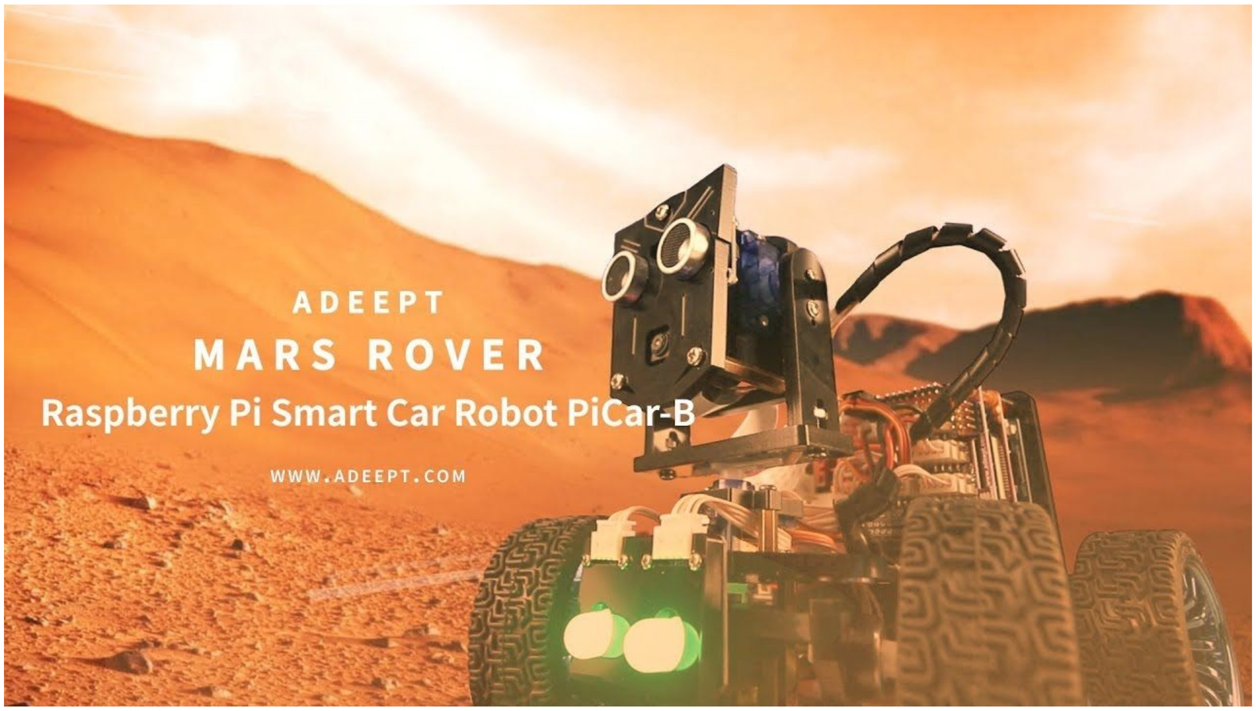


Multimedia Technology of PiCar-B Mars Rover

December 04, 2019
– CSC 461

Shuwen(Helen) Li
Zijian(Jay) Chen

Executive Summary

Picar_B Mars rover is a complete artificial intelligence robot kit based on the Raspberry Pi, and this robot car is a collection of speech recognition, video transmission and computer vision. This report investigated the background technology of Picar-B Mars Rover to get a deep understanding of the practical application in multimedia technology by combining the theoretical knowledge learned from the class with the technologies of this robot car [1]. The Raspberry Pi robot is composed of electronic components, and is controlled by the mainframe programming of Python code and a precise GUI program, thereby achieving the following functions：

- Speech Recognition：Picar can understand human's words and then execute the commands
- Automatic Obstacle Avoidance: Avoid obstacles ahead and find the next path based on ultrasonic sensor
- Object Recognition and Tracking: Track objects of a specific shape or colour based on OpenCV
- Real-time Video Transmission: Transfer the real-time images taken by the Raspberry Pi camera to a remote computer
- Remotely Controlled by APP: Remotely control the robot through the buttons on the keyboard or the virtual buttons on the GUI

We installed and tested the above functions one by one and studied the multimedia principles behind them. We also hope that the comparison and analysis of different obstacle detection methods can help us to get a deep understanding of obstacle avoidance techniques for the autonomous driving model.

Table of Contents

1. Introduction

1.1. Topic and Purpose

This report investigated practical application in multimedia technology through Analysis of the multimedia technologies of Picar-B Mars Rover.

1.2. Background Context

This PiCar-B Mars Rover is a prototype of a mockup real Mars Rover, and this robot car is made by Adeept Inc and its computer board is supported by Raspberry Pi and its system is run by Raspbian [2]. The purpose of doing this prototype is to let study to learn about how to use the multimedia systems to control the robot car or other kinds of space equipment remotely [1]. This project will be considered a useful project for people who are interested in studying space exploration and space transportation.

1.3. Problem Definition

1.3.1 Problem Needed to Solve

Basic on Preliminary research , A qualified robotic vehicles should have a walking function at all angles, a voice recognition function, and an obstacle monitoring and avoidance function. And Picar contains several modules such as Gear Motor, camera and ultrasonic sensor module. In order to explore the connection between these components and the control system, we propose the following questions :

- What specific functions can these corresponding electronic components perform in Picar?
- In addition, the user needs to set and control various functions of this robot car on the computer. What kind of technology is needed for command transmission?
- Picar can recognize a single object or all objects ahead. Is there a difference between the two principles of identification?

- Are there any restrictions on a particular object when Picar is tracking it? From the technical perspective, how does this robotic car determine that this object is the one to be tracked?
- In the process of speech recognition, how can sound waves be transformed into a language that can be recognized by Picar?

1.3.2 Goals

The goal of this report is to research and explore the multimedia technology and related code of PiCar-B Mars Rover through solving the above questions.

1.3.3 Constraints

- Time limitation: We must finish the project by the end of class.
- Coding support: The client code and server code are provided by Adeept, so I can't modify too much on the provided code.
- Technical limitation: The PiCar-B is a mockup, so it has its own limit, so we must follow the default and do not overpower it.

1.3.4 Potential Benefits

The self-driving vehicle has video cameras to detect traffic lights, read road signs and keep track of other vehicles, while also looking out for pedestrians and other obstacles. Meanwhile, ultrasonic sensors in the wheels can detect the position of kerbs and other vehicles when parking [3]. Studying the two corresponding functions of the PiCar can help us to understand how the driverless technology works when detecting obstacles, so as to better understand the application of multimedia technology in real life.

2. Methodology

2.1 Preliminary Research

Based on online searching, we found the Picar contains several modules, for example, camera, microphone, and ultrasonic sensor module. So this robot car able to use the camera to track the real-world situation and we can use the OpenCV library to track the object and the specific shape or colour [4]. Furthermore, we are able to write an AI program to make this car become an AI robot and it can run by its module sensor to track the road and to avoid obstacles ahead and find a possible path for a move. In addition, we can use the microphone system and the Python Speech Recognition library to accept the human voice as a command to control the robot car is moving forward or backward [1]. Therefore, we need to set up the program and study how the library works with the microphone system. Through obtaining information on PiCar's characteristics above, we want to combine the line tracking module, ultrasonic sensor, and camera together to make them can cross data and share the data with each other, then simulate an auto-driving robot car [5].

2.2 PiCar-B Installation

The installation of the PiCar-B is divided into two parts, hardware and software. The correct installation of the hardware ensures the normal operation of subsequent commands, and the display of all software functions are also performed by hardware [1]. This is the package list of PiCar:

1. 1 Set Acrylic Plates
2. 1x Adeept Motor HAT V2.0
3. 1x Raspberry Pi Camera(with Cable)
4. 1x USB Microphone
5. 1x Ultrasonic Sensor Module
6. 2x Adeept RGB LED Module

7. 4x Adeept WS2812 RGB LED Module

8. 1x Adeept 3CH Line Tracking Module

9. 3x Servo

10. 1x Gear Motor

11. 4x Wheels

12. 1x Battery Holder

13. 1x Cross Socket Wrench

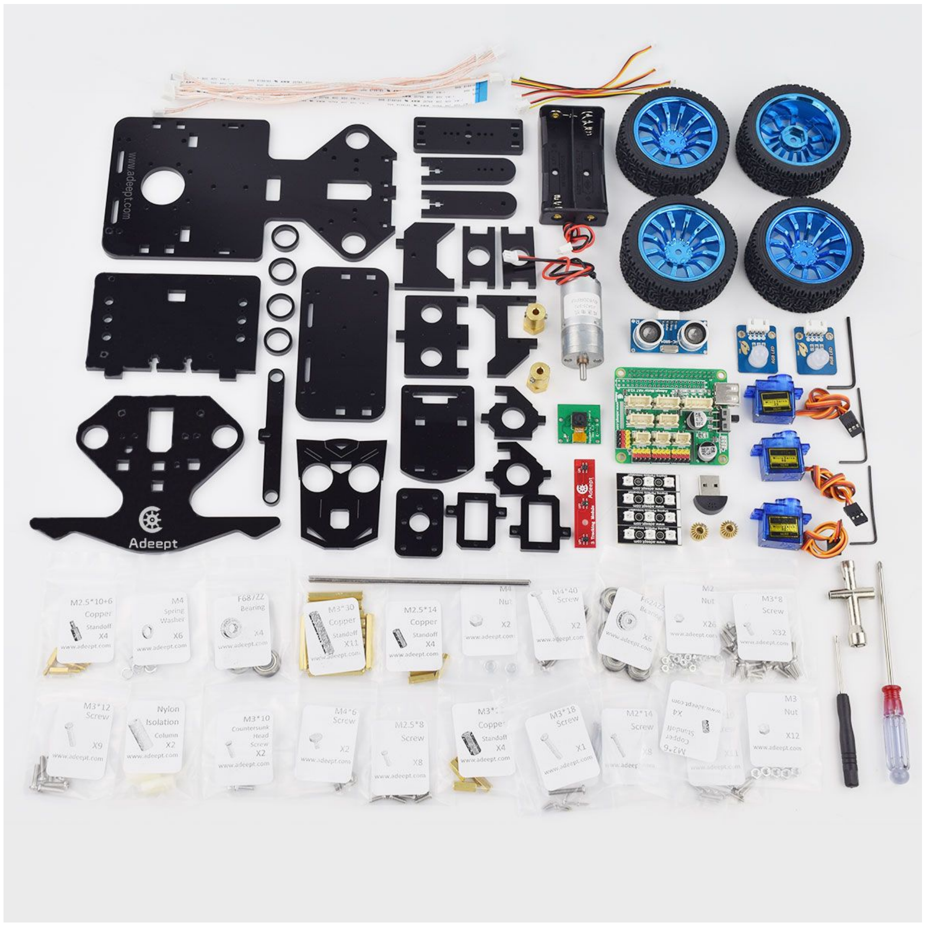
14. 2x Cross Screwdriver(Small and Large)

15. 1x Winding Pipe

16. 10x Bearing(6*F624ZZ + 4*F687ZZ)

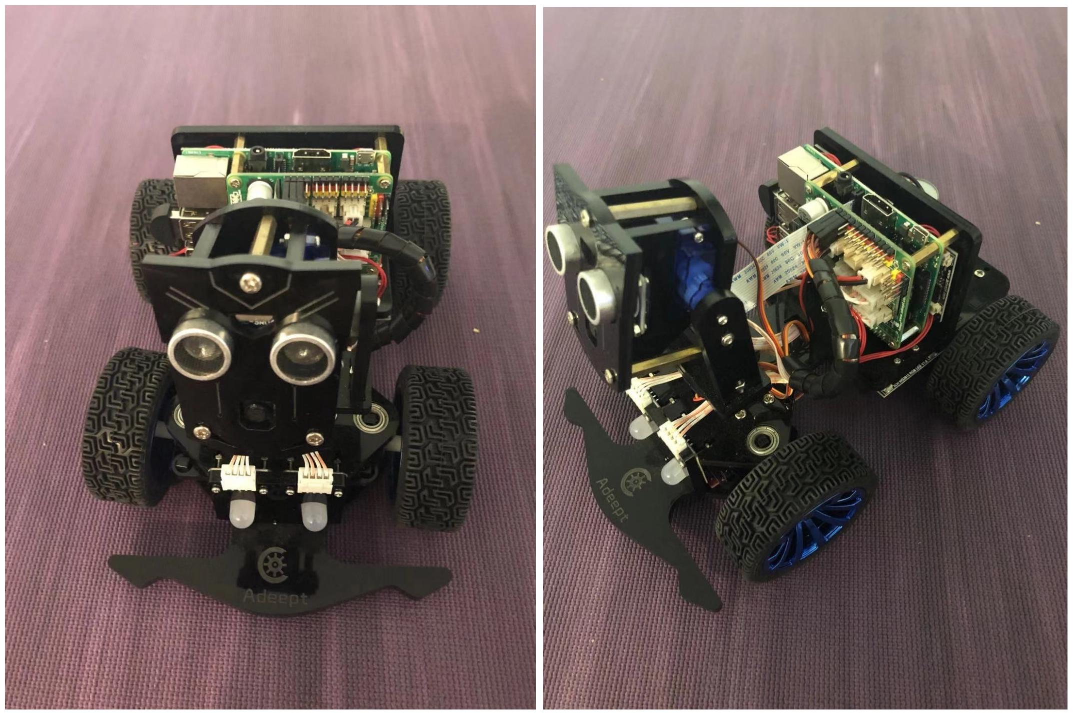
17. 2x Umbrella Gear Set

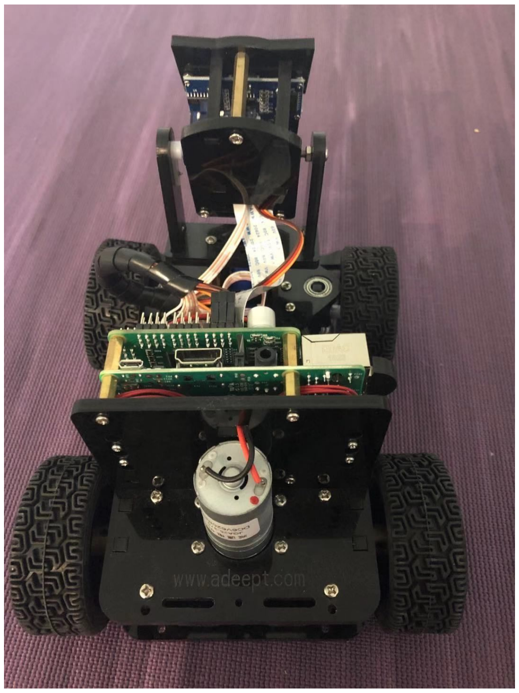
18. Other necessary accessories(Wires, Nuts, Screws, Copper Standoffs, Couplings)

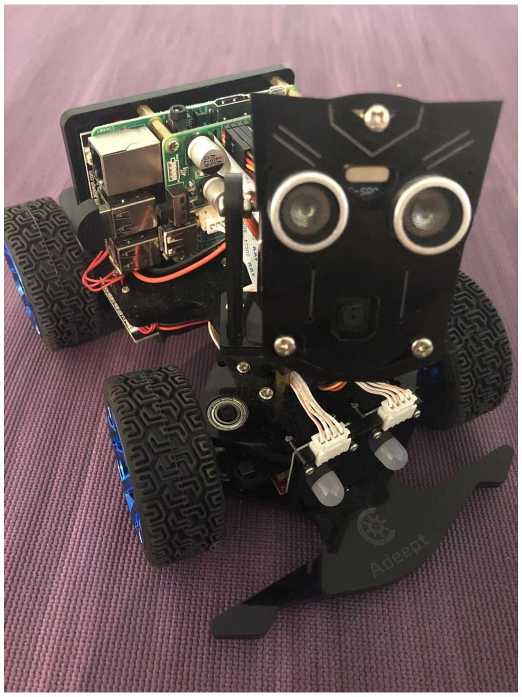


And also, we need to provide a Raspberry Pi single-board computer:

The hardware part of the PiCar is assembled through combining the basic parts with the Raspberry Pi single-board, and the following picture shows the completed Picar Mar Rover:

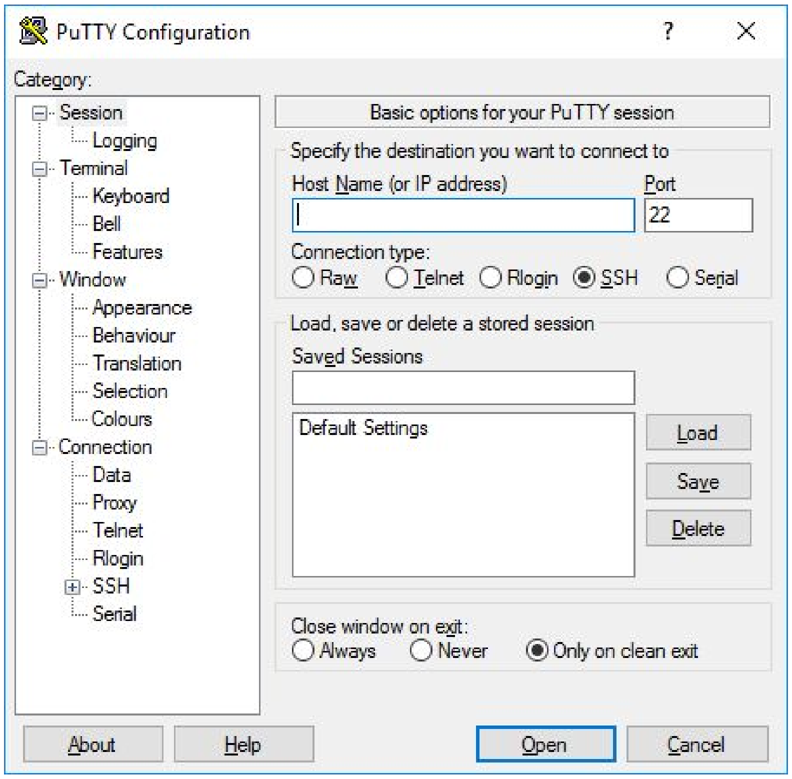






```
country=US
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
network={
        ssid="WIFI"
        psk="PASSWORD"
        key_mgmt=WPA-PSK
        priority=1
}
```

Now, we turn on the computer and log in to our network services profile and find out the IP address assigned by our internet router. Once, we found the IP address of the Picar then we can use the PuTTY application to login to the computer remotely.



Specify the destination you want to connect to

Host Name (or IP address):

Port: 22

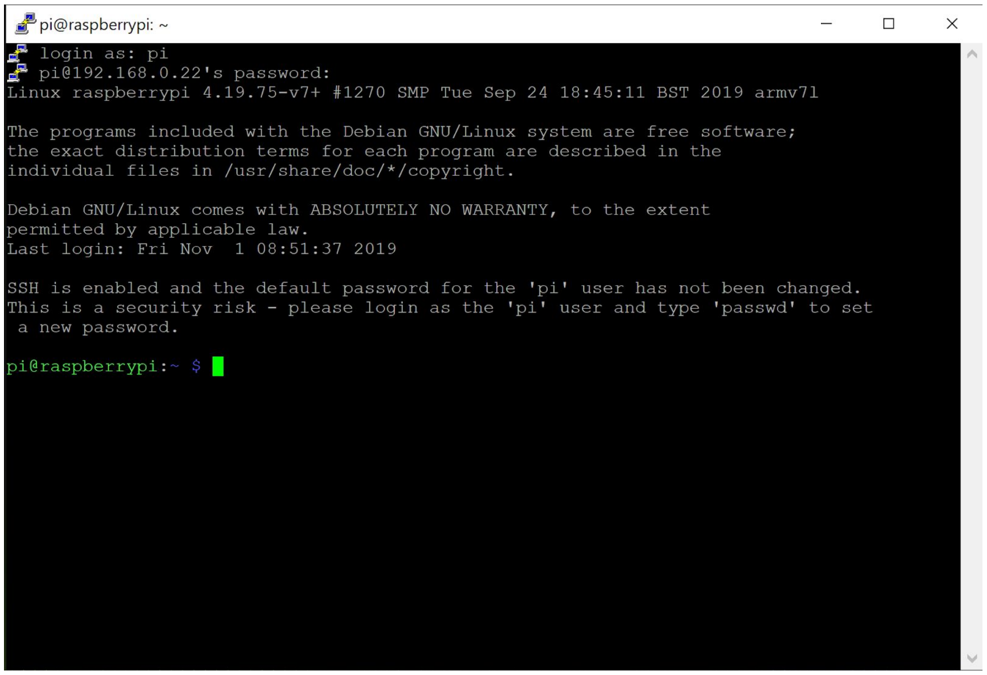
Connection type: Raw Telnet Rlogin [✓ SSH] Serial

Load, save or delete a stored session

Saved Sessions: Default Settings

Close window on exit: Always Never [✓ Only on clean exit]

Finally, we able to login to the computer by using the PuTTY application, and now we finish the assembly part of the Picar, and the next step is to install the application and python library on my desktop and Raspberry pi computer. That will make us allow to transfer data between two computers.



```
login as: pi
pi@192.168.0.22's password:
Linux raspberrypi 4.19.75-v7+ #1270 SMP Tue Sep 24 18:45:11 BST 2019 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Nov  1 08:51:37 2019

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
 a new password.

pi@raspberrypi:~ $
```

2.3 Environmental Package

The Adeept company provided us with a lot of code related to multimedia transmission, tracking and control module. Before running and analyzing this code, we need to install the adapted environment package. Firstly, we need to update and upgrade our software system on the Raspberry Pi computer, and that make sure we are using the latest version updated software. And also, we need to enable some settings on the computer, and that makes us allow the use of those library functions in our system environment [2]. Secondly, we installed that support application and python library on both computers. For example, we need to install OpenCV and Numpy library to let the Picar allow us to write a function that makes the camera track the object and detecting the real-world environment. And these support application and python library show in following:

1. I2C-Tools - it using for check the external devices are connected successfully and are it working or not.
2. Python drive PCA9685 - it using to control the PWM servo/LED.
3. Python package Setuptools - it is a python package using for facilitating packaging.
4. Python package Wheel - it is a dynamic object-oriented programming language.
5. Python package PyAudio - it is a python package for allowing us to use a microphone to accept our voice command.
6. Flac - it is to convert the sound to a .flac file for speech recognition and make the Picar follow our order.
7. Sphinxbase and Pocketsphinx - it is a part of CMU Sphinx Open Source Toolkit for Speech Recognition.
8. Bison - it is a program that can convert computer language grammar into a C language program and to make the computer can execute.

9. ALSA - Advanced Linux Sound Architecture is a software framework and part of the Linux kernel that provides an application programming interface (API) for sound card device drivers.
10. Swig - is used to connect computer programs or libraries written in C or C++ with Python in our project.
11. Python package SpeechRecognition - is a library for performing speech recognition, with support for several engines and APIs, online and offline.
12. Python package OpenCV - it is a library of Python bindings designed to solve computer vision problems.
13. Python package libatlas-base-dev libjasper-dev libqtgui4 python3-pyqt5 libqt4-test - re Python packages for FPV functions and sending jpeg stream to PC.
14. Python package zmq - it carries messages across inproc, IPC, TCP, TIPC, multicast.
15. Python package pybase64 - it provides a fast base64 implementation for base64 encoding/decoding.
16. Python package rpi_ws281x - is a Raspberry Pi library for controlling WS281X LEDs.

After we finish the support application and python library installing on Raspberry pi computer, we need to install related software on the user's desktop. In the user's desktop, we just need to install some software related to the movement and tracking of the car such as Setuptools, PyAudio and Swig. At this point, all the basic settings include hardware and software perspectives have been completed. The continuous running and testing of the related code will help us better understand the multimedia technology of the PiCar.

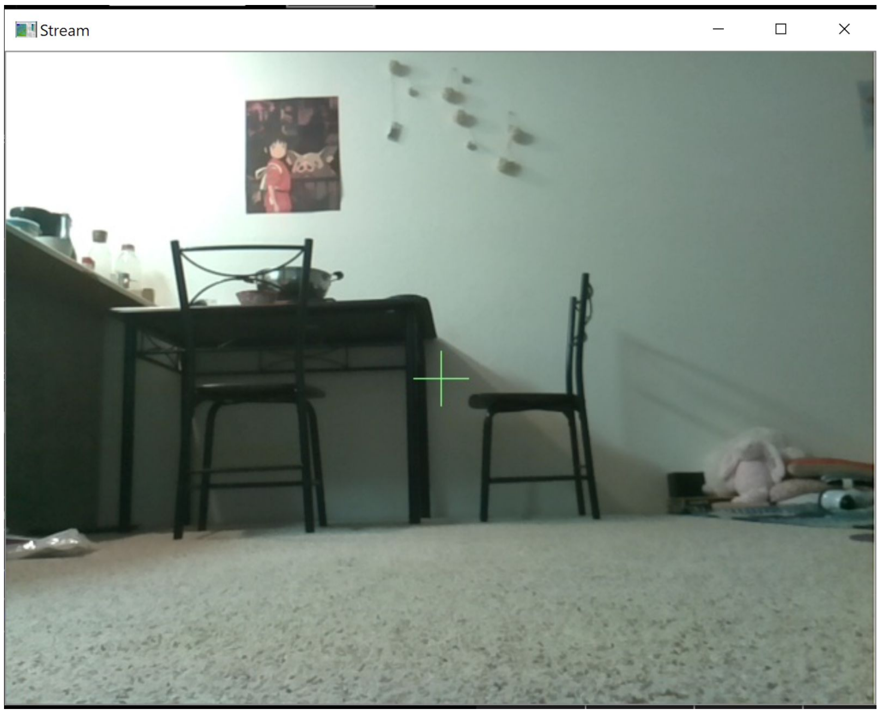
The next section of the report presents the results of our study methods and a discussion of how the background technology of PiCar works.

3. Results

3.1 Findings

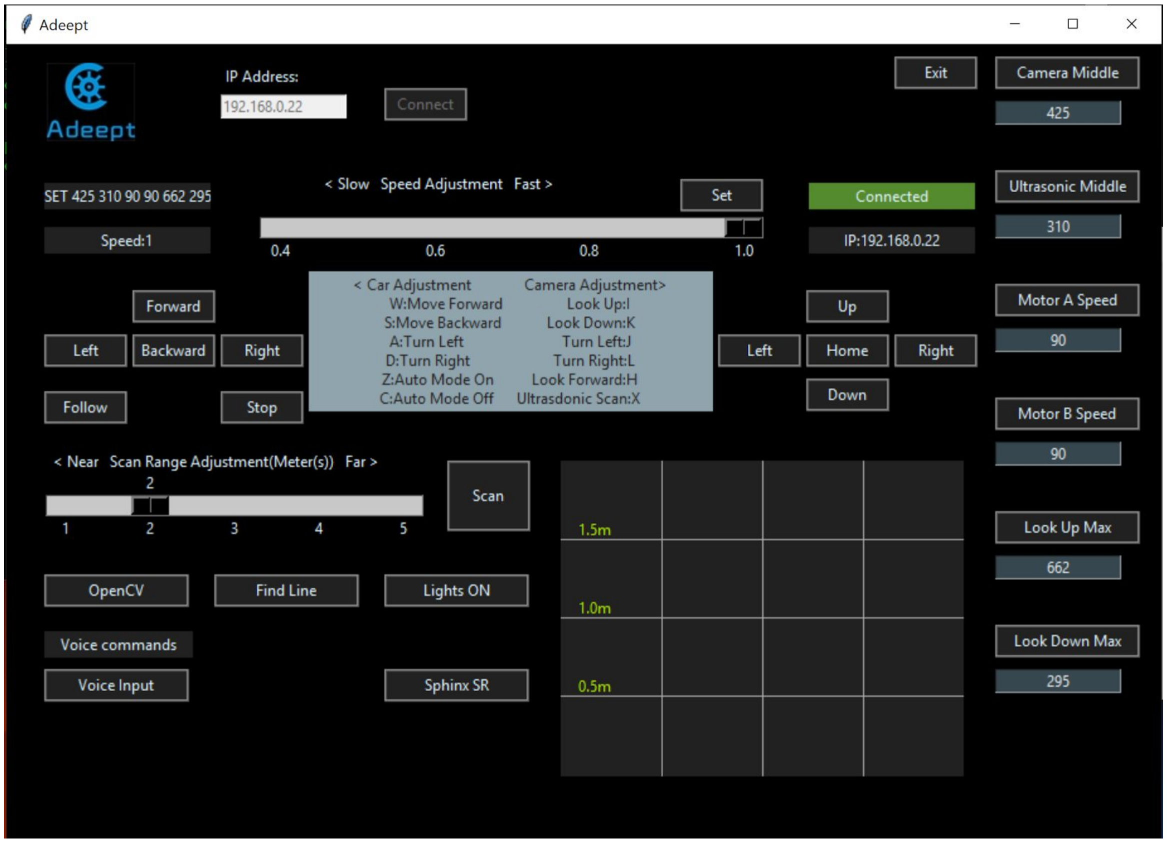
3.1.1 Real-time Video Transmission

The real-time video transmission on the PiCar is using mainly use TCP/IP to transfer the video frame from PiCar to our local desktop. The video media is a kind of sequence of the image, so every frame of the video is equally a large binary data set. So, in our project it using Numpy to deal with these large data sets, and to use TCP to send this data set or video data to our local desktop. The TCP will guarantee every byte will be delivered to our client, so it will happen lost data and that will make our client lost frames. So, PiCar uses the TCP to do the real-time video transmission will not perfect, and sometimes the video will slow and not really real-time. There are some delays in video transmission, as we mentioned in class, the online video stream is better to use RTP. So, based on our results we find out there is something that can be improved on the real-time video transmission of the Picar-B.



3.1.2 Remotely Controlled by APP

The remotely controlled is based on python GUI function, and the user can use the app to control the PiCar remotely. GUI is a form of user interface that allows users to interact with electronic devices through graphical icons and an audio indicator [6]. Once the user clicks the bottom and the controller will automatically send a message to the server, the server-side once accepts the message command it will automatically run the correct the program and give respond to our client-side. For example, if the user clicks the forward bottom the controller will send to the client a message forward and send a message stop to our server, and the server accepted this command it will power up the PiCar and start the gear motor go forward and stop. If the user holds the forward bottom the client will just send the forward to our server until the user releases the bottom and PiCar will be stopped. There are many functions on the GUI controller, such as the auto-follow model, the OpenCV model, and the voice command model [4]. Also, there is an ultrasonic scan function, once the client sends this command to our server and the server will start to scan the font of the PiCar and will send back the response back to our client and it will draw a visual graph to show how the ultrasonic wave detection. That user can use this graph to control the Picar without seeming the real-world environment.



IP Address: 192.168.0.22

SET 425 310 90 90 662 295

Speed:1

Connected

IP:192.168.0.22

Camera Middle 425

Ultrasonic Middle 310

Motor A Speed 90

Motor B Speed 90

Look Up Max 662

Look Down Max 295

< Car Adjustment
W:Move Forward
S:Move Backward
A:Turn Left
D:Turn Right
Z:Auto Mode On
C:Auto Mode Off

Camera Adjustment>
Look Up:I
Look Down:K
Turn Left:J
Turn Right:L
Look Forward:H
Ultrasdonic Scan:X

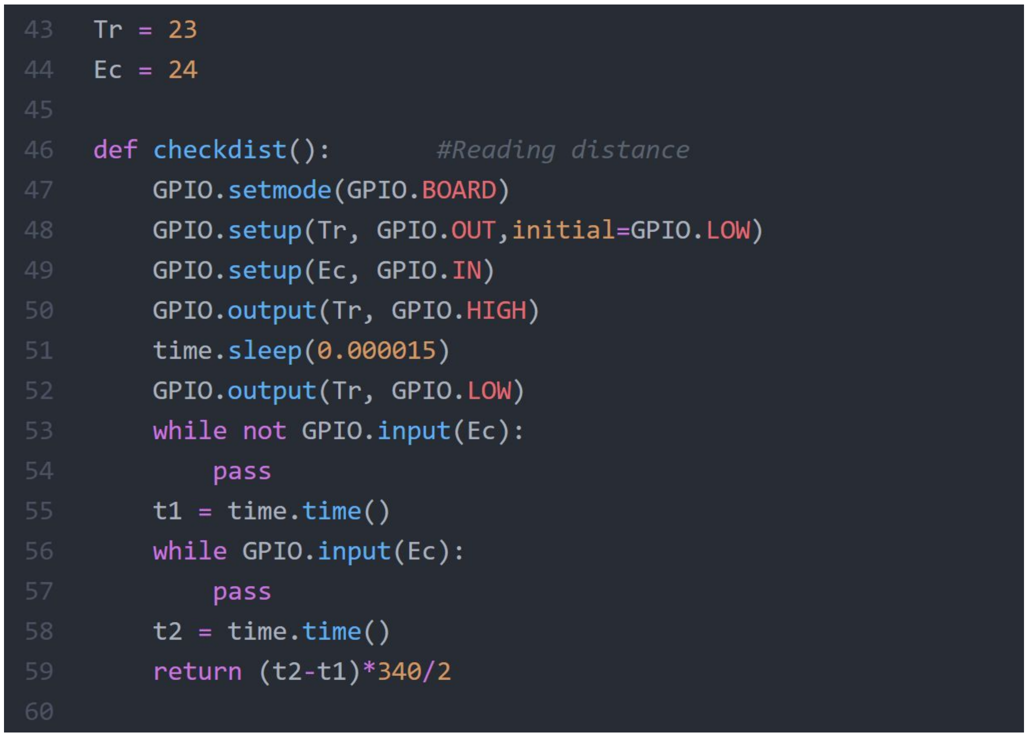
3.1.3 Movement of Picar

On the power transmission system, a gear motor is an important part that is a homogeneous and compact unit consisting of a gear unit and a motor [7]. The central role in the gear motor is performed by the gear unit and its gear unit stages, the gear pairs. These features transmit the force of the motor from the input end to the output end, so the gear unit, therefore, functions as a converter of speed and torque. On the PiCar, a gear motor is responsible for power up the picar to move forward or backward. And micro servo can rotate approximately 180 degrees (90 in each direction) [8], so the micro servo on PiCar is to control the front wheel to make a turn, and also move the head of PiCar to look up or down or left or right.

3.1.4 Automatic Obstacle Avoidance

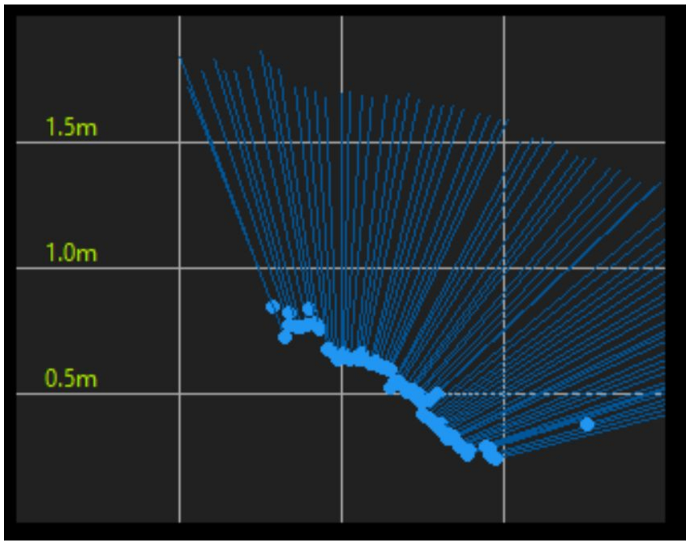
The PiCar is using the ultrasonic sensor to do the obstacle scanning. The PiCar will detect the front of the PiCar with about 180-degree angle. Once the object is detected the echo wave will give the response back to the ultrasonic sensor, and it to calculate the distance based on the time required. The general formula is Distance = Time * Sound of speed * 1/2. Normally the 20 C the sound of speed is 343 meters/second [9].

On the provided code, there are shows it set up the 23 and 24 pins for the reserve to the ultrasonic sensor. If we need to use the ultrasonic sensor we need to use GPIO.LOW to set up the sensor as initial, which is ready to activate the ultrasonic waves [10]. To start the ultrasonic sensor we need to set it to GPIO.HIGH, and once we did that the sensor will start sending out the ultrasonic waves [10]. After sent out the waves we need to let the sensor sleep little times because we don't want the echo detector to get a response too quickly. After the sleep, we start to accept the echo-response and calculate the distance by using the formula.



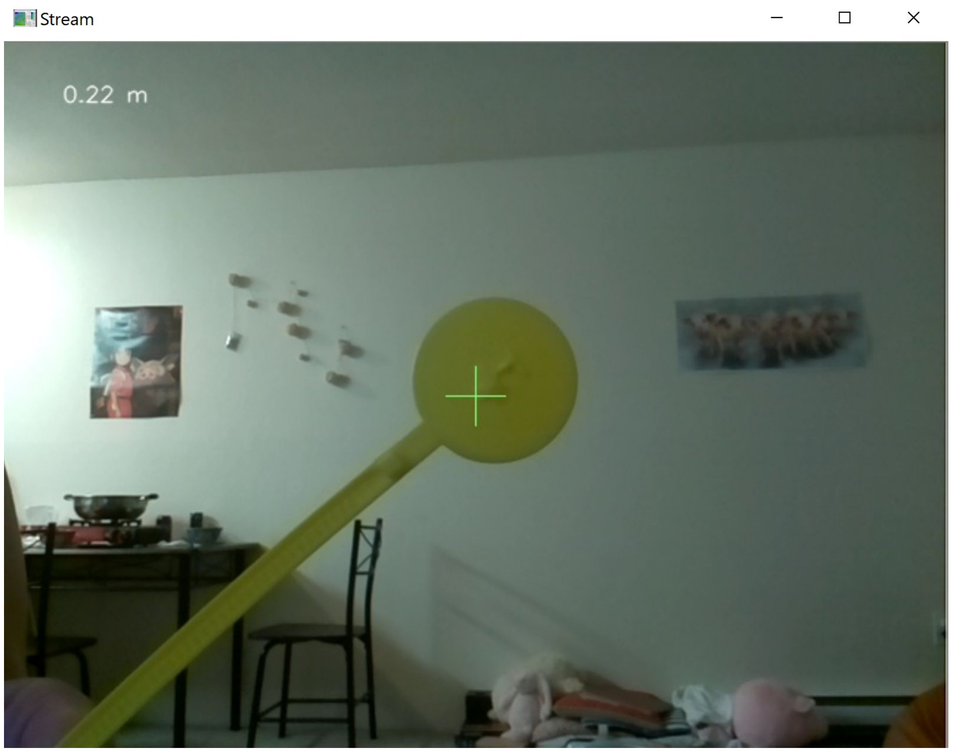
```python
Tr = 23
Ec = 24

def checkdist():       #Reading distance
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(Tr, GPIO.OUT,initial=GPIO.LOW)
    GPIO.setup(Ec, GPIO.IN)
    GPIO.output(Tr, GPIO.HIGH)
    time.sleep(0.000015)
    GPIO.output(Tr, GPIO.LOW)
    while not GPIO.input(Ec):
        pass
    t1 = time.time()
    while GPIO.input(Ec):
        pass
    t2 = time.time()
    return (t2-t1)*340/2

```

After accepting all the echo-response the graph will draw out the visual graph to show how the environment looks like. For example, the graph below shows the many blue lines connect to many blue points, and the points which is exactly the object right at, and the lines are the body of that object that ultrasonic wave can be detected. In this example, we set the furthest range to 2 meters, so the sensor will not detect further than 2 meters. The furthest range is 5 meters for this PiCar, and if we set the range larger than 5 meters the response will have error estimation.



3.1.5 Object Recognition and Tracking

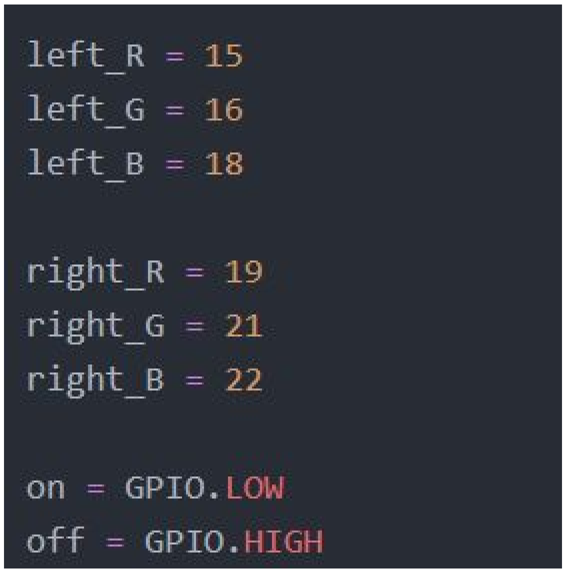
The PiCar supports the camera, so we use the Python library OpenCV to detect object and tracking. In our project, we set the program to detect the largest yellow object on the screen, so if we use other colours the OpenCV will not active and PiCar will not give any response [4]. Once I hole a yellow object into the screen and the camera will start tracking the object and PiCar will follow it. Also, the OpenCV can calculate the distance between the object and itself and shows on the top left of the screen. In the following example, there are shows 0.22 m, and we already set the default good distance to keep at 0.4 m, so the PiCar will do backward for this case.



In our project, the PiCar is run by the client and server both sides, so these two sides need to communicate with each other so this video stream is transferred by using TCP. As we know, TCP is not best for video streaming, so there are always delay in my video streaming. if we just look at the video stream and control the PiCar by remotely and that may have the issue on the control side.

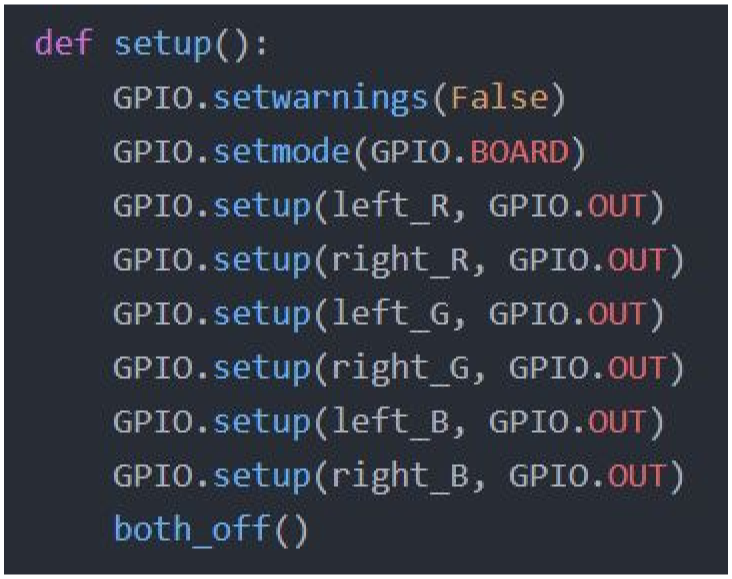
3.1.6 LED Morse Code

In the presentation video, we show how to use PiCar led light to delivered morse code, and this code is modified by our team members. The LED we use in the PiCar left LED we set the pin to 15, 16, 18 [1]. Once we use 15 and the PiCar will turn on the left light with a red colour, 16 will be green colour and 18 will be blue [10]. On the right LED it will use pin 19, 21, and 22 for turn on different colours [10]. Also, if we use GPIO.LOW means to power down, which means turn off the LED.



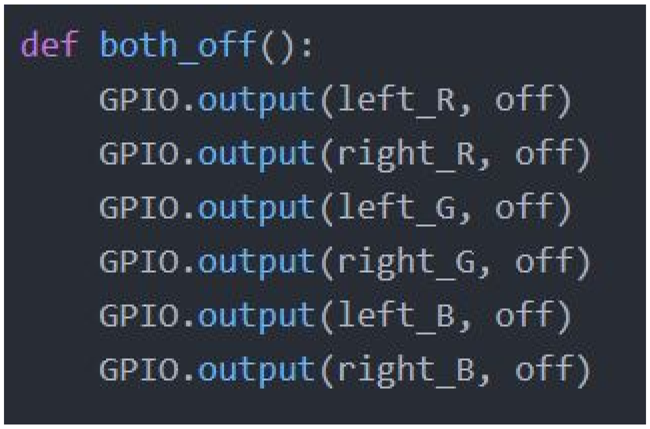
```
left_R = 15
left_G = 16
left_B = 18

right_R = 19
right_G = 21
right_B = 22

on = GPIO.LOW
off = GPIO.HIGH
```

Also, we need to initial setup all the LEDs if we want to use the morse code. The purpose of doing this is to clear out the data and reset the LED because sometimes there will be residual instructions unclean, so doing this will clear out the residual data and make sure everything is doing right.



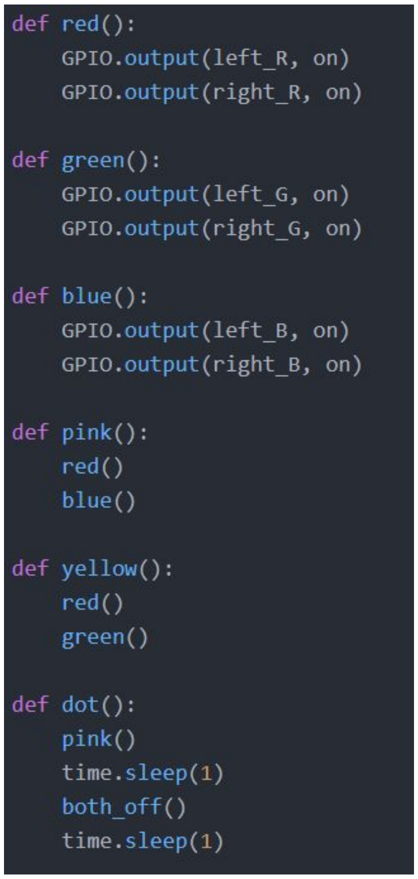
```
def setup():
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(left_R, GPIO.OUT)
    GPIO.setup(right_R, GPIO.OUT)
    GPIO.setup(left_G, GPIO.OUT)
    GPIO.setup(right_G, GPIO.OUT)
    GPIO.setup(left_B, GPIO.OUT)
    GPIO.setup(right_B, GPIO.OUT)
    both_off()
```

After the setup, we still need a turn off the function to turn off everything. If we want to turn off the LED we just set them all off, which is GPIO.LOW.



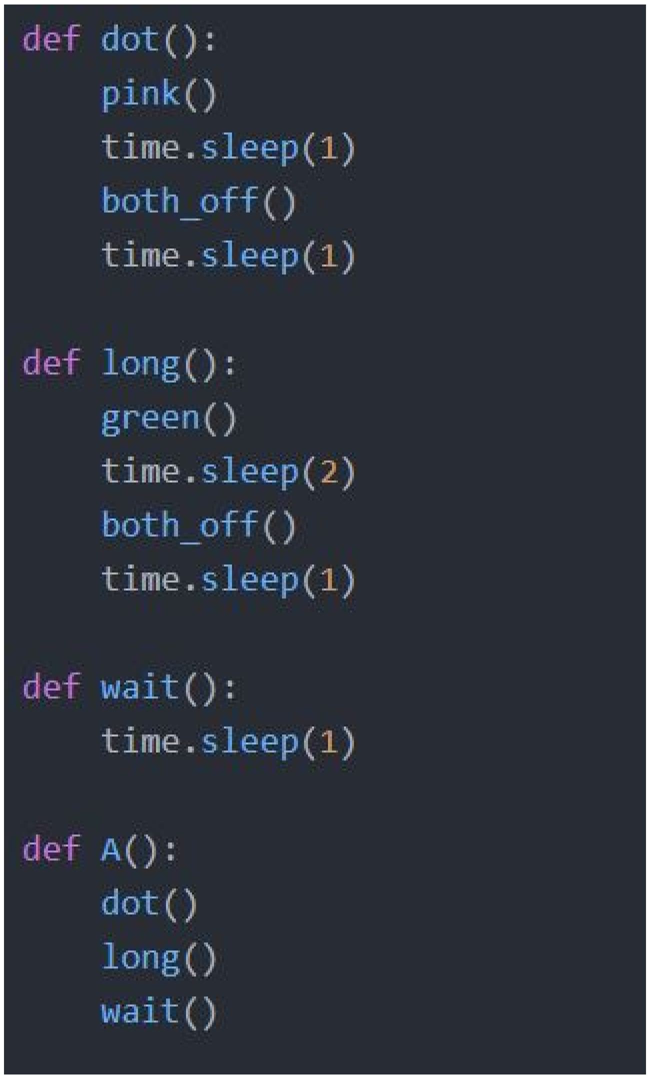
```
def both_off():
    GPIO.output(left_R, off)
    GPIO.output(right_R, off)
    GPIO.output(left_G, off)
    GPIO.output(right_G, off)
    GPIO.output(left_B, off)
    GPIO.output(right_B, off)
```

If we want the simply RGB colour function, we just simply use the original colour. If we want some other colours like pink and we need to mix red and blue together, and that will make a pink colour [11]. The yellow colour is mixed with red and green, and if want some other colour just mix them together and will get that colour you want [11]. There is some colour not really the same as the original colour because there is a limitation on the LED module, but most majority colour will able to mix up.



```
def red():
    GPIO.output(left_R, on)
    GPIO.output(right_R, on)

def green():
    GPIO.output(left_G, on)
    GPIO.output(right_G, on)

def blue():
    GPIO.output(left_B, on)
    GPIO.output(right_B, on)

def pink():
    red()
    blue()

def yellow():
    red()
    green()

def dot():
    pink()
    time.sleep(1)
    both_off()
    time.sleep(1)
```

After that, we need a dot and long function used to represent the morse code. So, the dot we use pink colour to represent and let it wait for 1 second then turn it off. The long will represent a green colour and it will wait for 2 seconds [12]. Also, we need a wait function use for split out every single letter for humans more readable. For example, if we wait to use morse code on letter A, and it will be one dot one long and wait. That is how I use this code to transfer the message to other people remotely, and it is how I use it in our presentation video.



```
def dot():
    pink()
    time.sleep(1)
    both_off()
    time.sleep(1)

def long():
    green()
    time.sleep(2)
    both_off()
    time.sleep(1)

def wait():
    time.sleep(1)

def A():
    dot()
    long()
    wait()
```

3.2 Discussion

3.2.1 Comparison of Ultrasonic scan and OpenCV model

Ultrasonic scan and OpenCV model in there are both designed to detect the distance between the object and Picar itself. But the principle is different between the ultrasonic scan and the OpenCV model [4][5]. The ultrasonic scan is based on the ultrasonic sensor, it can detect all objects in the range without judging what an obstacle is. In addition, we can set the largest range that the PiCar detects. However, Picar can only detect the presence of obstacles in this range, but cannot detect the specific distance between the obstacles and itself. So it can help the PiCar to avoid all obstacles in front and then choose the safest path to walk. But the OpenCV model is based on camera, It can use objects of a specific colour, shape or name as obstacles, only detect the distance between the car and a specific obstacle and ignore other objects that also in the range. Overall, because the detection speed of the ultrasonic scan is faster and this way cannot detect specific obstacles, it is more suitable for situations that need to quickly avoid all obstacles and find the best path. The OpenCV model uses a camera for detection. Although the detection speed is slower than the ultrasonic scan, it will be more accurate and can detect the distance to a specific object. So the OpenCV model is more suitable for tracking or dodging a specific object while ignoring other objects.

4. Challenge

It is undeniable that this research did not directly get the result and we also encountered many challenges. On the process of using SSH to connect to the server remotely, we can not connect successfully, and then we tried this to check all the data settings and found that it still couldn't be run. In the end, we found the problem, we can not connect to the server correctly due to WIFI settings. After changing the WiFi settings, we successfully connected to the server remotely by using ssh.

In our survey of multimedia systems for cars, we found that many installers were not used correctly because some of Python's packets were outdated. And the computer and Raspberry pi can't match because the applications don't match each other. We start with an outdated Python package and look for the original file. After exploring many official websites and technology websites, we found the latest package that can replace outdated Python packages. For the matching problem between the computer and the Raspberry pi, we modified the environmental data according to the existing situation so that it can match.

Meanwhile, we still encountered some unsolved technical problems, but we have tried our best to analyze the reason for these problems. In TCP and IP transmission, we found some lost packets when transmitting real-time detection scenarios. This will cause the transmitted video to have a delay and lose frames. We think we can use the jitter processing method we studied from class and use playback to help reduce the delay and then make video smoother. We are still exploring specific technology solutions to achieve this.

5. Further Exploring

Now that we have learned a lot of background information and related code about the PiCar, and we want to explore more multimedia technologies in the future. According to the results of investigations, the ultrasonic scan is based on the ultrasonic sensor. Although the detection speed of the ultrasonic sensor is quick, can picar detect the fast-moving object surrounding it? We will try to find more information and further research. And also, ultrasonic sensors offer advantages when sensing clear objects such as clear liquor or Highly corrosive liquid [13]. So we want to explore more about the application of Picar to detect high-risk objects. In addition, because the ultrasonic scan and OpenCV model have different advantages in terms of detecting the distance between obstacles and PiCar, we want to further study how to mix the two modules while taking the advantages of the ultrasonic scan and OpenCV model. The submission of this report will not be the end of this project, and we are still researching and testing to get more information about the multimedia technology of PiCar.

6. Conclusion

The PiCar-B is a complete artificial intelligent robot car that combines many multimedia technologies. In this research, we explored the hardware installation of the Picar including Raspberry Pi single-board computer inside. And we chose the adapted software environment package, support application and Numpy library to let the Picar allow us to write a function that makes the camera track the object and detecting the real-world environment. After the basic setting, web analysis and tested several functions that PiCar has: In real-time video transmission, we analyzed how to transfer the real-time images taken by the Raspberry Pi camera to a remote computer. In the remotely controlled by APP, we use the GUI controller Remotely to control the robot through the buttons on the keyboard or the virtual buttons. In the movement of Picar, we explained how to use gear motor and macro servo to control PiCar-B move and look around. In automatic obstacle avoidance, we researched how to avoid obstacles ahead and find the next path based on the ultrasonic sensor. In the object recognition and tracking, we explored how to track objects of a specific shape or colour based on OpenCV. In the LED morse code, we designed and written code to use PiCar led light to delivered morse code. After analyzing the above result, we reviewed all the challenges we met in this research and then try to explore more related information such as how to mix the ultrasonic scan and the OpenCV model in the future.

In conclusion, we studied and analysis the multimedia technology of PiCar-B Mars Rover and related code. We also specifically compared the differences between the ultrasonic scan and the OpenCV model and combined the driverless technology to analyze the corresponding applications in PiCar. Through this research, we get a deep understanding of the practical application in multimedia technology by combining the theoretical knowledge learned from the class with the technologies of Picar-B Mars Rover. And we will keep researching the Further Exploring part to get more technology of PiCar, We also hope that it is helpful to use the ultrasonic scan of PiCar on detecting high-risk objects.

7. References

[1]https://www.adeept.com/adeept-mars-rover-picar-b-wifi-smart-robot-car-kit-for-raspberry-pi-3-model-b-b-2b-speech-recognition-opencv-target-tracking-stem-kit_p0117_s0030.html

[2]https://www.raspberrypi.org/

[3]https://www.ucsusa.org/resources/self-driving-cars-101

[4]https://opencv.org/

[5]https://www.bannerengineering.com/in/en/company/expert-insights/ultrasonic-sensors-101.html

[6]https://www.computerhope.com/jargon/g/gui.htm

[7]https://www.sew-eurodrive.ca/products/gearmotors/gearmotors-3.html#was_ist_ein_getriebemotor

[8]https://www.nxp.com/products/power-management/lighting-driver-and-controller-ics/ic-led-controllers/16-channel-12-bit-pwm-fm-plus-ic-bus-led-controller:PCA9685

[9]https://www.arrow.com/en/research-and-events/articles/ultrasonic-sensors-how-they-work-and-how-to-use-them-with-arduino

[10]https://www.raspberrypi.org/documentation/usage/gpio/

[11]https://randomnerdtutorials.com/electronics-basics-how-do-rgb-leds-work/

[12]https://www.newworldencyclopedia.org/entry/Morse_Code

[13]https://www.rosen-group.com/global/solutions/services/inspection/Corrosion-Detection.html

8. Appendices

8.1 Contribution of team-member

Shuwen Li is the team leader of this project, and she designed and organized the process of the project and have done an online research first. And then she tried to find the adapted environment packages of all the modules of Picar. In addition, she created and updated our project website.

Zijian Chen is the programming support of this project. He installed the hardware part of PiCar and tried to connect the local desktop and PiCar itself. And also, he applied the code of Picar and analysis the background information of PiCar.

We did the weekly meeting and test all the multimedia technologies of PiCar together. And through the experience of completing this project, we get a deeper understanding of multimedia systems.

8.2 Website of This Project

The URL of the website is https://helenli99.github.io.

And also we update weekly reports and related code on the Project detail page.

8.3 Timeline

	Oct 6 -Oct 12	Oct 13 -Oct 19	Oct 20 -Oct 26	Oct 27 -Nov 2	Nov 3 -Nov 9	Nov 10 -Nov 16	Nov 17 -Nov 23	Nov 24 -Nov 30
Collecting information	🟥	🟥	🟥					
Updating website	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦
Writing code						🟥	🟥	
Weekly meeting	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦
Explore and improve the technology			🟥	🟥	🟥	🟥		
Writing final report						🟦	🟦	🟦
Making powerpoint							🟥	
Final Editing and submit								🟦

8.4 Summarize of Weekly Report

Time	Weekly Report Number	Comment
Oct 6-Oct 19	Weekly Report one	Finish the basic survey, but change write coding to find suitable application from official website
Oct 20 -Oct 26	Weekly Report two	Finish the assblem part of the Picar-B we build up it and install the system into the Raspberry pi.
Oct 27 -Nov 2	Weekly Report three	Finish the installation of required software and python library
Nov 3 -Nov 9	Weekly Report four	Test out the Ultrasonic sensor and use the code provided for us and study for it.
Nov 10 -Nov 16	Weekly Report five	Test out the OpenCV and study and edit the code they provide for us, and we may add some new functions on it or improve the function.