Smart Cane For Visually Impaired People

Submitted By : Arvin Munkh-Ochir

Supervised By : Tsolmon Yadmaa
Budragchaa Sosorbaram
Orgilbold Dashdorj

2024

Acknowledgments

This project was supported by Tsolmon Yadmaa and Budragchaa Sosorbaram, engineers from ONDOSPACE, who contributed much to the physical development and logistics of the smart cane. Gratitude is extended to the individuals who participated in testing the device. Special thanks are given to Chantsaldulam, a visually impaired individual, for her valuable feedback during the design process.

Abstract

Due to the slow improvements of IoT devices designed for individuals with disabilities and the lack of infrastructure improvements in our city for visually impaired people, a smart cane device has been developed. This device is specifically designed to assist visually impaired users. Based on extensive surveys and analyses, the primary challenges faced by visually impaired individuals were categorized into three distinct key areas: water detection, obstacle detection, and sudden hole detection. The cane is equipped with five innovatively structured infrared sensors through the cane body and a water sensor embedded at the front, all controlled by a microcontroller. It can detect three types of front obstacles: those at curbstone height, obstacles higher than a curbstone, and larger obstacles like walls that are over 100 cm from the tip of the cane. Additionally, downward-facing infrared sensors are used to detect holes in the user's path by measuring the distance between cane and ground. All sensors are housed in a durable plastic case for protection. The device provides real-time feedback via a speaker, conveying information about detected obstacles and conditions. Initial testing has shown the device to be effective. This paper presents the first version of the smart cane, with further development and improvements currently underway.

Chapter 1

Introduction

1.1 Contextual Overview

In 2024, there are approximately 12,600 visually impaired individuals in Mongolia, with nearly 4,000 living in the capital, Ulaanbaatar. The city's infrastructure for visually impaired people is significantly underdeveloped: many tactile pavements have been damaged or demolished, and essential public facilities—such as elevators and bus stops—lack support. As a result, many visually impaired individuals avoid going outside frequently, relying heavily on others for assistance and guidance. This challenge is not only occurring in Mongolia. While there has been progress in developing smart canes globally, many of these solutions remain inaccurate or impractical for daily use. To address these issues, we have identified three primary obstacles that individuals face in their daily lives and improved features of prior studies. This study aims to design an affordable, effective, and user-friendly smart cane that addresses these problems and improves the mobility and independence of visually impaired people.

1.2 Purpose of the Study

- **Detection and Identification**: The smart cane needs to detect front obstacles and differentiate between their heights (e.g., tall or short objects), as well as detect sudden holes and water hazards during walks, especially in summer, autumn, spring.
- **Improvement in Feedback Mechanism**: Instead of using vibration, which can become bothersome for users, this study explores using an audio-based feedback system (via speakers) to alert the user about obstacles.
- **Durability Enhancement**: To ensure long-term use, the study incorporates protective cases for sensors and other components, increasing the overall durability of the device.

Chapter 2

Problem Statement and Proposed Solution

2.1 Obstacle Detection

The device will be equipped with two obstacle-detecting and distance-measuring sensors positioned at different heights. One sensor will be mounted at the average curbstone height, while the other will be placed higher. This setup will help distinguish between curbstones and actual obstacles. The sensors will have a detection range of 70–90 cm, utilizing appropriate sensor technology to effectively perform this task.

2.2 Hole-Stair Detection

For detecting holes and stairs, the device will be equipped with a short-range distance measuring sensor arranged in a specific configuration. These sensors will ensure that holes directly in front of the user's path are detected. This enhances the user's ability to identify potential sudden holes in real time.

2.2 Water Detection

In Ulaanbaatar, during the summer, autumn, and spring, mud and rain are common due to the underdeveloped drainage systems on roads and bridges. To address this issue, the device will embed a water sensor at the tip of the cane to help users detect water hazards in their path.

2.4 Urgent message signaling

Various methods can be used to communicate messages to visually impaired individuals, including Braille, vibrations to specific parts of the body, or buzzer signals. For immediate and urgent notifications, however, the cane is equipped with an embedded audio system that delivers conversational voice messages. These messages are designed to alert the user in their native language, ensuring clear and effective communication in critical situations.

Chapter 3

Electric Components

3.1 Sensor Selection

3.1.1 ***Sharp GP2Y0A02YK0F*** **(USED)**

Working Voltage	DC 4.5V - 5V
Working Current	33 mA
Min Range	20 cm
Max Range	150 cm
Operating Temperature	-10 to 60 °C

Figure 1. Sharp GP2Y0A02YK0F

Table 1. Sharp GP2Y0A02YK0F

Review : This sensor was well-suited for obstacle detection. The accuracy of the values remained stable, and with some averaging and filtering, the readings were consistently clear within the 20–130 cm range.

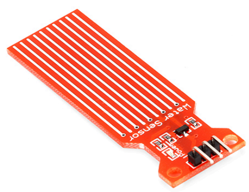
3.1.2 ***E18-D80NK IR Proximity Sensor*** **(USED)**

Working Voltage	DC 5V
Working Current	25 mA
Max Range	80 cm
Diameter	18 mm
Operating Temperature	-25 to 70 °C

Figure 2. E18-D80NK **Table 2.** E18-D80NK

Review : To prevent the device from identifying every object as an obstacle, an obstacle-sensing sensor is implemented. The sensor outputs a low signal when an obstacle is detected and a high signal when there is none. For stationary obstacles, the sensor will continuously output a low signal. This prolonged low signal can indicate the presence of a permanent obstacle. This mechanism allows the device to distinguish between stationary obstacles and temporary objects passing by.

3.1.3 *Water Sensor* (USED)

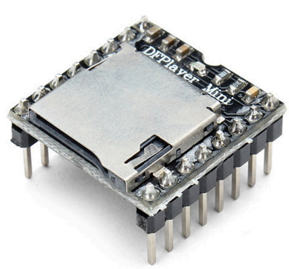


Working Voltage	DC 5V
Working Current	20 mA
Weight	3 gr
Dimension	40 x 16 mm

Figure 3. Water Sensor **Table 3.** Water Sensor

Review : To enable water detection, the cane required a reliable water sensor. While various water related sensors exist, such as water level sensors and moisture sensors, this particular analog sensor was chosen for its features. Its flat and elongated design makes it ideal for integration into the cane's structure. Moreover, the lightweight of this sensor is critical for maintaining the overall weight of the cane. The sensor's water-detecting section is positioned on one side, allowing for easy mounting at the tip of the cane.

3.1.4 *DFPlayer Mini* (USED)



Working Voltage	DC 4.2V
Working Current	15 mA
Supporting File Type	MP3 , WAV
Operating Temperature	-40 to 80 °C
File System	FAT 16 , FAT 32
Protocol	UART

Figure 4. DFPlayer Mini **Table 4.** DFPlayer Mini

Review : To play audio files, cane requires a component capable of reading from a microSD card. The DFPlayer module was suitable for this purpose, as its capacity and file organization system aligned with the needs of the cane. Additionally, it initialized the SD card quickly, ensuring efficient operation.

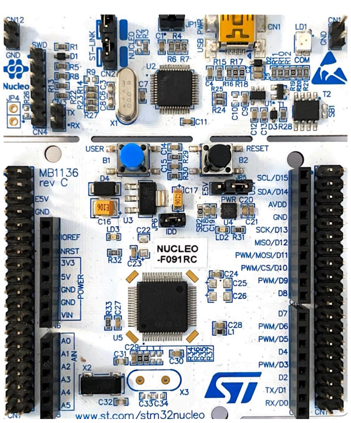
3.1.5 ***E18-D200NK IR Proximity Sensor*** **(USED)**

Working Voltage	DC 5V
Working Current	25 mA
Min Range	20 cm
Max Range	200 cm
Diameter	18 mm
Operating Temperature	-25 to 70 °C

Figure 5. E18-D200NK **Table 5.** E18-D200NK

Review : A long-range proximity sensor has been embedded to detect distant objects, sensing obstacles up to 200 cm away. The sensor outputs a low signal when an obstacle is detected and a high signal when there is none. For stationary obstacles, the sensor will continuously output a low signal. This prolonged low signal can indicate the presence of a permanent obstacle. This mechanism allows the device to distinguish between stationary obstacles and temporary objects passing by.

3.1.6 ***Nucleo64 STM32F091RC*** **(USED)**



Processor	ARM 32-bit Cortex
Memory	256Kbytes Flash memory
Clock Speed	48MHz
Peripheral Set	I2C, Uart, SPI
Analog Feature	12 bit

Operating Temperature	-40 to 85 °C

Figure 6. NUCLEO64 STM32F091RC **Table 6.** NUCLEO64 STM32F091RC

Review: The choice of the STM32F091RC board for the cane's design is well-justified due to its application of the On-Board Computer (OBC) typically found in satellite systems. Also, the board's capabilities make it ideal for developing a minimum viable product (MVP), enabling the prototyping process while maintaining the performance standards. Moreover, the board’s various set of peripheral interfaces, including ADC, UART, and I2C, along with its optimized low-power modes, makes it an excellent fit for this portable device. The STM32F091RC’s wide operating temperature range, from -40°C to +85°C, is a critical feature for ensuring reliable performance in Mongolia’s harsh climate. This selection ensures reliable performance and efficient power management, both crucial for the cane's functionality and usability.

3.1.7 *HCSR04* (NOT USED)

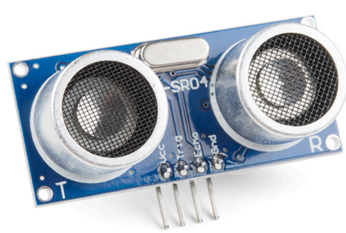


Figure 7. HCSR04

Working Voltage	DC 5V
Working Current	15 mA
Min Range	2 cm
Max Range	400 cm
Dimension	45x20x15 mm

Table 7. HCSR04

Review : The required measurement range for our device is 70–90 cm for obstacle detection and 9–15 cm for accurate hole detection. However, the sensor was not suitable for either task, as it did not meet these specifications. The accuracy of the values fluctuated significantly, especially around the 50–60 cm range. As a result, the device did not incorporate this sensor.

3.2 Sensor Placement Design

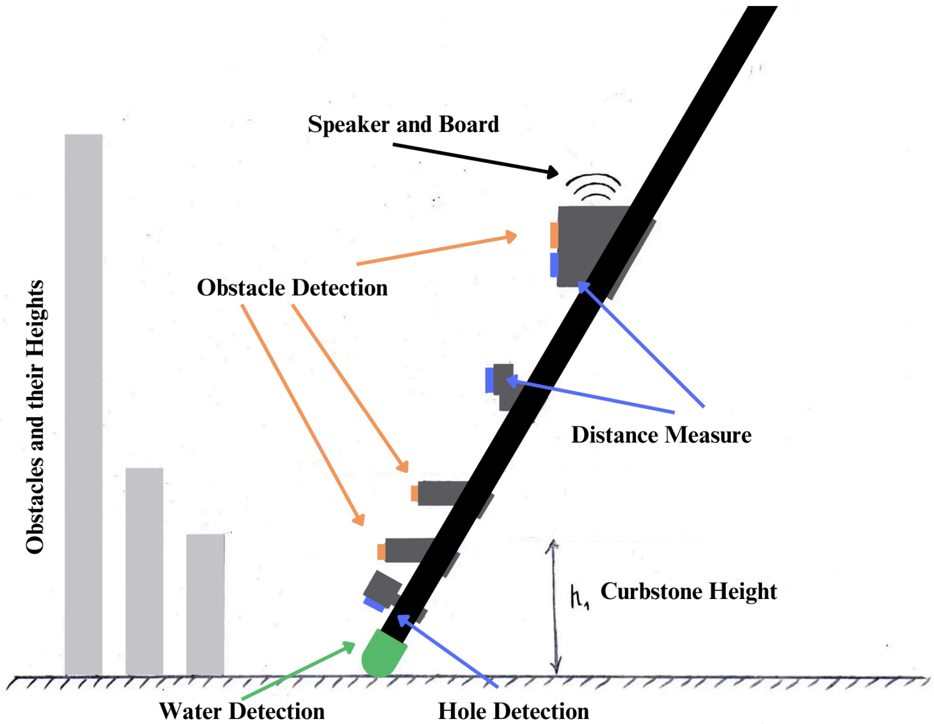


Figure 8. Sensor Logic

1. Water Detection

- **Sensor**: Water Sensor
- **Placement**: Positioned at the tip of the cane and protected within a protective 3D-printed casing.
- **Function**: The water sensor operates by detecting changes in the electric field through probe lines on its surface, allowing it to sense the presence of water. When water is detected on the ground, the sensor alerts the user to potentially hazardous wet surfaces. This functionality is essential for safety applications where early detection of water can prevent slips of users.

2. Obstacle Detection

- **Sensors**: E18-D80NK (2 units), E18-D200NK (1 unit)
- **Placement and Range**:

    - Two **E18-D80NK** sensors are mounted at lower and medium heights, detecting obstacles within **70 cm** and **80 cm** distances, respectively.
    - One **E18-D200NK** sensor is positioned at the top, with a **200 cm** detection range.
- **Function**: Three infrared sensors are positioned to detect obstacles at different height levels, providing information on obstacle height. If an obstacle remains at a consistent detected height over time, the sensors classify it as an obstacle and signal to the user whether it is low, medium, or high. This distinction enhances environmental awareness and improves safety.

3. Signaling Device

- **Components**: DFPlayer Mini, Speaker, and Microcontroller Boards
- **Placement**: Mounted at the top of the cane.
- **Function:** The DFPlayer module is controlled by output commands from the sensors, which specify the audio file to play. The speaker, powered by the DFPlayer Mini, emits audio signals placed near the user for clear, immediate feedback. With a microSD card embedded in the DFPlayer, this setup enables real-time, understandable feedback on detected hazards or terrain changes.

4. Hole and Drop Detection

- **Sensor**: Sharp GP2Y0A41SK0F
- **Placement**: Installed at the tip of the cane and facing to the ground.
- **Function:** The Sharp GP2Y0A41SK0F sensor continuously measures the distance to the ground by emitting and receiving infrared signals. Any sudden change in the measured distance indicates a potential drop-off, or steep hole, or stairway, alerting the user to changes in ground level. An increased distance suggests a hole, while a decreased distance indicates an upward step or stair.

5. Distance Measure

- **Sensor**: Sharp GP2Y0A02YK0F (2 units)
- **Placement**: Installed at the mid of the cane and facing forward.
- **Function:** The Sharp GP2Y0A02YK0F sensor will start measuring the detected obstacle distance and processes how far is the obstacle from the tip of the cane.

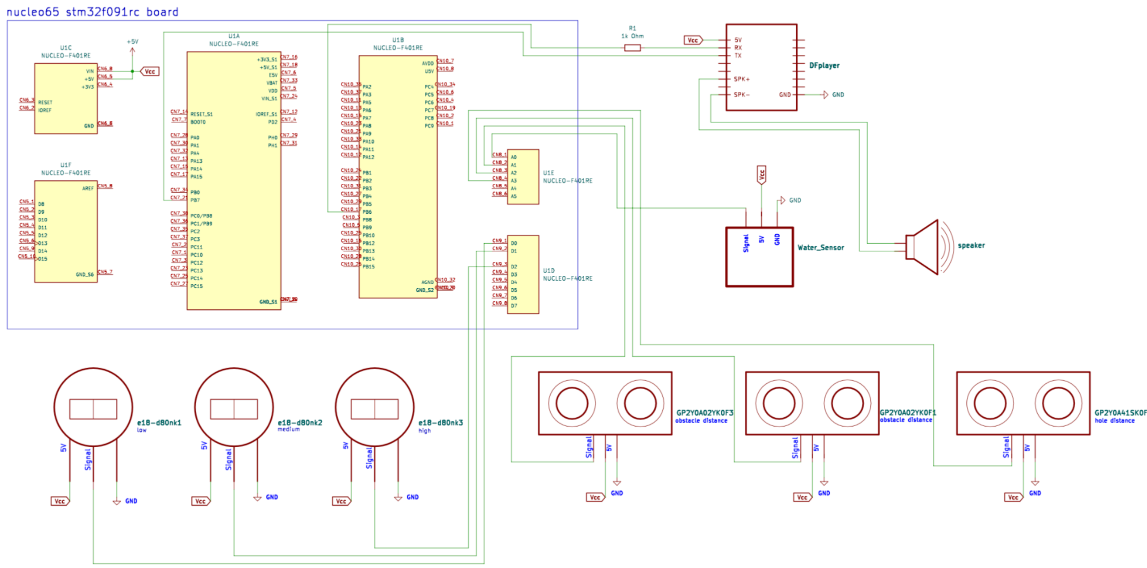


Figure 9. Schematic diagram of the whole system

Section	Sensor	Pin	Type
Obstacle Detection Upper	Sharp GP2Y0A02YK0F	**PA1**	Analog
	Sharp GP2Y0A02YK0F	**PB0**	Digital
Proximity Low	E18-D80NK	**PA8**	Digital
Proximity Medium	E18-D80NK	**PB10**	Digital
Proximity Upper	E18-D80NK	**PB4**	Digital
Water Detection	Water sensor	**PA0**	Analog
	Voltage Pin	**PB8**	Digital
Speaker	DFPlayer Mini Rx	**PB6**	Tx Pin of Board
	DFPlayer Mini Tx	**PB7**	Rx Pin of Board
Hole Detection	Sharp GP2Y0A41SK0F	**PA4**	Analog

Table 9. Microcontroller data pins and sensors

3.3 Hardware Design

3.3.1 3D-Printed Case

A **B** **C**

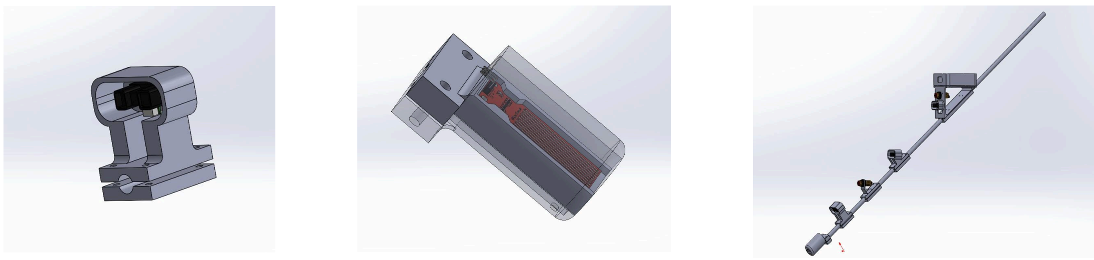


Figure 10. 3D designs of protective cases for sensors: **(A)** Case for hole-detecting distance sensor, **(B)** Case for water-detecting sensor, **(C)** Overall cane assembly

3.3.2 Real Life Cane

A B C D E

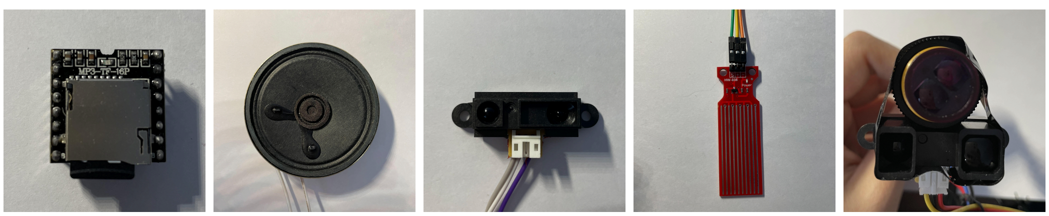


Figure 11. Items cane has been used: **(A)** DFPlayer, **(B)** Speaker **(C)** Distance measuring sensor **(D)** Water detecting sensor **(E)** Obstacle detecting sensor and Distance measuring sensor

Figure 12. (A) First and **(B)** Final Prototype

Chapter 4

System Architecture and Coding

4.1 System Architecture

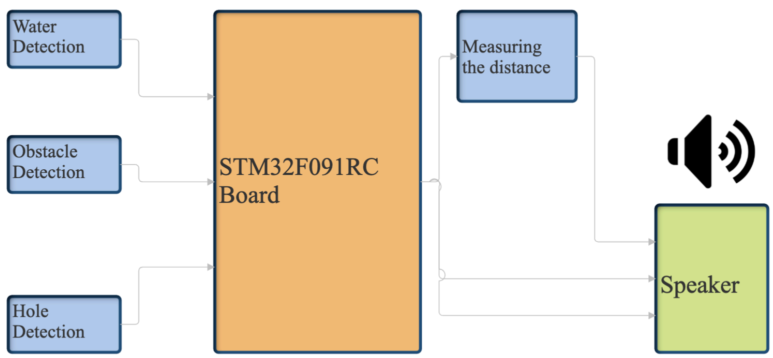


Figure 13. Block diagram of full system.

Figure 13 presents the overall structure of the system. The water, obstacle, and hole-detecting sensors operate asynchronously and continuously, triggering subsequent functions based on their readings. The speaker announces every detected signal, whether from water, obstacles, holes, or stairs. **Figures 14–17** illustrate the cane system's algorithm through flowcharts. Tasks shown in **Figures 15, 16, and 17** run continuously, calling other functions as needed. For instance, the *Obstacle_Detection* task identifies obstacles at various heights (low, medium, or high) and invokes the *Count_Time* function to determine if an obstacle is moving or stationary by calculating the duration of the detection signal. If an obstacle remains at a constant height for over 4 seconds, it is classified as a permanent obstacle, and the *Distance_Find* function is triggered. Additionally, the speaker task, which is named as *Signal_Wait* task, awaits signaling commands and, once received, directs the *Speaker* function to initiate audio feedback through the speaker.

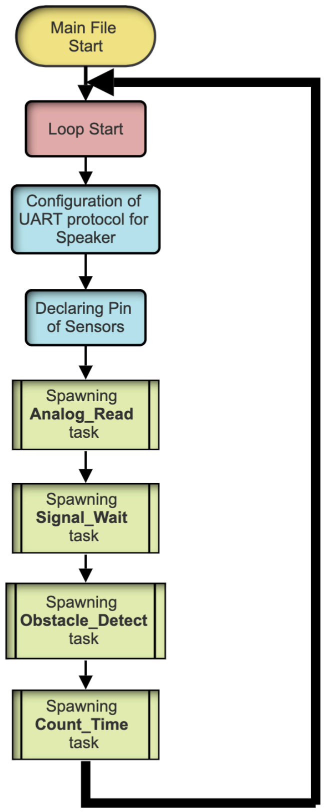


Figure 14. Flowchart of Main file

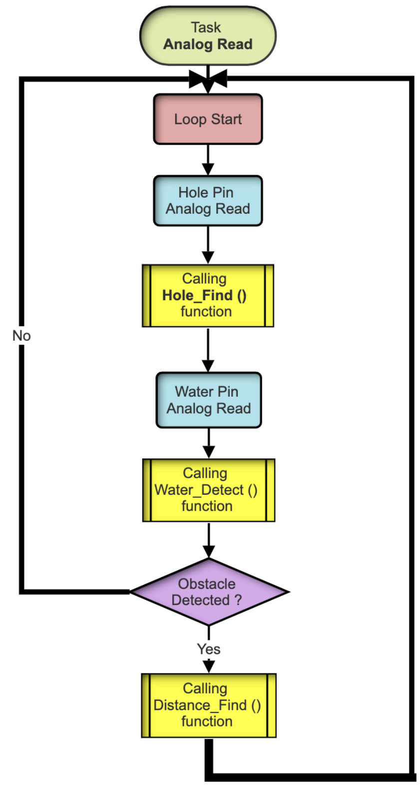


Figure 15. Analog Read Task

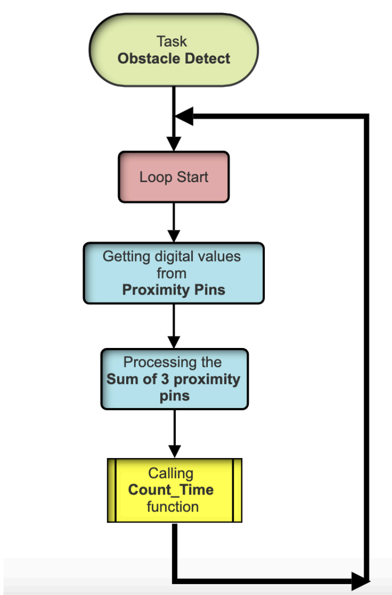


Figure 16. Obstacle Detecting Task

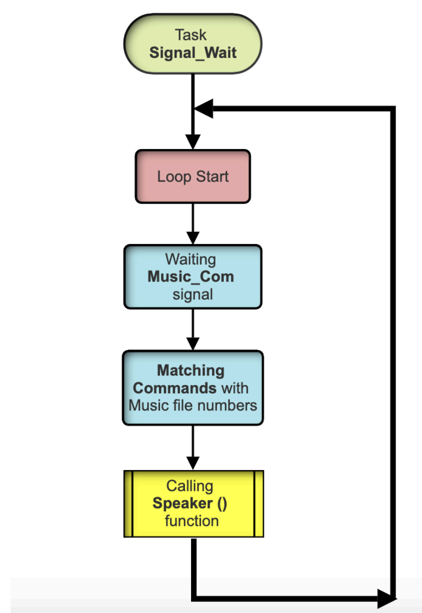


Figure 17. Signal Wait Task

4.2 Coding

```
config.baudrate=9600;
config.parity=usart::Parity::ParityNone;
let mut uart: Uart<'_, embassy_stm32::mode::Async> = Uart::new(p.USART1, p.PB7, p.PB6, Irqs2, p.DMA1_CH2, p.DMA1_CH1, config).unwrap();
let (mut tx: UartTx<'_, Async>, mut rx: UartRx<'_, Async>) = uart.split();

//Speaker
let mut ack_buffer: [u8; _] = [0u8; 10];
let mut ack_buffer2: [u8; _] = [0u8; 10];

//Initializing inserted SD card to make the DFPlayer detect it
let command_initialize: [u8; 10] = [ 0x7E, 0xFF, 0x06, 0x3F, 0x01, 0x00, 0x02, 0xFE, 0xB9, 0xEF ];
                    match tx.write(&command_initialize).await {
                        Ok(()) => { info!("Initialization has written");      }
                        Err(e: Error) => {  info!("Error reading ACK: {:?}", e);      }
                    }

                    match rx.read(&mut ack_buffer).await {
                        Ok(()) => {info!("ACK Response: {:X}", &ack_buffer);  }
                        Err(e: Error) => {  info!("Error reading ACK: {:?}", e);      }
                    }
                    Timer::after_millis(4000).await;

//Putting the volume to its highest note
let command_vol: [u8; 10] = [ 0x7E, 0xFF, 0x06, 0x06, 0x01, 0x00, 0x1E, 0xFE, 0xD6, 0xEF ];
                    match tx.blocking_write(&command_vol){
                        Ok(()) => { info!("Volume has written ");               }
                        Err(e: Error) => { info!("ZZZZ");                           }
                    }

                    match rx.read(&mut ack_buffer2).await {
                        Ok(()) => { info!("ACK Response: {:X}", &ack_buffer2); }
                        Err(e: Error) => { info!("Error reading ACK: {:?}", e);       }
                    }
                    Timer::after_millis(100).await;
```

Figure 18. Initializing DFPlayer on main file

```
// Upper Medium Lower Proximity sensors
let mut prox_lower=Input::new( p.PA8, Pull::Up);
let mut prox_medium=Input::new( p.PB10, Pull::Up);
let mut prox_upper=Input::new( p.PB4, Pull::Up);

// Hole Detecting Pin
let mut hole = p.PA4;

// Obstacle Distance Measuring pin
let mut obstacle_measure = p.PA1;

//Water Sensing pin
let mut water = p.PA0;

//ADC read part
let mut adc: Adc<'static,ADC1> =Adc::new(p.ADC1,Irqs1);

spawner.spawn(analog_read(hole,water,obstacle_measure,adc)).unwrap();
spawner.spawn(signal_wait(tx, rx)).unwrap();
spawner.spawn(proximity_read(prox_lower, prox_medium, prox_upper)).unwrap();
spawner.spawn(count_time()).unwrap();
```

Figure 19. Declaring Pins and Calling Task on main file

```rust
#[embassy_executor::task]
pub async fn signal_wait (mut TX: UartTx<'static, embassy_stm32::mode::Async> ,mut RX: UartRx<'static, embassy_stm32::mode::Async>) {
    loop{
        let val: u32  = SHARED.wait().await;
        if val > 0 && val < 22 {
            let mut command_play: [u8; 10] = match val {

                1  => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x01, 0xFE, 0xF6, 0xEF ], // 1 - Height : Low
                2  => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x03, 0xFE, 0xF4, 0xEF ], // 3
                3  => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x05, 0xFE, 0xF2, 0xEF ], // 5
                4  => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x07, 0xFE, 0xF0, 0xEF ], // 7
                5  => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x09, 0xFE, 0xEE, 0xEF ], // 9
                6  => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x0B, 0xFE, 0xEC, 0xEF ], // 11
                7  => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x0D, 0xFE, 0xEA, 0xEF ], // 13 - Height : Medium
                8  => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x0F, 0xFE, 0xE8, 0xEF ], // 15
                9  => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x11, 0xFE, 0xE6, 0xEF ], // 17
                10 => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x13, 0xFE, 0xE4, 0xEF ], // 19
                11 => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x15, 0xFE, 0xE2, 0xEF ], // 21
                12 => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x17, 0xFE, 0xE0, 0xEF ], // 23
                13 => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x19, 0xFE, 0xDE, 0xEF ], // 25 - Height : High
                14 => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x1B, 0xFE, 0xDC, 0xEF ], // 27
                15 => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x1D, 0xFE, 0xDA, 0xEF ], // 29
                16 => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x1F, 0xFE, 0xD8, 0xEF ], // 31
                17 => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x21, 0xFE, 0xD6, 0xEF ], // 33
                18 => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x23, 0xFE, 0xD4, 0xEF ], // 35
                19 => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x25, 0xFE, 0xD2, 0xEF ], // 37 - Water Detected
                20 => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x27, 0xFE, 0xD0, 0xEF ], // 39 - Ground Level Uprising
                21 => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x29, 0xFE, 0xCE, 0xEF ], // 41 - Ground Level Downrising
                22 => [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x2B, 0xFE, 0xCC, 0xEF ], // 43 - Nam saad bro

                _=> [ 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x03, 0xFE, 0xEC, 0xEF ],
            };
            speaker(&mut TX, &mut RX, &mut command_play).await;
            Timer::after_millis(1000).await;
        }
    }

} fn signal_wait
```

Figure 20. Calling commands for the audios to play

A

B

```toml
[package]
name = "smartcane"
version = "0.1.0"
edition = "2024"

[dependencies]
# Change stm32f091rc to your chip name, if necessary.
embassy-stm32 = { version = "0.1.0", path = "/Users/arvinamunku/D
cortex-m = { version = "0.7.7", features = ["inline-asm", "critic
cortex-m-rt = "0.7.0"
defmt = "0.3"
embedded-hal-1 = { package = "embedded-hal", version = "1.0" }
embassy-embedded-hal = { version = "0.2.0", path = "/Users/arvina
defmt-rtt = "0.4"
panic-probe = { version = "0.3", features = ["print-defmt"] }
embassy-sync = { version = "0.6.0", path = "../../../embassy-main
embassy-executor = { version = "0.6.0", path = "/Users/arvinamunk
embassy-time = { version = "0.3.1", path = "/Users/arvinamunku/Do
atomic_float = "1.1.0"

[profile.release]
debug = 2
```

Figure 21. (A) Project Crate and its Modules (B) Used Embedded Rust crates

The system’s code is written entirely in Embedded Rust, chosen for its secure memory management and compatibility with real-time operating system features. To optimize performance for embedded applications, various Embedded Rust crates were used in this project. *Figure 21.A* provides an overview of the system's functions and main files, highlighting the modular organization of functions within separate files (mod files).

The system is structured into three main modules: **dfplayer, analog_read_pins,** and **proximity,** each declared inside the mod file. The **analog_read_pins** module reads data from all sensors via an ADC (Analog to Digital Converter). Since the system operates on a single ADC channel, it reads sensor data sequentially by switching pin channels every 300ms, allowing it to capture three pieces of sensor data per 1 second. The **proximity** module processes the digital outputs of proximity sensors to detect obstacles and subsequently signals the **analog_read_pins** module to activate the distance-measuring function. Once these two modules have processed data, audio commands to alert the user are routed through the **dfplayer** module, which interfaces with a speaker. All modules rely on a shared signaling variable to manage. Commands sent to the **dfplayer** module are then played through the speaker, providing feedback to the user. The UART protocol is used exclusively within the DFPlayer module, as shown in *Figure 20*, with the parity set to "None" and the baud rate configured to 9600. Other modules do not use a transmission protocol.

The main program file declares tasks across the three modules as needed, ensuring a structured and efficient workflow throughout the system as shown in *Figure 18 and Figure 19.*

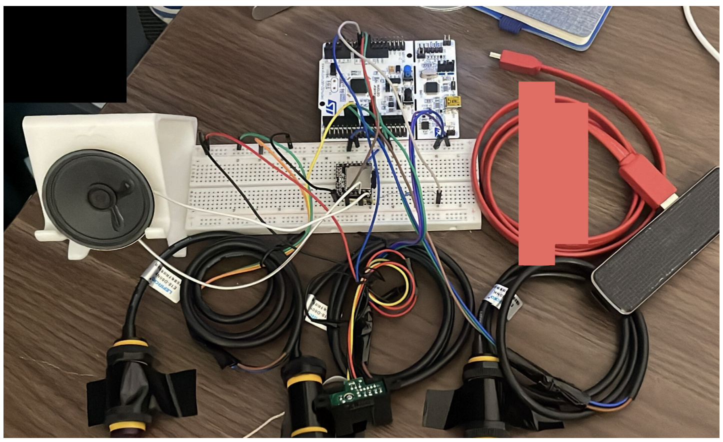
Figure 21.B illustrates the embedded Rust crates used in this smart cane. These crates were important to the functionality of the system, enabling efficient coding and hardware integration in the Rust language.The **embassy-stm32** crate was used to monitor and control the Nucleo-64 STM32 board, providing methods and support for asynchronous tasks of the microcontroller’s architecture and enabling the interaction with the board’s peripherals. Additionally, the **atomic-float** crate was used to handle atomic operations on floating point numbers, an important feature for ensuring safe calculations during real time sensor data processing. Together, these and other downloaded crates optimized the development process, making the use of Rust’s safety, concurrency, and performance features for the smart cane system.

Chapter 5

Testing and Conclusion

5.1 Testing

Figure 22. Combined Test of Sensors



All required sensors were connected to the board using a breadboard to manage connections to each pin. Code was compiled and successfully tested. The speaker, three proximity sensors, a hole-detection distance sensor, and obstacle-measurement distance sensors all performed as expected according to the planned operation. However, the DFPlayer module initially overheated, so it was replaced with a new one. After replacement, all components functioned as intended.

A **B** **C**

Figure 23. Protective cases and sensors **(A)** Water Sensor, **(B)** Frontal Picture, **(C)** Side Picture

5.2 Validating as in real use

Figure 24. Testing it in real life uses : Different liquids, Varied Height obstacle, Upstairs

5.1.1 Water Detection

The cane’s water detection capability was tested using muddied water, pure water, and various liquids with sediments. In each of 20 trials, the water sensor inserted in the cane’s tip accurately detected the presence of water. This result confirms the cane’s reliability in detecting water under different conditions.

5.1.2 Hole Detection

The cane’s ability to detect holes was evaluated on curbstones, stairs, and inclined surfaces. It successfully identified sharp downward edges, such as curbstones and stairs. However, the system showed limitations in detecting gradually inclined holes. Moreover, its performance was less reliable when encountering surfaces made of reflective materials, such as glass walls and floors.

5.1.3 Obstacle Detection

The cane's obstacle detection system was tested for various obstacle heights and distances under different lighting conditions. It demonstrated accurate performance indoors and during the evening. However, the system experienced challenges in bright daylight conditions with high sunlight intensity, resulting in less accurate measurements.

5.3 Conclusion and Future Work

This project presented an Internet of Things (IoT) based smart cane system designed to assist visually impaired individuals by detecting obstacles, water, and holes. The system demonstrated several advantages, including the effective use of sensors at varying heights, which improved accuracy in detecting different objects. The protective plastic casings ensured durability, making the cane tolerable for various use conditions. Additionally, the use of a speaker for audio warnings provided a simple and efficient means of alerting users in real time.

However, several limitations that needed for future improvements were identified. The inclusion of multiple sensors positioned at various heights, along with protective casings, significantly increased the cane's weight. This added weight could make long term use unbearable for individuals. Furthermore, the distribution of sensors, particularly those located near the tip, made the cane unbalanced, adding strain on users when holding it. A redesign focusing on sensor placement towards the middle or upper sections of the cane could improve ease of use.

Another challenge was the limited range of the sensors, which could only detect objects up to a maximum distance of approximately 100 cm. This short range required users to stop and change their direction within a constrained space, which could interrupt their journey and reduce overall efficiency. Enhancing sensor range to detect obstacles at greater distances would enable users to navigate more smoothly and proactively.

Finally, while the smart cane system successfully addressed several key needs of visually impaired individuals, it also highlighted areas for improvements. Future versions of this cane will focus on reducing weight, improving sensor placement, and extending detection range to overcome the identified limitations and provide a more user-friendly experience.

References

[1] Flyron Technology. (2024). *FN-M16P Embedded MP3 Audio Module datasheet* (Revised Sept. 2024).

[2] Sharp. (2011). *GP2Y0A41SK0F* (OP13008EN). [Revised Oct. 2024].

[3] e-Gizmo Mechatronix Central. (2016). *E18-D80nk Long Range Adjustable IR Sensor datasheet* (Revised Sept. 2024).

[4] DIY Robot Keyes. (2024). *Water Sensor Module User's Manual* (Revised Aug. 2024).

[5] Shin, H., Chang, S., Yu, N., Jeong, C., Xi, W., & Bae, J. (2022). *Development of a backpack-based wearable proximity detection system* (Master's thesis). Hanyang University.

[6] Ghani, F. B., & Zariman, A. B. (2022). *Smart cane based on IoT* (Master's thesis). Universiti Pendidikan Sultan Idris, Malaysia.

[7] Embassy.dev. (n.d.). *STM32F091RC | Embassy Documentation*. Retrieved June 2024, from https://docs.embassy.dev/embassy-stm32/git/stm32f091rc/index.html

[8] STMicroelectronics. (n.d.). *STM32F091RC*. Retrieved September 2024, from https://www.st.com/en/microcontrollers-microprocessors/stm32f091rc.html