

Computer Networks: Final Project

Fall 2020



• **Muhammad Shakeel Zuhaib**
└ Computer Science and Engineering
└ 2017314461

• **Moon Wonjun**
└ Computer Science and Engineering
└ 2017315482

• Contents:

1) Abstract	3
2) Introduction	3
3) Related Work	5
4) Architecture Diagram	8
5) Hardware Implementation	9
6) Software Implementation 1: HSV Detection using OpenCV	13
7) Software Implementation 2: Socket Programming	17
8) Software Implementation 3: Full Integration	19
9) Issues and Future Implications	21
10) Achieved Project Milestone	21
11) Reference List	22

1) Abstract

Over the last couple of years, there has been an increase in people using IV drip stands and going through home oxygen therapies. According to a research done by M. Meena, D. Kewlani and D. Sharma around 15% of elderlies around the world suffer from such health problems [1]. Furthermore, those old patients also go through another serious problem as it is hard for them to pull/push their oxygen tanks or IV drip stands especially those patients who are recommended for bedrest. Concerning this situation, this project aims to develop a Following Assistant Robot (FAR) that can be used for those patients. FAR uses a set of cameras to detect a specific tag's color attached to the patient's back and follows it automatically without the need of any physical force from the user.

2) Introduction:

Object detection and robotics is a popular and a growing field in computer science. The field of robotics has gone through a lot of development making it one of the most integral parts of our current society and the future. Based on these fields, this section will provide an introduction to our project which includes the motivation behind choosing this topic, key idea of this project and the expected results.

Currently, many hospitals changed their system to so-called 'Smart-Hospitals'. But in case of the disabled or extremely sick patients, when they are hospitalized, nurses are

assigned to them explicitly so that they can move their IV drip stands/ oxygen tanks and sometimes the patients are forced to do it by themselves as shown in the following figure.



Fig1. Patient Struggling with his IV Drip Stand

Our team thought that we can develop a robot that can effectively follow the patients wherever they go without the need of exerting any mechanical force. The key idea of this project is making a full system (both hardware and software) that can automate the previously mentioned process. We found several tools that we can use to implement our goal and chose the most appropriate tool for our project which is a simple RPI Camera along with using an OpenCV model [2].

With this project, we are expecting to make a more-efficient way for hospitalized patients' movement. Moreover, we want to offer a very general robot that not only can be used in hospitals but also can be used in shopping carts and future human-following suitcases.

3. Related Work

In this part we will focus on describing the related work that has been done in this area in a more systematic way. It is really essential to mention that most of the research that has targeted this field is either dependent on Statistical Machine Translation (SMT) or Computer Vision. Thus, it is really important to describe those mechanisms before diving into describing their applications.

- 1) Statistical Machine Translation (SMT): It is regarded as a subfield of machine learning which investigates how to automatically classify numerical data using probabilistic models estimated from parallel corpora which needs large quantities of training data. In simple probabilities the formula can be described as follows [3]:

$$\begin{aligned}\hat{T} &= \arg \max_T P(T|S) \\ &= \arg \max_T \frac{P(S|T)P(T)}{P(S)} \\ &= \arg \max_T P(S|T) P(T)\end{aligned}$$

Translation model

Encodes the faithfulness
of T as a translation of S

Language model

Encodes the fluency of T
in the target language

Fig2. Statistical Machine Translation

2) Computer Vision: It is a field of artificial intelligence that trains computers to interpret and understand the visual world. Using digital images from cameras, videos and deep learning models, machines can accurately identify and classify objects — and then react to what they “see.” It is essential to mention that those machines are usually trained with a Convolutional Neural Network described as follows [4]:

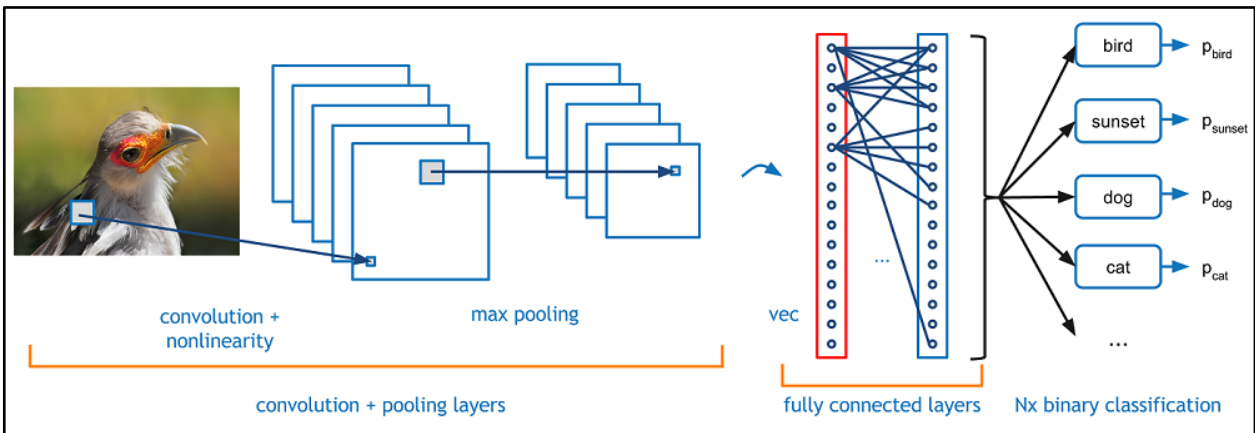


Fig3. Convolutional Neural Networks

One of the most notable works that has been done in this area was published by the University of VIT in an IEEE conference under the title of “A Line Following Robot for Hospital Management” in which they proposed a custom line tracking robot system which uses a specialized colored lines and IR sensors to follow a specific path with an accuracy of 98% [5]. On the other hand, the practicality of such robot is debatable as it is only able to follow lines rather than users which in fact is inconvenient and has very limited uses. Moreover, it uses IR sensors which has no explicit user detection. To compare this paper to our approach, we are inferring the fact that this project only focuses on the automation of robots in hospitals rather than considering actual problems that need to be solved.

Another interesting paper was published in ROBOMECH Journal under the title; “Mobile follower robot as an assistive device for home oxygen therapy – evaluation of tether control algorithms”. In this paper, the approach is quite unique; the users have to be attached to the robot via a tether/cable to make sure the user stays in the sight and range of the ultra-sonic sensors which are used to maintain a certain distance from the patients [6]. On the other hand, this approach has three main flaws:

- 1) The inconvenience of using it as many patients complained about the tether.
- 2) The absence of explicit object or color detection.
- 3) The use of ultra-sonic sensors which have very limited range.

That being said, newer papers have suggested better approaches using advanced CNNs and Machine Learning models. And our approach via FAR is based upon it. In our approach, we are taking the same Computer Vision techniques and enhancing it by focusing on using cameras and an OpenCV model for both object detection and obstacle avoidance. By using such approach we will make sure to provide the maximum convenience and explicit object detection.

4. Architecture Diagram

The main algorithmic workflow for our FAR's implementation is shown in Figure4.

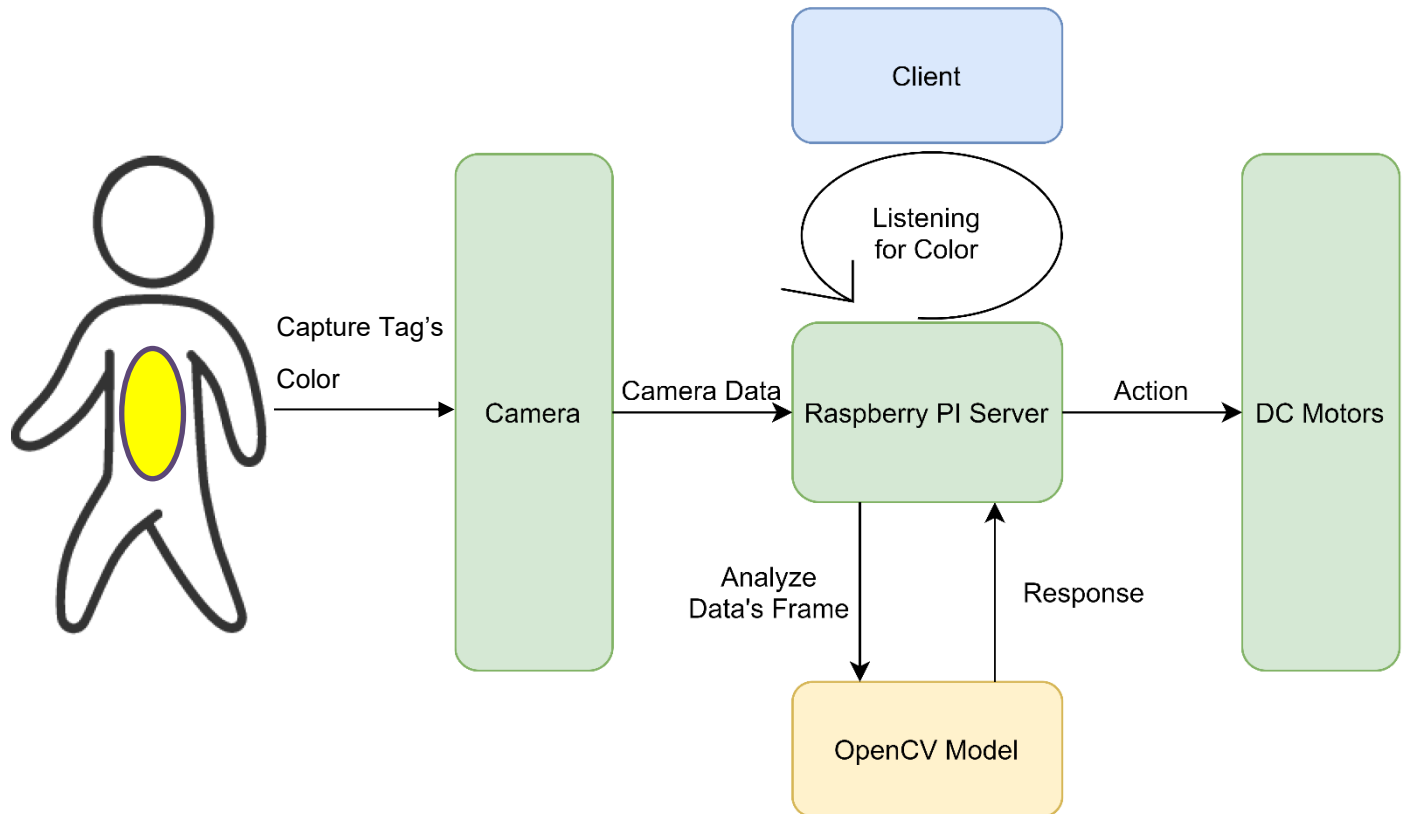


Fig4. Workflow of FAR

As shown in above figure, the Raspberry PI acts as a server which keeps listening until a client connects to it and sends some color to be detected. Once the RPI server receives the color from the client it starts capturing footage using the camera and then analyzes each frame using the running OpenCV model underneath. Since this model can detect the correct object's color (tag color), It returns a response to Raspberry pi to either move the servo motors or not while keeping a fixed distance from the leader object. In case of an obstacle, the OpenCV model will automatically fail and hence the robot will stop following the leader object hence creating a simple obstacle avoidance.

5. Hardware Implementation

5.1. Robot Chassis:

The car chassis was designed in AutoCAD with the following 3D design blue prints.

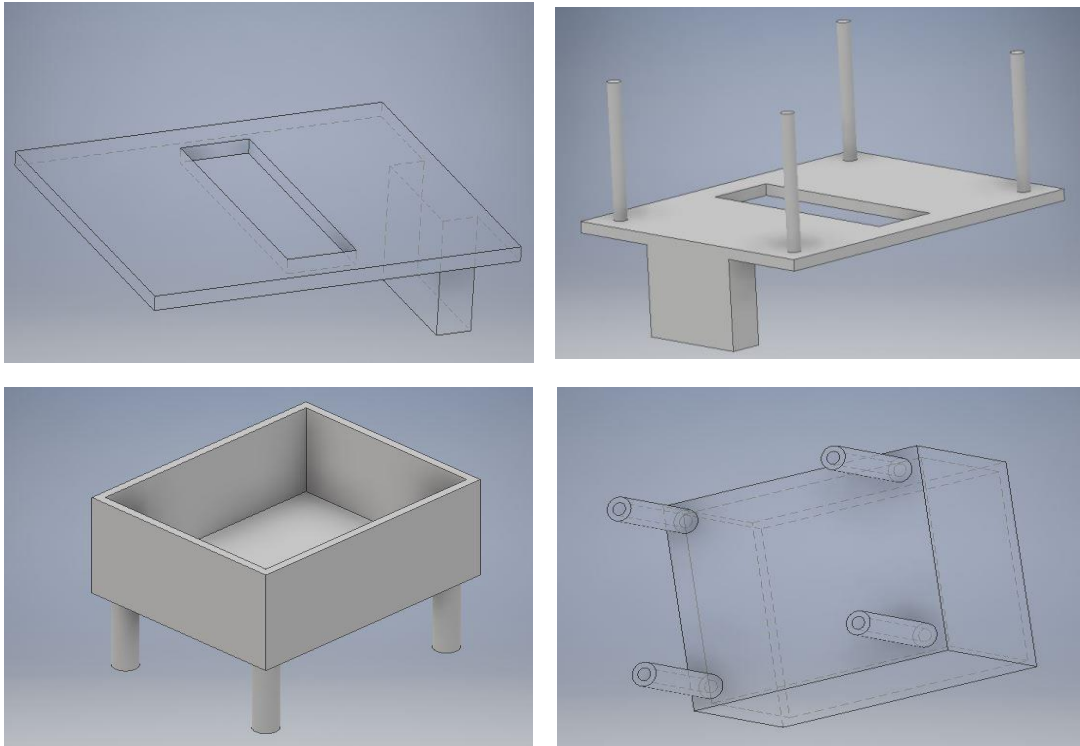



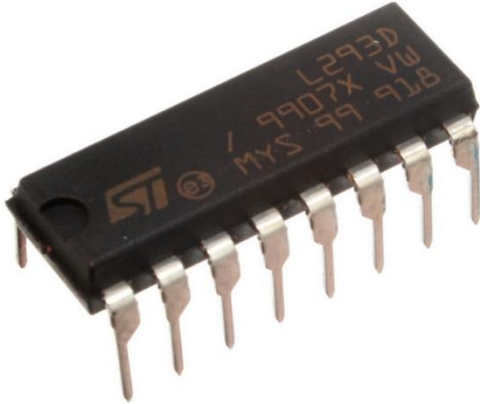
Fig5. AutoCAD 3D Designs of Car Chassis

Here are the results of the 3D printing using Acrylic plastic.



Fig6. 3D Printed Parts

5.2. Used Parts:

Part	Picture	Purpose
Raspberry Pi 3		Acts as a server for socket connection and is the main brain for running the OpenCV model and FAR's algorithm.
L293D Driver		Provides a simple interface to assign a separate power supply for the motors. Also, it enables the RPI to control the motors simultaneously.

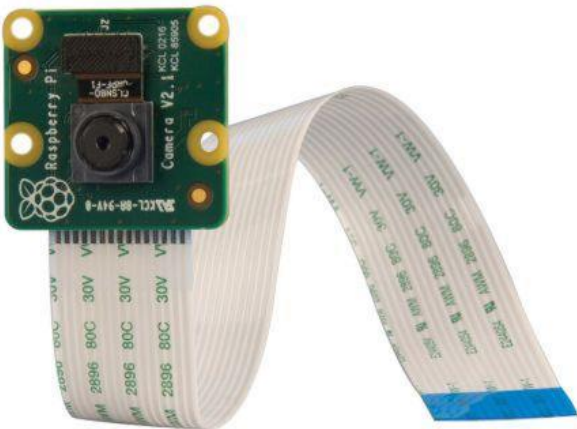

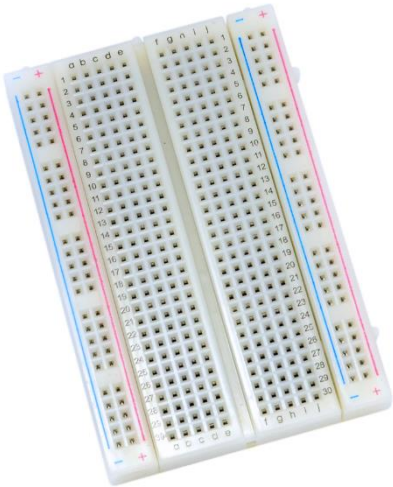
<p>Raspberry Pi Camera Module V2</p>		<p>1080p recording of the scenario.</p>
<p>DC Motors</p>		<p>360 degrees bidirectional movement</p>
<p>Peripherals (Breadboard, Batteries, Wires, etc.)</p>		<p>Power supply provision and cable connections.</p>

Table1. Used Parts

5.3. Circuit Diagram and Assembly:

The circuit diagram for our project is quite simple to follow as shown in Figure 7. What is more important is to notice that an external power supply is used to power both DC motors.

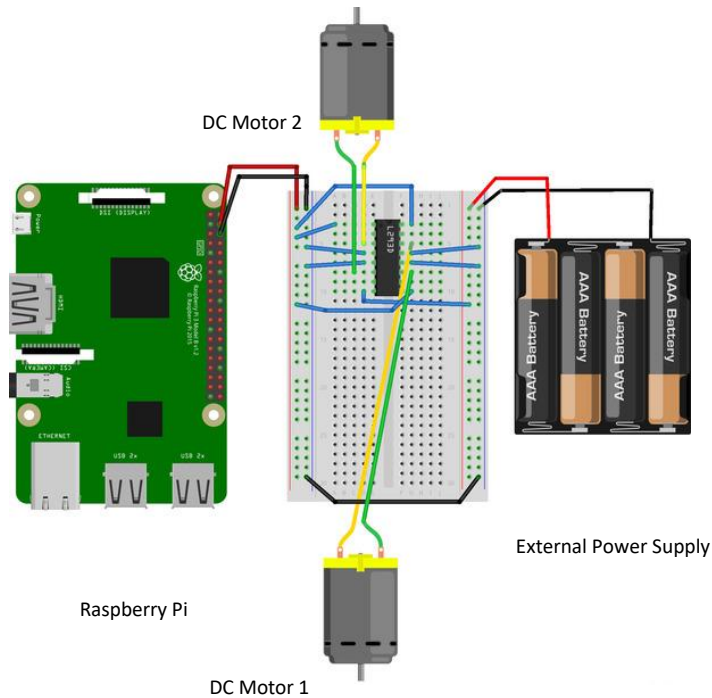


Fig7. Circuit Diagram

The final assembly of all parts is shown in the following figures.

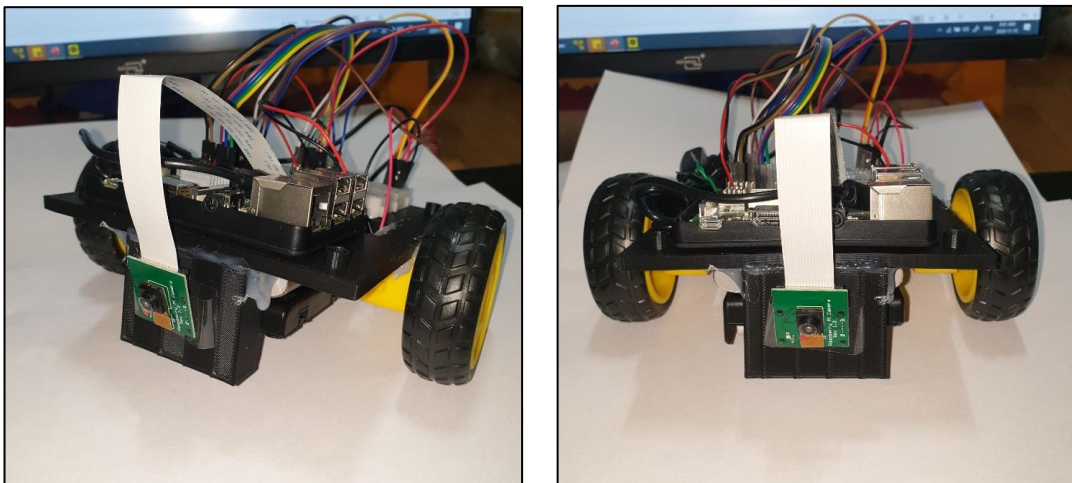


Fig8. Final Assembly

6. Software Implementation 1: HSV Detection Using OpenCV

6.1. OpenCV:

Most machine learning frameworks including OpenCV use HSV (Hue, Saturation, Value) color scheme for color detection. The reason behind this is that a single value of Hue is needed to represent the color while saturation only represents the shades of grey color and Value represents the intensity of the color hence these two parameters can be ignored as shown in Figure9.

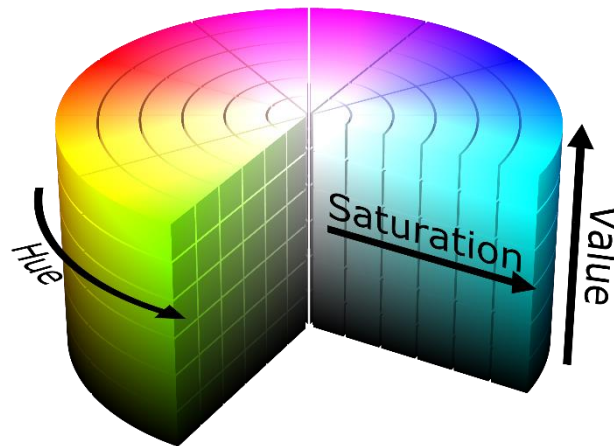


Fig9. HSV Color Space

On the other hand, since RGB color space is coded using three channels as shown in Figure10 it is more difficult to segment an object based on its color.

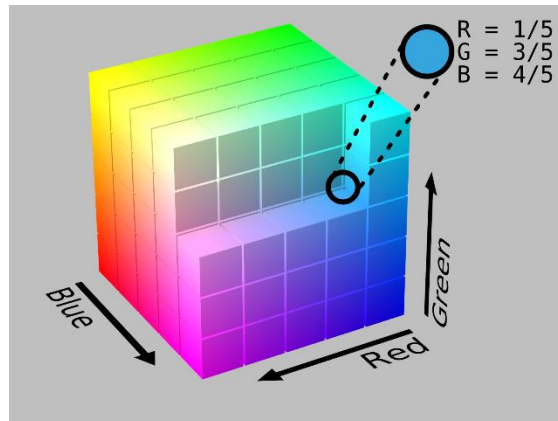


Fig10. RGB Color Space

6.2. Algorithm Description:

We used different colored balls to build the prototype and to figure out their HUE value we used OpenCV. This library provides two simple APIs that can be used for extracting our differently colored objects from the captured footage:

- a) `cvtColor()`: This API is used to convert the RGB captured footage to HSV color space as shown in Figure 11.

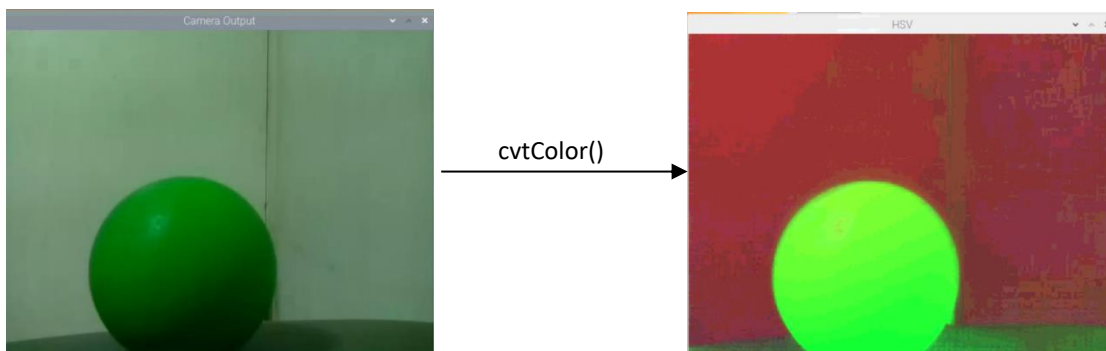


Fig11. RGB to HSV Conversion

- b) `inRange()`: This API applies a threshold to the HSV frame so that only the colors within a specific range are masked as shown in Figure 12.

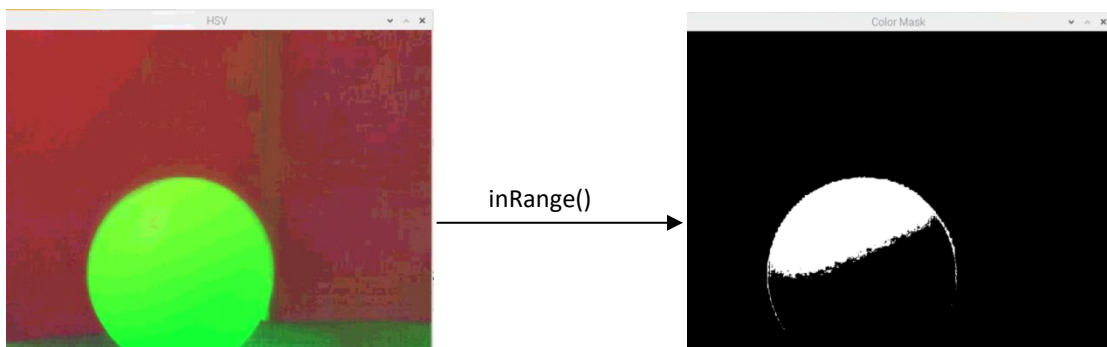


Fig12. Masking of HSV Frame

c) `findContours()`: This API finds 4 important information about each mask (the white area) which are (boundaries, area, position_x, position_y). It is essential to mention that FAR's algorithm which will be explained in a later section uses the area and the coordinates of the position to make a decision regarding whether the robot should move or not. The following figure gives an idea about the result of this function.

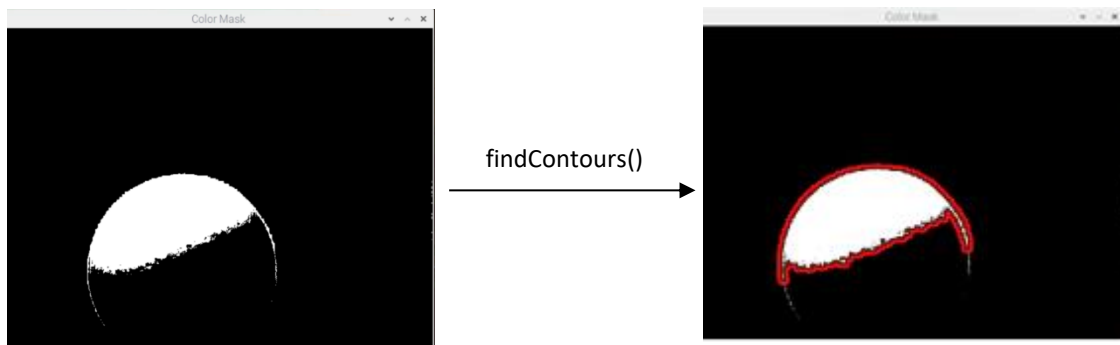


Fig13. Finding Contours of Mask

6.3. Code:

Using the above mentioned description we were able to identify the Hue value for the balls used in the making of the prototype and here is the implementation code for this part.

```
1 # import the necessary packages
2 from picamera.array import PiRGBArray
3 from picamera import PiCamera
4 import time
5 import cv2
6 import numpy as np
7
8
9 # initialize the camera and grab a reference to the raw camera capture
10 camera = PiCamera()
11 camera.resolution = (640, 480)
12 camera.framerate = 32
13 camera.rotation=180
14 rawCapture = PiRGBArray(camera, size=(640, 480))
15
16 while True:
17     while True:
18         try:
19             hue_value = int(input("Hue value between 10 and 245: "))
20             if (hue_value < 10) or (hue_value > 245):
21                 raise ValueError
22         except ValueError:
23             print("That isn't an integer between 10 and 245, try again")
24
25     lower_red = np.array([hue_value-10,100,100])
26     upper_red = np.array([hue_value+10, 255, 255])
27
28     for frame in camera.capture_continuous(rawCapture, format="bgr",
29         use_video_port=True):
30         image = frame.array
31
32         hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
33
34         color_mask = cv2.inRange(hsv, lower_red, upper_red)
35
36         result = cv2.bitwise_and(image, image, mask= color_mask)
37
38         cv2.imshow("Camera Output", image)
39         cv2.imshow("HSV", hsv)
40         cv2.imshow("Color Mask", color_mask)
41         cv2.imshow("Final Result", result)
42
43         rawCapture.truncate(0)
44
45         k = cv2.waitKey(5) && 0xFF
46         if "q" == chr(k & 255):
47             break
```

Code1. HSV Test

7. Software Implementation 2: Socket Programming

7.1. Algorithm Description:

Our second goal to accomplish after testing OpenCV was to implement a simple TCP/IP server where RPI acts as a server and waits for a client to send a specific message and in our case that message is the color to be followed and its HUE Value. In our implementation, we used a Linux client running on a virtual machine which connects to RPI Server using an IP address and a port number. To be more specific, the details of the workflow can be seen in Figure 14.

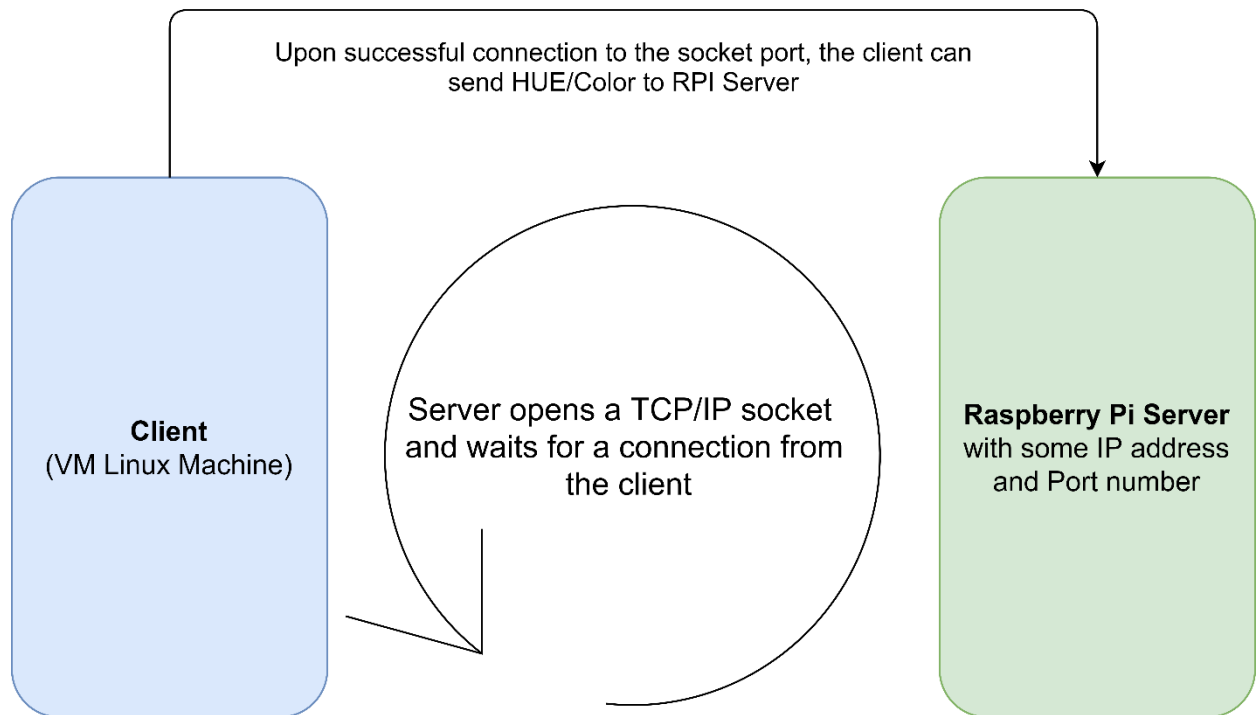


Fig14. TCP/IP Implementation

7.2. Code:

Here is the code for the Raspberry Pi server. The server was simply implemented using TCP/IP sockets in python as the following.

```
1 import socket
2 import re
3
4 s = socket.socket()
5 host = "192.168.219.156"
6 port = 12338
7 s.bind((host, port))
8 s.listen(5)
9
10 while True:
11     try:
12         clientsock, addr = s.accept()
13     except OSError:
14         continue
15     message = clientsock.recv(1000)
16     color=message.decode("utf-8")
17
18     val=re.findall('\d+', color)
19
20     print (val[0])
21
22     clientsock.close()
```

Code2. Server Side Code

Furthermore, here is the code for the Virtual Linux Machine client.

```
1 import socket
2
3 s = socket.socket()
4 host = "192.168.219.156"
5 port = 12338
6 s.connect((host, port))
7 output = 'Green HUE:64'
8 print(output," is sent")
9 s.sendall(output.encode('utf-8'))
10
11 s.close()
```

Code3. Client Side Code

8. Software Implementation 3: Full Integration

8.1. Workflow of FAR:

The workflow of FAR's algorithm can be represented in the following figure.

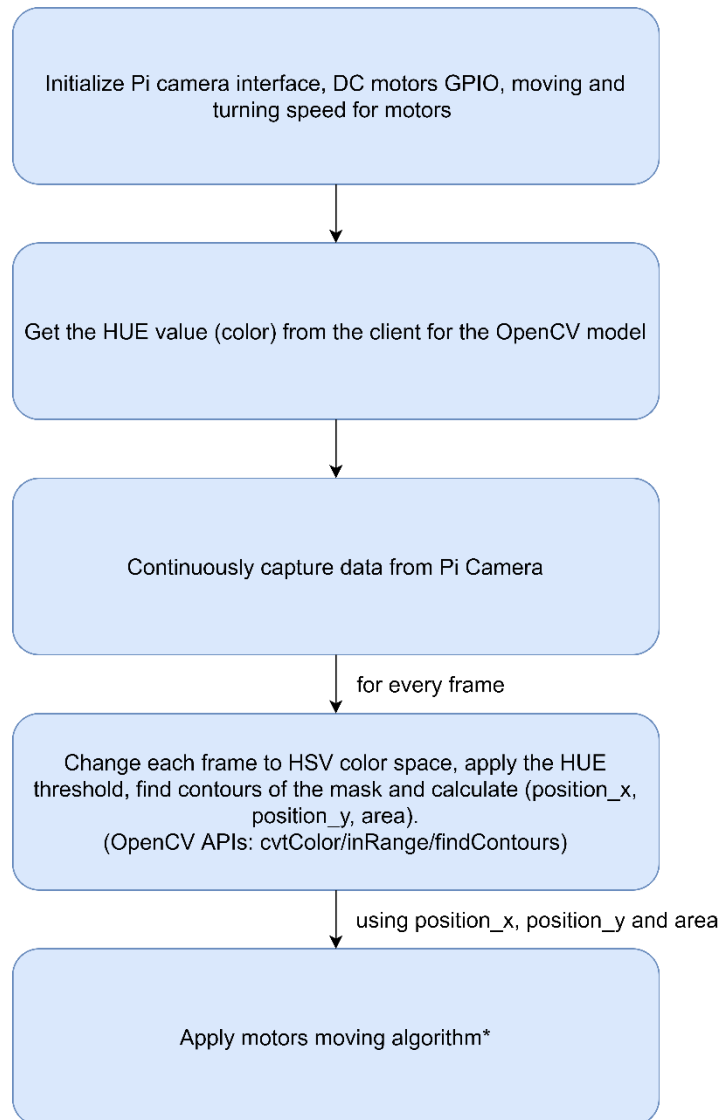


Fig15. FAR's Algorithm

8.2. Motors Moving Algorithm:

Using (position_x, position_y, area) retrieved from the OpenCV model, FAR takes a decision based on:

- a) A certain preset threshold (area_threshold): if the area is greater or equal to area_threshold that means the object is close enough and the robot stops.
- b) position_y's location: if the Y coordinate of the mask lies in the right portion of the frame then the robot turns right, if it lies in the left portion of the frame then robot turns left and if it lies in the middle portion the robot moves forward.

In the case of the “No Object” or “Incorrect Object”, the OpenCV will fail to generate a mask which will result in the robot moving 360 degrees around itself so that it continues searching for the object. To summarize there are 5 scenarios as shown Figure16.

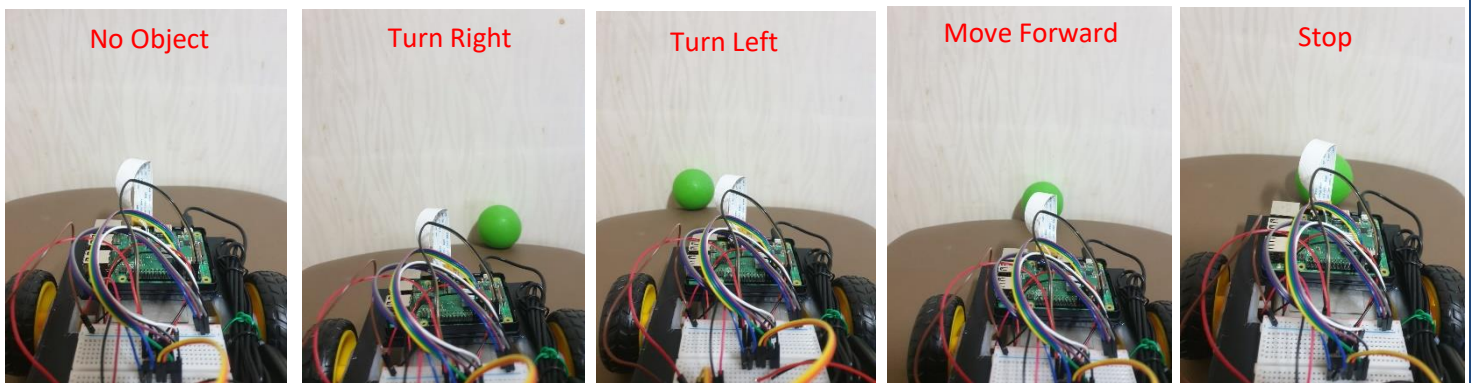


Fig16. Motors Moving Algorithm

9. Issues and Future Implications

There are a few issues with FAR that can be fixed in the future. To start with, we are using 1080p low quality camera which makes masking and HUE detection of the object sometimes tough especially in harsh lighting conditions. Moreover, FAR doesn't have a complex obstacle avoidance system thus some ultra-sonic sensors, better cameras and LIDAR can be used to implement the algorithm more accurately or even implement more complex features such as room mapping. That being said, the results achieved currently are extremely promising showing that FAR can potentially be a viable solution in the market.

10. Achieved Project Milestone

Phase	Achieved Deadline
Conceptualization & Survey & Proposal	10/10
Project Proposal Presentation & User Requirements Documentation	16/10
Environmental Setting & Design 1	30/10
Design 2	10/11
Development of the Backend	1/12
System Integration & Deployment & Testing	10/12

11. Reference List

- [1] M. Meena, R. Dixit, J. Kewlani, S. Kumar, S. Harish, D. Sharma, and M. Singh, "Home-based long-term oxygen therapy and oxygen conservation devices: An updated review," *National Journal of Physiology, Pharmacy and Pharmacology*, vol. 5, no. 4, p. 267, 2015.
- [2] Brahmbhatt, S. (2013). Introduction to Computer Vision and OpenCV. Practical OpenCV, 3-5. doi:10.1007/978-1-4302-6080-6_1
- [3] P. Lison, "Introduction to Statistical Machine Translation," *Maskinl ring meetup at Language Technology Group of University of Oslo*, May 2016.
- [4] "Parallel architectures and computer vision," *Computer Vision, Graphics, and Image Processing*, vol. 46, no. 1, p. 136, 1989.
- [5] Shrim, L. (2019). Case Study: A Line Following Robot for Hospital Management.
- [6] Endo, G., Allan, B., Iemura, Y., Fukushima, E. F., Iribe, M., Takubo, T., & Ohira, M. (2015). Mobile follower robot as an assistive device for home oxygen therapy – evaluation of tether control algorithms. *ROBOMECH Journal*, 2(1). doi:10.1186/s40648-014-0026-3.