Location Sensing Smart Backpack

Submitted By : Arvin Munkh-Ochir

Supervised By : Tsolmon Yadmaa
Budragchaa Sosorbaram

2023/2024

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 also 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

As urban landscapes change with the construction of new buildings and roads, visually impaired individuals face growing challenges in adapting to these changes. Sudden developments, such as unfamiliar structures and altered pathways, often restrict their mobility and independence particularly when traveling to new locations without guidance of others. Beyond adapting to these difficulties, they also encounter barriers to fulfilling basic needs, such as knowing essential information like time.

This paper presents the development of a smart backpack device designed to address these challenges and enhance the daily lives of visually impaired individuals. Featuring capabilities such as location sensing, time telling, and location sharing. This smart backpack provides a portable, user-friendly solution that improves safety, and boosts confidence, independency of users.

The design prioritizes not only resolving current challenges but also ensuring long term use. This study goes into the development process of the smart backpack and evaluates its feasibility in becoming a usable tool. By seeing this device from these different angles, this project aims to empower visually impaired individuals, fostering their growth, independence, and their inclusivity in an ever-changing world.

Chapter 1 : Introduction

1.1 Contextual Overview

Over the past two years, Ulaanbaatar has experienced significant growth in urban development. In early 2023, there were approximately 66,000 registered entities in the city, a number that grew to nearly 73,000 by mid 2024. This annual increase of 5,000 – 7,000 entities creates the rapid addition of new buildings, offices, and locations, transforming the landscape and expanding the city's infrastructure. This increase in new buildings and locations in Ulaanbaatar has exacerbated challenges for visually impaired individuals. Mongolia is home to approximately 12,600 visually impaired people, with nearly 4,000 living in the capital. For these individuals, traveling through an evolving urban landscape has become increasingly difficult, often leading to confusion and without the help of someone. This dependency limits their ability to travel independently and constrains their mobility. Moreover, accessing basic information, such as knowing the time, remains a challenge for old visually impaired people. To address these issues, this paper identifies key services designed to redress these problems. The proposed smart backpack integrates features such as location signaling, time-telling, and location sharing, making the backpack an impactful tool to empower the independence and confidence of visually impaired individuals.

1.2 Purpose of the Study

- **Accurate and Fast Location Alerts**: The device processes latitude and longitude values to provide precise location alerts. Locations are pre-calculated and coded to ensure accuracy for the user.
- **Simple and Portable Design**: The device is not permanently attached to the backpack. Its portable design allows it to be used with different backpacks or standalone gadgets, enhancing flexibility. Additionally, battery life is optimized to ensure extended operation.
- **Convenient Audio Signaling**: Location information is conveyed through audio alerts, with pre-recorded messages stored on an SD card for fast and effective playback.
- **Durable Protective Casing**: All components, including buttons, speakers, and boards, are surrounded in a protective shell to enhance durability and ensure long-term usability.

Chapter 2 : Addressed Issues and Proposed Solutions

2.1 Location Sensing

The device uses embedded GPS to track precise latitude and longitude coordinates, allowing users to identify their exact location, including which side of a building they are on. Custom algorithms process location data to provide detailed and accurate location names. This feature is also versatile, potentially assisting children or other groups in need of reliable location sensing.

In the first version of this backpack, the implementation focused primarily on common routes frequently used by visually impaired individuals. As further developments are made, the entire capital will be mapped and integrated into the coding.

2.2 Time Telling

For most people, checking the time is an instant task. However, for visually impaired individuals, this often involves a number of steps. Younger users may rely on smartphone apps with accessibility features, while older adults often depend on Braille watches to determine the time. To simplify this process, the smart backpack has embedded a button. With a single press, the device audibly tells the current time through a speaker, making it quick and convenient.

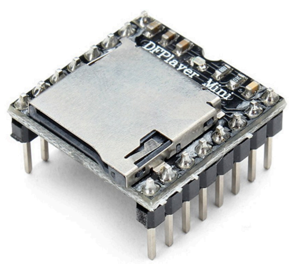
2.3 Location Sharing

The device is equipped with a SIM card to facilitate real time location sharing. When users are lost, their coordinates are automatically sent to a predefined contact in a Google Maps compatible format. This eliminates the need for complex video calls or additional steps to communicate the user’s location.

Chapter 3 : Electric Components

3.1 Sensor Selection

3.1.1 *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 1. DFPlayer Mini **Table 1.** 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.

3.1.2 *Button* (USED)

Type	Momentary
Radius Size	11mm

Figure 2. Button **Table 2.** Button

Review: The large radius and soft pushing force of the button made it highly suitable for our device. Its size suits for a wide range of finger sizes, ensuring accessibility for all users.

Additionally, the round and smooth shape was particularly effective for tactile feedback, making it easier for visually impaired individuals to locate and tap on it.

3.1.3 ***Micro Loudspeaker*** **(USED)**

Speaker Watt	8 Ohm, 1 Watt
Dimension	6cm x 5cm x 2 cm

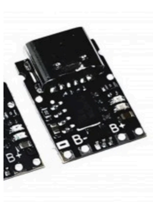
Figure 3. Micro Loudspeaker

Table 3. Micro Loudspeaker

Review : The size of this speaker made it suitable for integration into a small device that is needed to get mounted on a backpack. Its dimensions, comparable to the button, were small enough to keep the device portable while still producing sufficiently loud sound to alert users effectively. Moreover, its lightweight design made it an excellent choice for wearable applications.

3.1.4 ***Charging Module*** **(USED)**

A



B

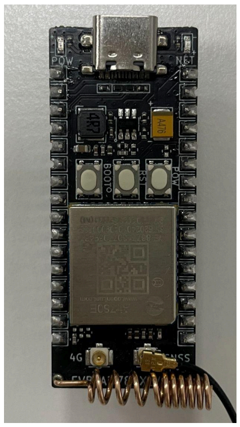
Type	Charging Discharging module
Battery	18650 lithium battery power boost
Port	Type C

Figure 4. (A) Charging Module, (B) Booster

Table 4. Charging Module Features

Review : The device needed to be portable, which required the inclusion of a charging module. Although the module's datasheet indicated it could provide 5V, its actual maximum output was 4.1V. To fix this problem and ensure the device operated at the required 5V, a DC-DC step-up converter was used to boost the voltage to the desired level.

3.1.5 *Air780EG development board* (USED)



Supply Voltage	5V
Available Lua Memory	256k
Script Space	128k
Peripherals	UART

Figure 5. Air780EG board

Table 5. Air780EG board

Review : The location-sensing backpack required GPS functionality, cellular connectivity, and a small, lightweight microcontroller board to ensure portability. The Air780EG board proved to be an ideal choice as it integrates built-in cellular connectivity and GPS functionality, eliminating the need for an external GPS module which can make the device case more bigger. Additionally, the board's low-power consumption made it particularly suitable for a battery-powered, portable device, ensuring extended usability without frequent recharging.

Section	Definition	Pin	Type
Button	Location Detect	**GPIO11**	Digital
	Message Send	**GPIO09**	Digital
	Time Signal	**GPIO01**	Digital
Speaker	DFPlayer Mini Rx	**UART1_TX**	Tx Pin of Board
	DFPlayer Mini Tx	**UART2_RX**	Rx Pin of Board

Table 6. Components and Connections on Board

3.2 Component Placement Logics

Figure 6. Sensor Logic

1. Signaling Device on Shoulder Strap

- **Components:** 3 Buttons, 1 Micro Loudspeaker
- **Placement:** Positioned on the upper part of the shoulder strap
- **Functionality and Design Considerations:**

  The signaling device is strategically placed on the shoulder strap for accessibility and user convenience. The three buttons are located close together, making them easy to tap with a hand, while the micro loudspeaker is positioned near the user's ear for clear audio output. Due to the device's sensitivity and its frequent interaction with the user, placing it on the front side of the body ensures optimal functionality and responsiveness.

  The compact case, measuring 40mm x 20mm x 70mm, is small enough to fit seamlessly on the strap. It houses and protects the buttons and speaker, while a robust clamping mechanism on the back ensures secure attachment to the strap.

2. Location-Sensing Device in Other Parts of the Backpack

- **Components:** Air780EG Board, Charging Module, Lithium-Ion Battery, DFPlayer
- **Placement:** Positioned in a stable, non-moving section of the backpack

- **Functionality and Design Considerations:**
  The location-sensing module is a core component of the device, containing the Air780EG board for GPS functionality, a charging module, a lithium-ion battery, and a DFPlayer to interface with the signaling device on the strap. To ensure stability and good performance, this module is placed in a stable section of the backpack, such as an interior pocket or compartment. The design allows for easy integration into various sections of the backpack without compromising mobility, ensuring it remains securely in place during use.

3.3 Hardware Design

3.3.1 3D-Printed Case

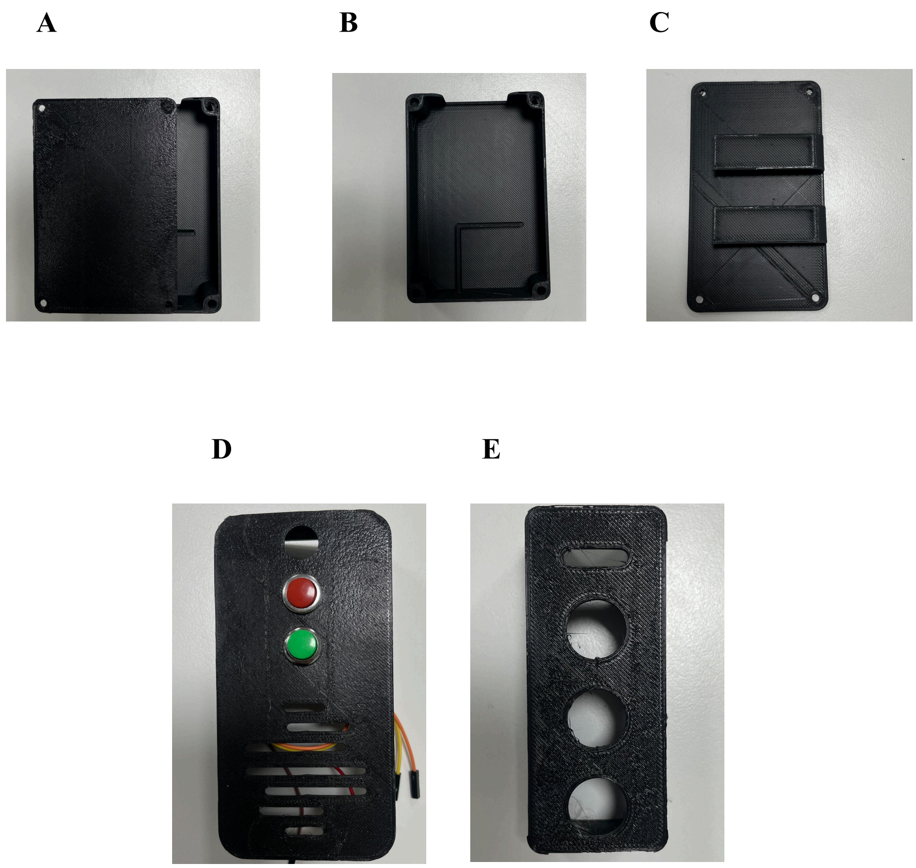


Figure 7. Case: **(A)** Location sensing device case outside, **(B)**Location sensing device case inside, **(C)** Signaling device case back, **(D)** First version of signaling device case front, **(E)** Second version of signaling device case front

3.3.2 Device Components

A **B** **C** **D** **E**

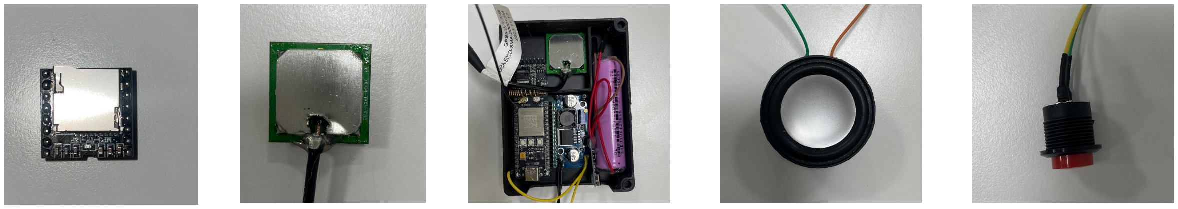


Figure 8. Items that cane has used: **(A)** DFPlayer, **(B)** GPS without case, **(C)** Location Sensing device inside, **(D)** Speaker, **(E)** Button.

A **B**

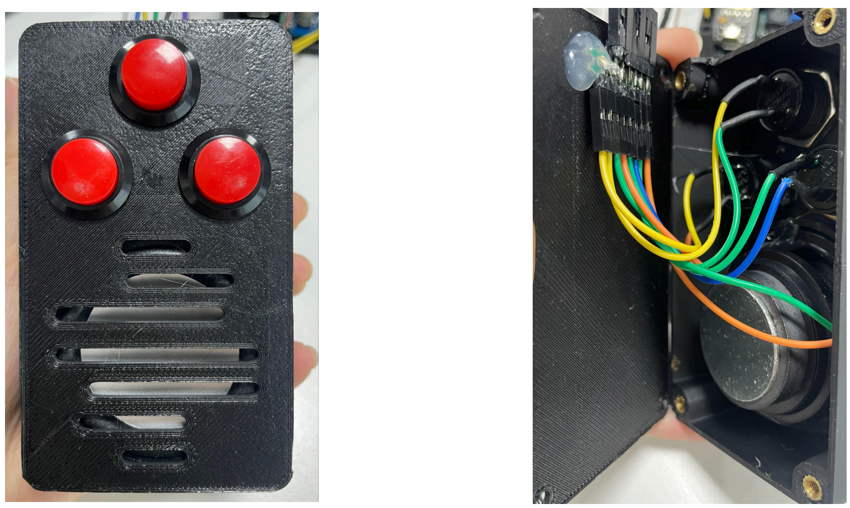


Figure 9. Items that cane has used: **(A)** Signaling Device , **(B)** Inside the signaling device.

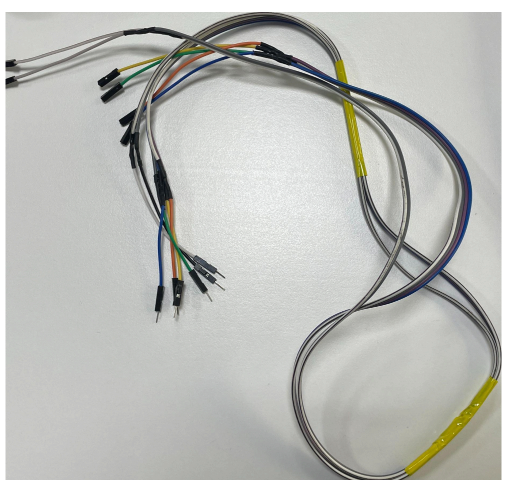


Figure 10. 65 cm long wire to connect signaling device to location sensing device on backpack

3.4 Power Consumption and Financial Cost

Component	Current consumption (Active mode)	Current consumption (Low power)	Quantity
Air780EG	4.7mA	0.47mA	1
DFPlayer Mini	15mA	15mA	1

Table 7. Current consumption for different operating mode

The device consumes approximately **20mA** of current, enabling it to operate continuously for about **125 hours** on a **2500mAh** battery without needing a recharge.

Component	Quantity	Price in tugriks / ₮	Price in dollars / $
Air780EG	1	53’000	15.7
DFPlayer Mini	1	20’000	2.68
Charging Module	1	8’000	2.33
Step - Up Booster	1	4500	1.31
Battery	1	10’500	3.06
Button	3	1’500	0.4
Speaker	1	9’000	2.62
Wires	30	1’000	0.29
All Combined :	**39**	**107’500**	**28.39**

Table 8. Price calculation of device

Chapter 4 : System Architecture and Coding

4.1 System Architecture

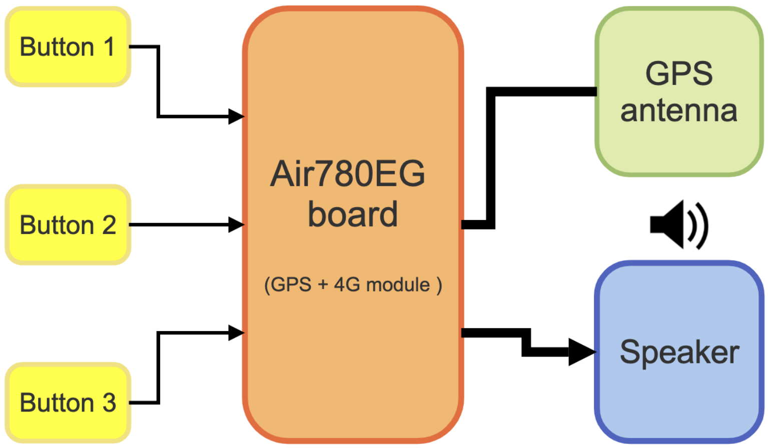


Figure 13. Block diagram of full system.

Figure 13 illustrates the block diagram of the system, which is controlled through three distinct buttons and all of them assigned to a specific role.

Button 1: Initiates the task for telling the current time.

Button 2: Activates the location detecting functionality.

Button 3: Triggers the message sending process.

The system operates based on the state of the buttons. When a button is tapped, the corresponding task is executed to complete its assigned role. This workflow is depicted in *Figures 14 to 17,* which present the flowcharts for the main program and the three specific tasks. These tasks are:

1. **Time Telling**: The system detects the button press, gets the current time from GPS, selects an appropriate pre-recorded audio file on the sd card. Then it sends the signal to the DFplayer, and plays the audio through the speaker.

2. **Location Telling:** After the button is pressed, the system gets the current location, processes it into a location name by filtering process. Then, plays an audio message describing the location.

3. **Message Sending**: This task involves obtaining the device's latitude and longitude coordinates, converting them into a string format, and appending a Google Maps link. The combined information is then sent as a text message to a defined phone number. Each of these tasks operates continuously in the background and controls it with these button states.

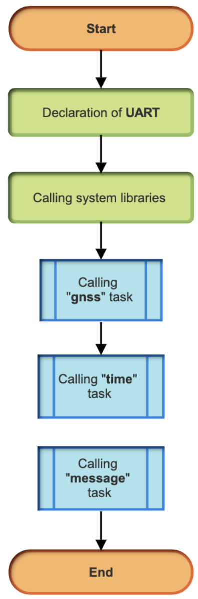


Figure 14. Main File

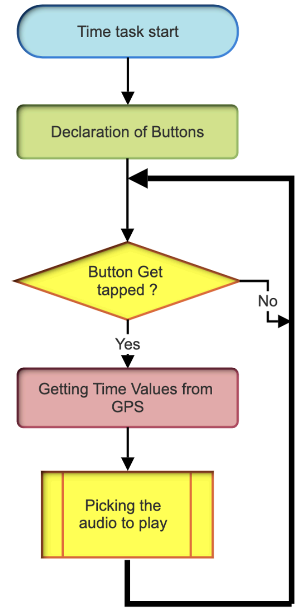


Figure 15. Time Telling Task

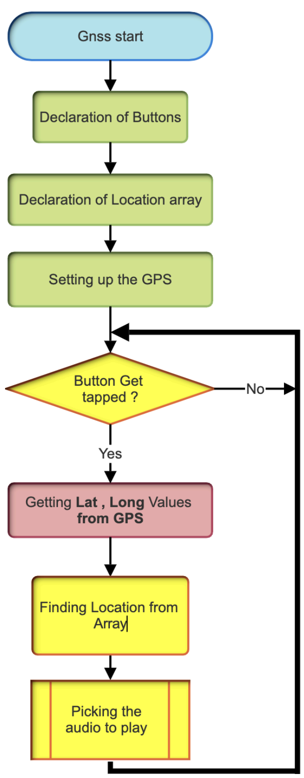


Figure 16. Location Finding Task

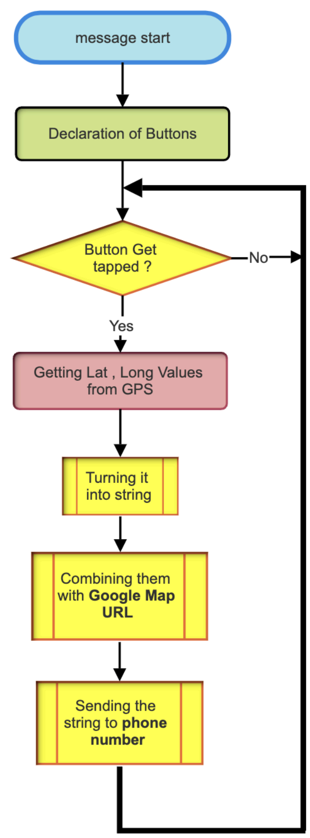


Figure 17. Message Task

4.2 Coding

```lua
PROJECT = "backpack"
VERSION = "1.0.1"
PRODUCT_KEY = ""

_G.sys = require("sys")
require("sysplus")

_G.gps_uart_id = 2

require "gnss"
require "time"
require "message"

sys.run()
```

Figure 18. Main Code

```
BACKPACK01
  gnss.lua
  main.lua
  message.lua
  time.lua
  uc6228.lua
```

Figure 19. Modules and Tasks

The selected board for this project is the Air780EG, which requires scripts written in the LuatOS programming language. Thus, all codes and modules were developed using Lua Programming language. *Figure 18* illustrates the main script file, where the tasks ***gnss, time,*** and ***message*** are called. These tasks are stored in the same folder, with corresponding filenames, as shown in *Figure 19.*

```lua
-- Setting up UART
local uartid = 1
uart.setup(
    uartid,
    9600,
    8,
    1
)

sys.taskInit(function()

    -- When UART receives signal
    uart.on(uartid, "receive", function(id, len)
        local s = ""
        repeat
            s = uart.read(id, 128)
            if #s > 0 then
                log.info("uart", "receive", id, #s, string.toHex(s))
            end
        until s == ""
    end)

    -- When UART send signal
    uart.on(uartid, "sent", function(id)
        log.info("uart", "sent", id)
    end)
```

Figure 20. UART declaration and call-back functions

Figure 20 demonstrates the UART initialization, detailing the configuration of parameters such as the baud rate and other related settings. It also showcases the implementation of callback functions, which handle receiving and transmitting signals, displaying them in the terminal.

4.2.1 Gnss task

A

```lua
libgnss.clear()
LED_GNSS = 24
gpio.setup(LED_GNSS, 0)
local gnss = require("uc6228")

 -- Setting Up the Location Arrays
local locs = 34
local vals = 4
local location_array = {
    {"MPM", 0, 0, 0},
    {"BUyan Road", 0, 0, 0},
    {"Edelwiss", 0, 0, 0},
    {"Continental", 0, 0, 0},
    {"Olympic Gallery", 0, 0, 0},
    {"Ypon Elch Said Hajuud", 0, 0, 0},
    {"Hurimiin ordon hajud", 0, 0, 0},
    {"Hurimiin Orden deer", 0, 0, 0},
    {"Shangrilla Hajuud", 0, 0, 0},
    {"Shangrilla Hajuud", 0, 0, 0},
    {"Shangrilla", 0, 0, 0},
    {"Enkhtaivan Horoolol", 0, 0, 0},
    {"Park Hajuud guur", 0, 0, 0},
    {"Park Hajuud guur", 0, 0, 0},
    {"Park Hajuud guur", 0, 0, 0},
    {"Park Hajuud guur", 0, 0, 0},
    {"Park zogsool", 0, 0, 0},
    {"Park dotor", 0, 0, 0},
    {"IC tower", 0, 0, 0},
    {"UB buyan", 0, 0, 0},
```

B

```lua
 -- Checking if the button is tapped
if gpio.get(BTN_PIN) == 0 then
        -- Lat
        if ( 47.91383 >= latitude and 47.91273 <= latitude) then  -- MPM 1
            location_array[1][3] = 1
        end
        if ( 47.91472 >= latitude and 47.91383 <= latitude) then -- Buyan Road 2
            location_array[2][3] = 1
        end
        if ( 47.91501 >= latitude and 47.91461 <= latitude) then -- Edelwiss 3
            location_array[3][3] = 1
        end
        if ( 47.91276 >= latitude and 47.91198 <= latitude) then -- Continental 4
```

C

```lua
 -- Long
if ( 106.92963 >= longitude and 106.92813 <= longitude) then -- MPM
    location_array[1][4] = 1
end
if ( 106.92965 >= longitude and 106.92925 <= longitude) then -- Buyan Road 2
    location_array[2][4] = 1
end
if ( 106.91495 >= longitude and 106.92690 <= longitude) then -- Edelwiss 3
    location_array[3][4] = 1
end
if ( 106.92648 >= longitude and 106.92414 <= longitude) then -- Continental 4
    location_array[4][4] = 1
```

Figure 21. Gnss task : **(A)** Declaration of location arrays **(B)** Checking the current latitude value **(C)** Checking the current longitude value

```lua
for i = 1, locs do
    for j = 3, (vals) do
        if location_array[i][j] == 1 then
            location_array[i][2] = location_array[i][2] + 1
        end
    end
end

for i = 1, locs do
    if location_array[i][2] == 2 then
        log.info("Your location is ", location_array[i][1])

         -- Picking the Audio File to Play for detected location
        if location_array[i][1] == "MPM" then
            uart.write(1, string.char( 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0xA9, 0xFE, 0x4E, 0xEF ))

        elseif location_array[i][1] == "Buyan Road" then
            uart.write(1, string.char( 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0xAB, 0xFE, 0x4C, 0xEF ))

        elseif location_array[i][1] == "Edelwiss" then
            uart.write(1, string.char( 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0xAD, 0xFE, 0x4A, 0xEF ))
```

Figure 22. Checking third and fourth indexes of the location array

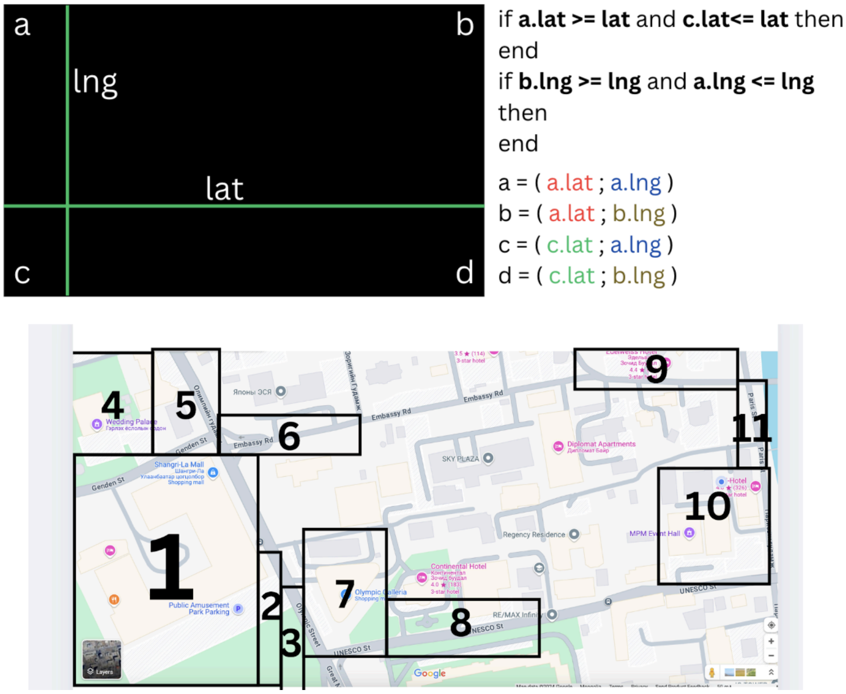


Figure 24. Logic of Location detecting

The **Gnss** task, as shown in *Figure 24*, used the system's algorithm to determine the user's location by checking if the current latitude and longitude values fall within a defined square area. These square areas are defined by four corner points (a,b,c,d) and are pointed with a specific longitude and latitude. If the current coordinates are within the boundaries of a defined square, the system identifies the user as being in that area of location.

As shown in *Figure 21*, previously defined locations are stored in a two dimensional array created up of four columns and 34 rows and each row represents a distinct location. The first column contains the location names, while the remaining three columns store flags used for range checks. These flags indicate whether the current latitude and longitude values fall within the maximum and minimum boundaries of a defined square area. For latitude, if the value lies within the defined range, the corresponding flag is set to 1. The same logic applies for longitude. Once the latitude and longitude are individually validated using if statements, a for loop iterates through the array to identify which location has a total of two "1" values in its third and fourth columns, as shown in *figure 23*. When such a location is found, the algorithm outputs the location name to the user through a speaker.

```lua
if ( 106.92227 >= longitude and 106.9162 <= longitude) then -- Shangrilla
    location_array[11][4] = 1
end
```

Figure 25. Example of longitude range checking

```lua
if ( 47.91412 >= latitude and 47.92227 <= latitude) then -- Shangrilla
    location_array[11][3] = 1
end
```

Figure 26. Example of latitude range checking

For example, if the latitude is 47.913368 and the longitude is 106.921654, then when these values are passed through the if statements, as shown in *Figures 25 and 26*, the third and fourth indexes of the array will be set to 1.

4.2.2 Time task

(A)

```lua
-- Loop Start
while 1 do

    -- Check if button is pressed, and if it's pressed it will retrieve time value.
    if gpio.get(BTN_PIN) == 0 then

        -- Get UTC hour and adding 8 on it. Because Mongolia is in UTC + 8 time zone.
        hour = libgnss.getRmc(2).hour
        hour = hour + 8
        min = libgnss.getRmc(2).min
        min = min
        log.info("Time now :", hour, min )
        sys.wait(1500)
```

(B)

```lua
elseif hour == 25 then -- 01:00
    uart.write(1, string.char( 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x01, 0xFE, 0xF6, 0xEF ))
elseif hour == 26 then -- 02:00
    uart.write(1, string.char( 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x03, 0xFE, 0xF4, 0xEF ))
elseif hour == 27 then -- 03:00
    uart.write(1, string.char( 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x05, 0xFE, 0xF2, 0xEF ))
elseif hour == 28 then -- 04:00
    uart.write(1, string.char( 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x07, 0xFE, 0xF0, 0xEF ))
elseif hour == 29 then -- 05:00
    uart.write(1, string.char( 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x09, 0xFE, 0xEE, 0xEF ))
elseif hour == 30 then -- 06:00
    uart.write(1, string.char( 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x0B, 0xFE, 0xEC, 0xEF ))
elseif hour == 31 then -- 07:00
    uart.write(1, string.char( 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x0D, 0xFE, 0xEA, 0xEF ))
end
sys.wait(2000)
```

(C)

```lua
-- Check hour value, and play the audio
if hour == 1 then
    uart.write(1, string.char( 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x01, 0xFE, 0xF6, 0xEF ))
elseif hour == 2 then
    uart.write(1, string.char( 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x03, 0xFE, 0xF4, 0xEF ))
elseif hour == 3 then
    uart.write(1, string.char( 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x05, 0xFE, 0xF2, 0xEF ))
elseif hour == 4 then
    uart.write(1, string.char( 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x07, 0xFE, 0xF0, 0xEF ))
elseif hour == 5 then
    uart.write(1, string.char( 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x09, 0xFE, 0xEE, 0xEF ))
elseif hour == 6 then
    uart.write(1, string.char( 0x7E, 0xFF, 0x06, 0x03, 0x01, 0x00, 0x0B, 0xFE, 0xEC, 0xEF ))
```

Figure 25. Time task : **(A)** Get time values in hours and minutes**, (B)** Showing the equivalent versions of hours that exceeded 24 value , **(C)** Sending the appropriate time audio signal by checking what time is it

The **time task**, as shown in *Figure 25*, demonstrates how the system retrieves the current time and plays it through the speaker. The GPS module provides time in UTC format, so the code adds 8 hours to equalize it with Mongolian time. The adjusted time value is then validated using an if statement. Once the validation is complete, the corresponding audio file signal is sent through UART to the speaker for playback.

4.2.3 Message task

```lua
while 1 do

    if gpio.get(BTN_PIN) == 0 then
        local lat = libgnss.getRmc(2).lat
        local long = libgnss.getRmc(2).lng
        local lat_as_str = tostring(lat)
        local long_as_str = tostring(long)
        local combined_str = "https://maps.google.com/maps?q=loc:" .. lat_as_str .. "," .. long_as_str

        log.info("sending sms")
        sms.send("97699088991", combined_str)
        sys.wait(700)
```

Figure 26. Message task

The **message task**, as shown in *Figure 26*, illustrates the process of sending location information. A pre-inserted SIM card is used to send data to a specific phone number. The location information is composed of the latitude, longitude, and a Google Maps link. These components are concatenated into a single string, which is then sent using the sms.send() function.

Chapter 5 : Validation and Testing

5.1 Testing

A **B**

Figure 26. Device : **(A)** Combined components, **(B)** Taped GPS module

Tested all components by connecting it to the computer and monitoring the terminal. Everything operated as planned. The speaker's loudness and audio quality were exceptionally clear, free from any environmental noise. The components fit perfectly within the case. However, the GPS module required securing with polyimide tape.

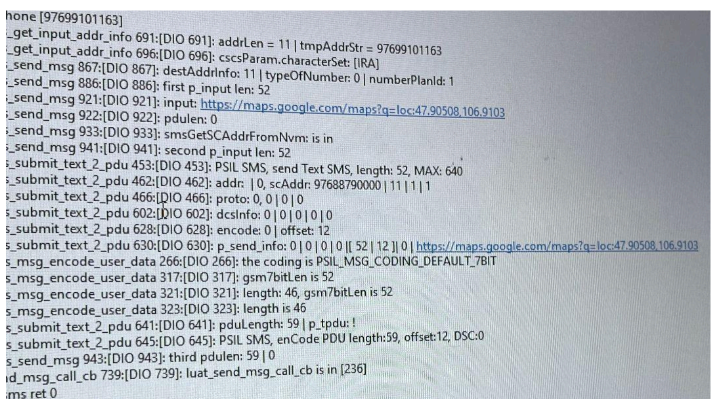


Figure 27. Message sending sms function shows its configuration on terminal

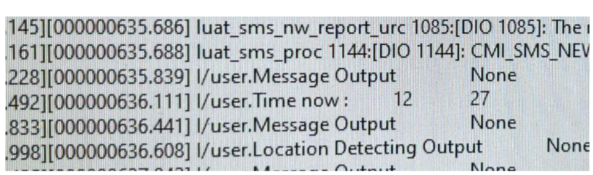


Figure 28. Time is showing up on terminal (12 : 27)

5.2 Validating it in real use

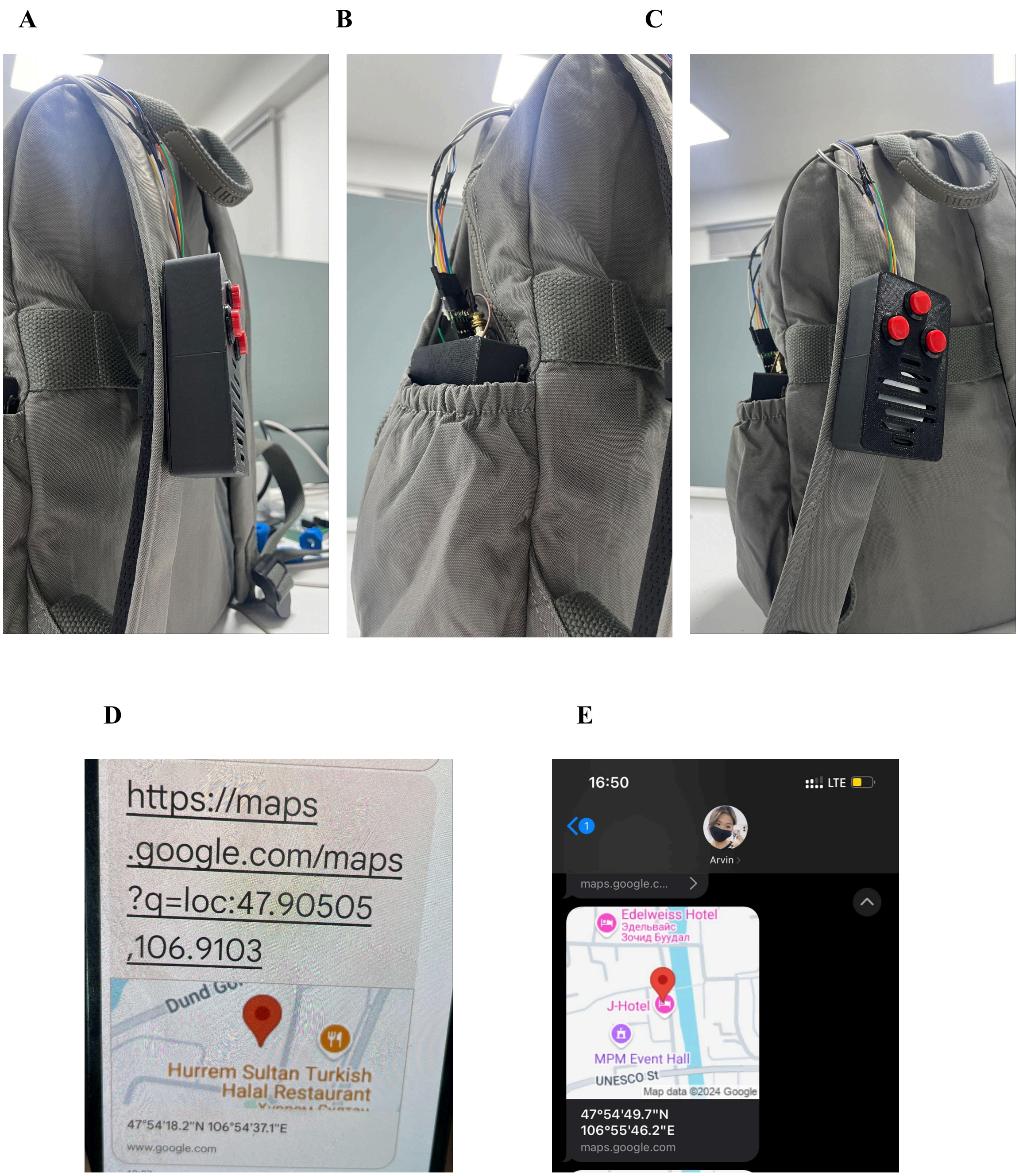


Figure 29. Real life Prototype : **(A)** Side view of the signaling device case, **(B)** Location-sensing device placed in the water bottle pocket of the backpack, **(C)** Combined setup and side view, **(D)** Location message displayed on a Samsung device, **(E)** Location message displayed on an iPhone.

As shown in *Figures 29.A to 29.C*, the final design of the backpack was tested outdoors in Mongolia at approximately -5°C. The overall operation of the device went well, particularly with the time telling and SMS sending functions. As shown in *Figures 27 and 28*, the SMS and time functions operated without any errors, neither announcing incorrect times nor missing messages. Also, the location links were successfully sent to both Samsung and iPhone devices, as shown in *Figures 29.D and 29.E*. On all these devices, the Google Maps links showed up properly.

However, challenges were observed with the **RMC data** collected from the GPS module, particularly during transitional movements or indoor use. When GPS is turned on it requires 2-3 minutes to get its values, also GPS's latitude and longitude values were not able to change while walking. This issue impacted the accuracy of location announcements in specific scenarios.

Tested Locations	Correct Locations (First Attempt)	Success Percentage
68	57	89.1 %

Table 9. Accuracy of device

The device was tested at 34 distinct locations and each location went through twice. So it means the device was tested in 68 locations. During the first attempt, the device successfully said the names of 57 locations, achieving an initial accuracy of 89.1%. However, at 11 locations, the device failed to say the location correctly on the first attempt. Further testing showed that on the second or third attempt, the device was capable of accurately identifying all locations. The initial errors were caused by pauses in updating the GPS module's RMC data, which includes latitude and longitude values, during transitional movements, and it led to insufficient accuracy for real time location announcements. Additionally, seeing the exact latitude and longitude of incorrectly announced locations was challenging during outdoor testing. Because outside, it was difficult to carry and monitor a computer terminal in real time while carrying the backpack.

5.3 Conclusion and Future Work

This Internet of Things based device addresses the challenges that visually impaired individuals face in daily life. From navigating unfamiliar locations to accessing basic information such as time and sharing their locations through smartphones. This device embedded these functionalities into simple and short actions that just use buttons and transmit their information. While primarily designed for visually impaired users, the device also has potential applications for other groups, such as school children or elderly individuals with cognitive impairments. Its portability and rechargeable nature make it suitable for long term use.

The device successfully achieved its intended results, demonstrating reliable performance in time telling, location detection, and location sharing. However, further enhancements are necessary to expand its utility and effectiveness.

- Expanded Location Latitude and Longitude Area Data:
  - Integrate a more comprehensive dataset covering the entire capital city rather than focusing on common routes used by visually impaired individuals.
- Size Reduction:
  - Miniaturize the device to improve portability and user comfort.
- Bus System Integration:
  - Develop a feature that leverages GPS data to connect with Ulaanbaatar's bus system application.

This enhancement would enable the device to announce the directions of approaching buses at bus stations, significantly increasing its impact and utility beyond the current services of location detection, time-telling, and location sharing.

In conclusion, the current version of the device successfully fulfills its mission of enhancing accessibility for visually impaired individuals. However, there is considerable potential for future updates and adjustments to further refine the device and broaden its scope of application. The next version of the smart backpack will aim to embed these improvements and make it a more impactful and versatile device.

References

[1] LuatOS Wiki. (n.d.). *AIR780E MCU documentation*. Retrieved October 2024, from https://wiki.luatos.org/chips/air780e/mcu.html

[2] LuatOS Wiki. (n.d.). *AIR780EG Additional instruction*. Retrieved October 2024, from https://wiki.luatos.org/chips/air780e/air780eg.html

[3] National Statistics Office of Mongolia. (2024). *Population statistics of enterprises*. Retrieved March 2024, from https://www.1212.mn/mn/statistic

[4] Wahab, M. H. A., Talib, A. A., Kadir, H. A., Johari, A., Noraziah, A., Sidek, R. M., & Mutalib, A. A. (2011). *Smart cane: Assistive cane for visually-impaired people*. Faculty of Electrical and Electronic Engineering, Universiti Tun Hussein Onn Malaysia.

[5] School of Project Management. (2022). *Feasibility study of project - Watch it now or fail later* [Video]. YouTube. Retrieved from https://www.youtube.com/watch?v=pN6ZZ51QCPU&t=285s

[6] Eagle News. (2024). *Challenges faced by blind citizens who are overlooked by both the government and society*. Retrieved from https://eagle.mn/r/131610

[7] 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.