicle is moving. This requires our system to utilize a compass to determine the heading when the UAV is hovering in place.

Our software will implement a predefined GPS flight path option. This will allow a user to set a full flight path before the vehicle has taken off. The user of the system will need to guarantee that the given GPS flight paths are widely unobstructed.

The GPS data will also be used for fail-safe backup when all communication is lost. The UAV will be able to backtrack the path already taken to find its way home. This also requires the user to guarantee that the path backward is always unobstructed.

Maxbotix LV-EZ0 Ultrasonic Sensor

We will be utilizing a Maxbotix LV-EZ0 ultrasonic range finder to determine the distance to the ground. This specific range finder has a maximum distance of 255 inches and a precision of 1 inch. We will use this sensor for take-off and landing. This sensor will be a crucial part of our fail-safe system when all communication is lost with the aircraft. If the desired fail-safe is a landing, the ultrasonic sensor will play a key part in determining distance to ground. This sensor is shown in Appendix II as item 8.

Honeywell HMC6352 Compass Module

As mentioned earlier, GPS heading data is only valid when the vehicle is moving. For autonomous flight, we will need to utilize a compass to determine the heading when the vehicle's velocity is near zero. This module is shown in Appendix II as item 9.

We have decided to use the Honeywell HMC6352 compass module. Our decision is based mainly on price. This compass is regular price but has no tilt compensation. Compass modules with tilt compensation will typically be 5 times more expensive. Because of the tilt limitation, we will have to be careful when relying on this module. Our algorithm for determining the heading will need to utilize predictive GPS data and an average of recorded compass data. Once a heading estimate has been made, the vehicle will begin to move and the GPS data will become more accurate and our reliance upon the compass will decrease.

Freescale MMA7260Q Triple-Axis Accelerometer

We will be using a Freescale MMA7260Q 3-axis accelerometer to determine the tilt of the helicopter. This will be used for stability when the PWM signals are not in the neutral position. We will use a digital feedback controller to adjust the servo position according to the desired tilt. The tilt measurements will also be the main means to controlling the direction of the helicopter. The desired tilt in the feedback system will be set to the proper tilt to give the desired direction of flight. The accelerometer is shown in Appendix II as item 10.

Miscellaneous Equipment

We will most likely be fitting the UAV with some additional equipment as needed or desired. For example, we will most likely put simple sensors on board to measure temperatures on the helicopter and the ambient air temperature. We would like to also have an auxiliary servo mounted on the aircraft for functions that are now unspecified. One example could be a pan & tilt servo system for the webcam. With this we could have the flexibility of viewing different positions without physically pointing the helicopter in a different direction. Another option for an auxiliary servo is an arm that can drop something. This could be anything from flower petals to rocket propelled grenades.

Power System

Even though the helicopter already has an on-board lithium polymer (LiPo) battery, it isn't good practice to tap into its power. We have decided to add another LiPo battery to aircraft for all the on-board electronics. Due to the dangerous nature of LiPo batteries we will be making a circuit to monitor the battery's voltage and temperature. Upon a low voltage or high temperature signal, the CPU will request an emergency landing from the ground station. Upon extreme conditions, the CPU will just land the UAV without permission from the ground station.

The voltage monitor and temperature sensor will be a simple circuitry connected to the battery itself. The signals of such will be fed into the SSPU.

Software Implementation

A diagram of the entire software interface is shown in Appendix III. The software of our system is broken into the following sections:

Aircraft Central Processing Unit

The Central Processing Unit (CPU) is the major processing portion of the aircraft. This is implemented on the VIA Technologies EPIA PX x86 board. It will be responsible for implementing all flight plans, paths, and commands and maintaining the flight controls. It will also be responsible for the webcam video feed and statistical feedback to the ground stations. The processing portion of this system will be implemented in C/C++. We will use VideoLAN VLC for the webcam video capture. VLC is able to capture the DirectShow interface and send it to the server. It has many options for transcoding and encoding formats and sizes.

Aircraft Sensor & Servo Processing Unit

The software for the Sensor & Servo Processing Unit (SSPU) will be implemented on the ATmega 1280 running as an Arduino environment which uses a C/C++ like language. This system will decode commands from the CPU and set the corresponding servo positions accordingly. It will also retrieve and report all sensor data to the CPU on a regular basis. Because the microcontroller will have approximately 10 hardware interfaces, we'd like to offload as much processing as we can to the CPU. This microcontroller will basically be a hardware interfacing module. The only real computation it will be doing is translating the CPU's directional and velocity commands into servo positions. This will be accomplished using the feedback system previously mentioned in the accelerometer section.

Forwarding Server

We will utilize a ground-based server to enable multiple client connections to the helicopter. This will allow several client connections without requiring additional power or computational resources on the helicopter, or more bandwidth on the cellular data link. The helicopter will only communicate with this server, and thus will only need to send one video stream and maintain only one command and status connection. Clients connect to the ground-based forwarding server and request one or more services. Only one client will have control of the helicopter at a time, but many clients may be able to concurrently view the video feed or status readings. At a minimum, we plan to run a VLC server which will listen for incoming requests for the video feed and forward the video stream to the clients. If time permits, we will also implement a similar system for status information.

Private Ground Station

The private ground station will control the helicopter through the forwarding server. We will implement its software in a C#-based GUI. The interface will show status information in a cockpit-like instrument view. The instruments will correspond to each of the sensors on the helicopter. This interface will also allow command input, in the form of simple raw commands (such as servo positions or rotor angles), and more advanced path commands (such as relative and GPS-based path control). The video feed will not be a part of the C# program, but will instead be web-based as described in "Public Ground Station" below.

Public Ground Station

The public ground station will provide certain data to multiple clients in an html-based interface. The web server is run on the forwarding server. At a minimum, multiple clients will be able to connect to the forwarding server and view the live video feed. We may choose to extend this and allow the sensor data to be viewed as well. This will either be a view-only version of the private ground station software, or a simpler web-based instrument view.

Schedule

Our project has five major milestones. Our first milestone is to obtain all hardware and have a complete high level system design. After this is complete we can begin working on low-level hardware and software interfaces and protocols. This milestone is scheduled for 5/5/2010.

Our second milestone is only eleven days later. It is to have declared all the software interfaces and protocols for our system. This will enable us to work on separate portions of code while adhering to predefined standards. This milestone is scheduled for 5/16/2010.

Our next milestone is to have a working draft for the basic control system. This milestone will allow us to start testing autonomous flight. After this we will perfect our draft design until it is deemed stable. This milestone is scheduled for 7/16/2010.

Our next milestone is to properly fly the UAV autonomously. This is a crucial milestone because we will need solid autonomous flight data in order to create stable sets of code for the various software areas. This milestone is scheduled for 8/10/2010.

Our last milestone is demonstration day for all senior projects. This is when we need everything working as planned. At this point all hardware and software implementations should be completed. Demonstration day is scheduled for 12/9/2010.

A complete flow chart of our schedule and tasking can be found in Appendix IV.

Bill of Materials

The major cost portion of our project is the Helicopter assembly itself. L3 Communications has given us approval to use their RC Helicopter. They have also agreed to buy the Helicommand 3A stabilization kit. This sponsorship contributes $2,500 to our project.

The second largest cost to our project is the VIA EPIA PX embedded board. VIA Technologies has sponsored this portion of our project. This sponsorship contributes $250 to our project. To get the full x86 system working, we had to buy a SO-DIMM memory module and a hard drive.

The rest of the project costs will consist of several smaller hardware devices, many of which have already been obtained. All of the hardware needed for this project will be used on the aircraft. The ground stations consist of computers only and are free for us.

A full bill of materials is shown in Appendix V.

Risks

Several risks are involved in this project. Almost all of them come from the following project characteristics: It's ambitious, we are a two-person team, non-trivial costs are involved, and we're flying a helicopter. Obviously, we may wreck the helicopter. Other projects using helicopters have done this before. They are not easy to fly manually, or especially autonomously. If our helicopter is disabled we may have no alternate vehicle with which to complete our project. The other major risk is our inability to complete our full specification in the allotted time. To help with this, we are planning on working through the summer and have set early deadlines.

Most of the other major risks from early on have been minimized. We were unsure if we could acquire a vehicle and pay for our equipment. We already have almost all of our hardware and a helicopter is on the way. We were also unsure of the hardware interface between the cell network and servos, but now we have a clear understanding of how to make this work.

We accept all risks because of the payoff.

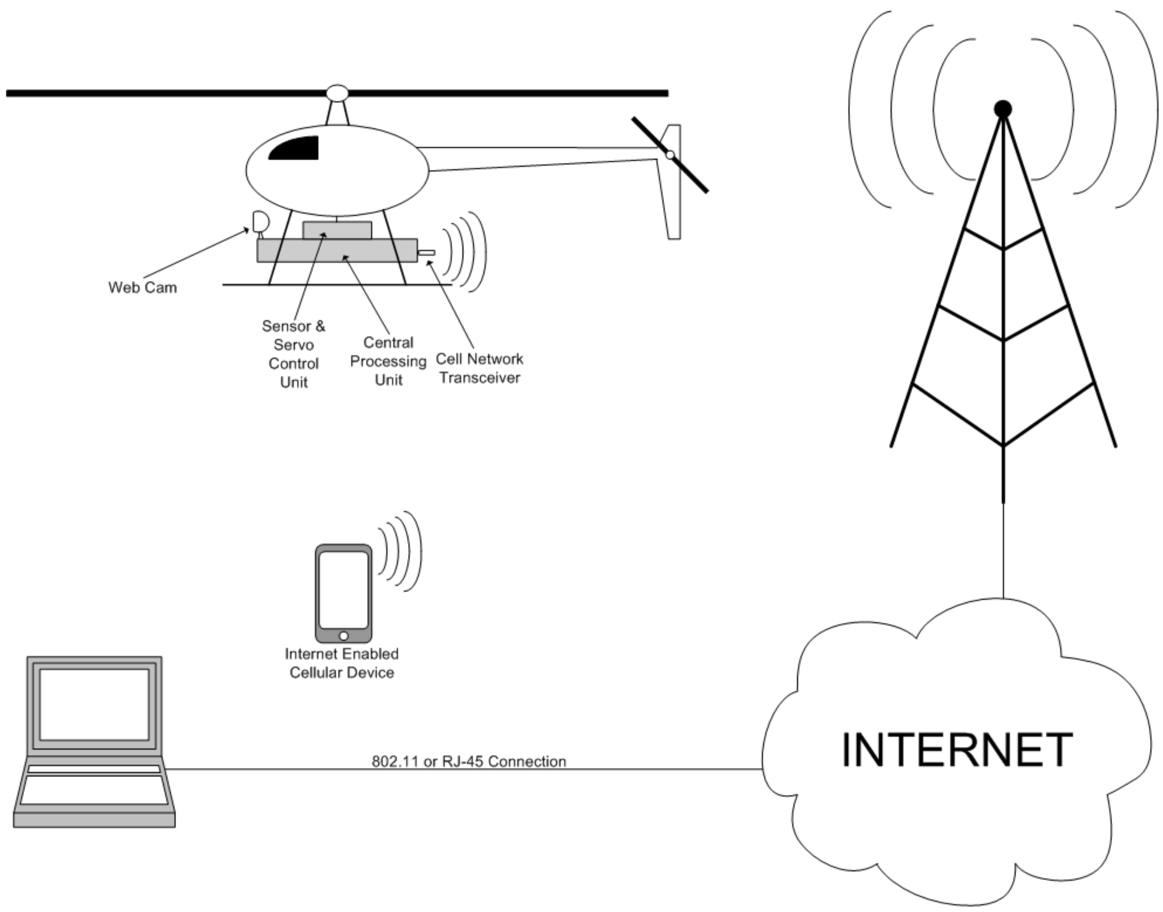
Conclusion

Unmanned Aerial Vehicles are useful for many purposes, but require long-range communication links and fault-tolerant logic when the communication links are inoperable. This kind of technology is largely inaccessible to non-military applications. We have presented a communications system for a helicopter UAV that utilizes the existing public cellular network infrastructure. We have also presented the hardware and software requirements to support control, monitoring, and fault tolerance for this vehicle, and outlined the specific design strategy to implement our solution. We have shown the impact of this project to be worth the risk, and plan to have a fully-completed project by December, 2010.

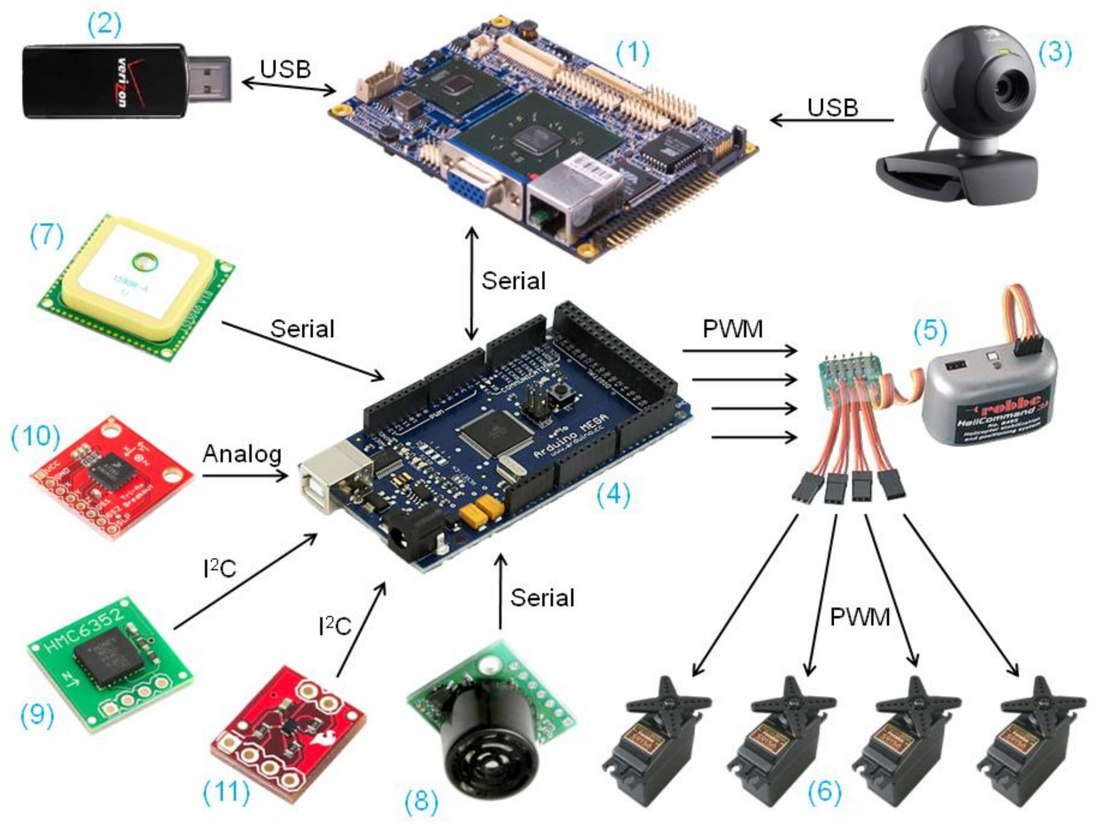
Acknowledgements

We would like to thank our sponsors, L3 Communications and VIA Technologies, without whom our project would have been impossible.

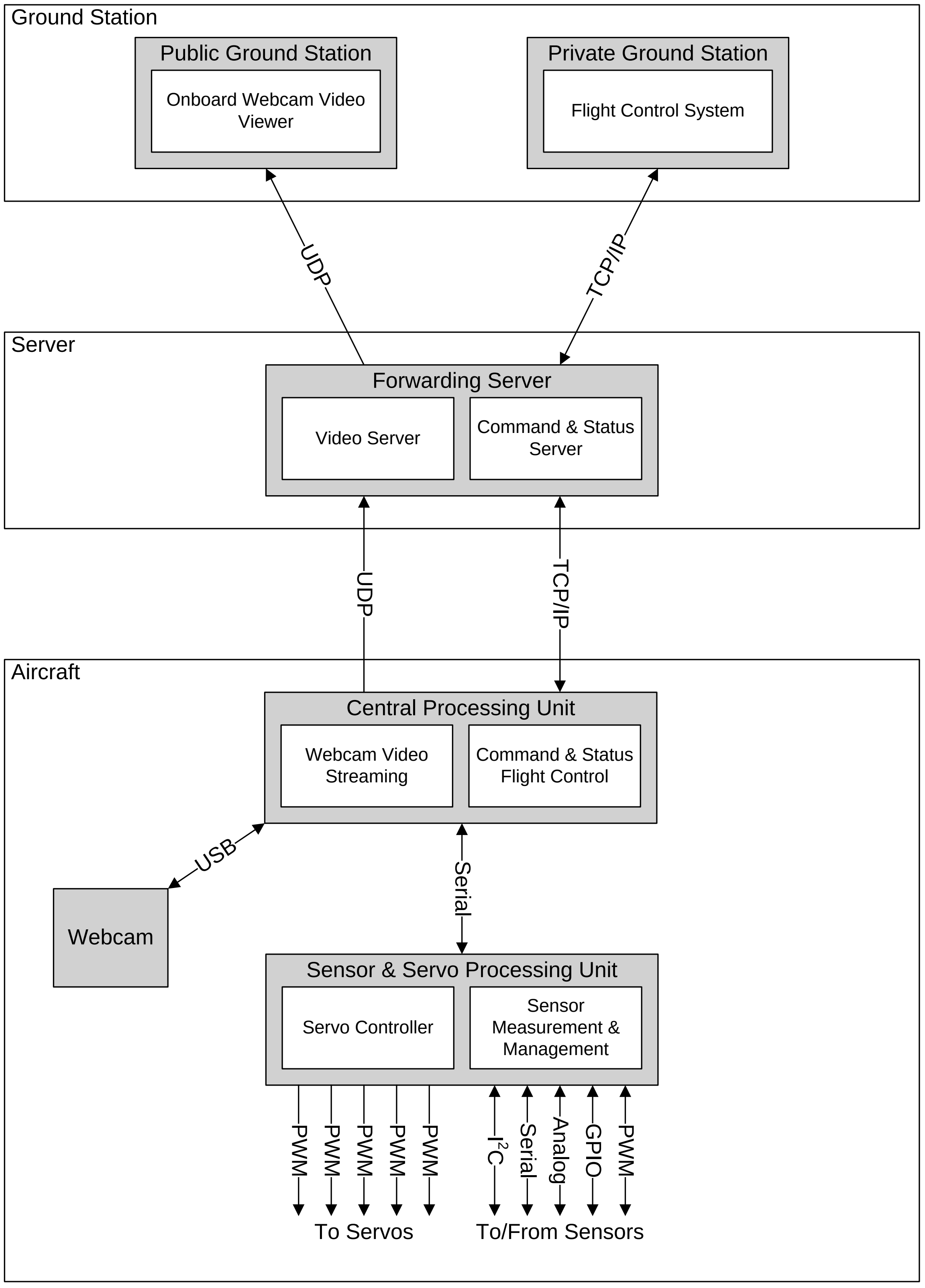
Appendix I – High-Level System Diagram


Web Cam
Sensor & Servo Control Unit
Central Processing Unit
Cell Network Transceiver
Internet Enabled Cellular Device
802.11 or RJ-45 Connection
INTERNET

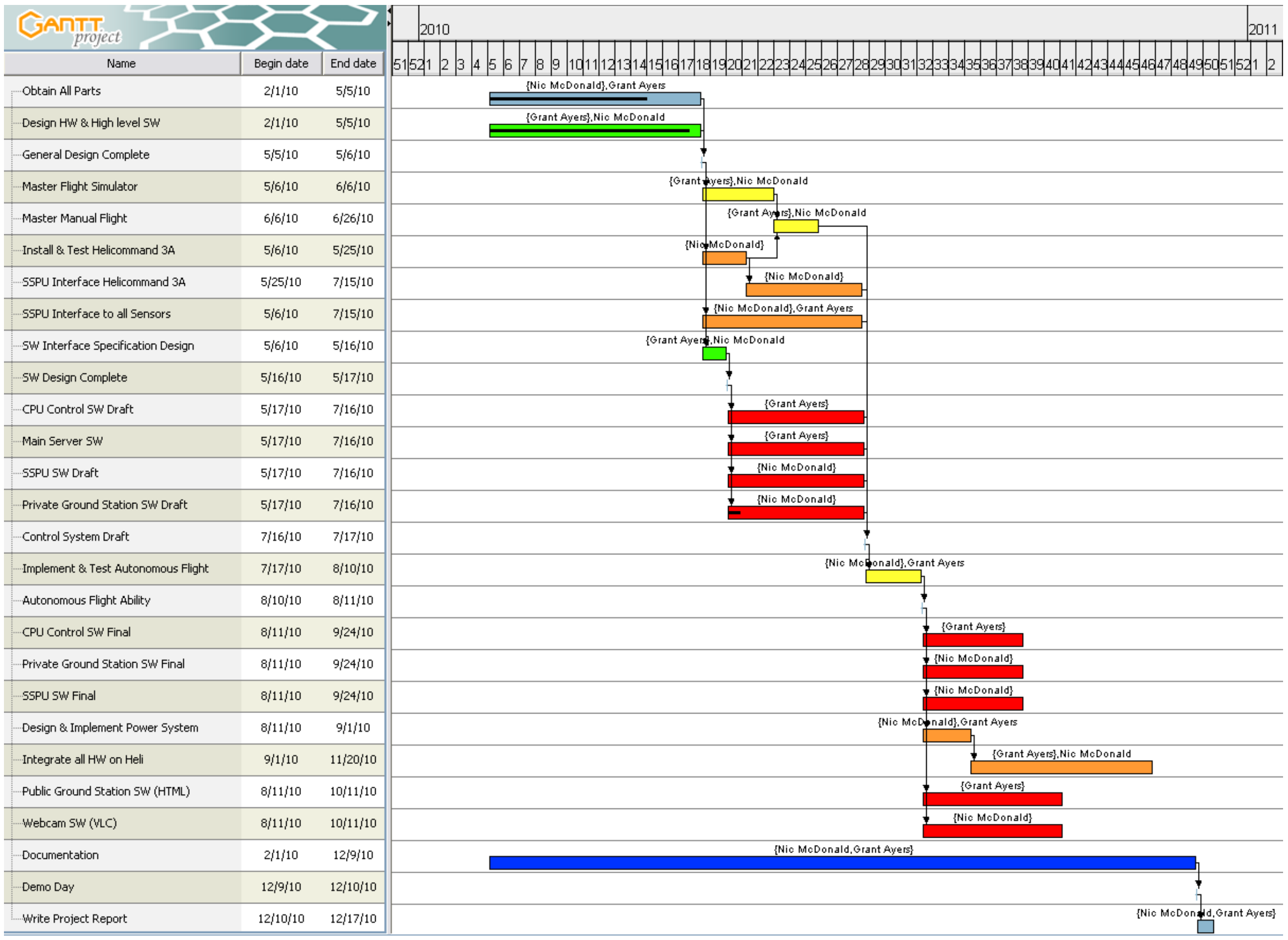
Appendix II – Hardware Interface Diagram



Appendix III – Software Interface Diagram



Appendix IV – Schedule Flow and Tasking



Name	Begin date	End date
Obtain All Parts	2/1/10	5/5/10
Design HW & High level SW	2/1/10	5/5/10
General Design Complete	5/5/10	5/6/10
Master Flight Simulator	5/6/10	6/6/10
Master Manual Flight	6/6/10	6/26/10
Install & Test Helicommand 3A	5/6/10	5/25/10
SSPU Interface Helicommand 3A	5/25/10	7/15/10
SSPU Interface to all Sensors	5/6/10	7/15/10
SW Interface Specification Design	5/6/10	5/16/10
SW Design Complete	5/16/10	5/17/10
CPU Control SW Draft	5/17/10	7/16/10
Main Server SW	5/17/10	7/16/10
SSPU SW Draft	5/17/10	7/16/10
Private Ground Station SW Draft	5/17/10	7/16/10
Control System Draft	7/16/10	7/17/10
Implement & Test Autonomous Flight	7/17/10	8/10/10
Autonomous Flight Ability	8/10/10	8/11/10
CPU Control SW Final	8/11/10	9/24/10
Private Ground Station SW Final	8/11/10	9/24/10
SSPU SW Final	8/11/10	9/24/10
Design & Implement Power System	8/11/10	9/1/10
Integrate all HW on Heli	9/1/10	11/20/10
Public Ground Station SW (HTML)	8/11/10	10/11/10
Webcam SW (VLC)	8/11/10	10/11/10
Documentation	2/1/10	12/9/10
Demo Day	12/9/10	12/10/10
Write Project Report	12/10/10	12/17/10

Appendix V – Bill of Materials

Item	Description/Notes	Status	Vendor	Cost	Totals	
Vehicle:						
Align T-Rex 600	Electric Helicopter	Got it	L-3 Sponsored	$2,000		
Helicommand 3A	Stabilization system	Got it	L-3 Sponsored	$500		
					$2,500	
CPU:						
VIA ARTiGO A1000	Mini x86 System	Got it	VIA Sponsored	$250		
1GB SO-DIMM RAM	x86 Memory	Got it	VIA	$45		
2.5" 80GB SATA	x86 Hard drive	Got it	Newegg	$40		
					$335	
SSPU:						
Atmega1280	Arduino Mega	Got it	Sparkfun	$65		
					$65	
Peripherals:						
Cell Network Modem	USB	Borrowing	Friend	$100		
Locosys LS20031 GPS	Serial		Sparkfun	$60		
Ultrasonic Range Finder	Serial, Analog, PWM	Got it	Sparkfun	$30		
3-Axis Accelerometer	Analog	Got it	Sparkfun	$20		
HMC6352 Compass	I2C		Sparkfun	$35		
Temperature Sensor	I2C		Sparkfun	$6		
Logitech C200 Webcam	USB		Amazon	$30		
Pan & Tilt Servo Kit	PWM		Trossen	$30		
					$311	
On Board Power:						
14.8v Battery	Lithium Polymer (4-Cell)		Hobby-Lobby	$32		
Battery Charger	1-4 Cell LiPo Charger		Hobby-Lobby	$30		
					$62	
						$3,273

References

T-Rex 600 Picture

http://www.alignrcusa.com/index.php?main_page=product_info&cPath=3&products_id=1491

Pulse Width Modulation Diagram

http://www.ermicro.com/blog/?p=771

Hardware Interface Pictures

(1) http://www.via.com.tw/en/products/embedded/ProductDetail.jsp?productLine=1&id=472&tabs=1

(2) http://www.verizonwireless.com/b2c/mobilebroadband/

(3) http://www.logitech.com/en-us/webcam_communications/webcams/devices/5865

(4) http://www.sparkfun.com/commerce/product_info.php?products_id=9152

(5) http://www.mscomposit.com/elektronika_79/ruzne_200/helicommand-3a.html

(6) http://www.jtecrc.com/futabaproducts.htm

(7) http://www.sparkfun.com/commerce/product_info.php?products_id=8975

(8) http://www.sparkfun.com/commerce/product_info.php?products_id=8502

(9) http://www.sparkfun.com/commerce/product_info.php?products_id=7915

(10) http://www.sparkfun.com/commerce/product_info.php?products_id=252

(11) http://www.sparkfun.com/commerce/product_info.php?products_id=9418