SAFER (Safety Autonomous Following Escort Robot)

Michael Burton, Jonathan Dillard, Jordan Harty, and Megan White

Dept. of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816

***Abstract* — SAFER (Safety Autonomous Following Escort Robot) provides an alternative way to get students home safely. It features on-board lights to show the surrounding area, audio to broadcast a warning to nearby people, and a mobile application that uses Bluetooth technology to allow the robot to follow behind the student. On the mobile application, an Emergency Mode can be initiated by the student if necessary, sounding an alarm and flashing the lights on the robot to draw attention when necessary.**

***Index Terms* — Autonomous vehicles, Bluetooth, mobile applications, robot vision systems, human-robot interaction.**

I. Introduction

Whether it be late night studying or visiting a friend, students often travel across campus after dark. For those who feel unsafe walking alone on campus at night, resources like SEPS (Safe Escort Patrol Service) and Knight Ride are there to help make sure you make your way home successfully. Sometimes, these resources are not available or desired. The purpose of the SAFER Knights Project is to provide an alternative to such pre-established precautions on the UCF campus. An autonomous vehicle equipped with various sensors and lights follows students home to provide a feeling of security when walking alone at night. It will follow the student to wherever they need to go on campus. Features such as: an emergency protocol, lights, and speakers are available in order to provide peace of mind to students. Emergency mode involves the user pressing a button on their phone to enable it, and the robot will start flashing lights and start a siren to alert any nearby passerby. Since the robot should be user-friendly, so a mobile app will be developed to easily connect with the robot and provide further information and access to Emergency mode. The robot will be mainly using a Bluetooth connection and PixyCam to follow the user.

From a technical perspective, the robot has to be of a decent size to hold necessary equipment and move fast enough to keep up with the user. The vehicle base is therefore a Power Wheels, usually a toy for young children, which has now been retrofitted to reach our design objectives. To follow the user, a PixyCam plugged into a Raspberry Pi will use computer vision to communicate with the custom PCB. It will know where the user is via a color code, and through coding will know how far away to stay from the user to avoid collision. A speaker and lights will be mounted on the robot as well to light the surroundings and sound an alarm when necessary. The robot will also have space on the back to allow the user to place their backpack or something small, so they do not have to carry it during travel. All of this is meant to keep the user safe in the best way possible while still being a passive observer.

II. Autonomous Driving

In order to move across campus, a vehicle base must be either built or bought to achieve our goals. Since building a base would be timely and possibly costly it was in our best interest to purchase a vehicle that is cheap, functioning, and small enough to get around campus without being an annoyance. Therefore, the Power Wheels was chosen, a toy meant for small children to be able to emulate driving a car. Fairly common, these vehicles can only be used for a short time before the child outgrows them, so plenty of used ones can be found online for relatively cheap. A used Ford F-150 Power Wheels was acquired. To get variable speed and autonomous steering, however, some changes needed to be made to fit our needs.

A. Rear Wheel Drive

The rear wheels of the Power Wheels are the driving force for the car. Originally, the Power Wheels had one speed mode that was enabled when the pedal was pushed down. Two h-bridges were added between the batteries and the DC motors to control the power delivered to the back motors. The h-bridges receive signals from the custom PCB in the front of the vehicle and will alter the speed of the motors when necessary.

B. Steering

A major design challenge came in converting the manual driving to an autonomous steering system to allow the robot to turn on its own when needed. The steering wheel was rid of and holes were drilled into the axle to allow for the new autonomous set up. A 12 Volt Gear Head DC Motor is attached to the bottom of the vehicle and can rotate its axle clockwise or counter-clockwise

depending on the signal received from an h-bridge connected to the custom PCB. As the DC motor axle rotates, a gear on the axle moves in a linear manner to turn the car's axle, which is connect to the moving gear, either left or right as needed. In addition, a 10 kΩ sliding linear potentiometer is mounted to the bottom of the vehicle to track where the moving gear is. The potentiometer sends a signal back to the microcontroller on the custom PCB between the values of 0 to 1023, and the code knows which values mean that the wheels are turned right, left, and center. With this, an RC Mode can run where the user can simply tap to command the robot to go left, right, and forward.

Figure 1. Steering Mechanism with mounted 12 V DC Motor and Sliding Potentiometer

III. Following Capabilities

The main function of the robot is to follow a user. Several ways of doing this were looked into, including infrared, GPS tracking, and computer vision. Computer vision was chosen.

A. Computer Vision

Figure 2. Color Tracking using the PixyCam

Using a PixyCam, a Raspberry Pi will be used to handle the video processing. The PixyCam is a useful fast vision sensor that is made for computer vision. It comes ready out of the box able to sense objects by their colors or bar codes. For simplicity's sake, color was chosen. Initial testing of the camera showed that learning an object by color can lead to false positives if objects of the same color are within the cameras field of vision. Luckily, the PixyCam has the ability to create color codes, which involves using two or more color tags close together. That way the camera will only register the desired object to be followed when the correct color combination is seen, such as orange and green.

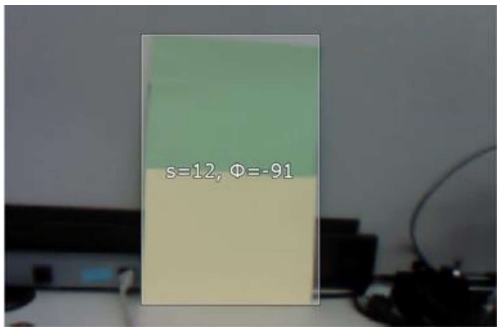

Fig. 3. Yellow and Green Color Code using PixyCam

To follow the user, the camera has to be able to know when the user is left, right, and center. To do this, the user will wear a color code board, which will come with the robot, on their back. The PixyCam is also able to determine the dimensions of an object that it is tracking and its X and Y coordinates in pixels. Since the board stays a constant size, its current dimensions can be measured and used to determine distance. The farther away the person is, the smaller the board will seem, and the robot will know that the user is far away. The code will also know how far left or right the color code is from center, so the robot knows when to turn and when to stay centered.

B. Bluetooth

We will be using a Bluetooth connection to remotely control the robot. We have a HC-05 Bluetooth module. This module utilizes a serial connection to connect to the ATmega328P chip. The Bluetooth typically communicates over a hardware implemented serial bus, but we utilized a specific library called SoftwareSerial that allowed for communication over standard input/output pins and allowed software to emulate a serial bus. When in the main loop of the program, the Bluetooth connection is polled to see if we are receiving any new data. If so, we take the appropriate action. The module is powered by a 5-volt connection on the PCB. This communicates at a default baud rate of 9600 and can be configured to be a master or slave. We use it in the master mode. We have encountered some issue with the connection stability, but

we will talk more about that in the mobile application section.

IV. Additional Features

The main functions and core of the project was to create an autonomous vehicle that could follow a user, but there are several other features that were added to add safety and other benefits to the project as well. This includes a speaker to produce audio when needed, LED lights to allow for a brightly lit path for the user, and an Emergency Mode that the user can enable via mobile application if necessary.

A. Speaker

The speaker is mounted underneath the hood of the vehicle with holes drilled into the opposite side for improved audio quality. An audio amplifier was necessary to power the speaker and therefore needed to be implemented on our custom PCB. Since we can store audio files on the Pi, it is possible to play multiple files whenever we choose to do so. A 9 mm audio cable connects the Pi to the custom PCB and another one goes from the PCB to the speaker as seen below.

B. LED Lights

Spotlights are the first set of lights used in the project. The spotlights are mounted, one on each side of the vehicle, in order to provide light in the forward direction. A separate battery pack, charging circuit, and relay complete the spotlight's subsystem as shown below. When the relay is switched on, it connects the positive wire from the battery to the positive wire of the spotlights. When the relay is off, the switch in the relay is creates an open circuit.

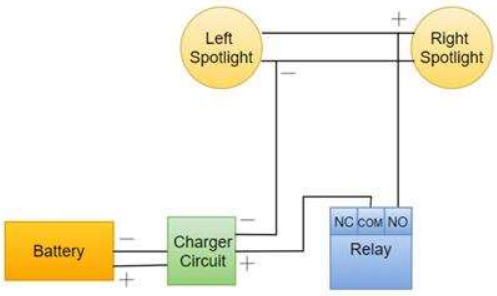

Figure 4. Spotlight Circuit with Relay

The relay is controlled through a PWM signal to the MCU for the purpose of allowing the spotlights an option to flash when emergency mode is activated.

The other lights used in the project are 12 Volt LED strips. Due to the ability to cut the strips to any length necessary and the practicality of mounting, they were clearly the best option in providing light to the sides and rear of the vehicle.

C. Emergency Mode

As mentioned before, SAFER is able to follow the user or be controlled remotely by a mobile application, but there is also another state the robot can be in. This extra state is called emergency mode. To trigger this mode the user will press the emergency button on the companion app. The robot will then go into emergency mode. The robot will then flash its headlights and LED light strip. The robot will also utilize its speaker to start playing a siren sound. These effects will continue until the user chooses to disengage emergency mode using the same button. This feature is meant to scare away any hazards and to draw attention to the user for any potential help. To achieve this mode, we are utilizing all three of our software environments. The mobile app will send the signal to the PCB to engage in emergency mode. The PCB will then start strobing the lights and send a command to the Raspberry Pi. The Raspberry Pi will take care of playing the audio until the user turns off emergency mode.

V. PCB

The custom PCB is the brains of the robot. For this project, the aim of the PCB is to communicate with the Pi, control the steering and rear wheel driving, as well as controlling the lights and audio for the robot. Audio files will be held on the Raspberry Pi and executed when necessary. The basic block design is a simple model to show the connections coming into and out of the custom PCB.

A. Microcontroller Unit

The Atmega328P is a popular and well-known option as it is the microcontroller used in the Arduino Uno board. Widely available and used by many hobbyists and engineers alike, this was a good option since there are plentiful resources online. It has a 32 KB program memory size, 14 general purpose input/output pins, and 6 analog input pins. It supports UART, SPI, and I2C

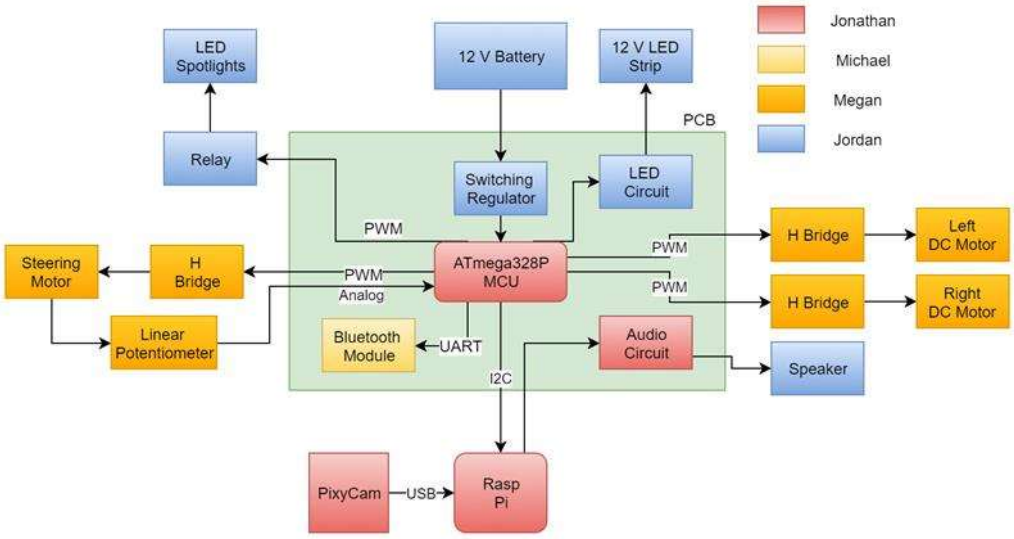

Figure 5. PCB Connection Diagram

connections. A major advantage was that testing for this microcontroller can be done using the Arduino Uno board, which is easy to program.

B. LED Light Strip Circuit

To control the LED light strips that line the sides of the vehicle, the LED light strips need 12 volts of power as well as a signal connection to the microcontroller unit so they can blink when necessary. The light circuit shown below allows for controlling the LED light strips by sending a signal to the base of the transistor, which will allow current to flow when the signal is put forth to turn the lights on. This way, when Emergency Mode is triggered, the lights can be toggled on and off to create flashing.

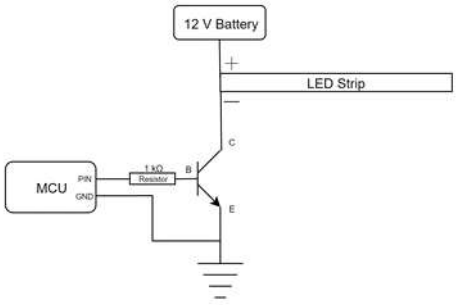

Figure 6. LED Light Strip Circuit

The headlights themselves will be toggled on and off by sending a signal through a relay that is mounted, and these lights can be recharged when necessary.

C. Audio Amplification Circuit

For the audio amplification circuit, the LM384 5 W Audio Amplifier chip is being used from Texas Instruments. On the data sheet was a reference design to use for typical consumer application of a 5 W amplifier using an 8 Ohm Speaker. This circuit connects from the Raspberry Pi, which holds the audio files to be played, amplifies the sound, and then plays it through the speaker mounted to the robot.

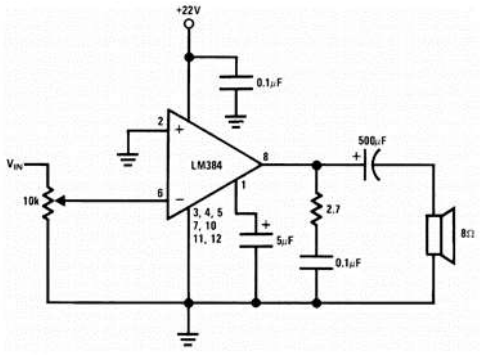

Fig 7. Audio Amplification Circuit from [3]

D. Raspberry Pi

The Raspberry Pi has two main jobs: computer vision and audio implementation. For computer vision, the Raspberry Pi receives information from the PixyCam via

USB connection. An I2C bus is implemented between the Raspberry Pi and the ATmega328P. The I2C connection implements the Raspberry Pi as the master while the ATmega328P is a slave. The Raspberry Pi will then relay the processed vision information through the I2C connection to the ATmega328P on the custom PCB while receiving certain flags from the ATmega328P over the connection upon request. Depending on the received flags the Raspberry Pi will implement multithreading to play audio without impacting the performance of the other aspects of the program.

E. Schematic & Board Layout

Below is the schematic for the custom PCB, made with EagleCAD. On it is the circuit components necessary for the ATmega328P to work, the LED light strip circuit talked about previously, and the audio amplification circuit. There are also various pin outs to connect to steering, driving, power, etc. There are also pin outs for a possible GPS module, compass module, Microsoft Kinect, and a USB hub, which were not used for the scope of this project. There are three on-board voltage switching regulators, two 2 Amp regulators and one 1 Amp regulator. The 1 Amp regulator takes the 12 Volt input and regulates to 5 Volts with a 1 Ampere max current, and this is used for the ATmega328P microcontroller unit. The 2 Amp regulators also regulate the 12 Volts down to 5 Volts but with a 2 Amp max current, and this supplies power to the Microsoft Kinect and USB hub.

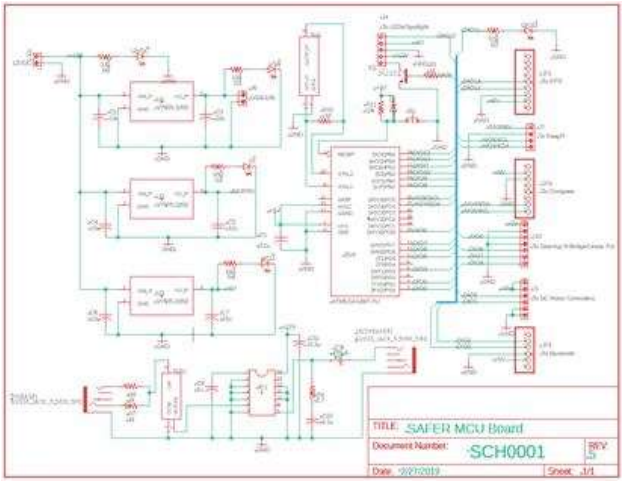

Figure 8. Overall Schematic

The Printed Circuit Board (PCB) was then designed based on the schematic. Due to limitations of the version of EagleCAD that was being used and the cost limitations of the PCB manufacturer we decided to go with a board that is 80 mm x 100 mm. The PCB houses the many main components of the SAFER system including the ATmega328P, audio amplifier, voltage regulators, and component connectors. Most of the passive components including resistors, capacitors, and diodes utilized a 1206 footprint to allow for easier hand soldering. The PCB also features DIP sockets for the ATmega328P and the LM384 for easy replacement of any damaged units. The PCB also contains the amplifier circuit that utilizes the LM384 amplifier chip that allows for the Raspberry Pi that connects to the PCB to play audio through the connected speaker.

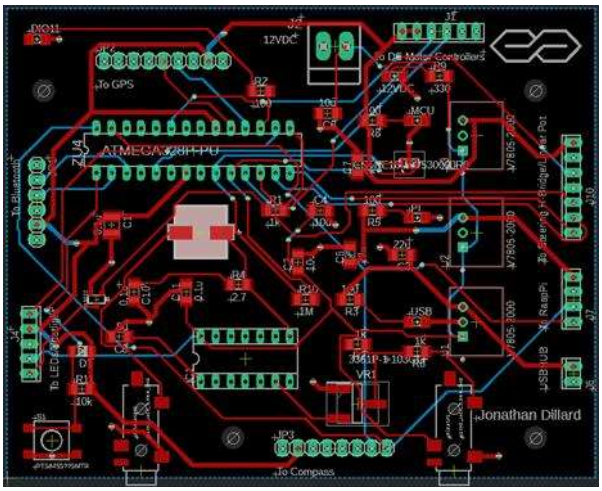

Figure 9. PCB Board Layout

VI. Power

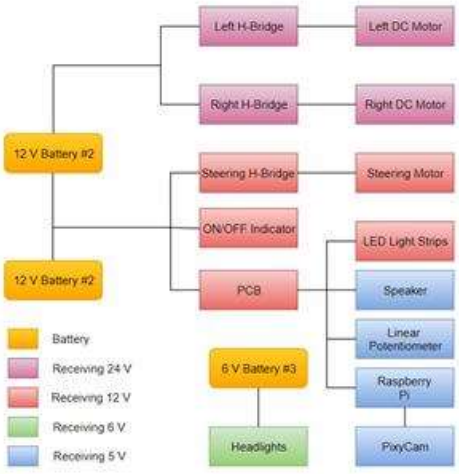

Figure 10. Power Diagram

The two main sources of power are a pair of 12 Volt batteries. These batteries are in series and provide 24 volts to the rear motors and 12 volts to the custom PCB, LED strips, and steering motor's H-bridge. 5 Volt regulators, on the PCB, step down the 12 volts from the battery. This 5 Volt supply then powers the Raspberry Pi, Bluetooth, MCU, and potentiometer; as seen by the PCB layout. Additionally, power is then drawn from the Pi to the PixyCam. Lastly, the headlights subsystem consists of a battery pack, relay, and charging circuit.

VII. Software

There are a lot of different types of software running the robot. The piece that the user will interact with is the Mobile Application, but there is also custom software that is running on the ATmega328P and Raspberry Pi. These three components communicate in various different ways and have different roles.

A. Mobile Application

The user will need a way to communicate with the robot. There a lot of options to do this from an interface directly on the robot, voice commands, or through a mobile application. As the heading suggests we chose to use a mobile application because it offers two main advantages. The first advantage is the ease of use for the users. They will be able to easily issue commands like turning the robot on and off, switching between the remote-control mode and following mode, as well as read information the robot is sending to the user. The second advantage is most everyone has a smart phone so our application will be able to be deployed easily and updated when needed.

We originally used an application development platform called Blynk. It is meant for internet of things (IOT) maker projects, but it had all the functionality we needed for out project. Using this platform saved us from rolling out our own app and user interface, but it ended up working too poorly due to a Bluetooth connectivity issue. The phone and robot were unable to maintain a connection longer than 10 seconds. They would become out of sync and we would get a "packet too big" error. This seemed to be an issue we could not work around because it was baked into the Blynk platform.

In the end we had to create our own phone app. We contemplated which platforms to develop the app and decided to choose Android Studio to make the app. Due to the restrictions including development environment, language limitations, and deployment cost we decided not to implement the application for iOS devices.

Our app will prompt the user to connect to the robot's Bluetooth module. Once connected the user will be able to see messages from the robot about its status. The user then will be able to enable the robot with a master enable switch. Once the robot is enabled the user can choose to activate Remote Control (RC) mode or Follow mode. In Remote Control mode the user can use a slider to select the speed of the robot and set the direction of the robot using a few buttons labeled center, full left and soon on.

Most of the time the robot will be in follow mode. In this mode the robot will be completely autonomous. The user will still be able to turn the master switch on and off, but the movement will be controlled by the following algorithms. At any time, the user can click the emergency mode button to activate the emergency mode. Once the user is done using the robot, they can then disconnect form the Bluetooth connection.

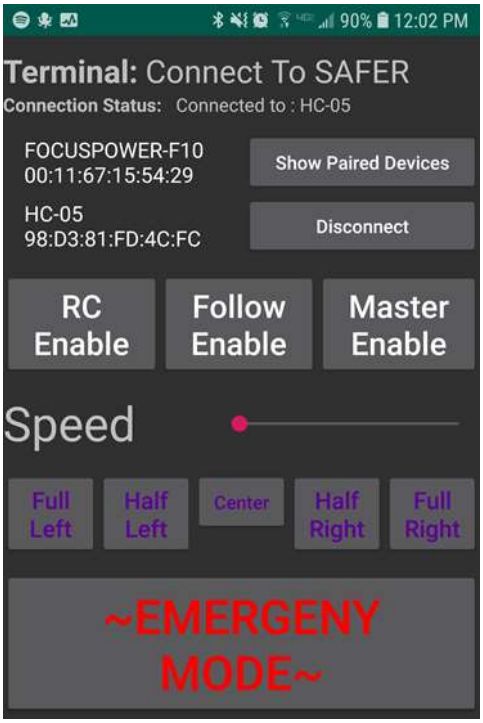

Figure 11. SAFER Mobile Application

B. ATmega328P (Robotics Controller)

The ATmega328P microcontroller is running custom code we wrote. This software is responsible for *controlling* the hardware like steering and the rear wheels. It also connects to the mobile application via Bluetooth and the Raspberry Pi via an Inter-Integrated Circuit (I2C) connection.

The code starts by initializing the connection to the phone and Raspberry and setting up all the variables used in the main loop. It also will get the proper pins for each

of the connected devices and set if they are input or outputs. The robot has two rear motor controllers and one steering controller. All three of these are H Bridges controlled via pulse wide modulation (PWM). There is a Bluetooth module also connected to the ATmega328P.

Once the main loop starts it will first start by poling the Bluetooth connection. If the user has sent a command to the robot it will take appropriate action. For example, the user pressed the follow button, so it will be read in and the following function will initialize and start running. The connection with the Raspberry Pi is an I2C connection. Their communication is interrupt based. This means when the Raspberry Pi sends data to the ATmega328P chip it will stop what it's doing to read that data. This is good for our operations because it means we will always be working off the most recent computer vision data. This connection will work both ways, so the ATmega328P code will send data to the Raspberry Pi when things like the emergency mode is initialized.

For the actual mechanical operation of the robot there are two modes. One where the user can remove control and one used for following. In the remote-control mode, the user will press button to issue commands that correspond to function in the code that steer or adjust the motor speeds. In the following mode the Raspberry Pi will send directional and speed data many times a second. The ATmega328P will read in this data and adjust its internal variables so that when the follow function is called it will change speed and direction appropriately.

To turn the robot, we have a linear potentiometer mounted to the bottom of the vehicle along with a motor that produces linear motion traditionally used to move car seats. As setting arm moves left and right it will also slide the potentiometer. As we are steering the left and right, we are also reading in the values of the potentiometer. This allows us to steer to a very precise degree. If we were not to have this linear potentiometer, we would not be able to know what the position the wheels are in and how to adjust them. This is one of the disadvantages of motors over servos. To engage the rear motors, we can just send their values between 0 and 255 and allow the PWM to do the rest. One issue that we have had to address is blocking while steering. That is, when the robot was issued a steering command it would be stuck in that loop unable to execute any other code until it was done.

This issue is compounded because the steering motor is not particularly fast, so by the time it achieved its goal seconds could have passed. This results in the robot drastically steering off path. To fix this issue we utilized timer interrupts. Utilizing the 16 MHz clock on the PCB, we can set up a timer interrupt to occur at a predetermined time offset. On a set amount of millisecond an interrupt is triggered, and it will check the current position of the steering against a global variable set by the follow or RC modes. If that position is met it will turn off the motor and if it is not met it will send a signal to move in the correct direction. This means instead of being stuck in a loop to check the steering position we are free to run other code. This makes the system much more real time. It has made a significant improvement on the operation of the robot.

A. Raspberry Pi (Computer Vison Controller)

The Raspberry Pi is the brains of the operation. The Raspberry Pi is running Ubuntu operating system with various software packages. The main responsibility of the Raspberry Pi is the computer vision. The Raspberry Pi is connected to a Pixy Cam, a powerful camera with robust opensource libraries used for computer vision. These two modules allow us to write some Python scripts to capture the camera information, process it and then determine how far the user is and if they are in the center, left, or right of the frame.

The user is tracked by an object that they hold or is pinned to them as they walk. This object is a colorful pattern of defined size. It is colorful, so it is easy to distinguish its pattern from other objects in the frame and it has a defined size so that we can calculate the distance to the robot based on its size in the frame. As the object becomes smaller, we know it is getting farther away.

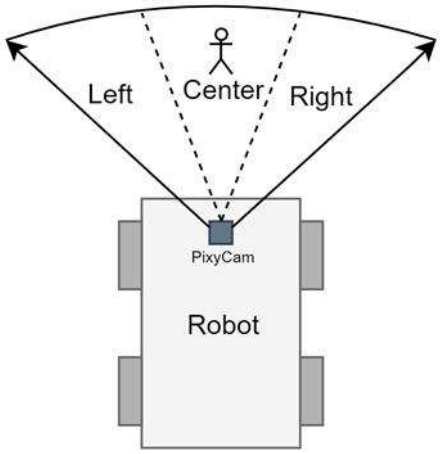

Figure 12. Robot Following Field of Vision

Each frame of video is analyzed to find the following object. Once it is found its size and placement is calculated. If the size of the object is within a defined rage the Raspberry Pi will tell the ATmega328P to speed up or slow down. The placement of the bottom right corner of the following object is marked and if it is within a defined

rage it will tell the robot to turn. For example, if the following object is small and to the left of the frame the Raspberry Pi will tell the ATmega328P to speed up and turn left. The ideas situation is to have following object the save size and placement in the frame. This would be perfect following.

As stated, before the Raspberry Pi will send its data over an I2C connection. The Raspberry Pi will also get data over this connection. When the user engages emergency mode, it will raise a flag that will be sent to the Raspberry Pi where the Pi will play certain audio files on a separate thread.

VIII. Conclusion

The hope is that this vehicle could provide a sense of safety for those who need it while walking to their destination, but it also served as a great way to do research and gain experience in areas we were interested in. Autonomous driving is getting more and more attention these days, as well as computer vision. Having a custom mobile application allowed for various modes to be implemented, so not only does the vehicle have a Follow Mode, it can also be controlled in Remote Control mode. We also managed to hit our main objectives. Providing a sense of safety to the user, giving students another option to protect themselves when walking alone at night. Building a robot that uses Bluetooth capabilities via phone connection while also providing light to see around the robot. Having an Emergency mode in the mobile app meant to help deter possible assailants and alerting nearby people and using a speaker to act as a siren as necessary. Finally, designing a custom PCB. By hitting these objectives, we hope to provide a sense of security at night, because every Knight deserves a SAFER night!

Acknowledgement

The authors wish to acknowledge the assistance and support of Mike Young.

References

[1] “ATmega48A/PA/88A/PA/168A/PA/328/P.” Microchip Technology Inc, 2018.
[2] “UM10204 I2C-Bus Specification and User Manual.” NXP Semiconductors, 4 Apr. 2014.
[3] “LM384 5W Audio Power Amplifier.” Texas Instruments, 2016.