Online-Journals.org

iJOE International Journal of Online and Biomedical Engineering

iJOE | eISSN: 2626-8493 | Vol. 19 No. 17 (2023) | OPEN ACCESS

https://doi.org/10.3991/ijoe.v19i17.42875

PAPER

Electronic Prototype of Autonomous Learning for the Crossing of Pedestrians with Visual Disabilities in Lima

Benjamin Luque-Milera[1], Alcides Alejandro Tocas-Taipe[1], Estefano Jose Luis Florian-Amaro[1], Axel Estrada-Ventocilla[1], Brahyan Fabián Salinas-Carbajal[1], Jhony Miguel Sanchez-Ramirez[1], Maritza Cabana-Cáceres[1,2], Cristian Castro-Vargas[1,2](✉)

[1]Facultad de Ingeniería, Universidad Privada del Norte, Lima, Perú

[2]Facultad de Ingeniería Electrónica e Informática, Universidad Nacional Federico Villareal, Lima, Perú

cristian.castro@upn.pe

ABSTRACT

Difficulties related to vehicular chaos and obstacles in public spaces hinder the orientation of visually impaired individuals, limiting their autonomy and exposing them to potential accidents. Considering these factors, the objective was to develop a prototype that facilitates autonomous learning by utilizing different electronic components. The aim is to ensure the safe movement of blind pedestrians, promote self-reliance, and minimize the risk of accidents. The proposed prototype is based on the concept of implementing intelligent traffic lights that detect the presence of pedestrians, allowing for safe crossing for both pedestrians and vehicles. The proposed circuit utilizes two ESP32 modules. One module is placed in the traffic light and configured as a Bluetooth master to transmit signals. It is also equipped with an ultrasonic sensor. The other module is located in the user's wristband and configured as a Bluetooth slave to receive signals. It is also equipped with a horn. The communication between the modules has been developed using the C programming language for microcontrollers in the Arduino IDE development environment. A master-slave communication system was implemented, resulting in the constant reporting of the distance between the pedestrian and the sidewalk within the pedestrian crossing by the ultrasonic sensor. This system controls the safe crossing by regulating the traffic lights. The HC-SR04 ultrasonic sensor can detect distances ranging from 2 cm to 450 cm. Therefore, the prototype can be used as a foundation for future advancements in various cities and contexts, ultimately benefiting blind pedestrians by improving their mobility.

KEYWORDS

ESP32 master-ESP32 slave, visual impairment, HC-SR04 ultrasonic sensor

1 INTRODUCTION

Currently, Peru lacks projects that promote the self-sufficiency of visually impaired individuals, address the vehicular chaos that creates noise, and fix the potholes in sidewalks and lanes. All of these issues hinder their orientation and make it difficult for them to navigate the streets of Lima. Visually impaired individuals are unable to

Luque-Milera, B., Tocas-Taipe, A.A., Luis Florian-Amaro, E.J., Estrada-Ventocilla, A., Salinas-Carbajal, B.F., Sanchez-Ramirez, J.M., Cabana-Cáceres, M., Castro-Vargas, C. (2023). Electronic Prototype of Autonomous Learning for the Crossing of Pedestrians with Visual Disabilities in Lima. *International Journal of Online and Biomedical Engineering (iJOE)*, 19(17), pp. 115–133. https://doi.org/10.3991/ijoe.v19i17.42875

Article submitted 2023-07-06. Revision uploaded 2023-09-10. Final acceptance 2023-09-10.

detect crosswalks on their own, and the mentioned obstacles force them to rely on a companion, putting them at risk of accidents. Traffic lights are essential for improving orientation and preventing accidents. In Peru, more than 47,600 traffic accidents were recorded in 2022 due to the majority of traffic lights being out of service or, in some cases, unavailable [1].

Several research studies have been carried out on the needs of visually impaired individuals in terms of safety and self-sufficiency in navigating public spaces [2]. Developing prototypes and alternative solutions can help facilitate their inclusion in the labor market and society. One of these initiatives is proposed in the research [3], where a portable assistance system based on artificial intelligence was developed for visually impaired individuals. This system utilizes edge computing to process data in real time. This system includes a pair of smart glasses, a smart cane, a mobile application, and a cloud-based management platform. Based on edge processing, data is analyzed, and responses are generated instantaneously, providing support and enhancing their experience in the environment where they operate.

Similarly, the smart traffic lights mentioned in [4] aim to improve traffic flow using advanced technology. Both initiatives share the use of innovative approaches to improve social aspects, such as labor inclusion and road efficiency.

In [5], the opinions of blind users were collected regarding the accessibility of community services and information services. The study highlighted inconveniences in finding people and their lack of friendliness, which hinders interaction with businesses. In [6], it was proposed to implement an intelligent preventive signaling system on the Serpentín de Pasamayo road. Through an exhaustive analysis of traffic volume, accident rates, and issues related to heavy vehicles on curves, we collected relevant information. Using Civil 3D's Vehicle Tracking software, we simulated the curve-turning maneuver. The obtained results supported the need to implement the proposed system in order to reduce accidents and enhance safety on the Serpentín de Pasamayo road. In addition, efforts have been made to enhance the accessibility and mobility of individuals with disabilities in urban areas by constructing accessible infrastructure and raising public awareness [7]. In addition, it aims to investigate the activities conducted by individuals with visual impairments and identify the factors that impact their growth. It also aims to identify the urban architectural barriers that hinder the orientation and mobility techniques used by these individuals [8].

In [9], a proposal was made to develop an AI-based cane that would assist visually impaired individuals. This cane would utilize a combination of camera and visual recognition technology. The integration of artificial intelligence and computer vision into this device could have a positive impact on the quality of life for visually impaired individuals.

In addition, as documented in [10], an advanced traffic light system was implemented to address the problem of traffic congestion in Kurt, Iraq. This system utilizes electronic components, including an Arduino microcontroller, a camera, and an infrared sensor. The system succeeded in reducing waiting times at all intersections and effectively coordinating traffic flow at a four-lane intersection. This coordination involves precisely adjusting the duration of the green light based on the volume of cars in each lane. This strategic adaptation facilitated the achievement of the project's main objective. In [11], an intelligent traffic control system is developed based on density principles to effectively regulate vehicular traffic. This system utilizes multiple components, such as ultrasonic sensors, an ESP8266 Wi-Fi module, an Arduino microcontroller, a Raspberry Pi3, an infrared sensor, a low-power microcontroller, a comparator, a radio communication module, and a cloud server. The implementation incorporates various testing methodologies, including functional, unit, integration, compatibility, and system testing. The objective of this system is to optimize traffic

flow, alleviate congestion, and minimize waiting times at traffic lights. The obtained results validate the system's ability to accurately detect traffic density. Furthermore, the system demonstrates its capability to dynamically adjust traffic light priorities based on the number of vehicles in the queue, thereby enhancing its efficiency.

These initiatives collectively demonstrate a dedication to addressing the requirements of individuals with visual impairments. They aim to enhance their independence, safety, and overall well-being within urban settings while also promoting their inclusion in different facets of society.

Therefore, the objective is to conceptualize and implement a prototype for traffic control. This prototype aims to help visually impaired pedestrians cross crosswalks safely, enabling them to develop autonomous skills. First of all, there is a need for a system of loudspeakers or sound devices that emit a distinctive sound to indicate when it is safe to cross the street. This system would allow users to autonomously learn to differentiate traffic light signals. In addition, a highly sensitive sensor system has been implemented to detect the presence of pedestrians in the crosswalk and enable them to cross when the corresponding traffic light for automobiles is on. This prototype of an electric autonomous learning system can enhance road safety and enable safe crossing for pedestrians who are blind.

2 METHODOLOGY

2.1 System design

The prototype utilizes two ESP32 modules with integrated Bluetooth: an ESP32 master that emits the control signal and another ESP32 slave that acts as the receiver. The operation of the traffic light is implemented using code written in the C language for Arduino. It involves the use of LEDs and 220-ohm resistors, a piezoelectric transducer known as a buzzer that can emit continuous tones, intermittent tones, or more complex sound patterns, and an ultrasonic sensor (HC-SR04) to measure the distance between the sensor and an object. This measurement is based on the principle of echolocation, as shown in Figure 1.

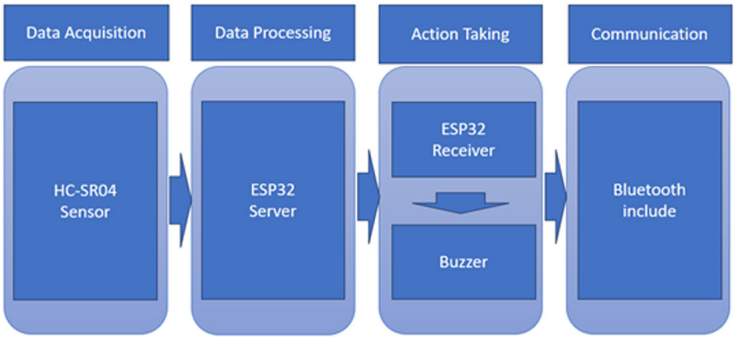

Fig. 1. System block diagram

The traffic light circuit we propose utilizes an HC-SR04 sensor to detect pedestrians in the crosswalk and adjust the timing of the traffic lights accordingly. Figure 2 and Table 1 show the characteristics of the sensor and its image.

The sensor works by emitting ultrasonic sound waves through its transducers and then detecting the resulting echo when these waves bounce off a nearby object. The echo is picked up by the second transducer. The distance is determined by measuring the time it takes for the echo to return to the sensor [12].

Fig. 2. Sensor HC-SR04

Table 1. Features of HC-SR04 sensor

Characteristics	Sensor HC-SR04 Technical Specification
Measuring range	2 cm–450 cm
Voltage	5 VDC
Minimum trigger pulse time (TTL level)	10 µS
Output echo pulse time (TTL level)	100–25000 µS
Communication time	20 milliseconds
Detection angle	15°
Precision	+– 3 mm
Ultrasonic frequency	40 Khz

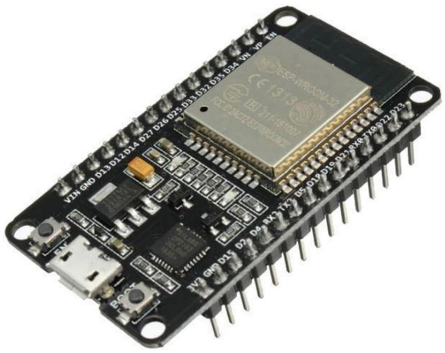

Fig. 3. ESP32 module

The ESP32 module enables wireless communication between devices, such as smartphones, computers, laptops, and other devices, using Bluetooth communication technology. It efficiently and securely transmits data [13]. Figure 3 and Table 2.

Table 2. Features of ESP32

Characteristics	ESP32
Platform	IoT development module
Microcontroller	Tensilica LX6 dual-core microcontroller
Inputs and Output	Increased number of digital and analog pins
Communication	Wi-Fi, Bluetooth, Arduino communication
Clock speed	Up to 240 MHz
Memory	Increased amount of RAM and flash storage
Operating voltage	3.3 V

2.2 Algorithm development

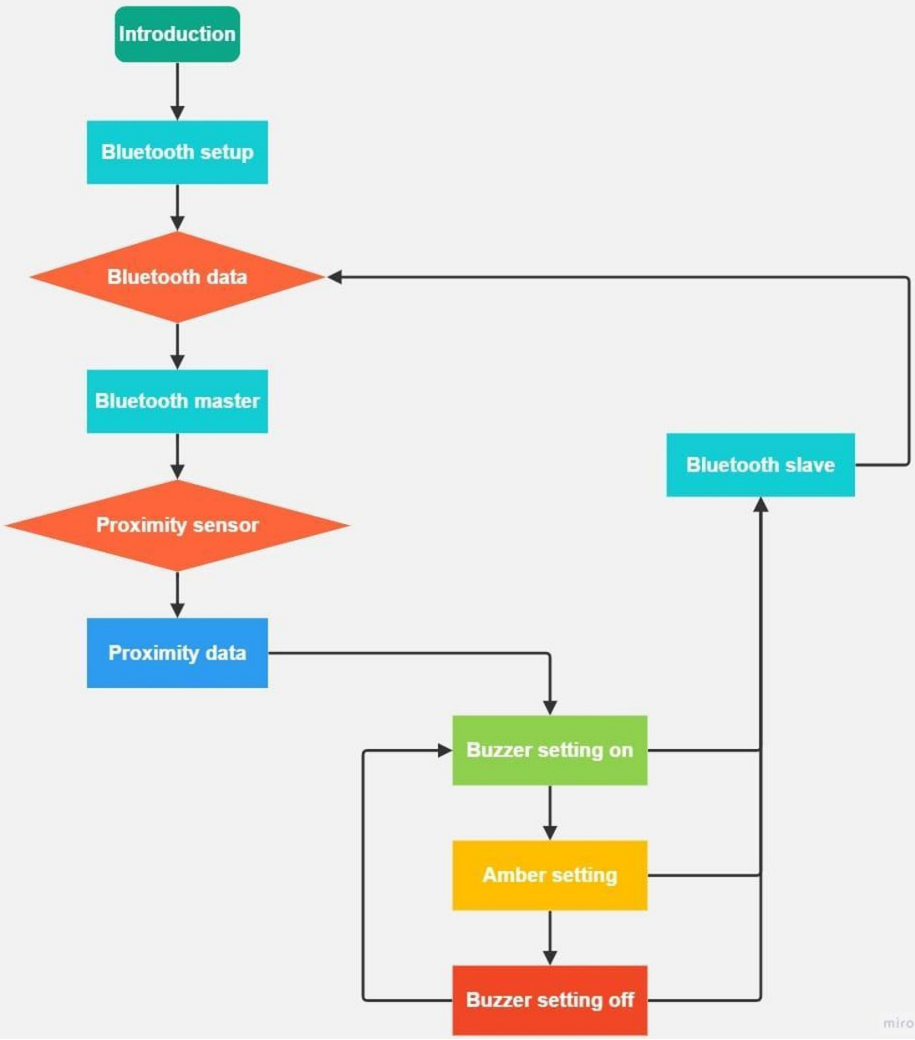

Fig. 4. Flowchart showing the development of the algorithm

The programming begins with the Bluetooth configuration of the ESP32 modules, and the operating modes and connection parameters are defined [14].

The transfer of proximity sensor data is initiated. The ESP32, acting as a Bluetooth master, collects sensor distance data to make decisions for traffic light control. Through programming, it can also detect nearby objects by receiving sensor data over the Bluetooth connection from the ESP32 slave.

The ESP32 slave receives sensor data in this function and utilizes that information to control the three light settings of the green, amber, and red traffic lights. These lights will be activated based on the data received from the Bluetooth slave. For example, if the proximity sensor detects an object nearby, the ESP32 slave will send a signal to the traffic light to activate the red light. If no object is detected, the green light will be activated [15]. To perform proximity reading with the sensor, we will utilize the following mathematical logic, along with their corresponding notations, as shown in equation 1 and Table 3.

$$v = \frac{2d}{t};\ d < y = m;\ d > y = n \tag{1}$$

Table 3. Notations

Notation	Meaning
d	The distance traveled is twice from pedestrian to traffic light in centimeters.
y	The distance required at a crosswalk is in centimeters.
t	Ultrasound travel time to pedestrian.
v	The speed of sound is 340m/s.
m	The red LED of the vehicle traffic light will remain on and the green LED of the pedestrian traffic light will remain on until the reading indicates d > y = n.
n	The vehicle and pedestrian traffic light LEDs will perform their regular sequence.

This creates a continuous cycle where the traffic light is updated in real time based on the sensor data.

The combination of the ESP32 and the proximity sensor enables the creation of an intelligent traffic light that adjusts to the surrounding conditions in real-time. This is shown in Figure 4.

2.3 Programming

The traffic light system with Bluetooth communication consists of two parts: the ESP32 Master and the ESP32 Slave. It is illustrated in the ESP32 master code configuration (see Figure 5).

```
//MASTER
#include "BluetoothSerial.h"
#if !defined(CONFIG_BT_ENABLED) || !defined(CONFIG_BLUEDROID_ENABLED)
#error Bluetooth is not enabled! Please run `make menuconfig` to and enable it
#endif
//ULTRASONIC SENSOR
const int trigPin = 5;
const int echoPin = 18;
//define sound speed in cm/uS
#define SOUND_SPEED 0.034
long duration;
float distanceCm;
//traffic light
int greenPin= 27;//green
int yellowPin = 26;//yellow
int redPin = 25;//red

int verdePin=33; //green
int amarilloPin=32;// yellow
int rojoPin=13;//red
BluetoothSerial BT; // Bluetooth
String clientName = "ESP32_LED_Control";
bool connected;
//**********************
void setup()
{
//ULTRASONIC SENSOR
  Serial.begin(9600); // Starts the serial communication
```

Fig. 5. ESP32 master code

Each step in programming the ESP32 master mode is detailed below.

```
#include "BluetoothSerial.h"
#if !defined(CONFIG_BT_ENABLED) || !defined(CONFIG_BLUEDROID_ENABLED)
#error Bluetooth is not enabled! Please run `make menuconfig` to and enable it
#endif
```

Fig. 6. Library inclusions and Bluetooth verification

This section includes the "BluetoothSerial.h" library that allows Bluetooth to be used in the ESP32. Next, it checks if the Bluetooth configuration is enabled. If it is not, an error message will be displayed (see Figure 6).

```
//ULTRASONIC SENSOR
const int trigPin = 5;
const int echoPin = 18;
//define sound speed in cm/uS
#define SOUND_SPEED 0.034
long duration;
float distanceCm;
//traffic light
int greenPin= 27;//green
int yellowPin = 26;//yellow
int redPin = 25;//red

int verdePin=33; //green
int amarilloPin=32;// yellow
int rojoPin=13;//red
BluetoothSerial BT; // Bluetooth
String clientName = "ESP32_LED_Control";
bool connected;
```

Fig. 7. Definition of pins and variables for ultrasonic sensor and traffic light

This section defines the pins used for the ultrasonic sensor and the LEDs of the traffic lights for both blinds and vehicles. A Bluetooth object named "BT" is also created, and a Boolean variable named "connected" is defined to verify whether a Bluetooth connection has been successfully established as illustrated in Figure 7.

```
void setup()
{
//ULTRASONIC SENSOR
  Serial.begin(9600); // Starts the serial communication
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
//*******************************************
  BT.begin("ESP32_client", true); // Name of your Bluetooth device and in master mode
  Serial.println("The Bluetooth device is in master mode. Connecting to the host ...");
   pinMode (redPin, OUTPUT);
   pinMode (yellowPin, OUTPUT);
    pinMode (greenPin, OUTPUT);// Change the LED pin to OUTPUT


   pinMode (verdePin, OUTPUT);
   pinMode (amarilloPin, OUTPUT);
   pinMode (rojoPin, OUTPUT);

  connected = BT.connect(clientName);
  if(connected) {
    Serial.println("¡Successfully connected!");
  } else {
    while(!BT.connected(10000)) {
      Serial.println("Could not connect. Make sure the remote device is available and within range, then restart the application.");
    }
  }
}
```

Fig. 8. Setup() function

The setup() function is executed only once at the start of the program. Here, the pins are configured as input or output as required. In addition, the Bluetooth connection to the remote device is established using the name “ESP32_client”. If the connection is successful, a success message will be printed. If the connection cannot be established within 10 seconds, an error message will be displayed, as shown in Figure 8.

```
void loop()
{
//ULTRASONIC
 // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculate the distance
  distanceCm = duration * SOUND_SPEED/2;
  // Prints the distance in the Serial Monitor
  Serial.print("Distance (cm): ");
  Serial.println(distanceCm);
   delay(1000);
//************TRAFFIC LIGHT FOR THE BLIND**************************
  BT.write(48);// VERDE // Envía 1 en ASCII
 Serial.println("48-8000  TRAFFIC LIGHT FOR THE BLIND START");
  digitalWrite(greenPin, HIGH);
  digitalWrite(yellowPin, LOW);
  digitalWrite(redPin, LOW);
  //CARTS
   digitalWrite(rojoPin, HIGH);
  digitalWrite(amarilloPin, LOW);
  digitalWrite(verdePin, LOW);
   delay(8000);
 Serial.println("48-8000  s TRAFFIC LIGHT FOR THE BLIND FINISHED");
  BT.write(49);//AMARILLO
  Serial.println("49-4000 TRAFFIC LIGHT FOR THE BLIND START");
  digitalWrite(yellowPin, HIGH);
  digitalWrite(redPin, LOW);
  digitalWrite(greenPin, LOW);
//CARTS
   digitalWrite(amarilloPin, HIGH);
  digitalWrite(verdePin, LOW);
  digitalWrite(rojoPin, LOW);
   delay(4000);
 Serial.println("49-4000 s TRAFFIC LIGHT FOR THE BLIND FINISHED");
    BT.write(50); // ROJO // Envía 0 en ASCII
   Serial.println("50-2000  STARTING");
  digitalWrite(redPin, HIGH);
  digitalWrite(yellowPin, LOW);
  digitalWrite(greenPin, LOW);
//CARTS
    digitalWrite(verdePin, HIGH);
  digitalWrite(amarilloPin, LOW);
  digitalWrite(rojoPin, LOW);
if(connected==true){
    while (distanceCm < 7)
                {
                ultra();
                delay(300);
                Serial.print("pedestrian crossing: ");
                Serial.println(distanceCm);
                }
}
  delay(2000);
  Serial.println("50-2000  s FINISHED");
  Serial.print("____log in connection_______");
  connected = BT.connect(clientName);
  Serial.print("____leaving connection_____");

  if(connected)
  {
    Serial.println("¡Successfully connected!");
  } else {
   }

}
```

Fig. 9. Loop() function

The loop() function runs continuously after the setup() function. Here, it reads the data from the ultrasonic sensor and calculates the distance in centimeters using echolocation. The distance is displayed on the serial monitor, and a one-second delay is performed before continuing. Next, the blind traffic light control is performed using the pins that were defined above. The state of the LEDs is adjusted based on the specified order and duration for each phase of the traffic light. If a Bluetooth connection is established, a while loop is activated to continuously check the distance of the ultrasonic sensor. If the distance is less than 7 cm, the "ultra()" function is executed, and the distance is displayed on the serial monitor. Afterward, there is a two-second delay, and a Bluetooth reconnection attempt is made, as seen in Figure 9.

```
void ultra()
{
//ULTRASONIC
 // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculate the distance
  distanceCm = duration * SOUND_SPEED/2;
  // Prints the distance in the Serial Monitor
  Serial.print("Distance (cm): ");
  Serial.println(distanceCm);
  delay(1000);
}
```

Fig. 10. Ultra() function

The ultra() function is used to perform a single measurement of the ultrasonic sensor. It performs the same procedure explained in the loop() function to calculate the distance and display it on the serial monitor (see Figure 10).

The ESP32 slave mode configuration is shown below.

```
// SLAVE
#include "BluetoothSerial.h"
#if !defined(CONFIG_BT_ENABLED) || !defined(CONFIG_BLUEDROID_ENABLED)
#error Bluetooth is not enabled! Please run `make menuconfig` to and enable it
#endif
//BLIND
int LED1 =26;//green
int LED2= 33;//yellow
int LED3 = 32;//red
int buzzerPin = 23;
BluetoothSerial BT; // Bluetooth
void setup() {
  Serial.begin(9600); // Initialization of the serial connection for debugging
  BT.begin("ESP32_LED_Control"); // Name of your Bluetooth device and in slave mode
  Serial.println("The Bluetooth device is ready for pairing");
  pinMode (LED1, OUTPUT);
   pinMode (LED2, OUTPUT);
    pinMode (LED3, OUTPUT);// Change the LED pin to OUTPUT
      pinMode(buzzerPin, OUTPUT);
}
void loop()
{
  if (BT.available()) // Check if we receive anything from Bluetooth
  {
    int incoming = BT.read(); // Read what we received
    Serial.print("Received:... ");
    Serial.println(incoming);
```

Fig. 11. ESP32 slave code

Each step in the programming of the ESP32 slave mode, as shown in Figure 11, is detailed below.

```
#include "BluetoothSerial.h"
```

Fig. 12. Library import

This line includes the "BluetoothSerial.h" library, which enables Bluetooth communication on the ESP32. Figure 12.

```
#include "BluetoothSerial.h"
#if !defined(CONFIG_BT_ENABLED) || !defined(CONFIG_BLUEDROID_ENABLED)
#error Bluetooth is not enabled! Please run `make menuconfig` to and enable it
#endif
```

Fig. 13. Verification of the Bluetooth configuration

These lines check whether Bluetooth is enabled in the ESP32 configuration. If it is not enabled, an error message will be displayed as seen in Figure 13.

```
int LED1 =26;//green
int LED2= 33;//yellow
int LED3 = 32;//red
int buzzerPin = 23;
```

Fig. 14. Pin statement

These variables represent the pins to which the LEDs and the buzzer are connected. Figure 14.

```
BluetoothSerial BT; // Bluetooth
```

Fig. 15. Object of the BluetoothSerial class

An instance of the "BluetoothSerial" class called BT, is created to enable Bluetooth communication as seen in Figure 15.

```
void setup() {
  Serial.begin(9600); // Initialization of the serial connection for debugging
  BT.begin("ESP32_LED_Control"); // Name of your Bluetooth device and in slave mode
  Serial.println("The Bluetooth device is ready for pairing");
  pinMode (LED1, OUTPUT);
   pinMode (LED2, OUTPUT);
    pinMode (LED3, OUTPUT);// Change the LED pin to OUTPUT
      pinMode(buzzerPin, OUTPUT);
}
```

Fig. 16. Configuration in the setup() function

In the setup() function, the components are initialized and the pins are configured. The serial communication is initiated at a baud rate of 9600, which is followed by the activation of Bluetooth with the name "ESP32_LED_Control." Additionally, the LED and buzzer pins are set as output pins, as shown in Figure 16.

```
void loop()
{
  if (BT.available()) // Check if we receive anything from Bluetooth
  {
    int incoming = BT.read(); // Read what we received
    Serial.print("Received:... ");
    Serial.println(incoming);
```

Fig. 17. Main loop loop()

In the main loop(), it checks if data is available via the Bluetooth connection. If data is available, the received value is read and stored in the incoming variable, as shown in Figure 17.

```
    if (incoming == 48)  // 8000
    { //GREEN
      for (int i=0; i <10; i++)
      {
        digitalWrite(LED1, HIGH); // LED ON
       //BT.println("LED1 ON");
        Serial.println("48-GREEN ON");
        buzz(buzzerPin, 500, 800);
      }
     digitalWrite(LED1, LOW);
     Serial.println("48-GREEN- OFF");
    }
/*****************************************************/
    if (incoming ==49)
    {  //YELLOW
      for (int i=0; i <10; i++)
      {
      digitalWrite(LED2, HIGH); // LED ON
      Serial.println("49-YELLOW- ON");
        buzz(buzzerPin, 1500,365);
      } digitalWrite(LED2, LOW);
      Serial.println("49-YELLOW- OFF");
    }
    if (incoming == 50)
    {   //RED
      for (int i=0; i <10; i++)
      {
       digitalWrite(LED3, HIGH); // LED ON
       Serial.println("50-RED- ON");
       buzz(buzzerPin, 2500, 200);
      }
      digitalWrite(LED3, LOW);
      Serial.println("50-RED- OFF");
    }
  }
}
```

Fig. 18. LEDs and buzzer control

Depending on the value received in “incoming”, specific actions are taken to control the LEDs and the buzzer. If the incoming value equals 48, the green LED will be turned on and off, and a beep tone will be emitted. If the incoming value is equal to 49, a similar control is performed for the yellow LED. If the incoming value is equal to 50, a similar control is performed for the red LED, as illustrated in Figure 18.

```
void buzz(int targetPin, long frequency, long length)
{
long delayValue = 1000000/frequency/2; // calculate the delay value between transitions
long numCycles = frequency * length/ 1000; // calculate the number of cycles for proper timing
for (long i=0; i < numCycles; i++){ // for the calculated length of time...
digitalWrite(targetPin,HIGH); // write the buzzer pin high to push out the diaphram
delayMicroseconds(delayValue); // wait for the calculated delay value
digitalWrite(targetPin,LOW); // write the buzzer pin low to pull back the diaphram
delayMicroseconds(delayValue); // wait againf or the calculated delay value
}
```

Fig. 19. Buzz() function

This function is used to generate sound tones using the buzzer. It takes three arguments: the buzzer pin (target), the frequency of the tone in hertz (frequency), and the duration of the tone in milliseconds (length). It utilizes the on/off cycles of the buzzer pin, along with calculated delay times, to generate the desired tone (Figure 19).

2.4 System design and implementation

The ESP32 master module transmits information via Bluetooth to the ESP32 slave module, which then receives the information. Additionally, the HC-SR04 ultrasonic sensor will establish a connection with the ESP32 master, which will verify if there are no pedestrians crossing the road. In addition, the buzzer will emit a unique sound when the corresponding-colored LEDs illuminate. Both ESP32 master-slave circuits can be seen in Figures 20 and 21. On the other hand, if pedestrians are crossing the road, the vehicular traffic light will remain on the red LED because the sensor detects the presence of a pedestrian at a close distance. If the sensor does not detect the presence of a pedestrian, the vehicular traffic light can continue its normal process by turning on the green LED and allowing vehicles to proceed. This system improves both pedestrian and vehicular safety.

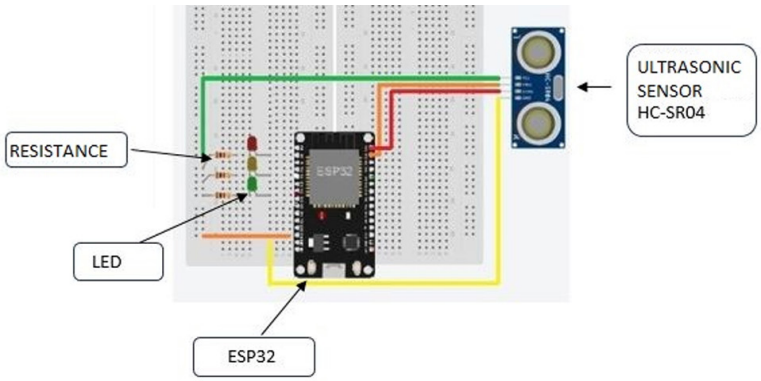

Fig. 20. ESP32 master circuit

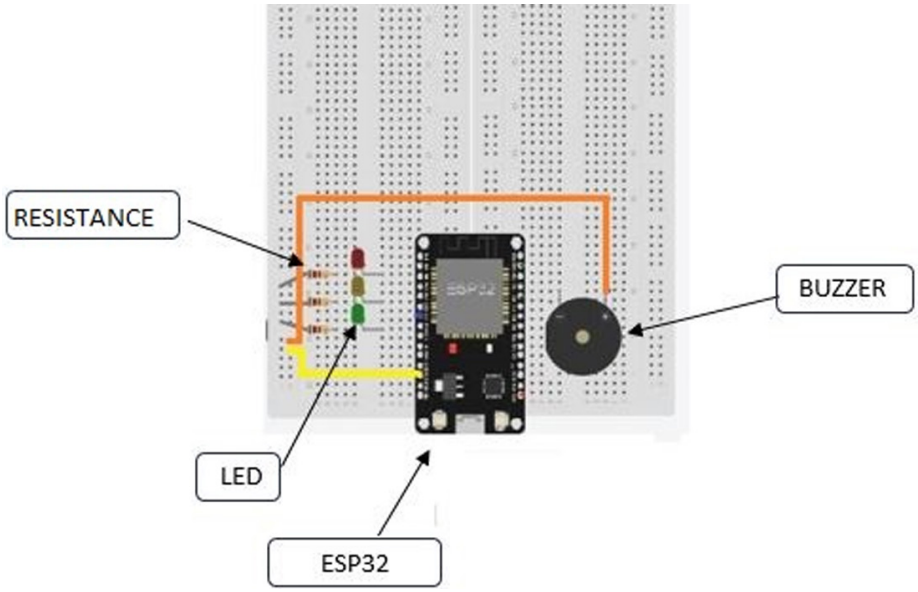

Fig. 21. ESP32 slave circuit

3 RESULTS AND DISCUSSION

The prototype was developed using two ESP32 master-slave modules that have integrated Bluetooth. This eliminates the need for an additional Bluetooth HC-05 module. The master ESP32 emits the signal, which is then received by the slave ESP32. We implemented the operation of the traffic light using code in the Arduino IDE application. The prototype also includes a piezoelectric transducer that emits continuous and intermittent tones [16]. You can observe the simulation and the mode of use in Figures 22–25. To measure the distance, we utilize an ultrasonic sensor (HC-SR04). This sensor operates on the principle of echolocation to calculate the distance between the sensor and an object. It is used to control the crossing of cars, allowing or restricting their movement. If the sensor detects pedestrians crossing the road when the traffic light is green, it triggers a code process, as shown in Figure 9. Overall, the project demonstrates the feasibility of enhancing the quality of life and promoting self-sufficiency among individuals by facilitating their daily activities. Electronic boards, such as Arduino and HC-05, can also be used because this smart traffic light technology is related to road safety. These boards incorporate sensors, cameras, and fiber optic networks to prevent possible accidents.

Fig. 22. Watch with ESP32 slave

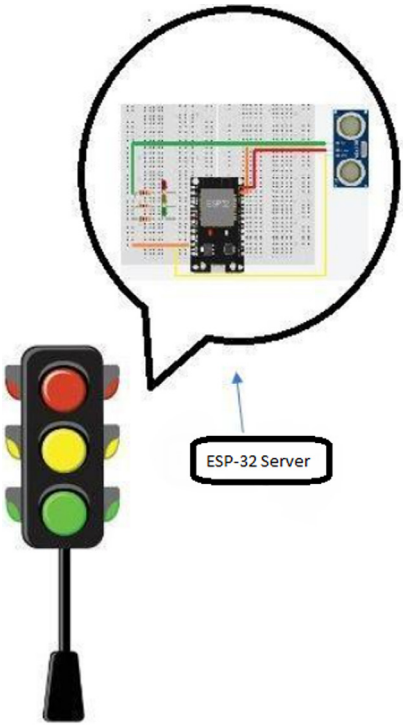

Fig. 23. The function of the Esp32 master in the traffic light

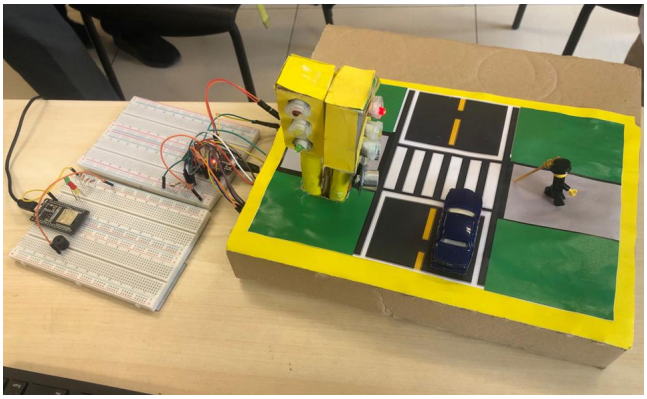

Fig. 24. Model

Fig. 25. Simulation

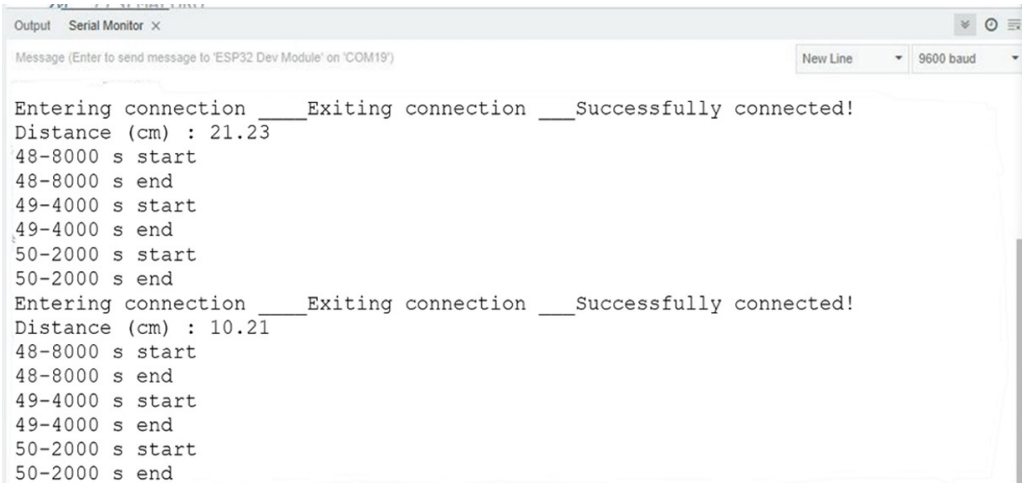

Fig. 26. Esp32 master sending information to Esp32 slave

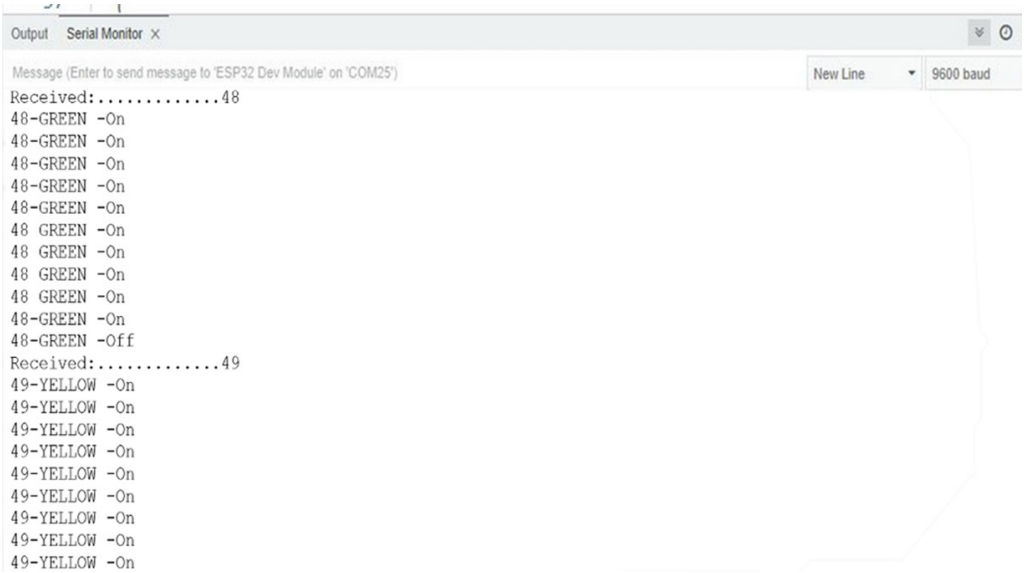

Fig. 27. Esp32 slave receiving the information sent from the Esp32 master

The implementation of the circuit has proven to be beneficial for individuals with visual impairments. It works using ESP32 technology, facilitated through a Bluetooth-enabled circuit. This is illustrated in Figures 20 and 21. When the traffic light LED displays red, green, or yellow signals, the slave ESP32 receives the corresponding data from the master ESP32 and generates an audible signal to indicate the color difference using a buzzer. Concerning the pedestrian detection test, fast communication is established as the master ESP32 transmits its signal to the slave ESP32. This is evident in the graphs presented in Figures 26, 27, 28. When the LED signal is red for vehicles, they stop, while a green LED activates pedestrian traffic. These states are maintained until the last person finishes crossing the crosswalk. Subsequently, when no pedestrians are detected, the sensor resumes the regular sequence of traffic light changes for vehicles. The sequence ensures proper synchronization in the data flow.

Table 4. Module comparison

Characteristics	Arduino	ESP32
Platform	Electronic development board	IoT development module
Microcontroller	ATmega328P (Arduino UNO)	Dual-core Tensilica LX6 microcontroller
Inputs and Outputs	Limited digital and analog pins	Increased number of digital and analog pins
Communication	UART, SPI, I2C, serial port	Wi-Fi, Bluetooth, Arduino communication
Clock speed	Up to 20MHz	Up to 240MHz

The ESP32 is a more powerful and versatile option compared to the Arduino UNO, as shown in Table 4. It features a dual-core microcontroller, a larger pin count, Wi-Fi connectivity with a second core for handling more complex IoT projects, Bluetooth capability, and compatibility with the Arduino development environment.

Fig. 28. Detection graph in centimeters

Table 5. Results table

Metrics	Result
Functionality	Tactile and auditory feedback based on simulated lights with Bluetooth.
Detection accuracy	85% accuracy in the detection of simulated traffic light states via Bluetooth.
Bluetooth connection	Stable and reliable connection between the traffic light and the simulated device.
Usability	Intuitive and accessible design and interaction for blind users.
Feedback	Positive feedback on the usefulness and effectiveness of tactile and auditory feedback.
Social impact	Improved road safety and accessibility for the visually impaired.

The system in question is a solution that provides tactile and auditory feedback based on simulated traffic lights through a Bluetooth connection. Its detection accuracy reaches 85% in identifying simulated traffic light states, and the Bluetooth

connection is characterized by its stability and reliability between the traffic light and the simulated device [17]. Table 5 highlights its usability, intuitive and accessible design, and interaction for blind users. The comments received have been mostly positive, with many highlighting the usefulness and effectiveness of tactile and auditory feedback. In addition, this system has a significant social impact by enhancing road safety and accessibility for individuals with visual impairments.

4 CONCLUSION

- The technology based on microcontrollers facilitates learning by providing necessary orientation for blind people.
- This prototype focuses on portability, as the bracelet is adaptable and interchangeable among pedestrians who require orientation.
- It also has a positive impact on traffic and in areas with high transportation demand. Additionally, it helps ensure the safety of pedestrians, which is particularly important in densely populated cities that face congestion and traffic accidents.
- Digital wireless communication converts visual signals into auditory signals that a blind person can interpret and understand.
- In short, this technological solution addresses the prevalent lack of technology in the country, thereby assisting visually impaired individuals. At the same time, it emphasizes the significance of including citizens, ensuring their safe transit and benefiting them.

5 REFERENCES

[1] Caretas Nacional, "Estudiantes crean equipo tecnológico para detectar accidentes interprovinciales en tiempo real – Caretas Nacional." https://caretas.pe/nacional/estudiantes-crean-equipo-tecnologico-para-detectar-accidentes-interprovinciales-en-tiempo-real/ [Accessed Jun. 23, 2023].

[2] Oscar Daniel Segura Medranda, "Escuela Superior Politécnica De Chimborazo Facultad De Informática Y Electrónica."

[3] W. J. Chang *et al.*, "An AI edge computing based W earable assistive device for visually impaired people zebra-crossing walking," in *Digest of Technical Papers – IEEE International Conference on Consumer Electronics*, Institute of Electrical and Electronics Engineers Inc., 2020, pp. 1–2. https://doi.org/10.1109/ICCE46568.2020.9043132

[4] Patricia Mendoza, "SOLUCIÓN PARA LA DETECCIÓN DE CRUCES PEATONALES Y OBSTÁCULOS POR PERSONAS CON DISCAPACIDAD VISUAL."

[5] María De los Ángeles TERRAZAS GARCIA, "Accesibilidad a la información de usuarios con discapacidad visual a la Biblioteca Central 'Pedro Zulen' de la UNMSM," 2018.

[6] E. Garibay Ortiz and B. Garibay Ortiz, "UNIVERSIDAD PERUANA DE CIENCIAS APLICADAS FACULTAD DE INGENIERÍA PROGRAMA ACADÉMICO DE INGENIERÍA CIVIL."

[7] EL COMERCIO PERÚ, "Personas con discapacidad: El desafío de ganarle a la mala accesibilidad en las calles de Lima | LIMA | EL COMERCIO PERÚ." https://elcomercio.pe/lima/sucesos/personas-con-discapacidad-el-desafio-de-ganarle-a-la-mala-accesibilidad-en-las-calles-de-lima-noticia/

[8] YUREMMY MARIELY VILCANQUI APAZA, "VILCANQUI_YUREMMY_MOVILIDAD_DISCAPACIDAD_VISUAL".

[9] "Un bastón 'inteligente' para que los ciegos puedan ir a la compra o sentarse en un bar." https://www.eldebate.com/salud-y-bienestar/salud/20230129/baston-inteligente-ciegos-puedan-compra-sentarse_87557.html

[10] Abdul Hadi M. Alaidi *et al.*, "View of design and implementation of a smart traffic light management system controlled wirelessly by Arduino," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 14, no. 7, pp. 32–39, 2020. https://doi.org/10.3991/ijim.v14i07.12823

[11] Mashael Khayyat *et al.*, "The Simulation and prototyping of a density-based smart traffic control system for learning purposes," *International Journal of Online and Biomedical Engineering (iJOE)*, vol. 16, no. 12, pp. 70–81, 2020. https://doi.org/10.3991/ijoe.v16i12.16849

[12] "ARDUINO: Sensor ultrasónico HC-SR04 | Blog de Tecnología – IES José Arencibia Gil – Telde." https://www3.gobiernodecanarias.org/medusa/ecoblog/fsancac/2018/02/06/arduino-sensor-ultrasonico-hc-sr04/

[13] "NodeMCU-32 30-pin ESP32 WiFi." https://naylampmechatronics.com/espressif-esp/384-nodemcu-32-30-pin-esp32-wifi.html

[14] "Los módulos inalámbricos ESP32 simplifican el diseño IoT | DigiKey." https://www.digikey.com/es/articles/how-to-select-and-use-the-right-esp32-wi-fi-bluetooth-module

[15] "ESP-Now conecta dos o más ESP32/ESP8266 – ElectroSoftCloud." https://www.electrosoftcloud.com/esp-now-conecta-dos-o-mas-esp32-esp8266/

[16] "Buzzer | Arduino." http://ceca.uaeh.edu.mx/informatica/oas_final/OA4/buzzer.html. [Accessed Jun. 24, 2023].

[17] "Semáforos inteligentes, una apuesta necesaria para optimizar la movilidad – CCIT – Cámara Colombiana de Informática y Telecomunicaciones." https://www.ccit.org.co/articulos-tictac/semaforos-inteligentes-una-apuesta-necesaria-para-optimizar-la-movilidad/

6 AUTHORS

Benjamin Luque-Milera is a student of Computer Systems Engineering at the Universidad Privada del Norte, Lima, Peru, focused on project management, application development (E-mail: N00306888@upn.pe).

Alcides Alejandro Tocas-Taipe is a student of Computer Systems Engineering at the Universidad Privada del Norte, Lima, Peru, focused on project management, application development (E-mail: N00296738@upn.pe).

Estefano Jose Luis Florian-Amaro is a student of Computer Systems Engineering at the Universidad Privada del Norte, Lima, Peru, focused on project management, application development (E-mail: N00245643@upn.pe).

Axel Estrada-Ventocilla is a student of Computer Systems Engineering at the Universidad Privada del Norte, Lima, Peru, focused on project management, application development (E-mail: N00282574@upn.pe).

Brahyan Fabián Salinas-Carbajal is a student of Computer Systems Engineering at the Universidad Privada del Norte, Lima, Peru, focused on project management, application development (E-mail: N00295486@upn.pe).

Jhony Miguel Sanchez-Ramirez is an electronic engineer graduated from the Technological University of Peru, specializing in programming and development of microcontroller systems, also work as supervisor of electronic engineering laboratories, networks and automation in engineering program of the Universidad Privada del Norte also as professor of industrial electronics at the Instituto Superior Metropolitano Lima-Peru (E-mail: jhony.sanchez@upn.edu.pe).

Maritza Cabana-Cáceres holds a PhD in Education, Master's in University Teaching, Electronic Engineer (UNAC). She has experience as a University Chair, coordination in Electronic Engineering, and Professor Researcher with publications in journals indexed. Reviewer of articles in indexed journals, Thesis Advisor, and Jury for undergraduate and postgraduate courses. She also has experience as a Consultant in research, thesis, electronic projects, IA, automation, and telecommunications (E-mail: maritza.cabana@upn.pe).

Cristian Castro-Vargas is an Electronic Engineer who graduated from the National University of Callao, and has a Doctorate in Education, as well as a Master's degree in university teaching, with extensive experience in research, developing electronic projects, networks, automation, and telecommunications and has publications in Scopus. He carries out university teaching using active methodologies of different specialties in the line of electronics and systems at various universities in Peru, a member of the IEEE-RAS (E-mail: cristian.castro@upn.pe).