Autonomous Targeting Sentry (ATS)

December 5, 2011

Group 12

Ethan King

Daniel O'Hara

Stephen Rodriguez

James Van Gostein

Table of Contents

List of Figures

List of Tables

1.0 Executive Summary

In the United States, defense has always been an important factor in the budget. In 2011, defense attributed for nearly 25% of the U.S. federal budget, more than any other individual factor.**[1]** With so much emphasis being placed on defense, it is evident that a constantly improving defense system is required. Autonomous sentry guns provide great protection to the area they surround, but do not rely on the use of people to do so. This aspect is what makes unmanned or autonomous defense systems such an important field of research. These defense systems can be deployed, not only in military settings, but in home defense, as well. The primary goal of this project is to simulate a realistic autonomous turret as a defense system to protect a high priority area.

The group's personal motivation in accepting this project is that it contains fields from both electrical and computer engineering, providing a complex task which challenges the skills obtained throughout our education. Also, the autonomous target sentry gun has many components in which the group members have limited experience, so throughout the process of the project, the designers will learn these technologies in preparation for our future careers.

The key traits to an autonomous sentry gun that make it such a valuable defense system are reliability, accuracy, intelligence, and efficiency. Reliability is vital because an autonomous system will not have constant monitoring from a human, so it must be able to function on its own for long periods of time, without the risk of component failure. Accuracy and efficiency are major factors because any targets need to be taken down before they can pose any kind of threat to the turret itself. Lastly, the system needs to be intelligent so it cannot be bypassed by someone using a decoy or some sort of distraction.

The objective of the project is to research, design, and prototype an Autonomous Targeting Sentry (ATS). The purpose of the turret design will utilize knowledge of hardware components such as servo motors, microcontrollers, capacitive touch sensors, lasers, web cameras, and printed circuit board design. The system will use the openCV computer vision libraries to detect, track, and eliminate targets. The turret will have the ability to be controlled manually from an application running on the on-board laptop. The user can take control of the turret at any time to operate the turret or shutdown the machine. Through openCV, facial recognition software will be developed as a security protocol to prevent unauthorized users from accessing the manual controls of ATS.

The turret will contain two servo motors for pan and tilt of the paintball gun along with another motor for trigger control. High definition web cameras will be used in conjecture with openCV to detect and track enemy targets. When a target enters the field of view of the web camera, the system begins tracking the target. The target will then be warned that they are entering a restricted zone. The on-board computer continually updates the targets coordinates while the microcontroller translates coordinates to pulse widths for the servo motors to move the turret into place. After a second warning to the target, the system begins to fire upon the target until the target exits the field of view or the system determines the target is no longer a threat.

The team has organized this document to cover the research, design, and testing of ATS. The research aspect of the project covers several component comparisons and methods to reaching our objectives and specifications. The design section will discuss the group's component decisions and acquisition as well as detailed software design and implementation. The testing section will discuss the team's strategy to test individual components, the overall system, and environmental variables. ATS is an intriguing, challenging and an instructional project, and the group is eager to construct the final product.

2.0 Project Description

2.1 Motivation and Goals

The motivation behind our project stems from the ever increasing budgets in both the government and private military sectors. The autonomous turret has many applications in security, military combat and un-manned vehicles. ATS features a perfect blend of software and hardware to complement the team's computer and software engineers. By incorporating hardware components such as microcontrollers, servo motors, and capacitive touch sensors ATS tests the group's electrical engineering skills; while motion tracking and the onboard control center will allow the groups computer engineers to acquire the skills necessary for real world software design and development.

The goal of ATS is to incorporate our skills gained over the past four years to research and design a functional prototype of an automated turret. The prototype will be lightweight and low cost and will demonstrate the need and functionality of a real world automated turret. The system will have intuitive controls and interface so that anyone can operate and monitor the turret. The system will have built in security measures including a kill switch and a facial detection security system in order to access the turrets manual control operations. In order to demonstrate the need for such a turret the system must be accurate, reliable, and have quick response times; any lapse in functionality will result in massive risk to the zone ATS is covering.

2.2 Objectives

To ensure the ATS systems meets a stringent quality factor, a system of specifications must be ensured. For any weapon system, accuracy should be high and consistent. For the webcam to properly function with the object tracking software, a specified maximum resolution must be specified. To keep up with a moving object, the servo motors must be able to operate at a specified speed. In order to limit the amount of fire, the ATS system must have specifications for the firing rate. In order for the tracking software to efficiently track a target, an ideal time to identify and track a target will be specified. These primary specifications along with secondary specifications are defined as the following:

- System will be cost efficient
- System will be able detect, track and target incoming objects
- ATS will be light weight
- Easily portable
- The ATS system will be reliable
- Ability to prioritize targets with high threat levels
- Easy set-up and usability
- Ability to track multiple targets
- Follow targets moving with high speeds
- System will be able to differentiate between allied and enemy targets
- Facial recognition provides an extra security measure required to access the manual control class.
- Easy accessible components with expectation of future upgrades
- Components will be interchangeable
- The weapon system will have high levels of accuracy
- Demonstrate real world applications
- The system will have two types of alarm systems, one continuous sounding alarm and an alarm to verbally warn incoming intruders
- The ATS will have the option to be manually controlled via capacitive touch sensor, keyboard or application buttons
- The base of the system will be capable of housing all the electronic components
- The printed circuit board will be encased in a plexiglass box to protect the design
- System will be able to determine when the target is no longer a threat.
- Microcontroller must control all the servo motors
- The manual control application will have a video stream from the turret's web camera embedded
- System alerts user when out/low on ammunition
- A laser pointer will aim down the barrel to indicate the target

2.3 Requirements and Specifications

In order to make a successful prototype that can demonstrate real world applications, the ATS system must first be affordable. With a wide variety of potential targets, the ATS system should be capable of tracking multiple moving targets either fast or slow. Since many industries push newer and more advanced components, the ATS system must have components that are easily swappable. The ATS system must also be well protected by a sturdy base construction. To meet these requirements the following objectives were created.

2.3.1 Hardware Requirements

Just like real world applications, specifications need to be established prior to any prototyping. This is also the case for the design team; the designers need to establish hardware specifications to ensure the ATS system is working properly. By doing so, the ATS system can prove to be a reliable system that will produce efficient results. Below is a list of requirements the designers established as hardware requirements.

- Weapon shall have a hit ratio greater than 70%
- Webcam shall provide a video stream with 1280 x 720resolution
- The entire turret system shall weigh no more than 30 pounds
- The turret base shall not exceed a height greater than three feet
- ATS shall cover a lateral angle of tracking of 135 degrees
- ATS shall cover a vertical angle of tracking of 90 degrees
- The turret shall be battery operated and operate for a minimum of 30 minutes before a recharge is required
- The servo motors shall be able to move at least 4 radians per second with no load
- The paintball gun hopper shall be able to hold 200 paintballs for ammunition
- The alarm system shall be audible from at least 100 feet away
- The high pressure aluminum alloy tank shall contain 20 oz of CO2 to pressurize the paintball gun
- The barrel of the gun shall not be longer than 12 inches long
- The system shall have a lag of no more than 0.25 seconds upon command via capacitive touch or the onboard laptop
- The battery system shall not output more than 12 V at any given time
- The gun shall be able to fire 3-5 paintballs per second in burst mode
- Turret should be able to detect and keep track of a minimum of 3 incoming targets
- Laser pointer shall be visible on targets within 100 feet
- Within 100 feet, the laser pointer shall be accurate within ±1% of the distance

2.3.2 Software Requirements

The ATS system is a compilation of both hardware and software components. A major component for the hardware to work properly is efficient working software.

The software is the initial step for a working prototype. The requirements below are a list of software specifications the design team established for the ATS system.

- ATS shall play an audio file to warn incoming intruders five seconds after entering the field of vision.
- ATS will play a second audio file, fifteen seconds after the target enters the field of vision, to warn the target to alert them the gun is about to fire.
- ATS will have a response time of 1.5 seconds from finding a target to aligning the gun
- Targets will be prioritized in the order they enter the web camera's field of vision.
- The firing rate of the gun shall be adjustable with three different settings which are automatic, burst, and single shot.
- The Facial Recognition class will lock the manual control class after five failed attempts.
- The object tracking class will transmit the coordinates of the target to the microcontroller with an accuracy of one degree.
- The system will determine when a target is no longer a threat by the amount of green paint on the target.

2.4 Responsibilities of Team Members

2.4.1 Ethan King

The roles and responsibilities that Ethan King, electrical engineer, will undertake are the selection of electrical components including resistors, capacitors, inductors, operational amplifiers, and voltage regulators. He is also responsible in choosing servo motors and working with Stephen Rodriguez to make sure each servo motor properly communicates with the microcontroller. Ethan king is also primarily responsible for the design and implementation of the printed circuit board with help from Stephen Rodriguez. Ethan King will also work with Stephen Rodriquez in programming, designing, implementing, and testing of a capacitive touch manual controller. Minor responsibilities include assisting Daniel O'Hara, Stephen Rodriguez, and James Van Gostein with getting the onboard computer to communicate with the microcontroller controlling each servo motor.

2.4.2 Daniel O'Hara

The roles and responsibilities that Daniel O'Hara, computer engineer, will undertake are mostly on the software side of the project. He will be responsible for the computer vision with the help of James Van Gostein. To be more specific, Daniel will undertake the object detection, object tracking, and color detection components of the ATS. This means that he will be responsible for the computer vision from the point that the system receives the image to where it must make a decision based on if any objects are tracked and what color they represent. James Van Gostein will take the lead on the software following that part. He will be responsible for both the design and documentation of these algorithms, with the help of the OpenCV software that can be used for this project. Other minor responsibilities will include website creation, capacitive touch versus android application research, and assistance with the project hardware as requested.

2.4.3 Stephen Rodriguez

The primary roles and responsibilities that Stephen Rodriguez, computer engineer, will undertake are implementation and a programming of the microcontroller. From there, he will tie the power source, servo motors and alarm system to the microcontroller's pins. Together with Ethan King, he will be responsible for the majority of the hardware components, this includes, the implementation and design of the power sources, alarms, web cameras, servo motors and architectural structures. Ethan King will take the lead on the implementation and design of the printed circuit board layout and servo motors. Other smaller responsibilities will be to assist both Daniel O'Hara and James Van Gostein in their software implementation since a large part of this project is software oriented.

2.4.4 James Van Gostein

James a CpE major will be working on the software side of the project with Daniel O'Hara. James will design and create the manual control application and the facial recognition security system. He will then work with Daniel to finish the object detection and tracking classes. James will be in charge of the testing and tweaking of the manual control operations and facial recognition. James will assist the group with the building of the base and gun mounting. James will also manage the budget and track the group's expenditures as well as log meeting times and progress throughout the two semesters.

3.0 Research Related to Project Definition

3.1 Similar Existing Technologies

The ATS system as a whole is not a new technology. Different aspects of the entire system have previous been done for a variety of reasons, such as the motion tracking cameras and a gun system that intercepts incoming targets. These were the original unique technologies that when all integrated together can help build an autonomous targeting sentry. The influence of these similar existing technologies has broadened the range for the expansion of the different prototypes for future optimizations.

The web camera on board the system has had several similar functions in the past. The web camera for the ATS system is similar to those of any video surveillance cameras. For example, in banks, the surveillance camera is continually streaming a video. If the videos need to be replayed, the users can use facial recognition to identify particular people of interest. The only difference between the cameras of the ATS system compared to these surveillance cameras, is the ATS system will not have a database to store these recordings, instead the designers wish to implement a more advance functions including color recognition, range detection and motion tracking.

Motion tracking is another similar existing technology. Motion tracking has been used for several different reasons such as projections in the schools and homeland security. Typically, these cameras have the ability to move on command. Even in the universities, the instructors have manual control of the cameras for those that record their classes. This is the same for homeland security, the user can move the directions of the camera to focus in different areas, and this increases the range of visibility which makes it practical instead of installing several other cameras.

The military actually has several technologies similar to the ATS system. The web cameras along with facial recognition and color detection make the ability for determining friend and foes targets. High tech security systems can include finger print and retinal scanners, these is to ensure the person of interest is who they're supposed to be. To demonstrate this same idea, the designers intend to have the ability to remotely enable or disable the system when the incoming target is a friendly. The intention is for only friendly targets have the remote with a specific password to power down the target, similar to a garage code where only those who are supposed know will know the correct sequence.

The military actually has a similar system called the Medium Extended Air Defense System (MEADS). **[2]** This system is a tri-nationally owned system, the United States of American, Italy and Germany, and the system is just like the ATS system. MEADS has the ability to detect and track an incoming missile and fire its missile to intercept the incoming target. The intent of MEADS is to provide safety for the city or military base that it is stationed in.

By reflecting on all the similar existing technologies previous demonstrated successful in the past, the designers of the ATS system intend to integrate several of them to make a specific design. The system will have several functions, mainly as a defense turret to protect. This system will be able to be further optimized depending on its use but initially can be used for homeland security and local security for businesses or home owners. It can even be used in military situations such as protect a military base from incoming armies and also have the ability to successfully intercept incoming planes, helicopters, and missiles. The ATS system's concept does have several uses, it can protect your possessions or it can save a life.

3.2 Similar Senior Design Projects

The creation of an autonomous turret system has been previously successfully completed in the past. Several other systems have been created, both on a production level, for sales, and a project level for varies opportunities such as senior design projects. These projects are related to the ATS system and can be found online for extensive research. The exposure of previous project's successes and failures are determined by researching the design of these projects which will further improve the ATS system intended to be constructed.

The idea of creating the ATS system was originally developed by the discovery of The Sentry Project. This prototype is an automatic targeting paintball gun intended for purchase for the curious buyers. This was the original inspiration for creating the ATS system that will also consists of a laptop to autonomously control the servo motors that directs the paintball gun to incoming targets. The ATS system's concept is similar to The Sentry Project because it also uses web-cameras for of range targeting while continuously streaming video to the laptop so the user can have the perspective of what the gun see and targets. **[3]** From an overview, the ATS system has a lot of similarities to The Sentry Project however The Sentry Project is business owned and intended for profit. However, the ATS system's design will vary significantly and the intent of the prototype is to demonstrate mastery of knowledge instead of profit.

After future investigation into the creating an autonomous gun, it was found that the University of Central Florida has had several groups create such a prototype recently for senior design projects. By reviewing previous completed projects, all the projects have similarities. Of course all the prototypes detect incoming targets autonomously however; each group used different approaches to reaching this goal. The group that graduated spring of 2008, created a prototype called the Paintball Targeting System. **[4]** This group used color recognition to trace incoming targets. This knowledge may be useful because color detection is one of the many different targeting techniques the ATS system will be using to detect targets.

Another group that completed their senior design project in the spring of 2009 created the G8 Sentry Gun. **[5]** This project was unique from the others because it was able to distinguish between friendly or foe targets. This was done by implementing GPS positioning. The friendly would carry a mobile device that would alert the turret that a friendly is entering the targeting zone; those without this device would automatically be considered a foe.

The final group researched completed their project in the spring of 2011. The Autonomous Turret, like the others, main purpose systems was to detect, track and target an incoming intruder. **[6]** The main difference that separated this project from previous prototypes is that the designers used an off-board server to record the video history. This is a very useful implementation for video tracking for those whose intentions are to recognize the incoming targets. By using an off-board server, the video files are stored in a different location and this will better equip the owner of the Autonomous Turret to report the intruder to local officials who can further decide the plan of action to prevent such intrusions.

All of this research will be very helpful for the designers, however; the use of previous project's research alone is not sufficient enough to successfully construct the ATS system from the ground up. Although the ATS does plan to incorporate some aspects of previous projects, it will be solely up to the designers. All research completed by previous projects is merely a spring board that can propel the ATS system as one of the more advanced video tracking systems.

The ATS plans to incorporate color detection, like the Paintball Targeting System, and friend/foe detectors, like the G8 Sentry Gun, but the approach will be entirely different. ATS also will not be incorporating an off-board server to record the video feed instead the designers intend to make the video tracking more efficient and not track only by color. The ATS system shall incorporate more advanced ways of tracking incoming intruders such as facial recognition and night time tracking. Another unique aspect is the ATS shall have a capacitive touch remote that should have the ability to control the turret remotely, which includes the ability to manually target, initiate and deactivate the system. The intent is take an already completed prototype and optimize it to make it the best it possibly can be.

3.3 Possible Architectures and Related Diagrams

3.3.1 The Base

Possible options for the base include creating a tri-pod with a mounting for the gun at the top. The electronics would be housed in a plexiglass box and be hooked up to the on-board laptop. This architecture is very similar to a camera mounted on a tripod. The tri-pod must be sturdy enough to remain standing after the rapid firing of ATS, if the tri-pod rocks too much ATS's accuracy will suffer greatly. This design would raise the gun off of the ground level and allow for more vertical coverage. The group's second option is to create a rotating platform for the gun. This option would require a less complex gun mount but would be heavier requiring greater torque to get it to turn about. The base would be composed of the same plexiglass housing which would act as the base while a rotating plate hooked up to the horizontal servo. The top section would consist of the gun mount and ammunition. Another option was to create a fixed base with a pan and tilt system mounted on top to mount the gun. Pan and tilt servos are commonly used to mount cameras and other lightweight objects. Unfortunately the pan and tilt system used for cameras only support a maximum of two pounds, this is not enough to support the gun and ammunition and withstand the recoil of the gun when firing.

3.3.2 The Gun Mount

Creating a lightweight durable gun mount is a major design challenge the group faces. The gun mount must be very strong to hold up to the vibration and stress of the automatic firing rate. If the gun mount is too heavy the servo motors will have a hard time rotating the gun, but if the mount is too light there is a chance it will break or severely decrease the accuracy of ATS. The group has a few different options for mounting the paintball gun. The first option is to create a frame for the gun to rest in. This can be done by creating a box made of lightweight metal rods. The rods will support the barrel and the handle of the gun. Two more metal rods will be used to keep the gun straight and accurate. The second option the group is considering is a side mounting for the gun. The mount would be positioned on the left or right side of the gun and have the ability to swivel left, right, up and down. This is very similar to the system used by the paintballsentry project the group has researched online. **[3]** This mounting would be very lightweight and keep the sentry compact and offer a less cumbersome mounting. The pan and tilt option requires that the group builds a platform to rest the gun on, unlike the camera pan and tilt option the groups platform must support more than two pounds.

3.4 Hardware Research

3.4.1 Microcontroller Research

A microcontroller is an integrated chip that has a wide variety of functions that enable them to provide unique control to a specific design. This is different from a microprocessor, which is a multifunctional chip that completes a variety of tasks. Therefore, a microcontroller is intended to be more self-contained and dedicated to specific tasks. Microcontrollers have the ability to execute a set of stored instructions and will carry out tasks defined by the designer. Micro-controllers also have the ability to access external memory chips and read/write data to and from the memory. Microcontrollers consist of a processor core, memory and programmable input/output peripherals. In its entirety, a microcontroller is essentially a basic computer in a single integrated circuit.

There are a variety of the microcontroller designs available for completing the necessities for the ATS system; however, there are 3 main options that will be sufficient for the design requirements. Each microcontroller has its advantages and disadvantages. The first option for a possible microcontroller is the PIC16F1503, Peripheral Interface Controller, from Microchip Technologies Inc. **[7]** These microcontrollers have the Harvard architecture – in which instructions and data come from separate sources. This simplifies the timing and design which helps optimize clock speed and power consumption. These microprocessors have RISC architecture, a built in oscillator with adjustable speeds and a wide range of interfaces with free Integrated Development

Environment and several commercial compilers available for the designer. The disadvantages for these microcontrollers are that it is an older design but it is a simple and powerful architecture. These microcontrollers only have one accumulator and operations and registers are not orthogonal, meaning some instructions can only use the accumulator, while others can only address the RAM and/or immediate constraints. However, for the necessities of the system, these constraints should not be a problem for overall design of ATS.

Another main consideration for selecting the PIC family of microcontrollers is the programming language. These microcontrollers are provided with a free assembler package with Microchip MPLAB Integrated Development Environment. It can also be programmed at higher levels of programming such as C. Free C compilers are also available as well as low-cost development tools depending on the preference of the designer. Both are offered in design support from Microchip and have a technical support and training available twenty four hours a day, seven days a week.

The second option is the MSP430F5172 from Texas Instruments. **[8]** These microcontrollers are 16-bit and RISC based but the stand out advantage of the MSP430F5172 microcontroller is that is designed specifically for ultra-low-power consumption. This is a major advantage because the MSP430F5172's peripherals enable ultra-low-power optimization which will ultimately extend battery life. This feature validates TI's competition amongst the leading microcontroller distributors. TI's integrated communication peripherals and high-performance analog make this microcontroller a great option for controlling servo motors that will be a key component in the system design. These 16 bit chips range from 1KB to 256KB of flash memory. Another advantage for this microprocessor is the TI Launchpad development board kit.

The MSP430F5172, like the PIC, can be programmed in a high level like C with Code Composer Studio. There is also an alternative C compiler which will serve as an efficient multi-platform compiler called mspgcc. This is a free and unlimited C compiler for TI's MSP430 series of microcontrollers. The MSP430 microcontrollers also have online training, tools and software to help support the designer.

The last option of microcontroller manufacturers to consider is Atmel Corporation. For the ATS system to be optimal with a minimum response time, simple calculations from the microcontroller need to be performed efficiently to reduce the lag time of targeting the incoming intruder. One of the perks of choosing an Atmel's SAM9 product line is this particular line is designed for high speed processing. With the line chip performing at approximately 240 MHz, this makes this microcontroller more than efficient for computing the simple calculations needed for the ATS system. **[9]** The major setback of choosing this microcontroller is it is much larger and consumes more power than smaller microcontrollers that are equally capable of completing the tasks needed for the ATS system. So this makes it a more expensive microcontroller needed for the application. So even though this microcontroller series is designed for high speed processing, it doesn't appear to be a necessity for the ATS system to perform sufficiently.

Table 1 contains the three different microcontrollers the designers intended to use. The key differences to note are the comparison in number of pins as well as the program memory. The designers of the ATS systems do not need an excessive amount of pins or a large memory, so the designers intend to implement Microchip's PIC16F1503.

Comparison of microcontrollers			
Product	**SAM3S1A**	**MSP430F5172**	**PIC16F1503**
Developer	Atmel Corporation	Texas Instruments	Microchip
Number of Pins	48	29	14
Memory Type	Flash	Flash	Flash
Program Memory (KB)	64	32	3.5
Operating Voltage (V)	1.62 to 3.6	1.8 to 3.6	1.8 to 5.5

Table 1 - Comparison of the different Microcontrollers

3.4.2 Printed Circuit Board Research

A printed circuit board (PCB) is typically a flat plate of insulating material that interconnects electronic components designed for a specific purpose. Connected by a conductive material, these electronic components have unique functions that control the device. And so, the PCB will be a key component in controlling the ATS system as a whole for it will contain the microcontrollers that will control the servo motors for the mobility of the ATS system. There are several types of PCB manufacturers available but two options were looked into for implementation for the ATS system, ExpressPCB and 4PCB.

ExpressPCB provides a free, designer assisting, CAD software that includes two parts, ExpressSCH and ExpressPCB. **[10]** The ExpressSCH program is for drawing schematics. This is to begin the process of designing and to familiarize the designer of drawing schematics. The library provides common electrical components so when drawing the schematic; it can be as simple as placing the components on the page and wiring the pins together. ExpressPCB allows the user to develop a printed circuit board that corresponds to the schematic previously created. This is a possible because ExpressSCH can be linked with ExpressPCB to ensure continuity. This makes laying down traces for the printed circuit board easier. ExpressPCB has a package of three identical 3.8” x 2.5” PCB that can be purchased for $51.

A second option to use for the ATS system is 4PCB. **[11]** Like ExpressPCB, 4PBC offers free software called PCB Artist with tutorial videos to help the design the PCB. However, 4PCB offers a 60 square inches, approximately an 8” x 7” piece, of PCB for $33. This is much larger than the PCB manufactured by ExpressPCB but the benefits of a cheaper option makes it a definite consideration. 4PCB also allows student buyers to purchase a single board which ultimately will benefit our budget.

3.4.3 Servo Motors

3.4.3.1 Motor Control Options

To effectively control the autonomous turret, a system of motors should be implemented to allow for full motion and firing. Since the turret is designed to be stationary, it is best to think of the turret as the origin of a spherical coordinate system. This will allow one motor to be dedicated to the theta direction, a second motor dedicated to the phi direction and a third motor dedicated to pulling a trigger.

Since the turret will remain stationary at the origin, rotational motors will provide easiest control. There are two types of motors that primarily stand out. These choices are a standard DC motor and a signal controlled servo motor, both of which have their own advantages and disadvantages.

Advantages to the DC motor include a full 360 degree range of motion, one input, and the availability of high torque. However there are large drawbacks when used in a controls environment. The largest of these drawbacks is the low precision. The motor is either on or off where speed can be adjusted based on the input. In order to accurately control the position a highly accurate microcontroller will most likely be needed. Another large drawback is the significant cost of higher torque motors.

Advantages to the signal controlled servos include a lower cost when compared to DC motors, a signal controlled position, and multiple similarly previous projects to be the starting point of research. Like DC motors, the signal controlled servos has drawbacks. The largest drawback to servo motors is quickly increasing cost for the increase in torque. Another large drawback is that most stock servo motors only have a 90 degree range of motion. In order to gain a 180 degree range of motion additional charges may apply.

In order to keep the cost low and simplicity high, servo motors were chosen to control the theta and phi directions of the turret. For pulling the trigger, a low cost, low torque servo can be utilized.

3.4.3.2 Servo Control Method

Most standard servos have three leads, positive power, negative, and signal. The power lead not only acts as the power source for the servo, but can also be utilized to turn the servo either on or off. The typical input voltage for power is between 4.8 volts and 6.0 volts. The negative power lead should be common ground. The signal lead will control the direction of the servo.

The primary method of controlling the servo is to send a pulse-width modulation along the signal lead. This pulse-width modulation signals is a fifty hertz square width. The length of each pulse of the square wave controls how far the servo will rotate. For example a pulse of 600 microseconds will rotate the servo arm -90 degrees and a 2400 microsecond pulse will rotate the arm positive 90 degrees. **[12]**

3.4.3.3 Open Loop versus Closed Loop

For a servo motor, there is a significant difference in an open loop and closed loop control system. In an open loop control servo control system, the pulse widths control how far the servo rotates in a specified amount of time. In other words, the length of the pulse width modulation controls how fast the servo rotates, not the position. For example, a 600 microsecond pulse may rotate the servo 90 degrees counter-clockwise in 0.15 seconds while a 1000 microsecond pulse may rotate the servo 45 degrees counter-clockwise in the same 0.15 seconds.

In a close loop servo control system, the length of each pulse controls the position, instead of how fast the servo rotates. For example a 600 microsecond pulse may rotate the servo to the 90 degrees counter-clockwise position in 0.15 seconds while a 1000 microsecond pulse may rotate the servo to the 45 degree counter-clockwise position in 0.075 microseconds.

Most standard servo motors can only rotate 90 degrees and can be stretched to 180 degrees for an additional cost. These rotational limitations are placed by a potentiometer built into the servo motor. As the potentiometer rotates with the servo, the voltage across the potentiometer changes allowing this voltage change to be used for feedback to control the position. The potentiometer can be disconnected to achieve a full 360 degree continuous rotation, however the feedback to control the position is lost and an external circuit will be required. Since it was specified that the turret will rotate below 180 degrees, a continuous rotation is unneeded; this allows for the utilization of the built in closed loop system. **[13]**

3.4.3.4 Servo Gear Material

There are four major gear materials used in servo motors. Nylon is in the lowest tier of gear material is generally the most common in stock servos. Karbonite is slightly above nylon in strength and durability and also runs quieter than Nylon. Metal gears are stronger than karbonite but have lower durability. Titanium gear is the highest tier of gear material. Titanium gears provide extremely high strength and durability, but at a significantly higher cost. These different gears can be purchased and swapped in as needed, however gear sets for each material do not exist for every servo combination.

3.4.3.5 Digital versus Analog Servos

Like many components in the electronics world, servo motors come in standard analog and digital varieties. Functionally speaking, a digital servo is a standard analog motor with a built in microprocessor that analyzes incoming signals to control the motor. Digital servos have two distinct advantages over their analog counter parts. With the built in microprocessor, the servo performance can be better optimized depending on servos function. Also because of the built in microprocessor, the pulse width modulation sent from the microprocessor operates at a higher frequency than the standard 50 Hz used for analog servos. This leads to higher accuracy, smoother acceleration, and the availability to hold higher torque. However, because of the addition of the microprocessor the servo comes with disadvantages. Since the digital servo operates at a higher frequency for higher accuracy, the power consumption also increases. The price of digital servos is also significantly higher than their analog counter parts. **[13]**

3.4.3.6 Torque Calculation

In order to calculate the required torque needed for the servo motors, the angular torque formula must be utilized.

$$T = (AngularAcceleration)(MomentofInertia)$$

The approximate weight of an unloaded paintball gun used as references was 2.25 pounds, or approximately 1.134 kilograms. The maximum allowed weight for one paintball gun is 3.5 grams, or 0.0035 kilograms. It assumed that 100 paintball guns will be loaded into the hopper for a weight of 0.350 kilograms. A lightweight paintball gun hopper, the component that holds, the paintball gun ammo, is advertised as approximately one pound. Since paintball gun hoppers vary largely, it is assumed the hopper will weigh two pounds, or approximately 0.9072 kilograms. The total mass of the paintball gun, paintballs, and hopper, is approximately 2.392 kilograms. For simplicity, this mass of a loaded paintball gun is estimated to 2.500 kilograms. The length dimensions of the paintball gun are estimated to be 24 inches long by 3 inches wide by 8 inches high. For simplicity, the paintball gun will be assumed to be a cuboid. The moment of inertia must be calculated across two axis, the pitch and the yaw. The calculations can be found in table 2.

It is assumed that the base platform will be constructed of quarter-inch plywood. According to the APA plywood specifications, 0.842 inch thickness of unsanded plywood weighs approximately 3 pounds per square foot. **[14]** Quarter-inch plywood can be estimated to be .75 pounds per square foot, which is also a rule of thumb. Assuming the base platform is 24 inches by 24 inches, or 0.6 meters by 0.6 meters, of half inch-plywood, the approximate is estimated to be 3 pounds, or approximately 1.5 kilograms. Similar to the paintball gun, the base platform will moments of inertia are calculated in table 2.

The base tower holding the paintball gun will be constructed of quarter-inch unsanded plywood. As previously used, quarter-inch plywood can be estimated to 0.75 pounds per square foot. It is assumed that the base tower compose of two quarter-inch plywood slabs that will be approximately 8 inches long by 0.25 inches wide by 16 inches high for a total weight of approximately 1.5 pounds, or approximately. The two plywood slabs will be placed parallel, 4 inches from the center to the outer of the plywood, or 3.75 inches from the center to the inner side of the plywood.

Table 2 consists of a summary of the dimensions, mass, and moments of inertia of the three major moving components of the autonomous turret. All units are provided in meters, kilograms, and newton-meters. “L” represents the length in meters, “W” represents the width in meters, “H” represents the height in meters, and M represents the mass in kilograms. Since the base platform will not rotate along the pitch direction, the moment of inertia can be ignored in the pitch direction. The moment of inertia is given in Newton-meters. According to the above equation, the angular torque is calculated by multiplying the moment of inertia and angular acceleration, assumed to be 60 degrees per 0.15 seconds-squared or approximately 6.9 radians per seconds-squared. The total torque in either direction can be summed by the principle of superposition The torque listed in the table below is converted to oz-in by multiplication of 141.6.

Moment of inertia and torque calculations						
Part	**L**	**W**	**H**	**M**	**Moment of Inertia**	**Torque**
Paintball Gun	.61	.08	.20	2.5	$I_{yaw} = \frac{m}{12}(W^2 + L^2)$ $I_{pitch} = \frac{m}{12}(W^2 + L^2)$	$T_{yaw} = 77.2$ $T_{pitch} = 84.5$
Base Platform	.61	.61	.006	1.5	$I_{yaw} = \frac{m}{12}(W^2 + L^2)$	$T_{yaw} = 91.8$
Base Tower	.20	.006	.41	0.7	$I_{yaw} = \frac{m}{12}(W^2 + L^2)$ $I_{pitch} = \frac{m}{12}(W^2 + L^2)$	$T_{yaw} = 6.98$ $T_{pitch} = 6.98$
Total	NA	NA	NA	NA	NA	$T_{yaw} = 182$ $T_{pitch} = 91.5$

Table 2 - Dimensions, mass and moments of inertia of moving components

According to the calculations above, the amount of torque required to turn the autonomous turret in 182 oz-in in the yaw direction and 91.5 oz-in in the pitch direction. These numbers are calculated from heavy estimations due to large variations in paintball gun weight and unconfirmed base design. However, all these estimations were on the higher end in order to receive a higher required torque that should theoretically be needed.

3.4.4 Web Cameras

The web camera will be a vital component in the system design. The sole purpose of the web camera is to provide optics for the ATS system, from there the software will be implemented so the system has object detection, object tracking, motion tracking, color detection and facial recognition. For these implementations to occur, a high definition camera will need to be used. The system will be utilizing a Logitech C310 HD Webcam. The webcam has a widescreen video at 720 pixels HD Resolution, a built-in mic and auto adjustment for poorly lit settings. The webcam will be attached to paintball gun looking down the barrel for accuracy of the target. The webcam will also be connected to the laptop via USB and have a real-time video capturing that will display on the laptop. **[15]**

3.4.5 Alarm Systems

The alarm system will be strategic component to warn incoming intruders that they are entering a restricted area. The alarm system needs to be loud enough to ensure the incoming targets can hear the potential warning. Two option have been considered as possible candidates for the alarm, a continuous audible alarm and a speaker that can play an audio file stored on the onboard control system.

The first option to consider is the Turbo series TMC-86-530-W, a panel mounted alarm by Floyd Bell Inc. **[16]** This option is a continuous sounding alarm that has a wide arrange of operating voltages ranging from 5Vdc to 30Vdc. The alarm is a non-descriptive alarm meaning it just outputs a continuous noise. The typical operating current ranges from 2mA at 5Vdc to 10mA at 30Vdc. In table 3 below, the important working specifications are provided. The output sound and current consumption is linearly related, so the higher the input current, the louder the alarm is. This option is a cheap viable option that only cost $9.64 per alarm component.

Turbo series TMC-86-530-W	
Mounting	Panel Mounted
Operating mode	Loud – Continuous
Operating current	2mA at 5 Vdc 10 mA at 30 Vdc
Termination	Wires
Surge Voltage	20% over the maximum rated voltage for five minutes

Table 3 - Specifications of Turbo series alarm

The second option for consideration is to just use a simple speaker that plays a unique audio file from the system. This gives the option to warn incoming intruders of specifically what the system's intentions are. For example, the system can verbally warn the intruders that they are entering a danger zone. This option gives a variety of audio files to instruct the intruder to exit the area and warn of an imminent attack. A basic set of computer speakers such as the Insignia 2.0 Stereo Computer Speaker System will be a satisfactory option for the necessity of the system and these speakers are available for $19.99 from Best Buy. **[17]**

There are advantages and disadvantages to using either option. The advantages of using the Turbo series TMC-86-530-W will perform better outdoors because its casing is resistant to salt spray, humidity, dust and vibrations. However, the disadvantage is the alarm cannot inform incoming intruders of the system's intent to fire upon them. The major advantage of using a speaker that reads an audio file from the computer is the alarm can verbally alert intruder that they are in a potentially dangerous zone if they continue to proceed forward. The disadvantages are the alarm will not be as loud and will not be as outdoor resistant. Therefore, the ATS may use both alarm components to serve the purpose of alerting incoming intruders.

3.4.6 Power Sources

There is a wide range of batteries available and each has certain advantages and disadvantages. Different types of batteries range from Alkaline batteries to lithium- ion batteries but for the purpose of the ATS system, the designers have narrowed the choice of battery to two options, lead - acid and nickel-metal-hydride batteries. Both of these types of batteries are matured and have proven quite reliable in the past. Therefore, there is a wide range of previous knowledge that the designers can turn too if problems arise during implementation of either power source.

First, the lead-acid batteries, typically known for being used as car batteries, are used for a lot of applications. CSB Battery technologies Inc. builds a 12V/1.3Ah Sealed Lead Acid battery available at RadioShack for 15.99.**[18]** This is a smaller version of a typical car battery so the size should not be a problem for the ATS application. The lead acid battery has a an extremely high boiling point, temperatures far higher than the system will every experience and the battery is easy to work with. Under normal operating conditions, the internal material will not be hazardous, only if exposure or leakage of the internal material, will the battery be hazardous but exposure is very unlikely.

One advantage of this is how small this battery is. It has a length of 3.8 inches, a height of 2.05 inches and width of 1.89 inches and the battery only weighs 1.3 pounds so this won't affect the weight or size dimensions of the ATS system. However, the major advantage is this battery is also rechargeable which makes an optimal battery. Rechargeable batteries typically have a higher initial cost but can be recharged cheaply and used over and over. Therefore, the battery will have a lower total and also benefit from environmental damage compared to disposable batteries.

The second option, is the nickel metal hydride battery. These batteries work by using a hydrogen absorbing alloy for negative electrode This makes the capacity larger than other batteries. The energy density is similar to a lithium-ion battery, but the rate of self-discharge is much higher compared to a lithium-ion battery. This is one reason the lithium-ion batteries were not considered for the ATS system, they tend to be every expensive.

Nickel-metal-hydride batteries come in all different shapes and sizes, so the next decision would be to determine how much power is needed. The designers determined that 12V would be more than sufficient to power the system. A suitable nickel-metal-hydride battery is the Tenergy 12V 1400mAh battery pack available through total power solutions on All-Bettery.com.**[19]** Like the lead-acid batteries, this battery pack is small, actually smaller with the dimension of 2.3 inches long, 1.3 inches wide, 1.1 inches tall and only weighs 8 ounces. A special feature of this battery pack is that it has a long life cycle and rapidly charges. However, the original intent for this battery pack is for RC aircrafts or mini Robots so the battery pack has 16 AWG Wires to connect to these usual platforms. This makes it a challenge to connect the nickel-metal-hydride battery pack directly to the printed circuit board where the microcontroller will be. The designers will be able to splice the wires to connect directly to the printed circuit board but then has to determine how the battery pack will be capable of being recharged.

3.4.7 Range Detector

In order to accurately prediction the location an object to be tracked, a range sensor would need to be implemented. Naturally, there are various sensor types to determine the range of an object. One range sensor type is infrared range sensors. Typically infrared sensors cannot measure distance past thirty feet. The advantage to IR range sensors is the low price. However, IR sensors are not very accurate and tend to be limited to less than thirty feet. This is much less than the desired one-hundred feet making the IR sensor impractical.

Another type of range sensor is a laser range detector. The laser range detector is used in a variety of applications from recreational sports to military defense. Laser range detectors have a wide band of ranges well within the desired one-hundred feet and are highly accurate. Even though a laser rangefinder would work, it is difficult to find one with a reasonable price and the ability to work with the microcontroller. Typical laser range detector prices start around $100 and exceed the $1000 mark.

Another type of range sensor is an ultrasonic range detector. Like the laser range detector, the ultrasonic has a wide band of ranges within the desired one-hundred feet. Ultrasonic range detectors are typically cheaper than the typical laser range detector. However, unlike the laser range detector, the ultra-sonic range detector is typically inaccurate and will have issue detecting the distance when multiple objects are moving.

Of these options, the best and only reliable range detection sensor typical would be the laser range detection. However, for a starting price of approximately $100 and difficulty of incorporating said sensor with a microcontroller, the options of not having range detection must be considered. Ideally an autonomous turret will be in a relatively small confined space smaller than 100 feet by 100 feet. The average paintball gun should theoretically fire straight pass 100 feet. Although the turret will not be able to predict the location of the tracked object, it should still be able to predict the direction of the tracked object. Although a range detector would be a great advantage for the autonomous turret to have, the cost is not justified since the direction of an object and already be predicted. Thus, no range detector will be implemented.

3.4.8 Manual Control Options

There are multiple methods of manual control for the autonomous turret. One method is to develop a website that offers a full control of the turret. This custom website would have options to control the pitch, yaw, and firing rate via a custom built GUI. The advantage to having a custom website is the easy expansion of various options. A video feed from the web cameras, preset firing rates, and accessibility to mobile devices are a few examples that could be implemented. The disadvantage to the having a custom built website is the need for a secondary computer to host a server and the need for additional networking hardware.

There are also multiple methods to control the servo motors with a physical “handheld’ controller. One method is to purchase a pre-built servo motor joystick controller from servocity.com. A two servo joystick controller from servocity.com currently cost $99.99. The advantage to this controller is no additional need for another circuit design. Power gets plugged directly into the controller and the controller outputs operating voltage power, ground, and signal. The disadvantage is the ridiculous cost for plastic and wiring, and the complete bypass of the microcontroller controlling the servo motor position and firing.

Another “handheld” controller method is to develop a controller similar to the servocity.com two servo joystick controllers. However, with the increase popularity of capacitive touch devices, an updated controller replacing the joysticks with a capacitive touch controller will make an interesting controller. One advantage to building a custom capacitive touch controller is the ability to design the controller to work cooperatively with the position and trigger microcontroller at a much lower cost. Another advantage is the possibility to incorporate custom modifications including predefined settings, custom firing rates, and a current position display. The main disadvantages are the huge step in difficulty and the need for an additional microcontroller.

A large deciding factor in choosing manual control method will be the cost. According to the Texas Instruments website, a capacitive touch booster pack with a MSP430 microcontroller Launchpad cost roughly $15.00. The MSP430 microcontroller and booster pack combination was chosen since the group is already somewhat familiar with the product. When compared to the prebuilt two servo joystick controller from servocity.com, the clear winner is a custom built capacitive touch controller. In addition to the capacitive touch controller, a custom built website will be developed for further control options.

3.4.9 Manual Controller Communications

With the selection of a capacitive touch controller, a method of communication between the position and trigger microcontroller and the capacitive touch controller must be incorporated. There three primary methods of are onboard, wired, and wireless.

The easiest method of communicating between the two microcontrollers is simply building the capacitive touch controller directly onto the turret. The capacitive touch controller will simply be attached to the turret, translate the capacitive touch to pitch and yaw degrees, and then send this information to the microcontroller controller the servo motors. The servo microcontroller will simply translate the pitch and yaw degrees to the correct pulse width modulation.

Although it is possible to transmit data over a wired or wireless connection, the difficulty of implementing a wired or wireless connection within a semester timeline may be difficult since the difficulty of implementing a real world capacitive touch controller remains largely unknown. The microcontroller implemented with the capacitive touch controller should be programmed heavily with functions in order have the communication and calculations separate.

3.4.10 Laser Pointer

A laser pointer is another component the designers wish to implement to the ATS system so the users can visually see where the gun is pointing at. However, there is a wide range of laser pointers to choose from but the typical distinction between each other is the laser color. The color of the laser is based off of the wavelength intervals from the visible light spectrum. The designers have narrowed the options down to three different options, a red, a green and a blue-violet laser pointer.

The first laser pointer is the red laser pointer, with a wavelength interval between 635 – 700 nm. **[20]** The red laser pointer pen from LasersMan.com actually outputs 5mW of power and has a wavelength of 650nm. But since the red laser point was one of the original laser pointer technologies and the growth of technology, the red laser pointers have no longer become a first choice option. The laser pointers lack the power and distance of new models but are still a cheap viable option, especially for simple uses likes pointing at a projector for presentations. Since the red laser pointer lacks the distance factor, the designers have opted not to use this design. **[21]**

The next option was the blue-violet laser point. With a wavelength interval between 400 – 490 nm, the blue-violet color is on the opposite end of the visible light spectrum. **[20]** This laser is still a new cutting edge technology, and has only recently been available in the market partly due to the creation of the blu-ray, high definition video technology. LaserPointPro.com has a adjust focus blue-violet laser pointer pen that has a larger range than the red laser pointer, which would benefit the system for targeting the incoming targets further away. **[22]** Although the laser pointer is a new technology, it is also an expensive one, costing $73.99. So the designers of the system will not need to impact their budget to meet the requirements that other laser pointers can achieve.

The final option is the green laser pointer. With a wavelength interval between 490 – 560 nm, the green color is right in the middle of the visible light spectrum. **[20]** Today, this is laser pointer is far brighter than red lasers. Known for such long ranges, it has actually become a felony for individuals that would aim green lasers at planes flying for distracting the pilots. LaserPointPro.com has a mid-open green laser pointer available that has a wavelength of 532nm and operates at 3-4.2 volts. **[23]** So the designers have decided the mid-open green laser pointer will be sufficient enough to demonstrate compliance for the system.

In table 4 below, is a comparison chart of the three different lasers the designers were intending to use. The key things to note are the laser wavelength, laser range, and cost. The laser wavelength is what determines the color and range. As you can see by the range comparison, the green and blue-violet laser pointers are significantly larger than that of the red laser pointer. After that, the cost was the determining factor, and since the mid-open green laser pointer is significantly cheaper, the designers have chosen to use this model.

Laser Pointer Comparison			
Product	**Mid-open Green Laser Pointer**	**Red Laser Pointer Pen White**	**Adjust focus Laser Pointer**
Model number	HK-E03358	A0877000AV0107	HK-E03508
Laser Color	Green	Red	Blue-violet
Laser Wavelength	532nm	650nm	405nm
Laser output power	20mW	5mW	150mW
Power Supply	2 x LR03 AAA 1.5V batteries	2 x AAA batteries	1 x 18650 2200mAh 3.7V battery
Working Voltage	3.0 – 4.2 volts	3.0 volts	3.0 – 4.2 volts
Working Mode	Continuous Wave	Constant wave	Constant wave
Dimensions (Dia. x L) in centimeters	1.295 x 14.198	1.27 x 13.716	2.2 x 14.3
Laser Range (meters)	500 – 5000	Approx. 1600	500 – 5000
Cost	$14.99	$3.39	$73.99

Table 4 - Comparison of different laser pointers

3.5 Software Research

3.5.1 Computer Vision

Computer vision is an increasingly popular application and has a wide variety of uses. Computer vision has found its way into robotics, surveillance, defense systems, and other applications. The process of computer vision is generally the same for all applications with variances for each specific use. The usual steps of computer vision software follow the pattern of image processing, object detection, object tracking, and, lastly, decision making. The image has to be processed first so when the object detector receives the image, it is in the best condition for the detector to find what it is searching for (edges, corners, faces, etc...). Then once objects have been detected, the algorithm must keep track of those objects for various reasons. In surveillance, tracking objects can be used to keep the camera focused on any movement so if there are intruders, they will be caught on the camera and more easily identifiable. Then the last step is to make a decision. This part is very application specific, and for the ATS the decision to be made is whether to fire upon the object or not.

Image processing is the initial step in all computer vision applications, and is essential to be able to utilize computer vision algorithms. The processing that actually happens depends on the application, and what is needed. Image processing covers a broad range of features such as image filtering, image transforms, structural analysis, shape descriptors and feature detection. Once the image has been processed, the object detector must view the image and determine whether or not there are objects and, if so, where they are located. Object detection is most commonly executed by detecting specific features within the image, called haar-like features. These features represent the image that is sought after (for example a human face). Using this, the algorithm can compare what it has learned through machine learning, and find where the objects are located.

Once objects have been found in the field of vision, each object must be tracked. The basic way of motion tracking would be to just find out where pixels changed, because if a certain pixel changed colors over the course of just one image frame, then most likely movement occurred at that location. A more robust method is to track the objects calculating optical flow. Optical flow is the pattern of motion of the things you are trying to track. It can be calculated through various methods such as the Lucas-Kanade method (two versions), the Horn-Schunck method, Farneback's method, or the block matching method. All of these methods find the optical flow, but utilize different approaches to complete the task at hand. The iterative Lucas-Kanade method with pyramids is probably the most popular and likely to be the choice for the ATS.

The last step in the computer vision application is to make a decision based on the information gathered from the image or stream. This is the vital step in computer vision and is the reason behind the need for computer vision. The ATS will need to determine whether or not there are targets to shoot at. Then determine if any of them pose a threat (based on various factors which are discussed later on). Examples of other decisions in computer vision applications include a pass/fail on an inspection, or alerting the user of the system that something is out of the ordinary. The decision making process can be based on a variety of factors that were determined within the processing, detection, and tracking done previously.**[24]**

3.5.1.1 Object Detection

Once the image has been acquired and pre-processed the first step to determine if there are targets or not would be to check if there are any objects in the image. An object can be used to represent a person, a face, or to be more general, a blob. Object detection can be done in various ways and has varying levels of depth. Facial recognition is a specific type of object detection that will be discussed in 3.5.1.5.

The first method to detect objects would be to separate the background and foreground objects. A camera or defense system would constantly be monitoring the same area so most likely there would be some normal image with no objects of importance within the image. By knowing what the background is normally, the system can then determine if the pixels of an area of the image has changed sufficiently enough to represent an object. Wherever this occurs the pixel can be labeled as foreground. Then, the entire image can be converted to a binary image, where the background and foreground are black and white, respectively. From this point, the process becomes relatively simple. Based on a threshold set for the minimum size of an object, the system will determine if there are any objects or not. If so then it will create a target and record the position of the object in the image.

Another way to detect objects is to use machine learning. If the system already has an established knowledge of what type of objects it is looking for, and then finding them in images become easier. Machine learning uses a series of positive images of the object to be detected and negative images that are not the object sought after. To be more specific, the software will learn based on haar-like features which consider adjacent rectangular areas in a specific area of the image. For example, the haar-like features to detect a face could be the darker rectangles of the eyes, compared to the cheeks just below. These consistent features of the object being detected allow the system to understand what object it is scanning the image for, so detection of the object becomes less complicated from that point. The reason haar-like features are so popular for object detection is the fast computation speed they allow. Using a two-dimensional look-up table, or integral image, the calculation of a rectangular area within the image only requires four look-ups. This speed is vital to the system, so it can operate in real-time to make a decision based on the object it detects within the image. Since the ATS will be placed in a singular position with the camera directed at a static background, images can be taken of the scene and used for the machine learning process.

The previously discussed methods only take into account local features within the image. If global features are also taken into account, it can provide a more accurate object detection algorithm. To find the global features of the image, the system would take a look at the image in its entirety and find major features within the image. However, this method requires a large amount of complex mathematics and this method is in early stages of research for computer vision. **[25]**

OpenCV's object detection method uses machine learning and haar-like features to detect a person's face and then creates an object based on that. It runs the image through a cascade of classifiers which eliminate non-objects through each stage. The first stage will eliminate most non-objects, but it is essential that it does not miss real objects. Then the following stages continue to eliminate non-objects that may be closer representations of the object sought after. Once the classifiers are completed then the only remaining items should be objects of interest.

In figure 1, the process that the frame from the web camera is shown step by step. In the first picture the standard RGB image that will be received from the web camera is shown. Next, in the second and third pictures, it will be searched for objects (blobs, faces, etc...) and any objects found will have a rectangle drawn around them to denote where in the frame they are located. Using this process to detect objects will allow the team to find all of the objects that are of importance without using an extremely robust algorithm that would require much more computational power. **[26]**

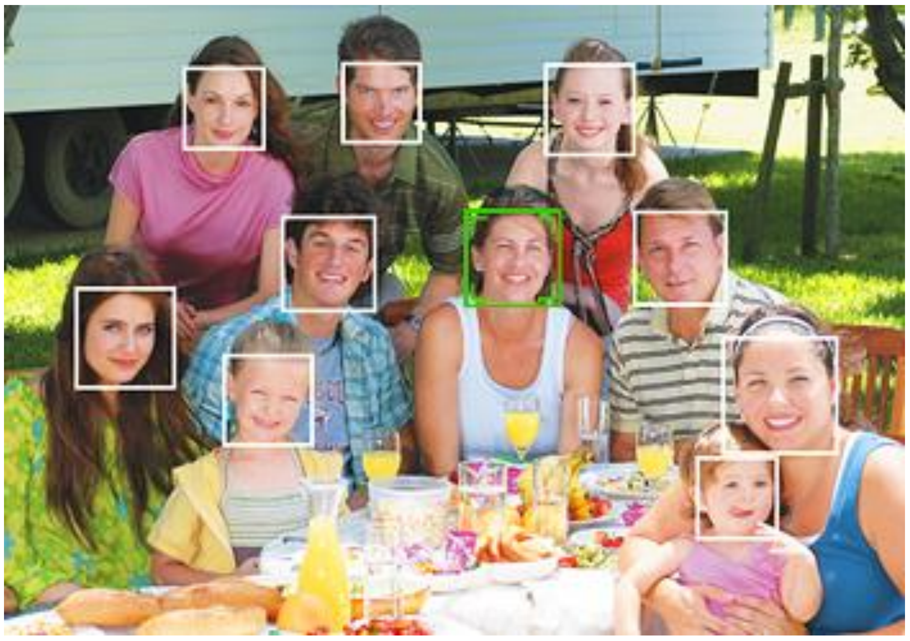


Figure 1 - Result of Object Detection algorithm

3.5.1.2 Object Tracking

Once the objects within the image have been detected, the next logical step is to track those objects. The basic concept of tracking the objects will be to assign each object an x-position, a y-position, and a velocity. For every frame of the image, the system will update each object's position and velocity. Once a few frames have passed, data association can be used to determine the future velocity and position of the object being targeted, allowing for an increase in accuracy since the ATS will be leading its target. Object tracking and motion tracking complete the same step in the computer vision process; however there is a tradeoff for which one the system utilizes. Motion tracking can simply be finding out which pixels changed from frame to frame, and wherever that occurs, motion has occurred. This method would make the process simpler because object detection would not be needed, but the tradeoff is that the algorithm would not be as robust as object detection and tracking algorithms are. For the specifications of the ATS, a more robust algorithm would be necessary to handle real world simulation scenarios.

By detecting and then tracking objects, rather than just tracking motion, the system will be able to assign values to the objects, such as color, or target versus non-target. This will allow the system to make more knowledgeable decisions based on each individual object, rather than just firing wherever enough motion to pass the threshold occurs. In section 3.5.1.4, color detection is discussed, so if it is implemented in the system, different values will be assigned to objects so targets and non-targets can be differentiated, making object detection a requirement.

There are actually a few different algorithms in which object tracking can be done. The most basic of these methods is blob detection and tracking. In this method, the image is scanned and searched for regions in the image that differ from the pixels surrounding it. Once this has been found, the algorithm creates a "blob" based on the size of the region found on the image. OpenCV's blob tracking uses a method of finding foreground and background pixels, then searching for adjacent regions of foreground pixels to create the blobs. Then assigns ID's to each blob found in the image and then records the motion that occurs as each frame goes by. This algorithm loads most of the work in the first stage, blob detection, and then uses more simple algorithms for the tracking of each blob.

Features can also be used to track and object. Features, such as color, edges, texture, and optical flow, offer methods of keeping track of an object. For example, tracking humans by the color of their shirt can be done because the shirt is generally a set distance below the face, so once the object detection has detected a face object, it can then get a set of pixels below it and find what color the shirt is. Then wherever that same segment of pixels is found, is where the object is. The OpenCV algorithm uses the most popular method of tracking by optical flow. The algorithm is known as the Lucas-kanade algorithm, which is a differential method. Another optical flow algorithm is the Horn-Schunck method which uses smoothness, but this method is more sensitive to noise than other methods because it is global rather than local. **[27]**

Optical flow can be calculated using a wide variety of methods. The first, and most popular method, is the Lucas-Kanade method. In this method, it is assumed that the flow of nearby pixels is constant and then uses the least squares criterion to solve the basic optical flow equations. There is also a modified version of this method that uses iteration with pyramids to find the optical flow of the object. Another method that can be used for finding the optical flow of an object is the Horn-Schunck method. In this method, a more global approach is taken to solve the problem, rather than the locality used in the Lucas-Kanade method. Also, a global constraint, smoothness, is introduced to solve the aperture problem, which is where it is impossible to tell what direction the object is moving because the ends of the object are unseen by local methods. To continue, the Farneback method uses polynomial expansion to do the approximation of nearby pixels. It provides a displacement between the two frames being considered. Pyramids can also be used here to detect larger displacements within the image. The last method of optical flow to be considered is the block matching method. In this method, a series of overlapping blocks are made of the image and then run through a series of calculations to compare the overlapping blocks to one another. The function then calculates the optical flow of these overlapping blocks and then uses that to find the optical flow for the image.

Overall, object tracking proves to be quite a challenging task due to all of the variables that can occur within the image. The major factors that make object tracking difficult are object occlusion, noise, complex object motion and shape, changes in scene lighting, and the requirements of real-time processing. All of these factors make object tracking a very memory and CPU intensive process, requiring a large chunk of RAM.

3.5.1.3 Motion Tracking

Motion tracking is the process of measuring an object's position and orientation in physical space. [**28]** The first step to tracking objects is to detect and distinguish objects from the background image. There are many methods to detect motion including mechanical, magnetic, optical and video. Video tracking is most relevant or our project as it is used in many security and surveillance systems. Video tracking is the process of comparing the position of an object in the current frame to the position in a reference frame. This is done by comparing individual pixels and identifying the number of different pixels. Since there are many variables such as lighting or outdoor conditions such as wind pre-processing may be necessary. Once an object has been marked as moving the object can then be tracked using a video tracking algorithm. Video tracking algorithms fall into two categories; target representation and localization or filtering and data association.

Target Representation and localization algorithms are generally less complex then filtering methods. To represent a target first a feature set must be chosen. This can be chosen from the color or size of the target to be identified. Two of the most popular methods are blob tracking and kernel tracking. For ATS blob detection will prove to be more useful due to its ability to track the dynamic changes found in human motion. Blob detection attempts to identify regions in an image that differ in color or brightness.

Filtering and Data Association is a top down process that uses prior information of the object or surroundings. The most widely used methods are the Kalman Filter and particle filters. The Kalman filter is a set of recursive equations to estimate the state of a process. The filter supports estimation of past and future events meaning this filter can be used to estimate where a moving object came from or where the object is heading. This is very useful when tracking objects that move behind obstacles or determining the path the object came from or is heading. The filter works by using time update equations and measurement equations; measurements are updated and used by the time equations to predict the future path of the object. **[29]** If the process is non-linear an extended Kalman filter is used instead to track and predict target paths. [**30]**

The best tracking algorithms often incorporate both Target Representation and localization with a filtering method for path prediction. For ATS's needs blob detection will be the method of choice because of the ability to track humans effectively. OpenCV contains the cvBlobsLib which is a library to detect and filter blobs.

The optical flow of an object through a video sequence is shown in figure 2. In the figure, the optical flow vector can be seen going up and to the right, following the path of movement of the object. As each frame of the video sequence goes by, the object has a new position. So the changes in the location of the object are recorded for use in calculating the optical flow. This can be used to assist in finding out where the object will be in the upcoming frames, thus allowing the program to predict the movement and become more accurate when firing upon any object.

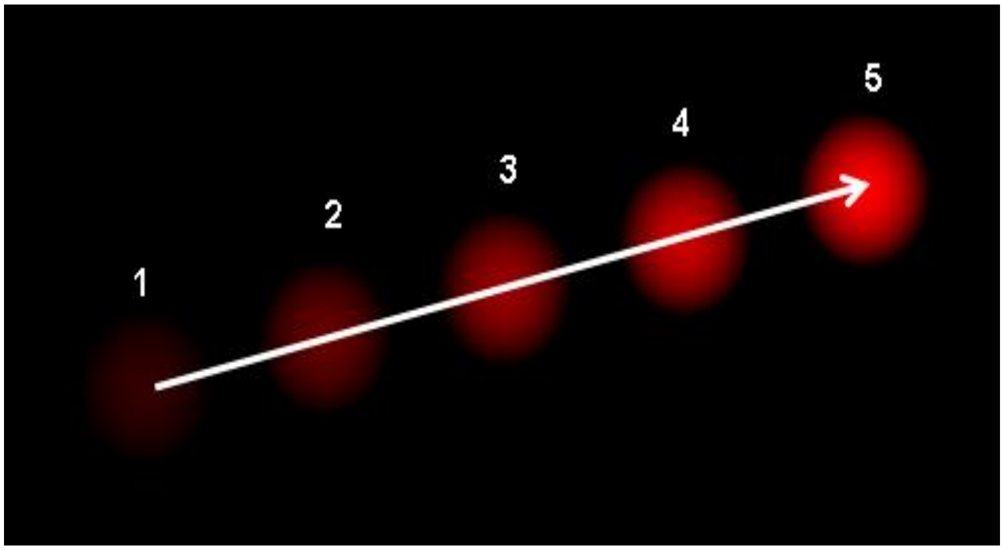


Figure 2 - Optical Flow of an object

3.5.1.4 Color Detection

Color detection is another important aspect in ATS' computer vision. By detecting various colors, the system will be able to differentiate between allied targets and enemy targets. By specifying a certain range of color to represent an allied target, the system will be able to calculate the average color of the target and determine if it is a threat or not.

There are multiple ways in which an objects color can be determined through computer vision. The first, and most straightforward, method of color detection is to take the pixels that represent the target, average their RGB values, and then check if it meets the requirements of the threshold. For example if allied targets were represented by green colors, comparisons would be made with pure green (0, 255, 0) and depending on what the threshold is set to, it would then determine if the target is an ally or an enemy. To make a more complex threshold, the Euclidean distance can be calculated between the received pixels and the target color. A challenge in this method of color detection is that it makes it difficult to detect various shades of the same color without making a large threshold. Another possible solution for detecting colors is using HSV instead of RGB. The primary benefit of converting the RGB pixel into a pixel represented by HSV is that it would be relatively easy to detect the various shades of a single color. A relatively small threshold can be set for the hue, or the tint of the color, and then larger thresholds can be set for saturation and value to detect all of the various shades of the color specified.

Once the color of the target has been found there are various ways to proceed with determining how to handle the target. One way, would be to convert the image into a binary image based on where the color specified has been found. Then it would be much easier to handle the image since, the various other colors of the image have been removed from the frame. This is effective and simple method, but is difficult to utilize for tracking more than one object of the specified color.

To fix the problem that multiple targets would cause, the system could find targets first and then assign them a color value based on what is found within each target. OpenCV's blob detection can already locate multiple targets in a single frame so by taking that data and then assigning objects color, rather than finding out the color first would make handling multiple targets more simple. Then once the target has been assigned a color value, the sentry gun can continue to track all targets, but it will check the color value of the target before determining to fire upon the target or leave them alone.

Overall, there are a couple of different approaches for determining the color of an object. The RGB method offers a more simple method, but comes with a couple weaknesses due to its simple nature. On the other hand, the HSV method handles those flaws, but would be a more complex algorithm. Then, once the color of the object has been detected, there are various ways to approach the decision making. Once again, the tradeoff for having a simpler algorithm of converting the image to a binary image would be that the system could not be able to detect more than one target. The other approach of finding all targets and then assigning them a color seems to be the more practical approach for the application of a defense system because multiple targets is a probable occurrence in the real world.

In figure 3, the before and after of the color detection process is demonstrated. This particular color detection method uses the idea that there will be a shirt a certain distance below the face that was detected. **[31]** So by checking those pixels, the color of the shirt is used as the color of the object. In the picture on the left, the standard RGB image taken from the web camera is shown. Once the object detector uses the frame to find all objects, the color detector will look a specific amount below those objects (which in this case are faces) to find the color of the objects (the shirt). Using hue, saturation, and value, the color will be easily determined due to the fact that hue represents the “tint” of the color and will give the group information that is needed for color detection. So, the algorithm will check for a wide range of hues but only a small range for saturation and value so it can return any of the shades of the specified color, rather than just one very specific shade of a certain color. **[32]**

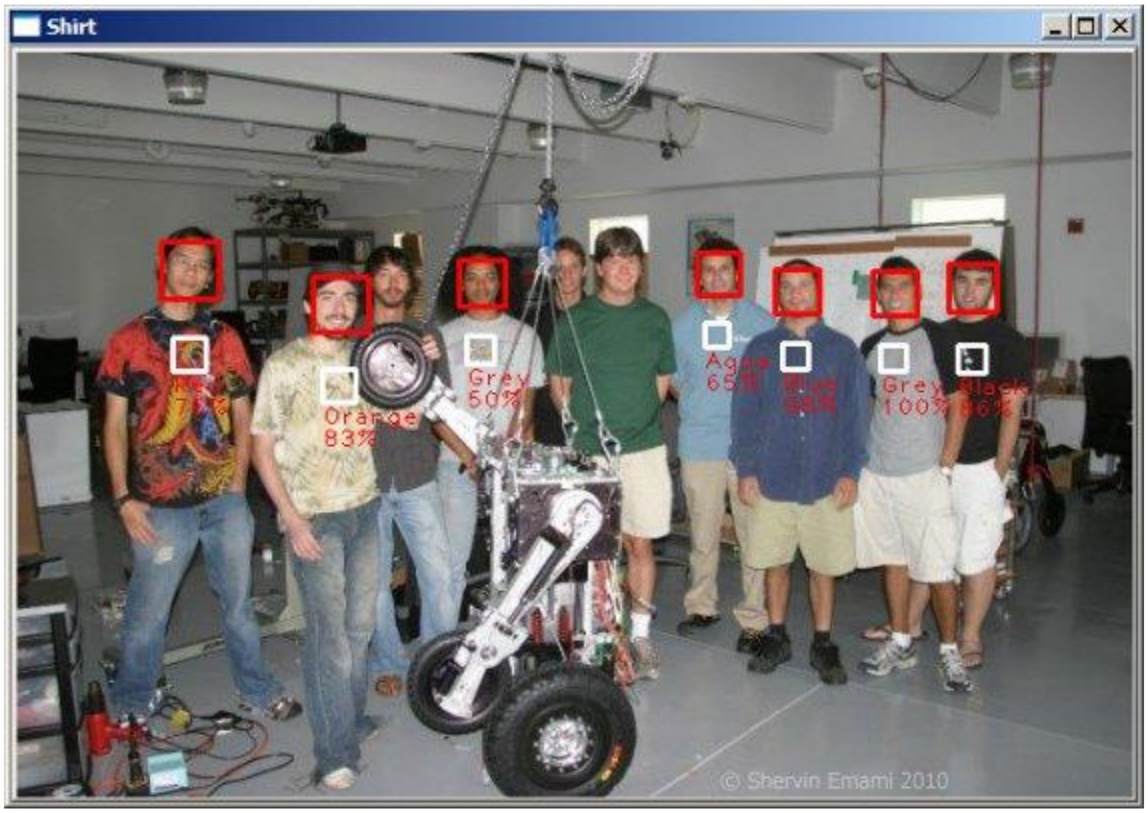


Figure 3 - Result of Color Detection

3.5.1.5 Facial Recognition

Facial detection and recognition has many real world applications from biometric identification to security systems, detecting criminals, and more recently in online photo albums to detect and tag recognized faces. The area that pertains to our project is biometric identification and security systems. Identifying particular users as friendly or dangerous will allow the designers to restrict the turret from firing upon friendly targets. Facial detection is a specific class of object detection Facial detection software determines the presence, location, and sizes of human faces. **[33]** Once a face is detected in an image, the software can then try to find a matching face in a database, this is known as facial recognition. There are various methods to finding and detecting faces they involve the detection of one or more of the following variables: appearance of the face, shape of the face, and motions of the face. In most methods of facial detection a binary pattern-classification task transforms content in the image into features. Then one or more classifiers decide whether the region is a face. Face models are often used to train the classifiers and are contained in databases. Face models include a wide range of faces with different skin color, expressions, shape, and motions of the face. The most widely known method used today is the Viola Jones method.

Classifiers often look for features such as the eyes of the face since the region around the eyes is generally darker than other regions of the face. Classifiers may also try and detect movement of a face including blinking, raising eyebrows, flared nostrils, wrinkled forehead, and an open mouth. Skin color is not used as a classifier since there are so many variations on skin color to be a valid option. In the best facial recognition software multiple classifiers are used to search for and identify a single feature this is known as a weak classifier. **[34]** The classifiers are then layered together in an attempt to identify a face. Facial detection is not an exact science there are many variables and scenarios that can result in false-positives and failures in detecting a face. If the viewing angle of the face is more than twenty-five degrees off from the front portrait of the face the classifiers may not be able to detect features. Other weaknesses include poor lighting, sunglasses, long hair, and facial expressions.

In 2001 Paul Viola and Michael Jones proposed a real time solution to the facial detection problem. In their report "Robust Real-time Object Detection" they document the steps and processes necessary to creating fast reliable facial detection software. **[35]** Unlike other facial recognition software at the time the Viola-Jones method does not use image intensities. The process involves finding the integral image representation for images, creating weak classifier using AdaBoost to locate and mark Haar-like features which are features that appear on the human face, and finally combining multiple classifiers in a cascading structure to increase speed of the detection by focusing on important regions of the image.

Finding the integral image representation requires a few operations per pixel but allows the group to detect facial features easier. The integral image at location (x, y) contains the sum of pixels above and to the left of (x, y) inclusive. The integral image can be calculated in one pass over the original image. This allows the team to use the rectangles calculated in the integral image to compare rectangles by calculating the difference between two rectangular regions. These rectangles are compared to search for Haar-like features which derive its name from Haar wavelets. Since the number of rectangle features of each sub window is greater than forty five thousand computing each one is very inefficient and not feasible in a real time system. Therefore Viola and Jones use a variation of AdaBoost short for Adaptive Boosting is an algorithm to train a machine to identify features of an object using weak classifiers. The idea is that multiple weak single characteristic classifiers can be combined to form a strong classifier. The algorithm searches over the image and returns the perception with the lowest classification error. The biggest challenge comes from assigning weight to each of the classifier and determining the importance of each facial feature.

The last step of the Viola-Jones facial detection algorithm is to enhance performance and reduce processing time. This is done by setting up a cascade of weak classifiers. Each sub-window is analyzed by a classifier if the feature is detected then another classifier is used, if at any point the classifier fails to detect a feature the sub-window is rejected. The idea is that after several stages of classification the number of sub-windows is reduced greatly. From here more advanced recognition methods can be performed on the remaining windows; this saves both time and processing power. Ideally the classifiers used are in increasing complexity that way if a window fails the first classifier test it is discarded reducing the need for further examination. As with most object detection methods there are tradeoffs; more classifiers means you will achieve more accurate results with fewer false positives and fewer errors, but more classifiers means more processing power for the CPU increasing time and required to compute an image. Viola and Jones suggest a false positive rate is established and one classifier is used to start. Sample data is then used if the false positive rate is not met another classifier is added and so on until the thresholds are met.

An effective detection method involves using Haar-like features that can be used to detect contrasts in the human face most notably the detection of the darkness around the eyes contrasted with the color of the person's cheeks. OpenCV uses an improved upon method of the Viola-Jones method in which a cascade of AdaBoosted classifiers are used to detect the Haar-like features of the human face. Classifiers output a "1" if the sub-window of the image is likely to contain a face and "0" otherwise. If a "1" is returned then the sub-window moves on to the next stage of processing continuing on until a face is detected or a classifier rejects the sub-window. **[36]** In order to check for images of different scales the image must be scanned several times with the classifiers must be resized to the scale. Facial recognition begins with the process of facial detection. Once a face is detected a database of faces can be scanned to try and find a match.

In figure 4, a few examples of the haar-like features of a face are shown. The haar-like features are broken down into three sections including edge features, line features, and center-surround features. Haar-like features are used to minimize the amount of computation needed for object detection. When dealing with only image intensities, searching an image for haar-like features by summing up the pixels of adjacent rectangles within a specific window of the image can be useful in finding the desired objects. By using a summed area table, the calculation of haar-like features becomes very fast and efficient, requiring only four lookups to the table. Viola and Jones detail haar-like features in their publication on "Rapid object detection using boosted cascade of simple features." **[37]**

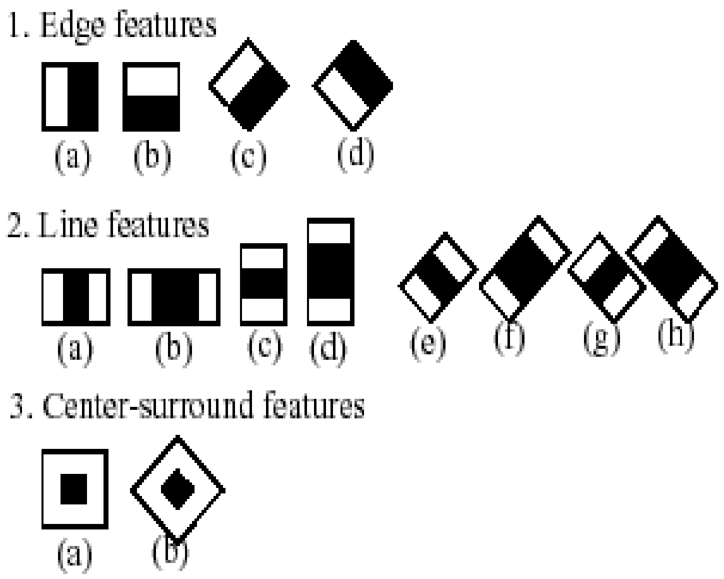


Figure 4 - Haar-like Features

3.5.1.6 FPGA or Computer Vision

The biggest decision the group was faced with was whether to use computer vision or an FPGA system to track targets. Each option has its advantages and disadvantages, a hardware FPGA system would have quicker processing times and would be more reliable for real time tracking, while computer vision has multiple other features the group can implement into the design such as facial recognition or path prediction tracking. Furthermore FPGA systems require specialized algorithms including Fourier transforms, Hough transforms, and stereo vision algorithms. These algorithms would take a great deal of time and research to implement whereas computer vision libraries have methods and functions to handle the calculations and algorithms.

FPGA systems are commonly implemented with a CMOS image detector to send the image to the FPGA board where the processing can take place. In Kazuhiro Shimizu and Shinichi Hira paper published by IEEE the implementation of an FPGA system with CMOS sensors is discussed. **[38]** The concept is that successive images or frames are captured by the CMOS imager and sent to the FPGA board where a vision algorithm is implemented. The FPGA then selects features of the image such as the hue or luminosity and from here can process the data since the feature data is much less than the full image data. According to the report the sampling must occur at 1000Hz in order to capture enough data for real-time tracking to be feasible. In order to reduce processing time and power filters are put into place along with a pipelining processing system.

Computer vision libraries are a popular way to implement motion tracking and image processing since computer vision libraries come with complete libraries and built in functions to handle the tedious processing and computation. The biggest flaw in computer vision systems is that they rely on a computer and operating system. Computer vision does not have the processing speed or reliability of a FPGA system. While computer vision is not without its flaws it is still a great way to implement motion tracking. Computer vision libraries and the built in methods and functions handle all of the processing and computation allowing the group to focus on other features to incorporate into the turret. Computer vision allows the group to add features such as facial recognition for security, multiple target tracking, and target path prediction meaning targets can be tracked behind partial obstructions or even if they are hidden from the camera.

The group came to the decision to use OpenCV which is an open-source computer vision library. It was decided that even if software requires more processing and achieves lower response times than the hardware solution that the extra features and built in algorithms OpenCV is the best choice. The OpenCV libraries are available online along with instructions to install the libraries and tutorials to get the group started with learning the functions and features. The second option was to use the aforget.net libraries, but in the end it was decided that OpenCV was more established featuring more tutorials and information. It was also decided that the group would prefer to use the C++ programming language used by OpenCv rather than C# used in the aforge libraries.

3.5.2 Android Application

Having a manual override for ATS is necessary for safety precautions. If the system were to malfunction for whatever reason, it would be essential to be able to disable it from firing at will on anyone who approaches it. For ATS, there are two options being researched for the manual override. The first is an android application and the other is a capacitive touch sensor. Both will feature ways of manually controlling the turret, but the android application will be more robust just due to its nature. The capacitive touch sensor will have limited features, but its hardware aspect makes up for it.

The android application would consist of features such as a kill switch, directional controls to manually aim the turret, and a button to set the fire rate. To get the main program and the android application to communicate, a small database would be needed to retrieve the changing data from each program. The android application would set variables in the database when the respective buttons are pressed in the application, and then the main program would retrieve those updated variables and act accordingly. Even though the application is quite basic, the real challenge of including this feature would be the database and interfacing the C++ program and the android application to communicate with each other.

The capacitive touch sensor would not consist of all of the features in the android application. It would be able to move the turret with the scroll wheel and determine when to fire with the capacitive touch button. One such capacitive touch sensor operates with the MSP430. The capacitive touch sensor would be a good option to add to the ATS for the hardware aspect of it.

Overall, both options are going to be challenging additions to the project, but will add another component that will test the capabilities of the team. The choice really comes down to hardware versus software. The project has already become software heavy so it may be a good idea to go with the capacitive touch sensor to help balance out the project load between software and hardware.

3.5.3 OpenCV

Open Source Computer Vision Library, or OpenCV, is a major component in the software part of the ATS. The system will utilize a large variety of the computer vision programs to go from the web camera image to the decision of when and where to shoot. Object detection, Object tracking, motion tracking, color detection, and facial recognition are all featured within OpenCV. With these open source algorithms, the system will be able to make well informed decisions based on all targets found in the range of the ATS. Using these algorithms will also allow the system to be more advanced because of the time saved from having to develop computer vision algorithms for the specific application. The OpenCV object detection algorithm, discussed in more detail in section 3.5.1.1, utilizes machine learning to be able to detect haar-like features of the object within the images it receives from the input, which, in our case, is the web camera. The OpenCV object tracking algorithm is more popularly known as the Lucas-kanade algorithm. It is the most popular object tracking algorithm, and does so by using optical flow. The color detection algorithm uses the HSV method as described in section 3.5.1.4. **[24]**

Aforge was another option for the open source computer vision software.**[39]** However, OpenCV was determined to better fit the needs of this project and team due to various reasons. OpenCV is coded in C++, while Aforge is in C#. The team only has experience in C, but it was decided that C++ would be the better route. Also OpenCV has a wider variety of extra computer vision features such as color detection and facial recognition. Lastly, OpenCV is more established and therefore has more efficient algorithms.

4.0 Design

The ATS system contains both hardware and software components. The design team was split equally between hardware and software. Hardware design was primarily lead by Ethan King and Stephen Rodriguez. Software design was lead by Daniel O’Hara and James Van Gostein. The following sections will discuss in further detail the hardware component selection and software design layout.

4.1 Hardware Design

Overall, the ATS system is a structure that has been demonstrated in the past, only the computer vision software is relatively new software. The hardware design on the other hand has similar components that have been completed for decades as engineers design new up and coming technologies. The hardware design is split into 4 different sections, physical layout, power systems, PCB integration and communications. Below in figure 5 is a hardware block diagram.

The power systems are simple design. The onboard laptop has an internal battery already installed so the designers will just use this for convenience. However, this battery system will not be sufficient to power the entire system, specifically the printed circuit board. Instead a 12 volt battery will be used to power the components on the PCB, mainly the microcontroller. From there, the entire system as a whole should have enough power to successfully perform.

The physical design is similar to building blocks. The designers need to draw out their preferred dimensions of the entire system. This includes the base structure which will ultimately hold up the entire system. From there, the base will contain servo motors which will turn the system to aim in the preferred position. Tied to these motors will be the gun itself and consequently how the gun system is mounted. The designers need to ensure the gun mounting is stabile because it will be moving because it is attached to the servo motors

The PCB integration is the brain the system. This needs to work to ensure the system works properly. The primary concern is the implementation of the microcontroller since both the servo controls and the status indicators will be soldered directly to the pins of the microcontroller. There are three different controlling servos, the pitch, the yaw and the trigger pull. Each will have to be attached to its own microcontroller pin. As far as the status indicators, the microcontroller will be also reserve I/O pins for the alarm system, led status and laser.

The final hardware design is the communications. The microcontroller will have to communicate directly to the computer system. On the computer system, there are two key functions, the sensors and the network. The sensors are the webcam and ammunition count. For the networks, the manual control, via the onboard laptop and capacitive touch, and kill signal will all be implemented to communicate directly to the computer system and then the microcontroller.

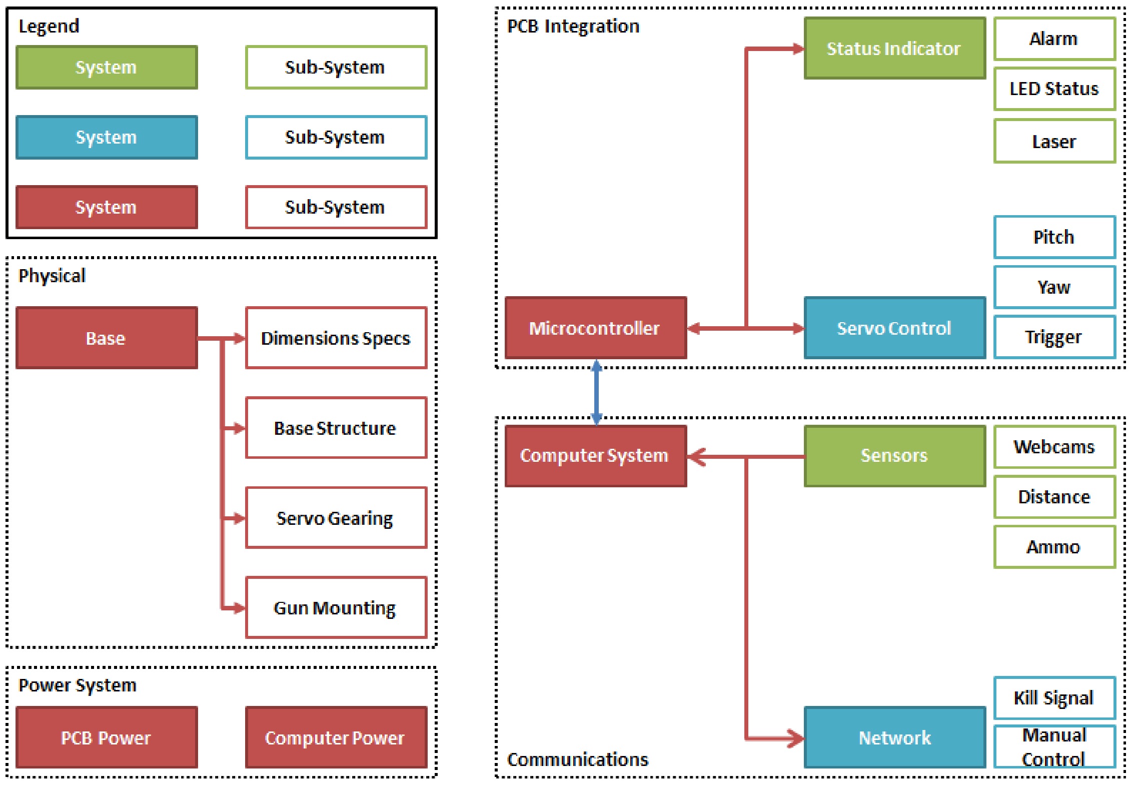


Figure 5 - Hardware block diagram

Looking further into the printed circuit board design, the following in figure 6 is generated. The manual controller will be controlled by a Texas Instruments capacitive touch booster pack with a Texas Instruments MSP430G245 microcontroller. A Microchip PIC16F1507 will translate incoming coordinates form either the manual controller or the laptop and convert the coordinates into pulse widths in the pitch and yaw directions. The PIC16F1507 will also toggle the component alarm and laser targeting on or off. The three servo motors, HS805BB, HS815BB, and HS81, will be controlled by the PIC16F1507 with a unity gain buffer between each servo motor and the PIC16F1507. The TL084 will be used to implement the unity gain buffer.

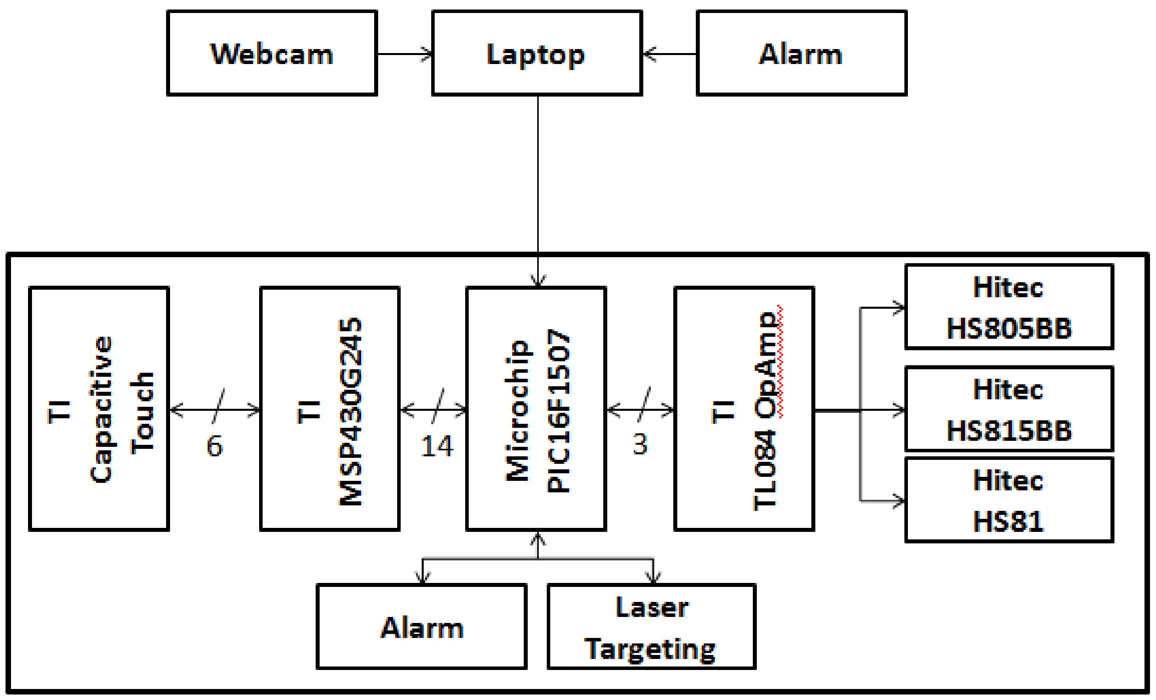


Figure 6 - Circuit control block diagram

4.1.1 Microcontroller

The system design will need a microcontroller to successfully obtain a signal from the on board lap top, process all the data successfully and to output a signal to control the servo motors of the system. After researching several different types of microcontrollers, the microcontroller chosen for the ATS system is model PIC16F1503 from Microchip Technologies Inc. It is a sufficient design suitable for the system that is capable of meeting the necessary specifications of the system design.

The microcontroller is a 20-pin flash, 8 bit microcontroller and unlike some microcontrollers, the PIC16F1503 is not already installed on a standard printed circuit board. Instead, the designers of the ATS system intend to solder the microcontroller directly on to a printed circuit board of their choosing. This will create a unique processor designed only for the ATS system so this will not be reused for other projects, unless the PCB is redone which means re-soldering and starting from scratch.

The concept of implementation should be a fairly simple task; however, depending on the expertise of the designers, the implementation may end up to be a complex task with significant importance that essentially determines the success or failure of the ATS system. The primary concerns are soldering the microcontroller properly to the printed circuit board and having the necessary pins of the microcontroller properly soldered to ensure the complete functionality for the system. The most crucial components are going to be soldering the power source, a connection to the vertical controlling servo motor, a connection to the horizontal controlling servo motor and a connection to the trigger control servo motor.

The microcontroller operates in a voltage range from 2.3 V to 5.5 V so the microcontroller will be directly connected to the power source. The device will need to have a soldered connection from the power source to the VCC pin; this will enable power to the microcontroller. Like all circuits, the microcontroller will need to be grounded so the ground pin will have a soldered connection to the ground of power source to ensure a completed circuit. After these two connections are established, a connection will be soldered connecting the microcontroller to the horizontal controlling servo motor. Each servo motors will need at maximum of approximately 5 volts to operate so this shouldn't be an issue for the microcontroller. The same process will be completed for the vertical controlling servo motor as well as the trigger pulling servo motor. The layout of the servo motors will depend on the designer of the system so the assignment of specific pins shouldn't matter as long as it is an input/output (I/O) pin.

Another advantage for using this microcontroller is having so many more pins then actually needed. Of course there are much larger microcontrollers with several more pins but having free pins on this microcontroller leaves room for expansion for future prototypes. The designers have the option to use several other I/O pins at their disposal. A variety of sensors can be tied to the microcontroller or potentially an alarm system, so the possibility of optimizing is available. Below is the pin layout for the PIC16F1503 microcontroller.

4.1.2 Servo Motor Selection and Acquisition

In order to effectively control the autonomous turrets, a system of three servo motors must be incorporated. One servo motor will be dedicated to the pulling the trigger, another servo motor will be dedicated to controlling the pitch of the turret, and the final servo motor will be dedicated to controlling the yaw of the turret. For the below, all prices are estimated using servocity.com. Servo motor acquisition will be discussed at the end of this section.

When choosing the appropriate servo motor, there are multiple parameters to keep in mind. The primary servo motor parameters include torque, speed, dimensions, stock angle of rotation, power input voltage, input signal voltage, and gear type. Concerning torque, there are multiple servos ranging from 11 oz-in up to a monstrous 611 oz-in. The torque must not only be high enough to hold the paintball gun, but the base of the turret and the ammo for the paintball gun as well. In general, servo speed is measured in seconds per 60 degrees of rotation. For the prototype, a high speed servo motor must be selected for the trigger to allow for responsive firing while the pitch and yaw have the option to be slower for reduced cost. The dimensions each servo is an important factor to consider since the base will be designed around the dimensions of each servo. Most stock servo motors only rotate 90 degrees in one rotation with the option to rotate 180 degrees at an increased cost. For the trigger, a minimal degree of rotation will be needed. For the pitch, 90 degrees of rotation is adequate considering the need to fire straight at the ground or straight into the air will be rare. For the yaw, the servo motor rotation must be greater than 90 degrees but less than 180 degrees in order to maximize field of view while still able to utilize the built in closed-loop system of the servo. The input power voltage must be noted in order to know what voltage regulator will be used to power the servo motor. The input signal voltage must be noted in order to know what voltage the microprocessor must output and if the need for an amplifier is needed. For the prototype, standard nylon gears should be adequate. This will also offer a wider range of servo selection at a reduced cost.

For the trigger, a high speed servo with moderate torque with a reasonable price. The dimensions and angle of rotation will be mostly irrelevant. There are a few servos that fit these requirements. The first is the Hitec HS-225BB servo motor. The HS-225BB is an analog servo with a torque of 54 oz-in, a moderate speed of 0.11 seconds per 60 degrees, and a cost of $17.99. Another choice is the Hitec HS-322 servo motor. The HS-322 is an analog servo with a torque of 51 oz-in, a slower speed of 0.15 seconds per 60 degrees, and a low cost of $7.99. The third option is the Hitec HS-81 Micro servo motor. **[40]** The HS-81 Micro is an analog servo with a torque of 42 oz-in, a fast speed of 0.09 seconds per 60 degrees, and a cost of $12.99. Of these options, the HS-81 Micro offers the fastest speed while maintain a moderate torque and low cost making it the optimum choice for a trigger servo. The HS-81 micro has an operating voltage of 4.8 to 6.0 volts and a required pulse of 3.0 to 5.0 volt peak square wave.

For the pitch, a moderate speed servo with high torque within a reasonable price will be sufficient. The angle of rotation should be 90 degrees in order to provide a reasonable firing angle while still keeping a minimum cost. The first option is a Hitec HS-8055BB Mega power servo motor. **[41]** The HS-8055BB is an analog servo with 274.96 oz-in torque, a moderate speed of 0.19 seconds per 60 degrees and a cost of $39.99. Another option is a HS-5805MG servo. The HS-5805MG is a digital servo with a torque of 274.96 oz-in torque, a moderate speed of 0.19 seconds per 60 degrees, and a cost of $79.99. The third option is a Futuba S3306 servo. The S3306 is an analog servo with a torque 266.5 oz-in, a moderate speed of 0.20 seconds pre 60 degrees, and a cost of $39.99. The two Hitec servo motors offer a slightly higher torque and slightly higher speed than its Futuba counter parts. Between the HS-8055BB analog servo and the HS-5805MG digital servo, the only difference is the increase in cost. Although the HS-8505MG digital servo may be more accurate, the double in price is discerning. For this reason, the Hitec HS-8055BB analog servo will be incorporated to control the pitch. The HS-8055BB has an operating voltage of 4.8 to 6.0 volts and a required pulse of 3.0 to 5.0 volt peak square wave.

For the yaw, a moderate speed servo with high torque within a reasonable price will be sufficient. The angle of rotation should be greater than 90 degrees, but less than or equal to 180 degrees. The first option is the Hitec HS-8155BB Mega

Sail Arm. The HS-8155BB is an analog servo with a torque of 274.96 oz/in, a moderate speed of 0.19 seconds per 60 degrees, and a cost of $44.99. The HS-8155BB has a stock rotation of 140 degrees, but no option to rotate 180 degrees. **[42]** The second option is once again the Hitec HS-805BB Mega power servo. The HS-805BB servo rotates 90 degrees at stock but can be stretched to 180 degrees for a total cost of $49.99. Since most webcams do not have a 180 degree field of view, a servo rotating a full 180 degrees is not necessarily needed. In order to slightly reduce cost, the HS-8155BB Mega Sail Arm will be incorporated to control the yaw. This servo has an operating voltage 4.8 to 6.0 and a required pulse of 3.0 volt peak square wave.

As previously mentioned, all servo motor prices were estimated using servocity.com. When actually purchasing the servos, there are multiple websites offering the chosen servos at different prices and shipping cost. Popular websites include ServoCity, Amazon, and HobbyHorse. Table 5 has a summary of the cost of each servo and shipping cost per order.

According to the frequently asked questions at the ServoCity website, all orders under $200.00 are shipped by the United States Postal Service and may take up two eight days. **[43]** If expedited shipping is needed, the customer must contact ServoCity by phone which may be a bit cumbersome during a time crunch. All orders under $200.00 have a shipping cost of $6.99. According to the ServoCity website, a Hitec HS-81 servo motor cost $17.99, a Hitec HS-805BB servo motor cost $39.99, and a HS-815BB servo motor for $44.99.

At the Amazon website a Hitec HS-81 servo motor cost $14.09, a Hitec HS-805BB servo motor cost $36.52, and a HS-815BB servo motor for $44.33. There are multiple shipping methods offered by Amazon including standard three to five business days and next day shipping. According to the help section at the Amazon website, standard shipping cost $4.99 per shipment and next day shipping cost $15.99.

At the HobbyHorse website a Hitec HS-81 servo motor cost $12.69, a Hitec HS-805BB servo motor cost $39.99, and a HS-815BB servo motor for $44.99. Similar to ServoCity, the HobbyHorse website only has standard shipping readily available with no mention of the possibility of next day delivery. According to the shipping section at the HobbyHorse website, standard shipping cost $7.00 per order on all order under $125.00. **[44]**

Table 5 is a summary of the servo motor cost and shipping cost per order at three popular websites: servocity.com, amazon.com, and hobbyhorse.com. Amazon currently has the best deal on the HS-8055BB and HS-815BB servo motors. Hobbyhorse has the best deal on the Hitec HS-81 servo motor. However, since hobbyhorse has an additional $7.00 charge for shipping to purchase the HS-81 servo motor, the cost surpasses the $14.09 cost of the HS-81 servo motor from amazon.com. The cheapest method is to purchase the servo motors from amazon.com

Summary of Servo Motor Cost			
Servo motor	**servocity.com**	**amazon.com**	**hobbyhorse.com**
Hitec HS-81	$17.99	$14.09	$12.69
Hitec HS-805BB	$39.99	$36.52	$39.99
Hitec HS-815BB	$44.99	$44.33	$44.99
shipping per order	$6.99	$4.99	$7.00

Table 5 - Cost summary of servo motors

In summary, the trigger will be controlled by a Hitec HS-81 servo motor for $14.09, the pitch will be controlled by a Hitec HS-805BB servo motor for $36.52, and the yaw will be controlled by the Hitec HS-815BB servo motor for $44.33. Shipping will cost $4.99 if the servo motors are packed and shipped together. The total cost of the servos is estimated to be $99.93. Currently, all servo motors will be purchased from amazon.com

4.1.2.1 Servo Motor Specifications

The Hitec HS-81 servo motor has an operating voltage between 4.8 and 6.0 volts. With an operating voltage of 4.8 volts, the servo has a stall torque of 36.10 oz-in, a speed of 0.11 seconds per 60 degrees, and a 220 mA current drain. At an operating voltage of 6.0 volts, the servo has a torque of 41.66 oz-in, a speed of 0.09 seconds per 60 degrees, and a 280 mA current drain. This servo requires a 3 to 4 volt peak to peak square wave pulse. The HS-81 servo motor uses the standard pulse duration ranging from 900 microseconds to 2100 microseconds. A 900 microsecond pulse will set the servo motor to 0 degrees. A 1500 microsecond motor will set the servo motor to 45 degrees. A 2100 microsecond servo motor will set the servo motor to 90 degrees clockwise. **[45]** Assuming the rotation in degrees is linear with the pulse width, the position can be estimated by pulse width = 900+40/3*A where A is the desired angle in degrees. The connector wire length is only 6.29 inches, so an extension may be needed when connecting the servo motor to the PCB. The HS-81 servo motor is 1.16 inches high, 1.17 inches long, and 0.47 inches wide. The mounting holes are 0.176 inches in diameter.

The Hitec HS-805BB servo motor has an operating voltage between 4.8 and 6.0 volts. With an operating voltage of 4.8 volts, the servo has a stall torque of 274.96.10 oz-in, a speed of 0.19 seconds per 60 degrees, and a 700 mA current drain. At an operating voltage of 6.0 volts, the servo has a torque of 343.01 oz-in, a speed of 0.14 seconds per 60 degrees, and an 830 mA current drain. Assuming the servo will have a one inch arm, running the servo motor at 5.0 volts should theoretically meet the requirement for 91.5 oz-in. This servo requires a 3 to 4 volt peak to peak square wave pulse. The HS-805BB servo motor uses the standard pulse duration ranging from 900 microseconds to 2100 microseconds. A 900 microsecond pulse will set the servo motor to 0 degrees. A 1500 microsecond motor will set the servo motor to 45 degrees. A 2100 microsecond servo motor will set the servo motor to 90 degrees clockwise. Assuming the rotation in degrees is linear with the pulse width, the position can be estimated by pulse width = 900+40/3*A where A is the desired angle in degrees. The connector wire length is 11.81 inches, which should be well within length to reach the PCB. The HS-805BB servo motor is 2.26 inches high, 2.59 inches long, and 1.18 inches wide and weighs 5.36 ounces. The mounting hole diameter is unlisted but will be assumed to four 0.176 inch diameter holes.

The Hitec HS-8155BB servo motor has an operating voltage between 4.8 and 6.0 volts. With an operating voltage of 4.8 volts, the servo has a stall torque of 274.96.10 oz-in, a speed of 0.19 seconds per 60 degrees, and a 700 mA current drain. At an operating voltage of 6.0 volts, the servo has a torque of 343.01 oz-in, a speed of 0.14 seconds per 60 degrees, and an 830 mA current drain. Assuming the servo will have a one inch arm, running the servo motor at 5.0 volts should theoretically meet the requirement for 91.5 oz-in. This servo requires a 3 to 4 volt peak to peak square wave pulse. The pulse width duration is not specified which may cause an issue since this servo does not operate a standard ninety degrees. It is assumed that the servo motor operates similar to a standard ninety degree servo motor stretched to a length of 140 degrees. The HS-815BB servo motor is assumed to use the standard pulse duration ranging from 900 microseconds to 2100 microseconds. A 900 microsecond pulse will set the servo motor to 0 degrees. A 1500 microsecond motor will set the servo motor to neutral 70 degrees. A 2100 microsecond servo motor will set the servo motor to 140 degrees clockwise. Assuming the rotation in degrees is linear with the pulse width, the position can be estimated by pulse width = 900+60/7*A where A is the desired angle in degrees. The HS-815BB servo motor is 2.28 inches high, 2.59 inches long, and 1.18 inches wide and weighs 5.36 ounces. The mounting hole diameter is unlisted but will be assumed to four 0.176 inch diameter holes. These comparisons are below in table 6.

Summary of Servo Motor Specifications							
	HS-81		HS-805BB		HS-815BB		
Oper. Voltage	**4.80**	**6.00**	**4.80**	**6.00**	**4.80**	**6.00**	**V**
Torque	31.1	41.7	275.0	343.0	275.0	343.0	**oz-in**
Speed	0.11	0.09	0.19	0.14	0.19	0.14	**S/60°**
Current	220	280	700	830	700	830	**mA**
Required Pulse	3.00	5.00	3.00	5.00	3.00	5.00	**$V_{pk\text{-}pk}$**
Pulse Equation	$900+\frac{40}{3}A$		$900+\frac{40}{3}A$		$900+\frac{60}{7}A$		**µS**
Wire Length	6.29		11.81		11.81		**in**
Height	1.16		2.26		2.26		**in**
Width	0.47		1.18		1.18		**in**
Length	1.17		2.59		2.59		**in**
Hole Diameter	0.176		0.176		0.176		**in**

Table 6 - Servo Motor Specifications

4.1.2.2 Servo Motor Implementation

The connector wire for each servo has three pins. One pin is dedicated to the operating voltage which ranges from 4.8 to 6.0 volts. The operating voltage will be discussed in detail further ahead. The second pin is dedicated to ground. This pin will share a common ground for all analog devices that require ground. The third pin is dedicated to the pulse width signal. This pin will be discussed further ahead.

In order to function to the specifications provided online, the servo motors must receive the appropriate operating voltage. A linear voltage regulator is capable of producing an output of a voltage between 4.8 and 6.0 volts. Assuming an operating voltage of 5.0 volts, a commonly used linear voltage regulator would be the LM7805 from National Semiconductor. According to the national semiconductor datasheet, the minimum input voltage to maintain regulation for the LM7805 is 7.5 volts. Assuming the operating voltage for the servo motors is at 5 volts, the minimum voltage drop across the LM7805 would be 2.5 volts.

While operating at 5 volts, the servo motor required approximately 750 milliamps. The current going into a linear voltage regulator can be estimated to the current going out of the linear voltage regulator, or approximately 750 milliamps for the pitch and yaw servo motors. The power dissipated across the LM7805 is approximately 1.875 watts.

Power dissipation in the watts range is somewhat problematic for a linear voltage regulator. According to the LM7805 datasheet, the maximum power dissipation can be calculated from the formula below. Since the LM7805 will be in a somewhat enclosed box possibly outside, the ambient temperature, TA, can be approximated to 35 degrees Celsius. From the datasheet, the maximum junction temperature, TJMAX, is equal to 125 degrees Celsius, and the junction-to-ambient temperature is equal to 39 degrees Celsius per watt assuming no heat sink. From the equations below the maximum power dissipation is 2.308 watts. This would theoretically work, however it was assumed that the input voltage was only 7.5 watts. The maximum input voltage before passing the maximum power dissipation is calculated to be 10.5 watts. If the entire PCB would be ran on a 12 volts power supply, the LM7805 linear voltage regulator would burn the instant the servo motors were required to turn. This is only assumed for the pitch and yaw servo motors. For the trigger servo motor, the maximum current drain is theoretically 280 milliamps. Assuming a voltage drop of 7 volts and current drain of 280 milliamps, the power dissipation is calculated to be approximately 1.96 watts, still within the max power dissipation of the LM7805. **[46]**

$$PDMAX = (TJMAX - TA)/\theta JA.$$

$$PDMAX = (125 - 35)/39$$

$$PDMAX = 2.308$$

The alternative to a linear voltage regulator is a switching voltage regulator. A switching regulator rapidly switches elements on and off so that the each element is neither fully conduction or fully switched off. This leads to extremely low power dissipation and the reason to why switching regulators will be more suited for the high power dissipation environment caused by the servo motors. The main advantage to using the switching voltage regulator is its low power dissipation allowing the servo motors to be properly powered. Theoretically, all three servo motors can be connected to the same switching voltage regulator. However, a large drawback is that the output will cause a ripple voltage that needs to be properly filtered to prevent a constantly changing operating voltage and to prevent any possible electromagnetic interference.

Like linear voltage regulators, there is a wide variety of options to choose from. However, the switching regulator most familiar to the group is the LM2576 simple switcher voltage regulator from National Semiconductor. The LM2576 is a switching regulator frequently used in the electronics II laboratory at the University of Central Florida. According to the datasheet, the LM2576 has 3.3V, 5V, 12V, 15V, and adjustable output versions ranging from 1.23V to 37V. The LM2576 has a guaranteed three amp output current, which is much higher than the linear regulator counterparts. For this reason, the LM2576 will be adjusted to 5 volts output and power all three servo motors. **[47]**

Another pin for each servo motor is dedicated to the pulse width modulation signal in order to properly position the servo motor. Since each servo motor requires a 3 volts peak-to-peak to 5 volts peak-to-peak voltage signal, the microcontroller controlling the servo motors should be able to output at least 3 volts peak-to-peak, on three pins. The current microcontroller to complete this task is the PIC16F1503 from Microchip Technologies Inc. Since the PIC16F1503 operates at 2.3 volts to 5.5 volts and has 17 bidirectional pins, the PIC16F1503 should easily complete this task. The pitch servo signal will be placed on pin 10, port RB7, the yaw servo motor signal will be placed on pin 11, port RB6, and the pitch servo motor will be placed on pin 12, port RB6. The high voltage for each pin will be defined as 4 volts peak to peak.

However, between each servo motor and the microcontroller controlling the servomotor, a unity gain buffer will be implemented to help prevent any current going into the voltage driven signal line. If, for some reason the PIC16F1503 microcontroller cannot deliver a 4 volt peak-to-peak pulse, the unity gain will be converted to a simple non-inverting amplifier with the appropriate gain. For either case, the operational amplifier chosen must have a very high slew rate. Choosing an operational amplifier with a high slew rate will quickly allow the output voltage to switch on in off. The input noise should remain largely unimportant since the signal line is driven by the width of the pulse and not the amplitude. Two high slew rate operational amplifiers the group is familiar with include the TL084 operational amplifier by Texas Instruments and the LF351 by National Semiconductor.

The TL084 operational amplifier by Texas instruments has a typical slew rate at of 13 volts per microsecond and implemented with bipolar junction transistors. With a 13 volt per microsecond slew rate, the TL084 can typically switch from the high of 4 volts to a low of 0 volts in approximately 300 nanoseconds. The main advantage of TL084 is the four operational amplifiers built into one IC chip. This will not only minimize the number of supply voltage inputs but reduce the amount physical space taken up by buffers. Assuming the TL084 will have the same supply voltage as the operating voltage of the servo motors to reduce the number of voltage regulators, a large disadvantage is brought forward by the bipolar junction transistor implementation. Typically with a BJT operational amplifier, the output cannot reach the supply voltage. This may be problematic if the total voltage lost surpasses two volts. In other words, if maximum peak output voltage is less than 3 volts due to a 5 volt supply voltage, the servo motor will not function correctly. **[48]**

The LF351 operational amplifier by National Semiconductor has a typical slew rate of 13 volts per microsecond and implemented with both junction gate field-effect transistor and bipolar junction transistors. Similar to the TL084 operational amplifier, a 13 volt per microsecond slew rate will allow the LF351 to switch from the high of 4 volts to a low of 0 volts in approximately 300 nanoseconds. The main disadvantage to the LF351 is that there is one operational amplifier per IC. Although this could be an advantage to replace a broken component, it will take up more space and require multiple lines for supply voltages. Furthermore, the LF351 operational amplifier was recently discontinued and may be difficult to acquire. However, the group currently has access to multiple LF351 operational amplifiers. **[49]**

Considering the options, the choice boils down to preference of trading space between an additional voltage regulator to power a TL084 a single operational or to use an additional voltage regulator to power the LF351 with multiple lines to power multiple operational amplifiers. Of the options, incorporating the TL084 operational amplifier for a buffer is the preferred choice.

The TL084 has a total for four operational amplifiers and 14 pins. Pins 4 and 11 will be connected to a LM7808 linear voltage regulator outputting 8 volts. Pin 1 will be connected to the pitch servo signal line, pin 7 will be connected to the yaw servo motor signal line, and pin 14 will be connected to the trigger servo motor signal line. Pins 3, 5, 12 of the TL084 will be connected to pins 13, 12, and 11 of the PIC16F1503 microcontroller. Pins 2, 6, and 13, will be connected to pins 1, 7, and 14, respectively to provided negative feedback to complete the unity gain buffer.

4.1.3 Capacitive Touch Controller

One of the popular functions of a capacitive touch device is wheel scrolling where a user can simply rotate their pointer clockwise or counter clockwise to scroll through a list or change the angle of a device. There have many past projects that incorporated wheel scrolling and simulated button pressing have been incorporated onto the same capacitive touch device. For manual controlling an autonomous turret, it was decided that wheel scrolling, although fun, is not the most functional method for controlling a direction. Furthermore, since the servo motors operate will be operating at a specified speed, wheel scrolling faster or slower will not turn the turret faster or slower. This further diminishes the functionality of wheel scrolling.

Instead of wheel scrolling, a system of five simulated buttons on the capacitive touch booster back will be used in order to control the pitch, yaw, and trigger. For controlling the direction there will be two methods. Tapping the controller will turn the turret one-degree. Holding the controller will turn the turret 10 degrees per second. This will result in the turret able to turn a full 140 degrees in the yaw direction in approximately fourteen seconds. Touching the controller towards the left will rotate the turret counter clockwise in the yaw direction. Touching the controller towards the right will rotate the turret clockwise in the yaw direction. Touching the controller towards the top will rotate the turret up in the pitch direction. Touching the controller towards the bottom will rotate the turret down. To fire the turret, the capacitive touch controller should be tapped towards the middle.

Currently, it is planned to have the capacitive touch controller built onboard the autonomous turret along with the microcontroller controlling the servos. In or methods der for the touch controller to communicate with the microcontroller, a method of transmitting servo degrees must be implemented.

One option is when the capacitive touch controller is touched, the current angle will be transmitted to the servo microcontroller. For example, the capacitive touch controller will transmit 45 degrees, then 46 degrees, then 47 degrees on repetitive touches. The microcontroller controlling the servos will convert these degrees into pulse width modulation square wave to turn the servo motors to the appropriate angle. The advantage to this option is the accessibility of adding additional hardware such as a small LED display to display the current angle or to have status LED when the servo motor has reached its turning limit. However, to parallel transmit a range of 140 degrees up to a resolution of one degree, multiple bits must be used. The number of bits needed can be calculated by the following formula:

$$2^n = 140 \rightarrow n = log(140,2) \rightarrow n = 7.123$$

According to above formula, the minimum number of bits needed to parallel transmit 140 degrees with a resolution of one degree is eight bits for the yaw direction. To calculate the number of bits required to transmit the pitch 90 degrees parallel, the same formula can be used:

$$2^n = 90 \rightarrow n = log(90,2) \rightarrow n = 6.492$$

According to the above formula, the minimum number of bits needs to parallel transmit 90 degrees with a resolution of one degree is seven bits. However, to distinguish between pitch and yaw another bit must be dedicated to either pitch or yaw. For example, a 0 could represent pitch while a 1 can represent yaw. Overall, a minimum number of 9 bits must be used. Assuming the most significant bit represents either pitch or yaw, a transmission of 101011010 will translate to 90 degrees along the yaw direction and a transmission of 001011010 will translate to 90 degrees along the pitch direction. Since the MSP430 has 10 input and output ports, it is theoretically possible to transmit nine bits in parallel. However some of the input and output ports are dedicated to the capacitive touch controller. A summary of this method can be found in table 7.

Summary of Angle Transmission for Method 1									
Pin	**P8**	**P7**	**P6**	**P5**	**P4**	**P3**	**P2**	**P1**	**P0**
Yaw Starting	1	0	0	0	0	0	0	0	0
Yaw Ending	1	1	0	0	0	1	1	0	0
Pitch Starting	0	0	0	0	0	0	0	0	0
Pitch Ending	0	0	1	0	1	1	0	1	0
Fire	1	1	1	1	1	1	1	1	1

Table 7 - Angle Transmission for Method 1

Another option is combine the pitch and yaw into only eight bits. This method will not have a bit dedicated to either the pitch or yaw direction, but instead use all eight bits to transmit both the pitch and a yaw. Since the range of the yaw servo motor is limited from 0 degrees to 140 degrees, the eight bits controlling the angle, disregarding the most significant bit, would range from 00000000 to 10001100. This leaves the binary numbers of 10001101 to 11111111 currently unused. Theoretically, it is possible to use this binary range for the pitch. Since the 140 degrees will be dedicated to the yaw direction and 90 degrees will be dedicated to the pitch direction, the total number of degrees is simply calculated at 230 degrees. In other words, the binary range from 00000000 to 10001100 will represent the degree range from 0 degrees to 140 degrees in the yaw direction while the binary range from 10001101 to 11100111 can represent the degree range from 0 degrees to 90 degrees in the pitch direction. Using this method could the binary range of 11101000 to 11111111 various combinations of firing methods when the capacitive touch controller is touched towards the center. A summary of this table can be found in table 8.

Summary of Angle Transmission for Method 2								
Pin	**P7**	**P6**	**P5**	**P4**	**P3**	**P2**	**P1**	**P0**
Yaw Starting	0	0	0	0	0	0	0	0
Yaw Ending	1	0	0	0	1	1	0	0
Pitch Starting	1	0	0	0	1	1	0	1
Pitch Ending	1	1	1	0	0	1	1	1
Fire	1	1	1	1	1	1	1	1

Table 8 - Angle Transmission for Method 2

The other option is for the capacitive touch microcontroller to only transmit the change in angle whenever the capacitive touch controller is touched. One bit will be dedicated to the change in degrees. For example, a 0 will represent a one degree turn, while a 1 will represent a ten degree turn per seconds. A second bit will be dedicated to the direction of turning the servo meters. For example, a 0 will represent a clockwise turn, and a 1 will represent a counter-clockwise turn. A third bit will be dedicated to which direction must be changed. For example a 0 will represent a change in pitch and a 1 will represent a change in yaw. However, since this method is only transmitting three bits to represent the change in degrees for the servo motors, the microcontroller controlling the servo motors is unable to determine if the capacitive touch controller is being touched or still transmitting its previous command. In order to resolve this, a fourth bit can be dedicated to this case. For example a 1 can represent the capacitive touch controller being touched while a 0 can represent the capacitive touch controller not being touched. When this bit pin goes high, an interrupt service routine will occur to update the angle. An example of this scheme can be seen in table 9.

Summary of Angle Transmission for Method 3				
Pin	**P3**	**P2**	**P1**	**P0**
Bit	**Degree change**	**Direction change**	**Pitch or Yaw**	**touched?**
High	10°	counter-clockwise	yaw	touched
Low	1°	clockwise	pitch	not touched

Table 9 - Angle Transmission for Method 3

Considering the options, the preferred method is to transmit the correct angle is method two. The main advantage to using method two over method one is the ability to gain an extra pin if needed. However, the disadvantage over using method two over method is the slight increase in difficulty to recognize the difference between a pitch and yaw transmission. The advantage to using method one over method two is that the capacitive touch microcontroller always knows what the angle is. This will allow the microcontroller controlling the servo motors to simply read the capacitive touch microcontroller whenever an update is needed. The main disadvantage to using method two over method three is the need for four additional extra bits. Because of the above reasons, method two is the balance between the number of pins needed and ease of transmitting the correct angle.

The MSP430 Launchpad, values at $4.30 plus shipping, generously donated by Texas Instruments comes with two different MSP430 microcontrollers, a MSP430G2211 and a MSP430G2231. The capacitive touch booster pack, valued at $10.00 plus shipping, also generously provided by the Texas instruments comes with a MSP430G2452 microcontroller. The MSP430G2452 microcontroller comes pre-programmed with an example case and will need to be replaced to properly control the autonomous turret.

The MSP430G2452 is a replacement for the MSP430G2231 that comes standard with the LaunchPad. The MSP430G2452 microcontroller that comes with the capacitive touch booster pack requires a supply voltage between 1.8 volts and 3.6 volts. When operating at 1 MHz clock and 2.2 volts, the microcontroller only drains 220 µA. For the autonomous turret, the MSP430G2452 will have a supply voltage of 2.2 volts with a 1 MHz clock frequency. **[50]** This clock frequency was chosen mostly for a round number.

In order for the capacitive touch controller to receive and transmit commands, proper pin allocations must be generated. The capacitive touch booster pack has six touch sensors: up, left, right, down, middle, and proximity. Each of these sensors will correspond to one pin on the microcontroller. According to the MSP430G2x52 data sheet, there are 6 purely general-purpose digital input-output pins. Proximity will be dedicated to port 2.0, left will be dedicated to port 2.1, down will be dedicated to port 2.2, right will be dedicated to port 2.3, up will be dedicated to port 2.4, and, if needed, middle will be dedicated to port 2.5. Disregarding the six purely general-purpose digital I/O pins, there are an additional ten pins that can be used as general-purpose digital I/O pins to transmit the capacitive touch controller degrees. Port 1.7 will represent the most significant bit of method two. Port 1.0 will represent the least significant bit of method one. There are two additional required pins in order to properly power the capacitive touch controller. The supply voltage is found on pin 1 and will be set to 2.2 volts. The ground reference is found on pin 20 and will share common ground with other components. A summary of the pin allocation can be found in the table 10 below.

MSP430G2452 Pin Allocations					
Pin #	**Port Name**	**Description**	**Pin #**	**Port Name**	**Description**
1	DVcc	Direct voltage 2.2V	11	P2.3	Right Sensor
2	P1.0	1st LSB of Angle	12	P2.4	Up Sensor
3	P1.1	7th MSB of Angle	13	P2.5	Middle Sensor
4	P1.2	6th MSB of Angle	14	P1.6	2nd MSB of Angle
5	P1.3	5th MSB of Angle	15	P1.7	1st MSB of Angle
6	P1.4	4th MSB of Angle	16	RST	Reset button
7	P1.5	3rd MSB of Angle	17	TEST	NOT USED
8	P2.0	Proximity Sensor	18	Xout	Crystal Oscillator
9	P2.1	Left Sensor	19	Xin	Crystal Oscillator
10	P2.2	Down Sensor	20	DVSS	Ground reference

Table 10 - MSP430 Pin Allocation

In order for the capacitive touch controller to communicate with the PIC16F1503 microcontroller controlling the servo motors, the two components must be tied together. Since pins 11, 12, and 13 of the PIC16F1503 microcontroller are dedicated to the servo motors, one option is to dedicate ports RC0 to RC7 to communicate to ports P1.0 to P1.7 on the MSP430G2452 capacitive touch microcontroller. For example, Port RC0 will be tied to port P1.0, port RC1 will be tied to Port P1.1, and so forth. The pin locations tied together relative to the MSP430G2452 and the PIC16F1503 microcontrollers can be found in table 11.

MSP430G2452 to PIC16F1503 Pins				
Description	**Pin #**	**Port Name**	**Pin #**	**Port Name**
1st MSB of Angle	15	P1.7	9	RC7
2nd MSB of Angle	14	P1.6	8	RC6
3rd MSB of Angle	7	P1.5	5	RC5
4th MSB of Angle	6	P1.4	6	RC4
5th MSB of Angle	5	P1.3	7	RC3
6th MSB of Angle	4	P1.2	14	RC2
7th MSB of Angle	3	P1.1	15	RC1
1st LSB of Angle	2	P1.0	16	RC0

Table 11 - Manual Control and Servo Motor Controller communication pins

The MSP430G2452 will be developed in the Texas Instrument's free limited version of Code Composer Studio version 4.1 and on the MSP430 LaunchPad. Multiple functions will be developed in order to keep the capacitive touch sensors and communication to the microcontroller controlling the servo motors. The functions are as followed: tap_up, hold_up, tap_right, hold_right, tap_down, tap_left, hold_left. Each function will set the appropriate pins to either high or low. For the tap functions, a tap will be considered a touch less than 20 milliseconds. Since the clock frequency is at 1 Mhz, 1 millisecond second should translate to (10^-3)/(10^-6) or 1000 cycles. Each tap function will increase the angle by one degree. For the hold functions, a hold function will be considered when a touch is greater than 20 milliseconds. The hold function will be released when the user releases the touch controller. Each hold function will increase the angle by 10 degrees. When the microcontroller is started, the MSP430 will transmit first transmit 45 degrees both in the yaw and pitch direction. A summary of the functions are found in table 12.

MSP430 Function Summary	
Function Name	**Function Description**
tap_up	increase pitch angle by 00000001
tap_left	decrease yaw angle by 00000001
tap_right	increase yaw angle by 00000001
tap_down	decrease pitch angle by 00000001
hold_up	increase pitch angle by 00001010
hold_left	decrease yaw angle by 00001010
hold_right	increase yaw angle by 00001010
hold_down	decrease pitch angle by 00001010

Table 12 - MSP430 Function Summary

In order to clarify proper use of the microcontroller a few user cases must be assumed and solved. If the user taps more than one sensor, the microcontroller will only consider the first touch. For example, if the user holds the left sensor while tapping the up sensor, the microcontroller will ignore the up tapping until the user has stopped holding the left sensor. If the turret has a zero degree position in the yaw direction, or a 900 microsecond pulse width, and the user continues to either hold or tap the left sensor, the capacitive touch microcontroller will continue to transmit hold left or tap left. However, the microcontroller controlling the servos must ignore any further hold left of tap left function that attempt to reduce the pulse width modulation below the minimum 900 microseconds.

4.1.4 Base

Before designing the base, a large factor must be decided before continuing. Similar to how a longer wrench is easier to turn that a shorter wrench because of the principle of torque, turning the base towards the outside of the base should be easier to turn. This will result into relieving some of the torque off the servo motors. The advantage to designing the base with reducing torque in mind is the possibility to increase the weight from the paintball gun ammo, compressed air tank, and the base itself. This will allow for a larger choice of sturdier and heavier base materials. However, the main disadvantage is the slower speed. In other words the tradeoff for reduced torque on the servo motor is the reduced speed in turning the base.

Alternatively the base can be designed with no plans to reduce the torque applied to the servo motors. The main advantage is the ease of designing the base is substantially easier while its disadvantage is the possibility of running into a torque issue once the base, ammo, and compressed air are attached to the paintball gun. The workaround for this is to offset the weight of the paintball gun is to have the compressed air tank attached to the stationary part of the base. The compressed air tank can be attached to the paintball gun by a simple hose. The weight of the paintball gun can be further reduced by simply reducing the amount of ammo loaded into the paintball gun at any time.

Considering the two options, the preferred method will be to reduce the amount of torque on the servo motors by reducing the weight of the paint ball gun and base. Once again, the compressed air tank will be attached to a stationary part of the base and the amount of ammo will be limited. Considering the servo motors have a significantly higher torque than the estimated torque found in the research section, this method should be safe. Also, the torque can be increased or decrease by changing the operating voltage.

Ideally, the base will have three sections. The bottom section, or the storage section, will simply store the battery, the compressed air tank, spare ammo, and all of the electronics. The electronics found inside the bottom section will include the on-board tracking computer, yaw servo motor, the power source, and the printed circuit board. The tracking computer, power source, and PCB should be easily accessible in order to swap said components easily. In order to reduce the likelihood of the paintball gun recoil tipping the base, the bottom section will contain the majority of the base weight. The majority of this weight should be located towards the front of the base.

The middle section of the base will consist of a simple rotating plate connected to the yaw servo motor. The current plan is to use the Hitec HS-815BB Mega Sail Arm servo motor. Traditionally this servo motor is used to control the direction of the sail of hobbyist boats. This application is very similar to turning the paintball gun in the yaw direction.

The top section of the base will be attached to the middle section and will house the paintball gun and currently loaded ammo. In order to attach the paintball gun to the top section, two rods will pinch the back of the paintball gun and another set of two rods will pinch the front of the paint ball gun. The four rods will be attached to a simple rotating plate similar to the one found in the middle section. The servo motor will be attached to this rotating plate so that when the servo motor rotates, the paintball gun rotates accordingly.

4.1.5 Power Source

The power source is an essential component for powering the ATS system. It is needed since the system will portable and self-sufficient. The battery chosen to power is the 12V sealed lead acid battery. Similar to a car battery, just significantly smaller, this battery will provide enough voltage to ensure reliable energy to the ATS system.

Before designing and implementing the battery unit, the designers must first determine to place the battery. The base of the ATS system will be divided into three sections, top, middle and bottom. The bottom section will be considered the storage unit and since the battery unit is one of the largest components needed for the system, the bottom is a good location to keep the battery. This has multiple reasons for the location. Since most of the bottom section will store several components, including the battery, the bottom will weigh the most which will help stabilize the system from the recoil of firing.

Another reason for keeping the battery below with the PCB and onboard laptop is the designers will have the chance to hide the components both for cosmetics and to protect from environmental issues that could potentially affect the performance of the electrical components. So this is effective for several reasons, especially since the PCB will be stored with the battery.

The implementation of the battery will be a simple connection to the PCB. The main components on the PCB that will need the energy provided will the PIC16F1503 microcontroller from Microchip. The designers should be able to just solder the connections to the input power pin to provide power, and solder connections from the ground pin to ground the power. Once the power is implemented onto the microcontroller, the ATS system is essentially powered to perform as designed.

The battery has the capability to recharge, so the direct connections to battery will be easy for the designers to disconnect and remove from the system. This is also if the owners decide to exchange the battery. Since the battery is similar to a car battery, the designers can use similar technologies to those of the battery connections where connections can be surrounded in an isolative material, some kind of rubber, where the designer can easily remove by hand.

4.1.6 Alarm system

Before designing the alarm system, a decision needed to be made based on sound volume. Since the ATS system would need to successfully warn the incoming targets, it was crucial that alarms were heard before being fired upon. To do this, the designers have opted to implement two types of alarm systems, a continuous alarm and a set of speakers to play an audio file that will verbally warn incoming targets.

The first alarm is going to be the continuous alarm. This alarm system's loudness will be based on the input current voltage, the higher the input current voltage will result in a higher decibel level, making it louder and easier to hear. This alarm will be directly connected to the microcontroller so when the software detects the incoming target, a signal will be sent from the microcontroller triggering the alarm. The connection will be soldered to one of the I/O pins directly. Like all electrical devices, a ground wire will be connected to a shared ground for all other devices.

The second alarm is going to a set of speakers. The decision to implement a set of speakers is because it has adjustable volume settings and can verbally warn incoming targets that they are within firing distance. The speakers will be connected directly to the laptop just like any other speaker system. So when an object is within 75 feet of the system, the software will trigger this alarm by playing a recorded audio file. This warning is intended for incoming humans so that they can properly understand the system's intent.

The designers choose to use both of these alarms systems to replicate real life alarm systems. Generally, when a continuous sounding alarm goes off, it is loud and high pitched, which usually initially grabs the attention of the audience. At the same time, verbal commands are being announced warning of the imminent danger or providing specific instructions. For this reason, the designers of the system chose to use both alarm systems.

4.1.7 Web camera

The web camera is typical camera but a vital component in the ATS system. It is the initial component in that will kick start the targeting. So the web camera needs to be properly implemented to the system to ensure the vision is provided to the laptop so the software can detect incoming targets. The web camera will be connected directly to the laptop via a universal serial bus (USB) cable, just like the mouse or a thumb drive is connected to a laptop.

On the other end of the cable, the camera itself will be tied to the gun. It will be tied looking down the barrel of the gun to replicate the sights a human would use while aiming a gun itself. By doing this, the live video feed will stream directly to the laptop creating the illusion of the user looking down the barrel of the gun themselves. This will greatly increase accuracy because this method is used for accurately shooting objects. So calculating the trajectory of the projectiles won't have to be too complex which will increase accuracy.

4.1.8 Laser Pointer

The laser pointer will be connected through the microcontroller to be initiated. This will have its own dedicated to the operating voltage. Like every electrical component, the laser pointer will also tie to the ground pin, which will be shared for all devices that need to be grounded.

The laser pointer has an operating voltage between 3 and 4.2 volts. So in order for the design to meet this range, a linear voltage regulator would be used to manage the necessary voltages needed. The designers will adjust as necessary to ensure the laser pointer is performing to its abilities. Once satisfied, the laser pointer will then be tied onto the barrel of the gun. By doing this, laser pointer can aim down range towards incoming targets. **[51]**

Ideally, the shot should hit the exact location of the later pointer; however, just because the laser pointer will be aiming down field doesn't mean the gun will hit exactly where the laser is pointing to. This is caused mainly by the trajectory of the paintball. There are other factors that could affect the accuracy of the paintball such as distance and weather elements. The main objective is determining the prime location to attach the laser pointer onto the system. Of course it will be aiming down the barrel, similar to tactical weapons used in the armed forces.

The difficulty in determining this location is based off of two main reasons, distance from the barrel and angle. The slightest angle tilt can largely affect where the laser points further way from the gun system. Ideally, the laser pointer should be parallel to the barrel, the distance between the barrel and laser point will affect the overall sights of the gun. So the combination of the two is what will provide the greatest results. To achieve this, the designers will go through extensive testing to determine the optimal location to place this laser pointer. This will be discussed furthermore in the hardware testing sections.

4.1.9 Paintball Gun Components

The following sections discuss the paintball gun components in further detail. A majority of the components are currently possessed such as the spyder paintball gun, Co2 tank and hopper. Components such as the paintball ammunition, coil hose and Co2 gas will need to be acquired for the completion of the ATS system.

4.1.9.1 Spyder Paintball Gun Classis Series

For the gun of the sentry a Spyder .68 caliber paintball gun will be fitted to the base. The gun was a good choice because one of the group members already owned one and it is lightweight and durable. The gun is unmodified with its original twelve inch barrel. The gun is only usable with Co2 gas and in order to keep the base lightweight a hose will be fitted from the tank to the gun. The trigger is metal, which is a requirement to withstand the constant abuse from the trigger servo motor. The top of the gun is fitted with an aiming sight on which a laser sight will be affixed. The gun also has a place to feed the paintballs from an attached hopper. It is imperative that the gun is maintained during testing; if the gun is not cleaned regularly to prevent jams and paint build up the paintballs could break in the barrel which would render the sentry useless. The gun is by not top of the line, but improvements can be made in the future by adding a longer barrel or mounting a new gun entirely. The gun is a great choice because it is lightweight and is previously owned by a group member, it will be used in the build to show functionality of ATS. Paintball gun being used is displayed in figure 7.

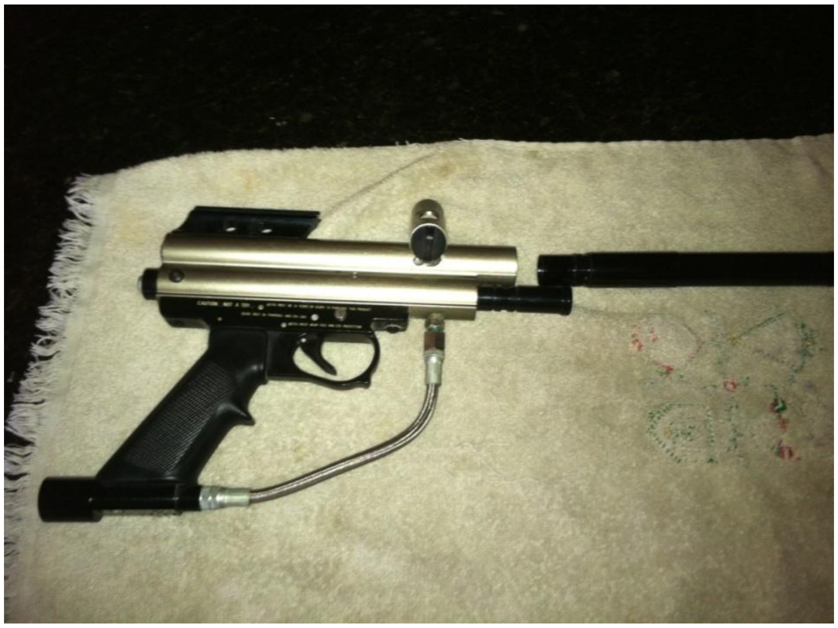


Figure 7 - Spyder .86 caliber paintball gun

4.1.9.2 Co2 Tank

The Co2 tank that will be used for ATS is a 20 oz Striker High Pressure Aluminum Alloy tank. The tank should provide enough Co2 for about one thousand shots, this is highly dependent on how clean the gun is and the velocity the gun is set to. The tank is composed of an aluminum alloy cylinder and a valve. The tank can be refilled at any sporting goods store or paintball gun supply store for fewer than five dollars. The Co2 tank being used is displayed in figure 8 below.

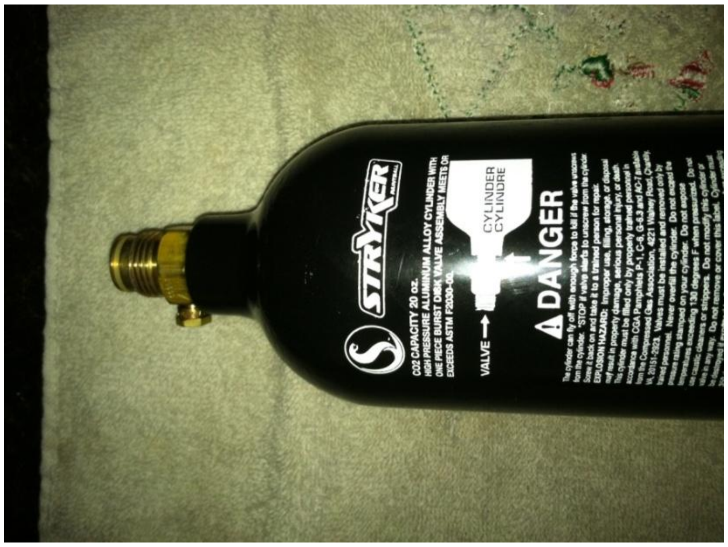


Figure 8 - 20 oz Striker High Pressure Aluminum Alloy tank

4.1.9.3 Co2 Tank Remote Coil Hose

In order to keep the base lightweight a coil hose will be used to attach theCo2 tank to the gun. The hose used for this project will be the JT Tactical C3 Remote Coil Hose. **[52]** The hose can be bought from around thirty dollars on Amazon and features many positive reviews giving an overall rating of four out of five stars.

4.1.9.4 Paintball Hopper

The paintball hopper is used to hold the paintballs. It is essentially the equivalent of a magazine for an actual gun. The hopper the team is using is a ViewLoader 200 ct. hopper. Since the hopper can hold 200 paintballs the team will initialize the ammo count to 200 and keep track of it as the gun is fired.

4.1.9.5 Paintballs

Paintballs are the ammunition that will be used by ATS. They are small gelatin capsules filled with non-toxic paint. Paintballs vary very much in size and quality. The paintballs needed for ATS must be .68 caliber balls. The quality of the paintballs is dependent on the hardness of the shell and the spherical shape. High quality paintballs are very thin so they break on contact and almost perfectly spherical. For the purpose of ATS the quality of the paintballs is not a high priority since the function of the turret is to show functionality. For testing purposes the group will purchase a cheap case of RPS 05630 Stinger Paintballs, a 2000 ct. case sells for $24 on Amazon and other online retailers. **[53]** For demonstration purposes the group will buy a nicer case of Diablo Heat paintballs which retails for $42 for a 2000 ct. online at zephyrpaintball.com. **[54]** This will ensure that on demonstration day high quality accurate paintballs are being used in order to ensure that are sure to mark the target.

The color of paintball the team has chosen will also play an important role in the project. Since ATS defines targets that are non-green as the enemy, green colored paintballs will be used. Once the target is covered in a sufficient amount of green paint ATS will determine the target is no longer a threat and cease fire. This will be vital in saving ammunition and to avoid over firing on targets unnecessarily. The amount of green paint required to confirm a kill is still to be determined.

4.2 Software Design

Overall, computer vision is a new and constantly growing software field. In recent years, the popularity has grown dramatically. Also, with the help of open source software like OpenCV or Aforge.net, knowledge of computer vision and how it works has become easily accessible. The ATS' software will consist of a computer vision application that will make decision on whether or not to engage targets found within the field of vision based on the targets threat level. It will also have a facial recognition security measure to access the manual control application.

In figure 9, the software activity diagram, the process that the ATS will go through to determine whether or not to fire is demonstrated. First, the web camera will capture the video stream from the field of vision. This stream will be brought into the program from the videoCapture class. The VideoCapture class will then set up the device with the program so grabbing frames from the stream will be simple. Then that frame will be taken into the cvtColor function. A traditional RGB image will be transformed into an HSV image, which will make color detection easier, by being able to classify a wide range for hue and smaller ranges for saturation and value.

Once the frame has been converted, it must then be searched for good features to track. This will be done using an OpenCV algorithm. This function will then pass the image and an array of corners to cornerSubPix. CornerSubPix will then improve upon what was found in the previous function and return with it a corner count. At this point, objects have been detected and now need to be tracked once there have been two frames sent from the web camera. The algorithm chosen for ATS' computer vision application is the iterative Lucas-Kanade with pyramids. It will calculate the optical flow for the sparse feature set that was sent to it from the previously described functions.

At this point, trackTargets is called to update all of the positions and velocities for any and all targets found within the field of vision of the ATS. After the positions have been updated accordingly, the threatLevel must be set for each target. The threat level will be based on the color detected from the object. If the object is green, then it will be considered an ally, and any other targets will be considered enemies. After that, all targets have been appropriately updated, and now the system must chose a target to attack. It searches for the highest priority amongst all targets, and then determines if that target passes the threshold for a dangerous target. If so, then the sentry gun will engage the target immediately. Otherwise, the turret will wait until a target with enough priority for it to attack enters the field of vision of the system.

In the end, the videoWriter class will write what has happened to a file that can be viewed later by the user of the ATS. The rest of the functions detailed in the software design are auxiliary functions used to complete various tasks to support the main functions of the computer vision application. For example, the rectangle and line drawing functions can be used to demonstrate what the object being tracked actually looks like to the program when displayed on the user interface. Another extra feature will be an alarm. Once an object has been detected on the screen, an alarm will play alerting the target that it is in range of the system. After a short period, the target will be attacked until it leaves the field of vision of the ATS. **[24]**

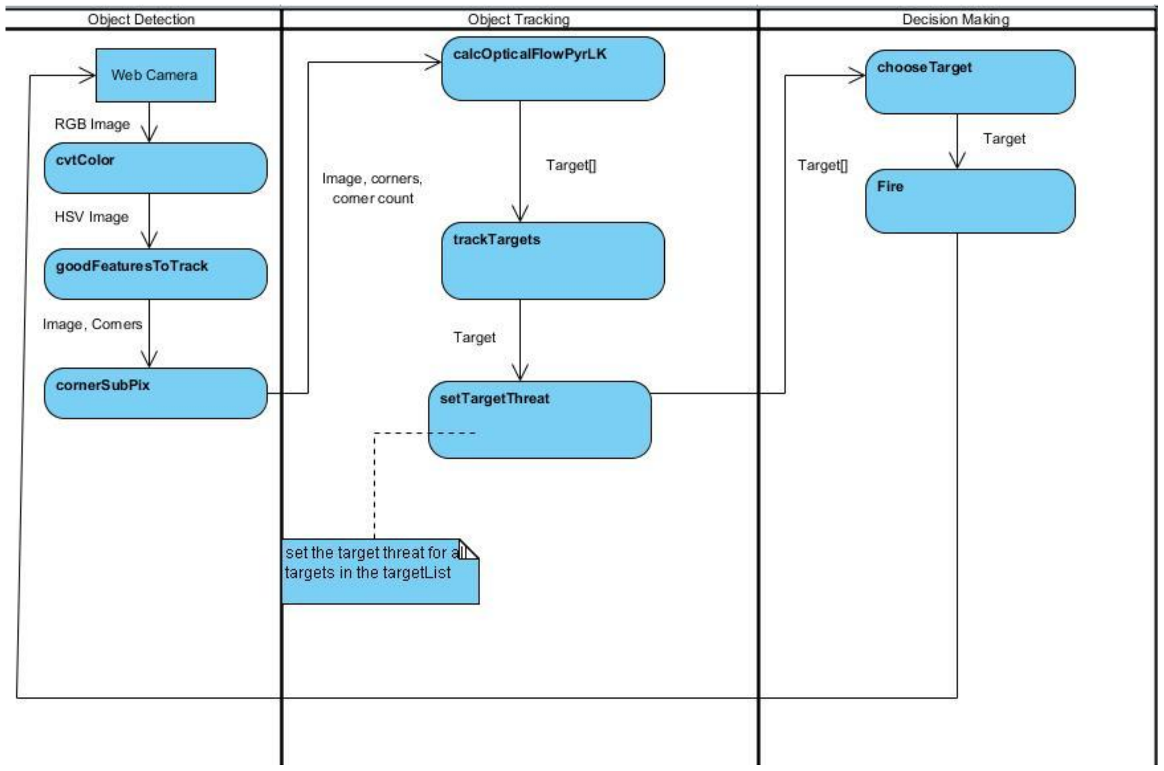


Figure 9 - Software activity diagram

The following figure 10 diagrams the process of the software development for the ATS. In each step of the process, the software developer can return to a previous section for further enhancement. The system has already been analyzed and designed so the next step is to write code for what has been designed. Once the code is written, initial testing can begin. Depending on how the testing goes, the next iteration can go back to analysis, design, or coding. If all of the tests that were run were successful, then coding can be continued. However, if something went wrong during the testing, the system could need redesigning or the analysis of the system could need to be altered also. The process will be iterative and constantly develop over time. Many of the phases will be occurring simultaneously. To explain, someone could be coding part of the program, while another member is testing a portion of the system. This modularity will allow for faster development and improve the ability for the system to be upgraded and maintained in the future. Once all test cases have been passed, the system still needs to be maintained so the process never really reaches an end. Other software development processes were considered, such as the agile method or the waterfall method, but neither of them fit well with the application of the ATS. Overall, it has been decided that this software development process provides the best solution to the specific application of the ATS.

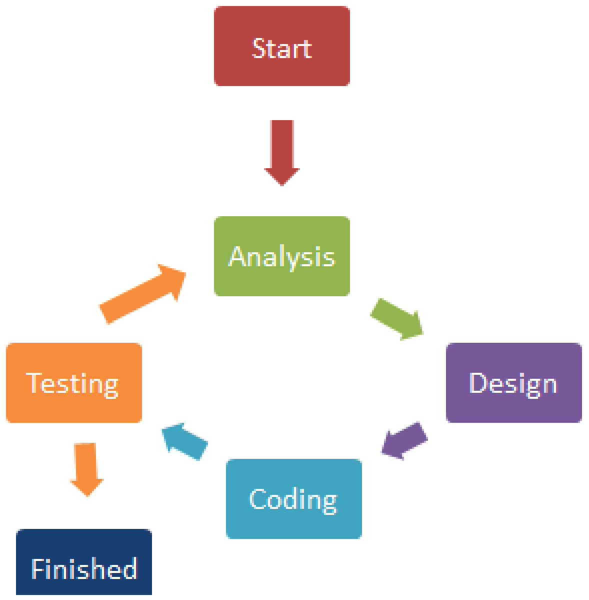


Figure 10 - Software development process

4.2.1 Object Detection Class

The Object Detection class will directly follow the video Capture class in the code, but will be the first class to be coded. This class will rely heavily on OpenCV algorithms to detect objects and various features of those objects. It will also be responsible for determining the color of each object so each target can be deemed as a threat or non-threat. To detect objects, in this specific case people will be the objects, the goodFeaturestoTrack and cornerSubPix methods will be used. The first of these two methods takes an image and finds all of the vital features of the image that stand out as objects (i.e. foreground objects). Then the cornerSubPix continues this process and refines the locations of each output from the previous method. Then the line or rectangle methods will be used to draw those features on the image so the user can see what is happening during the object detection process on the user interface. As for the color detection part of the object detection, first a cvtColor function will be used to convert the image from the standard RGB image format into an HSV image. This is helpful because it is quite simple to detect a wide range of shades of the same color using the hue with small thresholds for the saturation and value components. Once the image has been converted it will send it to the detectColor function that will be called when targets are being created to set the threat level of the target. The color green will represent non-threat targets while all other colors will remain as threats. This is because the paintball color will be green either the target is an ally (wearing green uniform) or has been hit sufficiently by the paintball gun and therefore can be deemed as a non-threat.

voidcvtColor(InputArray**src**, OutputArray**dst**, int**code**, int**dstCn**=0)

Converts an image from one color space to another.

Parameter Name	**Parameter Description**
src	Source image: 8-bit unsigned, 16-bit unsigned (CV_16UC...), or single-precision floating-point.
dst	Destination image of the same size and depth as src
code	Color space conversion code. See the description below.
dstCn	Number of channels in the destination image. If the parameter is 0, the number of the channels is derived automatically from src and code.

Target[ ] createTargets(frame **currImage**)

searches an image for targets.

Return value - a dynamic array of all targets found. If no target was found, null will be returned.

Parameter Name	**Parameter Description**
currImage	a single frame from the video stream.

voidgoodFeaturesToTrack(InputArray**image**, OutputArray**corners**, int**maxCorners**, double **qualityLevel**, double **minDistance**, InputArray**mask**=noArray(), int**blockSize**=3, bool**useHarrisDetector**=false, double **k**=0.04)

Determines strong corners on an image.

Parameter Name	**Parameter Description**
image	Input 8-bit or floating-point 32-bit, single-channel image.
corners	Output vector of detected corners.
maxCorners	Maximum number of corners to return. If there are more corners than are found, the strongest of them is returned.
qualityLevel	Parameter characterizing the minimal accepted quality of image corners. The parameter value is multiplied by the best corner quality measure, which is the minimal eigenvalue or the Harris function response. The corners with the quality measure less than the product are rejected. For example, if the best corner has the quality measure = 1500, and the qualityLevel=0.01, then all the corners with the quality measure less than 15 are rejected.
minDistance	Minimum possible Euclidean distance between the returned corners.
mask	Optional region of interest. If the image is not empty (it needs to have the type CV_8UC1 and the same size as image), it specifies the region in which the corners are detected.
blockSize	Size of an average block for computing a derivative covariation matrix over each pixel neighborhood.
useHarrisDetector	Parameter indicating whether to use a Harris detector
k	Free parameter of the Harris detector.

voidcornerSubPix(InputArray**image**, InputOutputArray**corners**, Size **winSize**, Size **zeroZone**, TermCriteria**criteria**)

Refines the corner locations.

Parameter Name	Parameter Description
image	Input image.
corners	Initial coordinates of the input corners and refined coordinates provided for output.
winSize	Half of the side length of the search window
zeroZone	Half of the size of the dead region in the middle of the search zone over which the summation in the formula below is not done. It is used sometimes to avoid possible singularities of the autocorrelation matrix. The value of (-1,-1) indicates that there is no such a size.
criteria	Criteria for termination of the iterative process of corner refinement. That is, the process of corner position refinement stops either after criteria.maxCount iterations or when the corner position moves by less than criteria.epsilon on some iteration.

voidline(Mat&**img**, Point **pt1**, Point **pt2**, constScalar&**color**, int**thickness**=1, int**lineType**=8, int**shift**=0)

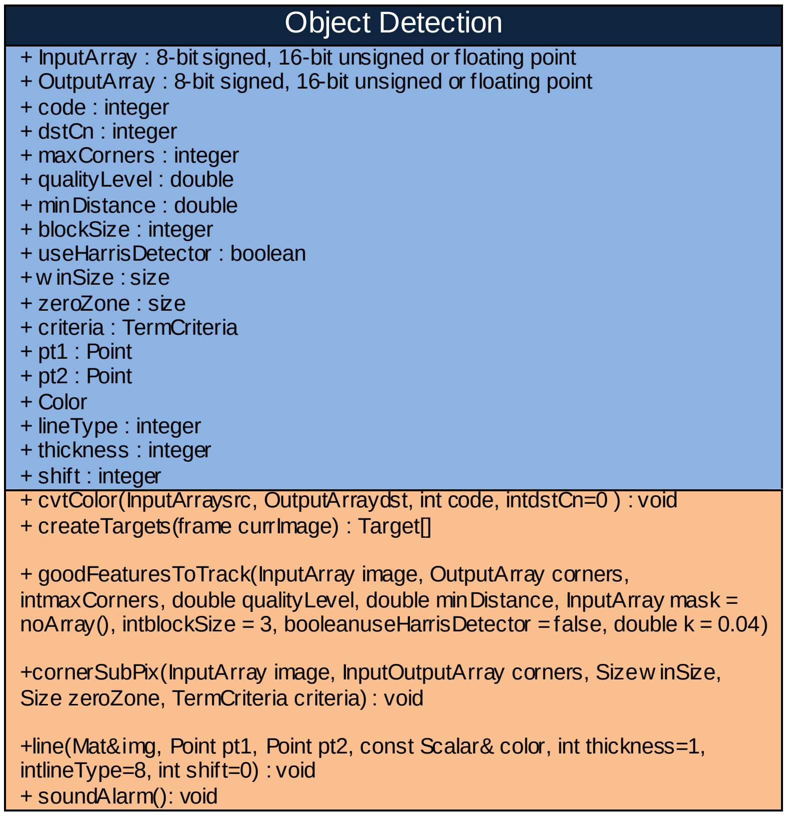
Draws a line segment connecting two points.

Parameter Name	Parameter Description
img	Image
pt1	Vertex of the rectangle
pt2	Vertex of the rectangle opposite to pt1 .
color	Rectangle color or brightness (grayscale image).
thickness	Line thickness
lineType	**8** (or omitted) - 8-connected line. **4** - 4-connected line. **CV_AA** – anti-aliased line.
shift	Number of fractional bits in the point coordinates.

voidrectangle(Mat&**img**, Point **pt1**, Point **pt2**, constScalar&**color**, int**thickness**=1, int**lineType**=8, int**shift**=0)

Draws a simple, thick, or filled up-right rectangle.

Parameter Name	Parameter Description
img	Image
pt1	Vertex of the rectangle
pt2	Vertex of the rectangle opposite to pt1 .
color	Rectangle color or brightness (grayscale image).
thickness	Thickness of lines that make up the rectangle. Negative values, like CV_FILLED, mean that the function has to draw a filled rectangle.
lineType	Type of the line
shift	Number of fractional bits in the point coordinates.


Object Detection
+ InputArray : 8-bit signed, 16-bit unsigned or floating point
+ OutputArray : 8-bit signed, 16-bit unsigned or floating point
+ code : integer
+ dstCn : integer
+ maxCorners : integer
+ qualityLevel : double
+ minDistance : double
+ blockSize : integer
+ useHarrisDetector : boolean
+ winSize : size
+ zeroZone : size
+ criteria : TermCriteria
+ pt1 : Point
+ pt2 : Point
+ Color
+ lineType : integer
+ thickness : integer
+ shift : integer
+ cvtColor(InputArraysrc, OutputArraydst, int code, intdstCn=0) : void
+ createTargets(frame currImage) : Target[]
+ goodFeaturesToTrack(InputArray image, OutputArray corners, intmaxCorners, double qualityLevel, double minDistance, InputArray mask = noArray(), intblockSize = 3, booleanuseHarrisDetector = false, double k = 0.04)
+cornerSubPix(InputArray image, InputOutputArray corners, Size winSize, Size zeroZone, TermCriteria criteria) : void
+line(Mat&img, Point pt1, Point pt2, const Scalar& color, int thickness=1, intlineType=8, int shift=0) : void
+ soundAlarm(): void
Figure 11 - Object detection class diagram

4.2.2 Motion/Object Tracking Class

The Motion/Object Tracking class will be the next class after object detection to handle the image frames. The responsibility of this class will be to take all objects detected from the object detection class and update their positions, velocities, ranges, and threat levels based on the differences found between the previous and current frames from the web camera. This task will be completed by calculating optical flow using the iterative Lucas-Kanade method with pyramids which is detailed in the object and motion tracking research sections. The other major method of the motion/object tracking class is to track the targets once they have been created. This method will be called later in the program once the targets have actually been created so it will be implemented later than the other part of the class. OpenCV functions will be used in this class to assist with the tracking. The trackTargets class will be responsible for the physical updating of all of the variables for each target in the targetList. If the targetList is null, then it will just return immediately.

voidtrackTargets(Target[] **targetList**, frame **prevImage**, frame **currImage**)

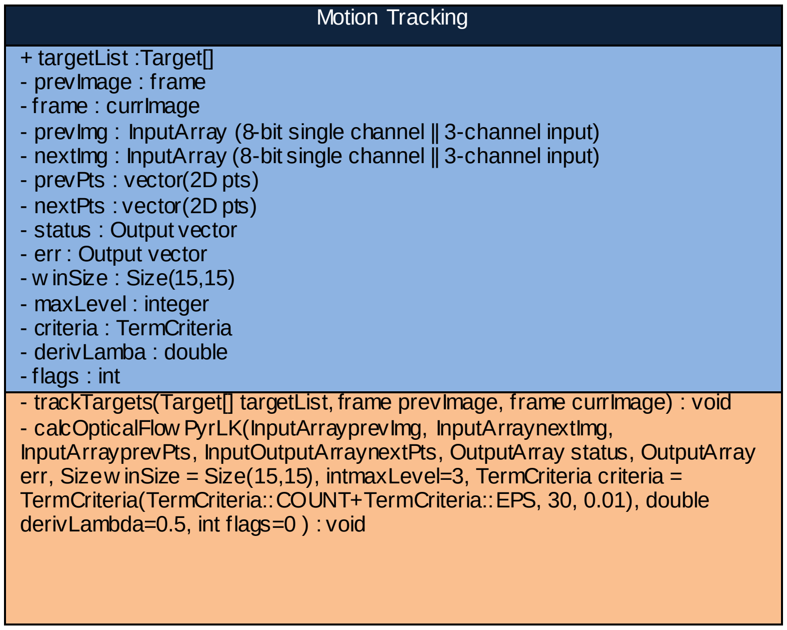
updates the positions and velocities of each target in the array.

Parameter Name	Parameter Description
targetList	the list of targets received from findTargets. If null, return.
prevImage	the last image from the video stream.
currImage	the current image from the video stream.

void calcOpticalFlowPyrLK(InputArray**prevImg**, InputArray**nextImg**, InputArray**prevPts**, InputOutputArray**nextPts**, OutputArray**status**, OutputArray**err**, Size **winSize**=Size(15,15), int**maxLevel**=3, TermCriteria**criteria**=TermCriteria(TermCriteria::COUNT+TermCriteria::EPS, 30, 0.01), double **derivLambda**=0.5, int**flags**=0)

Calculates an optical flow for a sparse feature set using the iterative Lucas-Kanade method with pyramids.

Parameter Name	**Parameter Description**
prevImg	First 8-bit single-channel or 3-channel input image.
nextImg	Second input image of the same size and the same type as prevImg.
prevPts	Vector of 2D points for which the flow needs to be found. The point coordinates must be single-precision floating-point numbers.
nextPts	Output vector of 2D points (with single-precision floating-point coordinates) containing the calculated new positions of input features in the second image. When OPTFLOW_USE_INITIAL_FLOW flag is passed, the vector must have the same size as in the input.
status	Output status vector. Each element of the vector is set to 1 if the flow for the corresponding features has been found. Otherwise, it is set to 0.
err	Output vector that contains the difference between patches around the original and moved points.
winSize	Size of the search window at each pyramid level.
maxLevel	0-based maximal pyramid level number. If set to 0, pyramids are not used (single level). If set to 1, two levels are used, and so on.
criteria	Parameter specifying the termination criteria of the iterative search algorithm (after the specified maximum number of iterations criteria.maxCount or when the search window moves by less than criteria.epsilon.
derivLambda	Not used.
flags	Operation flags: **OPTFLOW_USE_INITIAL_FLOW** Use initial estimations stored in nextPts . If the flag is not set, then prevPts is copied to nextPts and is considered as the initial estimate.



```
Motion Tracking
+ targetList :Target[]
- prevImage : frame
- frame : currImage
- prevImg : InputArray (8-bit single channel || 3-channel input)
- nextImg : InputArray (8-bit single channel || 3-channel input)
- prevPts : vector(2D pts)
- nextPts : vector(2D pts)
- status : Output vector
- err : Output vector
- winSize : Size(15,15)
- maxLevel : integer
- criteria : TermCriteria
- derivLamba : double
- flags : int
- trackTargets(Target[] targetList, frame prevImage, frame currImage) : void
- calcOpticalFlowPyrLK(InputArrayprevImg, InputArraynextImg,
InputArrayprevPts, InputOutputArraynextPts, OutputArray status, OutputArray
err, Sizewin Size = Size(15,15), intmaxLevel=3, TermCriteria criteria =
TermCriteria(TermCriteria::COUNT+TermCriteria::EPS, 30, 0.01), double
derivLambda=0.5, int flags=0 ) : void
```

Figure 12 - Motion Tracking class diagram

4.2.3 Target Class

The Target class will create targets with all of the necessary information that a target must possess to be tracked. Each target will have a position, velocity, threat level, range and an ID. The ID will be a simple counting variable to differentiate all of the targets in the system and will be incremented each time a new target is encountered. The position and velocity fields will be used to keep track of the target on a frame by frame basis so the turret can predict the future movement of the target to gain a better accuracy. They will keep track of past values of their respective variables so an average can be determined from the multiple values. The range will be used to calculate threat level (the closer the target is, the higher the threat level). The threat level will also be determined by the color detection class. If the target is wearing green it will be considered a non-threat or ally. The only methods that the target class has are access methods that get and set all of the variables that a target possesses. This will be coded before the turret class but after the basic object detection and tracking classes are in place due to the need for the information of the target before one can be created.

Variable Type	Variable Name	Variable Description
position	targetPosition	the coordinates, within the image that the target is located, that bound the target in a rectangular box.
velocity	targetVelocity	the speed and direction in which the target is moving.
int	targetThreat	a calculation of a target's threat based on its distance from the turret and the color detected from the target. Used to determine who to shoot out of multiple targets.
int	targetRange	the distance the target is away from the base of the turret.
int	targetID	an Identification number to keep track of all of the targets.

Position getPosition();

gets the current position of the specified target.

Void setPosition(position **newPosition**);

updates the position of the specified target.

Parameter Name	Parameter Description
newPosition	the new position of the specifed target.

velocitygetVelocity();

gets the current velocity of the specified target.

Void setVelocity(velocity **newVelocity**);

updates the velocity of the specified target.

Parameter Name	Parameter Description
newVelocity	the new velocity of the specified target.

IntgetThreat();

gets the threat level of the specified target.

Void setThreat(int**newThreat**);

updates the threat level of the specified target.

Parameter Name	Parameter Description
newThreat	the new threat level of the specified target.

IntgetRange();

gets the current distance away from the turret of the specified target.

Void setRange(int**newRange**);

updates the range of the specified target.

Parameter Name	Parameter Description
newRange	the new range of the specified target

IntgetID();

gets the ID of the specified target.

Void setID(int**initID**);

sets the initial ID of the target.

Parameter Name	Parameter Description
initID	the value of the ID for the target. Will be 1 greater than the last target.

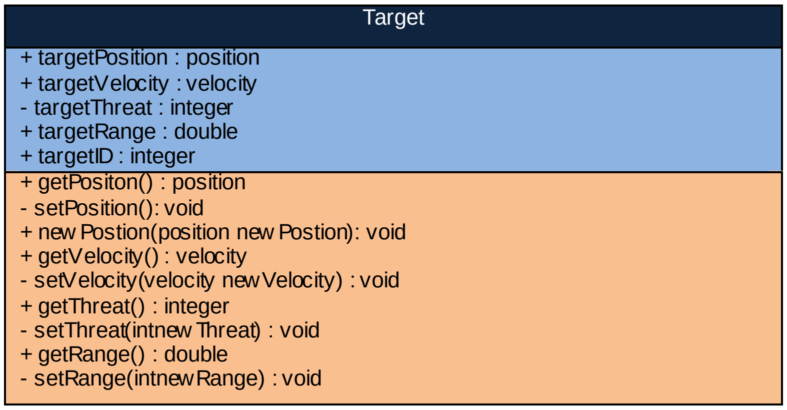

Target
+ targetPosition : position
+ targetVelocity : velocity
- targetThreat : integer
+ targetRange : double
+ targetID : integer
+ getPositon() : position
- setPosition(): void
+ new Postion(position new Postion): void
+ getVelocity() : velocity
- setVelocity(velocity newVelocity) : void
+ getThreat() : integer
- setThreat(intnew Threat) : void
+ getRange() : double
- setRange(intnewRange) : void

Figure 13 - Target class diagram

4.2.4 Turret Class

The turret class will be the main class of the software. It will instantiate the turret and initializes it with the given data upon the first execution of the program. It will then continue to keep track and update all of the vital information of the system. It will also control the firing, target selection, determine the angles needed for the servo motors, send audible alerts, and keep track of all of the targets in the field of vision. The fire function will take the given target and fire rate and attack that target, firing at the specified rate. The choose target method will make a comparison of all existing targets, and determine which one has the highest priority, returning that target. The alertWarning and alertEngaged functions will set the physical alarm and play an audio file to alert the target of its current status before attacking it. The rest of the functions are used to get and set the various variables such as the angles, ammoRemaining, fire rate, and if any targets exist at the current time. This is one of the most important classes and will be a high priority in the software of the ATS.

Variable Type	**Variable Name**	**Variable Description**
boolean	engaged	true if currently attacking a target.
boolean	targetsExist	true if there are any targets of any threat level in the field of vision.
Int	ammoRemaining	an estimate of how much ammunition is left in the weapon. Initialized to a set amount upon program start and decremented based on the fire rate selected.
Int	fireRate	the rate at which the gun should fire. 0 = automatic, 1 = burst, 2 = single shot

Void fire(Target **currTarget**, int**firerate**);

Assigns the turret to fire at the target with the highest threat.

Parameter Name	**Parameter Description**
currTarget	the target in the field of vision with the highest threat level
Firerate	the rate at which the gun should fire

Target chooseTarget(boolean**targetsExist**, Target[] **TargetList**)

Selects the target which has the highest threat level of the TargetList.

Return Value - the target with the highest threat level in the field of vision. Null if targetsExist == false.

Parameter Name	Parameter Description
targetsExist	true if there are any targets of any threat level in the field of vision.
TargetList	a dynamic array of all targets found. If no target was found null will be returned.

voidalertWarning();

plays the warning audio file if a target becomes in the tracking range of the sentry gun.

voidalertEngaging();

plays the firing audio file to alert the target it is about to be engaged.

IntgetHAngle();

returns the current angle of the horizontal servo motor.

voidsetHAngle(int**angle**)

sets the angle of the horizontal servo motor.

Parameter Name	Parameter Description
angle	the value of the horizontal angle that the servo motor needs to be set at to shoot the target.

IntgetVAngle();

returns the current angle of the vertical servo motor.

voidsetVAngle(int angle)

sets the angle of the vertical servo motor.

Parameter Name	Parameter Description
angle	the value of the vertical angle that the servo motor needs to be set at to shoot the target.

Boolean getEngaged();

returns whether or not the turret is engaging a target.

Void setEngaged(boolean**engaged**);

sets the engaged flag for the turret.

Parameter Name	Parameter Description
engaged	true if the turret is engaged in combat, false otherwise.

Boolean getTargetsExist();

returns whether or not targets are in the field of vision.

Void setTargetsExist(booleantargetsExist);

sets the turretsExist flag for the turret.

Parameter Name	**Parameter Description**
targetsExist	true if there are targets in the field of vision. False otherwise.

IntgetAmmoRemaining();

returns the estimated amount of ammunition left in the weapon.

IntgetFireRate();

returns the rate of fire in which the weapon is set to.

Void setFireRate(intfirerate);

sets the fireRate of the turret to the specified value.

Parameter Name	**Parameter Description**
Firerate	the rate at which the user wants the turret to fire at.

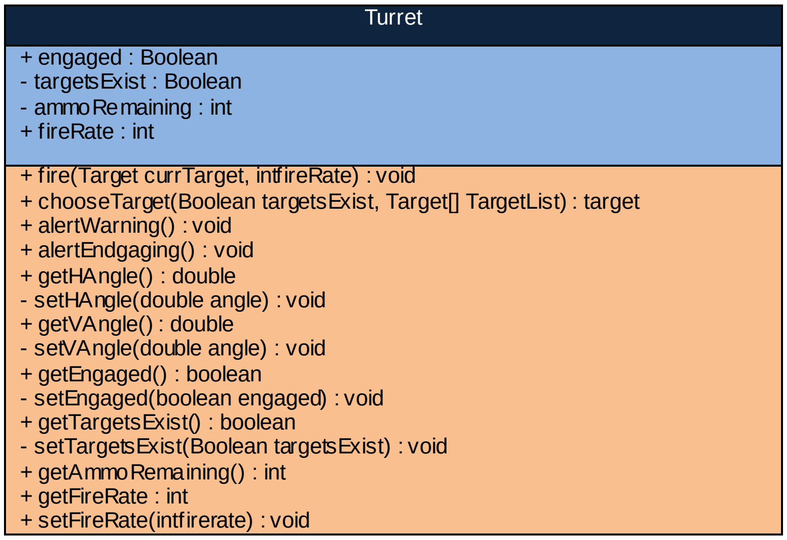

Turret
+ engaged : Boolean
- targetsExist : Boolean
- ammoRemaining : int
+ fireRate : int
+ fire(Target currTarget, intfireRate) : void
+ chooseTarget(Boolean targetsExist, Target[] TargetList) : target
+ alertWarning() : void
+ alertEndgaging() : void
+ getHAngle() : double
- setHAngle(double angle) : void
+ getVAngle() : double
- setVAngle(double angle) : void
+ getEngaged() : boolean
- setEngaged(boolean engaged) : void
+ getTargetsExist() : boolean
- setTargetsExist(Boolean targetsExist) : void
+ getAmmoRemaining() : int
+ getFireRate : int
+ setFireRate(intfirerate) : void

Figure 14 - Turret class diagram

4.2.5 Video Capture Class

The Video Capture class is the first class that will be accessed by the ATS software program. Its objective will be to take the video stream from the web camera and interface it with the program. This step is vital in the process so the frames can be operated upon to find the targets within the field of vision. The video Capture class completes this task by connecting to the device that is the web camera either with the constructor or with the open method. The frames are then taken from the stream by using the grab method or the retrieve method. The retrieve method decodes the frame as well as grabbing it. Also, the get and set functions for the property ID allow the access to a wide variety of variables of the video stream such as hue, saturation, brightness, frame rate, and other useful variables. Lastly, the release method detaches the device from the program so it can be safely disconnected. This will only be used when the program has been terminated (either by error or intentionally).

VideoCapture::VideoCapture(int**device**)

VideoCapture::VideoCapture(conststring&**filename**)

VideoCapture::VideoCapture()

VideoCapture constructors.

Parameter Name	**Parameter Description**
Filename	name of the opened video file
Device	id of the opened video capturing device (i.e. a camera index). If there is a single camera connected, just pass 0.

boolVideoCapture::open(int**device**)

Open video file or a capturing device for video capturing

boolVideoCapture::isOpened()

Returns true if video capturing has been initialized already.

voidVideoCapture::release()

Closes video file or capturing device.

boolVideoCapture::grab()

Grabs the next frame from video file or capturing device. Returns true if grab was success.

boolVideoCapture::retrieve(Mat&**image**, int**channel**=0)

Decodes and returns the grabbed video frame. Returns true if success, or false and a null pointer if unsucessful.

Parameter Name	**Parameter Description**
Image	The image being captured.
channel	The channel of the stream that the image is coming from. (set it to zero if only one channel).

boolVideoCapture::read(Mat&**image**)

Grabs, decodes and returns the next video frame. Returns true if success, false otherwise.

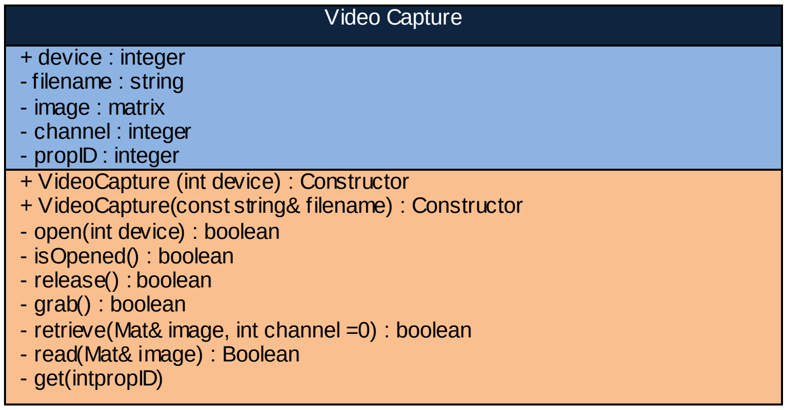
doubleVideoCapture::get(int**propId**)

Returns the specified VideoCapture property

Property Identifier	Description
CV_CAP_PROP_POS_MSEC	Current position of the video file in milliseconds or video capture timestamp.
CV_CAP_PROP_POS_FRAMES	0-based index of the frame to be decoded/captured next.
CV_CAP_PROP_POS_AVI_RATIO	Relative position of the video file: 0 - start of the film, 1 - end of the film.
CV_CAP_PROP_FRAME_WIDTH	Width of the frames in the video stream.
CV_CAP_PROP_FRAME_HEIGHT	Height of the frames in the video stream.
CV_CAP_PROP_FPS	Frame rate.
CV_CAP_PROP_FOURCC	4-character code of codec.
CV_CAP_PROP_FRAME_COUNT	Number of frames in the video file.
CV_CAP_PROP_FORMAT	Format of the Mat objects returned by retrieve() .
CV_CAP_PROP_MODE	Backend-specific value indicating the current capture mode.
CV_CAP_PROP_BRIGHTNESS	Brightness of the image (camera only)
CV_CAP_PROP_CONTRAST	Contrast of the image (camera only).
CV_CAP_PROP_SATURATION	Saturation of the image (camera only).
CV_CAP_PROP_HUE	Hue of the image (camera only).
CV_CAP_PROP_GAIN	Gain of the image (camera only).
CV_CAP_PROP_EXPOSURE	Exposure (camera only).
CV_CAP_PROP_CONVERT_RGB	Boolean flags indicating whether images should be converted to RGB.
CV_CAP_PROP_WHITE_BALANCE	Currently not supported
CV_CAP_PROP_RECTIFICATION	Rectification flag for stereo cameras (note: only supported by DC1394 v 2.x backend currently)

VideoCapture::set(int**propertyId**, double **value**)

Parameter Name	Parameter Description
propertyId	see Table #
Value	The value that the property id should be set to based on the specified values for each property.


Video Capture
+ device : integer
- filename : string
- image : matrix
- channel : integer
- propID : integer
+ VideoCapture (int device) : Constructor
+ VideoCapture(const string& filename) : Constructor
- open(int device) : boolean
- isOpened() : boolean
- release() : boolean
- grab() : boolean
- retrieve(Mat& image, int channel =0) : boolean
- read(Mat& image) : Boolean
- get(intpropID)
Figure 15 - Video Capture class diagram

4.2.6 Video Write Class

The Video Writer class is similar to the Video Capture class in multiple ways. It operates the same way opening and closing the device or file and writes/receives frames in the same ways. The Video Writer class will write frames to a file so it can be viewed later by the system user to see what occurred in the ATS' field of vision. This class will be used when the turret is engaged to a target so the user can view all engagements on a file by file basis. The file will be opened upon the initial firing of the target and then closed once the weapon ceases its fire. The file is created either by the constructor or the open method and the isOpenedmethod is to check if already has been opened. The write method will write a single frame to the video file and will be called on each loop through the program. This will be one of the last classes to be implemented in the software.

VideoWriter::VideoWriter()

VideoWriter::VideoWriter(conststring&**filename**, int**fourcc**, double **fps**, Size **frameSize**, bool**isColor**=true)

VideoWriter constructors (first is an empty constructor and the second is when it is known what type of video is going to be written)

boolVideoWriter::open(conststring&**filename**, int**fourcc**, double **fps**, Size **frameSize**, bool**isColor**=true)

Initializes or reinitializes video writer.

Parameter Name	Parameter Description
filename	Name of the output video file
fourcc	4-character code of codec used to compress the frames. For example, CV_FOURCC('P','I','M,'1') is a MPEG-1 codec, CV_FOURCC('M','J','P','G') is a motion-jpeg codec etc.
fps	Framerate of the created video stream.
frameSize	Size of the video frames.
isColor	If it is not zero, the encoder will expect and encode color frames, otherwise it will work with grayscale frames (the flag is currently supported on Windows only).

boolVideoWriter::isOpened()

Returns true if video writer has been successfully initialized.

voidVideoWriter::write(constMat&**image**)

Writes the next video frame

Parameter Name	Parameter Description
writer	Video writer structure (OpenCV 1.x API)
image	The written frame

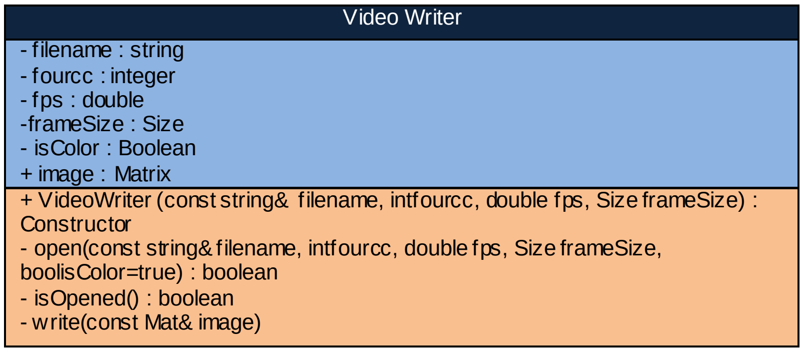

Video Writer
- filename : string
- fourcc : integer
- fps : double
-frameSize : Size
- isColor : Boolean
+ image : Matrix
+ VideoWriter (const string& filename, intfourcc, double fps, Size frameSize) : Constructor
- open(const string& filename, intfourcc, double fps, Size frameSize, boolisColor=true) : boolean
- isOpened() : boolean
- write(const Mat& image)

Figure 16 - Video Writer class diagram

4.2.7 Manual Control Class

The manual control class will function as a built in user interface on the turret controls computer. The Manual Control Class will be implemented using a Windows Form Application designed in Microsoft Visual Studios 2010. The application will provide a friendly user interface with buttons to rotate the gun, fire the gun, and sound the alarm. The application will also include the video feed from the web camera, along with an emergency stop button and list to select the rate of fire for ATS.

Several methods will be implemented in the Manual Control class, there are methods to rotate the turret right or left as well as up or down this will be achieved by adding buttons to the application as well as mapping the controls to the arrow keys. The Fire() method will be used to fire the gun, this will be implemented by another button on the application as well as the spacebar key on the computers keyboard. Buttons for playing audible files will be added to the alertWarning() and alertEngaged() methods and lastly a kill switch button will be implemented to bring the turret offline if need be.

Return Type	**Method Name**	**Method Description**
Double	getXpositon()	Returns the x coordinate the servo is currently set to.
Double	getYPosition()	Returns the y coordinate the vertical servo is set to.
Void	rotateRight()	Rotates the turret right by a set degree.
Void	rotateLeft()	Rotates the turret left by a set degree.
Void	panUP()	Tilts the turret upwards.
Void	panDown()	Tilts the turret downwards.
Void	Fire()	Fires the turret.
Void	alertWarning()	Plays an audio file to warn intruders that they are entering a dangerous zone and that they should proceed to leave.
Void	alertEngaged()	Plays an audio file to warn intruders that they are about to be fired upon by ATS.
Void	setFireRate(intfir erate)	sets the firing rate of the turret 0 = automatic, 1 = burst, 2 = single shot.

Variable Type	Variable Name	Variable Description
Double	xPosition	The x coordinate the horizontal servo is currently set to.
Double	yPosition	The y coordinate the vertical servo is currently set to.
Integer	fireRate	Controls the rate of fire of ATS.

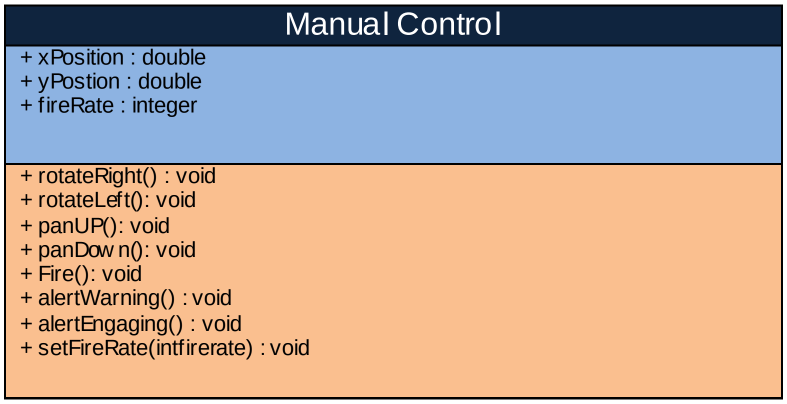


Figure 17 - Manual control class diagram

Figure 18 shows an early prototype for the manual control class. The Windows Application form was made in Visual Studios 2010 and contains buttons and controls to fire and move the gun, alert and warn intruders, an emergency stop and a rate of fire selection window. The blank area on the top of the application is reserved for the video feed; this feature is to be implemented still. This application is designed with future improvements in mind including visual improvements and detailed readouts including the coordinates the gun is currently aiming at, number of targets in the field of view, weather and wind conditions, threat list, and target highlighter which would illuminate targets in the embedded video.

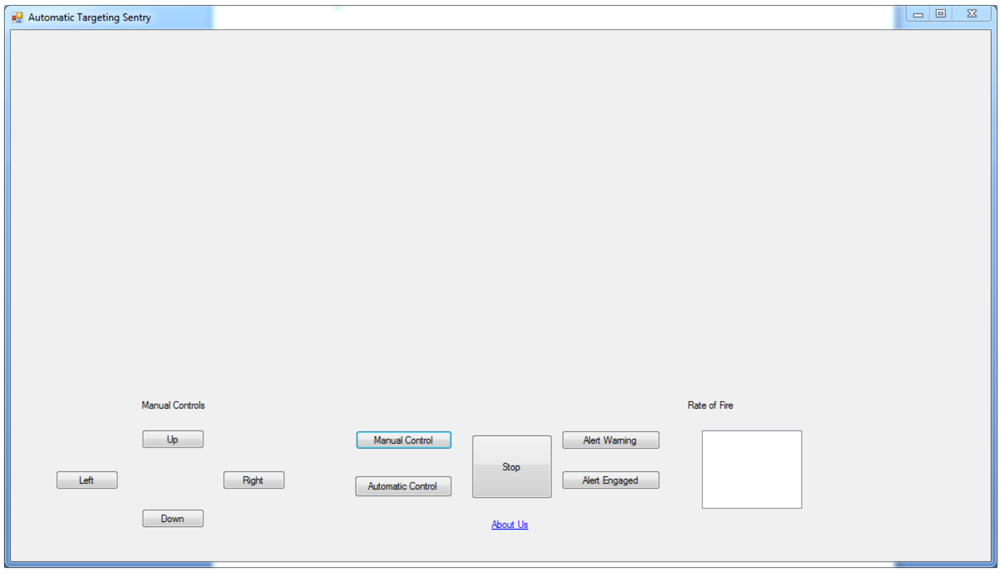


Figure 18 - Initial layout for manual control

4.2.8 Facial Detection and Recognition class

The facial detection and recognition class will be used as a security system to access manual control. This is a necessary step to securing the system from any threats that manage to reach the base of the turret. To ensure the turret remains under the teams control the system will require that the user undergoes a facial scan and confirmation.

When accessing the manual control class the user will first be asked to position his or her face in front of the on board computers web camera. The system will then scan the image in an attempt to detect a face using various Haar-like features. Once the class acknowledges the face it will attempt to find a matching face in the face database the team creates. The face database is nothing more than a collection of images of the teams faces. To grant additional users access to the manual control class an image of the person must be entered into the facial database. This class is an important security system that safeguards the turret from malicious users.

Void **detect_and_draw(**IplImage* image) – draws an object from an image.

cvMemStorage – Creates memory for calculations. Memory is automatically released when there are no objects referring to it.

cvHaarClassifierCascade – Create a new Haar Classifier. Haar classifiers are used to detect human facial features such as the dark area under the eyes.

HaarClassifierCascade**cvLoad**(cascade_name, 0, 0, 0) – Load the HaarClassifier Cascade. The cascade works by using a sequence of haar-classifiers to eliminate entries that are not faces quickly and to ensure accuracy.

The cascade starts with simple classifiers in order to eliminate false entries quickly but then images must pass tougher tests to be considered a face. This saves processing time and power.

cvShowImage(window, image) – shows the specified image in the window.

IplImage[] **detectFaces** (IplImage *image) – detects faces using Haar Features. Returns an array of images that are considered to be faces.

Void rectangle(Mat&image, Point pt1, Point pt2, constScalar&color, int thickness=1, intlineType=8, intshift=0) – draws a rectangle, the rectangle will be drawn around objects considered to be faces.

IplImage**matchFaces**(IplImage *image) – returns a matching face from our face database.

cvResize -used to resize the input image

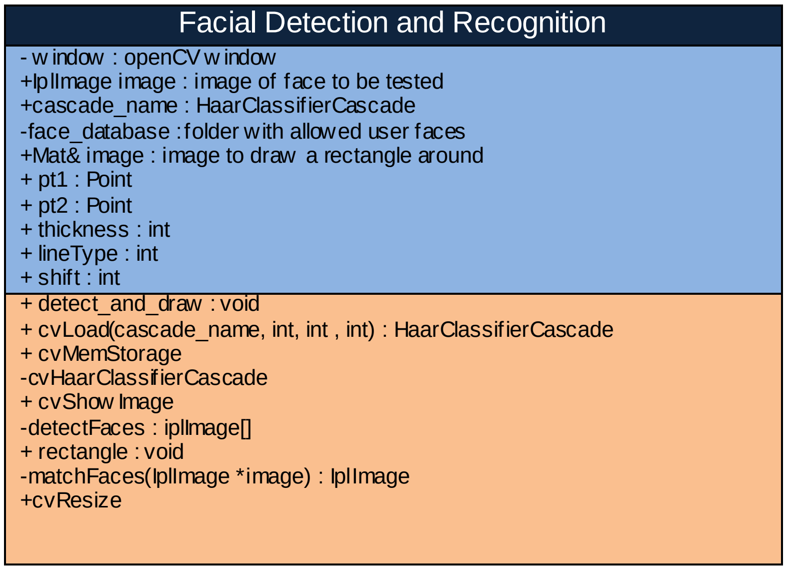

Facial Detection and Recognition
- window : openCV window
+IplImage image : image of face to be tested
+cascade_name : HaarClassifierCascade
-face_database : folder with allowed user faces
+Mat& image : image to draw a rectangle around
+ pt1 : Point
+ pt2 : Point
+ thickness : int
+ lineType : int
+ shift : int
+ detect_and_draw : void
+ cvLoad(cascade_name, int, int , int) : HaarClassifierCascade
+ cvMemStorage
-cvHaarClassifierCascade
+ cvShow Image
-detectFaces : iplImage[]
+ rectangle : void
-matchFaces(IplImage *image) : IplImage
+cvResize

Figure 19 - Facial detection and recognition class diagram

4.2.9 Data Structures

Position – A position object will consist of 5 integer values. These 4 values will represent the 4 corners of the rectangle that enclose the target. These values will be used in keeping track of where targets are, as well as, informing the sentry gun on where to shoot if the target is in the firing zone. The 5th integer value will represent the center of rectangle, which is the specific spot where the turret will be aiming for. This center value will be updated every frame from the other 4 values. Every frame that goes through the system will check for the targets, then update the position if the target is found to still be in the frame and if not the position's will all be set to -1.

Velocity – A velocity object will contain 2 components, speed and direction. The first component will be a floating-point value that represents the speed in which the target is moving. This will be calculated from the change in position over the course of 2 frames where the target has been spotted. The 2nd component will represent the direction in which the image is moving. The direction will be represented by an integer, from 1 to 360, which represent what direction the image is moving. [Example: If the direction was equal to 180 the target would be moving from right to left within the field of vision.

Turret – There will only be 1 turret object in the program so it will not need an Identification number to represent it, like targets do. It will have boolean flags to keep track of whether or not it is engaged to a target and whether or not there are any targets in the field of vision. It will also have a counter to keep a rough estimate of how much ammunition is left in the weapon. The last component it has is an integer value that represents the fire rate of the weapon. The options are 0 = automatic, 1 = burst, 2 = single shot, and these are only used during the manual control of the system. If the system is acting in an automated setting, then the fire rate will be 0, or automatic. Once a target is no longer able to be located, it will be deleted from the system.

Target – A target is any object in the camera's field of vision that is foreign to the view and passes a certain threshold. For example, trees that have been in the field of vision the entire time won't be considered targets and neither will very small moving objects (for example, squirrels). Since there can, and most likely will be, multiple targets each target needs its own identification number. The ID will be continually incremented, starting from zero, for each target that gets detected. Target's will also have position and velocity objects, which are detailed above. Lastly, each target will have two more integer values representing the range and threat. The range of the target will be how far it is located from the turret. The threat value of the target depends on its range and color. As a target gets closer to the turret, its threat level will increase. If the target is wearing certain colors, it will have a threat level of zero, no matter what. If the colors are anything else, then the threat level will be based on solely the range of the target.

4.2.10 Constants

The following list are constants that will be defined and used throughout the ATS software program.

- **UP** – This represents a 0 and when passed from the manual control class to the turret class, the turret will move up a set amount.
- **DOWN** - This represents a 1 and when passed from the manual control class to the turret class, the turret will move up a set amount.
- **LEFT** - This represents a 2 and when passed from the manual control class to the turret class, the turret will move up a set amount.

- **RIGHT** - This represents a 3 and when passed from the manual control class to the turret class, the turret will move up a set amount.

- **AUTO** – This represents a 0 and when passed to the setFireRate function, it will set the fire rate to automatic.

- **BURST** - This represents a 1 and when passed to the setFireRate function, it will set the fire rate to automatic.

- **SINGLE** - This represents a 2 and when passed to the setFireRate function, it will set the fire rate to automatic.

- **HUE_MAX** – This represents the upper bound for the hue of the color that represents an allied target. The value has yet to be determined.

- **HUE_MIN -** This represents the lower bound for the hue of the color that represents an allied target. The value has yet to be determined.

- **MANUAL** – This value represents a 0, and, when passed to the turret, it will set the manual control on.

- **AUTONOMOUS** - This value represents a 1 and, when passed to the turret, it will set the manual control off.

- **WARNING_DIST** – This value represents the distance at which targets will be warned of the ATS' intentions. The value is yet to be determined.

- **FIRING_DIST** – This value represents the distance at which the turret will begin firing on enemy targets.

- **CEASE_FIRE_DIST** – This value represents the distance a target that has been engaged by the turret must reach before being disengaged as an enemy.

4.2.11 Software Components

Since the system software is a majority of the prototype, the system will require a variety of programming languages and software environments. The openCV classes will be developed in visual studio 2010 using C++. The PCB will be designed using EagleCAD. The microcontroller IDE that the designers will use is MPLAB.

4.2.11.1 C++

The developers will use the C++ language. OpenCV is written in C++ therefore it will be the language of choice for our project. The language is object oriented and since most of the group is familiar with object oriented concepts and the C language C++ is a good fit.

4.2.11.2 Visual Studios 2010

Visual Studios is the most complete IDE available to developers working with the C++ language. It supports multiple languages and makes creating the Windows application for our on board computer a simplified task. There are extensive tutorials on adding the OpenCV libraries to a project and through our research it is the preferred IDE when working with OpenCV. The IDE contains IntelliSense which is an excellent syntax highlighting tool, and a debugging tool which debugs your code as you write. Visual Studios was provided via Microsoft's DreamSpark which provides software to students.**[55]**

4.2.11.3 EagleCAD

EagleCAD will be used to design our printed circuit board. EagleCAD's website provides training with tutorials, videos, detailed manuals, and forums. EagleCAD will provide everything needed for designing and setting up our board and laying the traces. **[56]**

4.2.11.4 Microcontroller IDE software MPLAB

The microcontroller will require embedded software for implementing simple instructions for the microcontroller to perform. Microchip Technology Inc. actually provides free design support which includes MPLAB integrated Development Environment (IDE) and a free C compiler. Along with the free software, technical documentation and 24/7 technical support and training are available. The MPLAB IDE is a free, integrated toolset available for the development of the microcontroller. It includes free software components for application development and a full featured debugger. Also available, depending on the designer of the system, MPLAB also provides a free C Compiler that may suite the programmer more depending on their preferences. This compiler is a highly optimized compiler for the higher performance microcontrollers. **[57]**

4.2.11.5 TortoiseSVN

TortoiseSVN is a popular subversion software for Windows. This subversion software allows the development team to upload their latest version of the source code and share instantly with other members working with the same code. TortoiseSvn can be integrated into Microsoft Visual Studios using a third party plugin such as VisualSVN. **[58]**

5.0 Design Summary

The hardware design summary will contain the selection and specifications of servo motors, microcontrollers, web camera, laser pointer and alarm systems. The software design summary will discuss the various classes including methods and parameters used throughout the system.

5.1 Hardware Design Summary

ATS' hardware design consists of only a few components that will work in together to construct a successfully working turret. The main components are the turret itself, which will be a paintball gun firing paintballs at incoming targets. The targets will be located through a web camera attached to the paintball gun and linked to an onboard laptop. This will be a real time video feed and display on the monitor, from there the software will detect the incoming targets.

Once the software successfully detects the incoming targets, the paintball gun will aim at their present location and track their movement. To do so, the system will consists of three servo motors, one for lateral movement, one for vertical movement and one for the trigger pull. The software will track the incoming target and relay the positioning to the microcontroller on the printed circuit board. The microcontroller will do the simple calculations for determining where the gun itself needs to point and passes the signal to its respective servo motor.

The turret will start tracking at 100 feet, when a target comes within 75 feet; an alarm will sound warning the target that they are entering a dangerous area. There will be two types of alarms on the system on board, a continuous sounding alarm and also a speaker system that will play an audio file that will verbally warn the incoming target of their increasing risks. The continuous sounding alarm will have be connected directly to the printed circuit board, so when the software identifies the target within 75 feet, the signal will be sent from the microcontroller triggering the alarm. Also when this happens, the speaker system, which is connected directly to the laptop, will start its verbal warnings in case the incoming target is human and can understand.

Like all other electrical components, that ATS system will have a battery powering the turret. The power source will be directly connected to the printed circuit board. From this connection, the main concerns are to power the microcontroller and the servo motors specifically. A 12 volt battery will be sufficient enough to power these components and will be kept near the base of the structure. The only other component that will need a power source will be the laptop. However; the laptop will not need to be tied to this battery instead it will rely on its own battery pack. The only remaining hardware of the ATS system will be the base which will be a light weight and easily portable.

The architecture of the stand will be divided up into three sections. The top section, the firing section, is where the gun will be stored with the loaded ammunition. The middle section will consist of a rotating plate connected to the yaw servo motor and the bottom base will act like a storage section. In the bottom section, there will be room to store the spare ammunition, battery, onboard laptop and all other electrical components such as the PCB. This is with the intention of making the system base bottom heavy to reinforce stability.

5.2 Software Design Summary

ATS' software design consists of a computer vision application with some extra features to enhance the quality. The process of computer vision will be completed by making a series of operations upon each frame received from the input, the web camera. First the web camera will be interfaced to the application by the Video Capture Class. Then the frame received from that class will be sent to the object detection class. The object detection class will then convert the RGB frame into a HSV frame to make color detection easier. It will then search the frame for any and all targets, and return an array of the targets (which will be null if no targets are found). Then each target gets assigned a threat level based on its range from the turret and color. The color will be used to determine allied targets from unknown targets, where any target wearing a certain green color will be deemed no threat.

Once the targets have been detected by the system, the application will need to keep track of their locations. It will accomplish this by comparing two frames and calculating the optical flow of the video stream using the iterative Lucas-Kanade method with pyramids. After the motion has been detected, each target will then have their position and velocity updated, accordingly. To make sure the program knows which target is which, they will be assigned ID numbers to differentiate them from one another in the system. The targets range will also be recorded in order to determine its importance as a target. This value will be used to choose which individual target should be fired upon, amongst a group of targets.

At this point, the turret now knows where all targets are and now must select what target, if any, to shoot. It searches the targetList for the highest threat level, and if it passes the threshold for a dangerous enemy, then it will set the location the turret needs to aim at and give the command to fire. The program will continuously check all threat levels, and update targets, as the turret is engaged to a target so if the priority changes, the turret will disengage its current target for the more threatening one in the field of vision. The turret will also keep an estimate of how much ammunition is left in the weapon by keeping a counter. Once the counter reaches zero, the turret will play an alert to inform the owners that the system is out of ammunition.

An additional feature of an alarm will be added to the software. There are two distances in which a target will be alerted by a sound file played, asynchronously, by the program. The first is the warning distance, in which targets will be tracked, and the second is the distance in which the target will be engaged. The distance at which the turret will disengage the target will be greater than the firing distance but less than the warning distance, to prevent a back and forth action of engaging and disengaging the target. The video writer class is another additional software feature. It will write the video history to an image, which can be opened later for viewing what occurred within the system during the time the turret was engaged with a target.

OpenCV's computer vision methods and classes will be used throughout the software application to enhance the feature set and quality of the ATS' software. OpenCV has a wide variety of ways to complete the process of computer vision, and the algorithms that will be implemented within this system are some of the most efficient and effective in the aspects of object detection, motion tracking, and machine learning.

6.0 Prototyping

The prototyping process will be followed by figure 20**,** show below. The program has already been researched and designed so the next step is to begin building the prototype. An evolutionary approach will be used to make the ATS' software component. The first iteration will consist of a program that can detect objects. Once that is complete, the object tracking will be implemented, but for manual control first. After another iteration, the autonomous tracking will be added. In the last iteration, the extra features, color detection and facial recognition, will be added to the software. Once each iteration of building/refining finishes, the system will be tested to see how well it functions. If successful, the next step in the evolutionary process will begin. However, if the current iteration does not work properly, then it must be fixed before moving on to the next iteration in the process.

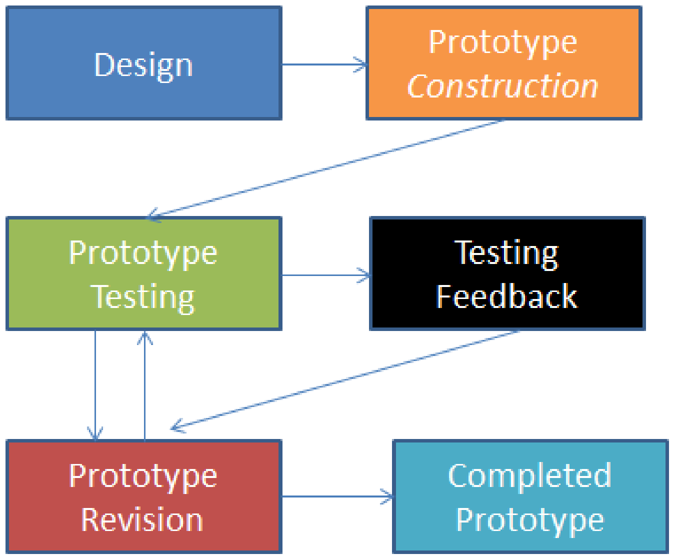


Figure 20 - Prototyping process

The prototyping process will be organized into modules for both hardware and software. Each module will be built to run individually and then will be integrated into the overall system once completed. The hardware modules will consist of the capacitive touch controller, the base and weapon system, the servo motors, microcontrollers, and the PCB. On the other hand, the software modules will be based on the classes that are defined in the software design. Object detection, object tracking, turret, color detection, and facial recognition will be the five major modules of the ATS' software prototype.

Each module will be further broken down into sub-modules that are a small part of the whole module. For the software of the ATS system, these sub-modules will be the individual functions and methods used to complete the overall computer vision application. For example, for the color detection module, there will be two sub modules. The first sub-module will be converting the image from an RGB image to an HSV image to make color detection easier. Then, the second module will actually determine what the color of the object is by checking a certain area of pixels of the object and comparing them with the threshold value, specified in the constants section. For the hardware, the sub-modules will be broken down into specific parts, where applicable, of the modules. For example, the base and weapon system module will be broken down into a sub-module for the base and then sub-modules for each component of the weapon, which consists of the gun, ammunition, co2 tank, hopper, barrel, and trigger.

Once all of the sub-modules for a given module are completed, the module can then move to testing to see how it functions. The prototype of the specific module will be tested based on its own merits to determine if it completes all required tasks that were detailed for that given module. If all of the modules tests that are listed within the testing section are successful, it can then be integrated into the overall system. After all modules have been completed, more integration testing must be done to see how each module interacts with the rest of the ATS system.

In Figure number 21 below a proto-typing model is laid out. The group will follow this process model with the customer being the group. The initial requirements are the design specifications and objectives. Once a component or system is designed a prototype will be implemented following by an evaluation. If the group is satisfied with the results the team will move to development and testing. The last step is to properly maintain the product.

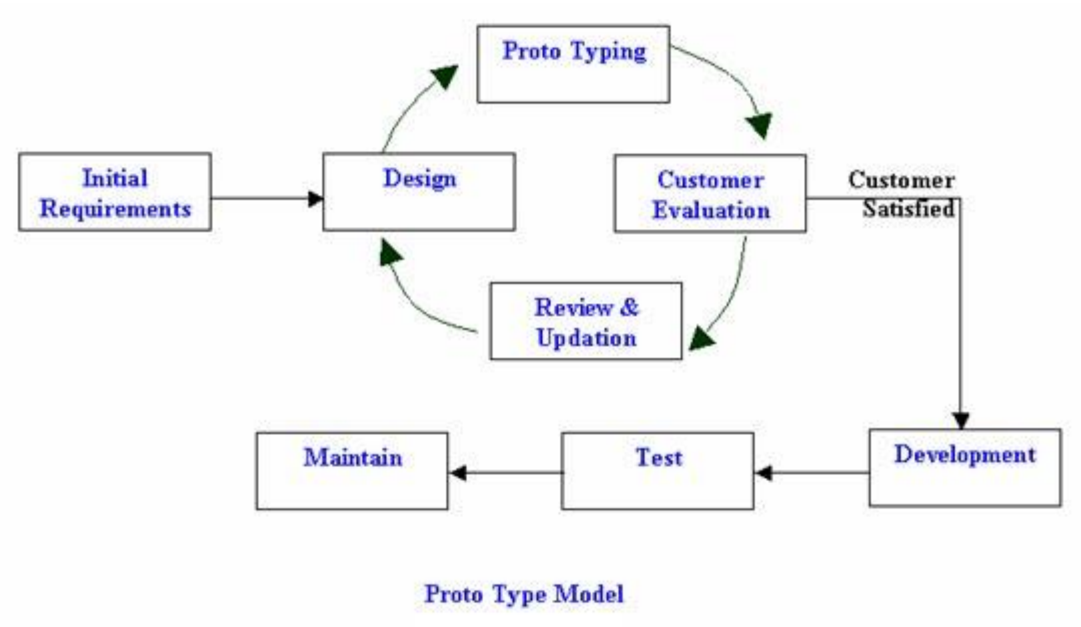


Figure 21 - Proto Type Model

7.0 Testing

The ATS system will undergo extensive testing on both hardware and software components. Hardware components will be tested individually prior to implementation. After each component passes with compliance, they will be integrated within the system. The software classes will be tested individually. Once both software and hardware are fully tested, they will be combined and the entire system will undergo overall system testing.

7.1 Hardware Testing

7.1.1 Servo Motors

Before attaching the servo motors to the base, the accuracy of each servo motor must be measured. Since the autonomous turret will only rotate while remaining in the same location, errors in direction become amplified at further distances. An assumed target is located thirty degrees counter clockwise from the center ten feet away. In other words the target is located five feet from the center. If instead error was introduced and the servo motor was pointed thirty-one degrees counter clockwise from the center, the turret will fire 5.150 feet from the center as opposed to the previous 5.000 feet resulting in an error of 0.150 feet. Table 13 represents the resulting error at ten, twenty, fifty, and one-hundred feet at one, three, and five degrees.

Error/ Distance	**1.000**	**3.000**	**5.000**	**Degrees**
10	0.150	0.446	0.736	ft.
20	0.301	0.892	1.471	ft.
50	0.752	2.231	3.678	ft.
100	1.503	4.462	7.355	ft.

Table 13 - Error percentages

It should be noted that error percentages were not calculated at various distances since the error percentage remains the same. Instead, the errors were left in differential feet to give a more tangible result. As seen in the table above, the maximum error, assuming a max error of five degrees, results in 7.355 feet. Assuming the turret it attempting to fire at about human width, approximately two feet, the turret has a chance to miss at fifty and one-hundred feet, once again, assuming a max error of five degrees.

Error in the servo motor degrees will come from two primary sources, an incorrect pulse width modulation from the servo motor microcontroller and the inaccuracy of the servo motors themselves. The inaccuracy from the servo motors themselves will most likely not be able to be perfectly fixed. Instead, these inaccuracies must be accounted for on the servo motor controller if possible. The ideal servo motor testing procedure can be found in the proceeding statements. The servo motors will have its operating voltage set to the planned operating voltage, currently five volts. The signal line will receive a four volt square wave from an oscilloscope at various pulse widths. Two sets of errors must be measured for this circumstance. Small shifts in the pulse widths must be calculated to see if the servo motors are responsive at turning a distance of less than five degrees. Large shifts in the pulse widths must also be calculated to see if the servo motors are accurate when moving from one side to the other, an assumed differential of thirty degrees or more. If there are large errors, special circumstances must be programmed into the servo motor microcontroller.

Since the servo motors operate at fifty hertz, a 20,000 microsecond period, with pulse width ranging from 900 to 2100 microseconds, there is a duty cycle ranging from 4.5% to 10.5%. In other words, for the 95.5% to 89.5% of the period that is not taken up by the pulse width signal will be taken up by the dead band. During this dead band, the microcontroller controlling the servo motors cannot go high to maintain its fifty hertz frequency. Theoretically the servo motors will only be able to be updated every 20 milliseconds. Although 20 milliseconds appear small, the servo motor will not be able to keep with rapid short back and forth movements.

For testing, the servo motor dead band should be measured in seconds as best as possible before connecting the servo motors to a load or any microcontroller. It is expected that any analog servo motors running at fifty hertz will be able to turn once every 20 milliseconds. The servo motor dead band will be again being tested with the servo microcontroller attached but still without a load. Finally the dead band will be tested with both the servo microcontroller and a load weight. As previously mentioned, analog servo motors are expected to be able to turn every fifty hertz, however it is likely that a higher dead band may be caused by the component lag from within the servo motors.

Once the above is tested, the accuracy from the previous calculated equation to predict the required pulse width modulation for a required degree position must be tested. This is required since the equation was not specified from the supplier. For a standard ninety degree rotation servo the equation was calculated to be pulse width = 900+40/3*A, where A is the required angle. For the 140 degree rotation servo the equation was calculated to be pulse width = 900+60/7*A, where A is the required angle. Each servo motor will be tested three times at divisions of fifteen degrees. The three tested will be averaged together. Table 14 below will be filled after the testing is completed for both the oscilloscope and the microcontroller.

Summary of theory and tested pulse widths							
Degree	**HS-81**		**HS-8055**		**HS-8155**		
	Theory	**Tested**	**Theory**	**Tested**	**Theory**	**Tested**	
0	900		900		900		µS
15	1100		1100		1028		µS
30	1300		1300		1157		µS
45	1500		1500		1285		µS
60	1700		1700		1414		µS
75	1900		1900		1542		µS
90	2100		2100		1671		µS
105	NA		NA		1800		µS
120	NA		NA		1928		µS
135	NA		NA		2057		µS

Table 14 - Angle testing v theory

The other source of degrees will result from the microcontroller sending an incorrect pulse width. This error theoretically should be easy to solve by having tighter timings on the microcontroller. As seen in the table 14, every inaccuracy in microseconds will result in the incorrect position. This will result in larger errors at further distances.

Once these errors are accounted for, the servo motors can be attached to the base. The above procedures must be repeated to see if there is a large error when attaching the base. Since there will be added torque, slight inaccuracies when attaching the servo motors to the base will be expected.

Once the servo motor is attached to the base, the servo motor speed must also be measured in order to determine if the speed is reasonable. The speed specifications provided by the manufacturer are with no load attached. If the turning speed of each servo is low, the speed can be slightly increased by increasing the operating voltage.

7.1.2 Microcontroller

Testing is always a critical step in designing a working prototype. By successfully testing the individual components of the ATS system, the designer can demonstrate reliability and confidence when constructing the prototype to properly achieve its specific purpose. Therefore, testing the microcontroller, which essentially controls the mechanics of the ATS system, is an important procedure. The testing of the microcontroller should be a straightforward process. The entire testing process will consist of two sequential steps, implementation onto the printed circuit board and tying the microcontroller to the servo motors.

The first step in testing is to successfully implement the microcontroller onto a printed circuit board. A proper design will allow the designers to calculate the ideal situation scenarios, so the designers will draw out a sufficient design unique for the ATS system prior to placing on the printed circuit board. Once the design properly replicates the layout design, a multimeter will be used to measure the microcontroller is receiving the sufficient amount of power. This is to ensure the microcontroller was properly soldered onto the printed circuit board and no power is being lost between connections.

Once the microcontroller is properly inserted onto the printed circuit board, the second step for testing can take effect. Since the microcontroller is working up to the standards of the designers, the designers can then solder the connections fo the servo motors to the proper pins. Once all the servo motors are tied to the necessary microcontroller pins, a multimeter will again be used to make sure that the servo motors are receiving the signals sent from the microcontroller. If the servo motors are receiving the signals from the microcontroller, then the testing should be complete for the microcontroller component at least. From this point, the designers can move onto further testing of the entire system such as the manual control testing.

7.1.2 Power Source

The battery source is no different than any other component when it comes to testing. Testing is always crucial to test per component as well as the entire system. The testing of the battery source should be significantly easier then other component because to test the battery, it only needs to be connected properly as well as testing the longevity of the battery. The battery connections can be tested to ensure that no power is lost due to a faulty connection or improper soldering. This can be done using a multimeter to measure the amount of power being output from the battery as well as the amount of power entering the microcontroller. The make sure no power is lost, these two values should be the same number.

Once the connection is properly connected, the designers next need to test the longevity of the battery. This is a simple process of just letting the system run until the battery is completely drained. The system only requires the system to last and perform for at least thirty minutes. This shouldn't be a problem since the designers should be able to run the system over and over to ensure it meets the system requirement of longevity. Once the baby is drained, the designers can replace or recharge the battery since it will be a simple process.

7.1.3 Alarm system

The alarm system testing will be a simple test. The first alarm to be tested is the set of speakers that will play an audio file. The designers will take a specific audio file warning the incoming target that they are in danger and should turn around. Once this audio file is created, the designers can demonstrate its performance by first manually looping the file over and over, just to ensure it works properly and can be heard from 100 feet away from the speakers.

Before attached, the second alarm to the ATS system, the designers will need to first ensure it is properly connected to the microcontroller. Once connected to the microcontroller, a multimeter will be used to make sure enough power is being passed through the microcontroller to the alarm. Again, the designers need to make sure the alarm is audible from 100 feet away. Once these two alarms are successfully demonstrated compliance, they can be attached to the system directly.

From here, the range detecting software will be incorporated. The key threshold as far as the alarms concern is 100 feet from the system. So once the target is within 100 feet, the alarm will sound until the target moves out of this range. When the object is within this threshold, the range detecting software will send 2 signals, one to the microcontroller triggering the continuous alarm and the other to the audio file to play and continuous loop.

To test, the designers will measure out exactly 100 feet, and consistently come in and out of the alarming zones to demonstrate compliance of the requirements. When an object is 100 feet or closer, both alarms will be triggered and continuous alarm as long as they stay in that range. This is the minimum requirement, if the target is further than 100 feet away, the alarm may still alarm however, it isn't required.

7.1.4 Web camera

The web camera testing will happen in a series of steps. Initially, the camera will be connected directly to the laptop. Once connected, the designers need to ensure the camera streams the video feed to the laptop screen in real time. From this point, the designers have assurance the web camera works properly. The designers will continue using this video feed to implement the variety of software classes.

Once the designers are pleased with the results of the software implementation, the web camera will be tied to the barrel of the gun. Once secured, the design team will continue testing to demonstrate the web camera's vision. This is continued to show the camera is stabile on the gun and of course, the vision isn't impaired for any reason. If the camera meets these criteria, the designers will begin to successfully start tracking incoming targets when they come into the field of vision.

7.1.5 Laser Pointer

The laser pointer testing will be a simple, straight forward process. Like the alarm system and the servo motors, the laser pointer will be tied to the microcontroller. So, the first thing the designers must test for is to ensure the laser pointer turns on properly. To do so, they need the correct operating voltage is being provided, between 3 and 4.2 volts. This can be measured using a multimeter. Once they've determined the operating power is sufficient the next thing to test for is placement of the laser pointer for accuracy.

This is where the non-technical testing comes in. The designers will shoot the paintball gun at a stationary target to see where the paintballs hit. Since the paintballs will leave a trace of impact, the designers can easily just aim the laser pointer at the point of impact and then acquire the positioning to attach the laser pointer. It is a simple process but will be sufficient enough to ensure the laser pointer is in calibration to where the paintballs will impact the incoming target.

7.1.6 Capacitive Touch Controller

Before incorporating the capacitive touch controller onto the autonomous turret, the controller itself must be tested for various cases. To begin, each function related to controlling the autonomous turret must be tested to see if the correct bit sequences are transmitted. Each pin dedicated to the specific bits will be measured with a digital multi-meter. When the microcontroller is first activated, the capacitive touch controller should reset back to 45 degrees. If this correctly happens the capacitive touch controller should transmit 00101101 across the transmission pins tapping the left sensor should increase the transmission to 00101110, then to 00101111, then to 00110000. When the capacitive touch controller is tapped right three times, the controller should transmit 00101111, then 00101110, and then back to the neutral 00101101. When the capacitive touch controller is tapped up three times, the controller should transmit 10001110, then 10001111, then 1001000. When the capacitive touch sensor is tapped down three times, the controller should transmit 10001111, then 10001110, then 10001101.

For the hold functions, the capacitive touch controller should increase by 10 or binary 1010. The microcontroller will be reset which should reset the position back to 45 degrees in the pitch and yaw direction. When the capacitive touch controller is held right, the controller should increase from 00101101 to 00110111, then to 01000001, and so forth until it reaches 140 degrees at binary 10001100. The microcontroller should transmission bits should not increase if the right button is held. For the hold left function, the transmissions bits transmissions decrease by 10 degrees or binary 1010 similar to the hold right function. The process will be repeated for the hold up and hold down functions keeping minding to the starting and ending range found in the Summary of Angle Transmission for Method 2 table.

The capacitive touch controller will then need to be tested for the special user case if the user tries to press more than one sensor at a time. To test this special user case, the left sensor will be held and the up button will be tapped. If this special case works correctly, the capacitive touch controller should decrease to 00000000 until the left sensor is released. When the left sensor is released and the up sensor is tapped the touch controller should jump to a binary transmission between 10001101 and 111100111 in order to transmit the updated pitch angle. The capacitive touch controller will then be tapped in the middle which should cause the controller to transmit a 11111111 for 20 milliseconds and then return to its previous value.

Once each function and special user case has been tested, the entire process will be repeated once controller is implemented with the microcontroller controlling the servo motors.

7.2 Software Testing

The software testing for the ATS will happen in a series of phases, which are detailed in figure 22. Step 1 and 2 have already completed a single iteration, the tests have been planned and designed as per each test case listed below. The next step is to execute the tests that have been designed. To do this, the program must first be coded so this step will not be complete until later in the second semester. The test execution will be broken up into two major components. The first component is module testing. Each section of the computer vision application will be tested alone to see if it properly completes its individual task in the process. These modules consist of object detection, color detection, object tracking, turret, and facial recognition. Once each module has successfully completed its module testing, the modules can be integrated with one another to execute the full system test. If this test is a success then the testing for the ATS has been completed and the test summary can now be written for the system.

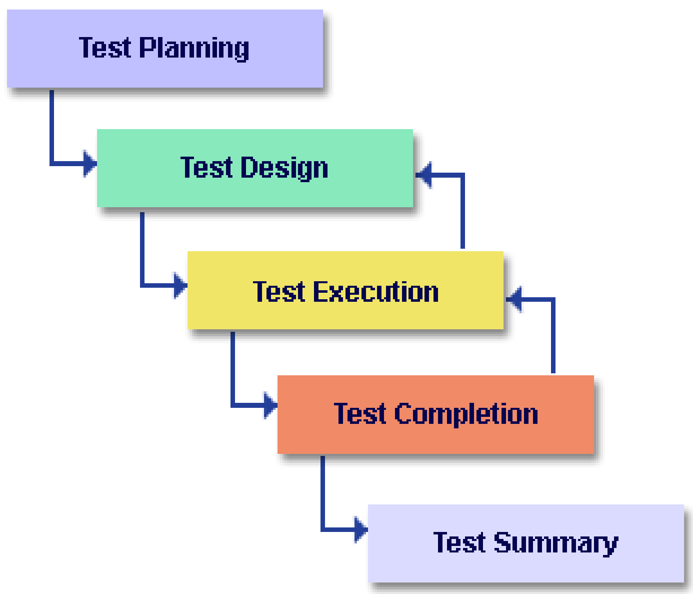


Figure 22 - Testing process

7.2.1 Color Detection Testing

7.2.1.1 Detecting the color of a single colored object

Test Case Description: Given a frame from the web camera, the color detection class should be able to accurately determine the hue of the given object.

Success – The class should return the corresponding value of the color of the object.

Failure – An incorrect value or no value is determined for the object.

7.2.1.2 Detecting the color of a multi - colored object

Test Case Description: Given a frame from the web camera with an object containing multiple colors the class should be able to determine the hue that is most prevalent in the frame.

Success – For this test to be considered a success the class must successfully return the most prevalent hue value of the object in the frame.

Failure – The class returns the wrong hue, minority hue, or no hue at all.

7.2.1.3 Detecting the color of multiple objects in the field of vision simultaneously

Test Case Description: Given a frame from the web camera with multiple targets the color detection class should be able to accurately determine the hue of each target.

Success -The class should return the corresponding value of the color of each object in the given frame.

Failure -The class does not identify each target, or an incorrect value or no color is determined for the objects.

7.2.2 Object Detection Testing

7.2.2.1 Detecting one object in the field of vision

Test Case Description: Given a frame from the web camera the object detection class must identify a potential target using Haar-like features, and placing a rectangle around the object.

Success –The object detection class places a rectangle around the object.

Failure – The class fails to recognize the target or places a rectangle in the wrong location on the frame.

7.2.2.2 Detecting multiple objects in the field of vision

Test Case Description: Given a frame from the web camera the object detection class must identify potential targets using Haar-like features, and placing a rectangle around each object.

Success – The class places a rectangle around each object in the frame.

Failure – The class fails to recognize targets or places rectangles in the wrong locations of the frame.

7.2.2.3 Detecting objects that leave and re-enter the field of vision

Test Case Description: When an object exits the field of vision of the web camera its rectangle and target identification is destroyed. When another object or the same object re-enters a new rectangle and target ID is created.

Success – The class recognizes that the object has left the field of vision and destroys the correct rectangle and corresponding target ID. The class creates a new rectangle and target ID for objects entering the field of vision.

Failure – The class fails to destroy the objects rectangle and target ID when they have exited the field of vision. The class fails to assign a new rectangle and target ID to objects entering the field of vision.

7.2.2.4 Detecting objects that are partially occluded from the field of vision

Test Case Description: A Kalman filter is a recursive filter that projects the path of objects by using the objects previous locations to determine its path. This is useful when objects are obscured and moving behind obstacles.

Success – The class must be able to successfully project the path of an object and update the position of the object's rectangle even when the object moves behind an obstacle or is obscured from the web camera.

Failure – The class loses the object and destroys its rectangle and target ID when moving behind an obstacle. The class incorrectly predicts the targets path.

7.2.2.5 No objects in the field of vision

Test Case Description: The ATS will be placed in an open area with no targets present in the field of vision. Everything in the area will be purely background.

Success – The sentry gun does not pick up anything else as a target and does not fire at anything.

Failure - The sentry gun determines that there is a target, when there is none, or fires upon anything in the field of vision.

7.2.3 Manual Control Testing

7.2.3.1 Rotating the sentry gun left and right

Test Case Description: This test involves sending pulse signals to the horizontal servo motor to rotate the turret to the left or right.

Success – The turret increases or decreases its horizontal angle the specified amount. The angle must change by the specified increment and with the different input methods capacitive touch, arrow keys, and the Windows Form buttons.

Failure – A failure is defined as the turret not responding to any of the three input methods. Failures also include the turret rotating too much or too little, and not stopping after rotating.

7.2.3.2 Panning the sentry gun up and down

Test Case Description: This test involves sending pulse signals to the vertical servo motor to rotate the turret to the left or right.

Success -The turret increases or decreases its vertical angle the specified amount. The angle must change by the specified increment and with the different input methods capacitive touch, arrow keys, and the Windows Form Application buttons.

Failure - A failure is defined as the turret not responding to any of the three input methods. Failures also include the turret rotating too much or too little, and not stopping after rotating.

7.2.3.3 Altering the rate of fire of the sentry gun

Test Case Description: This test consists of altering the rate of fire of ATS by using the list menu on the Windows Form Application. There are three rates of fire fully automatic, burst, and single shot.

Success – A success is defined by the turret's rate of fire. In order to pass this test the turret must accurately react to the changing of fire rate by adjusting the speed that the trigger servo fires and adding delays or stops as necessary.

Failure – The test is a failure if the turrets rate of fire does not change after selecting an alternative rate of fire. The test also fails if ATS fires at a rate that does not correspond to the selected rate of fire for instance the turret is firing in bursts when single shot is selected.

7.2.3.4 Firing the sentry gun.

Test Case Description: This test will ensure that the Manual Control class fires the turret when the user either presses the Fire button located on the Windows Form Application or the spacebar.

Success – A success is defined by the gun firing when pressing either the button on the application or the spacebar. The gun must also fire at the correct rate.

Failure – A failure occurs if the gun does not fire when using either the button or spacebar. A failure is also defined if the turret does not fire at the specified rate or does not stop firing once starting.

7.2.3.5 Alerting the target with a warning

Test Case Description: The alert button on the Windows Form application is used to warn intruders that they are entering a restricted area, and that they should leave the area immediately.

Success – A success occurs when the correct audio file is played through the speakers when pressing the alert button on the application.

Failure – The wrong audio file or no audio is played at all. The audio does not play clearly or plays in a continuous loop.

7.2.3.6 Alerting the target that the sentry gun is about to attack

Test Case Description: The Fire_Warning button on the Windows Form application is used to warn intruders that they will be fired upon by the turret.

Success - A success occurs when the correct audio file is played through the speakers when pressing the alert button on the application.

Failure -The wrong audio file or no audio is played at all. The audio does not play clearly or plays in a continuous loop.

7.2.4 Object Tracking Testing

7.2.4.1 Tracking a single, walking target

Test Case Description: One target will walk through the field of vision for a short period at a constant speed. The target will change directions on multiple occasions.

Success – The sentry gun keeps up with the target for the entire duration of the test, following all movements in any direction.

Failure - The sentry gun loses track of the target, aims in the wrong direction, or cannot keep up with the speed of the target.

7.2.4.2 Tracking multiple, walking targets

Test Case Description: Three targets will walk through the field of vision for a short period at the same, constant speed. Each target will change their direction and pass one another throughout the test.

Success - The sentry gun tracks the target with the highest priority and if another target becomes higher priority, it switches to that target.

Failure - The sentry gun tracks no targets, or tracks a target with lower priority than another, or if the sentry gun fails to switch targets upon a change of the highest priority target.

7.2.4.3 Tracking a single, running target

Test Case Description: One target will run through the field of vision for a short period at a fast speed. The target will change its direction on multiple occasions during the test.

Success - The sentry gun keeps aim on the target throughout the duration of the test and never loses track of it.

Failure – The sentry gun fails to keep up with the target, or does not correctly track the target.

7.2.4.4 Tracking multiple, running targets

Test Case Description: Three targets will run through the field of vision for a short period at a fast speed. Each target will alter its direction on multiple occasions.

Success – The sentry gun keeps up with the highest priority target and switches targets whenever another target becomes more important.

Failure – The sentry gun fails to keep up with the target it is tracking, tracks the wrong target, or fails to switch targets when another gains higher priority.

7.2.4.5 Tracking multiple targets with varying velocities

Test Case Description: Three targets will run, jog, or walk through the field of vision for a short period. Each target will change its speed and direction during the test.

Success - The sentry gun tracks the highest priority target and keeps track of them regardless of the speed. Also it must switch between targets when priority changes.

Failure – The sentry gun fails to track targets, keep up with them, switch targets when another gains priority or tracks the wrong target.

7.2.4.6 Tracking a high speed object

Test Case Description: An object will enter the system's field of vision, at a high rate of speed (a ball thrown by a person). The ball will cross through the field of vision from one side and exit on the other side.

Success – The sentry gun will detect, track, and hit the high speed target.

Failure – The sentry gun fails to register the object as a target, cannot keep up with the speed of the object or fails to aim at the target properly.

7.2.4.7 Tracking targets that stop moving for a period of time

Test Case Description: Three targets will enter the field of vision of the ATS and at some point during the test stop moving for a short duration and then continue moving.

Success - The sentry gun keeps firing at the stationary targets and then proceeds to track their movement once it resumes.

Failure - The sentry gun loses the target because it is no longer moving or fails to continue tracking it once movement resumes.

7.2.5 Fire Rate Testing

7.2.5.1 Single shot

Test Case Description: The fire rate will be initialized to represent a single shot rate of fire.

Success – The system registers the rate of fire command and sends the correct signal to the servo motors to replicate a single shot.

Failure – The system does not register the rate of fire command or inaccurately commands the servo motors to fire the weapon.

7.2.5.2 Burst fire

Test Case Description: The fire rate will be initialized to represent a burst fire (three to five shots) rate of fire.

Success - The system registers the rate of fire command and sends the correct signal to the servo motors to replicate a burst fire.

Failure - The system does not register the rate of fire command or inaccurately commands the servo motors to fire the weapon.

7.2.5.3 Automatic fire

Test Case Description: The fire rate will be initialized to represent an automatic rate of fire.

Success - The system registers the rate of fire command and sends the correct signal to the servo motors to replicate a fully automatic weapon system.

Failure - The system does not register the rate of fire command or inaccurately commands the servo motors to fire the weapon.

7.2.6 Threat Level Testing

7.2.6.1 A single non-threat target

Test Case Description: One target, that is a non-threatening target, will enter the field of vision and move throughout the area for a short duration.

Success - The target is detected, determined to be a non-threatening target and therefore not fired upon.

Failure - The target is not detected, or detected and determined to be a threatening target, or if the target is determined to be non-threatening but still fired upon.

7.2.6.2 A single threatening target

Test Case Description: One target, that is a threatening target, will enter the field of vision and move throughout the area for a short duration.

Success - The target is detected, determined to be a threatening target and therefore fired upon by the weapon system.

Failure – The target is not detected, determined to be a non-threatening target or is not fired upon entry to the field of vision.

7.2.6.3 Multiple non-threat targets

Test Case Description: Three non-threatening targets will enter the ATS' field of vision and move around for a short duration.

Success – The system accurately detects all targets in the field of vision and determines that they are of no threat and therefore does not fire upon any of the targets.

Failure – The system does not detect all targets, or if the system determines one of the non-threatening targets to be a threat. Also a failure occurs if the system fires upon any non-threatening targets.

7.2.6.4 Multiple threatening targets

Test Case Description: Three threatening targets will enter the ATS' field of vision and move around for a short duration.

Success – The system accurately detects all targets in the field of vision and determines that they are all threatening targets, and then chooses the one with the highest priority to fire upon.

Failure – The system fails to detect all targets or incorrectly determines any of their threat levels. Also a failure occurs if a target with the most priority is not fired upon.

7.2.6.5 Mix of threatening and non-threat targets

Test Case Description: Two separate tests will be run for this case. Two non-threatening targets and one threatening target will enter the field of vision and move around for a short duration. Then one non-threatening target and two threatening targets will enter the field of vision and move around for a short duration.

Success – The sentry gun determines the threat level of all targets in the field of vision and fires upon only threatening targets with the highest priority (in the case of one threatening target, it will only fire on that target, and, in the other case, will choose the most important of the two threatening targets).

Failure – The sentry gun incorrectly determines the threat level of any of the targets in the field of vision or fires upon a target that was deemed non-threatening. Also a failure occurs if a threatening target without the highest priority is fired upon.

7.2.7 Facial Recognition Testing

7.2.7.1 Detecting a Face Using the On-board Computer Camera

Test Case Description: When accessing the manual control class the user will be asked to position their face in the frame. The facial detection and recognition class must acknowledge that the image is indeed a face

Success -The class should recognize the presence of a face and move to step two which is matching the image with one in the face database. If no face is in the frame the class should prompt the user to position their face in the center of the screen. The identification processes will time out after one minute.

Failure –The facial detection class fails to recognize the face in a given frame. The class does not prompt the user to re-position or falsely identifies a non-face as a human face. The class fails to time out after one minute.

7.2.7.2 Matching a Face with a Face in the Database

Test Case Description: After confirming that the image is indeed a face, the class must correctly match the face with an image of the face in the database. The class must also refuse access to anyone trying to access the manual controls that is not in the database.

Success -The class correctly matches a face in the web camera frame to a face in the facial database. The class denies access to anyone not in the facial database.

Failure – The class falsely grants access to a person not in the database. The class denies access to those persons in the database.

7.2.7.3 Granting Access to Those in the Database

Test Case Description: After matching a face with one in the facial database the class will call the Manual Control class and grant the user access.

Success – A success is defined by granting access to the manual control class for those users located in the manual control class.

Failure – The class does not grant access to users that have images in the facial database.

7.2.8 Integrated Software System Testing

7.2.8.1 One target in the system

Test Case Description: A single target will enter the field of vision and move throughout it for a duration of time.

Success – the target should be detected, determined to be a threat or not, tracked, and if a threat fired upon. If not a threat then the weapon should not fire at all.

Failure – Any component of the software fails to complete its specified task. i.e. if the target is not detected, if the threat level is incorrectly determined, if it is not properly tracked, or if the target is fired/not fired upon when it should be the opposite.

7.2.8.2 Multiple Targets in the system

Test Case Description: Three targets will enter the field of vision and move throughout it, in varying speeds and directions, for a duration of time.

Success – all targets should be detected, determined to be a threat or not, tracked, and if any are threats, the weapon should fire on the most important target. If there are no threats then the weapon should not fire at all.

Failure – Any component of the software fails to complete its specified task. i.e. if any of the targets are not detected, if the threat level for any target is incorrectly determined, if the highest priority target is not properly tracked, or if the target with the most priority is fired/not fired upon when it should be the opposite.

7.2.8.3 Targets entering and leaving the system

Test Case Description: Three targets will enter the field of vision and move throughout it, in varying speeds and directions, for a duration of time. Then some will exit the field of vision, come back in, others will do the same a short time after.

Success – all targets should be detected, determined to be a threat or not, tracked, and if any are threats, the weapon should fire on the most important target. If there are no threats then the weapon should not fire at all. Whenever a target leaves the field of vision, all of its data should be freed and if the object happens to re-enter, then new fields should be created.

Failure – Any component of the software fails to complete its specified task. I.e. if any of the targets are not detected, if the threat level for any target is incorrectly determined, if the highest priority target is not properly tracked, or if the target with the most priority is fired/not fired upon when it should be the opposite. Also if the system fails to delete or create new data for targets exiting or entering the field of vision, respectively.

7.3 Software and Hardware Communication Testing

The software of the ATS will have to control the hardware on multiple occasions to get the intended result of the system. The microcontrollers and servo motors will both need to be sent signals to be told what to do.

7.3.1 Pitch Servo Motor Communication

Test Case Description: The servo motor that controls the pitch will be sent a series of signals that tell it how much to move. These positions will cover the full range that the servo motor is required to operate within.

Success – The servo motor responds correctly to all signals, moves to the correct position and rotates with the necessary speed.

Failure – The servo motor goes to a wrong position, the wrong signal is sent to the servo motor or if the servo motor rotates too slowly to keep up with moving objects.

7.3.2 Yaw Servo Motor Communication

Test Case Description: The servo motor that controls the yaw will be sent a series of signals that tell it how much to move. These positions will cover the full range that the servo motor is required to operate within.

Success – The servo motor responds correctly to all signals, moves to the correct position and rotates with the necessary speed.

Failure – The servo motor goes to a wrong position, the wrong signal is sent to the servo motor or if the servo motor rotates too slowly to keep up with moving objects.

7.3.3 Trigger Servo Motor Communication

Test Case Description: The servo motor that controls the trigger will be sent a series of signals that tell it when to pull the trigger. These signals will test all three of the fire rates to be used (automatic, burst, and single shot).

Success – The servo motor responds correctly to all signals, pulling the trigger to simulate an automatic, burst, or single shot fire rate as specified.

Failure – The servo motor does not simulate the correct fire rate or does not fire the weapon at all. Also a failure occurs if the wrong signal is sent to the servo motor.

7.3.4 Microcontroller Communication

Test Case Description: The microcontroller software will send the microcontroller signals to let it know what to do.

Success – The microcontroller accurately receives all signals sent to it and responds accordingly.

Failure – The incorrect signals are sent or received or the microcontroller operates in the wrong manner.

7.3.5 Web Camera Communication

Test Case Description: The web camera will be hooked up to the computer running the program and it will need to send its video stream into the program through the videoCapture class.

Success – The web camera sends its video stream to the videoCapture class and all frames of the video are received from the web camera and are able to be used in the computer vision application.

Failure – The web camera fails to send its video stream to the software, or the videoCapture class does not correctly interface the device with the program. Also a failure will occur if the frames received from the web camera are not able to be used in the program for whatever reason.

7.4 Overall System Testing

Once ATS's components are tested individually it is time to start assembling the individual components to test their interactions with one another. The first thing the group will test is the interaction between the microcontroller and the servo motors. In order to create an automated turret it is imperative that the microcontroller can send signals to the servo motors to move them. The next step is to start assembling the base; once this is completed the group can test the motors again to ensure they have enough torque to move the turret. The next step is to hook up the on-board computer and test the object tracking is working with the web camera. Various software components will then be checked starting with manual control, to ensure that the microcontroller is processing the information correctly and sending signals to the servo motors. Once the full system is assembled the group can begin testing the turret and the interactions that the individual components have with each other. The systems longevity and durability must also be tested. These tests must demonstrate that the turret will hold up to the recoil of the gun and the system will remain running at least thirty minutes on battery power.

Overall system testing will be done by the group as a whole. The test cases will be recorded and reviewed in order to correct problems in a timely matter. Having the entire group for testing allows each member to work on their respective components and correct issues more easily. Each member will keep a testing journal recording the results for each test and how their respective components or software classes functioned. To ensure functionality the tests will be repeated several times and in different testing environments to account for weather and environmental variables. System testing will occur throughout the semester assembling adding components on as testing occurs. Full system testing with all components is scheduled to take place for the entire last week of the semester. This will ensure that there is enough time to catch any problems that might arise and guarantee that ATS is ready for demonstration day.

7.4.1 Using manual control test the turret's movement

Description- To test the interactions between the onboard computer and the microcontroller, and the interaction with the microcontroller and the servo motors the turret will be moved left, right, up, and down.

Success – The turret moves in the specified direction incrementing the angle by a set degree every time. The turret must move in all four directions and do so in a time efficient manner.

Failure –The turret does not move, or moves in the wrong direction. The turret adjusts its angle by too much or too little. The turret gets stuck or the weight of the turret is too much for the motors. The turret does not stop moving once it has reached its correct position.

7.4.2 Using the object tracking class to detect objects and move the turret accordingly

Description – Once the manual controls are working the next step is to integrate the computer vision software to move the turret. The web camera will be used to detect targets while the object detection and tracking classes will send the coordinates to the microcontroller to process. Once the coordinates are processes the microcontroller will move the servo motors accordingly.

Success – The object detection class successfully picks up targets in the camera’s field of vision. The object tracking class continuously sends coordinates to the microcontroller which in turn moves the servo motors.

Failure – The object detection class fails to identify any targets or the object tracking class does not send the coordinates to the microcontroller. The servos fail to move or incorrectly interpret the coordinates and move in the wrong direction.

7.4.3 Battery testing

Description – The system must sustain thirty minutes of operation powered by nothing but the battery. The thirty minute duration will be tested multiple times to ensure the battery is not degrading and charging correctly.

Success – The system remains up and running for at least thirty minutes of tracking targets and regular use of the gun.

Failure – The system does not remain operational for thirty minutes or encounters an error that forces the group to recharge the battery and retest.

7.4.4 Durability testing

Description – The system including the base, hardware plexiglass box and the turret must hold up the vibration and rigor caused by the recoil of the gun. This stress test will take place throughout the rest of the testing. If the components are still functioning and the base is still sturdy the test is passed.

Success – The hardware components and base hold up to the week of system testing. This ensures that the turret will function on demonstration day.

Failure – The hardware components are malfunctioning due to vibration or the bases stability has been compromised. If the testing fails the group must reinforce the turret and re-test.

7.4.5 Single target testing

Description – The system will first be tested using a single target. Once the system is powered on a target will walk or jog at a temperate pace. Once the system has acquired the target and added the object to the target list it will warn the target that it is entering restricted five seconds after entering the field of vision. After fifteen seconds the turret will play another audio file informing the user it is going to fire. The turret will fire upon the threat until the target has exited the field of vision or until the target is covered in enough green paint to deem the target no longer a threat.

Success – ATS acknowledges the target and adds it to the target list once entering the field of vision. The warning audio file plays after five seconds and the commence firing audio file plays after fifteen seconds of acquiring the target. ATS begins firing on the target until the target either exits the field of vision or is marked with enough green paint.

Failure – There are numerous actions that would deem this test a failure. The first is that ATS does not recognize the target and does not add the threat to the target list. The audio files do not play, or the gun does not track the target are also considered failures. Issues concerning the gun not shooting or ceasing to fire are also considered failures.

7.4.6 Multiple target testing

Description – Once the group is confident that ATS is capable of handling single targets the multiple target testing will take place. Multiple target testing involves several objects or threats entering the field of vision. ATS will then add them to the target list and rank their priority by time spent in the field of vision. ATS will track the target that spent the most time in the system playing the warning audio file at five seconds and the commence firing audio file at fifteen seconds. Once the target ATS is tracking exits the field of vision or is deemed no longer a threat the turret will focus its attention on tracking the next target in the priority list.

Success –Multiple target testing is considered a success if all targets are either eliminated or exit the field of vision. The targets must be eliminated in the order they enter the system's target list.

Failure – ATS fails to recognize any targets or focusses the entire time on just one target. ATS does not delete eliminated targets from the priority resulting in only one target being shot at. The system gets confused with conflicting targets and loses the target it is currently tracking.

7.4.7 Demonstration testing

Description – The demonstration of the system should successfully demonstrate all of the features of ATS. The group will prepare the demonstration ahead of time including all of the features of ATS and test the full demonstration a minimum of ten times.

Success – ATS successfully demonstrates its capabilities and functionality all ten times.

Failure – ATS fails a step in the demonstration. If this occurs the group will resolve the issue and retest. It is imperative that the ATS demonstration succeeds ten times in succession to ensure without a doubt that the system will function on demonstration day.

7.5 Environmental Testing

Environmental testing is a method to ensure the ATS system is capable to withstand a variety of different environments. Since the weather is a dynamic element that is rarely understood and typically unpredictable, the designers need to test under different circumstances to make sure the system will perform properly. By exposing the system to a wide variety of environments, the designers can be reassured of the reliability which is an important aspect of any prototype owned.

7.5.1 Indoor testing

The ATS system will be tested both inside and outside, outside mainly to demonstrate the system will be able to continually perform even when exposed to a variety of environmental weather conditions. The ideal condition for the prototype testing will be within doors, mainly so the designers can control the environment. The system is intended for both indoor and outdoor use so by demonstrating the system can perform within doors where the external conditions are controlled, the designers will successfully demonstrate a working prototype for at least the ideal situation. Outdoor testing will add more difficultly just because the uncontrollable weather conditions.

7.5.2 Outdoor testing

The ideal condition for any prototype is a controlled environment such as within doors where the conditions are monitored accordingly. However, the ATS system is intended to have multiple uses, including properly functioning even in the outdoors. Therefore, the designers will test outside to ensure the system can handle the unpredictable weather when left exposed to natural elements.

7.5.2.1 Humidity

Humidity is the amount of water vapor in the air. This can be a problem for electrical components and the construction of the prototype. The initial ATS prototype will be constructed in Florida; often a very humid environment because it is surrounded by water and has extremely hot temperatures. Since the construction will be in Central Florida, humidity testing will be a simple process by just testing outdoors.

The main concern for humidity will be the functionality of electrical components. The designers intend to place the printed circuit board in a clear, plastic container what will hopefully keep out or reduce humidity. This is the main concern for the designers since the printed circuit board will be the key ingredient in controlling the ATS system. All other components such as the battery, servo motors and paintball gun should be unaffected by the humidity factor.

7.5.2.2 Rain

Several components of the ATS system will be waterproof, however; as a complete unit, the system will not be intended for rain to preserve the external corrosion of the system. Ideally, the system will perform in rain since the majority of the electrical components will be stored at the bottom of the base which will be sealed and covered from the rain but the system requirements have no need for operation in the rain so the designers will work around that particular requirement.

7.5.2.3 Snow and cold temperatures

As previously stated, the construction and all testing will be in central Florida. Although temperatures do drop near or below freezing, the cold weather will not be a factor for the ATS system, only snow has the potential to affect the performance of the system. The last time snow was recorded in central Florida was 1977, so the designers and tester of the system will assume snow will not be an issue and will not test for this environmental condition. Occasional freezing has occurred in central Florida, typically overnight, but the system will not be stored out overnight during these cold evenings.

7.5.2.4 Windy conditions

Wind can be a factor in affecting the performance of the ATS system, mainly for stability and accuracy. As far as stability, the designers will take into account the wind factor and design the architecture of the prototype with a wide base and the majority of the weight near the bottom which will make it difficult to tip over from the top. Of course, Florida is notorious for hurricanes and hurricane force winds, these wind conditions are not being accounted for and the designers have no intentions of the system being exposed to hurricane force winds.

As far as accuracy, the system will continue to perform properly except the wind may be a factor for the ammunition successfully hitting the target. Since the ammunition is a paintball, the wind may be strong enough to defer the path of the paintballs, especially for the longer range targets. For this to happen, the wind must be blowing pretty fast but this wind factor is a rare occurring factor so the wind should rarely affect the performance of the system.

7.5.2.5 Evening testing

The ATS will have the ability to perform in dim lighting so the system will be tested during evenings as well as day time settings. The camera is not a full on night vision camera so the system's performance maybe affected based on the amount of natural lighting during the night. Ideally, the system will perform with the same success as during day time testing, however; the designers will not know until the prototype is successfully built and the system can be tested. If it is completely black out and the camera will not be able to successfully identify the targets so the designers anticipate the ATS system will not perform under this specific situation.

8.0 Administrative Content

8.1 Timeline

The designers created a master schedule to follow to ensure the production of the ATS system. Two separate schedules were developed for the fall and the spring. The fall schedule primarily focuses in research and design. The spring schedule's main focus is development and testing. Each schedule is outlined with starting and completion target dates.

8.1.1 Fall 2011

In the first week of September, the group was chosen. Members were chosen based on the skills each person possessed, and how they could be applied to the project. After the group was chosen, the design proposal was written up over the next week. The month of October will be spent doing mostly research for the project and beginning the technical documentation required for the semester. The research will be broken up into sections for each member, and they will be responsible for relaying the vital information of that section to the rest of the team members. James and Daniel will handle the majority of the software research, while Stephen and Ethan will take the lead on the hardware research. The month of November will be dedicated to the design of the system, while finishing the technical documentation, simultaneously. The same team strategy will be utilized for the design section, which means that James and Daniel will handle software design and Ethan and Stephen will handle the hardware design. The whole team will collaborate on important design decisions but the details will be handled by individual team members in control of that section. The weekend before the submission deadline will be allocated for reviewing and formatting the document with a one day buffer for any last minute problems. This is the ideal scenario so that there are no last minute changes needed, but if there does happen to be, there is time to make the necessary changes. The fall semester will be a more strict schedule than the spring semester, meaning that the full allotted time for each section will be spent and no extra time unless necessary. This is to keep the design process organized and efficient and to prevent the group from falling behind at any point. Overall, the group has stuck to this schedule fairly well so far and plans to continue doing so. The fall schedule can be found in table 15.

Fall 2011 notional schedule		
Task	**Start Date**	**Completion Date**
Group Selection	September 5, 2011	September 9, 2011
Design Proposal	September 15, 2011	September 22, 2011
Research	October 3, 2011	November 4, 2011
Technical Documentation	October 17, 2011	December 2, 2011
Design	November 4, 2011	December 2, 2011
Review/format documentation	December 2, 2011	December 4, 2011
Document Submission	December 5, 2011	December 5, 2011

Table 15 - Fall 2011 Schedule

8.1.2 Spring 2012

The initial purchasing of materials will begin shortly after the Fall 2011 semester ends. Materials will still be obtained throughout the Spring semester, but the essential components will be bought in the first month. At the same time, the initial coding will begin with the basic object detection and object tracking algorithms. These will be completed by the end of January. The hardware assembly will begin at the start of January and continue for two months. Ethan will handle the PCB design and fabrication, while Stephen will work with the servo motors and microcontrollers. Once completed, they will combine their work to assemble the overall structure of the system. Also the turret and target classes will be coded in the month of January by James and Daniel. The base structure for the turret will be built in the second half of January. Once the rest of the coding has been completed, the extra features of color detection and facial recognition will be added during the month of February. At this point the coding modules will be integrated with one another to be tested in full. If testing is successful then the software will be ready to be integrated with the hardware for final testing. This will be followed by integrating the hardware and software to prepare for the final weeks of testing. Note that testing will happen throughout the process, but the period listed will be solely dedicated to testing all possible cases the ATS could encounter. Finally the website will be added and the final presentation will occur in the last week of April. The timeline for this semester will be flexible so if one section finishes early or takes longer than expected, the dates for the other tasks to be done can be adjusted accordingly to make the most of each day left in the build and test process. The spring schedule can be found in table 16.

Spring 2011 notional schedule		
Task	**Start Date**	**Completion Date**
Purchase Materials (initial)	December 15, 2011	January 20, 2012
Code the object detection and tracking algorithms	December 15, 2011	February 1, 2012
Servo and Microcontroller set-up	January 2, 2012	March 1, 2012
PCB Design / Fabrication	January 2, 2012	March 1, 2012
Code the turret and target classes	January 2, 2012	February 1, 2012
Build base structure	January 16, 2012	February 1, 2012
Code facial recognition and color detection algorithms	February 1, 2012	March 1, 2012
Integrate coding modules	March 1, 2012	March 10, 2012
Integrate Hardware and Software	March 10, 2012	March 19, 2012
Test/debug	March 19, 2012	April 13, 2012
Website	April 16, 2012	April 20, 2012
Submission	April 25, 2012	April 30, 2012

Table 16 - Spring 2012 Schedule

8.2 Budget

ATS is funded entirely by the group. The cost of the parts, materials, and service costs will be shared equally by the four group members. This was a big decision, but in the end the group felt that this was the best option because of the extra paperwork and stipulations of having a sponsor. The original budget was set at $1000, or $250 dollars each. While researching and comparing parts it was clear the team overestimated the budget. The cost as of now of all the required parts is $358 dollars. Many components including the paintball gun and all of its accessories are already owned by a group member. The group is willing to spend the original $1000 to account for unforeseen costs and replacement parts, in case they burn out or are damaged during testing. The extra money can also be used to upgrade our project and add additional features if there is time. It has not been decided yet as to who will inherit ATS once this project is over. It will likely be raffled off, or disassembled and parts can be distributed equally. Another option is to donate the project to give future students an idea of what senior design projects consist of. The tentative budget can be found in table 17.

Tentative budget for the ATS system			
Part	**Cost**	**Quantity**	**Total Cost**
Hitec HS-815BB	$44.99	1	$44.99
Hitec HS-805BB	$39.99	1	$39.99
Hitec HS-81	$17.99	1	$17.99
Logitec C310 HD	$29.99	1	$29.99
Framing Materials	$100.00	N/A	$100.00
PIC16F1503	$1.00	3	$3.00
PCB	$33.00	1	$33.00
Paint Ball Gun	Previously Owned	1	N/A
Hopper	Previously Owned	1	N/A
Co2 Tank	Previously Owned	1	N/A
Co2 Coil Hose	$29.99	1	$29.99
Paintballs	$30.00	1 x 2000 count	$30.00
HK-E03358Laser Pointer	$14.99	1	$14.99
Resisters, Capacitors, Diodes	$15.00	N/A	$15.00
Total (pre-tax)			$358.94
Estimated Budget			$500.00

Table 17 - ATS component planned expenditures

9.0 Reflections

9.1 Features left out

The ATS system has plenty of room for improvement, and, since it has been designed with modifiability and upgradability in mind, it is capable of handling those improvements. Also, the ability to have interchangeable parts is an objective of the ATS so switching to a new and improved part will be relatively easy. The first improvement that should be made is adding the range finder discussed in the features left out section. This would add another factor to the decision of when to attack and when not to attack the targets in the field of vision (this would be based on very specific ranges for warning, engaging, and disengaging).

The next improvement that should be made on the ATS is hardware upgrades to the weapon system. The paintball gun and equipment used for the design was already owned so to minimize the budget it was used. However, with a more hi-tech paintball gun, a more efficient system could be built around it. There are paintball guns that already have wiring to make the weapon fire automatically so that could be utilized to avoid the simulation of an automatic firing by the trigger servo motor. This would help make the system more lightweight and efficient. Also, in terms of hardware improvements, the servo motors only cover a certain range to rotate so it would be a great improvement to give the system a larger range of vision (if possible 360 degrees by rotating it around constantly). Being able to cover all directions of the area would be an improvement on just a wide range in front of the sentry gun. Furthermore, the base could probably be improved upon by adding a sturdier base that can not only house the components but is designed with sections to accomplish that task. This would only be a minor improvement though and would probably be one the lower end of the priority list of upgrades to the ATS system.

As for software improvements, the computer vision application could easily become more robust, making more complex decisions based on more variables. The system, as it currently stands, was designed to be in a middle ground of robust and lightweight. This was chosen to make the system feasible, given the time period, while still having a good feature set to make the software for the ATS unique. As of right now, the system only takes into account a few variables to determine whether or not to shoot the target such as color and size. So, if it were to base its decision-making process on more factors like distance, speed, movement direction, and other important variables, the system would have greatly enhanced functionality and application. On top of that, more extra computer vision features could be added, like the android application discussed in the research section or a database of engagement history could be stored so the user has a complete file on all of the activity of the system.

In conclusion, the ATS system is a great project but still has much room for improvement in both hardware and software sections. Only so many features and components that have been researched could be added to the system in the design phase. Due to the constraints of the project, which include budget, time, man-power, etc..., only so much could be added into the final design. If the group decides to take this project beyond the scope of this class, then these hardware and software upgrades are definitely some of the improvements that would be made to the system. The intent on doing so will be directly related to the success of the project during the class. Also, it is possible that if there is spare time left at the end of the semester, some of these future improvements get moved up into the final design of the project.

9.2 Future Improvements

Throughout the research and design process of the ATS, there have been many decisions on what extra features to utilize within the project to enhance its quality. Some were agree upon, while others did not make it into the design of the system. The first major design decision made was the choice to use openCV software or FPGA hardware to complete the task of computer vision within the ATS. This was perhaps the most difficult decision that the group made because it was the basis of the whole project and a choice between a hardware-heavy or software-heavy project. OpenCV was chosen because of the preference for software within the group and the enhanced capabilities of the computer vision application if software is used. Although, using hardware to complete the computer vision would have made that part considerably faster, due to hardware being faster than software, it did not fit the teams goals with this project. Even within the choice of software there was a decision to be made between openCV and Aforge.net software for the computer vision application. The group made the decision to go with openCV based on the fact that it uses C++ rather than C#. Also, openCV has a larger library of computer vision functions and has been more thoroughly tested so the group knows it can rely on the functionality of the openCV classes and methods that will be used in the ATS' software. Overall, openCV has the best characteristics out of the three computer vision options of FPGA hardware, openCV software, and Aforge software.

Another feature that did not make it into the final design is the android application for manual control. Two members have worked with android applications before and it seemed like an interesting way to control the turret manually, however the process of interfacing the on-board computer that is running the main program with the phone running the android application seemed to be a challenge. A database would have most likely been needed, which adds another large component to the project just for a small amount of data sharing between the two devices. Furthermore, the android application would basically operate exactly the same as a user interface for the on-board computer so it does really add a new element in that sense to the project. The database would be a new element in the ATS design, which could also be used for other features like recording engagement history statistics. In light of this decision, a capacitive touch controller has been decided to be used to control the turret manually, which can be seen in detail in the hardware research and design sections.

A range finder was also initially proposed in the design of the system. This would be used to find the range of the targets so the system could warn the targets when they were entering a danger zone. Also they would be warned before they were fired upon. The range finder was also going to be used to determine when to disengage a target (once they exited the range of 75 feet from the sentry gun). However, due to cost and budget limitations this is not planned for the final prototype of the ATS. It is still a possibility for the design of the ATS and is the first feature that will be added back in if there is a way to implement it, given the budget that has been set for the system. It adds a nice touch to the sentry gun without much overhead so hopefully it will find its way back in.

Lastly, some extra software features failed to make it into the final design of the ATS. The reason for this is because only so many could be implemented give n the time for the project, so some of the features had to be left out. Included in the design, color detection and facial recognition (for the manual control) add an extra layer to the project providing a more decision based system than without. One of the features that the team chose not to add was facial recognition of targets due to the challenge of recognizing faces at such long range. Instead the team came up with the idea of using it as a security measure for unlocking the ability to manually control the sentry gun.

Overall, the ATS has quite a few cool features added to its core concept of an automated sentry gun, but some others did not make it into the design. The main reasons for leaving some features out are time constraints, but for the range finder it is more of a budget issue rather than a time or other issue. The group wishes it could use the range finder as it enhances the projects capabilities significantly but just costs too much. The rest of the features were left out due to time constraints and replaced by other features that added similar benefits to the design. In the end, some of these features may make their way back into the project if there is extra time left at the end of the second semester, but if not they could be used for future improvements if the group decides to continue working on the ATS after senior design is complete.

10.0 Works Cited

1. 2011 US Federal Budget. Web. 03 Dec. 2011. <http://www.usgovernmentspending.com/welfare_budget_2012_4.html>.
2. MEADS International Inc. “MEADS Conducts Successful First Flight Test At White Sands Missile Range”. Web. 01 Dec. 2011. < http://www.meads-amd.com/>.
3. *The Sentry Project*. Web. 03 Dec. 2011. <http://www.paintballsentry.com/>.
4. "Paintball Targeting System." *Department of EECS, UCF*. Web. 03 Dec. 2011. <http://eecs.ucf.edu/seniordesign/fa2007sp2008/g11/>.
5. "Home." *G8 SENTRY GUN*. Web. 03 Dec. 2011. <http://eecs.ucf.edu/seniordesign/fa2008sp2009/g08/index.htm>.
6. “Autonomous Turret.” *Department of EECS, UCF*. Web. 01 Dec. 2011. <http://eecs.ucf.edu/seniordesign/fa2010sp2011/g17/index.htm>.
7. Microchip. “PIC16(L)F1503 14-Pin Flash, 8-Bit MCU Data Sheet”. Web. 01 Dec. 2011. <http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en553475>.
8. Texas Instruments. “MSP430F51”. Web. 01 Dec. 2011. <http://www.ti.com/lit/ds/slas619c/slas619c.pdf>.
9. Atmel Corporations. “SAM3S1A”. Web. 01 Dec. 2011. <http://www.atmel.com/dyn/products/product_parameters.asp?category_id=163&family_id=605&subfamily_id=2127&part_id=5001&ListAllAttributes=1>.
10. Expresspcb. “How ExpressPCB Works”. Web. 03 Dec 2011. <http://www.expresspcb.com/ExpressPCBHtm/HowWorks.htm>.
11. Advanced Circuits. “Full Spec 2-Layer Designs”. Web. 03 Dec 2011. <http://www.4pcb.com/index.php?load=content&page_id=130>.
12. Servocity.com. “How do Servos Work?“ Robotzone. Web. 02 Dec 2011. <http://www.servocity.com/html/how do servos work .html>.
13. Futaba-rc.com. “The Significant Operational Advantage of a Digital Servo“ Futaba. Web. 02 Dec 2011. <http:// http://www.futaba-rc.com/servos/digitalservos.pdf>.
14. Plumecreek.com “APA Plywood Specifications. Web. 02 Dec 2011.<http://www.plumcreek.com/Portals/0/downloads/productInfo/Y510.pdf>.
15. Amazon. “Logitech HD Webcam C310”. Web. 01 Dec. 2011. <http://www.amazon.com/Logitech-960-000585-HD-Webcam-C310/dp/B003LVZO8S/ref=sr_1_1?ie=UTF8&qid=1322785488&sr=8-1>.
16. Floyd Bell Inc. “TMC-86-530-W”. Web. 01 Dec. 2011. <http://www.floydbell.com/products/specifications/TMC-86-530-W>.
17. Best Buy. “Insignia 2.0 Stereo Computer Speaker System”. Web. 01 Dec. 2011. < http://www.bestbuy.com/site/Insignia%26%23153%3B+-+2.0+Stereo+Computer+Speaker+System+(2-Piece)+-+Black/9402283.p?id=1218100583100&skuId=9402283&st=Insignia 2.0 Stereo Computer Speaker System &cp=1&lp=1>.

18. Radio Shack. “12V/1.3Ah Sealed Lead Acid Battery”. Web. 01 Dec. 2011. <http://www.radioshack.com/product/index.jsp?productId=2103438>.
19. Total Power Solution. Tenergy 12V 1400mAh battery pack” Web. 01 Dec. 2011. <http://www.all-battery.com/12v1400mahnimhbatterypackwithbareleadsforrcaircraftandminirobots.aspx>.
20. Craig F. Bohren (2006). Fundamentals of Atmospheric Radiation: An Introduction with 400 Problems. Wiley-VCH. ISBN 3527405038. Web 01 Dec. 2011 <http://books.google.com/?id=1oDOWr_yueIC&pg=PA214&lpg=PA214&dq=indigo+spectra+blue+violet+date:1990-2007>.
21. LasersMan. “5mW 650nm Red Laser Pointer Pen White”. Web. 01 Dec. 2011.<http://www.lasersman.com/red-laser-pointer-seven-5mw/>.
22. lastpointerpro. “150mW 405nm Adjust Focus Blue-violet Laser Pointer Pen with Battery”. Web. 01 Dec. 2011.<http://www.laserpointerpro.com/150mw-405nm-adjust-focus-blueviolet-laser-pointer-pen-with-battery-p-523.html>.
23. lastpointerpro. “5mW 532nm Mid-open Green Laser Pointer”. Web. 01 Dec. 2011.<http://www.laserpointerpro.com/5mw-532nm-midopen-greenlaser-pointer-p-343.html>.
24. OpenCV. “OpenCV 2.3 Documentation” Web. 02 Dec 2011. <http://opencv.itseez.com/>.
25. Murphy, Kevin et al. “Object Detection and Localization Using Local and Global Features.” Web. 02 Dec 2011. <http://people.csail.mit.edu/billf/papers/localAndGlobal.pdf/>
26. Chesnokov, Yuriy. “Ultra Rapid Object Detection in Computer Vision Applications with Haar-like Wavelet Features.” Web. 02 Dec 2011. <http://www.codeproject.com/KB/audio-video/haar_detection.aspx/>
27. Wauthier, Fabian. “Motion Tracking in Image Sequences.” Web. 02 Dec 2011. <http://www.cs.berkeley.edu/~flw/tracker/> Sept. 08, 2011.
28. Maureen Furnis “Motion Capture” Web. 02 Dec 2011. <http://web.mit.edu/comm-forum/papers/furniss.html>
29. Greg Welch and Gary Bishop Dept of CS University of North Carolina “An Introduction to the Kalman Filter” Web. 02 Dec 2011. <http://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf>
30. Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer Siemens Coperate Research “Kernel-Based Object Tracking” Web. 02 Dec 2011.
31. Emami, Shervin “How to Detect the Color of a Person’s Shirt.” Web. 02 Dec 2011. <http://www.shervinemami.info/shirtDetection.html/> Sept. 13, 2010.
32. Utkarsh. “Tracking Colored Objects in OpenCV.” Web. 02 Dec 2011. <http://www.aishack.in/2010/07/tracking-colored-objects-in-opencv/>
33. Ragib Morshed Pomona College “Face Recognition in the Real World: A Compressive Sensing Prespective,” April 29, 2009 – Facial recognition research on both the front and profile of a face. Web. 02 Dec 2011.
34. Robert W. Frischholz “Face Detection Techniques” Web. 02 Dec 2011. <http://www.facedetection.com/facedetection/techniques.htm>.

35. Paul Viola and Michael Jones "*Robust Real-time Object Detection*," July 13, 2001. – Discusses face detection and the Viola Jones Facial algorithm. Web. 02 Dec 2011.
36. OpenCV documentation on facial detection Web. 02 Dec 2011. <http://opencv.willowgarage.com/wiki/FaceDetection>.
37. Paul Viola and Michael Jones "*Rapid Object Detection using a Boosted Cascade of simple Features*" – boosted cascades. Web. 02 Dec 2011.
38. Kazuhiro Shimizu and Shinichi Hirai Dept Robotics, Ritsumeikan University. IEEE xplore "Implementing Planar Motion Tracking Algorithms on CMOS+FPGA Vision System" January 15, 2007
39. Aforget.NET. Web. 02 Dec 2011. < http://www.aforgenet.com/>.
40. Servocity.com. "HS-81 Micro." Robotzone. ." Web. 02 Dec. 2011<http://www.servocity.com/html/hs-81 micro.html>.
41.Servocity.com. "HS-805BB Micro" Robotzone ." Web. 02 Dec. 2011<http://servocity.com/html/hs-805bb mega power.html>.
42. Servocity.com "HS-815BB" Robotzone. ." Web. 02 Dec. 2011. <http://servocity.com/html/hs-815bb_mega_sail_arm.html>.
43. Servocity.com. "Shipping." Robotzone. Web. 02 Dec 2011. <http://www.servocity.com/html/shipping.html>.
44. Hobbyhorse.com. "Shipping Information." Hobbyhorse. Web. 02 Dec 2011 <http://www.hobbyhorse.com/shipping.shtml>.
45. Hitecrcd.com. "Servo FAQs." Hitec. Web. 02 Dec 2011. <http://www.hitecrcd.com/support/faqs/faqs/servos/general-servos/index.html>.
46. National.com " LM340/LM78XX Series 3-Terminal Positive Regulators." Web. 02 Dec. 2011. <http://www.national.com/ds/LM/LM340.pdf>.
47. National.com " LM2576/LM2576HV Series Simple Switched 3A Step-Down Voltage Regulator." ." Web. 02 Dec. 2011. <http://www.national.com/ds/LM/LM2576.pdf>.
48. TI.com. "TL084 Operational Amplifier." . Web. 02 Dec. 2011. <http://www.ti.com/lit/ds/symlink/tl084.pdf>.
49.National.com " LF351 Wide Bandwidth JFET Input Operational Amplifier." ." Web. 02 Dec. 2011. <http://www.national.com/ds/LF/LF351.pdf>.
50. TI.com. " MSP430G2452 Datasheet." Web. 02 Dec 2011. <http://www.ti.com/lit/ds/symlink/msp430g2231.pdf>.
51. ermicroblog. "Building your own Simple Laser Projector using the Microchip PIC12F683 Microcontroller". Web. 01 Dec. 2011 < http://www.ermicro.com/blog/?p=1622>.
52. Amazon "Co2 Coil Hose" Web. 02 Dec 2011. <http://www.amazon.com/JT-Tactical-Remote-Coil-Hose/dp/B000FGCYU8>.
53. Amazon "RPS stinger paintballs" Web. 02 Dec 2011. <http://www.amazon.com/s?ie=UTF8&keywords=rps%20stinger%20paintballs&rh=i%3Aaps%2Ck%3Arps%20stinger%20paintballs&page=1>.
54. Zephyr Paintball. Web. 02 Dec 2011. <http://www.zephyrpaintball.com/product/PT-DIAB-HEAT/Diablo-Heat-Paintballs---2000-Count.html>.

55. Microsoft “Visual Studios 2010”. Web. 02 Dec 2011. <http://www.microsoft.com/visualstudio/en-us>
56. cadsoftUSA “EagleCAD” Web. 02 Dec 2011. <http://www.cadsoftusa.com/>.
57. Microchip “MPLAB X IDE” Web. 02 Dec 2011. <http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en019469&part=SW007002>.
58. TortoiseSVN “Subversion Control”. Web. 02 Dec 2011. < http://tortoisesvn.net/>.