Ahsanullah University of Science and Technology
Department of Computer Science and Engineering

CSE3216: Microcontroller Based System Design Lab

Project name: Mask Detector Door System

Lab Section: A2 **Group: 6**

Submitted to

Afsana Ahmed Munia
Assistant Professor
CSE,AUST

Nibir Chandra Mandal
Lecturer
CSE,AUST

Submitted by

Tahiya Ahmed Chowdhury	170204048
Jannatul Ferdaus	170204051
Tamanna Nazmin	170204052

Contents

1 Objective

With the continuing outbreak of COVID-19 pandemic, the practice of wearing a mask has pretty much became mandatory now. Although we are adjusting with the norms of the "new normal" and avoiding social gatherings, sometimes it becomes essential for us to visit hospitals,banks,shops,etc. So we have decided to build a mask detector system which can be installed into the doors of such places. Our objective is to make sure that people do not participate in social gatherings or crowded places without a mask to avoid the spreading and infection of COVID 19.

2 Social values

The COVID 19 pandemic had almost left the entire world in a standstill. More than 90 countries in the world went into lockdown following The World Health Organization's declaration of the outbreak as a Public Health Emergency of International Concern in January 2020 and later a pandemic in March 2020.Research and speculations on the development and launching of vaccines started way back. And finally Bangladesh received it's first shipment in January and started trails from February 2021. But it's yet unclear how well they will curb the spread of the coronavirus. Hence vaccinated people also have to wear mask for now. And as the vaccines are being produced in limited amount for now, the need for mask still remains.

3 Required Components

These following parts and tools are required for building this project:

• Arduino UNO R3:

Arduino Uno is a microcontroller board based on the ATmega328P (datasheet).

• LCD Display(LM016L):

A 16x2 LCD with 2 rows and 16 colums consisting of 8 data pins along with register select,read write,enable,etc pins.

• Buzzer 5v active:

A buzzer is a device that creates an alarm based on certain conditions.The noise level of buzzer depends on it's input frequency.

• Wire(male to male, male to female, female to female):

Wires are used for the sole purpose of establishing connection between devices.

• Motor-PWM SERVO:

A servo motor can be rotated by adjusting the angles in both clockwise and anticlockwise direction.

• Webcam:

A device just like camera which has the capability of monitoring things.

4 Design

Design of our project is given below :

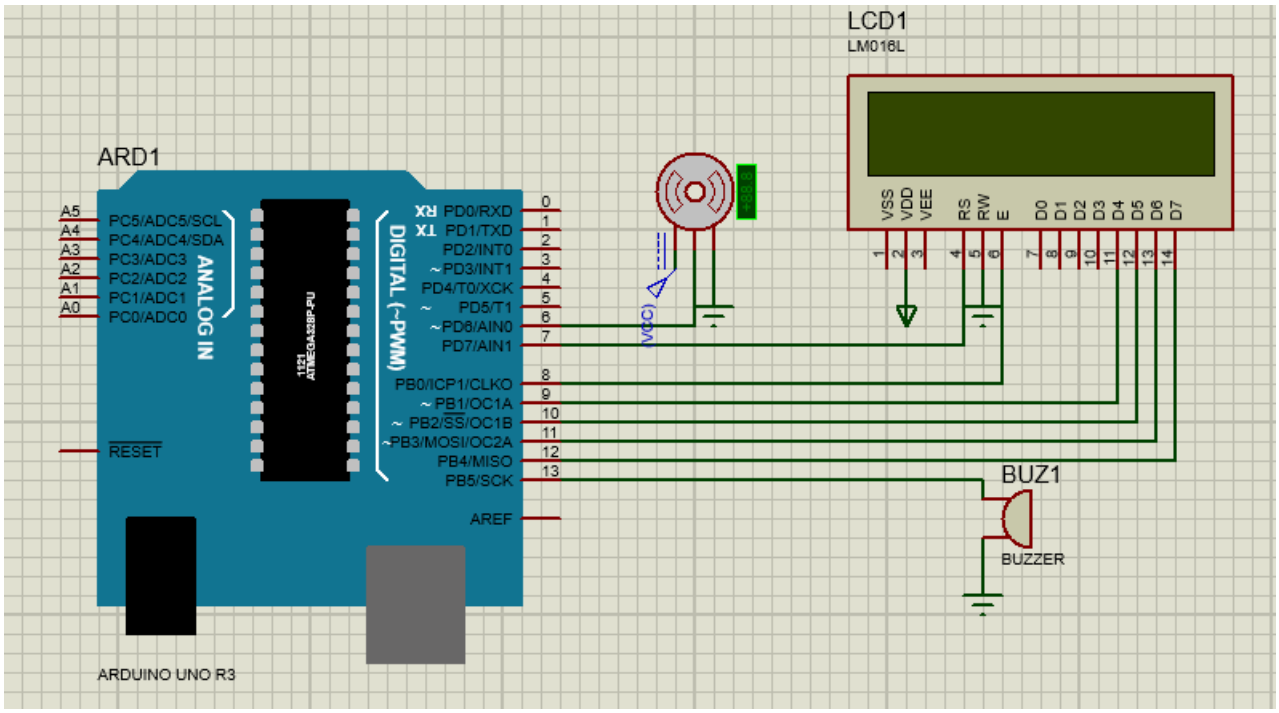

figure : design of mask detector door system in proteus

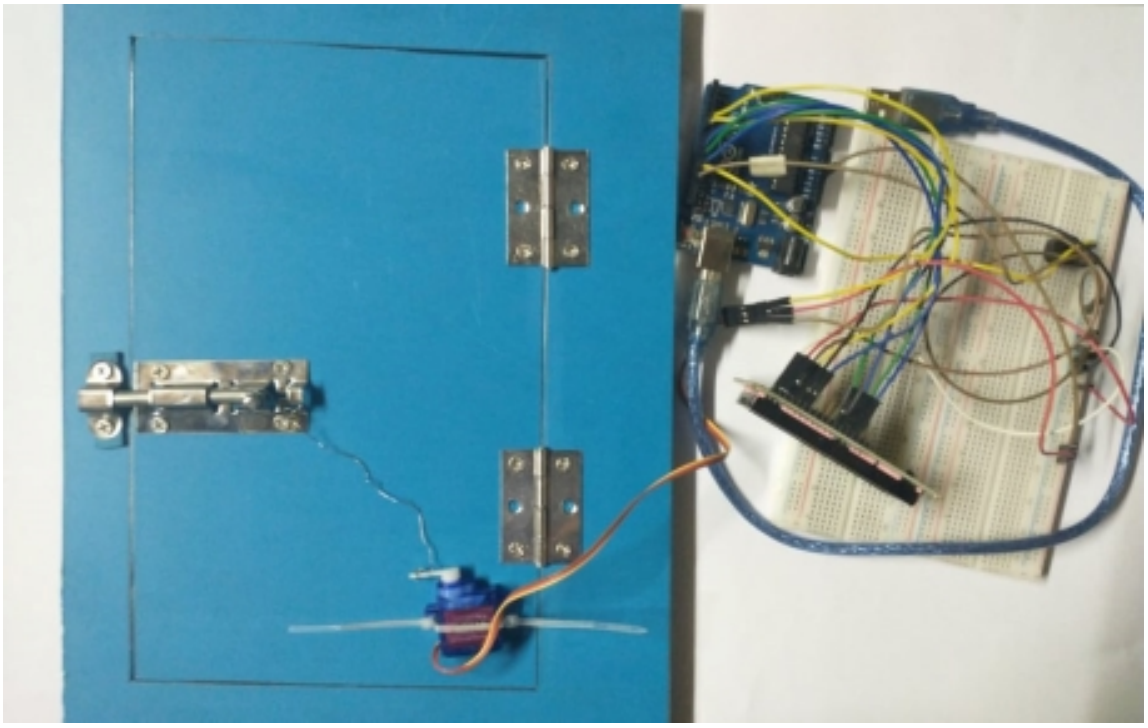

figure : design of mask detector door system in hardware

5 Working procedure

The basic components that react to the input are :

• Webcam :

The webcam of our pc detects whether the person is wearing a mask or not using convulational neural network by comparing the person's face with the other data sets i.e images analyzed by our mask detector model.

• Servo motor :

If mask is detected then the servo motor checks if door is open or not else it rotates clockwise from position 0 to open the door. Conversely, the servo motor makes sure whether the door is closed or not if not then it rotates anticlockwise from position 180 to close the door.

• LCD display :

At the same time, "Mask detected" is displayed in the screen of lcd if mask is detected or else "Please wear mask" caution is displayed.

• Buzzer :

In case of failure in mask detection, the buzzer turns on to notify the person.

6 Estimated Budget

Equipment	Quantity	Budget
Arduino UNO R3	1	430
Webcam	1	490
LCD Display(LM016L)	1	160
Buzzer 5v active	1	15
Wire(male to male, male to female, female to female)	as required	150
Motor-PWM SERVO	1	145

Total : Tk.1390

7 Code

Arduino Code:

```
include <LiquidCrystal.h>
include <Servo.h>
const int buzzer = 13;
int incomingByte;
const int rs = 7, en = 8, d4 = 9, d5 = 10, d6 = 11, d7 = 12;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
int pos = 0;
int val;
Servo myservo;
int countH=0;
int countL=0;

   void setup()
Serial.begin(9600);
myservo.attach(6);
pinMode(buzzer, OUTPUT);
lcd.begin(16, 2);


   void loop()

   if (Serial.available() > 0)
incomingByte = Serial.read();
```

```
if (incomingByte == 'H')
noTone(buzzer);
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Mask Detected");

if (incomingByte == 'L')
tone(buzzer,450);
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Please Wear Mask");


   if (incomingByte == 'H')
if(pos!=0  countH==0)
myservo.write(0);
delay(15);
countH=1;


   for (pos = 0; pos <= 180; pos += 10)
myservo.write(pos);
delay(2);

   else if (incomingByte == 'L')
if(pos!=180  countL==0)
myservo.write(180);
delay(15);
countL=1;


   for (pos = 180; pos >= 0; pos -= 10)
myservo.write(pos);
delay(2);
```

Mask Detector Model (using Convolutional Neural Network-CNN):

```
   from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import MobileNetV2
```

from tensorflow.keras.layers import AveragePooling2D
from tensorflow.keras.layers import Dropout
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Input
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.applications.mobilenet$_v2import preprocess_input$
$from tensorflow.keras.preprocessing.image import img_to_array$
$from tensorflow.keras.preprocessing.image import load_img$
$from tensorflow.keras.utils import to_categorical$
$from sklearn.preprocessing import LabelBinarizer$
$from sklearn.model_selection import train_test_split$
$from sklearn.metrics import classification_report$
$from imutils import paths$
$import matplotlib.pyplot as plt$
$import numpy as np$
$import os$

INIT$_LR = 1e - 4$
$EPOCHS = 20$
$BS = 32$

DIRECTORY = r"C:-Mask-Detection"
CATEGORIES = ["withMask", "withoutMask"]

data = []
labels = []

for category in CATEGORIES:
path = os.path.join(DIRECTORY, category)
for img in os.listdir(path):
img$_path = os.path.join(path, img)$
$image = load_img(img_path, target_size = (224, 224))$
$image = img_to_array(image)$
$image = preprocess_input(image)$

data.append(image)
labels.append(category)

lb = LabelBinarizer()
labels = lb.fit$_{t}ransform(labels)$
$labels = to_{c}ategorical(labels)$

data = np.array(data, dtype="float32")
labels = np.array(labels)

(trainX, testX, trainY, testY) = train$_{t}est_{s}plit(data, labels,$
$test_{s}ize = 0.20, stratify = labels, random_{s}tate = 42)$

aug = ImageDataGenerator(
rotation$_{r}ange = 20,$
$zoom_{r}ange = 0.15,$
$width_{s}hift_{r}ange = 0.2,$
$height_{s}hift_{r}ange = 0.2,$
$shear_{r}ange = 0.15,$
$horizontal_{f}lip = True,$
$fill_{m}ode = "nearest")$

baseModel = MobileNetV2(weights="imagenet", include$_{t}op = False,$
$input_{t}ensor = Input(shape = (224, 224, 3)))$

headModel = baseModel.output
headModel = AveragePooling2D(pool$_{s}ize = (7, 7))(headModel)$
$headModel = Flatten(name = "flatten")(headModel)$
$headModel = Dense(128, activation = "relu")(headModel)$
$headModel = Dropout(0.5)(headModel)$
$headModel = Dense(2, activation = "softmax")(headModel)$

model = Model(inputs=baseModel.input, outputs=headModel)

for layer in baseModel.layers:
layer.trainable = False

opt = Adam(lr=INIT$_{L}R, decay = INIT_{L}R/EPOCHS)$
$model.compile(loss = "binary_{c}rossentropy", optimizer = opt,$
$metrics = ["accuracy"])$

H = model.fit(
aug.flow(trainX, trainY, batch$_size = BS$),
$steps_per_epoch = len(trainX)//BS$,
$validation_data = (testX, testY)$,
$validation_steps = len(testX)//BS$,
$epochs = EPOCHS$)

print("[INFO] evaluating network...")
predIdxs = model.predict(testX, batch$_size = BS$)

predIdxs = np.argmax(predIdxs, axis=1)

print(classification$_report(testY.argmax(axis = 1), predIdxs$,
$target_names = lb.classes_)$)

print("[INFO] saving mask detector model...")
model.save("mask$_detector.model", save_format = "h5")$

N = EPOCHS
plt.style.use("ggplot")
plt.figure()
plt.plot(np.arange(0, N), H.history["loss"], label="train$_loss")$
$plt.plot(np.arange(0, N), H.history["val_loss"], label = "val_loss")$
$plt.plot(np.arange(0, N), H.history["accuracy"], label = "train_acc")$
$plt.plot(np.arange(0, N), H.history["val_accuracy"], label = "val_acc")$
$plt.title("TrainingLossandAccuracy")$
$plt.xlabel("Epoch")$
$plt.ylabel("Loss/Accuracy")$
$plt.legend(loc = "lowerleft")$
$plt.savefig("plot.png")$

Mask Detection Code (using open CV):

import the necessary packages
from tensorflow.keras.applications.mobilenet$_v2import preprocess_input$
$from tensorflow.keras.preprocessing.image import img_to_array$
$from tensorflow.keras.models import load_model$
$from imutils.video import VideoStream$
$import numpy as np$
$import imutils$

$importtime$
$importcv2$
$importos$
$importserial$
$importtime$

arduino = serial.Serial('COM3', 9600)
arduino = serial.Serial('/dev/ttyUSB0',9600)
lowConfidence = 0.75

def detectAndPredictMask(frame, faceNet, maskNet):
(h, w) = frame.shape[:2]
blob = cv2.dnn.blobFromImage(frame, 1.0, (224, 224),(104.0, 177.0, 123.0))
faceNet.setInput(blob)
detections = faceNet.forward()
faces = []
locs = []
preds = []
for i in range(0, detections.shape[2]):
confidence = detections[0, 0, i, 2]
if confidence > lowConfidence:
box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
(startX, startY, endX, endY) = box.astype("int")
(startX, startY) = (max(0, startX), max(0, startY))
(endX, endY) = (min(w - 1, endX), min(h - 1, endY))
face = frame[startY:endY, startX:endX]
face = cv2.cvtColor(face, cv2.COLOR$_BGR2RGB$)
$face = cv2.resize(face, (224, 224))$
$face = img_toarray(face)$
$face = preprocess_input(face)$
$faces.append(face)$
$locs.append((startX, startY, endX, endY))$
$iflen(faces) > 0 :$
$faces = np.array(faces, dtype = "float32")$
$preds = maskNet.predict(faces, batch_size = 32)$
$return(locs, preds)$
$prototxtPath = r"deploy.prototxt"$
$weightsPath = r"res10_300x300_ssd_iter_140000.caffemodel"$
$faceNet = cv2.dnn.readNet(prototxtPath, weightsPath)$

$maskNet = load_model("mask_detector.model")$
$vs = VideoStream(src = 0).start()$
$whileTrue:$
$frame = vs.read()$
$frame = imutils.resize(frame, width = 900)$
$(locs, preds) = detectAndPredictMask(frame, faceNet, maskNet)$
$for(box, pred)inzip(locs, preds):$
$(startX, startY, endX, endY) = box$
$(mask, withoutMask) = pred$
$label = "Mask"ifmask > withoutMaskelse"NoMask"$
$color = (0, 255, 0)iflabel == "Mask"else(0, 0, 255)$
$iflabel == "Mask":$
$print("ACCESSGRANTED")$
$arduino.write(b'H')$

else:
print("ACCESS DENIED")
arduino.write(b'L')

label = ": :.2fcv2.putText(frame, label, (startX, startY - 10),cv2.FONT$_HERSHEY_SIMPLEX$, 0.45, $color$, 2)
$cv2.rectangle(frame, (startX, startY), (endX, endY), color, 2)$
$cv2.imshow("FaceMaskDetectionbyKAREM - -qtoquit", frame)$
$key = cv2.waitKey(1)0xFF$
$ifkey == ord("q"):$
$break$
$cv2.destroyAllWindows()$
$vs.stop()$

8 Members Contribution

ID-17.02.04.048:
Arduino code, proteus design

ID-17.02.04.051:
Full Hardware setup

ID-17.02.04.052:
Face Mask Detection using Python, Tensorflow, OpenCV, pyserial

9 Difficulties

The difficulties during implementing the project is given below:

i)We couldn't successfully implement our idea in proteus so we implemented it in hardware and it is working perfectly.

10 Future Work

i)We can use our system in shopping mall, bank, hospital and so many other places.

ii)We can use better door lock to improve our hardware mechanism.

iii)We can integrate voice in our system to let people know that they are being refrained from entering as they are not wearing a mask properly.

11 Conclusion

Our country will still need sometime to fully fight and provide us a hundred percent protection against COVID 19. Till then we have to think of new adjustment techniques to buy more time. Maintaining the practice of always wearing a mask is one of them. Our project has been designed keeping that in mind.