University of Central Florida

STATS
Self-Targeting Autonomous Turret System

Senior Design Documentation

Elso Caponi

Michael Lakus

Ali Marar

Jonathan Thomas

Table of Contents

Table of Figures

Table of Tables

Section 1: Executive Summary

The 2014 United States federal budget was $1.1 trillion with over half of that, $573 billion allocated for military defense.[1] The U.S. budget for defense is equal to the next ten highest countries' combined. Needless to say, security and defense is a major structure in the United States of America. Autonomous sentry guns provide great protection without having any allies needed in the middle of the action. Unmanned automatic defense systems are a key area of research for the future safety of the country. Not only for military applications, autonomous turrets can also be utilized for the home and aid in keeping intruders from important possessions and family. The goal of this project is to take the design of automatic sentry guns and improve on what others have done before us.

The self-targeting autonomous turret system, or STATS, is a camera based weapon system that uses software to locate and attack a moving target. It will encompass a combination of hardware and software to match a mounted airsoft weapon's point of aim to the located target in the camera's view. Different from most turrets out there, this will all be controlled from a portable tablet device the user can move around wirelessly. The sentry gun which will have the ability to scan its field of view and fire automatically at moving objects. It will also include the option for manual control and firing. While in manual mode a real-time stream will display on the tablet. The user will be able to see the field of view that the gun sees. A four-way directional pad displayed on the tablet screen the user will be able to move the gun in any direction. A large fire button will also be on screen. Pressing the button will fire the gun, continuing to fire until released.

The turret gun will be connected wirelessly to a Windows tablet device. Using Processing language-based user interface and Arduino IDE, the tablet will detect movement, track objects, adjust the aim of the gun, and fire at a moving target. The user interface will display a live video stream that will act as the eyes of the gun. While in automatic firing mode the system will recognize a moving object, calculate its centroid, follow the target, and fire until the assailant is neutralized or out of site. By pressing a button on the tablet screen the turret will switch into manual control mode. Displayed on-screen is a directional pad the user can move the gun in any direction they please. With the live video feed streaming to the tablet the user can fire at any object with pinpoint accuracy.

The gun included in the project will be a lightweight, automatic airsoft gun. Atop the gun will be a warning siren, to signal incoming threats with flashing lights and loud noises. The gun will be connected to two servo motors that will allow it to move in both the X and Y plane in order to follow a target as it progresses across the camera's field of view. A third, smaller servo will be applied for the trigger pull of the gun. The gun and servos will be connected to a hand-built base, made up of mostly wood. The base will allow for appropriate movement of the gun in all directions including ranging from -45 degrees vertically below the horizon to 45 degrees above and a full 180 degrees horizontally. There will be a combination of internal batteries and AC-DC electricity powering the entire system. The video camera will stream a live video feed to the tablet by means of XBee

Wi-Fi modules connected to the camera and tablet. The tablet will send the signals to the PCB to move the servos via XBee 802.15.4 modules.

All of the elements including the gun, PCB, tablet, and programming software must not only be controlled individually, but in the end combined into one system. The following document covers the requirements, research, design, and testing of STATS. For every individual component of this project, there will be detailed information relating to every aspect pertaining to the whole turret project.

Section 2: Project Requirements and Specifications

2.1 Project Motivation and Goals

The motivation for this project, at its core, was developing a defense system that has real world implementations using the most current technology. Because of the group being split between two students studying electrical engineering and two studying computer engineering, it was important to choose a project idea containing an even blend of both hardware and software. The initial pitch appealed to each group member, with inspiration and concepts growing from the idea shortly after. With this accomplished, each member was given individual responsibilities for a sense of proprietorship over the portion tasked.

The central drive for the project was split between personal motivation and academic components. Personal motivation included an enthusiasm to develop a better understanding of not only the process of design, but a more profound knowledge of the components within the project. The ability to begin applying collegiate work to a design and receive feedback is an incentive to continue move forward towards completion. Going through a process of forming an initial idea to developing a prototype is a goal in itself, broken down into several subsections. Overall, it can be said without waver that each group member had a personal motivation for being part of a team bringing an idea into fruition. On the academic side, the senior design project is a gateway and a milestone. Aside from the apparent fact that this was a required assignment, the project in a sense gives engineers a perception of what is to come. This project is a milestone, bridging the gap between academic completion and the start of a career. It stimulates a drive within, knowing this project will have a meaningful impact.

The primary goal of the project was to develop a fully functional prototype which allows any user to quickly grasp the fundamental controls. The user interface was simple in design and intuitive. Because of the project being a defense turret system, it was agreed upon to have the physical design be militaristic in nature. The project, however, was not without boundaries. Secondary goals include budget and quality, which go hand in hand. Working with limited funds, the goal was to achieve optimal performance capabilities of the system using components that would keep the project budget within bounds. The prototype system was not to look aesthetically inexpensive either. Because of this, researching and testing became very important for choosing components that not only performed to the requirements, but also conformed to physical design of the system and were within budget.

Overall, the goals for the project were made to tangible, but challenging. Admitting, the sentry turret system is not supposed to be an original idea; it has been done many times over. The design process was not to be ‘reinventing the wheel,’ but rather improving upon an existing model while adding unique modification during development.

2.2 Fire Control

Fire control requires the use many subsystems within the sentry turret. In order to have the system firing properly, the microprocessor will send a signal to the appropriate servo/actuator, which will be located at the trigger of the gun. When ready to fire, the servo/actuator will pull back the trigger of the gun, thus firing the weapon.

The servo/actuator will have multiple firing modes which the operator can select. This would be semi-automatic fire and automatic fire. For semi-automatic firing, additional commands will be needed for the system to function correctly. This is due to the constant release of the trigger after every round is fired (semi-automatic fire requires the trigger to be pulled and released after every round). Thus every round would call for the servo/actuator to pull, release and repeat. Automatic firing simplifies this issue considerably. This firing mode utilizes a single trigger pull-and-hold to continuously fire rounds in a quick succession of one another. For this mode to function properly, the microcontroller will need to send two signals: one to start firing (pull the trigger back) and one to cease firing (release the trigger back to its original state). These two signals can be sent when a target enters into the field of view of the sentry turret, and when the target leaves the field of view. For the purpose of this project, the sentry turret will be programmed to only use automatic firing. The reason for this decision is to increase the hit/miss ratio of the system. If semi-automatic firing is utilized, the chance of a target hit is drastically reduced. This is directly due to RPM of the weapon, or rounds per minute. Semi-automatic firing will increase the interval time between rounds, while automatic fire will decrease this interval. Simply put, the more rounds that can be put down range in an allocated amount of time gives the system a higher chance of hitting its target. To show a simple breakdown of how this works, suppose a target is in the field of view for 3 seconds. First, the gun will fire semi automatically, with the ability to fire 6 rounds in the three seconds. Then, the gun is switched to fully automatic fire (with stock rounds-per-minute being 360 RPM). In the three seconds the target is in site, the gun will be able to fire 18 rounds (3s * 360 RPM / 60s). In any case, it is proven that there will be a higher hit percentage on fully automatic fire.

The firing control requirements and specifications can be broken down into three separate sections being auto fire control, manual fire control, and tablet firing control (including tablet interface design for firing). Each section will be partitioned into two separate divisions being the actual requirements the system must meet within certain parameters and specifications on how the requirements met and implemented.

2.2.1 Automated Fire Control

The first division, being the 'automated fire control,' will allow the sentry turret to perform all necessary operations without an operator present. This mode will be useful when the user is not present, but requires the sentry turret to maintain an active state to track and fire at its target(s). For automated fire control requirements, the sentry turret will have to locate targets within its field of view. When a target is detected, the system will automatically respond by engaging said target. However, before any rounds are fired, the sentry turret will be required to analyze the target to identify its state: either friendly or hostile. This can be determined by specific color or radio frequency identification, RFID. Once faction is known, the system will fire or be passive. Tracking and identifying targets will be made possible by software downloaded to tablet. From the camera, the data will be processed by the tablet and sent wirelessly to the microcontroller where the servos will be aligned to the correct coordinates. Therefore, the system is performing its primary calculations from the tablet's processor, and then the coordinate data will be relayed to the on board microcontroller to move the servos and fire the gun. The camera feed will also be stored within the tablet's internal memory in the form of a sentry turret status update.

For automated firing to work properly, this firing mode will mostly bypass the use of the tablet in terms of user interaction since the tablet will not need to wait for a command in order for the system to fire. Therefore, the tablet can be left untouched, and the system will track properly, hence the automated mode. As said in the requirements, the sentry turret will also have to calculate whether to fire or not based on the type of target. This process is initialized once the camera mounted to the main housing gains sight of a target. Using the tracking software, the video will then be deciphered by the tablet processor. Once locked on to the target, friend or foe can be identified by two different methods, color and RFID. If the target is made up of specific color, the processor will decipher the image and send the appropriate signal to the servo motors to fire or hold fire. For RFID, the system will fire if a specific frequency or small frequency range is not received. However, if the target holds the correct frequency, the sentry turret will remain inactive until the certain frequency is changed, turned off, or out of range. RFID will differ from color identification significantly since the system will have no need to utilize the camera to detect radio frequency. Instead, an RF reader will be installed to the system to pick out the certain radio frequencies no further than 50 meters away.

As part of the automated fire control requirements, a wireless connection is necessary for the sentry turret update. As the software on the tablet is in the process of identifying a target and performing calculations, the user will be able to observe the system running in automated mode on the tablet's screen. To let the user know when the turret system is tracking, the tablet will send off a warning signal when a target is in view. The warning will consist of a flashing screen and loud beep (or custom sound). The type of recording can be customized by the user. For live feed or video streaming features, bandwidth limitations will have to be taken into consideration. These additional features sent to the tablet will be ongoing in conjunction with the fire control.

2.2.2 Manual Fire Control

As the counterpart to automated firing control, manual fire control does require an operator to be present. This fire control will still utilize a tablet in conjunction with the microprocessor control inside the sentry turret housing. The difference is the tablet will now support hands on interaction for firing commands. The manual fire control scheme will allow the user to have complete control over the sentry turret, and will be discussed in more detail in the following section. This includes the ability to rotate the camera both vertically and horizontally and fire the gun. Having manual control of the sentry turret lets the user override the base commands the system would normally follow when in automated fire control. The operator will be able to move the turret and fire anywhere within the range of the servos. Manual mode will require some coding and the state of certain components in the system to be modified. Components requiring a change include the RF reader and processor operations for video feed. The coding libraries for tracking will require a separate library for manual control to allow for user input. Overall system operation will remain the same; the sentry turret will have identical range of motion and firing capabilities.

One firing control component requiring change is the RF reader. The reason this component requires change is because user input commands may conflict with built in commands. This conflict can be explained by a specific scenario where firing control must be modified. In this scenario, the operator is utilizing manual control. As a target approaches, the RF reader on the main system detects the correct radio frequency and does not fire. However, there is the possibility of the user deeming it necessary to proceed with firing, but is unable to because of the preset command for the RF reader. The remedy in manual firing mode will be to toggle the RF reader off to avoid user and system complications.

As stated previously, operation performed in manual mode will not be found in the automated mode. However, both modes will have very similar signal paths for the turret system to function. A wireless connection from the tablet to the microprocessor on the custom PCB will have to be maintained. The method of feeding video to the tablet will also require adjustments from automated firing mode. In automated mode, the tablet is used more as an alert and viewing device. In manual mode, the tablet has a more active role. When switched to this mode, the system camera will immediately begin a live feed to the tablet so the operator can see exactly what the sentry turret sees in real time. Now, the operator will be able to utilize the touch screen on the tablet to move the servos in real time as well. With respect to manual fire, the wireless video feed from the mounted camera to the tablet will have certain specifications, since this will directly affect firing control. The speed of the processor and bandwidth limitations have to be considered as this will be a source of cause for the video stream to lag behind when viewed on the tablet. As delay is decreased from the main system camera to the tablet, this will increase the effectiveness of manual mode hit/miss percentage. Therefore, it is optimal to limit the delay time. The user will not have to preemptively judge the position of the target as much when firing the sentry turret.

To implement manual control into the system and to have it work effectively for video streaming, a stable wireless connection must be maintained between the tablet device, the camera, and the PCB on the sentry turret. Any disruptions with the wireless signal will cause difficulties for the operator. For firing control the work efficiently, the effective range of the wireless connection will be at a minimum of 10 meters. There can be the likelihood that system operations may continue to work at increased ranges, but the user will run the risk of spotty connections due to environment and manmade obstacles.

Therefore, firing control will operate best when at a closer range will minimal disruption for the wireless signals to travel between the tablet, camera and onboard PCB.

In review, the main differentiating factors between firing modes are the way in which the tablet is utilized and the tracking software's command libraries. Besides this, both automated and manual mode commands will originate from the tablet device once the image from the camera is processed. The separate libraries for automated and manual fire control are tailored to each mode to account for any discrepancies between the two. The manual mode process will have a more complex design because of the additional commands the tablet must decipher from the operator using the touchscreen. Data will be sent from the camera to the tablet where the operator will choose a command. Then, the command signal is sent from the tablet back to the system PCB to perform the necessary function. In manual mode, this loop of information is continually updated to give the tablet an up-to-date picture of what the camera is viewing.

2.2.3 Processor Targeting control Requirements

The main sub-system of the turret will be the targeting control. The motor control system will be controlled by the Atmel mega 328 boot loaded with an Arduino IDE. The motor controller will receive signals that include objects position along with the anticipated position all sent wirelessly using XBEE RF technology. The servo motor controller will then move to the specified location using the pan-tilt servo motors. It is important to note that the motor controller will pan first then tilt. This seems to be a more efficient method in terms of maximizing the accuracy. This is due to the nature of moving sideways rather then moving up and down. Once the target is initialized a third motor will pull the trigger on the firearm. An interrupt will be sent back to the main computer vision processor indicating the object has been shot at. If the object still is still within range then the motor controller will move to the new position indicated by the tablet, fire and return an interrupt. This will continue to occur until the object is cleared form sight. The motor controller will always have continuous feedback to up keep with the target. This operation will only occur in the autonomous mode.

2.3 Weapon Systems

The housing design will allow for the sentry turret to utilize a multitude of platforms for target tracking. Anything weighing less than approximately 10 pounds can be mounted without sacrificing performance. Therefore, if the user prefers a non-projectile platform, a laser, flashlight, or something of that like can be modified to fit. For this project, it was clear that either a paintball gun or airsoft gun would be implemented into the design because of their availability, ease of use and target-hit-verification. However, these two weapon options have some differentiating factors that led the group must make a decision on.

2.3.1 Paintball Guns

Paintball guns (also known within the paintball community as a 'marker' because of the paint used to mark the target) are primarily broken down into two categories based on functionality. These categories are mechanical and electronic markers. Just as they sound, a mechanical paintball marker uses only physical components to fire a paintball, where an electronic paintball marker is aided by an onboard PCB.

Mechanical paintball markers are robust in design and can withstand natural elements better than electronic markers. An advantage of mechanical markers is in their ease of disassembly and reassembly. Troubleshooting can usually be easily fixed by user modification. The drawback is the firing mode. Because of the absence of an onboard PCB, the majority of mechanical markers are limited to semi-automatic firing where a trigger pull and release would be mandatory after each round. This specification would eliminate this model from the design since the requirements call for automatic firing capabilities for an increased hit percentage.

Electronic markers are more sophisticated than mechanical. The onboard PCB has a variety of useful functions for user customization. Most electronic markers come with an LCD screen showing gun status, and are pre-programmed with semi-automatic, 3-round burst, and automatic firing capability. This automatic firing mode would be ideal for this design, so this type of marker can be implemented. It can be noted that the gun can be fired without using the trigger since valve release is controlled by an electronic signal. Bypassing the trigger would let the system run only using two servos instead of three. However, it is only limited to the paintball marker and would cause redesign if other gun models were to be mounted. For the purpose of this design, the marker would be fired by using the third trigger/actuator on the trigger.

Both models can be powered by the same fuel; the most popular being carbon dioxide or compressed air. This is one of the drawbacks of using a paintball marker in this design. To power the marker, a tank of carbon dioxide or compressed air is required to be attached to the marker, or seated next to the marker (this can be accomplished by running a short extension hose from the tank to the marker). The tanks come in different volumes depending usage, so size is a factor. If the user wants the gun to fire for a longer period of time, a larger and heavier tank would have to be attached. Attaching the tank directly to the marker would not be an option since this would increase the weight of the weapon platform movement. Consequently, the servos may become strained due to the increased load and function poorly. It may even require higher torque servos which would not be a possibility for the design process. The only useable option would be to mount the tank to the side on the housing by use of clamps. Then the extension hose could link the tank to the marker. This feature is not included in the prototype because of the additional costs, but can easily be added in a later version of the model.

Additional factors to be taken into consideration are the paintball hoppers and the paintballs. Because of the larger diameter of a paintball to an airsoft BB (17.272mm compared to 6mm), storage is not as concealed and requires an additional component known as the ‘hopper’. This device is usually connected to the top of a paintball marker and will feed the paintballs to the gun. The additional weight of this component is not nearly as much as the air tank, but is still taken into consideration when selecting the servo motors. The combined weight of the hopper and paintballs is significantly more than a magazine filled with BBs. Therefore, servo selection will be based on a fully loaded paintball hopper since this is the heaviest platform the system will accept.

2.3.2 Airsoft Guns

Airsoft guns, as with paintball guns, are broken down into the two same categories, but function slightly differently due to the ammunition caliber and weight. The main difference is airsoft guns can fire BBs using internal air compression. This immediately eliminates the external tank required for paintball markers. Additionally, because of the BB size, an airsoft gun can hold more BBs with a smaller magazine. This cuts the need to a hopper to be added. Instead, airsoft guns use magazines that are functionally identical to real firearms. This not only adds to the realism of the airsoft gun and this design, but will make the overall sentry turret more concealed since a large hopper will not be protruding from the top. These magazines can hold upwards of 300 BBs, while hoppers are usually maxed out at 200. The increased capacity will let the sentry turret fire for longer duration and be reloaded less frequently.

Because a BB is smaller and weighs less, it can travel the same distance and speed as a paintball (usually even further and faster) using less air to propel it. Though there are positives and negatives to using BBs over paintballs, the pros outweigh the cons. A BB not being able to ‘mark’ a target can be considered a con, but is more scenario-specific. The only real negative factor is a BB’s weight. But this can be a two sided argument. The significant drop in weight compared to a paintball makes airsoft BBs much more susceptible to wind, rain and light foliage. Since both a BB and paintball can accelerate at the same rate, the mass is the only constant that is different. If an external force were to act upon both, inertial law governs that a BB would stop before a paintball. However, airsoft guns contain a simple component that gives the BB a straighter trajectory after leaving the barrel. This device is called ‘hop up’. Hop-up is no more than a small piece of rubber or composite material situated at the start of the barrel; within the airsoft gun. This device affects trajectory by putting a backspin on the BB as it travels through the barrel. This method is comparable to a pitcher throwing a baseball. As the BB leaves the barrel, the backspin will cut through the air and try to counteract the effects of gravity. This lets the BB travels straight over increased distances with most airsoft guns out to an effective 100 feet. In addition to BB hop up, airsoft guns allow for higher muzzle velocities. An average airsoft gun’s muzzle velocity can range anywhere from 250 to over 500 feet per second (FPS) depending on the force constant of the spring used to push and compress the air. On the other hand, paintball guns can reach above 300 FPS but it is impractical due to the amount of air required from the tank per shot. Therefore, airsoft BBs not only have an advantage in distance, but muzzle velocity as well. Next is to look at types of airsoft guns capable of firing at these velocities.

The mechanical, or pump action, airsoft guns are only semi-automatic fire capable. As with the mechanical paintball markers, these will be excluded from the design because of the lack of an automatic firing mode. Aside from the firing mode, mechanical airsoft guns require the user to cock the gun. The additional manual work excludes this option entirely. The action usually consists of the user pulling back a lever to compress a spring. Simultaneously, the airsoft gun's cylinder chamber is filled with air. Once done, a trigger pull will release the spring and force a piston down the air-filled chamber. This sudden compression will push the air towards the BB, propelling it out of the barrel. As intricate of a design it may be, electronics airsoft guns function similarly. The difference is the manual work is condensed to only pulling the trigger. The airsoft gun's internal motor will do the rest. Therefore, the choice of weapon for design implementation will be the automatic electric gun, or AEG. The AEG category contains several different real world models of gun designs, ranging from pistols to sniper rifles. However, for project requirements, the best choice weapon will be compact, lightweight, and exhibit good muzzle velocity.

Airsoft gun batteries, however, require more attention in selection. Since the airsoft gun is only regulated by its connected battery, giving the AEG's internal motor too much power can damage the components. On the other hand, not enough power will likely cause the AEG to fire poorly and increase the chance of a BB being jammed in the barrel. To work around this issue, it is important to review battery specification in conjunction with the AEG of choice. Airsoft batteries are rated in terms of voltage, V, and their storage capacity, mAh (milli-ampere-hours). The batteries rated voltage will determine how the AEG's internal motor will be able fire in quick succession. Therefore, an increase battery voltage will affect the gun's rounds per minute (RPM), and not its feet per second (FPS). The battery specification that is most important for this design is the rated storage capacity. Larger capacity batteries will allow for longer durations of testing before being charged again.

Based off of an AEG-modeled prototype sentry turret system, the airsoft gun of choice will have to meet certain requirements. The airsoft gun for this design should fit, but is not limited to the parameters in the table below.

Maximum Dimension (l x w x h) in inches	(24 x 3 x 12)in
Maximum Weight (lbs.)	10 lbs
Minimum Muzzle Velocity (FPS)	280 FPS
Minimum Battery Voltage (V)	7.2 V
Minimum Battery Capacity (mAh)	1200 mAh
Minimum BB Effect Range (ft)	25 ft

Table 2.1 – Airsoft Gun Requirements

The category of gun fitting the requirements best is the submachine gun, or SMG. The submachine AEG is a compact gun the will cut down on the form factor of the sentry turret considerably. Most SMG models are designed with close quarters combat in mind. Thus, this platform is an agile and lightweight weapon that can maintain comparable accuracy to a gun with a longer barrel. Because of these specifications, an SMG modeled electric airsoft gun will work best with the system.

2.4 Tablet Device

The tablet will run an application that will display a live video feed from the camera atop the turret. Also in the app will be buttons to allow the user to manually override control of the gun, giving the opportunity to aim the gun and fire at any target. The tablet must have the capability to communicate wirelessly with the camera and the processor. Meaning Wi-Fi, capability, Bluetooth capability, or at least one USB for wireless USB transmission.

A portable tablet with a large screen of eight inches or greater will be best for viewing the live feed from the camera atop the turret. A smaller sized, cheap tablet of five inches will operate sufficiently but with the goal of manually triggering via a portable device, the larger the screen the better. This will enable for more detail to be seen by the user which will assist in a more accurate shot.

2.4.1 Tablet User Interface

The idea of implementing a tablet into the turret design was for two reasons. The first is for portable video feedback of the gun, so the user can see exactly what the gun sees and where it is shooting. The second is for allowing manual override of the gun. A phone-sized device of around four inches will be serviceable but not ideal for viewing all the objects in the camera’s view. Thus a tablet with a large screen of a minimal eight inches will be required. With the application open it will be streamed a live feed from the video camera. This stream will be the “eyes” of the turret, showing to the user everything in its field of vision. The video will take up most, if not all of the real estate on the tablet screen, depending on the scaling ratio of the tablet screen size to the video feed. If the video feed is 4:3 then it will scaled and fit appropriately to the tablet screen without distortion.

With the video taking up nearly all of the screen, the buttons for controlling the manual movement of the gun will be semi-transparent. This will allow for maximum video size yet still able to see the manual override buttons without obstructing the video from the user. In the bottom left will be a directional pad with the directions of up, down, left, and right. This pad will allow adjusting the camera on the top of the turret to move in the direction the user selects when pressed. The user will have the option of holding in a direction, letting the gun move until the button is let go or pressing once for a minor increment. In the bottom right will be a simple “Fire” button. This will shoot the gun when pressed. Like the directional pad, a single tap will do a minor action, in this case shooting once. Holding in the fire button will allow for a continuous shooting until released. Finally, an indicator of where the airsoft pellets will land will be in the middle of the screen. The indicator will be a thin red crosshair so the user can line up their target with pinpoint accuracy.

2.4.2 Tablet Control Interface

Tablet control interface will have a role in both the automated and manual functions of the sentry turret operations. The automated firing control interface will not include all the features of the manual firing mode. However, automated mode will possess unique features that will not be found elsewhere.

As the sentry turret is in automated firing mode, the tablet will have a simplified working interface, but it will not allow the user to interact with the PCB and servos. In this mode, the user will also not have access to the main system camera, nor will they have a fire command button. Instead, the tablet will act on the lines of a tracking and alert device. As a target is in the camera's field of view, the tablet can alert a user of the activity. This alert can be activated by the tablet by vibration, sound, a flashing screen, or a combination of the three. This will ensure the tablet user receives the message and can take action.

Manual mode tablet interface will focus on interactive control to aid the user in engaging a target. Because of the required interaction between the user and the tablet for manual firing mode to function, the tablet firing control interface is taken into great consideration. Recorded video files found in the automated mode will be present in manual mode, but not as customizable. In its place, the tablet interface will make way for the interactive controls for camera and firing. The goal is to give the operator the best functionality possible with a simple, yet effective design. The feature set the tablet will have for firing control includes a firing button, 4-way directional pad, and a toggle between manual and automated firing modes. For design purposes, all controls will be accessible on a touchscreen-compatible tablet. This decision was made in order to streamline design, as adding a supplementary input device to the tablet for control would defeat the purpose of using a tablet and taking advantage of touchscreen technology. Additionally, any control hardware applied to the tablet runs a risk of compatibility errors and additional troubleshooting.

The firing command button layout is very simple in requirement. Whenever the user presses the button on the tablet, a signal will be sent to the PCB microcontroller, causing the servo to pull the trigger back. But more has to be taken into consideration when deciding the specifications of the fire command button.

The first specification reviewed for the button was the type of input. The two options include a toggle firing mode and push-to-fire firing mode. With a toggle firing mode, the button only has to be tapped once by the user to initiate firing. After this, the gun will continue to fire until the user taps to button again to cease fire. The other option would include a push-to-fire, where the firing would be based off touch. As long as the button is held down, the gun will continue to fire. The moment the user releases pressure from the button, firing will stop. By including automatic and manual mode shooting, this plays a factor in which button input is best suited for the operations of the sentry turret. With a toggle button specification, the user would be able to essentially fire and forget. However, this method carries more negative effects than positive as the user could accidentally leave the tablet while the sentry turret continues to fire. This poses a high risk factor and could put others in danger and potentially damage the system. Because of this, a push-to-fire design input system is implemented for the fire command button on the tablet interface. This input choice was not only chosen because of the toggle input design's flaws, but because of its positive attributes as well. The push-to-fire button interface is a smarter choice for design interface due to the fact in more closely resembles the actual firing mechanics of a gun trigger. Especially in the case of automatic firing, the gun will fire as the trigger is pulled back, and will stop firing when released. The push-to-fire design has very similar motion.

In addition to the type of firing button input is the physical location of the button on the tablet. Again, the choice is between two options. In this case, the button can either be digitally displayed, or mapped to a physical button on the tablet. For this project, it was decided to take advantage of touchscreen technology of a tablet and create a digital button for the user to interact with. The location for the fire command is located near the corner of the screen to allow the user maximum vision of the camera's field of view.

The next feature on the tablet for manual firing mode is the 4-directional control. This control feature is directly linked to the servo movement on the main system housing when in manual firing mode. The user will be able to decide which direction the servos will need to be moved based on where the target is located. The operator's job will be to simply press a direction on the tablet's screen and the servos will adjust themselves accordingly. There are four movement command signals the processor will be able to distinguish: upward movement, downward movement, left movement and right movement. These movements will be divided into and X and Y axis or horizontal and vertical plane. The user will also be aided by a center reticle for targeting. Because of the difficulty of mapping the movement to the physical buttons on the tablet, specification calls for the 4-way movement button inputs to be located on the tablet screen. The direction controls will be implemented with the touchscreen. The location of the 4-way direction input button can be either placed next to the fire command button or separated into the four corners of the screen. Both designs will still allow the user to have maximum field of view of the camera to spot targets while still maintaining functionality of the servo motors and firing control button.

The last section of the tablet firing control interface is requirements and specifications for the tablet switch control between the two firing modes: automatic and manual. The design requires a way to allow the user to easily transfer between the two firing modes without performing several tasks or clicking and pressing several buttons. To solve this complication, the tablet will have an onscreen button that will be programmed to toggle between the firing modes. The operator will have to do no more than make contact with the button on the touchscreen tablet to switch firing modes. The toggle button will always be on screen regardless of the mode and never hidden. This will guarantee the operator can switch as quickly as possible to avoid delays. The location of the button will follow suit with the other controls and be located in a corner of the screen to maintain maximum camera vision.

2.5 Video Tracking Requirements

2.5.1 Tracking Motor Control Requirements

The motion tracking begins with control of the movement of the entire turret itself. Through the use of strong servo motors the turret will be controlled in all directions by setting the two rotational motors in a pan and tilt format. The motors overall rotational distance will not need to be any more than 120 degrees because they just need to be able to rotate through the entire field of view of the camera for reasons described in the text below. The motors should have a set center point to return to after a preset amount of time has passed since the last time an object was detected in the field of view. This will improve initial tracking of new objects because no matter where they appear on the field of view they should be equal distance from the center point. The preset amount of time should allow the turret to remain near the location of the last moving object long enough to allow for the object to possibly come back into frame; thus speeding up tracking even further.

The servo motors should be of sufficient power to rotate a rather large rifle style airsoft gun at a speed of a running human object which on average is about 20 feet per second. The average weight of a full size airsoft rifle can depend on whether or not it is made of a metal housing or a plastic housing but will most likely be in the 6 to 8 pound range. This means that the motors will need to have enough torque to meet the required speed with at least 10 pounds of load on them. In order to effectively move the turret to accurately track a fast moving object; the motors will actually be required to move about twice the speed of the target in order to compensate for directional changes, target leading (for accuracy and shot placement), and multiple target acquisition. The motors will be supplied with a certain amount of voltage to control speed and rotational distance; determined again by distance and position of the target on the field of view. The third servo motor that will be used in this design will be designated to pulling the trigger of the airsoft gun. This motor does not need to be as powerful as the rotational motors because it will be performing a simple action. When the tracking system determines that it is on target the third motor will simply be sent a voltage pulse that keeps the trigger held down until the target is no longer in the field of view. The system processor will be in control of the voltages that control the servos as well as the tracking coordinates and commands.

2.5.2 Processor Requirements for Tracking

The control of the servos will be handled by the microcontroller that will be included on the custom PCB that will be built for this project. Using a software or hardware based tracking method the location of the target that appears on the video system must be sent to the microcontroller via direct line or wirelessly. While the camera system loops through at about 30 frames per second, any pixels that have changed from frame to frame will be sent to the microcontroller as coordinates for the servos to move to. The microcontroller will use pulse width modulation to send voltage signals to the servo motors in order to determine the amount of rotation needed as well as the speed required that would be determined by the speed of the object in the view of the turret.

The PCB design in this project must implement a wireless technology like Bluetooth, WiFi, or multiple other options in order to wirelessly communicate data between the camera, processor, and the tablet that will be used. The video sub system must be able to quickly communicate the location of the moving pixels on the screen to the microcontroller; then the microcontroller must determine where the servo motors must be moved to in order to align the turret with the target. The processor must have sufficient memory to store the information needed for the target tracking. Any memory size between 16kB and 32kB should be plenty for this project. The clock speed on the processor should also be high enough to allow high speed communication. The Atmega168PA or the ATmega328P have a 16kB and 32kB memory respectively and are viable options for this project.

2.5.3 Tablet Tracking and Control

A core element of this project is to include a tablet interface that allows the user of the turret to view and/or control the tracking and targeting of the airsoft gun. This will require multiple steps including a mobile application in order to link the device with the microcontroller. The output of the video subsystem must be sent to the processor in order for the tracking to work, but at the same time must be sent to the tablet in order for the user to see what the turret has in its field of view. The tablet will be in use at all times because it will have a direct feed from the camera and will also have the ability to manipulate the servo motors manually in manual mode. The manual mode will most likely have to be designed into the application that will be linked to the wireless output of the PCB. It should give the user the ability to take over control of the servos via an onscreen D pad; like found on most video game controllers. Another option could be a point and click style targeting method that would still need to use the tracking mechanism. If a D pad style control method is used then there must be a bypass used in order to turn off the auto tracking to allow the user to drive the servo motors manually and use something along the lines of an on screen button that sends a signal to the servo motor that is designated to the trigger in order to fire. If a point and click manual mode is used then it gets a little more complicated. The application would have to work along with the tracking method in order for a point and click targeting method to work. The selected area of the screen would have to be sent as the targeting area much like the moving pixels would be sent if auto tracking were turned on; then the processor would have to continue like it would in auto mode afterwards.

2.5.4 Video Subsystem Requirements

The camera that will be used must be of the lightweight and inexpensive variety to be of any use in this project. The camera must be able to be mounted onto the turret system and must have a way of connecting to the PCB in order to send video input signals to the processor so that the processor can implement the tracking software on the received images. The camera must have medium to low resolution and a relatively high frame per second ability in order for it to serve as a good motion tracking subsystem. Depending on tracking methods used in this project, the camera's location on the turret may be of importance. There are multiple ways of implementing motion tracking but some of them may require that the camera be stationary so the mounting location of the camera will have to be determined once the tracking methods are designed. Also the camera must be compatible with the user interface (tablet) that is chosen as well as the software used for tracking and targeting.

2.5.5 Laser Sight

This system will most likely need to implement the use of a laser sight for multiple purposes in the design. The functionality of a laser sight for this project will be mostly cosmetic but will also add the ability to see what the gun is pointing at from a distance without the use of the tablet interface. This will be most helpful in testing, as the laser sight can be seen on a target to test if the target is being shot at without actually using live fire.

2.5.6 Range Finding

The project will require some method of range detection so that the pan and tilt motors can be aimed accurately at the target at any given distance. It is expected that the airsoft gun as well as the camera system will be capable of shooting targets out to distances of up to 50 meters. At those distances a lightweight airsoft pellet would surely lose altitude due to gravitational pull. This is why some sort of distance measuring device must be utilized in the design of the tracking system.

2.5.7 Software Requirements

Since this project will be using a tablet as the user interface as well as the main computational source for the tracking code, the code must be compatible. If an Android based tablet is to be implemented then Android specific tracking algorithms must be implemented. Since the most common tracking methods used in similar turret designs involve OpenCV (Which is mostly windows based) it might be beneficial to use a windows based tablet since there are already a wide variety of libraries available for that source. Regardless of which type of device is selected, there must be a sufficient amount of tracking software that is capable of functioning on that operating system. Since a windows based tablet and an Android based tablet are already owned by the group members the choice of related software will most likely be between those two options.

2.6 Servos and Actuators

Servos will be the main component for positioning the turret in the correct direction. In this case, the servos will have to align the airsoft gun to the correct coordinates according to the camera's point of view. The separate firing modes have to be taken into consideration for the requirements and specifications of the servos. Manual firing mode may cause more strain upon the system as the user will have complete control over the airsoft gun's positioning. Consequently, the operator will be able to move the airsoft gun across its maximum firing angle quicker than when the system is in automated mode. Besides the differences in firing modes, the servos will perform to specification regardless of mounting option. Therefore, overall requirements for the servo motors will include speed, power consumption, size, and cost.

Before expanding on the servo requirements listed above, each servo function will be discussed. Because the servos for the turret system will not be the same depending on the role. The design of the turret calls for three servos. Two of these servos will be used for target tracking, and one servo will be used for firing control. The two servos used for tracking will be more robust and powerful in function. This is due to these servos being responsible for moving the weight of the airsoft gun to the appropriate coordinates. Therefore, the speed, torque, size and power consumption of these servos will be greater than that of the fire control servo. The firing control servo will only serve the function of pulling the trigger of the airsoft gun. In the case that something is mounted to the system that does not require a trigger pull, this servo will not have a use. Because of this servo's unique role, its design will be different than the two positioning servos. Primarily, speed is a factor. The response time for the servo to receive a signal to the actual trigger pull will be within under a second. Power also has a factor for this servo. Airsoft triggers do contain some degree of trigger pull. Therefore, the fire control servo is required to exert enough force to pull and hold a trigger effortlessly. Lastly, the size of this servo will affect how it is incorporated into the prototype design. Ideally, the form factor for this servo will be smaller than the target tracking servos. It is required that the servo be small enough to be mounted next to the trigger guard of an airsoft gun without having to perform extensive modifications to the airsoft gun itself in order for the system to fire properly.

Returning to overall servo requirements, the first includes speed. In order to guarantee an impressive hit-to-miss ratio, the delay time for the servos to position and fire will be at a minimum. The delay time will be required to be in the millisecond range. The time elapsed for the servo to move to the correct coordinates is measured in degrees per second. Therefore, if a camera has a view angle of 120 degrees, target tracking servos will be able to cover this distance in under one second. A similar rule applies for the fire control servo. Trigger pull will commence simultaneously as the tracking servos position themselves.

The next requirement for the servos is power consumption, or power required. The servos will require the most power over any other component in the design. For the turret system to be as mobile as possible, an onboard battery will be housed. Because of this design factor, it is very important for the servos to have the lowest power consumption possible without sacrificing performance. As stated before, the fire control servo will require less power. Additionally, the servos will be able to run for several hours without depleting the onboard power supply (tracking non-stop in both manual and automated modes).

The size and cost of the servos will be taken into consideration as well, but with lower priority. Understandably, servo motors not within the budget for this project will automatically be excluded since they are not practical. Reasonably priced servos with performance specification meeting the requirements are ideal. For the form factor of the servos, the size of the fire control servo has already been discussed earlier in this section. The size for the target tracking servos can be more lenient. The only requirement in regards to servo size is that there is no compatibility problems when mounted to the armature. The positioning servos will mounted to a joint arm system while providing enough space on the main platform for the servo control PCB and any additional features.

With all requirements met, the exact specifications of the servos can be expanded upon. Using the Table 2.2 and 2.3 below, the target tracking servos and the fire control servo will be within the shown specifications.

Minimum Speed	0.20 sec/90 degrees
Maximum Voltage	6 V
Minimum Servo Turn Range	90 degrees
Minimum Torque	120 oz-in
Maximum Size (l x w x h)	8in x 5in x 5in
Maximum Cost	$60 each

Table 2.2 – Target Tracking Servos

Minimum Speed	0.20 sec/90 degrees
Maximum Voltage	6 V
Minimum Servo Turn Range	45 degrees
Minimum Torque	18 oz-in
Maximum Size (l x w x h)	5in x 3in x 3in
Maximum Cost	$50

Table 2.3 – Fire Control Servo

2.7 Housing Design

Reviewing the housing design requirements and specifications for the sentry turret will ensure that the final housing model will meet the standards set for this project. The housing can be broken down into a section for the mounting platform, and a section for the main body. The stipulations for a good design will not be limited to only functionality and reliability. Aesthetics and streamlining will have a portion in the final design plan.

Before discussing the main body of the housing, requirements and specifications will be addressed for the platform mount. In order for the design to have the ability to connect to several devices, the mounting platform will be required to function seamlessly between switching devices or weapons. This calls for a universal type of mount. Seeing as the prototype will carry an airsoft gun, it is first necessary that the mount be able secure most weapon platforms (for this case both airsoft and paintball guns). It must also be noted that the mount cannot interfere with the trigger location on the weapon since a servo will be placed here and require enough room to function properly. The mount placement on the weapon will also have to be close to the center of gravity. This will ensure the aim of the weapon is not above or below the target being tracked. Therefore, the length and type of material used for the mount will be required to hold any weapon platform in place without any slipping when the servos are changing coordinates. For the design, a universal mount will be rubberized clamp. Having rubber in contact with most weapons and devices will increase grip capability to ensure that object does not slip when the servos begin moving. In addition, the rubber contact can also mitigate potential damage done to the secured object as it is subjected for lateral are vertical forces.

Besides weapon platforms, the mount will also be required to function with non-projectile devices. The mount will have the ability to hold a variety of lasers and flashlights. By over-compensating the mount to scale correctly with weapon platforms, the design will have no compatibility issues with these smaller devices.

Because of the mount working in tandem with the servo design, the servo housing will also have to meet certain requirements and specifications. As stated prototype design will be geared towards a mounted airsoft gun. Therefore, three servos will be required in the housing for the sentry turret to function properly. Two of these servos will make up the positioning system in the x and y plane. The third servo or actuator will be used exclusively for the trigger pull on the mounted weapon. The housing for the positioning servos will be integrated into an arm-like structure. Since this will be the only moving structure on the system, choice of material is important. Lightweight and durable metal or wood are the best options for building materials since servos can be mounted via screws. Metal or wood will also be able to withstand flex due from the rotational inertia caused by the mounted weapon or device.

The size of the servo mount will be as compact as possibly. The only factor posing a threat to optimizing vertical length will be the magazine from the airsoft gun. However, a magazine should span no more than 10 inches, so the servo mount system should be able to function if specified to 12 inches in height.

The first requirement for the housing is the ability for it to effectively mount or store all necessary components:

- The servo and trigger assembly
- PCB for servo control
- Onboard tracking camera
- Tablet mount
- Power Supply / AC-DC converter

Servos are to be integrated to the armature structure in the center of the system. The horizontal or x-axis coordinate servo will be positioned on the top, flat surface of the housing. Next to this servo is the PCB for servo control. This PCB will sit close to the servos to avoid lengthy wiring connections. However, it will be mounted directly behind the x-axis servo (opposite direction that an airsoft gun would be facing) to guarantee no interference when the mounted object is tracking a target. To prevent any damage done to the PCB, the design will be enclosed in a transparent case. A material able to external stress while providing internal protection and a clear view of the PCB is ideal. Therefore, either a clear reinforced ABS plastic or Plexiglas-like substance is the material of choice.

The top, flat surface of the housing will also be required to have a spot to mount the tracking camera. The camera will be as low profile as possible, and have a mounting position on the front facing end of the system. This allows the camera to have full field of view without being blocked by other components on the top of the housing.

For best mobility and decreased form factor, the housing is additionally required to carry a mounting spot for the tablet accessory and an internal storage location for a power supply and power converter. For the tablet, easy access is the first priority. The ideal spot for this mount is behind the PCB enclosure on a slanted platform. The degree of which the platform is slanted will match the viewing angle of a user standing directly behind the turret system. Having the tablet mount on the backside of the system also lets the user access the tablet without ever stepping in front of the camera and run the risk of injury due to accidental targeting. In addition to the tablet mount, the internal housing of the project will include enough space for multiple power supplies and any additional accessories or features the user would add. For the purpose of this project, however, the internal housing will only require enough room for a power supply and a power converter. Both of these options will be included to give the user the ability to move the turret system without losing power. When the system is mobile, or does not have access to an external power source (this project will use a standard 120V wall power outlet), the internal power supply will be able to provide the entire system with power. This includes the servos, PCB, camera and any additional small features. When an external power source is available, the turret system will utilize its AC to DC converter to provide the components with continuous power.

To allow the user of the turret system to switch out mounts and move the housing in a timely manner, the project will be modular to an extent. A modular design will not only allow for quick tear down and reassembly, but it is also receiving of additional mounting features since they can be added and removed easily. This feature will be most beneficial when changing the weapon platform from an airsoft gun to a paintball gun. With this change, the paintball gun’s air canister is required for it to function. To remedy this issue, the main housing will include a clamp-down mechanism suitable for variable sizes of paintball canisters. If the weapon platform is switched out, the user will be able to remove the mounted canister with ease and attach the next object.

The last main requirement for the housing design before discussing specifications is the mounting base. Because in the way the turret system functions, the main housing will be subjected to forces due to servo movement. The way in which the system is secured to the ground will determine the effectiveness of the target tracking and performance of the hit-miss ratio. Accordingly, the system will contain sturdy leg mounts to stabilize the housing under movement. The design will require a 4-legged retractable mounting system that combines both stability and mobility. The legs will have rubber on the end in contact with the ground as a rubber material reduced vibrations and can grip to a variety of surfaces.

With all requirements set, specifications for the housing are inspected. From the tables below, the shown specifications will keep the design within boundaries of all requirements. The design will be mobile, lightweight and contain a relatively small form factor. Additionally, materials used will be durable and within the allocated budget.

Maximum Length	2 ft
Maximum Width	2 ft
Maximum Height	1.5 ft
Maximum Internal Volume	4.5 ft^3
Tablet Mount Slant Degrees	45 degrees
Build Material	Wooden Cage with Metal Components
Mounting Accessories	Modular
Maximum Weight (without airsoft gun)	15 lbs

Table 2.4 – Main Housing Requirements

Maximum Length	1 ft
Maximum Width	0.5 ft
Build Material	Wood / Metal
Platform Mount Material	Rubber
Maximum Weight	5 lbs

Table 2.5 – Armature & Servo Mounting Requirements

Maximum Length	2 ft
Angle from Ground to Platform	60 degrees
Leg Build Material	Metal
Leg Base Material	Rubber
Maximum Weight	2.5 lbs each

Table 2.6 – Leg Stabilizers Requirements

2.8 Power Systems

After the specifications for all the components comprised in the sentry turret system, the power systems for the project can be mapped out. The components for the project all require different power ratings and the requirements for each can be varied. Therefore the power system must have strict specifications in order to effectively meet the requirements of the system as a whole. Power systems were looked at from a mobility standpoint. The system was to be designed to be moved from location, therefore multiple power options were considered. For this project, the sentry turret system will be able to be powered by two different mode. The first mode, being an internal battery supply, will house enough power for all components on the system. The battery choice will be based mainly off the servo ratings, since these will draw the most power out of all the components. The battery will be required to be rechargeable as recycling batteries and purchasing new ones can become a costly investment.

The other source of power for the system will be an AC-DC power converter. This converter will be implemented with the system to allow for continuous power supply. With this device implemented the turret will be able to run for as long as desired and no recharging will be needed. Neither power source, the internal battery supply nor the converter, will be considered the primary battery source. Either source will solely depend on location when the sentry turret system is deployed. Therefore, if STATS is deployed in an environment where no immediate external power sources are available, the user does have the option to use the internal battery to run the system. Vice-versa, when STATS is deployed in an area near a wall power source, the power cord provided with the system can let the user plug and play.

The power cord for the converter will have brief requirements and specifications. Primarily, the cord shall be ample enough in length to reach wall outlets at a maximum of 50ft away from the main housing of the sentry turret system. This distance will be sufficient to set up STATS in the desired location and reach the preferred wall outlet. It is important to note a relationship between peak performance, and an efficient power criteria. The autonomous turret is intended to work with numerous sub-systems for peak performance. With respect to power, a controlled source must be able to supply the power with no holdup. Sub-systems include a battery monitoring system, 3 fully integrated servo motors, 2 wireless interfaces, and an audio/light warning system. Specific power requirements include:

- 2 hour operating time
- Minimal current drain across components
- Recharge Circuit/Battery Meter
- PWM support

It is estimated that the three motors combined will have a combined current drain of 1.2 Amps which will account for the most amount out of all devices. A low power microcontroller will be used to control the motors. An estimated 20 LED's will be used with a combination of a buzzer to implement the onboard audio/alarm warning sub-system. Additionally the wireless interfaces will operate using nominal current values but since the interface will be turned on at all times, it may account for a higher then estimated current gain. The Turret power will be split into 2, the onboard PCB which will be powered by a primary battery, and will consist of the recharge circuit. The secondary battery will operate the off board devices including the camera, and wireless transmitters. The supply will be designed a manner such that feature expansion can be easily implemented.

Section 3: Research Related To Project Definition

3.1 Division of Labor

The work delegated to each project member was split evenly based on the amount of information the section would be comprised of. Each member was tasked with a portion of the project to follow through from the initial requirements to end design and testing. This method of dividing the workload allowed for each member to become well versed with their percentage of the project. Because the group is split evenly between two electrical engineers (Ali and Michael) and two computer engineers (Elso and Johnathan), the type of work delegated to each member was based off what their interests were. Below is a table showing the breakdown of assignments. Note that several parts of the project did require a joint effort and discussion in order to be successfully completed.

Project Component	Group Member
Tablet	Jonathan
Processor	Ali
PCB	Ali / Michael
Servos	Michael
Weapon Platforms	Michael
Camera / Laser	Elso
Power System	Ali
Software	Elso/Jonathan
Wireless Hardware	Ali/Elso / Jonathan
User Interface	Jonathan
Housing	Michael

Table 3.1 – Division of Labor

3.2 Existing Similar Projects

Autonomous sentry turrets come in many different variations. From basic beginner hobbyist tutorials to expert level designs, there was a lot of previous information out there to learn the most effective ways of doing this project are and improving those designs. Instead of an airsoft gun used in this project, some designs used different weapons like paintball guns, Tasers, or real guns when done on the professional level.

Our group's first exploration of research began with previous UCF Senior Design classes, looking for any projects related to turrets. From there we were able to find a number of previous assignments: Remote Touchscreen-Controlled Defense Turret during fall 2011, Autonomous Targeting Sentry (ATS) during fall 2011, and Automated Targeting Proximity Turret during fall 2010. Reading through those reports we were able to come up with the own version of an automated turret and expand on their ideas.

3.2.1 Remote Touchscreen-Controlled Defense Turret (RTCDT)

Remote Touchscreen-Controlled Defense Turret was a laser turret project which monitored a designated area, highlighting multiple moving targets and sent the given information to a Windows tablet via Wi-Fi. The group displayed the moving targets through an Open-CV interface on the tablet by highlighting each moving target with different colored outlines. The user is given the option of isolating one colored target by pressing a button on the tablet. Next they calculate the centroid of the target, then send the information to an Arduino microcontroller which moves the servo motors following the target and firing a laser pointer. In order to ensure the laser is correctly pointed at the target their group used a PID controller applied in the servos. When given the coordinates of the target from the Arduino, the PID controller took that information then calculated and fixed the movement errors of the servos. That signal was then sent to the servos which moved the laser.

Managing the imaging of objects on the tablet was done in three steps: object detection, object outlining, and object tracking. For object detection the group used a color recognition method. At first they used background subtraction but that was deemed too slow. With color recognition, the user selects an object on the screen, the program identifies the pixel and stores its color value. Then by setting a threshold for the color they were able to filter out objects of different color. Finally they converted all the pixels within the threshold to white and the rest of the image to black. Next the group outlined the object and calculated the centroid. They chose to outline the objects with a rectangle for ease of centroid calculation, but may not be ideal for certain shaped targets. The final step was tracking the object which was done by calculating the distance the centroid moved from frame to frame. This information was sent to an XBee radio module which used its library methods to convert and send the X-Y coordinates to the microcontroller. Finally the microcontroller converts the coordinates to PWM signals and send it to the servos which aim at the intended target.

The group was able to learn a lot of valuable information from this project and were able to narrow down the decisions for a number of aspects for the project. The design of the turret was a pre-built device which was manufactured to incorporate servos. This group did not attach a full sized gun to the turret, only a small laser which we took into consideration when deciding the turret structure. Learning about what method of object tracking was effective and what was not effective aided in choosing the proper method for this project while avoiding non efficient methods such as background subtraction. Giving a clearer idea on how to combine every part of the project into one was also useful information to us.

3.2.2 Autonomous Targeting Sentry (ATS)

ATS was another Senior Design project similar to ours. Unlike RTCDT, this project attached a paintball gun to the turret. By using high definition web cams and OpenCV libraries they detected and tracked moving targets. This project had the ability to be controlled manually by the user through a connected laptop. Also when the turret initially sees a target they are given a warning to get out of the way.

Where the RTCDT used an Arduino as their microcontroller, the group for ATS used a Texas Instruments MSP430. This allowed them to effectively communicate with the laptop using Microsoft's .NET Framework along with communication to the servo motors. The base structure for the turret was the front fork of a bicycle in a hanging position. The fork allowed the servos for movement for both the x and y axes. For the power supply the group used a 12V rechargeable battery which was sufficient for their needs. They attached two alarms to the project, one was simply a loud noise maker and the other was spoken audio played from speakers to warn humans.

Object detection was done with the use of several OpenCV libraries. With the use of color detection OpenCV methods, the group was able to distinguish their targets with greater accuracy. Like the previous group, they converted the RGB to grayscale for simpler tracking. Tracking was done by using OpenCV methods which took the targets from the object detection class, found the differences between every frame and calculated their velocity, direction, and range.

Manual control was done with a Manual Control Class created using Microsoft Visual Studios. The user interface is similar to ours with a video feed and buttons for rotating and firing the gun. They also had a button to initiate the warning siren. To ensure that not just anyone can access manual mode, the group implemented a facial detection and recognition class. By using more methods from OpenCV they were able to scan user's faces, sort through a database of authorized images, and determine if there was a match and allow access into manual mode. This project had a lot of schemes that the group is going for, especially a manual mode which was hard to come by during the research. Though this group did the project from a laptop and not a tablet we were able to come up with a lot of great ideas and narrow down the goals.

3.2.3 Automated Targeting Proximity Turret (ATPT)

A third similar Senior Design turret project was ATPT which was a paintball sentry gun with the added feature of using a server to store the attack history. This project also used range detection, when a target came into the field of vision it calculated the distance they were from the gun. When the objective came within a certain distance a warning signal occurred and if they did not leave then gun would fire.

Range detection was achieved by using an inexpensive laser pointer connected to a control board which was designed to specifically communicate the laser with the laptop. Three webcams were used for the image processing, one high definition and two low definition. The HD camera was mounted on top of the gun while the two low-def cameras were on the base. Their purpose was to keep track of the differences between frames, one focusing on the yaw and the other the pitch. Using AForge.NET libraries for motion detection and tracking and an Arduino microcontroller they were able to successfully detect motion and range of a person walking by.

A database was required in order to store the history of information when the turret is active. The group kept track of every time a target was hit, taking a screenshot of the video feed in the process. All of this was done using MySQL language. The base of the turret was made of wood with the standard three servos for movement. There was also a manual mode option for the user.

This project was another one of the few we came across that had a manual mode. Most of the previous senior design projects used an Arduino microprocessor which the group is unable to use, so we had to research more for alternatives. Overall, going through these three projects, among others, we were able to learn what methods were effective and what were ineffective and not waste the time with. Likewise, we were given options for techniques that did work and were able to decide which one we would like to use.

3.2.4 Lessons Learned

By reviewing a number a similar sentry tracking turret designs, the group was able to learn a lot of valuable information. Methods of research for former senior design projects included reading each group's Senior Design 1 paper and Senior Design 2 paper. This gave knowledge into designs that failed and the explanations for the failure. Seeing what others have done and where they fell flat on their designs gave the group a greater understanding of the overall goal of what this project should be. Not only failures, but successes were also noted giving the group viable options for various divisions of the design. Based off the previous project designs that were researched, a flowchart was necessary to overview the function of what the design should be. From the below diagram, seen as Figure 3.1, the basic process can be mapped out for the automatic firing mode of the sentry turret system.

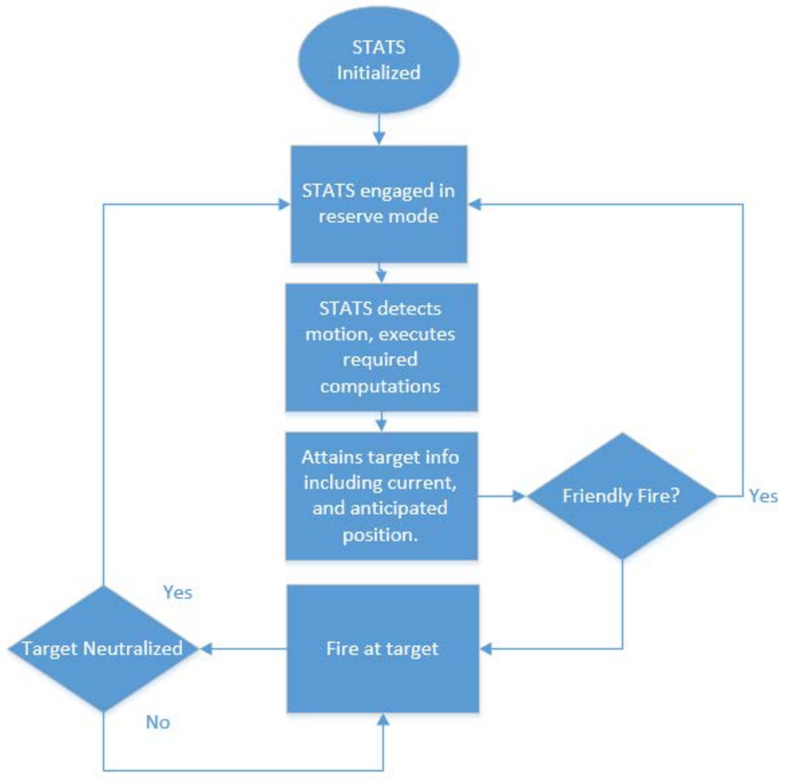

STATS Initialized
STATS engaged in reserve mode
STATS detects motion, executes required computations
Attains target info including current, and anticipated position.
Friendly Fire?
Yes
Fire at target
Target Neutralized
Yes
No

Figure 3.1 – Automatic Firing Overview

3.3 Software Research

In order to create a program that will track an object; first realize everything that is involved in that process. The video input should be constantly monitored by periodically updating changes in individual frames; possibly caused by wind or other slight movements that the system would not consider to be a moving target. As the frames are constantly being updated, a pre-calculated body mass such as a human adult would automatically be noticed because of the massive change in the input results. The tracking software will have to be able to recognize this body mass as a target, find and mark a center of mass for this target, as well as decide on a direction that the target is moving in order to compensate for the time between shots fired and target hits.

There are multiple ways of designing a tracking system; some involve using heat detection and infrared. Other methods would require multiple cameras and a mathematical equation to calculate the input from each camera system in order to send the proper location signals to the servo motors. The location of the mounted camera must also be considered because if it is not directly mounted to the weapon then there must be a method of informing the program where the turret is currently aimed. Another solution is to mount a laser pointer on the weapon with the exact line of sight as the barrel and program the tracking software to match the laser to the center of mass of the target.

When the user selects specific targets in “manual” mode there will be far less computation needed. The location selected would be mapped to the specific pixel cluster from the video feed and the processor must then command the servo motors. The motors can be commanded to either move until the laser pointer matches the target cluster, or set a new center of mass in the middle of that target cluster and remain stationary at that point of aim; until the user selects a new target.

3.3.1 Open Computer Vision

Computer vision is a very popular and widespread method of creating motion tracking abilities through the use of multiple open source libraries that can be utilized in order create a device that can identify and track a moving object. Open Computer Vision or OpenCV was developed in Russia by Intel and is free to use under the open source BSD license and can turn any linked camera into a motion tracking camera. It is cross platform so it can be used on most of the operating systems available for this project and it is also highly supported. Also most of the OpenCV libraries are explained in detail on many public sites which will aid in learning how to use the many tracking options for this project. OpenCV requires some basic knowledge of C/C++ which some members of this group have experience in. OpenCV makes it possible to detect object, track objects, and implement facial recognition, color recognition, as well as path prediction. Other benefits to using OpenCV is the fact that many robotic projects have been created with this software and many mistakes can be avoided by researching the work of others that have used this software in the past.

The object tracking algorithm used by OpenCV is more commonly known as the Lucas-Kanade algorithm that uses optical flow to track motion. The first step in any type of motion tracking involves image processing. Depending on the requirements of the tracking software, the image must be processed to determine shapes, detect object edges, distinguish colors, or detect heat when using thermal or IR methods. Using what are called Haar like features of an image the image processor can distinguish if the image contains thing as complex as a human face or as simple as a moving blob. The last step is for the algorithm to decide whether or not the moving object is a target, if so then the algorithm must focus on that target and follow it through the progressing frames. Then through the use of optical flow the algorithm must attempt to predict the path at which the object is traveling within the field of view. OpenCV has a number of features that enable a device to track objects in many different ways depending on the requirements of the project.

3.3.1.1 Color Tracking

One method of creating a motion tracker using OpenCV is by color filtering. Essentially the program must be set to ignore all background colors except for the one that is of importance. In order to properly do this the settings must be changed from the BGR (Blue,Green,Red) scale to the HSV(Hue,Saturation,Value) scale because colored objects are easier to filter in the HSV color space when compared to the BGR color space. Once the color scale is changed the max and minimum pixel values can be adjusted to filter out the colors that are not needed; creating an image on screen that only represents the color of choice. This method gives you a nice binary image that displays the specified color as white with all other objects and colors on the screen as black.

The next step in color based tracking with OpenCV would be to find the contours of the object that is to be tracked. This is done by using a predefined “FindContours” function within the libraries themselves. The input to this function will be the filtered binary image that would now be showing on screen, and the output of this function would be a vector of contours. This step would then be followed by the built in “Methods” function that would take that vector of contours and output the XY coordinates of the largest contour which would be defined by the objects inner area. This method of tracking is simple yet effective but requires a lot of computational power and might be too much for a small microcontroller to handle without the use of a laptop or other separate processor.

3.3.1.2 Facial Detection

Facial detection is also possible with OpenCV and can be used to determine if an object has a face so that the system can determine if it should be tracked. One such method of facial detection uses the “Haar Feature-based Cascade Classifiers”. Essentially the system is given a large amount of images with faces (positive images) and a large amount of images without faces (negative images) and then uses extracted features from those images to compare to the objects that appear on screen. Since it would take an extreme amount of time to compare all images in a screen shot to all of the different features of a face, the algorithm saves time by immediately discarding all images that could not be a face and focuses on only the pixels that can. This saves time because as stated in the paper “Rapid Object Detection Using a Boosted Cascade of Simple Features” by Paul Viola and Michael Jones, a 24x24 window would result in 160000 features. By ruling out non possibilities this algorithm rapidly speeds up the process and produces a facial detection system that is accurate as well as fast.

3.3.1.3 Facial Recognition

Along with facial detection OpenCV can also be used to recognize particular people using facial recognition. In order for a program to recognize a particular face the Haar-Cascade function must be implemented again. Then a face recognizer class must be implemented and then taught to learn all of the faces that are of interest to the user. This is done by calling a number of images of a particular face and assigning those images an id so that the Haar-Cascade algorithm can give those particular features an id and later relate that id to a face that enters the view of the camera. It is important orient the reference images the same way. The facial images being used for reference must have the same height and width so that facial features like the eyes and nose match up when compared side by side. There must also be a method of reshaping incoming image frames to match the format of the reference images so that they can be compared quickly and accurately. Facial recognition would be an added bonus to include in a sentry turret in order to create a list of friendly targets, but the speed of the turret system might be hampered by the extra weight of comparing incoming objects before firing at them. Therefore if facial recognition software is found to be too cumbersome to include then it will not be used in this project

3.3.1.4 Blob Detection

Blob detection is a popular method of detecting and tracking motion in OpenCV that has been used in similar sentry gun and turret projects in the past with success. Blob detection is a mathematical method that checks for regions in an image that are different in properties such as color or brightness than the surrounding areas of the image. A blob consists of a region in those changing properties of the image that have some constant value. So if a person were to cross in front of a still background in an image the area their body takes up would produce a relatively constant change in the background as it travels frame by frame. These changing regions can be used to notify the program of the presence of moving objects within the field of view and with the use of mathematical formulas the center of this blob can be used for targeting data.

In order to find the center point of a blob or "centroid" by setting a predefined shape such as a square or rectangle along the majority of the blob object. This shape must enclose the blob and then simple calculations can be used to measure and determine the center point of the object. This center point is important because in the case of a human object the center of the target will be the largest portion and would also be the easiest part to hit when firing the turret. Much like police are trained to aim for "center mass" in order to have the highest probability of hitting the target, this program must be able to locate and point at the center mass of a blob. An important library that must be used in order for this method to work is called cvBlobs which takes all consecutive pixels of a user selected color and represents them as rectangles in that chosen color. It is also important to preset the desired blob size in order to rule out any objects that do not meet the requirements. This method will result in an image that only shows the selected blob on screen and will negate out the rest of the background noise in order to create a very clean image of the desired target.

Once the image is processed down to a specified shape and most or all of the background noise is ignored the center of the object can then be calculated and used. Calculations for blob detection are done by the cvMoments function which calculates moments of an image on each frame. These moments are then related to the overall screen size in order to send the location of the object with respect to the field of view. When each frame is compared one after the other the different coordinates of the object can be used to determine where and how fast the object is moving.

3.3.2 Processing Programming Language

The Processing programming language has been around since the early 2000's and is used for a wide variety of visual applications. Originally designed for the use in visual arts, this programming language has become very popular for a multitude of applications including similar robotic projects. A focal point of this programming language is that it is designed for people that might not be great at programming and it is helpful because it gives the user instant visual feedback. Processing has plenty of online support and seems to get a lot of use in the online community. It appears to be just as likely to be used for motion tracking in this project as OpenCV.

3.3.2.1 JMyron Libraries

OpenCV seems to be the most popular method of implementing motion detection and motion tracking but another option for this is the open source library JMyron. JMyron is an external library for the "Processing" program language. The benefit of using the JMyron library and Processing instead of OpenCV is the fact that Java can be used as the main program language. Both computer engineering students in this group have more experience in Java than in C++ so this option might be a great alternative if OpenCV becomes too difficult to implement. Similar to how functions are called from the OpenCV libraries JMyron functions can be called as well. Currently almost anything that can be done using Intel's OpenCV can be accomplished using this library. Another benefit to this programming language is the fact that it is highly compatible with the Arduino platform. Since most of the testing for this project will be done using an Arduino Uno, this could be helpful because it would be simple to integrate the Atmel processor on the Arduino to the system.

3.3.2.2 GSVideo

Similar to the JMyron library, the GSVideo library is another cross platform external library for the Processing programming language. GSVideo is designed for video support including video playback, video capture and the creation of movie files. It can also be used as part of a system for tracking motion from a video source. GSVideo uses the GStreamer multimedia framework and it follows the API of the built in video library. This library seems to be less useful for motion tracking when compared to JMyron because much less information can be found on it online. For this reason the choice of software for motion tracking will most likely be between OpenCV and Processing with the JMyron library.

3.4 Hardware Based Tracking

Using a hardware based tracking method has multiple pros and cons about it. A benefit to using a hardware focused tracking system is speed. The hardware can talk to each other much faster than separate software can communicate with each part. A downside to using hardware based tracking is that it involves complex algorithms that must be researched and figured into what exactly this project needs. That can add significant time as well as introduce a higher chance of errors if the algorithms are not correctly implemented.

3.4.1 FPGA Motion Tracking

FPGA boards are one of the most popular tools to create a hardware based tracking system. The Field Programmable Gate Array better known as (FPGA) is equipped with multiple logic components that are designed to be programmed and are capable of calculating complex mathematical functions. Since matrix algorithms are one of the requirements of hardware based tracking this means that an FPGA is perfectly equipped to handle this task. The hardware based method requires that an image be obtained by a video subsystem and then those images are segmented and filtered so that an object can be identified and tracked.

Video frame analyses enable the FPGA to calculate the location, size, shape, speed and directional changes of an object in motion. There are multiple methods that can be implemented on an FPGA to create motion tracking. The first method is blob detection where a moving target is segmented and a shape is set to enclose the moving parts of the image resulting in a blob like target on the screen. Another method is Mean-Shift tracking. Similar to cluster analysis in computer vision, the mean-shift is a procedure that locates the maxima of a density function by using data that is sampled from that function. Basically this algorithm takes a histogram (Usually based on the color of a new image (frame)) and compares it to the previous image (frame) and uses this information to compare pixel density. A third option for FPGA tracking is the contour tracking method. Basically this algorithm finds that outer contours of an object as it progresses through each frame. This method uses the boundaries of a moving targets shape in order to determine its directional heading as well as speed and size. In order to use the FPGA method of tracking, all of these algorithms must be tested and compared in order to decide on a working design for this project.

3.5 Hardware & Software Tracking Systems

Another way to create a motion tracking system would be to use a microcontroller like found in an Arduino along with motion sensors and/or an external camera. Using a microcontroller along with external sensors is a valid method for motion tracking and the libraries found in the open source software that comes with many microcontrollers can be used to help code the specific algorithms needed to find a moving object.

Computer vision software like OpenCV can be used in conjunction with a microcontroller and a wired or wireless vision sensor such as a webcam or other small video systems that have the ability to connect to the microcontroller. Using the microcontroller to store all of the input and output data from the motion tracking software; including the location of the moving object as it moves through the pixel space can quickly overwhelm many processors. This is why most autonomous turret projects rely on the computational power of a laptop in order to function properly.

While there will be need of a laptop in order to program the microcontroller, one of the goals of this project is to create a standalone system that can be accessed remotely and left in an area to work as a self-sufficient sentry. If it becomes too cumbersome to store all of the tracking code directly on the board itself then a laptop computer will need to be connected at all times and there will be separate functionality handled by an Android tablet that will allow the user to be away from the turret itself while it is functioning.

3.6 Additional Sensors

There are a number of additional sensors that can and should be utilized in a sentry/turret gun project in order for maximum effectiveness to be achieved. Some of these will help with tracking motion and others might help in improving accuracy of the projectile that is being fired at the target.

3.6.1 Sonar Tracking

There might be a need for sensors including infrared sensors, accelerometers, lasers, and or sonar sensors. Sonar sensors are small and lightweight and are also reasonably priced. They use high frequency sounds which bounce off objects in the path and are returned to the receiving end of the sensor for processing. With the use of multiple sonar sensors a device can calculate where an object is relatively quickly. One problem with sonar is that it won't have the speed and distance that is needed to track a target out to distances over 20 meters, especially when compared to other sensors. Most sonar sensors that would fit in this project have a max range of about 3 meters which will not work.

3.6.2 RFID

Radio Frequency Readers and Radio Frequency Identification Tags (RFID) are a widely popular option for tagging specific objects in order to identify their type and importance. Using this technology in the turret project might allow for specific people or objects to be set as friendly targets. Placing an RF tag on a specific item like an air conditioner and programming the device to not shoot at that tag would prevent the airsoft gun from attacking a moving target if they pass behind it. A tag could also be placed on certain people to allow "friendly" objects to pass in front of the turret without fear of being attacked. In order to achieve this goal the project must implement an RF reader to the design. Essentially this project must either implement a prebuilt RF reader or design one from scratch; the same goes for the RF tags. Both can be found online but most of the time RF tags range between 30.00 and 50.00 dollars each and the RF readers can be in the hundreds of dollars. As of now this option is going to be pushed to the backend of the project and may be attempted if time and budget allows for it.

3.6.3 Infrared Motion Detection

IR sensors are a viable option to recognize motion because they pick up changes in the infrared signature of the area they are monitoring. PIR sensors (Pyro electric Infrared) are portable and effective IR sensors that could be used but there are drawbacks to this option as well. The first drawback is the fact that a PIR sensor has a warm up time and could slow the system down; though once the sensor is warmed up it is extremely sensitive to changes in the IR spectrum. A downside is this type of sensor does not give you the ability to track a moving object. Like in home security it is just there to detect motion so this option would not be used as the main method of tracking but could be used if properly implemented to awaken the turret system if movement is detected.

3.6.4 Range Detection Sensors

Range Finders are a useful tool that may be required in order to achieve maximum accuracy from the turret. Due to the flight path of most airborne objects there must be a way of calculating how far away an object is from the turret in order to judge the vertical angle that the airsoft gun must be set to in order to reach it. In order to maintain an accurate shot the range finder must be fastened to the moving parts of the turret; namely the firearm so that the distance can be adjusted as the target moves forward and backwards during evasion. There are a number of rangefinders available that would work for this project. There are sporting rang finders found on golf courses and around the necks of hunters but the size and weight constraints on this part limits it to either an infrared rangefinder or an ultrasonic range finder. Another option is to use software such as OpenCV in conjunction with a laser in order to determine distance.

3.6.5 IR Range Finders

Infrared Range finders were first considered because of their popularity among target tracking robot designs. IR range finders use triangulation in order to detect the distance of an object. They do this by sending a light pulse with a wavelength range of about 850nm outward towards the area in front of the device. When this light is reflected back and returns to the sensor, the angle of the reflected light is used to determine how far away the object is from the sensor. Figure 3.2 below shows how the reflected light is triangulated. It determines this angle by using a special lens that sends the light onto a CCD array which determines the angle and sends an analog signal to the connected micro controller.

A possible choice for an IR sensor might be the Sharp brand IR Rangefinder. It is priced low at about $18.00 and has very low power consumption. A benefit to using infrared technology is that ambient light as well as the color of the object will not negatively affect the effectiveness of the sensor. A downside to using an infrared range finder would be that most sensors have a range limitation. The Sharp sensor has a maximum range of a few meters which might not be enough for this project depending on location. If the turret is to be used inside a small enough room then this type of sensor might be utilized in the project.

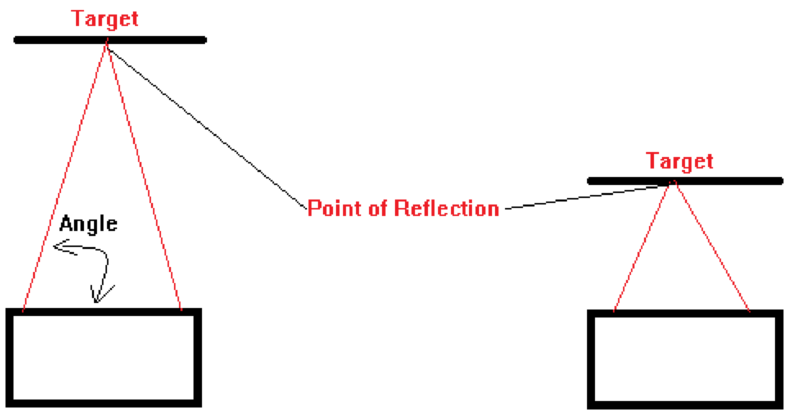


Figure 3.2 – Range Finder

3.6.6 Ultrasonic Range Finders

Ultrasonic Range Finders have been used in many robotic projects in the past and might serve as a good alternative to the IR rangefinder. Ultrasonic rangefinders use sound waves bounced off of targets in front of the emitter and then uses the angle of the returning sound waves to determine the distance of the target. This works very similar to the IR rangefinders but has a wider area so that narrow objects such as the side perspective of a person may be seen by the sensor better. Ultrasonic also suffer from range problems for this application. A decent ultrasonic sensor might be the Maxbotix Ultrasonic Rangefinder. Its maximum range is about six and a half meters with accuracy down to as little as an inch. It is also priced well at about 23.00 USD which would not hurt the projects budget greatly if implemented.

3.6.7 Laser & Camera Triangulation

The third option for range detection that was researched was adding software functionality that could implement distance calculations by using the laser device that will be installed onto the airsoft gun. Using a software based distance calculation will save in cost and will allow the mostly aesthetic laser to be used functionally in the tracking process. Basically the software would need to be able to determine where the laser pointer is contacting the object in the field of view of the camera. This would be achieved by finding the brightest pixel in the field of view; the laser dot on any object will be the brightest light source in almost any condition. The camera will then become the sensor that will work with the laser to determine distance. Similar to how the IR rangefinder works, the camera should be at a specified distance from the laser on the mount and when the software recognizes the laser in the field of view it can then determine the angle and thus determine the distance of the object. This option will require more complicated algorithms in the code but will achieve the greatest distance of the three options listed.

The distances that could be calculated using this method will far exceed the limits of the airsoft gun and thus would be more than adequate for the purpose of this project.

3.6.8 Laser Sights

There are multiple kinds of laser sights including weapon mounted sights and battery operated laser pointers; commonly used in classrooms. Weapon mounted laser sights are specifically designed to be mounted to a gun (airsoft rifle) which could be beneficial in this project. There are also multiple color choices that would be determined by visibility in day time, certain laser colors appear better during the day and would depend on the applications of the auto turret. This project may benefit from linking the laser sight with the power system of the PCB so that the sight turns on as soon as the turret is active. Another viable option is to use a laser sight equipped with a remote pressure switch that can be mounted between the trigger and the servo motor so that the laser engages only when the trigger is pulled. The optional addition of the laser sight can also be used as a warning system if needed. If a warning system is implemented in the turret then a laser sight pointed at center mass of a target along with other options like audible warnings or lights can be an effective method of deterring an enemy.

Laser technology has become very advanced in recent years and the price of such small lasers has been significantly reduced. There are many options when deciding on a laser but the most sensible when considering ease of use and price were basic weapon mounted lasers. Today there are mostly two options when deciding on a weapon mounted laser, and they both have to do with color choice. A typical laser designed for use on a weapon is a 620 - 640 nm red laser. The other option for a laser system would be to use a green laser which uses a 522 - 542 nm beam. The difference is that green lasers fall under a section of the visible light spectrum that is closer to the center and therefore can be seen better in any lighting by the human eye. This matters to this project because if a laser is to be used it must be visible in both direct sunlight and in a dark room. This is especially important if the laser is to be used in range calculations because if the turret is to be used outdoors then the laser might not be visible enough to be used in the calculations. Table 3.2 shows a brief list or possible laser choices that will fit this projects budget and needs.

Laser Systems		
Manufacturer	Laser Wavelength Spectrum	Price
LaserScope Tactical	532 nM (Green)	$35.00
BSA Laser Sight	650 nM(Red)	$19.99
NcStar Compact Laser	640 nM (Red)	$24.00

Table 3.2 - Lasers

3.6.9 Warning Alarm & Light

As previously mentioned, the turret will be utilized in 2 modes. The first being manual operation controlled by an Arduino device, while the second being the automatic mode.

Part of the automated mode will be a warning alarm/flashing lights that will serve as a warning to the intruder that a turret is in place and therefore be alert. This will commence shortly before fire and will persist throughout. Initially the warning sound/light will be set up to have police lights and siren. Research has been made to make sure that the siren does not violate any laws. As a conclusion, the law states that as long as one is not trying to imitate being law enforcement then the design can be produced.

When trying to implement this sub system, it was realized it would be a better idea to use a whoop buzzer. This is similar to what is used in car alarms, or the standard sound emitted by a fire alarm. A Whoop buzzer is considered a better choice as it is already embedded with the audio with in the solid state device. This would be more efficient then having to load the audio file into the program memory. A device considered for this part of the design is the Ultra Series lineup by Floyd Bell in which claim a sound level of 108 dB within 2 ft. Floyd Bell mainly engineers and manufacturer's solid state audio devices to be used in industrial applications as such of forklifts, ambulances, and factories. This device sells for $21.14, and will be utilized due to its unique characteristics in which include variable temperature operation, water proof, and is made for both indoor, and outdoor use. This device is a PC mount in which will have to be included with in the PCB design. As an alternative it may be best to have this on a dart board with leads to the main PCB. This is due to the PCB being enclosed therefore it will not be optimally heard.

As for the Lights, several tutorials were found online by simply flashing a red & blue LED light. A better option was to use several RGB lights. Different strobes of red, blue, and white will be flashed simultaneously in order to get the police light flash. Every LED will a rated resistor to limit the resistance to insure the LED does not croak itself. An LED that came up as an option was the RL5-RGB-D diffused tricolor led from super bright led. It has a low power rating and the manufacturer claims the colors change quick simultaneously with a diffused effect that allows for the glary police light feel. The lights will be surrounded with mirror like reflective sheets in all directions to amplify and direct the light intensity. The lights will be mounted on the same dart board with the Whoop buzzer, and will be place in a way such that the tracking camera is not affected by the light. Depending on the controller used, a microphone may be used to record sound. This will act as audio surveillance for further security and a reference for going archive

3.6.10 CMUcam

Another more interesting option is using or creating a separate board that does nothing but calculate the motion tracking software from a built in camera and outputs only location information. A recent kick starter by a company call Charmed Labs has created such a device called "Pixie" which is a small standalone board that can directly connect with a microcontroller and would allow for powerful tracking abilities in a small lightweight package. If we decide to create a system that is not laptop dependent then using or creating a device such as Pixie could be a viable option.

Pixie is a new image sensor (CMUcam5) that can be utilized along with a microcontroller such as an Arduino for example or any other microcontroller through the use of digital out, analog out, I2C, SPI, or UART Serial connections. Pixie is impressive because it is a board that contains a separate processor that's sole purpose is to store the important data on its own memory and only output the useful information to the microcontroller. The simple data such as location on an XY plane of pixels and even selected colors are sent through; while all of the calculations and other work are dealt with by Pixie. Even more impressive is that Pixie is able to send this information at a frame rate of 50 Hz which will allow the microcontroller to quickly respond and issue instructions to the servo motors thus creating a very agile turret system.

Pixie uses the hue based color system like the one described in the OpenCV section to determine if the objects on screen match any targeting data stored by the user. The most significant reason for using the hue based color system is due to the fact that lighting changes do not significantly affect the hue of a color in the HSV scale. Pixie can be a great addition to the auto turret if color coded targeting is to be assigned. Using the built in programming you have up to seven different color signatures that you can assign to specific targets but with the open source software you can program more color signatures manually.

Another interesting feature of Pixie that can be used in this turret system is the fact that you can teach it to track certain objects by their size, shape, and color quickly and easily. It does this by creating a statistical model of the object in front of its lens while you're pressing the button located on the top of the device. It will then store that data on flash memory and use it to track any object that passes its field of view. Along with that ability Pixie can also be programmed to recognize color combinations. In this project this could make for a very interesting friendly fire mode by attaching specific color coordinated shapes onto friendly humans that the device would recognize as non-threats.

PixieMon is the open source software that comes with this device and it can be completely modified in order to meet the needs for this project. While this device is very useful for tracking objects accurately and quickly it will still require further design in order to use it in an auto turret platform. It's currently being shown as a tool to create small robots that can track specific small items based on color and shape. Modifications would be needed in order to use this product to track moving human sized targets at distances of up to 30 meters, but this product would be a good place to start when it comes to the video subsystem of this project.

3.7 Processors

A vital segment of designing the turret will be choosing the correct method of processing for control and up keep. The programing environment will revolve around which architecture is used, and the program memory will be determined by the available memory offered by the chip. Accuracy of the turret will be affected due to different processing speeds, and its capabilities. The processor has to interface with the motor controller, camera, wireless interface, sensors, along with the Windows tablet.

3.7.1 Processor options

Key focus points that are to be concentrated on while choosing the right controller will be motor control, flexibility with Windows interface, design simplicity, and wireless interface. An important constraint that will have to be taken into account is time. It is critical to note that since prototyping will be done in the summer semester, there will be a shortage of 4 weeks due to the condensed summer semesters. With that said, it is essential that the simplicity in design will be maximized to avoid constraint of the final outcome. Motor control is key as a slight lag, or shortcoming due to processing would delay the control system and result in an inaccurate shot. This can also result in a harmful outcome as overshoot, or undershoot can cause other object near the projectile to be damaged. This can be correlated to the microcontroller as not being able to cycle through all the frames at a fast enough time. Therefore the target would have moved by the time the calculated position is determined.

A more complex option being a microprocessor was initially included as a prospect. A microprocessor, similar to a microcontroller, but can implement complex tasks that a microcontroller may not be able to handle. A microprocessor can often implement tasks at a much quicker speed compared to microprocessor. The core of any computer is powered by the central processing unit (CPU). Pertaining to STATS a microprocessor would be used to handle all the computer vision related tracking. Ideally this sounded as a great idea since computer visions libraries require very high computations, or complex functions that a microcontroller does not support. Along with many advantages of a microprocessor, disadvantages include size, and complexity of components. Compared to a microcontroller, a microprocessor will require off chip RAM, ROM, along with cooling, and would require an operating system. Although the features of a microprocessor are great, its tedious constraints definitely outweighs the benefits.

A FPGA is also considered as a possible choice and does help keep the cost down due to some components which can be programmed using a respective hardware language. FPGA structure contains logic blocks that can be configured to implement code. FPGA's execute all code in parallel, therefore a faster runtime would be expected along with a greater support for repetitive procedures. A drawback of FPGA's is the lack to control other peripherals controlled to the board. FPGA's are stronger at computations then standard microcontrollers, but are most definitely. Other limitations include the complexity of HDL over a language, such of C that is supported by OpenCV and high power consumption compared to microcontrollers.

It's determined that a microcontroller would meet the requirements and specifications for the project. A microcontroller would be adequate for a sentry turret application over a microprocessor, and an FPGA. The microprocessor would involve many additional components, while the FPGA requires much more hardware level programming. Both ideally would be possible if more resources, and time were allocated. Fortunately, since STATS does not necessarily require the use of top notch calculations, a fast microcontroller is sufficient.

3.8 Microcontroller Software Issues

Due to issues with finding a microcontroller that utilizes a floating point multiplier, it came to a conclusion to use a PC to process the tracking as it would be much faster and precise. An Atmel Mega 328 will still take care of the processing in terms of Android use, along with Servo motor control, alarm, and additional features. The Mega 328 will use serial transmission to communicate with the PC. The PC will transmit the location of the target to fire at. Utilizing the control system the Servo Motors will move to the object.

An important note that must be taken into account is the microcontroller will not necessarily have the most impressive clock speed. As mentioned earlier, the calculations are not off the top in terms of computations. With this said, most of the functions require minimal to average technical specs. The big dilemma will be providing a full floating point multiplier. This is not available on microcontrollers. However, this feature is available on processors, but since a microcontroller will be used, this cannot be implemented in the project design. A solution to the issue would be including a hybrid architecture in such a way that all the Computer Vision computations are left to a computer processor and will be communicating via USB in real time. This will be much more efficient and practical in terms of tracking as OpenCV is supported in the PC environment.

Choosing the controller was one of the more challenging tasks not only because there was a breach trying to maximize meeting the minimum technical requirements, but it was also the logistics and business involved. This includes such elements as cost, and lead time. Choosing the microcontroller in a timely manner would be an ideal choice for alpha testing of components. However, it took several group meetings along with a high level flowchart to realize that the right choice had been made.

3.9 Microcontrollers Considered

The following were microcontrollers considered to be part of the final design. Texas Instrument's MSP430, Atmega328, and Microchip PIC16F are the top of the line competitors in their respective field that would be sufficient for the turret.

3.9.1 TI MSP 430

The MSP430 is the first choice since it has been used in diverse different applications throughout Academia. It has been produced in many variants which allow more ease in terms of application. Low power consumption helps meet the reduced power requirements, along with extending the runtime of the system due to efficiency. Texas Instrument does an excellent job providing documentation along with reference designs for the MSP430. The TI E2E blog will be an aid in terms of debugging as the prototyping stage approaches. TI offers the MSP430 in over 100 options, including FRAM variations, LCD compatibility, different memory sizes, and I/O pin options. Different members of the group already possess an MSP430, which will lower project cost, and be beneficial to the prototyping stage. The group has previously developed for the MSP430; therefore this is the preferred choice. The MSP430 possess ideal features compared to many other embedded chips, and ships out at $4.30 straight from TI. This compared to other controllers is about a quarter of the price. Advantages of the MSP 430 is the ability to program in the C language which would ease the programming since all members have programmed in the C environment. Serial communication interface would be used to communicate with other devices or process in real time. The limitation considered is the support from the online community compared to many other microcontrollers including the Atmel Mega 328.

3.9.2 Atmel Mega 328

The Atmel Mega 328 is the favorite amongst hobbyist and robotic fans. Key specifications include an 8-bit RISC based architecture and 32 Kbytes flash. An interesting ability of the Atmel Mega 328 is its ability to achieve 1 MIPS per MHz. This allows for a good tradeoff between power, cost, and performance. The Arduino Uno currently utilizes the Atmega328 and functions perfectly in embedded robotic applications. Due to requirements, the Arduino Uno development board is not to be considered in terms of final product design, but the Arduino IDE is acceptable. The Arduino Computer Vision library does an excellent job with providing functions for low-resolution camera, and tracking algorithms. This would be a vital reason to prefer the Atmel Mega 328 due to its compatibility and resourcefulness to the Arduino Environment. Serial communication is also taken advantage of including a USART port.

3.9.3 Microcontroller-Computer Hybrid:

The Hybrid system will utilize a computer to process the computer vision algorithm while the microcontroller will serve as a controller to all the sub-systems. Utilizing the serial USART, the microcontroller and computer possess, information will be sent back and forth in real time. If the hybrid option is taken into account for the final design, then the microcontroller would have to support USART (universal synchronous/asynchronous receiver/transmitter). This is a hardware module that must be built into the IC to allow synchronous transmission that will allow a more efficient and faster communication line since there are no start and stop bits.

Generally, an operating system will not have much effect since terminal-like programs can run with any major platform. Therefore, any computer may be utilized to perform the computer vision algorithms as long as a USB 2.0 port, or RS-232 are present. Two options were being considered. The MacBook Pro is a well-built machine, in terms of performance, that can handle all the computations required for target tracking. Another option was to use a Raspberry Pi as the PC by loading a version of Linux onto its drive. The benefit of using the MacBook Pro is a camera would be utilizing the built in camera, therefore cost would be lowered along with the fact the group already possesses one. In addition, there online support for the product. On the other hand, the MacBook pro is larger, consumes much more power, and will not provide the mobile atmosphere originally proposed. As for the Raspberry Pi, performance is decreased due to a smaller processor, and less online resources. The Raspberry Pi is a fairly new product, released in 2012. Different OS can be loaded onto this platform including a light version of Linux called Raspbian. Since Linux is supported across many platforms, it is possible to utilize the Pi to implement the computer vision processing.

A variety of tablets were considered for the hybrid option. Requirements for a tablet include a long battery life, and full 32 bit operating system. The only 32 bit capable tablets are offered with a Windows operating system. Only a full version of windows 8 will be used compared to windows 8 RT since the RT version does not support the majority of the programs used including JAVA programming. In addition the window 8 RT mini USB serial transmission are not in complete synchronization with the processor as a full Windows 8 would be. The apple iPad, and all Arduino enabled tablets do meet the hardware compatibility in terms of processor. The dilemma would be dealing with input and output RS-232 transmission protocol. With that said a Windows 8 tablet will be used to execute the tracking, and interface with the motor controller.

Choosing the controller was one of the more challenging tasks not only because there was a breach trying to maximize meeting the minimum technical requirements, but it was also the logistics and business involved. This includes such elements as cost, and lead time. Choosing the microcontroller in a timely manner would be an ideal choice for alpha testing of components. However, it took several group meetings along with a high level flowchart to realize that the right choice had been made. It is agreed that the Atmel mega 328 microcontroller will be used to create the motor controller.

3.9.4 PIC Controller

As further research commenced on locating the most appropriate controller, it was inevitable to come across the 8 bit PIC microcontroller. PIC (Peripheral Interface Controller) was originally designed to be used to control small peripherals with very little computations. This would be ideal for the STATS project since most computations will be handled by a computer. The device most fitting for PIC control is the PIC 18F452 chip. This chip is an 8 bit C compiler, and MIPS based. It is widely known across hobbyist for being one of the stronger chips for motor control due to the ease of implanting PWM across most of the pins, along with its low power option. It is vital to consider the delay between commands as the servo motor must be able to quickly change direction. For the PIC, a servo command can be implanted for speeds as quick as 20 ms with the signal frequency set at 50 hertz. Another feature is the chip’s ability to support EUSART. EUSART (Enhanced Universal Synchronous Asynchronous Receiver Transmitter) module, allows for serial communication interface that is independent of the device program execution. This process is started by having all the components ready, including clock generators, shift registers, and data buffers necessary for serial communication without having to access the chip components.

An option considered for STATS is a SoC (System on Chip). A system on a chip consists of a microcontroller along with a set of additional components. All are fixed onto a package to be implanted in an embedded environment. This was ideal since the project requires the use of wireless interfaces, digital processors, serial interfaces, and possibly additional memory. When different SoC processors were looked at, starting with a variety from

Xilinx and TI, it was found that it would be far more cost effective to simply design the system separately using only the required components. Most chips were either all supported based on FPGA, or on ARM architecture which were out-ruled by design constraints. Since much of the processing will be handled by a computer an ARM architecture device would be too expensive and complex for an application such as a turret. Most of the SoC chips would not be used. That said, it would by unideal to use a SoC.

Table 3.3 summarizes all the considered microcontrollers for the project. Technical specifications are tabulated to help compare devices and make the best selection. The variations selection of the microcontrollers are found to be the most useful. All information was derived from the respective datasheet found on the manufacturer's website.

Processor	TI MSP430FR5739	Atmel Mega 328P	Microchip PIC16F178
Operating Voltage	3.6 V	1.8 - 5.5 V	2.3 – 5.5 V
DC Current per I/O	40 mA	40 mA	30 mA
FRAM	16 KB	32KB	7 KB
SRAM	1 KB	2 KB	1 KB
EEPROM	-	1 KB	256 Bytes
I/O Pins	15	14	25
Analog input Pins	10	6	10
Clock Speed	24 MHz	16 MHz	32 MHz

Table 3.3 – Microprocessor Comparison Table

3.10 Control System Processing

A feedback control system will be used to maximize accuracy. Generally this is more difficult done then said. A control system will be applied to further stabilize the system for further accuracy and to minimize the percent error. Due to portions of the turret system being mechanical it will take a few trials and analysis to further stabilize the system. Some options to be considered are to code up a class that will attempt to anticipate the location of projectile ahead of time. This can be implemented by comparing to frames that are read periodically, yet instantaneously. With this the position of an object can be viewed compared to the earlier frames. Depending on the position several calculations that are centralized around either differential equations, or a Gaussian plane will help predict the future coordinate. This system will run in the background to help further predict the position, without interrupting the motors. Using this method a projectiles acceleration, and velocity can be calculated as vectors for easier analysis for controller. Although this sounds a bit theoretical it can be easily implemented as long enough memory is present as the program will continue to save images into the memory.

The more images that are stored and compared, the stronger the feedback control system will be since the sampling interval is increased.

3.11 Tablets

One of the major objectives of this project is to have an option for the user to attack an incoming target remotely using a manual override function. The choice was to enable the user to be at some distance away from the incoming attack. Thus the selection of employing a tablet with the turret. Many previous projects used onboard microchips like FPGA or computers to process the incoming images. These complicated calculations take a lot of processing power so this narrowed down the choices of tablets for the group to work with. By researching through earlier senior design projects, it was determined that prebuilt libraries like OpenCV or Aforge.NET would be necessary for object tracking and detection. By considering these factors, it is important to choose a tablet that can accomplish all that.

There are three major tablet options on the market today, Apple, Android, and Windows. Apple applications are programed in Objective-C language. The group has no experience with Objective-C but after brief searching online there is an abundance of tutorials. Creating an Apple app using the iOS SDK (Software Development Kit) would be a great addition for the coders of the group to put on their resume. With some further research it is entirely possible to make an Apple app that is suitable to the project's needs. Android is mainly programmed in Java with XML used for the layout of the app's interface. With the programmers of the group already worked with and created Android apps for both classroom and hobby projects, creating an Android app would cut down on the learning curve significantly. Android is more open compared to Apple; meaning the developer has more freedom while programming. This is more helpful if down the line the group runs into bugs. Windows programs can be done in a variety of languages including Visual Studio, C++, and Java. Having completed nearly all UCF required programming courses for computer engineering, the two CE's will be able to code in Windows without any issue.

One more thing to take into consideration is not only experience but processing power. The tablet that will be employed in our project will be used not only for manual override control, but for the object detection and tracking algorithm calculations. This was taken into consideration when deciding the tablet for the project. Had the group decided to use an Android or Apple tablet, the processing power needed for the object detection and tracking calculations would not have been met. iPads currently use an Apple A5 chip. Androids vary from the Nexus 7's Qualcomm Snapdragon S4 Pro to cheaper tablets' lower end chips. These CPUs do not have the ability to process the video feed, tracking algorithm, and other processing needs for this project. If an Apple or Android tablet would be used, a computer or microcontroller capable of such tasks would need to be implemented such as a Raspberry Pi, Arduino, or BeagleBone.

With no one in the group ever making Apple apps or worked with Objective-C at all, Apple tablets were taken out of consideration. With the time constraints of this project already, learning a new language would be too time consuming. That left only Android or

Windows. Two members had experience with Android and there are an abundance of cheap options available out there. The idea of implementing a device such as a Raspberry Pi was intriguing, but in the end a tablet running a Windows operating system was chosen. The familiarity with programming on the OS with numerous different languages, giving the group a variety of choices for the final product made this decision much easier. Also the ability to utilize prebuilt libraries was a deciding factor along with USB ports for communication needs.

With Windows tablets becoming more popular, there are quite a few selections out there nowadays to choose from. Table 3.4 below compares a few of the ones the group was looking at. By examining the table, nearly any tablet available on the market will be sufficient for the processing power needs. It primarily boils down to price, operating system, and USB port readiness, as a micro USB port may not be applicable depending on the wireless USB device to apply. As seen in the same table, Windows tablets come in two different operating systems, the standard Windows 8/8.1 and Windows RT. Tablets running Windows RT use ARM processors which is cheaper and less powerful compared to Intel chips. RT is essentially a different operating system when viewing it from a programming standpoint. Software communication is more difficult and in some cases impossible. With this in mind, a tablet running the standard non-RT Windows is a necessity. New tablets on the market today now come with a 64-bit version of Windows. A 32 or 64 bit processor will not make a difference for this project, but the option is available.

	Lenovo Miix 2	**Microsoft Surface**	**Dell Venue 11 Pro**
Price	$249	$349	$899
Screen Size	8.0 in	10.6 in	10.8 in
Screen Resolution	1280 x 800	1366 x 768	1920 x 1080
Weight	0.77 lbs	1.5 lbs	1.75 lbs
Processor	Intel Atom Z3740	NVIDIA Tegra 3	Intel Core i5 4th Gen
RAM	2 GB	2 GB	4 GB
Operating System	Windows 8.1 – 32 bit	Windows RT	Windows 8.1 – 32 bit
USB Port	Micro USB	USB 2.0	USB 3.0

Table 3.4: Windows Tablet Comparison

3.12 Video Hardware

There are a few things to consider when selecting a camera for this project. Due to size and weight constraints the camera must be small and portable, avoiding adding any extra weight that could result in the turret itself being less portable. The camera will be mounted to a steady and secure location onboard the frame of the turret so it will therefore have no effect on the load that the servos must handle. It must be mounted so that it encounters as little vibration as possible from the movement of the gun and the recoil of the shots being fired. The reason for not mounting the camera onto the gun is that any movement to the camera itself can negatively affect the tracking ability. If the tracking software is looking for any pixels that are changing in order to tell where a moving object is located then the background must remain the same at all times. If the camera is not securely mounted then any time it moves it will be like the entire field of view has become a target and the turret will not be able to function properly.

3.12.1 Cameras

There are many things to consider when choosing a camera for the video subsystem of this project. Wireless vs. Wired and high definition vs. low definition all must be studied in order to make an educated decision on the camera choice. There must also be a strict limit on the budget for the camera because the group's allotted funding was set to less than $500.00 so price is key to the decision for the camera. There must also be a check for compatibility with programming languages and libraries that might be used for the motion tracking. Some programs might not work well with certain types of camera's especially USB connected cameras so when making the final decision on the camera these compatibilities must be checked.

3.12.2 USB Web Cameras

The first and most obvious choice for a camera in this system would be a basic USB wired webcam. The cost of a wired webcam is usually low and they are easy to find and connect to a computer or even the PCB itself. The most important aspect of the camera would be its power consumption, resolution range, frames per second and cost. The first thing to consider when deciding on resolution is what will be best for the tracking algorithms. Since video streaming can result in a massive amount of transferred data the resolution of the camera must be as low as possible while maintaining a relatively clear image. It was decided that a resolution of no more than 640x480 be used which would help keep the data rates low enough.

The next requirement is that the camera must have a sufficient frame rate in order to accurately keep up with a moving target. The group determined that frame rates in the mid to high 30 fps range would suffice for this project. This would give a fast enough refresh rate for the tracking algorithms to track objects traveling at maximum average speeds for a human adult. The third constraint is power and since most wired USB cameras are very low power (<2.5W) and since most companies do not advertise what their products power usage is the group did not use this information to determine a camera choice.

Once that was determined a number of USB webcams were researched and two options were found. The first choice for this project was the Inland brand webcam. It uses USB 2.0 technology which is easy to use and does not require a network connection to communicate with the user interface being used in this project. It has a manual focus which might be useful because some tracking programs cannot work when auto focus cameras are adjusting. It also has up to 30 fps capability and has adjustable resolution. The best part of this camera is the price which is about 16.00 USD at local stores. The second option researched was the Logitech HD C615 webcam. This camera has a higher resolution but can be adjusted down depending on specified needs. The best part of this camera is that it would be very easy to mount due to its size and shape. The only downside is that it is priced high for a wired webcam at about 70.00 USD locally.

3.12.3 Wireless Cameras

Wireless cameras are not as widely available for this type of use as standard USB connected cameras but they are still a valid option for this project. If this project is going to use a wireless camera then there will be more set up required. Some wireless cameras require a network connection to the internet and allow you to view the image through a website. This might cause to much lag in the system and will not be good for a tracking method. Another option is to design a separate board that can be the bridge between the user interface which would accept signals from the camera and send them to the tablet wirelessly. This might prove to be unnecessary work but would be interesting to attempt none the less. A third option is using a webcam that utilizes wireless USB. Depending on the tablet chosen for this project a wireless USB camera could in theory be directly connected to a tablet which would decrease the amount of work required to get the two devices to communicate with each other. If the tablet has a USB adapter on it then the camera could bypass all other circuits and talk directly with the tablet itself which should also improve the speed at which the tablet receives the video signal.

When it comes to options for wireless cameras there were few to choose from but two stood out as a possible option. The D-Link DCS-932L camera stood out as a decent choice. It is small enough to be mounted to the frame of the turret; it also has the proper resolution range with a max of 640x480. The downside is that it is a cloud based camera so it might be difficult to make a camera such as this communicate directly with the user interface or the microcontroller. The upside is that this camera is reasonably priced at about 60.00 USD which is surprisingly affordable for a wireless camera of this quality.

The second choice was the Adesso wireless USB web camera. This camera stood out as a great option because it has a resolution of 640x480 as well as a frame rate of 25 fps. It also would directly plug into the user interface creating a true plug and play set up for a range of up to 10 feet. This range seemed adequate because a person could be considered at a safe range from the turret at 10 feet. Another good thing about this camera is that it is well priced at just less than 40.00 USD. The only downside is that this model is currently set up to run on windows 7, XP, and Vista and does not list windows 8 as an optional operating system. This may cause compatibility issues because most windows based tablets these days are running windows 8.

3.12.4 Infrared Cameras

Infrared technology is very interesting and could be very useful in this project due to its ability to ignore the ambient light in the image and focus on certain IR signatures like body heat. Since the only targets that this turret would consider should be human (or at least be alive) it would simplify the tracking method if an IR camera were to be used. IR cameras use sensors to detect infrared energy (most commonly heat) and then convert that heat into an electronic signal that can be used to create a thermal image on the screen. It should be apparent why this could be useful because it would reduce the amount of background subtraction needed in most tracking algorithms. The cameras that have this technology were researched and a couple of options were found as possibilities.

The first camera researched was the Xeva-1.7-640 produced by Xenics infrared solutions. This camera appears to be very possible to integrate into this project. It is an electronically cooled InGaAs camera with a decent range out to over 45 feet. The benefit is that this camera can also accomplish smooth night vision as well as heat detection and it connects via a USB 2.0 port. This camera would be very useful to this project but no pricing could be found online and due to the fact that similar cameras cost multiple hundreds of dollars it is not likely that this camera will fit the budget of this project.

The second camera found during research fell into the more affordable range was the Wanscam Versio wireless IR camera. This is an affordable infrared camera found on the internet that has many capabilities. It is wireless and runs on the standard 802.11 WiFi which is easy to set up using the included software. The camera has a frame rate of up to 25 fps which should be sufficient for tracking. There are a number of downsides to using this camera that may limit its use in this project. Like most IR cameras the range is very limited. This camera has a maximum range of just 10 meters which would once again be only useful to this project if the turret is to be placed inside a small room. Another disadvantage to this camera is its size, it would be hard to mount anywhere on the turret itself because of the amount of space it would take. Though it might not be the best option its price range is fair at about 50.00 USD after shipping which makes it a possible choice if an IR camera is chosen for this project.

The main problem with using an IR camera is usually price and then the fact that most of them do not have a very long range due to the IR beams that are sent out of the camera. The average working distance of a reasonably priced IR camera is within 50 feet or less than 20 meters. That distance would be ok if the turret was to be used in an indoor setting but not if it were to be used outside; the camera would not be able to match the range of the airsoft gun in turn limiting the usefulness of the turret system. Another reason that an IR camera might not be needed is the fact that software like OpenCv can use prebuilt libraries to filter out regular light spectrums from ordinary cameras. This means that if the project needs an IR capability to help with tracking then buying a specific IR camera might not be required if the proper amount of software and coding is implemented. Table 3.5 below illustrates the differences between the cameras listed above. Judging by the research thus far it seems as though a basic USB 2.0 camera is most likely the easiest and most economical choice for the video subsystem.

	Make	Model	Max FPS	Price	Max Resolution
Wired Webcams	Inland	86200	30	$15.99	1600x1280
	Logitech	C615	30	$69.99	1280x720
Wireless Webcams	D Link	DCS-932L	30	$59.99	640x480
	Adesso	Cybertrack V10	25	$39.29	640x480
Infrared Cameras	Xenics	Xeva 1.7 – 640	25 or 90	N/A	640x512
	Wanscam	Versio	25	$50.00	640x480

Table 3.5 - Cameras

3.13 Remote Control

The remote control research section entails the examination of different methods of relaying a signal using remote control. In the initial phases of the project, the way in which to incorporate remote control was not clear. It was undecided whether the turret system would be completely wireless from table to servos, or if wired connections would be required. As research of a vast array of components commenced, design ideas began to fall in place. As a group, it was decided to use the tablet as the main accessory feature to the main turret system. The tablet would become the remote control portion of the project, consisting of a fully interactive touchscreen user interface which would allow for turret control (a feature only in the manual mode of fire). Because of this, the tablet would be required to be completely wireless, both transmitting and receiving data simultaneously. The tablet would first be receiving data wirelessly from the tracking camera mounted to the main housing. This transmission would show on the tablet as a live video feed. The user could then interact with the tablet to move the servos to a desired position and fire the attached weapon platform, thus performing remote control of the turret.

Other methods of remote control were considered which revolved around the tablet, but chosen not to be implemented. The main goal was to combine wireless remote control with touchscreen capabilities. Additional concepts included the attaching of control devices to the tablet, such as a joystick. However, this would defeat the purpose of using the tablet as the remote control. The tablet would only act as a signal transmitter. Overall, research and deliberation on remote control finalized the implementation of using the tablet as the only remote control device for the sentry turret system.

3.14 Motors and Actuators

Research for the appropriate components tasked to target tracking and firing fell upon three main choices: servo motors, actuators and stepper motors. Each component had varying methods for control. Therefore, to fully understand the function of each, the control mechanisms are examined. After reviewing several different models, it was found that servos could be used for all three devices (X-axis positioning, Y-axis positioning and trigger pull). Actuators were researched as a possible replacement for the trigger pull servo and stepper motors were researched for possible positioning servo replacement. Past projects were referenced as well to find suitable motors that were within the scope of this project.

3.14.1 Servo Motors

Servo motors were the first choice for positioning control and continued to show much promise when further researched. The reason for servo motors beating out the competition is their ability to have precise control. This precision is a key element to the project since tracking a target requires accurate coordinates which must be updated continuously. In addition to having great precision, servo motors are able to handle high speed, high torque applications better than comparable motors while maintaining power efficiency. Servo motors are an excellent choice for this project since the sentry turret platform may need to be turned at high speeds. Also, the power efficiency of the servo motors will not drain an onboard power supply as fast as competing motors.

Breaking down a servo motor, it is an angular positioning motor working in a closed-loop system; perfect for this project's application. The servo works by receiving an input from its current position and another input for its next, or desired location. These inputs are sent every 20 millisecond on average as electric pulses to the motor. Generally, servo motors have an angular range from 90 to 180 degrees (servo motors can have 360 degree rotation) and can switch direction. These motors can cover the angular distance in a short amount of time, too. The servo motors researched were all capable of spanning 60 degrees in under .20 seconds using 6 volts. The load each servo motor was able to hold varied more than speed. Overall, servo motors were found able to carry loads upwards of 10 pounds, making these ideal for the positioning control device.

3.14.2 Linear Actuators

Using an actuator for trigger fire control over a third servo motor was an option being considered. Actuators mainly differ from servo motors in that they work in linear motion, rather than angular. Because of this, an actuator could perform considerably well when pulling and holding a trigger. Actuators can be designed in multiple control forms including electric, hydraulic and pneumatic. An electric actuator was researched since it would need to be connected to the control PCB.

When comparing speed and force, actuators ranked above most servos. However, this was not necessarily an advantage. Searching for a linear actuator within the constraints of this project proved difficult. The main issue was the linear actuator's voltage rating and its pounds of force. The majority of actuators ran on 12V or higher with minimum force of 12lbs. These values far exceed the limits of what would be required to simply pull a trigger back. Since a trigger weight pull would be no more than 2lbs, a servo motor could work in place using an arm-extender to reach the trigger.

3.14.3 Stepper Motors

Stepper motors were researched as alternates for servo motor positioning, but ended up falling short in comparison. Stepper motors were an inexpensive and reliable solution for the servo motor. However, due to the mechanics of a stepper, implementing two of these into the project design would require more troubleshooting than using servo motors. The first downside of a stepper is its inability to return to a neutral point. Unlike servos, which are capable of this, stepper will not return to a specified point unless programmed to do so. Additional disadvantages of this type of motor include power consumption and its open-loop design. Stepper motors require a continuous supply of power even when holding a position and are far less efficient. Thus, they consume and require more power over any given interval of time when compared to a servo motor. This would likely cause modifications to the internal power supply in the main housing, which would not be ideal. An open-loop operated system also has disadvantages when compared to closed-loop control systems for this project. Open-loop systems do not have feedback; therefore the system is not self-adjusting. Thus, a stepper will take the input and produce an output with no looping feedback. On the other hand, stepper motor advantages include cost and speed variation. However, maximum speeds were still in range of comparable servo motors and lower speeds would not benefit the target tracking process. Price differentiation was noticeable but was not enough to warrant using two stepper motors over servo motors for positioning control.

3.15 Wireless Technologies

The goal of this project is to have a turret in any location and be able to control it remotely from a distance. In order for the user to be out of site of the incoming attacker, wireless technologies must be applied. There are two areas where wireless interaction need to take place. The first is between the turret structure and tablet. The second is between the video camera and tablet. While researching wireless equipment, various information was important to consider. Data transfer rate, range, and connectivity time were the top priority when deciding what to apply to the project. With the objective of having a distance of 10 meters between the user holding the tablet and the turret, it was necessary to find the right hardware that could do such a task. Another aspect to consider is the controls must react while attacking a target from such a distance in under 5 ms. The amount of data sent between the video camera and tablet will be very large and must be taken into consideration. The tablet to the processor will be relatively small so the options for that will be greater. With all that to consider, the group researched the most common methods for wireless communication, including Bluetooth, Wi-Fi, wireless USB, and ZigBee. Table 3.6 below compares these protocols.

	Bluetooth	Wi-Fi	Wireless USB	ZigBee
Range	10-100 m	250 m	10 m	50 m
Data Rate	24 Mb/s	54 Mb/s	100 Mb/s at 10m or 480 Mb/s at 3m	250 Kb/s
Power Consumption	Medium	High	Medium	Low
Connectivity Time	10 s	3 s	5 s	30 ms

Table 3.6: Wireless Technology Comparison [2]

3.15.1 Bluetooth

Bluetooth is a short range wireless technology that uses ultra-high frequency radio waves in the unlicensed ISM (Industrial, Scientific and Medical) band at 2.4 – 2.485 GHz between two locations. The most common use is personal connections between PCs, cell phones, mice, and keyboards. Bluetooth has many positives about it, including having a decent data transmission of 24 Mb/s along with a range of 100m (class 1). Disadvantages include a high risk of security issues, high battery drain, and a slow connectivity time of around 10 seconds. There is also a high risk of loss of connection, if any object comes in between the two devices the signal may be loss. Also from personal experience while using a Bluetooth headset for a cell phone, signal would be lost randomly for no apparent reason. There are three different classes for Bluetooth as seen in Table 3.7 below. As you move from class 3 to class 1, the range increases but so does the power consumption and the price.

	Class 3	Class 2	Class 1
Power Consumption	1 mW	2.5 mW	100 mW
Range	1 m	10 m	100 m
Price (per module)	Could not find	$10-20	$65+

Table 3.7: Bluetooth Categories [3]

3.15.2 Wi-Fi

Wi-Fi allows devices to connect to the internet in order to transmit data. With a range of upwards of 250 meters and speeds up to 54 Mbps, Wi-Fi has a lot of positives for the project's video camera transmission challenge. The downsides unfortunately outweigh the upsides. The user must be connected to the internet which means a router running continually is a necessity. Also the user must be within range of said router. For testing purposes when the final design is in effect, the Wi-Fi around the UCF campus would have to be used, which is unreliable at best. If multiple users are using the same Wi-Fi connection then speeds will drop and the video feed may not get to the tablet.

3.15.3 Wireless USB

Wireless USB (WUSB) is another short range form of communication. It allows a much greater amount of data transferred per second than Bluetooth or Wi-Fi. At a distance of up to three meters, the data transfer speeds can be upwards of 480 Mbps. When moved to a distance of ten meters, the speeds can go to 100 Mbps, nearly twice as fast as Wi-Fi. WUSB runs at a frequency from 3.1 to 10.6 GHz. This is the ideal form of communication for the camera-tablet connection. It is also one of the easiest methods of implementation, with no major configuration or set-up for the user. It is vital for the project objective to react instantly to the target passing by. There must be a minimal to no delay in the video feed from the recording camera to the tablet. One requirement will be the tablet must have a USB port and software that can interpret USB. The only disadvantage for WUSB is the range, once the user exceeds the 10 meter range, to data transmit may decrease, lag, or even stop completely.

3.15.4 ZigBee

ZigBee is a low cost, low power short range wireless protocol that creates wireless personal area networks. It uses radio waves at 2.4 GHz to produce a mesh network, allowing it to transmit data over long distances by passing through intermediate nodes. With a range of 50 meters, a super quick connection time, it can transfer data up to 250 Kbps. New ZigBee connections only take 30 milliseconds to connect, while once there is an association the connection time from the sleep to active state is 15 ms. This allows for significant savings in battery life. The ZigBee will wake up, communicate with other devices, and go back to sleep. For STATS, only a small amount a data is necessary for the interface to the processor exchange. Only when there is user input in manual mode will the module need to be running, the rest of the time it can be in sleep mode saving battery.

3.15.5 XBee

XBee is a product brand by Digi International that is almost identical to ZigBee. XBee modules usually come with the ZigBee standard along with added features. They are more user friendly but the convenience comes with higher price. The XBee family includes over 15 different variations, including the XBee 802.15.4, XBee Pro, and XBee Wi-Fi. In most cases XBees communicate with other XBees. The standard XBee is capable of 250 Kbps and a range of 30 m. This type of simple plug and play communication with low data rate is ideal for tablet to PCB communication. For camera to tablet, an XBee Wi-Fi would be necessary. This module allows for cloud connected Wi-Fi with data rates up to 72 Mbps.

3.16 Power System Research

3.16.1 Solar

Ideally, STATS would be a universally implemented device regardless of the conditions. This meant STATS would be used indoors or outdoors, irrespective of lighting conditions. This relates to the power system since it could be beneficial to utilize solar energy in conjunction with an indoor power outlet if available. It was also important to take the size constraints into consideration. It is determined that the battery will be the main power source for the project while the outlet and solar would be a backup with an additional function to recharge the device. As a solar panel in theory sound ideal for a device that would be used outdoors in the Florida sun, the design was over the size constraints. Several solar panels were considered for the design with the smallest of them all measuring to be about 2 ½ feet by 3 ½ feet in length. The area of a panel would require to increase the size of the project tremendously with the weight. Due to the application of the turret, it is vital to be light and mobile with minimal assembly and disassembly. Considering, UL solar 12V off grid solar panel do not supply the minimum desired estimated current. Due to the larger size, each of the servo motors would be requiring 7.4 Volts. Another drawback of using solar panels in the design would be cost. Most solar panels with the specifications desired usually cost around $120.00 on average, and require other components as DC-DC voltage regulators and controllers. This would ultimately increase the price of the project.

3.16.2 Battery

Whenever considering a battery, the following requirements were necessary to maintain proper size and optimize efficiency of STATS. It is important to note that the turret along with the housing will rotate on a bearing driven by the servomotors in each direction. The center of mass and weight ratio will have to be minimized to allow for smaller, more power efficient motors. Although it will be difficult to meet the requirements needed, it is the ability to maintain a reasonable tradeoff which would ultimately mark the decision. The following requirements are:

- Batteries must be lightweight to not slow down the movement of the turret
- Batteries must be the correct form factor.
- Batteries must be reliable for all weather conditions
- Batteries must safely recharge at a steady rate.
- Batteries shall be able to hold a charge for a minimum of 2 hours

It is important to consider the power system of any design. STATS will operate both indoors and outdoors. It is simple and easy to plug STATS into a standard wall power outlet. An obstacle to consider is most outdoors environments are without easy access to power outlets, and therefore internal battery operation would be ideal. Another proposed solution would be using PV panels as a form of solar power. However, the drawback of this would be cost and size.

For a standard wall plug, a transformer will be used to step down the voltage to the power rating required to operate, followed by AC-DC rectification. Several power sources were considered but were left out due to the many components to compensate for. Instead, a power supply will be designed to accommodate the needs of all subsystem including the microcontroller, airsoft gun, webcam, motors, and charge for the tablet. This will help lower cost and insure all devices are meeting the required power ratings. The airsoft gun will ship stock with battery operation. It will already include a rechargeable battery to operate the gun. The battery of the gun is rated to run for about 2 hours on standard operation. Thus the turret battery requirement will be at least 2 hours to match the operation of the gun.

Choosing the battery will depend on the application. Since there will be access to a wall at times, rechargeable batteries will be used to reduce maintenance. The system will be designed such that while the turret is plugged in, the batteries will charge and turret will operate at the same time. When looking into the different battery technology, automatically several batteries were ruled out due to not having the ability to recharge. It was concluded that a Lithium ion battery would be best choice since it contains a great power to weight ratio, and the fast recharging time. Standard AAA, and AA size Lithium ion were first looked into for including the Energizer power plus rechargeable battery packs. The batteries would deliver sufficient power to the load. But with having to buy several sets of batteries, along with a charger, it would begin to cost beyond what was initially budgeted for. In addition the energizer battery packs required to be charged, using the charger supplied by energizer would increase maintenance. Instead, a more ideal option was to use the SANYO 12v 1100mAH lithium ion battery from airsplat.com. Although the price of the battery is $39.99, it does not require to buy any external chargers and could be charged while the battery is on the device. Depending on the funds it is more likely two batteries will be purchased to allow for a backup in case a battery goes out. Table 3.8 illustrates the 12V 1100mAh battery from SANYO. Dimensions of the battery along with power ratings can be found in the table below.

Capacity	Primary battery	Secondary Battery
Voltage	12 V	3.7 V
Max charge current	1.35 Amps	1 A
Mililiamp hours	1100 mAh	1000 mAh
Discharge cut off voltage	13 V	4.2
Cell count	10	1
Cascaded Length	4 inches	NA
Length	12 inches	3 inches
Width	1.15 inches	5 iches
Cascaded Width	2.8 inches	NA
Weight	4 lbs	1 lb

Table 3.8 - Battery Table

Due to several components that require independent sources, including the fuel cell monitor and sub-systems, the system will run separately off the board. This is including warning module, XBEE wireless shields, and a low specification battery that will be required to drive there components. Several options besides off the shelf AA batteries were to use a polymer lithium ion battery from sparkfun.com for only $8.95. The specifications of the battery can be found in table 3.8 above. Although the battery can discharge up to 1 Amp, only a minimal amount of current would be used since the battery would be used as a secondary to drive most of the LED's and solid state speakers. Again, the battery would have to be rechargeable and would be charged using the same recharge circuit.

The tables below displays the components to be mounted on the printed circuit board. Table 3.9 shows the power ratings for the primary PCB which will contain all the main control units and motors. The expected total power consumption is about 5 watts, while the proposed battery supplies about 4.90 watts. Table 3.10 also shows the power consumption of all the components that will utilize the secondary battery. It is evident here that the XBEE modules along with the motors consume the most amount of power as proposed. Things to consider is the maximum current rating is used for the power calculation. Generally though the components will not all be operating at the peak current throughout the execution of the turret. Normally the battery will be able to provide the power if all components are using the maximum current, but ideally the sum of the power would be around 40% less.

Device	**Operating Voltage (V)**	**Max Current (mA)**	**Power (mW)**
Atmel Mega 328	5	50	250
HS-5685MH Servo Motor 1	7.4	170	1258
HS-5685MH Servo Motor 2	7.4	170	1258
HS-5055MG Servo Motor 3	4.8	70	336
XBee Pro	3.3	230	759
MAX1704	3.7	60	222
AVR Mini Programmer	2.6	50	130
LT1505	2.5	80	200
LED Lights	4	120	480
		Total Power	**4893**
		Proposed Battery Power	**4850**

Table 3.9 – Power Ratings of Components Used for Primary Battery

Device	Operating Voltage (V)	Max Current (mA)	Power (mW)
AudioLarm II	3	40	120
XBEE WiFi	4.2	230	966
Web Camera	3.3	215	709.5
LED Lights	3.3	120	396
XBEE Pro	3.3	230	759
		Total Power	2950.5
		Proposed Battery Power	3000

Table 3.10 – Power Ratings of Components Used for Secondary Battery

3.17 Recharge Circuit

A recharge circuit will be used to ease maintenance and implement a safe and proper method of recharging the battery. This will have to be able to link with the fuel cell gauge. Out of the many methods used to charge batteries, the CC/CV method which utilizes a constant voltage and constant current to increase the voltage on the battery was utilized. When the battery is plugged in, a constant charge will be directed to the battery. The current must be lower than the maximum current of the battery to maintain safety and proper charge. With constant current, the voltage will have a constant rise in voltage. When the maximum voltage is reached, the battery will stop charging unless the battery charge falls low again. Lithium ion batteries can be dangerous if proper procedure is not taken in to account for. Maintaining proper upkeep and charging procedure will prolong battery life and at times will maximize battery life past its manufacturer suggested lifetime. A device called the MAX1555 utilizes either an external DC power plug, or a mini USB to charge the battery. Standard mini USB supplies 100mA while a typical DC supplies 280mA. This is critical as the 100 mA would be too low for the SANYO 1100maH charge battery chosen. Therefore, a 280 mA wall outlet plug in charger will be used as the primary charge source. The DC sources will be wired to a mini-B USB head since they are easily found.

3.17.1 Fuel Cell Gauge

Since the turret is devolved to operate as a defensive system, it was desirable to minimize faults or any delay in operation. Since the motors operate at variable voltages it will be difficult to not only supply the optimal voltage at all times but to ensure the user is aware that the optimal voltage is low, therefore depicting a low battery. This is possible using a fuel cell battery gauge. Very little was found for resources on fuel cell gauges, as not too many semi-conductor companies manufacture them; and neither do many hobbyist and/or projects include them since of the early prototyping stage. A fuel gauge provides the SoC(state of charge), along with the SoH(state of health) of the battery. With the fuel cell gauge the remaining time left, remaining capacity, average current along with temperature can all be read from the battery and sent to the controller. There are several methods used to find this information which include coulombs count, and the battery discharge curve. Some issues with coulombs count is that it does not take into account the batteries life. This can result in an incorrect reading. The better choice would be the battery discharge curve which uses impedance tracking technology to keep up with the current. The downfall of devices as such would be cost. A more popular fuel gauge monitor is the Bq347110 manufactured by TI at a cost of $29.16. Unfortunately, this option is unrealistic when taking into account other components to build. A more convincing option is the MAX1700x which is specifically developed for microcontroller applications. The IC can be found for $2.87 from Mouser Electronics. The Max1700xx has 7 pins compared to 12 pin Texas instrument BQ347110 IC.

3.18 Mounting Options

Research required for mounting options included not only the component which would attach the weapon platform to the main assembly, but also the servo armatures as well. Since these parts will be the only ones to have to support the moving functions of the turret system, the ideal build material is lightweight and durable for the armature. The weapon platform mount should be a rubberized component.

There were two primary materials in researching the servo armatures. The first being metal, with the second being wood. During the building phase, wood would be a better option due its tolerance and cost. Several different models could be mapped out before determining one that would work best for the project. On the other hand, metal armatures would have greater durability but cost would increase. Therefore, it was decided to use wood for the alpha designs. Once a model was determined to be suitable for positioning control, metal would be substituted in for the final build process.

On the top of the servo armature, the weapon platform clamp is attached. The material for the clamp would consists almost entirely or rubber. Many home improvement stores carry a variety on clamps that would be suitable for the project. The main challenge was finding one large enough to encompass an airsoft or paintball gun, while still being able to hold smaller object such as flashlights or lasers. The rubber clamp would function similar to a wrench; able to adjust its tightness based on the size of the object.

3.19 Weapon Platforms

Because of a group member's former knowledge on the available weapon platforms for the sentry turret system, extensive research was not required to acquire a suitable model for the project. Research became limited to searching for an airsoft gun which would meet the specifications and requirements. With the airsoft model of choice leaning towards an SMG, other models were considered, but not researched in depth. The goal was to find a relatively inexpensive SMG airsoft gun that could meet the form factor and velocity mapped out in the specifications table. With a very limited selection of airsoft guns sold locally, ordering from an online store broadened the assortment of SMG models. When comparing SMG models, the most important factors to look at were muzzle velocity, battery size and magazine capacity. Search filters were put in place to only show airsoft guns with minimum velocities of 280 FPS. With any airsoft gun pushing velocities of 280 FPS or higher, metal gearboxes are usually used within. This is an additional feature that is not necessarily needed, but will add to the longevity and robustness of the SMG. Magazine capacity does have a factor in the duration of firing. Most AEG models purchased come with one magazine of high capacity, or hi-cap. A hi-cap magazine holds anywhere from 150 BBs to 300 BBs. It functions by utilizing a spring wheel located on the bottom side of the magazine. When spun, the BBs will become wound up in the internal storage of the magazine, letting the user fire off the entire clip without the need to reload the airsoft gun.

Battery size for airsoft guns come in different physical configurations and have a variety of power options. The three main physical battery configurations are neutral, stick, and nun chuck configurations. The reason for the different battery dimension depends on the space allocated for the battery in the internals of the airsoft gun. The normal voltage ratings for airsoft batteries start at 7.4V for AEGs and can surpass 11.1V. These higher voltage ranges are usually made for heavily modified AEGs, though. For a stock SMG airsoft gun, battery configuration and voltage rating is typically a stick form battery with voltage of 7.4V or 8.4V. Do note that nun chuck and neutral battery types can be used with an SMG if the power ratings are optimal. However, this would require the battery to most likely be mounted externally. Airsoft batteries' voltage rating will go hand in hand with a capacity value as well. This capacity is given in mAh (milli-ampere-hours). This number can range anywhere from 500mAh upwards of 5000mAh. For the purposes of this project, a capacity rating of 1200mAH or higher will be sufficient. With regards to an SMG airsoft gun, the ideal battery would be a stick type configuration rated at 8.4V and 1600mAh. Because of the duration testing may take, multiple batteries were purchased to avoid downtime when switching between the different tests required.

With baseline requirements set, reputable online airsoft stores were researched to find any airsoft guns within range. After browsing multiple stores, airsoftatlanta.com was chosen as the supplier due to their larger selection of SMG models to choose from and location. The store contained many submachine gun models under 200 dollars which also met minimum muzzle velocity requirements. From airsoftatlanta.com the SMG models were narrowed down to three different choices. The first SMG model was a Heckler & Koch (H&K) MP5K Competition AEG. This SMG met all the requirements for the project, but was also the most expensive of the three choices. The next two SMG models came in under 100 dollars, but with features that would possibly require modification in order to work. However, because of their price point, form factor and weight, the following two SMG were on par with the MP5K. Both models were built by the same manufacturer, JG, known for producing quality airsoft guns to compete with comparable models typically at double the cost. The SMGs from JG to be examined were the Scorpion VZ61 and the MAC-10. Both of these models were the 2013 edition and featured improvements from the previous models (this includes metal gearboxes). The one downside of both JG SMGs was their classification as an AEP SMG. AEP, standing for Automatic Electric Pistol, is still considered an SMG by design, but AEP models typically use smaller battery sizes. Hence, the voltage rating would be lower and the battery capacity would not have the length of charge as a normal sized battery. Fortunately, most airsoft guns do not require only one type of battery to function. Many airsoft enthusiasts will buy stock models of airsoft guns, and begin custom modifications. This almost always includes the battery. With further research of the JG AEP models, it was determined that their stock battery was able to be scaled up to a normal size battery with a longer life. Therefore, other than a battery modification, the two JG airsoft guns were directly comparable to the H&K MP5K. To map out all the features and specifications of the airsoft guns, the following table below has been produced. Please note, this is only including the base purchase of the airsoft gun. Separate accessories will be bought depending on which airsoft SMG is chosen for the final build. Additionally, this table reflects the airsoft gun choices available at the time this document. Some of these models may not be available for purchase, or new and improved models will be manufactured after release of this document. Therefore, these SMG choices cannot fully represent the final purchase for this project, but give a good indication of what will be purchased.

	H&K MP5K	**JG MAC-10**	**JG Scorpion VZ61**
Price	$149.99	$109.99	$99.99
Muzzle Velocity	~340 FPS	~280 FPS	~290 FPS
Horizontal Length	15.3 inches	12 inches	11 inches
Vertical Length	12 inches	13 inches	9 inches
BB Weight	0.20 grams	0.20 grams	0.20 grams
Battery Specifications	8.4V Stick-type	7.2V Mini	7.2V Mini
Magazine Capacity	200 Round Hi-Cap	450 Round Hi-Cap	70 Round Hi-Cap

Table 3.11 – Airsoft Gun Options

After reviewing all three models, the H&K MP5K or the JG MAC-10 will be used for the prototype. The JG Scorpion is excluded due the magazine capacity and battery location. The JG MAC-10 can still compete with the MP5K, even with the battery flaw. The dimensions on the MAC-10 are ideal for mounting it to the servo armature. In addition, the magazine capacity is more than double the MP5K. Muzzle Velocity falls behind slightly, but is not significant enough to deter this model from being a valid choice. Though, in order for the MAC-10 to be the prototype choice, a battery modification will be required.

Because of the ability of the MP5K to 'plug-and-play,' this airsoft SMG is the front runner. Other than price and magazine capacity, the MP5K marks well in all categories. The battery life is sufficient and muzzle velocity is far over the minimum value. Furthermore, the added length of the airsoft gun gives it the longest internal barrel length of all three choices. Increase barrel length will add accuracy to every shot fired to increase the hit-miss ratio. Depending on which airsoft gun is chosen for final design, an additional magazine and battery will be purchased. This will ensure that all testing will proceed with minimal interruptions due to the charge of the battery being depleted or the capacity of the magazine running empty.

Overall, the main reasons for not choosing a paintball marker for the prototype include cost, additional larger components, and ease of testing. Compared to airsoft guns, paintball markers contain more components, which raise the overall price of the platform. As stated before, these additions also add weight and volume. Overall size is a factor in design, and though a paintball marker can be mounted, it is more bulky. Because of the ammunition used, paintballs require more clean up if testing would be done (dry-firing the gun is an option, but is not recommended). The air tanks will also need refueling. This can only be done at professional shops or paintball retail stores. Compared to paintball, charging an airsoft gun's battery can be done from any standard 120 volt wall outlet. This can eliminate much of the downtime between testing. Generally, paintball equipment requires more maintenance and is more expensive than airsoft, making it non-ideal for the prototype. The benefit of choosing an AEG for this project is in the gun's compact design. The magazine that holds the BBs is smaller in comparison to a paintball hopper and is attached underneath to the airsoft gun. The battery, which is the AEG's source of power, is usually housed internally via a 2-pin Molex connector.

Section 4: Hardware and Software Design Details

4.1 Project Design Architecture

At the conclusion of specifications, requirements and research, the project will progress to the hardware and software design stages. In this section, all design details, including diagrams and flowcharts for all sections, will be included. The following hardware and software design details represent the direction of the project at the time of this documentation and are subject to change.

4.1.1 Hardware Block Diagram

The hardware block diagram shown below illustrates the implementation of all hardware components

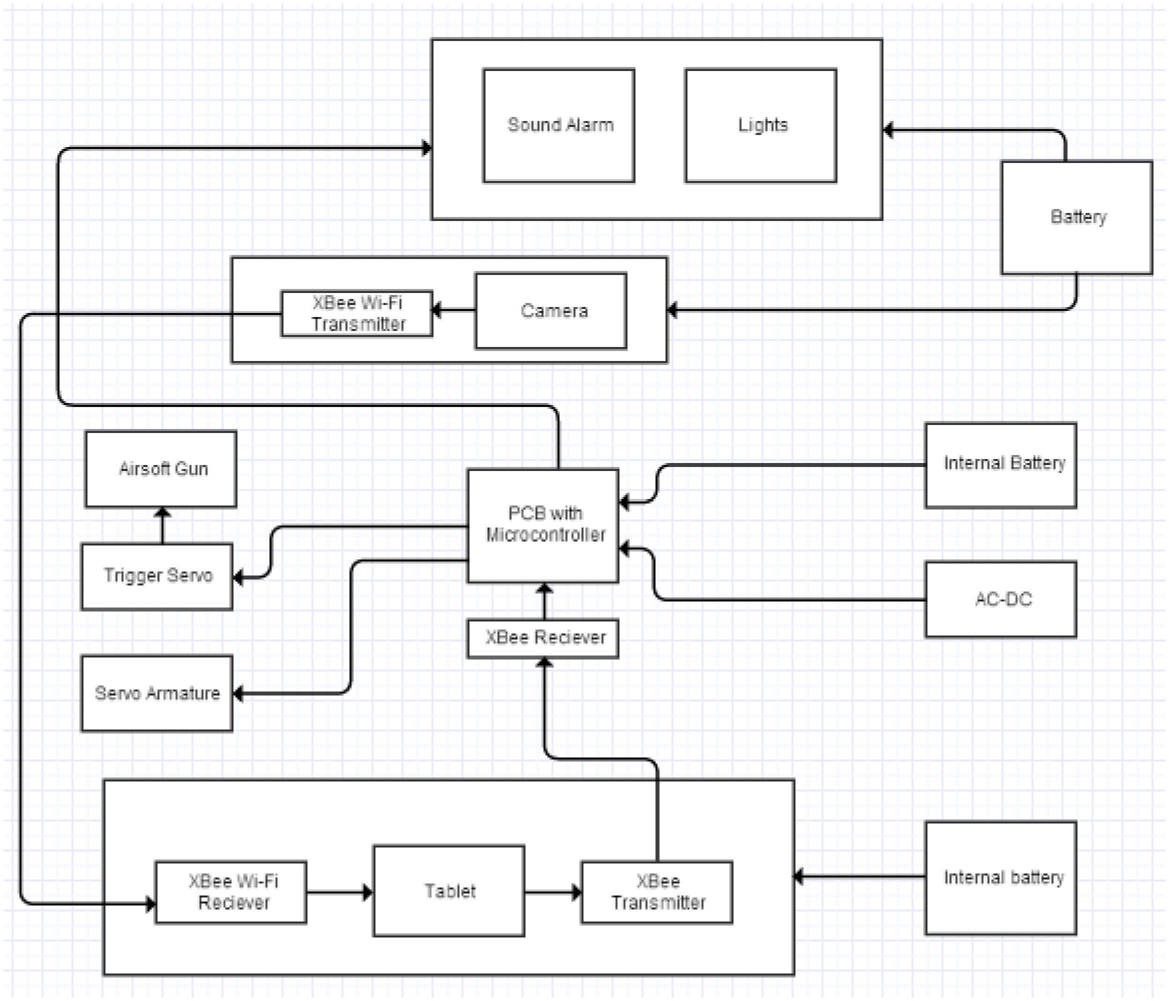


Figure 4.1 – Hardware Block Diagram

4.1.2 Software Block Diagram

The tablet device will be responsible for all the software requirements. By using the Atmel ATmega328 microcontroller the group was able to utilize the open-source Arduino library. There are abundant pre-existing Arduino libraries and functions that will help with this project's goals. The tablet device will house all the software responsible for the turret project from tracking to automatic firing to manual control. The process begins with the camera sending the video stream to the tablet for user display. While in automatic firing mode, the tracking algorithms are waiting to encounter a moving target. Once that happens the range finder determines the target's distance and sends the direction and speed signals to the microcontroller in order to move the servos. Once manual mode is activated, signals are sent to the microcontroller based on the user input.

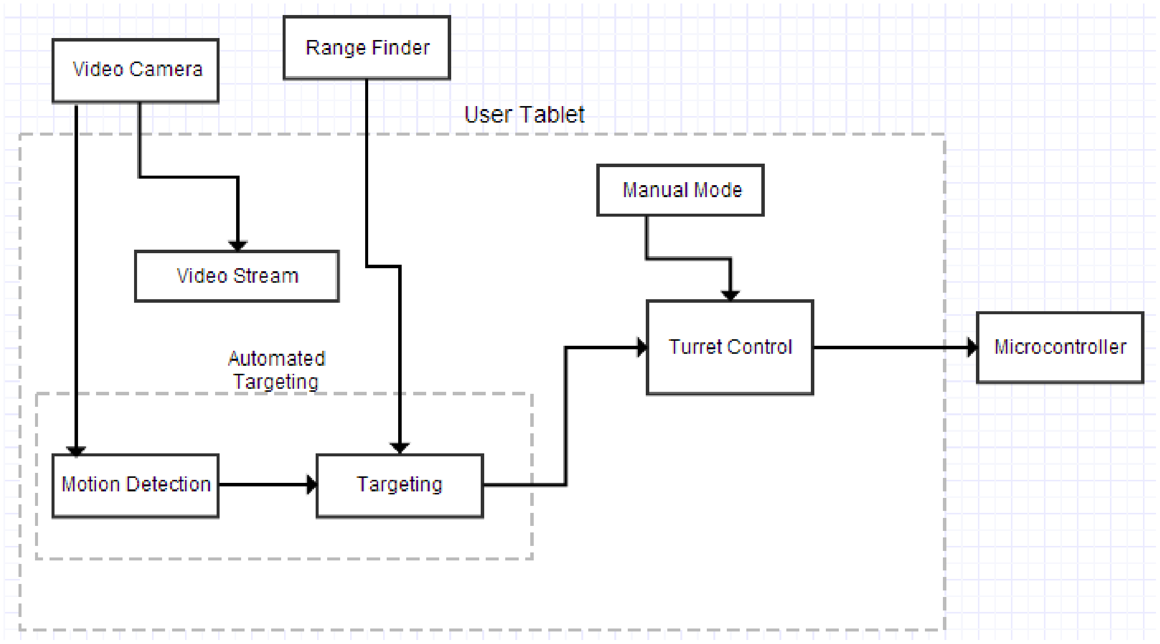


Figure 4.2 – Tablet-Software Communication Block Diagram

4.2 Main Housing

The main housing of the sentry turret system is the foundation for which all additional hardware structures will stem from. The design, as mapped out in the specifications and requirements section will be square-like in design with the back edge slanted for the tablet. The top surface of the housing will hold the PCB, servo armature and the tracking camera. Material used for the frame will be wood painted black. The housing will be built by hand, not preassembled. No components will be welded to the main frame. Only screws will be used to lock everything in place (and remove if necessary). Below the surface, the housing will contain internal space for the power supply. The top surface will have holes drilled to feed wires through to the power supply. All wiring will be tied down and color-coded to keep the surface clean and avoid confusion. The following diagram below depicts a top view of the surface to illustrate the position in which all the components will be laid out. The power supply shown in dashed line, will be stored in the back right corner with respect to the x-axis positioning servo. The power supply, shown in dashed line, will be stored it the back, right corner with respect to the x-axis positioning servo.

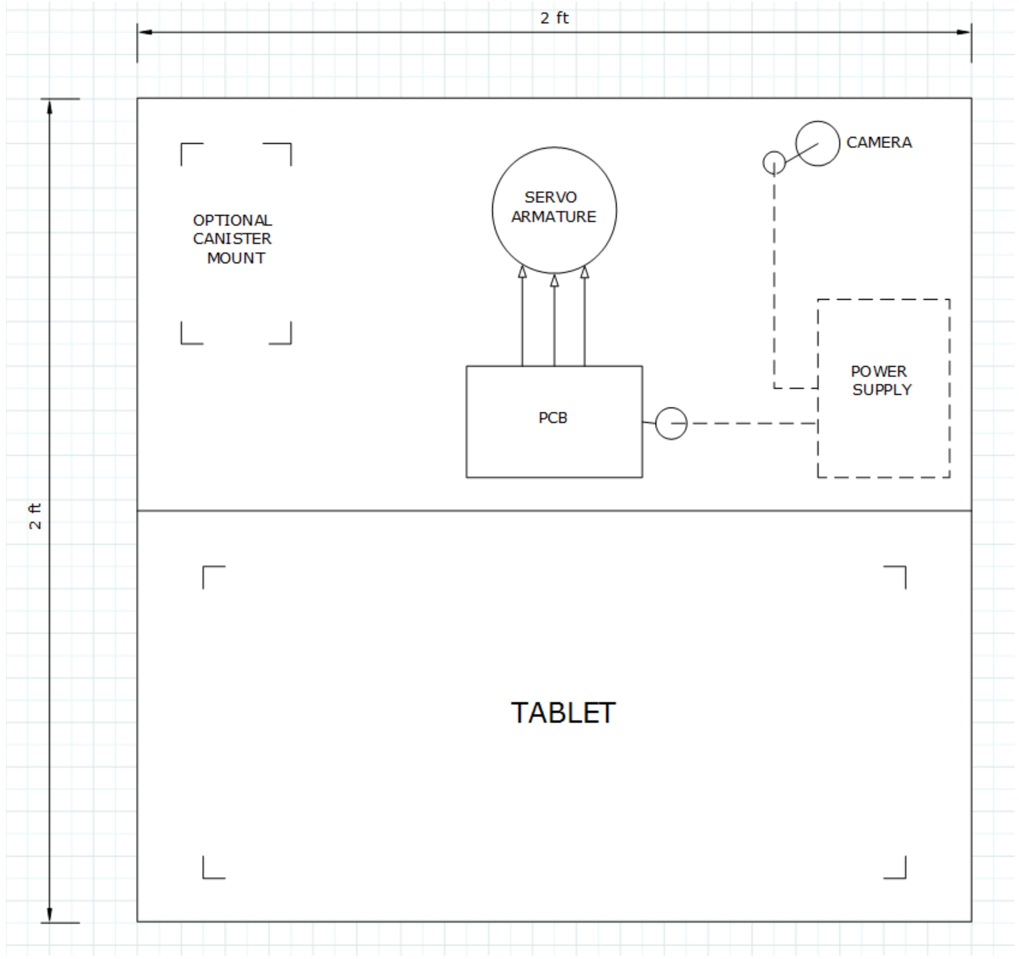


Figure 4.3 – Main Housing Top View

4.2.1 PCB Housing

To keep the PCB protected on the top of the main housing, a separate smaller cage will be used. For presentation purposes, the housing will be transparent to allow viewing of the custom built PCB and its features, but still provide ample securement. The PCB housing will constructed of Lexan, a substitute for Plexiglas. Lexan material can be purchased from home improvement stores and modified accordingly to house the PCB. Holes will be drilled into the sides of the housing to allow access for the power and servo control wires. The top of the cage will contain a latch for easy access to the PCB when assembly and disassembly is required.

4.3 Turret Design

The turret design will be broken down into three main sections consisting of the servo armature (including the weapon platform mount), the main housing, and the support legs. Based off of the requirements and specifications and research, the final design build will be modular, mobile, and accessory friendly. The aesthetic design will be robust, yet streamlined; representing a more militaristic look. For the initial design stage of the servo armature, material used will consist of wood. When an acceptable model is complete, the design will be represented in its final form using metal. The main housing will consist entirely of wood with all appropriate modular attachments for the camera and tablet. The support legs will be connected to the bottom of the housing and will be retractable in design. Material for the legs will consist of metal with rubber ends making contact with the ground surface for stability control. Below are two design diagrams for the sentry turret system. The first diagram represent a broad view of the system with the second diagram consisting of two close up views of the servo armature. Diagrams were designed using Google SketchUp.

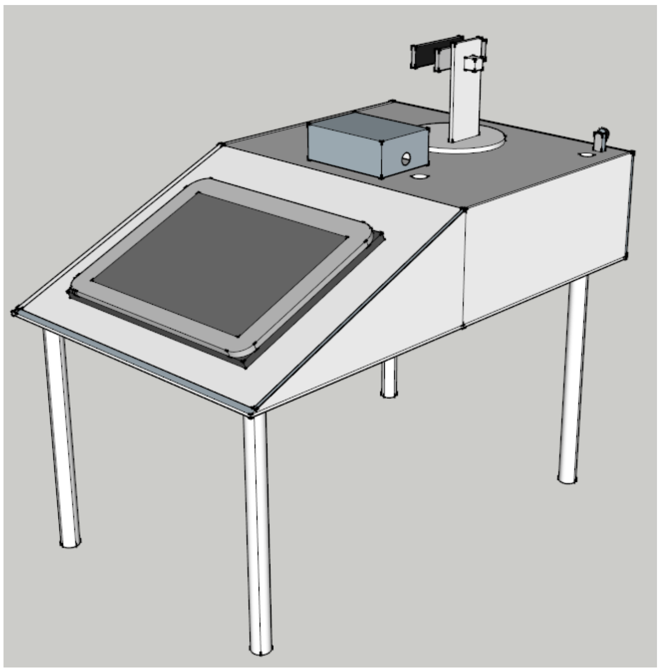


Figure 4.4 – Main Housing Overview

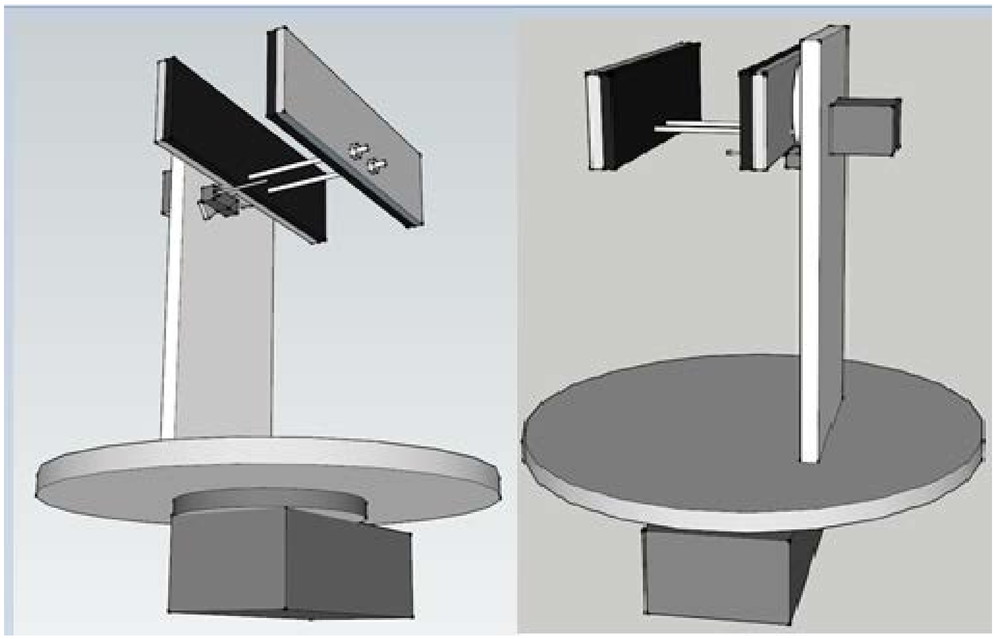


Figure 4.5 – Servo Armature Mount

4.4 Motors

From the results gathered in the research section for motors and actuators, servo motors were chosen as they were proven to be the most promising with regards to the design of this project. Using the servocity.com, the website contains a page for calculating the power and speed of the servo motors. Because the prototype design will house a weapon platform, as opposed to an object of negligible weight, the choice servo motor would have to compensate for the greatest weight the system could hold. If lighter servo motors were purchased, the design runs the risk of damaging the motors and target tracking would lag behind. Based on the webpage, the servo motor speed was calculated as the time taken to rotate 60 degrees and power was calculated in terms of (oz-in.) / 16 = pounds of force for an armature of 1 inch. Therefore, for an armature of greater length, simply divide the pounds of force by the total length of the armature.

The servos chosen for directional control were the digital Hitec HS-5685MH from servocity.com. Digital was chosen over analog for even faster response times and great performance out of the box with no additional programming needed. These servo motors are able to run on 4.8V to 7.4V. At 6 volts, the servo motors are capable 157 oz-in of power, which is close to 10lbs per inch. Additionally, the motor is rated at a speed of 0.20 seconds per 60 degree turn. Servo features include 180 degree turn radius, a neutral point, and multidirectional control. Two HS-5685MH servo motors will be integrates into the armature design; one for horizontal movement and another for vertical movement.

The design of the trigger servo will not need to compete with the specifications of the directional servos. The main design feature for this servo is the ability to pull a trigger with up to 2lbs of force and come in a small form factor. Using servocity.com, an appropriate servo would be the Hitec HS-5055MG. This servo motor comes in the sub-micro size, comparable to a quarter in size. At 6V, this servo is able to produce over a pound of force with its metal gear box. The servo is also able to span over 120 degrees (60 degrees both clockwise and counterclockwise), which will be more than a sufficient margin for trigger pull. The design of the trigger servo will allow it to be attached close to the y-axis positioning servo to grant it access to the trigger of the mounted weapon platform.

4.5 Software Design

For different aspects of this projects there will be different programming languages used. The first is called Processing which is an open source programming language and development environment. The language builds Java, but uses a simplified syntax and graphics programming model. It uses a sketchbook environment to create graphical and visual programs. Processing language is used for the object detection, tracking, and video streaming. The second is the Arduino IDE for tablet to microcontroller data transmission. By using the Atmel Mega328 microcontroller and creating a custom Arduino, the group is able to utilize the open-source Arduino environment.

4.5.1 User Interface

Through immense research into how to create motion tracking through a simple camera input the group agreed upon using the programming language Processing and its JMyron library to implement the user interface as well as the tracking methods required for this project to work. The reason the Processing programming language was the top choice is because there are a significant amount of open source tracking algorithms available that fit the needs of this project. Most prototyping for this design has been done using a laptop computer running Windows. The decision was made to continue using this method of tracking and target acquisition for the final design, which is why a Windows based tablet was chosen for the project. This allows the group to focus more on fine tuning the open source code instead of having to redesign the code from scratch to fit into an IOS or Android tablet operating system.

The user interface will be developed in the Processing program language and will display a simple yet clear option screen for the user to view. The first step in creating a user interface is creating windows that will house the commands, buttons, as well as the video feed from the camera. The first step to this is to create a main window that will be the lower layer of the UI. On top of this window will sit a separate pane that will house the video input as well as the touch sensitive selector boxes that will allow the user to switch between auto and manual modes, select a fire weapon command, and possibly enable an audio/video warning system. A prewritten Java GUI library is used to pull up button panes and image windows in the correct format. Once the GUI library is included, each UI object must be created, labeled and formatted to work. The main function to create the user interface is described below.

Function:	drawControlPanel()
Description	Method the creates user control panel
Parameters:	**colorScheme:** Contains color scheme for main panel **panelMain:** Main window used as destination of buttons and labels **button_Auto:** Selector Button for autonomous tracking, Boolean statement initiated to true when auto is selected. **button_Manual:** Selector Button for manual tracking, enables use of virtual D-pad which acts as a directional control for the on screen crosshairs. Click event sets Boolean statement to false. **button_Alarm:** Alarm toggle switch linked to a small library of .wav audio snippets, played randomly each time a mouse click event occurs on alarm button. **button_Fire:** Manual fire control linked to click event on "Fire" button. Attached to pin locations corresponding to trigger servo. **button_Standby:** Sets a Boolean function to true which enables a loop, pausing the system until Boolean reset to false. **label_DPad:** Label for directional control **label_Fire:** Label for fire button

Table 4.1 – Draw Control Path Function

This user interface class will be called at the same time as the main tracking program and will have to launch before the sentry will activate. This class will have much greater detail involved when it is actually created. This function will depend on imported files such as the GUI library as well as the necessary microcontroller information so that button clicks can send commands that will affect the turret directly. The end result will be similar to Figure 4.6 below, noting that buttons, video input, and manual control positions may be altered in final design.

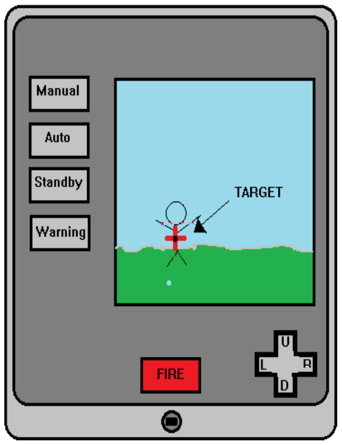


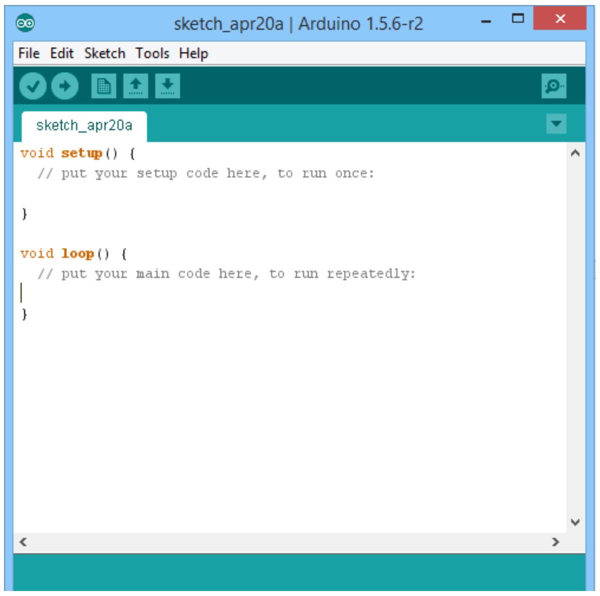
Figure 4.6 - Tablet Based User Interface Design

4.5.2 Arduino IDE

To program the custom-built microcontroller, the Arduino IDE version 1.5.6-r2 will be used, which is the latest version currently. This open source IDE from the Arduino website is written in Java. It can be run on Linux, Mac, and most importantly Windows, including 32-bit. By using the default environment it makes writing code and uploading it to the board in an easy fashion that is aimed at beginner programmers. The language the user programs in C or C++. The IDE comes with a library called Wiring which makes many common input/output operations very simple to use. The basic Arduino code must have two function, setup and loop, described below:

- setup(): a function run once at the start of a program that can initialize settings
- loop(): a function called repeatedly until the board powers off

Figure 4.7 below is a screenshot of a new Arduino program using the Arduino IDE. As seen in the figure a new program comes with setup() and void(). Programs written in this environment are called "sketches" and are saved with the file extension .ino. Also contained in the IDE is a compiler and options for different Arduino board options, for those that are using actual Arduino boards.



```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Figure 4.7 – Arduino IDE Interface

Copyright Arduino. Used under Fair Use

4.5.3 Software Development Life-Cycle

For the turret project the software design method was chosen to be agile. Agile development implies quick development to reach a prototype stage where continual testing and revising will occur. This does not infer that there is no planning, the goal of the group's agile method is to get some sort of working program up and running swiftly. Once an initial program is running this allows for adjustments on a larger scale. Also when it comes to the testing stage, this will allow for quick fixes on issues like servo aiming. Agile requires constant communication between group members as changes to the code will be regular. A basic outline of the agile design method is shown in Figure 4.8 below.

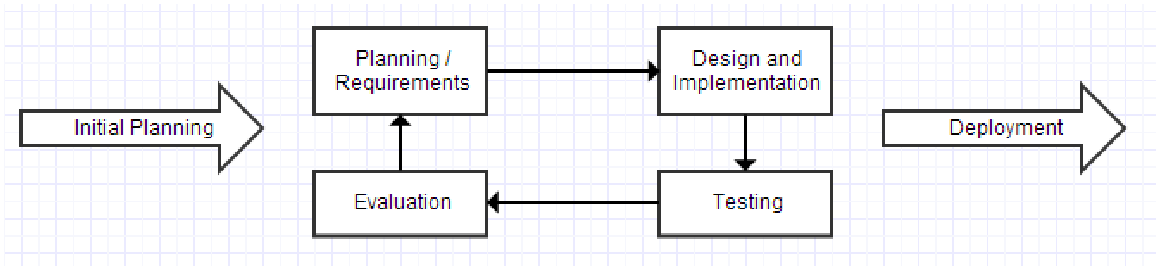


Figure 4.8 – Agile Software Development Method

4.6 Targeting Control

4.6.1 Servo Control Library

The group was able to utilize the open-source Arduino environment and prebuilt libraries by creating a custom version of Arduino on PCB. One of those prebuilt libraries include the Servo library. This library allows up to 12 servos, each at various angles between 0 and 180 degrees. Below in Tables 4.2-4.7 are an outline of the various functions that will be applied from the Servo library.[4] By applying these functions in the Arduino IDE, the below six methods are all that is needed for servo control.

Function	attach()
Description	Attach Servo variable to pin
Parameters	servo: a variable of type Servo pin: the number of the pin that the servo is attached to min: the pulse width, in µS, corresponding to the minimum (0-degree) angle on the servo max : the pulse width, in µS, corresponding to the maximum (180-degree) angle on the servo)

Syntax	servo.attach(pin)
	servo.attach(pin, min, max)

Table 4.2 – Arduino Servo Library Attach

Function	write()
Description	Writes value of angle (int) in degrees to servo
Parameters	servo: a variable of type Servo angle: the value to write to the servo, from 0 to 180
Syntax	servo.write(angle)

Table 4.3 – Arduino Servo Library Write

Function	writeMicroseconds()
Description	Writes value of angle (int) in µS to servo
Parameters	servo: a variable of type Servo uS: the value of the parameter in microseconds (int)
Syntax	servo.writeMicroseconds(uS)

Table 4.4 – Arduino Servo Library WriteMicroseconds

Function	read()
Description	Read the current angle of the servo (the value passed to the last call to write())
Parameters	servo: a variable of type Servo
Syntax	servo.read()
Returns	The angle of the servo, from 0 to 180 degrees

Table 4.5 – Arduino Servo Library Read

Function	attached()
Description	Check whether the Servo variable is attached to a pin
Parameters	servo: a variable of type Servo
Syntax	servo.attached()
Returns	True if the servo is attached to pin; false otherwise.

Table 4.6 – Arduino Servo Library Attached

Function	detach()
Description	Detach the Servo variable from its pin
Parameters	servo: a variable of type Servo
Syntax	servo.detach()

Table 4.7 – Arduino Servo Library Detach

In order for the board to communicate with the tablet, a second Arduino library must be incorporated. The Arduino Serial library allows transmission between the microcontroller and USB port on the computer which the computer to control the servos, siren, and any other apparatuses the group decides to add. Even though this project requires wireless transmission between tablet and board, this Serial library will be included. This library sends signals to the board as character data or in bytes. Below in Table 4.8 - 4.13 describes a few of the assorted functions that will be used by the Serial library.[5]

Function	available()
Description	Gets the number of bytes available in the serial buffer
Parameters	None
Syntax	Serial.available()

Table 4.8 – Arduino Serial Library Available

Function	begin()
Description	Sets the data rate in bits per second (baud) for serial data transmission
Parameters	speed: in bits per second (baud) - *long* config: sets data, parity, and stop bits.

Syntax	Serial.begin(speed)
	Serial.begin(speed, config)

Table 4.9 – Arduino Serial Library Begin

Function	end()
Description	Disables serial communication
Parameters	None
Syntax	Serial.end()

Table 4.10 – Arduino Serial Library End

Function	read()
Description	Reads incoming serial data
Parameters	None
Syntax	Serial.read()

Table 4.11 – Arduino Serial Library Read

Function	write()
Description	Gets the number of bytes available in the serial buffer
Parameters	val: a value to send as a single byte str: a string to send as a series of bytes buf: an array to send as a series of bytes len: the length of the buffer

Syntax	Serial.write(val)
	Serial.write(str)
	Serial.write(buf, len)

Table 4.12 – Arduino Serial Library Write

Function	flush()
Description	Waits for the transmission of outgoing serial data to complete
Parameters	None
Syntax	Serial.flush()

Table 4.13 – Arduino Serial Library Flush

4.6.2 Multidirectional Control

The servos receive the coordinates of the target from the tablet which is sent through wireless transmission. The coordinates sent from the tablet are analyzed and converted from an analog signal to digital by the Arduino Library analogWrite() function. For these servos to have smooth movements, a PID (proportional-integral-derivative) controller will be used. Figure 4.9 demonstrates how the servo system will be set up. The basics of it will be first the tracking algorithm or the user while in manual mode send the signal to the Arduino Library to be shifted. This initial analog signal gets converted to digital which then gets sent to the PID controller. The PID will run its algorithm then send the final signal to the servos. The reason a PID controller was decided to be implemented was their ability to calculate the error between two points of the servo movement, allowing for more accurate results. Attached to the servo, the signals sent from the tablet will go to the PID controller before forwarding the more precise direction to the servo. The PID controller is a closed loop feedback process that will give the device better response time, along with reducing the steady state error and sensitivity to the inertia. Improving response time and reducing steady state error will make the servos act faster and more accurate, giving the user a better chance at hitting their target.

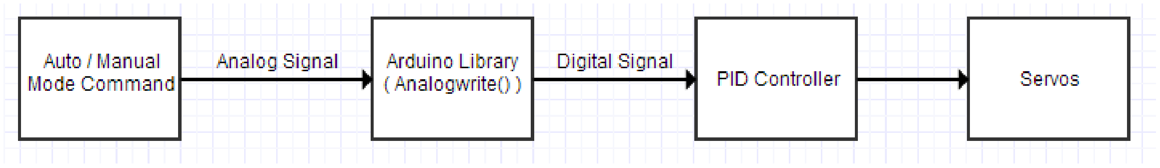


Figure 4.9 – Servo Control System

The PID controller is made up of three algorithms: P (proportional), I (integral), and D (derivative), summarized in Table 4.14 below. The derivative algorithm will be especially important for this design because of the fast moving targets and instantaneous movements that will be necessary. Figure 4.10 from Parker Motion the servo-PID controller control system.

Term	Function	Effect on Control System
P	$K_p * V_{error}$	The main drive in a control loop. K_p reduces overall error.
I	$K_i \int V_{error} dt$	Reduces the final error of the system. Summing even a small error over time results a drive signal large enough to move the system toward a smaller errors
D	$K_d \frac{V_{error}}{dt}$	No effect on final error. Counteracts K_p and K_i when the output changes quickly. Also helps reduce overshooting.

Table 4.14 – PID Controller Algorithms [6]

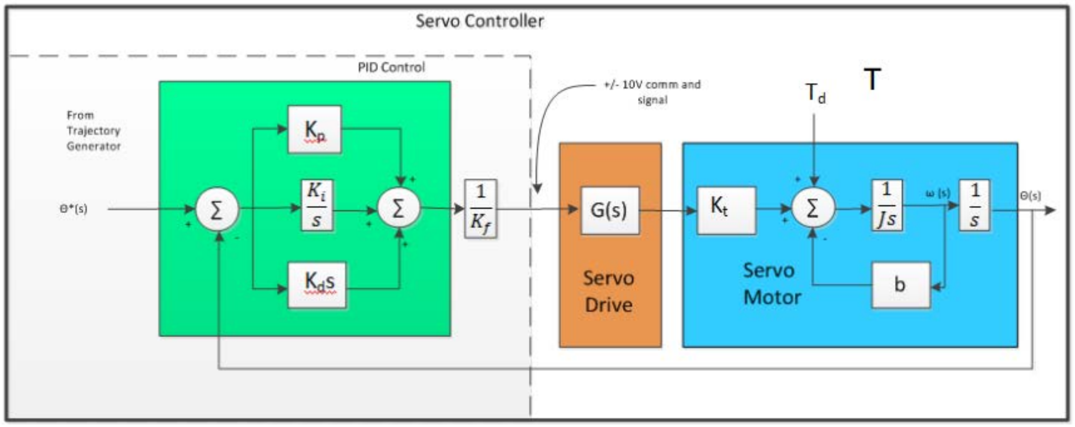


Figure 4.10 – Basic PID Servo Control Topology [7]

4.6.3 Arduino PID Library

The Arduino IDE also has a built in library for PID control. The user tells the PID what to measure (Input), where the user wants that measurement to be (Setpoint), and a variable to adjust that can make that happen (Output). The PID then adjusts the output trying to make the input equal the Setpoint. The three tuning parameters (K_p, K_i, and K_d) will be different for every user who utilizes this library. For that, there is a PID Autotune Library which helps determines the tuning parameters. The following tables are a few of the major functions that will be utilized in PID Library and PID Autotune Library.[8]

Function	PID()
Description	Creates a PID controller linked to the specified Input, Output, and Setpoint. The PID algorithm is in parallel form.
Parameters	**Input:** The variable we're trying to control (double) **Output:** The variable that will be adjusted by the pid (double) **Setpoint:** The value we want to Input to maintain (double) **Kp, Ki, Kd:** Tuning Parameters. These affect how the pid will chage the output. (double>=0) **Direction:** Either DIRECT or REVERSE. Determines which direction the output will move when faced with a given error. DIRECT is most common.
Syntax	PID(&Input, &Output, &Setpoint, Kp, Ki, Kd, Direction)

Table 4.15 – Arduino PID Library PID

Function	Compute()
Description	Contains the pid algorithm. It should be called once every loop(). Most of the time it will return without doing anything. At a frequency specified by SetSampleTime it will calculate a new Output.
Parameters	None
Syntax	Compute()
Returns	True: when the output is computed False: when nothing has been done

Table 4.16 – Arduino PID Library Compute

Function	SetMode()
Description	Specifies whether the PID should be on (Auto) or off (Manual)
Parameters	mode: AUTOMATIC or MANUAL
Syntax	SetMode(mode)

Table 4.17 – Arduino PID Library SetMode

Function	SetOutputLimits()
Description	The PID controller is designed to vary its output within a given range. By default this range is 0-255: the arduino PWM range.
Parameters	min: Low end of the range. must be < max (double) max: High end of the range. must be > min (double)
Syntax	SetOutputLimits(min, max)

Table 4.18 – Arduino PID Library SetOutputLimits

Function	SetTunings()
Description	Dictate the dynamic behavior of the PID
Parameters	Kp: Determines how aggressively the PID reacts to the current amount of error Ki: How aggressively the PID reacts to error over time Kd: How aggressively the PID reacts to the change in error
Syntax	SetTunings(Kp, Ki, Kd)

Table 4.19 – Arduino PID Library SetTunings

Function	Display Functions
Description	These functions query the PID internals to get current values. These are useful for display purposes.
Parameters	None

Syntax	GetKp()
	GetKi()
	GetKd()
	GetMode()
	GetDirection()

Returns	The corresponding internal value

Table 4.20 – Arduino PID Library Display Functions

Function	PID_ATune()
Description	Creates an Autotuner designed to create PID tuning parameters for the PID Library
Parameters	Input: The same variable that is connected to the PID Library as Input Output: The same variable thin the PID Library as Output
Syntax	PID(&Input, &Output)

Table 4.21 – Arduino PID Autotune Library PID_ATune()

Function	Runtime()
Description	Contains the Autotune algorithm. It should be called once every loop() during the autotune procedure.
Parameters	None
Syntax	Runtime()

Table 4.22 – Arduino PID Autotune Library Runtime()

4.7 Image Capture

The very first step in this motion tracking system is to capture the images that are in the field of view of the turret system. This information can then be processed by the image tracking software and used to send control signals to the servo motors as well as targeting data to the user interface so that a user can see what the tracking software can see. This is one of the first decisions that were required out of this project as well as one of the more daunting tasks to accomplish.

4.7.1 Camera

The various options for the camera on the video subsystem brought forward a lot of difficult decisions for the group. Wireless cameras mostly seemed dependent on the manufacturer's website in order to receive the video data wirelessly on a tablet. This did not seem like the best method of transmission due to the dependency on an internet connection. It seemed more logical to try and avoid using an established internet connection for device communication because the operating space of this system would most likely be outdoors. After researching multiple methods of sending video signal from a camera to a tablet separated by several meters of space, wireless radio transmitters (Xbee) were chosen as the best possible option. The camera to be used will be a simple Universal Serial Bus (USB) connected webcam like the Logitech C615 that will be linked directly to the main printed circuit board. The video images can then be received through a secure one way wireless frequency on the tablet and the tracking algorithms can begin.

4.7.2 Camera Schematic

It has been determined that the web camera is an adequate option therefore the camera will stream wirelessly utilizing a Wi-Fi X –BEE to transmit the signal to the tablet PC. This will be implemented by simply connecting a Wi-Fi XBEE to the cameras USB. A 5 volt secondary rechargeable battery will be used to power on both the camera and the XBEE Wi-Fi interface. A Micro USB plug will also be attached for a direct connect to the Laptop. The Camera schematic transmits to the windows tablet in which holds a receiver that has the exact same layout as Figure 4.11. The Receiver circuit though is powered by the internal battery of the Windows tablet PC rather than an external voltage source. The signal would be picked up at the tablet, and since the driver of the camera would be installed, the input to the tablet would appear as a wired camera input. It is critical that the voltage provided to the XBEE receivers and transmitters to be correct as any slight discrepancy would lower then bandwidth of the transmission.

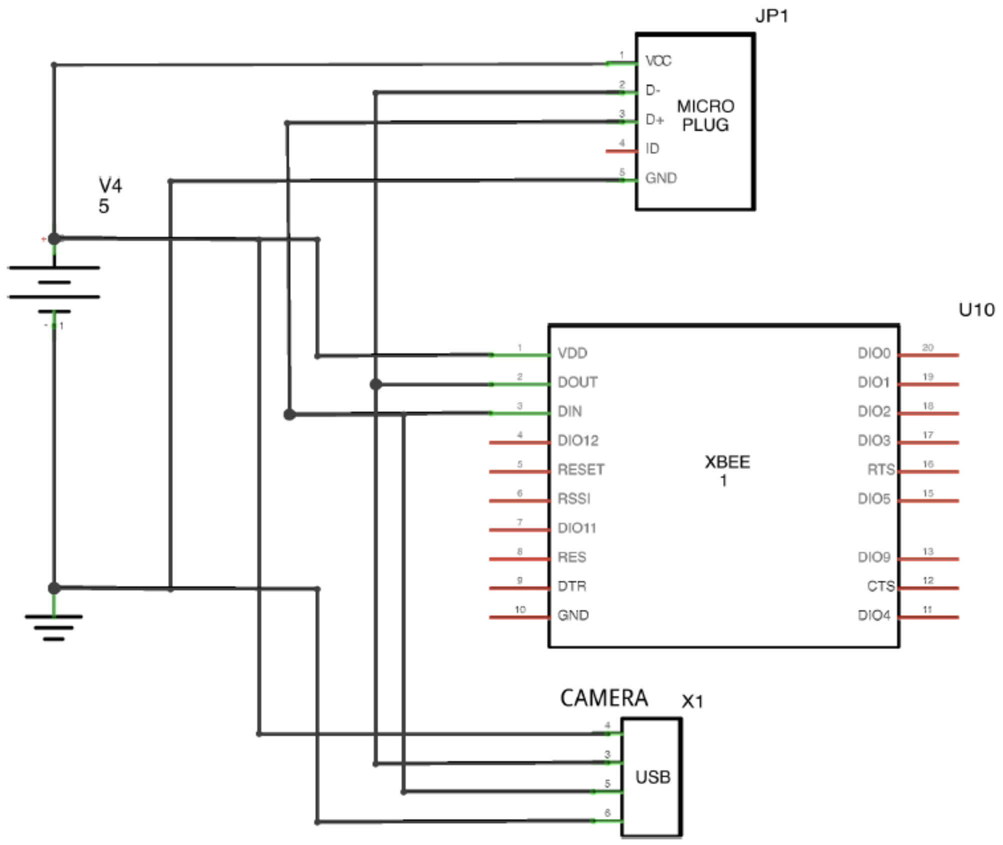


Figure 4.11 – Wireless Camera Schematic

4.8 Tracking System Design

The decision between the multiple tracking methods came down to using open computer vision software and libraries or the Processing programming language and its open source libraries. The Processing programming language won out due to its versatility, abundant online support and the fact that the source code is written in Java and not C++. The list of useful libraries that could be used for this project included many options that would aid in adding more complex abilities to this project. Among the multiple libraries that will be used for this project are the JMyron library, the BlobDetection library, and the serial communications library Processing.Serial. When the user selects to run the Processing IDE as well as the associated program files the program must borrow heavily from the pre-defined libraries in order to function. In Figure 4.12 the sequence of events that takes place after the user initiates the program is illustrated.

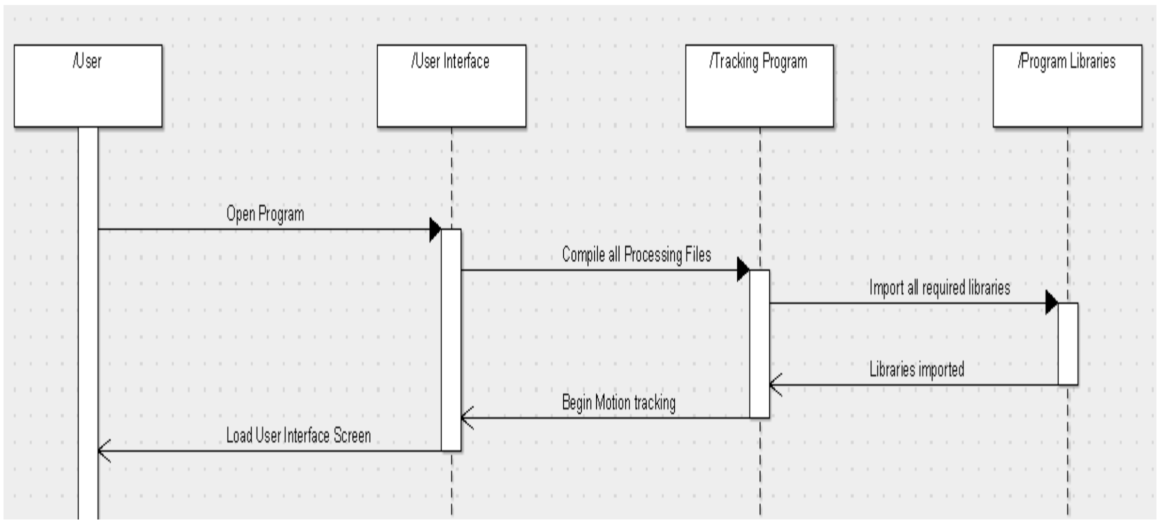


Figure 4.12 – Software Sequence of Events

The Processing based code will have multiple functions that it must complete at start up each time that it is run on the user's tablet. The main function will call upon multiple classes that each completes a separate task. The first class that will be utilized will be the user interface class that will load as soon as the user runs the program. Following this would be a class designated to making the connections with the microcontroller and designating instructions to pin locations. The JMyron library will be called to get camera input working and the blob detection library will be called to begin the motion tracking portion of the code. Each of these functions and classes will be use the information coming out of the large amount of required variables. Most variables will be partially populated by data received from the video system while the rest are pre-defined.

4.8.1 Class Functionality

The main class has multiple tasks mostly involving the calling of the other classes and loading them with information that is gathered from the video input. In the main class the camera's field of view window size is set, the libraries needed to complete the tracking algorithms are imported, and it is where the bulk of the tracking instructions needed for the motion tracking are located. Though the main class is where the motion tracking algorithm will be doing most of its work, there will be a separate class needed for features like target path prediction.

Since this path prediction is a special ability of the turret system, a separate class will be needed to implement it. This path prediction class will be called by the main function after the blob detection library has been imported and called. In order for it to work the X and Y coordinates must be compared in consecutive images. By comparing directional differences in the old position to the new position the program can predict that the next location will be in the direction that the object was traveling in the last two frames. This will allow the turret system to actively lead its target thus increasing accuracy by compensating for the time between firing and making contact with the target. The function can also use the amount of pixels that have changed between frames to determine how much acceleration the object has in order to send reactionary signals to the PCB. A brief layout of the path prediction class is listed below.

Function:	pathPredict()
Description:	Used to create target leading for accuracy
Parameters:	**dotP**: Variable used to set the amount of pixels that the target will be lead to compensate for travel time. **oldPossibleX**: Array of X coordinates used to predict future X value **oldPossibleY:** Array of Y coordinates used to predict future Y value **accX:** Array of X coordinates used in a comparator to determine a basic A=Velocity/Time using pixels and frame rates in the X axis. **accY:** Array of Y coordinates used for acceleration calculation in the Y axis **travelX:** Array of variables used to store the difference in X coordinates between consecutive image frames **travelY:** Array of variables used to store the difference in Y coordinates between consecutive image frames **prevTargetX:** Variable used to store the X coordinate of the last known location of the target **prevTargetY:** Variable used to store the Y coordinate of the last known location of the target

Table 4.23 – Path Prediction Function

In order for these calculations and tracking systems to be of any use, the tracking code must be in communication with the microcontroller. This is done through yet another short yet very important class that must be called by the main function as well. This class called controlConnect will be used to send the tracking information to the microprocessor which will then send its resulting commands to the servo motors so that the turret may begin to come alive. The following information describes how the controlConnect class works.

Function:	controlConnect()
Description:	Connects tracking software to the microprocessor
Parameters:	**portNum:** Used to store the number of available ports when the list.length function is called **connectFlag:** Used as a flag that notifies the program whether a microcontroller has been connected or not, used in conjunction with runNoConnect to allow video testing with the microcontroller not connected **runNoConnect:** Used in an if statement to allow the testing of the tracking software without the use of a connected microcontroller, sets **controlPort:** Stores current port number used for iteration through all ports **portChar:** Used to determine what values are currently being sent to a particular port **serialPortUsed:** Used to store pin location currently being used **miliStart:** Used to store a millisecond start time that creates a delay in the function

Table 4.24 – Control Connect Function

There are a few smaller classes that must be created including a class that stores sound files and that can be called upon when the user selects warning alarms. There must also be a class that allows settings to be saved so that if system failures occur they can be handled quickly and with little effect on the user. Once all of the code is combined it creates the necessary tracking ability needed for this project to work. As the code is more finely tuned over time, separate classes can be created to implement more advanced options like color tracking and facial recognition.

4.9 Wireless Communication

The basic wireless communication design of the project will have a linear scheme as illustrated in Figure 4.13. The video camera will only talk to the tablet for the purpose of a live video stream and activating the tracking software. The tablet will only send signals to the microcontroller for both automatic and manual mode. This will occur after the object detection and tracking algorithms have done their calculations. For both automatic and manual modes, the signal sent to the microcontroller is for movement of the servos. Two different types of XBee radio modules were chosen, XBee Wi-Fi for camera to UI, and XBee 802.15.4 for UI to microcontroller.

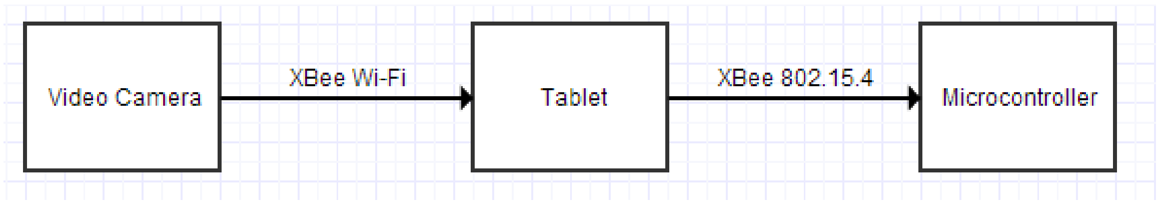


Figure 4.13 – Hardware/Software Communication

4.9.1 Camera to Tablet

As discussed previously, the amount of information needed to transfer a live video stream to a portable tablet will be extremely large. After considering all of the methods researched, the XBee Wi-Fi module was decided upon. The transmitter will be connected to the video camera via a serial adapter for the Wi-Fi module. The receiver will be connected to the tablet also via a USB serial adapter. This adapter is called the XBee Explorer Dongle and is priced at under $25 on SparkFun. This moderately priced XBee Wi-Fi module comes with two parts, a transmitter and a receiver. This is the ideal type of connection for the portable tablet, having no wires and only taking up only a few square inches. By following the instruction manual and spec datasheet of the XBee Wi-Fi, the video feed will appear on the tablet's screen with relative ease.

4.9.2 Tablet to Processor

The interaction from the tablet to the board in order to control the servos was chosen to be the XBee 802.15.4. These modules have incorporated PCB antennas which allows for easy placement on the custom board. Another reason XBee was chosen was the sleep mode that occurs when the module is not in use. This allows for greater battery savings. With the receiver module connected to the PCB board, transmitter will be connected to an XBee Explorer Dongle. This allows for direct connection to the tablet through USB port. Figure 4.14 below taken from Christopher Swingley shows the simple pin set up for connecting the XBee to an Arduino which is similar to the custom one the group will design.[9]

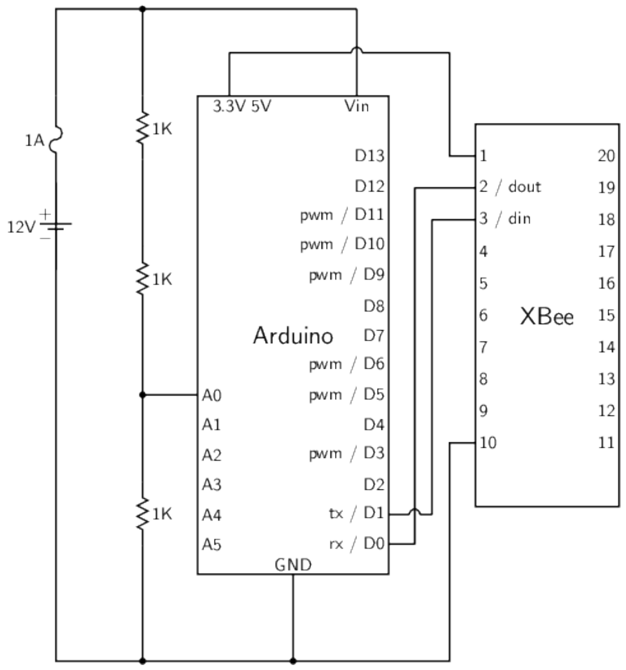


Figure 4.14 – XBee – Arduino Connection

4.10 Electrical Hardware

After much consideration it was finally determined that the Atmel Mega 328 processor will be the most ideal choice for the STATS project. The ATmega328 provided an adequate amount of resources in terms of I/O pins, clock speed, along with enough memory. Although the memory can be expanded, the provided memory will be sufficient. In addition the 5 different power saving modes will be utilized for the project. As stated STATS will operate in both manual, and automatic mode. With that said it is aware that the Automatic will utilize the microcontroller mode much more often, therefore the low power mode will be utilized in manual mode. The power mode will be set to active during the automatic mode to maximize the performance.

The table below realizes the device specifications. This will be used to limit the constraints of the project. The specifications in the table were derived from the mega 328 specification sheet found on Atmel's website. Memory will not be an issue as most of the heavy processing will be taken care of on the PC computer. The Arduino IDE boot loader will take up .5 KB memory on the chip. This will leave a sufficient amount of unused memory. Three PWM will be utilized to drive the servo motors. The operating voltage will be easily achieved using an on board power system The LM7805 Voltage regulator will be used to step down the voltage to the rated operating voltage.

Operating Voltage	5V
Input Voltage	7-12V
Digital I/O Pins	14 (6 PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz

Table 4.25 – ATmega328 Specifications

With all the pin layout now available with a description of each pin provided by Atmel, it is now simple to begin the design. As mentioned, Arduino IDE will be boot loaded to the chip. Before such the peripherals must be determined along with all the components. The chip will be programmed using ATP programmer from Sparkfun. This part will be acquired for $15.00. Due to Arduino being an open source, they were able to provide a great reference design to begin the design. As illustrated in the Figure 4.15 the Atmel Mega 328 pin out displayed along with a description to all the corresponding pins.

PORT	Description
Vcc	Digital Supply Voltage
GND	Ground
PORT B (PB7:PB0)	Port B is an 8-bit bi-directional I/O port with internal pull-up resistors.
PORT C (PC5:PC0)	Port C is a 7-bit bi-directional I/O port with internal pull-up resistors
PC6	If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin.
PORT D (PD7:0)	Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). T
AVcc	AVCC is the supply voltage pin for the A/D Converter, snd PC3:0. It should be externally connected to VCC, even if the ADC is not used.
AREF	AREF is the analog reference pin for the A/D Converter.

Table 4.26 – Atmel Mega 328 Pin Description

The microcontroller pin layout found in figure xxx indicates that the serial pins for the communication line are found on pins 1 and 2. These pins are responsible for any transmission between the mega 328 and other devices. Whenever a USB is plugged in, those 2 particular pins are used as the data lines. In the turret project, the goal is to transmit data to the controller wirelessly. It is important to insure the information is going back and forth in real time with no delays. The focus is to implement the wireless module onto the microcontroller.

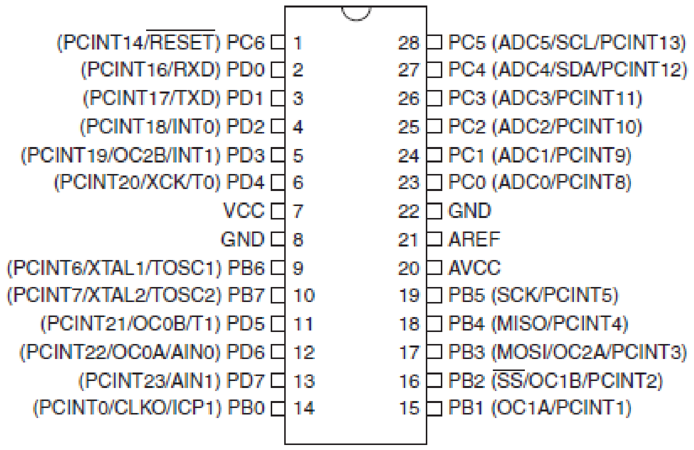


Figure 4.15 – ATmega Pin Out

The Atmel Mega 328 comes in different variants, one being a 28 pin while the other is a 32 pin. The main difference is the TQPF(32 pin) package includes a 4 bit analog to digital converter which may be utilized for analog signals. This less priced 28 pin will be used since the microcontroller will not utilize any analog to digital converter since all the analog inputs will be sent to the computer and digitally sent via serial transmission to the controller. With that said the 28 pin PDIP package will be used to prototype the project. Once the design is finalized, the final design including the PCB will utilize the smaller 28 pin MLF controller. This will help decrease board size, and also will allow better routing on the PCB since pins can be achieved from all sides equally.

Ultimately constructing the hardware for the turret will be split into sub stages to insure proper operation along with an opportunity to insure and debug and problems as they come up. With that said the microcontroller, along with all the hardware will be built on up from nil. When building the Mega 328 several components are absolutely required for optimal operation. The 328 datasheet did an excellent job documenting the microcontroller along with any necessary things to look out for. Figure 4.16 was derived from the reference design given by Atmel.

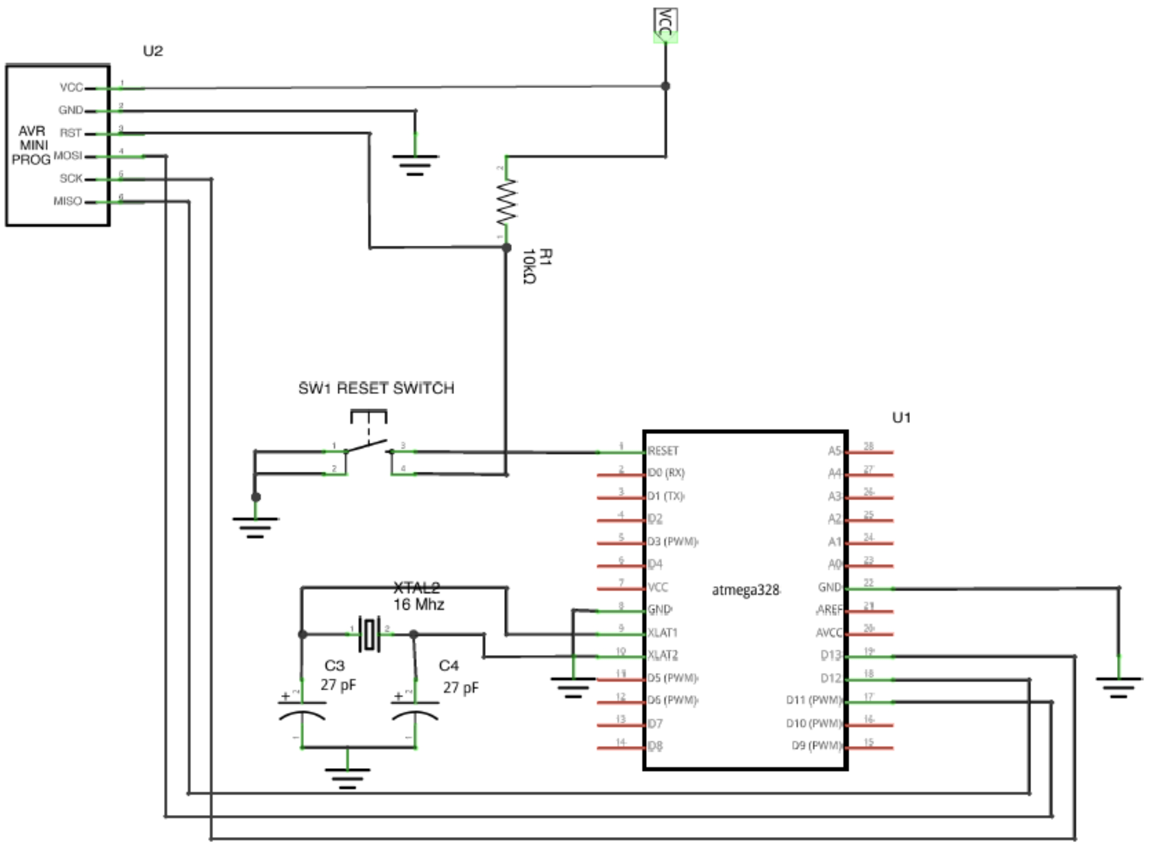


Figure 4.16 – Atmel Mega328P Reference Design

In the schematic found in Figure 4.16, it was realized that a 16 MHZ oscillator will be necessary to override the built in 8 MHZ oscillator since the Arduino boatload will require a 16 MHZ clock. In the begging it was a clear choice to use a 3 pin 16 MHZ ceramic resonator to implement the clock since it would be a cheaper implementation and did not require two capacitors which would also save time in the overall process. After much study it was found that it would be much more ideal to use a crystal oscillator. It is found that generally an oscillator has a much less percentage error compared to a ceramic resonator. Different sources noted that generally it would be fine to use a resonator over a crystal in an application such of a blinking LED, but do not prefer this option when dealing with an external primary clock for the control of the system. This is mainly due to percentage error which on average is about 1% in a resonator compared to .01% in a crystal. This is not such a big deal in most applications as stated but in terms of the USART, it will defiantly have an impact especially since a high baud rate will be used along with a high bandwidth video signal. With that said a 16 MHZ crystal oscillator will be used. The capacitor will be place in parallel with the negative pins grounded. These capacitors are necessary as they will be used to shift any unwanted signals from the oscillator and filtering out noise.

A simple pull down switch will be used to trigger the reset switch. This is denoted as SW1 in figure 4.16. A 10 K resistor is used to minimize the current from the power source to insure proper functioning and not burn the chip. An AVR mini programmer will be used to load and boatload the chip with the proper IDE. Although there are other options of loading the program into the EEPRom, an AVR mini USB programmer will be used since the surface mount chip variant of the 328 will be used. An option of purchasing the AVR mini programmer and embedding into the design was considered but a more efficient method of surface making and surface mounting the AVR programmer was preferred and would ultimately cost less since it would be embedded onto the same PCB. The AVR programmer power would be supplied from the same VCC of the Mega 328 and would only operate if and when any uploading would have to be done. The following schematic in Figure 4.17 from SparkFun will be used to implement the AVR Mini programmer.

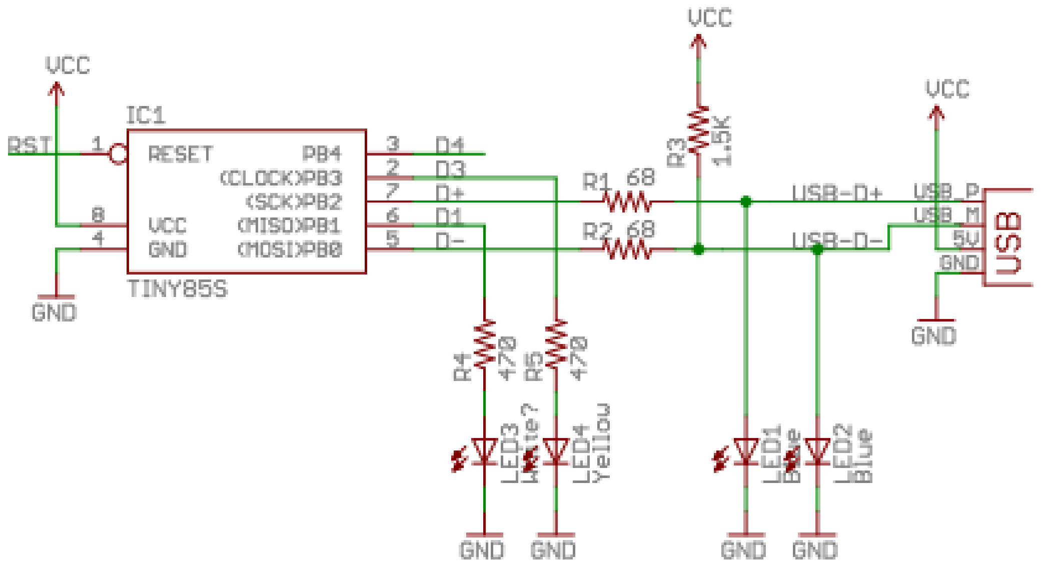


Figure 4.17 –AVR Mini Programmer

With reference to the Atmel mega 328 flowchart found on the Mega 328 datasheet, the figure in the datasheet illustrates the data path of the ATmega328 processor. All the ports are illustrated along with their leads into the controller. Here the leads for the motors can be found. It was mentioned earlier that the USART communication line will be essential for the motor control as there will be an Asynchronous transmission. With that said, port D will be used to control the motor. Although this is an 8-bit input line, only bit [0,1,2] will be utilized for motor control.

4.11 Audio and Alarm Design

The schematic found in Figure 4.18 implements the wireless transmission sub system along with sound, and visual alert. Several RGB LED lights will be used to make a variety of color strobes depending on the condition. Several RGB will be used to implement the light sub system. To keep in mind only 4 RGB led lights are illustrated in the schematic, but more will be added as necessary. This will depend on how much space left on the PCB, since only a limited surface area will be offered by the manufacturer for the education pricing. The resistor values will be changed accordingly as depending on the amount of LEDs. Again the LEDs' will be used to insure proper current ratings are flowing into the LEDS as extra current will burn the solid state device. Three Bipolar junction transistors will be used to control the LEDs' led by their respective pins on the PCB as illustrated. Since the design already hardwires the 5 volt VCC to the LED, the BJT will ground the LEDs' source. This will insure that the respective light will only turn on if the respective PIN is set high. The LED used will be the RL5-RGB LED from LED world.

The audio output modeled as a piezo buzzer (j1) in the diagram will be wired to pin 8 on the Atmel Mega 328. The signal will be amplified to insure optimal sound performance. The solid state speaker will be from the AudioLarm II line up manufactured by Floyd bell.

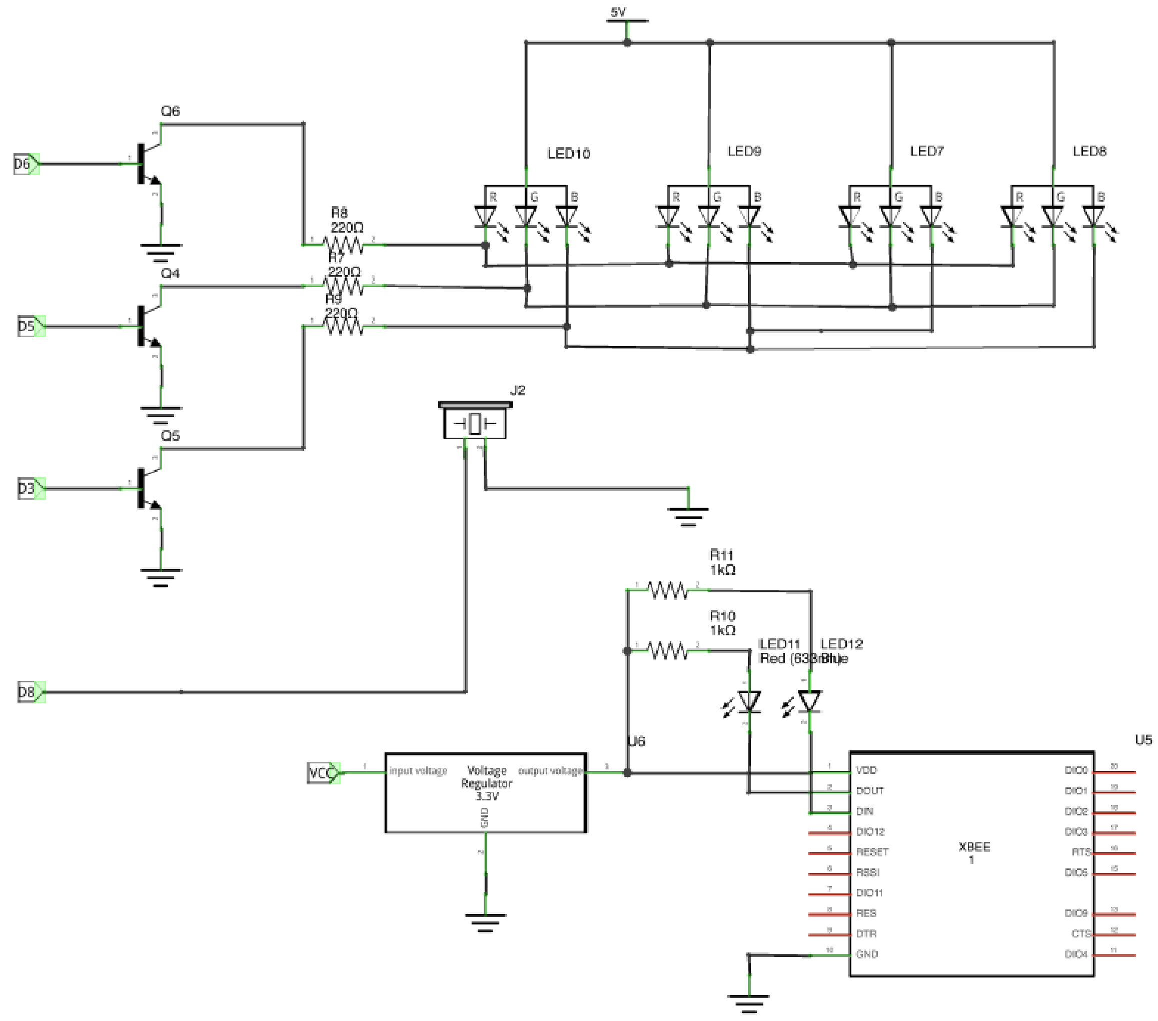


Figure 4.18 – Audio Alarm Schematic

The XBEE1 module will be a shield that will house the XBEE WIFI. The XBEE datasheet was used to create the schematic. The XBEE will powered using the secondary battery connected to the voltage regulator. The following XBEE WIFI will be used as a transmitter for the camera output. The 2 most important signal pins used are DOUT, and DIN which will take in and output the camera stream to the tablet as a WIFI RF signal. To insure transmission, 2 LEDs are place on the same node, and if the device is working properly, they should light up with the respective colors. With that said, if there is no light then there is no transmission. Several option were considered for the shield. Figure 4.18 illustrates how the XBee will be used to transmit the video signal. The camera feed will be sent to the XBEE, transmitted to the receiver, fed in as a USB. The power, a regulated 3.3V signal to both the receiver and transmitter will be supplied by the power system. A perforated board will be used to house the XBEE along with all necessary routings. As stated the XBEE explorer will be used to pair the XBEE protocols together.

4.12 Laser Device

The system requirement to have a laser device was established early on in the project design. Initially it was thought that it would be implemented in some kind of range detection algorithm. With further research into the abilities of common airsoft rifles, the addition of range calculation to this project does not currently seem like a necessary step. The distances that this turret will be effective in will be far less than the overall effective range of the airsoft gun. It was still decided that a laser should be implemented on the turret regardless, and would be used mainly for target testing. With the use of a weapon mounted laser, the accuracy and reliability of the tracking system can be tested without the need for live fire. This enables indoor calibration to be done to the system while the bugs are worked out of the code.

The laser that was selected by the group is the BSA weapon mounted laser sight. It is a cost effective 650 nM red laser that was priced just less than 20.00 USD online. The laser is adjustable so that the user can match laser alignment with the point of aim of the turret. It has the ability to mount directly to the airsoft rifle through the use of a standard picatinny rail system. Another advantage to this laser is that it will not require power maintenance because it runs on 2 AG13 batteries and like most low power lasers has a long battery life of over 20 hours. A second advantage to this laser is that it came equipped with a wired pressure switch that can be used to engage it remotely. The pressure switch will be mounted between the trigger mechanism of the airsoft gun and the servo gear arm that engages it. This will prevent wasted battery life and will only engage the laser when the trigger is pulled via pulse signals from the tracking software. This accessory is cheap and has many uses in the testing and final stages of the project and can be later used if needed

4.13 Audio Alarm

An audible warning alarm will be added to the autonomous turret as it spots object. The alarm will be implemented once an object begins to move and will continue sounding until the object is cleared out of sight. Some design constraints to be considered for were audio levels. The audio level has to be optimal enough to be heard within the viewing frame of the camera. With many great reviews, and many recent senior design groups using the AudioLarm II from Floyd bell, it became the preferred choice as there was much support from the company. Audible levels on the AudioLarm II are variable depending on the voltage level. Figure 4.19 illustrates the audio decibel level as the audio is increases. Five volts allows for an 85 dB tone, while 95 dB can be achieved with 30 volts. These sounds level translate into a hair blow dryer, all the way up to city traffic. Note that a sound level over 95 dB can cause harm to a human ear. Since the AudioLarm II is a solid state device it ships out with a preloaded tone, therefore a tone must be picked out before the device ships out. A very typical fast beeping alarm that sounds identical to a burglary alarm found in business and banks seemed to be the most accurate choice for use. The AudioLarm II is a simple device in terms of pins. A 5 V DC signal will be used to trigger the device for an 85+/- dB audio level. Adjustments to the sound may have to be made later which would result in stepping up the voltage.

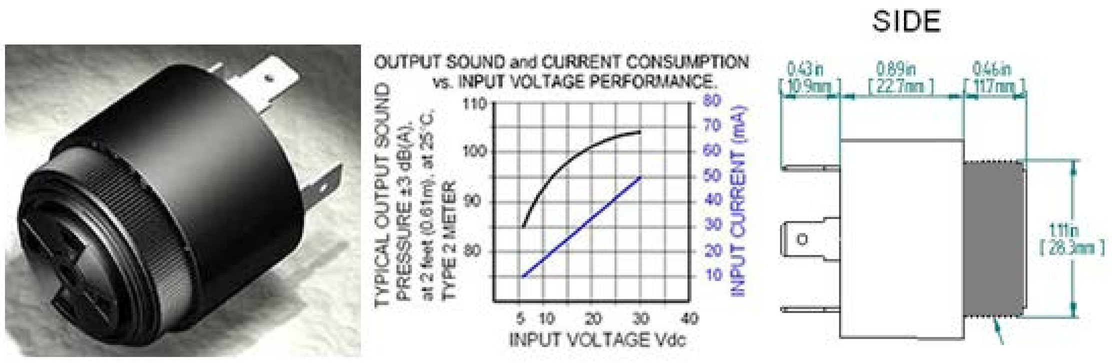


Figure 4.19 – AudioLarm II from Floyd Bell

4.14 LED Alarm

A sequence of 20 high intensity RGB LED lights will be connected to mimic emergency lights whenever the camera picks up a moving object. This will be the other deterrent to maximize defense. The RGB LED will be obtained online for whomever would be able to offer the best price for a large bulk. Figure 4.20 illustrates the general connection of the LED lights.

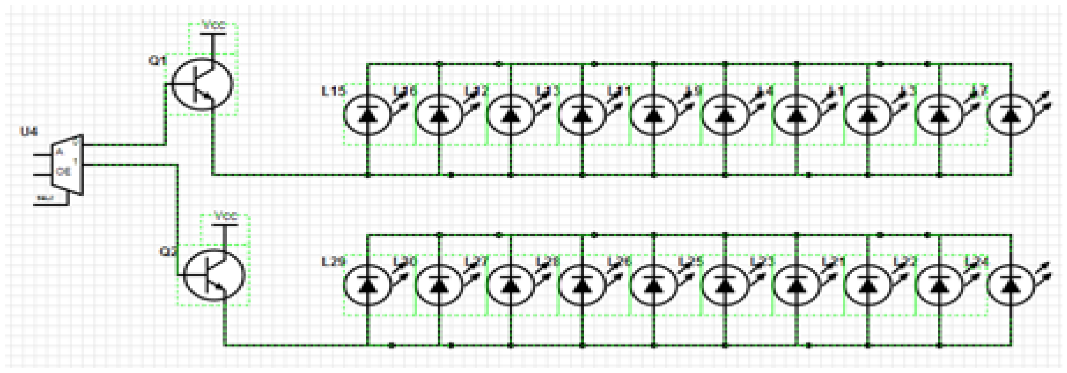


Figure 4.20 - LED Layout

A 1 input 2 output decoder will be used to control the LED lights. Whenever a moving object is sensed the control bit for the LED will modulate between 1 and 0 to this will implement the decoder to alternate between the lights and the BJT will amplify the voltage to allow for a desired current rating. This lights will keep on flashing until the object is cleared. Whenever there is no object then the LED light will return to standard cool white until another object is sensed. The LED operate at 3.2 Volts, but the voltage drop across the BJT must be taken account for. Four conductor RGB power wire strip will be purchased to neatly wire the LED lights together in a presentable manner

4.15 Power Supply Design

As mentioned the power design will supply the turret with all the power desired. A 12V SANYO 1100mAH, with a maximum discharge current of 1.35 Amps battery pack will be used to lead the circuitry. This main power house will be used to control the motor controller, which consist of the Atmel mega 328, and 3 servo motors. The power ratings of the devices that will be powered by the primary battery.

The Atmel mega 328 requires a 5 volt DC source. This will be done by the LM7805 voltage regulator. A voltage divider will be used to divide the voltage from the battery to the optimal supply voltage needed by 7805 for the 5 volt output. The output of the 7805 will be connected directly to the microcontroller. This can be reference by Figure 4.22. As for the motors the remaining 7 volts from the battery pack will be voltage divided again to each motor. With that said each motor will be supplied with its own separate voltage rating. Capacitors will be connected in parallel to reduce the noise of the battery.

The MAX 1704xx will be led by the secondary battery. This battery will account for minimal current drawing devices, which includes LED's, and the battery fuel cell gauge. It is important to note that if the battery for the fuel cell gauge is dead, the turret will still operate, but will not be able to keep track of the battery life. This design constraint was meant for as it would be ineffective for the device to fail if only one battery pack was running. This will optimize the operation as the device can work utilizing only one battery. LEDS will be connected to pins D2 and D4 which will be triggered with code that would be flagged using a Boolean that is triggered by a low voltage signal from the MAX1704XX IC. A blue LED would indicate the battery has optimal charge for the device to continue running, while a red LED would be triggered if the voltage drops under the minimum operating voltage. The MAX1704XX also contains an interrupt bit which in the case of this project will be grounded. If the interrupt bit on the MAX1704XX is triggered, then the code running on the Atmel Mega 328 will jump this is undesirable as the battery would be immediately larger when the voltage drops below operating voltage.

The turret will utilize a recharging circuitry that will use the constant current/constant voltage (CC/CV) method. This will be implemented by using the LT1510 -Constant-Voltage/Constant-Current Battery Charger from linear technology. The circuit will be implemented on the PCB. The reference design provided by linear technology will be used to build the device. Linear technology provides 2 designs to consider. Each schematic utilizes a different application. The once used in the turret will be the application in which supports up to an input of 20 volts. The reference design found in figure 4.21 was found in the LT510-5 datasheet. The device will be purchased from linear technology for $5.13.

Another prominent well-known lithium ion battery charger is the LT1510 from linear technology. The LT1510 is very cost efficient for hobby projects. In addition implementation is simple since there is only 4 pins. The LT1510 compared to the MAX1555 would be a more ideal choice due to the variation of voltage levels that could be used. The LT1510 supports up to an input of 20 volts compared to the MAX1555 of 8 volts. This may be undesirable for smaller size batteries but since the turret batteries are larger and high capacity, it would be preferred to have a higher input voltage. This would speed up the charge time. If the current divider across R1 and R2 is set up to where the charging current is 450mA, then it would take approximately 1.8 hours to charge the battery. This is found using the following equation.

$$CHARGETIME[h] = CHARGECAPACITY[mAh]/CHARGECURRENT[mA]]$$

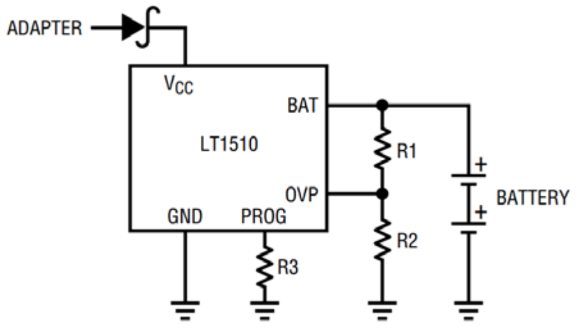


Figure 4.21 – LT1510 Battery Recharge Circuit

The Resistor value across R1 and R2 can be changed to reflect the desired charge current. This can be very useful since every battery's specification is different. In this case various battery across all sizes can be charged using the same circuit. This compared to the

MAX1555 is a big advantage since the MAX1555 only charges using a pre-set 280 mA current that cannot be changed. Using equation xxx, the battery charging time becomes roughly 3.7 hours, almost double of what is given by the LT1510.

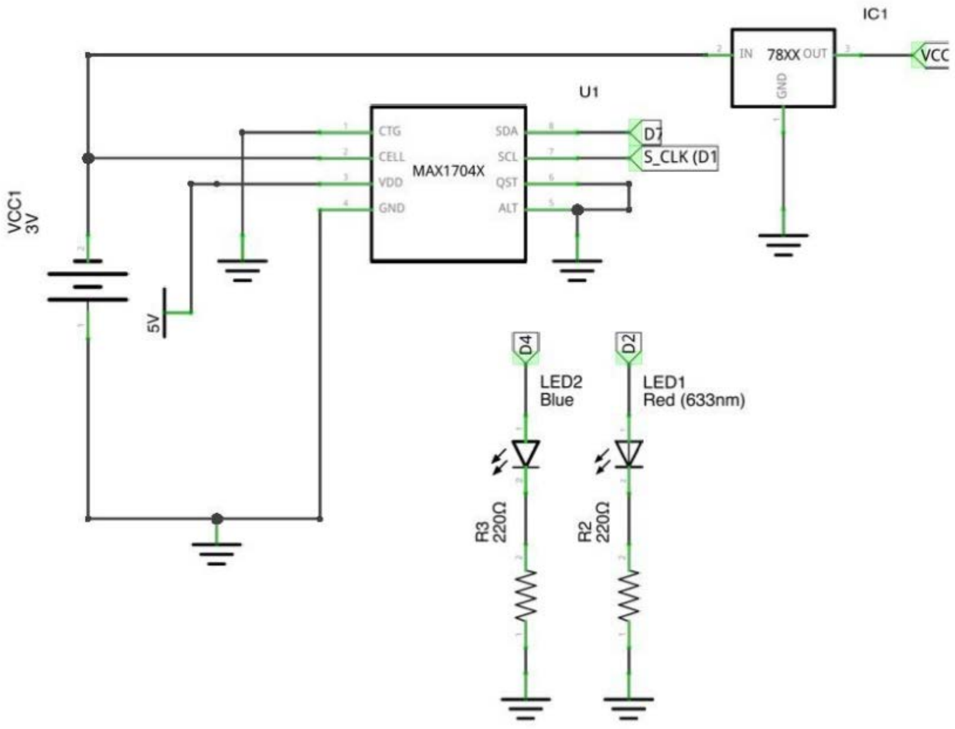


Figure 4.22 – Fuel Cell Gauge

4.16 Servo Motor Connection

The connection of the servo motor pins can be found in figure 4.23. Illustrated below is the ideal connection for all the PCB compnents in a grand scheamtic. The servo motors will have a 3 pin digital layout triggered by pulse width modulation. The positve terminals of each motor will be connected in parallel to the battery with a respective resistor. The value of the resisotor will be determined such that the each motor draws the ideal current need to operate for maximum performance. Motors (j5, j4) represent the HS-5685MH ultra torque motors that require a 4.8-7.4 Volt operating voltage. These are considered the larger motors in the turret, while motor (j3) hs-5055mg the small varient will be used to trigger the firearm. The hs-5055mg opeates at 4.8-6.0 Volt. All three motors require a 3-5 volt peak to peak square wave voltage. This signal will be supplied by the Atmel Mega 328 processor. The motors will be connected to the mega 328 PWM ports.

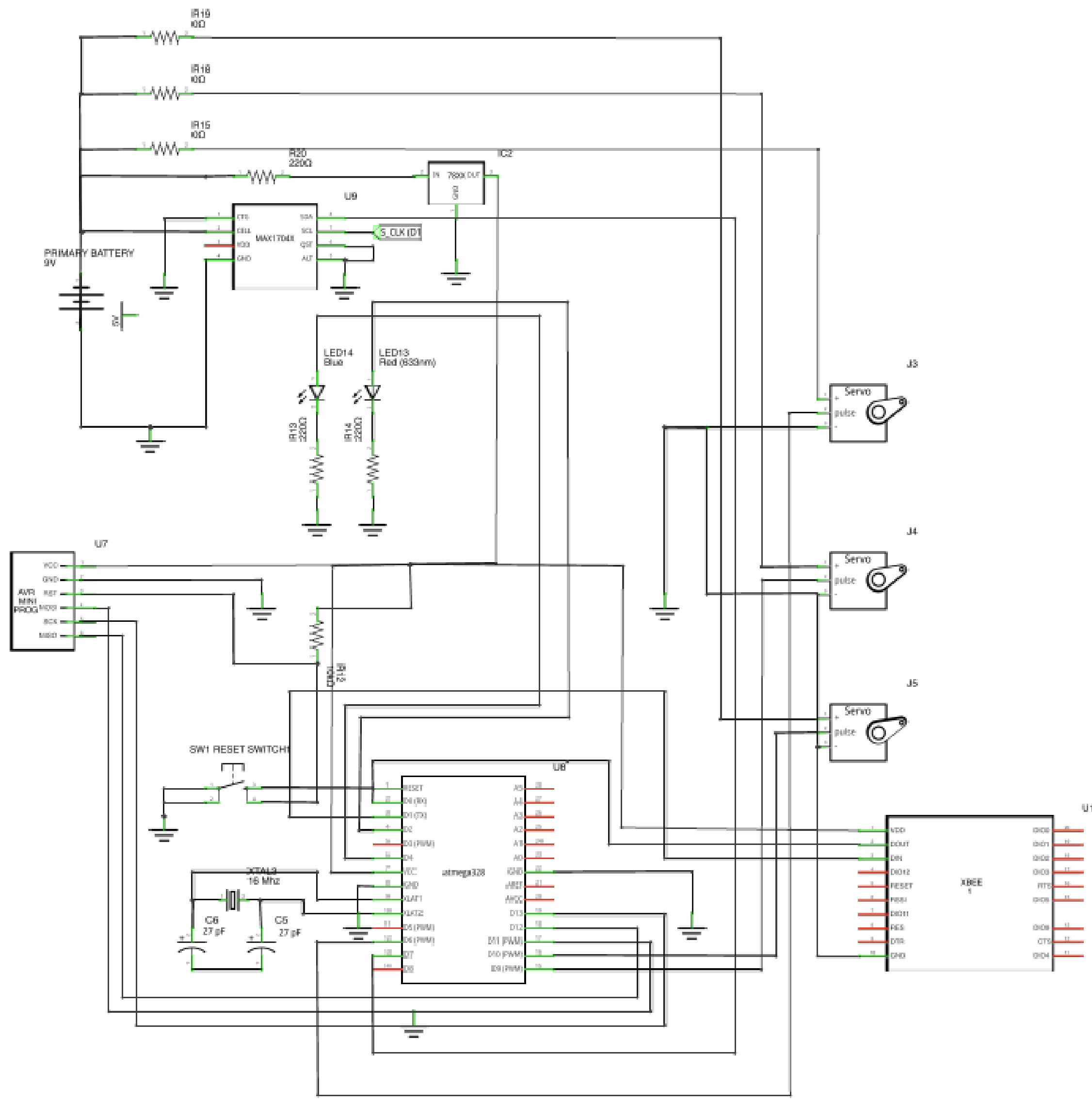


Figure 4.23 – Board Schematic

In the schematic above, it can be realized that the servos pulse is triggered by the mega 328 processor. The only way possible way to turn on the servo is by applying a pulse width modulated output which triggers the digital input. The mechanism works similar to a relay but at a much quicker implementation. In a relay an electromagnet operate a switch. The speed of the implementation is heavily based on how strong the voltage across the relay is. With that said a wireless signal will be received by the XBEE decoded by the mega 328, and will trigger the servos.

Section 5: Prototype Construction

Prototyping will be split amongst all the members and will take place throughout the course of the project. Since much of the project will be fresh concepts, it was best to attempt to have a working model before an idea was proposed. With that said, an initial model to control servos using a Windows computer and other mobile devices was used. After much research a working model of servo control was built using a mobile device and a microcontroller. Another example as such would be using OpenCV to track objects, and to return the location in coordinates. All in all this method would support adding the feature to the final design as long as partial working model was made. Prototyping of the final model will take place in Senior Design II during the summer semester. This will transpire once all major design details are considered and project materials are received.

5.1 Hardware Assembly

The hardware aspect of the turret project will entail smaller components coming together to form a whole. The base of the system will be hand built out of wood. This must be able to maintain the weight of everything and have slots for proper placement of all the necessary elements. The PCB housing will be simply a small encasing holding the board and wires. All these physical pieces will be combined together in the end to form the final project.

5.1.1 Main Housing

The main housing was custom build using piece parts from local hardware stores. The main frame assembly was put together using bolts located in the corners of the frame (marked in red). The surface panel of the housing has markings for pre-drilled holes for the mounting of the servo armature, camera and PCB housing. On the rear panel, markings were put in place for attaching the modular attachment for the tablet. Figure 5.1 below diagram depicts a blowout representation of the side panels and all markings. The tablet panel is excluded from this model to clearly reveal the innards of the design. Mounting locations for PCB, camera and power supply are marked in blue.

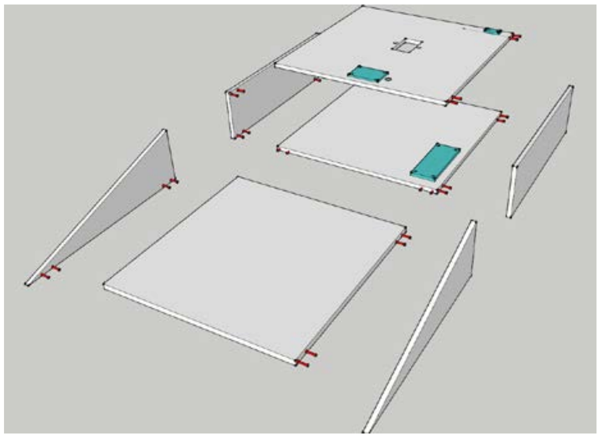


Figure 5.1 – Main Frame Blowout

The bottom panel contains the attachments for the four retractable legs. The legs were attached via ball joint shown in red in Figure 5.2 for maximum pivot angles to conform to a surface which may not be completely flat. Rubber pad on the bottom of each leg provide extra traction.

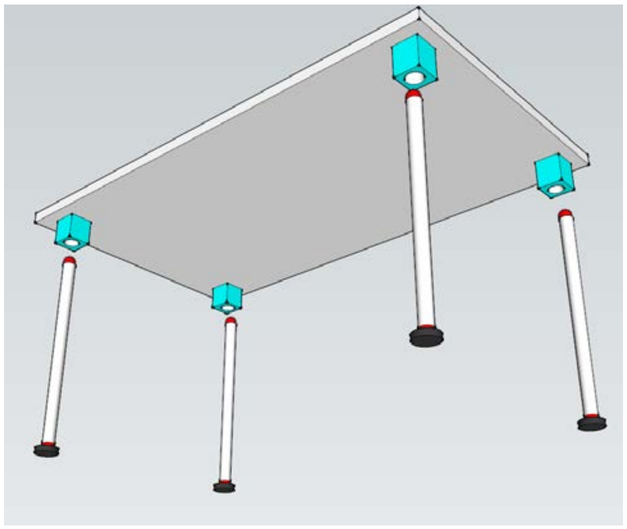


Figure 5.2 – Main Leg Frame Support

5.1.2 PCB Housing

The PCB housing is located on the top surface panel on the main housing and is bolted in securely. The top of the PCB housing has a level and latch marked in blue in Figure 5.3 to open the cage for access to the PCB. Holes are drilled into the sides of the cage for wiring purposes.

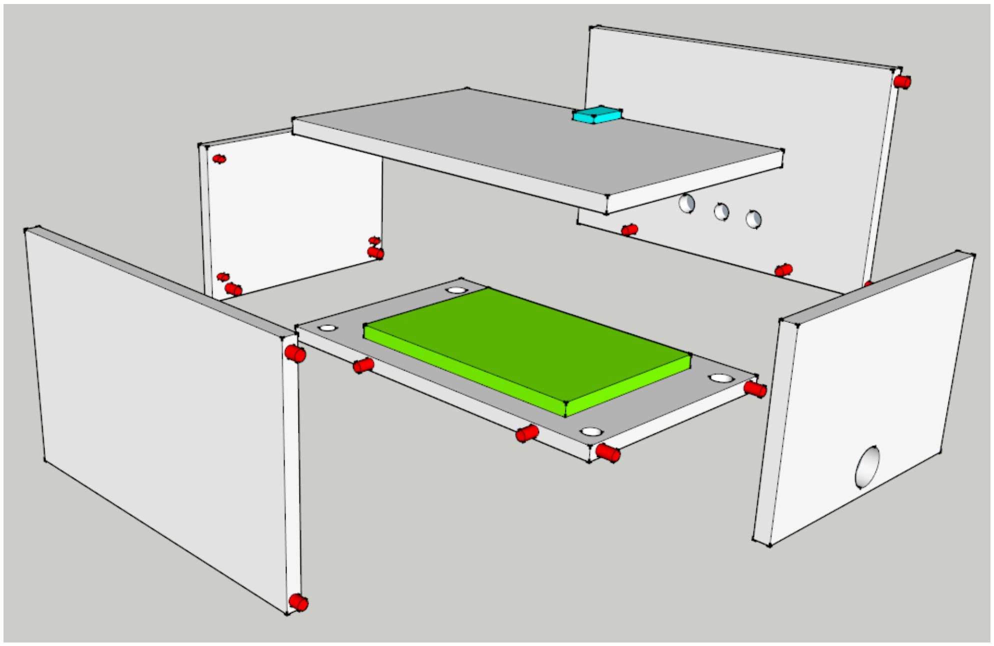


Figure 5.3 – PCB Housing Blowout

5.2 Weapon Platform Mount

The weapon platform mount design will be custom built using a modified clamp. The clamp will contain two large rubber pads roughly 28 in^2 (7in x 4in) in area attached to similar size metal plates using adhesive. The metal plates will then be attached directly to the large disk gear on the y-axis positioning servo via screws. Using a longer screw size (approximately 5 inches), the other metal plate will be connected. Adjusting the compression of the clamp will require the user to tighten or loosen the two screws holding the clamps together. This method will allow the clamp to conform to either a weapon platform or smaller laser light. The main challenge with the weapon platform mount is the addition of the trigger servo. For this function to work properly, the trigger servo must be mounted to the disk gear of the y-axis positioning servo, and not the clamp. For the purpose of prototype design, the trigger servo was designed to work with the weapon platform chosen for presentation. In order for the servo to be accepted by additional weapon platforms, slight positioning adjustment would have to be made to line up the armature with the trigger. The diagrams below shows the design details of both the clamp and trigger servo with respect to the y-axis positioning servo. Figure 5.4 depicts the clamp system utilizing a laser instead of an airsoft gun to show the versatility. Figure 5.4 also shows the mechanics of the trigger servo. As seen, the blue and red object represents a trigger and trigger guard. The green pin will be attached to the trigger servo armature and emit a force on the trigger.

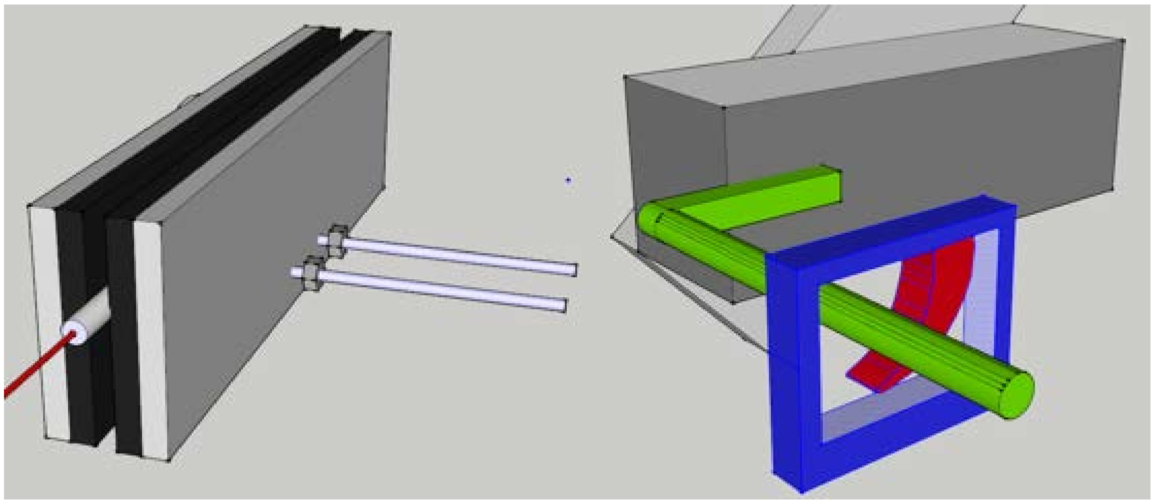


Figure 5.4 - Clamp and Trigger Servo

5.3 Servo Armature

The servo armature combines the use of three servos with metal rails for pivoting capabilities in horizontal and vertical direction. The bottom servo motor (x-axis) uses a large gear disk to mount the first vertical arm which extend up to the second servo motor.

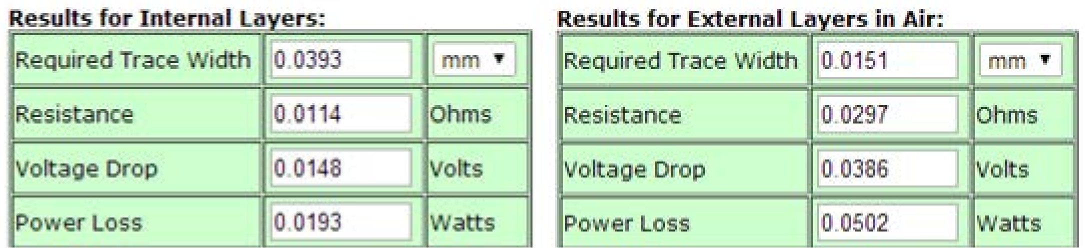
The y-axis servo motor is connected at the top of the metal rail and also includes a gear disk. The y-axis gear disk has mounting spots for both the trigger servo and the adjustable clamp.

5.4 PCB Software Design

Since printed circuit boards are part of the guidelines in Senior Design, a 2 layer PCB board will be implemented using Eagle by Cadsoft. It was a bit difficult choosing which option would be used to design since the turret would be the first project requiring a PCB. After reading much reviews and forums online from multiple sources including Cadence, Cadsoft, and Sparkfun it had been apparent that Eagle was the best choice over other options such as 4PCB, DipTrace, and ExpressPCB. It was also fortunate that there was a How to Use Eagle seminar during the mid-Spring semester. It was beneficial enough to begin having an idea of PCB design as this was never taught through all of the course work required. PCB design will be put off to the end as it will not affect the prototyping stage in any way. In addition all design details along with pin routing must be realized as this cannot be changed on the PCB once finalized.

5.4.1 PCB design

There will only be one major PCB for the design, with four major control units included on the board. Figure 5.5 illustrates the PCB block layout. The main control unit of the board will be the Atmel Mega 328, followed by the wireless mounts for the XBEE modules. There will be a total of 2 XBEE modules on the board in which each will use a different transmission protocol. The audio/light warning alert system, followed by servo drivers will be the smaller system in terms of complexity, but will use the most amount of current. It is estimated that the servo motors will draw about a maximum of 1 Amp if they are all working simultaneously. Considering the estimated 1 mm thickness along with estimated an estimated 1.3 Amp maximum current. Using these estimates, a web app was used to determine the trace width. The following results are tabulated below.



Results for Internal Layers:

Required Trace Width	0.0393	mm ▾
Resistance	0.0114	Ohms
Voltage Drop	0.0148	Volts
Power Loss	0.0193	Watts

Results for External Layers in Air:

Required Trace Width	0.0151	mm ▾
Resistance	0.0297	Ohms
Voltage Drop	0.0386	Volts
Power Loss	0.0502	Watts

Figure 5.5 – PCB Trace Width

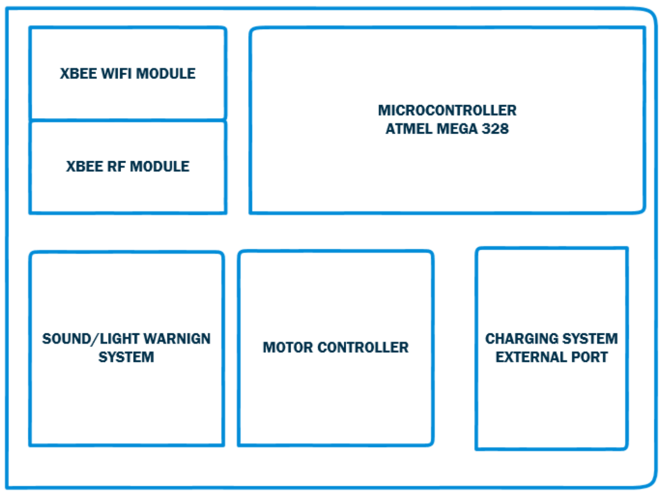

XBEE WIFI MODULE
XBEE RF MODULE
MICROCONTROLLER ATMEL MEGA 328
SOUND/LIGHT WARNIGN SYSTEM
MOTOR CONTROLLER
CHARGING SYSTEM EXTERNAL PORT

Figure 5.6 – PCB Block Layout

5.4.2 PCB Fabrication

It has been determined that the 2 layer printed circuit board will be adequate and will be ordered from 4PCB.com. 4PCB offers very competitive student pricing programs. With that though arise size constraints. Eagle will be used to design the PCB layout and the file will be sent to 4PCB for manufacturing. Eagle software is available at the UCF Senior Design lab, along with the smart lab therefore the PCB will be designed at UCF. Since an educational copy of eagle will be used, it limits the PCB board size to be only 3 inches by 3 inches. That said the board will be carefully designed to contain all the required components in an organized manner. Through hole mounting will be used since it requires less experience in soldering. If it is found though that through hole mounting will not fit adequately, then surface mounting will be considered. Compared to other PCB manufacturers, 4 PCB allows for having no quantity limit on boards at educational pricing. An extra board will be ordered for any faults that occur due to bad soldering.

Upon arrival of all components, assembly will take place in the Senior Design Lab. A Soldering Iron, solder, wick, paste and flux will all be obtained from the University. Capacitors, and resistors will all be obtained from the Electronics Lab on campus, or ordered online since some values are not located at the University. A reflow oven will be utilized for any surface mount technology. All components will be tested before mounting to decrease the percentage error. The motor pins will use special connectors that will be purchased from the manufacturer to insure proper installation and maintenance. The PCB will be mounted to an enclosure using mounting brackets. This will insure the board is protected from any vibration or physical damage if that arises.

5.4.3 Processor

The STATS team members have decided to utilize and Arduino Uno to prototype the design. Some factors that were considered was being that the main processor to be utilized will be an Atmel Mega 328 boot loaded with an Arduino IDE, it would be wise to use an Arduino Uno. Features include an I/O USB terminal for communication between the microcontroller and computer. This is essential as it will be utilized for transmitting the coordinates of the object tracked. A range of both analog and digital ports are also included. The board sells on the manufacturers’ website for around $40.00, but since a group member already owns one, theirs will be used. An Ethernet shield or XBEE WI-FI will be required to create a WIFI transmission line.

Since Arduino is open source, they seem to document everything very well, along with providing examples. Spec sheets, circuit diagrams, images, along with reference code was all found on the website which will be useful in the prototyping stage. It was found that Arduino also makes other embedded development board such as he Arduino Mega, but due to overwhelming specs such as abundant RAM, ROM, and masses of ports that would have been abundant and more costly for stats. There were additional development boards manufactured by Atmel, and other recognized Electronic Manufacturers, but it was the same dilemma in abundance of features that would not be utilized and simply just grounded.

5.5 System Hardware Wiring

It is evident that most of the wiring discussed in other sections refers to internal PCB wiring that will be routed using eagle. Along with this though, there will be physical 24 AWG wiring that will route the motor connections and power connections. The fire arm mount will be above the housing and will pan and tilt. The PCB will be placed near the firearm, but most of the wiring will be taken care of within the turret housing. A 9 volt AC-DC power cord will be routed in from the rear of the box to the PCB housing for charge. This will be a standard plug in that is found in many home electronics. Three sets of 3 24 gauge wires (positive, negative, signal) will be run from the each motor to the PCB. This will supply each motor with its required signals to operate. These wires will all be routed from within the turret housing. All the wires will be cable ties and strapped to the ides to no create clutter. Heat shrink to be attached around all wires, and proper American wire color coding will be applied.

5.6 Hardware and Software Design Summary

5.6.1 Hardware design summary

In the self-targeting autonomous turret system the following hardware technology was used, shown in the table below:

Sub-System	Hardware Design
Wireless Control	Radio-Frequency Transmission
	Tablet Interface
Mechanical Execution	Servo Motors
	Motor Controller
Hardware Control	Microcontroller
	Power System
Enclosure	Turret Housing

Table 5.1 – Hardware Design Summary

5.6.1.1 Wireless Control

One of the standout features of the turret is its ability to fully interface wirelessly in both manual and automatic mode. Two sets of XBEEs were used to implement wireless radio frequency transmission each using a different protocol. Camera transmission will be transferred via XBEE Wi-Fi to the receiver mounted to the tablet USB XBEE shield. In terms of interface with the microcontroller, an additional XBEE Pro RF module will be used to communicate with the USART port on the Atmel Mega 328. The interface will be able work up to 30 feet away of the device. A Windows based Tablet will execute the computer vision algorithm and will stream back the information in real time. A manual and automatic mode will allow for a friendlier interface.

5.6.1.2 Mechanical Execution

Top class digital servo motors will be used to synchronously real time autonomously track objects with help from the processor. The servo motors manufactured by Hitec to be purchased from Servo city with a maximum speed of 0.17 sec/60 degrees. A motor controller will be constructed utilizing the Atmel Mega 328P controller. Three pulse width modulated outputs will be used to trigger the pulse line on the motor. The motor controller can handle all three motors at a single time. That said the controller can pan tilt and fire with no disambiguate.

5.6.1.3 Hardware Control

In terms of hardware control, a microcontroller, and advanced power system will utilize top notch technology to power the sub-system, along with serve as the CPU to run the software. An Atmel Mega 328 will take advantage of the Arduino IDE to implement computer vision, and motor control libraries. The power system will be able to charge, and monitor battery levels. CC/CV charging methods along with a fuel gauge battery sensor will be utilized to maximize performance of the power system.

5.6.1.4 Enclosure

An enclosure will be custom built to satisfy the size constraints. A combination of both Lexan and lumber will be used to handcraft the enclosure. The motors will rotate the firearm in the x-direction utilizing minimal friction metal gear disks. The enclosure will encase the tablet, PCB, power supply, camera, firearm, along with any other peripherals necessary to maximize operation.

5.6.2 Software Design Summary

Sub-System	Software Design
Computer Vision	Tracking
	Object Detection
Automation	Servo Movement
	Control System
Input	User Interface
	Microcontroller

Table 5.2 – Software Design Summary

5.6.2.1 Computer Vision

For the object detection to work, computer learning must be implemented so that the software can determine that a foreign object has entered into the detection range. This is done by using the blob detection library that is imported to the Processing code. Essentially this creates a digital wrapper that encases the pixels that have changed during the progression of image frames. The comparison of pixel colors between frames is what notifies the program that an object has appeared. This digital wrapper which appears as a blob on the screen is then singled out and the tracking method takes over.

As the tracking method begins to function it implements algorithms that determine the distance that the pixels have moved from frame to frame. By keeping an array of previous pixel locations the program can begin to estimate the targets heading and compensate for the time differential between firing a shot and hitting the target. This information is then passed to a separate class which calculates the physical location that the servo motors must be set to in relation to the field of view. With the combined efforts of the tracking algorithm and the processor control code the sentry gun in theory will be able to predict the path of a moving target and accurately place shots on the target as long as the target remains in the turrets field of view.

5.6.2.2 Automation

Servo movement will be accomplished by utilizing the open-source Arduino environment and prebuilt libraries. One of those prebuilt libraries is called Servo Library. This library allows up to 12 servos, each at various angles between 0 and 180 degrees. By applying the six functions of Servo Library into the Arduino IDE, the signal sent from the tablet to PCB will activate the servo movement.

Path prediction is function used in Processing language. It will be called by the main function after the blob detection library has been imported and called. In successive images taken by the camera, the X and Y coordinates of objects get evaluated. By comparing directional differences in the old position to the new position the program can predict that the next location will be in the direction that the object was traveling in the last two frames. This will allow the turret system to actively lead its target thus increasing accuracy by compensating for the time between firing and making contact with the target. Path prediction also uses the number of pixels that have changed between frames to determine the acceleration of the object. This in turn to sends appropriate signals to the PCB.

5.6.2.3 Input

Communication between the tablet and microcontroller requires a second Arduino library to be implemented called Arduino Serial Library. This library allows transmission between the microcontroller and a computer, or for this project a tablet, via USB port. This makes controlling the servos, siren, and any other apparatuses the group decides to add possible. Even though this project requires wireless transmission between tablet and board, this Serial library will be included. A number of the given functions and methods will still be necessary as the fundamentals of the communication are still there.

The tablet interface that the user will use during manual mode includes a live video feed on the camera's field of vision, a four-way directional, and a fire button. The video will cover a large majority of the screen's real estate. Located in an easy to access location while the user is holding the tablet will be the directional pad and fire button. The user will be able to move the gun and aim in any direction of their choosing. Pressing the fire button will begin shooting the weapon. Until the user releases the on-screen button the turret will continue shooting. Processing language will be the programming language of choice. Prewritten Java GUIs exist that will help with button creation and overall blueprint of the UI.

Section 6: Prototype Testing

This project has a number of pieces and subsystems that will be combined together in the end to form the complete design. Each section of this project must be tested individually before slowly combining them together. This will save time when determining specific problems that will arise as each new part gets added to the final build. First individual hardware and software components will be created and assembled. Following all separate members being constructed will begin piecing together these parts, still alienating the hardware and software portions from each other. Once that occurs then the hardware and software can begin their combing process. At the same time the microcontroller and servos are being tested, the wireless communication and image tracking are all being tested independently from each other. By executing the testing procedure with this method this allows each member of the group to be working on something at all times.

This project requires that ideas be implemented physically in order to make decisions on a final design. The amount of testing required for this project is immense due to the number of separate parts that must communicate. In order to effectively test the tracking system, much of it was done using open source code, breadboards and development boards that would not be allowed in the final design. Testing on a laptop instead of a tablet and using various types of cameras are just a couple of examples. Due to this much of the materials and prototypes must be changed in order to get design credit for this project, and with these changes there can be errors that will occur in both the hardware and software. That is why function testing will be highly prioritized by the group and most of the "hands on" time will be spent on it.

The ensuing subsections describe each of the explicit steps necessary for every hardware and software component, along with a final subsection describing the testing for the project as a whole. A description of each test that will occur is stated at the beginning of the subsections, along with the steps that the group member will do for actual testing. After all the steps are completed, an explanation of what the expected result will be is stated. If the expected result or any of the conditions for success do not occur after the steps have been completed then the structure must undergo meticulous examination to determine the problem. After a solution is applied, the steps will be repeated for re-testing, continuing the cycle until the expected result and all conditions are met. Until that happens, the component cannot be added to the comprehensive design. This permits that every time two components come together, the merging process will be as smooth as possible.

6.1 Hardware Testing

6.1.1 PCB and Main Housing Testing

Procedure:

1. Perform an initial check of all wire connections from PCB housing to verify connections are secure
2. While actively tracking, verify servo wires do not snag or become tangled in the servo armature.
3. Determine top PCB housing box will open and latch properly
4. Verify length of wire connections from internal power supply is acceptable.

Required Outcome: Both PCB and main housing secure appropriate components and wire length is acceptable

Conditional Requirements:

- Housing remains stable during servo actuation

6.1.2 Power Supply and Battery Testing

Before electrical components can be tested the batteries and sources supplying the power to these components will need to be tested. From this test, it can be concluded that no devices are defunct and all systems on the sentry turret system will be provided with the appropriate power. This test is crucial before beginning tests of servos and warning systems.

Procedure:

1. Using a standard 120V wall mount, plug the AC-DC adapter into the wall with the power cord provided
2. Observe the PCB state and verify that the appropriate LED lights will turn on for the subsystems that the power source is connected to
3. Verify servomotors can move at rated speed
4. Unplug the power cord from the wall
5. Verify system powers down
6. Turn on the internal battery
7. Verify the PCB and subsystems power on
8. Verify servomotors can move at rated speed
9. Turn off system and verify all components turn off

Required Outcome: Using the external power from the 120V wall source and the internal battery supply, the sentry turret system and all subsystems are able to power up and run at required specifications. The system also will power down with no troubleshooting or further issues to report.

Conditional Requirements:

- All components receive the appropriate power

6.1.3 Servo Testing

Procedure:

1. Perform an initial dry run of servos to conclude no servos are dead-on-arrival
2. Mount servos to designated locations.
3. Connect the servos to varying power supply and PCB
4. Run steps 5- using 4.8 volts and repeat using 6 volts.
5. Verify the servos have full degree range of motion
6. Verify positioning servos have sufficient torque to rotate a load within its specifications without strain
7. Verify trigger servo can pull a 2lb trigger without strain
8. Cut power to servos after continuous movement for 10 seconds and verify servos return to the neutral position

Required Outcome: Servos respond accordingly to PCB inputs. All servos meet desired specifications and do not fail under maximum load.

Conditional Requirements:

- Servos have smoothly travel through there range of motion

6.1.4 Camera to Tablet Testing

Once the camera system seems to be functioning with the use of a laptop computer, then next step is to test that it functions with the wirelessly connected tablet. This is tricky because the camera will need to be connected to the PCB directly and the signal will need to be transmitted wirelessly to the tablet. This wireless transmission could affect the camera feed and picture quality so further testing must be completed.

Procedure:

1. Power on PCB and Tablet
2. Connect USB camera to the PCB
3. Establish a connection between PCB and tablet
4. Run Processing program on the Tablet
5. View video feed on the tablet using the user interface
6. Separate the tablet from the PCB and camera by a distance of at least 5 meters to check for effective range.

Required Outcome: The camera should function like it did when the laptop computer was used. There should be a clear image and there should not be any noticeable delay in the camera feed when viewing through the user interface. A reasonable distance can be place between the turret system and the tablet without loss of connection or transmission speed interference.

Conditional Requirements:

- Camera was able to establish a secure wireless connection with the tablet
- Video feed was accurate and clear
- Real time image representation was achieved from a distance of over 5 meters away from the turret system

6.1.5 Tablet-Microcontroller Testing

By the use of the XBee 802.15.4 data will be sent between the tablet and microcontroller. This test will ultimately decide whether the servos are receiving the correct signals from the PCB.

Procedure:

1. Attach XBee receiver module to PCB via the correct pin placement
2. Connect XBee transmitter to tablet
3. Power on all devices
4. Set up connection between transmitter and receiver
5. Run default software on tablet
6. Send basic signal to board
7. Observe if input and output are recorded

Required Outcome: The board responds to the tablet

Conditional Requirements:

- XBee connection between devices is correct
- Data is sent and received with zero delay
- Board responds instantly
- XBees go into standby when not in use

6.1.6 Weapon Platform Clamp Testing

The purpose for testing the clamp mechanics is to ensure that the variety of weapon platforms mentioned it this document will be held properly. This ranges from the prototype design model, being an airsoft gun, to smaller lasers.

Procedure:

1. Attach the rubber clamp to the two bolts so both clamps will be facing each other in parallel
2. Using the bolt nuts, tighten the clamp to avoid it from falling off the servo positioning armature system
3. Position the airsoft gun between the two rubber clamps. Do not attach the trigger servo
4. Using the nuts, tighten the clamps to form to the airsoft gun
5. Provide power to the sentry turret system and begin actively tracking a target
6. Verify that under movement, the rubber clamps will hold the airsoft gun stable and no wobble can be viewed
7. Repeat steps 1 through 6 with a standard stick laser

Required Outcome: The rubber clamp can hold both the maximum and minimum size of platforms for accurate targeting.

Conditional Requirements:

- All platforms remain stable under movement.

6.1.7 Airsoft Gun Testing

The airsoft gun testing will be comprised o only the mechanics of the gun. A separate testing section is required to judge accuracy and precision of the BBs fired.

Procedure:

1. Attach the internal battery to the airsoft gun
2. Load the airsoft gun's magazine with the required BBs
3. Fire the airsoft gun of semi auto mode
4. Fire the airsoft gun on full automatic mode
5. Detach the magazine
6. Mount the airsoft gun to the rubber clamps on the servo armature system
7. Repeat steps 2 through 5

Required Outcome: The airsoft gun's mechanics work properly under both user hand-held operation and clamp-held operation.

Conditional Requirements:

- Compression from clamps does not affect the performance of the airsoft gun.

6.1.8 BB Accuracy Test

Following testing of the actual mechanics of the airsoft gun, it is necessary to test the accuracy and precisions of the shots fired under operation. As with airsoft gun testing, procedure will be broken down with hand held operation first before attaching the weapon platform to the system. The reason for doing so is to easily adjust the airsoft gun's hop-up unit. Therefore, the airsoft gun will be properly calibrated for distance when it is attached to the rubber clamps. Additionally, since the system only requires full automatic mode, semi-automatic mode will not be included. The test for tis will be performed in a non-populated area with a sufficient amount of space to avoid any injuries. Please note that at increased ranges of testing, the BBs will naturally lose velocity and begin to drop. The main purpose for the test is the correctly calibrate the hop-up to prevent premature BB rise or fall.

Procedure:

1. Attach the battery and fully loaded magazine to the airsoft gun
2. Switch the firing mode from safety to full auto
3. Aim at a target of 25 ft in distance from the airsoft gun
4. Fire at the target in three round bursts at least 5 times
5. Adjust the hop-up accordingly based on the BB trajectory
6. Repeat steps 4 and 5 at a distance of 50 and 100 feet
7. Adjust the hop-up based on BB trajectory again

8. Attach the airsoft gun to the rubber clamps
9. Manually fire the airsoft gun at a target
10. Record hit and miss data

Required Outcome: The airsoft gun's hop-up is adjusted to its optimal settings and can hit targets at desired ranges.

Conditional Requirements:

- Compression from clamps does not affect the performance of the airsoft gun.

6.1.9 Microcontroller Testing

Using the Arduino IDE to implement a blink light program

Procedure:

1. Load program into IDE from sample code provided by Arduino
2. Compile and upload program using AVR programmer
3. Assign LED to the programmed output pin
4. Carefully monitor Blinking LED
5. Connect power supply and remove AVR programmer
6. LED should continue to blink
7. Using Hyper terminal transmit data using serial communication

Required Outcome: Microcontroller responds to serial transmission

Conditional Requirements:

- Led continues to blink with USB disconnected
- Serial transmission through hyper terminal exact
- Board instantly responds

6.1.10 Power System

To assure a stable output from power system, all batteries used in the system will be tested in their appropriate locations.

Procedure:

1. Connect battery to system
2. Using an oscilloscope measure DC waveform
3. Insure proper waveform
4. Remove battery
5. Plug in AC power source and USB port
6. Measure waveform using oscilloscope
7. Connect battery
8. Read the battery voltage from the microcontroller

Required Outcome: All components should receive peak voltage level

Conditional Requirements:

- Battery waveform matches AC waveform across motors
- All components work together
- Battery is able to recharge
- At optimal battery charge the battery stops charging

6.1.11 Siren Testing

Checking to see if the siren goes off when an object enters the camera's field of view.

Procedure:

1. Power on entire system
2. Open up user interface program on tablet
3. Turn on automatic firing mode
4. Have someone walk in front of turret

Required Outcome: When the person walks in front of the gun a siren will sound and lights will flash

Conditions of Success:

- As soon as a target enters the camera 's field of vision, the siren will go off
- No delay between target acquisition and siren signaling

6.2 Software Testing

6.2.1 Video Subsystem Testing

In order to test the tracking system of this project the first and most important part is the video subsystem. If the camera is not working well with the tracking program or the tablet that is implementing the program then nothing will work. The video system will first be tested on a Windows based laptop to check for bugs in the Processing based code.

Procedure:

1. Turn on the power system for the PCB and turn on the laptop
2. Check that any required software and drivers related to camera function are properly installed and working
3. Plug in the USB webcam to the USB port of the laptop
4. Connect the laptop to the PCB via USB cable
5. Run the program on the laptop and check to see if images from the camera are appearing in the user interface on the tablet
6. Have volunteers move in and out of the visual range at varying speeds and in varying patterns to check that frame rate and visual clarity are acceptable

Required Outcome: The video feed from the camera is appearing on the video feed window of the user interface on the laptop.

Conditional Requirements:

- Minimal delay in camera feed.
- Clear image that does not streak or blur
- Camera does not lose focus when multiple objects appear on screen

6.2.2 Object Detection Testing

Target acquisition is the first step in the motion tracking process and must first be tested on a laptop to insure that the program is functioning properly.

Procedure:

1. Power on PCB and Laptop
2. Connect laptop to the PCB via USB cable
3. Face turret and camera system towards an open area with no other moving objects
4. Secure camera so that turret motion will not create vibrations or movement in the recorded image
5. Run Processing IDE and run the tracking program
6. Have a volunteer enter the field of view and stop
7. Have two volunteers cross paths in front of the camera's field of view and stop then continue until out of sight.

Required Outcome: The targeting software should be able to pick up that a target has entered the field of view and place a crosshair symbol on that target. If multiple targets appear on screen then a priority target must be selected based on speed and distance from the borders of the field of view.

Conditional Requirements:

- Program can identify a target object is in the field of view
- Distinguishing mark is seen on target through the user interface
- Program can acquire multiple targets simultaneously and select a priority based on movement patterns

6.2.3 Motion Tracking Testing

The ability to track the objects that have been detected is the next logical requirement of this system. There must be a fast and accurate response from the tracking program as the objects/targets attempt to evade the system. If multiple targets are present then the system must follow the target that has priority, which is calculated by position in the field of view.

Procedure:

1. Power on PCB and Laptop
2. Open the Processing IDE and run the program
3. Have volunteer move across the field of view at multiple speeds and patterns.

4. Check agility of tracking system by having a volunteer move through the field of view in an erratic path of varying speeds.
5. Check the responsiveness of the system by having a volunteer run past the camera at maximum speed and varying distances.
6. Have a volunteer come to a complete stop at any point during the pass and see if the tracking system maintains fire on target.
7. Monitor the user interface to see that the targets are being followed while they remain in the field of view.
8. Check to see if system resets to an origin at the center of the field of view after a preset time with no targets located.
9. Move an inanimate object into the field of view and check to see how the tracking system reacts.

Required Outcome: Any targets detected in the field of view are followed by the turret system while the target remains moving in the viewing range. Targets are tracked and fired upon accurately while attempting evasive maneuvers. Targeting software detects threats even if they stop moving but are still in plain sight. Changes in background ignored after a period of time, items not matching target features not considered for attack.

Conditional Requirements:

- Little to no delay in tracking between target acquisitions.
- Smooth transition from standby to targeting modes.
- Center of target is marked as point of aim for the turret.
- Sentry returns to preset origin if no targets present for over 6 seconds
- Background refreshed periodically to account for changes in environment.

6.2.4 Tablet Tracking Software Testing

Once the camera system is properly connected to the tablet through the wireless interface, the tracking system must then be tested for use on the tablet. Since the tablet is going to be running the full version of Windows, integration should be simple. If the tracking program cannot be implemented in the same form as it was on the laptop then debugging and redesign might be required.

Procedure:

1. Power on the PCB and Tablet
2. Acquire wireless connection between tablet and the turret system
3. Open the Processing IDE on the tablet
4. Run the all necessary programs
    a. Close all other programs so that the tablet's processing power is devoted to the tracking software
    b. Check that no errors or compatibility warnings are present
5. Move tablet/user interface to a distance of at least 5 meters away from turret system.
6. Have a volunteer target enter the field of view of the camera
    a. Have volunteer target run at maximum speed from one end of the screen to the other.

b. Repeat last test but at average walking speed
c. Have volunteer attempt evasive maneuvers while passing through the field of view
d. Have volunteer hide behind an inanimate object midway through the path, and then return to medium speed across the field.
e. Test that program can handle targets from an oncoming direction by having a volunteer run directly at the turret.

Required Outcome: The motion detection and tracking program successfully compiles on the tablet and the user interface enables a first person view of the turrets targeting range. All user commands are functioning properly from the tablet and the tablet continues to work at a distance of over 5 meters.

Conditional Requirements:

- No compatibility related bugs occur from transition to tablet
- Targeting software is responsive and accurate
- User interface displays real-time tracking
- Tracking system can handle incoming targets as well as targets traveling along the X axis of the field of view.
- Distance from turret does not noticeably affect function or speed of target acquisition.

6.2.5 Arduino Servo Library

Using Arduino Servo Library, this test will convert analog input to digital output.

Procedure:

1. Connect microcontroller to tablet
2. Power on all devices
3. Open Arduino IDE
4. Open team created user interface, select Manual Mode
5. Send a set of coordinates to the board
6. Check output signal

Required Outcome: An angle value will be shown on screen

Conditional Requirements:

- The custom board convert analog data to digital
- The board keeps receiving signals until the user stops
- Proper angle of servos is given

6.2.6 Manual Mode

Using the Arduino Servo Library, this test will take user input from the tablet and manually move the servos.

Procedure:

1. Connect servos to microcontroller

2. Connect microcontroller to tablet
3. Power on all devices
4. Open Arduino IDE
5. Open group created user interface, select Manual Mode
6. Press directional arrows on user interface

Required Outcome: The servos will move in the proper direction that the user is inputting

Conditional Requirements:

- A proper connection is held between the board and tablet along with the servos and board
- Arduino code sends out the correct signals
- Movement of the servos is in the direction the user expects

6.2.7 Custom User Interface Video Stream

This test will **c**heck if the user interface created by the group recognizes and displays the video stream.

Procedure:

1. Connect camera to XBee Wi-Fi transmitter via Explorer dongle
2. Connect XBee Wi-Fi receiver to tablet via Explorer dongle
3. Power on camera and tablet
4. Run custom Programming language program so camera can begin transmission
5. Observe if video is shown and determine amount of delay

Required Outcome: The live video stream will be clear with no noticeable delay

Conditional Requirements:

- The XBee connection between the transmitter and receiver is steady
- The group created program picks up the signal of the video camera
- Video is shown on the tablet in real time

6.3 Overall Testing

After all previous tests in both hardware and software sections are run, the Overall Testing will be the final testing step. When combining all subsystems together, the complete design will be examined ensuring all requirements and goals are met. To accurately perform an overall test, it has been agreed upon to track a human target. This will most fully represent the unpredictability of motion from a target and let the sentry turret system work at its fullest potential. This process will span from battery supply to servo actuation and firing modes. Before the procedure for testing commences, the sentry turret system must be fully assembled and placed in an area deemed safe for testing. Because the test will involve the use of a fully loaded airsoft gun, all user testers and tester subject will also commence with safety measures. This includes the use of the following:

- Impact Glasses (to prevent a BB hitting the eye)
- Protective Clothing (for the target tester)
- Background tarp to catch stray BBs

With the following parameters set, the group can begin the overall test. The testing will be split into a section for automated firing and another for manual firing.

6.3.1 Automated Testing

Procedure:

1. Verify all power connections are working
2. Have one or more users stand behind the sentry turret to observe overall test
3. Have the test subject come into the field of view of the tracking camera
4. Verify all add on features (warnings) function
5. The test subject shall perform the following tests
    a. Backwards movement (increase distance)
    b. Side-to-side movement
    c. Random movement
    d. Stationary with quick movement intervals
6. Step out of field of view

Expected Results: The sentry turret system will track the target through every movement type and fire BBs.

Conditions of Success:

- The automated firing mode of the turret functions according to design.

6.3.1 Manual Testing

Procedure:

1. Verify all power connections are working
2. Have one user detach the tablet from the main housing
3. Switch the system to manual mode from the tablet
4. Test the direction of the servos by using the directional keys on the user interface of the tablet
5. Move the airsoft gun via control from the tablet to aim at a specified target
6. Pull the trigger of the airsoft gun using the fire command button on the tablet
7. Repeat steps 4 through 6 using a moving target

Expected Results: The sentry turret system is able to effectively track and fire upon a target with minimal delay time.

Conditions of Success:

- The manual firing mode of the turret functions according to design.

With all primary tests and the overall testing complete, it can be concluded that the main system of the sentry turret and its subsystems all will function properly when the project is presented and run in a live exercise. In the case a component does not meet the required specifications or fails to work, this outlined testing section can be repeated to find the root cause.

Section 7: Administrative Content

7.1 Project Milestone

The first semester required the group to be focused on quickly determining what would work for this project and what would not. This involved learning from the mistakes of similar group projects in the past. In order to get the most relevant data many small prototypes were built to narrow down the parts that would need to be researched. Research quickly became the most time consuming part of the first semester. With the wide array of programs, hardware, and design options it became clear that this project could be completed in a multitude of ways, the hard part was deciding on the best way that would result in a working product, as well as stay within the low budget that was set. The amount of relevant data regarding similar designs actually hindered the creative process at first because there were so many directions that the group could go in for a final design.

After the group decided on the sentry gun turret system the testing process began almost immediately. The group was fortunate as a whole due to the fact that a few of its members were in possession of many useful prototyping tools including bread boards and development boards like the Arduino Uno. Within a month of forming the group there had already been multiple meetings to discuss design ideas and formulate a solid plan for completing the project. The first step taken to begin testing was to purchase several low end servo motors and begin experimenting with the Arduino and some simple code based off of the open source libraries found online. As testing continued it allowed the group to get a better understanding of what needed to be done to get a full scale version of this design completed.

The next milestone was deciding that the entire system was to be run on a tablet and not on a laptop computer or similar computing device like most other projects had done in the past. The initial thought was it was to be developed on an Android based tablet using a custom made Android application. Since all prototyping thus far had been achieved using a Windows based computer it seemed more logical try and transfer the work to a Windows based tablet instead of completely starting from scratch in order to get the programs working on a different operating system. This is when the Windows tablet was selected.

Since the second half of this project will be held during the summer semester, the group has decided that work should be started immediately on the final design. This means that some of the short break between the spring and summer semesters will be used to begin building. Since the group project was not awarded with a very large budget, a lot of items must be purchased individually. Group members have volunteered to make some of the larger purchases like the Windows tablet and the airsoft rifle in order to save the budget for the hardware design and fabrication. Using the information found while researching parts and supplies, a list of components has been made and will be used to start ordering parts like the servo motors, power supply, and most of the small electronics needed to start laying out a PCB design. The next step is to create a bread board PCB design that eliminates the need for the Aruduino Uno. Since the group will be using the same microprocessor that the Uno uses, integration of current code and software should be simple and fast. This will allow the group to test the layout of the circuit before ordering a PCB and having all of the parts mounted. So far the progress on the project is continuing as hoped and most of the planning is set in motion. A layout of the milestones accomplished so far as well as the goals for the future are shown in the table below.

Semester	Week	Date	Milestone
1	1	01-07-14	First day of SD1
1	2	01-14-14	Group formed
1	3	01-21-14	Sentry Gun selected as project description
1	4	01-30-14	Funding request paper submitted in class
1	5	02-06-14	Paper divided among group members
1	6	02-13-14	Initial project design ideas due
1	7	02-20-14	Begin researching code and prototyping hardware
1	8	02-27-14	Test image processing using a webcam on a laptop and simple motion tracking code
1	9	03-13-14	Breadboard a power circuit to power the development board and servo motors
1	10	03-20-14	Servo control testing using Arduino Uno and laptop
1	11	03-27-14	Prototype tracking software combined with servos and tested for functionality
1	12	04-03-14	Group meeting to format rough draft of paper
1	13	04-10-14	Rough draft of paper submitted in class
1	14	04-17-14	Windows based tablet chosen for user interface and control
1	15	04-26-14	120 page rough draft due on Dropbox for formatting
1	16	04-29-14	Final draft of SD1 due
Break		04-30-14	Group meeting discussing parts & supply acquisition
		05-02-14	First test with tablet as main computational source
2	1	05-13-14	Turret housing completed
2	2	05-20-14	Tracking system completed
2	3	05-27-14	PCB Design completed
2	4	06-05-14	PCB mounted and tested
2	5	06-12-14	Rifle, servos, and power system mounted to housing
2	6	06-20-14	Website built
2	7	06-24-14	First live fire function testing
2	8	06-29-14	
2	9	07-04-14	Final design issues and bugs pinpointed & corrected
2	10	07-11-14	
2	11	07-19-14	Final Project Report submitted
2	12	07-26-14	Final Presentation Completed

Table 7.1 – Project Milestone

7.2 Finance and Sponsorship

As research, and final design began to come along, it became more evident to which components/parts would be required for the turret. Initially the first thought that came to mind is the turret would cost a very minimal amount compared to other proposed projects. As complexity began to grow it became more evident the project would cost more than originally assumed. Dr. Richie also advised that a usually project senior design adds up $20.00 at a time. It was then when the group applied for sponsorship to handle expenses. Luckily there was sponsorship openings from Boeing still open. After filling out the application, and a short while wait for the application to be processed, the group was rewarded $476.46 from the $761.89 initially requested. Although there was greater funding if the Duke Energy sponsorship was used, but since the project was not environmental in any way, it was assumed that more likely the request would be denied.

Along with all the components/parts necessary to construct the turret, it was also necessary to consider any tools that were obligatory to test and prototype the circuits. These included oscilloscopes, function generators, power supply, soldering iron, and cables. Luckily the UCF department of Electrical Engineering provides the senior design lab for students to store and work on projects. Although it is not obligatory to work in the senior design lab, it would be preferred due to tools being already present. As for the rest, everything will be either purchased or collected from already owned products. Retailers such as Skycfraft Parts & Surplus will be used to purchase housing parts, and any other parts that may be necessary since they seem to have a variety of everything. To minimize cost, many electrical components, and parts will be ordered online since it would be more cost effective. Things to consider though is purchasing all the parts together would reduce the cost of overall shipping. As for the PCB, 2 boards will be ordered at a single time. This will reduce the cost in case something goes wrong to the first one. A table below organizes all the parts, price, and where they will be purchased from. A miscellaneous category is added which will consist of minor parts purchased that include, screws, connecters, and etc.

7.2.1 Budgeting

Component	Quantity	Price	Extended	Acquired
Atmel Mega 328	2	$3.23	$6.46	Yes
PCB	2	$33.00	$66.00	No
RGB LED (10 Pack)	3	$4.95	$14.85	No
AudioLarm II	1	$21.27	$21.27	No
Web Camera	1	$35.00	$35.00	No
Primary battery pack	1	$39.99	$39.99	No
Secondary battery pack	1	$8.69	$8.69	No
X-BEE PRO	2	$37.95	$75.90	No
X-BEE Wi-Fi	2	$39.00	$78.00	No
Microphone	1	$4.00	$4.00	No
USB Mini-B	1	$3.95	$3.95	No
Arduino UNO	1	$39.99	$39.99	Yes
Windows Tablet	1	$319.99	$319.99	Yes
LT1510	1	$5.13	$5.13	No
Maxim 1704	1	$3.00	$3.00	No
HS-5685MH Servo Motor	2	$39.99	$79.98	No
HS-5055MG Servo Motor	1	$17.99	$17.99	No
Wooden Legs	4	$4.00	$16.00	No
Lumber	1	$30.00	$30.00	No
Miscellaneous	1	$50.00	$50.00	No
AVR mini programmer	1	$9.95	$9.95	Yes
LM7805 (3 Pack)	2	$0.99	$1.98	Yes
		Grand Total	**$928.12**	

Table 7.2 - Budget

Section 8: Appendices

8.1 Bibliography

“About Charmed Labs” Charmed Labs , n.d.Web.
http://charmedlabs.com/default/?page_id=2

"A Look at the Basics of Bluetooth Technology." *Basics | Bluetooth Technology Website*. Bluetooth SIG, Inc, n.d. <http://www.bluetooth.com/Pages/Basics.aspx>

"Arduino - Serial." *Arduino*. N.p., n.d. Web. <http://arduino.cc/en/reference/Serial>

"Arduino - Servo." *Arduino*. N.p., n.d. Web. <http://arduino.cc/en/reference/servo>

"ATmega328." *ATmega328*. N.p., n.d. Web.
<http://www.atmel.com/devices/atmega328.aspx>.

Beauregard, Brett. "Arduino PID Library." *Arduino Playground*. Arduino, n.d. Web.
<http://playground.arduino.cc/Code/PIDLibrary>

"Bluetooth Basics." *Learn.SFE*. SparkFun, n.d. Web.
<https://learn.sparkfun.com/tutorials/bluetooth-basics/wireless-comparison>

"Bluetooth Power Classes." Bluetooth Insight. N.p., 11 Jan. 2008. Web.,from:
<http://bluetoothinsight.blogspot.com/2008/01/bluetooth-power-classes.html

“Facial Detection using Haar Cascades” Docs.OpenCV.n.d.Web.
http://docs.opencv.org/trunk/doc/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html

Hannifin, Parker. "Fundamentals of Servo Motion Control." *Parker Motion*. Parker Motion, n.d. Web.
<http://www.parkermotion.com/whitepages/ServoFundamentals.pdf>.

“How Servo Motors Work” JameCo Electronics, n.d.Web.
http://www.jameco.com/Jameco/workshop/howitworks/how-servo-motors-work.html

Library Tutorial: JMyron” Silently Crashing , n.d .Web.
http://www.silentlycrashing.net/p5/libs/jmyron/

Ochoa, Gerardo H. "A Swarm of Xbees! Arduino Xbee Wireless & More." *Bildr*. N.p., 15 Apr. 2011. Web. <http://bildr.org/2011/04/arduino-xbee-wireless/>

“Pixy (CMUcam5): a fast, easy to use image sensor” Kickstarter, n.d.Web
https://www.kickstarter.com/projects/254449872/pixy-cmucam5-a-fast-easy-to-use-vision-sensor

"Pocket AVR Programmer PGM-09825 RoHS Open Source Hardware Share Use This URL to Share: Share on Twitter Share on Facebook Share on Google+." *Pocket AVR Programmer*. N.p., n.d. Web. <https://www.sparkfun.com/products/9825>.

"Processing G. SHRIKANTH , K. SUBRAMANIAN "Implementation of FPGA-Based Object Tracking Algorithm" gatech.edu, N.p ,Apr. 2008.Web. http://www.cc.gatech.edu/~ksubrama/files/FPGA_Report.pdf

"Sensors – Sharp IR Range Finder" Society of Robots , n.d.Web http://www.societyofrobots.com/sensors_sharpirrange.shtml

"Servo Power & Speed." *Servo Power & Speed*. ServoCity, n.d. Web. <http://www.servocity.com/html/servo_power___speed.html>.

"Stepper vs Servo Motors." *Stepper vs Servo Motors*. N.p., 01 May 2007. Web. <http://www.cncroutersource.com/stepper-vs-servo.html>.

Swingley, Christopher. "Remote Sensing with Arduino and XBee." *Swingley Development*. N.p., 15 May 2011. Web. <http://swingleydev.org/blog/p/1896/>

"The PID Controller." *ECircuit Center*. ECircuit Center, n.d. Web. <http://www.ecircuitcenter.com/Circuits/pid1/pid1.htm>

Wallbank, Derek. "Congress Unveils $1.1 Trillion Plan to Fund Government."*Bloomberg Business Week*. Bloomberg, 14 Jan. 2014. Web. <http://www.businessweek.com/news/2014-01-14/congress-unveils-1-dot-1-trillion-measure-to-fund-u-dot-s-dot-government>

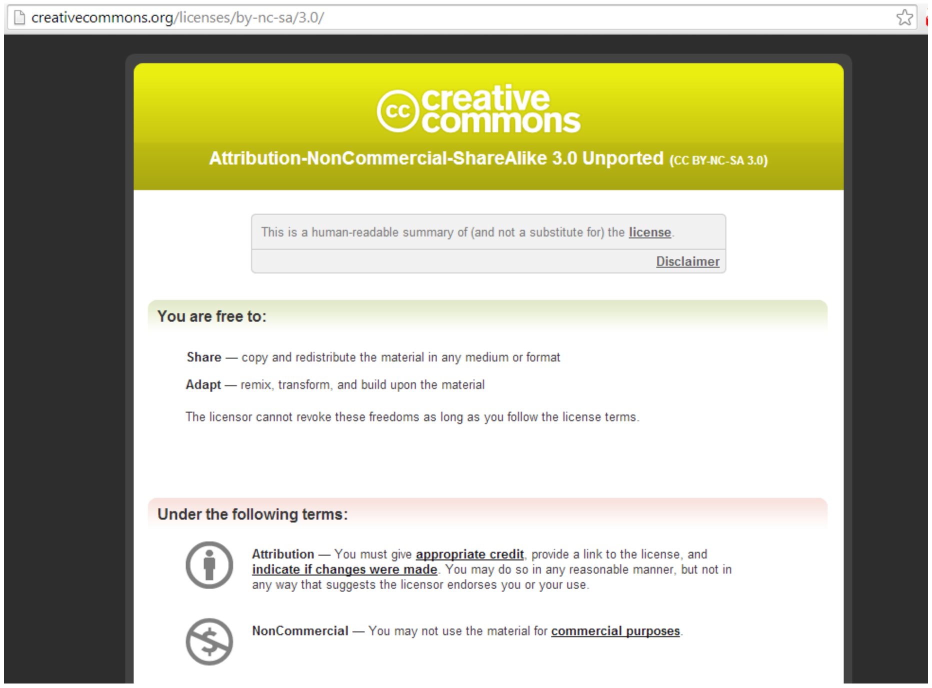
"Wireless USB FAQ." *Everything USB*. Everything USB, n.d. Web. <http://www.everythingusb.com/wireless-usb.html>

"Xeva-1.7-640 SWIR Camera" Xenics Infrared Solutions, n.d.Web http://www.xenics.com/en/infrared_camera/visnir-nir_camera_-_visual_near_and_near_infrared_cameras_-_ingaas/xeva_near_ir_camera_-_high-res_-_ingaas_fpa.asp

"ZigBee Technology." ZigBee. ZigBee Alliance, n.d. Web. <https://www.zigbee.org/About/AboutTechnology/ZigBeeTechnology.aspx>

8.2 Authorization

Authorization for SparkFun:



Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0)

This is a human-readable summary of (and not a substitute for) the license.

Disclaimer

You are free to:

Share — copy and redistribute the material in any medium or format

Adapt — remix, transform, and build upon the material

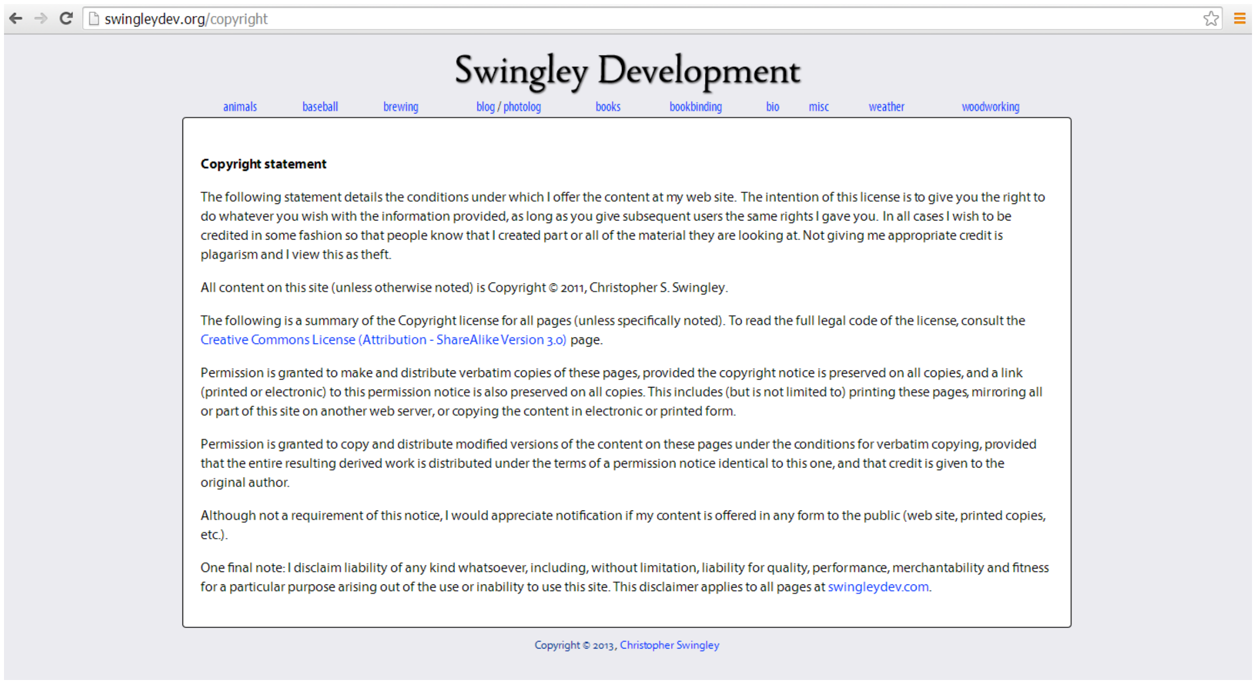
The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

Attribution — You must give **appropriate credit**, provide a link to the license, and **indicate if changes were made**. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

NonCommercial — You may not use the material for **commercial purposes**.

Authorization from Christopher Swingley:



Copyright statement

The following statement details the conditions under which I offer the content at my web site. The intention of this license is to give you the right to do whatever you wish with the information provided, as long as you give subsequent users the same rights I gave you. In all cases I wish to be credited in some fashion so that people know that I created part or all of the material they are looking at. Not giving me appropriate credit is plagarism and I view this as theft.

All content on this site (unless otherwise noted) is Copyright © 2011, Christopher S. Swingley.

The following is a summary of the Copyright license for all pages (unless specifically noted). To read the full legal code of the license, consult the Creative Commons License (Attribution - ShareAlike Version 3.0) page.

Permission is granted to make and distribute verbatim copies of these pages, provided the copyright notice is preserved on all copies, and a link (printed or electronic) to this permission notice is also preserved on all copies. This includes (but is not limited to) printing these pages, mirroring all or part of this site on another web server, or copying the content in electronic or printed form.

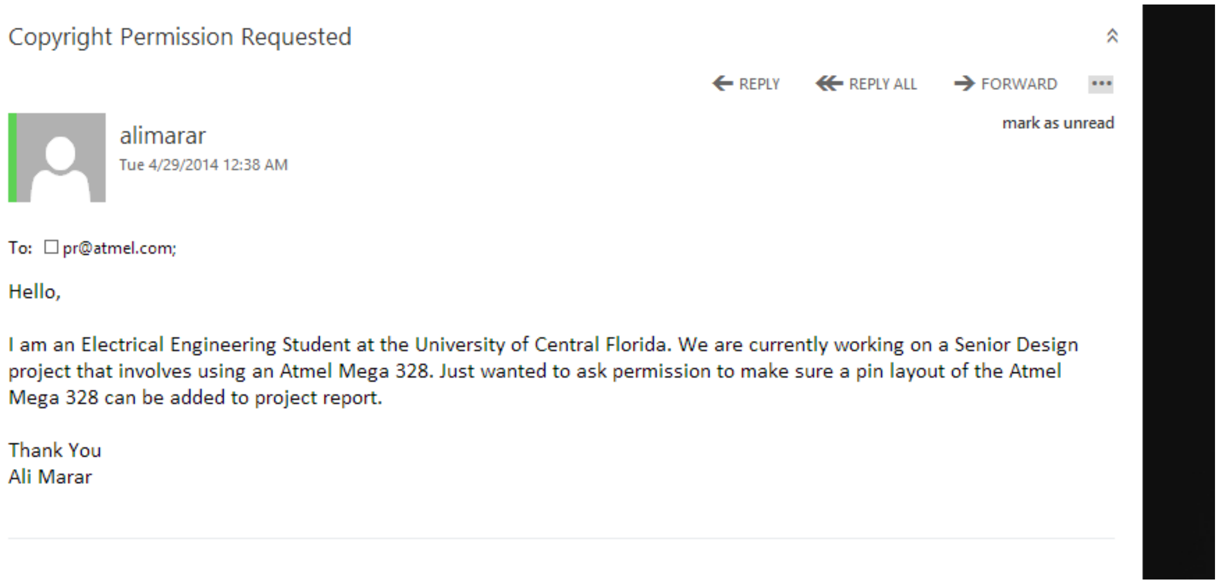
Permission is granted to copy and distribute modified versions of the content on these pages under the conditions for verbatim copying, provided that the entire resulting derived work is distributed under the terms of a permission notice identical to this one, and that credit is given to the original author.

Although not a requirement of this notice, I would appreciate notification if my content is offered in any form to the public (web site, printed copies, etc.).

One final note: I disclaim liability of any kind whatsoever, including, without limitation, liability for quality, performance, merchantability and fitness for a particular purpose arising out of the use or inability to use this site. This disclaimer applies to all pages at swingleydev.com.

Copyright © 2013, Christopher Swingley

Requested Permission from Atmel Corporation:



Copyright Permission Requested

REPLY REPLY ALL FORWARD

mark as unread

alimarar
Tue 4/29/2014 12:38 AM

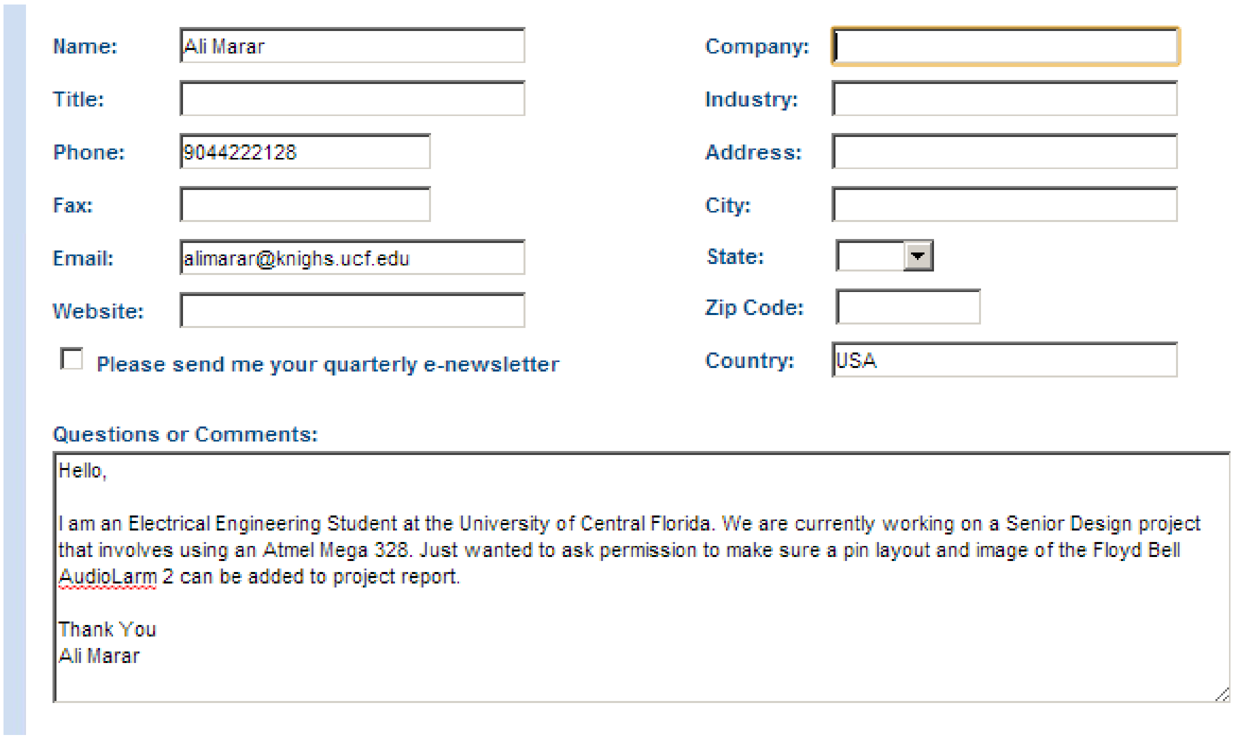
To: pr@atmel.com;

Hello,

I am an Electrical Engineering Student at the University of Central Florida. We are currently working on a Senior Design project that involves using an Atmel Mega 328. Just wanted to ask permission to make sure a pin layout of the Atmel Mega 328 can be added to project report.

Thank You
Ali Marar

Requested Permission from Floyd Bell:



Name:	Ali Marar	Company:	
Title:		Industry:	
Phone:	9044222128	Address:	
Fax:		City:	
Email:	alimarar@knighs.ucf.edu	State:	
Website:		Zip Code:	
☐ Please send me your quarterly e-newsletter		Country:	USA

Questions or Comments:

Hello,

I am an Electrical Engineering Student at the University of Central Florida. We are currently working on a Senior Design project that involves using an Atmel Mega 328. Just wanted to ask permission to make sure a pin layout and image of the Floyd Bell AudioLarm 2 can be added to project report.

Thank You
Ali Marar

8.3 Annotations

[1] Wallbank, Derek. "Congress Unveils $1.1 Trillion Plan to Fund Government."*Bloomberg Business Week*. Bloomberg, 14 Jan. 2014. Web. <http://www.businessweek.com/news/2014-01-14/congress-unveils-1-dot-1-trillion-measure-to-fund-u-dot-s-dot-government>

[2] "Bluetooth Basics." *Learn.SFE*. SparkFun, n.d. Web. <https://learn.sparkfun.com/tutorials/bluetooth-basics/wireless-comparison>

[3] "Bluetooth Power Classes." Bluetooth Insight. N.p., 11 Jan. 2008. Web.,from: <http://bluetoothinsight.blogspot.com/2008/01/bluetooth-power-classes.html

[4] "Arduino - Servo." *Arduino*. N.p., n.d. Web. <http://arduino.cc/en/reference/servo>

[5] "Arduino - Serial." *Arduino*. N.p., n.d. Web. <http://arduino.cc/en/reference/Serial>

[6] "The PID Controller." *ECircuit Center*. ECircuit Center, n.d. Web. <http://www.ecircuitcenter.com/Circuits/pid1/pid1.htm>

[7] Hannifin, Parker. "Fundamentals of Servo Motion Control." *Parker Motion*. Parker Motion, n.d. Web. <http://www.parkermotion.com/whitepages/ServoFundamentals.pdf>.

[8] Beauregard, Brett. "Arduino PID Library." *Arduino Playground*. Arduino, n.d. Web. <http://playground.arduino.cc/Code/PIDLibrary>

[9] Swingley, Christopher. "Remote Sensing with Arduino and XBee." *Swingley Development*. N.p., 15 May 2011. Web. <http://swingleydev.org/blog/p/1896/>