SMART BLIND MAN'S WALKING STICK

KANCHON GHARAMI

A project paper
submitted in partial fulfillment of the
requirements for the degree of

Bachelor of Science
with Departmental Honors

Dept. of Computer Science & Engineering
Rajshahi University of Engineering & technology
August 2019

Supervisor: Dr. Mir Md. Jahangir Kabir

Approved by: ______________________________

(Dr. Mir Md. Jahangir Kabir)

Presentation of work given on August 5th, 2019.

To everyone who has help me to fulfill this project,
with gratitude.

ABSTRACT

Background: It is hard for a blind man to walk freely everywhere without help of others. Obstacles, stairs, water, slob, etc. can become a big problem for a blind man to go with any new path. With the help of modern technology and sensors, we can build a smart blind man stick which can remove all these trouble of walking and with this technology blind man get the ability back to walk in a busy path easily.

This smart blind man's walking stick is working with background Arduino command and measure obstacles or banister with help for total 2 sonar system and detect slob with moisturizer sensor. When the smart stick found obstacles or other any which need to informed to the blind man, the stick will inform its owner with sound bip of buzzer's sound. If the blind man fallen down or the stick fallen down, then it starts alarm loudly to help its owner by informing other people.

Methods: This smart blind man stick detect front obstacles with help of front faced sonar which send a high frequency sound and receive echo of that sound came back from impacted with obstacles; the Arduino in pre-programmed to receive the time interval of sound send and echo's came back and with this interval we can easily determine the distance (here we know the sound speed) of front's object. If the front object's distance is enough safe (customized as blind man wish) for walking blind man through it then the stick doesn't do anything but at the moment it gets object inside it's defined safe distance it will send alert and ring the buzzer.

This stick detects stairs with the same distance measurement technique. Here I set a sonar faced to downward; when it doesn't get any surface within 1.2 meter (can be customize as blind man wish), this mean we have a stairs or holes in front, the Arduino will have sent command to ring the buzzer.

The water detection part fully based on sensors sensitivity. It gives us value of taken object's moisturizer value belong to range from 0~1024. Here I want to set a sensitive system checking with 1000 boarder value. when it gets moisturizer value below 1000 it sends signal to ring the buzzer.

Results: The project can be used as a very useful and smart blind man stick. This system has following features:

i. Ability to detect obstacles and inform to blind man about that.
ii. Ability to play siren for wanting help when the blind man fallen down by any accident.
iii. Ability to detect water or slob which is risky for blind man's walking.
iv. It can help blind man to cross risky area or crowd place.

Conclusion: We estimate that this project could create a cost-effective budget with huge additional features in the future modified version; Which is very optimistic for all the blind people facing problems in their daily lives.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1

INTRODUCTION

1.1 Summary and Motivations

It is hard for a blind man to walk freely everywhere without help of others. Obstacles, stairs, water, slob, etc. can become a big problem for a blind man to go with any new path. With the help of modern technology and sensors, we can build a smart blind man stick which can remove all these trouble of walking and with this technology blind man get the ability back to walk in a busy path easily.
This smart blind man's walking stick is working with background Arduino command and measure obstacles or banister with help for total 2 sonar system and detect slob with moisturizer sensor. When the smart stick found obstacles or other any which need to informed to the blind man, the stick will inform its owner with sound bip of buzzer's sound. If the blind man fallen down or the stick fallen down, then it starts alarm loudly to help its owner by informing other people.

1.2 Problem

For this project, mainly we need to face 2 problems; how to measure distance of any object and how to measure moisture value of any object/surface. If we can determine distance of any object, then we can easily determine the front's object stay in a safe distance or not. And if we can determine that the moisture of the front surface of walking then we can determine that the next step surface is dry and clean enough for walk or not.

1.3 Solution

We can solve the problem with help of ultrasonic sensor (sonar) and moisture sensor. The ultrasonic sensor sends high frequencies sound by giving signal in its trig pin and when it receives/get the echo of that sound then it give output signal by echo pin. With calculating the time interval of sound sent and echo receive we can determine the distance by following equation-

$$2d = vt \quad \ldots\ldots\ldots\ldots\ldots\ldots \text{i}$$

$$\text{or, } d = vt/2 \quad \ldots\ldots\ldots\ldots \text{ii}$$

Here, d = distance of front object from ultrasound sensor.
v = speed of sound at that time (we can consider it $346ms^{-1}$ in $25^{o}C$)
t = time interval (we get it from help of sonar)
When we get $d > 2$ meter then we say it as safe distance else it go as close distance which means need to ring alarm.
The moisture value of surface determines by moisturizer sensor and actually it is very easy. The moisture sensor can give analog input which range $R_f = [0,1024]$. The value 1024 indicates fully dry and 0 indicate fully wet. We set the alarm range border value at 1000 as safe zone cause lower then this value the surface start to became wet slowly.
By this basic principle the full project can be make.

1.4 Overview

This smart blind man's walking stick is working with background Arduino command and measure obstacles or banister with help for total 3 sonar system and detect slob with moisturizer sensor. When the smart stick found obstacles or other any which need to informed to blind man, the stick will inform its owner with vibration or voice speech. If the blind man fallen down or the stick fallen down, then it starts alarm loudly to help its owner by informing other people.

This system has following features:

a. Ability to detect obstacles and inform to blind man about that.

b. Ability to play siren for wanting help when the blind man fallen down by any accident.

c. Ability to detect water or slob which is risky for blind man's walking.

d. It can help blind man to cross risky area or crowd place.

Using a properly commanded Arduino microcontroller the system can find obstacles with help of sonar. The sonar receive soundwave reflected from obstacles and calculate its distance. Mainly with this distance the whole project works. We estimate that this project could create a cost-effective budget with huge additional features in the future modified version; Which is very optimistic for all the blind people facing problems in their daily lives.

2

EQUIPMENT'S OVERVIEW

2.1 Equipment's List

In this project we use following equipment-

a.	Arduino Uno R3	1 piece
b.	ultrasonic sensor (Sonar) HC-SR04	2 pieces
c.	Moisturizer sensor	1 piece
d.	Moisturizer sensor module	1 piece
e.	9-volt battery	1 piece
f.	Buzzer	3 pieces
g.	Switch	1 pieces
h.	LED	
i.	Jumper wires	
j.	Old man sticks (as main base)	

Beside these we need following tools too-

a. Soldering Iron
b. Glue gun
c. Cutter
d. Connection Wires
e. Bread board / Project board
f. Rung, glue stick, tapes etc.

With this equipment and help of these tools we can make our smart blind man sticks. The main equipment's description in included below.

2.2 The Arduino Uno

Arduino Uno is a microcontroller board based on the ATmega328P (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. You can tinker with your UNO without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again.

Figure 2.1: Arduino Uno R3 and USB cables

"Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform; for an extensive list of current, past or outdated boards see the Arduino index of boards.

Pins of Arduino UNO

General pin functions:
LED: There is a built-in LED driven by digital pin 13. When the pin is high value, the LED is on, when the pin is low, it's off.
VIN: The input voltage to the Arduino/Genuino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
5V: This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 20V), the USB connector (5V), or the VIN pin of the board (7-20V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board.
3V3: A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
GND: Ground pins.
IOREF: This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.
Reset: Typically used to add a reset button to shields which block the one on the board.
Special pin functions:
Each of the 14 digital pins and 6 analog pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller. The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function.[7]

In addition, some pins have specialized functions:

Serial / UART: pins 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL serial chip.
External interrupts: pins 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.
PWM (pulse-width modulation): 3, 5, 6, 9, 10, and 11. Can provide 8-bit PWM output with the analogWrite() function.

SPI (Serial Peripheral Interface): 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
TWI (two-wire interface) / I²C: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.
AREF (analog reference): Reference voltage for the analog inputs.

2.3 Ultrasonic Sensor (Sonar)

An ultrasonic sensor is an instrument that measures the distance to an object using ultrasonic sound waves. An ultrasonic sensor uses a transducer to send and receive ultrasonic pulses that relay back information about an object's proximity.
Ultrasonic ranging module HC - SR04 provides 2cm ~ 700cm non-contact measurement function, the ranging accuracy can reach to 3mm.Ensured stable signal within 5m, gradually faded signal outside 5m till disappearing at 7m position. The HC-SR04 ultrasonic sensor uses sonar to determine distance to an object like bats do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package. It comes complete with ultrasonic transmitter and receiver modules.

Ultrasonic Sensors are best used in the non-contact detection of:

- ✓ Presence
- ✓ Level
- ✓ Position
- ✓ Distance

Non-contact sensors are also referred to as proximity sensors.

Figure 2.2: Ultrasonic ranging module (Sonar) HC - SR04

Pin Function of ultrasonic sensor-

i. TRIG: Trigger Pulse Input
ii. ECHO: Echo Pulse Output
iii. GND: Ground
iv. VCC: 5V Supply

Table 2.1: Ultrasonic sensor features table

Working Voltage	DC5V
Working Current	16mA
Working Frequency	40Hz
Max Range	2cm ~ 700cm
Min Range	2cm
Trigger Input Signal	10uS TTL pulse
Echo Output Signal	Input TTL lever signal and the range in proportion
Dimension	46x20.5x15 mm

The ultrasonic sensor uses sonar to determine the distance to an object. Here's what happens:
The transmitter (trig pin) sends a signal: a high-frequency sound.
When the signal finds an object, it is reflected and…… the transmitter (echo pin) receives it.

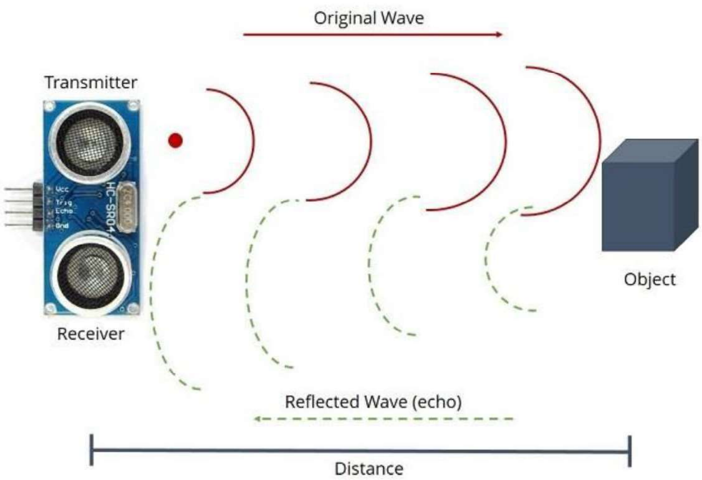

Figure 2.3 – Ultrasonic sensor (sonar) working principle

The time between the transmission and reception of the signal allows us to calculate the distance to an object. This is possible because we know the sound's velocity in the air.

2.4 Moisturizer sensor

Soil moisture sensors measure the volumetric water content in soil. Since the direct gravimetric measurement of free soil moisture requires removing, drying, and weighting of a sample, soil moisture sensors measure the volumetric water content indirectly by using some other property of the soil, such as electrical resistance, dielectric constant, or interaction with neutrons, as a proxy for the moisture content.
The relation between the measured property and soil moisture must be calibrated and may vary depending on environmental factors such as soil type, temperature, or electric conductivity. Reflected microwave radiation is affected by the soil moisture and is used for remote sensing in hydrology and agriculture. Portable probe instruments can be used by farmers or gardeners.

Figure 2.4 - Soil moisture sensor

The soil moisture sensor consists of two probes which are used to measure the volumetric content of water. The two probes allow the current to pass

through the soil and then it gets the resistance value to measure the moisture value.
When there is more water, the soil will conduct more electricity which means that there will be less resistance. Therefore, the moisture level will be higher. Dry soil conducts electricity poorly, so when there will be less water, then the soil will conduct less electricity which means that there will be more resistance. Therefore, the moisture level will be lower.

Specifications:
Table 2.2: The specifications of the soil moisture sensor FC-28 are as follows

Input Voltage	*3.3 – 5V*
Output Voltage	0 – 4.2V
Input Current	35mA
Output Signal	Both Analog and Digital

A Pin Out – Soil Moisture Sensor: The soil Moisture sensor FC-28 has four pins

i. VCC: For power
ii. A0: Analog output
iii. D0: Digital output
iv. GND: Ground

The Module also contains a potentiometer which will set the threshold value and then this threshold value will be compared by the LM393 comparator. The output LED will light up and down according to this threshold value.

2.5 Others Equipment

In this project we need some other product too, such as-

a. Buzzer
b. LED
c. 9-volt battery
d. Switch
e. Jumper wires

Buzzer

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric (piezo for short). Typical uses of buzzers and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke.

Figure 2.5 - Buzzer

LED

A light-emitting diode (LED) is a semiconductor light source that emits light when current flows through it. Electrons in the semiconductor recombine with electron holes, releasing energy in the form of photons.

Figure 2.6 - LED

Nine-Volt Battery

The nine-volt battery format is commonly available in primary carbon-zinc and alkaline chemistry, in primary lithium iron disulfide, and in rechargeable form in nickel-cadmium, nickel-metal hydride and lithium-ion. Mercury-oxide batteries of this format, once common, have not been manufactured in many years due to their mercury content.

Figure 2.7 – 9V battery

Jumper Wire

A jump wire (also known as jumper wire, or jumper) 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.

Figure 2.8 – Jumper Wire

Switch

In electrical engineering, a switch is an electrical component that can "make" or "break" an electrical circuit, interrupting the current or diverting it from one conductor to another. The mechanism of a switch removes or restores the conducting path in a circuit when it is operated.

Figure 2.9 – Switch

Glue Gun

Hot melt adhesive (HMA), also known as hot glue, is a form of thermoplastic adhesive that is commonly sold as solid cylindrical sticks of various diameters designed to be applied using a hot glue gun.

Figure 2.10 – Glue Gun

Soldering Iron

A soldering iron is composed of a heated metal tip and an insulated handle. Heating is often achieved electrically, by passing an electric current through a resistive heating element. Cordless irons can be heated by combustion of gas stored in a small tank, often using a catalytic heater rather than a flame. Simple irons less commonly used today than in the past were simply a large copper bit on a handle, heated in a flame.

Figure 2.11 – soldering iron

3

HARDWARES ASSEMBLING

Here we need to assemble 2 sonars and 1 moisturizer sensor, 3 buzzers and a battery with microcontroller Arduino Uno. The connections are given below.

The front distance measuring ultrasonic sensor (sonar) defined as soner1 in here. The pin connection with Arduino in below-

Table 3.1

Soner1 pin	Arduino pin
Vcc	13
Trig	12
Echo	11
Gnd	10

When we need to ring alarm by getting alert from soner1 we ring a buzzer named Buzz. It's assembling pin with Arduino in below-

Table 3.2

Buzz pin	Arduino pin
Vcc	3
Gnd	2

The downward distance measuring ultrasonic sensor (sonar) defined as soner2 in here. The pin connection with Arduino in below-

Table 3.3

Soner2 pin	Arduino pin
Vcc	7
Trig	6
Echo	5
Gnd	4

When we need to ring alarm by getting alert from soner2 we ring a buzzer named Buzz2. It's assembling pin with Arduino in below-

Table 3.4

Buzz2 pin	Arduino pin
Vcc	A1
Gnd	A0

Here I set the moisturizer module and moisturizer probes connection as below-

Table 3.5

Moisturizer Module	*Arduino*
AI (Analog Input)	A2
Vcc	A4
Gnd	A3

Moisturizer Probes	*Moisturizer Module*
+ ve	positive
(-ve)	Negative

When we need to ring alarm by getting alert from moisturizer module we ring a buzzer named Buzz3. It's assembling pin with Arduino in below-

Table 3.6

Buzz2 pin	Arduino pin
Vcc	8
Gnd	9

Finally, the battery is connected with Arduino power jack by passing with a switch. The moisturizer module and probes distance is long so we use extra chinese joined cable with jumper.

4

CODING AND UPLOADING IN ARDUINO UNO

4.1 Algorithm Making

The algorithms of the full project is derives in below-

The pseudo code of main loop-

1. Start
2. Sonar1_dist = front sonar distance
3. If sonar1_dist < 70
    a. Buzz_vcc power up:
    b. For i in range sonar1_dist to 0
        i. delay 10 ms
        ii. Buzz_vcc power low
    c. For i in range sonar1_dist to 0
        i. delay 10 ms
    d. buzz_vcc power low
4. Sonar2_dist = downward sonar distance
5. If sonar2_dist < 110
    a. Buzz2_vcc power up
    b. Buzz2_gnd power low
    c. Delay 500ms
6. msvalue = analogRead of Moisturizer Module;
7. flag = false
8. if msvalue >= 900 and flag = false then
    a. Buzz3_vcc power low
    b. Flag = true
    c. Delay 1s
9. if msvalue <= 900 and flag = false then
    a. Buzz3_vcc power high
    b. Flag = high
    c. Delay 2s
10. Delay 1s
11. Continue loop

4.2 Algorithm Implementation in Code

The smart blind man stick algorithm implementation can be start with basic pin declaration for our sensors and modules. Here we use Arduino sdk.

Front faced ultrasonic sensor's pin can declare like below-

```
//soner 1 declear
const int soner1_VccPin = 13;
const int soner1_trigger = 12;
const int soner1_echo = 11;
const int soner1_GroundPin = 10;
```

downward faced ultrasonic sensor's pin can declare like below-

```
//soner 2 declear
const int soner2_VccPin = 7;
const int soner2_trigger = 6;
const int soner2_echo = 5;
const int soner2_GroundPin = 4;
```

moisturizer module pin can declare like below-

```
int msensor = A2; // moisture sensor is connected with
the analog pin A1 of the Arduino
int msensor_vcc = A4;
int msensor_gnd = A3;
int msvalue = 0; // moisture sensor value
boolean flag = false;
```

Buzzers are connected like below pins-

```
//Buzz declear
const int Buzz_vcc = 3;        // for soner 1
const int Buzz_gnd = 2;
const int Buzz2_vcc = A1;      // for soner 2
const int Buzz2_gnd = A0;
const int Buzz3_vcc = 8;      // for moscirisor
const int Buzz3_gnd = 9;
```

Extra integer used in code are declare like below-

```
int soner1_dist;
int soner1_time_taken;
int soner2_dist;
int soner2_time_taken;
```

our pin mood setup can be done like below-

```
void setup() {
  Serial.begin(9600);

  // Buzzer bip power
  pinMode(Buzz_vcc,OUTPUT);
  digitalWrite(Buzz_vcc,LOW);
  pinMode(Buzz_gnd,OUTPUT);
  digitalWrite(Buzz_gnd,LOW);

  pinMode(Buzz2_vcc,OUTPUT);
  digitalWrite(Buzz2_vcc,LOW);
  pinMode(Buzz2_gnd,OUTPUT);
  digitalWrite(Buzz2_gnd,LOW);

  pinMode(Buzz3_vcc,OUTPUT);
  digitalWrite(Buzz3_vcc,LOW);
  pinMode(Buzz3_gnd,OUTPUT);
  digitalWrite(Buzz3_gnd,LOW);

  //Soner 1 setup
  pinMode(soner1_VccPin, OUTPUT) ;
  digitalWrite(soner1_VccPin, HIGH) ;
  pinMode(soner1_echo, INPUT);
pinMode(soner1_trigger, OUTPUT);
  pinMode(soner1_GroundPin, OUTPUT) ;
  digitalWrite(soner1_GroundPin,LOW) ;

  //Soner 2 setup
  pinMode(soner2_VccPin, OUTPUT) ;
  digitalWrite(soner2_VccPin, HIGH) ;
  pinMode(soner2_echo, INPUT);
  pinMode(soner2_trigger, OUTPUT);
  pinMode(soner2_GroundPin, OUTPUT) ;
  digitalWrite(soner2_GroundPin,LOW) ;

  // moisture sensor setup
  pinMode(msensor, INPUT);
  pinMode(msensor_vcc, OUTPUT) ;
  digitalWrite(msensor_vcc, HIGH) ;
  pinMode(msensor_gnd, OUTPUT) ;
  digitalWrite(msensor_gnd,LOW) ;
  pinMode(Buzz3_vcc, OUTPUT);
}
```

Now we need to declare our distance measuring functions with help of sonar module-

```
void soner1_calculate_distance(int soner1_trigger, int
soner1_echo)
{
  digitalWrite(soner1_trigger, LOW);
```

```
  delayMicroseconds(2);
  digitalWrite(soner1_trigger, HIGH);
  delayMicroseconds(10);
  digitalWrite(soner1_trigger, LOW);

  soner1_time_taken = pulseIn(soner1_echo, HIGH);
  soner1_dist= soner1_time_taken*0.034/2;
  if (soner1_dist>300)
  soner1_dist=300;
}

void soner2_calculate_distance(int soner2_trigger, int
soner2_echo)
{
  digitalWrite(soner2_trigger, LOW);
  delayMicroseconds(2);
  digitalWrite(soner2_trigger, HIGH);
  delayMicroseconds(10);
  digitalWrite(soner2_trigger, LOW);

  soner2_time_taken = pulseIn(soner2_echo, HIGH);
  soner2_dist= soner2_time_taken*0.034/2;
  //if (soner2_dist>300)
  //soner1_dist=300;
}
```

And finally our main loop part is below-

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

  //front obstrucle decteting with soner 1

soner1_calculate_distance(soner1_trigger,soner1_echo);
  if (soner1_dist<70)
  {
    //Serial.print(soner1_dist);
    //Serial.println("\nObject Alert");
    digitalWrite(Buzz_vcc,HIGH);
    for (int i=soner1_dist; i>0; i--)
      delay(10);
    digitalWrite(Buzz_vcc,LOW);
    for (int i=soner1_dist; i>0; i--)
      delay(10);
  }

  //gap in next step detecting with sonet 2

soner2_calculate_distance(soner2_trigger,soner2_echo);
  digitalWrite(Buzz2_vcc,LOW);
  if (soner2_dist>110)
  {
```

```
      //Serial.print(soner2_dist);
      //Serial.println("\ngap found");
      digitalWrite(Buzz2_vcc,HIGH);
      digitalWrite(Buzz2_gnd,LOW);
      delay(500);
   }

   // mostarisor sensor part
   msvalue = analogRead(msensor);
   digitalWrite(Buzz3_vcc, LOW);
   Serial.println(msvalue);
     if ( (msvalue >= 900  ) && ( flag == false ) )
   {
     digitalWrite(Buzz3_vcc, LOW);
     flag = true;
     delay(1000);
   }
     if ( (msvalue < 900  ) && ( flag == true ) )
   {
     digitalWrite(Buzz3_vcc, HIGH);
     flag = false;
     delay(2000);
   }
     delay(100);

}
```

4.3 Full Code Overview

The final code which in uploaded to Arduino is-

```
//soner 1 declear
const int soner1_VccPin = 13;
const int soner1_trigger = 12;
const int soner1_echo = 11;
const int soner1_GroundPin = 10;

//soner 2 declear
const int soner2_VccPin = 7;
const int soner2_trigger = 6;
const int soner2_echo = 5;
const int soner2_GroundPin = 4;

//Buzz declear
const int Buzz_vcc = 3;       // for soner 1
const int Buzz_gnd = 2;
const int Buzz2_vcc = A1;     // for soner 2
const int Buzz2_gnd = A0;
const int Buzz3_vcc = 8;     // for moscirisor
const int Buzz3_gnd = 9;

int soner1_dist;
int soner1_time_taken;

int soner2_dist;
int soner2_time_taken;

int msensor = A2; // moisture sensor is connected with
the analog pin A1 of the Arduino
int msensor_vcc = A4;
int msensor_gnd = A3;
int msvalue = 0; // moisture sensor value
boolean flag = false;


void soner1_calculate_distance(int soner1_trigger, int
soner1_echo)
{
  digitalWrite(soner1_trigger, LOW);
  delayMicroseconds(2);
  digitalWrite(soner1_trigger, HIGH);
  delayMicroseconds(10);
  digitalWrite(soner1_trigger, LOW);

  soner1_time_taken = pulseIn(soner1_echo, HIGH);
  soner1_dist= soner1_time_taken*0.034/2;
  if (soner1_dist>300)
  soner1_dist=300;
}
```

```
void soner2_calculate_distance(int soner2_trigger, int
soner2_echo)
{
  digitalWrite(soner2_trigger, LOW);
  delayMicroseconds(2);
  digitalWrite(soner2_trigger, HIGH);
  delayMicroseconds(10);
  digitalWrite(soner2_trigger, LOW);

  soner2_time_taken = pulseIn(soner2_echo, HIGH);
  soner2_dist= soner2_time_taken*0.034/2;
  //if (soner2_dist>300)
  //soner1_dist=300;
}


void setup() {
  Serial.begin(9600);

  // Buzzer bip power
  pinMode(Buzz_vcc,OUTPUT);
  digitalWrite(Buzz_vcc,LOW);
  pinMode(Buzz_gnd,OUTPUT);
  digitalWrite(Buzz_gnd,LOW);

  pinMode(Buzz2_vcc,OUTPUT);
  digitalWrite(Buzz2_vcc,LOW);
  pinMode(Buzz2_gnd,OUTPUT);
  digitalWrite(Buzz2_gnd,LOW);

  pinMode(Buzz3_vcc,OUTPUT);
  digitalWrite(Buzz3_vcc,LOW);
  pinMode(Buzz3_gnd,OUTPUT);
  digitalWrite(Buzz3_gnd,LOW);


  //Soner 1 setup
  pinMode(soner1_VccPin, OUTPUT) ;    //tell pin 13 it
is going to be an output
  digitalWrite(soner1_VccPin, HIGH) ; //tell pin 13 to
output HIGH (+5V)
  pinMode(soner1_echo, INPUT);     //tell pin 11 it is
going to be an input
  pinMode(soner1_trigger, OUTPUT);    //tell pin 12 it
is going to be an output
  pinMode(soner1_GroundPin, OUTPUT) ; //tell pin 10 it
is going to be an output
  digitalWrite(soner1_GroundPin,LOW) ;//tell pin 10 to
output LOW (0V, or ground)


  //Soner 2 setup
  pinMode(soner2_VccPin, OUTPUT) ;    //tell pin 13 it
is going to be an output
```

```
  digitalWrite(soner2_VccPin, HIGH) ; //tell pin 13 to
output HIGH (+5V)
  pinMode(soner2_echo, INPUT);     //tell pin 11 it is
going to be an input
  pinMode(soner2_trigger, OUTPUT);    //tell pin 12 it
is going to be an output
  pinMode(soner2_GroundPin, OUTPUT) ; //tell pin 10 it
is going to be an output
  digitalWrite(soner2_GroundPin,LOW) ;//tell pin 10 to
output LOW (0V, or ground)


  // moisture sensor setup
  pinMode(msensor, INPUT);
  pinMode(msensor_vcc, OUTPUT) ;
  digitalWrite(msensor_vcc, HIGH) ;
  pinMode(msensor_gnd, OUTPUT) ;
  digitalWrite(msensor_gnd,LOW) ;
  pinMode(Buzz3_vcc, OUTPUT);

}

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

  //front obstrucle decteting with soner 1

soner1_calculate_distance(soner1_trigger,soner1_echo);
  if (soner1_dist<70)
  {
    //Serial.print(soner1_dist);
    //Serial.println("\nObject Alert");
    digitalWrite(Buzz_vcc,HIGH);
    for (int i=soner1_dist; i>0; i--)
      delay(10);
    digitalWrite(Buzz_vcc,LOW);
    for (int i=soner1_dist; i>0; i--)
      delay(10);
  }

  //gap in next step detecting with sonet 2

soner2_calculate_distance(soner2_trigger,soner2_echo);
  digitalWrite(Buzz2_vcc,LOW);
  if (soner2_dist>110)
  {
     //Serial.print(soner2_dist);
     //Serial.println("\ngap found");
     digitalWrite(Buzz2_vcc,HIGH);
     digitalWrite(Buzz2_gnd,LOW);
     delay(500);
  }

  // mostarisor sensor part
```

```
  msvalue = analogRead(msensor);
  digitalWrite(Buzz3_vcc, LOW);
  Serial.println(msvalue);
    if ( (msvalue >= 900  ) && ( flag == false ) )
  {
    digitalWrite(Buzz3_vcc, LOW);
    flag = true;
    delay(1000);
  }
    if ( (msvalue < 900  ) && ( flag == true ) )
  {
    digitalWrite(Buzz3_vcc, HIGH);
    flag = false;
    delay(2000);
  }
    delay(100);


}
```

5

CUSTOMIZING FOR ANY PERSON'S DEMAND

5.1 Customizing Distance

In this project we use front border line distance at 70cm in the 2nd line of following part-

```
soner1_calculate_distance(soner1_trigger,soner1_echo);
  if (soner1_dist<70)
  {
    digitalWrite(Buzz_vcc,HIGH);
    for (int i=soner1_dist; i>0; i--)
      delay(10);
    digitalWrite(Buzz_vcc,LOW);
    for (int i=soner1_dist; i>0; i--)
      delay(10);
  }
```

Here we can change the distance value or we can develop the code more with define a new integer "Sonar1_border_distance" as the place of 70 in the code and can define the value in first declaring. It gives the code more user friendly.

Besides, we use downward border line distance at 110cm in the 3rd line of following part-

```
soner2_calculate_distance(soner2_trigger,soner2_echo);
  digitalWrite(Buzz2_vcc,LOW);
  if (soner2_dist>110)
  {
     //Serial.print(soner2_dist);
     //Serial.println("\ngap found");
     digitalWrite(Buzz2_vcc,HIGH);
     digitalWrite(Buzz2_gnd,LOW);
     delay(500);
  }
```

Here we can change the distance value or we can develop the code more with define a new integer "Sonar2_border_distance" as the place of 110 in the code and can define the value in first declaring. It gives the code more user friendly.

5.2 Customizing Moisture Value

In this project we use border moisture value at 900 in the 4^{th} and 10^{th} line of following part-

```
msvalue = analogRead(msensor);
  digitalWrite(Buzz3_vcc, LOW);
  Serial.println(msvalue);
    if ( (msvalue >= 900  ) && ( flag == false ) )
  {
    digitalWrite(Buzz3_vcc, LOW);
    flag = true;
    delay(1000);
  }
    if ( (msvalue < 900  ) && ( flag == true ) )
  {
    digitalWrite(Buzz3_vcc, HIGH);
    flag = false;
    delay(2000);
  }
```

Here we can change the moisture value or we can develop the code more with define a new integer “moisture_value” as the place of 900 in the code and can define the value in first declaring. It gives the code more user friendly.

6

FULL MODEL ANALYSIS

The smart blind man sticks can be make as a strong and useful material for blind man's walking.

The final blind man sticks have following factures:

i. Ability to detect obstacles and inform to blind man about that.

ii. Ability to play siren for wanting help when the blind man fallen down by any accident.

iii. Ability to detect water or slob which is risky for blind man's walking.

iv. It can help blind man to cross risky area or crowd place.

It is hard for a blind man to walk freely everywhere without help of others. Obstacles, stairs, water, slob, etc. can become a big problem for a blind man to go with any new path. With the help of modern technology and sensors, we can build a smart blind man stick which can remove all these trouble of walking and with this technology blind man get the ability back to walk in a busy path easily.

This project is a cost friendly project so that we can use it as business model. This full project can be made within 1,400 takas and we can easily have marketed it more that this amount. The main parts price list (for business use) is below:

Table 6.1 – Price list of the main parts:

Parts name	Parts type	Quantity	Price(per piece in taka)
Arduino Uno R3	Microcontroller	1	420
Ultrasonic sensor	Sonar sensor	2	90
Soil moisture sensor	Sensor	1	350
Buzzer	Sound maker	3	8
Old man sticks	Main base	1	300
Jumper wires (F~M)	Wires	20	40
Switch	Connection joiner	1	6
			Total = 1,320

This smart blind man stick is a effective and useful material for any blind man within a cost friendly budget.

7

AFTERWORD

This project, smart blind man sticks have done with basic command and function. It can be developed more and more in future. The more efficient codes and hardware setup will make it more effective for use.

Some Drawbacks of this project:

i. This project need a battery indicator cause if the battery running out when the blind man uses it on the street then blind man cannot be noticed that the smart stick is stopped working. It may be causes for an accident.
ii. The ultrasonic sensor (sonar) of this project are very effective but it has some drawbacks too. This sensor is really useless with calculating distance from any ultra-soft surface like foams, carpet, etc other things, so that when blind man walking through any carpet or soft surface like it then sonar send wrong signal and buzzer rang without any reasons.
iii. This project is not sufficient for walking in rain time. The sensitive moisturizer sensor always senses moisture and biped buzzer as fond any water/ pond like obstacles.

In future edition we can add extra other sensors like pir motion sensor which can use for tracking busy road where many vehicles run fast. We can add battery level indicator buzzer with it, cause if the battery suddenly stops without any notification then it can be make a huge trouble for the blind man. Adding extra other features like this we can make it more efficient in future.

8

CONCLUSION

With the proposed architecture, if constructed with at most accuracy, the blind people will able to move from one place to another without others help, which leads to increase autonomy for the blind. The developed smart stick that is incorporated with multiple sensors will help in navigating the way while walking and keep alarming the person if any sign of danger or inconvenience is detected. The developed prototype gives good results in detecting obstacles paced at distance in front of the user; it will be real boon for the blind. At the same time global positioning system (GPS) can be linked with the voice stick for navigation, so that person can know his current position and distance from the destination which will be
informed to users through voice instructions.

In this paper, we introduce a smart stick system for assisting blind people. The smart stick comes as a proposed solution to enable visually impaired people to find difficulties in detecting obstacles and dangers in front of them during walking and to identify the world around. The system is designed to act like an artificial vision and alarm unit The system consists of two sensors: ultrasonic sensor, water sensor, microcontroller (Arduino Uno R3) to receive the sensor signals and process them to short pulses to the Arduino pins where buzzers, vibrator and sound alarms are connected. We seek in our project to provide a smart stick affordable and suitable for most blind people, and also it is light in weight. It can be made available to all segments of the society and their families who need them.

We estimate that this project could create a cost-effective budget with huge additional features in the future modified version; Which is very optimistic for all the blind people facing problems in their daily lives.

Appendix

A Materials and Equipment’s

A.1 Uno R3:

Arduino Uno is a microcontroller board based on the ATmega328P (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller;

A.2 Sonar:

An ultrasonic sensor is an instrument that measures the distance to an object using ultrasonic sound waves. An ultrasonic sensor uses a transducer to send and receive ultrasonic pulses that relay back information about an object’s proximity.

B Keywords

B.1 Bip:

Sound created by buzzer.

B.1 vcc:

Voltage source.

B.1 gnd:

Ground voltage.

REFERENCES

[1]. Sung Jae Kang, et al." Development of an Intelligent Guide-Stick for the Blind", Proceeding of the IEEE international Conference on Robotics & Automation, 2001.

[2]. Y. Kawai and F. Tomita, "A support system for visually impaired persons to understand three-dimensional visual information using acoustic interface", IEEE Conference on Pattern Recognition, Vol.3,pp.974-977,2002.

[3]. J. M. Sáez, F. Escolano, and A. Peñalver, "First steps towards stereo- based 6DOF SLAM for the visually impaired," in IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), San Diego, USA,2005.

[4]. Shruti Dambhare, et al., "Smart stick for Blind: Obstacle Detection, Artificial vision and Real-time assistance via GPS", 2nd National Conference on Information and Communication Technology (NCICT), 2011.

[5]. Mohammad Hazzaz, et al., "Smart Walking Stick- an electronic approach to assist visually disable persons", International Journal of Scientific & Engineering Research vol. 4, No. 10, 2013.

[6]. AlbertoRodriguez, et al., "Obstacle avoidance system for assisting visually impaired people", in proceeding IEEE Intelligent Vehicles Symposium Workshop, 2012.

[7]. S.Innet,N. Ritnoom" An Application of Infrared Sensors for Electronic White Stick" 2008 International SymposiumonIntelligentSignalProcessingandCommunicationSystemsBangkok,Thailand.

Books:

[1]. Jan Axelson, Programming and Customizing the AVR Microcontroller – Dhananjay V. Gadre Parallel Port Complete

[2]. Priyanka Patil, Smart stick for blind man Department of Information Technology, K.J. Somaiya College of Engineering, Mumbai, India

To complete the project, I get many online help from the sites-

[1]. wikipedia.org/wiki/Wiki

[2]. www.arduino.cc/

[3]. youtube.com

[4]. circuitdigest.com/microcontroller-projects/arduino-smart-blind-stick

[5]. instructables.com/id/Smart-Blind-Stick/

[6]. https://pdfs.semanticscholar.org/df35/9ab8b894f5180e844a1ff24f186c 7ed75a67.pdf

[7]. www.iosrjournals.org