

Project: Pan-Tilt-Zoom Camera

Group 26

Chisom Ikejiani - Computer Engineering

Andre Samaroo - Computer Engineering

Michael Serignese - Computer Engineering

Table of Contents

Title Page.....	i.
Table of Contents.....	ii.
Executive Summary.....	1
Project Description.....	1
-Motivation.....	2
-Objectives.....	3
-Requirement Specifications.....	4
-Standards and Constraints.....	5
-Marketing requirements.....	7
Technology	
Investigation.....	10
-Existing Products.....	10
-Camera.....	12
-Audio Systems.....	16
-Motors.....	18
-Electronic Speed Control.....	23
-Mount/Frame.....	29
-Main control board.....	30
-Power/Data Link.....	33
-Remote Control.....	37
Design.....	40
-Prototype.....	40
-Hardware.....	45
-Software.....	55
-Testing.....	79
Administration.....	86
-Budget.....	86
-Milestones.....	87
-Vendor.....	89
-Design	
issues.....	90
-Appendix.....	96

Executive Summary:

Motion control rigs are basically robotic cameramen. They control the camera's movements and even the camera settings. These systems allow you to execute difficult camera shots with the ease of programming the movements. This also allows us to take camera shots from difficult places and remotely.

The tall tripod-mounted action camera is meant to act as a sports camera that can rapidly pan, tilt, and zoom (also called a PTZ camera), rapidly responding via the user interface in order to capture fast-paced events from a vantage point that allows effective coverage of a large sports arena. The implementation we looked at was designed by Qwikcut. It is rather pricey for a system that does not quite meet a desirable maintenance schedule, where three cables for power, audio, and video appear to be grafted onto the camera system in such a way that they often fail. Additionally, the waterproofing for the system is lackluster and may cause failure on an incident of rain, a detrimental feature for a camera system that is meant to capture events often held outdoors. Therefore, the goal of this project is to offer a new design and implementation of the product that is lower cost, more tightly integrated, has superior waterproofing, and requires less maintenance than the initial design.

The camera will output audio and video data to a workstation computer that will be responsible for recording to disk, livestreaming the audio-video media content in real time, and issuing commands to the camera via the user interface of a game controller. The inputs of power and commands to the camera as well as the media output from it will flow over a USB cable capable of conducting the signal and wattage necessary for the device.

The camera will be controlled via a game controller, using two joysticks and at least four buttons to control pan, tilt, and zoom, starting and stopping the recording or livestream, increasing or decreasing the brightness of the video stream, and finally powering on and off the device. If there are additional interfaces on the controller that remain unused, the goal for these features is to reserve them to attain maximum control over the camera through the controller, and only deferring to the workstation computer as a secondary control mechanism.

Since this is an action camera, the responsiveness of the device both in terms of software control features and also in mechanical agility are paramount. The controller-camera logical connection, in tandem with the pan-tilt motor agility, must be bridged in such a way that commands transmitted via the user interface of the controller reach the camera.

As a stretch goal, we would like to automate the control rig in some way. We thought that a computer vision based tracker would be best. The camera will lock onto a subject in frame and then basically follow that person until the feature is turned off. This leads to another stretch goal of making the game controller communicate with the workstation wirelessly to allow for the person to control the camera remotely.

Project Description

Motivation:

In the PTZ camera world there exists many high-end cameras that can capture and stream high-quality sports action video phenomenally. Today's cameras can smoothly and safely capture under the roughest of conditions from the strongest of winds to the fiercest of rainstorms. Inside tight weather-proofed housing there are no worries or risk of damage to the investment. During sporting events, the cameras can track real-time movements like baseballs after being hit by a bat through the sky or a football player charging down a field at a high speed. Everything related to these PTZ cameras seems almost perfect except their ultimate costs, they can cost anywhere from \$1500 to \$10k+. For big sports organizations that price point may be miniscule but for the local high school or small youth teams it can be near impossible.

Our implementation of the PTZ camera seeks to solve the issue of price by combining and assembling a PTZ camera that preserves many of the qualities the high-end cameras have such as weather-resistance, real-time tracking, and remote controlling. The proposed camera will achieve its low price point by sourcing budget products such as Arduino and standard consumer cameras while using cheaper 3d printed materials for the housing and design. Our design will also cut out expensive controlling software used today by using a standard gaming controller as the remote and simple custom video software to both control and display video from the camera. It will be cheaper, more portable, and practical than any of the other PTZ cameras available today.

Our PTZ camera will be easy to use and setup, remote controlled camera that only needs a single laptop to work with. The light weightiness of design makes it perfect for the traveling camera man or local sport institutions. The targeted customer would be those without the financial means or incentive to invest in a high-end expensive PTZ camera, such as youth sports teams, local high schools, and general action sport hobbyists. The target group would find that the camera assists with not having to lug around heavy equipment and can be used without having to go through a steep learning curve since the remote control and software are simple to use.

Additionally, alternative PTZ cameras today are also equipped with complex and expensive control and streaming software that our project eliminates all together by using a simple to use and easily available gaming controller combined with custom computer software that will display video and push commands to the systems motors. Another problem our project indirectly solves as a byproduct of our minimalist approach is the cluster and entanglement of wires that occurs when using the bigger more expensive cameras and systems.

Objectives:

A key to having a successful project is to maintain clear objectives and goals that guide the mission, design, and implantation of the project at hand. After a few goals are established, the project will have enough direction and purpose to continue successfully.

Some of the objectives of this project are to assemble a PTZ camera that can shoot high quality video, withstand harsh weather, track real time tracking, maintaining a low price

point, and portability. However, the three of them are essential to the success of fulfilling the purpose of this project are listed below.

1. Shoot high quality video
2. Have suitable weather resistance
3. Be cost-effective

The first objective of the PTZ camera is the ability to shoot high quality video comparable to the standard set by alternative PTZ cameras today. The lowest quality considered high definition would be 720P at 30 fps. This quality is achievable in a wide selection of cameras and is within the budget of the project. The camera selected will also have to consider our minor objective such as portability, so the light-weight options will be preferred.

The second objective of the PTZ camera is to achieve a weatherproof design, able to capture and withstand winds 10 mph or less and heavy rain. This is one of the most essential objectives as some PTZ cameras used in sports must capture video steadily no matter the weather without damaging equipment. Device damage is inevitable without proper weatherproofing, so achieving dependable weatherproofing is a must.

The third objective of the PTZ camera is to remain cheap and cost-effective when compared to other PTZ cameras on the market. The low price point is achievable by first purchasing the desired quality of camera and focusing on using 3d printed parts for the housing, low-cost microcontrollers, and low-cost motors. The camera should also be able to maintain its quality without opting for cheapness where it is unnecessary.

Altogether the desired objectives work in tandem to fulfill a design that works to keep the cost cheap while providing, high quality video, harsh weather resistance, and portability. These goals and objectives serve as a guideline which will help keep the team directed and in agreement in order to ensure the success of this project.

Requirement Specifications:

Tripod-1

The tripod must be sturdy enough to provide mechanically stable video in wind gusts of 10 miles per hour.

Housing-1

The device housing must be waterproof in moderate rain, at 0.10 inches of rain per hour.

Video-1

The camera system must be capable of sampling 1080p at 30 frames per second, with a delay of less than one second for the samples to arrive.

Video-2

The camera subsystem must be under 1 pound.

Audio-1

The audio system must be able to sample audio in the following formats, or an equivalent format that offers comparable quality parameters: WAV (44.1kHz, 16 bit), MP3 (320Kbps), or OPUS (128 Kbps). The samples must not be delayed longer than 1 second to arrive.

Mount-1

The mounting connectors between the tripod, motor mounts, and camera system must be able to withstand minor force of 5 ft-lbs without being damaged.

Motor-1

The motor must be able to pan and tilt with an attached mass of at least 1 pound.

Controller-1

The controller must be able to connect to a workstation computer via USB2 or newer, and control the camera via the joysticks and buttons on the device.

Controller-2

The controller must be able to pan, tilt, zoom, and otherwise control the camera on the full range of the device's capabilities, initiating the commands from the controller in under 1 second.

Workstation-1

The workstation software must transmit the commands received from the controller in less than 0.5 seconds.

Workstation-2

The workstation software must process packets received from the camera system to prevent a packet backlog of over 1000 packets from forming.

Standards and Constraints:

We will be using multiple control boards to perform different functions. These boards will communicate through pin connections from the general-purpose input/output (GPIO) ports. Most devices have a standard layout for these pins. On Raspberry Pi devices, the pin layout uses a 2 by 20 design located to one side of the board. On Arduino, the pins are split into 1 by 20 blocks, one on each side of the board. There may also be additional pins such as RVR pins for I2C connections. Pin block designs can be found documented in the control board's specification sheet. [15]

We will be using soldering to attach pin connectors to the GPIO ports to allow the PWM board and main control board to connect easily. We will follow IPC soldering standards to make sure that our pin connections are secure and that we remain safe while soldering. IPC-J-STD-001 defines standard soldering practice for PCB's. However, the IPC standards are not freely available so we will look at the NASA standards which adopt many of the IPC standards, but are available to the public for free. This standard advises us on what kind of solder we should use. The advice is to go to lead-less solder which is less toxic. These lead-free alloys use tin and silver in place of lead. [12][13]

To properly solder the pin connections, we must hold the pin headers to the ports and heat the pin heads. Heating the pins before applying solder ensures that the solder sticks to all the surfaces. If we solder to a cold surface, the solder will just cool on contact and just form a droplet on the surface of the pin and pad. The best practice for applying solder is to melt it on the soldering iron and then apply the small drop to the area around the pin head. Applying small amounts of solder helps us avoid flooding the board and forming an unintentional short in the circuitry. Just enough solder should be applied to cover the GPIO pad around the port to avoid creating bridges and shorts between the ports. The section on through hole soldering is most applicable to our project since we are getting mostly assembled boards that only need the appropriate pin headers to be attached.

We are transmitting data over wi-fi and USB. Each of these transmission methods has standards with regards to how to encode data and how to interface the devices at each end of the transmission. For wi-fi, it is IEEE 802.11. This is a standard defined by IEEE that defines how a wi-fi signal should transmit, collision control, and what channels the device can use in the radio wave spectrum. IEEE 802.11b defines the maximum raw data rate at 11 Megabits per second on the 2.5 GHz channel though this is later updated in 802.11g to 24 Megabits per second with backward compatibility for the previous standard. This standard helps us define the maximum data transfer between the camera and laptop and also the motors and remote. Standards following 802.11g all define how to use the wi-fi spectrum for wi-fi 2 and up which allows us to use the 5GHz spectrum and gives us more available channels to use for our signal. All purchasable products are IEEE 802.11 compliant, so we won't need to build an 802.11 compliant device, but these standards are helpful in determining what specs to look for in purchasable devices and how to properly use the services provided when we code the control boards to transmit and receive over the wi-fi signal network. [23]

IEEE 802.3 is the meta standard for ethernet devices and communication. Similar to IEEE 802.11, this standard has several standards that define the communication protocol. We specifically want to look at IEEE 802.3af and 802.3at which pertain specifically to PoE cables. This standard says that PoE cables transmit 44 volts at 350 milliamps [4] [5]. This means that the voltage would need to be wound down for simple 5-volt devices. Also, it defines that only sufficient power should be delivered. So, all 802.3 compliant devices have a protocol for turning on for sufficient voltage.

The release of USB 2.0 in the year 2000 signified the increase of the data rate to 53 Mbyte/sec, allowing a current of 1.5 A while performing high-bandwidth, high-speed communication, or a more power-heavy load of 5 A while sacrificing data transfer speed. The USB 2.0 Micro connector supports a similar pin-out scaled down to fit a smaller, differently shaped port. Power always flows from the host to the secondary device. These specifications are standard through all USB designs so we can be expected similar performance from all versions of USB connectors. These can help us also determine the constraints of using USB connections. [23]

One constraint in regard to USB connectors is the length of the cable. USB cables are known to drop signal over distances longer than 5 meters (16.4 feet). Therefore, our design can't use a single long USB cable to provide power. This constraint forces us to use a repeater, an amplifier, or a different cable because we want our system to stand on a 25-foot tripod. When designing the system, we must be mindful of the limitations of the different connection technologies and use alternate options when necessary, such as a standard AC power converter with a barrel plug. [16]

There are constraints with regard to power and current that applies to the technologies that we have bought for the project. Each motor has an operating voltage that determines the minimum required voltage for a motor to achieve full speed, so the power supply for the control board needs to be able to power the motors as well as the boards. Each board needs its own power too, so whichever control boards we use need to provide the proper power and current distributions to each part of the system supplied by a single power source.

We also have a constraint on how the power supply should be connected. The main power is routed through the PWM board and then driven to the main controller through the 5-volt and ground GPIO pins. We need to make sure that the amount of voltage through the port does not go too high to prevent overloading and damaging the board. 5-volt compliance is a standard that says that a higher voltage power supply will be regulated down to 5 volts. Most of the control boards available are designed either this standard in mind through the main power supply port but driving power through the GPIO port will bypass this regulator and present the possibility of burning the board. We should either make a regulator or keep the voltage in mind when connecting to the 5-volt rail.

Another constraint that we have to be aware of is the size. We want to use 3D printing to make the housing unit for our system. This means that we can design it how we want to, but we are limited in size because of the size of the printer. 3D printers are constrained to print objects within the size of the print platform. Larger prints must be broken into pieces and assembled outside of the printer. We would like to make a single solid print of all our parts to maintain the integrity of the weatherproofing, so we must constrain our design to the size of the printing platform.

One of the biggest constraints we have is the budget. We want to design a system that can compete on the market with other similar devices. This means that we need to stay in the same price range or below. Most competitors range between \$300 and \$1500. A majority of the budget goes into a high-quality camera and precise motors. Similarly, we can expect our budget to inflate in the same areas. Therefore, we need to be mindful of these parts and try to drive the cost down for the purpose of a commercial sports camera.

Marketing Requirements:

The client's assumed intent is to use the device to perform videography on high-speed sports events that require a bird's eye view of a wide area that can be covered quickly by the action camera system. We intend to deliver this product as an easily controllable bundled package that relies on commonly available computer hardware. Including prebuilt options like cameras and controllers defers the research costs to other companies, ensures

ease of part replacement in the case of failure, and reinforces the likelihood that users will be familiar with the parts and operation of the device.

Impacts

The impact of cost will filter the users of the device to include those willing to invest more than those satisfied with taking snapshots from the ground. The benefits of the elevated device must outweigh the extra cost of developing the tripod-based system. This is the basis for competition with those ground-based video capture products. Alternatively, there are other tripod-type camera systems available on the market, but they are often homebrew-style devices without tight integration of the parts. Ideally, this device's quality choice of parts with pre-planned integration will increase the competitiveness, product viability, and competitiveness of our project.

This will make such a camera model available as a bundled package without the need to implement it from scratch. As such, communities, schools, sporting venues, or otherwise bird's eye videography will become more available to the public if drone video shooting is undesirable but shooting from the ground is not sufficient to cover the area being filmed.

Safety

The tripod camera system, when used as intended, does not cause harm to the user or to individuals nearby during operation. The electronic interference is negligible and does not take advantage of large-scale radio transmission technology. Only the proper physical handling of the device is required in order to establish safe conduct.

Steps should be taken to ensure that in high wind conditions, the tripod is fixed securely enough that it does not topple over, which could prove an impact hazard to people or property. A brief safety pamphlet noting the potential dangers of handling electricity would cover the bases for any power sources involved in setting up the device.

Manufacturability and Sustainability

The camera, the tripod, the controller, the workstation, and the connecting hardware are all standardized and widely available. The software written for the project can naturally be duplicated freely by the manufacturer. Therefore, the project ecosystem as a whole is supported by a highly manufacturable industry and faces no general threat of shortage.

There is a chip shortage as of 2021, but this is less likely to threaten access to embedded parts. It is likely that after this shortage passes, parts for regular maintenance or repairs will continue to be available to the general public.

Given that there are not many moving parts beyond the pan and tilt motors, the product should overall not need replacements until the lifetime of the individual components passes. If necessary, new motors could be sold as separate parts for the user to perform maintenance. Alternatively, a system where damaged parts are shipped back to the producer, repaired, and returned to the customer could also be implemented, allowing the manufacturer to determine the cause of the defect in order to provide insights into improvements on the device.

If there are updates to the software or otherwise security vulnerabilities discovered in the product after publication and distribution, the software maintained by our team runs on a typical desktop-style workstation, so updates could be pushed over the internet if necessary.

Requirements to Use

Users will be familiar with the technology involved if they have general experience with cameras, ordinary computer use, and possibly a background in using video game console controllers. The most important background is in the use of the controller device as this is the most specialized. In order to capture video with an action camera on a fast-paced sports game, a certain minimum level of fast reflexes is required. Therefore, the operator should be alert and dexterous enough in order to capture these events as they occur.

For those users that have prior experience with video game console controllers, they are likely to be familiar with the PS4 controller that is used to operate the camera controls; this will ease the learning process for the device.

House of Quality

The house of quality, in Table 1, illustrates the tradeoffs between the marketing and engineering requirements. By comparing the tradeoffs between the two requirements we are able find what can be accomplished conceptually before actually physically building the project.

The marketing requirements are defined as the needs and wants of the product in the point of view of the consumer. For example, the consumer of our PTZ camera would like a camera that is low in cost, easy to use, enabled with a reliable user interface, and high in both audio and video quality.

The engineering requirements on the other hand consist of the needs of the system. Engineering requirements unlike the marketing requirements are in the perspective of the engineer instead of the consumer. The less the system weighs the more portable and easier to use the device would be. Another interesting requirement is the amount of power to be used, as the total power goes down the cost will also, however the quality of audio and video will also see a decrease.

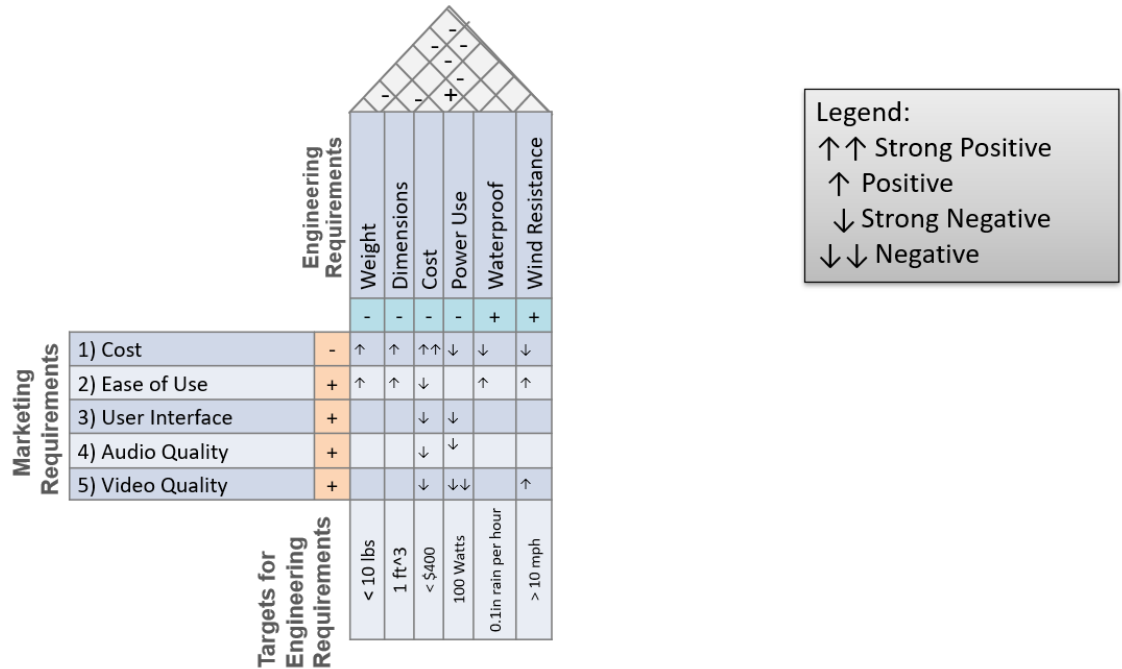


Table 1: House of Quality Diagram

Technology Investigation

Existing Products:

During initial product and project research we were able to find a few PTZ cameras like our own in the sports and security industries and one hobbyist project open sourced to the internet. In sports PTZ cameras usually consist of a 4k camera inside weather-proofed housing that is attached to the top of a tripod mount. The security industry takes a slightly different approach as most of their cameras are powered using PoE and don't move nearly as fast as a sports PTZ if they can move at all. The hobbyist was able to mount a DSLR Canon EOS 250D to a self-designed and 3D printed mount controlled by a controller and moved by stepper motors. Our camera's design takes advantage of 3D printing and available electronic gadgets while also implementing a solution that can keep up with the panning and tilting speed of a sports camera.

Sports PTZ

Panasonic's Professional PTZ Cameras are a line of PTZ camera that are remotely used to handle the quick and rugged sports filming job. The AW-UE150 4k 60p Professional PTZ Camera is one of Panasonic's top high-end cameras. Shown in Figure 1 in white and black variants this camera comes in supporting SRT Protocol for 4k 60p video streaming, up to 20x optical zoom and image stabilization. The Panasonic PTZ camera also supports simultaneous video output with various 4k interfaces as output such as 12G-SDI, HDMI, Optical Fiber, and IP. The biggest drawback for this camera is the hefty MSRP at \$11,400. Our goal is to source our own project parts while also costing less than 5% of the price while still being able to stream high-quality video.



Figure 1: IP camera powered by over ethernet.
CC0 1.0 Universal (CC0 1.0). Public Domain Dedication.

Security PTZ

Shown in Figure 2, Security PTZ are security cameras mounted to houses, businesses, and anywhere that requires monitoring that can Pan, Tilt, and Zoom usually at a slow pace. These types of cameras are made to keep reliable non-stop surveillance, have human motion detection abilities, and night vision. They can interact with applications to remotely notify the user when motion is detected or alert of noise if applicable. Security PTZ differ from other PTZ as they are almost always connected to a closed network or CCTV by power over ethernet and aren't made to track fast moving objects or shoot very high-quality video depending on their price.



Figure 2: Reolink PTZ Camera

As pictured in Figure 2, The Reolink PTZ Outdoor camera is a cheaper security camera that can shoot and record surveillance footage at 5 mega pixels in 2560x1920 Super HD. The camera features a 360-degree pan, 90-degree tilt, and 4x optical zoom. It has complete

weatherproofing and since it will be mounted to a wall using one wire cabling. The PTZ camera is also able to alert the user as soon as any motion is detected and has the ability for the user to set the range, they want the camera to detect. The biggest drawback of this type of camera is its inability to work with non-native software and its original design not being geared to catch action sports.

Open-Source PTZ

During our research we found an open-sourced pan-tilt-zoom mount on YouTube that utilized 3D printing, printed circuit boards, a microcontroller, and gaming controller to assemble a self-made PTZ camera. The microcontroller used was an Arduino Nano which handled the control of everything and the monitoring of inputs. For panning and tilting two Nima 17 stepper motors were used which were controlled by the A4988 stepper driver board. Another insightful function of the camera was the edition of embedded sensors that reside in the pan and tilt axis that would reset the stepper motors to its original position when the device was powered on. This project provided a lot of insight but differs from the two of our main objectives as it isn't as cost-effective, and it doesn't have any type of weather resistance.

PTZ Camera	Price	Type	PoE Enabled	Weather Resistant
AW-UE150 4k 60p Professional	\$11,400	Sport	Yes	Yes
Reolink PTZ Camera	\$229	Security	Yes	Yes
Open-Sourced PTZ	\$350+	General Purpose	No	No
Our PTZ	>\$300	Sport / General Purpose	No	Yes

Table 2: A comparison of similar products to our project

Cameras:

PTZ PoE Security Cameras:

The first consideration for the camera we analyzed is a straightforward power over Ethernet pan-tilt-zoom (PoE, PTZ) camera, typically used in settings for security. These devices typically feature remote controls over TCP/IP on the HTTP protocol. A web interface is offered pre-built by connecting over HTTP on port 80.

The motor is also included in the system, meaning that the system is capable of covering the full range of motion for capturing action events. With respect to the power-over-Ethernet power supply, the IEEE802.3af Ethernet standard can support powered devices that do not draw more than 15.4 watts [5].

One consideration on the motor is that these cameras are primarily designed to be used as wall-mounted security cameras. This is a hint at the fact that they often do not provide audio out of the box, and if they do, it is often not high fidelity enough to qualify it for selection. These cameras also may not pan rapidly enough to qualify as an action camera.

Therefore, if this is meant to be a viable option, our own audio subsystem would have to be implemented on top of this.

Highlighted in Figure 3, finally, since the prebuilt turret-style security camera almost universally supports features that are not relevant to our project, such as wireless imaging, remote monitoring over a web interface, SD cards that are built into the camera, noise-emitting self-contained alarm systems, and so forth, we pay extra for features in a product that we will simply not be using. The prices for example security cameras running over PoE, with the required features, but also with the consequential defects listed above are compiled in a table below.



Figure 3: Simple turret-style PoE security camera.
Computer model licensed under CC SA.

Camera	Resolution	Framerate	Price	Weight
Amcrest	3 Mpixel	30fps	\$109.99	1.4 lb
Montavue	4k	30fps	\$219.95	2.0 lb
Hikvision	3 Mpixel	30fps	\$191.15	1.5 lb

Table 3: PTZ camera comparison

<https://www.electronics-notes.com/articles/connectivity/ethernet-ieee-802-3/standards.php>

DSLR:

DSLR cameras have a high-quality lens and an enhanced capability for digital zoom that allows for a clear picture out of the box. Supporting advanced features correcting for brightness in overcast weather and image auto-stabilization in windy conditions, there are clear benefits for the DSLR camera.

One particular advantage of the DSLR camera is the support for many types of electronic and mounting accessories. Most relevant to our use case is accessory audio systems. Although DSLR cameras typically contain a small pinhole port for audio sampling, these assume the sample is close-by and that the source has limited noise interfering with the signal. There are many consumer attachments for DSLRs that allow for superior audio sampling over the stock pinhole models that help in canceling interference from wind.



Figure 4: DSLR-style camera with attached microphone.

As shown in Figure 4, the ideal DSLR camera for taking video is the mirrorless camera. These are more expensive than the DSLR camera, but the advantage in the auto-focusing system is clear. In DSLR cameras with a mirror, the mirror has to be shifted away in order to unblock the auto-focusing element.

However, DSLR cameras that are waterproof are less common than in action cameras. A DSLR camera meeting the waterproofing requirements, that is mirrorless, and that falls within the constraints of the project budget narrows the range of choice in cameras significantly.

The ultimate problem of the DSLR mirrorless camera is the cost. Taking the example of the Canon EOS M2, this is a camera at the low end of the price spectrum. An older camera from the year 2013, we find a reasonable price at about \$200 used. The requirements for quality are satisfied, and we are able to take audio and video at 30 frames per second on 1080p; however, the camera itself is very heavy due to the size of the frame and battery, and most importantly the requirements for remote control are not satisfied. The part's availability is low because they can only be bought used; furthermore, they are no longer in production and have been superseded by more expensive, newer models.

On the other end of the spectrum, we examine the second element of the case study: the Fujifilm X-T30. This camera does tick all the boxes for the engineering requirements. With 4k video at 30 frames per second, a lightweight frame and battery clocking in at under one pound, and high-quality digital zoom, the point of failure is the mirror image of the Canon EOS. This greatly exceeds the budget to satisfy the marketing requirements, costing over \$800.

The obvious solution might be to find a middle of the road camera that lies just above the lower cost mirrorless DSLR but has the features required by the project. However, I could

not find many like this that were not used goods in the same manner as the lower bounds camera.

In conclusion, the DSLR camera suffers from lack of parts availability or extreme price when considering the overall budget of the project.

Sports Action Cameras:



Figure 5: GoPro HERO Action Camera.

The advantages of the premade action camera are clear. Figure 5 shows the outline. The device is waterproof out of the box, the video capabilities support 60 frames per second or 30 frames per second in regard to the framerate requirements, and 1080p or 720p resolution is standard. This is the most popular consumer-grade action camera, so the support for attachments, mounting setups, and documentation is widely available [2]. The downside to using a flagship camera in the center of the action camera spotlight is the price.

The action camera family also supports external microphones over 3.5mm jacks. This would increase the sound quality and reduce interference in audio sampling, but cursory research indicates attaching an external microphone requires removing or altering the waterproof housing of the device. If the waterproofing is compromised, one of the major benefits of the action camera would be liquidated.



Figure 6: GoPro with external microphone and wind muff.
Reproduced under CC BY-NC.

Displayed in Figure 6, the Sony FDR X3000R is an alternative action camera to the GoPro with comparable features. Water resistance and video modes are directly comparable at a reduced price. The battery life on the Sony FDR is not as high-powered as on the GoPro or

some of its competing alternatives, but this is mitigated by the attached charging cable in our configuration.



Figure 7: Sony FDR X3000R Action Camera.
Reproduced under CC BY-NC.

Above is the general form of the Sony FDR X3000R in Figure 7 as described.

Camera	Housing	Resolution	Framerate (frames/sec)
Sony FDR X3000R:	Plastic	4k	30
Garmin VIRB Ultra	Plastic	4k	30
AKASO V50 Pro Action Camera	Plastic	1080p	60fps
Yi 4k +	Plastic	4k	30

Table 4: Comparison of camera housing, resolution, and frame-rate Source: amazon.com

Audio Systems:

Many premade consumer microphones are available with a variety of connectors to support our application: from the camera’s fixed location, record sound at the target location of the video recording. This has two immediate implications: first; the microphone would be a cardioid-style microphone, that is, it will pick up sound in a monodirectional fashion. Second, the microphone mount will most likely be mounted on or nearby the camera in order to passively take advantage of the panning and tilting to determine the direction that sound is read from.

Microphones have many different connection standards. For commercial applications and high-fidelity audio, the professional XLR connector is used. The XLR connector is popular because it is capable of providing current approximately up to 10 amps at 24 volts; at over 200 Watts, this more than doubles our requirements for the ceiling on our power system design. This is because the XLR connector can double for stage lighting and other cinematography applications that exceed the needs of our camera system.

On the other hand, the connector is not used routinely on single-board computers or embedded devices like the Arduino. cursory research shows little results for XLR connectors on embedded devices including the Arduino platform, the TI MSP systems, the Adafruit platform, and the Beaglebone platform. Additionally, requiring that an XLR device be used would require additional hardware, either XLR to USB or XLR to 3.5mm barrel jack in order to attach them.

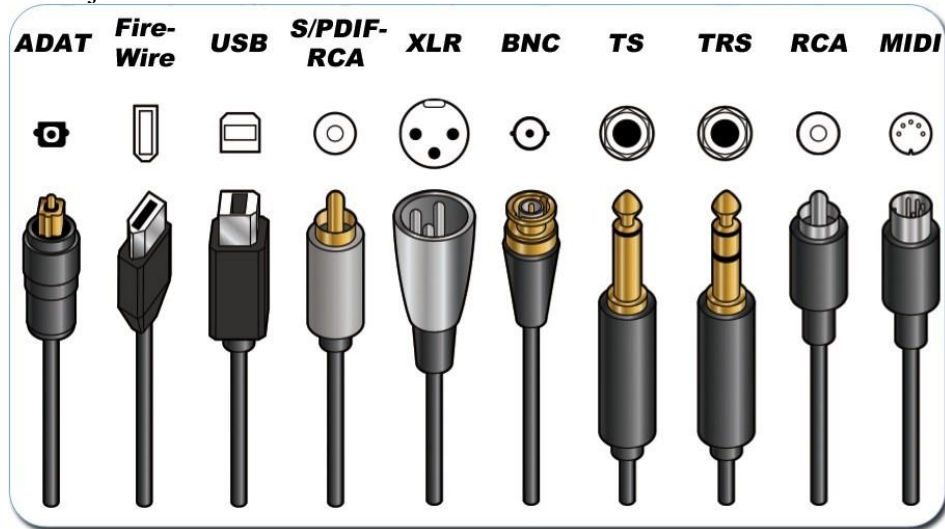


Figure 8: Many type of audio connectors.
Reproduced under public domain.



Figure 8: Microphone XLR connector to (a) USB; (b) 3.5mm jack.
Reproduced under CC BY-NC.

Figures 7 and 8 show the situation. One solution to the problem of connectors is to link an XLR microphone directly to the workstation hardware. This would open possibilities regarding types of using higher fidelity microphones as the audio pickup source on the action camera [3]. The issue is that the increased cost of the additional hardware, further alienation of less technically competent users who may not be comfortable with adding custom hardware to their workstation, or even requiring certain types of motherboards which have additional slots necessary to accept accessory hardware; although advantageous, these factors do not necessarily justify the complications introduced by such a card. Therefore, a solution outside of the XLR direct connection is necessary.



Figure 9: PCI Microphone connector card that directly supports XLR.
Reproduced under CC BY-NC.

Figure 9 highlights the problem. Both cases would defeat the power requirements or add even more additional hardware in order to connect to a PC. Since this is not widely supported either on PC or on embedded devices, most likely the project will not employ this connection standard even though it is very widespread in the microphone world.

Taking into the considerations into the study on XLR and observing that connection standard without the need for additional hardware would be highly beneficial to the design of the microphone subsystem, we defer to the reference figure to provide the most natural answer of USB (Universal Serial Bus). Note the universal keyword which will be far more flexible than the more obscure XLR-style connector.



Figure 10: Monodirectional USB microphone and typical mounting system.
Reproduced under public domain.

Figure 10 shows one form factor. In contrast to the previous XLR microphones, which threatened to introduce an entirely new ecosystem of software and hardware due to connector incompatibility, the USB interface is widely available, comes in multiple size form factors, including miniaturized versions, and can draw a reasonable amount of power

(5 watts at minimum from computer USB source ports) directly from the connected workstation.

Alternatively, many embedded computers such as the Arduino or Raspberry Pi we intend to use can also be purchased with either full USB or micro-USB support. These generally max out at 5W, but fortunately many of the consumer-class USB microphones use less than 5W of power for operations.

Figure 11 demonstrates the solution to noise. The altitude of the microphone outdoors requires a system to cancel noise introduced by the wind. Our requirements specify the wind-speed conditions the device must be able to operate under with minimal interference; in moderate wind conditions outdoors, a wind muff will be sufficient to suppress noise adequately.



Figure 11: A microphone (using the XLR connector) with a fluffy wind-muff. Reproduced under public domain.

Motors:

The camera motion rig system we aim to create requires a motor system in order to perform the pan and tilt functions. 2 motors are required. One will point down into the base to rotate the camera rig and perform the pan function. The other will point into the camera and connect to the bracket in order to perform the tilt function. Both motors will connect to the electronic speed control unit in order to control the speed and direction of the pan or tilt and the full range of each function. As such, the motor system is dependent on the electronic speed control as a constraint on its power and current consumption. The motors should also rotate at a sufficient speed to capture athletic activities at a range of 25 feet up. The motors should also allow the camera to tilt from 0 to 180 degrees and pan from 0 to 180 degrees as well.

Brushed Motors:

Brushed motors use an electromagnetic design to drive the axel. A wire is coiled around an armature to create the electromagnet and the armature encases the commutator which houses the axel. The brushes are connected to the commutator and drive the current to the electromagnet. They control the direction of the current and thereby control the polarity of the magnet in the rotor so that they oppose or align with the permanent magnets outside of the rotor. These motors have two wires that deliver voltage, one for voltage and one for ground. This also means that it is easy to change the speed and direction by changing the voltage strength or direction, so in some cases a speed controller is not necessary. Brushed

motors are also cheap because the materials are cheaper and don't require expensive sensors. However, brushed motors also have shorter life spans because of the friction between the axel and brushes. The friction also makes it harder to rotate from rest. [9] [8]



Figure 12: A brushed motor

The Micro Gearmotor is a brushed motor. An example form factor is given in Figure 12. The Micro Gearmotor model requires a minimum of 6-volts and has a maximum current draw of 1.6 amperes. Depending on the model, they can deliver 90 to 175 rotations per minute (rpm). The high operating voltage is not ideal since most boards support 5-volt power supplies. However, the small motor size is good for a compact design and the eternal gears make it easy to interface with gears on the frame that the camera sits on.

The Hobby motor is a brushed motor with a 1-volt minimum voltage and a 0.8 ampere maximum current draw. It can deliver 6600 rpm. This motor is by far the cheapest at a price of 2 dollars. This also makes it cheap to buy replacement parts since brushed motors have a shorter lifespan compared to other motors. We also get the lowest power consumption at a maximum of 2.4 watts at 3 volts from this motor which is ideal for a board that supports a 5-volt power supply.

The P28UK12 Piborg motor is a brushed motor with a 12-volt operating voltage and a 0.5 ampere maximum current draw. This motor can deliver 590 rpm at 12 volts. This motor is part of the P28UK12 series which comes in a variety of sizes so we can choose the appropriate size without changing the specifications of the motor. However, the 12-volt operating voltage makes this the most demanding brushless motor. It is saved by the low current consumption from being the most power-hungry motor.

Brushless Motors:

A brushless motor is a type of motor that has permanent magnets around the motor and no brushes. In place of brushes, they have control circuitry to detect the position of the rotor, hence the third pin. Brushless motors don't have the burden of friction between the brushes and the axel, so they last longer and are more efficient. The use of a position sensor also makes them more accurate, so they have greater control for functions such as braking. They also have more power, so they spin faster too. However, because of the sensor and the

parts, these motors tend to be more expensive. Speed controllers for brushless motors tend to be more expensive too, especially if they require a PWM. [8] [9] [10]



Figure 13: A Brushless motor

The Exceed RC Rocket 3015 is a brushless motor (example shown in Figure 13) with a 12-volt minimum voltage and a 35-ampere maximum current rating. It provides 10,080 rpm. This motor is meant for remote control cars, so it is designed to take battery power. The high rpm is a little unnecessary for this project since the motor only needs to rotate the camera mount 180 degrees. It is also one of the more power-hungry motors that may not have proper support from a microcontroller as it would from a 12-volt battery supply.

The Exceed RC Rocket Airplane 86MB88 is a brushless motor similar to the Rocket 3015, but it is designed for remote control airplanes. It has a 12-volt operating voltage and a 32-ampere maximum current rating. It delivers a 9000 rpm. It is also suited to battery power. This model is a slightly less power-hungry motor but has all the same issues of the other Exceed motors.

The Exceed RC Optima 75M92 is a brushless motor meant for use in remote control airplanes. It has an 11-volt minimum voltage and takes a maximum current of 27 amperes. It delivers 47703 rpm. This is another high-power option that is suited to battery power and share similar issues with the others of this model. This is however, the most power hungry of the set at 330 watts.

The AEO Micro Gearbox is a brushless motor system with a gear assembly attached to the axel. It has a 7-volt minimum voltage and a 4.8-ampere maximum current rating. It can deliver at least 49140 rpm. This option has the three wires needed for a brushless speed controller, but no barrel connections. This motor would require either a barrel fitting for the end of the wires or a control board that uses screw terminal ports so we can use the bare wires. Also, since it has the gear assembly, it makes it easier to use. Consuming only 40 watts, this is most power efficient brushless motor. However, the 7-volt operating voltage still may be too much for the simpler control boards we are looking at for our project.

Servos:

Servos are similar to brushless motors in that they have 3 pins, but the control on a servo is much greater control. Servos are different from regular DC motors because they use pulse width modulation (PWM), so they require a more precise control board than regular motors. The PWM signal allows you to rotate the servo with more precise control by telling it what position to move to instead of controlling the speed of the rotation. This comes at the cost of freely rotating. [7]

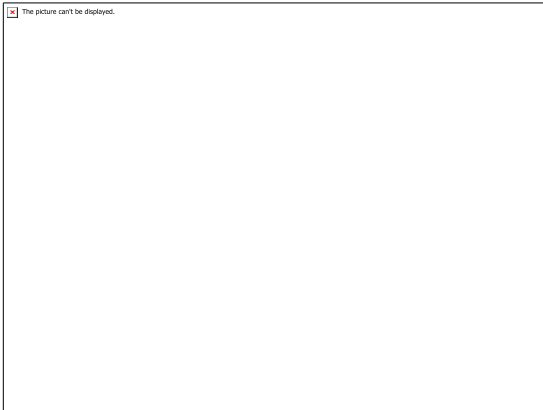


Figure 14: A micro servo

As in Figure 14, The Adafruit micro servo SG92R is a more controlled option with a range of 180 degrees of motion. It has a 3-volt minimum voltage requirement. To control the speed and position of the micro servo, the control board would need a PWM device. Micro servos are small, so they are easy to design a mount around. While there is no free rotation, the requirements of the pan and tilt functions don't require full 360 rotation.

The Hitec HS-422 is a full-sized servo motor. It has a minimum operating voltage of 4.8 volts and a maximum current draw of 0.8 amperes. Compared to the micro servo, this one is slightly slower but otherwise similar. It offers the same range and precision of motion for slightly more power consumption. While larger, the servo size is still easy to design around and fits in most pre-made mounts.

Comparison:

The best option is the servo motors, though they are closely followed by brushed motors. Brushless motors seem like a good option because they are highly efficient and long lasting, which are desirable in a product meant for long term use. However, these motors also require too much energy. They are also meant to be used with 12-volt batteries on remote control vehicles and consume a lot of power to work efficiently. Servos require significantly less power and have the accuracy of brushless motors because of the position sensor. They also have high torque which makes them easy to move from rest. Brushed motors are also a good low power alternative, just with less control precision because it has no position sensor. All options chosen were in a similar price range, so cost is not a determining factor in the decision.



Figure 15: HS-422 servo

Examine Figure 15 and compare it to the alternative. Between the micro servo and the HS-422 servo, the HS-422 is the best option. While the micro servo is slightly faster and cheaper it may also have less durability. The HS-422 is a full-sized servo that does take more voltage, but the difference is not significant. Also, it is more resistant to being burned-out by voltage fluctuations.

Motor	Price	Type	Voltage	Power (watts)	Current (amps)	RPM
Micro Gearmotor	12.95	brushed	6 -12	*0.42 – 19.2	0.07 – 1.6	175
Micro Gearmotor	12.95	brushed	6 - 12	*0.42 – 19.2	0.07 – 1.6	90
Hobby motor	1.95	brushed	1 – 3 (12 max)	*0.11 – 2.4	0.11 – 0.8	6600
P28UK12 Piborg motor	19.46	brushed	12	*5.964	0.497	590
Exceed RC rocket 3015	10.16	brushless	12 - 14.8	300	1.2-35	*10,080
Exceed RC rocket airplane 86MB88	11.86	brushless	12 - 14.8	280	0.6 - 32	*9000
Exceed RC optima 75M92	10.95	brushless	~11.1V	330	2.3 – 27.5	47730
AEO micro-gearbox	21.95	brushless	7.4 – 8.4	40	4.1 - 4.8	49140
Adafruit Micro Servos	5.95	servos	3 - 6			*100
Hitec HS-422 Servo Motor	14.99	servos	4.8 – 6	*0.038 – 4.8	0.008 - 0.8	*62.5

Table 5: Comparison of motor options. The * means that the value was calculated. The two formulas used were $V = IR$ and $K_v = \text{RPM/Volt}$.

Electronic Speed Control:

The electronic speed control unit is part of the motor system and controls the speed and direction of the motors. While part of the same system as the main control board and motors, the electronic speed control is a separate component that can come from different manufacturers. This component takes input from the main control board and interpret it to drive voltage, and in some cases signal input, to the appropriate motor. These control units have a current rating that determines the maximum allowable current through the component. This would dictate the maximum current from the main control board and the motors as a constraint to prevent the board from being overloaded. Some control units have pulse width modulators to interpret the signal width and pass position information to brushless motors or servos. The speed control unit also has to be small enough to fit on the main control board.

RC speed controllers:

These small basic controllers are best suited for brushed motors on RC vehicles. They have the basic function of controlling the voltage delivered to the motors to control the speed of rotation. Some controllers can be advanced with integrated wireless communication, but any control board can have added functionality by connecting to another board through the connection pins. Figure 16 shows the speed controller.



Figure 16: The Sparkfun Qwiic speed controller

The Readytosky RC motor controller is made specifically for remote control vehicles. It uses a 2-pin design for the motor interface, so it is best suited for controlling brushed motors. The board also has a 2-pin battery interface to get voltage and ground from the main board. It requires a minimum of 3 volts and has a maximum current rating of 20 amperes. The board also has a receiver integrated for wireless control so only one wire for power is needed. The product listing sells the controller in pairs, so the cost is about half that of other options.

The Sparkfun Qwiic has 4 pin connections for the motors and 2 pin connections to get power from the main board. This means it's best suited for a pair of brushed motors, though you could adapt the pins for a brushless motor or servo and use 2 boards instead of one. However, this work-around would require getting a precise clock signal from the main control board to emulate a pulse width modulator for servos. The board also has screw port connections, so you only need to plug in the bare wire. No need to match up special connectors. It has a minimum requirement of 3 volts and a maximum current rating of 1.5 amperes. The 3-volt operating voltage requires a logic converter to interface with a 5-volt logic board. (control through I2C channels)

PWMs:

Pulse width modulators (PWMs) are needed for motors with a signal pin such as brushless motors or servos. These devices control the frequency of the signal and the motor interprets this information to control the speed or position. The 2 PWMs discussed are the Toshiba TB6612FNG H-Bridge and the PCA9685. Both are reasonable to use and do the same job, but how the boards use them is the main deciding factor. For instance, these boards take an external 5-volt power source to drive the motors in addition to the 3.3 volts the board needs to work, most use a typical barrel plug or terminator block, but some draw the power through I2C connectors or the GPIO pins. We want to minimize the number of input wires going to the device so we want a board that can support a single power source and has low power consumption. Also, some of these control boards can support extra functions with extra pins. This may be unnecessary for our needs and thus would just make our design more cumbersome.

Toshiba PWM:

The Sparkfun Wireless Motor Driver supports 2 and 3 pin connections, so it can be used with any motor option. The Toshiba TB6612FNG H-Bridge can support 2 brushless DC motors at the same time and the GPIO pins are able to support 3-pin devices like brushless motors or servos. The board has a barrel plug to deliver power to the motors and can deliver power to the main board through the GPIO pin connections. If the motors have low power demands, then the driver can draw power from the main board through the VIN pin connection. The controller doesn't support wireless communication, so it needs to go through the logic board with transmission capabilities or use another product such as an RC transceiver. It is also best suited for use with Arduino and Sparkfun products. Examine the form for more detail in Figure 17.



Figure 17: The Ultraborg servo driver

PIC16F1828 PWM:

The Ultraborg PWM servo control board requires a 5-volt power source separate from the main logic board. However, it can also interface with any kind of board: Pi, Arduino, or other I2C devices. This is the most minimal board out of all the options with support for 4 motors and 4 I2C connections. The minimal design also means that it does not carry any support for a communication device. Any wireless communication would have to be supplied through the main board and communicated through the I2C connection.

PCA9685 PWM:

The Sparkfun Servo pHAT is a controller meant for 3-pin motor control. It has a minimum operating voltage of 2.3 volts. This HAT is made to interface with the GPIO pins on Raspberry Pi logic boards especially the Pi Zero, but it also has I2C ports to use with any brand of logic board. It's USB-C port is used to drive power to both motors and the logic board so only one power line is needed. This board also has power protection so it can protect power sources from damage.

The PCA9685 Servo PWM for Pi Zero is a control board made to control 3-pin motors. It can easily attach to the GPIO pins on Pi devices. It has a 2.3-volt minimum operating voltage and can take a maximum of 0.4 amperes. Unfortunately, the board requires a separate power supply with a common ground. Only low current power can be driven through the GPIO pins. It also only works with analog RC servos, so it limits the options of motors we can use. The PCA9685 is the PWM.

The Adafruit series of PCA9685 boards are similar to the Pi servo PWM and are made to work well with Pi devices. All devices in this series can support up to 16 motors and have 3.3-volt minimum operating voltages and 0.4 ampere maximum current ratings. There are a range of sizes thought which allows different capabilities. The smallest is the Adafruit bonnet which is made for use with the Pi Zero and has the choice of a barrel plug or screw port to supply power. This is also the cheapest option from all the speed controllers. The Adafruit PWM servo driver has a similar size and capability but only provides the connection for a screw port power supply. Then there is the Adafruit servo mini kit. This is a larger board meant for Pi 2 and up but carries a lot of extra pins that would be overkill for this project. The best option of this series would be to use the bonnet because it has the cheapest price and the minimum necessary capabilities to power the motors for this project.

Comparison:

Using a PWM control board seems to be the best option. Since servo motors are the best option from the previous section, we need a controller that supports the 3-pin connection and PWM for the motor control. We want low voltage with large enough current to avoid burning out. Most of the PWM control board options are 5-volt compliant. This means that they support servos that need 5 volts to operate and supply 3.3 volts to the board. Since all the PWM options share this specification, it cannot be a determining factor. We also want a design that is minimal but has enough flexibility to support other devices such as a wireless transmitter. The Ultraborg is the most minimal of the options, but it is also one of the more costly options. The Adafruit servo bonnet is the cheapest option and is minimal in design. It also offers flexibility to work with Raspberry Pi zero and up. The Sparkfun servo pHAT is also comparable in price but has the flexibility to use I2C connections to power the main board. Alternatively, we have the Sparkfun wireless driver, which is the most expensive option, but it has an easy-to-use design. The Sparkfun driver is made to take a wireless communicator and power the main control board through the GPIO pins. However, it can only work with the Sparkfun control unit. The best option would be to go with the Sparkfun servo pHAT because of its flexibility of connections and its minimal design. The wireless communication can be handled by the main control board, possible a PI Zero W which has built-in Wi-Fi capabilities. We see the form in Figure 18 below.

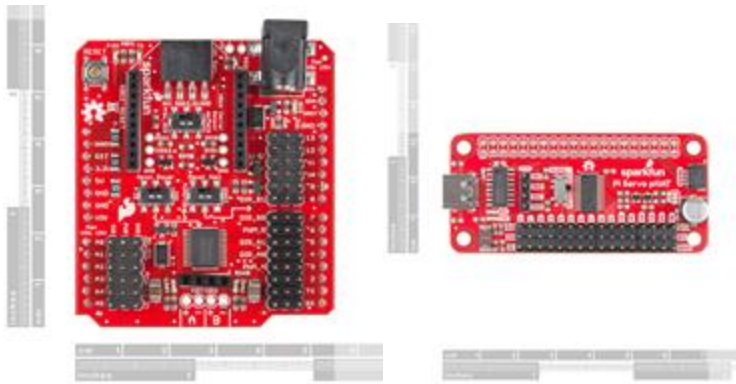


Figure 18: (a)Sparkfun motor driver, (b) Sparkfun servo pHAT

control unit	cost	voltage	current (amps)	note
Readytosky RC motor control	14.99	3 – 9.4	20	2 pieces, meant for RC brush motors
Sparkfun Qwiic	14.95	3.3 – 5 (11 max)	1.2 -1.5	1 board supports 2 motors
Sparkfun wireless motor driver	22.95	0.2 – 6	1.2 – 3.2	2 and 3 pin support
Sparkfun servo pHAT	10.95	2.3 – 5.5	0.025	power through RVR or usb
Ultraborg PWM servo control	18.39	5		minimal design
Servo PWM Pi Zero	15.04	2.3 - 5.5	0.025 – 0.4	
Adafruit PCA 9685 servo driver	14.95	2.3 – 5.5	0.025	
Adafruit PWM/Servo bonnet	9.95	3.3 - 5	0.025 1.3 back power	good for Pi zero
Adafruit 16-channel PWM/servo HAT for Pi	17.50	5 – 6	1.3 (back power)	can support multiple servos/motors

Table 6: Comparison of motor control options

Mount/Frame:

The frame is the part of the system supporting all of the electronic components. This includes the mount holding the camera, the gears connecting the tilt motor to camera bracket, and the mount connecting the main system to the tripod mount. Most of these parts can be 3D printed in ABS plastic for a custom fit, but the main frame supporting the camera can be made of metal if it is bought. The frame is constrained by the size of the system. All the components need to fit within the frame and the camera need to be able to fully articulate. In order to get the full range of the tilt, the camera needs to be high enough to see almost directly beneath it.

Bought option:

Metal parts, more durable. Easy to find replacement parts for mass manufacture. But have to fit design around availability. Parts come in standard sizes. Expensive. Metal corrosion in poor weather depends on material options. All options come in kit versions, bundle with servos. Shared design

The Steering Gear mount is an aluminum mount meant for remote control boats and cars, but the design is generic enough to be adapted to our purposes. The aluminum material is good for durability and weather proofing. We can also find this mount design in the Mallofusa 2 DOF kit where the mount is bundled with 2 MG995 servos. By bundling these 3 items together, we can reduce the cost slightly. The mount is made of a C frame and servo platform with the panning servo serving as the base. The minimal design leaves the servos exposed and we also have to design a platform for the panning servo, so it sits on top of the control board without getting in the way of the electronics. The design is visible in Figure 19.



Figure 19: The Steering Gear mount

The Lynxmotion kit is another alternative with the same design but is bundled with the Hitec HS-422 servo. This servo is more power efficient and has a housing that is more weather resistant than the ones in the Mallofusa kit. However, this is the most expensive mount option.

The Dagu mini pan and tilt kit has 2 C frames and a mount for the tilting servo with the panning servo acting as the base. This design has issues similar to the mount option before. However, this kit uses micro servos, so the price is almost half that of the previous options. Smaller servos are also easier to design around.



Figure 20: The Dagu min pan/tilt kit

In contrast to Figure 20, the Adafruit Pan-Tilt kit also uses micro servos, but the frame is designed more tightly against the servos making it hard to add weatherproof housing around the servos. The smaller design also makes it difficult to mount larger cameras. The mount is made out of plastic, which is less durable and weather resistant as opposed to the metal options.

Built option:

Instead of buying the mount pre-made, we can also design a custom mount for the servos with 3D printing. This would allow us to use a custom design that would make it easier to house the whole system in a weatherproof casing. It would also be cheaper since we would only pay for the material to print the parts. However, this option has its own problems. The mount would be printed in ABS plastic which is not as durable as the metal pre-made options. Also, the plastic will degrade in extended sun exposure which isn't good for a camera meant for long term use in the Florida sun. We could always replace parts when they degrade, but since these are custom printed parts, they won't be as easily to replace as pre-made products. The only practical option would be to buy an expensive 3D printer to keep replacing the parts. Also, we would need to use the 3D printer available in the Texas Instruments workshop or the library curriculum materials center on campus. That printer has limited availability, so we need to schedule enough time to get the parts printed, especially if they are detailed because the process could take hour if not days.

Comparison:

Buying a pre-made mount for the servos would be fast and easy but designing a custom mount would give more freedom to place the components. The best option would be to buy the parts since the plastic material poses so many issues to a long-term use device such as ours. It also would take too long to print the full design which would make it hard to test the prototype in a timely manner. The pre-made metal products will still need some parts to incorporate the mount with the control board and the housing.

All the pre-made options are available in a kit that bundles the servos with the mount, so this reduces the chance that the servos won't fit with the purchased mount. Also, this helps to reduce the cost slightly since buying a kit is cheaper than buying the parts separately. Among the kits, the best option would be the Lynxmotion because it has a metal frame and

the servos included draw less current than the ones included in the Steering gear. If we wanted to use the Hitec HS-422 with the Steering Gear mount, then the price would go up to 44 dollars, thus making the Lynxmotion the cheaper option. However, I we decide on a small enough camera, the next best option would be the Dagu kit. It uses micro servos, so it can't support larger cameras, but with a small enough camera it is the cheapest and most durable option. Confirm this by examining Figure 21.



Figure 21: Lynxmotion kit

Product	Price of mount	Kit price	Material
Steering Gear Pan and Tilt	13.99	23.49 (Mallofusa)	aluminum
Lynxmotion Pan and Tilt kit	9.95	31.39	aluminum
Dagu min Pan and Tilt kit	N/A	14.16	metal*
Adafruit Pan-Tilt kit	4.95	18.95	plastic

Table 7: Comparison of Pan-tilt mount kits

Main Control Board:

The motor that is responsible for panning, tilting, and zooming the camera system is operated indirectly, with commands conceptually streaming from the controller, through the workstation software, to the embedded MCU device, through the motor servo shield board, finally reaching and controlling the motors. In practice, our motor will be controlled by PWM (pulse width modulation).

Arduino Uno:

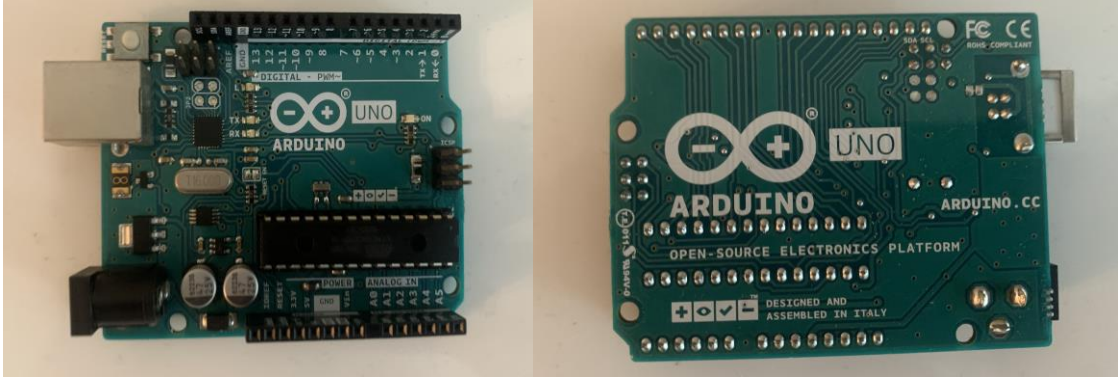


Figure 22: The Arduino Uno.

The form factor of the board can be referenced above in Figure 22. The Arduino can draw sufficient power for our application from the USB port of a workstation. The Arduino Uno Specs are as follows:

Input Voltage	8 V
Operating Voltage	5 V
PWM Digital IO Pins	6
DC Current for IO Pins	20 mA
Maximum-Voltage Output Pin	5 V
Flash Memory	32 kB
SRAM	2 kB
Mass	25 g

Table 8: Specifications for the Arduino Uno [1] [21]

Observe that the mass of the device is 25 grams. This is significantly smaller than the weight of many common motors we have examined in our research. Effectively, the mass of the MCU plus the shield chosen can be ignored.

IDE for Developing Arduino Software:

Another benefit of the Arduino is the level of software support for programming and flashing the device with the desired program. The Arduino IDE is written in C, C++, and Java and manages code, libraries, binaries, compilers, and handles connection to the Arduino over JTAG and serial interfaces that are abstracted away from the IDE user [1]. This means that the Arduino IDE handles all connection details of the board for us. A nice feature of the well-supported development environment is the logical distance from the hardware during the programming process. The compiler libraries included support register-level programming, but you do not have to probe that closely to the hardware if you do not want to. Instead, a high-level digital interface abstracts those details away, allowing you to perform digital writes over several protocols, including I2C, with a simple function call

Sparkfun Redboard:

The Sparkfun Redboard is an Arduino based device that supports servo drivers with the split 1 by 20 pin design. This board can handle power from 7 to 15 volts with 5-volt compliance voltage. It takes between 0.15 to 0.8 amperes of current through the voltage rails. The Redboard combines the functionalities of 3 Arduino products into a simple board with the best features. It's compatibility with Arduino code makes it easy to use with lots of support for building a project from the online community. This board is also designed to be forward compatible with any new drivers, so it has longevity for long term use before needing to be replaced. However, the Arduino based design also means that it won't interface with Raspberry Pi based HAT or control boards, so it is limited to drivers that support this design. Examine the layout in Figure 23.

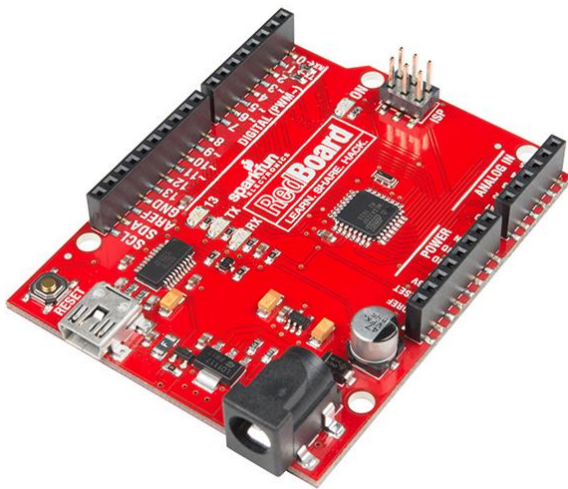


Figure 23: Sparkfun Redboard

The Pi Zero W:

The rival family to the Arduino product line is the rising family of Raspberry Pi modules. Raspberry Pi produces single-board computers that are capable of running a full operating system but has special boards that may be called competitors to the Arduino boards, in particular the Pi Zero W.

The Pi Zero W, in a technical sense, cannot be called a direct analogue to the Arduino line because of their support for running full, non-real-time operating systems such as Linux. However, since they are miniaturized computers that allow us to write directly to the GPIO header pins, they provide all the support needed to control a motor shield just as the more bare-bones Arduino family. [22]

A supremely visible marketing advantage over the Arduino Uno is the price, coming it at \$10 before tax, two times cheaper than the Uno. Although it may seem the Pi's ability to run a full operating system in tandem with the lower price beats out the Uno immediately,

there is still hope for the Arduino perspective because of its extensive software libraries and shield support. [22]

Power/Data Link(connectors):

To power and transfer data from the computer to the camera we needed to choose an appropriate connector based on the current standard used with PTZ cameras today. After some research we found that it is common for both USB and power over ethernet connectors to be used. Our original goal was to find a suitable way to implement power over ethernet after using an ethernet adapter that would combine power and the data from the ethernet IN to an ethernet OUT which would connect to the camera itself, this device is known as a power over ethernet injector. Through research we learned that the way we prototyped was not standard in the industry and the easier and more productive way was to use a USB connection of some sort. By using a USB connection, we easily can pass data and power through one cable without the need for an adapter.

Power Over Ethernet Injector:

In our earlier theoretical prototypes for this project we planned to use power over ethernet which created the need to inject power into the ethernet cable transferring data from the laptop. Upon search we found the power over ethernet injector that perfectly fulfilled are theoretical need. The adapter first takes two inputs of power via dc power cable and data via ethernet cable and converts it to supply power and data via one single ethernet cable to PoE devices without the need for any other type of power. In some variations the injector is also able to detect the required power up to 15.4W after connected, which was in the bounds of our project.



Figure 24: TP-LINK 802.3af Gigabit PoE Injector

As seen in Figure 24, the TP-Link injector easily enables PoE to any non-PoE device without the need to replace any components. It also has fast data transfer with both 1 Gigabit input and output ethernet speeds. Power can be delivered up to 328 ft away if desired. This injector also can auto detect power and deliver the required power supply for the powered device up to 15.4W.



Figure 25: iCreatin Wall Plug PoE Injector

As seen in Figure 25, the iCreatin Wall Plug PoE Injector is the direct to wall injector variant that reduces overall price by getting rid of excess wiring altogether. For this design all you need is two separate ethernet cables and a DC power wall outlet. This injector also includes short circuit protection for Over Currents and Voltages. It can also protect devices from overloads over 500mA and high voltages exceeding 50V.



Figure 26: Cudy 30W Gigabit PoE+

As seen in Figure 26, the Cudy 30W Gigabit PoE+ converts a non-PoE Gigabit port to a PoE or a PoE+ Gigabit port. PoE+ differs from PoE as Power over Ethernet + can deliver up to 30 watts over ethernet cables compared to PoE's 15.4 W. This injector also supports backwards compatibility so even PoE devices will work and be auto detected by the system.



Figure 27: Cudy POE400 90W Gigabit Ultra PoE++

As seen in Figure 27, the Cudy POE400 90W Gigabit Ultra PoE++ comes equipped with auto sensing technology that delivers up to 90W to a PoE++ device all while supporting backwards compatibility for PoE+ (60W and 30W) and PoE (15.4W). It also has 1 Gigabit Ethernet input and output ports supporting a full duplex of 2 Gbps.

Comparison:

For our project anyone of these injectors would work perfectly with our overall system, so if were to use PoE the best choice would be Tp-Link’s 802.3af Gigabit PoE injector. The reason this injector is the most suitable overall is its lower price and high speed. The iCreation injector is cheaper but the speed of only 100 Mbps may impact the video quality being sent over the ethernet. The worst choice would be Cudy’s POE400 90W Gigabit Ultra PoE++ because its specs are needlessly high-end and the power it can supply is far greater than needed. The price is also impacted as it is \$62 more expensive than the closest injector in cost. Finally, Cudy’s 30W Gigabit PoE+ would be a good option if we required slightly more power but the Tp-Link is slightly cheaper and fulfills our needs.

Injector	Price	Power	Speed
Tp-Link 802.3af Gigabit PoE	\$16.99	Up to 15.4W	Up to 1000 Mbps
iCreatin Wall Plug PoE	\$8.98	Up to 24 W	Up to 100 Mbps
Cudy 30W Gigabit PoE+	\$17.90	Up to 30 W	Up to 1000 Mbps
Cuddy POE400 90W Gigabit Ultra PoE++	\$79.90	Up to 90 W	Up to 1000 Mbps

Table 9: Comparison of PoE injectors

Ethernet Splitter Adapter:

Another device that came up in one of our drafts was the ethernet splitter which basically reverses the process of a power over ethernet injector. In the case that the laptop used were to provide power over ethernet and the camera only used the ethernet connection for data this option would split the power from the ethernet. Ultimately the discussion of adding this adapter was also the decisive moment where we decided to pursue the most standard cable connection in cameras today, the USB.



Figure 28: iPolex Active PoE Adapter

As seen in Figure 28, in the situation where an ethernet splitter would be used a good option is iPolex's Active PoE adapter. It comes equipped to handle an input DC voltage of 44-57V and includes overvoltage protection. The data rate is 100 Mbps and its compatible with IP camera, IP phone wireless AP and most 12V Non-PoE devices. This adapter also ships in a pack of 2 and is competitive in price to single shipped alternatives.



Figure 29: Amcrest Active PoE Splitter Adapter

As seen in Figure 29, an alternative to the iPolex splitter is Amcrest's Active PoE splitter that includes an adjustable DIP switch for 5V, 9V, and 12V adjustable output. It can deliver power up to 328 ft and includes interchangeable cord tips to serve a multitude of different camera power types.



Figure 30: Steamemo Active PoE Power Over ethernet Splitter Adapter

Finally, as seen in Figure 30, the Steamemo's Active PoE splitter adapter is the most cost-effective option out of the alternatives and has almost near identical features when compared to the iPolex adapter. This splitter comes with multiple protections such as overvoltage protection, short-circuit protection, and overcurrent protection. It also has a built-in isolation transformer, which enables it to protect and supervise regular operating of the equipment.

Comparison:

In the scenario where using a splitter is feasible the best choice would be the Armrests' splitter, the speed needed for transferring video is almost essential to be 1 Gbps when dealing with high quality video. Steamemo's price and quality is hard to compete with but the speed alone disqualifies it from consideration. The iPolex splitter is individually

cheaper than the Steamemo adapter however it is only sold in packs of two doubling the price making it an unreasonable choice as the specs are almost identical.

Splitter	Price	Output	Speed
iPolex Active PoE Adapter (2-pack)	\$16.99	12 V	Up to 100 Mbps
Amcrest Active PoE Splitter Adapter	\$19.98	Adjustbale 5V, 9V, or 12V	Up to 1000 Mbps
Steamemo Active PoE Splitter Adapter	\$8.99	12 V	Up to 100 Mbps

Table 10: comparison of PoE splitters adapters

Remote Control:

For this project we had three initial options for the remote controlling of our device, the Xbox One controller, the PS4 controller, and a generic wired controller. Originally in theory the PS4 controller was the most favored as many open-sourced options have sought to handle the porting of Sony’s PS4 controller to the non-native Windows operated PC. Since Microsoft designed both Windows and the Xbox One controller it was natural the great amount of support available to it on its own platform.

The generic controller was also a good option since it was designed to work with as many platforms as possible, such as Android OS, multiple versions of Windows, and the PS3 console. When comparing the three the most practical to implement when using Windows happens to be the Xbox One Controller. On the Windows platform the Xbox one controller has the most up to date and well-maintained API library provided by Windows and it has the best ability to establish connection wirelessly. The API that would be responsible to connect the PlayStation 4 is no longer maintained or documented, even though it is stable and can be implemented the Xbox’s API is more desirable. Even though the generic controller is compatible with the Xbox’s API and cheaper its inability to connect to the controlled device wirelessly puts it a step behind the Xbox One controller.

PlayStation 4 Controller:



Figure 31: PlayStation 4 Controller

As seen in Figure 21, the PlayStation 4 controller, also called the DualShock 4, is manufactured and designed by Sony a Japanese multinational conglomerate which is one of the world's largest manufacturers of consumer and professional electronics and related services. Since we are using PC, we originally assumed that the ps4 controller would be easier to implement as it has a lot of open-sourced adaptations. However, the windows operating has an API that interacts with the Xbox One controller better. Since support for the ps4 controller is no longer maintained we opted to use the latter. Also, the PlayStation 4 controller supports three-axis gyroscope and three-axis accelerometer which are used for motion detection.

Xbox One Controller:



Figure 32: Xbox One Controller

As seen in Figure 32, the Xbox one controller is manufactured and designed by Microsoft an American multinational technology company that specializes in computer software among other related services. Most of Microsoft software is made for the PC which is the type of computer we have decided to use for this project. For this controller independent developer support was implemented from day one which allows for easy interaction with only a little experience in a programming language such as C++. Unlike the other controllers the Bluetooth capabilities of the controller are free to use for independent developers making wireless connection possible. The version of the controller we are using released in 2020 also has Micro-B connection capabilities as an alternative to Bluetooth.

Generic controller:



Figure 33: Generic Wired Gaming Controller

As seen in Figure 33, the generic controller is a wired gamepad that is compatible with Windows, Android OS, and the PlayStation 3 console. Its USB cable is 6.5 feet long which allows for ample space from the device being controlled. Equipped with dual vibrators, a feature that could be used to notify of reaching the end of the PTZ cameras range by sending vibrations at the limits of motion. The controller also allows for customizable buttons that repeat commands even after they are pushed. The major draw to this controller is its low price and ability to handle the needs of this project.

Comparison:

To serve the purposes of this project the Xbox one controller is the most suitable. The Xbox One Controller is closer in relation to the operating system most PCs run on in Windows and is heavily supported by most apps that use controllers when compared to Sony's PlayStation 4 controller. The difference in support on Windows is clearly seen when noticing that the PlayStation controller is only compatible with a legacy Microsoft API called DirectInput which is no longer maintained and isn't compatible with today's Windows Store apps.

However, the Xbox one controller uses a well maintained and up to date API known as the XInput Library, made specifically for it. Finally, the last thing considered when deciding which controller to use was the price. Currently, the Xbox One controller sells for \$54.99, slightly cheaper than the PlayStation 4 controller's \$59.99 price tag. The generic controller also would use the DirectInput API and must be wired which are both undesirable attributes not worth its lower price point.

Controller	Price	Battery Life	Charging Interface	Wireless	Note
PlayStation 4	\$59.99	4-5 hours	Micro USB	Yes w/ Dongle for PC.	Controller batteries are non-removable and rechargeable
Xbox One	\$54.99	10-20 hours if removable batteries are used	USB	Direct Bluetooth	
Generic Controller	\$26.99	Unlimited	USB	No	Must always be wired to device.

Table 11: Comparison of remote controls

Design

Prototype:

Our design is made of two systems. The camera system controls the audio and video stream, connects the camera to the streaming software, and controls the zoom. The motor system controls the motors and the movement of the camera, the pan and tilt. Both systems are controlled by the laptop and remote control. The motor system takes external power from a plug-in power converter and receives instructions from the remote through a Wi-Fi receiver on the main control board.

The camera system is connected through USB cable to get instructions from the remote and transmit the video stream to the streaming software through the laptop connection and provide continuous power to the camera.

The hardware block diagram in figure 37 illustrates the physical connections between subsystems in the device in our original design. The capture device is the controlling system for the audio-video subsystems, as well as directing the motor to the appropriate angles in order to frame the shot correctly and quickly. Any commands and media data are transmitted over the power-data line, where commands are brought in from the controller, and where the media is written to disk. The mount and tripod support the capture system, the weight of the cables connecting the workstation, capture device, and motor, providing stability and the altitude for media capture. Some details have changed between this

original design and the one discussed in the hardware and software sections, which reflect our final design.

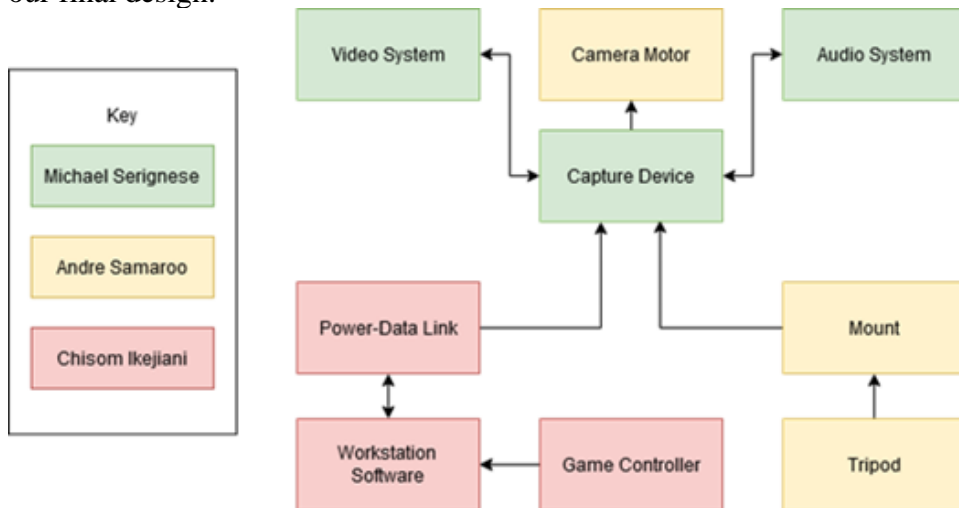


Figure 34: Hardware Block Diagram

Housing:

Our project needs to be used outdoors so we need a housing design that will keep the parts dry and withstand light wind. We decided on making a custom housing unit with 3D printing. This allows us to build our motor mechanism how we wish without being constrained by a pre-built product. The housing for our pan-tilt camera needs to keep the electric components dry in light rain conditions and withstand 15 mph wind as outlined in the requirement specifications. The housing unit also needs to be strong enough to stand up to the Florida sun without melting or getting sun rot. We also want the design to be easy to take off so that if the motor system or camera needs repairs, we can easily remove the housing and then put it back on. The housing unit design needs to meet all these requirements while staying affordable and easy to manufacture so that we can meet our budget and manufacturing requirements as outlined in our market research section of the report.

We can fabricate our housing unit in a few different ways. One would be to build the housing out of sheets of plastic or aluminum and cut the sheets based on how we want to shape the housing and pasting the pieces together with weather resistant adhesive. This method is less than desirable because the flat sheets can't form curved surfaces easily such as for a sphere. Another option is thermal plastic. Thermal plastic is type of plastic that when heated, can be molded into whatever shape you want. This would be able to make curved surfaces more easily and can be molded around the motor system easily. However, thermal plastic is difficult to smooth out, the shapes are often lumpy unless you use a hard mold to form the plastic. This would make our product look unprofessional. This method also requires a person to hand make the housing unit which is not good for manufacturability. If a client would need to replace the housing because of a broken part, a new housing unit would need to be hand made as opposed to buying or even machine manufacturing a new unit. Our third option is to use 3D printing to make our design replicable, customizable, and easy to build.

3D printing uses a thermal plastic in the shape of filament to build a computer designed object by layering the plastic filament up until the full design is printed. Using a Dremel 3D printer is the best option for a fast and delicate design. While highly detailed and large designs can take days to print, our design is a simple shell casing for a small pan-tilt system with an action camera. We could print in materials such as metal, but that requires a specialized type of printer, and we really only have access to printers that take plastic filament unless we seek out a 3D printing service that would make our design and ship it. In the long run, finding a service would be best for manufacturing and allow clients to order replacement parts easily through the manufacturing pipeline, but for this simple project prototype, using the university's printing services will suffice. There are different plastics we can use as well. The most popular are polylactic acid (PLA) and acrylonitrile butadiene styrene (ABS), though plastics like nylon and thermoplastic elastomers (TPE) also exist but are more expensive. ABS is the best option because of its strength and flexibility is better compared to PLA, though ABS still has its limits. It is known to shrink and warp during the cooling process and the machine needs to be in a well-ventilated space because of the toxic fumes from the melted plastic. For our purposes, the advantages outweigh the disadvantages. [17]

To make our design, we need to use two software applications for modeling and slicing. Modeling is the actual design and construction of the object. Slicing is taking the model file and slicing it into layers that the 3D printer will use to build the object. There are many modeling options to choose from including Fusion 360, FreeCAD, even Blender. What is most important is the program's ease of use and file format. Since our housing design will be simple, we don't need a modeling program with lots of features for complex modeling. Simple geometry manipulations will be good. Fusion 360 is the best option since it is free to UCF students, and it is used by all industry professionals. Slicing software will take the modeling file and turn it into G-code for the 3D printer to functionally use. This conversion will also give the printer data on the filament, nozzle temperature, and print height. Some programs include Ultimaker Cura, Slic3r, and ideaMaker. These are all open-sourced, so they are free and work with most any printer. When selecting the slicing software, we need to consider our printer. We will be using the printer in the school's production lab or the TI innovation lab. [19]

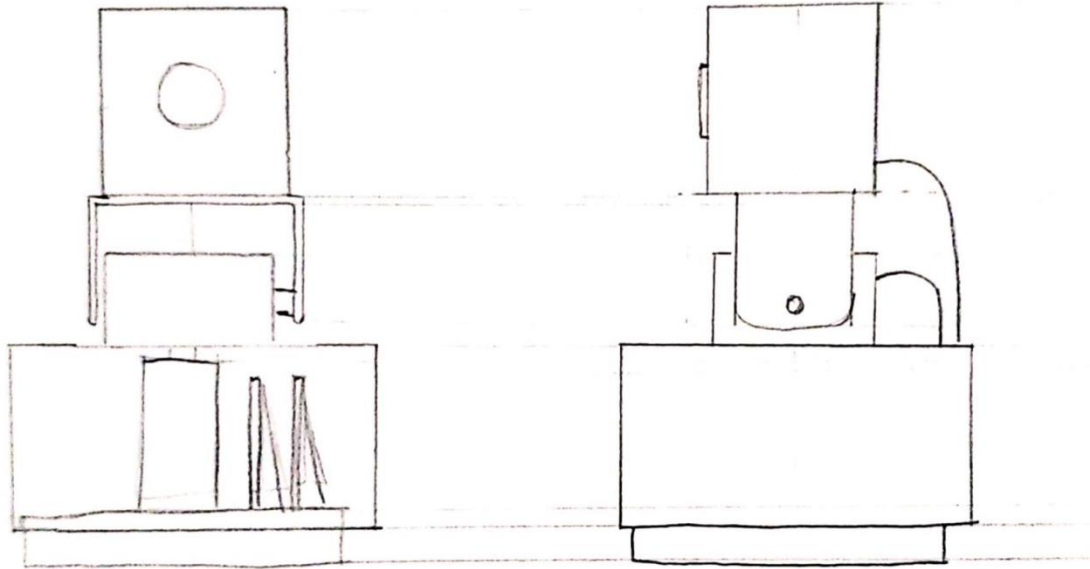


Figure 35: Box design

Design 1 – box

One design option is to make a simple box casing that covers the main control board and panning servo and another box casing around the camera and tilting servo. The box covering the panning servo will have a separate bottom that will allow the housing to move with the panning motion. Dividing the housing into two parts allows the servos to move freely while keeping the covering tight to the servos to prevent water from getting in. This minimal design is also easy to print since it requires almost no support structures. The printing time will also be very short since this design keeps simple flat planes, the best geometry for fast 3D printing. The boxes would be printed in two parts that use interlocking connectors to snap together. This allows for easy assembly and disassembly for repairs. The interlocking design will also keep water out since water will follow the channel along the seam and stay out of the main containment area. The boxes will also have small holes for the wired components between the two housing units. The tilt box has a wire for the tilting servo and a wire for the camera power. The panning box has the tilting wire running into it and the power cable running out of it. In order to give the wires enough slack while keeping the wires in place, a rubber holder will secure the wires to the tripod so that it doesn't wear the cables down when they move.

One critique of this design is that it is not aerodynamic. This design exposes a lot of surface area and that makes it susceptible to strong winds. When wind hits hard edges or goes through tight spaces, it creates wind resistance. This will make the camera system shake which will at least ruin the video stream footage and at most may break the system off of the tripod. It also leaves the wires exposed which could be a point of entry for water. The

best solution is to use rounded shapes such as cylinders and spheres to reduce the wind resistance. We can also use rubber stoppers around the wiring to seal off the holes from water.

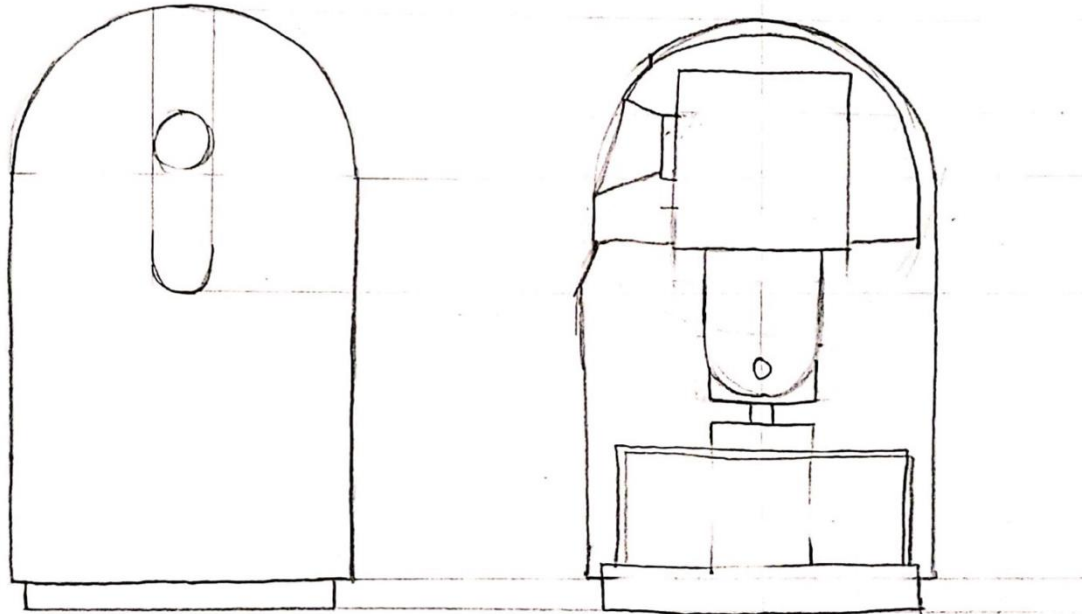


Figure 36: Turret design

Design 2 – turret

The turret design is based on similar PTZ security cameras. The casing consists of two parts: the panning base and the tilting camera cover. The base is a rounded cylinder that covers the entire system, but at the top it has a slit opening to allow the camera to see out of. The camera cover is a half sphere that sits inside the main base and covers the slit where the system would be exposed. As the camera tilts, the cover maintains covers from the part of the sphere that is inside the main base. The bottom of the base is detached similar to the previous design. This design encapsulates the whole system so that none of the wires are exposed and the single solid body is best against the wind because it won't create as much drag as two separate pieces. This design will also use the rubber holders for the wire from the previous design. However, since all the components are under one unit, only one outlet for the power cables is needed at the bottom by the base of the tripod mount.

However, the rain can get in more easily through the slit in the top. The camera cover still needs room to move, so rain can still fall between the gap between the camera cover and the base cover. This would damage any of the exposed electrical components. A possible solution is to create a casing for the control boards separate from the outer housing. The servos come in a plastic casing that doesn't leave much room for water to get to the electrical components, we could similarly encase the Pi and the Sparkfun controllers so that they are protected without obstructing the motion of the servos.

weather proofing:

The parts of the housing unit will be printed in parts for easy assembly and disassembly as discussed previously. However, this means that the seams where parts meet are vulnerable to water penetration. We can prevent this in a few ways. One would be to make interlocking channels. This design has a small recess in the lip of the housing unit that will lock with the lip of the other half of the unit. This creates a close seal, but also prevents water from getting by trapping it in the recess and draining out. There are many manufacturers that use this design for plastic casings, so we know it's a viable option. The issue is that this design is very hard to open without a way to break the seal. You can think of this as like Tupperware without the raised corner to help open it. This can lead the client to break the casing if they are removing it for cleaning or part maintenance.

Another option is to simplify the design and just make a flat overlap between the parts and use screws to keep them together. The overlap will prevent the water from seeping in similar to the channel, but since we are using screws to keep the housing together, it will be easier to take the pieces apart for maintenance. However, we still can get some water through the gaps if they are large enough. A simple solution is to use rubber stoppers. Similar to how your house door keeps the air from seeping out, the rubber stoppers will be a watertight seal. We can also use them around the openings for the wires between the parts of the system to make sure water does not travel down the wire and into the electrical components.

In addition to the housing for the entire system, we also need to design the frame that will hold all the components together. In both designs, the bottom is a separate piece that allows the upper casing to pan freely. This bottom section will hold the panning servo and the two control boards. It will also connect to the tripod mount. To hold the control boards, the bottom needs to have 2 pairs of arms that will align with the mounting hole on the boards. These arms will have holes for small screws to secure the boards, so they won't shift or move while the system is in use. Next to the control boards, the panning servo will be mounted by screwing it into the base. The casing provided with the servo already has holes for screws to secure the servo to a mount. The base will need holes drilled where the servo will sit. To connect to the tripod mount, the simplest option would be to just drill a hole. Tripods usually come with a mounting plate that uses a small screw to secure the device to the plate and then the plate is secured to the tripod. This is an acceptable option but since we are printing the housing, we could just print the plate right to the bottom structure. Most tripods have a standard plate that works between different tripod brands. Therefore, we could make the plate in our 3D design and print it as part of the housing unit. This would make it more secure since the whole design would be one piece. There are some specialty camera stands that use a special plate connection, but these camera stands don't have the screw-in base either, so they would need a specially designed camera mount connection. Most clients won't need this, so it's safe to go with the more popular design and print the mount onto the housing unit.

Hardware:

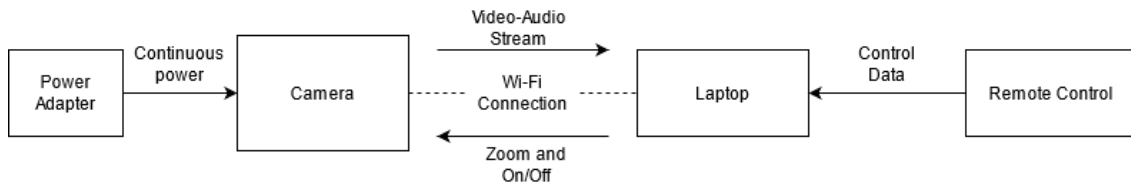


Figure 37: Block diagram of camera sub-system

The camera system uses an action camera which can be powered over USB connection and can transmit over Wi-Fi or USB cable as well. The camera will take power through the USB cable to provide continuous power. The USB cable will also carry the audio and video data from the camera to the laptop. In order to use USB as both a power source and data transfer line, the USB cable will connect to the USB port of the laptop which keeps power from its charging cable.

The laptop will use OBS streaming software to live stream the video to users through the internet. An Xbox One remote will connect to the laptop and control the on, off, and zoom functions of the camera.

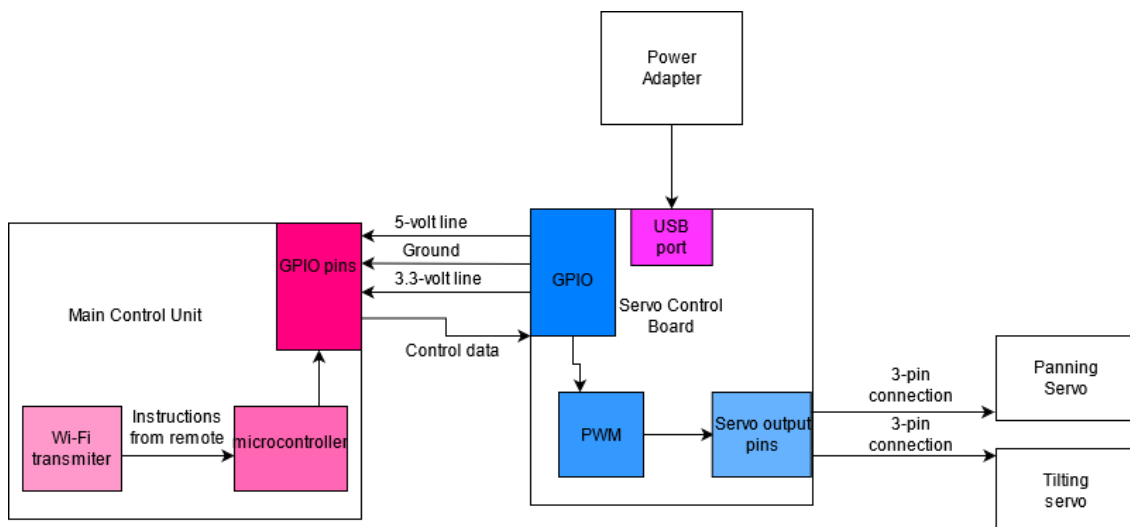


Figure 38: Block diagram of the motor sub-system

Motor control:

The motor system consists of 2 servo motors, an aluminum camera mount, a PWM board, and a main control board. The servos control the pan and tilt functions and connect to the mount physically to perform those functions. The servos draw power and signal response from the pin connections to the PWM board. The PWM changes the width of the on signal to tell the servos what position they should rotate to between 0 and 180 degrees as opposed to changing the speed.

Speed can also be changed by simply changing the pulse width signal slower which has to be interpreted by the remote software. An external PWM is necessary for most control boards because they use Linux as an operating system. Linux doesn't have a reliable clock especially when interfacing with multiple boards and devices. For servos, it's important to

have a consistent clock so that the sensor on the servo doesn't mistake an inconsistent clock with a shorter pulse width and cause the servo to jitter.

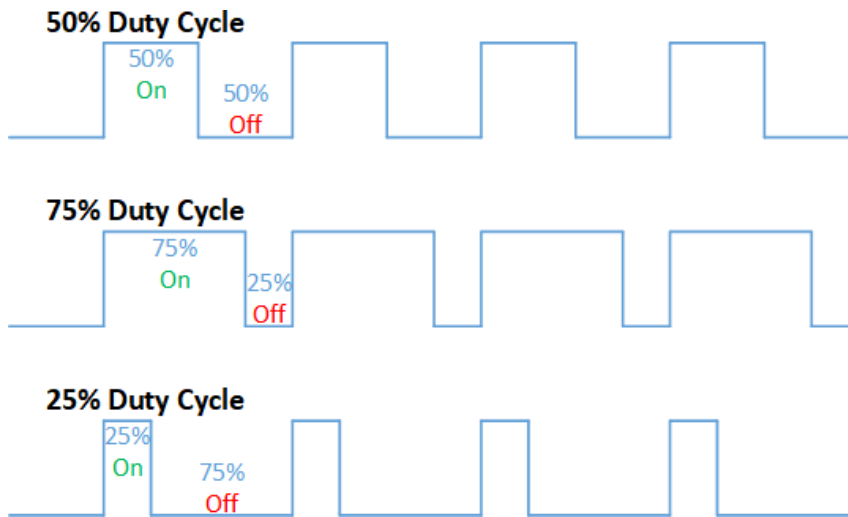


Figure 39: PWM example. Reproduced under public domain.

Pulse-width modulation controls the voltage provided to a device in a specified manner that decreases the amount of power relative to a constant DC signal of the magnitude of the peaks. Examine Figure 39. According to the duty cycle, which is the ratio of the time the voltage is in the 'on' position vs. the 'off', a device receives many small bursts of power per second which enables us to control devices in a gradual manner. Some devices, such as very small motors or display panels, require that pulse-width modulation is used; if a DC-peak voltage were applied to the device, it may overheat, burn out, or otherwise damage the component.

Particularly in regard to motors, pulse width modulation allows us to control how far the motor pivots and how quickly it ends up traveling there. This in effect allows us to control the position and speed of the motor with minimal control hardware inside of the motor itself. The task is simply reduced to sending this alternative voltage waveform in the duty cycle that yields the speeds and position required.

This waveform can be crafted directly from a MCU's (microcontroller unit) power supply and, using software libraries or even implementing it on one's own, software written to control this process. However, entertaining the idea of implementing one's own PWM signal raises some alarm bells related to reinventing the wheel.

The concept of PWM is so widespread that there are accessory boards designed specifically for controlling motors that implement PWM for us. This comes as no surprise for those with a background in robotics, where many so-called "hats" or "shields" on MCUs are already designed and available for purchase. These shields are accessory hardware to commonly available MCUs and implement popular functions in hardware for the designer's convenience.

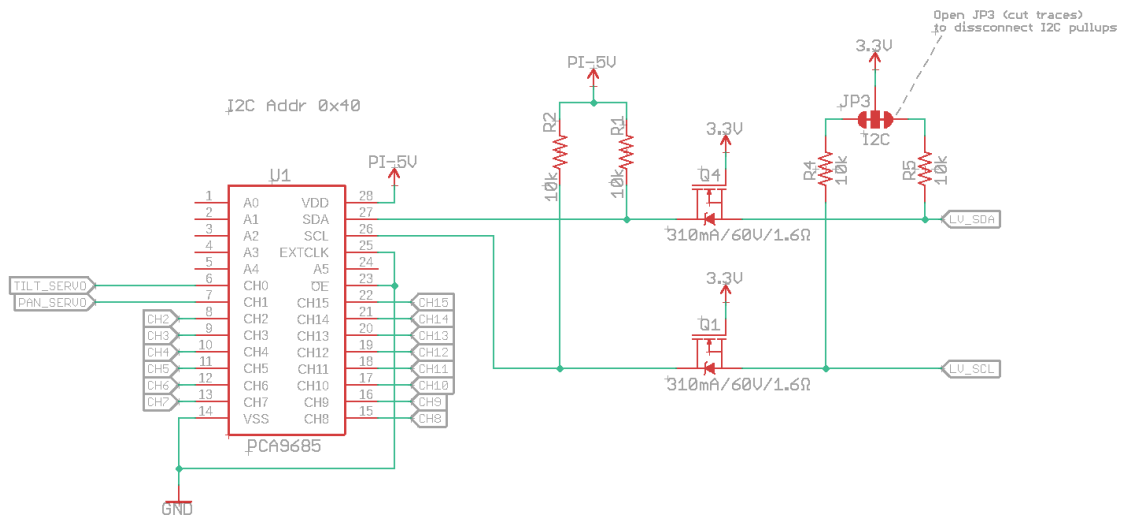


Figure 40: PWM on Sparkfun servo pHAT. Reproduced under creative commons

The PWM needs to connect to the main control board to receive the control signals from the remote. The Sparkfun servo pHAT has three ways of interfacing with the main board. One is to use the GPIO pins. The interface through which the main MCU and the shield communicates are the GPIO pins. There are usually a few dozen GPIO pins on an MCU; since we only intend to control one motor-servo setup, we will have many pins to work with to provide power to our shield, and then indirectly from there to our motors.

Power distribution:

The PWM board draws power from a USB 3 plug with an outlet power adapter. This is used to drive 5-volt power to the servos and draw 3.3-volt power to the board. The PWM board is also driving power to the main control board through the GPIO pins: the 5-volt power pin, the ground pin, and the data pins. Since power is delivered through the GPIO pins, there is no power security to prevent overloading and burning the board.

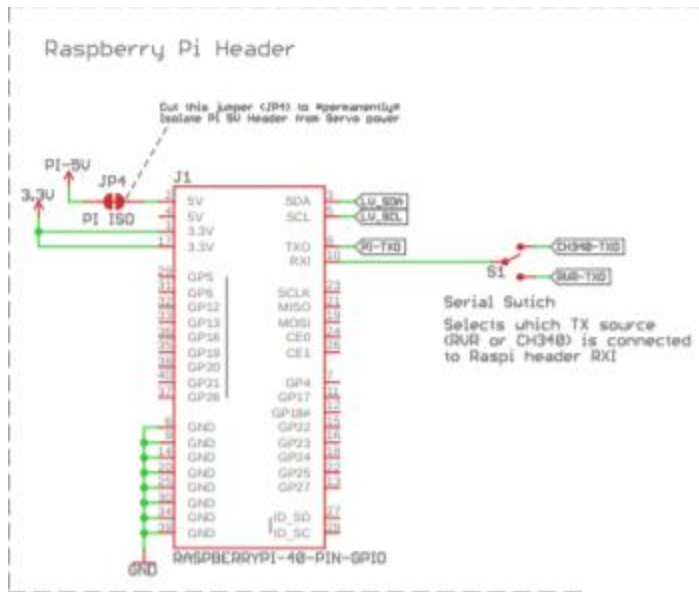


Figure 41: Pi GPIO header pins on Sparkfun servo pHAT. Reproduced under creative commons.

An alternative is to use the I2C connections to integrate with the main board. The GPIO pins also use I2C, but in a more general sense since all the pins are connected even if we don't use them. With the pins on the qwiic port, there is only ground, Vcc, input signal, and output signal as shown in the diagram below (add pin outputs). These are the bare bones of an I2C connection. The Vcc supplies the power for the 5-volt rail on the board and the ground provides a common ground among all the devices. I2C allows peripheral devices to communicate with control devices with minimal port accesses. It supports methods like chaining devices or with our HAT, stacking devices, so we would be able to use multiple peripherals. However, the qwiic port connections would still use the GPIO pins like the header. The only difference is that instead of using the whole header, the qwiic port would only need to use 4 pins: ground, 5-volt rail, and two programmable pins for signal transmission and reception. This kind of port is more suited for use with other Sparkfun products that also use the qwiic system to quickly integrate peripherals with I2C while using as few ports as possible to maximize the number of peripherals we can use. Since we are only using the Pi Zero W and the Sparkfun servo pHAT, we don't need to worry about more peripherals. If we decide to integrate the camera into one control device, then we would use the micro-USB port or the mini-HDMI port. [14]
 (what is I2c and how does it do power protection?)

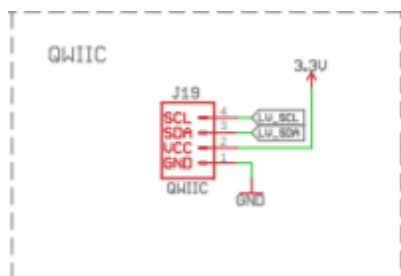


Figure 42: Qwiic I2C port on Sparkfun servo pHAT. Reproduced under creative commons

We can also use the USB port to interface the devices. We want to use the USB to carry the power to the main system, but USB is also capable of transferring data as well. This can be compared to how you can charge your phone and transfer files from your camera through a USB charging cable. Similarly, the USB 3.0 port on the Sparkfun servo pHAT can connect to the micro-USB port on the PI Zero to take 5-volt power from the Pi to the Sparkfun and split that between power for the servos and power for the board. Using this connection method helps to reduce wires by bundling the power and data lines into one port, but also providing power protection since USB can carry a maximum of 5 volts. USB would also be good because the distance between the two devices is short. USB cables can't transmit signals over distances over 5 meters so the short distance is best for this kind of connection. However, if the 2 control boards are using the USB ports to communicate, then we can't use the port on the PI to integrate the camera if we wish and we can't use the port on the Sparkfun pHAT to power the system.

USB-to-Serial Converter

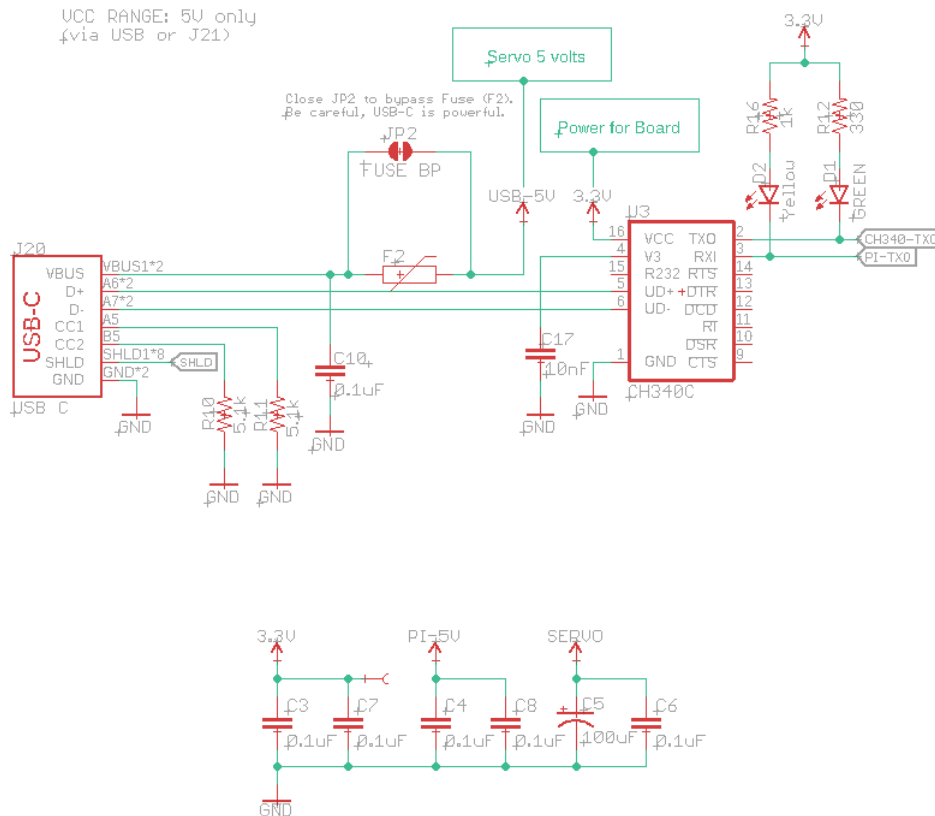


Figure 43: USB port on Sparkfun servo pHAT. Reproduced under creative commons

The PWM connects to the servos through two of its 16 servo ports. The main control board will be programmed to look at the two that we are using. The port addresses will also help to define the pan servo from the tilt servo so that the position signals from the remote doesn't confuse the two.

Wi-fi control:

In order to reduce the number of wires needed to run to the system, we are implementing wireless control of the motor system through wi-fi. The main control board will support the wireless communication to the remote. Wi-fi control can be easily implemented using by syncing the transmitter and receiver on the same network. Once on the same network, the device will use internet protocol to search the local network for the remote's IP address. With the IP address stored, the remote and the servo control board will use internet protocol to transmit information in internet frames. This method may pose challenges in areas with slow wi-fi or areas with none at all, so it is important to have a wired connection as a backup plan.

Wi-fi is used in order to transmit the video and receive the motor control signals. The Pi Zero does this through the Broadcom BCM43438 chip, which supports 2.4GHz LAN as well as Bluetooth. The chip uses a method similar to how you would use a wireless printer to print directly from your phone. The board communicates with the remote by taking a wi-fi address on the same LAN as the remote. Then, when the remote and workstation are first initialized on the network, they will search for the IP address connected to the Pi device. Once the two systems have each other's address, they can pass data over the network router. We can also apply a similar principle using Bluetooth instead. [20]

The local area network (LAN) is the internet network tied to a single endpoint, in our case a router. Internet protocol gives each device on the network a unique IP address with the same base address. The base address is to define the router so having the same base address means that all devices are on the same router. Since there is only one router, the devices connected on the LAN can easily search for each other by going through the addresses and seeing if the device matches the search specifications. Raspberry Pi's are a distinct device that the workstation can be easily programmed to look for through the LAN. Once the workstation has the address, it can freely communicate to the Pi without searching the network again. We don't need the Pi to store the IP address of the workstation because the communication is only one way. The motors would never need to send information to the workstation.

Similarly, Bluetooth can connect these devices, but instead of using the LAN network, the device communicates directly to each other through a link established at the time of initialization. The benefit of this method is that the devices are talking with each other over a private line instead of using a wi-fi network that could have dozens of users on at once, especially at public sporting events. Also, Bluetooth establishes a connection at the time of activation, so we don't need to program a script to allow the device to connect to different networks at different venues. We can just do a signal sensing operation at the beginning and the computer will detect the motor system. This is also good because most commercial laptops support Bluetooth already, making it an option with a low barrier to entry. However, Bluetooth has limited capabilities. Bluetooth has low bandwidth and slow speeds. The motor system may not need the same speed as the camera for wireless communication, but it still needs to respond to the operator's controls quickly to follow the action being captured on the camera. It's also necessary if we want to implement our stretch

goal of using auto tracking because the motor system needs to respond to the video stream in a timely manner to maintain view of the moving target.[18]

For our purposes, wi-fi control is best since it offers high quality fast data transfer. Also, since the camera will also use wi-fi to transmit the video, the laptop will not have to manage two different protocols for communication. There is also a lot of resources to help make the communication program over wi-fi that will make development quicker and easier. The camera will be transmitting on a different wi-fi signal, so we hope to integrate both in a single simple interface to make it easy for the operator to work with.

Usb to barrel connector:

We want to use the USB port on the Sparkfun servo controller to deliver the power. However, USB cables don't carry signal further than 5 meters (16 feet). In order to use this control board, we would need to use a USB cable with an amplifier, use multiple USB cables with repeaters, or use a converter to take power from a barrel plug and fit it to the pins on a USB port.

The best option is to use a converter because they are cheap and simple to use. Signal amplifiers are a specialty device that can be expensive and hard to replace for a client if theirs breaks. AC power converters with barrel plugs are ubiquitous and cheap. They can also carry power in excess of 30 feet. This makes them easy and cheap to replace when necessary. The plug converter is a simple female barrel plug to male USB port device. It just maps the power provided by the AC power converter to the proper pins on the USB port. However, since the cable only carries power, the only pins that will be important are the voltage and ground. We could also switch control boards to one that has the barrel plug terminal to avoid the converter. Many of the option from the technology investigation section had a barrel plug terminal, but they also have less flexibility with the board-to-board communication methods.

What is Power over Ethernet?

Today's PTZ and security cameras are mainly powered using power over ethernet. In the year 2000 Cisco invented the Switch Cisco Inline Power device, the switch provided ethernet ports that provided PoE conversion. In the early version of the switch the twisted pairs inside the ethernet cord that weren't being utilized were used to send power over the cord. In 2003, the Institute of Electrical and Electronics Engineers (IEEE) and CISCO standardized the first PoE standard known as 802.af, able to power up to 15.4 W of DC power on each port. However, as technology advanced the need to install more connected devices and provide only one source of power was desired on a multitude of devices connected to a single network. By 2009, a new standard of Power over Ethernet was implemented and named 802.at (Type 2) or PoE+. Type 2 PoE supplied 30W of power and is today the most common type of PoE used in the industry. Around 2011, Cisco was again able to allow for more power by using all four twisted pairs of wire in the ethernet cable to be used for power. Cisco's 2011 implementation was named UPOE and was alone theirs, it allowed for 60W of power. Also, in 2011 IEEE was able to standardize Cisco's latest design and it was named PoE++ or 802.bt (Type 3). PoE++ can provide 60W of power through a single ethernet. PoE++ enabled the powering of a switch to a switch which was

a major implementation that provided the use of less resources to power many devices. In 2018, PoE++ was adapted to allow for 90W of power down a single cord.[27][28][29]

The main two different systems both attributed as PoE, Active and Passive. Switches that run active PoE negotiate with end devices to negotiate with the end device to decide the amount of power to discharge. Cisco Discovery Protocol (CDP) was the first implementation of an active switch, it acts as a bridge between the switch and device to calculate the amount of power discharged. CDP was standardized industrially and called the Link Layer discovery Protocol (LLDP) which works the same. In a passive switch the power is always on, unlike active that can differentiate between devices that should and shouldn't be served power a passive system will pass power to all devices connected. If a device that doesn't use PoE is connected to a passive system, the device is liable to take damage. Passive systems are still used today but are likely to become legacy equipment in the future.

Why Use Power Over Ethernet?

One of the main goals of our PTZ project is to be able to shoot high quality video, as we are dealing with network equipment the use of the industry standard PoE was commonplace. For our project the software used to control the camera will be hosted on a portable laptop which would have been connected to the camera over ethernet and the microcontroller over USB. The problem that initially arose was the need to power the PTZ camera and pass data at the same time while also minimizing the number of cords and direct power sources needed for the entire projects design. Also, the original sponsored design of the project by QwikCut called for PoE functionality. As an industry standard, many cameras support PoE functionality so the using a PoE injector would solve our laptop power problem. PoE would provide 1 Gigabit data transfer and at least 15.4W of power between the camera and the laptop which was perfect for our design. In the end we decided against using PoE as the portable camera we decided to use can be powered and transfer data over one USB-C cable between the camera and the laptop.

Model	TL-POE150S
Standards	IEEE802.3, IEEE802.3u, IEEE802.3ab, IEEE802.3af
Ports	1 10/100/1000Mbps Auto-Negotiation RJ45 LAN port(LAN IN) 1 10/100/1000Mbps Auto-Negotiation RJ45 PoE port(POWER+DATA OUT) 1 power socket(DC IN)
Network Media	10BASE-T: UTP category 3, 4, 5 cable (maximum 100m) EIA/TIA-568 100Ω STP (maximum 100m) 100BASE-TX: UTP category 5, 5e cable (maximum 100m) EIA/TIA-568 100Ω STP (maximum 100m) 1000BASE-T: UTP category 5, 5e, 6 cable(maximum 100m)
Basic Function	<ul style="list-style-type: none"> Auto-Sensing Algorithm enables providing power with 802.3af PD

	<ul style="list-style-type: none"> • Delivers power and data over the single cable up to 100 meters • Auto-determine the necessary power requirements • Plug-and-Play
Power	Input: External 48VDC power adapter Output: Auto-determine the necessary Power requirements (max. 15.4W)
LED Indicator	ON: Supplying power normally Flashing: NO PD connected/Supplying power abnormally OFF: No power supply
Dimensions (W x D x H)	80.8 x 54 x 24mm
Certification	FCC, CE, RoHS
Package Contents	TL-POE150S, Power Adapter, RJ45 cable, Installation Guide
Environment	Operating Temperature: 0°C~40°C (32°F~104°F) Storage Temperature: -40°C~70°C (-40°F~158°F) Operating Humidity: 10%~90%RH non-condensing Storage Humidity: 5%~90%RH non-condensing

Figure 44: Tp-Link DataSheet

Power over Ethernet Design

In order to implement PoE, we need to use power sourcing equipment (PSE) and power splitting at the powered device (PD). PSE are the devices that supply the power and ethernet signals in PoE. We use PoE injectors to combine the two signals a single ethernet cable. The PD uses a PoE splitter or intermediary device to split the signals and wind don the voltage for the board to use. For PSE we really only need the injector, however there are several devices we can use for the PD.

Most commercial control boards don't carry Ethernet ports. The ones that do, like the Raspberry Pi, don't inherently support PoE, just signal data. There are a few options to use. We can use a passive PoE splitter that will split power from data and then we can use the built-in ports separately. We could also use a HAT that will use the ethernet ports on the main control board and use its connection pints to convert the ethernet power to the 5-volt power needed for the board. Another option would be to build a board from scratch that uses PoE components and splits the power and data.

A passive splitter just takes a single ethernet input and splits the signals into data and power. This is the simplest solution, but also the most limited. We might even consider an active splitter if we need to include power protection and 802.3 compliance, but both are one-way solutions which means that data can only be transmitted to the system not from it. It also requires the control board to have an ethernet port which, as we've discussed, is not common.

An alternative is to use an interpretive device like a HAT or a shield. Similar to the servo control board, an extra device that uses the GPIO pins to interface with the main control

board can perform the necessary functions to enable two-way PoE through the interpretive devices as well as provide power protection. For the Pi, there is PoE HAT that uses the ethernet port on the Pi 3, so it mainly manages the power protection function. For Arduino, there is an ethernet shield which has its own port, so it manages the power-data splitting. Both devices use GPIO pins on their respective boards and can be stacked with other devices using a pass-through pin connection. However, the Pi HAT is limited to the Pi 3 B model and up so it limits our choice of main control board. The Arduino shield requires the PoE module which is an additional part that will inflate the budget and it is also third party technology that doesn't have open source support.

The final option is to build a custom PBC board with PoE capabilities. Since the board is custom, we could also wrap the PWM servo control and power into the design as well. This option would be good because we could build a single board with all the functions that we need. However, it provides an issue for the consumer. When the board needs to be replaced, the customer would need to rebuild the custom board again. This can be especially true for a product where the function is highly specific and has low demand.

Before deciding against the implementation of Power over Ethernet for this project, the design for it was thoroughly researched and created. Our original goal to implement Power over Ethernet consisted of the Tp-Link's 802.3af Gigabit PoE injector, two Ethernet cables no longer than 328 feet, and a 48V DC power input wall plug. The PoE injector solves the issue with standard laptops not being designed to support PoE out of the box, by creating an intermediary connection it will output Power and Data into one ethernet cable before finally reaching the PTZ camera. The injector would supply up to 15.4W of power depending on the amount the LLDP calculated was required for the connected camera. A minor issue that occurs is the loss of portability that occurs when needing a walled power source to power the injector alone.

In the final design, we chose not to use PoE because it would strain our budget and was incomparable with the desired technology. Most high-quality cameras don't support PoE and building an interpretive device for the camera seemed to be too challenging for the scope of this project. Therefore, we chose to use the wi-fi option instead because even though a hard-wired connection would provide consistent connection, we would still need wi-fi to broadcast the stream to other users anyways.

Software:

The PTZ camera project software consists of three subsystems. The diagram in Figure 38 shows how the remotes software, interact with the laptops command center to finally interact with the main control board. The laptops command center and the remote's software both are hosted on the laptop. The three subsystems run in tandem which allows the subsystems to stay in sync. In detail, this grants the physical camera system the ability to make precision movements as not long after the remote sends them, and the computer handles the transfer.

As the remote sends the command data to the computer, the computer's command center sends it over to the main control board for interpretation while also processing the video data being received from the camera via USB. The main control board interprets the data received based on pre-written code stored in its RAM. The translated commands are then passed to the speed controller which directly controls the tilting motor and the panning motor. Finally, at the end of the desired movement the status and position of the motors are then reported back to the main control board.

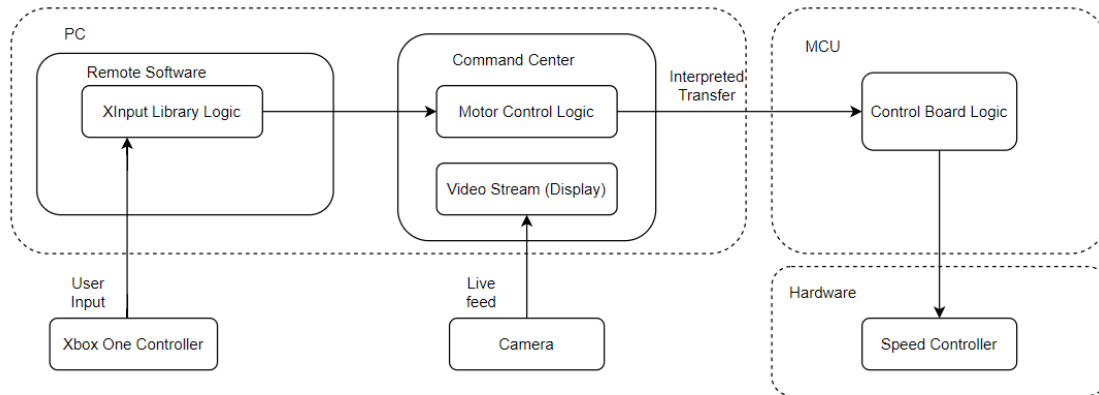


Figure 45: In-depth Software Diagram

Remote Software:

The remote software takes advantage of a Windows API called XInput which enables applications to receive input from an Xbox Controller on Windows. The XInput API consists of functions that allow you to retrieve the controller state, this alone is the foundation of interpreting the controller's commands. Retrieving the controller's state is done very often to keep track of changes which are later transformed into motor movements.

Command Center:

The command center is a custom written software application that is responsible for handling the video data being received from the camera and transferring the remote software output over to the micro control board. The video data is transferred to the command center via direct USB connection. In the command center the video data is displayed live and has the option to be recorded for later use.

Main Control Board Software:

The Control Board Software is a custom API that translates the Motor Control logic in the command center into code the speed controller will be able to use. Since the motor controller will be an Arduino or Raspberry Pi the translation of code will be relatively easy as C and C++ are closely related. The control board software will also keep track of the position of the pan and tilt motors and have safeguards in place to stop the arm from moving out of bounds.

Windows API

As mentioned before this project takes advantage of the Windows API which is a library of functions used freely for developers to interact with their projects. The API is a collection of commonly used functions when programming interfaces under the Windows operating system. The API receives support from a software development kit, the Microsoft windows API, which provides documentation and descriptions based on the Windows API.

The API is responsible for handling a lot of serial connections and applications with respect to the Windows operating system. Even though API's belonging to this family lose support they are still stable enough to be used in development.[30]

XInput Library

The XInput Library is a cross-platform API to be used on Xbox and Windows. When on windows it is provided as a DLL (Dynamic-link library) file directly installed on the operating system. There currently exists three versions of the XInput DLL, they each provide slightly different functionality and should be chosen based on the versions of Windows needed to be supported and the programming language to be used.[31]

XInput version 1.4 is packaged as a part of Windows 10 and should be used when building Universal Windows Apps, which are applications that can be used across the compatible Windows Devices, such as PCs, tablets, Xbox One console, and smartphones. Version 1.4 is also the only version that can be used in C++ or DirectX Window Store apps. XInput version 9.1.0 is a part of Windows Vista, Windows 7, and Windows 8. The 9.1.0 version should be used when the desired desktop app would be run on those versions of Windows while using basic library functionality. Finally, XInput 1.3 is considered legacy as it used to support Windows Vista, Windows 7, and Windows 8 before it was dropped. It is still possible to run version 1.3 to support down-level versions of Windows that aren't supported by version 9.1.0.

DirectInput Library

The DirectInput Library is a part of Microsoft's DirectX API that is used to process data from a multiple of controllers. The advantage this library has over others is its ability many different types of controllers even ones not directly native to the Windows platform such as the PlayStation 4 controller. DirectInput works by enabling an application to retrieve data from any kind of input device, through a process known as action mapping. Action mapping allows an application to pull data from an input device without having to know anything about the device itself. The biggest drawback to using the DirectInput Library is its drop of support and replacement by the XInput Library. The use of DirectInput is still highly common among developers as it is a powerful software needed to connect many different types of devices to Windows.[31]

Comparison

In the case of choosing between the XInput and DirectInput libraries the former is the better choice. The controller of choice for this project the Xbox One controller is compatible with both these libraries however, the XInput library was made to support all features available to the Xbox one and later controllers.

While the DirectInput library can handle basic interpretations from the Xbox one controller it can't implement some of its newer features such as querying for the state of the controller, vibration effects and sound input and output functions. However, if the PlayStation 4 controller was used the only option would be the DirectInput library.

Library	Multiple Controllers	Supports PS4 Controller	Supports Xbox controller	Note
XInput	Up to 4 at a time	No	Yes	As of 2021, the XInput library is still supported and maintained.
DirectInput	1	Yes	Yes	No longer maintained or supported for troubleshooting

Table 12: Comparison of API libraries for the controllers

Programming Languages

When researching the requirements for the software going to be written a need to select an appropriate programming language arose. As most problems can be solved in many ways so can programming problems, however in some cases there is a language that is suitable to solve the problem the best. There are three software systems that are a part of this project so one will require a properly selected language.

The primary choice of language for the workstation software should be easy to implement and maintain, fast enough to keep up with the remote audio-video data, and well-supported enough to provide the media and graphical software libraries necessary to complete the project with minimal reinvention of the metaphorical wheel.

A clear choice satisfying these parameters, especially in the context of the Windows operating system, is C++. The C++ API is tightly integrated with the Microsoft Windows API. Furthermore, for the remote controller software system, the XInput library is the best-supported and well-documented library for controller interaction on Windows; it is also written in C++. This makes a very good case of using a pairing of C++ with the XInput library.

The second software system is the Command Center. This is the central hub of the workstation software that coordinates controller input, motor controls, and camera data flow on the workstation computer. Since the grounds for using C++ in the project are well established, the logic for the command center may also be implemented in the C++ programming language; however, an alternative for the graphical interface, which may shave off key development time, is NodeJS. NodeJS is basically a desktop version of JavaScript that runs outside of the web browser. There are many technologies that allow easy-to-design graphical interfaces available as simple libraries in NodeJS. Therefore

NodeJS will also be considered in the implementation of the command center if the extreme flexibility of C++ is not required.[26]

Finally, the software on the main control board. As this is an embedded system, a typical choice of software platform is ordinary C. There are also options for C++ in an embedded system, but the toolchain that is best-supported on Arduino, the Pi family, and especially other, less well-known MCUs is the GNU C toolchain.

Software Flowchart:

In figure 39, the general overview of the software flow is awaiting commands from the workstation. The device awaits a power-on signal, and then listens for input related to powering off, audio and video sampling, PTZ actions for the motor, and responds to them appropriately.

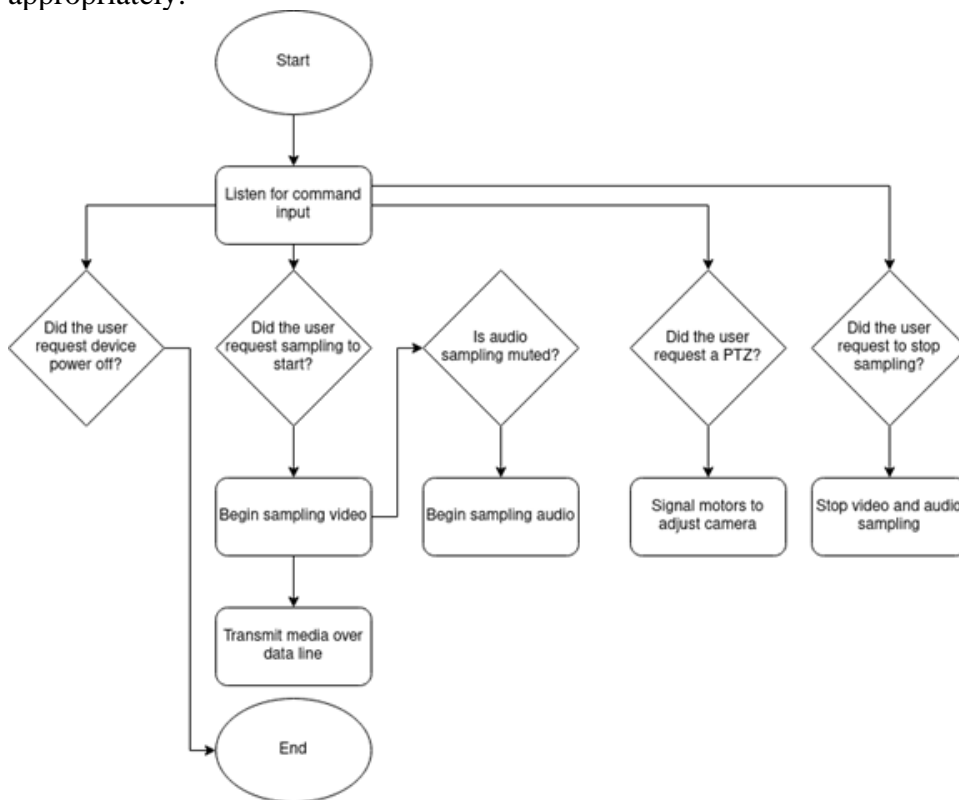


Figure 46: Software Flowchart

Controller Software Design

As specified before we decided to use the Xbox one controller as the remote control for the Pan and Tilt functionalities of the camera. A relatively new model of the controller was chosen only just released on November 10, 2020. The official name of the controller is the Xbox Core Controller which is now the primary for the controller for the Xbox One, Xbox Series X and Xbox Series S. The first step in controller connection is to open a serial port or transfer of data between the controller and computer programmatically. To access the Windows API in C++, '#include <windows.h>' is written inside the cpp file which allows use of functionality that can connect devices to the operating system serially. For everything else related to pulling the data from the controller the XInput library is imported

using `#include <xinput.h>`. To make sure the controller is connected before translating button clicks and joystick movement the state of the controller is frequently pinged, using a function called `XInputGetState()`. `XInputGetState` takes two parameters `dwUserIndex` and `pState`, with a single return value. The first parameter `dwUserIndex` takes a value from 0 to 3 and determines how the value maps to indicators on the controller, this is useful when connecting to multiple controllers the limit being four in total. The second parameter is `pState` which is a pointer to an `XINPUT_STATE` structure that receives the current state of the controller. Inside the `XINPUT_STATE` struct is the type `DWORD` `dwPacketNumber`, the type `XINPUT_GAMEPAD` `Gamepad`, and the `XINPUT_STATE` pointer. The `dwPacketNumber` states the packet number which indicates whether there have been any changes in the state of the controller, if the number has changed in sequentially returned structures then the controller state has changed. Finally, the `Gamepad` structure contains the current state of the controller using a byte to represent if a button has been pressed. Inside the `GamePad` structure is the `WORD` type `wButtons`, the `Byte` type `bLeftTrigger` and `bRightTrigger`, the `SHORT` type `sThumbLX`, `sThumbLY`, `sThumbRX`, and `sThumbRY`. The `wButtons` variable is a bitmask of the controller's buttons digitally, when a bit is set it means the corresponding button is pressed. The `bLeftTrigger` and `bRightTrigger` variables are responsible for the holding the current value of the left and right trigger analog controls. The `sThumbLX`, `sThumbLY`, `sThumbRX`, and `sThumbRY` variables control represent the position of each thumbstick in both the x-axis and y-axis; each holding a value between -32768 and 32767. By calling the `XInputGetState` function frequently we can track every time there is a change in the controller. After the data is received and processed it can be passed on to the microcontroller for motor control.

Servo Control Overview

The data received from the controller will be translated using simple logic to work with a Raspberry Pi Zero library in python called `gpiozero`. The `gpiozero` library helps the programmer write custom code to control the GPIO pins. The library takes advantage of the many common uses of GPIO pins to provide modules which simplify the control of many hobby parts such as Servos, Motors, and LEDs. The `gpiozero` contains a class named `Servo` that simplifies the forward and backwards controlling of the motor through code. The `Server` module extends the class `CompositeDevice` which extends the class `Device`. The `Device` class represents a single device of any type; GPIO-based or SPI-based and more. The classes hierarchy is shown below in Figure ??? the light blue represents concrete classes while the dark blue consists of concrete classes. `Device` functions as the base class of the device hierarchy and is used to define all the basic device services. The `CompositeDevice` class represents a device composed of multiple devices like H-Bridge motor controllers and HATs. Finally, the `Servo` class represents a PWM-controlled servo motor connector to a GPIO pin. After the proper connections are made between the raspberry pi and the servo, code can be used to move the servo between its minimum, maximum, and mid-point positions. For example, to create a servo object the constructor `Servo(pin#)` can be called which returns a servo object and the `pin#` would be the number of the pin used in the connection. With a servo object simple functions like `servo.min()`, `servo.mid()`, and `servo.max()` are used to position the servo at its minimum, maximum, and mid-point positions. To position the servo more precisely the `value` property of the servo

object is used, for example the command `servo.value = 0.9` would move the servo to that position.[25]

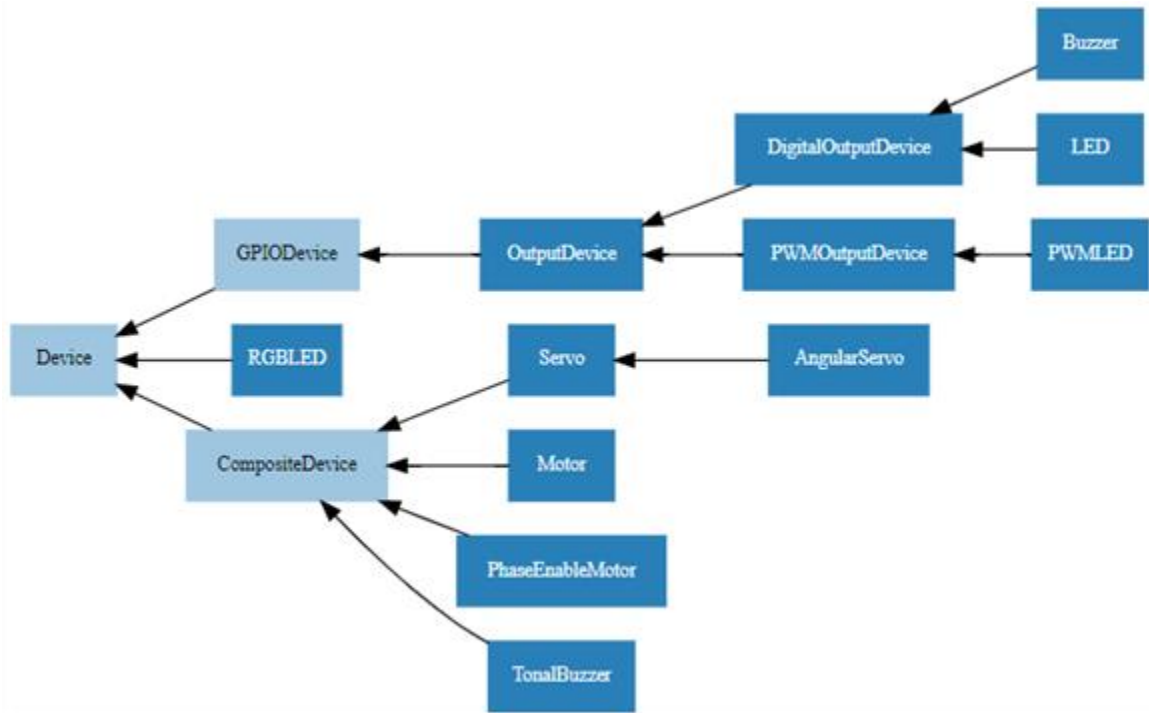


Figure 47: Partial class hierarchy diagram

Servo Control Design

Servos offer precise position control in a closed loop system. To take advantage of the remote control and the motors together some programming manipulation was needed. The data from the controller is first parsed for information that signifies changes such as button clicks or joystick movement. For this project the purpose of the controller is to adjust the pan and tilt of the housing by moving the joystick in the direction desired. Control of the system can be accomplished but for a more complete feel both joysticks are enabled for control. When a joystick on the controller is moved a number correlating to where it is two numbers representing its position on the x and y axis are then proportioned to the range of pan and tilt system respectively. The joysticks's range is anywhere between -32768 and 32767 which is converted to the servos range. The information on the range of the servos may be included in the servo specification but this isn't always the case. To calibrate the servo's, range, the pulse width for the minimum and maximum rotation must be set. The servos take PWM data signals so either a hardware or software generated signals can be used. Initially a software solution will be used but a hardware solution will be implemented instead if more precision is needed. To find the minimum and maximum of the servo set $\text{maximum} = (2.0 + \text{degree}) / 1000$ and $\text{minimum} = (1.0 - \text{degree}) / 1000$, then pass it into the Servo constructor respectively. The degree variable will be adjusted until the maximum and minimum rotation of the servo is found. After finding the max and minimum pulse width values the `servo.value` attribute can be used to move the servo arm precisely to its maximum and minimum positions. Finally, the range of the joystick can accurately be

adjusted to the range of the servo and the joysticks movements will control the servos as desired.

Graphical User Interface

Another essential part of the software's design is the graphical user interface. A graphical user interface (GUI) is a user interface that allows users to connect with electronic devices through icons and more. GUIs were the solution to difficulty of learning to use command-line interfaces which call for more typing with a keyboard. GUI icons in general perform an action as an effect of interacting with the graphical element of the icons. GUIs are not only used in computers but with many mobile devices such as gaming devices, smartphones, and household appliances. Two major goals when designing GUIs are usability and simplicity. To make sure a graphical user interface is useable appropriate visual widgets that relate to the kind of data they hold should be used. Maintaining simplicity when designing a GUI should also be considered as the more complicated and clustered a GUI, the lower the user experience.

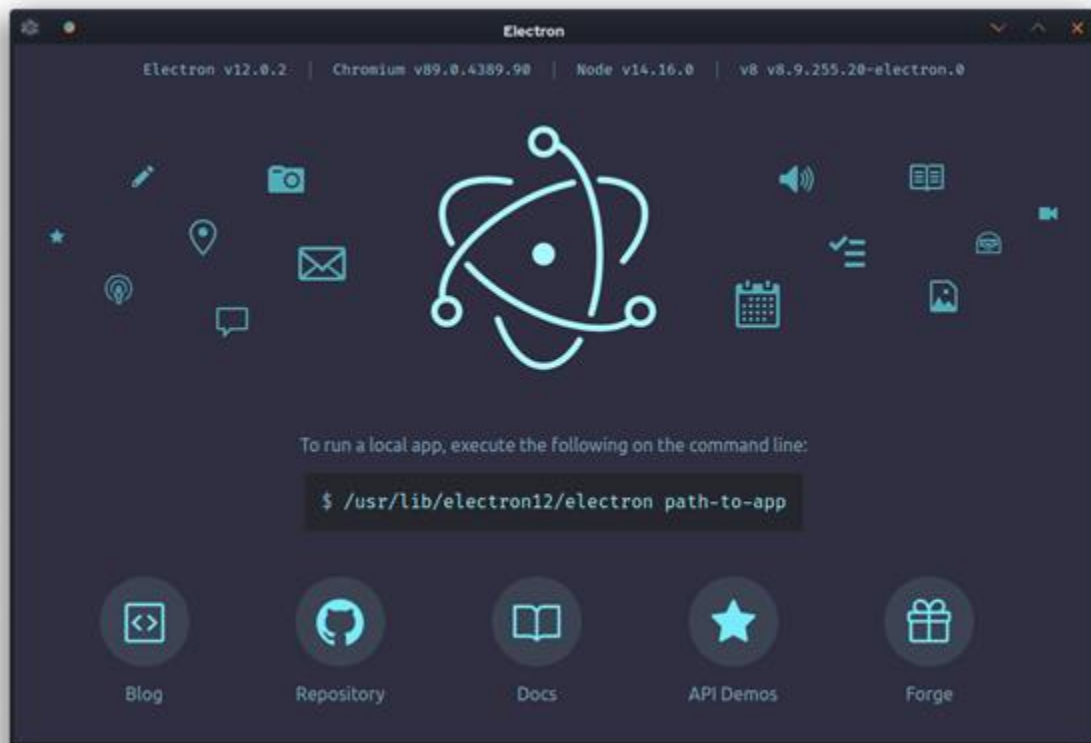


Figure 48: Electron

This image is licensed under the Expat License, free to use.

Node.js and Electron

The command center is the name of the GUI software customized purely for this project. It's able to display the received video from the camera and encapsulates controller and servo software. For the front-end user interface node.js was the chosen language used for the design. Node.js is an open source, cross-platform, back-end JavaScript runtime environment, its purpose is to execute JavaScript code outside of a web browser. The main

advantage to Node.js is its open-sourced frameworks such as Express.js Socket.IO, and Feathers.js which help speed up the development of applications. The choice of software framework for our GUI is Electron, the main advantage of Electron is its ability to use web technologies to develop GUI applications.

When developing a desktop application an inexperienced developer will run into many challenges as they are more complex and operating system oriented when compared to web applications. Those used to developing web applications must deal with packaging, installation, and the managing of updates. However, unlike a desktop developer a web developer will commonly have to design their applications to run on the many different browser types that browse the internet. The Electron software saves developers from having to use specialized tools for multiple platforms or the need to switch between programming languages. Electron in a nutshell is a framework that combines modern web technology with the native experience of a desktop application. Using Electron developers can create cross-platform applications using JavaScript, HTML, and CSS. The framework provides JavaScript APIs that handle the communication between different operating systems while also using webpages as the base for creating user interfaces. Electron apps are minimal web browsers with the ability to access the local file system. With Electron everything is a part of the local packaging meaning anyone using the application built by it will be using the same set of code.

Electron apps are a kind of a multitude of processes, usually a main and then a few or several renderer processes. The main process is used to run application logic and trigger multiple render processes. HTML, and CSS are also rendered as the window the user can see and interact with which simplifies development of the user interface by magnitudes.

GUI Software Design Details

The GUI for this project will be programmed in Node.js and designed with the Electron software framework. It will feature three main functions: start/stop switch, video display, and speed controls. The start/stop switch will be able to begin or halt all physical capacity of the system. The visible speed controls will be able to adjust how fast or slow the camera can move after receiving signal from the physical controller. The video display will be an extra feature that allows the user to see the current live feed natively. Graphical user interfaces made in electron feature any design that can be made in a modern webpage like menus, notifications, support of multiple platforms, and the powerful debugging tools available to webpage software.

Start/Stop Switch

The GUI will feature a start and stop switch with the control to either ready the system for use or stop all digital and physical functionality. There will be a looping function that checks the status of tasks to determine the availability of the buttons. For example, the stop function will not execute if there are critical tasks active but will be able to schedule one as soon as they are finished. The ability to click either button will be disabled and feature a grey color when functionality of either is not enabled.

Speed Control

An essential feature of the GUI will be the ability to control the speed at which the camera's arm can pan and tilt. After testing there will be a minimum and a maximum speed that the

system will not allow the user to surpass. The user can use clickable arrows to adjust the speed or simply enter the speed value as a number and hit enter to accomplish the same task. Finally, the design of this code must be efficient and adhere to certain time complexities as delays in response times is very undesirable.

Video Display

The final feature of the graphical user interface is a video display that includes a resizable and pop out screen. The video display on the GUI is an extra feature that will enable the user to fully work from the software after the design is full initialized. The PTZ camera system will also use OBS streaming software for video recording but once the recording has started the user can work entirely from the GUI. To implement our video display design, we are relying on an HTML5 API called *getUserMedia*, this allows us to pull both live audio and video from a connected camera device. After to define the initial window size the *webkitGetUserMedia* function is used to set the minimum/maximum width and height.

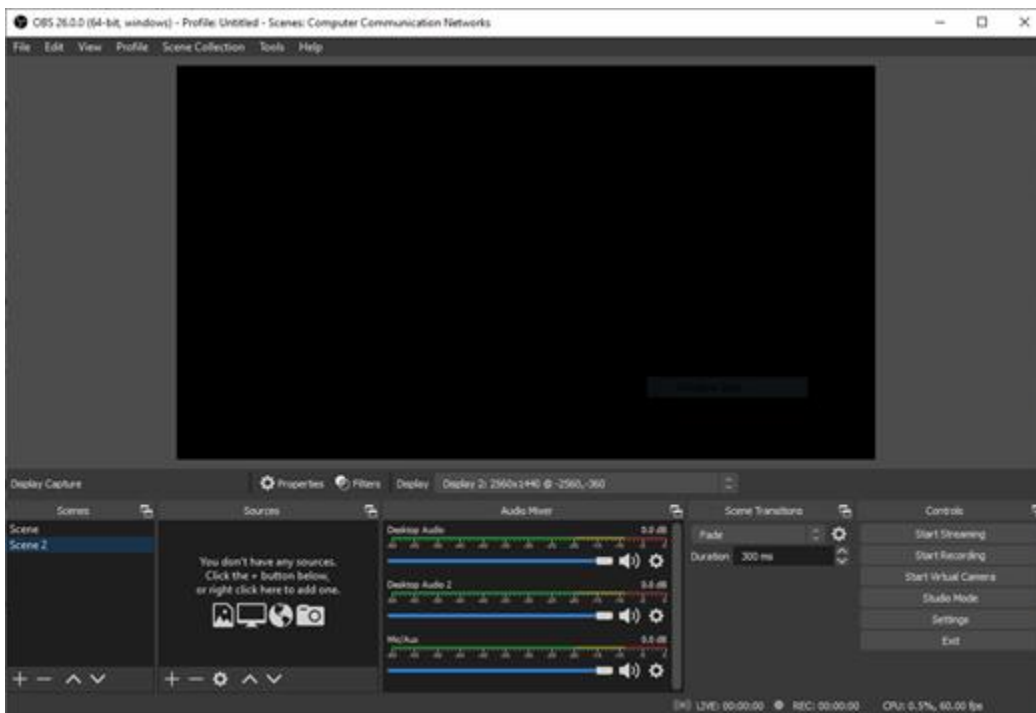


Figure 49: OBS Graphical User Interface

Open Broadcaster Software

The Open Broadcaster Software (OBS) streaming software will be used to record and store the video and audio that is coming directly from the camera. OBS is a free open source live video production software, it can live stream and record video and audio. OBS is supported by large community of developers worldwide, its cross-platform and designed using the Qt webkit. The OBS application allows for real-time device capture, encoding, recording, broadcasting, and scene composition. It transfers data using the Real Time Messaging Protocol and can send to any RTMP enabled destination. For video encoding, OBS can use x264 free software library, x264 free software library, Intel

Quick Sync Video, Nvidia NVENC and the AMD Video Coding Engine to encode into the H.265/HEVC format and the H.264/MPEG-4 AVC format. OBS also supports the addition of plugins which extends its functionality to include features such as VST plugins and stream deck controls. Finally, the PTZ optics OBS plugin provides features that allow the control over PTZ cameras directly from the OBS software.

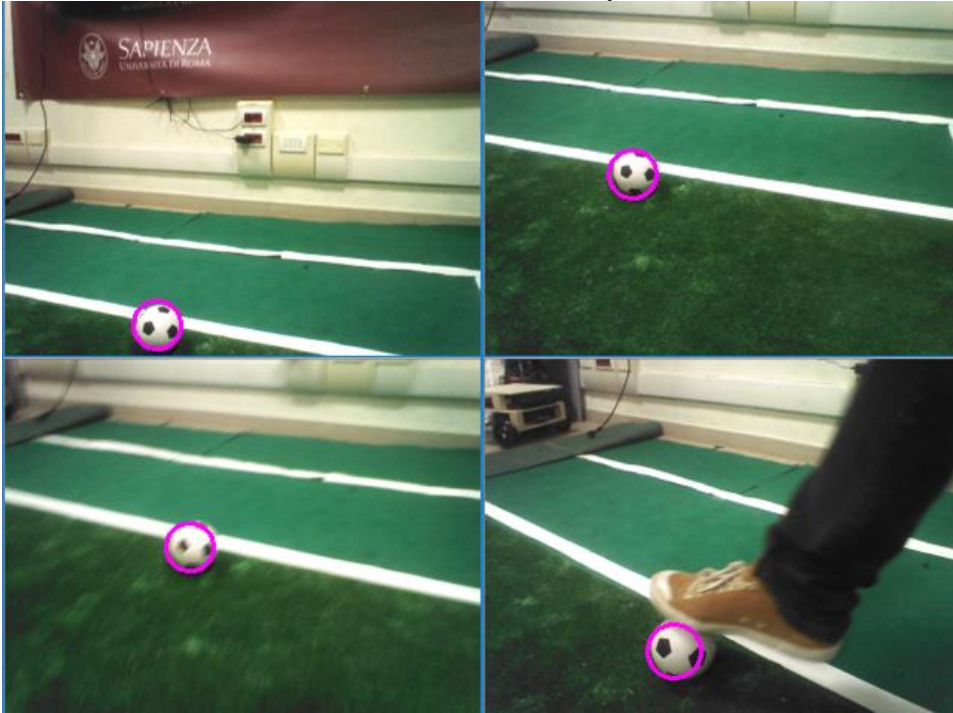


Figure 50: OpenCv detecting a soccerball.

Tracking Software

One of our stretch goals is to prototype, test, and design camera tracking software that would identify a given object and follow it through motion. Object tracking works by first taking an image still of a video and identifying the parts of the image depict the same object but in different frames. Tracking can be broken down into subcategories such as similarity measurement, correlation, correspondence, matching/retrieval, and data association. Tracking is also trying to learn the model of the object's appearance and motion. By knowing the appearance of the object, the tracking software can track that object independently of anything else in the image frame. If the motion of the object is known the software may be able to make predictions of the directory of the object. Luckily for developers today there exists much open-source software that handles all the algorithms and theory for us.

To preform motion tracking requires a field of study called computer vision. Computer vision is when we take visual input, such as photos or video, and we let the computer decide what is in the image and how to respond according to some set of features. In order to do this, we need to define the input, define the features, find the objects, and then preform a response depending on the objects and their state.

The input is defined as a static photo or video. In our case we want to process video. Since video is a set of images playing in sequence, we can think of this as processing a series of photos. This extra consideration actually increased our processing requirements because instead of needing to just run on a limited set of photos, we are running the program on 30 photos per second, which comes from the frames per second. Of course, we can reduce processing requirements by lowering the sampling, but this would make the result less reliable in the same way that a video with fewer frames per second looks lower quality. In addition to the processing, we also have to consider that the images in each frame are not independent of each other. The succession of photos shows a discrete version of continuous time so the actions in one photo are directly related to the action the next. This is important to keep in mind for the purposes of tracking because when we identify an object in one photo, we can track its state as long as it is in the next phot and we can assume that the states are related in a cause-and-effect manner.

A large part of computer vision is defining features. Features are the parts of an object that make it distinct form all the other objects. In the context of computer vision, this means the pixels in the image that define a separate object. Computers only see in pixels not the context that the pixels are in. For that we use features. Features can be as simple as template matching, or they can be complex vectors that define the color, gradient and relationship to other pixels. Some common features are edges and shadows. Most computer vision features use some sort of edge detection because they define solid shapes and give us some depth information. As mentioned previously, features need to be processed as vectors or tensors of pixels. This is because we want to look at patches of pixels as they change because a single pixel is a light on your laptop, but an array of pixels makes an image. Looking at patches of the image at a time lets us see how the pixels change across the image, so we can also see how the relationship between the pixels change. We also want to look at the equality of the features. A common traditional computer vision approach called SIFT says that features should be unique, unambiguous, and should be scalable. Uniqueness means that no features in the same image should be confused for the other so we should avoid duplicate features if necessary. Unambiguity means that features should not be too large because large features have too many data points to match to and can cause under-fitting or over-fitting based on the program. Scalability menas that features can be transformed, such as rotated, shrunk, or stretched, and the program can still recognize the features as being the same.

With these features, the program can do abject detection through classical methods or machine learning. Most of the computer vision libraries use some form of machine learning, though there are some classical models. Machine learning requires the program to take in training data, make determinations about the object classification, is told if it is correct or not, and then the program adjusts it parameters to guess correctly next time. This is specifically called supervised learning. Once the program is trained, we can use it to identify objects such as people or a ball and track its movement based on its position frame-to-frame. Even if we don't build our own machine learning model, we can do the training to make our tracking accurate to our specific needs.

The final step is for the system to respond to the changes in the frame. Since the object will move, it can move out of frame, so it is important that the servos can respond quickly and efficiently to the change in frames. While the machine learning model is great, it is also processor heavy and may require a high-grade processor like an 8 core CPU or a GPU modification. In this case it would be best to use one of the classical models in one of the CV libraries.

The Python programming language has a thriving set of tools and libraries for use in the computer vision community. Open CV, SimpleCV, and Imutils are some of the leading image processing, and computer vision libraries available for use currently. Open CV is a widely known and open-source computer vision library it is also referenced and used in most of the other libraries, even SimpleCV, and Imutils use it. OpenCV includes modules for video analysis, image processing, object detection, camera calibration, three-dimensional reconstruction, gesture recognition, mobile robotics, motion tracking, facial recognition, and augmented reality.

However, there is a steep learning curve when learning to use OpenCV and that is out of the scope of this project. Instead, we will be using SimpleCV an open-source framework for creating computer vision applications. SimpleCV provides access to OpenCV and other high-end computer vision modules without the need to learn about the intricate details such as color spaces, bit depths, and matrix versus bitmap storage.

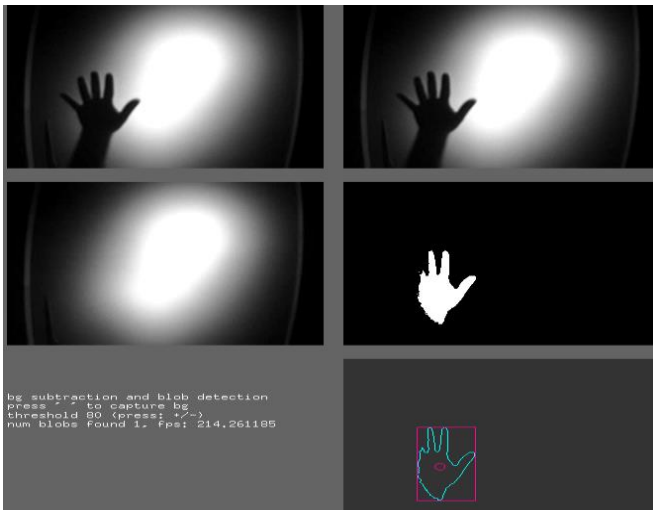


Figure 51: SimpleCV image manipulation using stretch function. Attribution-ShareAlike 4.0 International (CC BY-SA 4.0).

SimpleCV Implementation

As SimpleCV is very advanced software our first prototype was to simply implement a white ball tracker. Using SimpleCV this advanced feature only takes a few lines of Python code to implement. It's done by first by importing the SimpleCV module and then creating a display object. In python this is simply done by capturing the returned value from the SimpleCV.Display(). The camera being used to track the object must also be initialized using SimpleCV.Camera() and finally set the display mode to normal. A color distance is

also useful which will push all the colors the furthest distance away from black as it is a white object we are trying to find. The return value of `colorDistance (SimpleCV.Color.BLACK).dilate(3)` accomplishes dilutes the colors away from black and returns the new image as a return value, the `dilate(3)` function expands the pixels in the image which helps the computer differentiate between colors. Next, we use the `stretch` function to filter to darkens any pixels out of a set range to isolate the object. Finally, we can show the isolated object in the image with a circle to show that the software was able to successfully identify it. SimpleCV also supports motion tracking naturally so the circle will follow the object as it is in motion.

Automated Positioning

Another one of our stretch goals is to implement a few pre-set automated positioning buttons. The automated buttons would work by positioning the camera to stored coordinates on command at the speed desired. The positioning functionality would be useful for religious, educational, and active events. For example, say camera is set to track an object that has moved across a room, to move back to another area without to physical move the camera further enhances the total automation of the PTZ camera system.

The design for this system will be done using the same Raspberry Pi Zero library used in the initial servo control design and button functionality from Electron. The `gpiozero` module is responsible for controlling the servos in our PTZ camera system so the design to automate buttons will rely on functions from this library to interact with buttons built in Electron. As Electron uses web-based software to create graphical user interfaces the implementation of buttons will be done using HTML code for the graphical interface and JavaScript for the underlying commands. First the button must be able to reasonably describe the positioning it will move the user to when clicked, the custom feature to set the name and positioning of the button is essential. By design the user can set a name and add as many buttons as they need to the GUI by seamlessly pressing an addition button and writing the name in a text box. To set positioning all the user would have to do is move the camera by controller to a position and click the set button which would translate the coordinates and save it as a button.

After the button is set and ready to be used as a user clicks it code on the backend will send the data over to the microcontroller to be executed. However, before the code can be executed it needs to be interpreted in a way the microcontroller can understand. Using the `gpiozero` library the actual servos can be controlled in a way as we did before. This is done using the `Servo` constructor to imitate a servo object, after which it's possible to set the servo objects value. When the servo object's value is set the servo then moves to the specified positions. Finally, since we set the positioning when the camera was there no translation needs to be done expect giving the servo value attribute the saved number.

Workstation Graphical Interface

Overview

The workstation will run the software that is responsible for receiving the media transmission signal over WiFi, displaying a live feed of the video to the workstation user

for immediate review during transmission or broadcast, and negotiates controls of the motors for panning and tilting, initialization or stopping of camera recording, and camera zoom. Therefore, the most prominent elements of the graphical interface will pertain to the status and operation of:

- The battery level of the camera.
- The battery level of the motor system.
- Whether the camera is recording or not.
- The live video feed displayed.
- Whether the camera is panning, tilting, or zooming, and the direction.

A possible configuration of these graphical elements is presented here as a mockup interface.

Figure: A graphical layout example.

The battery level of the camera will be transmitted from the camera to the workstation computer over the wireless interface. After the packets are decoded to indicate battery life, the amount of battery life left will be represented as a percentage for accuracy reasons, and as a graphical ideal battery image split into three sections. All three sections being lit green will be understood to mean the battery is fully charged. Bars are removed from the display as the battery level decreases until there are no bars left. An empty battery should be displayed as red. This will also be the generic definition of a battery life indicator for the scope of this section. The location of the battery bar is prescribed to be the top right.

The battery level of the pan-tilt motor system will be transmitted from the motor system to the workstation over a wireless interface. After the packets are decoded to indicate motor battery life, the amount of battery life remaining will be represented as a percentage, and as an image of a gear with a generic battery life indicator as described above. This indicator will be displayed in the top right of the display, to the right of the camera battery indicator.

The camera-recording indicator will be a standard flashing red circle pulsing at 1 Hz when the camera is in fact recording. To the right of the circle will be the red text “REC” to reinforce the idea of recording. When the camera is not recording, the circle will be a dark grey color, and the text will be absent. This indicator will be in the top left of the display.

The live video feed will be in the center of the display. The media being received from the camera over wireless will be displayed in real time as a sort of preview to the media technician operating the equipment. If no media is being transmitted at the time of rendering on the graphical interface, a lightly greyed-out box should be displayed instead of the content.

The pan, tilt, and zoom indicators show whether the motors are currently panning or tilting, and when the camera is performing digital zoom. The pan and tilt features will be represented as small arrows in the arrangement of the cardinal directions. When they are

not active, they will be a dark grey; when the pan or tilt features are activated, the arrow in the direction of the pan or tilt will be colored a dark gold for the duration of the motion. These controls will be located in the bottom left of the interface on the control panel area.

The zoom indicator will be located in the bottom left hand corner of the interface. It will be represented by a magnifying glass on the left, followed by a plus sign and then a minus sign. The plus or minus sign will be dark grey when inactive; when activated, the sign color will change to a dark gold.

As is typical for a desktop application, the program will use a standard top window decorator as prescribed by the operating system desktop environment, containing the title of the application, and the minimize, maximize, and exit buttons. The behavior of these buttons will conform to what a typical user would expect. The title of the application will make it clear that this is the camera control command center.

Disability Accommodations

The colors of the graphical interface will be neutral in tone, and attempts will be made to limit the number of green elements in order minimize the effect on colorblind users. Red will be used instead of green in this case as it is widespread to use red to represent the fact that a camera is recording.

The font size will be adjustable out of the view menu in order to accommodate users with limited eyesight or other vision impairments. If the feature is supported by the host operating system, high contrast mode can be enabled in order to provide greater visibility between the text of the interface and the background of the program in order to provide a visual distinction between the two aspects of the interface. On Windows, this generally juxtaposes an electrified purple color with a muted grey.

A stretch goal for this project would be a voice-controlled aspect of the interface. This would allow the workstation microphone to be enabled and basic aspects of the software, such as file-saving plus starting and stopping recording, to be done using voice recognition libraries. This would enhance the usability by handicapped operators, but also allows more hands-free operation for everyone. This goal has limitations in that a fast-paced action camera target may not be easily controlled as rapidly as necessary through voice recognition. The controller is a superior option for control because of this, and so the voice controls would be a secondary method only.

Finally, a text-to-speech interface can assist users who are hard of hearing. Menu items that are highlighted or controls that are utilized can be read aloud using TTS (text-to-speech). This has the additional benefit of making what operations are being performed in rainy or foggy weather when visibility is limited.

Menu Bar

The menu bar located at the top of the program window will have the following menu entries:

- File
- View
- Help

Covered below are the actions that will be stored in the respective menus:

File

1. Show Recordings

This will open the directory where the recordings are being stored in the user's file manager.

2. Settings

This will open a secondary settings window that will allow the user to configure various options about the behavior of both the graphical interface, file output, and also related to the behavior of the camera.

3. Exit

Exits the program, writing back any important user data back to disk before exiting.

View

1. Fullscreen

This will make the users' window take up the full screen in order to better show the content.

2. Show Overlay

This will toggle whether the indicators such as battery life and pan-tilt-zoom controls are shown.

Help

1. Help Portal

This will provide the user with brief documentation on how to connect the hardware devices to the workstation, and also on how to use the software to record media.

The response time of the graphical elements should be rapid, with the changes occurring within half a second on the computer with the hardware as described in the requirements and normal load placed on the system.

Operation Manual:

Initialization:

This system has 3 parts: the camera, the motors, and the laptop control. All parts are connected via wi-fi and requires a continuous wired power connection. Each part will come

with its own power cable that needs to be connected before using the product so that the system can power up. Once the power is on, we need to establish the wi-fi connection for each part. The laptop will connect the remote to the wi-fi automatically. The camera will connect to the wi-fi through the settings panel and transmit video to the user interface which will run on the laptop. The motors connect to the wi-fi via the Pi Zero. To access Pi, use the user interface on the laptop to set the wi-fi network. The user interface on the laptop should be able to find the Pi device and store the IP address for the controller. The camera will use the built-in app to transmit the video, so the laptop just needs to run the app through the user interface to stream it. The controller will be connected to the laptop via wired USB connection or a wireless wi-fi connection similar to the camera. Then, open the camera control dashboard on the system interface.

Once all the parts are connected, we can set up the tripod. Mount the camera to the motor system with the screw in the base. Attach the motor system to the tripod using the standard tripod mounting latch. Then connect the power cables through the holes in the housing unit. Allow enough slack to allow the camera to pan fully. The best way to do this is to test the camera pan function and see how much slack the wire needs. Use one of the rubber stays to secure the power cable to the tripod pole to keep the slack. As you raise the tripod to the full height, attach more rubber stays at 2 foot intervals to keep the power cables from swaying in the wind and causing damage to the cable.

At this point, it is best to test the camera feed. This will help ensure that the feed is stable and that you have the view that you want. The camera stream should run through the user interface on the laptop. Use the controller to test the pan and tilt functions to ensure that it is connected and that your camera has the correct view of the scene. If you need to adjust the camera's perspective in any way, please lower the tripod to below 6 feet and move the tripod base. Moving the tripod while at full height will be dangerous and ill advised.

Features:

The laptop and remote control are what provide the main functions of the system. The remote controls the pan, tilt, and zoom functions. Using the right-hand joystick, the user can control the pan and tilt functions. By pressing the joystick up the user will tilt the camera up and, likewise, pressing down will tilt down. The degree to which you push the joystick will determine the speed of the tilt. The left-to-right controls will control the pan function in a similar fashion. The left joystick will control the zoom function through the up-down direction similar to the tilt function. The laptop itself will control the streaming and connectivity functions. The connectivity functions were discussed previously. To stream, use the system interface to easily stream through the UI. Through the laptop app, OBS is used for sending out the live feed to other users. All OBS features should be available through our custom user interface for ease of use and all communication between the camera stream and OBS are handled within the interface. You can also use the implemented automation stretch goal features through the custom user interface.

For the auto-tracking feature, press the auto track button. The interface will prompt you to make a window selection to identify the subject you wish to track. Simply place the cursor by the subject and drag it on the screen. A box will appear. Make sure the subject is fully

in the selection box. After confirming the subject, the feature should allow the camera to follow a person as they move using computer vision. It's important to make sure that the camera is not too zoomed in on the subject because if the subject moves too fast, they will leave the frame and the auto-tracking will fail. To end the auto-track function, simply click the end button on the screen. An alternative to focusing on a player in a sports event, it may be best to focus on the ball. This function will require either a powerful computer processor or an external GPU to handle the computer vision computation. It is best to test this functionality with your desired computer beforehand to make sure that it has adequate processing power. If you require an external GPU, use the settings tab in the interface to set the processing device by its access port and device type. As always, make sure your computer specifications can support the functions of this system.

The record motion function captures the control information you use to move the motors and replays the same control information again for perfect replication of a camera shot. For the recorded motion function, press the motor button to record the motion sequence of the motors and then press again to end the recording. The system will save all the controls you made during the motion recording session. Click the play button to make the system replay the motion sequence it stored. You can either select the loop setting or the play once setting. If you only want to play the motion sequence once, the system will do the motion sequence and return to the dashboard. If you choose the loop setting, the system will replay the motion sequence until you hit the stop button. As a stretch goal on top of this, we will try to store multiple motion sequences for easy reusability. In this case, the motion sequence recording will ask you to name the recording and store in a motions tab where you can select previous recordings to use.

Maintenance:

As with all things, our system will need maintenance and care. While the system is made to be weather proof, it is still possible for parts to damage from use. To replace parts, simply disassemble the housing unit by unscrewing the parts from top to bottom. The wires are simple to unplug from parts that need to be replaced. If you need to replace a wire, you need to also replace the rubber stopper that seals the hole in the housing unit. All parts of the system were chosen for both performance and their commercial availability, so finding replacement parts is as easy as googling the name. This makes it easy for a customer to fix the system without the need to order custom parts. The only custom part is the housing unit. However, the 3D printing file is easy to use and the customer can simply use a 3D printing service to make a new unit using the same file. Also, the parts are easy to disassemble. The PWM board and Pi Zero are connected using pin headers, which plug and unplug easily. The servos use the same pin header connections in the servo header style. The wires supplying the power are simple AC power converters that use a header adapter to plug into the USB port of the devices. The controller is plug-and-play, so it's easy to replace.

When you replace the Pi board or the camera, you will need to make sure it can communicate to the laptop software. For the Pi, you need to make sure that the new Pi device has the SD card programmed with the SSH software to connect to the laptop through the proxy. The user interface will handle the proxy for ease of use. If you can't reuse the

SD card from the previous Pi device, we can supply the code for a new SD card. Simply buy a replacement card and then burn the provided code onto it. The code should run at power up. If you still have trouble with the replacement device, use a physical connection via USB micro to USB connection with the laptop to directly access the Pi device. This requires choosing a device in the interface menu. If you are replacing the camera with the same version, then no extra setup is needed. All action cameras will use the same wi-fi app to transmit video and audio. If the new camera is a different model, then it may require some configuration. A different model will have its own wi-fi app that, in the basic version of our system, is not supported. However, a stretch goal is to make our interface amenable to all wi-fi cameras with a way to easily select an app through which the interface will get the streaming data on setup. Our system will not support cameras that require a hardwired data connection because of the signal drop off from long distances.

Camera Wi-Fi:

The camera transmits the media over 802.11 wireless radio. Both audio and video are encoded in a transport stream that is broadcast using a mobile radio transmitter built into the body of the camera. An antenna pickup on the receiving end, namely the workstation computer, takes the signal as an input and decodes it into the video and audio signals. This media is displayed on the live console within the workstation software.

Although the camera supports built-in video display, this feature is unused in general circumstances because of the altitude of the device. Typical use cases do not make it convenient for the user to use the built-in display, but for debugging purposes it can be useful to have the screen built-into the camera.

For reverse engineering of wireless signals emitted by the camera in the case that there is incomplete documentation of the action camera, a popular libre and gratis tool Wireshark is widely available. Wireshark allows the decomposition of all types of network packets, using the 7-layer OSI model as a frame of reference. The primary layers that we will focus on in the development of the workstation software are from layer three and above.

	Layer	Protocol data unit (PDU)	Function ^[20]
Host layers	7 Application	Data	High-level APIs, including resource sharing, remote file access
	6 Presentation		Translation of data between a networking service and an application; including character encoding, data compression and encryption/decryption
	5 Session		Managing communication sessions, i.e., continuous exchange of information in the form of multiple back-and-forth transmissions between two nodes
	4 Transport	Segment, Datagram	Reliable transmission of data segments between points on a network, including segmentation, acknowledgement and multiplexing
Media layers	3 Network	Packet	Structuring and managing a multi-node network, including addressing, routing and traffic control
	2 Data link	Frame	Reliable transmission of data frames between two nodes connected by a physical layer
	1 Physical	Bit, Symbol	Transmission and reception of raw bit streams over a physical medium

Figure 52: Types of network layers.
Screenshot from public domain content.

The libraries we intend to use to implement the workstation software support easy-to-use datagram oriented transport layer APIs that form a barrier of abstraction over the gritty details of the lower layers. Wireshark is a graphical frontend to the libpcap library in the UNIX world, implemented as Winpcap on the Windows side.

According to the documentation, many types of wireless interfaces are supported on Windows using the Winpcap drivers. The interface of Wireshark is presented here:

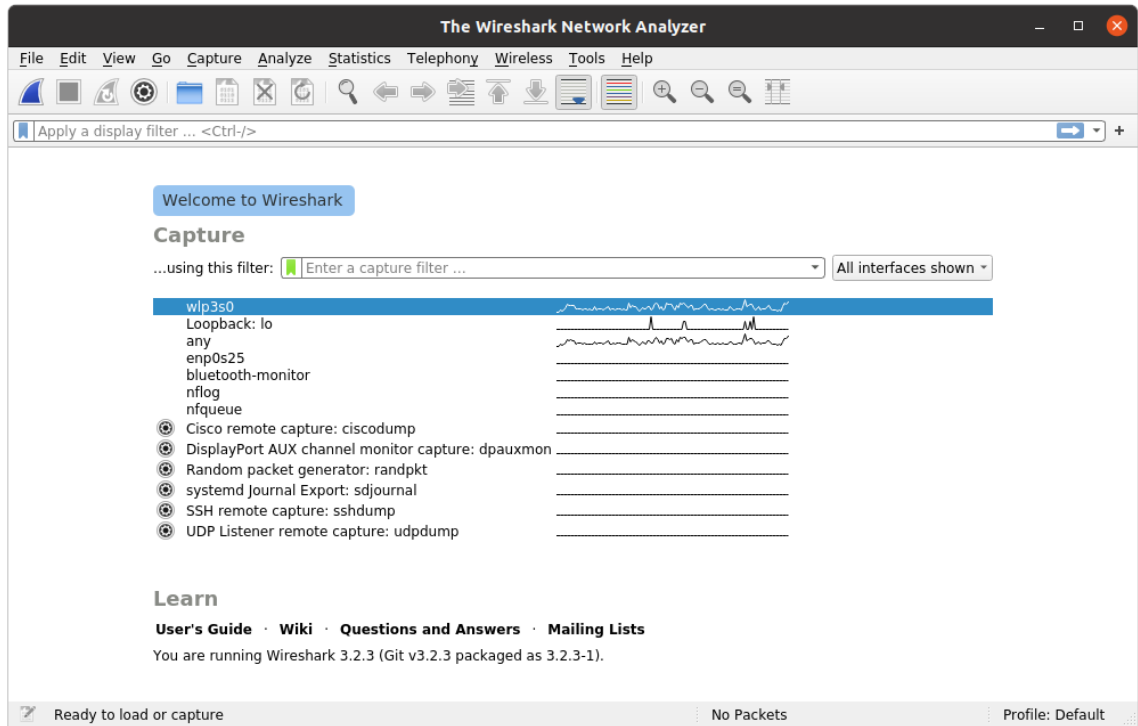


Figure 53: Wireshark interface.
Screenshot from GPL3 licenced software.

In this case, the wireless interface that the packets will arrive on is wlp3s0, indicating that the network card recognized by the operating system is using a Broadcom brand wireless radio. Packets arriving on this interface will be mapped to the appropriate network port prescribed 5th layer of the OSI model on the session layer. This will determine what port the packets arrive on. Our workstation software will be configured to listen on that destination port and move the packet into our buffer for further analysis. The most important part of the payload containing any audio or visual data, status of the camera including battery life, and changes in camera state are parsed out of the datagram and viewable in the Wireshark interface. This will greatly ease debugging and allow us to obtain a more direct visualization of the network traffic flow from our device to the workstation.

Wireshark allows logging of the data received so a track record can be created and one of our requirements can be satisfied on packet backlogs. We state that a packet backlog cannot be formed of over 1000 packets and this software can be used to verify whether transmitted packets and their accompanying media are being processed on time, or whether they are entering the buffer and being placed in a wait queue. For example, if additional media packets are arriving, and the workstation software is configured to record the media to disk, we should see without unreasonable delay all of these media packets being committed to

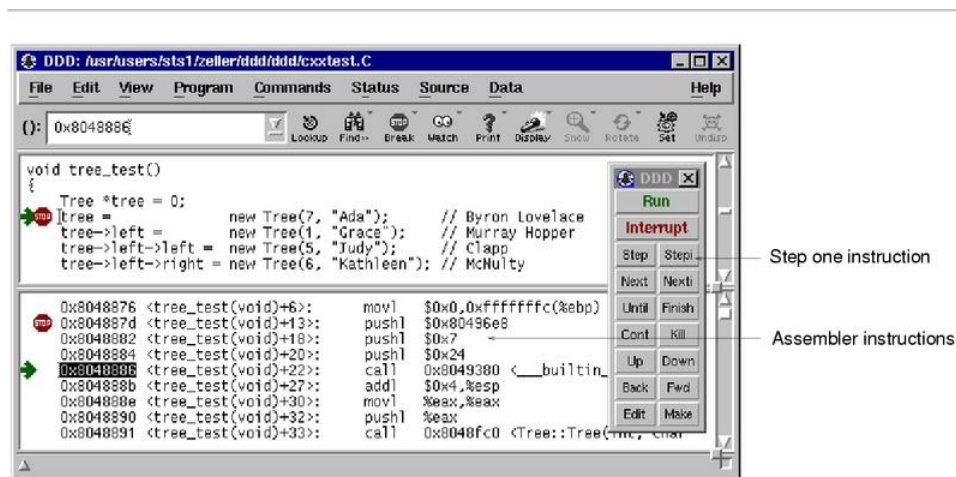
disk. If there is any indication that a backlog is forming, further analysis from the perspective of the workstation command center software can indicate whether there are excessive packets waiting in the pool.

Debugging Software

The size of a software development project is typically measured in SLOC (source lines of code). This particular metric usually omits whitespace in such a way that only syntactically significant lines are counted and redundancies are omitted. With that definition, Every software project of significant size as measured in SLOC contains bugs: many bugs. Studies over the decades have projected averages of anywhere from 1 to 25 bugs or errors per 1000 SLOC in significant software projects. Even one bug can cause undefined behavior, crashes, performance issues, or even security vulnerabilities on the end-user's computer, and with a workstation connected to the internet for live streaming, a competent software marketer strives to avoid bugs where he can. Several tools are used for debugging software from different debugging perspectives, separated further by the programming language in question.

DDB

DDB is also known as the GNU debugger, and has been used for decades in stepping through and running tracebacks on C and C++ code. DDB is a live debugger in that the program being tested is run and the user debugging the software can set breakpoints throughout the code, manually stepping through the execution, examining the states of variables or memory locations, and from there determining the reason why a program may be behaving in a peculiar way. This makes it a very hands-on debugging approach, but is extremely useful for placing the state of execution under a microscope.



Showing Machine Code

Figure 54: The tried-and-true DDD interface to DDB.

Static Analysis

From a different software debugging perspective, we can perform static analysis on our source code outside of the realm of execution in order to determine automatically if the source contains poor idioms, coding constructs that are often misused, out-of-bounds array accesses, and many other classes of errors, without even compiling the program. This is not as manually involved as live debuggers such as ddb.

An example of a static analysis tool is the software known as cppcheck. From the cppcheck website, “Cppcheck is a static analysis tool for C/C++ code. It provides unique code analysis to detect bugs and focuses on detecting undefined behaviour and dangerous coding constructs. The goal is to have very few false positives. Cppcheck is designed to be able to analyze your C/C++ code even if it has non-standard syntax (common in embedded projects).” So cppcheck is a libre and gratis static analysis tool for C++ that will potentially help eliminate classes of errors from our frontend implemented in C++.

We intend to use both ddb and cppcheck in order to perform live debugging, syntax checks, and static code analysis in order to sweep for hidden errors in our C++ frontend system.

NodeJS Tools

Our high-level language chosen for the project in order to simplify the development process is NodeJS. Since ddb and cppcheck are intended for use in compilation of C and C++ code to native machine code, the virtual machine oriented NodeJS will require a different suite of software in order to perform linting, debugging, and the equivalent of static analysis.

Node Inspector

Node Inspector allows for live debugging and stepping through the program while in execution. It allows examination of variables, breakpoints, inspection of network state and requests in flight, and CPU plus memory loading effects during the execution. In this way it could be called a beefier version of ddb for its live performance.

Build Systems

Once all the libraries, object files, headers, source files, and their dependencies are collected, and a software project is complete, one may call the compilation or build process easy. The complete set of functions, modules, or libraries is known, and given that the version is unchanging, may be built just as easily with a shell script as any other tool. However, software is never unchanging. Projects need to be built in an extensible fashion from the very beginning as a form of gradual testing and feature addition. New versions of the software will emerge with heightened functionality, and this begs the question of what kind of new functions and libraries will need to be implemented by the software developer in order to support these features. The question must be asked of how these dependencies will be rectified, and software build systems are there to rescue us.

Software build systems allow source files, libraries, and dependencies to be described in such a way that they are managed automatically and in a modular fashion, satisfying two fundamental properties. First, the build system should be able to recognize the library resources or dependencies available on the build computer, according to the way it was configured. The user no longer must manually coordinate every object file; it is

automatically included in the project when the build system runs. The second property of the build system is modularity in that adding functions, libraries, or other dependencies does not break the build process in anything more than a superficial manner. Using an appropriate build system, we gain the ability to smoothly and and remove features without having to crawl through long compiler commands by hand to generate the resultant code. We continue with examples of build systems that are commonly available and widely used.

Make

Make is the tried-and-true build system that has existed for decades. A series of make commands are entered into a text file, and the level of abstraction away from the compiler is generally low. Although advanced implementations of Make such as GNU Make allow for macros and other basic syntactic manipulation, this build system alone has shown its age. In order to support automatic checking and management of dependencies as indicated in the fundamental properties of build systems above, GNU Autotools is often used in combination with the bare Make software. However, Autotools is well-known for being somewhat clunky software that is a bit of a hack. Although it is a standard in many free software projects, many systems are moving on from this piece of work.

```
edit : main.o kbd.o command.o display.o \  
      insert.o search.o files.o utils.o  
      cc -o edit main.o kbd.o command.o display.o \  
          insert.o search.o files.o utils.o  
  
main.o : main.c defs.h  
      cc -c main.c  
kbd.o : kbd.c defs.h command.h  
      cc -c kbd.c  
command.o : command.c defs.h command.h  
      cc -c command.c  
display.o : display.c defs.h buffer.h  
      cc -c display.c  
insert.o : insert.c defs.h buffer.h  
      cc -c insert.c  
search.o : search.c defs.h buffer.h  
      cc -c search.c  
files.o : files.c defs.h buffer.h command.h  
      cc -c files.c  
utils.o : utils.c defs.h  
      cc -c utils.c  
  
clean :  
      rm edit main.o kbd.o command.o display.o \  
          insert.o search.o files.o utils.o
```

Figure 55: Highlighting Make's simplicity.

CMake

CMake is another build system that was first released in the year 2000, but the community support as libre and gratis software is tremendous. Many top companies rely on the CMake build platform in order to make commercial-off-the-shelf and government-off-the-shelf software run. CMake could, in many ways, be called the successor to GNU Make, but its scope is far greater than the simple build script of ordinary make. The syntax of the latest version of CMake is highly expressive and far more readable than enormous makefiles generated with GNU Autotools. As far as the community sentiment

goes, simply avoiding autotools is a legitimate reason to use CMake. Cmake can be used to generate Ninja output.

Ninja

Ninja can be compared to the assembly language of the build process that still must be translated into machine code of executables, like an intermediate stage. However, the benefit of using ninja is the separation of build configuration with executable building proper. Emitting ninja intermediate build components speeds up the process and is generally used whether you are using CMake or the alternative to CMake mentioned in this document below, Meson

Meson

Meson is a simple alternative to CMake created by a Google employee that many argue has a more readable syntax than CMake. Its acolytes also state that Meson is faster than CMake. Both of these charges appear to have diminished in community favor with the release of the newest version of CMake, where it has caught up in speed and syntactic readability. Importantly, CMake has much more community effort and manpower behind it. So CMake wins out over Meson in this case.

Build System Summary

CMake emitting Ninja intermediate compilation files appears to be a simple, rich method of building our software that will allow us to quickly adapt to our project's needs.

Testing:

Our project is to create a Pan-Tilt-Zoom camera for outdoor use. To test the functionality of our system, we will test the responsiveness of the motors, the responsiveness of the remote, and the durability of the system mounted to the tripod.

The remote will transmit signals to the laptop through a wireless connection that gets re-transmitted over wi-fi to the camera and motor systems. To test the response time, we will need to benchmark the data transmission rate. We can issue a series of commands from the remote control to see if the laptop can pick it up. Then if the laptop responds, we can see if the wireless transmission can send the signal out to the system. The time between the control and the movement of the motors or the zoom of the camera should be measured. We also want to see how well the system responds to opposing requests. For example, switching the motor from 0 to 180 degrees or making the camera go from a wide shot to full zoom. Testing these extremes will help determine if the response speed is fast enough to suit the needs of a sporting live stream.

We also want to see how well the transmission can send the video and audio information. Since we are sending the stream over wi-fi, frames can get lost or have a delayed response. This is compounded by streaming delay. It's important that the camera transmits the data smoothly so that we don't have issues when we stream the input to the public. The best test would be to run an internet reliability test. This will send the camera a number of internet packages and receive back the response packages. The number of frames dropped will be

the error rate and the speed of the data will be the transmission rate. Ideally, we want a low loss with high speed.

Another important aspect of the project is the durability of the design. This camera has to survive the Florida weather and not shut down or short circuit. The electrical components will be in a weather-proof housing to protect them. To test the housing, we will use the mini wind tunnel at UCF that the mechanical engineering school uses. This will test the wind resistance of the housing unit and its strength to hold onto the tripod mount.

In addition to the weather in its home state of Florida, the device should be equipped to readily adapt to other worldwide climates, including extremely hot and dry regions, regions that are even more humid than Florida, snowy areas with a very low temperature, and so forth. Electronics typically have no issue running in cold climates; in fact, the lower temperature is usually welcome for cooling the device and helping to dissipate excess heat. This will especially be the case for the motors, the likely largest source of heat after the camera system or any batteries present. However, due to the way snow can drift, including coming in from the bottom of the device and flying upwards to make contact with the bottom surface, this produces additional challenges in regard to waterproofing after the camera is probed from all angles.

In the case of very hot regions that are dry, we are less concerned with the general concept of weatherproofing because IPX standards deal in the relationship with water [6]. Instead, power dissipation resulting in excessive heat release that may result in the device overheating. Since our camera system uses passive cooling in order to dissipate heat released during normal operation, it is depended on the temperature of the environment in order to reach the necessary operating temperatures to ensure the device will work properly. Therefore, there will be an additional test needed in order to prove that it will operate normally under these hot conditions without pushing the temperature of the device outside of nominal limits.

For snowy conditions we must take into consideration special waterproofing conditions involving a totally three-dimensional picture of weatherproofing. This is because snow can drift up and make contact with the system in unusual patterns, unlike rain which generally falls straight down with a single angle associated with it. Snowflakes are more aerodynamic than rain in such a way that they can be blown upward by light wind and land on surfaces normally inaccessible by raindrops.

We also want to test the system's resistance to the rain so we would need to set up a small rain simulation using a hose or sprinkler. The housing will go through a test without the electrical components first to see if the case can keep water out. If the case remains dry on the inside, then we will test the system with the electronics on and operating. We will test the system to see if it fails under the conditions in either the motors, the camera, or the transmission.

Test Plan

In order to test the operation of the device and confirm that it is working as directed, each of the engineering requirements must be applied in some manner to the camera system in a concrete way. Therefore, in this section, we refer back to the requirements and create a testing protocol for each of these.

Test Plan for Tripod-1:

Test Statement

This requirement deals with video stability in the presence of small gusts of wind. Since the tripod is tall, forces on the tower structure are amplified; so special consideration must be put into the design of the stand and feet of the tripod to ensure that not only does the structure not topple over in these gusts, but also that the tripod holds the video system steadily enough such that there is limited jitter or shakiness introduced to the video feed that would interfere with the quality of the capture system.

Test Environment

The fully-mounted camera-tripod system is configured to run as if it were in production. All cables and power systems are attached to realistically simulate the maximal wind flux surface area. The device should be set to record video and store it on the workstation computer. Ensure that the tripod feet are placed on a reasonably sturdy level ground.

As a result of the nature of the sports action camera, it is important to define what will be meant by sturdy here. Any sturdy tripod placement point will be free of debris; however, both concrete or gravel and a grassy sports field must be included in the definition for sturdy in order to simulate realistic conditions for actual use of the device. The grass can be astroturf or false grass; a soccer field; or an ordinary lawn that has been tested for levelness.

Test Description

For the wind environment, three tests are possible depending on the suitability of nature for the testing session. The first is the most scientific in that it is the most thorough and repeatable: the device is placed in a wind tunnel which is configured to blow at ten miles per hour for 30 seconds. The recorded video feed is then analyzed by a team member for video stability. However, gaining access to such a large wind tunnel is likely to be unfeasible for the scope of our project. Therefore, simpler alternative methods will be considered as well, in order to simplify the testing process while ensuring compliance with the standard.

The second method for testing will sacrifice accuracy with respect to the wind speed. The tripod system is placed outdoors in the test environment when the wind speed is approximately ten miles per hour. Since the wind speed may deviate during the test, the actual wind speed should be measured in order to ensure it is reasonably in bounds during the test at plus or minus 3 miles per hour.

A failure above ten miles per hour indicates that it is necessary to perform more testing in order to determine whether the failure was a result of wind speeds outside of the boundaries

for our testing. The experiment will be repeated after wind conditions are remeasured, and they are within the boundaries for the test procedure as specified.

The final methods for testing will sacrifice accuracy with respect to the surface area of the device. An industrial fan may be hoisted up in such a way that the box fan is lined up with the camera system mounted on top of the tripod. In particular, the box fan should be raised such that only approximately 10% of the wind generated blows above the camera system; the rest will fall on the camera and on the tripod mounting system and perhaps some of the legs of the tripod. The fan used for this purpose should blow at approximately 10 miles per hour, plus or minus 3 miles per hour, in order to ensure that the requirement is satisfied. The video will be analyzed by a team member for stability, and the tripod should not tip over in this or any other test.

Test Plan for Housing-1

Test Statement

Since the sports action camera system is designed in part to be used outdoors, special steps must be taken in order to ensure that the system is water-resistant enough to operate as normal in light rainy conditions. Most sports within the scope of this project are placed on hold in extreme rainy weather conditions, so an absolute waterproofing requirement is not necessary. However, with respect to this test, the device must continue to operate normally without water failure after being subjected to a moderate rain of 0.1 inches per hour for at least 1 minute.

Test Environment

The most relevant portions of the device for waterproofing is the motor system, accessory control boards, and the camera proper. Therefore, the tripod and workstation connections are a lower priority for waterproofing tests, and we do not require that these are included for the test to be declared as successful. The camera system that is included should be mounted upright for the duration of the tests as it will be during normal operation. We intend to use a hose perforated at regular intervals wrapped in a shape like refrigerator condenser coils in order to apply the steady rainfall simulation to the device. The input of water will come from a standard residential water supply and holes of an appropriate diameter will be made in the hose in order to produce the correct amount of water off the residential pressure.

Test Description

A source of water, simulated to be rain such as through a metal screen that produces droplet-type water flow, should be placed above the device and made to sprinkle down on the camera system at the specified rate of 0.1 inches per hour for one minute.

Test Plan for Video-1

Test Statement

The camera must be able to transmit the video frames while sampling 1080p at 30 frames per second.

Test Environment

The camera must be configured to record at the desired frame rate and resolution stated above. The camera should be set to record a scene that is realistic for a sports action camera, such as a residential neighborhood with cars moving under 30 miles per hour. This is because encoding algorithms such as H.264 place variable load on memory and processors based on the complexity of the input; a static scene requires much less memory and processing power.

Test Description

The camera is set to record video for one minute. The live feed is examined and compared to the objective source of video in the spectator interface of the workstation software. If there is a delay of greater than one second between the live feed and the objective video source, the test result is a failure. Otherwise, if the video is running smoothly at the specified parameters without a delay of one second, the test result will be declared a success.

Test Plan for Video-2

Test Statement

The camera system, including the video camera setup, the motors, any accessory boards, or other devices above the tripod mount, must weigh under a total of one pound.

Test Environment

The weighing session should be done with a clean, working scale without excessive wind currents that would interfere with accurate measurement. The scale can either be of the analogue or digital variety. The model intended for use in this test is the Taylor precision products analog scale with 11-pound capacity, which has a steel basket and an analog zeroing knob in order to tare the scale to zero before weighing.

Test Description

The camera system as described in the statement is placed on the scale and weighed. The result is recording in the testing documentation and compared to the system requirements for weight. If the measured weight exceeds one pound, the test has failed; otherwise, it is a success.

Test Plan for Audio-1

Test Statement

The audio system must be able to sample in WAV, MP3, or OPUS, at the specified bitrates or quality factors. The sample must not be delayed longer than one second.

Test Environment

The audio sampler must be configured to record at the desired bitrate or quality stated above. The audio sampling setup should be provided a realistic audio source that can reasonably be tested for quality, such as human voices or music. Additionally, the tester should use headphones in order to hear the output clearly.

Test Description

The audio sampler is set to output live for one minute using the realistic input sources described above. Simultaneously, the tester listens for a delay longer than one second. If there is such a delay, the test case is a failure.

Test Plan for Mount-1

Test Statement

The camera mount to the tripod-motor system must be able to withstand minor forces. This could include the force from wind, accidental pressure or other forces applied laterally to the camera. This test case uses five lateral ft-lbs as a minimum for how much force the camera mounting system must be able to sustain.

Test Environment

The camera mount setup is fixed in place so that it does not move freely during testing. The force is applied using a rod or an otherwise solid object.

Test Description

A lateral force is applied on one side of the mount base of 5 ft-lbs. This process is repeated for all of the directions of the mount base to ensure the strength is symmetrical.

Test Plan for Motor-1

Test Statement

The motor must pan and tilt moving a weight of one pound without excessive strain in terms of gear noise or overheating for one hour.

Test Environment

The area the test is performed in should be room temperature. For additional flexibility in environments, a secondary test at 40C can be conducted.

Test Description

A weight of one pound is fixed to the motors, and the motors are operated to pan and tilt in their full range of motion for one hour. The temperature on key points of the motor, such as the power input and the motor proper, is taken and compared to the data sheets for the motor on maximum operating temperatures. If the temperature observed exceeds the amount stated in the data sheet, then the test is considered a failure.

Test Results

This has not yet been tested.

Test Plan for Controller-N

Test Statement

The controller must connect over USB to the workstation and control the camera and motors using the buttons and joysticks to implement panning, tilting, and zooming.

Test Environment

None required.

Test Description

First, the controls for panning are applied. When the left directional pad button is pressed, the camera should pan to the left, from the perspective of the camera, at a constant rate of speed until the maximum range of motion for the camera is completed. Then, the right directional pad button is pressed; in the same manner, the camera should pan at a constant rate of speed to the right from the perspective of the camera until it once again reaches the maximum range of motion on both sides. The delay for panning should be under half a second in order to be considered a success.

When the up directional pad button is pressed, the camera should tilt upwards, from the perspective of the camera, at a constant rate of speed until the maximum range of motion for the camera is reached. Then, the down directional pad button is pressed; in the same manner, the camera should tilt at a constant rate of speed downward from the perspective of the camera until it once again reaches the maximum range of motion on both top and bottom. The delay for tilting should be under half a second in order to be considered a success.

Test Results

This has not yet been tested.

Test Plan for Workstation-1

Test Statement

The workstation software must transmit the commands received from the controller in less than half a second.

Test Environment

None required.

Test Description

An arbitrary command is placed into the transmission buffer for the workstation software. This can be inputted either by controller or by artificially placing such a command into the queue through software located on the workstation. The command is dispatched, and a timer is started. When the device physically begins responding to the command, the timer is stopped. If the recorded time is greater than half of a second, the test is a failure.

Test Results

This has not yet been tested.

Workstation Features Testing

Another very important step in this project is the testing of software which ensures the executions the system makes results in successful responses. In order to test the software everything must be connected properly and powered on, the workstation software needs to be initially connected and able to receive and pass information on to the hardware components. It is expected that the software will be able to accurately measure, and display data related to hardware which also gives clues to the status of the software's accuracy. The workstation software needs to display the current speed limit, camera battery level,

motor battery level, recording status, and current coordinates on the x-y plane. The features of the workstation software are very interconnected with the hardware as a whole; as such it is in our best interest to properly test each feature before finalizing our project.

Objective: The objective of this test is to the features of the workstation software and ensure they are functioning as desired.

Environment: The testing of the workstation software will be tested on the personal computers of our group members on the Windows 10 operating system. The software will be tested on at least 3 different computer systems to avoid any undesired behaviors.

Procedure: In order to test the workstation software, the steps below will be done in sequential order:

1. Power on the camera device and ensure the motors are connected to their power source.
2. Check to see that the video from the camera is being displayed through the workstation software at an acceptable frame rate by waving an object or a hand in the camera's view. The object should move seamlessly in the playback without any lag.
3. Test the workstation's software ability to interpret and send controller data by making simple movements with the controller's joystick to ensure the camera is moving in a logical fashion.
4. While the camera is moving also observe the workstation's feature that tracks the current position of the camera. The current position should be displayed while the camera is stationary and be in a state of change as it moves.
5. Next, hit record and observe the workstation's software indicator to make sure it accurately notifies the user when the camera is or isn't recording.
6. Observe the camera battery level display to make sure it can accurately track battery level as power is used.
7. Make variations in the speed limit of the camera in the workstation software to test if the speed can be properly controlled through software.
8. Finally, repeat these steps on at least 3 different Windows 10 machines.

Conclusion: If we find failure in any of these steps or there is undesired behavior, further maintenance and debugging will need to be done to ensure success of the workstation software. After all the steps can be executed correctly the workstation software can be assumed to be functioning as desired.

Power Testing

In order to ensure that the motor, servo, power supply MCU shields, and camera are drawing power for the devices within nominal limits, a standard multimeter will be used.

Administrative

The Administrative section serves as a documented timeline of how our group organized workload and budget for the success of this project. The milestones and expected weeks of completion are logged from the formation of teams to the final presentation. The budget

for the project is also discussed, it describes the contrast between initial budget given by the potential sponsor and our own design budget.

Budget:

The budget shown in Table 13 displays options for both ready made products and parts for systems we would build ourselves. There are three major categories for the products. There are the parts for the camera, parts for the controller, and parts for the mount. The computer workstation will just be the user’s computer with a user interface. A good option for a bought camera is a --- because it supports wireless video transmission and has the durability to withstand harsh weather conditions.

Our proposed budget shown below frames the calculated costs for the project. The prices listed in the table are not estimates and are actual costs associated with the parts purchased and to be used in the assembly of the PTZ camera. As there was no sponsor of this project the total cost was split between each group member. For now, the budget seems to be enough for the total cost of the entire project including spare parts. If there is an increase of cost that goes over the budget for any reason the cost will again be split between all group members.

We will build the motor system, so we are buying the motors and control units separately. This allows us to design our own system custom suited to our needs. While we will buy the mount, we hope to make the housing unit by 3D printing the parts with ABS plastic. Our budget also includes some spare parts, the servos, just in case we get malfunctioning servos from the kit. Our goal is to also get a spare control board, but it depends on the rest of the budget since we want to keep the cost below \$400.

Item	Quantity	Price
Camera	1	\$200
Usb-to-barrel adapter	2	\$13
Power adapters	2	\$20
Main Control board	1	\$10
Servo control board	1	\$10
Game controller	1	\$56
Raspberry Pi pin header	1	\$1
Mount kit	1	\$33
Additional connection cables		\$15
Total	--	\$358

Table 13: Budget

Milestones:

The timeline for the milestones were set around the schedule and weekly responsibility of each member to provide a productive and timely schedule. The project milestones are split

into two categories the first for Senior Design 1 which is focused on research and design; the second category is for Senior Design 2 which is focused on implementation.

Unlike the years before this year of Senior Design is to entirely online, at times the online only model and social distancing rules negatively impact team productivity. However, by properly setting well timed milestones the initial hurdle was overcome for Senior Design 1. As for Senior Design 2, restrictions for social distancing have slowly been lifting and will more than likely not be an obstacle for the physical implementation portion of this project. For now, we plan to physically assemble the project as a team, however this can change if new restrictions are again placed during that time.

The progression of this project has been well documented as it was important to overall productivity to have things done in a timely manner. We started this semester with first a formation of teams, this was done with the help of the professor in an online classroom setting in which project an idea was discussed and any interested students would try to join the group of the student who presented it. After our group was formed, we were tasked with coming up with coming up with a few ideas individually before bringing them all together before the group. After much discussion we decided on one idea completable within our time constraints and skills.

The initial idea we came up with was a drone that could reconstruct the terrain within its vision in 3D. The drone would be equipped with lidar and sonar sensors, a microcontroller, and a raspberry pi 4. The mapping drone would be completely controlled by the operator and overcome many conditions in sensor mapping today such as light conditions unsuitable for lidar and photogrammetry, the limited power supply available to drones, the range limits of sonar mapping, and the expensive cost of lidar sensors. However, after the Divide and Conquer 1 check-in with the professor we were advised that the amount of time and cost it would take to complete this project were out of the scope of this course. In response, we chose a modified version of the PTZ camera design proposed by Qwik Cut a potential sponsor for our group.

After weeks of research the initial refined project idea was solidified as reasonable and ready for design and intrinsic research of the more specific parts. The project was divided among the group members into three groups; the video system, capture device, and audio system were given to Michael; the power-data link, workstation software, and game controller were assigned to Chisom; while the responsibility for the camera motor, mount, and tripod was assigned to Andre.

Table 14 outlines our milestone goals week by week. A majority of the first semester is spent on documentation and developing an initial prototype and writing the proper documentation. This semester we looked at all the possibilities to solve the problem of a long term PTZ sports camera. We developed a solution using our own motor system and wireless video transmission.

A majority of the work was spent on researching different systems and how we could emulate them and writing the report. The second semester is about building the prototype,

analyzing the initial results, and developing an improved system. We hope to have our parts by May, when the second semester starts. That way we can get to work on our prototype right away. Our goal is to test the prototype and have time to revise the design so that we may implement our stretch goals.

SENIOR DESIGN 1	
WEEK	Milestone
1	Form teams
2	Brainstorm ideas
3	DC1
4	Investigate project idea
5	DC2
6	Determine what products are needed for project
7	Draft parts requirements
8	Divide report research into parts
9	Write research reports based on division of topics, start first draft
10	Put research together and edit
11	Work on report
12	45 page draft, order parts
13	Write final pages
14	80 page draft
15	Final draft
SENIOR DESIGN 2	
1	Software plan, work on building motor sub-system
2	Implement workstation software, communicate with motors wirelessly
3	Communicate with camera, prototype housing
4	Finish first prototype
5	Initial testing
6	Assess and revise prototype
7	Remodel and rebuild housing, improve software
8	Implement stretch goals if all goes well
9	Second test and assessment
10	Write final report
11	Revise
12	Final presentation

Table 14: Milestones

3D Printing Vendor

In the search for a suitable 3D printing vendor, we focused on finding certain necessary traits that would assure the quality and price point of the design. The vendor chosen would need a fast turnaround time, fair pricing, good quality, and good reputation. For those reasons the vendor we decided to send our design to be was brain3D. After research on top

vendors brain3D stood out as the reviews described a company that was punctual, quality driven, and price efficient.

As we are making a custom design the pricing for 3D printing is different on a case-by-case basis. Below are the best estimates and listed qualities with the information available to us before ordering:

- Standard PLA (polylactic acid), 100% infilled - \$22.99 for spool
 - Bio-degradable
 - Usually 2-day turnaround
 - Heavy and durable
 - Commonly used in cups, plastic bags and cutlery.
- Standard ABS (acrylonitrile butadiene styrene), 100% infilled - \$21.99 for spool
 - Strong and flexible
 - Usually 2-day turnaround
 - Strong and flexible
 - Higher heat resistance
 - Commonly used in LEGOs and electronic housings.

Since pricing is given after the design has been created and received by the vendor these are only estimates based on industry standard. Brain3D also allows for only infilled printing which needs to be accounted for in weight and price.

Specifications:

- All prints ordered from brain3D must be 100% infilled.
- Choice of acrylonitrile butadiene styrene, amorphous melting point, tensile strength 27 MPa, and 3.5-50% elongation.
- APS has density from 1.0 – 1.4 g/cm³ and has a glass transition temperature of 105 degrees Celsius.
- The PLA option has a 173 degrees melting point, tensile strength of 37 MPa, and 6% elongation.
- PLA has a density of 1.3 g/m³ and has a glass transition temperature of 60 degrees Celsius.

The 3D printed housing unit are designed in such a way that it is initially easy to assemble and disassemble the parts. However, the method used for waterproofing may inhibit that initial ease of assembly depending on which sealing method is used.

Design Issues

Power and Data Delivery

One of the most significant challenges in realizing the design is in the cabling used to link the systems together. Since the distance from the base workstation to the camera is at least 25 feet, the simplest method of USB cable for power and data delivery is challenged, because even for low power devices, the power dissipation incurred by the

low-voltage desktop PC port over a long USB cable is very expensive. Attempts to reconcile this with common classes of devices available online was difficult as either the power problem was not solved, the data bandwidth of the line was not high enough, or the hardware had to be created from scratch.

Camera Connectivity

Since power-over-Ethernet cameras are often security cameras, they have features that make them less accessible, less compatible, or more expensive than necessary for an action camera. Features on security cameras such as built-in alarm systems, infrared recording, and so forth were a type of virtual litter that made it more difficult to find systems meeting our criteria within the budget. Faced with the difficult choice of either building our own power over Ethernet system from scratch, compromising on a camera loaded with unnecessary features, or entering a wasteland of duplicate USB cables, we decided that it would be instead quite liberating to use wireless communications for data and a simpler low-power battery charging USB system for the camera. This has an additional benefit that makes this camera more desirable and easier to use. Not only is a prebuilt action camera widely available for part replacement in case of damage, defaulting to a battery and using WiFi for communication allows the user to disconnect the camera assembly from the workstation while filming.

Duplicate Cabling

One of the primary directives of the project as an idea was to avoid unnecessary extra cables, whether that be power or data cables. In the minds of some project team members, it was an anathema to use two wires where one could be sufficient. This was only natural as this could double the wire length necessary in a system that involves a tall tripod. However, sometimes practice can override principle, and we found that the most straightforward solution was to send power over two separate wires instead of shoehorning them into one.

To Reinvent the Wheel or Not

A recurring theme of the project design was whether to buy a part or to create it ourselves. Much effort was invested into moving further down the rabbit hole into obscure component and PCB design. However, we decided eventually that much of the progress we had made building the custom design was simply leading back to readily available consumer hardware with similar features. Therefore, we decided to invest more of the project resources into software development than on hardware design.

Team Meetings

Availability of all the team members, even though this is a small project composed of only three members, can be challenging in creation of meetings that are critical for composing our work as a group effort. When an individual member is unable to make it, a fractal-like chain of rescheduling can occur that can wreak havoc on group coordination, potentially causing delays in productivity. We found it important to remain flexible and understanding with our team members, and stay far ahead of deadlines, in order to minimize the risk impact of a delayed page count in the face of due dates.

Collaboration Platforms

Since concrete collaboration is key in any group effort, and because of restrictions on gathering under the COVID-19 pandemic, online software collaboration platforms were critical for staying on track in our project, and we knew this from the beginning. However, it took using these particular platforms, such as Discord, Google Docs, Google Drive, and Microsoft Teams, in order to subjectively determine what platform would be most beneficial and usable to our group members. We found that Discord was sufficient to conduct group meetings, but for real-time file sharing with updates we found it to be insufficient. We moved the document sharing activities over to Google Docs, which was quite an upgrade from the rather static Discord system but found Docs to be inflexible and cumbersome as the document grew larger. Towards the end of the collaboration on the document, we transitioned to Microsoft Teams, which behaves in a similar manner but allowed better compatibility on document formatting. Additionally, we found that Microsoft Teams appeared to handle the larger page load and sidenotes in a more fluid manner. Overall, Teams was more featureful comparable to a full-fledged word processor in the browser over Docs.


Works Cited

- [1] *Farnell Company*. “Arduino Uno Datasheet,” 2018, <https://www.farnell.com/datasheets/1682209.pdf> , Accessed 28 March 2021.
- [2] *Gopro, Inc.* “Gopro Hero Product Information,” 2020, https://gopro.com/content/dam/help/hero/manuals/UM_HERO_ENG_REVB_WEB.pdf , Accessed 20 March 2021.
- [3] “*The RF Connection*. Microphone connectors,” 2017, <https://rfconnection.com/microphone-connectors/> , Accessed 20 March 2021.
- [4] *Electonics Notes*. “Ethernet IEEE 802-3 Standards,” 2020, <https://www.electronics-notes.com/articles/connectivity/ethernet-ieee-802-3/standards.php> , Accessed 21 March 2021.
- [5] *Cisco Corporation*. “Power Over Ethernet Standards,” 2020, <https://www.cisco.com/c/en/us/solutions/enterprise-networks/what-is-power-over-ethernet.html> , Accessed 28 March 2021.
- [6] *UEnclosure Company*. “IP Ratings Explained,” 2017, <https://www.enclosurecompany.com/ip-ratings-explained.php> , Accessed 28 March 2021.
- [7] *TotemMaker*, “Electric motor comparison (DC vs Servo vs Stepper),” (n.d.), <https://totemmaker.net/blog/electric-motor-comparison-stepper-vs-servo-vs-dc/> , Accessed 12 March 2021.

- [8] *Jameco Electronics*. “Motor Buyers Guide,” (n.d.), <https://www.jameco.com/Jameco/workshop/ProductNews/motor-buyers-guide.html> , Accessed 5 March 2021.
- [9] *Electronic Design*. “What’s The Difference Between Brush DC And Brushless DC Motors?,” 16 February 2012, <https://www.electronicdesign.com/technologies/electromechanical/article/21796048/whats-the-difference-between-brush-dc-and-brushless-dc-motors> , Accessed 5 March 2021.
- [10] *Association for Advancing Automation*. “Brushed DC Motors Vs. Brushless DC Motors,” 24 January 2017, <https://www.automate.org/blogs/brushed-dc-motors-vs-brushless-dc-motors> , Accessed 12 March 2021.
- [11] *Electronics Notes*. “WiFi Standards: IEEE 802.11,” (n.d.), <https://www.electronics-notes.com/articles/connectivity/wifi-ieee-802-11/standards.php> , Accessed 1 April 2021.
- [12] *Tempo*. “The H Revision: New Soldering Standards in Telecommunications PCB,” 4 February 2021, <https://www.tempoautomation.com/blog/the-h-revision-new-soldering-standards-in-telecommunications-pcb/> , Accessed 1 April 2021.
- [13] *NASA*. “SOLDERED ELECTRICAL CONNECTIONS,” 18 January 2001, <https://nepp.nasa.gov/docuploads/06AA01BA-FC7E-4094-AE829CE371A7B05D/NASA-STD-8739.3.pdf> , Accessed 2 April 2021.
- [14] *Sparkfun*. “I2C,” (n.d.), <https://learn.sparkfun.com/tutorials/i2c/all> , Accessed 1 April 2021.
- [15] *NXP Semiconductors*. “UM10204 I2C-bus specification and user manual,” 4 April 2014, <https://www.nxp.com/docs/en/user-guide/UM10204.pdf> , Accessed 1 April 2021.
- [16] *Reichelt Elektronik Magazine*. “Guide on USB connections over long distances,” (n.d.), <https://www.reichelt.com/magazin/en/usb-connections-over-long-distances/> , Accessed 16 April 2021.
- [17] *All3DP*. “3D Printer Filament Buyer’s Guide,” 6 February 2020, <https://all3dp.com/1/3d-printer-filament-types-3d-printing-3d-filament/> , Accessed 16 April 2021.
- [18] *Diffen*. “Bluetooth vs. Wi-Fi,” (n.d.), https://www.diffen.com/difference/Bluetooth_vs_Wifi , Accessed 16 April 2021.
- [19] *3D Printing.com*. “Software For 3D Printing,” (n.d.), <https://3dprinting.com/software/#Simplify3D> , Accessed 16 April 2021.

- [20] *All About Circuits*. "The Raspberry Pi Zero W Adds Wireless Capabilities with Wi-Fi and Bluetooth," 28 February 2017, <https://www.allaboutcircuits.com/news/the-raspberry-pi-zero-w-adds-wireless-capabilities-with-wi-fi-and-bluetooth/>, Accessed 16 April 2021.
- [21] *Mouser*. "Arduino Uno Microcontroller Board," 11 November 2020, <https://www.mouser.com/new/arduino/arduino-uno/>, Accessed 1 April 2021.
- [22] *Raspberry Pi*. "Raspberry Pi Zero W," (n.d.), <https://www.raspberrypi.org/products/raspberry-pi-zero-w/>, Accessed 1 April 2021.
- [23] *USB.org*. "USB 2.0 Specification," 21 December 2018, <https://www.usb.org/document-library/usb-20-specification>, Accessed 16 April 2021.
- [24] Pulli, Kari; Baksheev, Anatoly; Korniyakov, Kirill; Eruhimov, Victor (1 April 2012). "Realtime Computer Vision with OpenCV". Accessed 28 March 2021.
- [25] B. Nuttall and D. Jones, "gpizero", 2015. Available: <https://gpiozero.readthedocs.io/en/stable>, Accessed March 24, 2021.
- [26] L Orsini. "What You Need To Know About Node.js," <https://readwrite.com/2013/11/07/what-you-need-to-know-about-nodejs/>, 7 November 2013. Accessed March 23, 2021.
- [27] *IEEE standard 802.af-2003*, June 2003. Accessed April 1, 2021
- [28] *802.3at Amendment 3: Data Terminal Equipment (DTE) Power via the Media Dependent Interface (MDI) Enhancements*, 2009. Accessed April 1, 2021
- [29] "*IEEE P802.3bt DTE Power via MDI over 4-Pair Task Force*", 2016. Accessed March 29, 2021.
- [30] *Microsoft TechNet* "Inside Native Applications", November 2006. Accessed 1 April 2021.
- [31] *Mirosoft*. "XInput and DirectInput," 17 September 2012, Available: <https://docs.microsoft.com/en-us/windows/win32/xinput/xinput-and-directinput?redirectedfrom=MSDN>. Accessed February 25, 2021.

Appendix A: Sparkfun permissions.

RE: Quote - Order #n/a - Guest [ref:_00Do0aduJ_5001J11BaeR:ref] 



Support <support@sparkfun.com>
to me ▾

Thu, Apr 22, 12:38 AM (5 days ago) ★ ↶ ⋮

Hello Andre,


Thank you for your feedback. We are an open-source company so you are free to use any content we have on our site. We only ask that you give us credit when you do reference us or use a photo.

Kind regards,
Nicole

----- Original Message -----
From: SparkFun Support Form [no-reply@sparkfun.com]
Sent: 4/20/2021 11:51 AM
To: support@sparkfun.com
Subject: Quote - Order #n/a - Guest

New Support Request!

Appendix B: NitroRCX Permissions.

Permission to use product photos 



Andre Samaroo
Hello, I am a student at UCF writing a robotics report. In my report, I discuss using some of your brushless motors. I wanted to ask for permission to use some

Tue, Apr 20, 3:00 PM (7 days ago) ☆



NitroRCX
Ok, no problem. Sincerely, NitroRCX.com/Xhell.com/Nitroplanes.com/Hobbypartz.com America's Top Online Retailer & Distributor for RC hobby

Tue, Apr 20, 3:04 PM (7 days ago) ☆



Andre Samaroo <andre.samaroo@gmail.com>
to NitroRCX ▾
Thank you!

Tue, Apr 20, 3:07 PM (7 days ago) ★ ↶ ⋮

Appendix C: Raspberry Pi Permissions



Nicola Early info@raspberrypi.com via freshdesk.com
to me ▾

Tue, Apr 20, 9:30 AM (7 days ago) ★ ↶ ⋮

Andre

Thank you for your interest in Raspberry Pi and for taking the trouble to check this with us.

Our documentation is released under a Creative Commons licence. The schematics occur in our documentation which is licensed under a Creative Commons Attribution-NoDerivatives 4.0 International (CC BY-ND) (<https://creativecommons.org/licenses/by-nd/4.0/>).

Kind Regards

Nicola

On Fri, 16 Apr at 7:28 PM , Andre.samaroo <andre.samaroo@gmail.com> wrote:

Your name	Andre Samaroo
Your email	andre.samaroo@gmail.com
Subject	Something else
Your message	Hello, I am a student writing a report for a robotics project. I would like to use the Pi Zero W pin diagram schematics in my report and I wanted to reach out for permission to reprint it in my report. The schematics will be cited in the appendix. Thank you for your time.

Appendix D: The Pi Hut Permissions



Richard (The Pi Hut Support)

Apr 22, 2021, 11:39 GMT+1

Hi Andre

No problem at all - I've attached our larger versions for you as well in case that helps :)

Thanks

Rich


Attachment(s)

[DSC_0454_5dd8180f-e117-4ec2-9f23-f8027e692e6c.jpg](#)

[DSC_0455_2f737141-6970-4f70-8a54-d031af5f59ad.jpg](#)

[DSC_0456_46bb1fbd-e6d0-4577-b161-5bcfd638b4c5.jpg](#)


Appendix E: Adafruit Industries Permissions

Re: [[EDUCATOR INQUIRY]] - andre.samaroo@gmail.com 2021-04-16 14:47:24 



Adafruit Industries <support@adafruit.com>

to me 

Fri, Apr 16, 3:16 PM (11 days ago) 

feel free to, credit adafruit.

On Fri, Apr 16, 2021 at 2:47 PM Andre Samaroo <contact_us_forms@adafruit.com> wrote:

contactname : Andre Samaroo

email address : andre.samaroo@gmail.com

Message : Hello,

I am a student at UCF and I'm writing a robotics report. I discuss some of your products, specifically the pan-tilt kit and the micro servo. I wanted to ask for permission to use the product images in my report. I would also include the proper citation in the appendix.

Thank you for your time.

Client IP: 104.136.57.207

Appendix F: RobotShop Permissions.

Your request (T720276) has been updated. To add additional comments, reply to this email.

There is an attachment at the bottom of this message.

Mariane (RobotShop)

Apr 19, 2021, 10:21 EDT

Hi,

Thank you for your reply Andrea.

You may use the two following images of our product the [Lynxmotion Pan and Tilt Kit / Aluminium](#) (see attachments).

However, we do not retain the rights for the pictures of the Dagu Mini; you will need to contact the manufacturer Dagu directly and request their autorisation.

Please let us know if you have further questions.

Regards,

Mariane
RobotShop inc.

Appendix G: Reolink Permissions

You have received a message from the Amazon Seller - ReolinkDirect

Dear Chisom,
Glad to support you.
It is ok for you to use it in a right way.
Appreciate your support on Reolink products.
Have a nice weekend!

Thanks and best regards,
ReolinkDirect-Sabrina

Appendix H: Ipolex

You have received a message from the Amazon Seller - ipolex

Hi Chisom,

Thanks for reaching out.

Sure. You can use the primary display images for the iPoex Active PoE Adapter for your project. And please note the picture are only just for learning reference only , shall not be used for commercial purposes. All copyrights belong to Ipolex.

Thank you!
Ariel

Appendix I: Steamemo Permissions

You have received a message from the Amazon Seller - STEAMEMO

Hello

Do you mean you need to use our pictures?

If these pictures are helpful to you and are not for commercial use, you can use it


Appendix J: Cudy Permissions

You have received a message from the Amazon Seller - Cudy

Dear Chisom,

Thanks for contacting us. You could use the picture for your college project if it is not for commercial use.

Appendix K: TP-Link Permissions

 **Chisom Ikejani** Fri, Apr 23, 3:48 PM (4 days ago) ☆
Hello Brian, The image would be used as a visual of representation of the product. We are comparing different PoE injectors by price, transfer speed, and output

 **Brian Martinez** Fri, Apr 23, 3:56 PM (4 days ago) ☆ ↩ ⋮
to me
Hi Chisom,
For educational purposes the use of the image is fine. Please credit TP-Link.
Thanks!
—
Brian
xxx
