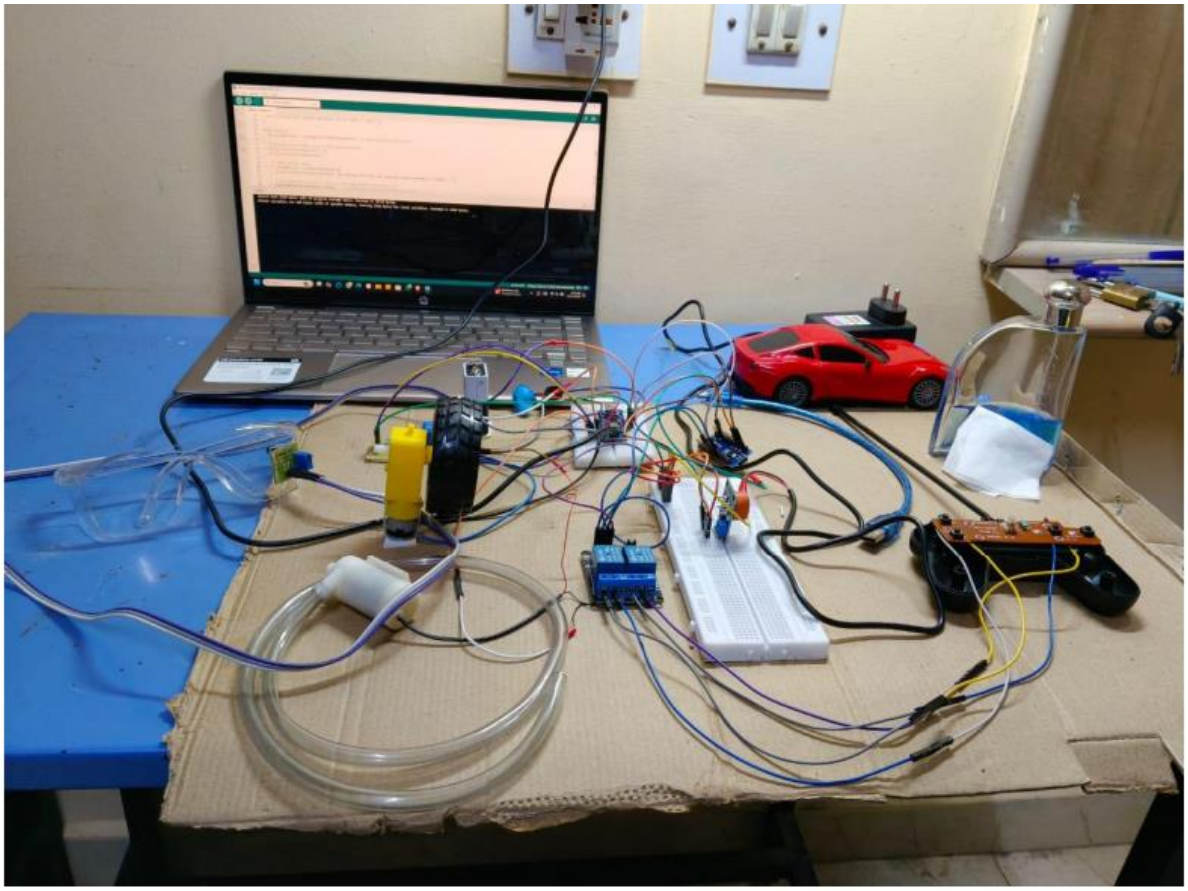
Detection of Alcohol and Sleeping in Driving System using IoT

Arani Vineela

Abstract

Driving under the influence of alcohol and driver fatigue are major factors contributing to road accidents worldwide, posing critical risks to both drivers and pedestrians. Despite stringent regulations and extensive public awareness campaigns, impaired and fatigued driving remain pressing safety issues due to a lack of reliable, real-time detection and intervention systems. Many current detection approaches are constrained by their reliance on post-incident analysis or manual monitoring methods, which do not address the need for immediate preventive action. This study introduces an IoT- and image processing-based system designed to detect alcohol consumption and drowsiness in drivers in real-time. Leveraging the combined capabilities of alcohol sensors and in-car cameras, the system assesses the driver's breath alcohol content and analyzes facial features to identify signs of drowsiness. Once impairment or fatigue indicators are detected, the system initiates alerts or other preventive measures, such as activating in-vehicle alarms or restricting vehicle controls, aimed at reducing the likelihood of accidents.

Hardware Set UP



1. Arduino Code for Alcohol Detection, Self-Parking, and Water Sprinkling

This code handles alcohol detection using the MQ3 sensor, performs self-parking by controlling

the relays, and triggers water sprinkling when required.

```
// Pin Definitions
const int alcoholSensorPin = A0; // Analog pin for MQ3 alcohol sensor
const int leftRelayPin = 3; // Relay control pin for Left button on remote
const int forwardRelayPin = 4; // Relay control pin for Forward button on remote
const int waterMotorControlPin = 6; // Pin to control water motor
// Threshold for Alcohol Detection
int alcoholThreshold = 400; // Adjust this based on sensor calibration
void setup() {
 Serial.begin(9600); // Initialize serial communication
 pinMode(alcoholSensorPin, INPUT);
 pinMode(leftRelayPin, OUTPUT);
pinMode(forwardRelayPin, OUTPUT);
 pinMode(waterMotorControlPin, OUTPUT);
 // Start with relays and water motor off
 digitalWrite(leftRelayPin, LOW);
 digitalWrite(forwardRelayPin, LOW);
 digitalWrite(waterMotorControlPin, LOW);
 // Initial check to ensure "no alcohol detected" at startup
 while (analogRead(alcoholSensorPin) > alcoholThreshold) {
 Serial.println("Alcohol detected on startup. Waiting until no alcohol is detected...");
 delay(1000); // Wait and check again every second
 }
 Serial.println("No alcohol detected. Car is ready to start.");
```

```
}

void loop() {

 int alcoholLevel = analogRead(alcoholSensorPin); // Read alcohol sensor value

 // Display the alcohol level on the Serial Monitor

 Serial.print("Alcohol Level: ");

 Serial.println(alcoholLevel);

 // Check alcohol level

 if (alcoholLevel > alcoholThreshold) {

 Serial.println("Alcohol Detected - Activating left turn and stopping forward movement...");

 // Activate both relays for self-parking

 digitalWrite(leftRelayPin, HIGH); // Simulate Left button press

 digitalWrite(forwardRelayPin, HIGH); // Simulate Forward button press

 delay(2000); // Keep both buttons "pressed" for 2 seconds

 Serial.println("Self-parking complete. Stopping the car.");

 digitalWrite(leftRelayPin, LOW); // Turn off Left button relay

 digitalWrite(forwardRelayPin, HIGH); // Keep Forward button relay on to prevent movement

 // Activate water motor after parking

 Serial.println("Activating water motor...");

 digitalWrite(waterMotorControlPin, HIGH);

 delay(2000); // Run water motor for 2 seconds

 digitalWrite(waterMotorControlPin, LOW);

 }

 delay(100); // Small delay to stabilize readings

}
```

2. Arduino Code for Drowsiness Detection, Buzzer, and Engine Locking

This code handles drowsiness detection using an IR sensor, activates a buzzer for alert, and locks

the engine.

```
// Pin Definitions
const int drowsinessInputPin = 5; // Pin to receive drowsiness signal from IR sensor
const int buzzerPin = 11; // Pin to control buzzer
const int enginePin = 6; // Pin to control DC motor (engine)
// Time for alerts (in milliseconds)
const int alertDuration = 2000; // Duration for buzzer and engine lock
void setup() {
 Serial.begin(9600); // Initialize serial communication
 pinMode(drowsinessInputPin, INPUT);
 pinMode(buzzerPin, OUTPUT);
 pinMode(enginePin, OUTPUT);
// Start with buzzer and engine running
 digitalWrite(buzzerPin, LOW);
 digitalWrite(enginePin, HIGH); // Assume HIGH keeps engine running
}
void loop() {
 // Check if drowsiness is detected via IR sensor
 if (digitalRead(drowsinessInputPin) == HIGH) {
 Serial.println("Drowsiness Detected - Activating buzzer and locking engine...");

 // Activate buzzer and stop the engine
 digitalWrite(buzzerPin, HIGH); // Turn on buzzer
 digitalWrite(enginePin, LOW); // Stop engine
 delay(alertDuration); // Run for alert duration
 // Deactivate buzzer and restart the engine
 digitalWrite(buzzerPin, LOW); // Turn off buzzer
 digitalWrite(enginePin, HIGH); // Restart engine
```

```
 }
 delay(100); // Small delay to stabilize readings
}
```

Drowsiness detection using image processing code:

```
from scipy.spatial import distance as dist
from imutils.video import VideoStream
from imutils import face_utils
import imutils
import time
import dlib
import cv2
import serial
# Initialize Serial Communication
arduino = serial.Serial('COM3', 9600) # Replace 'COM3' with the correct port
time.sleep(2) # Wait for connection to establish
def eye_aspect_ratio(eye):
 A = dist.euclidean(eye[1], eye[5])
 B = dist.euclidean(eye[2], eye[4])
 C = dist.euclidean(eye[0], eye[3])
 ear = (A + B) / (2.0 * C)
 return ear
def final_ear(shape):
 (lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
 (rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]
 leftEye = shape[lStart:lEnd]
 rightEye = shape[rStart:rEnd]
 leftEAR = eye_aspect_ratio(leftEye)
 rightEAR = eye_aspect_ratio(rightEye)
```

```
 ear = (leftEAR + rightEAR) / 2.0
 return (ear, leftEye, rightEye)
# Constants
EYE_AR_THRESH = 0.25
EYE_AR_CONSEC_FRAMES = 30
COUNTER = 0
# Initialize Dlib's face detector and facial landmarks predictor
print("-> Loading predictor and detector...")
detector = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
predictor = dlib.shape_predictor('shape_predictor_68_face_landmarks.dat')
# Start video stream
print("-> Starting Video Stream")
vs = VideoStream(src=0).start()
time.sleep(1.0)
while True:
 frame = vs.read()
 frame = imutils.resize(frame, width=450)
 gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
 rects = detector.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5,
minSize=(30, 30),
flags=cv2.CASCADE_SCALE_IMAGE)
 for (x, y, w, h) in rects:
 rect = dlib.rectangle(int(x), int(y), int(x + w), int(y + h))
 shape = predictor(gray, rect)
 shape = face_utils.shape_to_np(shape)

 ear, leftEye, rightEye = final_ear(shape)
 # Draw contours on the eyes
```

```
 cv2.drawContours(frame, [cv2.convexHull(leftEye)], -1, (0, 255, 0), 1)
 cv2.drawContours(frame, [cv2.convexHull(rightEye)], -1, (0, 255, 0), 1)
if ear < EYE_AR_THRESH:
 COUNTER += 1
 if COUNTER >= EYE_AR_CONSEC_FRAMES:
 cv2.putText(frame, "DROWSINESS ALERT!", (10, 30), cv2.FONT_HERSHEY_SIMPLEX,
0.7, (0, 0, 255), 2)

 # Send signal to Arduino 1
 arduino.write(b'1')
 else:
 COUNTER = 0
 cv2.putText(frame, f"EAR: {ear:.2f}", (300, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0,
255), 2)
 cv2.imshow("Frame", frame)
 key = cv2.waitKey(1) & 0xFF
 if key == ord("q"):
 break
cv2.destroyAllWindows()
vs.stop()
```