Design and Implementation of a Smart Military Shoe: An Intelligent Footwear for Modern Warfare

Md. Faiyaz Abdullah Sayeedi*, Sabrina Afrin†, Maz Bin Hossain‡, Md. Kamrul Hassan§ and Molla Md. Sabit¶
Dept. of Computer Science and Engineering
United International University, Dhaka, Bangladesh
011212049*, 011212131†, 011212133‡, 011212153§ and 011212167¶

***Abstract*—To increase the security of soldiers in warfare, the usage of modern technology has been increasing day by day. Many researchers, scientists, and engineers are working in this context. Following this, we design and implement a smart military shoe system to enhance soldiers' situational awareness and give them real-time feedback. This system has been implemented on a shoe with some sensors and hardware components. Such as - GPS module, soil moisturizer sensor, ultrasonic sensor, etc. This military shoe system offers several important features such as - mine detection and disabling, location tracking, measuring Heartbeat and SpO2, and balancing spikes. Overall the smart military shoe will represent a hopeful performance and safety for soldiers.**

***Index Terms*—Military, Smart Shoe, Arduino, GPS, Location, Mine, Heartbeat, SpO2, Mission, Warfare**

I. INTRODUCTION

[Problem Statement] It is essential for soldiers to be informed about their surroundings and make correct decisions quickly, as military operations usually take place in dangerous, hazardous, and unpredictable environments. Actually, traditional military footwear doesn't give the opportunity to make soldiers aware of their surroundings, and that sometimes puts them at risk. To overcome these problems, there is a need for modernized solutions on the battlefield.

[Literature Review] Researchers have been studying this issue for a huge time. Several research has been done regarding - the development of smart footwear for military soldiers. For example, a research paper written by S. Jessica Pauline et al. (2021) says that to know the position, status, and location of a soldier's radio, a line communication system has been used, which is not that much effective for them. [1] To get out of this situation, they proposed a system that could track military personnel in an effective way. Wu et al. (2019), in their research paper, proposed a smart helmet system that can detect dangerous threats in the environment of a soldier. [2] In another research paper, Abdelhakim et al. (2019) proposed and developed a smart insole that can monitor soldiers' movement and steps. [3] Another study by Li et al. (2020) developed a smart footwear system that can inform soldiers of dangerous environments, such as - minefields. In the study of D. Janson et al. (2019), he shows more concern and interest in safety. [4]

[Creativity] The total overview of our literature review shows the importance of a technology-based smart military shoe to improve the performance of soldiers on the battlefield. For this, we have come up with an innovative solution which is our "Smart Military Shoe." The aim of this smart military shoe is to help soldiers to face challenges on the battlefield.

II. PROJECT OVERVIEW

The Smart Military Shoe System is designed to enhance soldiers' situational awareness and give them immediate feedback on their surroundings. It will make a soldier aware while there is mine and disable it, which can save the life of military personnel. It will also track the location of the soldiers and continuously send the update to the headquarter. Furthermore, It will measure the heartbeat and SpO2 of the soldiers. Lastly, it will help a soldier to keep balance in muddy areas and mountains.

- Mine Detection and Disabling:

  The Smart Military Shoe System is designed and equipped with a sensor that can detect mine set under the land. A metal mine detector is an instrument and a feature that detects the nearby presence of metal. Metal can be present in the form of mines, too, it is useful to find any kind of harmful mines and metals on the surface of soil. It works by transmitting signals from the source to a buzzer.
  Any metal objects within the electromagnetic field will let the signals active and re-transmit an electromagnetic field of their own, with the help of a buzzer. So, the presence of metal can only be identified when the buzzer sounds up.

- Location Tracking and Sending SMS consisting of location:

  The Smart Military Shoe System is programmed to periodically obtain the GPS coordinates of the location of the device using the GPS module. We are going to use an Arduino nano board, SIM800L 2G

Module, etc.
The system is designed to operate in power-saving mode to minimize battery power. Additionally, the system is programmed to turn off the GPS module and his SIM800L module when not in use to further conserve power. The received location data will be sent to a specific phone number via SIM800L 2G module.

- Measuring Heartbeat and SpO2:

If the military mission army wants to check the current situation of their health regarding blood pressure, heart beat rate, or oximeter rates, their personal health assistant has to assist them by giving updates continuously, which is granted as time-consuming, and the assistant may not be available there.
So this feature helps to give an instant solution that if the person is feeling sick, they can immediately check two things: Blood pressure and oximeter. It will immediately check the above things anytime and anywhere. By this, there will be a specific measurement and boundary of the parameters of the BP that, if these exceed, will give the information that the health condition is alarming or not, and then that alarm will update the following army members about the seriousness of him.

- Balancing Spikes:

The Smart Military Shoe System is designed and equipped with spikes that can get out from the sole of the shoe to help the soldiers maintain their balance on muddy or uneven land. This feature can be very useful, especially in challenging environments like - muddy or uneven lands, mountains, etc.

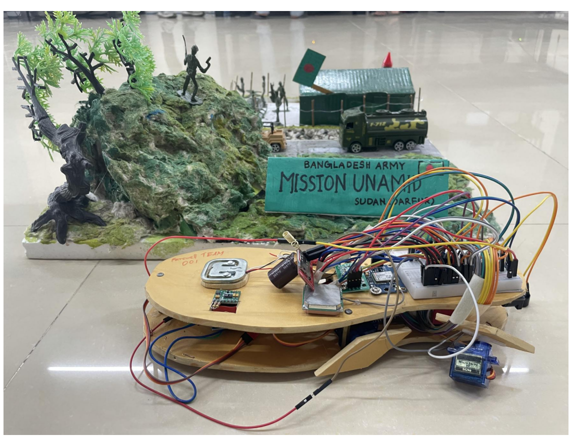

Fig. 1. Smart Military Shoe

III. Component List

We used the following components to make the project "Smart Military Shoe":

1) Arduino (UNO)
2) Open Circuit
3) Ultrasonic Sensor Module and ULN2003
4) 16x2 LCD
5) Buzzer
6) GPS Module and GSM Module (SIM800L 2G)
7) MAX30100 Pulse Oximeter Sensor
8) Soil Moisture Sensor
9) Load Cell
10) Servo Motors

A. Arduino (UNO)

Arduino Uno is a microcontroller board and an open-source electronics platform. Arduino boards are capable of reading inputs and giving outputs. We can give instructions to the Arduino board through Arduino programming language and Arduino IDE.

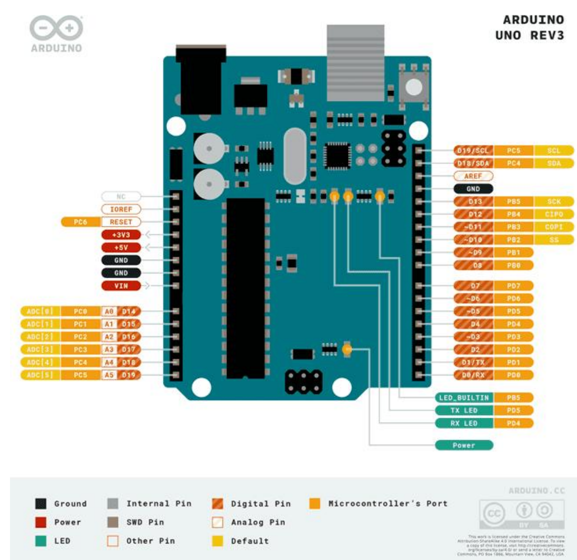

Fig. 2. Circuit Diagram of Arduino

B. Open Circuit

Here we use a DC power source with a LED which will work as a buzzer and an open wire. On the other hand, this is a closed circuit where the metal is detected and, under the electromagnetic field, is re-transmitting signals as stated above.
We split the circuit throughout the shoe sole to activate the circuit throughout the entire shoe sole to work evenly. Series connections of the circuits are to be connected. By this, there will form a series of connections of circuits according to the show sole and work evenly.

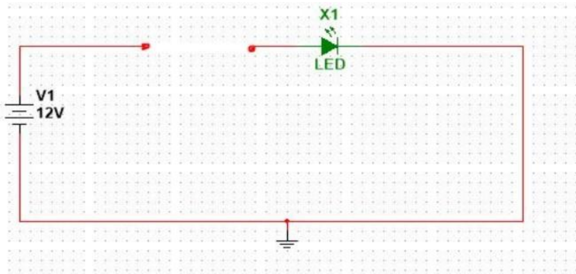

Fig. 3. Circuit Diagram of Open Circuit

C. Ultrasonic Sensor Module

The Ultrasonic sensor module's "trigger" and "echo" pins are directly connected to pins 10 and 11 of Arduino. Control pin RS, RW, and En are directly connected to Arduino pin 7, GND, and 6. And data pin D4-D7 is connected to 5, 4, 3, and 2 of Arduino, and the buzzer is connected at pin 12.

At pin 8 of Arduino, ULN2003 is connected for turning on or turning off the doors in between the chambers of Plaster of Paris powder and water.

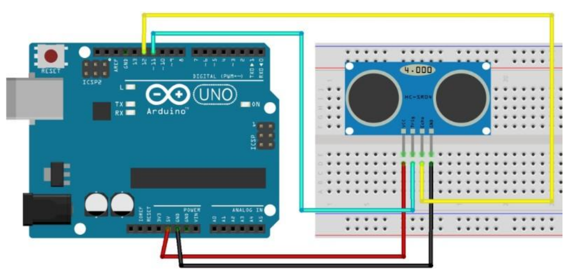

Fig. 4. Circuit Diagram of Ultra Sonic

D. 16x2 LCD

The full meaning of LED is Light Emitting Diode. It's a semiconductor device that gives off light when electricity flows through it. In order to produce light, holes from p-type semiconductors mix with electrons from n-type semiconductors.

E. Buzzer

A buzzer is an electric device that produces sound. There are two types of buzzers piezo and magnetic.

The piezoelectric material mechanically deforms when a voltage is supplied across the two electrodes, which is similar to how the ferromagnetic disk in a magnetic buzzer or the speaker cone of the piezo disk inside the buzzer moves to produce sound.

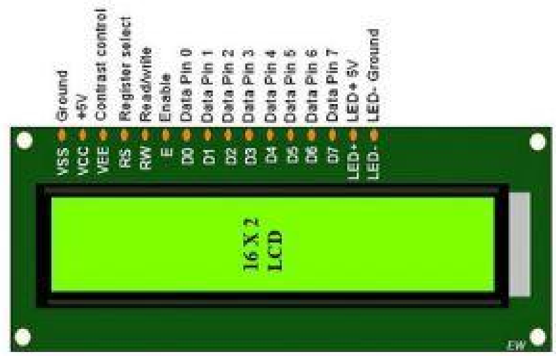

Fig. 5. Circuit Diagram of LCD

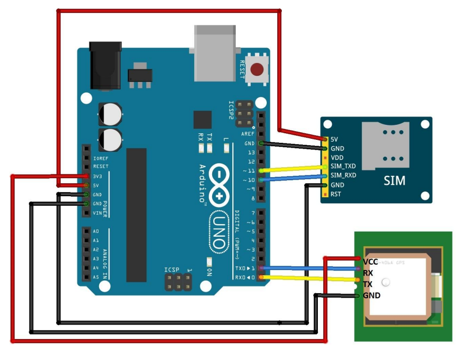

Fig. 6. Circuit Diagram of GPS and GSM

F. GPS Module and GSM Module (SIM800L 2G)

The SIM800L 2G module sends the location data to a designated phone number or email address upon receiving the 'get location' command message. This message is sent from the user's mobile device to the SIM card inserted in the smart tracker. The message is then processed by the Arduino Nano board and interpreted by the software to initiate the location data collection and transmission. Process. The 'Tiny GPS++' and 'FONA' libraries used in the programming of the device provide functions to parse GPS data and communicate with the SIM800L module, respectively. These libraries help to simplify the programming process and improve the accuracy and reliability of the device.

G. MAX30100 Pulse Oximeter Sensor

The MAX30100 Sensor is capable of measuring Blood Oxygen and Heart Rate. We can use any display, like a 16×2 LCD Display, to view the value of SpO2 and BPM.

The MAX30100 is a Pulse Oximetry and heart rate monitor sensor solution. It combines two LEDs, a photodetector, optimized optics, and low-noise analog signal processing to detect pulse oximetry and heart-rate signals.

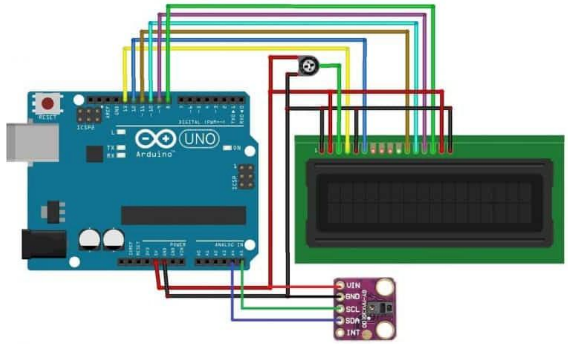

Fig. 7. Circuit Diagram of MAX30100 Pulse Oximeter Sensor

H. Soil Moisture Sensor

A soil moisture sensor is a sensor that measures the volumetric water content in the soil. It is mainly used to check the moisture of the soil.
The Soil moisture sensor has four pins VCC, GND, AOUT, and DOUT. These four pins can be used to get the soil moisture data from the sensor. We just need to stick the fork-shaped conductive probe to the soil, which has two exposed conductive plates which will act as a variable resistor, and the value of the resistor will depend on the water content of the soil. The more water in the soil will cause more conductivity which will resist lower resistance.

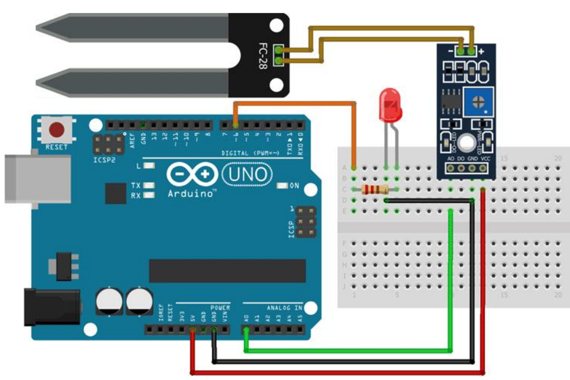

Fig. 8. Circuit Diagram of Soil Moisture Sensor

I. Load Cell

The load cell is a transducer that transforms force or pressure into electrical output. The magnitude of this electrical output is directly proportional to the force being applied. Load cells have a strain gauge, which deforms when pressure is applied to it. And then strain gauge generates an electrical signal on deformation as its effective resistance changes on deformation. A load cell usually consists of four strain gauges in a Wheatstone bridge configuration. Load cell comes in various ranges like 5kg, 10kg, 100kg, and more. Here we have used Load cell, which can weigh up to 40kg.

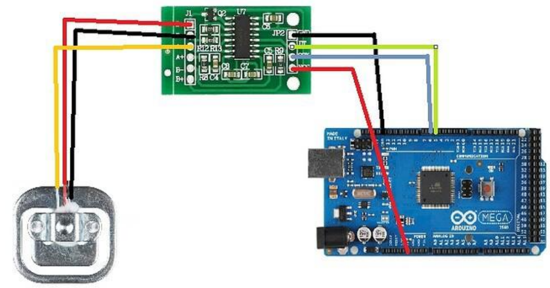

Fig. 9. Circuit Diagram of Load Cell

J. Servo Motors

A servo motor is a straightforward electric motor that is managed by servomechanism. A Servo motor is used to rotate an object at a certain angle or distance. A motor is referred to as a DC servo motor if it is powered by a DC power source and an AC servo motor if it is driven by an AC power source. We used this servo motor in our project with spikes.

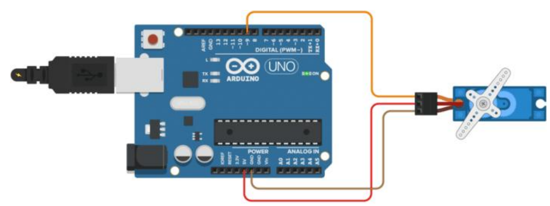

Fig. 10. Circuit Diagram of Servo Motor

IV. System Flowchart

The system flowchart for this project consists of some hardware components that are merged together to develop the system. Before the system flowchart, here we present our working procedures/steps.

They are as follows:

- Identifying the Purpose of the System
- Selecting the Sensing Components
- Determining the sequence of operations
- Feature Implementation
  - Mine Detection and Disabling
  - Location Tracking
  - Measuring Heartbeat and SpO2
  - Balancing Spikes
- Data Collection from the Sensor
- Data Processing
- Real-Time Feedback

Now, Our system flowchart is as follows:

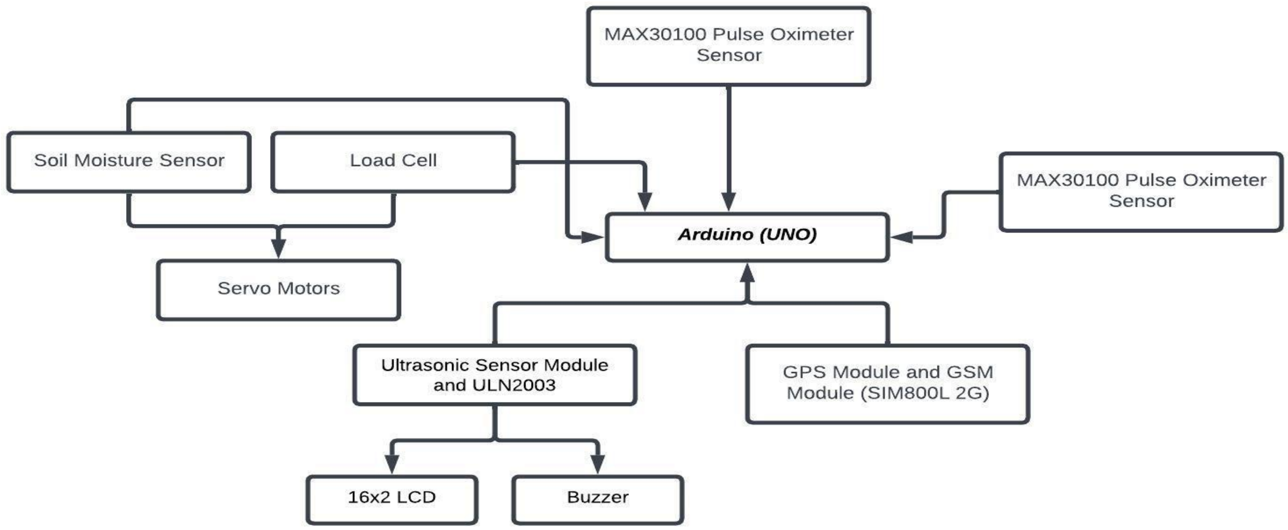

Fig. 11. Flowchart of System

V. CODE

Our system code:

```
#include<Servo.h>
#include<Wire.h>
#include <Wire.h>
#include "MAX30100_PulseOximeter.h"
#include <SoftwareSerial.h>
#include <SoftwareSerial.h>
#include <TinyGPS.h>
#include <TinyGPS++.h>


SoftwareSerial sim800l(12, 13);
SoftwareSerial mySerial(10,11);
TinyGPS gps;
#define REPORTING_PERIOD_MS     750

PulseOximeter pox;
uint32_t tsLastReport = 0;

#define GPS_BAUD 9600
#define SIM800_BAUD 9600

SoftwareSerial gpsSerial(10, 11);
SoftwareSerial sim800Serial(12, 13);
TinyGPSPlus gps1;

Servo s1, s2, s3;
int soil;
int moist;
int pos = 0;
const int waterSensorPin = A1;
const int buzzerPin = 8;
int buttonPin = 9;
int buttonState;
Servo gate;
//int pos = 0;

void setup(){
  Serial.begin(9600);
  s1.attach(2);
  s2.attach(3);
  s3.attach(4);
  s1.write(0);
  s2.write(0);
  s3.write(0);

  pinMode(buzzerPin, OUTPUT);
  pinMode(buttonPin, INPUT);
  digitalWrite(buttonPin, HIGH);
  gate.attach(5);
  gate.write(0);

  Serial.begin(9600);
  sim800l.begin(9600);
  mySerial.begin(9600);
  Serial.begin(9600);
  delay(3000);
  Serial.print("Initializing pulse oximeter..");
  if (!pox.begin()) {
    Serial.println("FAILED");
    for(;;);
  } else
    { Serial.println("SUCCESS");
  }
```

```
  pox.setIRLedCurrent(MAX30100_LED_CURR_50MA);

  gpsSerial.begin(GPS_BAUD);
  sim800Serial.begin(SIM800_BAUD);


  sim800Serial.println("AT+CMGF=1");
  delay(100);
  sim800Serial.println("AT+CNMI=2,2,0,0,0");
  delay(100);
}

void loop(){
  soil = analogRead(A0);
  moist = map(soil, 0, 1023, 100, 0);

  if(moist>50)
  {

     s1.write(90);
     delay(2);
     s2.write(90);
     delay(2);
     s3.write(90);

  }
  else
  {

    s1.write(0);
    delay(2);
    s2.write(0);
    delay(2);
    s3.write(0);
  }

  int sensorValue = analogRead(waterSensorPin);
  if (sensorValue >570)
  {
    int outputValue = map(sensorValue,
    570, 800, 0, 255);
    Serial.println(outputValue);
    digitalWrite(buzzerPin, HIGH);
  }
  else
  {
    digitalWrite(buzzerPin, LOW);
    buttonState = digitalRead(buttonPin);
    if (buttonState == 0)
    {
      gate.write(90);
      // delay(2000);
      // gate.write(0);
    }
    else
    {
      gate.write(0);
    }
  }

  //delay(250);
    pox.update();

  if (millis() - tsLastReport >
  REPORTING_PERIOD_MS) {
    Serial.print("Heart rate:");
    Serial.print(pox.getHeartRate());

    Serial.print("bpm / SpO2:");
    Serial.print(pox.getSpO2());
    Serial.println("%");

    int heartRate = pox.getHeartRate();

    if (heartRate > 130)
    {
      bool newdata = false;
      unsigned long start = millis();
      // Every 5 seconds we print an update
      while (millis() - start < 5000)
      {
        if (mySerial.available())
        {
          char c = mySerial.read();
          //Serial.print(c);
          if (gps.encode(c))
          {
            newdata = true;
            break;
          }
        }
      }
      if (newdata)
      {
        gpsdump(gps);
        Serial.println();
      }
    }
    tsLastReport = millis();
  }

    if (sim800Serial.available())
    { String sms = sim800Serial.
    readStringUntil('\n');
    if (sms.indexOf("location") != -1)
      { sendLocationSMS();
    }
  }

  while (gpsSerial.available()) {
    if (gps1.encode(gpsSerial.read()))
      { sendLocationSMS();
```

```
    }
  }

}

void gpsdump(TinyGPS &gps)
{
  long lat, lon;
  float flat, flon;
  unsigned long age;
  gps.f_get_position(&flat, &flon, &age);
    Serial.println("Sending SMS...");
  sim800l.print("AT+CMGF=1\r");
  delay(100);
  sim800l.print("AT+CMGS=
  \"+8801704054900\"\r");
  delay(500);
  sim800l.print("http://maps.google.
  com/maps?q=loc:");
  sim800l.print(flat == TinyGPS::
  GPS_INVALID_F_ANGLE? 0.0 : flat, 6);
  sim800l.print(",");
  sim800l.print(flon == TinyGPS::
  GPS_INVALID_F_ANGLE? 0.0 : flon, 6);
  sim800l.print((char)26);
  delay(500); sim800l.println();
  Serial.println("Text Sent.");
  delay(5000);
}

void printFloat(double number, int digits)
{
  // Handle negative numbers
  if (number < 0.00)
  {
     Serial.print('-');
     number = -number;
  }


  double rounding = 0.50;
  for (uint8_t i=0; i<digits; ++i)
    rounding /= 10.00;

  number += rounding;

  unsigned long int_part =
  (unsigned long)number;
  double remainder = number
  - (double)int_part;
  Serial.print(int_part);



  if (digits > 00)
    Serial.print(".");

  while (digits-- > 0)
  {
    remainder *= 10.00;
    int toPrint = int(remainder);
    Serial.print(toPrint);
    remainder -= toPrint;
  }
}

void sendLocationSMS()
  { sim800Serial.println("AT+CMGF=1");
  sim800Serial.println("AT+CMGS=
  \"+8801747904424\"");
  delay(100);
  sim800Serial.print("Latitude: ");
  sim800Serial.println(gps1.location.lat(), 6);
  delay(100);
  sim800Serial.print("Longitude: ");
  sim800Serial.println(gps1.location.lng(), 6);
  delay(100);
  sim800Serial.write((char)26);
  delay(1000);
}
```

References

[1] Paulinea, S. Jessica, et al. "Smart Shoe for Tracking and Monitoring of Army Soldiers." Smart Intelligent Computing and Communication Technology 38 (2021): 251.

[2] Ko, Li-Wei, et al. "Development of a smart helmet for strategical BCI applications." Sensors 19.8 (2019): 1867.

[3] Poulose, Alwin Senouci, Mohamed Han, Dong. (2019). "Performance Analysis of Sensor Fusion Techniques For Heading Estimation Using Smartphone Sensors." IEEE Sensors Journal. (2019).

[4] Janson, D., S. T. Newman, and V. Dhokia. "Next generation safety footwear." Procedia Manufacturing 38 (2019): 1668-1677.