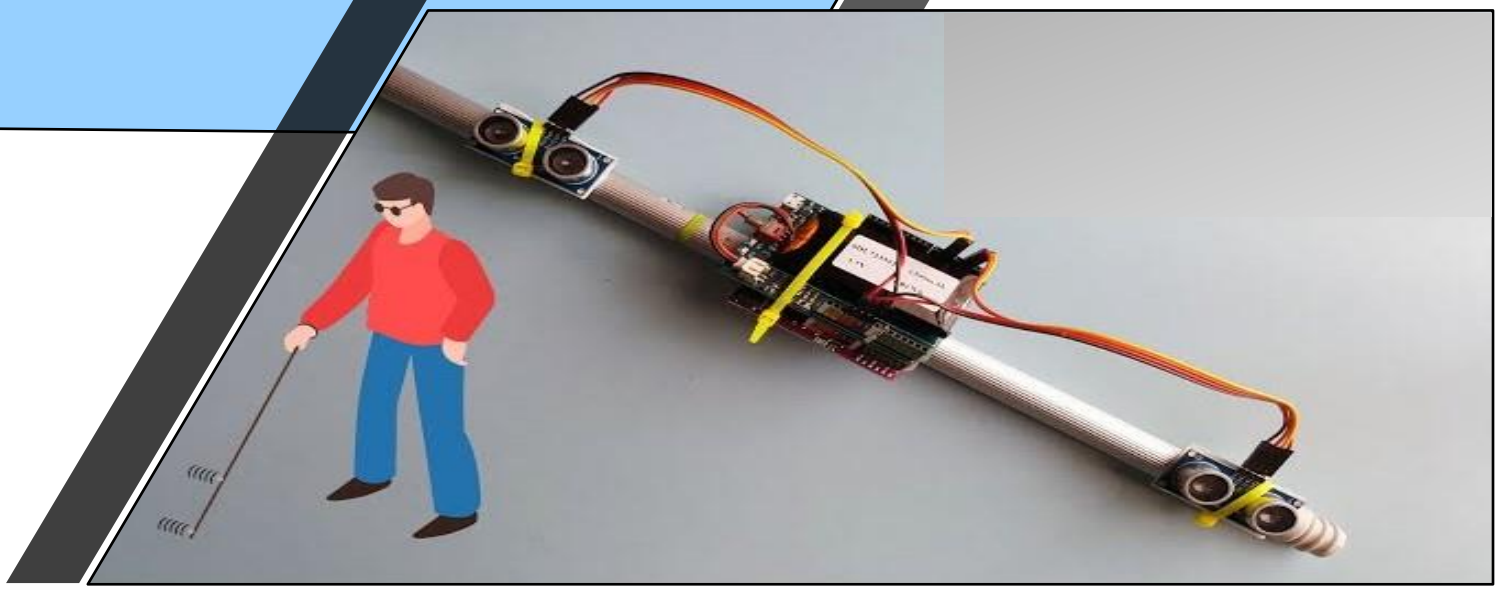
Kelistan Ketheeswaran [AS2020943]

Embedded System

Smart Blind Stick (Progress Report)



Content

Introduction

❖ Background and Significance

The Embedded System Smart Blind Stick project is a transformative endeavor that addresses a critical need within the community of visually impaired individuals. Visual impairment is a widespread and life-altering challenge, impacting millions of people worldwide. Navigating through various environments, whether indoors or outdoors, can be daunting and fraught with obstacles for the blind and visually impaired.

The primary motivation behind this project is to enhance the independence, safety, and overall quality of life for those with visual impairments. The project aims to design and implement an intelligent, sensor-driven system that assists users in detecting obstacles, providing real-time feedback, and enabling more confident navigation.

❖ Objective

The main objective of this project is to develop a Smart Blind Stick that leverages cutting-edge embedded technology to mitigate the challenges faced by visually impaired individuals when navigating their surroundings. By creating a reliable and adaptive solution, our goal is to empower users to explore and move through their environment with increased ease and confidence, ultimately improving their mobility and fostering a sense of self-sufficiency.

❖ Purpose

The purpose of the Smart Blind Stick project is to enhance the independence and safety of visually impaired individuals by providing them with technologically advanced mobility aid. This innovative device is designed to detect obstacles, both above ground and at ground level, and to provide real-time feedback to the user through auditory or tactile signals. By addressing the challenges faced by the visually impaired in navigating their surroundings, the project aims to empower them with improved mobility, increased confidence, and a greater sense of self-reliance, ultimately leading to a more inclusive and accessible world for all.

Scope of the system

❖ Scope of the Smart Blind Stick system key aspects:

- ✓ **Navigation Assistance:** The primary objective of the system is to provide comprehensive navigation assistance to visually impaired individuals. This includes detecting obstacles and guiding the user safely in indoor and outdoor environments.
- ✓ **Obstacle Detection:** The system is designed to detect a wide range of obstacles, such as walls, furniture, stairs, curbs, and low-hanging objects. It uses an ultrasonic sensor.
- ✓ **Real-time Feedback:** The system provides real-time feedback to the user through a combination of haptic (**Vibration Motor**), auditory (**Buzzer Tone**), and/or tactile feedback mechanisms. This feedback informs the user about the location and nature of obstacles and helps them navigate around them.
- ✓ **Environmental Adaptability:** The system is adaptable to different environmental conditions and lighting situations. It can function effectively both indoors and outdoors and can accommodate variations in lighting and terrain.
- ✓ **User-Centric Design:** The Smart Blind Stick is designed with the user's needs and preferences in mind. It is ergonomically designed for comfort and ease of use. The feedback mechanisms can be customized to suit the user's preferences, such as adjusting the intensity of vibrations or the type of auditory cues.
- ✓ **Scalability and Upgradability:** The Smart Blind Stick is designed with the potential for future enhancements and updates. This includes the ability to add new sensors, improve algorithms, and incorporate user feedback to continually enhance its functionality.
- ✓ **Limitations:** It's important to acknowledge the limitations of the system. For example, the system may not be able to detect transparent obstacles, and it may have constraints related to the accuracy of obstacle detection at different distances.

Requirements

❖ Functional Requirements:

- ✓ **Obstacle Detection:** The device must detect obstacles in the user's path using sensors.
- ✓ **Distance Measurement:** Provide distance information about detected obstacles to the user.
- ✓ **User Feedback:** Offer audio or tactile feedback to convey obstacle location and proximity.
- ✓ **Mobility Support:** Enable safe navigation in various environments.
- ✓ **Switchable Modes:** Have different modes for different scenarios.
- ✓ **Battery Management:** Efficient power usage and battery warnings.
- ✓ **User Interface:** User-friendly controls and configuration options.
- ✓ **Durability and Portability:** Lightweight and rugged design.
- ✓ **Voice Assistance:** Optional voice commands or audio feedback.

❖ Non-Functional Requirements:

- ✓ **Reliability:** The system should consistently and accurately detect obstacles.
- ✓ **Safety:** It must not cause harm through false alarms or malfunctions.
- ✓ **Accuracy:** The device's measurements and detection should be precise.
- ✓ **Real-time Responsiveness:** The system must provide immediate feedback.
- ✓ **Usability:** The user interface should be designed for individuals with visual impairments and be easy to use.
- ✓ **Environmental Considerations:** The system should operate in various environmental conditions.
- ✓ **Regulatory Compliance:** It must adhere to relevant regulations and standards.
- ✓ **Cost-effectiveness:** The project should be affordable in terms of production and maintenance.

Hardware Components

- **Arduino Uno -** I used an Arduino Uno board for this project. It's commonly known as the easiest product to program and to apply hardware connections.
- **Two ultrasonic sensors –** The ultrasonic sensors detect the distance in front of them, so I use them to detect how far away obstacles are.
    - In this prototype I will work only on the stick, for this I need two ultrasonic sensors, one attached to the stick to measure the **depth** under the stick and the other to a **servo motor** to detect obstacles in front of it at an **angle of 180 degrees**.
- **LDR Sensor -** It is used to indicate the presence or absence of light or to measure the intensity of light.
- **Servo Motor -** A servo motor is used to rotate the ultrasonic sensor to detect obstacles through an angle of 180 degrees.
- **ESP8266 ESP01 -** Android App (**RemoteXY**) Detects (**Vibrate & Sound**) Blind Stick system directly from app via Wi-Fi connection.
- **Printed (Copper) Circuit Board -** I used a printed (copper) circuit board and turned it into a tiny little sub-system.
- **Buzzer & Vibrator -** These are used to provide information to the user. These vibrate and provide information on the location of obstacles through sound (**Different Tone types**).
- **Resistor's –** It is used to lower the flow of current, divide voltages, block transmission signals, and bias active elements.
- **LED –** Blink rate is used to identify obstacle distance status and computer on/off status.
- **Jumper wires –** Connect the Sensors to the Arduino Uno board system.
- **Two 9 Volt Battery –** Power (Voltage) to the system work.
- **Wood Stick –** Fix Blind Stick System stuff on the stick.
- **Nylon cable ties –** Attach the wire connection tightly to the stick.
- **Switch** – Power on/off the system.

Concept of Working

- **Two Ultrasonic Sensors:**
  - Sensor 1 detects obstacles in front and provides distance measurements within a 180-degree range, ensuring obstacle awareness.
  - Sensor 2 measures the stick's depth to the ground, helping maintain a consistent height and avoid tripping.
- **LDR Sensor (Light Dependent Resistor):**
  - The LDR sensor detects ambient light, allowing the system to adjust feedback based on lighting conditions for user comfort and awareness.
- **Buzzer and Vibrator for Alert Purpose:**
  - The buzzer and vibrator provide immediate alerts to the user when obstacles are detected, enhancing safety through audible and tactile feedback.
- **ESP8266 ESP01 and Android App (RemoteXY):**
  - The ESP8266 module connects to an Android app via Wi-Fi, enabling remote control, alert customization, and real-time data monitoring for the user.
- **Arduino Uno:**
  - Arduino Uno processes data from sensors, calculates obstacle distances, and coordinates alerts and communication with the Android app, ensuring accurate and timely feedback for users.

- The Embedded System Smart Blind Stick incorporates two ultrasonic sensors, an LDR sensor, a buzzer, a vibrator, an ESP8266 module with an Android app, and an Arduino Uno to assist visually impaired individuals in their navigation. The first ultrasonic sensor detects obstacles and measures the distance to objects in the front 180-degree range, while the second sensor determines the depth to the ground below the stick. The LDR sensor detects ambient light conditions. When obstacles are detected, the system triggers alerts through the buzzer and vibrator, adjusting their intensity based on obstacle proximity. The ESP8266 module connects to an Android app via Wi-Fi, allowing users to remotely control the stick and receive feedback. The Arduino Uno processes sensor data, calculates obstacle distances, and coordinates the alerts, ensuring users can safely navigate their environment with real-time guidance and customized feedback.

Block Diagram

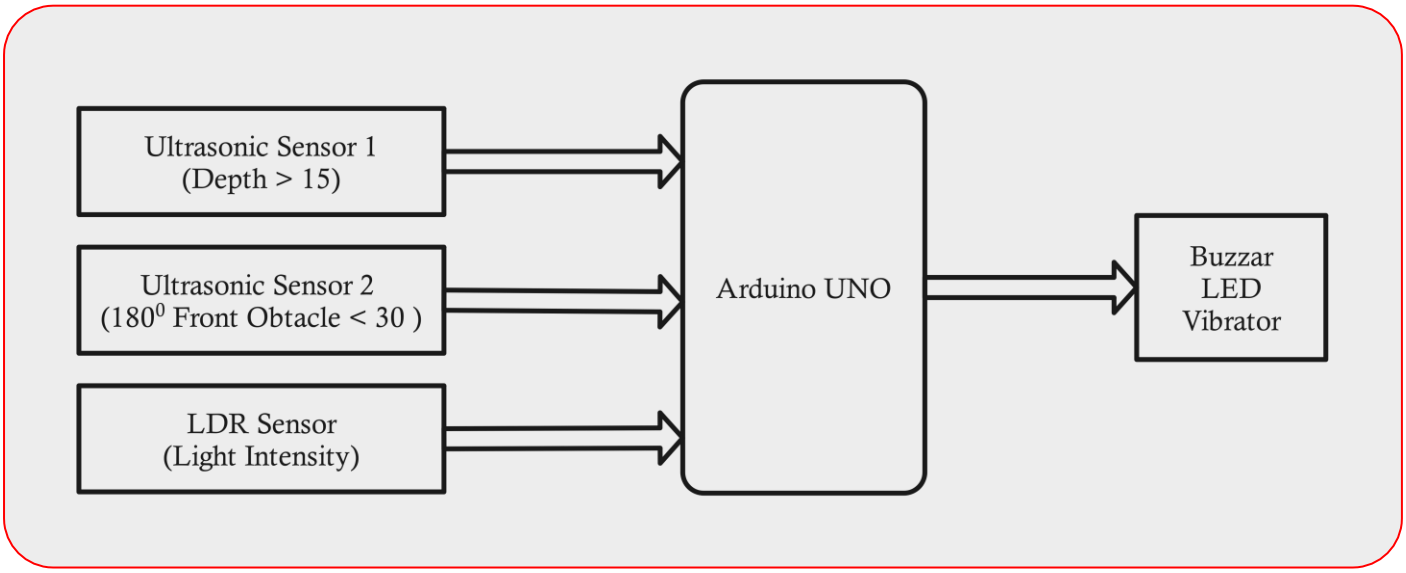


Figure 1: Blind Stick System

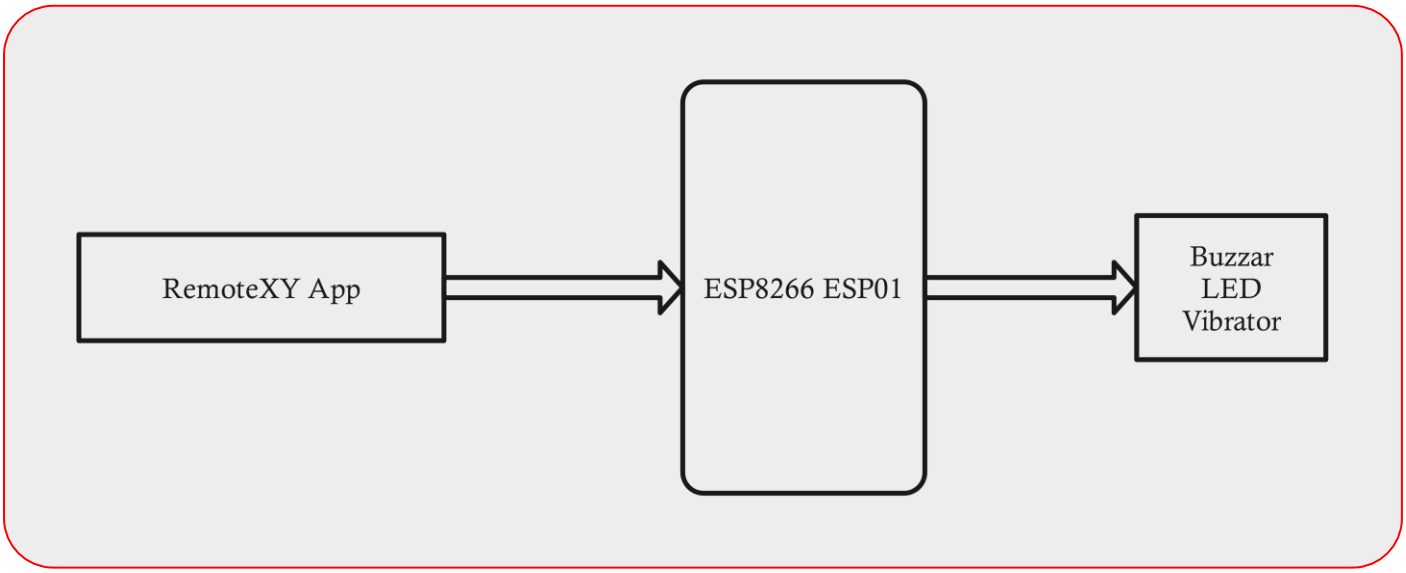
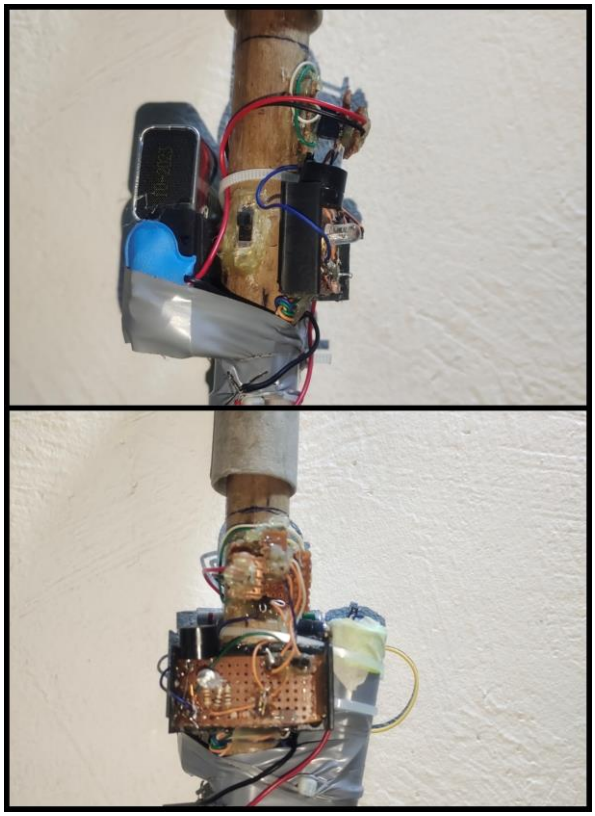
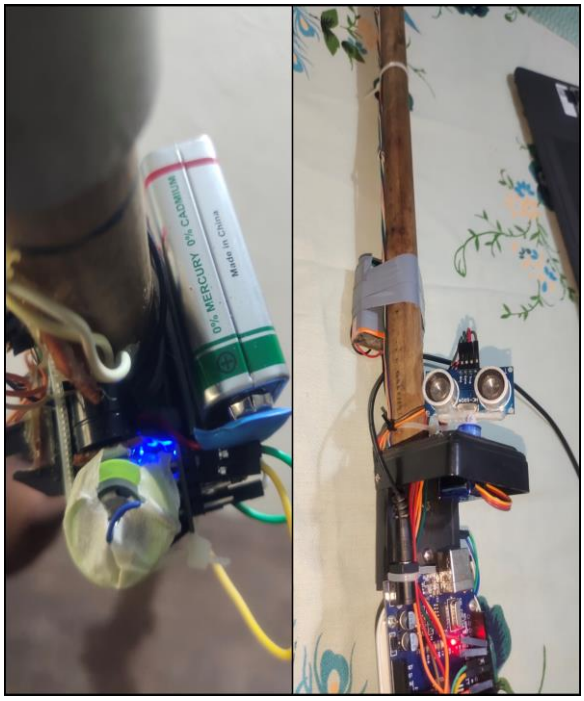
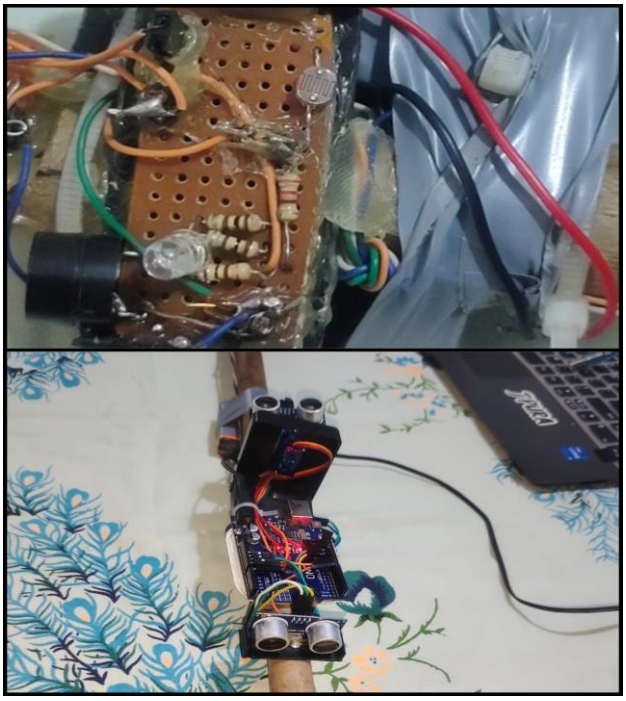


Figure 2: Find Stick System

Prototype







Source Code

❖ **This is the code for the Smart Blind Stick prototype.**

➢ Main function file Code

```
#include <Servo.h>
#include <Calculation_Function.h>
#include <Tone.h>

////////////////////////////////
//Global Variable Declaration
////////////////////////////////
const int Depth_trigPin = 10;
const int Depth_echoPin = 11;
const int Distance_trigPin = 12;
const int Distance_echoPin = 13;

const int BuzzarPin = 8;
const int LDRpin = A0;

int distance, depth, LightIntensity;

Servo servo;
int angle = 0;

////////////////////////////////
//-----------------------------------------------------------------//
//Instances of Classes
DistanceSensor forwarddistance(Distance_trigPin, Distance_echoPin);
DepthSensor depthdistance(Depth_trigPin, Depth_echoPin);
LightSensor lightintensity(LDRpin, LightIntensity);
BuzzerControl buzzercontrol(BuzzarPin);
//-----------------------------------------------------------------//
```

Source Code

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);

  pinMode(Distance_trigPin, OUTPUT);
  pinMode(Distance_echoPin, INPUT);

  pinMode(Depth_trigPin, OUTPUT);
  pinMode(Depth_echoPin, INPUT);

  pinMode(BuzzarPin, OUTPUT);

  servo.attach(7);
  servo.write(angle);

  SystemStatus();
}

////////////////////////////////////////////
//       System Status (ON/OFF) Identify   //
////////////////////////////////////////////

void SystemStatus() {
  tone(BuzzarPin, 500);
  delay(500);
  tone(BuzzarPin, 500);

  for (int i = 0; i < 180; i++) {
    servo.write(i);
  delay(5);
 }
```

Source Code

```
  for (int i = 180; i = 0; i--) {
    servo.write(i);
    delay(5);
  }
  delay(1000);
}
////////////////////////////////////////////
//       System Status (ON/OFF) Identify            //
////////////////////////////////////////////

void loop() {
  // put your main code here, to run repeatedly:
  Servo_Motor();
}

void Looping() {

  LightIntensity = lightintensity.readIntensity();
  Serial.print(LightIntensity);
  Serial.println(" I");
  if (LightIntensity < 5) {
    buzzercontrol.intensityTone(BuzzarPin, LightIntensity);
  }

  distance = forwarddistance.readDistance();
  Serial.print(distance);
  Serial.println(" Cm");
  if (distance < 30) {
    buzzercontrol.forwardTone(BuzzarPin, distance);
  }
```

Source Code

```
depth = depthdistance.readDepth();
  Serial.print(depth);
  Serial.println(" M");
  if (depth > 15) {
    buzzercontrol.depthTone(BuzzarPin, depth);
  }
}

void Servo_Motor() {

  for (angle = 0; angle < 180; angle++) {
    servo.write(angle);
    delay(15);

    Looping();
  }
  for (angle = 180; angle > 0; angle--) {
    servo.write(angle);
    delay(15);

    Looping();
  }
}
```

Source Code

- ESP 8266 ESP01 Wi-Fi Module Code – Blid Stick Finding Purpose

```
//////////////////////////////////////////////
//        RemoteXY include library          //
//////////////////////////////////////////////

// RemoteXY select connection mode and include library
#define REMOTEXY_MODE__ESP8266WIFI_LIB_POINT
#include <ESP8266WiFi.h>

#include <RemoteXY.h>

// RemoteXY connection settings
#define REMOTEXY_WIFI_SSID "RemoteXY"
#define REMOTEXY_WIFI_PASSWORD "12345678"
#define REMOTEXY_SERVER_PORT 6377


// RemoteXY configurate
#pragma pack(push, 1)
uint8_t RemoteXY_CONF[] =   // 27 bytes
  { 255,1,0,0,0,20,0,16,31,1,2,1,11,47,43,26,1,26,31,31,
  79,78,0,79,70,70,0 };

// this structure defines all the variables and events of your control
interface
struct {

    // input variables
  uint8_t Find; // =1 if switch ON and =0 if OFF
```

Source Code

```
    // other variable
  uint8_t connect_flag;  // =1 if wire connected, else =0

} RemoteXY;
#pragma pack(pop)

/////////////////////////////////////////////
//           END RemoteXY include          //
/////////////////////////////////////////////

#define PIN_FIND 2
void setup()
{
  RemoteXY_Init ();

  pinMode (PIN_FIND, OUTPUT);

  // TODO you setup code
}

void loop()
{
  RemoteXY_Handler ();

  digitalWrite(PIN_FIND, (RemoteXY.Find==0)?LOW:HIGH);

  // TODO you loop code
  // use the RemoteXY structure for data transfer
  // do not call delay(), use instead RemoteXY_delay()
}
```

Source Code

❖ Library file Codes

- ➢ Library of Calculation_Function
  - ○ Calculation_Function.h

```
#ifndef CALCULATION_FUNCTION_H
#define CALCULATION_FUNCTION_H

#include <Arduino.h>

class DistanceSensor {
public:
  DistanceSensor(int trigPin, int echoPin);
  int readDistance();
private:
  int trigPin;
  int echoPin;
};

class DepthSensor {
public:
  DepthSensor(int trigPin1, int echoPin1);
  int readDepth();
private:
  int trigPin1;
  int echoPin1;
};

class LightSensor {
public:
  LightSensor(int LDRpin, int LightIntensity);
  int readIntensity();
```

Source Code

```
private:
  int LDRpin;
  int LightIntensity;
};
#endif
```

- Calculation_Function.cpp

```
#include "Calculation_Function.h"
#include <Arduino.h>

#define NOTE_L0 2000
#define NOTE_L1 20

DistanceSensor::DistanceSensor(int trigPin, int echoPin) {
  this->trigPin = trigPin;
  this->echoPin = echoPin;
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}

DepthSensor::DepthSensor(int trigPin1, int echoPin1) {
  this->trigPin1 = trigPin1;
  this->echoPin1 = echoPin1;
  pinMode(trigPin1, OUTPUT);
  pinMode(echoPin1, INPUT);
}

LightSensor::LightSensor(int LDRpin, int LightIntensity) {
  this->LDRpin = LDRpin;
  this->LightIntensity = LightIntensity;
}
```

Source Code

```
int DistanceSensor::readDistance() {
  //Distance calculation code here
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);  //send ultrasonic wave
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  long duration = pulseIn(echoPin, HIGH);
  int distance = (duration * 0.034 / 2);

  return distance;
}

int DepthSensor::readDepth() {
  //Depth calculation code here
  digitalWrite(trigPin1, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin1, HIGH);  //send ultrasonic wave
  delayMicroseconds(10);
  digitalWrite(trigPin1, LOW);

  long duration1 = pulseIn(echoPin1, HIGH);
  int depth = (duration1 * 0.034 / 2);

  return depth;
}

int LightSensor::readIntensity() {
```

Source Code

```
 //Light Intensity calculation code here
  int LightIntensity = analogRead(LDRpin);  // read the value from
the sensor
  return LightIntensity;
}
```

- Tone Library
  - Tone.h

```
#ifndef TONE_H
#define TONE_H

#include <Arduino.h>

class BuzzerControl {
public:
  BuzzerControl(int buzzerPin);
  int forwardTone(int buzzerPin, int distance);
  int depthTone(int buzzerPin, int depth);
  int intensityTone(int buzzerPin, int LightIntensity);
private:
  int buzzerPin;
  int distance;
  int depth;
  int LightIntensity;
};

#endif
```

  - Tone.cpp

```
#include "Tone.h"
#define NOTE_L0 2000
```

Source Code

```
#define NOTE_L1 20
#define NOTE_U0 800
#define NOTE_U1 970

BuzzerControl::BuzzerControl(int buzzerPin) {
  this->buzzerPin = buzzerPin;
  pinMode(buzzerPin, OUTPUT);
}

int BuzzerControl::forwardTone(int buzzarPin, int distance) {
  int melody[] = {
    //tone array
    NOTE_U0, NOTE_U1
  };
  for (int i = 0; i < sizeof(melody); i++) {
    tone(buzzarPin, melody[i], 250);  //play tone
    delay(80 + 5 * distance);         //delay
    noTone(buzzerPin);                //tone off
  }
}

int BuzzerControl::depthTone(int buzzarPin, int depth) {
  tone(buzzarPin, NOTE_L0);
  tone(buzzarPin, NOTE_L1);
  tone(buzzarPin, NOTE_L0);

  if (depth < 30) {
    delay(100);
  } else {
    delay(50);
  }
```

Source Code

```
  noTone(buzzarPin);
}
int BuzzerControl::intensityTone(int buzzarPin, int LightIntensity)
{
  if (LightIntensity < 5) {
    tone(buzzarPin, NOTE_L0);
    tone(buzzarPin, NOTE_L1);
    tone(buzzarPin, NOTE_L0);
    delay(80 + LightIntensity * 10);
    noTone(buzzarPin);
  }
}
```

Future Implementation

- **Voice Assistant:** Integrate a voice assistant feature that can provide real-time spoken information to the user, such as details about their surroundings, weather conditions, nearby landmarks, and more. This feature can enhance user interaction and provide valuable auditory cues.
- **Location Tracking:** Implement GPS or other location tracking technologies to allow caregivers or family members to track the user's location in real-time, ensuring their safety and enabling quick assistance when needed.
- **Find User Functionality:** Develop a feature that allows the user to request assistance when they are disoriented or lost. This feature can send a signal to a designated contact or caretaker, helping locate and guide the user to safety.
- **Rechargeable Battery:** Replace disposable batteries with a rechargeable battery system. This upgrade can make the smart blind stick more cost-effective and eco-friendlier while ensuring longer, more reliable operation between charges.
- **Obstacle Recognition and Avoidance:** Enhance the stick's obstacle detection and avoidance capabilities through advanced sensors and machine learning algorithms. This will further improve the user's safety and mobility.
- **Customizable Feedback:** Allow users to customize the feedback provided by the smart blind stick to suit their preferences. This could include adjusting the volume, pitch, or language of spoken instructions, and enabling vibrations or auditory cues.
- **Mobile App Integration:** Develop a dedicated mobile app that pairs with the smart blind stick, enabling users to configure settings, access additional information, and receive alerts on their smartphone.
- **Connectivity with Smart Devices:** Integrate the smart blind stick with other smart devices in the user's environment, such as home automation systems, to provide seamless interaction and enhance the overall user experience

Conclusion

In conclusion, the Smart Blind Stick project holds significant promise for improving the lives of visually impaired individuals. It addresses the critical challenges faced by this community by providing an innovative solution that enhances mobility, safety, and independence. Through a combination of carefully selected components and advanced technology, the smart blind stick offers a comprehensive set of features that aid users in navigating their surroundings with increased confidence and reduced risk.

The project's scope covers essential functions such as obstacle detection, real-time feedback, and user-friendly operation, ensuring that it caters to the unique needs of the visually impaired. As the project progresses, we anticipate further refining and expanding the system's capabilities to offer a more comprehensive solution.

By providing an overview of the project's components, problems it aims to solve, and its purpose, we have laid a solid foundation for understanding the project's context and objectives. Additionally, the concept of working has been explained, elucidating the core principles and technologies that drive the smart blind stick.

Moving forward, we aim to continue our efforts, addressing any challenges that arise, and working towards the realization of our goals. With ongoing development and testing, we plan to fine-tune the system, ensure its reliability, and ultimately make a positive impact in the lives of visually impaired individuals.

The Smart Blind Stick project represents not only a technical endeavor but also a means of promoting inclusivity, accessibility, and empowerment for a marginalized community. As we advance further, we are committed to refining and optimizing the system, and we look forward to the positive change it can bring to the lives of those it serves.