Robofan

07/28/18

Group 8/13

Team

Ryan Rossbach
William Terry
Floyd Thormodson II
Ivan Sarmiento

Table of Contents

List of Figures

List of Tables

1.0 Executive Summary

Workshops can get extremely hot, especially if airflow is sub-par. Even worse, you could be working out of a garage in Florida! This team is comprised of four young Floridians who are striving to combat every Floridian's two greatest enemies the heat and humidity. Many members of this team have had to work in environments where airflow was less than optimal. The air pushing apparatus at our disposal, did not meet our satisfaction. Hence the passion. Large fans might not always fit in a workspace, and if you are moving around you will get limited effect from a stationary fan or oscillating fan. We are going to design a fan that does not fall into these same pitfalls. We are introducing the Robofan. It is a ceiling mounted fan armed with sensors that can track a user around a room. This was accomplished by equipping the Robofan with mechanisms to allow it to tilt and rotate. What separates this from a standard oscillating fan is that it is not be limited by a 180-degree operating threshold, instead the Robofan has the capability to continuously rotate. With this ability, just a single Robofan can cover an entire room, never letting a user escape its cooling gaze. An accompanying smart phone application allows users to control the Robofan remotely. Manually pointing the fan, controlling the speed of the fan, and assigning users are among the actions that can be taken with the smart phone application.

The Robofan's main components are the mechanism that controls its tilting motion, the mechanism that controls its panning motion, a sensing technology to track the user, wireless communication technology, and a microcontroller to tie it all together. The selection of components is crucial. Therefore, this document details the technology comparisons made in order to choose the optimal parts for the Robofan. This project is self-funded so great consideration is taken to make sure the components, software, and tools are the best for our price range. The Robofan must be able to keep up with the speed of the target/user at a reasonable walking pace to simulate its primary use for a potential consumer.

To achieve this functionality a specially designed firmware is implemented to communicate with the specifically chosen hardware as well as the communication module, used to connect with the smart phone application for optimal consumer experience. To gain the most from our modest hardware, due to our financial constraints, clever implementation of firmware is used to gain the optimum output from each individual piece of hardware. A detailed and concise explanation of all the programming methodology utilized within the scope of the Robofan is found in this document. Throughout this document we have set requirements, milestones, and propose design architectures to use as the foundation of the Robofan.

2.0 Project Description

This sections goal is to analyze our project description. The goal of a project description is to fully detail the scope of our development, and provide a high level overview of what we intend to accomplish. We accomplish this by looking at the motivation for creating our product, and why we feel this project is a necessary addition to the world. The motivation is shown to be very objective, and applicable to many different people that could be interested in the device. The motivation for why the market needs our Robofan is examined. We discuss the requirements necessary to create a functional prototype that shows our devices effectiveness and usefulness. The requirements are chosen based on the constraints written about in section 5.0. Additionally, requirements are written as a list of measurable quantities that The house of quality is shown and examined in detail. The house of quality is important for visualizing the cost to benefit trade off of different aspects of the project, as one aspect goes up it could affect another aspect either positively or negatively. Finally, we go into the potential broader impacts our device can have. Here, we will look at if it has any potential social implications, and if it can be effectively used by our target demographics or other groups. This section effectively shows the overall impact of our project on the intended user group.

2.1 Project Motivation

Workshops can get extremely hot, especially if airflow is sub-par. Even worse, you could be working out of a garage in Florida! Most garages or workshops aren't equipped with air conditioning like the rest of a house due to the garage door and possibly not being insulated to the outside. Large fans might not always fit in a workspace, and if you are moving around you will get limited effect from a stationary fan or oscillating fan.

2.2 Objectives

Our project was to design a compact ceiling mounted fan that tracks a person in a 360-degree area. The fan is able to pan and tilt so as to point anywhere in the room. This allows for a smaller fan to have a much more precise and noticeable effect. The fan then tracks the operator via a Pixy camera mounted on the front of the fan and a patterned necklace, hat, or shirt on the user. We designed a printed circuit board (PCB) that can take coordinates from the Pixy camera and tell the motors how they need to move to keep the fan aimed at the user.

The board also has a Bluetooth module so that it can communicate with a companion app. The easy to use app was programmed in android and be able to change the fan mode(auto/manual/pattern/etc.) and control the fan in manual mode. The application also allows the user to change the fan's position manually.

The pixy camera can gauge distance so it is possible to have the fan blow at different power levels to keep a certain perceived effect on the operator. We also planned on integrated a thermometer or use the phone's thermometer to adjust fan power. The application also has an option to change how much effect they want or the user can lock the fan at a power level.

2.3 Requirements Specifications

Robofan does not need to be highly accurate, as the column of air the fan produces is fairly wide. We should try to get the fan accurate within half the width of the fan, to ensure it is centered enough on the user, but also not constantly making slight and unnecessary adjustments. Power should be able to be cut from the device through a physical switch. This ensures that the device does not rely entirely on the Bluetooth application or its own programming to know when to turn off. The device shall be controllable through an android app connected via Bluetooth. This allows the device to be remotely and manually controlled without the use of multiple hardware buttons. The fan shall be programmed to systematically shut off features to conserve energy. First, the fan is powered off if no user is detected within 10 minutes. The device completely powers off if no user is detected within an hour. The device completely meets relevant safety standards. As shown in table 2-1 below are the numerical requirements, a description of why the value was chosen, and the target value.

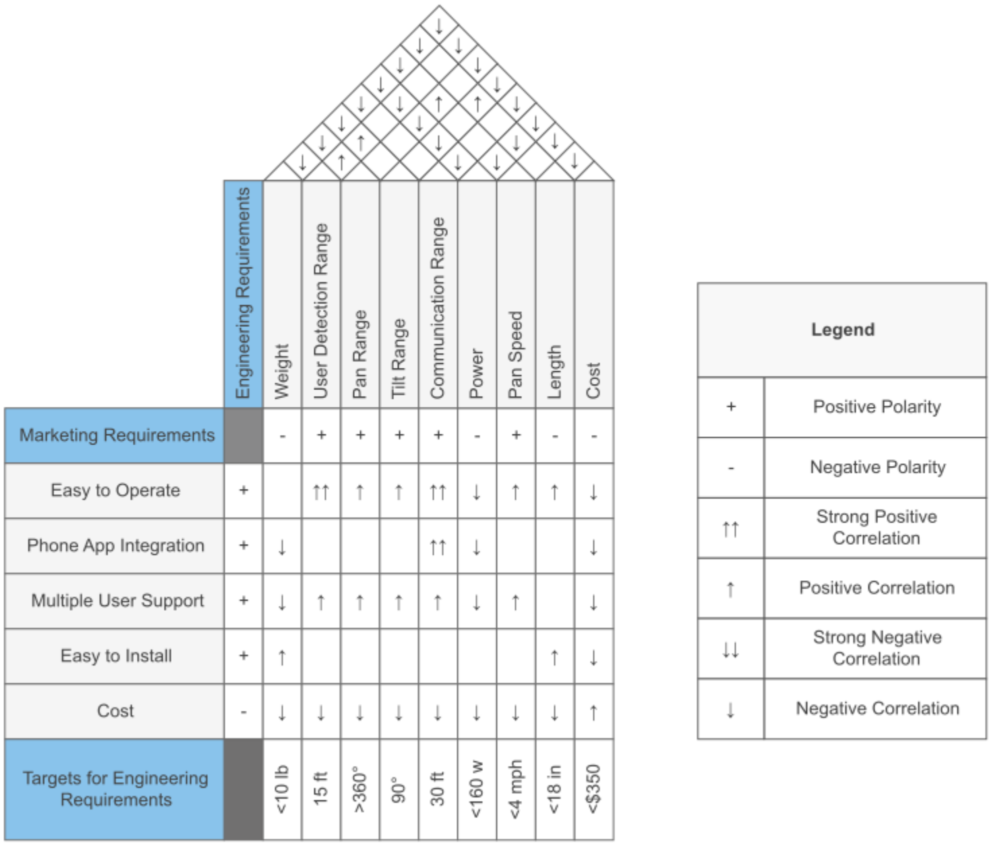
Table 2-1: Numerical Requirement Specifications

Name	Description	Target
Pan Speed	The device turns fast enough to keep up with an average walking speed of 3.1 mph	>4 mph
Detection Range	The detection range is suitable for an indoor workshop or garage	15 feet
Communication Range	The user app is able to function at a greater range than the device's detection range.	>20 feet
Power	The device does not use much more power than what an ordinary oscillating fan uses. Wattage varies based on fan speed and how actively the fan/user moves	<160 Watts
Length	The length of the device (from mount to top of fan) is suitable to work within an average American home with a ceiling height of 8 to 9 feet.	<24 inches
Pan Range	The pan motor's degree of rotation does not be limited to a specific threshold. The device is able to rotate in any direction continuously.	>360°
Tilt Range	The tilt of the elbow joint allows the fan to point directly below the mount. The fan is able to tilt close to 45° from this position in either direction.	<45°
Weight	Since the device is mounted to the ceiling, the lighter it is the better.	<10 pounds

2.4 House of Quality

The house of quality is a tool that can help optimize the design of a project. Figure 2-1 is a house of quality specifically for the Robofan. It shows the relationship between the various marketing and engineering requirements. The more upward pointing arrows there are the better the situation is. This means that more requirements are positively correlated. The more downward points arrows as the situation becomes less optimal.

A quick glance at the diagram shows that there are several downward pointing arrows. These represent negative correlation between two requirements. The fewer there are the better as it means improving one requirements worsen the other. However, on further inspection most of the downward arrows appear in boxes that relate to cost, and power; these two requirements are often at odds with the remaining requirements in any project. So, disregarding those two for now, shows that there are very few downward facing arrows. This means the remaining requirements either have positive correlation or have no correlation at all. This is an advantageous situation since there is no balancing act required to improve those requirements. When examining engineering requirements' relation to each other most of the downward arrows correspond to power, cost, and size. Again, these are all requirements that tend to have negative correlation when compared to others. Disregarding those for a moment, shows that all other engineering requirements have no correlation or a positive correlation. This makes it easy to separate and simplify the project into smaller sections, without worrying about whether one section affects the other.



	Engineering Requirements	Weight	User Detection Range	Pan Range	Tilt Range	Communication Range	Power	Pan Speed	Length	Cost
Marketing Requirements		-	+	+	+	+	-	+	-	-
Easy to Operate	+		↑↑	↑	↑	↑↑	↓	↑	↑	↓
Phone App Integration	+	↓				↑↑	↓			↓
Multiple User Support	+	↓	↑	↑	↑	↑	↓	↑		↓
Easy to Install	+	↑							↑	↓
Cost	-	↓	↓	↓	↓	↓	↓	↓	↓	↑
Targets for Engineering Requirements		<10 lb	15 ft	>360°	90°	30 ft	<160 w	<4 mph	<18 in	<$350

Legend	
+	Positive Polarity
-	Negative Polarity
↑↑	Strong Positive Correlation
↑	Positive Correlation
↓↓	Strong Negative Correlation
↓	Negative Correlation

Figure 2-1: House of Quality for Robofan

2.5 Broader Impacts

This project ideally was used by anyone with a workshop space that cannot afford to have it properly cooled. It would act as a quality of life improvement for workshop users that would allow them to function in a "hot" environment for sustained periods of time. Additionally, it would be an ideal fan for disabled people that may have difficulty moving a fan to whatever position they may move to in a room, while staying useful.

The fan itself could technically be replaced with anything that needs to be pointed at an individual. This device could thusly function, with some modifications, in various different capacities depending on the needs of the user. For instance, it could be modified to have a heating element added to the fan to function as an area heater if a user is too cold.

This device’s goal was to provide a cheaper alternative to a larger cooling unit for any individual or company that wishes to save money. This is useful as extending central cooling to a workshop, or providing a large area cooler is very expensive. Additionally, some areas such as warehouses that have frequently opened large doors are not cost effective to cool no matter what. This device would focus on cooling individuals as they moved in the warehouse, rather than cooling an entire area. This goal was to hopefully make it a cheaper alternative that would see use by large companies.

3.0 Research Related to Project Definition

Our project is composed of several components. Many of these components had other options that could have replaced them. This section discusses our inspirations for certain design choices. We cover products that already exist in the market. We examine these preexisting products for similarities we can use in developing our own device. We also look at projects made in other Senior Design groups that could cover a similar niche as our intended design. The goal of looking at these similarities is to determine if there are other avenues of design to take when creating our design document, and final product. We are primarily looking for other devices that utilize a form of vision to locate and examine another object in real time. We also want to look for products that have some form of interior locomotion such that they can move in place while installed to something by utilizing various motors. Additionally, this section covers what the competing choices for given components. Those alternative choices have clearly defined positive reasons to utilize them, as well as negative reasons not to. We clearly show the pros and cons of each choice, and the final decision for why we did not use any related research presented here. We examine components with close regards to our constraints as determined in section 5.0. It is important to research and compare with clear goals in mind. By following our constraints, it is easy to show what would constitute a good component and use of our effort, and what would be wasteful or out of our design scope. The goal of this section is to prove that we chose the most effective design choices we could find, by showing our research related to the project's definition.

3.1 Possible Architectures and Related

There are many considerations for other architectures that could be used in designing the Robofan. The main competing research was a camera that saw in infrared, such as a generic security camera. Infrared is incredibly useful, as it is possible to design a vision system that looks for specific beacons. An infrared beacon can be designed to operate at a specific frequency. The vision system could be set up to look for certain frequencies and track those. This very specific searching method could be used to avoid misidentifying a target. However, this was not chosen because it would be more complicated than utilizing the Pixycams built in functionalities. To properly use the IR camera, an entirely independent vision codebase would need to be built to attempt to specifically identify the IR beacons. Further, we would need to actually construct an IR beacon to go along with the project. This increases overhead, as well as makes the device more difficult to use. This added object is something that can be lost that would prevent the device from properly functioning. For these reasons, we did not choose an IR camera.

An alternative to utilizing an IR camera, is a camera that would only have a visual feed. The visual feed would need to be utilized for object recognition. The best way to recognize objects is to develop a machine learning software. The machine learning would be developed in Python due to existing frameworks, and ease of use. A machine learning system is a rather difficult thing to construct alone, even given the existing knowledge base surrounding it. However, designing a machine learning algorithm is a very nice thing to have done, and can be put on a resume. The system, once built would have to be trained to recognize certain objects. This would give us many choices about what to look for. The choices that were considered are heads/faces, torso, or an entire human body. It is difficult to predict exactly what pieces of a human is visible in a workshop environment, given protective gear that is occasionally worn. This makes deciding how to train the system a difficult task. Despite the fun that using this architecture might be, it was decided against.

Wi-Fi was a possible design choice for the communication medium. An advantage Wi-Fi has, is a much larger range of communication. Wi-Fi additionally has multiple connection frequencies: 2.4, 3.6, 5 GHz, as compared to Bluetooth's single frequency of 2.4 GHz. It also has a much higher maximum bandwidth capacity than Bluetooth's. However, all of these features are not necessary for the simple device that is our project. The main goal of Wi-Fi is also to connect a device to the internet, and we only want to connect to another device. Wi-Fi additionally appears to be more difficult to work with and design around the existing IEEE standards. So while Wi-Fi is a very powerful option, we would not be utilizing it in any capacity that justifies the choice. Therefore, Wi-Fi was not chosen.

4.0 Related Standards

Within all branches of STEM (science technology engineering mathematics) fields there are sets of standards that are used to keep practices within said fields to a level of quality and uniformity that the community has generally agreed upon. Standards are put into place for ethical, health, safety, optimization, and uniformity purposes. Standards almost universally comply with the laws of the country they are published in. This may sound obvious, but standards help to regulate designs within the legal boundaries as to protect the designer from legal trouble. In the same vein, standards also keep designs within a realm of health and safety regulations. By adhering to standards within engineering, designers of business and consumer goods prevent harm from befalling the user. Aside from legal and ethical concerns, standards help to create optimal functionality for designs by setting guidelines made by working professionals with experience to know what pitfalls to avoid and what constraints hold true for certain design types. Standards in general provide a better product in every aspect in a design.

4.1 Power Supply Standards

For AC to DC power supplies there are standards that focus mainly on health and safety first and foremost. Health and safety are concerns with every standard imaginable, but when dealing with power and power distribution the safety concerns are the most apparent and anything not done to a set of industry wide standards could prove immediately disastrous not just from a functionality viewpoint. Power supply standards cover not only the power supply specifically but how the device as a whole protects against hazardous electric shock. Although most devices do, not all devices use basic grounding as protection against electrical shock. Some standards even go into the safety and standards of the lab equipment used to test the device so as to provide the most accurate readings form a safety point of view.

One particular standard for power supply standard is produced by CUI. The company known as CUI is known for producing electronic components, such as power supplies. CUI covers a wide range of components and have produced a standard for most all of their different components they produce. The CUI standard known as *Power Supply Safety Standards, Agencies and Marks* compiles different safety standards from other sources that CUI uses and extracts the power supply portion of those general safety standard. The standard IEC 60950-1 sets the boundary for the voltage of office machinery to be less than 600 V. For the Robofan, no singular portion of the device comes anywhere close to that level of voltage with the fan portion utilizing the highest operation voltage of 120 V. Within this same standard is a set of classifications of devices based on voltage level and how the device protects against electrical shock. The Robofan falls under the category of a Class I device. A Class I device protects the user from electric shock by being grounded along with every single electrical component being grounded or as the document specifies “Earth Grounded”. The design of the power supply for the Robofan follows the health and safety standards when planning and constructing this portion of the device.

IEC 60950-1 is an international standard on safety protocols for a whole spectrum of electrical design aspects, one of them being power supply design. Disconnection from the main power supply provided by design is vital for safety of the device and most importantly the consumer. When a power surge happens or an electrical disruption as severe as a lightning strike it is critical that the power supply has a fail-safe measure to prevent as much current flow as possible. Disrupting the current flow in a power supply follows the same logic as a breaker in a residential home’s circuit breaker being flipped to prevent as much damage as possible. For the Robofan a fuse may be the best way to implement such a fail-safe while still managing to be cost efficient as possible. Implementing a fuse into power supplies are a common method of protecting the device it provides power to.

4.2 Bluetooth Standards

Bluetooth standards aren’t as well established as some of the other standards within this section. This is due to the recent development of Bluetooth communication compared to the age of other technologies such as power supplies, and PCBs. Bluetooth standards primarily focus on the frequency range that a communication signal must be to be considered as Bluetooth. Due to Bluetooth’s usage of high frequencies it is easy for the signal to cause interference or to gain interference from other signals in devices such as wireless routers in a residential home or a device not even used for communication such as a microwave oven. The FCC has put strict regulations on what types of signals are allowed to be produced and the parameters they must follow. The penalties for not adhering to these regulations can range from fines to more severe consequences. That is why codes and regulations for Bluetooth and other communication are very rigid and blatant. The Bluetooth module used in the Robofan for communication strictly follows these standards with absolute compliance. There is normally a narrow band of frequencies on the high and low end of any communication signal range that is left unused as to not cause interference with other signals in the air waves. Besides interference the other issue that Bluetooth standards cover in detail are security concerns. One of the first industry set standards was set by IEEE and is still considered the most widely followed communication standard.

One of the most relevant standards used for Bluetooth today is IEEE 802.15.1 which is a subsection of a large and heavily detailed standard provided by IEEE relating to multiple uniformities wireless communication signals must follow. The main issue that becomes a recurring theme throughout this standard is interference. All wireless signals operate within a specified frequency range set by that countries communication regulation department, for the US this department is the FCC. The FCC sets the range for Bluetooth at 2402 MHz to 2480 MHz with a narrow range of frequencies on the high and low ends of that range to prevent interference with other wireless signals that may fall within the range for the Bluetooth module. The guard band at the lower end of the spectrum is 2 MHz wide and the guard band at the higher end of the spectrum is 3.5 MHz wide making the total range from 2400 MHz to 2483.5 MHz including the guard bands. These frequency constraints must be applied to the Robofan, otherwise the Robofan may not get a signal from the user's phone application in the garage due to Wi-Fi signals coming from the bedroom. This one example is one of hundreds that could occur if the proper range isn't set for the module. Luckily this wasn't as much of an issue as it sounds, when testing the Bluetooth module for the Robofan the frequency must be set to the proper range otherwise the devices won't communicate with each other i.e. the Robofan and the smart phone. But that doesn't exempt this team from making sure the standard frequency range is utilized. When designing the software for the phone application along with the firmware these standards were followed.

The security aspects of Bluetooth are addressed in the IEEE 802.15.1 as a main concern when designing a product with Bluetooth. The official website hosted by Bluetooth, the company, also provides their own custom standard which brings up the issue of security as one of their main concerns as well. Security for Bluetooth communication is vital for protecting the information of the user. If the Bluetooth communication is breached not only is the real-time video of the Robofan in question but all of the information attached to the smart phone in use. Security within the code of the firmware of the Robofan and the software of the smart phone application were implemented and discussed in their respective sections within this document. While Bluetooth hacking isn't terribly common it was implemented on the software side of the design.

The limitations of Bluetooth's data rate are brought into consideration in the last portion of the IEEE 802.15.1 standard. An extension known as the EDR (enhanced data rate) extension is widely used to solve this particular problem. These extensions are used to phase shift the envelope of the signal by a quarter-pi phase shift or an eighth-pi phase shift depending on the extension used. The smart phone application for the Robofan utilizes this extension as to optimize the communication of data transferred between the smart phone application and the Robofan's Bluetooth module.

4.3 PCB Standards

Standards for PCBs mainly focus on factors that aid in the safety and efficiency of how a printed circuit board operates and to ensure a certain level of uniformity is used within the industry. Adhering to standards for PCBs help to prevent faults when designing a PCB and avoiding costly errors that would nullify the proper operation of the PCB. Health and safety concerns are another important factor when designing a PCB that industry regulated standard keep in check. The PCB is one of the most integral parts of nearly any electronic device or appliance. If a PCB is designed with improper grounding or is not sound from a thermal point of view, then the entire construct could break beyond repair or worse yet put the consumer of the device in danger. Many organizations produce standards for PCBs, primarily IPC and IEEE. IPC is the Association Connecting Electronics Industries, this organization is connected to thousands of electronic manufacturers that adhere to the standards of the organization which they themselves are members of. IEEE, Institute of Electrical and Electronics Engineers, is a professional organization that centers on aiding technological advances. The members of IEEE are, as the name implies, electrical and electrical adjacent engineering associates in the industry. Both of the aforementioned organizations consist of members within their field who create and innovate, this gives weight to the authority both organizations have in creating standards due to their own involvement in the community and their own design of PCBs.

Starting with the broadest possible design standard for PCBs, is the IPC-2221B standard titled *Generic Standard on Printed Board Design* produced by IPC. This standard can be applied to any PCB design because it covers all of the PCB basics. The first subject this standard goes over is the material selection for the PCB. The material isn't limited the plastic of the board itself but also the adhesive, film, laminate etc., the list goes on in extensive detail. For the design of the Robofan's PCB, a licensed and industry trusted PCB fabrication vendor was used to ensure that any illicit or unconventional materials are kept out of the board design. Stated within IPC-2221B is a section dedicated to the mechanical and physical properties of the board itself. Using a fabrication vendor clears up the portion of the initial board being sound or not, but the issue remains if the board would be physically stable with the components put on the board at a later date. It is our responsibility to make sure that the PCB design has the right amount of thickness to support any parts and components we plan on implementing and attaching to the board itself. IPC-2615 *Printed Board Dimensions and Tolerances* addresses the issue of physical limitations of a PCB as well but in a more specific framework. This standard talks about the size and dimensions of the PCB and how that affects the physical integrity of a board.

The electrical considerations with the board are also brought up in IPC-2221-B standard. The electrical portions of this standard focus mainly on the design aspects we have the most control over, unlike the previous mechanical and physical portions that are, for the most part, taken care of by the board fabricator service. Electrical performance and power distribution are the first concerns addressed within the standard as far as design is concerned. With these design constraints addressed the other portions of the design can be set up with the power and electrical performance taken care of first. Materials are brought up again as far as conductance is concerned. For the PCB of the Robofan copper was used within the PCB so this standard was met with no effort required. Parts and component placement are addressed in the electrical clearance section. The electrical clearance is important so as to avoid noise as much as possible. Combing the concerns of the electrical with the physical issues of the board itself brings up the concern of heat. In both IPC-2221-B and IPC-2223B *Sectional Design Standard for Flexible Printed Boards* the conversation of thermal management and the precautions to take are laid out. The overall design of the PCB is brought into question when deciding what to do about thermal management. Both standards bring up heat sinks and ventilation as options for more complex boards with components that are known for heating up to degrees that could prove troubling for the device. As far as the design of the PCB for the Robofan is concerned, ventilation is being considered the most effective form of heat management for our specific PCB. Other thermal management considerations may be need if the design of the PCB becomes intricate to the point where greater heat dissipation is required.

Assembly of the PCB with the parts and components are the last design constraints to consider when forming the design to industry standards. Appropriately in many standards this is also the last section of each of these standards such like IPC-2221B. The assembly of the PCB as whole after the board has been received from the fabrication vendor is in full control of the customer. In the case of the Robofan the customer is us and the main parts that need to be considered are the Bluetooth module and motor controllers. When mounting these parts, following the proper soldering regulations within this standard are crucial. Failing to solder on components correctly can easily ruin an entire PCB in mere seconds. The ways a board can be corrupted by haphazard soldering range from ruining the component or board by burning through it physically to making the board unstable from an electrical stance by essentially “crossing the wires” by making a connection between tracks with misplaced solder. One of the most common mistake with placing components is not leaving enough room during the design process. This happens when looking at the schematic and seeing only where each part and component goes without visualizing the actual size of the parts and components.

4.4 C Programming Language Standards

We are going to use the standard C programming language to run the firmware on our PCB. C is fairly basic as far as high level software goes, but is very useful to us. C has similar basic standards to most operating systems, and as there is no set universal standards list we referenced Carnegie Mellon University's C coding standard.[1] rules such as you must declare identifiers before using them, do not depend on the order of operation, do not try to read uninitialized memory, do not modify constant objects, etc. We don't need to go through every single rule of the language, as that would take far too long.

The standard C language has quite a few rules, and like most programming languages we need to follow them for the code to work. We need to follow the naming conventions in place, identifiers should not have leading or trailing underscores, as these are reserved for system commands. #Defines should be in all caps, and Enum are capitalized or in all caps. Identifiers such as variables and functions should start with a lower case character. Our code should have useful comments so that the whoever is reading it can understand what variables and functions are and how they are being used. Global variables should all be declared before any functions. Generally individual .c files should be no longer than 1000 lines, as issues start to occur beyond that. We plan to follow these rules and more while coding in C, to ensure that our code works properly and can be understood easily.

5.0 Realistic Design Constraints

This section analyzes the design constraints necessary to produce our project. The goal of constraints is to set realistic boundaries on the scope of our project, as well as a basic framework to begin design with. In this section of the design document, we examine the various types of constraints relevant to our project, and how they affect our design. The constraints that were taken into account and shown are relevant by research are: economic, time, social and political, ethical, health and safety, and sustainability. When all constraints are tightly set, designing a project becomes very simple. Because constraints show what the largest bounds of design can be, we end up with a scope to fill. In short, economic constraints are considered due to the cost of producing the project, where the goal of following them is to produce a very cost effective end result. Time constraints reflect our milestones for producing the Robofan, and how difficult any given component is to fabricate, receive, and/or attach to our device to produce a working result. This section becomes one of the most important constraints due to the very short two semester time allocation we have to design and create the Robofan. The social and political constraints reflect the implications of what we are designing, and if there are any norms that must be followed and what the result of doing that successfully, or not, could be. The ethical, health, and safety constraints show that our device must be constructed to be safe and designed to help a user without harming anyone. Finally, the sustainability constraints show that a product must be made in such a way that its lifespan is as efficient as possible in terms of initial, and sustained cost of operation. In summary, this section shows how we examined all constraints surrounding our project, and followed them in our designs.

5.1 Economic Constraints

Robofan does not have a sponsor, and is being financed by the group members. This leaves us with a fairly limited budget, whereas some other groups with a sponsorship might have a much larger budget. Our initial projected budget was almost $300 USD, and we have purchased $612.34 USD of that by the end of senior design 2. A more specific breakdown of our expected budget and current variance is discussed in section 9.3. As this project is financed by the group members, the less money we have to spend, the better off we are. This is also meant to be a consumer product as opposed to one that a company would buy and use, so the lower our unit cost the better the product would sell to consumers. It doesn't matter how nice the fan is and how many features it has, if our unit cost were to be $1000 USD it wouldn't sell well. Due to this, we wanted to get the cheapest parts that met our requirements to keep the per unit cost down. For example, we calculated about how much motor speed and torque we needed for the stepper motor, and we chose the cheapest stepper motor that meets the speed and torque requirements. This is also good practice because if we were to buy too big and powerful of a motor we might have trouble with the extra weight and power draw, and especially if we were to overbuild several parts. This would end up with our project too heavy and too expensive. Another factor to consider is that our project needs to be lightweight, as it would be hanging from a ceiling mount and moving around. The less the project weighs, the cheaper the parts we needed to move it around. Heavier duty brackets, joints, and gears would also be necessary if the Robofan weighs too much, and these would drive up the unit cost.

Another factor to consider in the economic constraints is the power that the Robofan uses. This is discussed from an environmental perspective later in section 5.6.1, but the more power our product draws the more expensive it is to run. This means we should try to make our motors, motor controllers, and power converter as efficient as possible. We can also keep our product efficient by minimizing its movements while tracking, and having it limit functionalities over time if it doesn't detect anyone in the room. The Robofan is also somewhat modular in its design. If the end user were to have a problem with any of the motors, joints, or simpler boards (not the main PCB) it would be fairly simple to replace the damaged part. This could either be done by the end user or by a technician familiar with the Robofan's inner workings. This would make it last much longer, as when one part breaks you don't have to buy a whole new product. This also reduces waste, as the whole product doesn't get thrown away and replaced, just the broken part.

5.2 Social and Political Constraints

Social and Political constraints may not be the most apparent constraints considered when thinking of a household appliance. Concerns recently have been coming to light with the new wave of “smart” appliances that operate based off of memorized information from previous interactions that the system utilizes to give the consumer a more convenient experience than appliances of earlier generations. This information that the smart appliances gather can sometimes be obtained by way of hacking of unsecure systems put in place by the device designer. The Robofan is not a smart device, yet it does use a smart phone application and uses a type of camera as a sensor. Both of these features bring up the social and political constraints of consumer privacy and the security thereof.

5.2.1 Privacy

Privacy has been a highly discussed concern in the public consciousness over the recent years. This concern is naturally most voiced when considering devices with cameras. One of the possible candidates for choosing a tracking method is using a Pixy Cam. The Pixy Cam becomes less of a concern for privacy than other cameras due to the fact that recording footage is not possible with this particular hardware. This would alleviate privacy concerns except for the fact that the Robofan also has a phone application. This brings up issues of real-time surveillance providing someone could hack into the phone application and access the current video the Pixy Cam is transmitting. Such breaches in privacy have been proven to happen in web cameras on consumer computers and smart phones. To ensure the Robofan does not breach the privacy of the user proper security measures were taken when programming the firmware and software of the device. Security was also to be put in place when concerning the Bluetooth wireless communication as discussed previously in section 4.1.3 Bluetooth Standards.

5.3 Sustainability Constraints

All design should revolve around constraints. One of the most important constraints to consider while in a design process is sustainability. Sustainability is defined as the ability to be maintained a certain level. We are concerned primarily with the life cycle of our project in this regard. We also need to consider power usage, and material components of our device.

The life cycle of a device is a 5 stage cycle.

1. Purchase of the device
2. Installation
3. Maintenance
4. Removal from service
5. Disposal

The goal of designing for sustainability is to prolong the life of the device before it is removed from service, and disposed. This is done by prolonging the period in which the device can be maintained. There are several ways to ensure the device can be easily maintained. All parts of the device should be individually replaceable. This is done by using a modular design, that focuses on having all parts be removable without removing other components as well. This means that every component should be as self-contained as possible.

Price of operation and maintenance is another strong aspect of sustainability design. If a product consumes too much power, or requires too much maintenance it can be considered too costly to keep in service. If a device is removed from service before it has fully lived out its projected life cycle duration, the sustainability design has failed. Power consumption can be controlled by effecting when the device is active and consuming power. A product should use as little power as needed, whenever possible. This can be achieved by producing a simple design with few components. Further, power consumption can be mitigated by design that allows for idle modes to be used as often as possible. These modes would consume less power than standard operation, and have a minimum extra power needed to resume normal function. A product should attempt to self-regulate power consumption to prolong its own sustainability. This is important because users can forget to control their own devices efficiently. Maintenance is an important part of sustainability as well. The product needs to utilize components that need little maintenance. As a product needs more maintenance, it becomes more expensive to operate which thusly reduces how long it stayed in service. However, a device should not utilize components that are so high quality, and require minimum maintenance, that they are prohibitively expensive. The product needed to be in a competitive and reasonable price range for what it does. Designing with sustainability constraints needs to consider the initial price of the device as the sum of man hours required to produce it, and the price of components. This initial price needs to be laid against the cost of maintaining the individual components. This is done by analyzing how often a component is expected to break or need to be repaired and how costly these are. It is important to recognize that no product lasts forever, and a realistic lifetime needs to be set for the device to fail within and have an acceptable removal from service.

Our device is designed with these sustainability constraints closely in mind. Every component of our device is replaceable. This is done by giving it a specific placement within the housing that it anchors to and can be removed from. The entire product can be taken apart in such a way that all components are reachable. This is possible because all components have a single specific purpose. Our device manages its own power consumption. The fan is only active when tracking a target that requires air. It goes into idle modes when not tracking targets very often. It only moves motors when absolutely necessary. All of these factors limit the products power draw. Admittedly, cheaper components were focused on more than pieces with more durability and ease of maintenance. This is because the baseline duration without maintenance for the motors, fan, and camera we selected is all acceptably long. Our upfront total cost as covered in the budget was already as high as we wanted to go without a sponsor. Additionally, because we are designing the prototype as a device for personal use, rather than commercial we felt that this design choice was acceptable. Any duration of use without maintenance or replacement parts is going to be within what we consider an acceptable duration. This is our sustainable design.

5.4 Health and Safety Constraints

The Robofan would be mounted on the ceiling, so it won't have the same safety concerns as a normal oscillating fan. It is unlikely that a child would be able to get to it and hurt themselves, and the fan is meant for a workshop or garage, which children shouldn't be allowed free roam in. Despite this, we should still design Robofan so that a child wouldn't be able to hurt themselves with it. The only thing that a child would be able to get to is the wall plug and the wall switch. This means that we need to make sure that the wall switch enclosure so that it cannot be opened by a child, and should require a screwdriver to open. We need to of course make sure there aren't any open wires, and make sure that a child or pet couldn't break the wires open easily. Another problem to avoid is the fan falling on someone or something. If the fan were to fall on someone's head they could get seriously injured, or if were to fall on something fragile it could break it. Also if the fan falling were to break wires it could leave live wires hanging, which could be an extremely dangerous situation.

To avoid any wire shorts or dangerous situations involving shocking, we are going to use a fused plug that blows the fuse if the wires are shorted anywhere, thereby minimizing the shock or short circuit. The circuit breaker for the outlet would most likely flip if the outlet was shorted, but we wouldn't want to leave fault detection up to the outlet, as we can't guarantee what conditions our product is used in. A problem we have encountered already with safety, is with the cage on the oscillating fan. The oscillating fan can be run just fine without the cage over the fan blade, but this leaves the blade open to make contact with anything that strays too close. Unlike a ceiling fan that is largely out of reach and moves a bit slower, the oscillating fan blade is smaller and spins much faster, making it more hazardous. This combined with the fact that Robofan moves around and reaches lower into the room and a normal ceiling fan, the blade could be very dangerous. Due to this, we have of course decided to leave the oscillating fan cage attached to the Robofan. This should make any contact it might have with a person's head nothing more than a slight bump. We are not able to guarantee that the Robofan won't bump into people if they walk under it, depending of course on the height of the ceiling. To alert the user of this, we would include in Robofan's instruction manual that the user 'should stand at least 1 foot clear of Robofan when turned on to avoid collisions that could damage the user or the fan. The cage from the oscillating fan comes in two pieces, a front piece and the one that is attached to the core of the fan. The joint between these two halves feels slightly flimsy, as is was not meant to move like we are planning. We are going to look in to securing this with a better connection, to avoid any possibility of the cage splitting while in use and creating a potentially dangerous situation.

Robofan needed to be tested, and this required something moving around for it to track. We of course cannot test on any kind of animal, as it might cause unnecessary stress to the animal. An animal would also be less than ideal, as we cannot control how and where they move, and this product was not meant for any type of animal. This leads us to the conclusion of testing Robofan on ourselves. This might sound unsafe, but there should be no real danger unless the user gets too close to it and a collision occurs. We had to be careful in our first tests and have someone ready to flip the off switch on the wall if any unintended behavior occurs. If this occurs for too long the Robofan could damage itself and possibly its surroundings. We must also avoid the use of any toxic substances. This includes but is not limited to toxic paints, lubricants, chemicals, lead paints, and mercury. Later in section 5.6.1 this is discussed relative to environmental concerns and that the product needs to avoid anything that would not be safe to throw away normally.

Robofan is not be able to be able to record and save any amount of video. The Pixy camera is not set up to export video to the PCB, and further the PCB would not have any way to store it. The Bluetooth connection that links the phone app to the PCB does not have enough throughput to transfer video at any reasonable speed, and the phone application does not even have permissions to connect to any outside device. This in combination with the passcode to link to the Bluetooth module prevents any individual or the creators from using the Robofan to spy on the user in any way.

6.0 Project Hardware and Software Design Details

The details of our hardware and software need to be researched so actual design work can begin on the Robofan. First the initial plans and diagrams for the project are discussed. Wiring diagrams, and software diagrams, are featured in this section. Next, 6.2 and its subsections have comprehensive comparisons for possible Robofan components. This portion of the report is important as it allows the team to select the best components for the Robofan. For example, what types of motors are used for the tilting motion and the panning motion? Are they even be the same, why or why not? These are the types of questions that need to be answered in order to create an optimized Robofan. The firmware subsection details how communication and signal delivery between the various components are handled by code. Power supply options are examined in section 6.5. The Pixycam has an entire subsection dedicated to it. The camera's detection capabilities, output, connection and wiring details are topics that are discussed further in this paper. Security features for the phone app is described in 6.7. Finally, ending section 6 is a comparison of microcontrollers and stepper motor controllers

6.1 Initial Design Architectures

To being the actual physical work of the Robofan some initial architectures need to be established. The overall design needs to be fully realized in a visual medium that communicates the functionality of the device. A three-dimensional representation was used to convey this. An ideal of how all the portions were wired together needs to be created before components and more specific design concepts can be realized. This was remedied by an initial wiring diagram to explain a broad idea of how the electrical system within the Robofan is connected. The following sections expand on these designs and how they'll be implemented.

6.1.1 3D Design

To model the physical whole of the Robofan a three-dimensional virtual model has been made to give the design the optimum visual representation before construction of the physical housing of the device begins. The size and dimension of the functional components are rigid values that have to be taken into account before the design takes place. Aside from the fan portion of the Robofan salvaged from a desk fan, the outer housing of the fan is all the user sees. This outer housing comes in three main sections as follows:

- Panning Base; housing the mount, wiring, panning motor, and slip ring
- Tilting Arm; housing the wiring, tilting motor, and fan shell bracket
- Fan Shell; housing the wiring, PCB with components, and fan motor

The three sections above can be constructed in a variety of different ways. The one that the 3-D model would pertain to the most would be 3-D printing out the pieces. Whether or not each piece was constructed by way of 3-D printing or not is discussed in their own respective sections of this document. When modelling the Robofan the ability to display each of the three sections as a whole as well as individually should be a top priority. Being able to isolate each section gives more degrees of freedom to the overall design when deciding on a course of action when constructing the physical housing of the Robofan. If the sections are designed where they can't be isolated, then the only way to 3-D print would be to decide that the housing as a whole needs to be printed. Having the sections separate allows for the option of printing certain portions and constructing the others in another manner, providing we chose to 3-D print at all. The ability to meet this need is largely dependent on the design itself and the capabilities of the 3-D modelling software.

6.1.1.1 3-D Modelling Software

To initialize the process of creating a three-dimensional object a choice of 3-D modelling software must be made. When choosing the software that we are going to use for the creation of the three-dimensional representation some considerations come to mind. The first is that the software creates files that can be easily given to a 3-D printing service for manufacturing. Having a software that is common within the engineering industry is optimum for explaining the design to other engineers, technicians, and potential clients for the product your describing. Using a software that falls in line with this field is great experience for entering the electrical and computer engineering field. Clarity is one of the largest issues engineers have to combat when expressing an idea to others, utilizing the proper software helps eliminate that barrier. Ease of use is a highly desirable trait in looking for a software since the members for the Robofan project team have limited to no experience with 3-D modelling.

6.1.1.2 AutoCAD

AutoCAD is a software that can holds a whole host of different design functions which include both 2-D and 3-D modeling. Since AutoCAD is used so extensively in the engineering industry utilizing AutoCAD probably yields results better suited for the design of an engineering project like the Robofan. The design toolboxes and applications within AutoCAD appear simple to use as being well as user friendly from the material and tutorials we have researched.

As far as utilizing a skill for the Electrical and Computer Engineering field is concerned, the use of AutoCAD would be highly beneficial and applicable to many career paths. AutoCAD is used extensively in designing electrical components and schematics, but could also be used in a mechanical approach such as housing for the Robofan. Having the ability to use AutoCAD effectively can greatly benefit the team in their future projects and endeavors moving passed the Robofan project. The clarity AutoCAD provides across engineering disciplines greatly contributes to its possible selection for 3-D modelling for the Robofan.

The usability of AutoCAD may be an easier transition than most software due to its popularity within industry. After a quick examination of the most popular 3-D printing services, every single printing service accepted CAD based software designs. Being as AutoCAD is one of the most common software choices out of the CAD libraries we would be at a loss trying to find a 3-D printing service that would not accept a CAD design. Converting a 2-D design is possible in AutoCAD with multiple free tutorials walking you through the steps are widely available online.

6.1.1.3 Solidworks

In the mechanical engineering community, Solidworks is a highly used program in industry for representing projects in a three-dimensional way and to show intricate moving parts individually and how they come together in the large scope of the project. The possibility of printing Solidworks designs are attainable because Solidworks can be saved to an STL file which is an industry standard within multiple engineering disciplines.

In a similar situation to the availability of AutoCAD in the previous section, many computers at the University of Central Florida have computers free to use for students that have the latest version of Solidworks as an application. The computers that contain Solidworks are available nearly 24-hour accessibility except for the rare scheduled events that occur within the campus. Availability and not having to add to the financial aspect of this project alleviates some constraints when factoring in time, effort, and money. Being as the 3-D model is only taking into account the mechanical physicality aspects of this project, this software is suitable for the needs of the Robofan.

6.1.1.4 Blender

Blender is a free three-dimensional modelling software compatible on both Microsoft and Apple based operating systems. The main appeal of this software is that every member can get this software regardless of the type of computer they are using. Blender is not offered on the computers of the University of Central Florida, but this is not concerning do to the convenience of having the 3-D modeling software on the team members' personal computers. This software seems catered to consumers more concerned with visual aesthetics rather than conveying engineering ideas and functionality. This would be optimum if this was an art project but may not be ideal for conveying ideas to others in the industry about the functionality of the design of the Robofan.

If this 3-D model is used for printing out portions of the housing for the Robofan this program allows for STL files to be made for 3-D printing. STL files are one of the file types that most commercial 3-D printing services utilizes outside of CAD files. This fills the requirement that the software we need to utilize a file type compatible with 3-D printing. When considering this software over the other choices in this section the choice of convenience and ease of use are weighed against the cost of efficiency and clarity.

6.1.1.5 Autodesk 3ds

A viable software to create our three-dimensional representation of our project is Autodesk 3ds. Out of all the software discussed this is the most intricate and variable with how the design is created. Similar to Blender, Autodesk offers a more artistic representation with different textures and a very user friendly interface for creating the model of the Robofan. The largest downside to using this program is that it is a licensed software that requires payment. Autodesk is also a distributor of the AutoCAD program so it possibly uses a shared or similar tool set and reference library for creating a 3-D model. The main consideration when comparing Autodesk 3ds to the other software options on this list is how much the final 3-D design benefits from the extra features Autodesk 3ds has against how much it costs financially and if it is worth the cost.

6.1.1.6 3-D Modelling Software Choice

The choice of 3-D modelling software for the design of the Robofan is AutoCAD. AutoCAD was primarily chosen due to the constant availability afforded to the members of this project. The availability for AutoCAD was a large deciding factor in choosing this particular software over the others. While Blender may have been free and there was a free trial available for Autodesk 3ds, both of these programs are less rigid and not as tailored to creating mechanically sound projects. They have quite a few features that allow for easy customization of the model but unfortunately in small batch manufacturing it would be more challenging to follow a design based in these programs. Solidworks wasn’t chosen over AutoCAD only because AutoCAD is used more than Solidworks in computer engineering and electrical engineering. The project members for the Robofan all have a computer or electrical background so the choice of AutoCAD over Solidworks is more of a choice in harnessing skills that serve the team members of this project in their future careers.

6.1.2 Initial Wiring Diagrams

Figure 6-1 on the following page, is a broad first draft of our wiring diagram. This image is very basic and is more for allocation of what each person worked on throughout the project. This image still has a few things that we have since changed, for various reasons. The notable changes are the security camera is now a Pixy camera for ease of use and programming. The fan motor is now powered directly from the AC power, as it is in the stock fan we are using as a base.

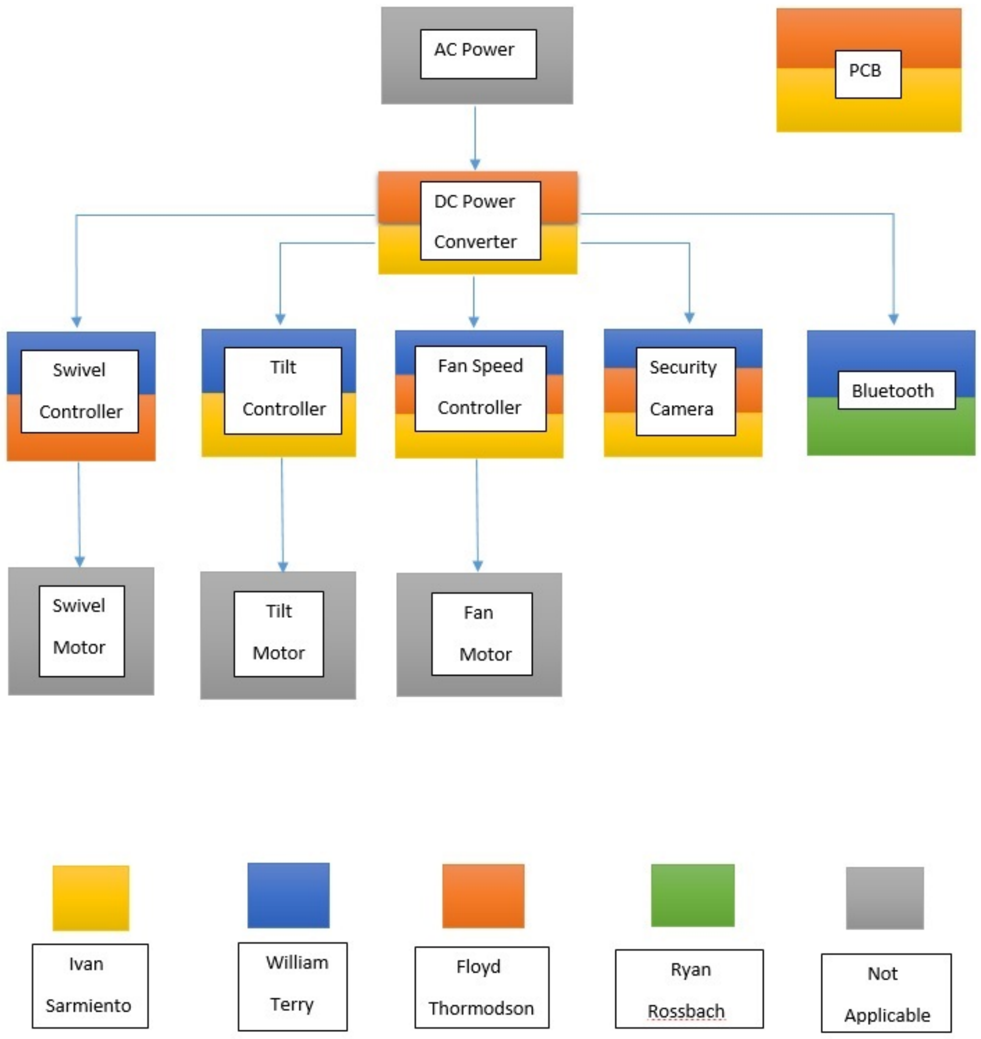


Figure 6-1:Initial Hardware Block Diagram

Figure 6-2 below encompasses all of the wiring in the Robofan. This diagram is more specific to the parts and connections than the previous image. The source power is a plug straight into normal US wall power. This runs at 120 volts AC, and is perfect for running the large fan motor that drives the fan blade. We considered using a battery to power the fan, however this would involve either a large battery or limited runtime, as the fan motor needs a substantial amount of power. A battery would also be more expensive and bring the price of the fan up considerably, whereas a wire into a plug is very cheap and easy to come by. The fan is also mounted to the ceiling and not meant to be moved, so a plug works well. Not shown in Figure 6-2 is the switch we are going to splice into the wire that goes to the wall plug. This switch would either stick to the wall or be screwed in, so that the user can turn the entire Robofan on or off easily and without using the companion app, provided this were put in for production. We considered putting the switch on the outside of the fan casing but that might be hard to reach, and especially if the fan is moving around.

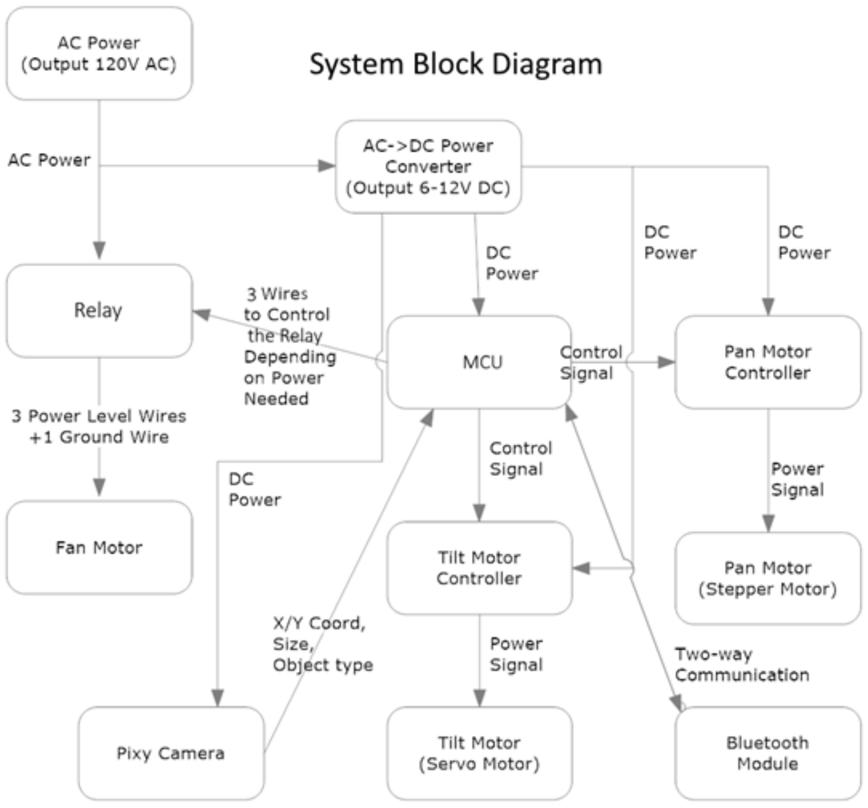


Figure 6-2:Hardware Wiring Diagram

The power to the fan motor is controlled through a series of relays. These are basically switches that are toggled electronically, and can handle plenty of current running through them. These relays are controlled from the MCU and there is one for each of the power levels which are essentially low/medium/high. If all three relays are toggled to off, the fan motor is getting no power and this makes a fourth state, off. We do have to be very careful connecting the relays to the PCB, as if it is connected improperly we could put 120 volts through the PCB and possibly anything connected to it. This could burn out these components and would cost valuable time and money. To avoid misconnecting the relays we used the much less expensive and readily available test board that we have, which is a Launchpad microcontoller. Figure 6-3 below is the side of the main fan motor. The four wires going directly out of the bottom of the figure are the wires that came connected to the four buttons, which we are connect to our three relays. The extra button was the off button that released the other three buttons when pressed, to turn the fan off. We did not need the fourth button as our MCU can just turn all of the relays off. The extra blue and red wires going off to the side attach to a large capacitor that is on the motor. This capacitor is for power factor correction to keep the motor efficient. The motor is running off of AC wall power, and is a mainly inductive/resistive load, so we need the capacitive load to cancel out the as much of the inductive load as we can, to minimize reactive power.

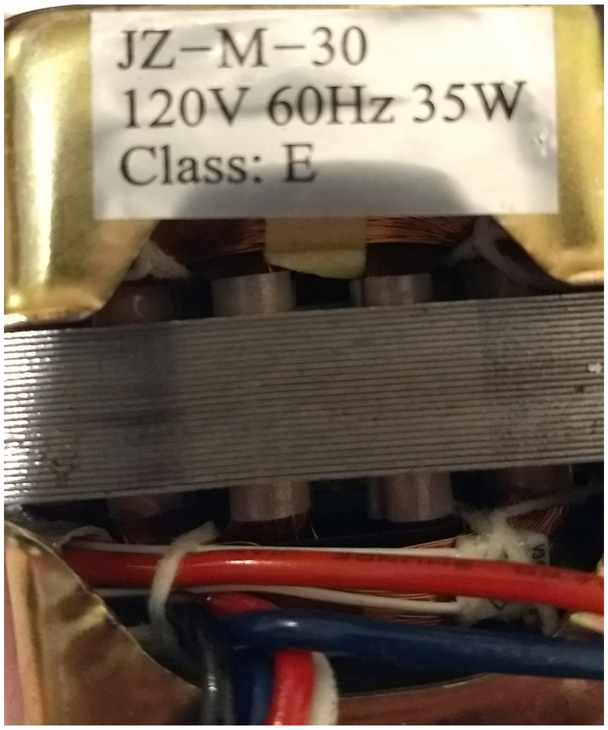


Figure 6-3: Fan Motor Connecting Wires

The AC/DC power converter takes in 120 volts AC and output DC at a much lower voltage, somewhere between 5 and 10 volts depending on what is needed. This supplies power to the stepper motor controller, the servo motor, the Pixy camera, the Bluetooth module, and the MCU. The Pixy camera takes in DC power from the power converter, at 5V. It is expected to take in about 140mA[2] average while running. The power connector is the white 2 pin connector at the bottom right of the board, shown in the image below. The pixy camera can also receive power through the USB plug (image top left), or the I/O ribbon cable (image top right). The Pixy camera is also connected to the MCU, where it sends its output. The Pixy camera outputs X and Y coordinates, the approximate size of the object, and the type of object based on what it has learned. The MCU can then use these coordinates and the knowledge of where the center of the fan is blowing to send out commands to the motor controllers to move the motors appropriately. To connect the Pixy camera to the MCU we have many options, but for this project we expected we would most likely end up using the generic pins which in the end is what we utilized.

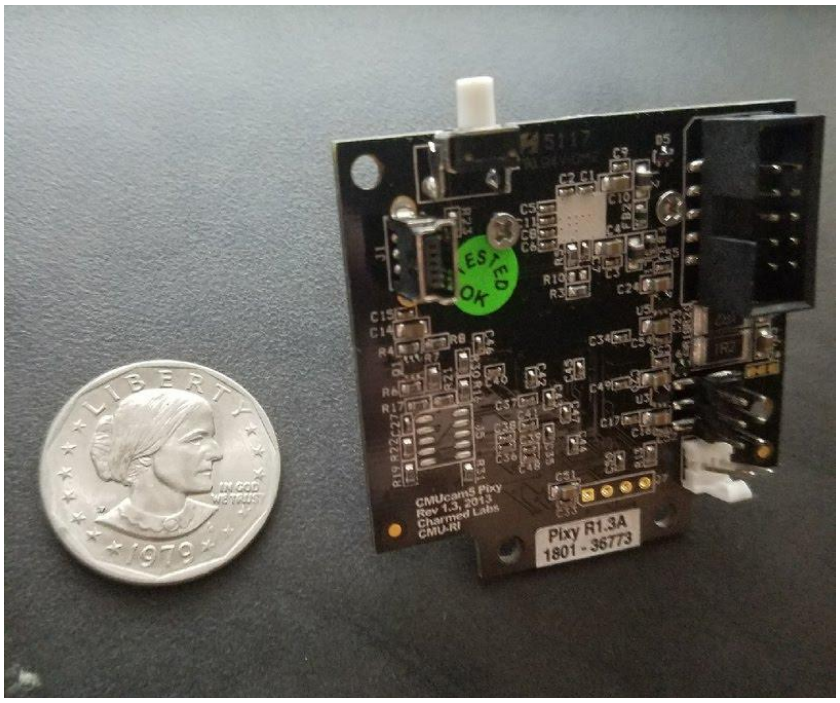


Figure 6-4: Pixy Camera Rear View

The stepper motor controller receives DC power from the power converter, and commands from the MCU. The stepper motor is connected to the controller via its 4 wire plug. The stepper motor isn't powered like a traditional DC motor where it just receives DC power. The stepper motor is instead powered by alternating current, over two phases. The black and green wires essentially power one phase, while the red and blue wires power the other phase. To turn the motor, the stepper motor controller alternates which phase is being powered, with each switch being a step. The faster the stepper motor controller cycles the faster the motor turns, and the more current the controller supplies, the more torque the motor has. We control these with the input from the MCU, and it pulls whatever power it needs from the power converter.

The servo motor is controlled differently than the stepper motor. We are using the plug that the servo motor comes with, which is a 3 pin connector. The red pin is the positive from the power supply, the black pin is the negative from the power supply, and the white pin is the control signal. The red and black pins come straight from the power supply, and supply whatever power the servo motor needs. The voltage on the white pin tells the servo where to position itself, for example +5V would be all the way to one direction, with -5V being as far as it can rotate in the other direction. Something to keep in mind is that since the servo motor relies on the analog voltage, so we need to keep the resistance of the connecting wire low to avoid voltage drop across the wire. This can be accomplished by keeping the connecting wire as short as possible.

The Bluetooth module has 6 pins which are labelled on the board. The GND connects to the negative terminal from the power supply. The VCC connects to the power supply positive. The VCC can handle between 3.6-6V as printed on the board, but we used the recommended 5V. The STATE pin reads to the MCU and is low if Bluetooth is not connected and high if Bluetooth is connected. The EN pin takes the command from the MCU of whether it is supposed to be the slave module or the Master module. More information on the Slave/Master aspect of Bluetooth can be found in section 6.3.2. The RXD pin is the receiver pin that receives data from the MCU to send to the paired Bluetooth device. The TXD pin is the data pin that outputs the data from the paired Bluetooth device to the MCU. The RXD and the TXD pins are what allow the Bluetooth module to have two-way communication to the MCU. In addition to the pins that go to the MCU for information, the Bluetooth module of course has two-way communication with the phone application through the connection to the phone's Bluetooth. This makes the bridge between the two code flowcharts shown later on.

6.2 Mounting and Gearing

The Robofan is a project being created by electrical and computer engineering majors, but requires a fair amount of mechanical implementation. The autonomous movement of the panning base and tilting arm of the device require electro-mechanical integration to achieve the desired result. The gearing and mounting portions of the Robofan represent how we plan to design the Robofan in its mechanical aspects, i.e. panning and tiling. The following sections go into detail about the subsections of the entire design from their specific mechanical function, how they attain this function, restraints of the design and what specific parts we integrated into the Robofan to obtain the desired range of motion and other functionality aspects.

6.2.1 Panning Base

There are two primary motions the Robofan uses when tracking an object i.e. the user, panning and tilting. The base of the Robofan allows for a continuous panning motion. The physicality's of how the continuous motion is obtained and how we address the issue is discussed in section 6.2.3. To obtain the desired panning motion a device, most likely as motor, must be implemented in order for the tracking to be swift and effective to the specifications of the Robofan. The precision of the panning mechanism was within a 3 degrees range high or low to the desired target.

6.2.1.1 Panning Mechanism constraints

The robotic/autonomous panning motion of the Robofan is electronically by way of a motor controlled by a sensor with programmed parameters. The panning mechanism required the highest amount of torque to allow the Robofan to operate within the design specifications defined within this project. The fan base is affected by the size of the motor we choose. The type of motor that fits within the constraints of the panning base limitations would be a stepper motor. The specific type of stepper motor chose for the panning base is discussed in section 6.2.7.

6.2.1.2 Panning Base Design

Designing the panning base of the Robofan is primarily dependent on the components within and attached to the swiveling base. The current design of the Robofan employs the use of a stepper motor to pan the base of the Robofan. The stepper motor must be utilized internally with respect to the housing due too cosmetic and safety reasons the motor would be housed within the center of the base of the current design to allow for the rest of the Robofan to swivel while the stationary portion remains mounted to the solid surface. With the current design the panning base must adhere to the constraints of the stepper motor. The design of the panning base has been done in the 3-D modelling and drafting program AutoCAD to show the shape and desired dimensions of the panning base.

To design the panning base of the Robofan, 3-D software that we used to provide the best visual representation with desired clarity for presenting the panning base in conjunction with the rest of the fan. The size of the stepper motor, the apparatus that allows for continuous rotation, and the wiring with the joint of the panning base was to be the deciding factor in the size of the panning base. Since we decided that the panning mechanism was to be a stepper motor, the size of our design is not as variable since stepper motors are typically larger than servo motors. On average, when looking at stepper motors for small appliances such as the Robofan, the dimensions rarely exceed than 5 x 5 x 3 inches in length, width, and height respectively. The most variable and crucial part of the panning base design is the opening where the tilting arm was to sit. The size of the base, as stated before, is largely dependent on the size of the swiveling based components within, but the real deciding factor was the circumference of the tilting arm. This stipulation in design requires the tilting arm dimensions to be set before the panning base can be fully realized.

6.2.1.3 Panning Base Construction

Construction of the panning base greatly affected the cosmetic design of the Robofan as far as how it looks to the user. The housing of the mechanical and electrical hardware is the only part of the device aside from the premade fan with attached cage that the user/potential customer would see. The panning base is one of the main visual focal points for the Robofan aside from the tilting arm, fan housing, and fan blades with premade cage. Cosmetic aspects of the project are a lower priority than functionality of the panning base There are three different methods of fabrication for the panning base of the Robofan. The first option is the take the 3-D model of the panning base created in AutoCAD and send it to a business that offers 3-D printing services. The second option would be to assemble the entire panning base, aside from the swivel/panning mechanisms within, by hand from bought materials from a hardware store or comparable hobbyist store. The final option for constructing the panning base is the salvage the base from a similar device such as a fan or light fixture and adapt the design of the other 3-D design portions. In the final product, the panning base was a set of gears connected to the base of the baring.

6.2.2 Tilting Arm

Aside from the 360 degrees of swivel motion of the Robofan, the most important aspect of motion is the tilting movement of the device. To obtain this motion a tilt joint is to be implemented to allow the motor to tilt up and down towards the direction where the pixy cam has detected the desired target. The tilt joint is controlled by a motor with an accuracy of 1.8 degrees per step. Allowing this level of accuracy, the fan should aim at the user with perceived pinpoint precision due to the nature of the fan. The choice of fan has been based on size and weight due to the size and weight constraints of the tilting joint and motor respectively.

6.2.2.1 Tilting Mechanism

The autonomous tilting motion of the fan happened electronically which means a motor must be implemented to move the joint up and down in a vertical fashion. The choice of motor for the tilting mechanism changes the design cosmetically, the design of the PCB, and the primary source of power for the tilting joint. The tilting mechanism had to meet a set of requirements and constraints to provide the tilting with the optimum viable part.

6.2.2.1.1 Tilting Mechanism Constraints

The three most applicable motor types that would be utilized in a tilting joint for a small appliance type device like the Robofan would be either a DC motor, a stepper motor, or a servo motor. The tilting mechanism chosen had to meet all of the requirements listed below:

- High torque, needs to be able to lift the entire fan mechanism ~ 5 lbs.
- Accuracy, required to lock onto the desired location within a 5-degree margin of error.
- Cost effective, at the very least should be comparable in price to other viable options.
- Within the physical dimensions of 3.5x3.5x3.5 inches of the total tilting mechanism to allow for a reasonably sized tilting mechanism.
- Low vibration, to not interfere with the other components as well as reducing noise as much as possible.

6.2.2.1.2 Tilting Option 1: DC Motor

A DC motor, as the name implies, is a motor that operates on direct current. Simplicity is the greatest advantage to using a DC motor. The speed of a DC motor is controlled by the amount of voltage you supply the motor. The initial investment in a DC motor is quite cost efficient when compared to a stepper motor or servo motor. Many DC motors can be purchased around $6 USD. While this may appear to be a very appealing option price wise, a gear box would be another investment that would be necessary for any motion of the tilting arm. The average price of a DC motor with a gearbox included is $14 USD which brings the average price comparable with both stepper and servo motors. Adding the necessary gearbox to the DC motor would make the dimensions of the DC motor larger than what we plan on designing the width of the tilting arm to be.

6.2.2.1.3 Tilting Option 2: Stepper Motor

Stepper motors are brushless motors that provide rotation in a series of equal steps. The torque in a stepper motor would be more than enough to lift the 5 lb. load of the fan mechanism of the Robofan. The size of a stepper motor would be within the feasible range if only the motor itself were to be the tilting mechanism. A gear box has to be applied in conjunction with the stepper motor to get the tilting joint of the arm to move. The width of the stepper is increased by a minimum of 2 inches with even a relatively small gear box. Adding a gear box brings the average width of a stepper motor to a value greater than 5 inches. Utilizing a stepper motor for stepper motor would limit the design to having the tilting mechanism on the outside of the joint, providing opportunity for the mechanism to become damaged and the design to be heavily limited.

6.2.2.1.4 Tilting Option 3: Servo Motor

A servo motor is a motor that uses either a linear or rotary actuator to provide position feedback to determine and control the final position of the motor. Accuracy is the leading advantage a servo motor has over a stepper or DC motor. The positional feedback allows for the motor to rotate in either direction if the position is not that of the desired input. Unlike the two previously motors, a servo motor does not require a gear box to move the tilting joint of the arm. Having the lack of gear box on the motor saves a lot of space when concerning the entire tilting mechanism. The average size of a servo motor with the adequate amount of torque for this project is less than 2x2x2 inches. The cost of a servo motor of the desired torque and size is rated at an average of $15 USD, which is within the same price range as the previous two options.

6.2.2.1.5 Tilting Mechanism Choice

When weighing the three options for a tilting mechanism the servo motor achieves all of the goals listed of fitting within the constraints listed in section 5.0. Cost effectiveness was met by all three options after researching a variety of motors. The total average cost range for the servo and the DC motor fell within $15-$20 USD including the gear box and mounting apparatus. The stepper motor became slightly more expensive when a gear box was factored in bringing the total price to nearly $40 USD. Having a gear box attached to a motor increases the size of the total tilting mechanism by at least 2 inches in width. The servo motor was the only tilting mechanism option that did not require a large gearbox. All motors in their price range and sizes previously mentioned are found in a variety that contain enough torque to meet the 5 lb. load weight of the fan mechanism. More detail about the type of specific servo motor to be used in the Robofan is found in section 6.2.5.

6.2.2.2 Tilt Arm Design

Designing the tilting joint of the Robofan is primarily dependent on the components within and attached to the tilting joint. The current design of the Robofan employs the use of a servo motor to tilt the joint within the tilting arm at the joint. The servo motor can be utilized externally with respect to the joint, but due too cosmetic and safety reasons the motor could be housed within the joint with the current design. With the current design the tilting joint must adhere to the constraints of the servo motor. The design of the tilting arm was done in the 3-D modelling and drafting program AutoCAD to show the shape and desired dimensions of the tilting arm.

The 3-D model has been designed within the rest of the whole design located in section 6.1.1 the Initial 3-D Model.

To design the tilting joint of the Robofan, 3-D software has been used to provide accuracy and the best visual representation of the tilting arm in conjunction with the rest of the fan. The size of the motor and the wiring with the joint of the tilting arm would be the deciding factor in the size of the tilting arm. Since we decided that the tilting mechanism is a servo motor, the size of our design can vary since servo motors are typically smaller than stepper motors. On average, when looking at servo motors for small appliances such as the Robofan, the dimensions are rarely larger than 2x2x1 inches in length, width, and height respectively.

6.2.2.3 Tilt Arm Construction

Construction of the tilt joint is one of the most crucial decisions that would affect the cosmetic design of the Robofan. The fan portion of the Robofan is prefabricated and won't change much, cosmetically, with the design of the rest of the device. The tilting arm is one of the main visual focal points for the Robofan aside from the fan and the base. Cosmetic aspects of the project are a lower priority than functionality of the tilting arm. There are three different methods of fabrication for the tilting arm of the Robofan. The first option is the take the 3-D model of the tilting arm created in AutoCAD and send it to a business that offers 3-D printing services. The second option would be to assemble the entire tilting arm, aside from the tilting mechanism, by hand from bought materials from a hardware store. The final option for constructing the tilting arm is to buy a premade arm assembly that comes with a movable joint that we can integrate into the design of the Robofan.

6.2.2.3.1 3-D Printed Tilting Arm

Utilizing the design from the 3-D software we used allows for the tilting arm to be constructed with a 3-D printer. Since the current price of owning a 3-D printer along with material for 3-D printing isn't cost effective, a service that offers 3-D printing could be used as the resource for our 3-D printed tilting arm. The time in printing a CAD design for most 3-D printing services average on 3-5 hours for small designs and as much as 24 hours for larger or more complex designs. The design of the tilting arm for the Robofan was not very complex because the tilting mechanism and wiring has been attached during the construction which is the most complex part of tilting arm. The joint is the only portion of the design that needs more attention, due to the minimum size requirement to house the servo motor and associated wiring.

6.2.2.3.2 Self-constructed Tilting Arm

Making the titling joint by hand allowed the tilting arm of the Robofan to adapt in design the easiest out of the other options for constructing the tilting arm. Compared to the previous methods constructing the tilting arm by hand would most likely give the most degrees of freedom. The choice of material, size and shape are all malleable when constructing the tilting arm. If an issue arises from the design process occurs, such as an issue with the allotted dimensions within the arm, then the proper adjustments can be made during the construction process rather than ordering a premade arm with joint or ordering a 3-D printed arm and having the entire arm be an issue. In this instance, creating the arm by hand would be more cost effective as well as time efficient. In general, making the tilting joint is less expensive than the other options but it won't be as time efficient as ordering a 3-D printed arm (providing no modifications are needs) or a premade tilting arm.

6.2.2.3.3 Premade Tilting Arm 1.3.3

Utilizing a premade tilting arm that can be bought or salvaged would be integrated and adapted to meet the needs of the Robofan. A wall bracket for a television set comes with flat bracket on the base that would normally mount to a wall and a bracket that the television would attach to. Changing or fitting the brackets to attach to the base of the Robofan and the fan mechanism would allow for simple mounting of the entire device. The price of the entire tilting arm without modification can be purchased between $20-$30 USD. This proves cost effective providing any adjustments and modifications don't cost more than the bracket itself. The main drawback of using such a joint is the rigid nature of the tilting arm would not allow any freedom to adapt the dimensions without welding, which is not reasonable for the scope of this project. The servo motor would be exposed by having no housing if a television bracket is utilized. The functionality would be the same as having he motor within the fan, but possible damage to the motor or the wiring is less than optimal for the Robofan.

6.2.2.4 Final Tilting Arm Decision for the Robofan

The tilting arm of the Robofan has been based off of the 3-D model created, but it was constructed in-house by hand. The versatility of creating the tilting arm by hand allowed for more diversity in materials to be used than if a 3-D printer made the tilting arm. The self-constructed arm was also made out of more durable material than the plastic substance a 3-D printed arm would be congealed out of. The plan for the material of the tilting arm is to be constructed from a hardwood or a similar material, as to be cost efficient. This met the right balance for our financial constraints and the desire to get the best product we can obtain.

6.2.3 Connecting Spinning Components

The Robofan needs to perform a panning motion from a fixed spot to keep track of a target. This doesn't refer to pan and scan which would require a rail system for the Robofan to move on. In this situation this refers to motion that is analogous to a person turning their head left and right. This panning motion brung about the question of how wires will connect on an object that twists and turns. There are three solutions to this problem that are proposed in this section. One solution is enforcing 390-degree rotation limit. The second proposal is using an electric slip ring. The final option is using a wireless charging module.

6.2.3.1 390 Degree Limit

The first solution is to power rotating components is to simply connect them with a wire. The wires have enough slack as to be allow panning motion of slightly more than 360 degrees, such as 390 degrees. Breakaway wires can be used in the prototyping phase to make sure the motor doesn't operate so fast that the wires tear. The 390-degree limit is chosen so the motor can compensate for whenever the user passes the 360-degree threshold. If the limit was kept at 360 degrees problems would occur whenever the user stands at threshold. Any degree past the threshold would cause the Robofan to immediately spin in the opposite direction. Of course, the Robofan could be programmed to not move until the user has moved a certain amount. This would lessen the accuracy of the Robofan, unfortunately. To provide a better explanation of the situation a diagram is provided. The following figure depicts a user standing at the 360-degree threshold.

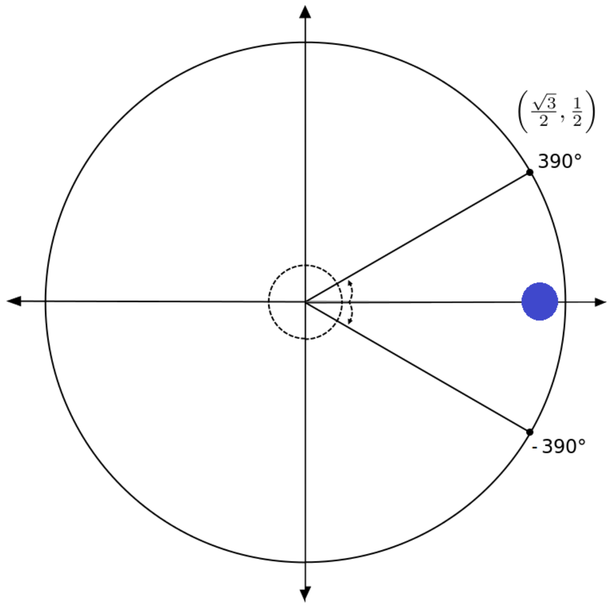


Figure 6-5: Diagram of the 390-degree Threshold

In the diagram above the user is represented by the blue circle and the Robofan is located at the origin. Imagine the user starts at 0 degrees and has traveled counterclockwise 360 degrees. The user is now back to where he or she has started. If the user continues in a counterclockwise direction Robofan would continue to follow them until they reach the 390-degree threshold. After that, the fan would need to reorient itself by rotating in the clockwise direction and relocate the user. This is done to prevent the internal wiring from twisting continuously in one direction.

Using the 390-degree limit is probably the simplest and most inexpensive option available. However, there are several drawbacks with this system. The diagram shows that a 390-degree threshold provides decent leeway. This only applies if the user is standing further away from the Robofan. Diagram shows that the closer the user stands to the origin the more difficult it is for the Robofan to track the user as well as adhere to the 390-degree limitation. Secondly, this adds complexity to the programing of the Robofan. In this setup, the Robofan always needs to be aware of its positioning. Thirdly, this system is inelegant. There is the issue of the panning motor being overworked since it must make close to a full rotation whenever the user passes the threshold. The speed in which motor reorients itself also has to be considered. What if the user moves past another threshold while the fan is reorienting itself? If the user remains conscious of the 390-degree threshold he or she can attempt to move in a manner that would help the Robofan avoid having to awkwardly rotate in the opposite direction. This is not ideal and the system is not very user friendly if they need to make a conscious effort to assist the Robofan in rotating in an efficient manner. In short, if at all possible this option should be avoided.

6.2.3.2 Electric Slip Ring

The second option is to utilize an electric slip ring to connect stationary wires to rotating ones. The following figure demonstrates how a slip ring works. Basically, it allows stationary wires to connect to rotating ones. This allows components to be powered while on a rotating platform.

This shows why a rotational limit must be implemented if this wiring configuration is used. The breaking of wires can be prevented if the Robofan unwraps the wires by rotating in the opposite direction. This is the configuration for the electric slip ring option. The wires to the right of the diagram are stationary. At the ends of these wires are contacts, such as carbon brushes, that are touching traces that cover the outside of a rotating disk. The wires located in middle, on top of the slip ring are connected to the rotating disk and spins along with it. In this wiring configuration the electronics on top of the platform is able to receive power while the platform continuously rotates.

This solution is far more elegant than the first option. With this system the Robofan can continuously rotate in any direction for as much it needs with no possibility of wires disconnecting. The programming of the Robofan's behavior is greatly simplified in this situation. The device no longer needs to keep track of its position. Since it can continuously rotate in any direction, the Robofan only needs to keep track of whether the user has moved left or right and proceed to turn in that direction. This solution is also much easier on the panning motor since it no longer has to reorient itself. It follows that power usage is reduced since motor usage is reduced by this feature. The electric slip ring does have drawbacks however. One negative is that the life expectancy of the slip ring must be considered. Overtime, continuous rotation eventually degrades and wears out the brush contacts. This in turn degrade the power transfer received by the electronics on the rotating end of the slip ring. Another problem to consider is that the sliding contacts produce some signal noise on the power obtained by the rotation parts. It is possible that the device is able to work even with the presence of noise depending on how good the signal-to-noise ratio is. If not, a circuit must be designed to clean up the power signal provided by the slip ring. A third drawback is that slip rings can be relatively expensive. Depending on the number of wires, the voltage rating, and current rating the price of a standard electric slip ring can be in the range of $15-$30. For a business class slip ring, with much higher life-expectancy the price can go higher than $100.

In summary, the use of an electric slip ring is preferred over wiring without one. There are some noteworthy drawbacks but the positives outweigh the negatives in this case. This is likely option for the powering the project's rotating components.

6.2.3.3 Inductive Charging Set

The final option is the use of an inductive charging module. As the name implies the module is a kit for creating projects such as phone chargers. Applications are not limited to chargers. The module can easily be adapted to supply power to a rotating object. The closer the coils are to each other the more efficient it is in transferring power. This option works similarly to the electrical slip ring. Just like with the slip ring using the inductive coils allows the Robofan to continuously rotate without the worry of wires getting wrapped up and breaking.

The inductive coils have several key drawbacks. The efficiency of power transfer is very low. An air-core transformer, which is what figure 6-7 is, has only about 40% efficiency. Furthermore, inductive coil sets available for purchase all have low voltage outputs. 5V is generally what the inductive chargers provide. 5V is not enough to power all the components necessary for Robofan's operation. The current output tends to be low as well. Many sets have current ratings lower than 1A. Again, this is output is not enough to power the Robofan.

6.2.3.4 Selection for Powering Spinning Components

The only viable option for sending power to electronics that are continuously rotating is using an electric slip ring. Connecting wires directly and using a rotation threshold is far too clumsy. As for the inductive coils, it is a nice option if only it could provide enough power to the project. The final selection of the slip ring that has been chosen with the project is one provided by SparkFun. The cost of slip ring comes to $29.95, this is without factoring the price of shipping. It is a 3-wire slip ring and is depicted in the figure below.

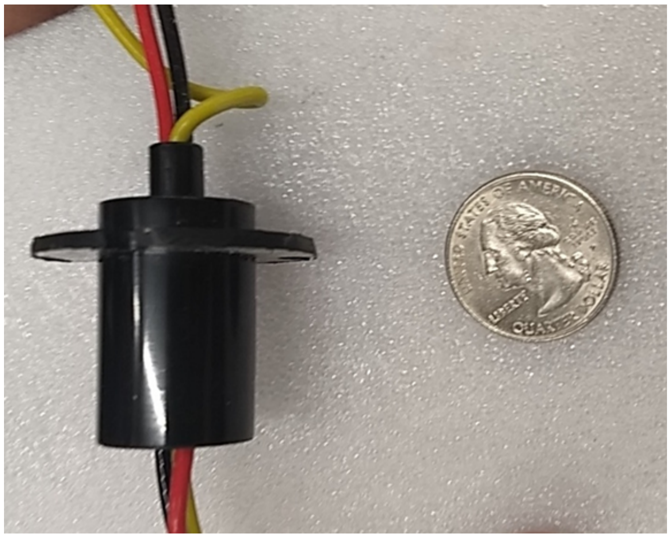


Figure 6-6: Three-Wire Slip Ring

Figure 6-8 is a picture of the slip ring that has been purchased for use in this project. The configuration of the Robofan only needs a few wires to connect to the rotating circuit board. Any unneeded wires have been tucked away and secured safely, perhaps even cut when design is finalized. The slip ring features the following specifications:

- 3 wires
- Voltage: 380 VDC/AC
- Current Rating: 10A
- Max Current Rating: 25A
- Operating Speed: 250RPM
- Compatible with data bus protocols
- Transfers analog and digital signals
- 22mm Diameter x 33mm Length w/ 44mm Diameter Flange

The voltage and current ratings for this slip ring is higher than what is necessary for this project. Unfortunately, more inexpensive slip rings don't have the current rating needed. In the search for a suitable slip ring there was none that was somewhere in the middle of the two extremes of providing much more or not enough.

6.2.4 Panning Motor

The Robofan requires a motor in order perform panning motions. Three types of motors are compared in relation to how well they can perform continuous rotation. The types to be examined are DC motors, Servo Motors, and Stepper Motors. A more detailed comparison can be found in the Titling Motor section but the comparison is in regards with tilting motions.

There are several requirements the panning motor must meet, in order for it to be suitable for the Robofan. Listed below are the requirements that must be met:

- High torque is not necessary, since the only resistance is at the rotating mechanism.
- High rotational speed (If the user is moving at 3 m/s at a distance of 1m, the Robofan needs to reach speeds of 180 degrees per second.)
- The motor must be accurate and controllable (The Robofan is aiming for a <5-degree error.)
- Motor must be responsive and have quick reaction times.
- Low power usage if possible.
- Low cost if possible.
- Produces minimal noise.
- Produces minimal vibration.

6.2.4.1 DC Motor:

The DC Motor fulfills many of the listed requirements. DC motors can provide more than enough torque for the situation. This type of motor is also able to produce high rotational speeds. A quick product search shows that a cheap $1.95 can provide a loaded speed of 4500 ±1500 rpm, which much more than is necessary. This also demonstrates the how inexpensive DC motors are. This type of motor is the most efficient of the ones being compared and uses the least amount of power. It is also the type that would produce the least amount of noise and vibration. When using a DC motor, it is harder to achieve high levels of accuracy. This is because only power signals can be sent to the motor. In this situation motors position can be predicated over time but it cannot be known exactly where it is. To overcome this the use of control theory is needed.

6.2.4.2 Stepper Motor:

The stepper motor meets all of the listed requirements for the panning motor, if not exactly as well the DC motor fulfilled some of them. For example, the DC motor produces less noise and vibration then the stepper motor. However, the noise and vibration level of the stepper motor is acceptable for this project. The design of stepper motors causes its torque to reduce at higher speeds. While this is a negative, it is one that can be forgiven in the case of a panning motor since it does not meet much resistance. Stepper Motors can be highly accurate because more than two wires can be sent to the motor, which means position signals can be sent. These types of motors are also relatively inexpensive with suitable stepper motors being sold at around $15-$20. The stepper motor would need a separate driver for controlling it.

6.2.4.3 Servo Motor

This type of motor is highly accurate and easy to use. They can be controlled with just a microcontroller and no other special circuitry. Servo motors have high torque even when they aren't powered, though this is not important in relation to the panning motion. Suitable servos in terms of torque and speed are in the range of $15-$20 much like the stepper motor. Unfortunately, this is a moot point since servos at that price range do not feature continuous rotation, a key component for the panning motion of the Robofan. Servos tend to have a limited range of motion. There are continuous rotation servos available but prices go up to $50 when considering suitable speeds. Servo motors produce the most noise and vibration of the three types of motors. Good code is needed in order to dampen movement whenever the servo comes to a stop. While the servo motor is not suitable for the panning motion of the Robofan, this type of motor may be of better for a different movement function. This is explained in more detail on the tilting motor section of this report.

6.2.4.4 Panning Motor Selection

The stepper motor is the only type of motor that meets all of the necessary requirements for a motor dedicated to a panning motion at an acceptable level. The next step is to compare different types of stepper motors and choose the most suitable one for this project. The following table is a comparison of stepper motors.

Table 6-1: Stepper Motor Comparison

Motor Type	Price	Size	Power Related Info	Torque	Additional Info
High torque 42 Stepper Motor 2 PHASE 4-lead Nema17 motor 42BYGH34	10.86 w/shipping	1.33 in	1.33 A 2.1 ohm	43 oz.-in	10 oz. Low Noise
Nema 17 Stepper Motor	14.99	3.7 x 2.5 x 2.3 in	2.1 A 1.5 ohm	92 oz.-in	14 oz. 1.8 deg step angle Step accuracy 5%
Nema 23 Stepper Motor	24.50	5 x 5 x 2.5	2.8 A 0.9 ohm	269 oz.-in	39 oz. 1.8-degree step angle Step accuracy 5%

The second motor listed is the one the team has decided to use. The NEMA 23 Stepper has the best of features and specifications of the motors being compared. The amount of torque is not needed for the panning motor, however. It is better to use the money elsewhere. Choosing a weaker motor is more cost effective and reduce the power usage of the Robofan. The first listed motor is the least expensive of the three. However, it features less than half the torque of the second motor. The accuracy and step size were also not listed making it less reliable choice.

The NEMA 17 Motor is the final selection for the panning motor. It is the most cost-effective motor. The specifications are all suitable for the needs of the Robofan's panning motion. The step of angle of 1.8 degrees allows for highly accurate and controllable movement. Its small size is very suitable for the Robofan panning motor. This NEMA 17 Stepper Motor features the following electrical specifications:

- Manufacturer Part Number: 17HS19-2004S1
- Motor Type: Bipolar Stepper
- Step Angle: 1.8 deg.
- Holding Torque: 59Ncm(83.6oz.in)
- Rated Current/phase: 2.0A
- Phase Resistance: 1.4ohms
- Inductance: 3.0mH+/-20%(1KHz)

The physical specifications are as follows:

- Frame Size: 42 x 42mm

- Body Length: 48mm
- Shaft Diameter: 5mm
- Shaft Length: 24mm
- D-cut Length: 15mm
- Number of Leads: 4
- Lead Length: 1m with connector
- Weight: 390g

6.2.5 Tilting Mechanism

For the purposes of the Robofan, a servo motor is used to control the joint within the tilting arm of the device. A servo motor was chosen for this function over a stepper motor due to the fact that servos are less complex than stepper motors in general. A stepper motor, in a general sense, has more torque than what is required than a stepper motor provides. Choosing a servo motor also conserves energy when compared to a stepper motor which would be better suited for the panning motion of the Robofan. Stepper motors would require more space than a servo due to the necessary gear box attached. Another advantage of utilizing a servo motor is that you can run it on DC power from the PCB with the use of a controller. The servo motor chosen for the Robofan must meet constraints realized by the design plan for the remaining portions of the fan.

6.2.5.1 Servo Motor Selection

As discussed in the tilting arm section, the servo motor was chosen because of its accuracy and compact size. Servo motors don't come in one uniform option, the varieties of servo motors vary in functionality and design. Universal commonalities of a servo motor are DC power consumption, feedback is used for high levels of accurate positioning, and to control any servo motor within this project a controller would need to be implemented on the MCU to control the position of the motor. As mentioned in the tilting mechanism constraints a servo motor was chosen as the mechanism. The servo motor chosen must try to optimize the following parameters.

- Lowest price possible, without sacrificing the other parameters.
- Smallest dimensions, available within the range of torque the motor needs
- Ease of use, the operation of each type of servo is what creates the distinction of that particular servo. A certain operation style may fit the Robofan more appropriately than others.
- Avoids obvious problems, if a type of motor is commonly known for having problems such as needing to be heavily ventilated to prevent overheating, vibrates excessively etc.

To select the appropriate type of servo motor three common varieties are considered. The motor type that fits the optimization criterion the best is the primary candidate for integration into the Robofan. The three choices, positional, rotation, and linear, are explored further in the following sections.

6.2.5.2 Option 1: Positional Rotation Servo

A positional rotation servo is the most common type of servo motor used and available for purchase. The output shaft of the positional rotation servo rotates, generally, in a half-circle range, or 180 degrees. To limit the range to the specified degree of rotation, the output shaft physically stops placed within the gear mechanism. This limit is imposed to the protect the sensor from physical damage. A controller is required to set the position of the servo motor. The controller would most likely be attached to the MCU These common servos are found in radio-controlled cars and water- and aircraft, toys, and similar applications to robotics and appliances. The applications used for a positional servo motor are similar to that of the Robofan which is a robotic appliance.

6.2.5.3 Option 2: Continuous Rotation Servo

A continuous rotation servo motor is a servo motor that utilizes open loop speed control. The operation of the continuous rotation servo differs from the closed loop positional system most servos use to determine the position of the motor. This is quite similar to the common positional rotation servo motor, except it can turn in either direction indefinitely. The control signal, rather than setting the static position of the servo, is interpreted as the direction and speed of rotation. The range of possible commands causes the servo to rotate clockwise or counterclockwise as desired, at varying speed, depending on the command signal. This functionality is comparable to the positional rotation servo motor.

6.2.5.4 Option 3: Linear Servo

A linear servo motor operates in a nearly identical fashion to the positional rotation servo in section 6.2.3.2 the main deviation in operation is in the rotation. A linear servo motor utilizes more gears than the previous two servo options. With additional gears (usually a rack and pinion mechanism) to change the output from circular too back-and-forth, hence giving the servo motor a linear control as opposed too rotational. The extra gears lead too larger dimensions for the servo overall. The dimensions for linear servo motors rarely exceed 3 inches in width, which is within an acceptable range, but the smallest dimensions the servo motor can be while maintaining enough torque is the optimal choice. Availability is a hindrance when using linear servo in a project. Linear servos can occasionally be found in hobby shops or salvages from older appliances and devices. Positional rotation and continuous rotation servos are much more common and have a greater selection when looking for variety in torque, price, and ease of use.

6.2.5.5 Servo Motor Decision

After evaluating the options of the servo motor types, a positional servo motor has been used for the tilting arm of the Robofan. This type of motor was chosen primarily based on its operation compared to the other two servo motor options. The accuracy of each variety explored had comparable accuracy, within degrees of each other. Since positional rotation servos are the most common type of servo motors the selection was the largest. This alone gave it the greatest advantage of having a specific motor that fit the criterion of the ideal servo motor the closest to peak optimization. The specific servo motor purchased is a Lewansoul LD-27MG full metal standard gear digital servo. The positional rotation servo motor chosen is pictured in Figure 6-9.

The motor in Figure 6-9 is a DS3218 full metal standard gear digital servo. This motor is a type of positional rotation servo that comes with a servo horn kit to allow the motor to attach directly to the joint of the tilting arm. The figure above shows the inner working of the servo motor; near the bottom of the motor a potentiometer can be seen as part of the assembly. The potentiometer is the mechanism providing the accuracy for this servo motor. The outer housing of the servo motor is fabricated out of aluminum to allow for fast and effective heat dissipation to protect the motor from thermal damage. Researching customer reviews for this particular servo motor showed consumers satisfied with the results for their various projects including robotics. This particular servo motor comes with the following specifications:

- Dimensions: 1.57 x 0.78 x 1.59 inches or 20 x 20 x 40.5 mm.
- Weight: 2.32 oz. or 65 g.
- Control Method: Pulse Width Modulator;
- Pulse Width: 500~2500, period of 20 ms.
- Rotation Range: 270 degrees
- Large Torque: 277.6 oz.*in or 20 kg*cm
- Rated Power ~ 5-6.4W DC; Rated Voltage 6-7.4 V and current of 1A

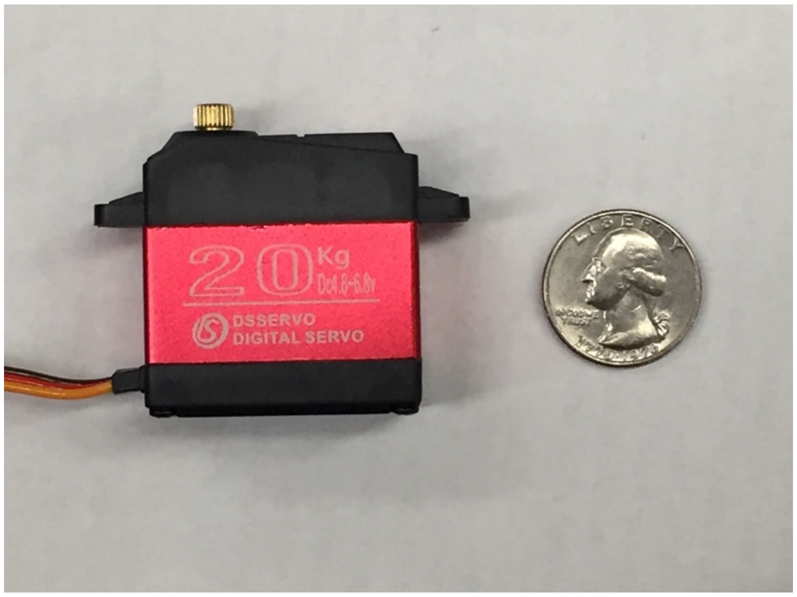


Figure 6-7 DS3218 Servo Motor

The specifications meet all of the needs that our servo motor must provide to act as the tilting mechanism. With 180 degrees here are more than enough degrees of rotational freedom to provide the tilting function of the Robofan. The torque in the motor is far beyond the requirements needed to lift the fan mechanism of the Robofan plus additional load if more features are added in the design process. This excessive torque is not a negative aspect as the servo motor itself does not pull a high enough voltage of current to constitute any marginal range of wasted power consumption.

6.2.6 Oscillating Fan

The base for our project is an oscillating fan. We need something that is lightweight, compact, can produce a good amount of airflow, and doesn't have open blades that could be dangerous. We knew that a traditional ceiling fan wouldn't work as they are too big and would be dangerous when moving around. We considered a Dyson bladeless fan as it would make for a very sleek design, but they are quite expensive and we have a limited budget. Normal oscillating fans are quite cheap and lightweight. Oscillating fans also come with a pre-built cage, motor housing, and AC motor that hooks straight into 120V wall power. If we had one or more Aerospace or Mechanical engineer in our group we would design the fan blade and assembly ourselves, but as we do not we decided to just buy an oscillating fan. This also saves us the trouble of purchasing a motor to run the fan blade, as we would probably spend more on that than the entire oscillating fan costs.

6.2.6.1 Fan Model

We shopped around for an oscillating fan that met our needs. We had decided that the smallest we could use was about 8 inches across, and largest we would be ok with was about 16 inches. Past this the fan assembly would start to get too heavy. We knew that we wanted a fan with as little extra pieces as possible, as we didn't want to pay for things we weren't going to use. This eliminated any tall stand oscillating fans. We had decided we didn't want a battery operated fan, so we had to make sure the one we got had a plug. Finally, we had to make sure that the fan we were getting fit within our budget, and the less we spent on the fan the more would be available for other, more technical components.

The specific oscillating fan that we ended up with is a Lakewood 12-Inch Oscillating Table Fan. This fan only cost us $19.88 as it was on sale on Amazon. It is a very manageable size and weight at 14 inches across and about 4 lbs. for the parts that we need. This model comes with a three speed button selection and oscillating functionality. The motor is rated for 120V 60Hz AC current at 35 watts, which should be plenty of power without the fan spinning so fast that it makes a lot of noise. The motor also has a large capacitor attached to it which seems to be for power factor correction to preserve efficiency. We briefly considered replacing the motor with a nicer one, but this one is really at the perfect speed and the fan is built around it.

6.2.6.2 Fan and Motor

The fan we got comes apart fairly easily with mainly screws, and we can isolate the parts that we need. The switch assembly comes out of the base fairly easily, and the wall plug goes straight into it, as shown below in Figure 6-10. The wires from the switches go straight into the fan motor. This switch plate has been replaced with a series of three relays in our final design so that we can change the fan power digitally without the use of a more complicated mechanism, and the four wires already go straight into the motor. We are also planning to replace the wall plug with a longer one that also has a switch on it that mounts to the wall above the outlet. This is so the user can mount the fan wherever they want on their ceiling and not have to use an extension cord. The wall switch is so that the user can turn the fan on and off without the app or a switch on the side of the moving fan.

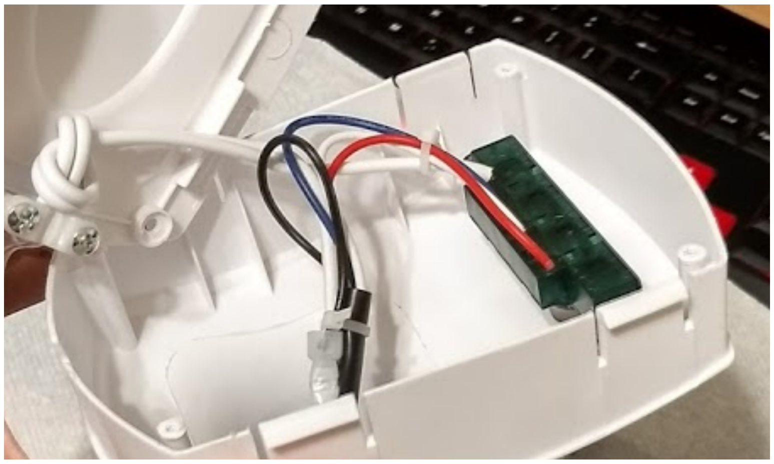


Figure 6-8: Oscillating Fan Switch Assembly

The oscillating part is driven by a motor post that comes off of the back of the fan motor and turns a gearbox. Once taken apart we were able to take this gearbox out, and we should be able to cut off the part of the post that comes out the back of the motor. This should give us enough space in the housing to fit some wires and a few of our boards, but most likely we would have to have a larger one fabricated so we would have enough room. We need to consider how closely the components are packed into there, but as this is a fan as long as we have vents on the backside and enough room for air to travel the fan blade pulls air through the housing and keep our components cooled off.

6.3 Firmware

In the following sections we discuss in depth the firmware of the Robofan. The firmware is the program which is hard coded on the MCU for basic functionality. In the following sections we discuss the different aspects of the firmware and the specific methods in which we implement it as far as control. The main three aspects of the autonomous fan tracking are panning motion, tilting motion, and tracking. The firmware ties in between the specific hardware pieces and the MCU to reach our desired specifications. Besides the hardware, the firmware also guides the Bluetooth communication between the Robofan and the smart phone application. The firmware is separate from the software which concerns the smartphone application. In general, the firmware mainly is used to communicate between the hardware to operate in conjunction with each other in order to function as a whole.

6.3.1 Modes

The firmware also has several different modes that determine what the fan does. The first and primary mode is the normal tracking mode. In this mode, the code's objective is to keep the user in the center of the frame, thereby keeping the column of air produced by the fan centered on the user. The algorithm that facilitates this is described in detail in Section 6.3.4, but essentially it uses the coordinates from the pixy camera to adjust the motors. As long as the user is still in view and the Bluetooth command is set to tracking mode, the device keeps tracking the user. If the fan cannot find the user, then the mode switchs to searching mode. In this mode the fan rotates in an attempt to locate the user, and if the user is located, it goes back to tracking mode.

If the Bluetooth module receives a command from the phone app to change between modes, the firmware changes modes as soon as possible and do whatever that mode entails. The two modes mentioned in the previous paragraph are the default modes, and the device tries those first if there is no Bluetooth connection. The other two modes, manual and pattern, are only accessible from the application.

The pattern mode performs a specific pattern and not use any input from the camera. Example patterns would be to rotate continuously at a specific speed and height. Another pattern would be to slowly change height while continuously rotating around the room. The pattern menu is quite customizable to meet the user's needs, and includes a few basic patterns with adjustable values. The manual mode locks the fan in place and allows the user to set its position. This would be useful if the user wants airflow in a specific spot where they may or may not be standing, or if they need airflow on something they are working on. This mode doesn't involve very much moving, so it is a bit more power efficient than the tracking and pattern modes. The code has been made fairly simple for this mode, as the MCU would just be moving to the values given by the phone application through the Bluetooth connection. The flowchart for the modes can be found below in Figure 6-11. This chart shows the different modes along with the cases in which the code switches between them, as discussed in this section.

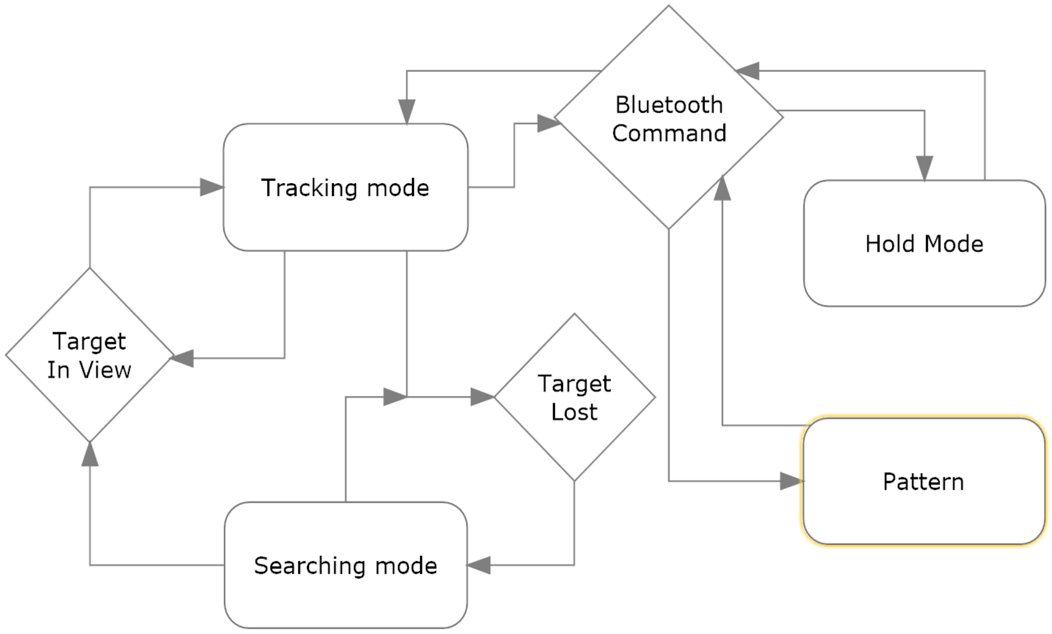


Figure 6-9: Mode Flowchart

6.3.2 Bluetooth integration

The PCB is going to have pin connections for the Bluetooth module as discussed above in section 6.1.2. The particular model we are going to use is the HC-05 Bluetooth module. This module can connect to other Bluetooth 2.0 devices as either the slave or master. Bluetooth uses this master and slave relationship to control when and where the modules can send data.[3] A module in the master mode can connect to up to seven different slaves, while the slave module can connect to only one master. The master modules can send and request data from any of its connected slaves. Slave modules, however, can only communicate with its connected master, and cannot communicate with other slaves. For the Robofan, the fan's Bluetooth module has been set to the slave mode, and the phone is set to the master mode. This retains the phone's ability to be connected to Bluetooth devices at the same time as the fan, such as wireless headphones or a phone headset. This also keeps our ability to connect to more than one fan in the future, if someone had a larger workshop and wanted more coverage. This would be possible in the reverse, but one fan would have to be the master and would have to communicate between the phone and the other fans, which would make that fan more complicated. This could also waste power as that fan's Bluetooth would always have to be running for any other fan to have Bluetooth.

The working voltage for the Bluetooth module is 3.6 to 6 volts. This is safely in line with the 5 volts that we are going to have our power converter output. The Default baud rate for our Bluetooth module is 9600. Baud rate is how many symbols it transfers per second, so it is essentially the amount of data that can get through every second. Bluetooth has several other baud rates that it can run at, but this is the default, and also happens to be towards the middle of the range. If the baud rate is too low, then the amount of data you can get through per second is constricted, but if you try to get too high of a baud rate the data might get damaged and you might use power that is unnecessary. The default pin for our Bluetooth module is 1234. The pin is used when connecting to the device, to make sure someone doesn't connect that isn't supposed to. This can be disabled so the user doesn't need a pin to use it, but we are going to keep a random pin and have it on a sticker on the side of the device. At worst, someone connects to the fan from outside your workshop and wastes some electricity, so needing to at least see the side of the fan should be enough security.

Every Bluetooth module has a unique address, made up of a 12-digit hexadecimal value. The first half of the address is a code that denotes who manufactured it, and the second half is unique to each module. This address has been presented to the user in the Bluetooth connection screen of their phone, unless the module is given a user-friendly name. Our first prototype has been named Robofan_1, and any future models that could be made have been named with the same convention and a number that signifies the order in which manufactured. We also use pairing such that when the Robofan is turned on, as long as the phone's Bluetooth is turned on the devices automatically pairs and go to the last mode that the user had on their app. If the Robofan is not connected it defaults to the search/track modes.

The Bluetooth module comes with UART (Universal Asynchronous Receiver-Transmitter) compatibility, which is an asynchronous way of message passing between two devices, either through wires, or in this case, wireless. The Bluetooth module utilizes UART to receive commands from the phone to change modes, position data for the manual mode, and pattern data for the pattern mode. The Bluetooth module also is able to pass information back to the phone about what position it is at, the distance to the user, and what user it is tracking. We can also have the Robofan pass to the phone more information for troubleshooting while debugging the Robofan. This is very useful as we can't connect the MCU to our computer while it's running, and it won't have a screen or any other way to output fault information.

6.3.2.1 Message passing from phone application

In this section we are going over the specific codes that have been used when passing messages from the phone application to the MCU. The Bluetooth protocol also handles the pairing and connection between the devices, so we do not have to worry about this. We do, however, have to worry about how we pass the commands back and forth between the two devices. We need to assign data values to each command that the fan and android application use, and the command has been coded or decoded on each side of the link. The values are 8-bit commands, as we are using UART to pass these back and forth and 8-bit is well supported. In Table 6-2 below, is listed the commands that have been used to transfer information from the application to the Robofan. We may need to add more commands later depending on what functionalities we end up adding, so this is more of a running list than a final list. This table is not necessarily sorted by code values.

Table 6-2: Message Passing Codes to Application

Command name	Description	Code
Sleep mode	Commands the Robofan to switch to sleep mode, which turns off the fan, camera, and motors	01000000
Tracking mode	Commands the Robofan to switch to tracking mode	01000001
Manual mode	Commands the Robofan to switch to manual mode	01000010
Manual Data - Tilt	The first byte of the manual mode data, this byte that describes the tilt position	00000000 - 1111111111
Manual Data - Pan	The second byte of the manual mode data, this byte that describes the horizontal movement	00000000 - 11111111
Manual Data - Fan power	The third and last byte of the manual mode data, which describes the fan power	00000000 - 00000011
Pattern mode	Commands the robofan to switch to the pattern mode	01000011
Pattern Data - Tilt	The first byte of the pattern mode data tells the robofan either what height to stay at, or what speed to tilt up and down at	00000000 - 11111111
Pattern Data - Pan	The second byte of the pattern mode data, tells the robofan which direction to move and how fast	00000000 - 11111111

Pattern Data - Fan power	The third and last byte of the pattern data, which includes the fan power that is needed and information about which directions to oscillate	00000000 - 00111111
Object list Request	The phone app asks the MCU to return the list of objects that the Pixy camera has remembered	00100000
Objects detected request	The phone app asks the MCU to return the list of objects detected	00110000
User Data Command	This Byte is a command that tells the Robofan which user to follow	00111000
User Data - Object Code	This byte contains the signature value of the object(user) that is to be tracked	00000000 - 11111111

Robofan does not take an explicit command to turn off via Bluetooth. This command is not really needed due to the wall switch that toggles power to the fan. Also a command that turns the fan off entirely could cause unintended behavior if the fan receives a faulty byte or data byte that it takes as a power off command. If the whole MCU were to turn off entirely there would be no recovering and the user would have to toggle the wall switch to resume operation. This command is replaced with a sleep command, which turns off power to everything except the Bluetooth module and the MCU. The MCU can exit this mode easily, as any mode change command takes Robofan out of sleep mode.

The data bytes that follow the command to switch to manual mode needs to convey the tilt, pan, and fan power that the user wants. The MCU is able to take an exact tilt value from the Bluetooth connection, because the MCU always know where the servo motor is based on the voltage the servo motor is receiving. The value that is transmitted over Bluetooth is somewhere between 0(highest) and 255(straight down). This one byte value is modified to a voltage and output to the servo motor. The Pan motor has been dealt with a bit differently, as the MCU does not keep track of where the unit is pointed in a horizontal direction. The Robofan can pan all the way around continuously thanks to the slip ring joint, so the MCU doesn't need to keep this information. This means that when the phone app sends a command to pan sideways, the number do not represent a specific position like with the servo. Instead, the pan command signifies relative movement in a direction. This has been encoded into one byte, and it is signed. If the first bit is 0, then it signifies a negative value, and if the first bit is 1, it signifies a positive value. With this configuration, an encoded 0 would signify a large movement in the negative direction, and 127 would be the slightest movement in the negative direction. On the positive side, 128 would be the slightest movement in the positive direction, and 255 would be the largest movement in the positive direction. The Fan motor's power can only be 0(off), 1(slowest), 2, or 3(highest). This number is encoded in the next and last byte that follows the manual mode command. This byte also has plenty of free space that can be encoded with any other information that needs to be transmitted for the manual mode.

The manual mode has three bytes of data that follow the manual mode command. This is 4 bytes total, whenever the phone app gives the command to switch to manual mode. Both the phone app and the MCU knows that the manual mode command is followed by these 3 bytes, and only these three bytes, so anything after that is a different command. This helps keep the MCU from confusing data values and mode commands. Every time the phone app wants to send the MCU an update on its previous manual mode command, it would just send the manual mode bytes again, followed by the new data byte. The manual mode settings on the app is very easy to control, and is able to be controlled in near real time. This means that every time the user changes something in the app, it is updated in the MCU, and the Robofan moves appropriately. If the user is constantly holding down a button, for example, to keep panning in one direction, the phone keeps updating and sending input to the MCU. At its fastest, this could happen at the baud rate of 9600 bits per second, which would translate to 1200 bytes per second, which further translates to 300 refreshes per second of the manual mode data. This is a theoretical maximum, but even if the real value is one tenth of this speed it feels like the Robofan is responding in real time.

The pattern mode has been coded similarly to the manual mode. The pattern mode command is followed by three bytes of data. The pan, tilt, and fan power has been coded in the same way as in the manual mode data. The difference between the pattern and manual codes is that the third data byte contains information about which functions are going to oscillate in the pattern. The third data bytes are split like so: '00XXYYPP'. The two bits marked 'XX' have been coded such that the MCU knows whether to oscillate at the tilt joint. If the 'XX' is coded as a '00', then the MCU takes the tilt value as a fixed value, and sets the servo motor to that value and leave it there. If the 'XX' is coded as '11', then the MCU takes the pan value as a speed, and oscillate at that speed. The tilt speed of course is restricted so the fan doesn't move up and down too fast and cause excess wear and excess power used. This leaves two unused combinations of 'XX', which are '01' and '10', which can be used to signify something later if needed. The two bits marked 'YY' have been coded such that the MCU knows whether to oscillate at the pan joint. When coded as '00', the MCU knows not to oscillate the pan motor, and use the pan motor data as a fixed amount to move before stopping. If the 'YY' is coded as '11', the pan motor oscillates at the speed and direction as denoted in the pan data byte. If the 'YY' is coded as '10', then the MCU knows to not pan anywhere in the pattern, and to just hold the stepper motor where it is currently. This leaves '01' unused as of now, but could be used for something in the future. The 'PP' at the end, is of course used to denote the power level wanted while oscillating. This is coded the same way as in the manual mode. Every time the user makes changes in the app to the pattern that they want the Robofan to be executing the phone app sends a whole new pattern dataset, starting with the pattern mode command followed by exactly three bytes, which are the updated pattern data.

The next few commands involve the object data from the Pixy camera. In short, the Pixy camera keeps a list of objects that it remembers. For each of these objects, it keeps a signature value, and a name for the object. We are going to add functionality to the phone application so that the user can see the names of the objects that the Pixy camera has. The phone app is also able to display which objects it can see in its current frame, and the user is able to choose which object to follow. This means we need to be able to communicate this information through the Bluetooth connection. The object list request command asks the MCU to return the list of objects' signature values along with the names that the Pixy camera has associated with each object. The objects detected request command is similar, but requests only the objects that the pixy cam is currently detecting. This command only needs the signature values of the objects, as it can reference this to the full object list to get their names. This means we do not need to send the names repeatedly which is a waste of bandwidth. The user data command tells the MCU which object to have the Pixy camera look for. This command is always followed by one data byte that is the signature value for the object that the user has chosen for Robofan to track.

6.3.2.2 Message Passing from MCU

To pass the information from the MCU to the phone application we use a method similar to the method from the Phone app to the MCU. The MCU does not have the same amount of data that it needs to send to the phone, as the phone application does not need to know where the fan is. The application is also mainly for button free control of the fan. We initially wanted the phone application to be able to display the video that the Pixy camera is taking and what it is tracking, but we ran into several complications with video streaming over Bluetooth. The Pixy camera would have no trouble outputting the video to our microcontroller, but Bluetooth doesn't have a high enough bitrate to be able to get real-time video in a reasonable resolution and framerate. We could have it save some video and transfer the files, but if we give up the real time aspect it loses much of its value. This would also open the user up to security risks, and we would have to encrypt the video and make the fan's overall security stronger.

If the MCU is not transferring video back to the phone, there are not many things that need to be sent back. Since the Phone application is sending and resending the mode and data multiple times a second, the MCU does not need to return confirmations of what it received. If the MCU receives a faulty command through Bluetooth it has very little time to move before the next packet arrives and corrects it. The phone app doesn't really need to receive what mode or data on where the motors are, because if the MCU is working properly, the phone app knows that what it is send the MCU is very close to what the MCU is doing.

The MCU needed to respond to the requests it receives from the Phone app. So far the requests we have are all related to the objects seen and known by the Pixy camera. The response to the object list request is the most complicated, as we have to transfer the name of each object matched with its signature value. To accomplish this, we use ASCII values to represent each number and letters. The transmission starts with a header, to signify that this is the start of the data stream. There is an ASCII coded space and then the ASCII characters for the signature value, then no space and the characters that make up the name that the pixy camera has stored for the object. Once the name is finished, there is a space then the signature value for the next object, and it continues until the last object's name has been transmitted. Then, instead of a space, we signify that the end of the text has been reached by transmitting the ASCII character for "end of transmission"(00000100). The detected request response works similarly to the list request response. The transfer is started with a header code specific to the detected request. This is followed by the ASCII space and the first signature value of the object that the Pixy cam detects at the moment. This is followed by a space and the next object, until the last signature value is transmitted. Then, an end of transmission ASCII character is transmitted. If the Pixy camera does not currently detect any objects, it transmits end of transmission directly after the header code. Below, in Table 6-3, the significant codes are listed. This list was added as we went through the building process and added functionalities to the application.

Table 6-3: Message Passing Codes to Robofan

Command name	Description	Code
List request response header	Signifies that the transmission that is starting is the list request response	10100000
List request - signature value	1-2 numerical ASCII characters that represents the signature value of an object that the Pixy camera knows (0-9)	00110000 - 00111000
List request - Ascii bytes	An unknown string of ASCII characters that represents the name associated with the signature value that it follows (A-z)	01000001 - 01111010
Detected request response header	This signifies the start of the transmission of the	10110000

6.3.3 Digital Motor Control

There are two motors that must be controlled digitally. They are the Servo motor, and the Stepper motor. They are controlled in accordance with the tracking algorithm that determines how they need to move. Firmware needs to be written in short simple methods for controlling the motors. Ideally, these motor motion functions have been written such that there are functions for each individual direction a motor can go in. As the stepper motor moves in literal steps, the associated functions send signals that move it a certain number of steps in a given direction. The servo motor is controlled by a voltage sent through the wire to modify the angle it is currently at. The associated functions were written such that there are several amounts of motion to move it based on the tracking function. It needed to be determined exactly how much motion is required at any given time.

These small functions for individual amounts of motion produce strong separation of code. The tracking algorithm needed to choose which methods to use for movement of the motors. This keeps the code very readable, as the functions have been named appropriately for ease of usage. Without this separation, the tracking functions would be forced to send specific motor controls several different times, which would produce rather unreadable code. Additionally, as each motor is used several times, their functions only need to be programmed once. This is very good practice.

The motors should only move when directed by either the searching or tracking algorithms. We are using a very simple processor that does not support multithreading. This prevents the usage of motors that run until the camera says not to, forcing to use functions that move them in amounts and check if that amount was correct, due to clock cycles.

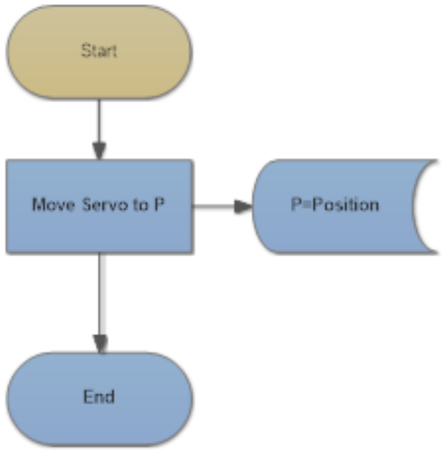
6.3.3.1 Desired Motion

The motion of the motors can be controlled by either the tracking firmware, or manual override from the phone at any given point in time. The tracking firmware must be written with special tolerances such that the camera does not over react to any movement of its target. The tolerance for movement of the target should be based on the size of what is currently being tracked. A larger target would need to move more to produce a movement of the fan. This is additionally important because visual tracking software is imperfect, and the target occasionally "shudders". This means that it appears to move slightly as the light being tracked could change slightly due to shadows or similar false movements. Having a higher tolerance for what is actually a movement of the target produces fewer false positives of movement and keep the fan more stable. It is important that the fan does not over track an object, because any movement of the fan is potentially noisy and distracting due to the motors that produce movement. In short, the desired motion caused by the firmware is largely controlled by thresholding to produce consistent, smooth movement without over reacting to any trivial change of object location.

Once the target is confirmed as moving by the tracking algorithm and tolerances, it must follow the target. The device should have variable speeds it attempts to move at, based on the targets distance from the center of the view as well as its perceived velocity. If the target is both near the center of the view, and moving slowly the camera should move slowly to follow it. If the camera is moving too fast, it could potentially overshoot the target. This would require the camera to stop its movement, and go the other direction. This accidental movement would both lower the overall precision of the device, and again be noisy. The perceived velocity of the target would be tracked during motion as the average of its movement across a number of frames. It is important to track this motion as the total distance of coordinates it has moved in a direction versus frames, rather than its distance from a previous location in a current frame. This is because the camera attempts to move to follow the target, thusly changing the frame of view and giving the target a different average velocity. This variable speed allows the camera to come to smooth stops, as well as starts. Smoother movement puts fewer stresses on the system, as well as look more aesthetically pleasing.

6.3.3.2 Coding Plan

The directional functions for motor control should be written as simply as possible. The servo motor is controlled by telling it what position it should be placed at. The position should be chosen based on several components of the tracking algorithms data. Because the overall goal is to produce smooth movement, the motors should primarily choose how to move based on the speed of the target being tracked. The servo new position should be calculated based on the current position, and the ratio of how long it will take to get to the targets new location versus how fast the motor can move. The simple implementation of the servo motor controller is shown in Figure 6-12. Because the camera is updating at a certain frame rate and the processor can cycle at a certain speed, the motors should not try to move the entire distance at once. Rather, they should attempt to move a certain amount of the total distance every so often. If the motors previous command was within a certain threshold of what a new command would be, such that any difference in movement is minor, the new command should be ignored. This will further add to the perceived smoothness of motion. The Stepper motor is controlled in steps in a given direction. It is possible to produce methods for either direction that each do attempt to do a certain number of steps. There are two possible implementations for the stepper control. First, either the tracking algorithm will guess at how many steps are the correct ratio for a given movement. The second is to have several methods for variations of fast, slow, or a medium amount of movement. The first setup will likely be a better control setup. This more ratio based control will likely be more complicated to produce, but yield better movement.



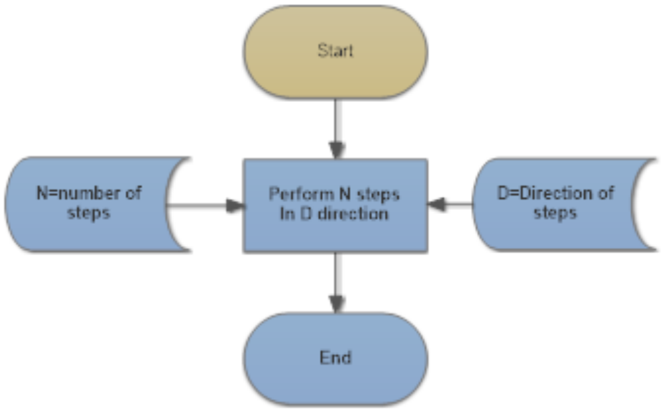


Figure 6-10: Stepper Motor Flowchart

6.3.4 Tracking

The fan tracks a target by using the Pixycams XY output. An algorithm needs to be written that sends signals to the motors with the goal in mind of keeping the target centered on the camera. For more precision, the targets distance from center should affect how fast the fan spins to follow. This prevents harsh stoppages if the fan only follows at a single speed and the target starts and stops rapidly. Changes in X can be handled by the stepper motor, changes in Y is handled by the Servo. These steps are shown in Table 6-13.

While in tracking mode, there are several steps the device must take to ensure the target stays centered. First, the camera must determine if the object is centered relative to the current view. This is done by checking if the object is at our relative x coordinate is equal to zero, as well as the y coordinate. If they are centered, then the device simply waits for them to no longer be centered. Once the target moves, this check fail causes it to enter the movement portion of tracking. These checks are ordered as follows, but can be implemented in any order or combination of checks. For instance, the tracking flowchart shows only one directional checking. This causes the camera to only move on one axis at a time for tracking. This means it takes longer to follow a target if they are moving in multiple relative axis to the camera. This slowed tracking can result in an easier lost target. The tracking algorithm should be improved as much as possible to minimize target losses. For smoother movement, these checks can be run in parallel as combinations of directions so that the camera can move diagonally. Diagonal movement speeds up the relative motion of the camera to match a target as it moves in real time, thus producing a more constant centering.

With regards to the motors being used to track, it is important to remember their limitations. The servo is very precise, but due to the housing is not able to rotate from a front facing to straight down to negative facing. Instead, if the target moves directly under the fan to behind it the fan must spin horizontally to track them. If it loses the target due to this weird movement it must re-enter searching mode. The stepper motor paired with slip ring joint allows continuous 360-degree rotation along the X axis. This imposes no limitations on tracking implementation. The tracking algorithm does need to prevent accidentally overextending in either angle of elevation or depression. These angles are hard limited by the servos maximum angles, as well as if the casing has any joint limitations. Currently, the camera checks if it needs to move right, and if so it does. After this, it checks for centering and repeats motion as necessary. This process is repeated in order for left, up, and down. If it ever discovers it is centered, it reverts to the state where it waits for the target to move so that it can begin tracking again.

There is an important addition to the tracking algorithm that still needs to be made due to a limitation of our servo motor and fan housing. Currently, the servo motor cannot angle the fan straight downwards. This results in a case where a user can move under the fan and fail to be tracked as the target would be lost due to exiting the field of view as the camera gets stuck. This case needs to be very specifically addressed. First, tracking algorithm should keep track of where the motors are currently pointing the camera in relation to a certain origin point. The motion of the camera should be tracked, and any movement that attempts to move the camera into the dead zone should be prevented. Instead, the camera should assume the target is going to move to the opposite side of the camera. Rather than revert to searching mode, the camera should rotate to the opposite point of where the target was lost crossing the threshold. This motion should be done as fast as safely possible. Once the camera arrives at the assumed location, it should wait temporarily for the target to arrive. If the target does not appear in view in a certain delay period, it should exit tracking mode, and re-enter searching mode. This failure to find can result from either the target changing direction, or remaining directly under the fan. In either case, they are eventually picked up by the searching mode assuming they leave the underside of the fan.

6.3.5 Firmware Coding Plan

In this section we discuss how each part of the firmware fits together and communicates, along with the different functions and what variables they keep. The firmware has been coded in standard C, as we all have some level of experience in it, it is easy to use, and has very little overhead to worry about. Our code is not very complex either, and we don't need to use objects. Some other languages we considered for the firmware code are assembly, C++, and Java. We decided not to use assembly because it's hard to read and program in. We decided we liked C++ over Java if we were to use an object-oriented language, but even then we don't really need an object-oriented language or to make the code more complex than it really needs to be. C is also easier to get running on a low memory and processing power board, but still has all of the readability and usability that assembly is lacking.

Now that we've decided we're going to be coding in C, we can start building functions. The main functions are where the code starts and ends, as this is how C works by default. This is where we have all of the global variables declared and call the necessary functions from. The main function is where our program repeats forever. The Robofan is only turned off by the wall switch that toggles the power for the entire fan, so our code is never supposed to turn off completely. Instead, the PCB keeps turning off functionality ending in a lowest power wait-and-listen sleep mode. In this mode the fan intermittently turns on the camera and Bluetooth to look for connections, and wait in-between. Since our program is never supposed to end, we can use a 'while(1)' loop in our main function. Most of the work is done in the refresh function. This is a function in the main while loop that calls several different functions in order to update everything that has changed or needs to change since the last refresh function. This is a useful programming tactic since we can't use multithreading to run multiple functions at the same time, so we need the functions that have to run share the CPU and not stick on one function for too long. If the CPU gets stuck on one function for too long there can be a visible delay in some movements of the fan, which could cause shuddering and unnatural feeling movement. Another factor working in our favor, is that we have a very easy time keeping the motors moving fluidly as we can just set the outputs to those motors to what we want them to get to and while the PCB CPU is working on refreshing other things the motors are still be moving because they each have their own boards controlling them. The stepper has an external controller that deals with this and the servo motor has an internal chip. Using UART to communicate with the Pixy camera and the Bluetooth module also assists us in reducing movement lag, as we do not have to use code interrupts whenever the UART receives a command, because UART is asynchronous. This means that the messages that are passed are stored in a buffer until the MCU wants to read them. If we were using synchronous message passing, we would have to have interrupts in the code so that the CPU can switch over and read the message as it is coming in. This would cause large delays in tracking as if too many signals come in at once the CPU would be bogged down trying to read those and wouldn't be able to do anything else for a time.

The refresh function is in charge of calling several functions that keep Robofan moving fluidly. The refresh function starts by calling a function to read the bluetooth UART. We call this new function bluetoothRead(). This function also calls a bluetoothWrite() if it reads a request from the phone that it needs to respond to. After bluetoothRead() we need to get the latest object location from the Pixy camera. To do this we call cameraRead(). This reads the UART from the Pixy camera, and if in the future we need to send information back to the Pixy camera we can use a cameraWrite() function here. Now that we have the new location of the object from the pixy camera we need to update the position/speed of the motors. This has been implemented in two separate functions. To update the servo motor we have servoUpdate() and for the stepper motor we have stepperUpdate(). We also have a function for adjusting the power of the fan, which is called fanUpdate().

The bluetoothRead() function checks and see if there are any UART messages in the buffer waiting to be read. If there is a mode change command, this function will resolve it and change the necessary variables. the bluetoothRead() function calls itself as long as there are messages in the buffer, unless it reaches a preset number of actions. Each command the MCU receives from the bluetooth module(not each data bit) counts as an action, and we will keep track of the number in a local variable named actions (int). This gets cleared at the start of each run of the bluetoothRead() function, and we adjust the maximum number this variable can reach as we need to avoid the MCU getting stuck reading inputs and not being able to adjust the motors. If bluetoothRead() takes in a request command, it calls bluetoothWrite() and passes it the request code. the bluetoothWrite() function can then take that command and respond as needed.

The cameraRead() function reads in the latest values from the Pixy camera and change the necessary variables. The first two variables changed are the global variables that store the X and Y coordinates of where the object is, which are named objX(int) and objY(int). Next it changes the global variable for the object's size, which is named objZ. This is because we are only really be using the object's size to gauge its distance from the fan. The last variable that is changed is the global variable that holds the signature value of the object that is being followed, which is called objID. The cameraWrite() function may be used in the future if we need to send information back to the Pixy camera, such as teaching it new objects from the application, or telling it which object ID to track. If we are in the pattern or manual modes the Pixy camera is turned off so there shouldn't be any communications from it, but the cameraRead() function is not called in these modes.

The servoUpdate() function updates the data value that goes directly to the servo motor. If we are in the tracking mode, this function compares the objY with the preset border values. If the objY value is outside of these border values then the servoUpdate function must change the servoP (double) variable to move the servo and adjust the objY. The servoP variable is the double value that is exported as a voltage to the data pin of the servo motor. This value represents the position that the MCU wants the servo motor to be at and the board on the servo motor makes adjustments to be at that position. If we are in the search mode then the search() function determines how far and how often the servo motor moves. If we are in the pattern or manual modes the servoUpdate() takes directions on where to position and how to move from the Bluetooth commands.

The stepperUpdate() function works in much the same way as the servoUpdate() function. If we are in the tracking mode, stepperUpdate() compares objX with the preset border values, and if objX is outside of those values the function sends commands to the stepper motor controller conveying which way the motor needs to move and how fast. If we are in the search mode then the stepperUpdate function either moves at a fixed speed in one direction, or not moving at all depending on the current delay and how long we have been searching. If we are in either the pattern or manual modes, the movement of the servo motor is dictated from the Bluetooth commands that are received.

The fanUpdate() function takes the current objY and compares it to a predefined list of ranges for each fan power level, and sets the fanP (int) based on that. To prevent the fan motor power from changing too frequently, we have a base duration between fan power changes. This delay was most likely on the order of a few seconds, any lower and the changes in fan power wouldn't have enough time to even spin the fan up to the different speed, much less have a noticeable effect. If we are in the search mode, the fan is on if it's within a predetermined amount of time, or past that time the fan motor turns off to save power. We don't want to spin the fan down immediately as if the fan loses someone for a few seconds we don't want it to have to spin all the way down and back up. If we are in the manual or pattern mode, the fan power is determined based on what the Bluetooth commands say.

There are many global variables that we use to keep track of important things that are needed by several different functions. The first of these global variables is the mode variable. This is set to 1 for tracking mode, 2 for searching mode, 3 for manual mode, and 4 for pattern mode. If we add more modes in the future we can add a new number choice for each new mode. The mode variable is used in every function, as the mode that the fan is in has an effect on every function. The remainder of the global variables are explained above in their respective sections, and table 6-4 below shows which functions need each of these variables.

Table 6-4: Global Variables Used by each Function (x means used)

	objX (int)	objY (int)	objZ (int)	objID (int)	servoP (double)	stepperP (double)	fanP (int)
bluetoothRead( )	x	x	x	x			x
bluetoothWrite()				x			
cameraRead()	x	x	x	x			
cameraWrite()				x			
servoUpdate()		x			x		
stepperUpdate( )	x					x	
fanUpdate()			x				x

6.4 Application

The application is designed for android devices. Android was chosen because it was what the developer owns, and is thusly easily testable and usable. The goal of the application is the have a mobile platform that controls the device from a nearby position via a Bluetooth connection. The application should be easily usable, with self-explanatory controls. The application is lightweight due to its small scope. This is useful as requiring less memory to install, and fewer resources to run leads to easier running on the phone. Ideally, it does not consume much battery power, leading to a prolonged activity time. The application is necessary to utilize the fan, because all functionality is controlled via signals sent from the application to the device.

6.4.1 Backend

The android app has been programmed utilizing the IDE Android Studio. This was chosen because it is the official IDE recommended by Google for Android devices. This utilizes the Java language, which must be separately installed on the development system. An additional benefit of this IDE, is that it allows emulation of an android device. This provides an easy method for testing of the application in development without the hassle of constantly uploading it to a device manually.

The application connects to the device via a Bluetooth module. This app is password protected by a code written on the device. Without the code attached to the fan, the application has no device to connect to. Because it does not have any devices to connect to, the application is not able to utilize any features of a fan. The Bluetooth module has a limited range which acts as an additional layer of security. For example, if your phone was stolen with the application available while outside of your workshop. In this scenario the phone could not connect to the fan, which in turn prevents a potential thief from viewing the camera feed and seeing into your work zone.

The system is going to have an entry screen, that takes users to the main control page. There are 3 modes that the application must support: Searching, Tracking, and Manual. The phone sets the fan to search by default. Searching and tracking automatically transfers control of the fan to each other as necessary. Manual mode can only be entered and exited by the user. The backend has been designed to consume as little memory as possible, and only send signals back and forth when absolutely necessary. The only signals that are relevant are the Bluetooth connection, any settings or timers, and the video view, when in video mode. This means that the application should request video when in video mode, and not while in other modes.

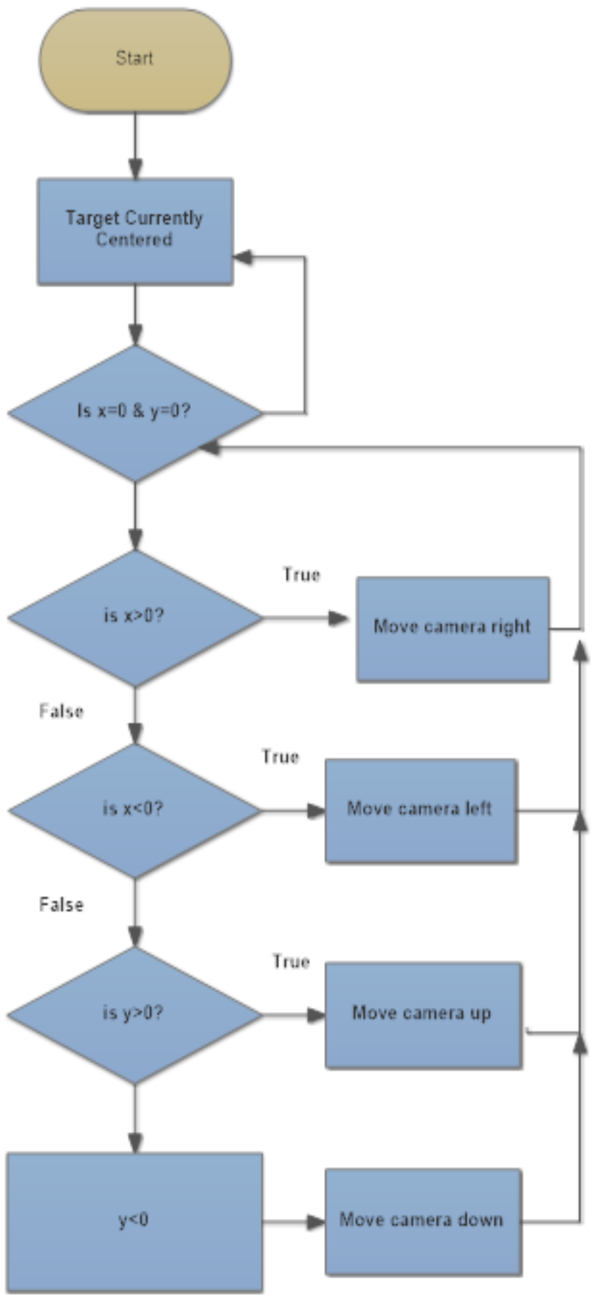


Figure 6-11: Servo Motor Flowchart

6.4.2 User Interface

The User Interface should be made as such, the goal of the user interface for the phone application is to be simple to use and easily convey information. The user interface starts with a welcome screen on launch of the app. This transitions to a menu with 4 buttons: Toggle fan active, Toggle manual control, View Camera

Feed, and connect to device. The view camera button takes the user to a new screen, that shows the camera feed while in automatic mode, and shows additional controls while in manual mode. This screen also says what mode the fan is currently running in. Pressing the connect to device button takes the users to a new screen that prompts them for a code located on their fan. Once this prompt has been filled, and the device receives confirmation that it is connected to the fan, the screen flashes an alert showing this success. No buttons on the application function while it is unconnected to a fan device, except for the connect to device button. There are also be a section dedicated to running settings. This settings screen controls delay between searching cycles, and what color pattern to look for, from a color list we added to the device.

The user interface should be light colors that do not strain the eyes to view. Ideally some primary blue color on a background of black has been used. All visual aspects of the application and UI has been designed using Android Studio. The buttons are laid out in specific orders for each menu. The main menu has the most used buttons placed higher up on the menu. Similarly, the remote view section has the camera feed placed above the buttons. This placement allows the buttons too not get in the way of the view, and the view to be easy to see. This is done to allow the user ease of use. This ease of use stems from the idea that people read left to right, top to bottom, thusly seeing the more useful options first. Given more time to research, we would gather data about how frequently buttons are used versus what we assume they are. This data could be used to place buttons in better spots.

6.4.3 Modes

Searching mode is the default mode for the device turning on. The fan is not active during searching mode by default, but can be enabled by the application. It rotates the device continuously at a slow speed until it finds a target to track. When it finds a target matching a pattern it knows, it enters tracking mode. If it does not find a target within 3 rotations, it begins to add a delay to the searching movements. It will add a 1-minute delay every time it fails to find a person, up to a maximum of 5 minutes. This is done to conserve power. The application can also override any delays added, via the settings button. The delay can be manually set in a range of zero to ten minutes. These delays are held until the next time the device changes modes.

When the device enters tracking mode, it changes behavior. First it turns the fan portion active, and second it focuses on the target it has located. While in tracking mode its primary goal is to keep the target in the center of the camera. It does this by sending signals to the fan motors based on the changing XY coordinates of the target. Additionally, the tracking mode can utilize the size of the target to estimate how far away it is. It can use this to automatically adjust the fan power output, to provide a relatively constant amount of air to a target. This feature can be overridden on the application to set a constant airflow. A basic flowchart showing this searching and tracking algorithm is shown below.

The third mode is for manual control via the phone. There is a page on the application that always shows the cameras view. While in manual mode this page also has controls for the fan. It is able to set power, and manually move the fan in its 4 directions. This manual control does not allow the user to move the fan below its maximum depth as limited by hardware. The same applies for upwards elevation. At a maximum elevation or depression, the relevant button ceases to function.

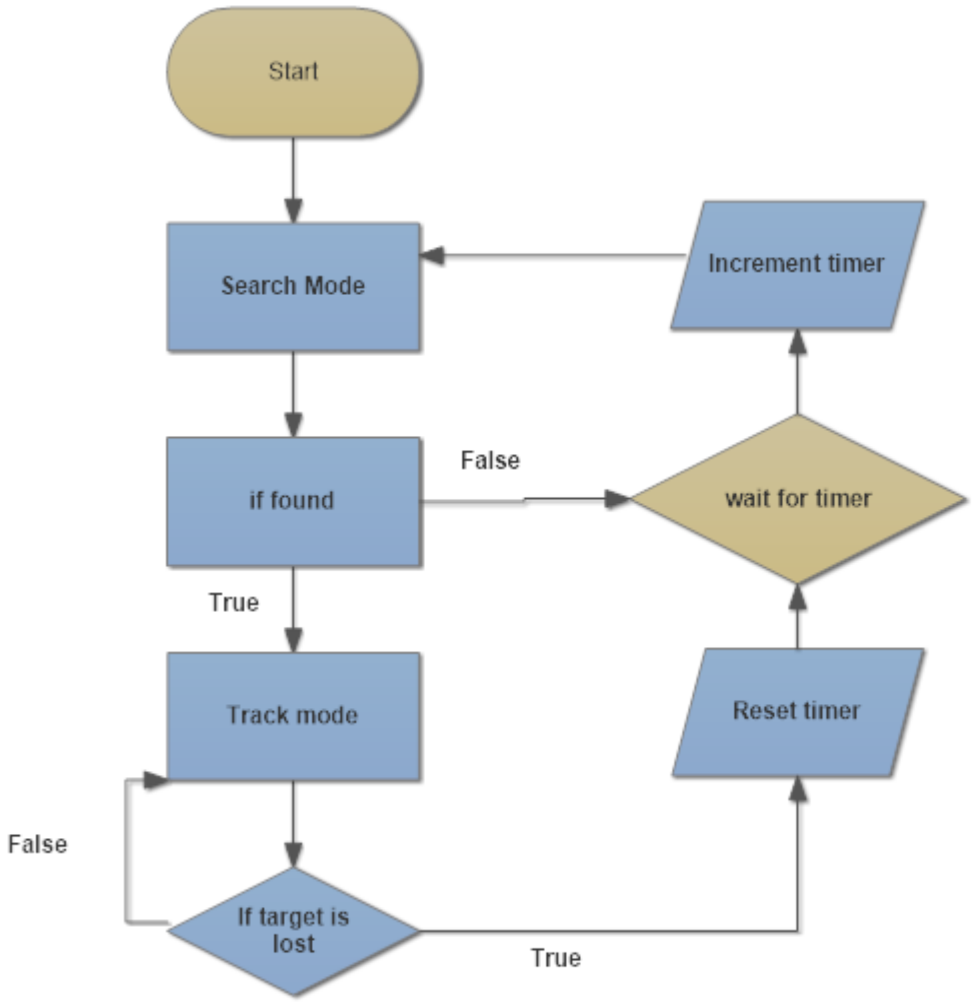


Figure 6-12: Manual Control Flowchart

6.4.3.1 Pixy Camera Mounting

The PixyCam has been mounted on the front center of the fan. It is shipped with brackets, screws, and mounting points. These have been used to attach the camera to the fan. Due to the fact that the PixyCam is largely an exposed circuit board, an additional casing needs to be fabricated to secure the camera and keep it safe. This casing has been designed using a 3D CAD software, and 3D printed. The casing should be as small as possible due to printing constraints, and have an aperture for the camera lens. The casing needs to have exit points for the wiring as well, because it needs to be routed in a specific direction across the fan cage. The casing needs to be designed to be easily removable from the camera, in the event that adjustments need to be made. The casing does not need to be very thick, as it ideally has no external stresses on it. This is largely due to the fact that it is being used as a cover against the environment and is not needed to withstand stress due to the fact that the installation is hanging from the ceiling where nothing should hit it. Because the camera is going to be mounted on the front of the fan, wire placement becomes an issue. The wire connecting the camera to the microcontroller needs to be securely placed and fastened along the fan cage. The wire should be as short as possible so that it has fewer things to catch on and potentially snag and break. The strong anchoring of the wire is also important due to the fact that it is in the direct wind flow of the fan, adding potential stress to the fastening. Other than the wire being exactly as long as necessary to reach the microcontroller, no additional focus needs to be placed on the length. Wire length is relevant to other sections of the design, such as when attached to the servo motor.

6.5 Power Supply

Alternating current is the source of the power this device receives in the same fashion most appliances. The components within the device such as the motors, Pixy cam etc. require direct current to operate. To combat this obstacle an AC/DC (alternating current/direct current) power supply has been implemented to turn the alternating current from the wall outlet into consumable direct current for the device. The base of the Robofan is the fan portion itself. The fan's motor runs on alternating current power. A constraint when designing this power supply is considering the placement and integration of the power supply with the alternating current consuming fan. Appliances such as fans vary in different fashions of power supply implementation.

AC/DC power supplies are implemented in different fashions for devices that operate on direct current. Adapters can be implemented within the power cord itself. To save on the size of the device and the interior and component spacing there can be an AC/DC adapter at the beginning of the wall outlet or, more commonly near the middle of the wall outlet cord. Some devices are able to implement their power supplies in a less obtrusive way.

6.5.1 In house PCB Power Supply

Premade circuit boards can harbor AC/DC power supplies in a less obtrusive and cosmetically less noticeable way. Power supplies often times are on a PCB (printed circuit board) within the device itself as part of a larger PCB with other functions or on a completely separate PCB. The need for a separate PCB would be needed if the device such as the Robofan had a PCB too large or was too functionally delicate to house the power supply on the main PCB. These premade power supplies can be bought or built from scratch, but the bought power supply would be a separate board from the main PCB. There are sometimes needs to choose the PCB option or the cord option depending on the need of the design of the device.

Time and financial constraints are the main aspects to consider when choosing the specifically tailored power supply design route. The transformer can be quite large when trying to meet a voltage 6V with current of 1 A or 2 A, which happens to be the largest demand of any single one DC dependent component within this project. Going from 120 V AC to 6 V DC at only 1 A would require a step down motor that would not easily fit in a space the size of the enclosure of the Robofan. To allow for a physically smaller step-down transformer we would have to implement a certain type of power supply that would allow for the implementation of a power supply on the printed circuit boards. A power supply type that would satisfy this needs would be a switched-mode power supply.

It can be described that in a switched mode power supply that nearly all of the power supply resides on the PCB. The input going into the power supply is high voltage with a low frequency coming from the wall outlet, this feeds into a rectifier such as a standard full bridge. The rectifier gives a DC voltage output that is not regulated. A capacitor is used to smooth out the signal of the input rectifier or filter, this is known as power factor correction. The smoothed DC power of the filter capacitor output, feeds into a high frequency switch, usually 20 kHz or higher. This switch turns the DC power into a perceived AC power, but instead of the low frequency of the original input it has the frequency of the switch. The switch then feeds into a physically small step-down transformer. After having the voltage stepped down to the appropriate voltage level, the new output goes through another rectifier or filter in the same manner it did in the first step in this process such as a coil or possibly another transformer. The new rectified DC voltage is smoothed out with a smaller set of capacitors. The white bars on between the capacitor and transformer as well as the ones between the switch and coil are the primary and secondary heat sinks respectively.

6.5.2 External Pre-made Power Supply

An external pre-made AC/DC power supply is the less complex option for the Robofan as far as design is concerned. The main two positive aspects of having a pre-made AC/DC power supply are convenience and price. Having a pre-made power supply would be very convenient to integrate into the project for the Robofan. To meet the power requirements for all of the components within the Robofan the transformer would be too large to logistically justify putting it on a PCB. In general, pre-made power supplies come with a housing that would provide another constraint when it comes to the size of the Robofan enclosure when designing that portion of the 3-D model of the Robofan. An external power supply and corresponding schematic are pictured in the figure below.

To understand how an external pre-made power supply as well as giving a graphic representation of the signal as it processes through the mechanism. The AC source in the case of this project would be coming from a wall outlet at 120 V with a frequency of 60 Hz. The AC source feeds into the “primary” side of the transformer which is stepped down to a lower voltage on the portion of the transformer labeled “secondary”, for the Robofan this would be stepping down from 120 V to 6V. Four diodes form a full bridge rectifier used to rectify the power transmitting from the secondary half of the transformer. The output of the node following the bridge rectifier has an output of constant positive polarity. After the signal has been rectified to provide a consistent polarity in the voltage, the capacitor smooths out the output of the full bridge rectifier to provide a DC current for the load. The load for this project would be the components that operate on DC current found in Table 6-5.

6.5.3 Robofan Supply Choice

When weighing the options for the Robofan, a salvaged pre-made external AC/DC power supply has been utilized rather than building one the main PCB itself from scratch. Finding and salvaging a premade AC/DC power supply is the most convenient and cost effective method for the Robofan project. The difference in the integration of this power supply from its original design is that it is with the housing of the Robofan. This design choice stems from the need for the fan motor to receive AC current. All of the other electrical components within the Robofan are listed in the table below.

Table 6-5: DC Components

Component	Voltage	Current	Power
Lewan Soul Standard Digital Servo	6 V – 7.4 V	1A	6 W – 7.4 W
Nema 17 Stepper Motor	2.8 V	2A	5.2 W
Hiletgo HC-05 6 Pin Wireless Bluetooth RF Transceiver Module	3.3 V – 6 V	10 mA	33 mW – 60 mW
Pixy Cam (CMUcam5)	5V	140 mA	700 mW

The table above displays the different components within the Robofan that operate on DC power. The currents and power of the components in Table 6-5 have a varying range of voltage and current demands. This gives the power supply responsibility for providing three components with DC power. The current range doesn't exceed 1 A for DC which should be manageable to provide the correct current to each device without having to implement anything complex. The most favorable aspect of the components the power supply is responsible for is the voltage levels of each component are very similar. Since the components power specs are comparable it is justified that this project utilizes the more cost effective route of salvaging a premade power supply and adapting it to fit within the Robofan. The power supply is only responsible for the DC power consuming components, not the remaining AC power components.

6.6 Voltage Regulator

The Robofan's DC components are rated at 12v and 5v. To get the needed 5v a voltage regulator is necessary. A switching regulator was used since linear regulators are not rated for much higher than 1 amp. This would not work since the Robofan needs more than 1 amp to run all its 5v DC components. The regulator that was chosen for this project uses the D24V25F5 integrated circuit. It is designed to output 5v when it is given a 12v input. The regulator is rated for 2.5a, which more than what is needed for the Robofan. Switching regulators are also more efficient than linear regulators. Depending on the load a switching regulator can be have efficiencies of 85% to 95%.

6.7 Temperature and Humidity Sensor

The Robofan has the ability to automatically change the speed of the fan based on its surroundings. The DHT11 humidity and temperature sensor was used to provide this functionality. By measuring electrical resistance between two electrodes the sensor can measure humidity. The module measures temperature by using a built-in thermistor. The module sends this data to the microcontroller and our firmware adjusts the fan speed based on this data.

6.8 PixyCam

For the motion sensor of the Robofan, a Pixycam has been utilized to track the desired target/user. The Pixycam is useful because it has a thorough premade API. The Pixycam is very small, (1.5 x 2.25 x 2.25 in) so it easily fits on the front of our fan. PixyCam is developed by the company CMU. The project has open source code in the form of the PixyMon software.

6.8.1 Detection

The PixyCam can detect colors by a detection software. Colors, and color patterns can be programmed in as a recognized object by utilizing this detection feature. We are going to use this feature to teach it color sets that are easily identifiable from a range, as well as in an environment that may have many things going on. The camera itself has a manually adjusted focus. This is used to set the range at which the camera gets the best view of its targets. The focus is adjusted by rotating the lens at its attachment point on the camera. The lens is attached by a screw system, and as such has a very wide range of distances it can focus to. However, the focus is difficult to properly adjust once the camera is installed on the device. This is because the camera is going to be placed inside a protective housing. This necessitates the focus to be set during testing to a suitable range. Due to utilizing colors, rather than shapes the focus is not critical to correctly identifying a target. However, the closer to the correct focus distance the target is, the easier it is to keep it centered on the camera view. This is because a target that is out of focus has edges that are not well defined.

This detection feature can be adjusted based on parameters programmed into each color signature. The color saturation and inclusiveness can be adjusted to change the rate of false positives, and false negatives. The light receptiveness can be adjusted in the event of overexposure that causes bright spots to be unviewable. These bright spots can be problematic in areas with large amounts of natural light, or when there are reflective surfaces being viewed. By limiting the light sensitivity, overexposure can be reduced to normalize the colors being viewed. On the opposite end of the spectrum, if an area is too dark, or if the target color is too difficult to determine in a similar background, the light receptiveness can be increased. This causes an effect where all colors are perceived more brightly and contrasted.

We are targeting something neon unlikely to appear naturally, such as a combination of pink and green. We chose this by analyzing our target workshop for color saturation and decided this was the least appearing set of colors. This pattern is applied to both sides of a plain white shirt. This should be very easy to track due to both its size and color. Additional colors can be programmed in for usage depending on different environments that hold different colors. The environment being used should always be as brightly lit as possible, so that the camera has an easier time detecting the pattern it is looking for. These additional color sets can also be used for advanced modes that could track multiple targets and divide air time between them.

We do not need to manually program a machine learning system by using it. We only need to actually teach it. Additionally, the camera only needs to be configured to properly send the output we need from it. This allows us to control our PCB and microcontroller extremely efficiently. The output that is used for tracking is the label of the object found, and its XY coordinates. These outputs are used as specified in section 6.3.

The detection feature of the PixyCam is incredibly useful. Because the camera has a built in software for searching for colors and isolating them in a specified view frame, our group does not need to produce a program to solve this issue. This saves valuable man-hours from both research, and development stages. Additionally, producing a detection algorithm can be incredibly difficult for a single programmer. This is because the current Computer Science student is currently in a computer vision class, and has never worked with such software in the past.

6.8.2 Video Output

This is done through an installable called “PixyMon”. PixyMon is a user friendly interface that is used to control the information sent to the PixyCam. The interface allows for viewing of the camera’s video feed. This view can be set to show either the unmodified view in the top portion of Figure 6-15, or what the camera is currently recognizing as objects. This object view, referred to as “cooked” , the bottom section of Figure 6-15, in the interface is useful to see how well the camera is recognizing what it has been taught.

The two different view types are incredibly useful for debugging. The uncooked view is useful in getting a good idea of exactly how well and far the camera is currently seeing. Using this view allows for a good examination of the areas color saturation, as well as if it is too bright. Using this view allows for a good adjustment of the cameras physical focus as well. The cooked view shows that the camera is currently recognizing as objects, as well as being the mode used to teach it new objects. The following Figure 6-15 shows a cooked view being used to see a pink/green color combination. This view is useful for ensuring that you have taught the camera the correct color or combination of colors. Sometimes the chosen color for training could have been in a bad lighting or contrast area, causing recognition to suffer.

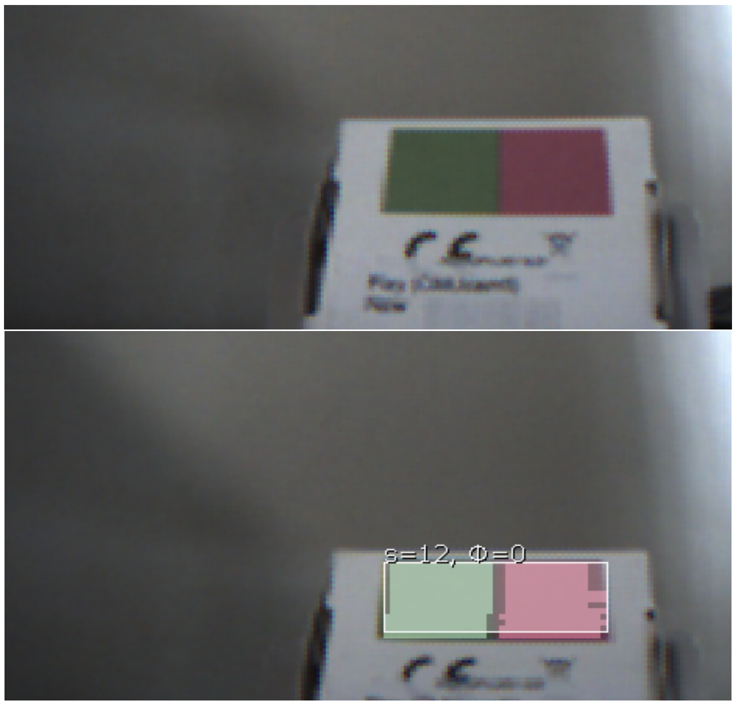


Figure 6-13: Pixycam View with and Without Detection

PixyMon and the camera itself can store data regarding what has been taught to the camera as a seeable object. This is useful for reproducing the project using a different PixyCam. It allows for the original color plan to be sent directly to a new camera, without having to redesign the pattern and program the ID of the new color. Further, because the camera itself has storage a new computer can download information from a previously taught camera.

The camera is going to send data such as X,Y Coordinates, size, and the tag we taught it for the color scheme as well as the value phi. Phi is unused in our implementation. We did not need the size function for this project, but we could use it for advanced versions. This advanced version would be featured as an extra mode that adjusts the fans power. The goal of this new algorithm and mode would be to keep a constant perceived fan power on a target. The primary usage of size would be in attempting to determine how far away the object is for determining the amount of fan power required. Basically, it would be used for a variable power mode to attempt to always get the exact same amount of air blown at someone.

The coordinates are used to control the pan and tilt of the camera. While it is on, it has a searching mode that utilizes the constructed 360 degrees of rotation to find a target, and then lock onto it. Once it finds a target, it exits search mode and enter a tracking mode where its sole goal is to follow a single target and keep it in the center of the camera.

6.8.3 Connecting the Pixy camera

The pixy cam can be used with many microcontrollers that are premade, or configured to use a custom microcontroller. This list includes Arduino, Raspberry Pi, and BeagleBone Black. In order to work with a different microcontroller, the camera can be configured to use serial communications, analog/digital, and USB connections. The serial communications sends information about the target, its location, and can accept information for setting specific servos for the camera that we are not using. The Arduino connection utilizes serial communications. The analog/digital connection has no protocols or interfaces, and must be manually configured to be utilized. The USB connection is designed to use more memory than either other connection, and can also be used to stream video. The USB connection is used for Raspberry Pi, and BeagleBone. It has a micro USB port that is used to teach colors, and that port is being used to attach it to a computer for said teaching.

The connection standards are useful to understand for the construction of the device. The aforementioned information demonstrates the uses of each type of connection that is possible by using the PixyCam. This project utilizes the serial communications from the PixyCam. The serial communications are chosen due to the similarity with UART, as well as ease of setup. It is very easy to follow the pin diagrams as documented by the PixyCam porting guide such that they can attach to our device. Additionally, our device utilizes UART for communications which follows a similar serial communication setup.

6.8.4 UART Communication

As mentioned above, we are using UART to pass messages between the Pixy camera and the MCU. We are using UART because it is asynchronous which allows us to read the messages passed whenever we want to, as they are stored in a buffer until they are read. UART also comes with a stop and start bit on either side of each byte, so that we do not mix up where bytes start or stop. We are not transferring a huge amount of information from the Pixy camera, but even if we wanted to transfer some video we would be able to as the UART from the Pixy camera can handle up to 230 kbaud. This means 230 thousand symbols per second, which would be sufficient to get some simple video to the MCU. We are most likely not going to do this as the Bluetooth module's UART connection is not able to handle it. The main information we are sending over the UART is the X and Y coordinates of the object and the object's signature value.

6.9 Security

The device has several layers of security. The first layer is the connection formed by Bluetooth. Bluetooth is a low range connection, so hacking into the device can only be done locally. The phone can only connect to the device by using a code printed onto the fan itself. This requires anyone that wishes to access the device from a distance to have acquired the code from the fan. Additionally, the fan only recognizes one connected device at a time. This adds an additional layer of security as the connection cannot be hijacked while it is already in use. The phone app also only require a Bluetooth access from the phone, and no additional components that could have vital information or control privileges. This limited scope for the application reduces the negative possible effects of any form of security breach.

Security is very important to consider when designing a device that connects both to a phone, and has a video component. If the video output gets hacked, an intruder gains visual knowledge of anything the camera can see. If the phone app gets hacked by an outside source, any permissions the app has are able to be abused by an attacker. Due to the control of the fan aspect of the application, anyone that accesses another person's fan could adjust control of the fan, preventing the intended user from having air blown at them.

6.10 Stepper Motor Controller

The panning motion of the Robofan is provided by a stepper motor, as explained in the previous section. An important difference a stepper motor has with a servo motor, the kind that is being used for the tilting motion, is that a stepper usually requires more power. This means that steppers can’t rely on a USB powered microcontroller to provide enough power. This is important to consider in the testing stages, since it means both the microcontroller and the stepper motor should be connected to a suitable external power supply. Another key difference, and the topic of this section, is that a stepper motor cannot be controlled by simply connecting it directly to the microcontroller. To operate a stepper, it must be connected to a motor driver integrated circuit. This IC is then connected to the MICROCONTROLLER. A suitable driver must be compatible with the project’s chosen NEMA 17 motor. The stepper motor characteristics that must be considered is the rated current per phase and the stepper type. Stepper motors typically don’t have a voltage rating. Using values as high as 35 volts is encouraged to increase the performance of the stepper motor. This might be useful in situations where the Robofan needs to keep up with a fast-moving user. The Robofan's stepper has a rating of 2 amps per phase and is a bipolar type stepper. In the following subsections, comparisons between possible drivers for the Robofan's stepper have been made below.

6.10.1 L293D

This integrated chip is a quadruple half H-bridge driver created by Texas Instruments. The L293D is often mentioned in stepper motor control tutorials. The idea of choosing this chip is appealing since usage is thoroughly documented and easily explained. Use of this driver does not require any more components than the wires used to connect it to the microcontroller, stepper, and power supply. The L293D requires three pins to control the motor. The state of the pins determines what the motor does. For example, a combination of high, high, and low may correspond to the motor turning clockwise. Unfortunately, this IC is an older model and is obsolete in many ways. For instance, it uses transistors instead of MOSFETS, like many of the other options. Voltage and current regulation is also not featured in the L293D. Current limiting is especially important, as it allows the use of higher voltages, that can improve the stepper's performance. The chip also does not feature any protection circuits for high temperatures, or other situations. Furthermore, the chip is only capable of driving currents up to 1 amp. It's clear to see that this chip is an unsuitable choice for driving the Robofan's stepper motor. Either the stepper motor does not function at all or the motor's performance is degraded greatly, if this chip is chosen. As such, the idea of choosing the L293D for the Robofan were not entertained. Still, it is useful to include this chip in the comparison as it illustrates the progress made in motor driver technology.

6.10.2 A4988

Unlike the L293D, the A4988 is an actual microstepping motor driver designed specifically for the use of operating stepper motors. This integrated circuit is developed by Allegro MicroSystems. MOSFET technology is used in this chip rather than the transistors found in the L293D. The A4988 has many protection features such as current regulation, thermal shutdown circuitry, and short-to-ground protection to name a few. This IC only requires two pins in order to operate the stepper motor. One pin to provide the step signal and the second pin to provide the direction signal. This motor driver features five different step modes, with a sixteenth being the smallest step resolution. In order to utilize this feature and use steps smaller than the default full step, three more pins are required. The maximum load supply voltage is listed as 35 V. Higher voltages can be used to increase the speed of the motor when necessary. Luckily, the motor only needed to reach high speeds if the user is circumnavigating the fan from a very small distance.

There are many breakout boards for the A4988, such as the Big Easy Driver from SparkFun. This is extremely handy for prototyping purposes as breakout boards make the integrated chip ready-to-use without any extra components. Every pin of the A4988 isclearly labeled too, eliminating the need them up on a datasheet. However, the main advantage is the adjustable potentiometer that sets current limit value. This current limit needs to match the stepper's rated current. Without the potentiometer resistors must be placed on the SENSE pins. Suitable resistor values are determined using this equation:

$$I_{TripMax} = V_{REF}/(8 * R_s)$$

In this equation $I_{TripMax}$ refers to the desired current limit, V_{REF} is the reference voltage and R_s is the sense resistor. It can be quickly determined that suitable resistor values are very low, less than an ohm is common. These types of values are not commonly found in the available labs. This is why the potentiometers found in breakout boards are useful. The adjustable current limit also allows for compatibility with stepper motors with different current ratings. While the breakout board certainly has many benefits, ultimately a standalone chip has been bought and integrated onto the Robofan's main circuit board.

The A4988 is looking like a suitable driver controller for the Robofan's panning motor. However, there is one detail that brings up some concerns. This integrated circuited is advertised as being suitable for stepper motors rated at 2 amps. When inspecting the datasheet, 2 amps is listed as the absolute maximum value for output current. It is better to ere on the side of caution on these matters; operating at max values would have no doubt bring about issues of overheating and lowered life expectancy. Overall, the A4988 is a decent option but it is preferable to find a driver that has maximum current value that is higher than the stepper motor's current rating of 2 amps.

6.10.3 DRV8825

This motor driver is developed by Texas Instruments. The DRV8825 integrated chip has many of the same features found in the A4988 as well as some unique to itself. This chip uses chip uses MOSFETS much like the A4988. In addition to the thermal shutdown found in the A4988, overcurrent protection, a Fault Condition Indication Pin, and VM Undervoltage Lockout are featured on this chip. The DRV8825 is capable of microstepping as well. Microsteps all the way down to 1/32 is possible; this is one resolution size smaller than the A4988 can provide. Again, much like the previous driver, the DRV8825 needs a minimum of two pins. To utilize microstepping three more pins are necessary, otherwise the motor only operates using full sized steps. The maximum supply voltage is 47 V, which is higher than the A4988's 35 V.

The DRV8825 can be purchased along with a breakout board as well. If the board features a potentiometer the current limit can be adjusted to be compatible with stepper motors of different ratings. When using a standalone chip the equation for finding suitable resistor values is similar to the one used for the A4988. The only difference is that the resistor value is multiplied by five instead of eight. Breakout boards for the DRV8825 often do not have the headers soldered in. Fortunately, some team members have soldering irons at hand. This makes the breakout boards suitable for prototyping.

Interestingly, the DRV8825 is frequently advertised as a replacement part for the A4988. This goes as far as being able to operate using the exact same code as the A4988. Pinout and interface are nearly identical as well. All this to say, that it is literally a drop-in replacement. Replacing an A4988 running at 1/16 steps with a DRV8825 defaults to 1/32 with the same configuration. This allows for possibly quieter operation with the same functionality. Greater functionality is provided when consider that this integrated chip contains more protection circuitry. With this in mind, it is not a stretch to say the DRV8825 is basically a straight upgrade to the A4988.

Despite all the bells and whistles, the main advantage the DRV8825 has over the A4988 lies somewhere else. According to the datasheet the absolute maximum value for output current is 2.5 amps. The increased output current value allows for some breathing room. Still, a 0.5 amp increase to the max current may not be enough to operate the stepper motor without considering the possibility of overheating and decrease life expectancy.

Stepper Motor Controller Selection

Two of the three examined integrated circuits are suitable for driving the Robofan's stepper motor. Of the two remaining one is a straight upgrade of the other. Not only that, the prices of the DRV8825 and the A4988 are nearly the same. On Digikey a single A4988 goes for $3.27, while the DRV8825 is being sold for $3.84. This is a difference of $0.57. The only possible advantage that the A4988 has is its smaller footprint. The size of the A4988 is 5 mm x 5 mm x 0.90 mm, this is compared to the DRV8825's dimensions of 9.70 mm x 6.40 mm (the third dimension is not provided by the datasheet). The following table summarizes this information.

Table 6-6: Stepper Motor Driver Comparison

	Max Output Current (amps)	Max Supply Voltage (volts)	Price (USD)
L293D	1.2	36	$2.95
A4988	2.0	35	$3.27
DRV8825	2.5	47	$3.84

The table makes it clear that the DRV8825 is the best choice of the available integrated chips. As mentioned in the subsection above the biggest problem with the DRV8825 is the 2.5 maximum output current. This is just 0.5 more than the stepper motor's rated current of 2.0. Although worries about heating and life expectancy are lessened compared with the A4988, they still need to be considered. In order to compensate for this, a heatsink can be attached to the top of the DRV8825. Proper spacing, air flow, and circuit board design can help alleviate issues with high temperatures.

Stepper motor controllers with a maximum output current higher than 2.5 amps are available. However, when going beyond 2.5 amps, controller solutions are much more expensive. Standalone integrated chips or breakout boards are no longer available at this current range. The higher output current ranges are usually reserved for larger devices specifically designed for powering multiple stronger motors. Such devices would not be suitable for the Robofan. For now, the DRV8825 is the best option.

The other notable feature of the DRV8825 is the 1/32 step mode. The Robofan's stepper motor has a step angle of 1.8 degrees. This means that it takes 200 steps to make a full 360-degree rotation. If the DRV8825 is running in 1/32 step mode 6,400 steps are required for a full rotation. This is allows for finer control and greater precision. Other than accuracy, the smaller step sizes provide an additional benefit. Smaller steps sizes can be utilized to ease the motor. When the Robofan pans from one position to another, it tries to cover the distance as fast as possible. This vibrates and shakes the Robofan if panning speed goes from one extreme to another. With proper coding, it may be possible to use the smaller step modes to decelerate the panning motor smoothly whenever the Robofan fan is close to reaching its target position.

6.11 Microcontroller

Selecting an appropriate microcontroller for the Robofan is a crucial decision. The microcontroller lies at the heart of Robofan. It is connected to every component of the project. The stepper motor and its driver, the servo motor, the Bluetooth module, and the Pixycam is connected to the microcontroller. The role of the microcontroller is to send signals from one component to the others. For example, the Pixycam sends a signal to the microcontroller detailing the x and y-coordinates of the Robofan's target. If the x coordinate is negative, the microcontroller sends a direction signal and a step signal to the stepper motor. This tells the stepper motor to begin moving in the counter-clockwise direction. Likewise, a signal is sent, by the microcontroller, to the servo motor based on the received y-coordinate. As for the Bluetooth module, a phone app can send it signals based on the user's desired position for the Robofan. The module sends a signal to the microcontroller, and again the appropriate direction signals and move signals are sent to both the stepper and servo motor.

This leads to the most obvious requirement needed for the Robofan's microcontroller; it needs to have enough input and output pins to accommodate all the necessary components. Three to eight pins might be used for the stepper motor, one digital input for the servo motor, and one pin for the three MOSFETs being used to control the fan's power. The microcontroller needs to have as many as twelve pins available and be able to accept two UART connections to accommodate all the Robofan's components. Ideally, the microcontroller should have lower power requirements to keep the overall power usage of the Robofan low. The microcontroller's performance also needs to be considered. Within comparable price ranges, the microcontroller with the highest clock frequency and available memory are chosen. Higher bit count is also preferred. However, it is possible that an 8-bit microcontroller will suffice since calculations being made are simple. It is important to remember that the Robofan is situated in very high temperature areas, such as a garage, so the microcontroller's recommended operating temperature must have a high maximum value. The final thing to consider in the microcontroller selection is the price of the chip as well as the test board. The test board is needed so prototyping and coding can begin immediately without waiting for the team's circuit board to be designed and delivered. Prototype board use was ceased when the Robofan's custom circuit board was made. Design and fabrication of the printed circuit board began as soon as May 30th.

6.11.1 Stellaris LM4F120

This board is designed around the LM4F120H5QR microcontroller. This controller is from Texas Instruments. Interestingly, this product is no longer being provided by TI. However, the team has been given one by Dr. Richie, the UCF Senior Design Professor. Right at the start, this microcontroller has the huge advantage of costing the team absolutely nothing. This chip features a 32-bit architecture. Relatively fast processing speed, at 80 Mhz. A massive number of pins are available; up to 43 input/outputs can be handled by this microcontroller. The operating range for temperature is between -40 degrees Celsius and 85 degrees Celsius. The maximum operating temp is suitable for the Robofan's intended area of operation. The information contained in this paragraph is compiled in Table X.

Table 6-7: Stellaris LM4F120 Specifcations

Specification	Rated Value
Clock Rate	80 MHz
Architecture	32-bit
Operating Temperature Range	-40 degrees C to 85 degrees C
Available I/O pins	43
Flash Memory	256 KB
SRAM	32 KB
Price (with test board)	$25.95

This chip has great specifications, as illustrated by table, and is already in the team's hands. There is one negative to this microcontroller, unfortunately. Since it is a discontinued item, the documentation is lacking. The actual datasheet for it can't be found on the Texas Instruments website since it leads to a broken link. Furthermore, CAD files for symbols and footprint are not available either. The process of designing a printed circuit board based around this microcontroller is incredibly frustrating. Because time is a major concern, none can be wasted troubleshooting several improperly made circuit board designs that are a result of lack of documentation and files. This microcontroller has several things going for it, and is more than enough for the purposes of this project. Regrettably, the time factor is a huge disadvantage and lessens the appeal of this project greatly. The microcontroller and its test board is used in the component testing stage. The servo motor, stepper motor, and Pixycam were all tested using this microcontroller while the final microcontroller selected is not available.

6.11.2 Arduino Uno

The ATmega328 is a microcontroller developed by Atmel. This is the chip that is found at the heart of the popular Arduino test boards. Specifically, the Arduino Uno has been looked into. All values discussed in this subsection are from the ATmega328 datasheet. The Atmel chip is based on an 8-bit architecture. The processing speed of 20 MHz is less than what the LM4F120 provides, but it is suitable for the Robofan's needs. The microcontroller mostly just received signals from the various components as well as transmitting signals whenever it is necessary for the situation. There are 26 available input/output pins, which meets the requirements of 12 pins. Other specifications include, a storage ROM of 32 kilobytes and SRAM of 32 kilobytes. The operating range of the ATmega328 is between -55 degrees Celsius and 125 degrees Celsius. This range of temperature is greater than what the LM4F120 can handle. The maximum of 125 degrees is a great feature since a Florida garage in the middle of the summer can get to very extreme temperatures. 125 degrees Celsius limit likely provides a longer life expectancy for the ATmega328 compared to the LM4F120's 85 degree Celsius maximum thermal limit. The information found in this paragraph has been summarized neatly in the next table.

Table 6-8: Arduino Uno Specifications

Specification	Rated Value
Clock Rate	20 MHz
Architecture	8-bit
Operating Temperature Range	-55 degrees C to 125 degrees C
Available I/O pins	26
Flash Memory	32 KB
SRAM	2 KB
Price (with test board)	$22.00

One advantage with choosing Arduino is that documentation is very well organized on the website. Every microcontroller, has eagle files and schematics provided. This simplifies the design process in Eagle since much of the work involved in creating a suitable circuit board is removing all the components that are not needed for the operation of the Robofan. Along with plentiful the official documentation, are the numerous user-generated documentation, tutorials, videos and tips. These boards are very popular, and have a very active online fan base that ranges from hobbyists to professionals. Many times, fully functioning, code for various operations can be found online. It is not rare to find, well commented, open source code that is encouraged to be used in by whoever needs it so long as proper credit is given. A quick search reveals that there are libraries dedicated to the operation of stepper motors and servo motors through the use of an ATmega328. This is especially handy for programing the Robofan.

The ATmega328 chip is promoted as a low power chip. This mostly refers to specific built-in features of the chip, such as sleep and standby modes. Other options to reduce power consumption is to use a lower Vcc to 3.3 volts thereby lowering current as well. Reducing the clock speed is another possibility for limiting power usage. The following table illustrates the effects of a combination of the discussed actions. All data is taken from the datasheet.

Table 6-9: ATmega328 Power Modes Comparison

Vcc (V)	Clock Speed (MHz)	Current (A)
5.0	20	13.92
5.0	10	9.03
3.3	20	6.48
3.3	10	3.87

The table shows that by running at lower clock speeds at a lower Vcc the current value is less than half of what the current is when running at the default settings of the microcontroller. That said, the lowest operating settings may prove to not be enough for the Robofan's needs. Lowering the Vcc to 3.3 volts may also reduce the reliability of signals. If this microcontroller is selected, testing has been done using low power settings to see if it is viable. It is possible that code to efficiently switch between operating modes, can reduce power consumption without sacrificing performance. The ATmega328 is a possible chip for the Robofan as it meets all of the necessary requirements.

6.11.3 MSP430 LaunchPad

The MSP430 line is a series of 16-bit chips developed by Texas Instruments. Three out of the four members of the Robofan team have used this series of microcontroller. More specifically, the M430G2553 integrated chip. The microcontroller was used in Embedded Systems along with the Launchpad test board. This means that prototyping and coding for this microcontroller has been smoother because of prior experience with it. Furthermore, this also means that the team has three units of the MSP430 already in possession, reducing the total cost of building the Robofan.

At 16 MHz, the processing speed of the MSP430 is lower than the ATmega328 and the LM4F120. It is possible that this may not be sufficient for proper operation of the Robofan. However, since the team is in possession of three of these microcontrollers already, actual testing with the components. The microcontrollers available to the group have 16 input/output pins available. Generally, all pins of the MSP430 series can be used as an input/output. However, some pins are used for specialty purpose. Care must be taken if these specialty pins are used for input/output instead. Since the chip already provides the required number of pins, this was not an issue, and unless additional components are added to the Robofan it won't be. This series of microcontrollers come in many variants, differing in things such as flash and RAM. The ones already available to the group have 16 kilobytes of flash and 512 bytes of RAM. The operating temperature range of the MSP430 microcontrollers are in between -40 degrees Celsius and 105 degrees Celsius. The maximum operating temperature is in between the LM4F120 and the ATmega328, which were 85 degrees Celsius and 125 degrees Celsius respectively. Life degradation of the chip due to overheating should not be a great concern. The most relevant specifications discussed in this paragraph have been organized onto Table 6-10.

Table 6-10: M430G2553 Specifications

Specification	Rated Value
Clock Rate	16 MHz
Architecture	16-bit
Operating Temperature Range	-45 degrees C to 105 degrees C
Available I/O pins	16
Flash Memory	16 KB
RAM	256 B
Price (with test board)	$9.99

The MSP430 seems to take pride in its low power consumption. These chips feature an ultra-low power mode. These microcontrollers are able to go into four different levels of low power mode. The following table illustrates the difference in current consumption of the various low power modes. All values are taken when the chip is running at a clock speed of 1 MHz.

Table 6-11: MSP430 Power Modes Comparison

Vcc (V)	Mode	Current (μA)
3	Default	300
2.2	Default	200
3	LPM0	55
2.2	LPM0	32
3	LPM2	17
2.2	LPM2	11
3	LPM3	0.9
2.2	LPM3	0.7
3	LPM4	0.1
2.2	LPM4	0.1

It is impressive that at the lowest power modes the current consumed can get as low as a fraction of a microampere. It goes without saying that using low power modes greatly reduce the performance of the microcontroller. Using a lower Vcc negatively affects the reliability of signals as it becomes more difficult for the microcontroller to differentiate between high and low signals. Still, with clever coding the low power modes of the MSP430 chips can be used to reduced power consumption without affecting performance. For instance, the low power modes could be activated whenever the Robofan is targeting a stationary user. Keeping it in low power mode instead of completely shutting off allows it to react to the user when they start moving. The datasheet lists this reaction time as less than a microsecond.

6.11.4 Raspberry Pi

The Raspberry Pi is a chip that is developed, quite fittingly, by the Raspberry Pi Foundation. The standard model boasts impressive specifications with a clock speed of 1.4 GHz and 1 gigabyte of memory. This is magnitudes greater than what is needed to get the Robofan up and running. The Raspberry Pi can also be operated with 64-bits or 32-bits depending on the installed operating system. It's no wonder that the Pi is often referred to as a single board computer instead of a microcontroller. Considering this small computer can accurately emulate classic video games, which requires more power than one might imagine, choosing this to control the motors of the Robofan is overkill. Not to mention power usage would be greatly inefficient.

This integrated chip is extremely popular, so it's not strange that this was an early suggestion for the Robofan. However, just as quickly as it was suggested, it was rejected. Separating the chip from the board is effectively impossible. The documentation that would make this possible is not available. Since one of the most important requirements for the Senior Design Project is to create a custom designed printed circuit board, the Pi cannot be considered. The raspberry pi was not featured in the final comparison table that is found in the section regarding the microcontroller selection.

6.11.5 Microcontroller Selection

Now that all of the microcontrollers that are being considered have been examined in-depth it is time to make the final decision of which is used as the central component of the Robofan. Other than the Raspberry Pi, which was eliminated immediately, none of the remaining microcontrollers have yet been rejected. Each microcontroller has enough pins to support the Robofan, have a suitable level of performance, and have high operating maximum temperature values. The positives and negatives of each microcontroller are measured between in this subsection of the report.

Taking a look at the specifications of each microcontroller shows that their performance varies quite a bit between them. The Stellaris LM4F120 leads the group by decent margin, with the Arduino Uno's ATmega328 following after it, and the Launchpad's MSP430G2553 at the back of the pack. For whatever reason, after skimming the website, the Arduino boards in general seem to have worst price to performance ratio. What's interesting is that the Stellaris does not cost much more than the Arduino Uno. In terms of the price to performance ratio the Stellaris comes out ahead of the other two microcontrollers.

In terms of pins available for connecting components, the Stellaris board once again comes out on top with 43 pins. This is about 60% more than the Arduino board which has 26 pins. Coming in last is the LaunchPad which has 16 available pins. Having an excess number of input/output pins may not seem important, but it does allow the team to have more freedom down the line. For instance, it might be the team improve the Robofan by adding additional functionality requiring more components. The upgrade process went much smoother is if the microcontroller is able to handle the new additions. That said, using multiplexers can increase the number of components that can be connected to a microcontroller when there are not enough available pins.

When comparing the specifications, the Stellaris is the best choice in terms of most parameters. Its only weak point is that the documentation is very scarce. It is no longer being supported by Texas Instruments. There are no cad library files available for download, making this chip much less attractive since much more time would have to be spent on Eagle to make sure footprints and symbols actually match the components specs. There are not many example codes available online for this chip. This means code must be developed from scratch before being able to prototype components reducing the amount of time the team's EE majors can prepare for designing the printed circuit board by wiring and connecting parts with a breadboard. On the opposite end of the spectrum, the ATmega328 and the Arduino board has an embarrassment of riches when it comes with documentation. Every board being sold on the Arduino website has CAD files available for download as well as PDFs of the schematics. The Arduino website has plenty of tutorials and example code from which the team can pull from. The website it even contains specific tutorial sections and example codes for both stepper motors and servo motors. This greatly reduced testing and prototype times. Code can just be pulled from the website to see if the motors are operating correctly. Unofficial resources are also plentiful due to the active online community. For example, a quick browse using an internet search engine reveals that a user on GitHub is making a processing library for smooth servo control, which may prove extremely useful for the project. As for the LaunchPad, documentation and tutorials are not really needed, as previously mentioned three members of the Robofan team have already worked with this microcontroller and prototype board. Fortunately, TI does provide CAD library files for this microcontroller. In this comparison the Arduino Uno has a clear and overwhelming edge.

Considering everything discussed in this section, the ATmega328 microcontroller is selected to be used for the Robofan. The only disadvantage it has is the price of $22.00, considering the other two microcontrollers are free since they are already in the team's possession. Still, it is not enough to make this option unappealing. In terms of its specifications, it meets all the requirements needed for the Robofan and does not fall far behind the Stellaris. The Arduino's greatest asset and the deciding factor is its extensive documentation and availability of CAD library files. This may seem like lazy reasoning but it is extremely important since time is such a major constraint since Senior Design 2 in the summer semester. Using Arduino reduced time during the testing stages. Time is also saved when using eagle. Having to manually create a symbol and footprint for a microcontroller is time consuming. Imagine if, inadvertently, one of the dimensions was inputted incorrectly and was not noticed before being sent to the manufacture. That is at least two weeks waiting on a circuit board only to realize that the microcontroller does not fit. This would be a disastrous situation that must be avoided at all costs. In conclusion, the efficiency and time saving opportunities using an Arduino provides is the reason the ATmega328 is the microcontroller that is used by the Robofan.

7.0 Prototyping

In this section we discuss our progress towards the initial prototype. We have not yet made a 3D diagram of Robofan, but below in Figure 7-1 is shown our initial 2D diagram. This diagram shows how the ceiling mount connects to the pan and tilt joints, and finally to the fan itself. The motor housing holds the motor and also holds many of our various boards and wires, to keep a clean appearance from the outside. The housing that came with the oscillating fan was not large enough to fit everything that we need, so we planned to have a larger one printed. The pixy camera is directly attached to the front of the fan cage and pointed in line with the flow of the air coming off of the fan blades. This ensured that we can use the center of the pixy camera feed as a guide where we want to keep the person we are tracking. We are also discussing our current stock of parts and components, our PCB design, software, and where we are ordering it from. We also discuss our final software coding plan in this section.

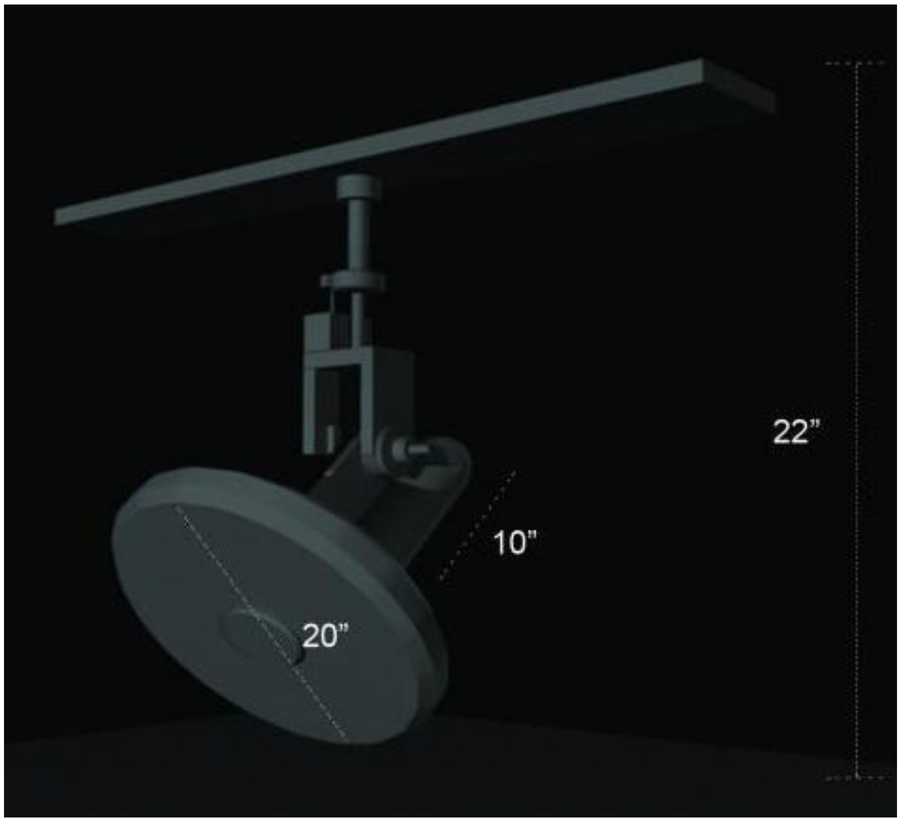


Figure 7-1; Robofan Prototype Sketch with Labeling

7.1 Parts Acquisition and BOM

As of the end of Senior Design 1, we have purchased and started working with several parts. The first part we have purchased is the oscillating fan. We ordered our base oscillating fan from Amazon and promptly took it apart to get a feel for the insides. The fan cage, blade, motor, and wires are going to be used, and most everything else has been stripped off, including the parts that made it oscillate. We are able to mount the fan from the mounting point on the motor, and all of the parts we need to use attach to the motor. We needed to make a new housing to enclose the motor and boards. We have also purchased the stepper and servo motors, and the stepper motor controller. We have started connecting them together and understand how they need to be connected. We have also purchased the Bluetooth module and experimented with how it would be connected. The Pixy Camera has been purchased and tested with quite a bit. The Pixy camera can be connected to a desktop computer quite easily and you can see the output of what the camera is seeing and what the camera is detecting. We have used this to experiment with the detection accuracy and detection range that the camera is capable of and have found it to be sufficient for what we need. For a full list of the prices and where the products were purchased from and when, visit section 9.3 - Budget and finance discussion.

7.1.1 PCB Fabrication Vendor

Once the PCB (printed circuit board) is finalized in the software we chose, the next step in the process is fabrication. We can’t fabricate the PCB on our own due to the equipment required to implement the printed wiring and other embedded components. To get the PCB manufactured for the Robofan we need to choose a PCB fabrication vendor. The two main factors to consider when choosing a vendor are the price and the expected time of delivery of the board. Pricing of a PCB is universally set per square inch with manufacturers in the United States, this unit varies in other countries but still adheres to charging per a unit of surface area. In general, the pricing for PCB fabrication is more inexpensive when ordering overseas than it is in the United States. When ordering a PCB from overseas the expected time of arrival can be up to several weeks as opposed to ordering domestically which usually results in an order arriving within a few days to one week.

7.1.1.1 Vendor Selection

The vendors chosen for consideration have been selected on two sliding scales of the amount of time it takes to receive after placing the order and cost. The reputability of a PCB vendor can only be quantized by the amount of positive and negative reviews the company has. For PCB fabrication vendors, there are not a lot of review forums to accurately get an aggregate score on the quality of a PCB fabrication vendor so the only metrics of which we can use are cost and time of arrival. The right PCB vendor for the Robofan can change depending on the schedule of the project and if a second round of PCBs were needed to be ordered.

Both of these options are thoroughly compared in the final PCB fabrication vendor selection.

There are different PCB types that can be ordered depending on the number of layers needed for the board. At this stage in the design process, a 2-layer PCB has been fabricated instead of a 4-layered board. Four fabrication vendor options discussed in the following sections are explored to see which vendors are appropriate for the Robofan's particular needs. When looking at the vendors below, only the price of 2-layered boards is discussed as the other options don't concern the scope of this project. Most PCB fabrication vendors offer deals to university students, these deals are the pricing parameters of the ordered PCB if applicable. The pricing that is discussed in the following sections are not based on bulk order, the pricing is based on the least number of PCBs that can be ordered. The following options will use the previously stated guidelines to optimize the best price per vendor.

7.1.1.1.1 OSH Park

OSH Park is a PCB manufacturer in the United States that provides for several industries such medical, industrial, and aerospace. The design software files accepted by OSH Park include Gerber files, kiCAD, and Eagle. The PCB design as discussed in section 0.0.0 is utilizing Eagle software, making OSH Park a compatible PCB vendor for the Robofan PCB. No special pricing options are listed for university students so the listed pricing options are what is the lowest cost. The pricing of a PCB order from OSH Park is rated by units of surface, $5 USD per square inch. At this pricing level the 2-layer PCB order comes with 3 PCBs of the single design submitted. The reason behind sending multiple PCBs is to ensure that the consumer receives a working PCB providing one had a manufacturing defect. If all 3 PCBs fail to function than there is a design problem. At the $5 USD per square inch price rate, shipping time is estimated to be under 12 days after the order is placed. Based on a 12 square inch PCB, the total cost of a PCB from OSH Park would be $60 USD. The shipping time for OSH park is moderately fast, if there is a PCB related issue within 2-3 weeks of the project submission a different vendor may I have to be selected to correct the issue in a timely manner.

7.1.1.1.2 4PCB

Advanced Circuits offers PCB fabrication through their 4PCB service. This service is located in Aurora, Colorado in the United States. The software that 4PCB accepts PCB design files from is called PCBArtist. PCBArtist is a design software provided free by 4PCB that has the capability of importing a design from Eagle or any other CAD project making it relatively easy to upload the design and get it fabricated. 4PCB offers a student discount for 2-layer PCB orders. The discount entails a flat rate of $33 USD per PCB ordered with no minimum amount of boards ordered. This discount is very cost effective but doesn't give as much security if the PCB is defective or if one is damaged during testing. Having a service that sends a small duplicate number of PCBs per order, such as OSH Park provides greater security that the project had a working PCB. The student discount requires that the PCB order must be sent to a University address to receive the discount. This could prove to be a problem if we need the PCB on a weekend and we could only get into the mail room on a weekday. The shipping time of a manufactured PCB form this vendor is roughly 5 days, but as soon as 1 day.

7.1.1.1.3 Express PCB

Express PCB is a PCB printing business located in Mulino, Oregon in the United States. The type of PCB software used by Express PCB is a unique and free to download CAD software in two varieties. Expresspcb Classic and Expresspcb Plus. The differences between the two software options are minute except for the pricing of using each software. For all of the options for this vendor, as far as this project is concerned, the PCB in both versions of this software does not come with the silkscreen or solder mask layers, this affects the soldering portion of the hardware design. Eagle is the program the PCB was designed in initially, which is also a CAD based software. If this vendor is to be used than the design created in Eagle would have to be exported or recreated to fit the specific software Express PCB utilizes, which would consume more design time.

7.1.1.1.4 Expresspcb Classic Pricing and Shipping Time

The pricing of a PCB using the software Expresspcb Classic is based solely on the size of the design and quantity. The order of the board starts with a quantity of 1 and goes up from there. Having one PCB is ideal for pricing with the risk of having a defective PCB with no replacement to test. The pricing per 2-layer board is a sum of the setup fee ($63 USD), the per board price ($1.02 USD), and $0.71 USD per square inch. With second day shipping rated at $9.85 USD the total cost of one PCB would be roughly $80 USD for a 9 square inch PCB. The time of shipping for this option is 2 days from the day the order is placed. This is the fastest delivery time out of the options explored as well as the most expensive.

A separate more cost effective option, called the Mini Board is available using this software providing the size of the board does not exceed the dimensions of 2.5 x 3.8 inches. The fixed price of this option is $51 USD and 3 identical boards could have been printed. When using this option, the pricing and shipping time is ideal but the size of the board is severely limited, if the dimensions of the design exceed these than the cost and time savings are a moot point.

7.1.1.1.5 Expresspcb Plus Pricing and Shipping Time

Utilizing the Expresspcb Plus software is similar to the classic version except with minute attributes and schematic changes. The pricing parameters are the same as with the Classic version except for the setup cost, the set up cost would have been less than that of the classic but the value of the setup cost would have been quoted when and only when you have the design for the PCB. The Mini Board version in Expresspcb Plus has all of the same pricing and shipping parameters as its Classic version counterpart. The only difference in Mini Boards is the software used to design each one.

7.1.1.1.6 PCB-Pool

PCB-Pool is a PCB manufacturing service provide by Beta Layout from the United Kingdom. The software PCB-Pool pool accepts are Eagle PCB design files. Since the PCB as been designed in Eagle software this PCB fabrication vendor ranks as one of the most convenient for getting the design made. A pricing calculator is used instead of a given rate, which did not list what sections were being charged for and at what rate. The online price calculator quoted a 2-layer, rigid styled PCB at $67.71 EUR (~$87 USD) with no identical copies sent unless ordered.

7.1.1.1.7 PCB Way/3PCB

PCBWay and its smaller subsidiary 3PCB are a PCB fabrication vendor with their headquarters stationed out of China. When investigating price comparisons for these two companies I noticed that when any information on pricing or specials regarding the pricing for getting a PCB fabricated both sites linked back to PCB Way even though I was sometimes on the website for 3PCB. From this point in the document, the information for 3PCB is the same for PCB Way unless specified otherwise. The PCB manufacturing is done by a tailor-made online calculator which uses mm^2 as the unit of measurement for surface area pricing instead of $inches^2$. On the calculator is a conversion tool that automatically places the correct translation from English standard units to metric. When obtaining the price with the best deal there was a deal that their minimum order of five boards was now the same price for 10 boards. The price of 10 boards with 3 x 3 inches as the dimensions was priced at $5 USD. The price is nearly unmatched when considering price per board. The $5 USD price is rated for standard manufacturing speed of 2-3 days. For express service, 24 hour build time, then the price changes to $39 USD for the same amount of boards. The shipping service for PCB Way is DHL, after the boards are built the shipping time is 3-5 days at $21 USD. The best deal with this fabrication vendor comes to a total of $26 USD for manufacturing and shipping, with the 10 PCBs arriving within 5-8 days.

7.1.1.2 Fabrication Vendor Choice

There are two PCB vendors that we used depending on the time constraints of the Robofan project at the time of ordering. A table compiled with the metrics of each viable PCB fabrication, discussed in the previous section, are displayed below.

Table 7-1: PCB Vendor Comparison

Vendor	Pricing	Shipping Time	Number of Boards	Additional Information
Osh Park	$5 per square inch	12 days or less	3	N/A
4PCB	$33 flat rate	1-5 days	1	Must have access to a university shipping address to take advantage of discount
Express PCB Classic: Standard	63 setup fee + ($1.02 * qty=1) + ($0.71 * board area * qty)	Business Day after Order	1	No silkscreen, No soldering mask
Express PCB Classic: Mini Board	$51 flat rate	1 Day	3	No silkscreen, No soldering mask
Express PCB Plus: Standard	(Less than $63) setup fee + ($1.02 * qty=1) + ($0.71 * board area * qty)	Business Day after Order	1	No silkscreen, No soldering mask
Express PCB Plus: Mini Board	$51 flat rate	1 Day	3	No silkscreen, No soldering mask
PCB Pool	Approximated $87	7 Days or longer	1	Overseas, Currency rate fluctuates.
PCB Way	$26	5-8 Days	10	N/A

For the initial order of the PCB the fabrication vendor OSH Park was used to fabricate the PCB for the Robofan. The primary choice for the first order of the PCB is based on cost efficiency that fit within reasonable constraints. OSH Park is the primary choice because it is time efficient enough to get the PCB with 12 days of ordering as well as the most cost efficient. The approximated cost of the PCB is calculated based on a surface area of 12 square inches with dimensions 3 x 4 inches as $60 USD. If the PCB can achieve dimensions of 2.8 x 3.5 inches or less than the primary choice would be Express PCB, specifically the Classic version of the Mini Board. Both primary choices come with 3 identical PCBs of the design submitted. On the initial test of the Robofan making sure the PCB is not defective is crucial, and the most time efficient way to check is to test one of the other PCBs received; these are the only two vendors within the table that offer duplicates for free.

On the second order for the PCB, providing one is needed, time of shipping was the main priority with cost as a distant second consideration. If the Mini Board was an option for the first order than the Mini Board is still the choice for the second order. If the dimensions of the PCB exceed that of the Mini Board then the second order will 4 PCB if the project can wait 5 days, if not than the Board will come from Express PCB. Like the Mini Board the PC ships fast, but the cost could be well over $90 USD. Using Express PCB for a standard board exceeding 2.8 x 3.5 inches is costly but is needed when time is limited.

7.1.2 ECAD Software Comparison

Designing printed circuit boards with the aid of software is usually comprised of two stages. The schematic stage and then the PCB layout page. Software that falls into this category are referred to as Electronic Computer-Aided Design or ECAD for short. The process starts with creating a schematic. Every component is represented using basic shapes. It is here the designers have decided what components were used and what connections need to be made. In this stage it is useful to organize things based on function instead of how they will actually be placed on the board. For example, users can place the portion of the circuit that relates to the power supply on one side of the sheet. One area of the sheet can deal with the part of the circuit that involves the microcontroller. Another part of the sheet can be dedicated to the portion that amplifies or filters a signal. In short, it is beneficial to break down and compartmentalize the circuit board, to simplify the design process. This allows the users to optimize and fix specific portions of the design without completely adjusting everything already placed on the schematic sheet.

The second part of creating a PCB is to use the newly created schematic with a PCB layout editor. In this stage it is now important to visualize the physical dimensions of the board. Every component and copper trace must be place in a way that is practical and makes sense, making it look much less tidy than the schematic. It is important to consider size constraints and where to place heatsinks if they are needed.

Choosing which software environment is most suitable for the team's needs is a crucial decision to consider. There are three vital requirements the software suite will need to fulfil for to be chosen as the team's PCB design software. The first and most important criteria is if the software has a reasonable price. The lower the price the better, and if free will likely be on shortlist of the programs that might be picked. The second requirement is how compatible the program is with different operating systems. The two EE majors who worked on the circuit board design have a Mac laptop and a Windows 10 laptop between them, both of which are 64 bit. It is preferable that the design software is compatible with both operating systems but obviously must work with at least one of those systems. Lastly, the final key factor that was considered is if the program is lightweight. Since the team was mainly working with laptops it is necessary for the program to be able to perform well on weaker hardware. In the event that none of the programs are compatible with the laptops available, desktops with better hardware are available to the team. Laptops are preferable since work can be moved around easier. It also allows the team meet and compare notes at the school campus. If none of the software that was looked at meet all three of the given criteria, then the one that meets the most was chosen. The requirements are listed in order of importance so a program that meets the first and second criteria was picked over a program that meets the first and third program and so on. In the event of a tie a more detailed examination will need to be made. The number of exclusive features, the ease of use, how intuitive the user interface is, and the number of tutorials and documentation are things that were considered to make the final decision.

7.1.2.1 Eagle

The first program that we looked at is Eagle, which stands for Easily Applicable Graphical Layout Editor. Autodesk is the creator of this program. Quite recently this company used to be referred to as Cadsoft Computer. Eagle features both a schematic editor and a PCB layout editor. Eagle's schematic editor has several features. One of which is the ability to create modular blocks for custom components. These can be saved and be used in other projects. This project will mostly use off-the-shelf parts so this feature will likely not be used. It is nice to have if the need for it arises. Eagle's schematic editor allows the use of multiple sheets for one design. This allows the users to cleanly organize each portion of the circuit board design in terms of their function. Eagle also allows the use of electronic rules checking in the schematic stage. This is a useful feature as it helps eliminate easy design mistakes the users may have overlooked before they can be carried over to the PCB layout stage. The schematic and layout editor of Eagle are synced. This means any changes made in the schematic view were reflected in the layout view and vice-versa. This makes switching between working on the schematic and layout seamless.

The PCB layout editor of Eagle has many features that try to simplify the process of laying out all the pieces of the circuit board. When placing components in layout view, Eagle will try to place wires according to what is created in schematic view. The program will try to make the connections use the least amount of bends as possible and automatically reroute when obstacles are placed in its way. The layout editor features optimization tools that can be universally applied to all designs. These tools include loop removal and concerning tools which can be turned after wires have been place or during the placement. All of these quality of life and automation features can be disabled if the user needs to place things manually or does not agree with Eagle's placement. The software helps users finalize the PCB layout by having a design rule check. This feature can be adjusted and customized to the user's design constraints. Autodesk's library of components is always growing. Using premade blocks from the library is useful since they are well labeled and present key stats about the component. Many of the objects found in Autodesk's library are connected to the manufacturers. This makes buying parts and compiling a price list easy. The last feature to be discussed is Eagles ability to generate three-dimensional models of the circuit boards. This is helpful for making sure the board fits a project's size constraints as well as making easier to create or buy an appropriate enclosure for the board.

Beyond that, Eagle offers a free two-year trial of the professional version to students. This program is also compatible with both Windows and Mac operating systems. If the recommended system requirements listed on the website are accurate, Eagle should perform well on the available laptops.

7.1.2.2 Express PCB

ExpressPCB is circuit board manufacturer that offers both a schematic editor and layout editor to be used exclusively for their website and PCB fabrication services. The programs are confusingly named with the schematic editor called "ExpressSCH classic" and the layout editor called "Express PCB Plus". The schematic and layout editors do not sync with each as in Eagle at this time. However, this is a promised feature when plus version of ExpressSCH Plus is released. The selling point of this product is in its simplicity as the website doesn't list any features to entice customers. Instead of a feature list, the schematic creation process is broken down to four steps on the product's website. Involved with the four-step process is the selection of components, the placement of components, the placement of wires, and the final editing of the schematic. The layout editor is also quite bare compared other PCB design software. Many of its features can be found in other software suites. Its biggest and exclusive feature is the ability to simplify the PCB ordering process by allowing users to order boards from the program itself.

Express PCB is offered for free. The two programs only natively support Windows operating systems and have no plans for supporting Macs. This software is likely the most lightweight of the ones being compared and works well with the available laptops. This software suite was not used by the team. However, Express PCB offers circuit board fabrication services with one-day lead time. This can be useful for quick testing of initial board designs. It's also something to keep in mind if time becomes an issue, which hopefully can be avoided with careful planning.

7.1.2.3 KiCad

KiCad is a set of programs created for Electronic Design Automation. It was first released in 1992 by Jean-Pierre Charras and development is continued by a dedicated KiCad Team. This software suite is a completely free and open source. No paid-version is offered. The package comes with two software, Eeschema and PcbNew; the former being the schematic editor and the latter being the layout editor. Despite being free, the KiCad software suite is fully featured and users don't have to deal with any the artificial limitations, such as board size, number of components/pins/layers, that are often found in the trial versions of other ECAD software.

Eeschema contains many features found in professional versions of design software. The program supports the use of subsheets, which can be used to break up and organize larger circuit designs. The schematics created by this software can be exported into several file types, such as PDF, Postscript, SVG, and HPGL. This makes it easy to import designs into other ECADs as well as making pictures of the design easy to place into documents. This schematic software features an electric rules check that looks for simple mistakes such as output pin conflicts, unconnected pins, missing drivers, etc. After the schematic has been designed the software can generate a bill of materials based on the components used. To go along with that feature, is a community maintained library that users can draw from to populate their schematics with commonly used objects and components.

The layout editor, PcbNew, has no board size limits or components limits as mentioned earlier. However, users are restricted in how many layers they can use. The software supports the use of 4 aux layers, 32 copper layers, and 14 technical layers. This is a very generous amount of layers and will not be hinderance to the design process of this project if this ECAD is chosen. The KiCad website doesn't highlight many features of this ECAD but does mention PcbNew's "push and shove" router feature. This feature partially automates the process of placing wire traces by pushing away already laid traces or by moving tracks around obstacles. This occurs in real-time, as the user is placing the traces on the PCB layout. Another featured tool is dedicated to trying to adjust trace lengths to a user-specified number. This is useful for optimizing the design. Users can attempt to minimize the length of traces or try to match the sizes of specific traces.

KiCad's software suite fulfills all of the team's requirements for choosing an ECAD software. Both PcbNew and Eeschema are completely free. The two programs natively support Windows and Mac operating systems. The KiCad website's listed system requirements are well within the capabilities of the team's available laptops. KiCad is looking like a possible choice of ECAD software.

7.1.2.4 DesignSpark PCB

This is another completely free ECAD software. It is developed and maintained by RS Components as well as Allied Electronics. There are very few restrictions other than a hard limit of 1m x 1m for the size of printed circuit boards. In the context of the team's project, the circuit board needs to be as small as possible so the program's limit of the physical size of the board will not be a problem. The software doesn't have any restrictions on the number of layers, nodes, or connections.

DesignSpark contains many of the same features as the programs listed earlier. It has an object library that users can obtain common components from to be used in schematics. A bill of materials can be generated after the design has been finished, giving an estimate of the entire price of all the used components. DesignSpark is able to import and export files into a number of different file types. This means schematics and layouts can likely be used in different ECAD suites. This is also helpful in making sure the designs created in DesignSpark would be in a format that the team's chosen PCB manufacturer's preferred formats.

The software does have one drawback that the other ECAD suites have seemed to avoid. DesignSpark needs users to be logged in. The program will no longer function if more than sixty days have passed since the user has been online. This is a very strange restriction considering that it is a free software without a professional edition available for purchase. While sixty days is a generous and reasonable span, it is a restriction that doesn't seem to have a purpose for existing. It is not a huge problem but it is one that can be completely avoided by choosing one of the listed ECAD software suites.

DesignSpark PCB meets two of the listed requirements for a suitable ECAD suite. It is inexpensive, in this case free, and it has low system requirements suitable for laptops. The one requirement it failed to meet is native support for Mac operating systems. Since this ECAD suite does not seem to have any sort of edge when compared to the other schematic and layout programs listed above, it is unlikely the team will choose this software over one's with native Mac support.

7.1.2.5 NI Circuit Design Suite

This is the final ECAD suite that were compared. This is a set of software created by National Instruments. The suite actually contains more than schematic and layout software but this comparison will only consider those two. The schematic software is called NI Multisim. This program should be familiar to all UCF students in an ECE major. Being able to use NI Multisim may speed up the process of designing the circuit board, since students will already be acclimated to its user interface. The PCB layout program is called NI Ultiboard. Unfortunately, the two programs are quite expensive, Multisim on its own costs upwards of $1700. A free trial is available for both, but they incredibly restricted. Free users are unable to export Gerber files, which means designs can't be provided to the circuit board manufacturers.

National Instrument's software suite only meets one of the team's requirements for a suitable ECAD. The software is expensive and it does not have native Mac support. The only checkmark it fulfills is low system requirements.

7.1.2.6 ECAD Conclusion and Selection

Five ECAD software suites are compared to find the most suitable one for this project's scope. Three requirements are identified, to help narrow down the possible choices. Of the five ECAD software examined, only two managed to meet all three requirements; they are Eagle and KiCad. The table below will quickly compare the five software suites based on the three requirements.

Table 7-2: ECAD Software Comparison

Software	Price	Windows Compatible	Mac Compatible	Low System Reqs.
Eagle	Free Two-Year Trial / $100 per year	Yes	Yes	Yes
Express PCB	Free	Yes	No	Yes
KiCad	Free	Yes	Yes	Yes
DesignSpark PCB	Free	Yes	No	Yes
NI Circuit Design	$41.95 (Student Ed)	Yes	No	Yes

As Table 7-2 demonstrates, Eagle and KiCad are the software suites that the team should consider using. Of the two, Eagle has the higher system requirements. This is likely due to the 3D model generator that the Eagle features. This feature is likely to be ignored because viewing 3D models smoothly would probably only need a weak but dedicated graphics card. Comparing the user interfaces does not provided an edge to either software as they are quite similar. Comparing the features of each does nothing to end the tiebreaker either. Looking further into the documentation of each software suite shows that there are very few exclusive software tools when compared. It is almost complete feature parity between the two. The only way one gets the edge over the other is when looking at tutorials and documentation available. In this category Eagle wins. The Autodesk site contains several documents dedicated to the use of Eagle. There are also several Eagle tutorials that can be found online. Autodesk provides a forum where users can ask for help or for troubleshooting. Eagle is also available on some UCF computers, meaning many professors are familiar with it. The team's final choice for ECAD software is Eagle.

7.1.3 Initial PCB Design

In this section an attempted start at a schematic and board design on Eagle are discussed. Figure 7-2 is a screenshot of the schematic. Please note this is an incomplete design with some components not yet featured. It is best to view this as the start of trying to learn Eagle's many intricacies.

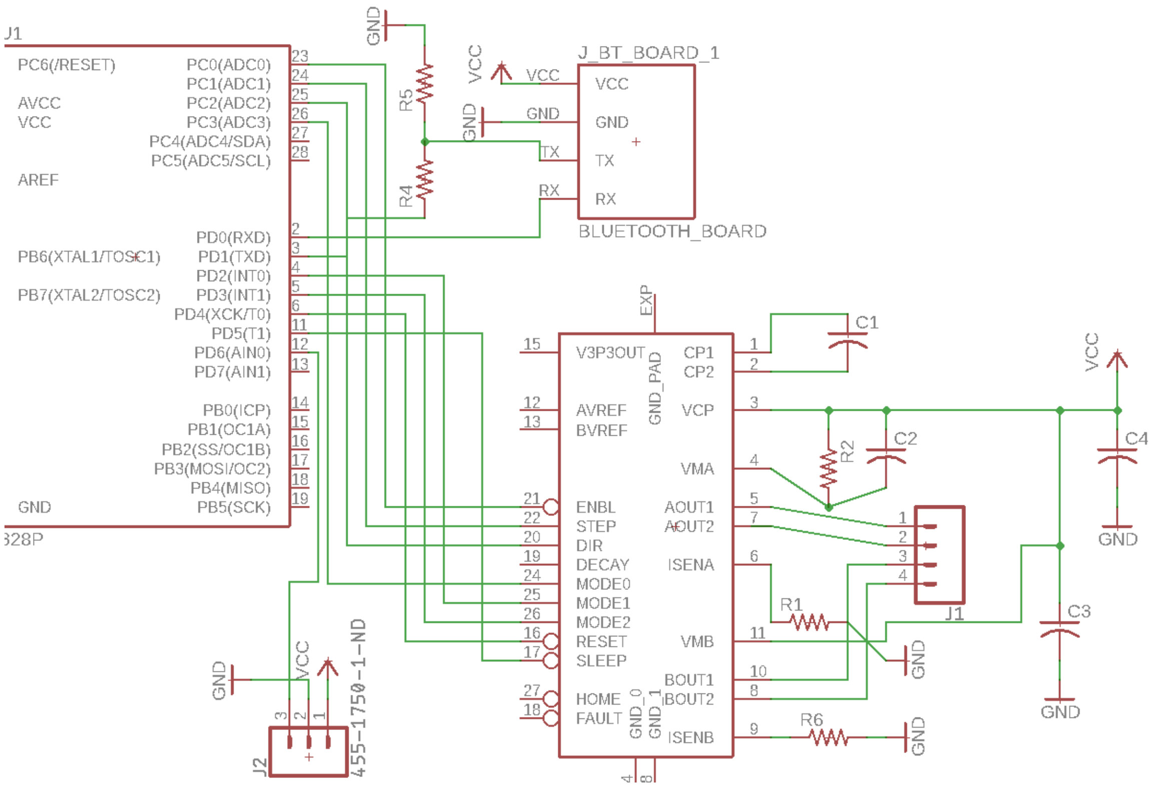


Figure 7-2: Preliminary Robofan Schematic

This is an early start at the Robofan's schematic created on Eagle. There are still some components that have yet to have been added. A socket for the Pixycam must be placed as well as the MOSFETs for controlling the fan speed. Little work has been done regarding power supply, other than placing Vcc and GND symbols wherever they are needed. This is why the left-half the schematic has been cropped out, since those pins of the ATmega328 are mostly related to power supply. Of what actually appears in the document it is likely that there are errors that must be corrected when a more complete design is done.

One may notice that the connections are poorly drawn. This is due to having to get used to Eagle's way of moving objects around. It is somewhat unintuitive compared to something like photo editing software. Moving an object requires clicking on its designated origin, represented with a crosshair. These can be difficult to see causing much frustration. In particular, moving a group of objects is still an elusive action. This requires clicking on the group tool and selecting the desired components. Afterwards, the move button must be clicked. Unfortunately, clicking on one of the grouped components tends to deselect the entire group. Holding the Ctrl is said to be the solution to this but it did not seem to have an effect. When moving around components already connected with nets, the connections are not redrawn. For example, if a component is moved so that a simple horizontal line can be used to connect it to its destination the net will not automatically be adjusted to do that. Instead, the portion of wire near the moving component is awkwardly stretched and bent at angle in order to stay connected.

The default libraries of Eagle leave much to be desired. Downloading and using the Sparkfun and Adafruit libraries are greatly recommended. Many of the products being sold at those storefronts are provided in their respective Eagle library zip files. The default library is not as labeled nicely as the ones provided by Sparkfun and Adafruit. The new downloaded libraries provide actual pictures of the components, detailed descriptions and links to where the parts can be bought. The added libraries are also easier to search through since they are better organized. It is also easier to compile a bill of materials when using these additional libraries. The default library tends to lack many common parts as well. It's possible they were already in standard library, but it was difficult to find any sort of connector. The Sparkfun library had whole a folder labeled and dedicated to connector types.

After the Eagle schematic was attempted the next step was to look at the PCB layout editor portion of the program. Figure 7-3 is a screenshot of the beginnings of trying to design the Robofan's circuit board. It is important to note that the Autorouter tool was used to produce the traces connecting the components. Initially, manual routing was used and the electronic rules check gave the design the okay. However, that design was much less presentable than the one generated by the Autorouter tool.

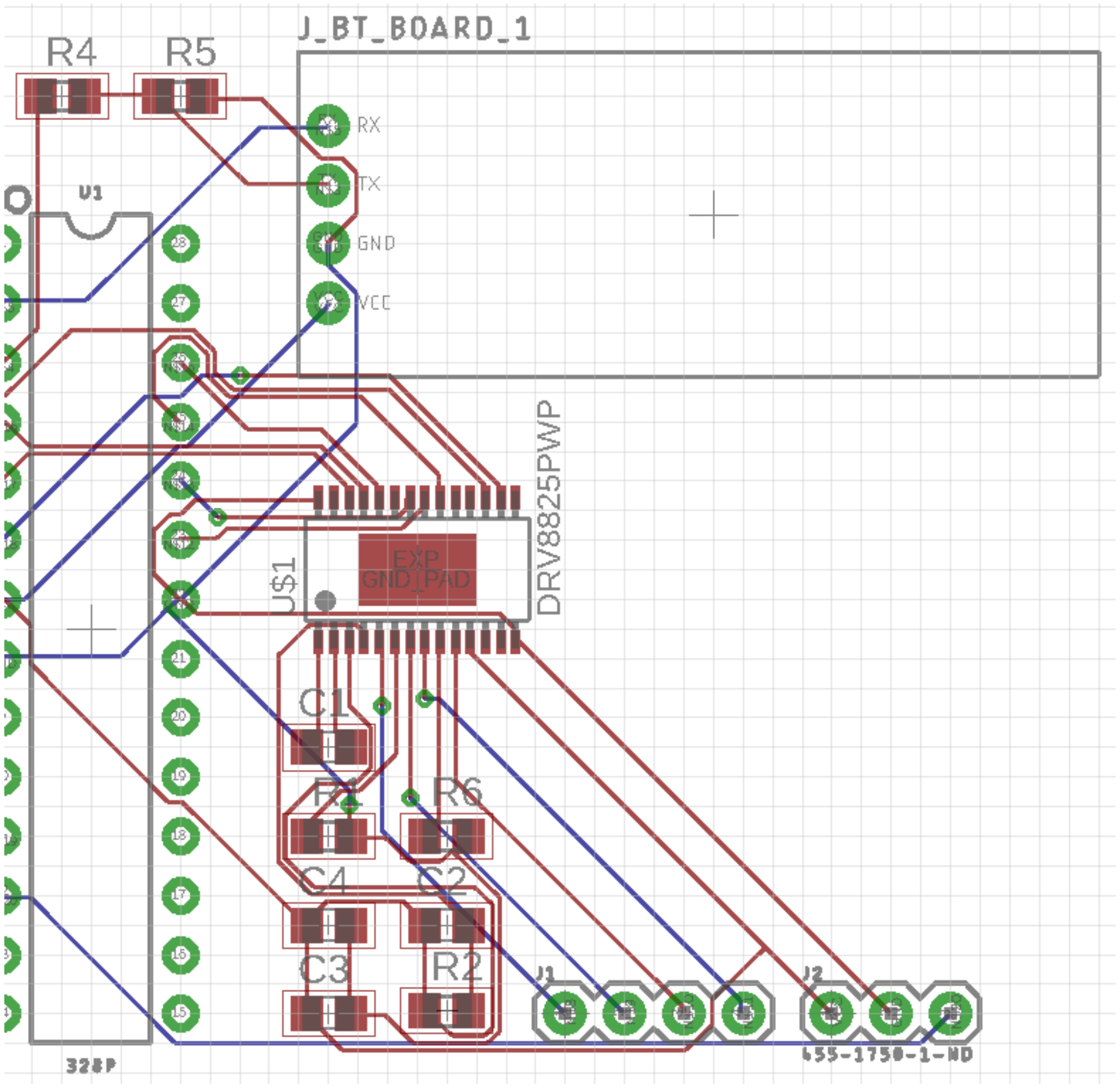


Figure 7-3: Preliminary Board Layout

Again, this is an incomplete design. Nothing more than rearranging the components and drawing the traces has been done. Admittedly, the Autorouter tool does an admiral job at the task. The tool even makes sure to avoid using 90 degree angles and instead uses 45 degree angles whenever possible. The above design was created without adjusting any parameters and just hitting the start button. The final design of the printed circuit board will very likely be a combination of using the Autorouting tool and manually adjusting whatever it generates.

More than just adding the missing components and power supply needs to be done on this layout. Calculations of the trace widths and lengths need to be made to optimize the design. This is because actual traces have a resistance value assigned to them, that is based on the characteristic resistivity of the copper traces, the length, the width, and the thickness. Voltage drop, power dissipation and temperature rise are all factors that must be considered when dealing with a resistance. The entire circuit board design can be a bust if these are not considered before sending the design off for manufacturing. This will severely use up precious time.

The physical dimensions of the board need to be decided. It best to keep traces as small as possible when they are handling signals so it usually a good idea to place the components close together when possible. However, it may also be preferable to spread out components to deal with heating problems. The correct balance must be decided when the actual design is being finalized.

7.1.4 Final PCB Design

The final PCB design was created using EasyEda instead of Autodesk Eagle. At the center of the schematic is the ATmega328p. The upper part of the schematic deals with the power. The 12-volts needed for the DRV8825 stepper driver is provided by a 12-volt power supply connected to the PCB via a standard barrel power jack. A 100 uF capacitor is used to filter any high-frequency noise that may appear in this power signal. To the right of that is the switching voltage regulator, which will provide 5 volts for the rest of the components. Once again, a decoupling capacitor is used for this power signal as well. The right side contains all the components that the microcontroller will interact with. This includes the stepper motor and driver, the servo motor, the temperature sensor, the relays, and the Bluetooth module. There is also a set of pins for the ICSP connection which will allow the PixyCam to be easily plugged in. It is worth nothing that the Bluetooth RX line uses a voltage divider comprised of a 1k ohm resistor and 2k ohm resistor. Signals going to the Bluetooth module must be at 3.3 volts. If it is left at 5 volts than it could damage the components or diminish its lifespan. Normally a logic level shifter would be used, but for our project only the Bluetooth sends signals to the MCU and never the other way around. Therefore, the voltage divider is there as a safety precaution. Figure 7-4 shows the final schematic created in EasyEda.

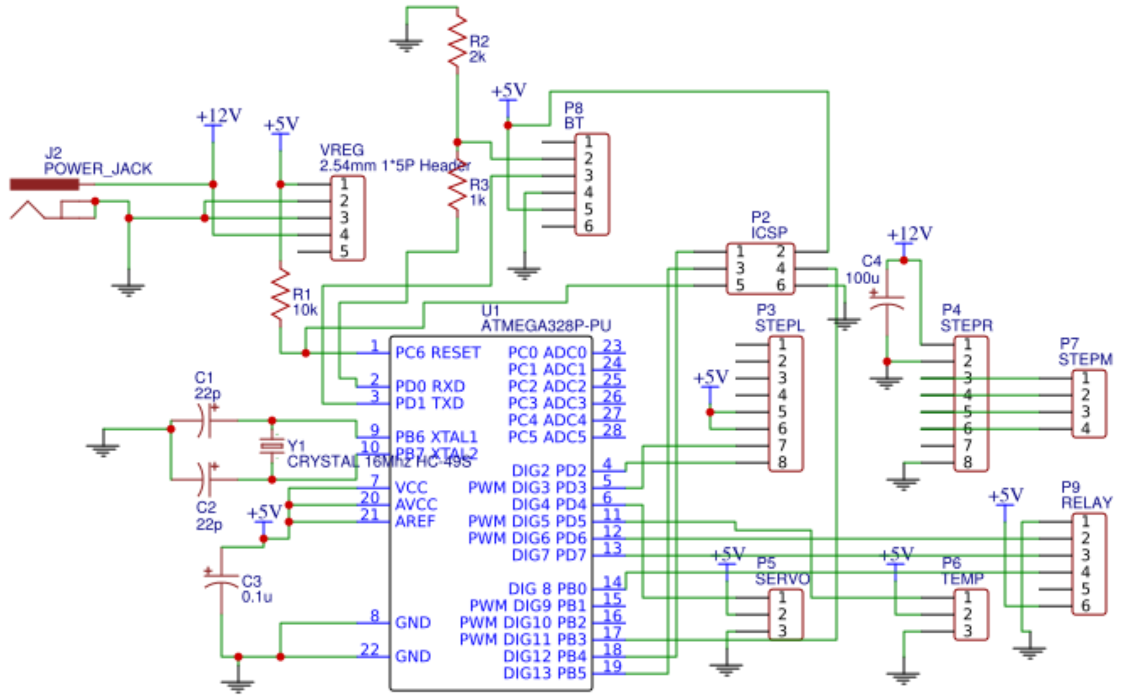


Figure 7-4 Final Schematic

7.1.5 Board Layout

The board ended up having a dimension of 71mm x 81mm. Size was not a major constraint in this project. It just needed to fit in the back compartment of the fan. This meant that the board's size was based on making sure it wouldn't be difficult to solder the components. Three different trace widths were used. A 1.300 mm trace is used for the stepper motor. For the trace for the servo motor's power a width of 0.800 mm was used. Finally, a trace width of 0.500 mm was used for the rest of the components. FIGURE X shows the board layout of PCB design used in the final demonstration.

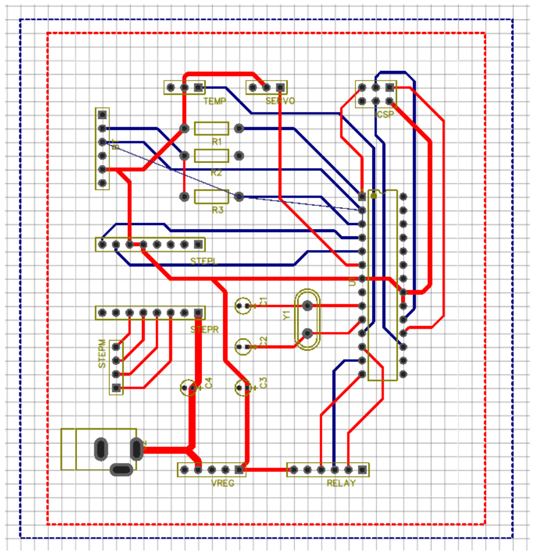


Figure 7-5 Final Board Layout

7.1.6 Surface Mount Layout

A second design was attempted using surface mount components. The biggest difference in this design is that the switching regulator is integrated into the board itself. The datasheet explains what external components need to be connected to the TPS5420 to get the desired voltage output. The circuit is designed to convert voltages in the range of 10-35 volts to 5 volts. The board produced from this design was not working. Instead of the desired 5 volts the output was always 3 volts. Looking back at the datasheet revealed that some components were missing and did not match the application circuit provided by the datasheet. FIGURE X shows the completed board layout for the non-functional surface mount design.

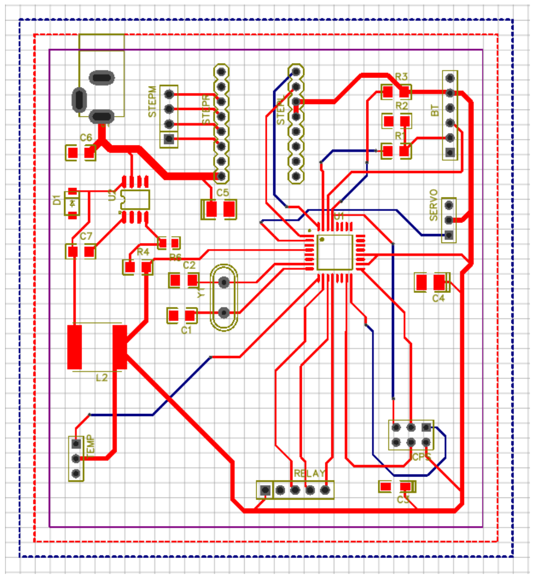


Figure 7-6 Surface Mount Layout

7.2 Final Coding Plan

The final coding plan consists of all design choices being used for the firmware and software. The firmware is programmed in C, and installed on the microcontroller. The firmware programming was done in the IDE Dev C++. The firmware consists of the motor controllers, the tracking algorithm, searching algorithm, and manual override. The firmware must be written with a single threaded processor in mind. This reduces the number of fancy tricks possible by multithreading. The software was written using the Android Studio IDE.

The firmware is going to be as simple as possible. It was written with as concise and small methods as possible. Every function should handle one task. Some functions were larger scale that require the results of other functions to work. The searching algorithm will check if the camera has found a target. If it has not, it will move the motors and check again. If it has, it will enter tracking mode. If the searching algorithm has completed a 360-degree loop, it will check the delay timer. It will wait a set amount of time before searching again. The delay will increase with every loop, until it hits a predetermined cap of 10 minutes. The tracking algorithm will focus on keeping the target centered. A target will only be considered to be moving if it moves more than 10% of its total size as distance away from the center. Once a target breaks this threshold, the tracking mode begins to control the motors. The motors should attempt to travel about 25% of the distance required to reach the center per second. It will accept that the target is centered within a 10% distance from the center of the target, based on its size. It will only check a given direction in X, or Y axis once per pass. However, if a motor has previously been given a move command it will not stop the motors movement to control the other one. If the target's speed changes such that a motor command will no longer cover the required distance, a new command is sent to the motor to update it to the new movement required. The stepper motor is controlled by a function that receives a distance to move, and a direction from the controlling modes. The control functions will then move the motor the correct amount. The Servo control function will receive a numerical value representing an amount to move. The function will then translate this into an amount to move the motor.

The software is going to be located on a phone as a companion application. The phone needs to handle very little. It will connect to the PixyCam via Bluetooth. The application is able to send a signal to the device to turn it on to searching mode. It is able to set the device to manual control mode, and then send manual control signals. The manual control signals will come from user input buttons. There is a user input for the color sets to look at while tracking. It will tell the firmware how to spend time between each color.

8.0 Project Prototype Testing

It is important to know how and where each component of the Robofan is tested. A designated area that contains the necessary equipment to properly power components, measure signals, and develop code is needed to make the prototyping process as smooth as possible. The UCF Senior Design Lab is one such location. The specifics of the laboratory are discussed in one of the subsections. A list of the available equipment is compiled in this section as well. It is necessary to compile testing plans on how to test all the important parameters of the Robofan's components. This is why later sections describe the number ways testing the microcontroller, motors, the Pixycam, and slip ring can be accomplished. Testing is a crucial stage for the Robofan project as it can answer many questions as well as reveal questions that need to be asked. A significant portion of time was dedicated to the prototyping stage and this section is a start at making sure that time is spent as efficiently as possible

8.1 Hardware Test Environment

The hardware for the Robofan will require a test environment conducive to testing proper functional requirements, power flow testing, and space to assemble the final product. The chosen testing area must meet a variety of criteria such as adequate access to multiple power outlets, professional testing equipment for electrical/ electronic components, internet access etc. Having enough space with proper lighting would be ideal for productivity in testing the hardware. The ABET accredited University of Central Florida takes the guess work and hassle of finding a proper hardware testing environment by providing a senior design lab for current students enrolled in the senior design courses. While certain aspects of the hardware can be tested in the homes of the team members, the senior design lab provides more than the average residential home provides in a given area.

8.1.1 UCF Senior Design Lab

The senior design lab is a hardware testing environment set aside specifically for students of the University of Central Florida enrolled in the Senior Design I and II courses. The senior design lab is located in room 456 of the Engineering 1 building within the University of Central Florida main campus. The senior design lab is a locked lab that requires the UCF student ID card and the PID of the same student enrolled in the program to enter. This lab is ideal for working in because it is accessible at all hours of the day and night to senior design students. The senior design lab comes fully furnished with a variety of testing equipment that can be utilized for testing power requirements and if the hardware has any faults from an electrical view point.

8.1.2 Testing Equipment

The testing equipment as far as the hardware is concerned for the Robofan project was provided by the senior design lab. The testing equipment in the senior design lab is tailored to testing hardware and other electrical components. Testing voltage, current, power flow, and signal analysis can all be achieved with equipment found at every workstation within the lab. The equipment found in each laboratory station are listed as follows:

- Tektronix MSO 4034B Digital Mixed Signal
- Oscilloscope, 350 MHz, 4 Channel
- Tektronix AFG 3022 Dual Channel Arbitrary
- Function Generator, 25 MHz
- Tektronix DMM 4050 6 ½ Digit Precision Multimeter
- Agilent E3630A Triple Output DC Power Supply
- Dell Optiplex 960 Computer

Each of these amenities and tools can be seen pictured below in Figure 8-1. The probes and connections for the testing equipment are checked out as a loan or rental to students currently enrolled in the senior design program like the team members for the Robofan project. The connections and probes are leant out for free with the stipulation of having to pay for damage if any befalls the equipment while in the care of the team in question, these regulations are to be expected to ensure the equipment is in working order for future team projects. In the image below is one of multiple work stations that comes with the equipment listed above as well as some of the parts we were testing with said lab equipment.

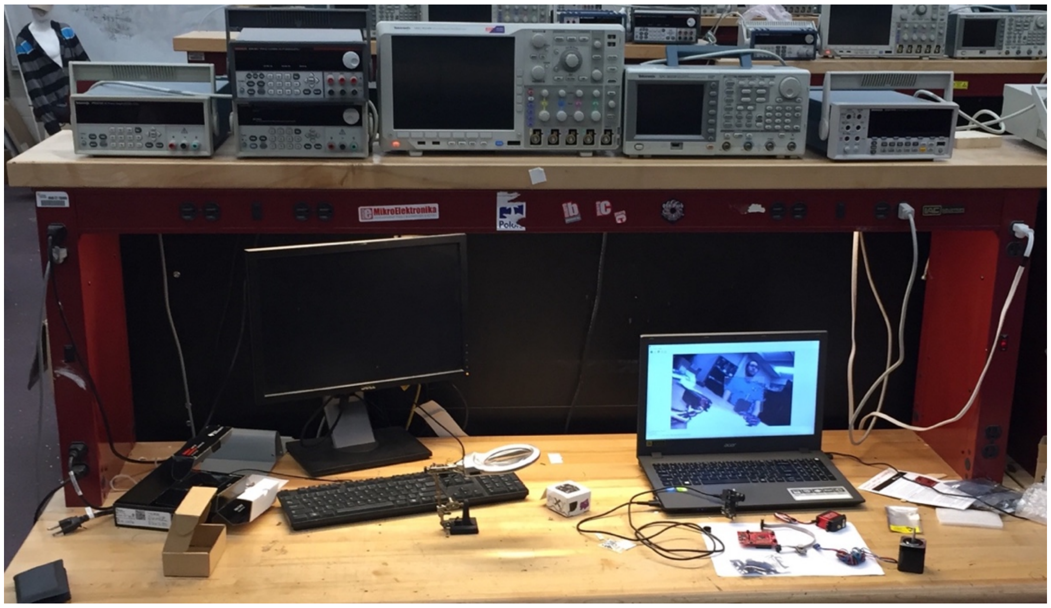


Figure 8-1: Senior Design Workstation with Testing Equipment and Robofan Parts

8.2 Hardware Specific Testing

Testing the hardware was done in two phases. The first phase in testing is to adequately check for proper component function that correlates with the specifications given for the specific part. An example of proper component function with regards to the Robofan is that the Nema 17 stepper motor operates at 6 V and that it moves at 1.8 degrees per step. It is necessary to do these preliminary tests on the individual parts because the whole of the project won't function properly if the smaller sections aren't functioning as they are intended to. Providing a specific component isn't functioning properly i.e. motors, pixy cam, etc. they can be returned and replaced early on to be as efficient as possible with time management as to give enough time to adequately test the working whole of the Robofan.

8.2.1 Motor Testing

The motors chosen for the motion of the Robofan were both specifically chosen with certain specifications in mind. The specifications mainly dealt with power consumption and motion accuracy. In the specifications when each motor was chosen and ordered the accuracy per step and the operational voltage and amperage were specified. When testing for these requirements we need to establish an environment that lacks room for error or interference. The equipment listed in the specific hardware testing environment section 8.1.2 aided in obtaining accurate test results.

The testing criterion for the tilting motor (Lewan Soul digital servo motor) begins with testing to see if the motor operates with a given amount of voltage and amperage as specified in the table within section 6.5 the Power Supply section of this document. To guarantee that the promised accuracy of 1.8 degrees per step is achieved a temporary rig had to be implemented with the tilting motor as the center. The simplest way to do this is to secure the motor in place and attach a make-shift arm to the rotational pin. Placing a protractor underneath the center of the motor and marking the distance the arm moves with an attached pencil allowed us to calculate the rotational angle per step if we provide enough power for one step of the motor. Providing the titling motor meets the criteria that was advertised, it is then ready to be integrated into the final hardware assembly design the Robofan.

For the panning motor (Nema 17 stepper motor) the testing criterion also began with functionality of power. A variable power supply found within the lab was utilized to increase the voltage and current provided to the panning motor as to not overload and/or destroy the motor during the testing process. When testing for the power functionality of the panning motor the proper grounding and controls were utilized in conjunction with the motor, this is necessary to not ruin this particular motor according to research gathered on this make and model. To guarantee that the promised accuracy of 1.8 degrees per step is achieved a temporary rig had to be implemented in the exact same manner as the tilting motor. The only difference in testing the panning motor from the tilting motor is the stipulation of having the proper controller for the motor attached before attempting to run the motor.

8.2.2 Pixy Cam Testing

The tracking sensor, Pixy Cam, had the most testing requirements out of the hardware within the scope of the Robofan project. The first criteria the Pixy Cam has to meet, just like the other components, is that it responds to the correct amount of power given to the component. The power the Pixy Cam consumes is far less than that of our other hardware like the motors and slip ring. Having the Agilent E3630A Triple Output DC Power Supply used in the senior design lab was critical due to the precise, low amount of DC current the Pixy Cam requires.

The first two tests of power and functionality can be done at the same time for this particular piece of hardware. The Pixy Cam has the capability of being powered by a computer with a USB cord. The Pixy Cam is at the same time be communicating with the UART. The real-time footage is then display on the UART. To test if the Pixy Cam functions a green and pink design is printed in the back of the packaging material to aid in the operational testing process. The input for the colors the camera should detect on the packaging are given in the instructions. If the UART displays two white boxes outlining the green and pink designs, then proper functionality has been achieved.

8.2.3 Slip Ring Testing

The slip ring is vital to one of the main goals of the Robofan's function, continuous rotation. Testing the slip ring needs a preliminary make-shift mount to test if the machine prevents a wire from snapping. The only two tests that the slip ring has to pass are that it functions when the right amount of power is provided to it and that given a finite amount of wire the wire does not snap or break when it is continuously rotated around a point. The mount keeps the portion of the slip ring that would be mounted in the final construct steady as the portion in motion spins.

The first test was to see if the slip ring works with the specified amount of power supplied to it. A variable power supply was used to supply amperage and voltage to the slip ring to simulate the final conditions the power supply provides the slip ring without having to wait for the power supply to be completely modified first. Once the slip ring proves it can function with the promised amount of voltage and current then we can test that it prevents wiring breakage. The test for wiring breakage requires the slip ring to be fastened or mounted to a solid surface that won't give way to the motion of the slip ring and that allows for complete rotation without hindrance of functionality. Once the slip ring has been temporarily mounted in place, the next step is to have the remaining wiring loosely held in place below the slip ring by a team member. The slip ring should then be able to rotate without effecting the structural integrity of the wiring. Once these tests have been fully satisfied without discrepancy, the slip ring is ready to be implemented into the final Robofan assembly.

8.3 Software Test Environment

The phone application software was tested in two locations. First, it was tested as each piece is developed utilizing the Android Studio. The testing methods are shown in section 8.4. Second, the application was tested as a final result installed on the developers' phone. Further, the testing should be done both while connected to the fan, and while isolated. The PixyCam was tested both connected to a computer, as well as to the microcontroller. While connected to a computer the camera must be tested for various outputs and target locations. This environment utilizes the software PixyMon to show outputs. The specific testing was repeated on a microcontroller. Here, the cameras outputs were read as labels and XY coordinates for testing.

These environments were chosen for several reasons. The most important consideration when picking test environments was robustness of the tests. All testing must be thorough to the point of mitigating as many potential errors in production as possible. It is imperative to have strong testing environments, because a strong environment allows control of as many variables as possible. Powerful control allows rigorous specific testing.

8.4 Software Specific Testing

The software must be tested both piecewise, and as a system. The following pieces need to be tested. Every individual button in the UI needs to be pressed and its behavior needs to be checked to see that it matches what is planned. The application needs to be launched, and show that it properly activates itself. The fan active button is required to turn the fan on to Searching mode, as well as disable it from any mode. The manual control mode button needs to force the fan to exit any mode it is previously in, or revert it to Searching mode. The view camera feed button needs to transition the application to a new screen that shows the camera view, as well as manual controls. All four directional manual control buttons need to move the camera in the specified direction. Additionally, there is a second toggle control button adjacent to the controls that toggles the controls. The return to main menu button present on the camera feed page needs to return the application to its home state. A button to connect the phone to a fan needs to open a new page that requests an input. This input was the Bluetooth password for the local fan, which was placed on the fan.

The PixyCam software needs to be tested as well. The firmware is dependent on the PixyCam outputting correct information on the targets it is seeing. The camera was tested for various outputs and target locations. First it needs to show recognition of a specific predetermined pattern. It then needs to follow that pattern as it moves around a space. This testing was done visually using the computer environment as specified in 8.3. The camera then needs to show that it can output these values to the microcontroller it is attached to. The microcontroller uses this target information to control the firmware for the searching and tracking algorithms.

Several components need to be tested to ensure the project is successful. Every individual component is going to be tested as well is possible, followed by the pieces assembled in as simple order as possible. A detailed testing plan was followed.

- The PixyCams ability to recognize patterns in the correct environment was tested using PixyMon software. (Can the camera output coordinates and labels for its trained patterns?)
- The Bluetooth modules ability to connect to a phone was tested. (Can data be transferred back and forth between phone and module?)
- The MicroControllers ability to send accurate signals to the motors were tested. (Can the microcontroller move the fan?)
- The ability for code to determine how to send correct signals based on the PixyCams output were tested. (Does the camera stay on target while using motors?)
- The ability for the PCB to function was tested. (Can the LaunchPad be replaced with no change in functionality?)
- The ability for the phone application to control the modes of the fan, and the fan itself was tested. (Can a user manually change settings using the application, and can they manually move the fan while in the related mode?)
- The ability for the fan to work were tested. (Does it blow air?)
- The ability for the fan to transition from inactive to searching mode
- From searching mode to tracking autonomously were tested.
- From tracking back to searching to relocate a lost target (All 3 modes should be tested using the Launchpad and PCB both)

9.0 Administrative

The effective administration of this group is integral to producing both a strong design document, as well as final product. This section discusses the groups actual management, our milestones and how we achieved them, our budget and how money was spent, and why our group members were interested in creating Robofan. The general overview of the group management was that every member has a voice, and we should utilize our small group to the best of its ability. Our group additionally had a very strong communication base, and frequent useful meetings. The groups work was all shared in the most efficient ways we could find, and members frequently assisted each other due to good management. This shows our strong management. Our milestones were set in such a way that we would naturally progress through them as we tackle each task that is set in front of us. Each person was slated to work on milestones that they would be able to effectively contribute to. The milestones reflect on excellent self-awareness of what an individual can either work with, or learn to work with given a timeframe. Our budget shows that we did not have a sponsor, and had to purchase all components of our end product personally. It is clear we chose options that used as little money as possible for this reason throughout the budget. Our budget shows wise use of group resources, with each member being able to purchase components that are relevant to their design sections while still spending comparable amounts of money. The groups individual motivations show that each member strongly wanted to work on this project. We have all dealt with environments that this product could improve the quality of. Further, there are clear indications that every member of the group was very interested in learning how to work with new technologies and tools. For these reasons, we made an effective group. This effectively summarizes our administrative content and what was covered in the following sections of paper.

9.1 Group Management

The first decision that needed to be made regarding the group's management was administrative positions. There are several positions that can be utilized in a group. Some such positions are manager, code master, build master, and document review. With a group this small, the only position that needs to be directly filled is the group manager. With only one primary programmer, code master does not need to be assigned, and this applies similarly to build master. As all members need to contribute equally to the end documents, and no one person is significantly better at writing, there does not need to be a designated document writer. The goal of a group manager is to keep the project moving forward with development. This of course has many components of varying importance. A leader needs to keep the team working cohesively, keep them working within the timeframe and deadlines, schedule and maintain meetings, and manage documents.

During our first meeting, our group held a vote regarding the position of manager. To facilitate communication, our group meets twice a week in person. We chose to meet from 3:00 PM, on Mondays and Wednesdays until we feel we have discussed all relevant topics. During this time, we discuss what we have accomplished between meetings. We plan what should get done per person for the next meeting. During this time, we also provide feedback and assistance where possible to other members. Some duties are swapped based on better understanding and time availability, such as who is writing about a given component. There are several advantages to bi weekly meetings as opposed to once a week. The most important for our group is that it allows for cancellations without drastically setting back progress.

An additional, yet very important facet of communication is handled without face to face meetings. Our group utilizes several platforms to share work and communicate via our computers and phones. The most prominent platform is Discord, followed by a group text. Additionally, communications can be handled via our work sharing in the form of in text comments functionality provided by google docs.

Discord is a free to use messaging software with both text and VoIP capabilities. All features are managed by a server's creator. The server creator can create and manage ranks to control which members have access to what server content. Thankfully our group is small at four people, which means content does not need to be strongly regulated. This allows for global permissions to all users and little time spent handling the server itself. It is very modular and allows for creation of multiple channels for voice and text communication. Our server utilizes a single text and a single voice channel. To improve our communication, we could split our single text channel into several different ones such as: completed work, asking for help, scheduling/deadlines, and off topic. Additionally, Discord has a mobile application that all group members have access to. This allows for further communication even if a member is not directly at their computer. This feature keeps our group in better communication, which allows for more constant feedback when needed. The second platform of communication is very useful because it provides a second way to reach another group member, in the event they do not notice the first attempt to get their attention on our other communication platform, texting.

We share work with each other via online resources. Currently, Google drive is used to share all text and picture based work. We chose Google drive because we are familiar with it, we find it easy to use, and it has a large capacity for storage. Our drive is split into several folders, of which all four members of the group hold edit access. Our folders are: Diagrams, meeting minutes, photos, research, and primary documents. Diagrams holds various tables that are inserted into documents. Meeting Minutes holds transcripts and notes taken during biweekly meetings. Photos holds prototyping pictures during development. Research holds documents of notes with links to be used for writing primary documents. Primary documents have all work that is to be directly submitted. All documents are able to be written on by other members of the group. This provides an additional layer of assistance that can be provided between group members. It is easy for a member to request a quick review of a section from another member. During this review the other member can leave comments on the page about what they think of the writing.

Programming work and projects are being shared utilizing a private repository on GitHub. GitHub is useful because it allows for simple sorting and version control of code. Further, it allows for easy cooperative work on code segments as required by our team's strengths and weaknesses. The private repository can also be made public if we ever choose to go open source with our project to provide a building path for future projects or modifications by endeavoring individuals.

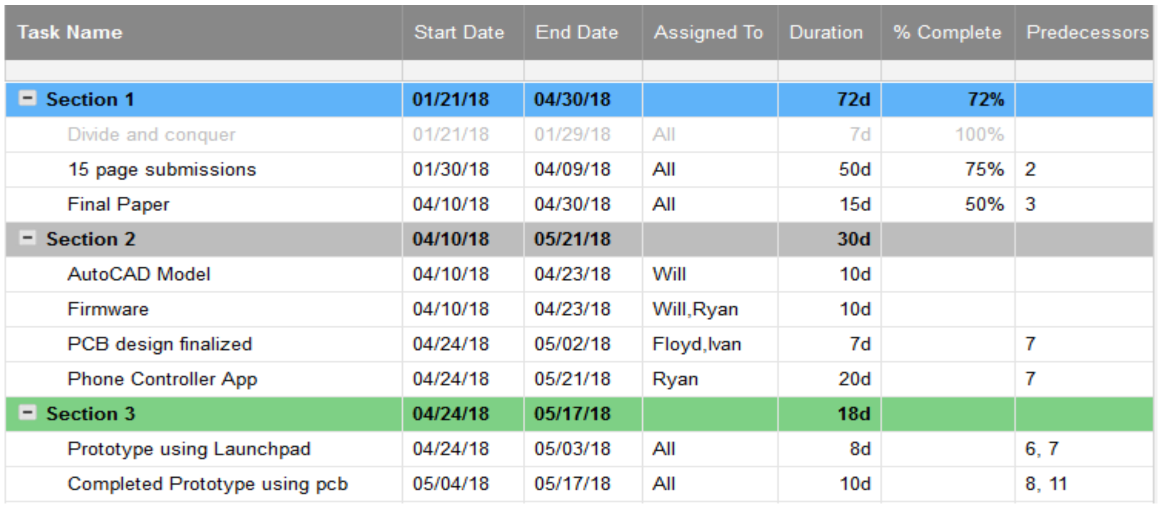
There were building meetings that are not regularly scheduled wherein group members met in a workspace to design and build the fan. These meetings took place in Will’s garage, with different goals set each meeting regarding progress. Some such milestones were: the fan motors can move the fan; the pixy can control the motors; and the fan can track the users. These meetings had an extra focus on getting feedback on the camera’s behavior, as well as the microcontrollers accuracy. This feedback is used to improve the programming of the camera algorithm. Further, the microcontroller must successfully handle all information passed through it without failing.

An additional tool being used to help keep the group both on task and up to date is PERT charts. PERT stands for Program Evaluation and Review Technique. PERT charts track various tasks and events, and show estimated slack and actual completion times. These times are used to develop a critical path, that is how long it could possibly take given the most pessimistic time usages. Once a PERT chart is developed a group can view its own time usages to see how they compare to what was planned. Our group is using these to ensure we stay on task and focus when we need to. Our PERT charts (shown below) are being used somewhat effectively. Our group placed a large focus on PERT charts, as they are very effective as shown by their historical usage.

Producing a PERT chart consists of 3 separate charts. First, a table showing the list of tasks and their estimated times must be produced. The following table (Figure 9-1) was produced using in text editor table making. Next, a Gantt chart must be made. This chart does visually show the overlap and time requirements of each item that needs to be accomplished. Our current Gantt chart (Figure 9-15) is shown below our time table. Finally, a Network chart should be made. Our group however, felt that stopping at a Gantt chart would be sufficient to provide the information required to keep us working on time. We are missing slack time information due to this cutback, but the time spent manufacturing the relevant charts could be better spent actually performing the work.

Table 9-1: Milestone List

#	Task	Predecessor	Opt	Norm	Pess
1	Divide and Conquer	-	4	5	7
2	15-page submission	1	15	30	45
3	AutoCAD Model	-	5	10	15
4	Firmware	-	5	10	15
5	PCB design finalized	4	4	7	10
6	Final Paper submitted	2	20	25	30
7	Prototype Launchpad	3,4,8	5	8	10
8	Phone App	4	10	20	25
9	Completed Prototype	8,7,5	6	10	14



Task Name	Start Date	End Date	Assigned To	Duration	% Complete	Predecessors
Section 1	**01/21/18**	**04/30/18**		**72d**	**72%**	
Divide and conquer	01/21/18	01/29/18	All	7d	100%	
15 page submissions	01/30/18	04/09/18	All	50d	75%	2
Final Paper	04/10/18	04/30/18	All	15d	50%	3
Section 2	**04/10/18**	**05/21/18**		**30d**		
AutoCAD Model	04/10/18	04/23/18	Will	10d		
Firmware	04/10/18	04/23/18	Will,Ryan	10d		
PCB design finalized	04/24/18	05/02/18	Floyd,Ivan	7d		7
Phone Controller App	04/24/18	05/21/18	Ryan	20d		7
Section 3	**04/24/18**	**05/17/18**		**18d**		
Prototype using Launchpad	04/24/18	05/03/18	All	8d		6, 7
Completed Prototype using pcb	05/04/18	05/17/18	All	10d		8, 11

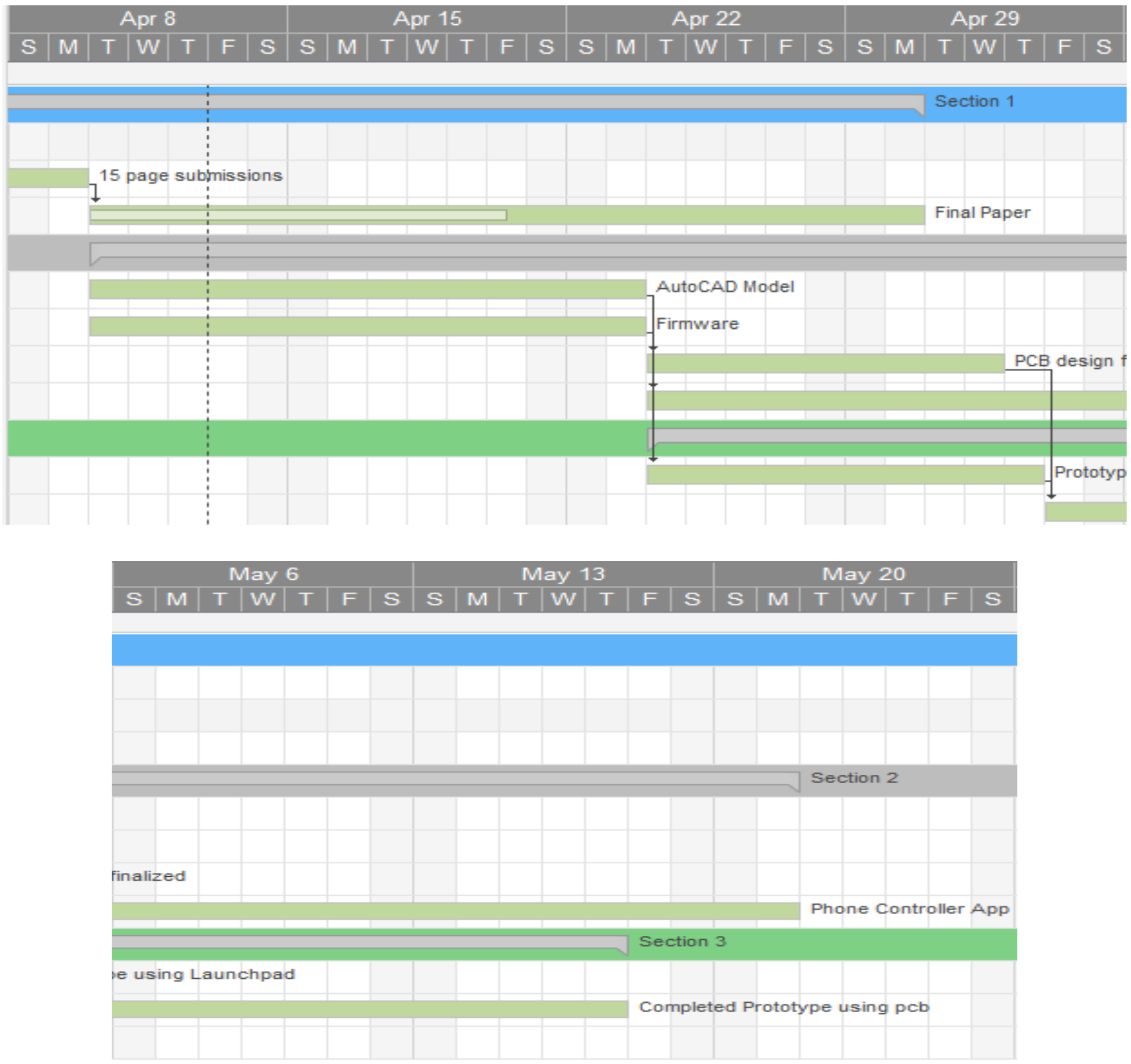


Figure 9-1: Segmented Gannt Chart

9.2 Milestone Discussion

Our milestones primarily focus on paper due dates as well as major design stages. The first milestone is the original Divide and Conquer paper. This paper is expected to be about seven to ten total pages. This paper holds most of our original design discussions, as accomplished before research has properly begun. During this time, we produced preliminary pitches, and ideas for how and what we wanted to achieve during Senior Design. We expected this paper to take about five days. We were not so lucky, but we did finish within our pessimistic assumption of seven days. This milestone was extremely useful in producing future milestones, as it provided a basis on which to build our paper, research, and designs.

The second milestone was our 15-page submission. This was estimated to take about 30 days' total. We were wrong. In total, about 50 days were spent on this milestone. Thankfully, this portion of our design had a very large amount of slack. This slack was due to the large amount of time between this portions completion, and its dependencies being required. The milestone could have been completed quicker if a stronger focused was placed on research and the actual writing of the paper.

The next two milestones are the 3D model of our prototype, and the firmware for the microcontroller. The 3D model is primarily being designed by the group's Electrical Engineers. They felt this would be a good investment of time and energy, as well as a source of learning about skills that they would not normally acquire during standard class taking. The firmware is being designed by the Computer Science, and Computer Engineer members of the group. They are working on this because it is content they are already somewhat familiar with, but would like to further reinforce their skills. These milestones do not have any prerequisites, which makes starting them very easy. Additionally, they are not technically required for the deadlines immediately following their start. This allows for additional slack time on completing them. Because these milestones are being worked on by different people, and are setup as precursors to further separate tasks they can be completed in parallel, without one stressing completion of the other.

The firmware was completed before the submission of the final paper. This is because the next major milestone that needs to be completed is the PCB design. It is imperative to finish the PCB design as early as possible, because once it is complete the part must be ordered and shipped which takes a considerable amount of time. Once our group has the PCB, we have to test it and ensure it works. This testing procedure historically shows that our design is flawed and requires a re-design to occur. This adds more time to an already stressed time period. The PCB has to be fully designed and tested before the prototype utilizing a premade microcontroller and board can be relocated to the PCB for final testing. This means that any slack and time overextension on this milestone can easily cause problems with the final result of the project completion.

Our final paper needs to be submitted about 25 days after completion of our 15-page submission. As our timetable shows, this gives our group very little slack on the final stage of writing and as such requires intense focus and work. While no future deadlines technically require the final paper as nothing is directly based on it, the deadline is still important as it gives us our grade for the semester. This milestone is helpful for future milestones though, because it contains our designs for the project and how we plan on hitting all of our goals. During the rest of our production process the group is going to continually refer to what we have written in our final paper submission.

Following the completion of the final paper, and all prior milestones except for the PCB design, the next thing that should be developed is the prototype utilizing the

TI Launchpad. This prototype utilizes all components, such as the camera, Bluetooth module, and motors. The goal of this prototype is to utilize all prefabricated components to produce a working project without the PCB that has a high incidence of failure. This is important as once the project works without the PCB, testing specifically for the PCB design failures becomes much easier. The phone application should be developed using Android Studio, as mentioned in the relevant section. The application should take about 20 days to complete once the required firmware for the Launchpad is finished. The app was developed by the single Computer Science student currently in the group. This sole developer is the reason for the expected long development time of this milestone. Only the completed prototype requires this milestone to be finished, which allows for a large slack on completion of the app.

The final prototype is the last progress milestone that is to be accomplished. This milestone is the culmination of all progress made prior to this point. We expected it to take approximately ten days to build and test the final design from the point where the prior prototype is complete. When this stage came up it took about 4 days to put together. At this point, all remaining slack was available until our final presentation and showcase. This milestone required the entire group to work together to ensure there are no remaining issues with the project.

Once all milestones are accomplished, the only remaining task is to present our work as required by our class for a grade. This is not currently included as a milestone as it is the culmination of all work done thus far. The presentation included a demo of our project proving it works, as well as the highlights from our paper put into a quick and understandable format.

9.3 Budget and Finance Discussion

Table 9-2 below shows our estimated budget. The prices are approximate as we don't know what exact parts we are going to use, or exactly which vendor we'll be getting them from and whether shipping is included. The quantities listed are what is currently planned as there is a chance some components do not work or get fried in the process of testing. This is something that we need to be very cautious of, as the less parts we need to reorder the more time and money we'll save.

Table 9-2: Initial Estimated Budget

Item	Quantity needed	Price($)	Total Cost($)
Oscillating Fan	1	25.00	25.00
AC-DC power converter	1	10.00	10.00
Stepper motor	1	25.00	25.00
Stepper motor controller	1	12.00	12.00
Servo motor	1	15.00	15.00
Servo controller	1	7.00	7.00
Fan motor controller	1	10.00	10.00
Pixy Camera	1	70.00	70.00
Bluetooth Module	1	8.00	8.00
PCB	2	35.00	70.00
ARM CPU	1	2.00	2.00
Power Switch	1+	Already have	0.00
Slip ring	1	10.00	10.00
Tilt joint	1	25.00	25.00
Wires and other consumables		10.00	10.00
Total			$ 294.00

Figure 9-3 below shows the components that have been purchased as of the end of the Senior Design 1 semester. Unlike the table above, this table includes the exact prices of items including shipping. This table also includes when they were purchased and where they were purchased from. Many of the parts have been purchased from Amazon, as they have competitive prices and we have access to Amazon Prime, so the shipping on these items is quick and free.

Table 9-3: Purchased Components Senior Design 1

Item	Price($)	Units	Total($)	Date	Purchased from:
Lakewood 12-Inch Oscillating Table Fan	19.88	1	19.88	Feb 19th	Amazon
Pixy camera (CMUcam 5)	69.00	1	69.00	Feb 19th	Amazon
DS3218 Digital Servo	17.99	1	17.99	Mar 15th	Amazon
Nema 17 Stepper motor Bipolar	13.99	1	13.99	Mar 15th	Amazon
HiLetgo HC-05 pin Wireless Bluetooth RF Transceiver	7.69	1	7.69	Mar 15th	Amazon
Hobbypower StepStick 4-layer DRV8825 Stepper motor controller	14.99	1	14.99	April 18th	Amazon
Running Total:			$143.54		

In the original document we have also kept who directly paid for which items, so that we can balance out the cost between the group members. This project is financed by the group members, so we need to keep track of each person's expenses so that it can be balanced out between group members at the end of Senior Design 2.

As shown in Table 9-4 below, we are far over our initial estimated budget. There were expenses we didn’t anticipate such as the large cost of lumber for the wooden stand to demonstrate the functionality of the Robofan. If prototyping were not needed in the future the total cost of the Robofan would be a fair deal less than the true total cost.

Table 9-4: Total Purchases

Item	Actual Price($)
Pixy camera (CMUcam 5)	69.00
DS3218 Digital Servo	17.99
Nema 17 Stepper motor Bipolar	13.99
HiLetgo HC-05 pin Wireless Blutooth RF Transciever	7.69
Hobbypower StepStick 4-layer DRV8825	14.99
Arduinos (2)	51.17
Slip Ring	32.53
Additional MCUs	28.19
Miscellaneous Components (Resistors, Wires, etc.)	114.17
Miscellaneous Building Materials (Wood, Glue,Tools)	242.74
Totals:	$612.34

A few variances have been gained through free shipping and finding parts for lower than we originally thought. Being able to find the stepper motor for almost half of what we anticipated was nice, as we ended up not needing as large of a stepper as we originally thought. The servo motor controller that we originally thought was necessary we have found out is unnecessary, so that is a large source of variance.

9.4 Individual Motivations

It is important to remember that all groups are comprised of individual members. Everyone working in a group has their own reasons for being there. An individual should always be aware of both their own, as well as their groupmates motivations. This knowledge helps people work together better, as they can empathize with their groupmates choices. Cooperation has been proven more optimal in situations where the members of a team are more familiar with their own respective goals and values.

9.4.1 Ryan Rossbach

I am motivated to complete this project, in large part because I think it will be fun. Most things I do are controlled by if I perceive it to be fun. I can use this to my advantage by perceiving everything to be fun. I am extremely interested in designing flowcharts that will assist me in developing complicated code algorithms. This allows me to work with new drawing software that I have never used. I have never worked on a phone application either, and I see that there is currently and there will be in the future, a very large need for application developers for mobile devices. Working with Android Studio will be a very useful skill for me to take forwards in my life. Florida is also a comically hot state, doubly so when in Will’s garage working on literally any project. I am desperate for that workshop to be less hot than Satan’s armpit. The fact that this project is required for me to complete my education at the University of Central Florida is a major motivation for me as well. I am very excited to complete this project both because it marks the end of my time at this school and the culmination of my learning, as well as the fact that this project can be added to my resume. I can use it to show that I work well in a team, as well as showing my ability to manage a group of differently skilled peers to complete a daunting task.

9.4.2 William Terry

My personal motivations for this project are to learn useful skills related to my major. Throughout our classes we learned a lot of material, but unless there was a lab section we didn't get much practice applying the materials. I am personally a more hands-on person and I learn better this way, so I feel like this project is a good chance for me to get some hands-on knowledge. I am also looking forward to programming and debugging the PCB, as I have never done something like this before and make it work as it would in a final product. I will be able to learn robust C programming on a PCB with limited memory and processing power, and having it work every time it turns on until it is turned off. Working with Bluetooth will also be a very useful experience, as I have never worked hands-on with any type of networking devices, and Bluetooth is used very widely nowadays. I am also not allowed to graduate unless I pass this set of courses, as with most of the courses I have taken. However, since this project is actually interesting to me as it was my original idea, I'm quite excited to build it and make it work in Senior Design 2. Lastly, I plan to keep the final project mounted to the ceiling in my workshop, as it will significantly increase my level of comfort while working on various projects.

9.4.3 Floyd Thormodson II

The main motivation, I have, to complete this Robofan project to strengthen and develop skills that pertain to the field of electrical engineering. The most relevant skill to develop in this project form an electrical engineering standpoint is PCB design. Multiple jobs in the industry involve the design of PCBs due to PCB technology being a relevant technology to most all electronic devices. Learning a 3-D modelling software for creating the concept for the entire device is intriguing due to my own research into careers within electrical engineering. Although no class within our University offer electrical courses that use AutoCAD within the main curriculum, most jobs prefer a graduate who has experience in AutoCAD or a similar 3-D modelling software. In the catalog of courses I have taken at this school, I have not had the opportunity to combine the skills and knowledge of my collective coursework and utilize them in a real-life project. Completing this project will be my first real-world step so to speak inn becoming a professional engineer. In addition to acquiring new engineering related skills, I have a more immediate motivation to pass this course and graduate from the University of Central Florida with a bachelor’s degree in electrical engineering.

9.4.4 Ivan Sarmiento

No more motivation than wanting to graduate and earn a bachelor’s degree in electrical engineering is needed. I can appreciate the opportunities a well-constructed senior design project can bring. A chance to learn new things in a real-world situation is always valuable. I will have to contend with less than perfect conditions and strict time constraints, something that I will no doubt experience much more in an engineering career. The most important opportunity, however, is the chance to convince myself that I can effectively apply the breadth of knowledge I have gained from all the classes I have taken at the University of Central Florida.

10.0 Appendix 1: Bibliography

[1]"360°Rotating Slip Ring | How does a Slip Ring Work • MOFLON", *Moflon.com*, 2018. [Online]. Available: http://www.moflon.com/slip-ring/slip-ring.html. [Accessed: 24- Apr- 2018].

[2]A. Industries, "Inductive Charging Set - 5V @ 500mA max", *Adafruit.com*, 2018. [Online]. Available: https://www.adafruit.com/product/1407. [Accessed: 24- Apr- 2018].

[3]"Driving a Stepper | All About Stepper Motors | Adafruit Learning System", *Learn.adafruit.com*, 2018. [Online]. Available: https://learn.adafruit.com/all-about-stepper-motors/driving-a-stepper. [Accessed: 24- Apr- 2018].

[4]"L293D Quadruple Half-H Drivers | TI.com", *Ti.com*, 2018. [Online]. Available: http://www.ti.com/product/L293D. [Accessed: 24- Apr- 2018].

[5]"Pololu - A4988 Stepper Motor Driver Carrier", *Pololu.com*, 2018. [Online]. Available: https://www.pololu.com/product/1182. [Accessed: 24- Apr- 2018].

[6]"DRV8825 2.5A Bipolar Stepper Motor Driver with On-Chip 1/32 Microstepping Indexer (Step/Dir Ctrl) | TI.com", *Ti.com*, 2018. [Online]. Available: http://www.ti.com/product/DRV8825. [Accessed: 24- Apr- 2018].

[7]"EK-LM4F120XL Stellaris® LM4F120 LaunchPad Evaluation Kit | TI.com", *Ti.com*, 2018. [Online]. Available: http://www.ti.com/tool/EK-LM4F120XL. [Accessed: 24- Apr- 2018].

[8]"Arduino Uno Rev3", *Store.arduino.cc*, 2018. [Online]. Available: https://store.arduino.cc/usa/arduino-uno-rev3. [Accessed: 24- Apr- 2018].
[9]"TI LaunchPad development kits | Overview | TI.com", *Ti.com*, 2018. [Online]. Available: http://www.ti.com/tools-software/launchpads/overview.html. [Accessed: 24- Apr- 2018].

[10]"ExpressPCB Plus Release Notes - ExpressPCB", *ExpressPCB*, 2018. [Online]. Available: https://www.expresspcb.com/expresspcb-plus-release-notes/. [Accessed: 24- Apr- 2018].

[11]"Eeschema Features", *Kicad-pcb.org*, 2018. [Online]. Available: http://kicad-pcb.org/discover/eeschema/. [Accessed: 24- Apr- 2018].

[12]"PCB Software", *Rs-online.com*, 2018. [Online]. Available: https://www.rs-online.com/designspark/pcb-software. [Accessed: 24- Apr- 2018].

[13]"NI Circuit Design Suite - National Instruments", *Sine.ni.com*, 2018. [Online]. Available: http://sine.ni.com/nips/cds/view/p/lang/en/nid/204742. [Accessed: 24- Apr- 2018].

[14]"SEI CERT C Coding Standard - SEI CERT C Coding Standard - Confluence", *Wiki.sei.cmu.edu*, 2018. [Online]. Available: https://wiki.sei.cmu.edu/confluence/display/c/SEI+CERT+C+Coding+Standard. [Accessed: 24- Apr- 2018].

[15]"Powering Pixy - CMUcam5 Pixy - CMUcam: Open Source Programmable Embedded Color Vision Sensors", *Cmucam.org*, 2018. [Online]. Available: http://cmucam.org/projects/cmucam5/wiki/Powering_Pixy. [Accessed: 24- Apr- 2018].

[16]"Bluetooth Basics - learn.sparkfun.com", *Learn.sparkfun.com*, 2018. [Online]. Available: https://learn.sparkfun.com/tutorials/bluetooth-basics/how-bluetooth-works. [Accessed: 24- Apr- 2018].

[17]"Porting Guide - CMUcam5 Pixy - CMUcam: Open Source Programmable Embedded Color Vision Sensors", *Cmucam.org*, 2018. [Online]. Available: http://cmucam.org/projects/cmucam5/wiki/Porting_Guide. [Accessed: 24- Apr- 2018].

[18]"OSH Park ~", *Oshpark.com*, 2018. [Online]. Available: https://oshpark.com/. [Accessed: 24- Apr- 2018].

[19]"Printed Circuit Board Manufacturer & PCB Assembly I Advanced Circuits", *4pcb.com*, 2018. [Online]. Available: http://www.4pcb.com/. [Accessed: 24- Apr- 2018].

[20]T. www.netpai.com, "LewanSoul", *Lewansoul.com*, 2018. [Online]. Available: http://www.lewansoul.com/product/detail-149.html. [Accessed: 24- Apr- 2018].