Eyevoce

FOR VISUALLY CHALLENGED

INTRODUCTION

Human visual system plays an important role in recognizing information regarding surroundings. Since visual signal provides with more data than auditory information, visual signals are more effective than auditory signals when the human being perceives information.

However, in case of blind people the lack of visual information constrains them in recognizing information. For a blind person to recognize a subject around him depends on the subject to speak something. In addition, even when the subjects speak it is difficult for the blind to recognize the subject.

OBJECTIVE

In this project, we will be making a prototype of a smart eyeglass that can help a visually challenged person to recognize the person in front of him and learn about obstacles ahead. This will be enabled by face recognition and distance detection features.

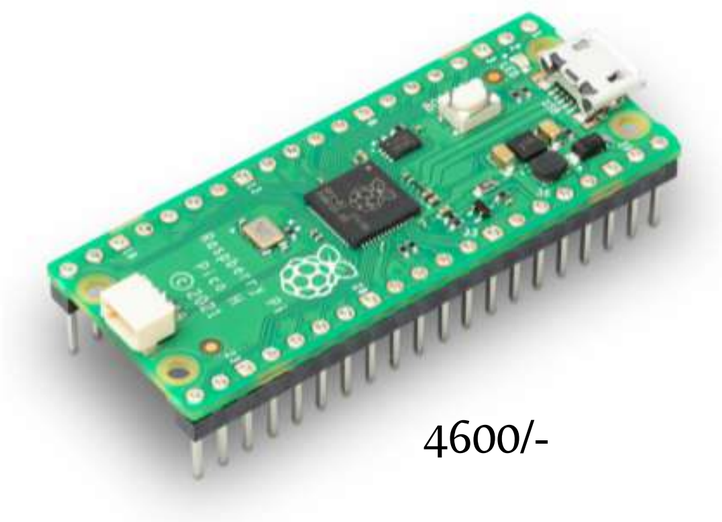
TIMELINE

Collecting the components	- week 1
Hardware implementation and coding	- week 2-4
Experimental trails	- week 5,6
Adjustments and adding features	- week 7
Final model	- week 8

COMPONENTS

- Raspberry Pico
- Ultrasonic Sensor
- Voice Module
- USB Cable
- Jumper Wires
- Battery 5V/Powerbank 5V

1.Raspberry Pico



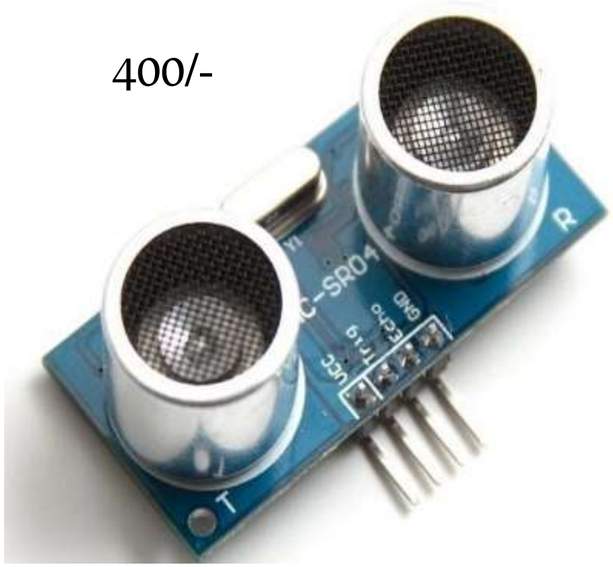
4600/-

The Pico is a low cost, high performance microcontroller board built around the Raspberry Pi RP2040 chip. The Pico features flexible digital interfaces and can be easily programmed over USB using C/C++ or MicroPython, thanks to a comprehensive SDK with software examples and full documentation.

2.Ultrasonic Sensor

HC-SR04 Ultrasonic Sensor has two eyes like projects in the front which forms the transmitter and Receiver. The ultrasonic sensor uses sonar to determine the distance to an object like bats or dolphins do. This module is a transmitter, a receiver, and a control circuit in one single pack!! It has very handy and compact construction. It offers excellent range accuracy and stable readings in an easy-to-use package.

400/-



3.Voice Module

250/-

The Recording Module Voice Board is the real easy way to add Voice Recording (and Playback) to your project. The Module can be operated directly by using the 3 Push-Buttons or with every microcontroller (ex. Arduino). A microphone is implemented directly on the board, and you can connect any 8 Ohm Speaker. Your recordings are saved even without power due to the non-volatile storage on the recording module.

4.Jumper Wires

A jump wire (also known as jumper, jumper wire, DuPont wire) is an electrical wire, or group of them in a cable, with a connector or pin at each end (or sometimes without them – simply "tinned"), which is normally used to interconnect the components of a breadboard or other prototype or test circuit, internally or with other equipment or components, without soldering.

5.USB Cable

The term USB stands for "Universal Serial Bus". USB cable assemblies are some of the most popular cable types available, used mostly to connect computers to peripheral devices such as cameras, camcorders, printers, scanners, and more.

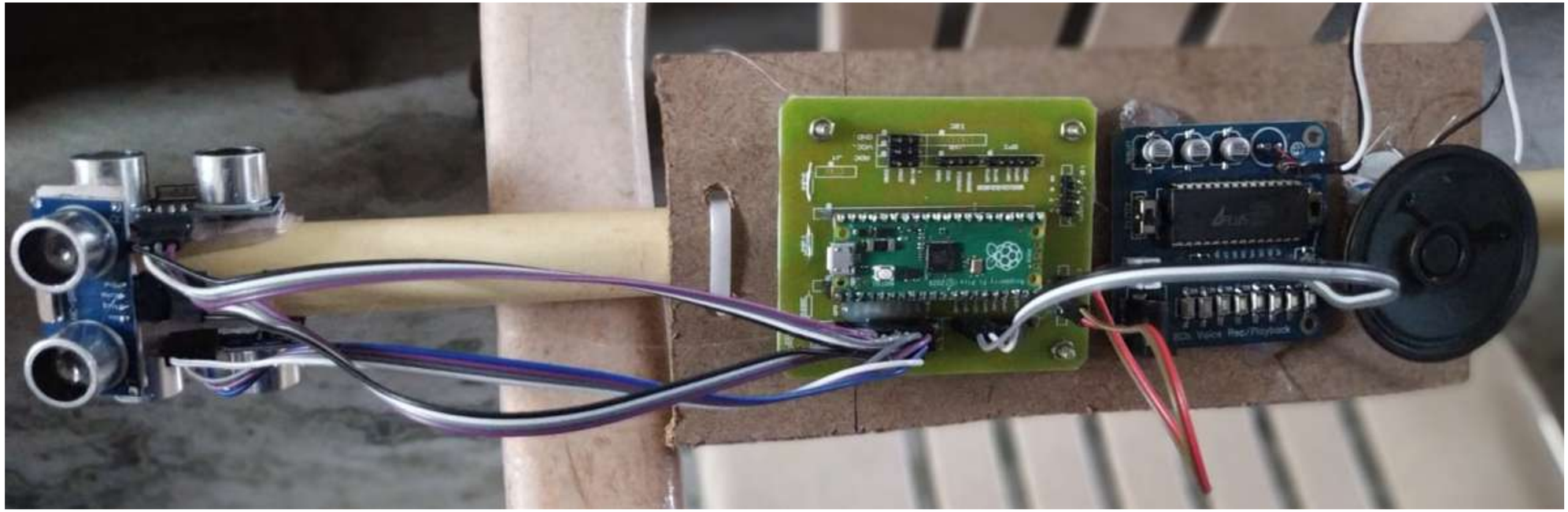
CODE

```
from machine import Pin,PWM #importing PIN and PWM
import time #importing time
import utime
# Defining motor pins
v1=Pin(10,Pin.OUT)
v2=Pin(11,Pin.OUT)
v3=Pin(12,Pin.OUT)
v1.value(0)
v2.value(0)
v3.value(0)
# Defining  Trigger and Echo pins
trigger = Pin(3, Pin.OUT)
echo = Pin(2, Pin.IN)
trigger1 = Pin(5, Pin.OUT)
echo1 = Pin(4, Pin.IN)
trigger2 = Pin(7, Pin.OUT)
echo2 = Pin(6, Pin.IN)

# Defining function to get distance from ultrasonic sensor
def get_distance():
    trigger.low()
    utime.sleep_us(2)
    trigger.high()
    utime.sleep_us(5)
    trigger.low()
    while echo.value() == 0:
        signaloff = utime.ticks_us()
    while echo.value() == 1:
        signalon = utime.ticks_us()
    timepassed = signalon - signaloff
    dist = (timepassed * 0.0343) / 2
    return dist
```

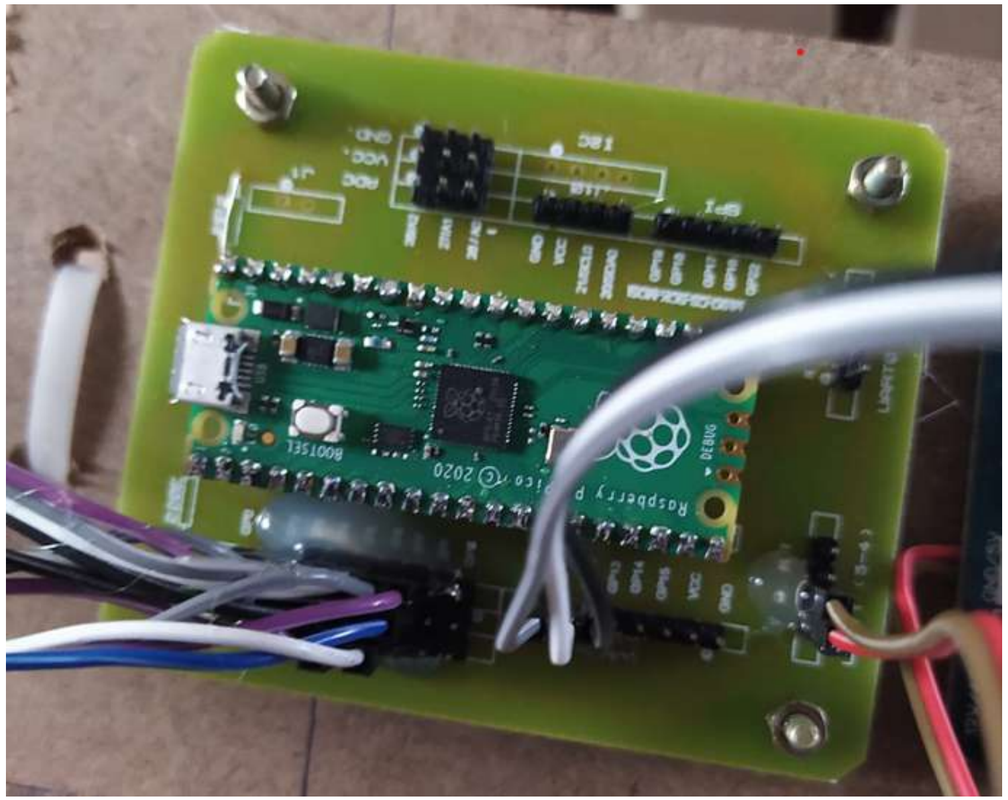
```
def get_distance1():
    trigger1.low()
    utime.sleep_us(2)
    trigger1.high()
    utime.sleep_us(5)
    trigger1.low()
    while echo1.value() == 0:
        signaloff = utime.ticks_us()
    while echo1.value() == 1:
        signalon = utime.ticks_us()
    timepassed = signalon - signaloff
    dist1= (timepassed * 0.0343) / 2
    return dist1
def get_distance2():
    trigger2.low()
    utime.sleep_us(2)
    trigger2.high()
    utime.sleep_us(5)
    trigger2.low()
    while echo2.value() == 0:
        signaloff = utime.ticks_us()
    while echo2.value() == 1:
        signalon = utime.ticks_us()
    timepassed = signalon - signaloff
    dist2= (timepassed * 0.0343) / 2
    return dist2
```

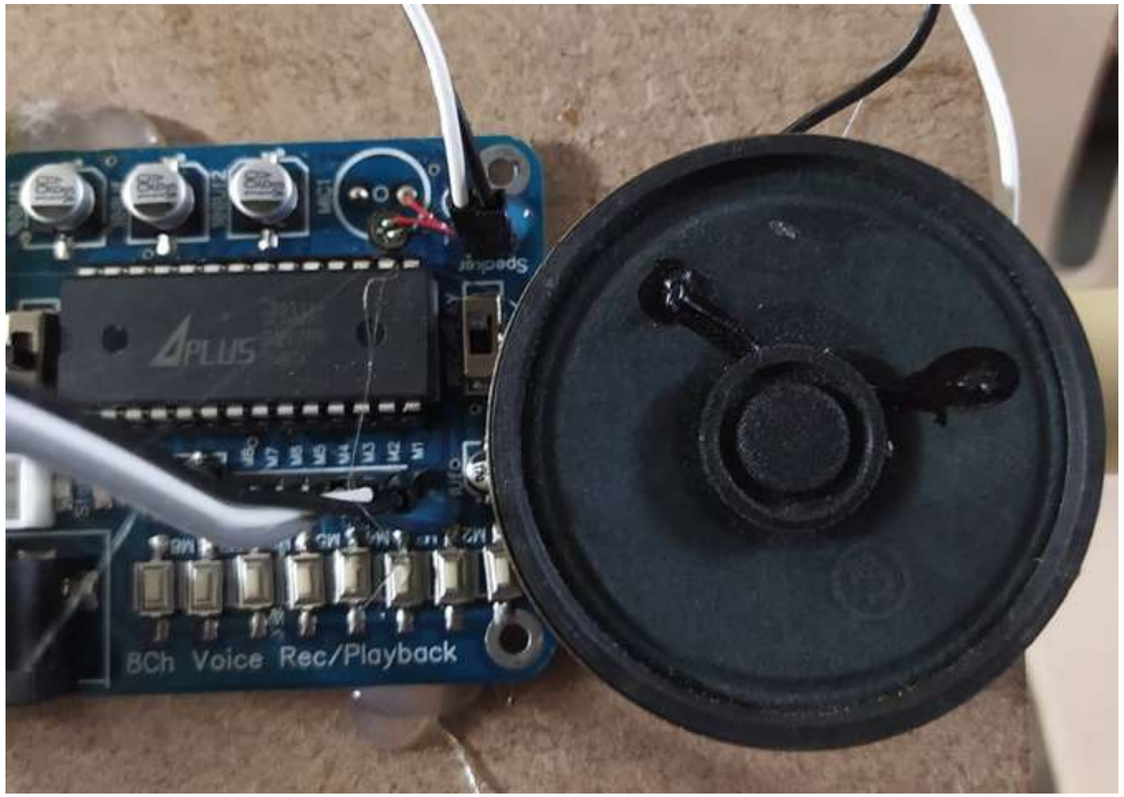
```
while True:
    distance=get_distance() #Getting distance in cm
    print(distance)
    distance1=get_distance1() #Getting distance in cm
    print(distance1)
    distance2= get_distance2() #Getting distance in cm
    print(distance2)

    #Defining direction based on conditions
    if distance < 15:
        v1.value(1)
        time.sleep(2)
        v1.value(0)

    else:
         v1.value(0)

    if distance1< 15:
         v2.value(1)
         time.sleep(2)
         v2.value(0)

    else:
        v2.value(0)
        time.sleep(0.2)

    if distance2< 15:
        v3.value(1)
        time.sleep(2)
        v3.value(0)
    else:
        v3.value(0)
        time.sleep(0.2)
```

WORKING MODEL



WORKING MODEL





COST

COMPONENTS	AMOUNT
Raspberry Pico	4600/-
Battery 5V	200/-
Ultra Sonic Sensor	400/-
USB Cable	180/-
Jumper Wires	100/-
Voice Module	250/-
GRAND TOTAL:	5730/-