GNG2101 Report

Project Deliverable – I

User Manual

Submitted by

Group C23 – TalkBox

Victoria Jancowski, 300203985

Chuanzhi Li, 300055864

Michael Hetu, 300209299

Date

April 11, 2021

University of Ottawa

Abstract

This report details the fundamental functions and features of the TalkBox hardware and software, created for clients Anthony and Roy. The user manual outlines how the features and function were developed through the design process and combined and implemented to create the final product. This report also explains the installation, operations, and maintenance of the TalkBox, and the future steps to improve the product design. This document will provide the user with all of the work completed through the duration of the project, and how all of the moving parts fit together. In the final section of the report, further recommendations and suggestions for future updates and additions are included to provide the reader with examples of improvement.

Table of Contents

List of Figures

List of Tables

List of Acronyms and Glossary

Table 1. Acronyms

Acronym	Definition
USB	Universal Serial Bus
OS	Operating System
GUI	Graphical User Interface
XLS	Microsoft Excel File Extension
XLR	External Line Return

Table 2. Glossary

Term	Acronym	Definition
Debian	-	A Linux Operating System.
Heat Sinks	-	Heat transfer medium.

1 Introduction

This User and Product Manual (UPM) provides the information necessary for users to effectively use the TalkBox and for prototype documentation. This report details also details all features and functions of the TalkBox created for clients Anthony and Roy. This product was designed to provide users with difficulty speaking and low dexterity the ability to communicate through their daily lives with the help of the TalkBox. With client feedback taken into account through multiple client meetings, a final functioning prototype was realized and created to ultimately fulfill the initially outlined design criteria. The fundamental features included in the final product include: a graphical user interface, a functioning text-to-speech ability, customization features for user navigation through a joystick, and customization of the graphical user interface. This is all provided to the user on a single board computer with an LCD Screen.

The objective of this user manual is to provide the user with information about how the fundamental functions and feature of the product were created, and how they are to be used for the optimal results. The caretaker options, maintenance and operations of the TalkBox are discussed through this report and are further discussed for future improvements to the product. This report discusses the development process, suggestions for future development of the product and the intended use of the product.

2 Overview

Many people with disabilities have difficulty speaking clearly enough for others to understand them, and to communicate well. This also includes communicating to smart home devices such as an Alexa or a Google Home. There is then a need for a way for certain phrases to be organized into categories such as Caretaker Interaction, General Living, etc., and then could be spoken aloud. People may also not possess the necessary dexterity that comes with operating a touchscreen and could find it difficult to navigate menus and banks of phrases. This will require the use of switches or another method to ensure the user is able to navigate phrase banks.

Disabilities range in all shapes and sizes, and it is important to provide future users with customizable options that suit the needs of the user. The TalkBox will provide users with the ability to customize phrases and categories and use a text to speech function so that communication is made easier.

The fundamental needs of a user will be the need for an easy navigation system through phrase banks, a way to “talk” without using their voice, and a way to interact with smart home devices. There is also a need for the modification of phrases banks suited to each individual user need.

The TalkBox provides users with a text to speech function, onscreen viewing of categories and phrases, options to modify the phrases in the phrase bank, two joystick functions that include a scanning menu and manual joystick movements and an option to modify the colour of the text onscreen and the color of the background.

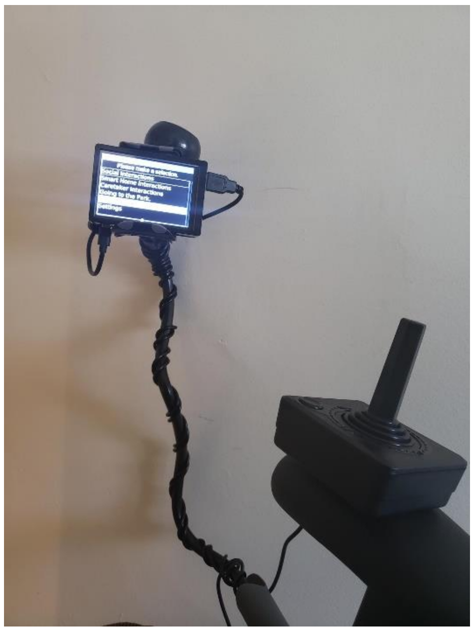
Figure 2.1-2 TalkBox from User View

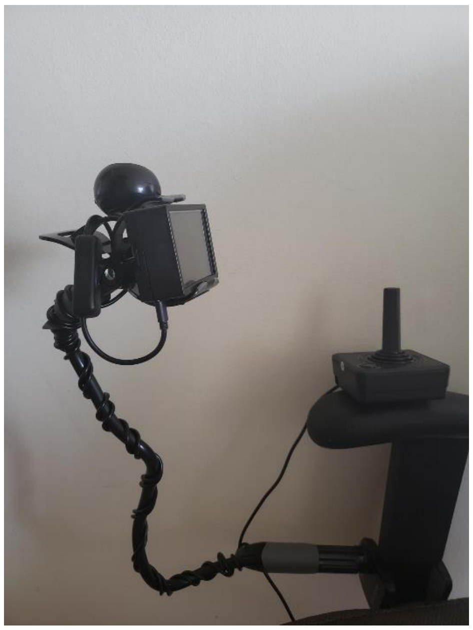
Figure 2.1-1 TalkBox from Side View

The key features of this product include but are not limited to a Raspberry Pi single board computer, and LCD Screen, a Graphical User Interface, a Speaker and a Joystick. The graphical user interface provides the user with an appropriate mode of access and the Raspberry Pi and Joystick provide the architecture of the system..

2.1 Cautions and Warnings

Please note that if the joystick and power supply are not connected to the TalkBox, the program file will not open. It is a requirement in the programming that both are attached for the program to function. Do not get this product wet as it is not yet waterproofed.

3 Getting Started

To operate the system, please first make sure that the TalkBox is attached to a power supply, and that both the joystick and speaker are plugged in. Open the file denoted “TalkBox” on the desktop and wait for the program to open. Once the program is open, the categories will appear on the menu, as will Settings and Exit TalkBox.

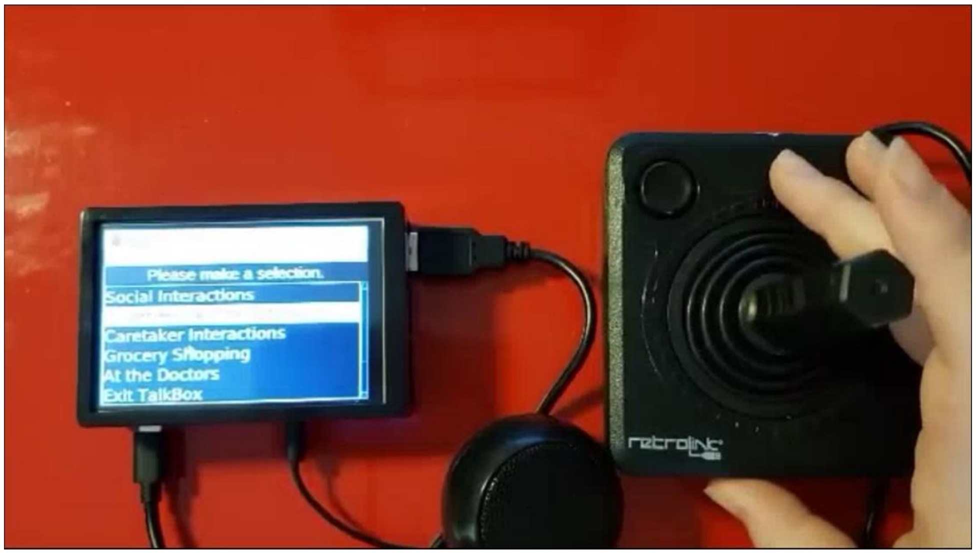

Figure 2.1-1 Main Screen of TalkBox

The automatic joystick function of the TalkBox is a scanning menu. To use this function simply let the scrollbar run through the options and bump the joystick any way to make a selection. If the user desires a way to navigate more quickly by using the complete functionality of the joystick, go to the settings menu and select scrolling options. There you will find the options for scanning menu, or joystick mode. Click on joystick mode. By pressing up on the joystick, you will be able to scroll up the options and by pushing down on the joystick you will be able to scroll down. To select in joystick mode, move the joystick either right or left.

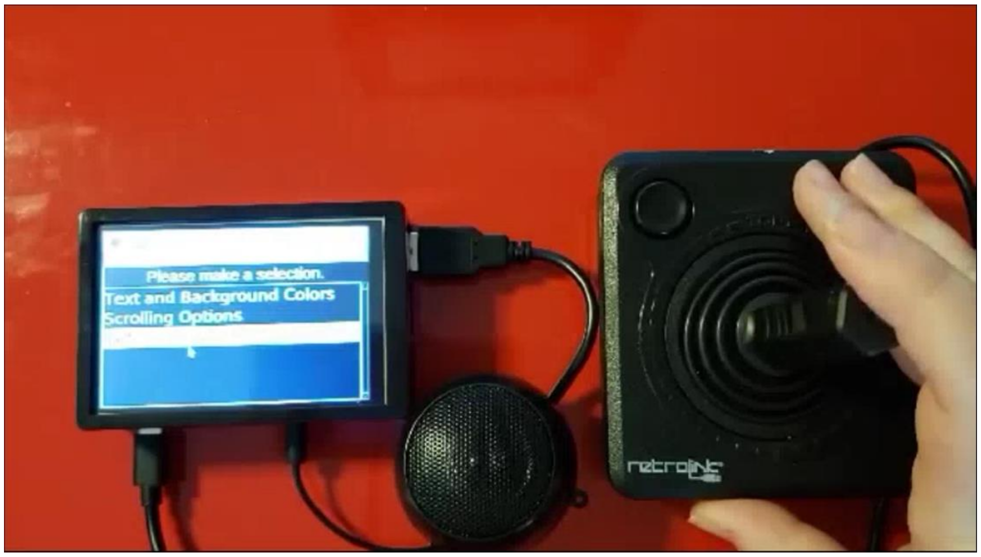

Figure 2.1-2 TalkBox Settings Menu

If the user desires a different background and text colors return to the settings menu. There you will find an option to change the color. The selections are Day Mode (white background and black text), Night Mode (black background and white text), and Black and Yellow (black background and yellow text).

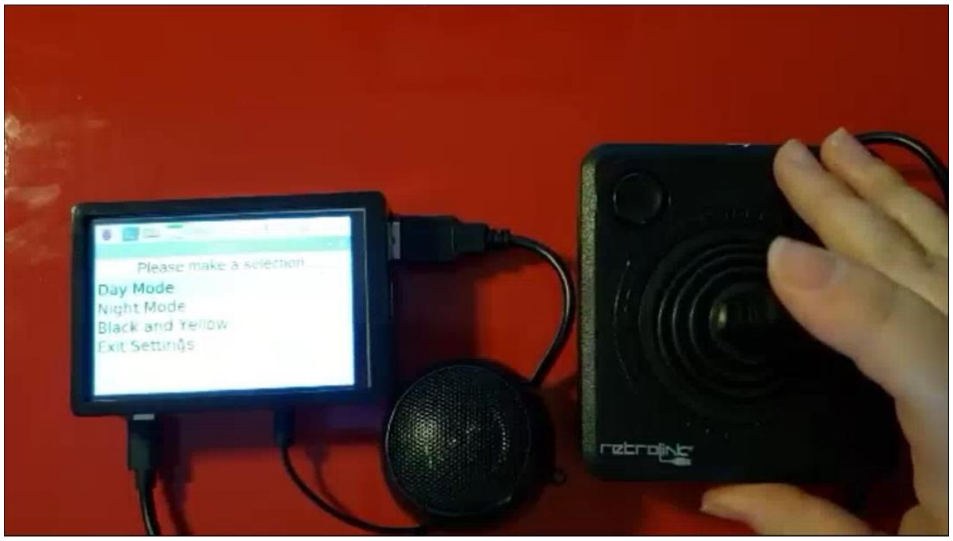

Figure 2.1-3 TalkBox in Day Mode

The text to speech function will operate when the user selects a category or phrase. Each category contains phrases associated with it. To access the phrases, select the category you want to explore. Note that there is an exit button at the end of each phrases menu that returns you back to the main menu.

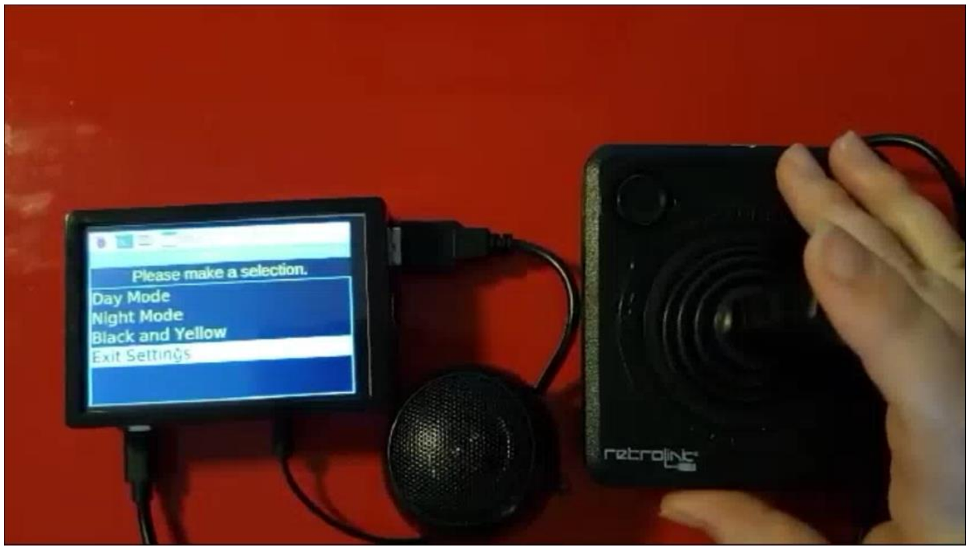

Figure 2.1-4 Exit Function at the End of Each Menu

To exit the system, press Exit TalkBox on the main menu screen.

3.1 Set-up Considerations

The devices used in the TalkBox are the TalkBox itself, the **clamp attached** gooseneck to hold the TalkBox, the power cord, the speaker, and the joystick. The gooseneck **clamps** on to the TalkBox holding it at the desired height for best viewing.

There is also the possibility of installing an optional 3D printed case for the TalkBox. The 3D printed cover allows for a more secure attachment to a gooseneck using the ubiquitous camera mount, commonly found on devices such as the ©GoPro. The gooseneck used for installation will have to be a **camera mount**, instead of clamp attached as mentioned earlier. If you would like to install the camera mount case, assembly instructions can be found in section **6.4.1**.

The joystick plugs into any of the four USB port s on the side of the TalkBox. The speaker plugs into the auxiliary port on the side of the TalkBox and the power cord attaches to the TalkBox using a micro-USB port on the side. Pictures are provided below with the appropriate set-up.

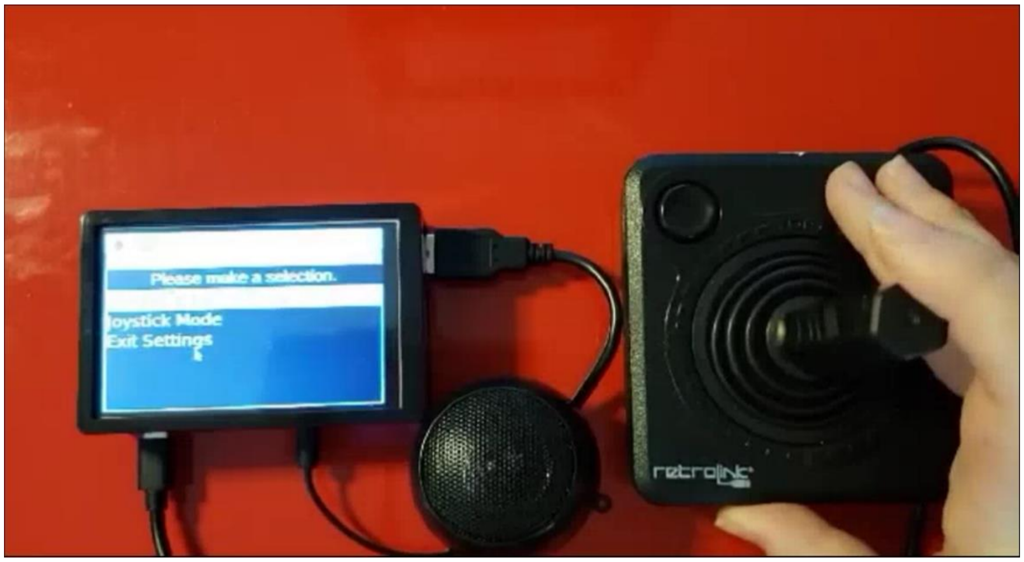

Figure 3.1-1 Speaker Connection, Joystick Connection and Power Connection

To modify the phrases or categories, the talkboxp.xls file will have to be copied onto a USB stick and transferred to a computer with Microsoft office. Open the file in the USB on the computer and add/remove or change any phrases or categories you desire in the document. Please do not change the name of the document, or the file will not work in the TalkBox and the program will crash. Once done editing the file on your computer, save it to the USB stick and transfer it back to the TalkBox using one of the free USB ports. Move the file from the USB drive onto the Desktop of the TalkBox and delete the old file. The TalkBox will automatically start reading the new file.

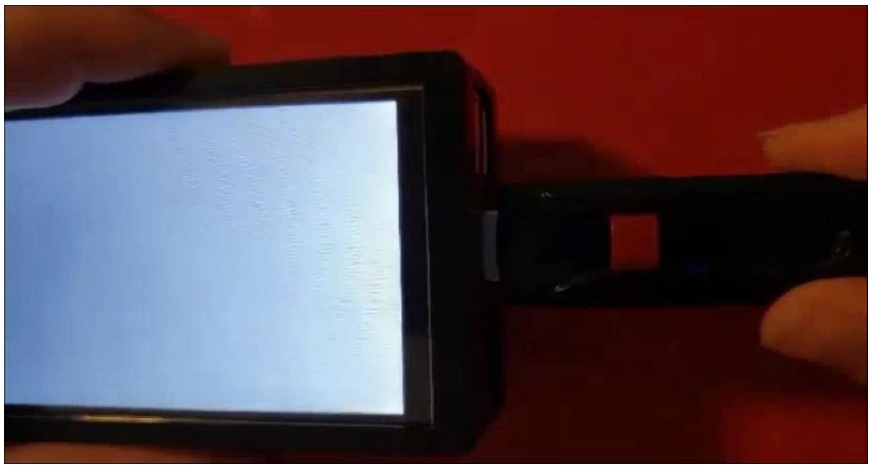

Figure 3.1-2 Plug USB Stick into the Side of the TalkBox

3.2 Accessing the System

If any programming changes are going to be made to the file, or any access to the Linux command lines are going to be used, please note that the systems username is “pi”, and that the password is “raspberry”. As the user, you may make any modification to the username and password as you wish through the general settings menu on the TalkBox.

3.3 System Organization and Navigation

3.3.1 Main Menu

The main menu consists of the categories of types of phrase banks, a settings menu, and an exit function. To navigate to any of the sub menus, simply select one of the options on the main menu. Please note that the exit function and the settings are located at the very bottom of the main menu.

3.3.2 Phrases

After selecting any of the categories on the main menu, you will be brought to a sub menu of phrases. To “speak” any of the phrases just select whatever phrase that is highlighted. To exit the phrases menu, press the exit option at the bottom of the list.

3.3.3 Settings

The settings menu is accessed from the main menu and brings the user to a new sub menu containing the “Text and Background Colors” and the “Scrolling Options”. The “Text and Background Colors” upon being engaged, take the user to another sub menu which provides the choice of “Day Mode”, “Night Mode”, and “Black and Yellow”. The “Scrolling Options” takes the user to the submenu containing “Joystick Mode” and “Scanning Menu”. To exit all menus detailed above, select the exit menu function at the bottom of the menu they are on.

3.4 Exiting the System

To exit the system, and completely turn off the TalkBox, the user only has to remove the power supply to the TalkBox, through the micro-USB slot.

4 Using the System

The following sub-sections provide detailed, step-by-step instructions on how to use the various functions or features of the TalkBox. They include: Text-to-Speech, Graphical User Interface, Joystick Navigation, Caretaker Modification and Customization.

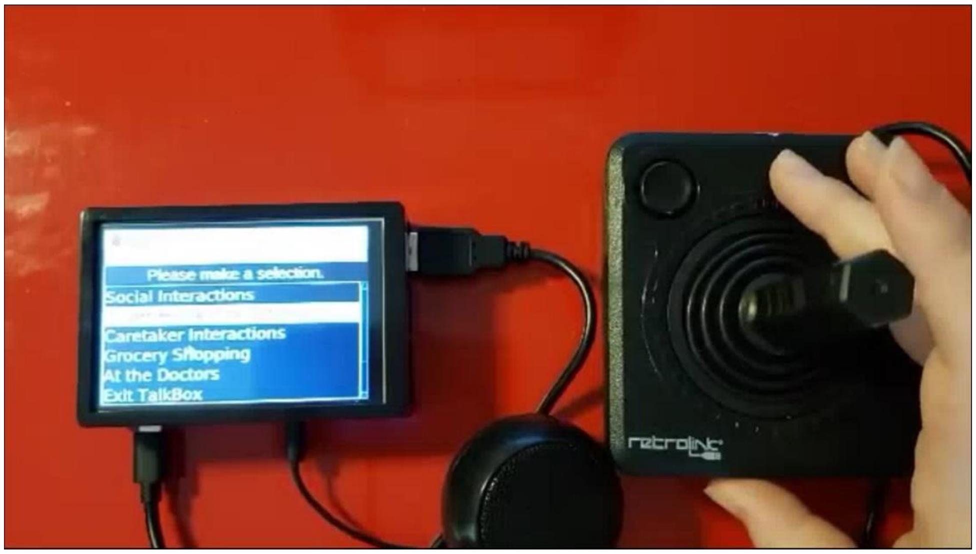

Figure 3.4-1 Starting Menu of the TalkBox

4.1 Text-to-Speech

The text-to-speech function is engaged when the user selects either a category on the main menu, or the phrase sub menu, respectively. The user need not do anything other than to select either a phrase or a category to use the text-to-speech function.

4.2 Joystick Navigation

The Joystick Navigation function is engaged to provide an easy way to control the device. Users can move the highlight up or down to navigate the phase or order they want. User can press the button on the top of the joystick to execute the function that the highlighted phase provided.

4.3 Caretaker Modification

The caretaker is free to modify all settings using the screen touch display.

4.4 Customization

The TalkBox offers users the ability to customize the joystick modes they desire, as well as the text and background colors on screen. This feature offers adaptability for all different kinds of users, with different levels of sight and dexterity.

4.4.1 Scrolling Options

There are two joystick modes: Scanning Menu, and Joystick Navigation. The scanning menu function is immediate upon opening the TalkBox Program. It scans through each item on the screen every two seconds and loops back around to the top of the list once it reaches the bottom. To make a selection, all the user needs to do is bump the joystick along any axis.

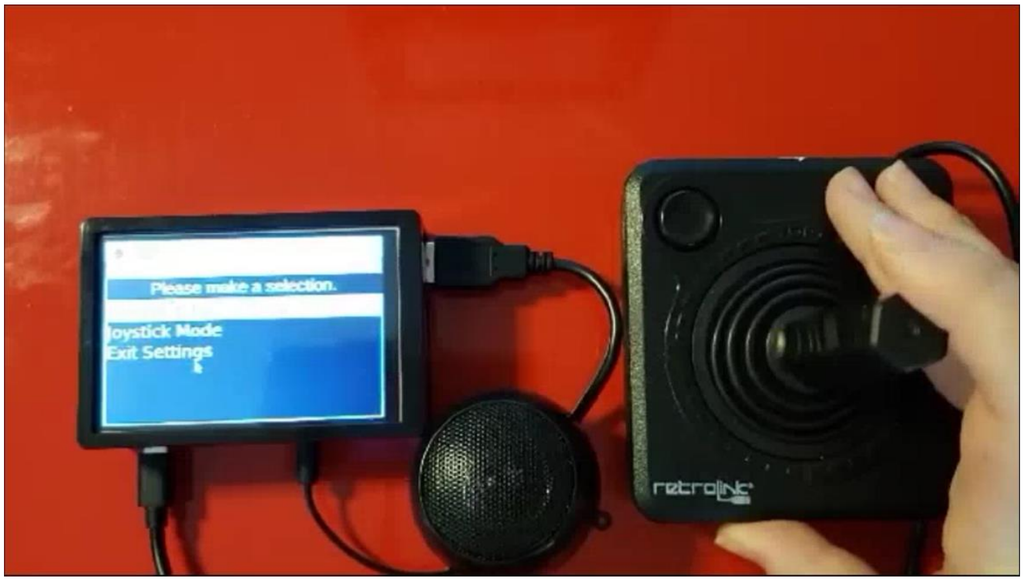

Figure 4.4-1 Joystick Menu

The Joystick Navigation function allows the user to use the joystick in a traditional way. The up and down axes allow the user to scroll up and down through a list, while the left and right axes act as selectors. The joystick modes can be found under the settings menu.

4.4.2 Color Modification

Under the settings menu in text and background colors, the user will find an option to change both the text and background colors. Once the change color option has been selected in the sub menu, the user can choose either: Day Mode, white background with black text; Night Mode, black background with white text; and Black and Yellow, black background and yellow text.

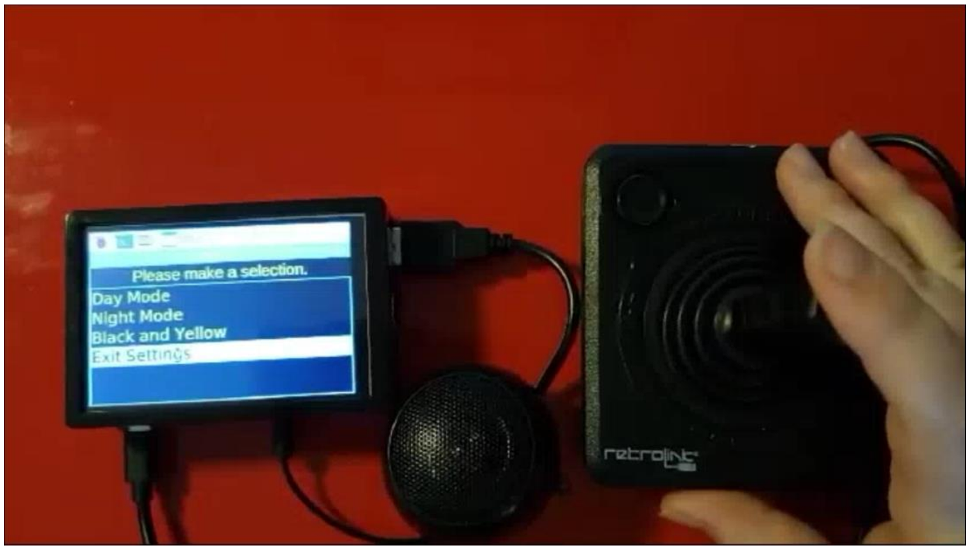

Figure 4.4-2 Color Modification Menu

5 Troubleshooting and Support

This section will provide users with the ability to troubleshoot any errors that may be generated through use of this product.

5.1 Error Messages and Behaviors

Please note, that if the joystick, power supply and speaker are not connected to the TalkBox upon start up, the program file will not compile. The file is dependent on the connection of these devices.

Please make sure when changing phrases and categories in the xls file that you only have one copy on the desktop of the TalkBox. The program will not compile if there are two xls files.

5.2 Special Considerations

Please do not get the device wet, as it is not waterproofed. If the TalkBox is not starting when the executable file is selected, restart the device.

If you are programming and updating the device, please not that you will need to be running a Linux system, or might need to partition you hard drive.

5.3 Maintenance

If the device is not being used for prolonged periods of time, please make sure to power off to avoid overheating. Please store in a dry, low dust, room-temperature environment.

5.4 Support

If there are any issues with the device, please email vjanc090@uottawa.ca. You will receive a reply in one to two business days.

6 Product Documentation

6.1 Product Code

Below is the code for the TalkBox. It was programmed using python 3. Please note that a Linux based system will be needed for any potential updates to the software as the TalkBox uses Debian – specifically Raspbian.

```
#!/usr/bin/env python
import pandas as pd
import pygame
import pyttsx3
from guizero import App, Text, ListBox
import os

settings1 = None
settingscolor = None
settingsjoystick = None

pygame.init()
pygame.joystick.init()
using_scanning_menu = True

def scrollsix():
    if (settings1 != None) and (settings1.visible == True):

settings1.tk.children["!listbox"].see(settingsitems.index(settings1.value))

    if current_phrases != None and current_phrases.visible == True:

current_phrases.tk.children["!listbox"].see(phrases_list[categories.index(cur
rent_phrase_list)].index(current_phrases.value))

    if category.visible == True:
        if category.value == None:
            print("nothing")
        else:
            print(categories.index(category.value))

category.tk.children["!listbox"].see(categories.index(category.value))

    if settingscolor != None and settingscolor.visible == True:

settingscolor.tk.children["!listbox"].see(colours.index(settingscolor.value))

    if settingsjoystick != None and settingsjoystick.visible == True:

joystickmenuitems.tk.children["!listbox"].see(joystickmenuitems.index(setting
sjoystick.value))

def dosomethingwithjstk():
    if using_scanning_menu:
        if jstk.get_axis(0) > 0.1:  # right
            scanningmenujstk()
        elif jstk.get_axis(0) >= -1 and jstk.get_axis(0) < -0.1:  # left
            scanningmenujstk()
        elif jstk.get_axis(1) >= 0.1:  # down
            scanningmenujstk()
        elif jstk.get_axis(1) >= -1 and jstk.get_axis(1) < -0.10:  # up
            scanningmenujstk()
    else:
        dexjstk()


jstk = pygame.joystick.Joystick(0)
jstk.init()


def scanningmenujstk():
    global settings1
    global current_phrases
    global category
    global settingscolor
    global settingsjoystick

    if (settings1 is not None) and (settings1.visible == True):
        if settings1.value is not None:
            settingsfunc(settings1.value)
    elif current_phrases is not None and current_phrases.visible == True:
        if current_phrases.value is not None:
            settings1command(current_phrases.value)
    elif category.visible:
        if category.value is not None:
            phrase_selection(category.value)
    elif settingscolor is not None and settingscolor.visible == True:
        if settingscolor.value is not None:
            txt_bg_col(settingscolor.value)
    elif settingsjoystick is not None and settingsjoystick.visible == True:
        if settingsjoystick.value is not None:
            choosejstk(settingsjoystick.value)

def dexjstk():
    if jstk.get_axis(0) > 0.1:  # right
        if (settings1 is not None) and (settings1.visible == True):
            if settings1.value is not None:
                settingsfunc(settings1.value)
        elif current_phrases is not None and current_phrases.visible == True:
            if current_phrases.value is not None:
                settings1command(current_phrases.value)
        elif category.visible:
            if category.value is not None:
                phrase_selection(category.value)
        elif settingscolor is not None and settingscolor.visible == True:
            if settingscolor.value is not None:
                txt_bg_col(settingscolor.value)
        elif settingsjoystick is not None and settingsjoystick.visible ==
True:
            if settingsjoystick.value is not None:
                choosejstk(settingsjoystick.value)
    if jstk.get_axis(0) >= -1 and jstk.get_axis(0) < -0.1:  # left
        if (settings1 is not None) and (settings1.visible == True):
            if settings1.value is not None:
                settingsfunc(settings1.value)
        elif current_phrases is not None and current_phrases.visible == True:
            if current_phrases.value is not None:
                settings1command(current_phrases.value)
        elif category.visible:
            if category.value is not None:
                phrase_selection(category.value)
        elif settingscolor is not None and settingscolor.visible == True:
            if settingscolor.value is not None:
                txt_bg_col(settingscolor.value)
        elif settingsjoystick is not None and settingsjoystick.visible ==
True:
            if settingsjoystick.value is not None:
                choosejstk(settingsjoystick.value)
    if jstk.get_axis(1) >= 0.1:  # down
        if (settings1 != None) and (settings1.visible == True):
            if settingsitems.index(settings1.value) + 1 > len(settingsitems)
- 1:
                settings1.value = settingsitems[0]
            else:
                settings1.value =
settingsitems[settingsitems.index(settings1.value) + 1]
        if current_phrases != None and current_phrases.visible == True:
            if
phrases_list[categories.index(current_phrase_list)].index(current_phrases.val
ue) + 1 > len(
                    phrases_list[categories.index(current_phrase_list)]) - 1:
                current_phrases.value =
phrases_list[categories.index(current_phrase_list)][0]
            else:
                current_phrases.value =
phrases_list[categories.index(current_phrase_list)][

phrases_list[categories.index(current_phrase_list)].index(current_phrases.val
ue) + 1]

        if category.visible == True:
            if category.value == None:
                category.value = categories[0]
            elif categories.index(category.value) + 1 > len(categories) - 1:
                category.value = categories[0]
            else:
                category.value = categories[categories.index(category.value)
+ 1]

        if settingscolor != None and settingscolor.visible == True:
            if colours.index(settingscolor.value) + 1 > len(colours) -1:
                settingscolor.value = colours[0]
            else:
                settingscolor.value =
colours[colours.index(settingscolor.value) +1]

        if settingsjoystick != None and settingsjoystick.visible == True:
            if joystickmenuitems.index(settingsjoystick.value) + 1 >
len(joystickmenuitems) - 1:
                settingsjoystick.value = joystickmenuitems[0]
            else:
                settingsjoystick.value =
joystickmenuitems[joystickmenuitems.index(settingsjoystick.value) + 1]


    if jstk.get_axis(1) >= -1 and jstk.get_axis(1) < -0.10:  # up
        if (settings1 != None) and (settings1.visible == True):
            settings1.value =
settingsitems[settingsitems.index(settings1.value) - 1]

        if current_phrases != None and current_phrases.visible == True:
            current_phrases.value =
phrases_list[categories.index(current_phrase_list)][

phrases_list[categories.index(current_phrase_list)].index(current_phrases.val
ue) - 1]

        if category.visible == True:
            if category.value == None:
                category.value = categories[0]
            else:
                category.value = categories[categories.index(category.value)
- 1]

        if settingscolor != None and settingscolor.visible == True:
            settingscolor.value = colours[colours.index(settingscolor.value)
-1]

        if settingsjoystick != None and settingsjoystick.visible == True:
            settingsjoystick.value =
joystickmenuitems[joystickmenuitems.index(settingsjoystick.value) + 1]

    scrollsix()

first_time = True

def joystickthings():
    global first_time
    events = pygame.event.get()
    #print(events)
    if len(events) > 0:
        event = events[0]
        if not first_time:
            if event.type == pygame.JOYAXISMOTION:
                #print('click')
                dosomethingwithjstk()
    first_time = False
    pygame.event.clear()


def settings1command(value):
    if value == 'Exit':
        go_to_main_menu()
    else:
        texttospeech(value)


def phrase_selection(value):
    global current_phrases
    global settings1
    global settingsitems
    global textappcolor
    global backgroundapp
    global current_phrase_list
    settingsitems = ["Text and Background Colors", "Scrolling Options",
"Exit"]
    if value == exit:
        app.destroy()
    elif value == settings:
        category.hide()
        settings1 = ListBox(
            app,
            items=settingsitems,
            scrollbar=True,
            width="fill",
            height="fill",
            command=settingsfunc,
            selected=settingsitems[0])

        settings1.text_color = textappcolor
        settings1.bg = backgroundapp
        settings1.text_size = 22
        settings1.focus()
    else:
        category.hide()
        texttospeech(value)
        current_phrases = ListBox(
            app,
            items=phrases_list[categories.index(value)],
            width="fill",
            height="fill",
            command=settings1command,
            scrollbar= True,
            selected=phrases_list[categories.index(value)][0])
        current_phrases.text_color = textappcolor
        current_phrases.bg = backgroundapp
        current_phrases.text_size = 22
        current_phrases.focus()
        current_phrase_list = value
```

```
def go_to_main_menu():
    global current_phrases
    current_phrases.hide()
    category.value = None
    category.show()
    category.focus()


def settingsfunc(value):
    global settings1
    global joystickmenuitems
    joystickmenuitems = ["Scanning Menu Mode", "Joystick Mode", "Exit 
Settings"]
    global colours
    global backgroundapp
    global textappcolor
    global settingscolor
    global settingsjoystick

    colours = ['Day Mode', 'Night Mode', 'Black and Yellow', 'Exit Settings']
    if value == settingsitems[0]:
        settings1.hide()
        settingscolor = ListBox(app, items=colours,
                                width="fill",
                                height="fill",
                                command=txt_bg_col,
                                selected=colours[0])
        settingscolor.bg = backgroundapp
        settingscolor.text_color = textappcolor
        settingscolor.text_size = 22
        settingscolor.focus()
    if value == settingsitems[1]:
        settings1.hide()
        settingsjoystick = ListBox(app, items=joystickmenuitems,
                                   width="fill",
                                   height="fill",
                                   command=choosejstk,
                                   selected=joystickmenuitems[0])
        settingsjoystick.text_color = textappcolor
        settingsjoystick.bg = backgroundapp
        settingsjoystick.text_size = 22
        settingsjoystick.focus()
    if value == settingsitems[2]:
        settings1.hide()
        category.show()
        category.focus()


def choosejstk(value):
    global joystickmenuitems
    global settings1
    global using_scanning_menu
    if value == joystickmenuitems[0]:
        using_scanning_menu = True
        settings1.hide()
        category.show()
        settingsjoystick.hide()
        category.focus()
    if value == joystickmenuitems[1]:
        using_scanning_menu = False
        settings1.hide()
        category.show()
        settingsjoystick.hide()
        category.focus()
    if value == colours[3]:
        settings1.hide()
        category.show()
        settingsjoystick.hide()
        category.focus()



def txt_bg_col(value):
    global backgroundapp
    global textappcolor
    global settingscolor
    global settingsjoystick
    if value == colours[0]:
        backgroundapp = "white"
        textappcolor = "black"
        settingscolor.bg = backgroundapp
        settingscolor.text_color = textappcolor
        category.bg = backgroundapp
        category.text_color = textappcolor
        settingscolor.text_size = 22
        app.bg = backgroundapp
        titleapp.text_color = textappcolor
    if value == colours[1]:
        backgroundapp = "black"
        textappcolor = "white"
        settingscolor.bg = backgroundapp
        settingscolor.text_color = textappcolor
        settingscolor.text_size = 22
        category.bg = backgroundapp
        category.text_color = textappcolor
        app.bg = backgroundapp
        titleapp.text_color = textappcolor
    if value == colours[2]:
        backgroundapp = "black"
        textappcolor = "yellow"
        settingscolor.bg = backgroundapp
        settingscolor.text_color = textappcolor
        settingscolor.text_size = 22
        category.bg = backgroundapp
        category.text_color = textappcolor
        app.bg = backgroundapp
        titleapp.text_color = textappcolor
    if value == colours[3]:
        settings1.hide()
        category.show()
        settingscolor.hide()
        category.focus()
```

```
# text to speech
def texttospeech(selectedphrase):
    titleapp.cancel(scannit)

    os.system("echo \"" + selectedphrase + "\" | festival --tts")

    titleapp.repeat(2500, scannit)


# import excel file, make sure format is xls
#"C:\\Users\\vjanc\\OneDrive\\Documents\\University of Ottawa\\Year 1\\2021
Winter\\Project Management - GNG2101\\Project\\talkboxp.xls"
#/home/pi/Desktop/talkboxp.xls
df = pd.read_excel("/home/pi/Desktop/talkboxp.xls")
df = df.dropna()
categories_big = df['Categories'].tolist()
phrases = df['Phrases'].tolist()
categories = categories_big.copy()


def removeduplicates(a):
    i = 0
    while i < len(a):
        j = i + 1
        while j < len(a):
            if a[i] == a[j]:
                del a[j]
            else:
                j += 1
        i += 1


removeduplicates(categories)
exit = "Exit TalkBox"
categories.append(exit)
settings = "Settings"
categories.append(settings)
phrases_list = []
i = 0
l = 0
are_we_done = False


def relate_phrases_categories():
    global l, i, are_we_done
    phrases_list.append([])
    a = l
    b = l + 1
    c = l
    while True:
        if a + 1 == min(len(categories_big), len(phrases)):
            phrases_list[i].append(phrases[a])
            phrases_list[i].append("Exit")
            are_we_done = True
            return
        if categories_big[a] == categories_big[b]:
            phrases_list[i].append(phrases[a])
            a = a + 1
            b = b + 1
            c = c + 1
        if categories_big[a] != categories_big[b]:
            phrases_list[i].append(phrases[a])
            phrases_list[i].append("Exit")
            break
    i = i + 1
    l = a + 1


while not are_we_done:
    relate_phrases_categories()

backgroundapp = "black"
textappcolor = "white"
app = App(title="Talkbox", bg=backgroundapp)
titleapp = Text(app, text='Please make a selection.', size=22,
color=textappcolor)
category = ListBox(
    app,
    items=categories,
    width="fill",
    height="fill",
    selected=categories[0],
    command=phrase_selection,
    scrollbar=True)
category.text_color = "white"
category.text_size = 22
current_phrases = None
category.focus()

current_phrase_list = categories[0]

titleapp.repeat(100, joystickthings)


def scannit():
    global settingscolor

    if not using_scanning_menu:
        return

    if (settings1 != None) and (settings1.visible == True):
        if settingsitems.index(settings1.value) + 1 > len(settingsitems) - 1:
            settings1.value = settingsitems[0]
        else:
            settings1.value =
settingsitems[settingsitems.index(settings1.value) + 1]

    if current_phrases != None and current_phrases.visible == True:
        if
phrases_list[categories.index(current_phrase_list)].index(current_phrases.val
ue) + 1 > len(
                phrases_list[categories.index(current_phrase_list)]) - 1:
            current_phrases.value =
phrases_list[categories.index(current_phrase_list)][0]
        else:
            current_phrases.value =
phrases_list[categories.index(current_phrase_list)][

phrases_list[categories.index(current_phrase_list)].index(current_phrases.val
ue) + 1]

    if category.visible == True:
        if category.value == None:
            category.value = categories[0]
        elif categories.index(category.value) + 1 > len(categories) - 1:
            category.value = categories[0]
        else:
            category.value = categories[categories.index(category.value) + 1]

    if settingscolor != None and settingscolor.visible == True:
        if colours.index(settingscolor.value) + 1 > len(colours) - 1:
            settingscolor.value = colours[0]
        else:
            settingscolor.value = colours[colours.index(settingscolor.value)
+ 1]

    if settingsjoystick != None and settingsjoystick.visible == True:
        if joystickmenuitems.index(settingsjoystick.value) + 1>
len(joystickmenuitems) -1:
            settingsjoystick.value = joystickmenuitems[0]
        else:
            settingsjoystick.value =
joystickmenuitems[joystickmenuitems.index(settingsjoystick.value) + 1]

    scrollsix()


titleapp.repeat(2500, scannit)
app.display()
```

6.2 Bill of Materials

Table 3 Bill of Materials

Materials*	**Cost ($)**	**Purchased (Y/N)**
Speaker	16.01	Y
LCD Screen	29.98	Y
Joystick	33.88	Y
Raspberry Pi	99.00	Y

Gooseneck	39.99	N
Adhesive Pads	8.99	N
XLR Cable to micro-USB	17.08	N

*See appendix for links to material.

6.3 Equipment List

- Computer
- Python 3
- Linux OS
- USB Stick
- Joystick
- Speaker
- LCD Screen and stylus
- Raspberry Pi
- Case for Raspberry Pi
- Gooseneck
- XLR Cable to micro-USB
- Micro SD card
- Heat sinks
- Power Cable
- 3D Printed case top - (Optional for camera mount)
- 3D Printed case bottom - (Optional for camera mount)
- M3 screwdriver - (Optional for camera mount)

- Four M3 5mm screws - (Optional for camera mount)
- Four M3 8mm screws - (Optional for camera mount)

6.4 Assembly Instructions

6.4.1 Hardware

Step 1: The code for this project is described as above. This section will cover the physical assembly of the TalkBox. Heat sinks were installed at the locations indicated by the blue arrows in Figure 6.4-1, and the micro-SD card installed in its correct location located by the yellow arrow.

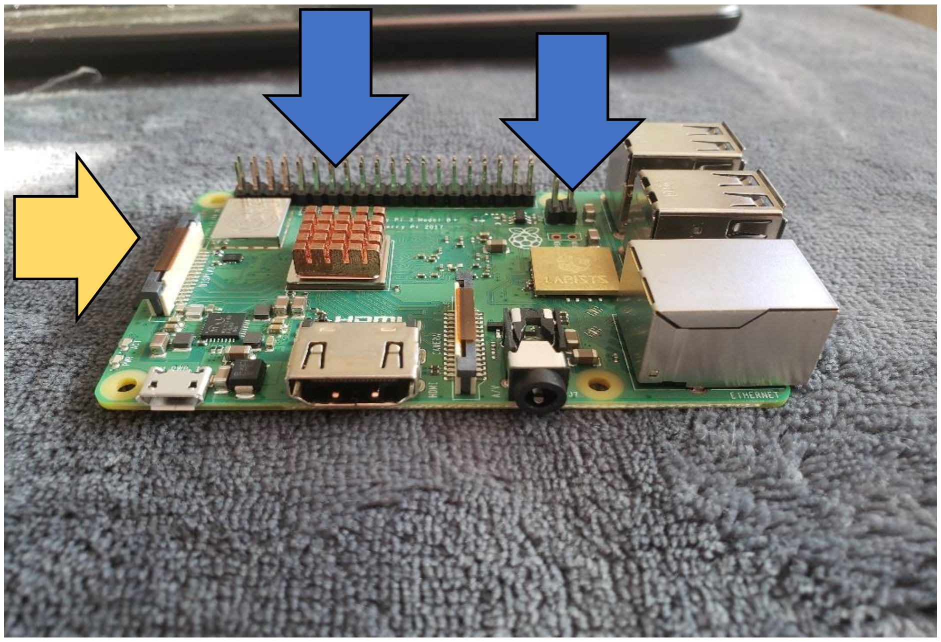

Figure 6.4-1 Raspberry Pi

6.4.1.1 Optional 3D printed camera mount cover:

The 3D printed cover allows for a more secure attachment to a gooseneck using the ubiquitous camera mount, commonly found on devices such as the ©GoPro. The default case which comes with the screen can be used instead of the 3D printed camera mount case, to simplify assembly.

IF NOT USING CAMERA MOUNT COVER: STEPS IN THIS SECTION (6.4.1) CAN BE SKIPPED TO SECTION **6.4.2**

Step 2 (camera mount): To assemble the camera mount cover, first 3D print the top and bottom parts from the STL files provided in the repository.

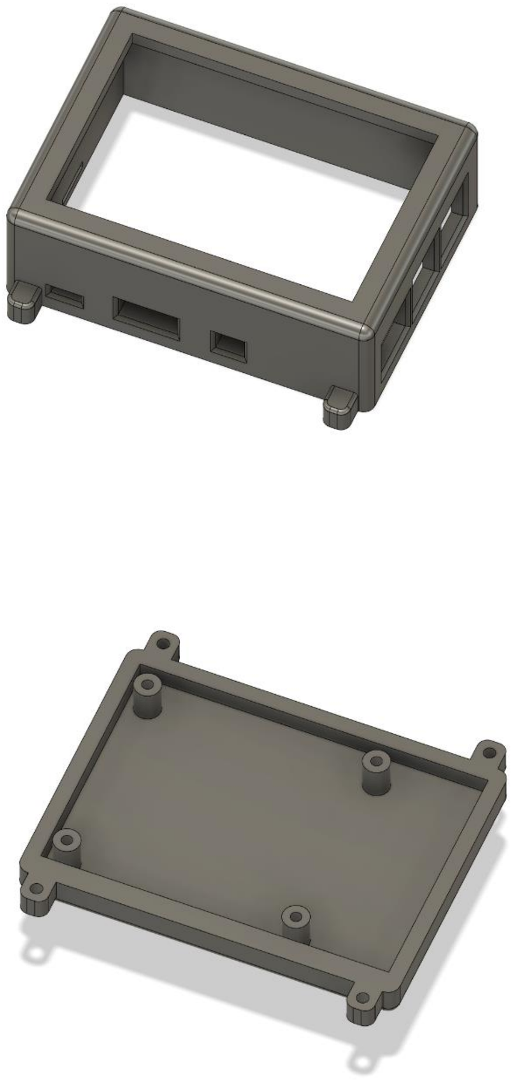

Figure 6.4.1.1 The 3D printed camera mount cover

Step 3 (camera mount): If using the camera mount case, attach the Raspberry Pi to the bottom cover screw mounts using four M3 5mm screws. The screw in locations are pointed out with red arrows:

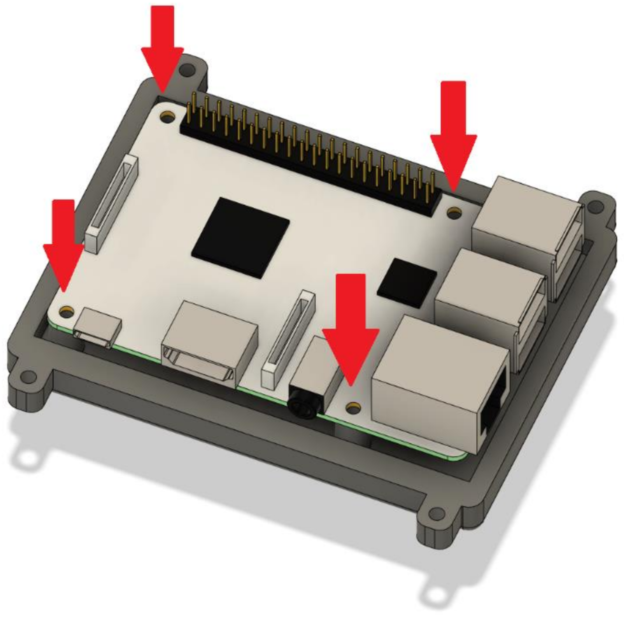

Figure 6.4.1.2 Mounting the Rasberry Pi to the bottom cover

Step 4 (camera mount): Next the screen was lined up with the Raspberry Pi and placed atop. The pin alignment starts at the end closest to the micro-SD card, or pin 1, and rests on top of the Raspberry Pi.

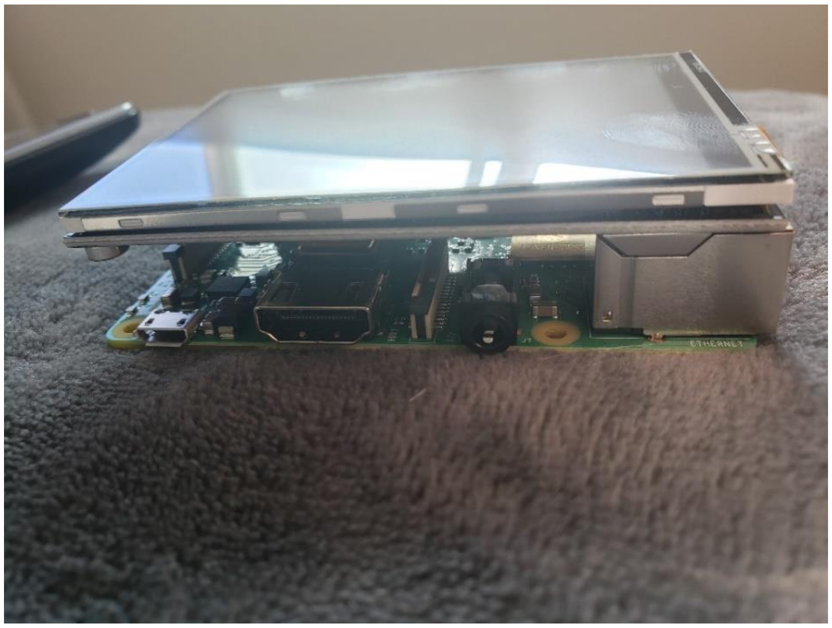

Figure 6.4.1.3 Mounting the screen

Step 5 (camera mount): Finally place the top cover over the Rasbery Pi and secure to the bottom cover using four M3 8mm screws. The M3 8mm screw in locations are shown below:

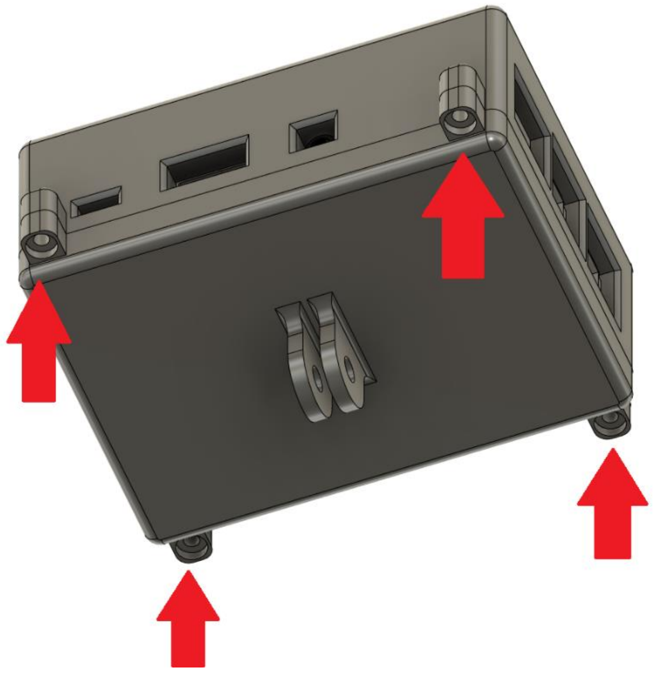

Figure 6.4.1.4 Securing the case

Assembled TalkBox with 3D printed cover:

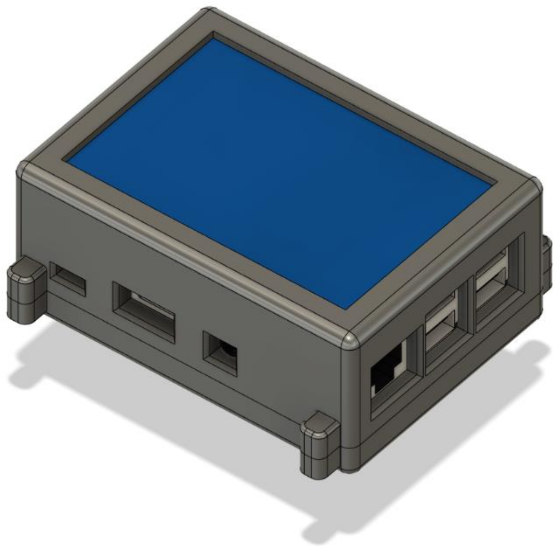

Figure 6.4.1.5 Assembled case

6.4.2 Default cover:

Step 2 (default mount): If NOT using camera mount the screen is lined up with the Raspberry Pi and placed atop. The pin alignment starts at the end closest to the micro-SD card, or pin 1, and rests on top of the Raspberry Pi.

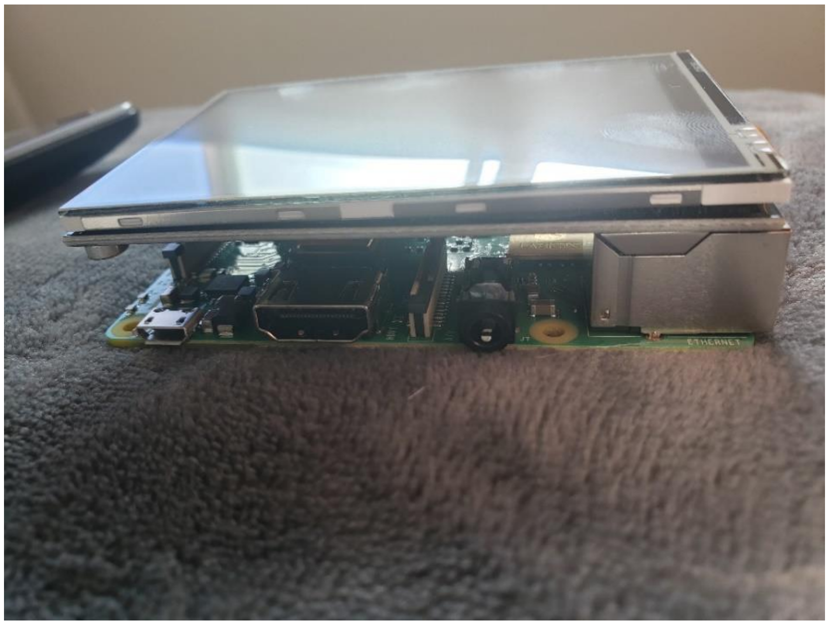

Figure 6.4.2.1 Mounting the screen

Step 3 (default mount): If NOT using camera mount, the case is simply placed on and around the Raspberry Pi. It leaves ports and sections for the associated inputs.

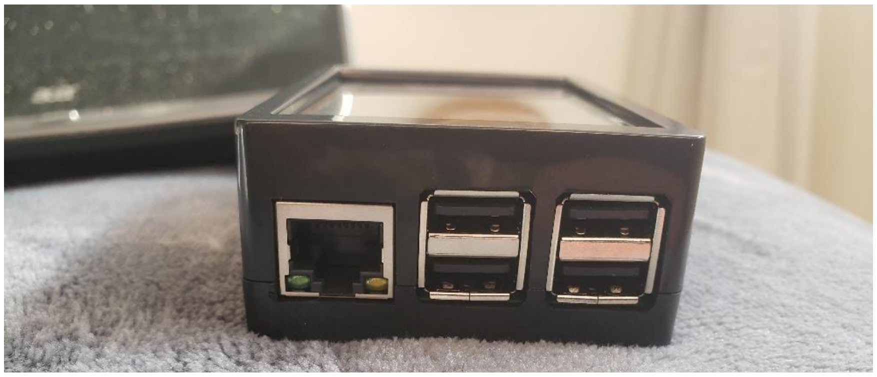
Figure 6.4.2.2 Assembled case

SOFTWARE:

Next, the power cord was attached into the micro-USB port. This is the only way to power the TalkBox. As you can see below, the LCD screen and Raspberry Pi are well contained within the case. To make sure the TalkBox will work, all packages in the code MUST be installed. This includes, but is not limited to:

- Guizero
- Pygame
- Pandas
- OS

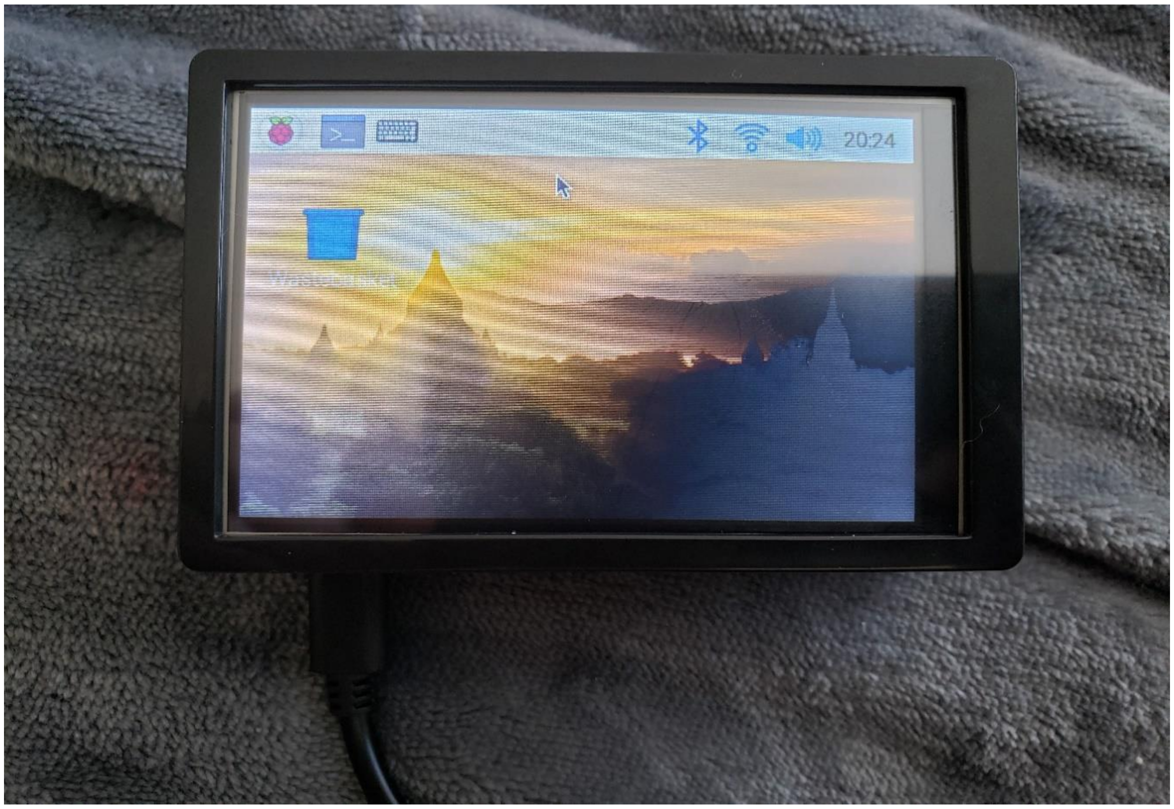

Figure 6.4-2 Raspberry Pi in Case with Power Cord

The next step is importing the code to the Raspberry Pi. This included tinkering around with the command line, downloading any missing packages. Please ensure that when you are installing that you are specifically downloading packages for python 3. After the program works on the device, the file was turned into an executable, and the xls file for the TalkBox was uploaded to the Desktop of the Raspberry Pi. Finally, the speaker was connect to the aux port, and the joystick was connected through the USB port.

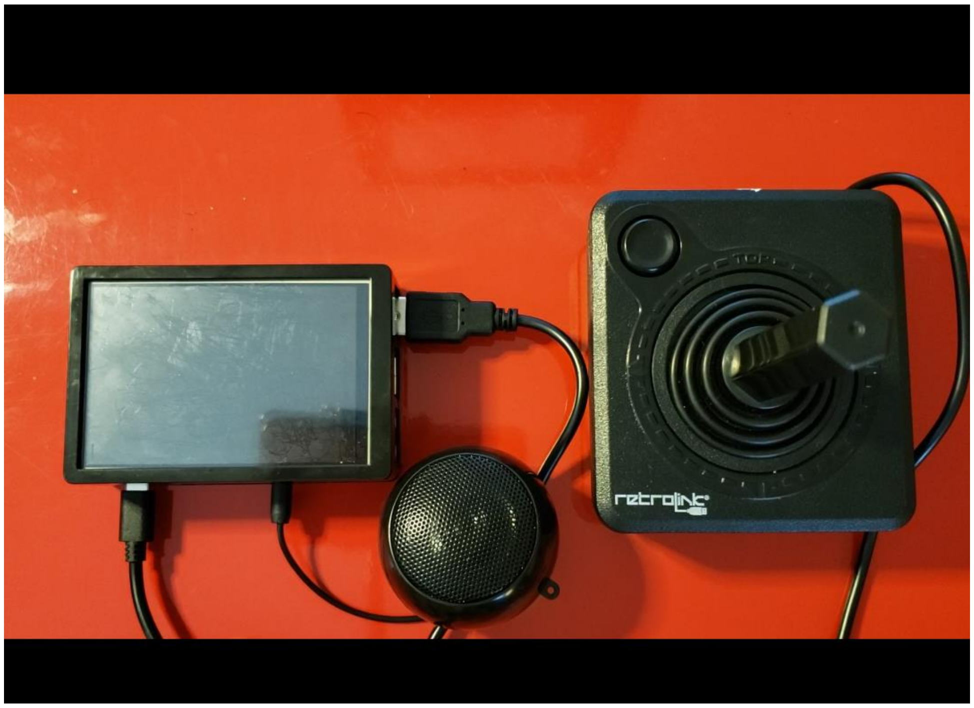

Figure 6.4-3 Final Assembly

The final step is attaching the product to a wheelchair. Connect the power supply to the battery using a USB or XLR cable and place the TalkBox in the gooseneck holder. Apply the adhesive pad to the joystick and place in the best suited location. This concludes the assembly of the TalkBox. If you have any further questions, please do not hesitate to reach out to vjanc090@uottawa.ca.

6.5 Testing and Validation

Table 4 Testing Essentials

#	Metric Tested	Ideal Value	Actual Value
1	Joystick Response Time	Very Fast	1s <
2	Languages available to user	English/French	English

3	Temperature of Raspberry Pi	15-30°C	0-30°C
4	Wire Length of Joystick	10ft	10ft
5	Storage Capabilities of SD Card	>10GB	32GB
6	Weight of product	>1lb	>1lb
7	Volume	~60db	0 - ~80db
8	Communicates with Smart Home Hub	Yes	Yes

7 Conclusions and Recommendations for Future Work

To conclude, this user manual documents the features and functions of the TalkBox, describes their realization and creation, explains the operation and maintenance of the product, and provides the user with recommendations for future developments. As well, this manual identifies issues that were encountered through the iterative design process, and the development of the product and provides advice on how to avoid these issues in usage. The manual also provides information key to the future development of this product.

Lessons learned over the course of this project included: time management skills, project management skills, communication, and teamwork. The most important lesson to be learned in this course, is that teamwork and communication are key factors in the success of a product.

Features that would be beneficial to this product would be waterproofing, an all-encompassing case, more user settings for customization, and a bigger screen. This would have provided the user with more features to mold to their needs and daily lives.

8 Bibliography

Table 5. Referenced Documents

Document Name	Document Location and/or URL	Issuance Date
Maker Repo	https://makerepo.com/vjanc090/898.talkbox	2021-04-07
Guizero	https://lawsie.github.io/guizero/about/	2021-04-07
Python 3	https://www.python.org/download/releases/3.0/	2021-04-07
Pandas	https://pandas.pydata.org/	2021-04-07
Pygame	https://www.pygame.org/docs/ref/joystick.html	2021-04-07
Espeak	http://espeak.sourceforge.net/	2021-04-07
Linux	https://www.raspberrypi.org/documentation/linux/	2021-04-07
Ubuntu	https://ubuntu.com/	2021-04-07
Hard-Drive	https://www.dell.com/support/kbdoc/en-ca/000143677/how-to-create-or-modify-a-partition-in-microsoft-windows	2021-04-07
Screen Driver	https://github.com/goodtft/LCD-show.git	2021-04-07

APPENDICES

APPENDIX I: Design Files

APPENDIX II: Materials Cost Links

XLR to USB:
https://www.amazon.ca/HDE-Balanced-Female-Instruments-Microphones/dp/B0050CEEIW/ref=rtpb_5?pd_rd_w=CmHvO&pf_rd_p=164a4ce0-05d6-485d-91f8-34f2925e3c30&pf_rd_r=0WM9RSYDT1W3GHY3P2M1&pd_rd_r=c4137c51-7fd4-484c-abed-65274692f6f4&pd_rd_wg=6DWoX&pd_rd_i=B0050CEEIW&psc=1

Gooseneck:
https://www.amazon.ca/Lamicall-3546-1/dp/B07S9JXQP2/ref=sr_1_6?dchild=1&keywords=Gooseneck+Phone+Holder&qid=1617807297&sr=8-6

Speaker:
https://www.amazon.ca/gp/product/B07MH1GG2B/ref=ppx_yo_dt_b_asin_title_o02_s00?ie=UTF8&psc=1

Raspberry Pi:
https://www.amazon.ca/gp/product/B07DGFH76Y/ref=ppx_yo_dt_b_asin_title_o03_s00?ie=UTF8&psc=1

LCD Screen:
https://www.amazon.ca/gp/product/B07P3GBWGL/ref=ppx_yo_dt_b_asin_title_o04_s00?ie=UTF8&psc=1

Joystick:
https://acgamesonline.com/products/n-accpc-0055

Adhesive Pads:
https://www.amazon.ca/Boao-Replacement-Compatible-Expanding-Accessory/dp/B07KC735RY/ref=sr_1_6?crid=BP3YH5AEHQDJ&dchild=1&keywords=double+sided+mounting+tape&qid=1617823688&s=electronics&sprefix=double+sided+%2Celectronics%2C173&sr=1-6