UNIVERSITY OF CALIFORNIA SAN DIEGO

COMPUTER SCIENCE AND ENGINEERING (CSE)

EMBEDDED SYSTEMS DESIGN PROJECT

A smart droid, RD-1

Author:
Ninh Tran.
nkt002@ucsd.edu

Author:
Khanh Pham.
kcpham@ucsd.edu

Author:
Ethan Nagola.
enagola@ucsd.edu

Author:
Andres Bernal.
a3bernal@ucsd.edu

June 2021

Abstract

The project RD-1 is the revolutionary start of implementing modern AI into everyday life. RD-1 is a retrieval robot that takes in a command from the user like ”find me a keyboard” and then processes and executes the command. The robot has self pathing and image recognition as well as a microphone and speakers to communicate its findings with the user. In this paper we will talk about the critical research and steps that went along with building RD-1 as well go into why we started with RD-1 in the first place.

Contents

Chapter 1

Introduction

1.1 Prior Research

The RD-1 started as a general robot that would provide simple assistance to common people. From this we narrowed the idea down to one specific task that would lead the way for future simple assistance tasks to follow. This simple task was for the robot to listen to a retrieval command and then search and find the object based on that command. For this to work we needed a few key things: a robot skeleton as well as a way for the robot to see, communicate, and move around.

1.1.1 Self Balancing Robot

In order to create RD-1 we wanted to find a base robot to build off of that we can connect our sound and camera system to easily and efficiently. After researching the different base robots that we could order and build quickly, we decided upon using ELEGOO Tumbller the self balancing robot (https://www.amazon.com/ELEGOO-Tumbller-Self-Balancing-Compatible-Arduino/dp/B07QWJH77V).The two-wheel self-balancing vehicle has a simple structure, flexible movement, easy driving, and convenient carrying, meeting the needs of energy-saving and environmental protection. We chose this robot because it was compatible with Arduino and worked nicely for our robot's independent motion. We also saw a way that we could implement a Raspberry Pi which would be useful for our communication and computer vision.

1.1.2 Computer Vision

For the vision of the robot we first looked into OpenCV as it is a great source for general object detection. However, we wanted the robot's computer vision to be adaptable as we refined its detection and maybe even limited the objects that it detects. Because of this we ended up using Tensorflow Object Detection which is a more general machine learning algorithm that can be applied on images as arrays in order to detect the images. This worked better than OpenCV because OpenCV is a computer vision specific library. Because Tenserflow is a more general machine learning based model we can train the model and have it adapt more fluidly than if

we were to use OpenCV. We then connected this algorithm to cameras that we added to RD-1 through our Respberry Pi

1.1.3 Voice Recognition

For the voice recognition that the robot would have, we looked into many different already created algorithms such as siri, google Text to Speech, DeepSpeech, and SpeechRecognition 3.8.1 . We ended up using a combination of gTTS(google Text to Speech) and SpeechRecognition 3.8.1 and connected the algorithm to a Bluetooth speaker with a built in Microphone and one extra USB Microphone through our Raspberry Pi.

1.1.4 Pathing

Because we used Tenserflow Object Detection for our general object detection, when it came time to implement the pathing we further used Tenserflow to calculate the distance between the robot to the objects that it detects. Then we made sure that RD-1 was able to move on its own and calculate how far it has moved in order to give the robot the ability to move towards an object once it has found the correct one. We did further research on more complex pathing such as avoiding objects with multi-path learning and pathfinding algorithms but were unable to implement that at this time. As a result, our robot currently needs a clear path towards the object that it is retrieving in order to succeed.

1.2 Contributions

Some key contributions that we have made include: creating an efficient and accurate object detection system with Tensorflow Object Detection, combined voice recognition techniques to create a unique algorithm for RD-1, and created a combination of computer vision, voice recognition, and movement that can be easily implemented into any common day robot. Overall, we believe that RD-1 is the start in the right direction of common day use of Artificial Intelligence and, once made more efficient, can be cheaply and easily made and expanded upon in order to make AI more accessible and applicable in our modern word.

Chapter 2

Technical Material

2.1 Hardware

We use all parts for the self-balancing robot from ELEGOO, such as motors, plastic frames, screws, nuts, and wheels. In addition, we also purchased fundamental control boards such as Arduino Uno, a sensor board, and a motor driver.

Figure 2.1: Parts for the self-balancing robot.

We decide to use Arduino Uno to control all the mechanical parts and read basic sensors to make this droid stand balance. Arduino Uno read data from MPU-6050 (GY-521) 3-Axis Accelerometer and Gyro change to the robot's angle with the ground. The PID algorithm is always executed to keep the robot's angle from the sensor equal to the set angle. Thus, RD-1 is auto self-balancing all the time if we set the set angle is zero degrees. The signal of output PID uses to control two motors by using Driver IC, TB6612FNG, for Dual DC motor. Arduino Uno does all the processing and controlling basic tasks so that RD-1 could work with many fundamental functions.

In order to do the smart tasks such as objects recognizing, robot speaking, and voice recognition, we use Raspberry Pi 4 to handle By separating tasks, basic tasks controlled by Arduino Uno, the Raspberry Pi 4 is free from the tasks that need high priority, such as self-balance or update sensor data. Thus, Raspberry Pi 4 has time to work with heavy tasks.

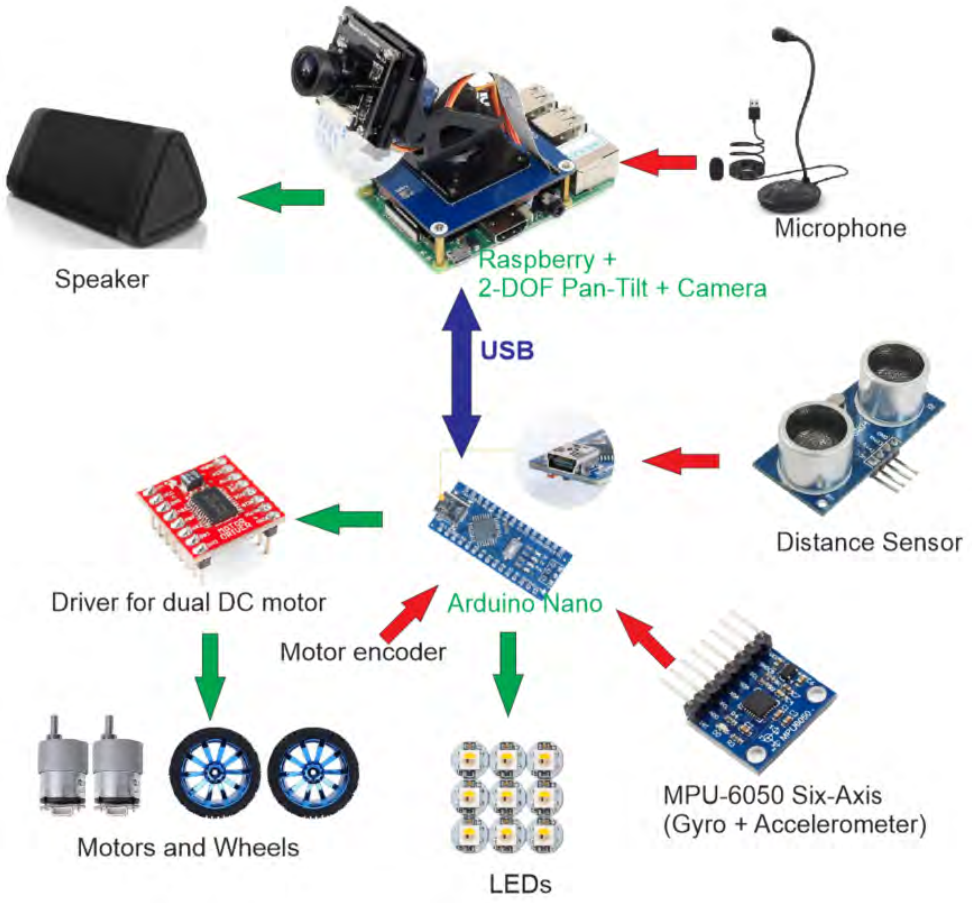

Figure 2.2: Parts for the self-balancing robot.

Raspberry pi can connect to a speaker via Bluetooth or audio 3.5 mm jack. We use a Bluetooth speaker in this project, so RD-1 doesn't need to carry this speaker. In addition, this method can reduce the weight of the robot. The microphone can hook with Raspberry pi via USB port or Bluetooth. We use two microphones in both types so that anyone can talk with RD-1 in various positions in the room.
In Figure2.2, Raspberry pi communicates with Arduino Uno via USB cable to send and receive commands. Thus, all sensor data, the robot's status from Arduino Uno send to Raspberry pi, and the control bytes from Raspberry pi send back to Arduino Uno.

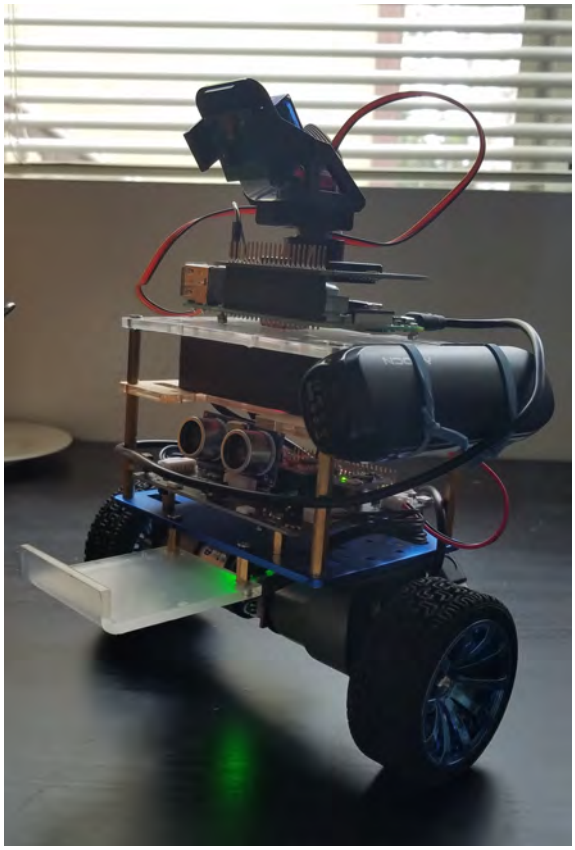

Figure 2.3: The body of the RD-1.

Figure 2.4: The complete hardware for RD-1.

2.2 Essential control functions

Ardunio of RD-1 handles all the essential functions.[1] supports the basic demo code for a two-wheel self-balance robot. Kalman filtering and PID control algorithm were applied to reduce the high-frequency interference of the accelerometer and the low-frequency error of the gyroscope, improve the response speed and control precision of motor to error, and ensure vertical control.

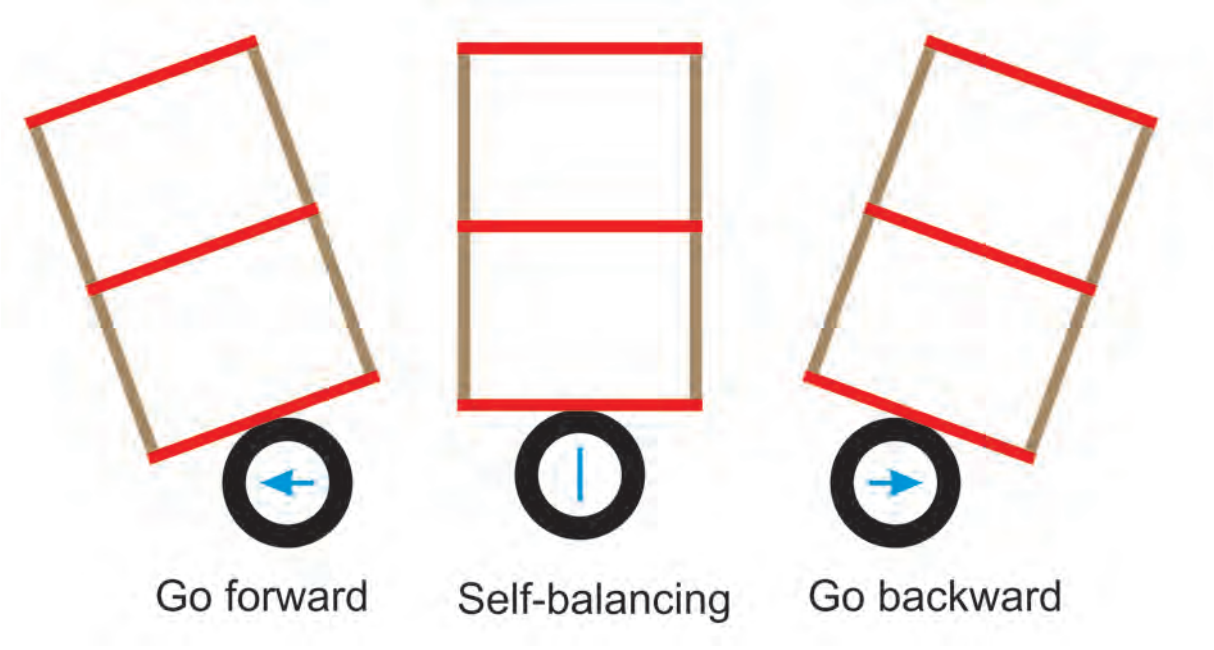

Figure 2.5: The principle of the balancing car.

Figure 2.4 shows how RD-1 could balance or go forward/backward by changing the setting angle.
RD-1 works on dynamic stability in which the gyroscope sensor placed inside the vehicle is used to detect the change of vehicle attitude. MPU-6050 (GY-521) 3-Axis Accelerometer and Gyro must place parallel with the ground surface so that it catches the change of the gravity center of a robot's body. After the fast calculation by an Arduino, RD-1 knows the angle between its plastic surface and the ground surface. The motors are driven to make the balance vehicle advance or retreat, with the turning realized by direction calculation. The servo control system is used to adjust the motor for a balance system. When RD-1 leans forward/backward/turning left/turning right direction, the motor rotate forward/backward/in the left/in the right direction to give the system a forward/backward/turning left/turning right acceleration. When RD-1 reaches equilibrium, motors nearly run in a small position to keep RD-1 standstill in the current position. RD-1 also has a motor encode to measure the distance that it traveled.
Adruino uses a byte as the status to control RD-1. This byte is a command byte receives from a serial port sent from Raspberry Pi. The value of bytes will decide what RD-1 must respond. For instance, ”f” is for moving forward, ”b” is for moving

backward, "i" is for turning right, "l" is for turning left, "s" is for stopping and balance at the current position, "0" and "1" is for following an object, "2" is for moving away from the obstacle, from "3" to "9" is for controlling LED. We also reserve two modes, "∗" and "#" for future development.

```
// Modify code from [1]
bool getserialData(){
  if (Serial.available())
  {
    char c = Serial.read();
    if (c != '\0' && c != '\n')
    {
      key_value = c;
      if (c == 'f' || c == 'b' || c == 'l' || c == 'i')
      {
        function_mode = BLUETOOTH;
      }
      if (c == 's')
      {
        function_mode = IDLE;
      }
      if (key_value == 'f' || key_value == 'b' || key_value == 'l' ||
          key_value == 'i' || key_value == 's' || key_value == '0' ||
          key_value == '1' || key_value == '2' || key_value == '3' ||
          key_value == '4' || key_value == '5' || key_value == '6' ||
          key_value == '7' || key_value == '8' || key_value == '9' ||
          key_value == '*' || key_value == '#')
      {
        return true;
      }
    }
  }
  return false;
}
```

2.3 Raspberry Pi for smart functions

Raspberry Pi 4 work with heavy tasks requiring a longs time to execute, such as computer vision for object recognition, speaking to communicate with a human, and speed recognition to understand a human command.

2.3.1 Voice Recognition and Speaking

RD1 can talk to communicate with a human. We use the library google Text to Speech (gTTS) . It is a light library and the best sound compared with other libraries. The code snippet shows how to use gTTS make RD1 can talk

```
text = "I saw " + text
myobj = gTTS(text=text, lang='en', slow=False)
myobj.save("audio.mp3")
os.system("mpg321 audio.mp3 >/dev/null 2>&1")
```

RD-1 uses library SpeechRecognition 3.8.1, performing speech recognition, with support for several engines and APIs to change speed to text. We use the for loop to find the active microphones so that RD-1 can detect two microphones. Thus, a human can stand at many positions in the room and command what RD-1 should do. The code snippet shows how to use SpeechRecognition to make RD-1 get a text from a human command. The parameter,phras-time-limit, is set to 4 seconds as the default. The timeout parameter is the maximum number of seconds that it will wait for a phrase to start before giving up and throwing a speech recognition.

```
with sr.Microphone(device_index=required) as source:
    audio = r.listen(source, phrase_time_limit = run_time)
try:
    input = r.recognize_google(audio)
```

In some initial tasks, RD-1 receives the commands such as forward/backward/turning left/ turning right direction/ stop while acting by a human. These speeds of humans are changed to the command bytes as "f" for moving forward, "b" for moving backward, "i" for turning right, "l" for turning left, "s" for stopping and balance at the current position. The command byte is sent to Arduino to execute.

2.3.2 Objects Recognition

We wanted the robot's vision to be adaptable as we refined its detection and maybe even limited the objects it detects. The runtime for object recognition execution is about 256 ms. [2] supports the way to test how long a Raspberry Pi executes Tensorflow Object Detection. Figure 2.5 shows the result of the running times.
[3] support the guide that provides step-by-step instructions for setting up TensorFlow's Object Detection API on the Raspberry Pi. By following the steps in this guide, RD-1 can use both live video feeds from a Picamera or USB webcam to perform object detection. [3] also guide how to retrain neural networks to identify specific objects. In order to increase the accuracy of object detection, RD-1 only focuses on

```
pi@raspberrypi:~/exp $ python3 classify.py
classify.py:74: MatplotlibDeprecationWarning:
The set_window_title function was deprecated in Matplotlib 3.4
specific methods instead.
  fig.canvas.set_window_title('TensorFlow Lite')
>>>  48.58  ms ( camera capture )
>>>  136.47  ms ( inference )
>>>  16.08  ms ( preview )
digital clock 0.08627450980392157
********************************
>>>  69.75  ms ( camera capture )
>>>  184.81  ms ( inference )
>>>  16.2  ms ( preview )
matchstick 0.10980392156862745
********************************
>>>  73.05  ms ( camera capture )
>>>  167.97  ms ( inference )
>>>  16.16  ms ( preview )
lipstick 0.13725490196078433
```

Figure 2.6: Running time test.

the common objects in my room, such as a keyboard, a cup, a bow, a backpack, a bottle, or a laptop. All these items must correctly recognize so that RD-1 can find and reach them. Fig 2.7 shows the result of RD-1 eyes. RD-1 can understand what the obstacles are in front of it.
Based on the demo code from [7] that can recognize some objects, We developed code to focus on the robot acknowledge and communicate with Arduino.Because RD-1 works on dynamic stability, that means the camera continuously vibrates even when RD-1 is standstill. We reduce the shuttle time to capture a picture to reduce the blurry. The method is a simple way to increase the accuracy of detecting an object.

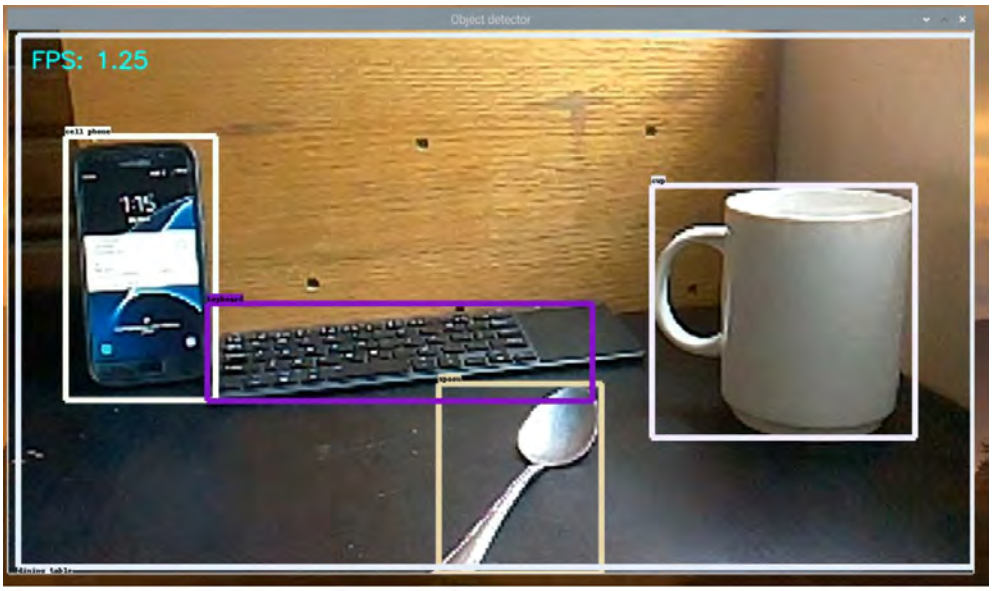

Figure 2.7: Objects recognition.

2.3.3 Complicate commands to find objects

In the limit of the range that RD-1 can move and reach, Rd-1 works well with a distance of fewer than 36 inches. However, the father distance can reduce the accuracy.

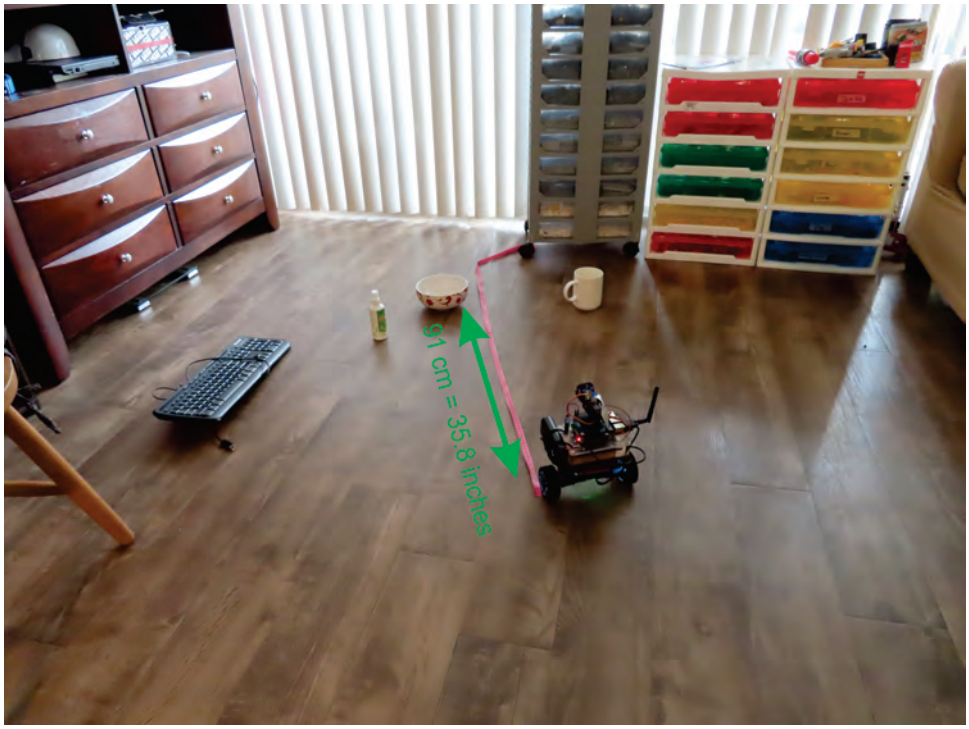

Figure 2.8: The distance with high accuracy.

RD-1 receives the commands to find an object in the room. We demo with four things setting around Rd-1. The human order to find an object. Raspberry uses SpeechRecognition to detect a word be the object in the sentence send a command to Arduino to random rotate left or right. Raspberry waits until the correct thing is detected, then It will move forward to reach the object.

Chapter 3

Milestones

3.1 Task Description and Analysis

The task is to describe how far we go on the project by looking at how well the robot can recognize/label objects, respond to a voice command from the user and move itself to a target from the origin. Our MVP is to have and evaluate the robot move itself from a 1 foot away location to the target after listening and understanding the voice command from a user. To accomplish this particular goal, we will need to look into three main modules: Computer Vision, Audio Command, Motion Planning.

3.2 Achievement

3.2.1 Computer Vision

Goal: The robot can recognize what kind of objects it sees and labels them with their label. In addition, the robot can determine the location of each particular object and determining how far from itself to the target

Done by: Andres, Ethan
Completed: We have completed this goal as expected

Evaluation: The robot can recognize and label objects as long as the objects are within the range of the camera. The time response is under 1 second, which is good enough for a real life task

3.2.2 Audio Command

Goal: The robot can process the voice commands and it should respond to the command by moving itself and making a voice response such as “turn left”, “turn right”, ”move forward”, ”move backward” ...

Done by: Ninh, Khanh
Completed: We have just completed the goal partially. Commands such as ”go pick up the cup”,or ”go to the cup” are still complicated for the robot to understand.

The delay time response is between 2-4 seconds depending on how long the audio command is.

Figure 3.1: Voice command for basic movement

Click HERE to see demo
Evaluation: The robot sometimes cannot recognize the voice because of the noise from surrounding environment, so user needs to repeat the command. Some commands such as ”go to the cup” is more complicated for the robot to understand. It needs to extract the voice to two sub-commands: ”find the cup” and ”move itself to the cup”.

3.2.3 Motion Planning

Goal : The robot can move itself to the target object after listening to the voice command
Done by: The whole group
Completed: We have completed this goal. The time for the robot to recognize the object and start moving to the object is under 4 seconds.

Click HERE to see demo

Evaluation: The robot can do some basic motion commands (left, right, forward, backward). However, with commands that require multiple steps, the goal has not been accomplished yet due to the accuracy from Audio Command module

Figure 3.2: Movement with complicate command

3.3 Plan for the Future

3.3.1 Computer Vision

We will need to increase the accuracy of this module by training the model with smaller number of objects. By doing this, the robot can process and recognize objects much faster.

Our group members, Andres and Ethan will be responsible for this part. We expect to complete this part by the end of week 8.

3.3.2 Audio Command

So far, this module is the key. We will need to process/extract these commands containing multiple stages of processing. If we can do this, we can make the robot execute each stage sequentially so that the robot can execute the command as expected.

Our group members, Ninh and Khanh will be responsible for this part. We expect to complete this part by the end of week 9.

3.3.3 Motion Planning

This module is the output module, which is directly connected and controlled by the Audio Command module. Once we can process the Audio Command module accurately, we can send the output to the Motion Planning so that the robot can understand how it needs to move itself to the target by converting the output from the

audio input to the sequence of steps (move forward-left-right-move forward-target). For this part, the whole group will need to collaborate and test the final product by the end of week 9. We expect to see the robot be able to move itself to the target object after listening to and processing a voice command.

3.3.4 Documentation

Once all goals are achieved, we all have the knowledge of how to build this RD-1 from the scratch. We're supposed to write out final documentation and prepare for the presentation. Everybody in the group is expected to have their best understanding about the parts they have been working on. This task will done by Thursday of week 10

Chapter 4

Conclusion

To reiterate we were able to develop the main features and goals for our RD-1 robot. As it's able to detect object in its proximity , able to receive voice commands given by humans and execute them such as to move a certain direction and it's also and able to repeat the command that its given as confirmation and calls out objects that it sees in its proximity.
Moving forward there's more progress to be done to RD-1 robot so that it can be fully functional to its full potential. There's some work to be on the path planning side so that the robot is autonomous and is able to move on its own without crashing into objects when given a command to find something. Another piece of work to be done is to optimize the time it takes the robot to find an item when given a command to look for something , as of now it's able to find different items but the execution time is not as fast we want it to be.
We believe that with these improvements to be done in the near future RD-1 can become a smart autonomous robot.We also hope that RD-1 robot can become a norm in households to help us accomplish every day task which would make our life easier and save us time.

Bibliography

[1] ELEGOO. https://www.elegoo.com/pages/arduino-kits-support-files.

[2] Jitesh. https://github.com/jiteshsaini/coral_USB_ml_accelerator.

[3] Boping Zhang Guoxi Wu . "Design of two-wheel self-balancing vehicle based on visual identification." EURASIP Journal on Image and Video Processing.

[4] Boping Zhang Guoxi Wu . "Design of twowheel selfbalancing vehicle based on visual identification." EURASIP Journal on Image and Video Processing.

[5] https://pypi.org/project/gTTS/

[6] https://pypi.org/project/SpeechRecognition/

[7] Evan. https://github.com/EdjeElectronics/TensorFlowObjectDetection-on-the-Raspberry-Pi.