MECHANICAL PROJECT

PROJECT NAME—ANTI-SLEEP ALARM FOR DRIVERS

COURSE—B.TECH(CSE)

YEAR-FIRST YEAR

SEMESTER-2nd

SECTION--A1

PARTICIPANTS NAME	ENROLLMENT NUMBER
PRASANT KUMAR	12021002001008
SWAPAN KUMAR SHEE	12021002001009
SHRUTI RAJDHAR	12021002001010
ROHIT CHAKRABORTY	12021002001017
HIMANSHU RAJ	12021002001020
KASHISH	12021002001023
MD DANISH KALAM	12021002001027

GUIDED BY---

VINOD YADAV

DEPARTMENT OF MECHANICAL ENGINEERING

Abstract

Feeling sleepy while driving could cause a hazardous traffic accident. However, when driving alone on the highway or driving over a long period of time, drivers are inclined to feel bored and sleepy, or even fall asleep. Nowadays most of the products of driver anti-sleep detection sold in the market are simply earphones making intermittent noises, which is quite annoying and inefficient. As such, there is a high demand for cheap and efficient driver sleep detection. Therefore, we came up with an idea and successfully developed a sleepy detection and alarming system, which could effectively meet.

Introduction

1.1. Project motivation and purpose

The goal of this project is to develop a system that can accurately detect sleepy driving and make alarms accordingly, which aims to prevent the drivers from drowsy driving and create a safer driving environment. The project was accomplished by a Webcam that constantly takes images of the driver, a beagle board that implements an image processing algorithm of sleepy detection, and a feedback circuit that could generate an alarm and a power supply system.

1.2. Functions and Features

This system has many features that make it unique and functional. These features include:

1. Eye extraction, use open and close to determine sleepiness .
2. Daytime and night detection .
3. Real time image processing and detection.
4. Sound and flashing LED warning system to redraw driver’s attention .

5. Little inference and potential hazard to driver's normal driving .

6. Portable size with car cigarette charger socket power supply.

Materials required:

1) Relay:

2) Piezo buzzer:

3) Wires:

4) 9v battery:

5) Gear motor:

6) Wheel

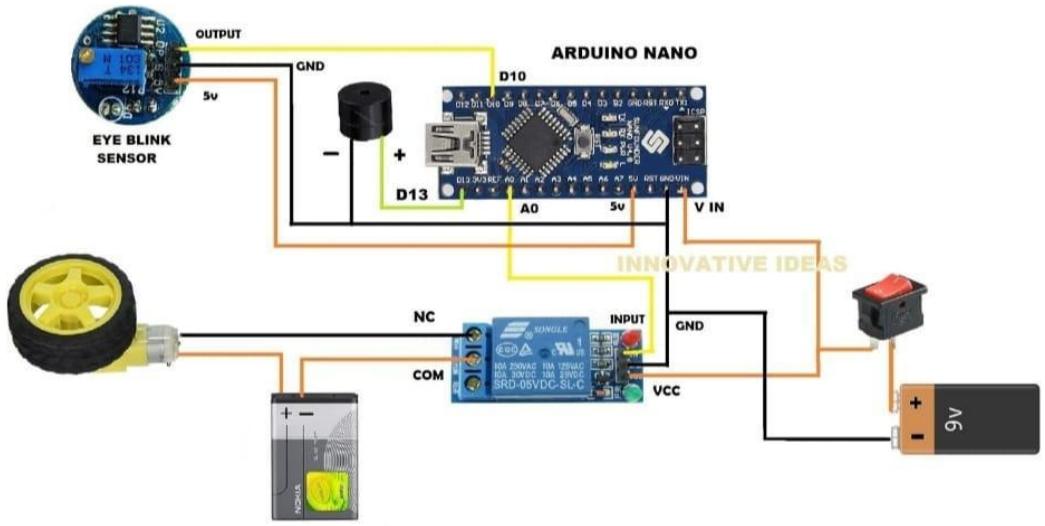
7) Arduino nano

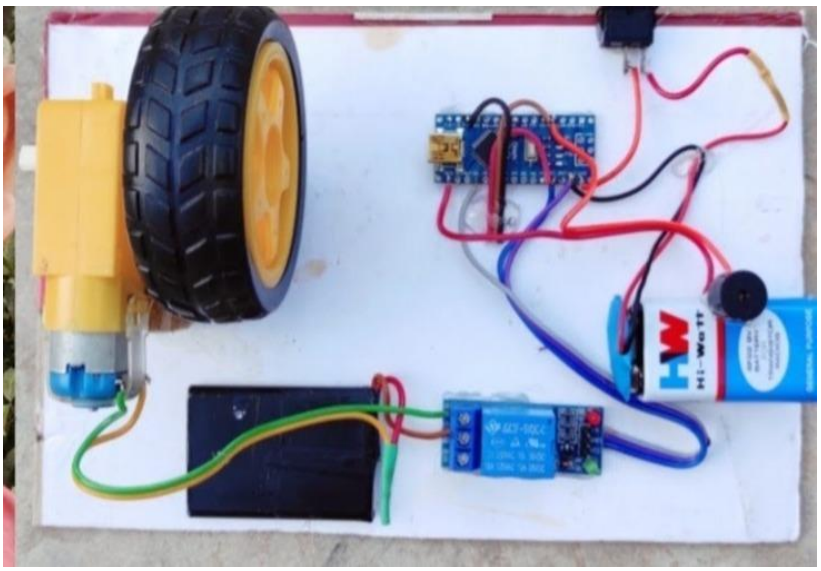
8) SPST switch:

9) Eye blink sensor:

UEM
UNIVERSITY OF ENGINEERING & MANAGEMENT
श्रद्धावान लभते ज्ञानम्
Good Education, Good Jobs

Circuit Diagram:





Results :

1. Physiological sleepiness detection devices
    - Sleepiness detection devices based on eye activity
    - EEG-based sleepiness detection devices
    - Devices based on electrodermal activity (galvanic skin resistance)
2. Sleepiness detection devices based on physical activity, including wrist activity and head movements
3. Devices based on behavioural measures, including driver or operator performance or task-related activity, and secondary task performance
4. Model-based predictions of sleepiness
5. Devices based upon a combination of more than one approach or measure.

Benefits:

This Glasses alerts the driver whenever he is getting into sleep while driving the vehicle. Since sleeping on wheels is dangerous sometimes it may convert into fatal accidents that can lead to death. to prevent such consequences of accidents we can use this gadget to alert the driver when he feels drowsiness.

Arduino code:

```
#define Relay 13
#define buzzer A0
static const int sensorPin = 10;                // sensor input pin
int SensorStatePrevious = LOW;                  // previousstate of the sensor

unsigned long minSensorDuration = 3000; // Time we wait before  the sensor active as long
unsigned long minSensorDuration2 = 6000;
unsigned long SensorLongMillis;                 // Time in ms when the sensor was active
bool SensorStateLongTime = false;               // True if it is a long active

const int intervalSensor = 50;                  // Time between two readings sensor state
unsigned long previousSensorMillis;             // Timestamp of the latest reading

unsigned long SensorOutDuration;                // Time the sensor is active in ms
```


```
//// GENERAL ////

unsigned long currentMillis;          // Variabele to store the number of
milleseconds since the Arduino has started

void setup() {
  Serial.begin(9600);                 // Initialise the serial monitor

  pinMode(sensorPin, INPUT);          // set sensorPin as input
  Serial.println("Press button");
  pinMode(Relay,OUTPUT);
  pinMode(buzzer,OUTPUT);
}

// Function for reading the sensor state
void readSensorState() {

  // If the difference in time between the previous reading is larger than
intervalsensor
  if(currentMillis - previousSensorMillis > intervalSensor) {

    // Read the digital value of the sensor (LOW/HIGH)
    int SensorState = digitalRead(sensorPin);

    // If the button has been active AND
    // If the sensor wasn't activated before AND
    // IF there was not already a measurement running to determine how long the
sensor has been activated
```


```
 if (SensorState == LOW && SensorStatePrevious == HIGH &&
!SensorStateLongTime) {
     SensorLongMillis = currentMillis;
       SensorStatePrevious = LOW;

      Serial.println("Button pressed");
    }

    // Calculate how long the sensor has been activated
   SensorOutDuration = currentMillis - SensorLongMillis;

    // If the button is active AND
    // If there is no measurement running to determine how long the sensor is
active AND
    // If the time the sensor has been activated is larger or equal to the time
needed for a long active
    if (SensorState == LOW && !SensorStateLongTime && SensorOutDuration
>= minSensorDuration) {
      SensorStateLongTime = true;
      digitalWrite(Relay,HIGH);
      Serial.println("Button long pressed");
    }
    if (SensorState == LOW && SensorStateLongTime && SensorOutDuration
>= minSensorDuration2) {
     SensorStateLongTime = true;
      digitalWrite(buzzer,HIGH);
      delay(1000);
      Serial.println("Button long pressed");
    }
```

// If the sensor is released AND


```
  // If the sensor was activated before
    if (SensorState == HIGH && SensorStatePrevious == LOW) {
      SensorStatePrevious = HIGH;
      SensorStateLongTime = false;
      digitalWrite(Relay,LOW);
      digitalWrite(buzzer,LOW);
      Serial.println("Button released");


    }

    // store the current timestamp in previousSensorMillis
   previousSensorMillis = currentMillis;

  }

}

void loop() {

  currentMillis = millis();    // store the current time
  readSensorState();           // read the sensor state       }
```