**REDHAAN REPORT**

**Introduction:**

Removing shoes in public places like temples and mosques is a mess and many people lose their shoes at those places. And also, in countries like Japan and China, it is their culture to remove shoes at their workplace. To make this task simple and efficient, we have come up with the idea ***“smart shoe rack”.***

The project name “***Smart shoe”*** rack Smart shoe rack is a robot. The foremost idea of the bot is to store the shoes of the person who visits public places and to return the shoes whenever the person returns after completion of the work.

Smart shoe rack bot features a camera used to detect the face and store the data in the database. After recognizing the data of the person then it picks up the shoes from the person and then stores the shoes of the person in the respective shoe rack which is vacant, where we use a line follower bot mechanism to move the bot from person to person rack. When the person returns for collecting the shoes it checks for the data in the database and then collects the shoes from the rack and gives it to the person.

An ESP32 camera is used which is used to recognize the image of the person visiting the public places and security is also provided to the shoes until he collects back his shoes from the smart shoe rack.

**Components Required:**

**Hardware:**

* ESP32
* channel relay
* Rack and Pinion
* L298N-1
* Side shaft motors
* Ultrasonic sensors
* IR sensors

**SOFTWARE:**

* Visual studio code
* Platformio

**COMPONENTS DESCRIPTION:**

**HARDWARE:**

**ESP32:**

The ESP32-CAM is a very small camera module with the ESP32-S chip, Besides the OV2640 camera, and several GPIOs to connect peripherals, it also features a microSD card slot that can be useful to store images taken with the camera or to store files to serve clients.

****

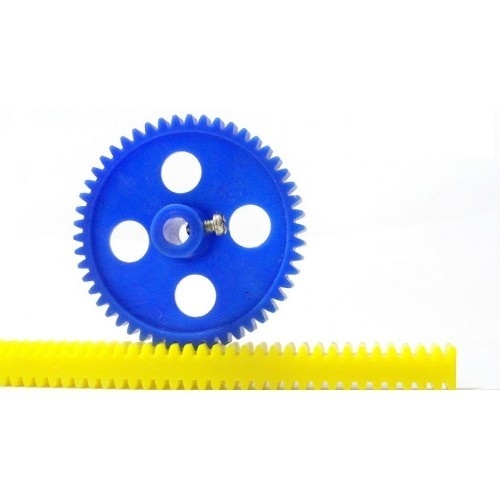
**channel relay:**

A relay is an electrically operated device. It has a control system (also called input circuit or input contactor) and a controlled system (also called output circuit ). It is frequently used in automatic control circuits. To put it simply, it is an automatic switch to controlling a high-current circuit with a low-current signal.

****

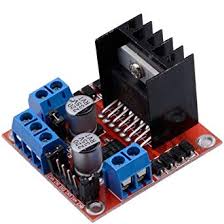
**Rack and Pinion:**

A rack and pinion is a type of linear actuator that comprises a circular gear (the pinion) engaging a linear gear (the rack), which operates to translate rotational motion into linear motion. Driving the pinion into rotation causes the rack to be driven linearly. Driving the rack linearly will cause the pinion to be driven into a rotation.



**L298N:**

The L298N is an integrated monolithic circuit in a 15- lead Multi watt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic level sand drive inductive loads such as relays, solenoids, DC, and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected and the corresponding external terminal can be used for the connection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

****

**Side shaft motors:**

200 RPM Side Shaft Heavy Duty DC Gear Motor is suitable for large robots/automation systems. It has sturdy construction with a gearbox built to handle stall torque produced by the motor. The driveshaft is supported from both sides with metal bushes. The motor runs smoothly from 4V to 12V and gives 200 RPM at 12V. The motor has an 8mm diameter, 17.5mm length drive shaft with a D shape for excellent coupling.



**Ultrasonic sensors:**

As the name indicates, ultrasonic sensors measure distance by using ultrasonic waves.

The sensor head emits an ultrasonic wave and receives the wave reflected from the target. Ultrasonic Sensors measure the distance to the target by measuring the time between the emission and reception.

Outline and detection principle

An optical sensor has a transmitter and receiver, whereas an ultrasonic sensor uses a single ultrasonic element for both emission and reception. In a reflective model ultrasonic sensor, a single oscillator emits and receives ultrasonic waves alternately. This enables miniaturization of the sensor head.



**IR sensors:**

An infrared sensor is an electronic device, that emits to sense some aspects of the surroundings. An IR sensor can measure the heat of an object as well as to detect motion. These types of sensors measure only infrared radiation, rather than emitting it which is called a passive IR sensor. Usually, in the infrared spectrum, all the objects radiate some form of thermal radiation. These types of radiations are invisible to our eyes, that can be detected by an infrared sensor. The emitter is simply an IR LED (Light Emitting Diode) and the detector is simply an IR photodiode that is sensitive to IR light of the same wavelength as that emitted by the IR LED. When IR light falls on the photodiode, the resistances and the output voltages will change in proportion to the magnitude of the IR light received.



**SOFTWARE:**

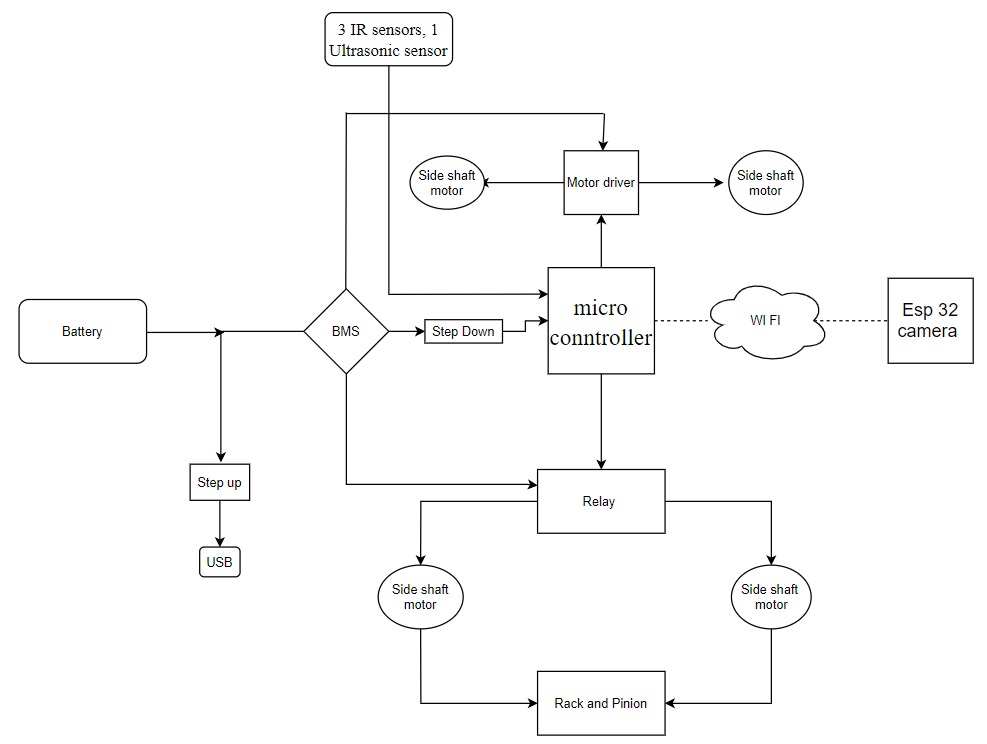
**Visual studio code:**

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS, and Linux. It comes with built-in support for JavaScript, TypeScript, and Node.js and has a rich ecosystem of extensions for other languages (such as C++, C#, Java, Python, PHP, Go) and runtimes (such as .NET and Unity).

**Platform:**

This is a cross-platform code builder and library manager with Arduino or MBED support platforms. They took care of toolchains, debuggers, frameworks that work on most popular platforms like Windows, Mac, and Linux.

**Circuit Diagram or Block Diagram:**



**Code:**

import cv2

import NumPy as np

import pickle

import os

import time

from copy import copy

from PIL import Image

import webbrowser

counter = 0

frame\_counter=0

currentName = ''

name = "no"

rack = {1:'',2:'',3:''}

pname = 1

face\_cas = cv2.CascadeClassifier('C:\my files\Langages\python\python atom\haascascade\haarcascades\haarcascade\_frontalface\_default.xml')

recognizer = cv2.face.LBPHFaceRecognizer\_create()

i=1

folder = "images/id"

recognizer.read("trainner.yml")

lables = {}

with open("lables.pickle", 'rb') as f:

lables = pickle.load(f)

lables = {v:k for k,v in lables.items()}

print(lables.items())

cap = cv2.VideoCapture(0)

while(1):

if frame\_counter > 200:

frame\_counter = 0

currentName=''

ret,frame = cap.read()

frame\_copy = copy(frame)

gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

faces = face\_cas.detectMultiScale(gray,1.3,5)

for (x,y,w,h) in faces:

roi\_gray = gray[y:y+h, x:x+w]

roi\_color = frame[y:y+h, x:x+w]

cv2.rectangle(frame, (x,y), (x+w,y+h), (255,255,0),2)

id\_,conf = recognizer.predict(roi\_gray)

if(conf>=45 and conf<=85):

#print(lables[id\_])

font = cv2.FONT\_HERSHEY\_DUPLEX

name = lables[id\_]

cv2.putText(frame, name,(x,y), font, 1, (0,0,255), 2 )

counter = 1

if currentName!= name:

currentName = name

for x,y in rack.items():

if name == y:

if(x==1):

webbrowser.open('192.168.4.1/one?')

rack[1]=""

print(1)

elif(x==2):

webbrowser.open('192.168.4.1/two?')

print(2)

rack[2]=""

elif(x==3):

webbrowser.open('192.168.4.1/three?')

print(3)

rack[3]=""

else:

if rack[1]=="":

webbrowser.open('192.168.4.1/one?')

print(1)

rack[1]=currentName

elif rack[2]=="":

webbrowser.open('192.168.4.1/two?')

print(2)

rack[2]=currentName

elif rack[3]=="":

webbrowser.open('192.168.4.1/three?')

print(3)

rack[3]=currentName

print(rack)

else:

print(counter)

counter+=1

try:

if counter>=20:

#time.sleep(2)

loc = folder + str(i)

os.mkdir(loc)

imgloc = loc+"/s.jpg"

cv2.imwrite(imgloc, frame\_copy)

i+=1

base\_dir = os.path.dirname(os.path.abspath(\_file\_))

image\_dir = os.path.join(base\_dir, "images")

tface\_cas = cv2.CascadeClassifier('haarcascadeFiles\haarcascade\_frontalface\_default.xml')

trecognizer = cv2.face.LBPHFaceRecognizer\_create()

x\_train = []

y\_lables = []

current\_id = 0

lable\_ids = {}

for root, dirs, files in os.walk(image\_dir):

for file in files:

if file.endswith("png") or file.endswith("jpg"):

path = os.path.join(root, file)

lable = os.path.basename(root).replace(" ", "-").lower()

if not lable in lable\_ids:

lable\_ids[lable]= current\_id

current\_id +=1

id\_ = lable\_ids[lable]

#print(lable\_ids)

pil\_image = Image.open(path).convert("L")

image\_array = np.array(pil\_image, "uint8")

#print(image\_array)

faces = tface\_cas.detectMultiScale(image\_array,1.5,5)

for (x,y,w,h) in faces:

roi = image\_array[y:y+h, x:x+w]

x\_train.append(roi)

y\_lables.append(id\_)

with open("lables.pickle", 'wb') as f:

pickle.dump(lable\_ids, f)

trecognizer.train(x\_train, np.array(y\_lables))

trecognizer.save("trainner.yml")

#time.sleep(2)

counter = 1

except(Exception):

pass

recognizer.read("trainner.yml")

lables = {}

with open("lables.pickle", 'rb') as f:

lables = pickle.load(f)

lables = {v:k for k,v in lables.items()}

print(lables.items())

cv2.imshow('vdo', frame)

frame\_counter+=1

print(frame\_counter)

print(currentName)

if(cv2.waitKey(1) == ord('q')):

break

cap.release()

cv2.destroyAllWindows()

#include <Arduino.h>

#define ir1 1

#define ir2 2

#define ir3 3

#define lf 4

#define lb 5

#define rf 6

#define rb 7

#define trig 8

#define echo 9

#define up 10

#define down 11

void getData();

void goHome();

void goTo(int);

boolean shoes();

void lower();

void rise();

void forward();

void backword();

void right();

void left();

void stop();

void setup() {

pinMode(ir1,INPUT);

pinMode(ir2,INPUT);

pinMode(ir3,INPUT);

pinMode(lf,OUTPUT);

pinMode(lb,OUTPUT);

pinMode(rf,OUTPUT);

pinMode(rb,OUTPUT);

pinMode(trig,OUTPUT);

pinMode(echo,INPUT);

pinMode(up,OUTPUT);

pinMode(down,OUTPUT);

Serial.begin(115200);

}

void loop()

{

Serial.println("loop");

getData();

}

void getData()

{

goTo(1);

}

void goHome()

{

backword();

delay(200);

while(digitalRead(1) >0 && digitalRead(2)<0 && digitalRead(3)>0)

{

if(digitalRead(1) <0 && digitalRead(2)>0 && digitalRead(3)<0)

{

forward();

}

if(digitalRead(1) >0 && digitalRead(2)<0 && digitalRead(3)<0)

{

left();

}

if(digitalRead(1) <0 && digitalRead(2)<0 && digitalRead(3)>0)

{

right();

}

if( digitalRead(2)>0 && digitalRead(3)>0)

{

right();

}

}

lower();

stop();

delay(500);

}

void goTo(int x)

{

int c=1;

if(shoes())

rise();

while(digitalRead(1) >0 && digitalRead(2)<0 && digitalRead(3)>0)

{

if(digitalRead(1) <0 && digitalRead(2)>0 && digitalRead(3)<0)

{

forward();

}

if(digitalRead(1) >0 && digitalRead(2)<0 && digitalRead(3)<0)

{

left();

}

if(digitalRead(1) <0 && digitalRead(2)<0 && digitalRead(3)>0)

{

right();

}

if(digitalRead(1) <0 && digitalRead(2)>0 && digitalRead(3)>0)

{

if(c==x)

{

right();

}

else

{

c++;

forward();

}

}

}

stop();

delay(500);

if(shoes())

lower();

else

rise();

goHome();

}

boolean shoes()

{

digitalWrite(trig,HIGH);

delay(100);

digitalWrite(trig,LOW);

delay(20);

int t = pulseIn(echo,HIGH);

int dist = t\*0.034/2;

Serial.println(dist);

if(t<10)

return true;

else

return false;

}

void lower()

{

Serial.println("lower");

digitalWrite(down,HIGH);

delay(2000);

digitalWrite(down,LOW);

}

void rise()

{

Serial.println("rise");

digitalWrite(up,HIGH);

delay(2000);

digitalWrite(up,LOW);

}

void forward()

{

Serial.println("forward");

digitalWrite(rb,LOW);

digitalWrite(lb,LOW);

digitalWrite(rf,HIGH);

digitalWrite(lf,HIGH);

}

void backword()

{

Serial.println("backword");

digitalWrite(rf,LOW);

digitalWrite(lf,LOW);

digitalWrite(rb,HIGH);

digitalWrite(lb,HIGH);

}

void right()

{

Serial.println("right");

digitalWrite(rb,LOW);

digitalWrite(lb,LOW);

digitalWrite(rf,LOW);

digitalWrite(lf,HIGH);

}

void left()

{

Serial.println("left");

digitalWrite(rb,LOW);

digitalWrite(lb,LOW);

digitalWrite(rf,HIGH);

digitalWrite(lf,LOW);

}

void stop()

{

Serial.println("stop");

digitalWrite(rb,LOW);

digitalWrite(lb,LOW);

digitalWrite(rf,LOW);

digitalWrite(lf,LOW);

}

**Working:**

Removing shoes in public places like temples and mosques is a mess and many people lose their shoes at those places. People don’t have security for their footwear in public places. Also, in countries like Japan and China, it is their culture to remove shoes at their workplace. To make this task simple and efficient, we have come up with a solution and we call it the smart shoe organizer.

The basic working of the bot is that, whenever a person leaves his shoes, his facial data is captured using a camera. the bot remembers his face and the shoes that left. then a vacant shoe rack is allowed to that person. The bot goes to the allotted rack and places his shoes inside the rack. The path towards the rack is based on the line follower mechanism using infrared sensors. And when the person comes back to collect his shoes, the facial data is checked with the database and the bot goes to his shoe rack which was allotted previously. the bot collects the shoes from the shoe rack and gives them back to the user.

The project is based on the Internet of things(IoT). the data transfer between the camera and the robot happens through a GET request.

some of the unique features of the robot are, the bot can recognize if the shoes are placed on it or not, and only when the shoes are placed, the bot accepts the request from the camera and starts moving, the robot is completely autonomous and doesn't require the internet for the data transfer through Wi-Fi.

Bot features EPS 32 is capable of hosting a 2.4GHz Wi-Fi access point natively which is the core concept of the whole project. the camera is connected to this network and the data transfer takes place through this network.

the bot is powered by 3s li-ion cells which gives an approximate voltage of 12V. To power the esp32 with 3.3v a DC-DC buckle converter is used to bring the voltage from 12V to 3.3V.

The lifting mechanism is designed using rack and pinion, where a single pinion is supported by two racks. A small tray is attached to the rack to place the shoes.

**Practical Implementation & Output:**

****

The bot works as a line follower bot to collect the footwear. It follows the dark line and reaches the person who wants to place the footwear in the rack after collecting the footwear it follows the same line and moves towards the shoe rack.



This is how it picks the footwear using the rack and pinion and it holds the footwear and places it in the rack.

**Existing competitors:**

The project is unique it follows its work. Using the image processing and picking the shoe and placing it in the rack and returning it to the person the same person makes it unique. especially in simplifying the task of the bot in reaching the shoe rack from person to rack and rack to the person using line follower is good to tackle. Although it is not accurate in a mess conditions it shows its uniqueness with a simple process rather than using high-end technologies like AI.

**Conclusion:**

We have successfully come up with a reliable solution to solve the problem of storing the footwear, which can be used in both public and private places like temples, parks, offices, houses, etc.…. we have made a bot which is useful to all the people in both public and the private places and which reduces the human effort and also saves the time of the person. We know that footwear is also theft by the people due to this it acts as clock room and people feel won’t get tense about their footwear and it gives a stress-free in searching the footwear before the pilgrimage places.

**Applications & Future Scope:**

We can make the bot even more intelligent by making it completely autonomous like detecting whether the given items are shoes or any other object.

It is used as a cloakroom by storing the footwear before the pilgrimage places. It is also used for the recognition of the people for allowing the particular places and places and certain occasions.