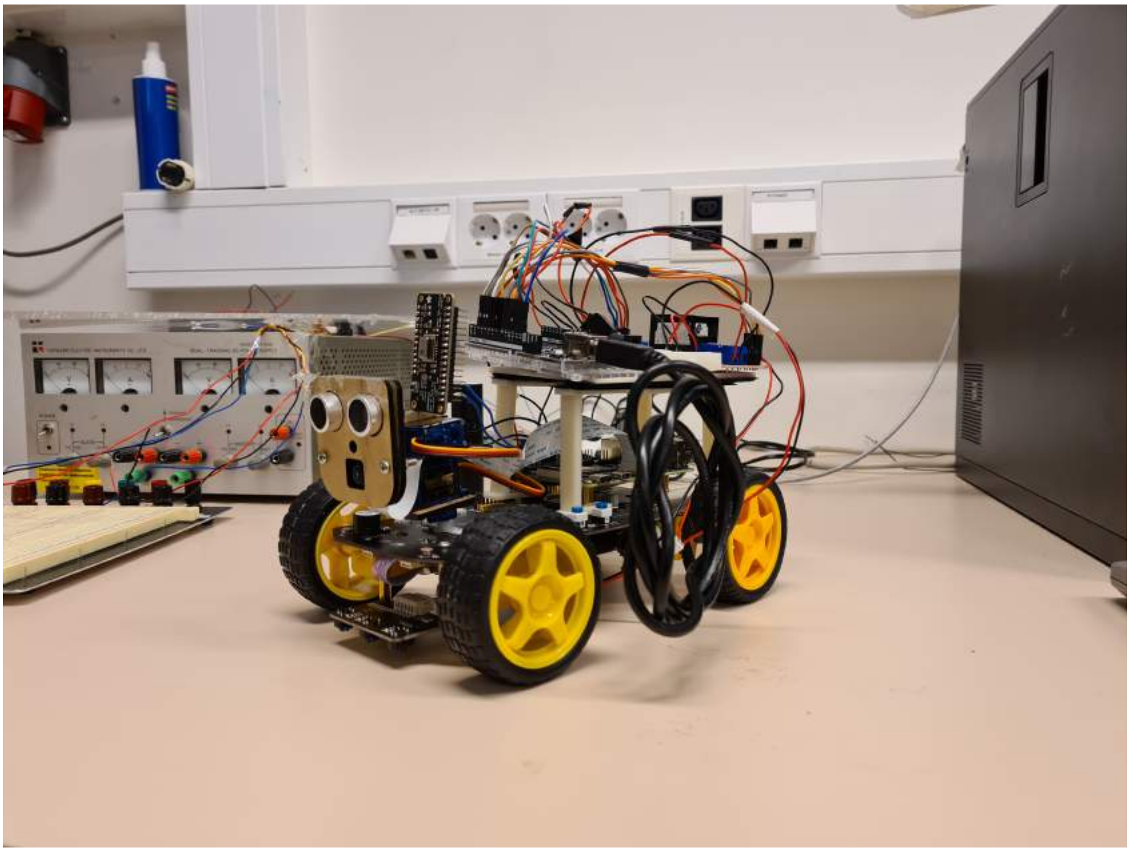
CHALMERS



Controllable car with Surveillance capabilities

Remote controll car with Night vision and Infrared camera

Bachelor thesis in Computer Science and Engineering

Gabraeil Mamoush

Michael Evanson Ngibuini

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
GOTHENBURG UNIVERSITY
Gothenburg, Sweden 2021

Bachelor's thesis 2021

Remote control car with Surveillance capabilities

Controllable car with live footage from an Infrared and Night vision camera

GABRAEIL MAMOUSH

MICHAEL EVANSON NGIBUINI

Department of Computer Science and Engineering
CHALMERS UNIVERISTY OF TECHNOLOGY
GOTHENBURG UNIVERSITY
Gothenburg, Sweden 2021

Remote control car with Surveillance capabilities
Controllable car with live footage from an Infrared and Night vision camera
Gabraeil Mamoush
Michael Evanson Ngibuini

Examiner: Jonas Duregård, Department of Computer Science and Engineering

Department of Computer Science and Engineering
Chalmers University of Technology/ University of Gothenburg
412 96 Gothenburg
Telephone: 031 772 1000

Department of Computer Science and Engineering
Gothenburg, Sweden 2021

Sammanfattning

Mobila enheter som skapar bilder med hjälp av det infraröda spektrumet för att skapa bilder är tillgängliga men till en hög kostnad. Det finns en lucka där enheter som använder det infraröda spektrumet för att skapa bilder samtidigt som de styrs av användaren men till ett billigare pris. Denna uppsats syftar till att skapa och tillhandahålla ett litet koncept för en liten övervaknings bil som kan fungera som ett övervakningsverktyg samtidigt som det är billigt och användarvänligt.

Projektet kommer att testa konceptet med hjälp av hårdvaran som finns tillgänglig på marknaden till ett billigt pris, ett kit som kan driva hårdvaran som styr kamerorna och innehåller verktyg för styrenheten som bestämmer robotens rörelser.

Den projekt kommer även att gå igenom de verktyg och hårdvara som används och hur de monterades tillsammans med flera sektioner som täcker implementeringsprocessen. Denna process kommer också att visa de valda konfigurationer. Resultatet av projektet är ett koncept av en övervaknings bil som är billig och kan mäta föremål med hjälp av det infraröda spektrumet. Bilen kan också fungera i miljöer med svagt ljus tack vare de utvalda verktygen som användes för att testa funktionen.

Resultaten visar att ett koncept för en liten övervaknings bil är möjligt men kan kräva hårdvara som inte finns till ett billigt pris och kräver skräddarsydda verktyg som kan driva hårdvaran längre. Det tas även upp de olika sätt som kan utforskas gällande förbättringen och tester. Dessa rekommendationer tas upp tillsammans med de frågor som dök upp under projektet när det gäller etik och hållbarhet. Det tas även upp hur detta arbete kan påverka enligt de tre dimensionerna kring hållbarhet.

Nyckelord: Infraröda Spektrumet, Övervakning, Arduino, Raspberry Pi, Mörkerseende, Infraröd Kamera.

Abstract

Mobile devices that create images with the help of the infrared spectrum to create images are available but at a high cost. There is a gap where devices using the infrared spectrum to create images while being controlled by the user but at a lower price. This paper aims to build and provide a small concept of a mobile car capable of performing as a surveillance tool while being cost and user friendly.

The paper will test the concept using hardware available in the market at a lower price point. The project will utilize an available kit capable of delivering power to the hardware while being price friendly. The project will only focus on using hardware that is easy to use on the Arduino and Raspberry Pi.

The paper will provide the tools and hardware used and provide multiple sections that cover the implementation process. The result is a concept of a surveillance car that is cheap and capable of measuring objects with the aid of the infrared spectrum. The car can function in low light environments and measure thermal radiation thanks to the provided tools used to test the feature.

The findings show that a concept of a surveillance car is possible but can require hardware that is not available at a lower price and require custom made tools that can power the hardware longer. Questions in terms of ethics and sustainability are brought up and discussed due to the use of cameras to observe areas that can be considered private.

Keywords: Infrared Spectrum, Surveillance, Low light, Arduino, Raspberry Pi, Night vision, Infrared Camera.

Terminology

PWM: *Pulse Width Modulation*

CPU: *Central Processing Unit*

GPU: *Graphics Processing Unit*

GUI: *Graphics User Interface*

GPIO: *General Purpose Input/Output*

DC: *Direct Current*

PSU: *Power Supply Unit*

USB: *Universal Serial Bus*

CCTV: *Closed-Circuit Television*

Acknowledgements

This bachelor thesis is done in the department of computer science and engineering at Chalmers Univerisity of Technology which is done in 2021 and comprises of 15hp. We would like to thank Jonas Duregård for the opportunity to carry out the project at the department and the information provided.

We also thank our familiy and friends for the encouragement thoughout the process of the thesis and with the help of aquiring the necessary information needed to carry out the project.

Last but not least we thank our supervisor Sakib Sistek for the provided assitance, purchasing of components and for the exeptional support provided during the project.

Gabraeil Mamoush, Michael Evanson Ngibuini, Gothenburg, June 2021

Table of Contents

List of Figures

All figures in the thesis are taken by the authors.

1
Introduction

This introductory chapter will provide background about the project and its limitations. It also provides the purpose, research questions and aim in multiple sections that go into greater detail.

1.1 Background

Multiple mobile devices that can utilize the infrared spectrum to create images have been developed but are not affordable for the average consumer. The consumer has to find an alternative way of constructing the device themselves, but this requires a lot of knowledge to achieve the wanted result. Another drawback is that devices found in the market are not mobile and need constant power to operate. The user accessibility decreases as the device becomes more expensive.

The project intends to develop a concept of a system that the average consumer can use at home with affordable products that can be used and improved to a limit that won't impede the ability to interact with them. The project will mainly focus on hardware construction but will contain hardware that needs to be programmed. A kit containing parts will function as the base of the hardware construction to provide increased accessibility with other hardware. An Arduino and Raspberry Pi will be a part of the project to connect the various hardware used. Two cameras that use the infrared spectrum will capture footage where one shows the thermal radiation, and the other detects objects in low light conditions. The whole construction will move based on user commands.

1.2 Purpose

The purpose is to develop a concept of a robot moving when receiving commands from the user. The robot will be able to detect objects in low-light environments while measuring the infrared radiation from them.

1.3 Research questions

The following research questions will be answered in the thesis

- *How is the mobility and controls of the robot?*
- *How is the footage quality from the thermal camera and can it be improved?*
- *How will the low light camera be used to display the footage and can it be improved?*

1.4 Aim

The aim is to develop a concept of a robot capable of moving based on user commands while showing the thermal radiation from objects when the second one shows them in low light conditions. The robot's movements will receive instructions sent from the user allowing it to move.

1.5 Delimitations

The project will be delimited to the following:

- No new hardware will be made, only purchased and compatible with the Arduino or Raspberry Pi.
- The movements of the robot will be simple and depend on the kit.
- The cameras chosen are only compatible with the Arduino and Raspberry Pi.
- The kit used will be compatible with either a Raspberry or Arduino that allows to be powered with batteries

The construction will mostly occur on the provided kit that is compatible with the Arduino or Raspberry pi. The project will not develop new software but program existing hardware to function according to the set specifications.

2

Theory

In this chapter, we'll cover all of the necessary hardware and knowledge needed to understand the contents of the project.

2.1 Infrared

Infrared is a region in the electromagnetic spectrum containing waves transporting energy through space by disturbing the electric or magnetic fields. These waves all travel at the speed of light that is $2.998792458 * 10^8$ m/s in vacuum. These waves usually are characterized by frequency and wavelength. The electromagnetic spectrum is in multiple regions where each region contains boundaries defined by the wavelengths. The region used is the infrared region located between 0.7 - 300 μm [1].

The locations of the regions are not always precise due to the limitations in sources, detectors and amplifiers. The area where each of them lies is divided based on their wavelength. It's known that the lower end of the wavelengths lies just close to the visible limit. These subregions usually are divided by the wavelength they reside in [2].

These regions are in three groups. There is the near-infrared located approximately at the wavelengths 0.78 - 2.5 μm. The mid-infrared is near the wavelengths 2.5 - 25 μm while the far-infrared is located approximately at the wavelengths 25 - 1000 μm [3].

2.1.1 Applications of infrared

Infrared has many applications by using wavelengths to identify objects. In this case, the spectrum will detect objects with the help of different wavelengths located on the infrared spectrum. There are two types of ways this can work. One is by enhancing the image with the help of the near-infrared to make it visible. The second is by detecting the far-infrared portion commonly emitted through heat by objects [4].

The enhancement of an image works by gathering infrared light to amplify it for an image to be received. The instrument used is called an image-intensifier tube that handles the amplification and distribution of images. The detection works by first using a lens that focuses infrared light from the objects in its view and later scanned by a detector that makes a thermal gram. The pulses are then converted for the signal processor to create an image [4].

The common types of thermal imaging devices are cooled and uncooled. The cooled are more sensitive and easy to damage after use. They offer the best resolution and sensitivity in terms of images due to the cooling temperature being below 0°C. The result is that the cooling devices are capable of seeing differences of up to 0.1°C from a distance of 300m. The uncooled devices are cheaper and more common where the detector is stored in a compartment that allows it to operate at room temperature [4].

2.2 PWM (Pulse Width Modulation)

Pulse width modulation is a technique that creates a variable power supply by switching the voltage on and off faster than the load can detect. The result is a reduction of the average power delivered. PWM is perfect for shiftless loads such as motors and servos due to them being the least affected by the switching. Different PWM techniques are applied based on the application. The common ones are Multiple PWM, Sinusoidal PWM (SPWM), Trapezoidal PWM, Staircase PWM, 60° PWM and Third Harmonic PWM [5].

PWM presents some advantages and disadvantages. One advantage is the lower power dissipation due to the load not reacting to the discrete switching. Another advantage is that it's easy to implement due to higher-order harmonics being present and implementation in hardware due to the increased compatibility with modern microprocessors. The disadvantage is that high-frequency components can occur that can cause damage to the load and can cause attenuation in the hardware. Filters that reduce the harmonic distortion are added in the hardware as prevention [5].

One application for PWM is controlling motors such as DC motors. PWM provides the ability to control the DC motor and is far more effective at delivering power with minimal power dissipation. A common way that people often try to do is to connect a variable resistor in series with the motor to control it. The result causes a large power consumption [6].

The result is an increased power and heat loss at the resistor. PWM manages to solve this by giving the user the ability to control the pulse width. Since PWM changes between a high or low voltage average, the motor will use more power as the pulse width increases. The reason is due to the changing of the timing of the pulses known as duty cycles which increases the speed of the motor based on how long the average power is high. The same happens but in the opposite meaning that they will rotate slower the longer the average power is low [6].

The commonly used duty cycles often provide sufficient power to run loads while reducing the power consumption. Typical duty cycles used are 25% and 50% that provide easy modification while requiring low maintenance [6]. The Arduino and Raspberry are both capable of creating PWM signals to power the motors. The waves of the duty cycles with the analogue value are in figure 2.1.

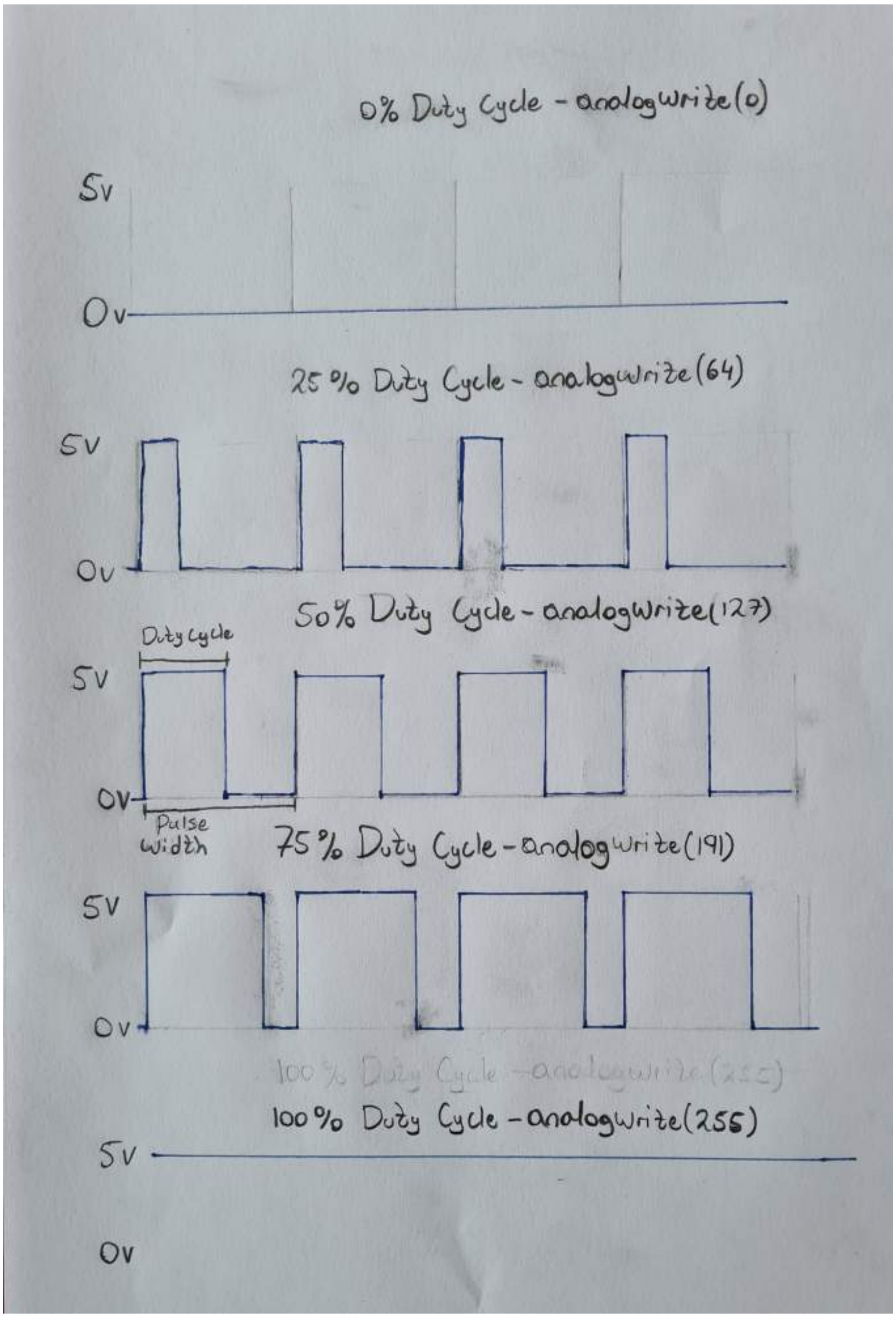


Figure 2.1: *The commonly used duty cycles with PWM.*

2.3 Hardware

The hardware used in the project will require knowledge of the functions and required configuration. The section below aims to provide an introduction to the hardware used with features of each hardware.

2.3.1 Raspberry Pi 4 model B

The Raspberry Pi 4 Model B is a single board computer that carries essential circuits such as the CPU, GPU and others that handle the communication between the ports. The components make the Raspberry Pi a small computer that is capable of many things. There is also a GPIO that helps to communicate with hardware that can receive signals from the raspberry pi [7]. Raspberry also supports peripherals through USB for keyboards or mice, HDMI for screens and an ethernet cable to connect to the internet. It provides multiple purposes such as operating a coffee maker or even function as a Web server [8].

The Raspberry Pi can also function as a controller for projects such as drones or robots. The Raspberry can send signals to servos and motors based on the user programming. The user also can design a GUI controlling of the motor or servo [10]. The low cost of the Raspberry Pi has made it easier to build simple constructions that allow easy integration with cheap components that will bring down the total cost of the project [7].

The Pi developed by the Raspberry Pi Foundation aims to increase the computing power available to people. The foundation has produced many models of the Raspberry Pi that have become even cheaper and able to be used in more complex projects [9].

In comparison to boards such as the Arduino Uno, the Raspberry Pi 4 Model B can control motors and servos. With the help of the onboard operating system, it is capable of monitoring and running different things without any problems [10]. The model used in the project contains 2 GB Ram and an ARM quad-core cortex CPU with a clock speed of 1.5 GHz. The Pi also comes equipped with Bluetooth together with Wi-fi [11]. Figure 2.2 shows the Raspberry pi with the specs mentioned above.

Figure 2.2: *Raspberry Pi 4 Model B to be used in the project*

The important part to note is the Raspberry Pi can run off a PSU or a USB with the recommended current supply. The model used can be powered with a PSU capable of delivering a current of 3.0 A. It can also be powered from the USB port if 1.2 A is the current. The lack of sufficient current can cause problems due to the board not getting enough power which can corrupt the SD card [12].

2.3.2 Arduino Uno R2 Wi-fi

The Arduino Uno R2 Wi-fi is a microcontroller-based board capable of being programmed by the user to perform tasks. Most common appliances such as a microwave or washing machine contain a microcontroller that controls them. The Arduino Uno can perform many different functions with the help of open-source tools broken down into multiple packages that are easy to access. The Arduino Uno is easy to program thanks to the IDE(Integrated Development Environment) and the programming language resembling C/C++. The result is an IDE that simplifies the learning and programming of the Arduino [13]. The Arduino used in the project has an integrated wi-fi module to ease connectivity. The board used is seen in figure 2.3.

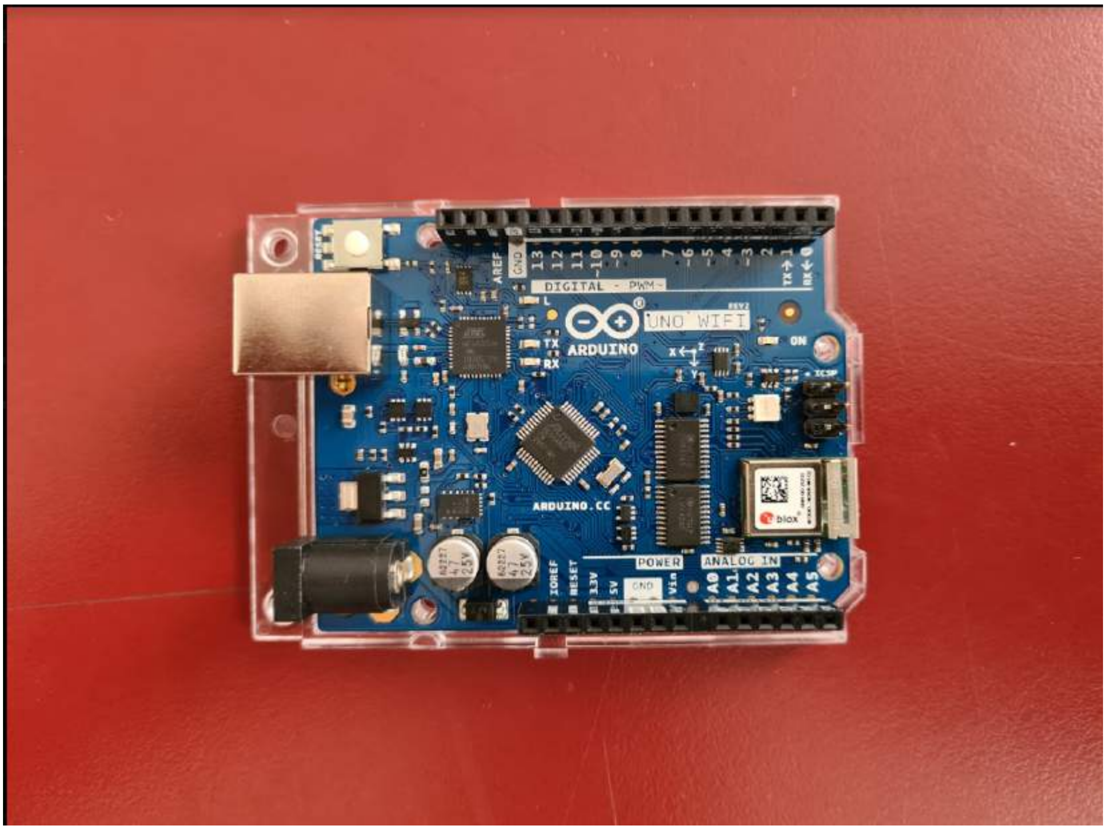


Figure 2.3: *Arduino board to be used in the project*

Due to the Arduino Uno being a simple computer, it's often compared to the Raspberry Pi. The difference between the two is the Arduino Uno contains an ATmega 328p with a preloaded bootloader that enables the ability to program it directly without the help of an external hardware programmer [14]. The Arduino does this with an IDE that compiles and programs the Arduino. Multiple add-ons increase the capability thanks to the open-source libraries and hardware developed by the community. A similarity to the Raspberry, the Arduino can control motors and servos with the help of additional hardware. The added hardware makes it easy to construct more complex projects by adding components that support expansion [13].

Compared to the Raspberry Pi, the differences are subtle but can be a factor depending on the application. The points below are the differences between the two:

- The microcontroller on the Arduino contains the CPU, RAM and ROM. The rest of the hardware seen on the board takes care of the connectivity and programming. The Raspberry instead has individual chips that are the graphics driver, processor, memory and storage [15].
- The Raspberry Pi 4 Model B requires an operating system to operate due to the separation of the processor, memory and storage. The Arduino only needs a file containing the compiled source code to be functional [15].
- The Arduino Uno has a maximum clock speed of 16 MHz when the Raspberry Pi can achieve a clock speed of 1.5 GHz. The higher clock speed gives room for running more strenuous applications [15].
- The open source hardware and software files of the Arduino provides the possibility of creating your own Arduino board. The same is not possible with the Raspberry Pi due to the files not being open source. [15].

The points mentioned above are the significant factors that impact the user when choosing aboard. They are other subtle differences still the impactful ones are listed above [15].

2.3.3 Freenove 4WD Smart car kit

The Freenove 4WD Smart Car kit is a kit that allows easy programming and contains tools that support multiple capabilities such as face tracking, line tracking, obstacle Avoidance, Colorful light and can even function as a wireless RC car with a camera. The kit comes with all of the components that support the functions mentioned [16].

The kit comes from the Freenove company that produces kits that are easy to program and build with the help of the provided tutorial that comes with each kit [17]. The hardware used in the project above comes with four DC motors that allow the user to control them individually, a camera compatible with the Raspberry Pi to record videos and take pictures, two servos that control the camera and two IR sensors [18].

Compared to the other kits that Freenove provides, the 4WD kit is easy to build and use due to the components being compatible with other microcontrollers or SBCs. The kit doesn't include a Raspberry Pi and requires two 18650 3.7V to provide power to the Raspberry Pi. The other requirement is that the capacity of the batteries have to be between 2000 - 3500 mAh for the kit to function [19].

2.3.4 Pi NoIR Camera V2

The Raspberry Pi NoIR Camera V2 is a camera that can interface with the Raspberry. It is capable of taking pictures and record video that is displayed directly on the Raspberry desktop. The camera takes and stores footage in the Raspberry if needed. It offers the same function as the regular camera but allows taking pictures in the dark with the help of a few infrared LEDs [20].

The NoIR V2 is a popular camera among various hobbyists. The reason is that it can help monitor nocturnal instances without disturbance. The camera is compatible with models 1, 2, 3 and 4. Various libraries allow users to access the camera with the help of python, one of them being the Picamera library which is popular [20].

The NoIR V2 camera contains an 8 megapixel Sony IMX219 sensor custom-designed for the Raspberry Pi. Thanks to the sensor, the NoIR v2 features a focus lens capable of resolutions of 3280 x 2464 for static images and supports video footage from 480p90 up to 1080p30. The advantage is that it is relatively small in size and weighs less than 5 grams. The camera is perfect for multiple applications such as a CCTV camera, taking low light photos and monitoring things such as plants. The camera has no infrared filter that makes it suitable for taking infrared pictures [21].

2.3.5 AMG8833 Featherwing IR thermal Camera

The AMG8833 Featherwing is an 8x8 thermal camera compatible with the majority of Microcontrollers. It features the I2C protocol that helps it to communicate with controllers such as the Arduino or Raspberry Pi. The Sensors located on the camera are from Panasonic that contains multiple sensors forming an 8x8 array where each sensor measures the temperature individually with the help of an internal thermistor. The Featherwing is an uncooled IR camera capable of operating at room temperature [22].

The protocol I2C used to communicate with the chip was created by Philips in 1982 to support their chips. The first version of the I2C could support speeds of 100 kHz with 7-bit addresses. The protocol has developed to provide frequencies of up to 400 kHz and 10-bit addresses. The current versions of the Arduino and Raspberry support the 400 kHz frequency with 10-bit addresses [23].

The AMG8833 can show temperature detection in the form of a two-dimensional array that contains 64 pixels. The sensor can lie in various home appliances such as microwaves, air conditioners, lighting controls and also automatic doors [24]. The AMG8833 is a good camera due to the aspects below

- Since the Operating Voltage is 3.3V DC, the camera can receive power from either the Arduino or Raspberry pi. [24].

- The Current consumption is 4.5 mA during operation and 0.2 mA during sleep mode. The consumption makes it efficient to use with the Arduino or Raspberry Pi. [24].

- The viewing angle is 60° with an optical axis gap of $+-5.60°$ [24].

- The range of Human detection distance is 7m with a maximum refresh rate of 10 Hz [24].

The camera is capable of measuring temperatures within a range of $0°C$ up to $80°C$ with an accuracy of $+-2.5°C$. The Featherwing also supports the possibility of interpolation to increase the resolution from the camera. In addition to the interpolation function, there is an interrupt pin that activates if the user wants a certain threshold to be passed or not [22].

2.3.6 L298N H-Bridge Motor Driver

The L298N is a dual H-bridge motor driver that allows speed and directional control of two motors. The L298N is a popular motor driver used to control motors on the Arduino. This motor driver is also compatible with the Raspberry Pi, which makes it suitable to control motors on either platform [25]. The described motor driver is in figure 2.4

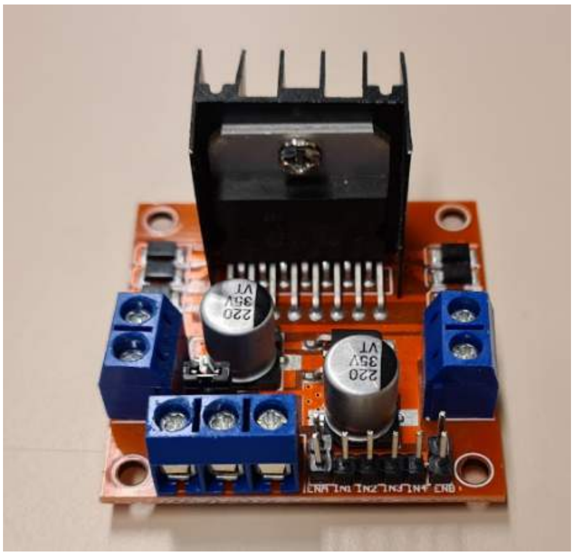


Figure 2.4: *One of the L298N to be used in the project*

The L298N consists of multiple parts used to help with the controlling of motors. The components each have a dedicated function that allows the whole motor driver to be compatible without damaging the other parts. These parts are:

- L298N Motor Driver IC that helps with the logic of controlling the motors and driving them [26].
- Power Supply which feeds the IC and motors with power. It consists of pins that supply the 5V logic together with the motor [26].
- Output Pins that allow the connection of the DC motors and supply them with power. These pins are located on the sides of the L298N [26].
- The control pins handling the direction of where the motor rotates [26].
- The speed control pins handle how fast the motors will spin. The pins are capable of sending an ON and OFF signal for the motors. [26].

The L298N provides an easy way to control motors on multiple platforms due to the ability to handle the connection. The result is an easy way to send signals between the platform and motors.

3 Method

The chapter will cover the materials used in the project and present the procedure. Materials used in the project are in the form of a list is in the section as well.

3.1 Materials

The materials used in the project were chosen based on time and cost impact. It was decided that a kit with complete materials and electronics were suitable due to the cost impact and would reduce the overall time of sourcing materials. This also meant that a control unit was required if the motors were to function as intended. The list below contains the components used in the project

- Freenove 4WD smart car kit
  - 4 hobby gear motors and 4 wheels
  - Mounting tools for the Camera and the Raspberry Pi
  - PCB to mount the Camera, Raspberry Pi, Motors and Batteries
- Arduino Uno R3 Wi-fi
- Raspberry Pi 4 model B
- Pi NoIR Camera V2
- AMG8833 Featherwing IR Thermal Camera
- 2 L298N H bridge motor drivers
- 2 18650 Samsung Rechargeable Batteries
- 2 9V Battery and 2 9V Battery contact.
- 2 remote controls containing IR leds
- 1 Power Bank with 5V USB output

The materials are chosen based on the previous expertise gained through the years and components that are compatible and easy to integrate.

3.2 Procedure

The procedure taken during the project was motivated by prior experience in term of dividing the workload into smaller parts. Each part would focus on a component where the goal was to understand and implement each component function and look for ways to integrate it into the final construction. The result is decreased time in debugging due to the extensive tests taken to ensure that the component is functioning. Each section is dedicated to specific hardware used in the project and brings examples of tests performed to achieve a suitable result used to perform measurements.

When specific hardware arrived, it was analyzed to see its function and implemented before adding it to the final construction. The chance of detecting non-functioning hardware increased due to the required task tested on its own. It took less time to assemble the hardware due to the components being compatible from the beginning. The complete system was tested by evaluating the function of each element in the final construction. Testing was done in three stages, the movement, the quality from the infrared camera and the NoIR camera in low light conditions. The goal of testing was to see the performance at each stage based on the implemented function.

4 Implementation

This chapter will cover the hardware function and the result of the implementation. Each section of the chapter will present the steps taken to program and tests performed to optimize each component to get the best performance possible. The last section will cover how all of the hardware is connected to form a fully functioning unit.

4.1 Tested setups

Upon receiving the components, the exploration of setups occurred to find the best one that provided the most convenient way to put everything together. The result found was about how the cameras, Arduino, Raspberry Pi and the freenove kit would function together as a unit. The setups tested were as follows.

- IR camera was to send data through the Arduino to the Raspberry Pi with a serial connection.
- The control unit and NoIR camera were implemented in the Raspberry Pi while the Arduino takes care of the IR camera.
- The Arduino functioning as the control unit while the Raspberry takes care of the NoIR and infrared camera.

The points mentioned are the multiple ways the system would come together. The chosen setup was with the Arduino functioning as the control unit while the Raspberry took care of the IR and NoIR camera.

4.2 Connecting the IR Camera to the Raspberry Pi

Due to the Arduino functioning as a control unit for the robot, the IR camera became connected to the Raspberry Pi. The I2C protocol functioned as a medium of communication between them. The initial step taken was to connect the IR camera to the Raspberry pi.

The pi supports the I2C protocol and providing support for up to 7-bit addresses. The camera connects to the pi according to the pins corresponding to which signal. The Raspberry would power and detect the camera. Four wires connecting the camera create a connection to the Raspberry Pi. The highest black wire connects to the SDA pin on the IR camera to the Raspberry. The lower black wire connects the SCL pin to the respective pin on the pi. The blue wires connected to the IR camera where the highest one is the voltage supply for the camera is 5V while the lower one is the ground wire. Figure 4.1 shows the pins connected to the Raspberry Pi and camera that support I2C.

Figure 4.1: *Connection of I2C on Pi (Left) and Pins on the camera (Right).*

After connecting the camera to the Raspberry Pi, The other step was to control if the Raspberry could detect the camera. The I2C address required to communicate with the camera is on the PCB of the Featherwing. It is in figure 4.3 on the right as "open 0x69". This address is required to be the same inside the Raspberry to access the camera. The camera required libraries installed and a higher frequency to provide stable footage meant using commands to open the settings to implement the changes. Figure 4.2 illustrates commands used and the verification that the detected address was correct.

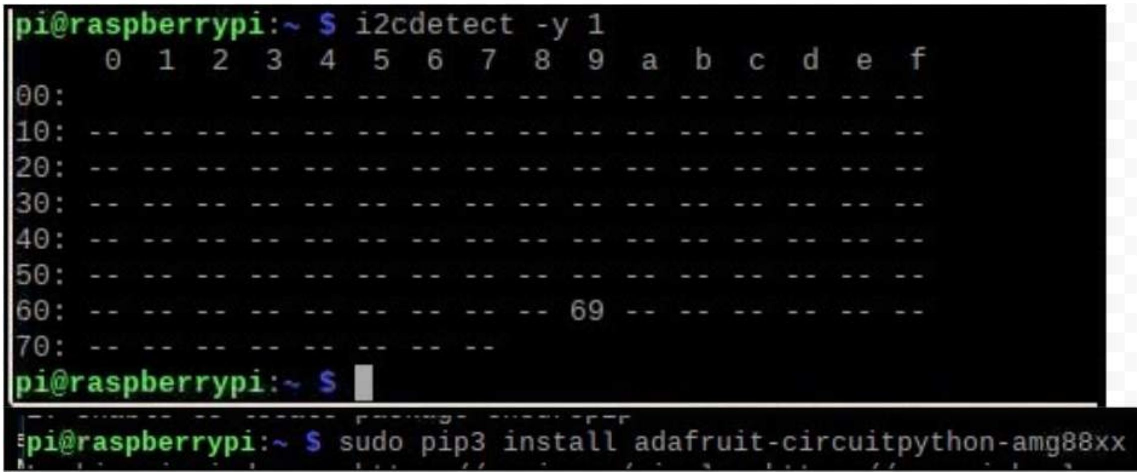


Figure 4.2: *I2C address in Raspberry Pi (Above) and the command to install the library for the Camera (below)*

As seen above, the Raspberry Pi was able to detect the camera without any problems because the address shown in the Raspberry Pi was the same as located on the PCB of the Featherwing. It indicated that the Raspberry could access the camera with the I2C address required to open it. The frequency was changed to 400 kHz to run the camera at the highest sampling rate possible. The frequency change in the config file was done by writing in the command line *sudo nano /boot/config.txt* to access the config file and changing the line to *dtparam=i2c_arm=on,i2c_arm_bauderate=400000*. The frequency change will ensure that the AMG8833 will run at its maximum sampling rate that is 10Hz. The Raspberry was rebooted for the change to occur.

With the above steps done, the next step was to program the camera to display the data seen when testing the camera in the Arduino. During the implementation, a python library called *matplotlib.pyplot* was used to plot the values with the help of a colour scale to see which parts are warmer than the others.

The commands inside the matplotlib.pyplot gave the possibility to chose colours from a list provided by the library. The name is required to be in the function parameter of *plot.imshow*, this function has multiple parameters where one of them is the colour used to plot the data. The chosen colour would then associate with each value in the data, thus creating an image using the colours. Figure 4.3 shows the result of the initial test with the commands in the library.

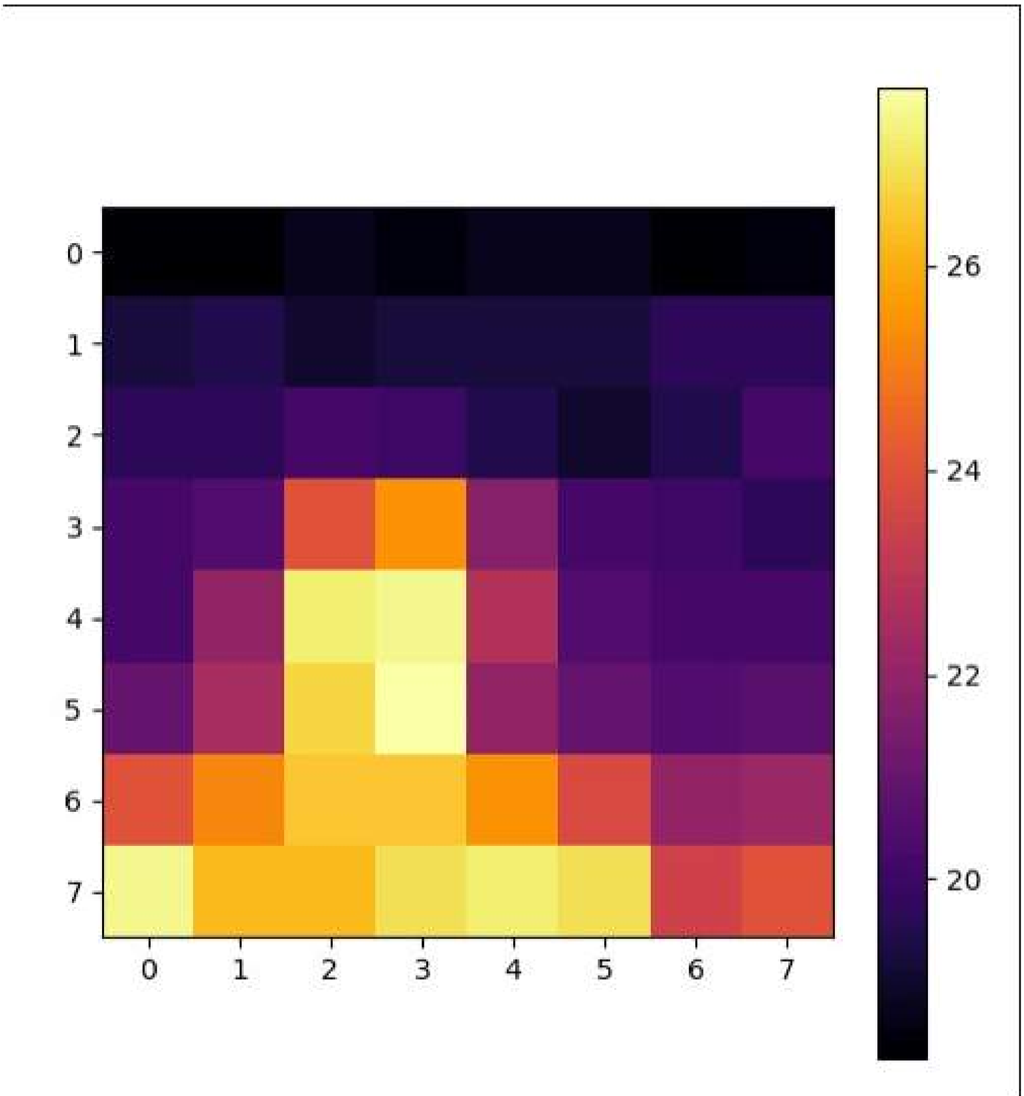


Figure 4.3: *Photograph of a person shown as the result of the plotted data with the help of the matplotlib.pyplot library.*

The image initially produced was grainy due to the pixels being visible. The issue was that the image created directly from the values from the sensor gives out results with image distortion. Improvement to the image quality was performed by interpolating the values from the camera.

With the help of the matplotlib.pyplot library, multiple interpolation methods available from a list provided by the matplotlib.pyplot library acted as a way to see which method provided the optimal result. The chosen interpolation method was hamming made it easier to distinguish the same image in figure 4.3. Figure 4.4 shows the outcome of the interpolation method on the same image.

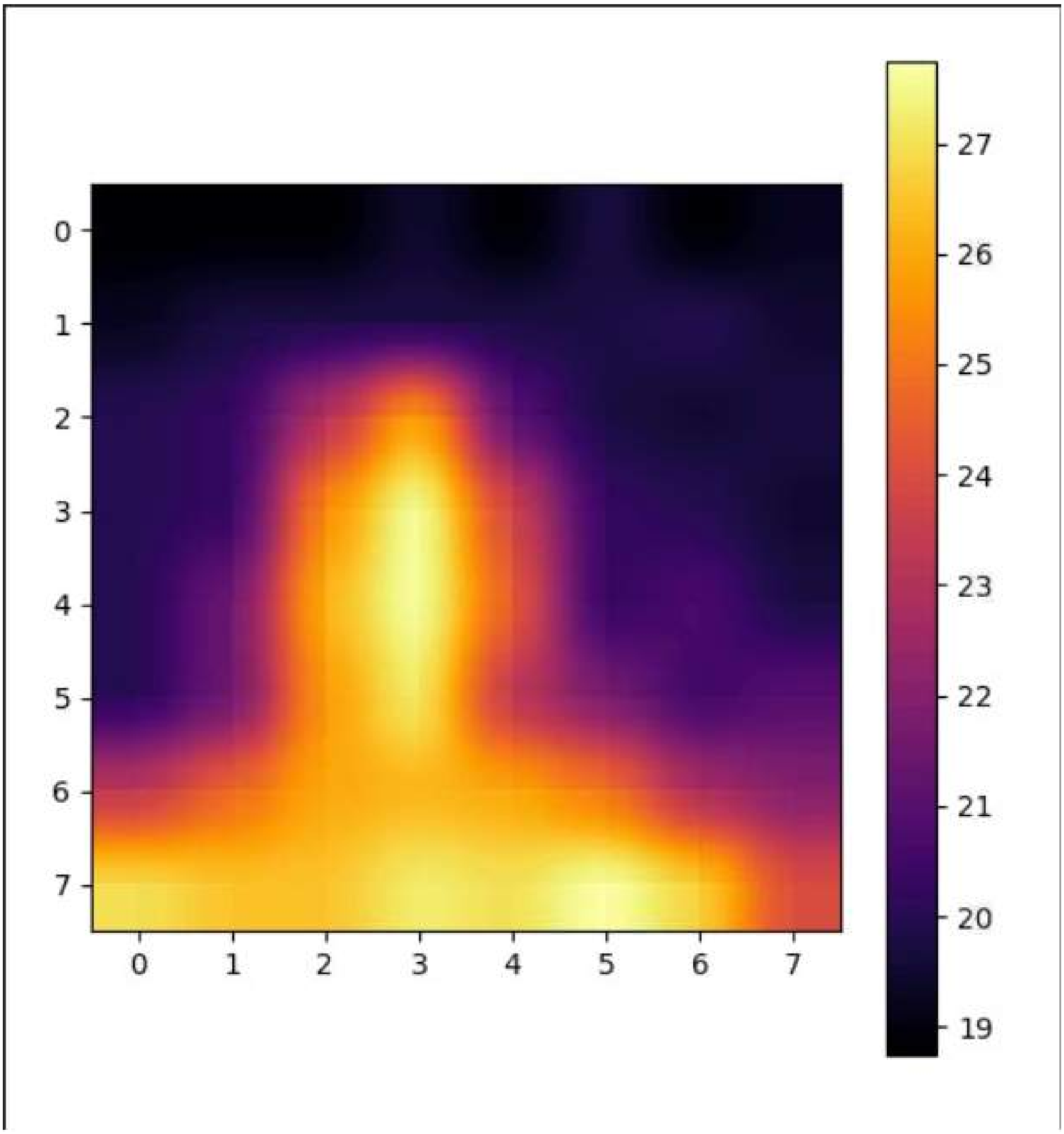


Figure 4.4: *Hamming result on the same observed image*

The difference here is that the observed figure is more defined in terms of lines to distinguish what the camera sees. Compared to the original image, the interpolated image using the hamming method with the help of the matplotlib.pyplot commands made the picture clear and easy to distinguish.

4.3 Programming the NoIR V2 Camera

With the addition of the IR camera to the Raspberry Pi, the NoIR v2 camera tests confirmed that it would function as intended and learn its capabilities to ease the implementation. The camera was connected to the camera to the Raspberry Pi module and is in figure 4.5.

Figure 4.5: *Connection of the Camera to the Raspberry Pi*

Since the Raspberry Pi contains a dedicated module that supports the addition of the camera, the connection made without any more problems. The camera was later tested by writing simple commands that would allow it to either take a picture or save a video. The first command used was *raspistill -o test1.jpg*. It took a picture and stored it in the home folder with the name given. The reason it became stored in the home folder was that there was no specified location. In this case, the image was called test1.

The second command was *raspistill -awb off -awbg 1.3, 1.2 -o test.jpg*. The instruction above took a picture with the AWB (Auto White balance) with preset values. It required turning off the AWB first, and the command also saved the image with the name test.

The last command was *raspivid -o video.h264 -t 5000* took a video with the H.264 codec with the name video. The *-t* is a flag that indicated the length of the video in milliseconds. In this case, the command sent took a 5-second video and saved it in the home folder.

The progress made was a good starting point due to the commands providing a function to see the ways needed to access the camera with the footage from it. Lines written in python helped monitor the entrance of the camera through the GUI. Adjustments made because the initial photos taken were discoloured but changing the AWB made it possible to get better images. Figure 4.6 illustrates the pictures taken with and without the AWB applied.

Figure 4.6: *Before adjusting the AWB (Left) and After adjusting the AWB (Right)*

As seen in figure 4.6, the changes to the AWB affect the lighting caused objects to appear in other colours than they are. Changing the gains to 1.3, 1.2 with the help of the command *awb_gains* sets the auto white balance gains for the camera. The setting for the AWB was set to auto to let the camera decide the AWB by itself.

After testing, early establishment indicated that the camera could either take a photo once it reaches the object or constantly streams the footage. With the help of the OpenCV library, the live video capture from the camera was possible just by following the many guides available online [27]. The guide gave a clear description of the necessary steps needed for OpenCV to access the camera for the live capture. The part that caused problems was the compilation process for the libraries required to run the python script.

The compilation took more than 1 hour due to the large size of the required python libraries. The reason is that the libraries are large that meant more RAM. To compile the libraries, the swap space needed to increase because the physical RAM wasn't enough. The swap space increased the systems accessible memory beyond the hardware capabilities.

To increase the swap size, the command *sudo nano /etc/dphys-swapfile* accesses the swap file. The line *CONF_SWAPSIZE=100* was changed to *CONF_SWAPSIZE=2048* to allow the Raspberry to use more of the swap space. The command *sudo systemctl restart dphys-swapfile* regenerated the swap file for the change to occur. The result is the remaining volume on the SD card is acting as memory. Figure 4.7 shows one of the significant parts of the installation process.

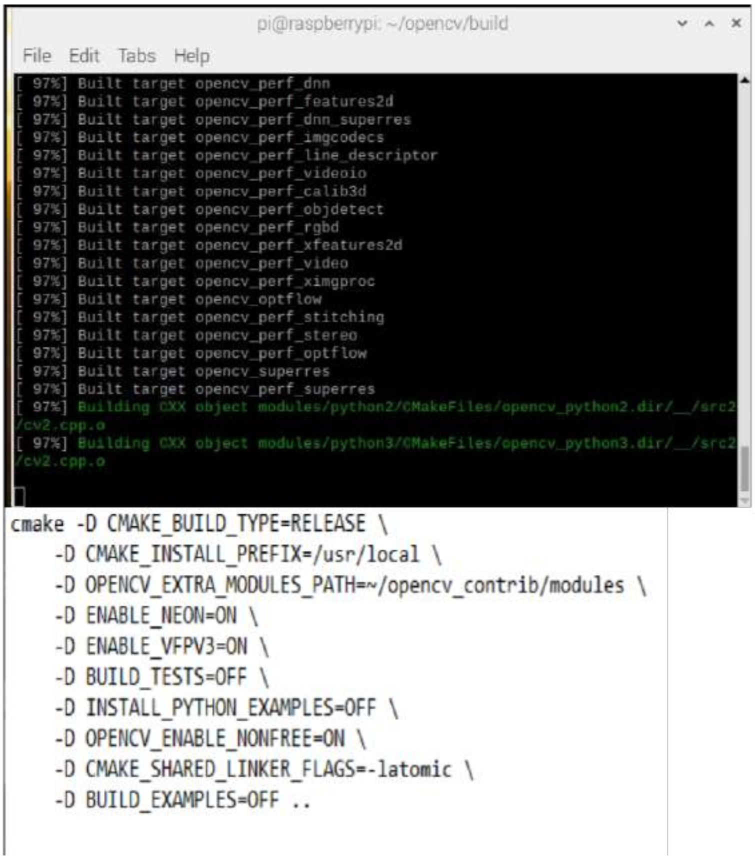


Figure 4.7: *Compilation of the large python libraries that support open cv*

After all of the necessary libraries were installed and compiled, code in python used to test the live capture feature with the help of various online tutorials. The code in appendix A provided a way to test the capabilities with the help of the open cv library and implement the final function. The code captures what the camera sees and develops the footage frame by frame with the method *capture_continuous*.

The function captures each frame and stores them in an array to use the data to display the image. The data is to a method in the open cv library called *cv2.imshow* to display the data inside the array in a frame. The OpenCV library also provided the option to close the window with the function *cv2.waitkey*. The function *capture.truncate(0)* clears the current frame and allowing for the next one to be displayed.

The code *cv2.waitkey(1) & 0xFF* works as a way to detect if the user has pressed the required key to close the window. The result is a stream from the camera that can shut down by pressing the required key. The code in appendix A managed to produce a stable stream of footage from the camera without any problems. Figure 4.8 shows the result based on the initial tests with a lower resolution.

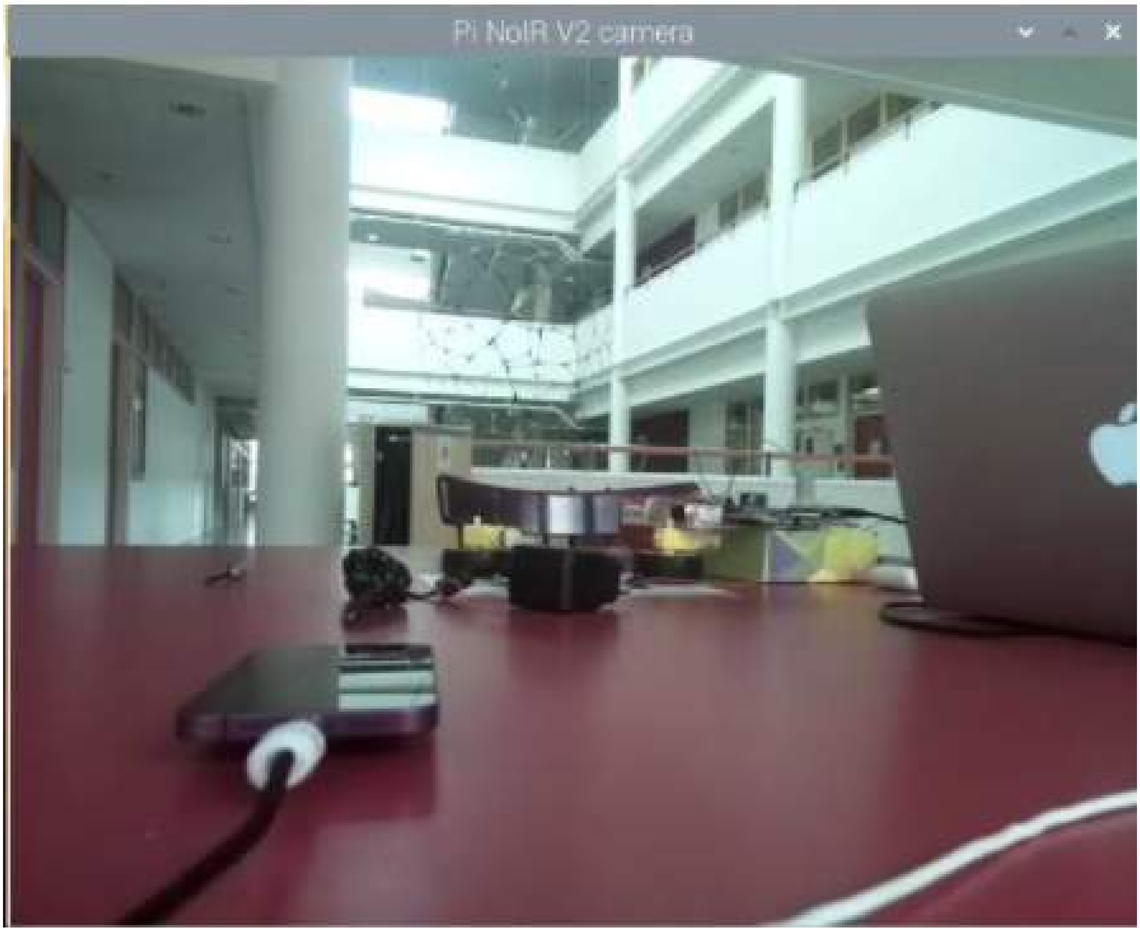


Figure 4.8: *Picture taken from the live capture of the camera in 480p*

As seen, the python code can interface the camera without any problems and is capable of displaying the footage on the window. The resolution shown in figure 4.10 is at 480p is clear enough to produce an image. The Pi NoIR v2 can support a resolution of up to 3280 x 2464 for pictures and 1080p30 for video.

The resolution was increased to the maximum supported by the camera in order to see if the video would provide a stable experience with the increased resolution. Figure 4.9 contains a snippet of the same observed instance but with a higher resolution and increased frames.

Figure 4.9: *Snippet taken from the 1080p30 footage from the camera*

As seen, the quality is higher, but the viewing angle is smaller. The reason is that resolution increased which made the viewing angle smaller. After testing multiple times, the chosen video resolution was 480p60 due to the viewing angle being less affected and allowing the window of the thermal camera to be visible at the same time.

Tests regarding the night vision capabilities were not as thorough compared to the other processes. Capabilities observed during the testing revealed odd-looking colours in the former figures. An example is in figure 4.9, the colour of the chair appears purple while it is black in reality. It proved that the camera could function in low light conditions if an infrared light source illuminates the subject. Figure 4.10 illustrates the assumption made above.

Figure 4.10: *IR leds being illuminated directly on the NoIR camera*

Figure 4.10 confirms the phenomenon noted in figure 4.9. The camera can see in low light environments even with a live capture function. The tests showed that the camera required illumination of the object with an IR light source to see it. Figure 4.10 shows that the camera can see the IR light and will be able to see an object should it be illuminated by the same source.

4.4 Building the Freenove 4WD Smart car Kit

The Freenove kit contained items that were also compatible with the Raspberry and Arduino. All of the included hardware was mounted to make sure that none was left out. It became easier to build and change connections depending on how the other components would accommodate the integration. The hardware was sorted based on the provided information from the manufacturer's tutorial. The result is that all of the required hardware that came with the kit was available and not lost. After sorting the hardware, the four motors were mounted with the provided brackets on the plate. The wheels were mounted on each side, and that made the construction stand on its own. Figure 4.11 shows the kit with the attached motors and wheels to the mounting plate.

Figure 4.11: *Motors and Wheels mounted on the included baseplate*

After installing the wheels, the line tracking became installed underneath the car with the provided standoffs. The connection of the line tracking module occurs with a pin cable attached to the base near the motor. After adding the line tracker, the next step was to install the Raspberry Pi used in the project. The implementation occurred with the help of the included standoffs to make sure that the Pi would sit securely when the car would move. A provided connection board in the kit connects the Pi with the base to power the Pi from the batteries.

With the motors and the included line tracker added, the module that supported the pan and tilt movements became added in conjunction with the NoIR camera and the IR sensor. The above ensured that the camera would be stable while capturing the footage. The mounting of the IR sensor and camera needed servos included in the kit that functioned as a link to the provided baseplate. The servos used worked as a way to mount the module containing the camera. Figure 4.12 illustrates how the kit looked like when adding the camera.

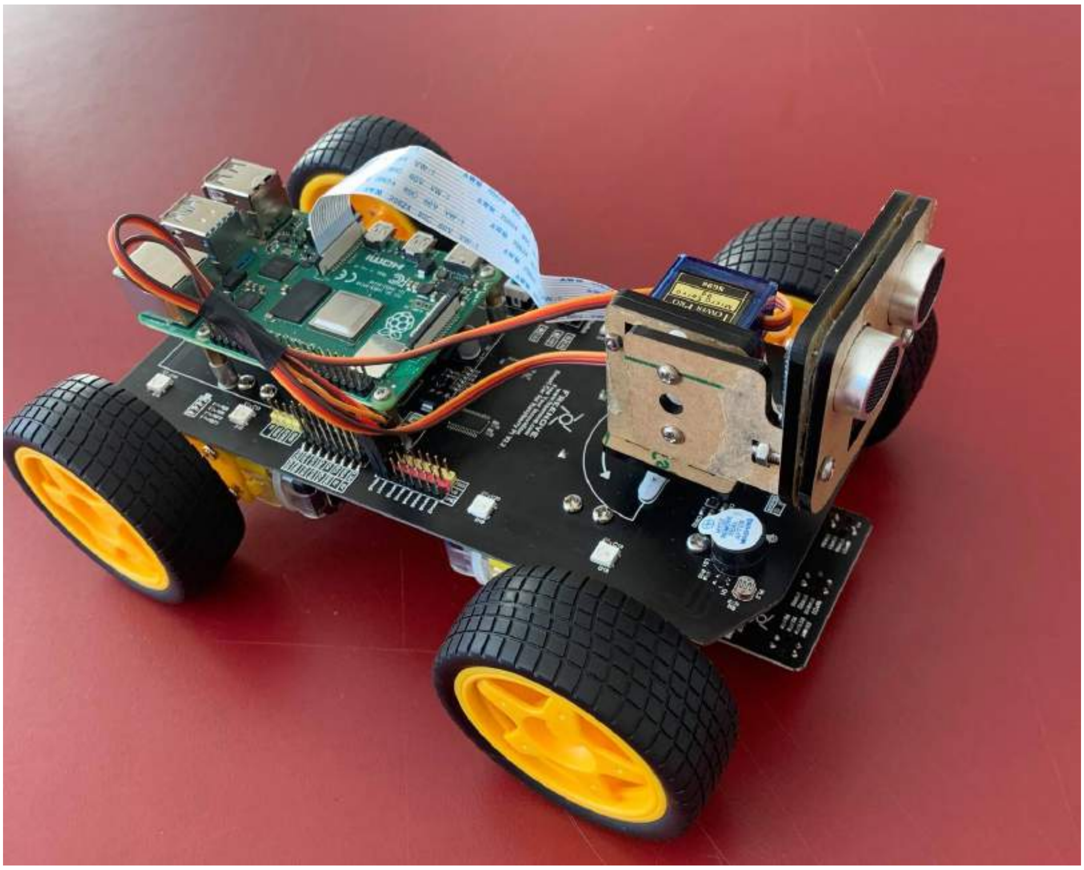


Figure 4.12: *The kit with the camera module attached*

The last step was to add the connection board that would ensure that the Raspberry Pi could receive power with the help of the batteries that made it more mobile. The connection board became modified to accommodate the IR camera. That meant wires required to be soldered to the respective connections to improve the accessibility of the IR camera. Figure 4.13 shows the connection board before modification.

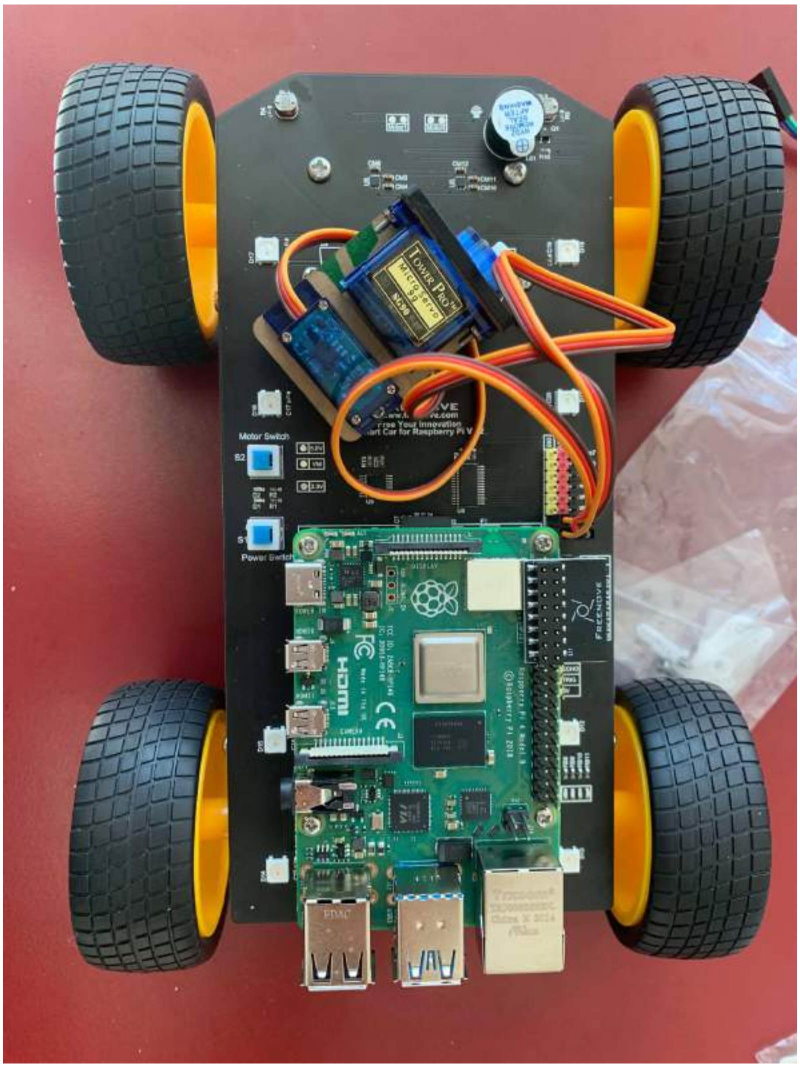


Figure 4.13: *Connection board on the Raspberry Pi before modification*

While testing the connection, the significant part was that the Raspberry Pi would receive power upon installing the batteries into the kit. The connection board was a bridge between the pins of the Pi and the batteries. The Raspberry Pi was able to receive power from the batteries and managed to function.

4.5 Building the control unit with the Arduino

The hobby gear motors provided with the freenove kit could move based on commands from the Arduino and the help of two L298N drivers that will give 4WD capabilities with the PWM signals from the Arduino. The first step was to simulate the motors to ensure that time spent debugging and checking connections became minimal. Tinkercad provided a simple simulation where the hobby gear motors accepted commands with the help of buttons located on a breadboard. The breadboard formed a bridge that simplified the connections between the different components. Figure 4.14 contains the simulated circuit.

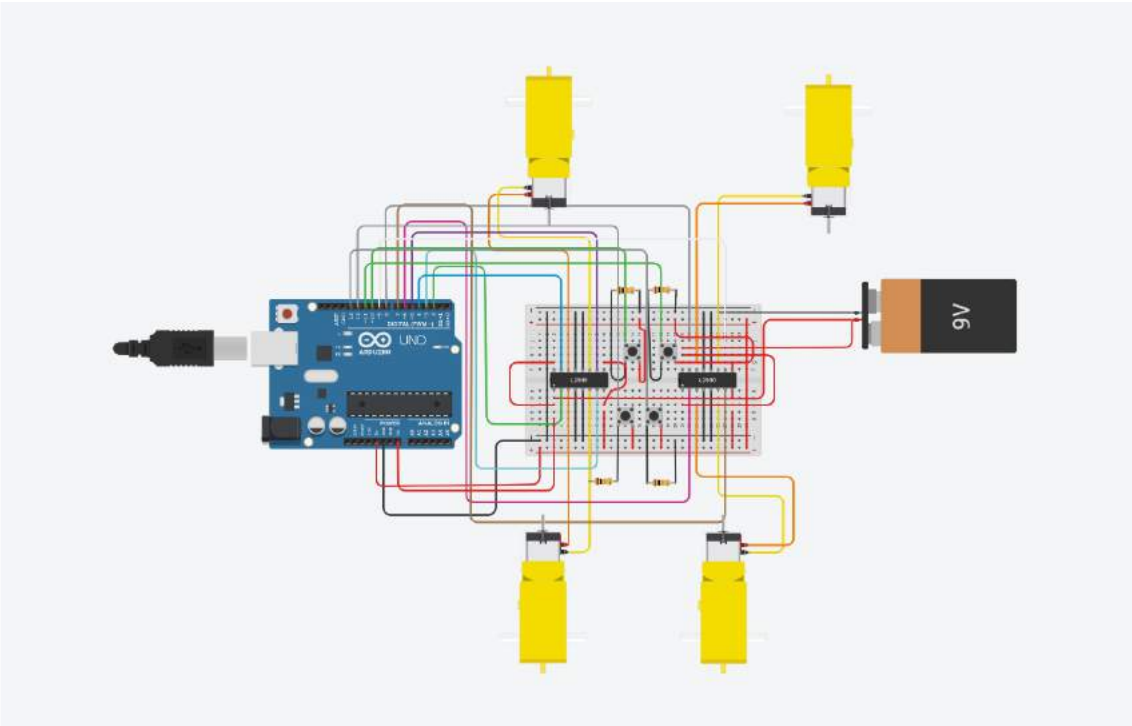


Figure 4.14: *A simulation of the circuit used to control the motors built in tinkercad to test the movement of the motors.*

The circuit shown in figure 4.14 functioned as a guide on how to connect the motors. It provided a way to check the pins for the motor drivers, Arduino, and the battery if they were connected correctly. Figure 4.17 also shows how the breadboard would work as a bridge connecting the other components. The buttons worked as a medium to test the directional control. After simulating and confirming that the code was working, the next step was to see if the hobby gear motors would accept commands with the help of an access point sending signals to the Arduino from the user. The webpage would show present buttons where the user can press to dictate the direction of the car.

When looking for libraries, a library called WIFININA provided examples of code capable of controlling hardware by creating an access point. The user will connect to the access point by adding the chosen network name and password. The access point created a way to send commands to the motors without interfering with the connections. A simple web page created with the help of the Arduino library made the whole process easier than expected. The result of the site is in figure 4.15.

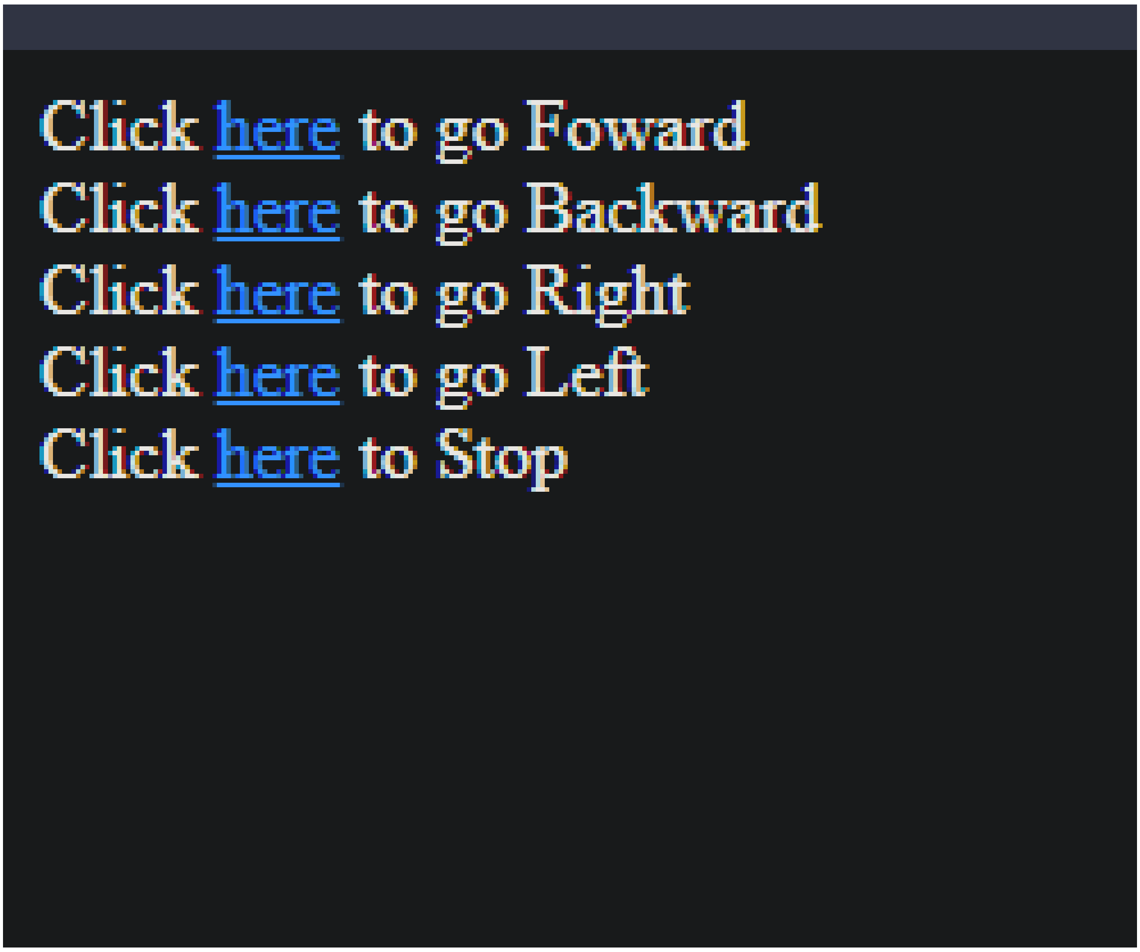


Figure 4.15: *Web page containing signals that sends signals to the Arduino.*

The created page sends commands from the user's device to the Arduino with the help of the created access point. The user's device has to connect to the created access point to access the web page. The Arduino itself was able to accept commands from the page with ease, and the motors were able to be powered by a 9V battery which was able to make the whole kit move based on the user's wishes.

4.6 Assembling everything together

With the individual hardware programmed capable of functioning alone, the last step included assembling everything to test that they worked together. The first step was to find a way to connect the control unit with the Arduino. The 3D printed mounting brackets adding the possibility of creating a connection to the Arduino with the rest of the kit. The addition made it possible of controlling the motors without any problems. Figure 4.16 shows the pieces used to add the Arduino and motor drivers to the freenove kit.

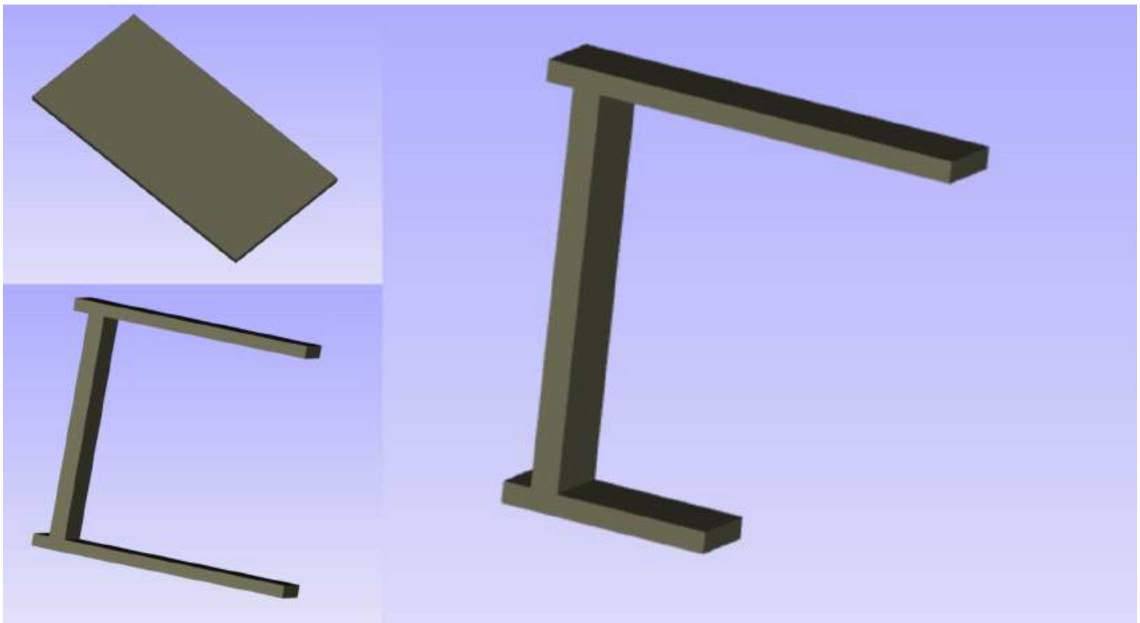


Figure 4.16: *3D printed Mounting brackets for the Control Unit.*

These ensured that the Arduino would be able to control the motors located on the kit. A small breadboard was used to connect the ground wires together in order to make sure that the connections were correct and would not cause any problems. The 3D pieces were mounted by drilling holes that provided the ability to place the brackets on the standoffs provided by the freenove kit.

When added to the construction, the 3D brackets provided support for the motor drivers, Arduino and breadboard without any problems. They managed to not interfere with the Raspberry Pi on the 4WD kit and remained stable while moving. They also didn't interfere with the onboard cameras while recording and provided access to the connection board.

The motor drivers, breadboard and Arduino are located above the Raspberry and can provide sufficient power to the wheels. The cameras can record due to being placed in front of the 3D bracket holding the Arduino, breadboard and drivers. The placement of the NoIR camera was suitable to capture footage at the same time as the IR camera. The 3D brackets provided a solution that resulted in a working unit that functions with minimal interference. Figure 4.17 shows the mounting on the final construction.

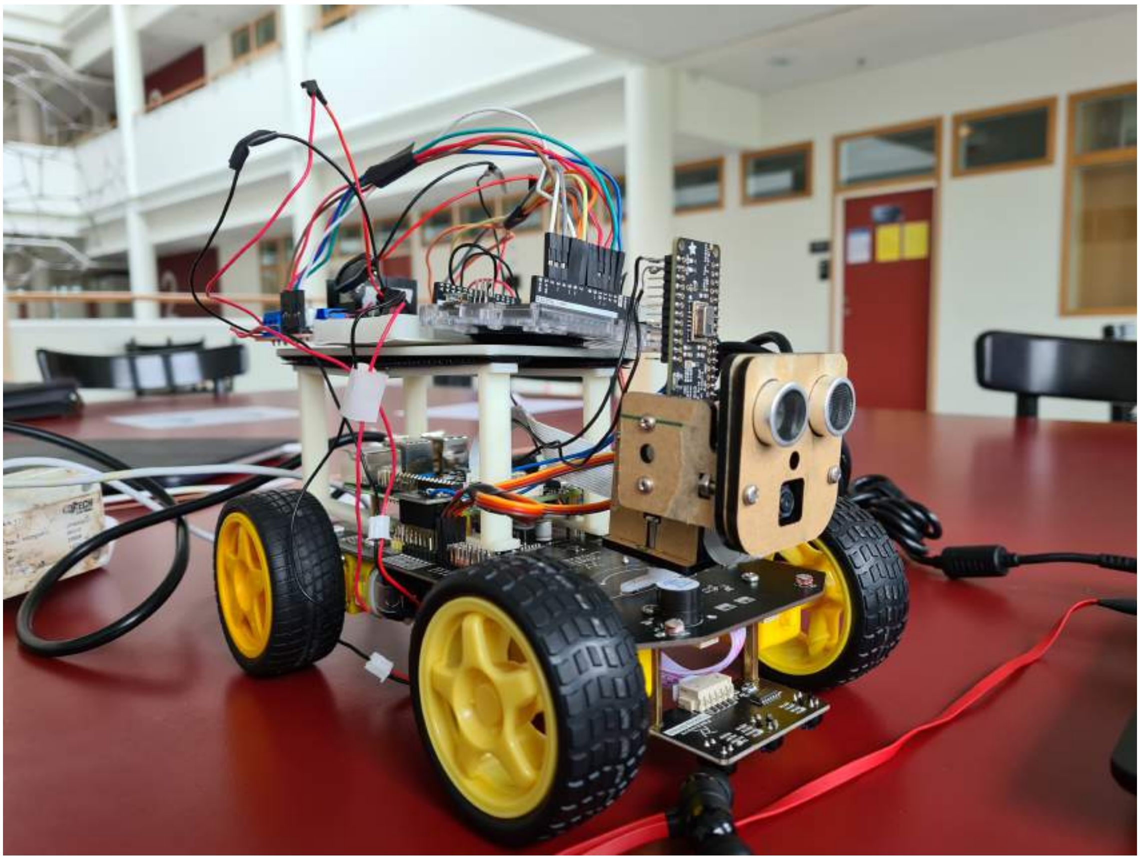


Figure 4.17: *3D printed Mounting brackets on the final construction.*

The other step was to combine the code base that controlled the NoIR Camera and the IR camera. The new code launched two windows that showed the footage from both cameras that the user would see. The task above was possible due to the help of the threading library in python that allows parallelism. The library allowed the execution of multiple hardware functions to run simultaneously. The code used to test and configure the cameras worked as the foundation for the implementation and ran simultaneously without any problems. The final code is in Appendix A with the commands used from the thread library.

A simple remote connection with RealVNC was able to provide a method to view footage from the cameras. The program did not require configuration due to the Raspberry came pre-installed with RealVNC that needed activation. The established remote connection provided the user with a way to observe what the cameras were seeing to the Raspberry Pi live without any problems.

The last step was to attach the Arduino and connect the wires from the motors to the L298H bridges that allowed the car itself to receive commands from the Arduino. An explored solution was connecting directly to the L298H bridges. Wires would come loose, resulting in a loss of power to the motors. A solution found included adding a small breadboard to form a bridge to the H bridges allowing a steady voltage.

Adjustments to accommodate the hardware added the possibility for increased mobility. One of them involved adding a power bank that powers the Arduino instead of using batteries to power it. The batteries instead were used to supply the L298H that made it easier to power them. The motor drivers, Arduino and breadboard all rest on the 3D printed standoffs capable of supporting the weight of the Arduino and L298H bridges. The power bank used to power the Arduino was attached with scotch tape that held it in place while the car was moving. The connection and location of the Arduino are in figure 4.18.

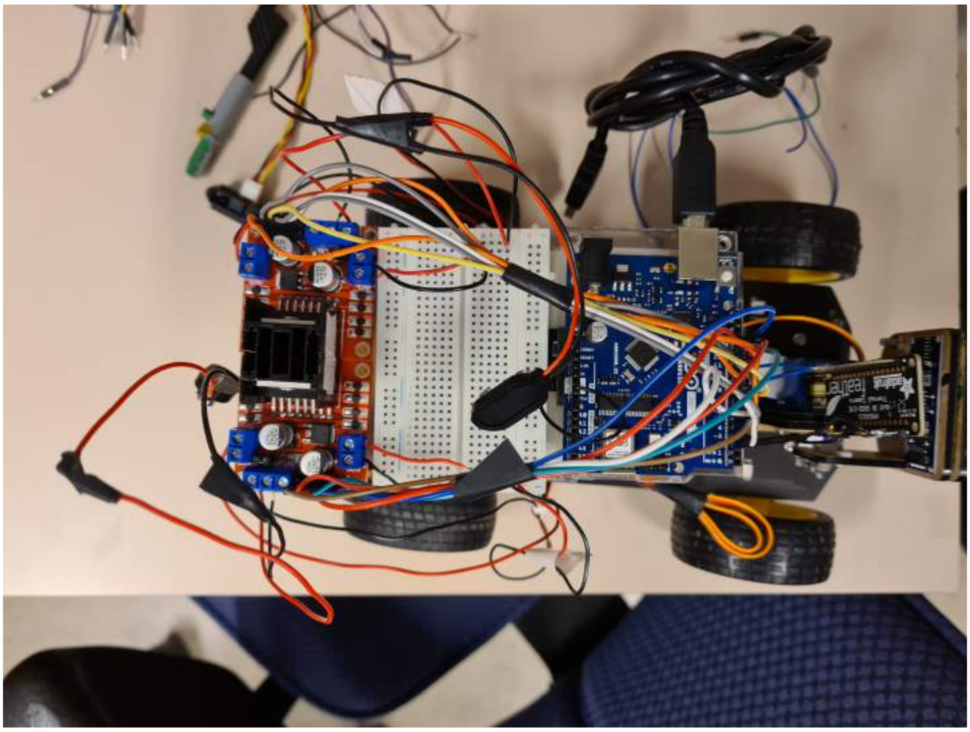


Figure 4.18: *Control unit attached to the kit*

5
Results

This chapter will show the findings made from the project and divide them into small areas which cover the areas tested on the robot. The result shows the movement, measuring thermal emission from objects and seeing objects in low light environments.

5.1 Movement of the Robot

The designed control unit was able to supply the motors with sufficient power to move the robot with ease and limited restrictions. The movement initially was lagging despite being fluent later on. The connections have to be secure to ensure that they will move the wheels without obstructions. The robot could left and right with ease while the batteries and power bank were enough to power the whole control unit. The chosen speed was suitable and managed to not affect the footage while moving.

In terms of turning, the car managed but struggled when the motors were not receiving sufficient power. The turning didn't disrupt any camera footage and managed to keep objects in the cameras viewing angles.

5.2 Measuring Thermal Radiation

The IR camera was able to measure the thermal emission from objects without problems. They served as a way to see how the camera would detect the thermal emission from the objects. Figure 5.1 shows a warm cup observed at a distance of 30 centimetres.

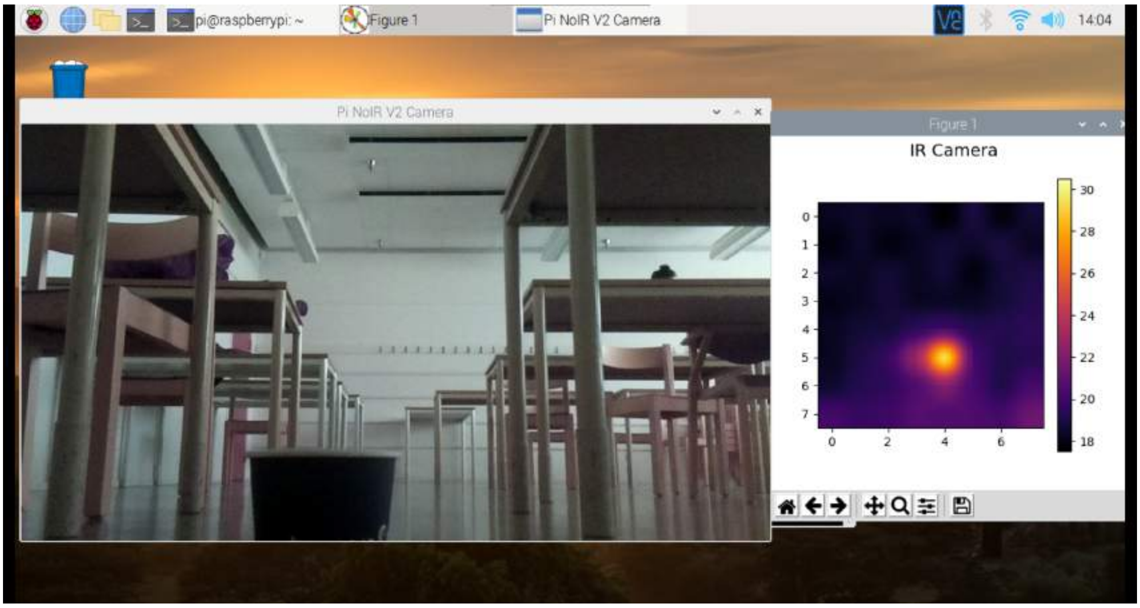


Figure 5.1: *Warm cup being observed on both of the cameras at close range.*

As seen in figure 5.1, the NoIR camera could see a small part of the cup. The IR camera was able to see the cup with the help of the thermal radiation emitted from the cup. The cup became observed at a distance to see how the camera would detect it. Figure 5.2 shows the same cup seen at a distance of 1 meter.

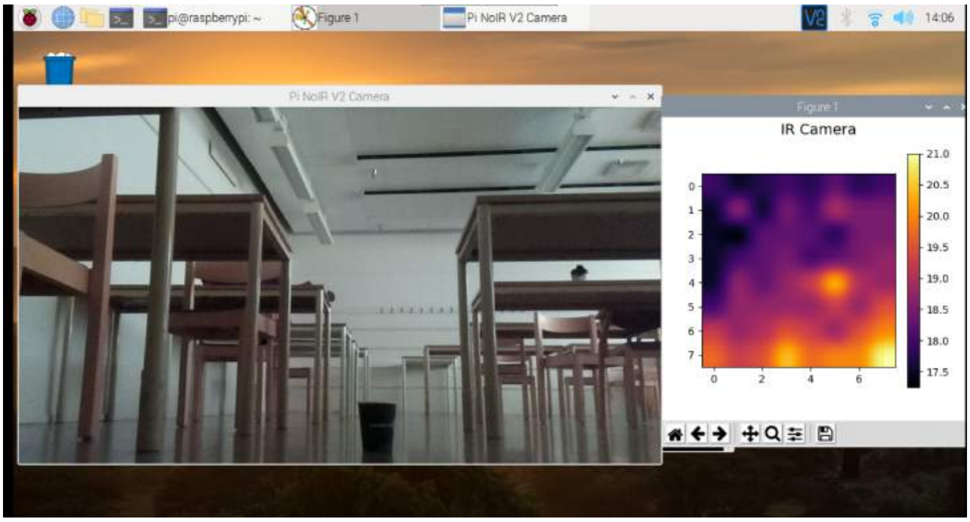


Figure 5.2: *Warm cup being observed by both cameras*

While the thermal radiation is visible, other objects in the area cause the camera hardly shows a clear enough picture of the heat radiating from the cup. The camera also captures the thermal radiation from nearby objects that causes the cup to be less visible than before. The same cup was later filled with cold water and held in front of a forearm. Figure 5.3 shows the cup with cold water in front of someone's forearm.

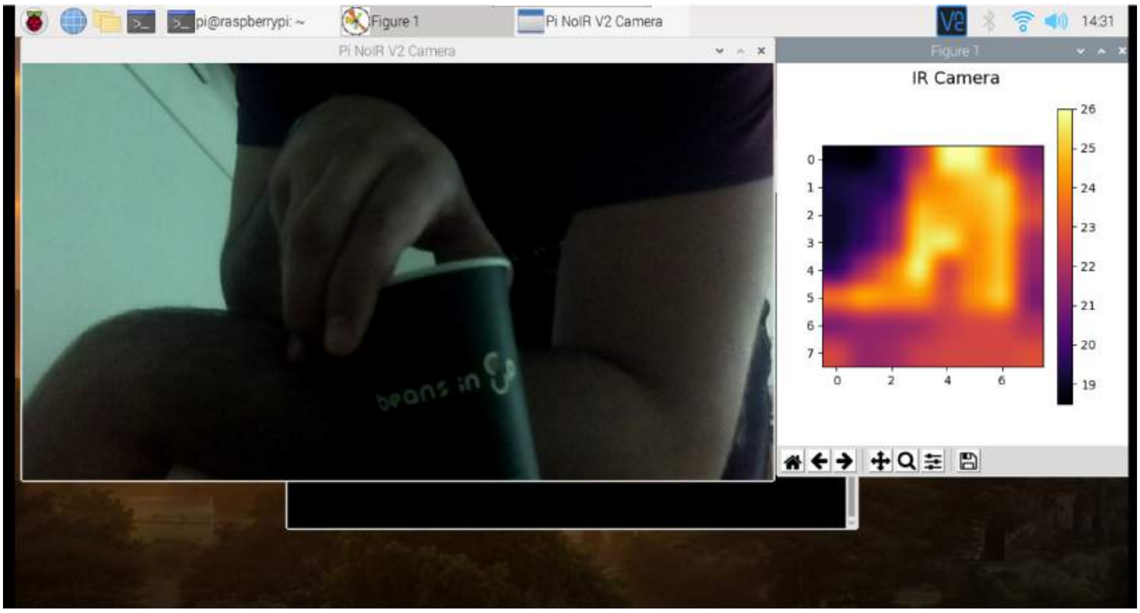


Figure 5.3: *Cold cup being held close to the camera in front of the forearm.*

The cup appeared on camera due to the temperature difference in front of someone's forearm. The camera detected it without any problems located on the right in the figure. The same cup with cold water became observed to see if the camera would still see the thermal radiation from the cup. Figure 5.4 shows the measurement mentioned above.

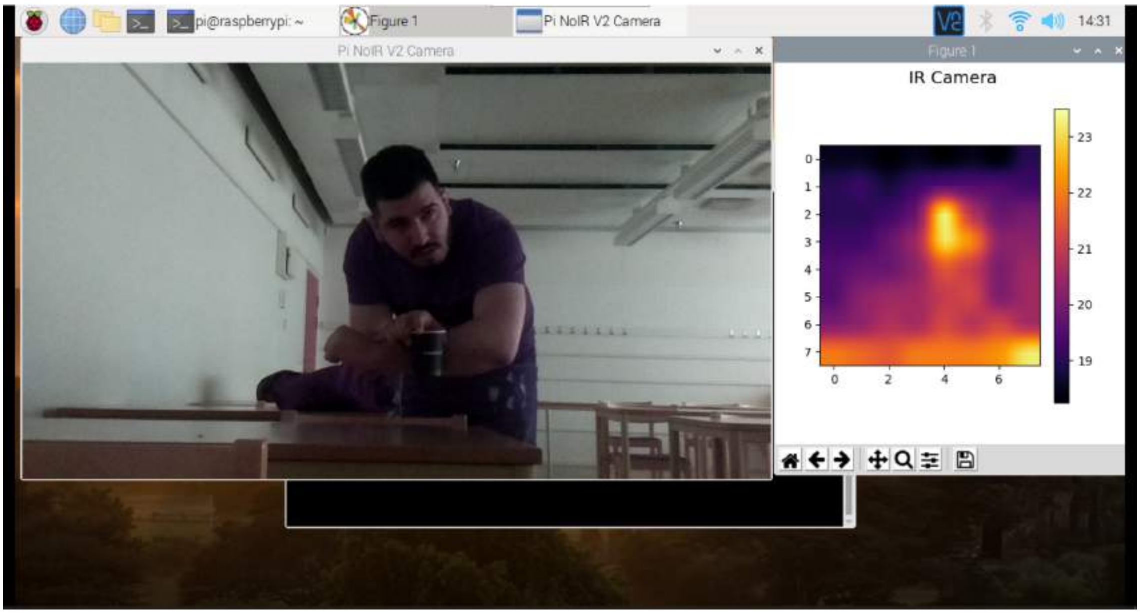


Figure 5.4: *Cold cup being held far from the camera in front of the forearm.*

The cup in the figure couldn't appear on the camera. Instead, the thermal radiation from the body is the most dominant one that the camera can see. It's also noted in the figure that the cup is barely detected on the IR camera while the other camera can see the cup.

A lighter became used to see how sensitive the IR camera would react and to see how the camera registered the flame. The test was done close to the cameras and far from them. Figure 5.5 shows the lighter up close to the car.

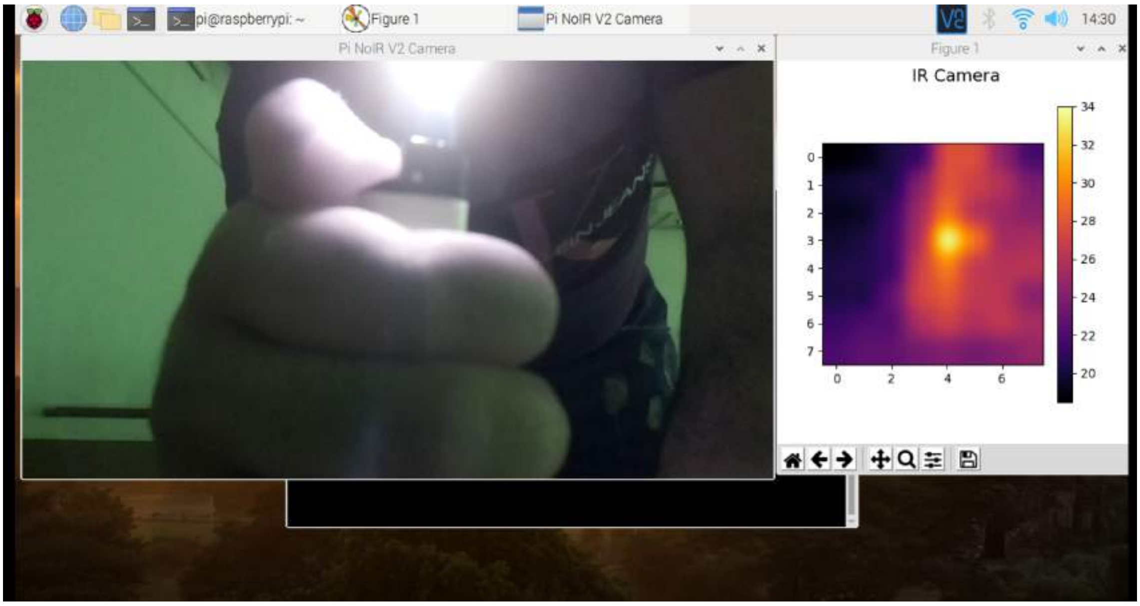


Figure 5.5: *Lighter up close to both of the cameras.*

The IR camera could detect the flame from the lighter. The window on the right shows the flame as a small light source warmer than the rest of the observed area. The camera was also able to see the underlying thermal radiation in the background without any problems. A test at a distance from the camera with the same lighter took place. The result is in figure 5.6.

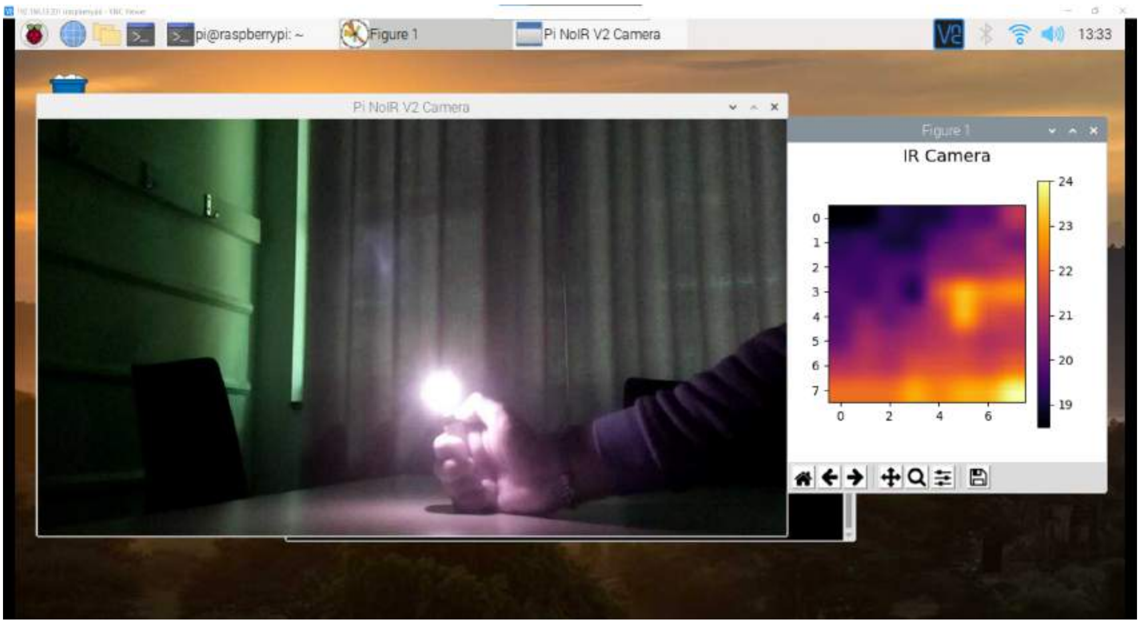


Figure 5.6: *Lighter located farther from the cameras.*

When placing the lighter at a distance, the flame couldn't appear due to the thermal radiation from the background. Instead, the arm was seen as the warmest object by the IR camera instead of the flame. The NoIR camera could see the glare but not as detailed despite managing to see the lighter itself.

5.3 Measuring in Low Light Conditions

The NoIR was tested in low light to see its capabilities of how the object is shown and detailed. The item used in the test was the lighter used to test the thermal radiation. The lighter was the only object used to measure in low light conditions due to its white colour. The two remote controls illuminated the lighter to help the camera see it.

The first test was when the car was far away from the lighter and moving closer. The second test includes when it was stationary and close to it. Both tests assisted in seeing how far the camera would detect the object when illuminated with the two remote controls. Figure 5.7 shows the camera close to the lighter.

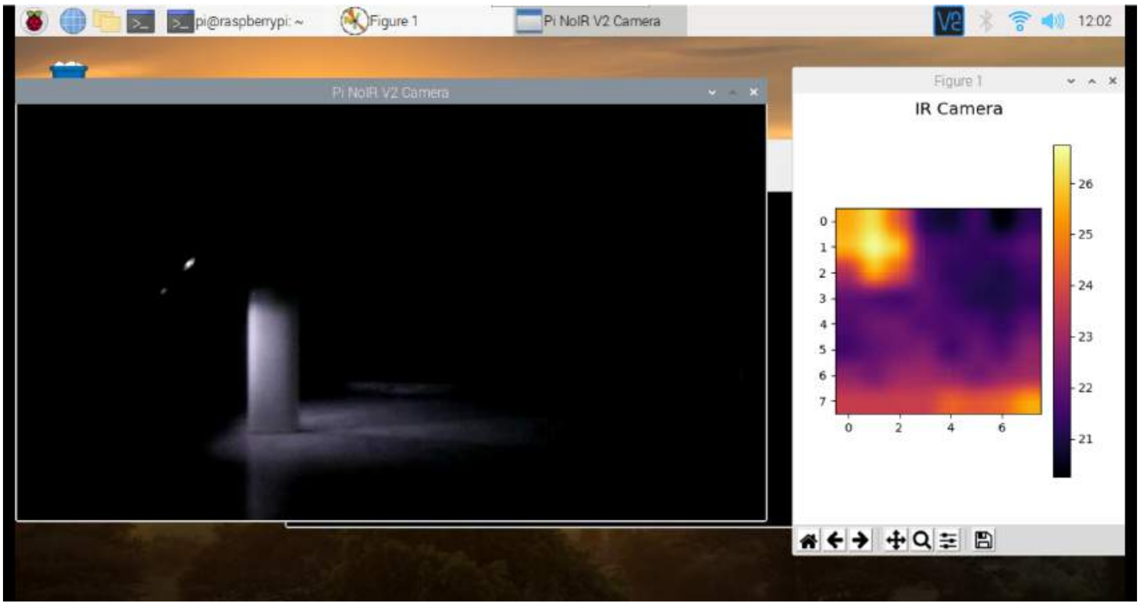


Figure 5.7: *Lighter in low light conditions close to the camera.*

The NoIR camera could see a small portion of the lighter due to how the two remote controls illuminated the lighter. The lower part of the lighter was lit up, which allowed the camera to see it. The same lighter was also observed at a distance to see how the camera would react. Figure 5.8 shows the same lighter but at a distance.

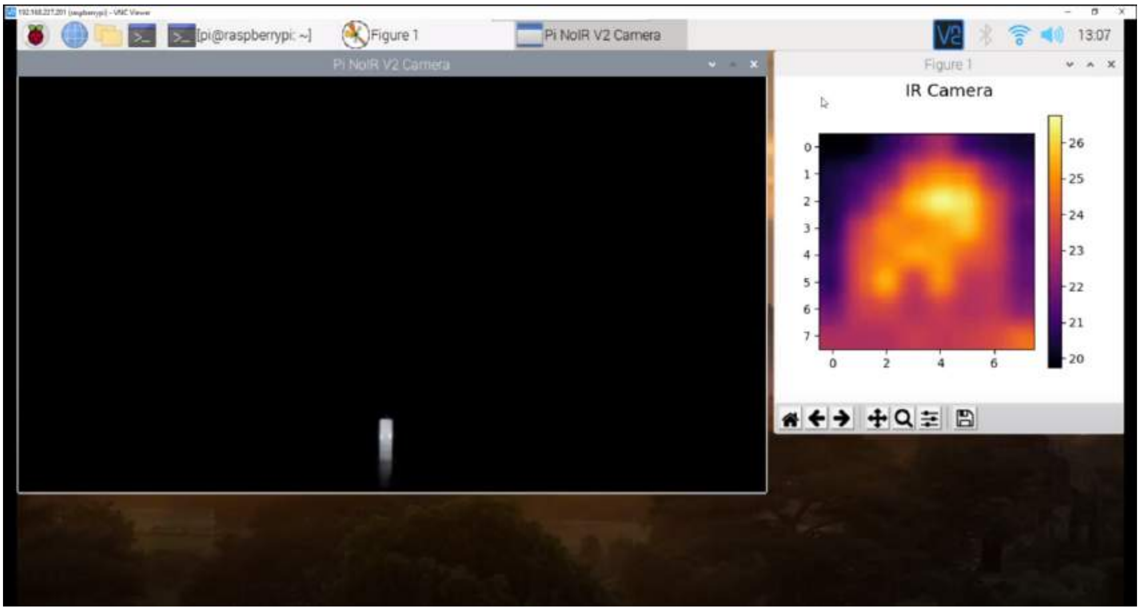


Figure 5.8: *Lighter in low light conditions far from the camera.*

The lighter observed in figure 5.8 can be detected with the NoIR camera and could see a portion with no problems. The IR camera on the right shows how the two remote controls are illuminating. The process of the illumination occurred by getting as close as possible to the lighter to make sure that the camera would see it.

6
Discussion

This chapter discusses the results of the implemented cameras and the implemented control unit for the car. Section 6.1 discusses the mobility and controls of the robot and suggests improvements and recommendations for future work.

Section 6.2 discusses the infrared camera and section 6.3 the low light camera while presenting the improvements and recommendations for future work. The last section discusses the sustainable development and ethical questions that arose during the project.

6.1 Movement and controls of the robot

In terms of the movement, the components could function properly when the car received commands to move. The added 3D brackets provided the stability that helped the motor drivers and breadboard provide stable current to the motors. The positioning of the drivers became a factor that affected the movement of the car because of the wires. The controls for the robot were responsive and could move the robot with ease.

The movement has room for improvement to become more stable and fluid. The initial idea was to use multiple servos to control the whole construction but abandoned due to the complexity of the code in terms of moving and changing direction. The current model could turn left and right but required more power which meant using two 9 volt batteries for the motors. The run time was also decreased due to the motors requiring more power. A recommendation would be to use batteries with a larger capacity to increase operation time.

The controls of the robot were stable and responsive than expected. The created access point provided a simple way to send signals to the Arduino by connecting to it. The page viewed on the browser was simple but could need more configuration to improve the look.

The connection between the Arduino and motor drivers was great but trying to power the motors arose. The signal from the Arduino was stable, but the car moved and turned slower due to the insufficient power provided by the batteries to the hobby gear motors. Another problem noticed was the number of wires that came loose unexpectedly that caused them to not function. A possible solution to explore in the future is to cover them or route them to reduce the chances of affecting other components.

6.2 Footage of the Infrared camera

The chosen camera provided footage that made it possible to see objects and note the sizable temperature differences. The downside was with the minor temperature differences, especially with various items. The detection range was also lower. An example is in figure 5.4, the thermal radiation of the cup wasn't visible while the radiation from the human body was visible.

The added hamming window function provided increased accuracy in detecting compared to the raw data acquired directly from the camera. The added filter caused a minor disadvantage when measuring the results. Some objects were hard to distinguish by when seen at long range. An example is when the cup was in front of the forearm. The applied window function could make large objects easier to see but, minor figures were less visible because of the loss of data from the filter.

The footage from the IR camera was stable during testing but, the detection of objects was a problem depending on the distance. It could detect objects from 10cm but, the camera picked up the thermal radiation from objects close by that obscured them from being seen the farther it became. It became difficult to see because the camera had difficulty seeing the item observed.

The Infrared camera was able to measure the human body with ease which is in the results but, there is room for improvements. One of them would be to measure each object with different filters to see the effect on the footage quality. The measurements would give a rough estimate of which filter is best to implement to see if the camera would focus on the thermal radiation of objects or human beings.

When it comes to using the infrared camera, it would be helpful to try the different window functions available in the *matplotlib.pyplot* library to see which filter has the best reception to detect thermal radiation from humans or objects. It would increase the accessibility while providing the user with the ability to choose filters based on the measurement. Another improvement would be to compare the footage of the Featherwing with another camera to see which can provide the best footage.

6.3 Footage on the low light camera

The NoIR camera proved to function better as expected with the initial tests done during the implementation. The available documentation found online provided enough understanding but lacked information about the low light capabilities. The tests done with the lighter in section 5.2 showed that the camera could see objects but required illumination from IR LEDs to see in dim light.

The tests done with the IR camera showed that the live footage quality was more than sufficient and was stable throughout. The other noticeable thing is the various colours that appear when exposed in illuminated environments. The solution was changing the auto white balance settings for the camera to compensate for the colours that appeared.

The live capture feature, however, came at a cost. It caused the Raspberry Pi to devote most of its resources to running the camera due to the size of the libraries. For future work, a recommendation is to look into other smaller libraries capable of delivering the same function but using fewer resources on the Pi.

When testing the capabilities in dark environments, the provided infrared light source was able to aid the camera is seeing the objects in front of it. Figure 5.7 proves that the camera can detect objects if they are shined with an infrared light source. During testing, the camera was also able to see it at a distance when shined. Figure 5,7 also proves that the NoIR camera requires a light source bigger than the two used IR LEDs for better illumination. The following points serve as recommendations for future work in terms of better performance

- Add the capability for the user to switch the camera settings by pressing a button. This setting will modify the camera to better support footage for both day and night.

- Explore ways to add a ring of IR LEDs around the NoIR camera for better low light footage while maintaining a reasonable distance from objects.

- Test the low light function with multiple objects to see which setting provides the best performance for which situation.

6.4 Sustainability and Ethics

During the project, multiple questions arose in terms of ethics and sustainability. Some ethical questions were about who would benefit the most, the precautions, and the implementation of the project in world scenarios. The project can help anyone who wants to apply a controllable surveillance car, but precautions need consideration to avoid privacy breaches.

Another issue is who stands to benefit from a project like this, especially if a company creates a similar product. One would be hesitant to use it due to not knowing who's controlling it. The result is a privacy issue and benefits a small group while absolving the responsibility.

In terms of sustainability, the three pillars of sustainability [28] were used to motivate the choices made. In terms of sustainability, the project strives to be inclusive by using hardware available to anyone. The hardware used can be found at a relatively low price anywhere, and they are guides that show how to use the hardware. The negative part is that the amount of electronics used is high, and the project tries to mitigate the issue by using platforms that provide increased accessibility of various components. The result is a minor reduction of resource consumption due to the amount of hardware needed is less.

7
Conclusion and Future recommendations

The project managed to develop a concept of a simple robot that could move based on the user input and capable of measuring objects in low light conditions and thermal radiation. The results proved that the chosen cameras worked together without any problems and provided footage that provided a clear distinction of what was measured. The measured results show that it's possible to create a small robot capable of being controlled remotely and providing surveillance but will require testing to increase the user experience while being sustainable and ethically viable.

While the project proved the concept to be possible, there are significant areas that can improve. These are the following:

- Find a way to add the capability to switch the camera settings to improve the footage quality.
- Add brighter Infrared LEDs to provide increased illumination.
- Test the low light function with multiple objects.
- Develop a platform that connects the Raspberry Pi and Arduino.
- Find ways to increase the operation time for the control unit.
- Analyze ways that the Arduino can handle the IR camera without taxing the control unit.
- Find and compare the different Infrared cameras available to find cameras that provide the best quality footage without affecting accessibility.

These recommendations are on the areas that can improve the project further. In conclusion, the concept developed in the project proved that it's possible to create a device capable of surveilling humans and objects with the help of devices located in the consumer market.

Bibliography

[1] Centre for Remote Imaging, Sensing and Processing,“Principles of Remote Sensing - Centre for Remote Imaging, Sensing and Processing, Crisp,” *Crisp.nus.edu.sg.* [Online].
Available: https://crisp.nus.edu.sg/ research/tutorial/em.htm,
Accessed on: 2021-03-26.

[2] V.Nordstrand, “Infrared Radiation,” in *Van Nordstrand’s Scientific Encyclopedia.* Hoboken, NJ, USA: John Wiley & Sons, Inc., 2007. [Online].
Available: https://onlinelibrary.wiley.com/doi/10.1002/0471743984.vse4181.pub2,
Accessed on: 2021-03-26.

[3] Libretexts, “Infrared:Application,” *Chemistry Libretexts*, August, 15, 2020. [Online]. Available: https://chem.libretexts.org/@go/page/1845,
Accessed on: 2021-03-28.

[4] M.Junedul, M. Muntjir, “Night Vision Technology:An Overview,” *International Journal of Computer Applications*, vol.167, no.13, pp. 37-42, Jun. 2017,
Accessed on: 2021-03-26. [Online]. doi: 10.5120/ijca2017914562.

[5] S. K. Sahoo, A. Ramulu, S. Batta and S. Duggal, "Performance analysis and simulation of three phase voltage source inverter using basic PWM techniques," in *IET Chennai 3rd International on Sustainable Energy and Intelligent Systems (SEISCON 2012)*, Tiruchengode, 2012, pp. 1-7. Accessed on: 2021-03-30. [Online]. doi: 10.1049/cp.2012.2223.

[6] Electronics Tutorials, “Pulse Width Modulation,” *Electronics-tutorials.ws.* [online].Available:https://www.electronics-tutorials.ws/blog/pulse-width-modulation.html, Accessed on: 2021-04-06.

[7] N. S. Yamanoor and S. Yamanoor, "High quality, low cost education with the Raspberry Pi," in 2017 IEEE Global Humanitarian Technology Conference (GHTC), San Jose, CA, USA, 2017, pp. 1-5. Accessed on: 2021-03-24. [Online]. doi: 10.1109/GHTC.2017.8239274.

[8] E.Upton, J. Duntemann, R. Roberts, T. Mamtora, and B. Eveard, *Learning computer architecture with raspberry pi,* 1st ed., Hoboken, NJ,USA: John Wiley & Sons, 2016. [Online]. Available:
https://ebookcentral.proquest.com/lib/chalmers/detail.action?docID=4653541#,
Accessed on: 2021-03-24.

[9] The Raspberry Pi Foundation, “Raspberry Pi foundation - about us,” *Raspberrypi.org,* Jan, 11, 2012. [Online]. Available: https://www.raspberrypi.org/about/, Accessed on: 2021-03-24.

[10] S.Monk, “Motors”, in *Raspberry Pi Cookbook: Software and Hardware Problems and Solutions.* 2nd ed., S. Conant, J. Bleiel, Ed. Sebastopol, CA, USA: O’Reilly Media, 2016, ch. 11, pp. 269 - 297.

[11] The Raspberry Pi Foundation, “Raspberry pi 4 model B specifications - raspberry pi,” *Raspberrypi.org.* [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-4-model-b/specifications/, Accessed on: 2021-04-14.

[12] The Raspberry Pi Foundation, “Power Supply,” *Raspberrypi.org.* [Online]. Available: https://www.raspberrypi.org/documentation/hardware/raspberrypi/power/README.md, Accessed on: 2021-04-14.

[13] B. Steward, *Adventures in Arduino.* 1st ed., Hoboken, NJ: USA: John Wiley & Sons, 2015.

[14] Components 101, “ATMega328P Microcontroller Pinout, Pin Configuration, Features & Datasheet,” Components101.com. [Online]. Available: https://bit.ly/3dqPor4, Accessed on: 2021-04-12.

[15] R. Teja, “What are the differences between Raspberry Pi and Arduino?,” Electronicshub.org, Apr, 05, 2021. [Online]. Available: https://www.electronicshub.org/raspberry-pi-vs-arduino/#Differences_between_Raspberry_Pi_and_Arduino, Accessed on: 2021-04-08.

[16] “Frenove 4WD Smart Car Kit for Raspberry Pi 4 B 3 B+ B A+, Face Tracking, Line Tracking, Light Tracing, Obstacle Avoidance, Colorful Light, Ultrasonic Camera Servo Wireless RC: Amazon.se: Electronics,” *Amazon.se.* [Online]. Available: https://bit.ly/3ezL5Jd, Accessed on: 2021-04-21.

[17] Freenove, “Freenove,” *Freenove.com.* [Online]. Available: https://www.freenove.com/, Accessed on: 2021-05-02.

[18] Freenove, “Tutorial,” *github.com.* [Online]. Available: https://bit.ly/3nI5zDD, Accessed on: 2021-04-30

[19] Freenove, “About Battery,” *github.com.* [Online]. Available: https://bit.ly/3eMahwk, Accessed on: 2021-05-02.

[20] The Raspberry Pi Foundation, “Pi NoIR Camera V2,” *Raspberrypi.org.* [Online].Available: https://www.raspberrypi.org/products/pi-noir-camera-v2/, Accessed on: 2021-05-02.

[21] Farnell, “Raspberry Pi NoIR Camera v2”, *farnell.com.* [Online]. Available: http://www.farnell.com/datasheets/2056180.pdf, Accessed on: 2021-05-02.

[22] Adafruit Industries, “Adafruit AMG8833 IR Thermal Camera Featherwing,” *Adafruit.com.* [Online]. Available: https://www.adafruit.com/product/3622, Accessed on: 2021-04-21.

[23] Sparkfun, “I2C - learn.sparkfun.com,” *Sparkfun.com.* [Online]. Available: https://learn.sparkfun.com/tutorials/i2c/all, Accessed on: 2021-04-21.

[24] AMG8833, “AMG8833 Temperaturgivare IC-krets,” *farnell.com.* [Online]. Available: https://bit.ly/3ebiZoD, Accessed on: 2021-04-21.

[25] Dejan, “L298N motor driver - Arduino interface, how it works, code, schematics,” *Howtomechatronics.com,* Aug, 08, 2017. [Online]. Available: https://howtomechatronics.com/tutorials/arduino/arduino-dc-motor-control-tutorial-l298n-pwm-h-bridge/, Accessed on: 2021-05-05.

[26] “Interface L298N DC Motor Driver Module with Arduino,” *Lastminuteengineers.com.* [Online]. Available: https://lastminuteengineers.com/l298n-dc-stepper-driver-arduino-tutorial/, Accessed on: 2021-05-05.

[27] A. Rosebrock, "Accessing the Raspberry Pi Camera with OpenCV and Python - PyImageSearch,"*Pyimagesearch.com,* Mar, 30, 2015. [Online].Available: https://www.pyimagesearch.com/2015/03/30/accessing-the-raspberry-pi-camera-with-opencv-and-python/., Accessed on: 2021-05-05.

[28] B.Purvis, Y.Mao, and D. Robinson, "Three pillars of sustainability: in search of coneptual origins,"*Sustain.Sci.,* vol.14, no. 3, pp. 681-695, 2019. [Online]. Available: https://link.springer.com/article/10.1007/s11625-018-0627-5#Abs1, Accessed on: 2021-05-31.

A

Appendix 1

A.1 Code for the two Cameras

```
import time
import busio
import board
import adafruit_amg88xx
import cv2
import matplotlib.pyplot as plot
from picamera.array import PiRGBArray
from picamera import PiCamera
from threading import Thread

def IR_Camera():
    #Initialization of i2C bus
    i2c_bus = busio.I2C(board.SCL, board.SDA)

    #Initialization of the camera sensor
    sens = adafruit_amg88xx.AMG88XX(i2c_bus)

    #Time to wait for the sensor to initialize
    time.sleep(0.1)

    while True:
        #Get data from the sensor
        data = sens.pixels

        #Drawing the figure with the help of interpolation
        figure = plot.imshow(data, cmap = "inferno", interpolation
  = "hamming")

        #Plotting the color bar for the figure
        plot.colorbar()

        #Title for the plot
        plot.suptitle("IR Camera", fontsize=14)

        #Running the GUI for a certain time
        plot.pause(0.1)

        #Clearing the figure
        plot.clf()


```

```python
def NoIR_Camera():
    #Intializing the Camera
    camera = PiCamera()

    #Camera resolution
    camera.resolution = (864, 480)

    #Setting the framerate
    camera.framerate = 60

    #Set the AWB mode, gains only be change if AWB is off
    camera.awb_mode = 'auto'
    #camera.awb_gains = (1.3 , 1.2)

    #Generating a 3D array which handles and stores the values based on decided resolution
    capture = PiRGBArray(camera, size = (864, 480))

    #Time to allow camera to start
    time.sleep(0.1)

    #Capture frames from the camera to create a Numpy array
    for frame in camera.capture_continuous(capture, format="bgr", use_video_port=True):

        #Grab the raw data from the  Numpy array
        image = frame.array

        #Display the data from the array
        cv2.imshow("Pi NoIR V2 Camera",image)

        #Wating for a key to be pressed
        key = cv2.waitKey(1) & 0xFF

        #Clear in order to allow for the next frame
        capture.truncate(0)

        #break the loof if a key is pressed
        if key == ord("e"):
            break

#Enabling multithreading to allow both to run at the same time
Thread(target = NoIR_Camera).start()
Thread(target = IR_Camera).start()
```

B

Appendix 2

B.1 Code for the Control Unit in the Arduino

```
// Control unit that sends signal to the motors via Wi-fi
// Created with the help of the AP Simple Web Server WiFi example from the WiFiNINA library
// Latest change: Changed directions for the motor in order to rotate in the correct direction and added a lower speed to allow turns
// Necessary libraries to use wi-fi
//http://192.168.4.1/ is the address to access the website
#include <SPI.h>
#include <WiFiNINA.h>
#include "arduino_secrets.h"


// Pins that control the motor Speed
#define ENA1  3 // Front Right Motor
#define ENB1  5 // Rear Right Motor
#define ENA2  6 // Rear Left Motor
#define ENB2  9 // Front Left Motor

// Pins for Left Motors
#define IN1_L2  2 // 1st input for Rear Left motor
#define IN2_L2  4 // 2nd input for Rear Left Motor
#define IN3_L1  7 // 1st input for Front Left Motor
#define IN4_L1  8 // 2nd Input for Front Left Motor

// Pins for Right Motors
#define IN1_R1 10 // 1st input for Front Right Motor
#define IN2_R1 11 // 2nd Input for Front Right Motor
#define IN3_R2 12 // 1st Input for Rear Right Motor
#define IN4_R2 13 // 2nd Input for Rear Right Motor

///////please enter your sensitive data in the Secret tab/arduino_secrets.h
char ssid[] = SECRET_SSID;        // your network SSID (name)
char pass[] = SECRET_PASS;       // your network password (use for WPA, or use as key for WEP)
int keyIndex = 0;                // your network key Index number (needed only for WEP)

// Speed for the car
int speed  = 127; //This is based on a 50% duty cycle
```

```
int status = WL_IDLE_STATUS;

WiFiServer server(80);

void setup() {

  //Initialize serial and wait for port to open:
  Serial.begin(9600);

  while (!Serial)
  {
    ; // wait for serial port to connect. Needed for native USB
   port only
  }

  Serial.println("Access Point Web Server");

  // Pinmode for the controlling of the motor speed
  pinMode(ENA1, OUTPUT);
  pinMode(ENB1, OUTPUT);
  pinMode(ENA2, OUTPUT);
  pinMode(ENB2, OUTPUT);

  // pinMode for the motors on the left
  pinMode(IN1_L2, OUTPUT);
  pinMode(IN2_L2, OUTPUT);
  pinMode(IN3_L1, OUTPUT);
  pinMode(IN4_L1, OUTPUT);

  // pinMode for the motors on the right
  pinMode(IN1_R1, OUTPUT);
  pinMode(IN2_R1, OUTPUT);
  pinMode(IN3_R2, OUTPUT);
  pinMode(IN4_R2, OUTPUT);

  // check for the WiFi module:
  if (WiFi.status() == WL_NO_MODULE)
  {
    Serial.println("Communication with WiFi module failed!");
    // don't continue
    while (true);

  }

  String fv = WiFi.firmwareVersion();
  if (fv < WIFI_FIRMWARE_LATEST_VERSION)
  {
    Serial.println("Please upgrade the firmware");
  }

  // by default the local IP address of will be 192.168.4.1
  // you can override it with the following:
  // WiFi.config(IPAddress(10, 0, 0, 1));
  // print the network name (SSID);
  Serial.print("Creating access point named: ");
  Serial.println(ssid);
```

```

  // Create open network. Change this line if you want to create an
    WEP network:
  status = WiFi.beginAP(ssid, pass);

  if (status != WL_AP_LISTENING)
  {
    Serial.println("Creating access point failed");

    // don't continue
    while (true);
  }

  // wait 10 seconds for connection:
  delay(10000);

  // start the web server on port 80
  server.begin();

  // you're connected now, so print out the status
  printWiFiStatus();
}

void loop()
{
  // compare the previous status to the current status
  if (status != WiFi.status())
  {
    // it has changed update the variable
    status = WiFi.status();

    if (status == WL_AP_CONNECTED)
    {
      // a device has connected to the AP
      Serial.println("Device connected to AP");
    }
    else
    {
      // a device has disconnected from the AP, and we are back in
   listening mode
      Serial.println("Device disconnected from AP");
    }
  }

  WiFiClient client = server.available();   // listen for incoming
   clients
 // if you get a client,
  if (client)
  {
    Serial.println("new client");           // print a message out
   the serial port
    String currentLine = "";                // make a String to
   hold incoming data from the client

     // loop while the client's connected
    while (client.connected())
```

```
 {
    // if there's bytes to read from the client,
   if (client.available())
   {
     char c = client.read();             // read a byte, then
     Serial.write(c);                    // print it out the serial monitor

     // if the byte is a newline character
     if (c == '\n')
     {
     // if the byte is a newline character
       // if the current line is blank, you got two newline characters in a row.
       // that's the end of the client HTTP request, so send a response:
       if (currentLine.length() == 0)
       {
         // HTTP headers always start with a response code (e.g. HTTP/1.1 200 OK)
         // and a content-type so the client knows what's coming, then a blank line:
         client.println("HTTP/1.1 200 OK");
         client.println("Content-type:text/html");
         client.println();

         // the content of the HTTP response follows the header:
         client.print("Click <a href=\"/F\">here</a> to go Foward <br>");
         client.print("Click <a href=\"/B\">here</a> to go Backward <br>");
         client.print("Click <a href=\"/R\">here</a> to go Right <br>");
         client.print("Click <a href=\"/L\">here</a> to go Left <br>");
         client.print("Click <a href=\"/S\">here</a> to Stop <br>");

         // The HTTP response ends with another blank line:
         client.println();

         // break out of the while loop:
         break;
       }
       else
       {
       // if you got a newline, then clear currentLine:
         currentLine = "";
       }
     }
     else if (c != '\r')
     {    // if you got anything else but a carriage return character,
       currentLine += c;      // add it to the end of the currentLine
     }
```

```
        // Check to see if the client request was "GET /H" or "GET /L":
        if (currentLine.endsWith("GET /F"))
        {
          forward();                // GET /F Makes the car go foward
        }
        if (currentLine.endsWith("GET /B"))
        {
          backward();                 // GET /B Makes the car go backwards
        }
        else if (currentLine.endsWith("GET /R"))
        {
            right();                // GET /R Makes the car turn to the right
        }
        else if (currentLine.endsWith("GET /L"))
        {
            left();                 // GET /L Makes the car turn to the left
        }
        else if(currentLine.endsWith("GET /S"))
        {
          stopping();
        }
      }
    }

    // close the connection:
    client.stop();
    Serial.println("client disconnected");

  }
}

void printWiFiStatus()
{
  // print the SSID of the network you're attached to:
  Serial.print("SSID: ");
  Serial.println(WiFi.SSID());

  // print your WiFi shield's IP address:
  IPAddress ip = WiFi.localIP();
  Serial.print("IP Address: ");
  Serial.println(ip);

  // print where to go in a browser:
  Serial.print("To see this page in action, open a browser to http://");
  Serial.println(ip);
}

void forward()
{
    //Starting and moving the Front Left Motor
```

```
    digitalWrite(IN3_L1, LOW);
    digitalWrite(IN4_L1, HIGH);
    analogWrite(ENB2, speed);

    // Starting and moving the Rear left motor
    digitalWrite(IN1_L2, HIGH);
    digitalWrite(IN2_L2, LOW);
    analogWrite(ENA2, speed);

    // Starting and moving the Front Right Motor
    digitalWrite(IN1_R1, LOW);
    digitalWrite(IN2_R1, HIGH);
    analogWrite(ENA1, speed);

    //Starting and moving the Rear Right Motor
    digitalWrite(IN3_R2, HIGH);
    digitalWrite(IN4_R2, LOW);
    analogWrite(ENB1, speed);
}

void backward()
{
  // Starting and moving the Rear left motor
    digitalWrite(IN1_L2, LOW);
    digitalWrite(IN2_L2, HIGH);
    analogWrite(ENA2, speed);

    //Starting and moving the Front Left Motor
    digitalWrite(IN3_L1, HIGH);
    digitalWrite(IN4_L1, LOW);
    analogWrite(ENB2, speed);

    // Starting and moving the Front Right Motor
    digitalWrite(IN1_R1, HIGH);
    digitalWrite(IN2_R1, LOW);
    analogWrite(ENA1, speed);

    // Starting and moving the Rear Right Motor
    digitalWrite(IN3_R2, LOW);
    digitalWrite(IN4_R2, HIGH);
    analogWrite(ENB1, speed);
}

void left()
{
 // Starting and moving the Rear left motor
    digitalWrite(IN1_L2, LOW);
    digitalWrite(IN2_L2, HIGH);
    analogWrite(ENA2, speed);

    //Starting and moving the Front Left Motor
    digitalWrite(IN3_L1, HIGH);
    digitalWrite(IN4_L1, LOW);
    analogWrite(ENB2, speed);

   // Starting and moving the Front Right Motor
```

```
    digitalWrite(IN1_R1, LOW);
    digitalWrite(IN2_R1, HIGH);
    analogWrite(ENA1, speed);

    //Starting and moving the Rear Right Motor
    digitalWrite(IN3_R2, HIGH);
    digitalWrite(IN4_R2, LOW);
    analogWrite(ENB1, speed);
}

void right()
{
   //Starting and moving the Front Left Motor
    digitalWrite(IN3_L1, LOW);
    digitalWrite(IN4_L1, HIGH);
    analogWrite(ENB2, speed);

    // Starting and moving the Rear left motor
    digitalWrite(IN1_L2, HIGH);
    digitalWrite(IN2_L2, LOW);
    analogWrite(ENA2, speed);

   // Starting and moving the Front Right Motor
    digitalWrite(IN1_R1, HIGH);
    digitalWrite(IN2_R1, LOW);
    analogWrite(ENA1, speed);

    // Starting and moving the Rear Right Motor
    digitalWrite(IN3_R2, LOW);
    digitalWrite(IN4_R2, HIGH);
    analogWrite(ENB1, speed);
}

void stopping()
{
  // Stopping the motor on the Front Right
  digitalWrite(IN1_R1, LOW);
  digitalWrite(IN2_R1, LOW);
  analogWrite(ENA1, speed);

  // Stopping the motor on the Rear Right
  digitalWrite(IN3_R2, LOW);
  digitalWrite(IN4_R2, LOW);
  analogWrite(ENB1, speed);

  // Stopping the motor on the Rear Left
  digitalWrite(IN1_L2, LOW);
  digitalWrite(IN2_L2, LOW);
  analogWrite(ENA2, speed);

  // Stopping the motor on the Front Left
  digitalWrite(IN3_L1, LOW);
  digitalWrite(IN4_L1, LOW);
  analogWrite(ENB2, speed);
 }
```

```
//Contents in arduino_secrets.h

// Both SSID and password must be 8 characters or longer
#define SECRET_SSID "Your SSID"
#define SECRET_PASS "Your PASSWORD"
```

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS