SMARTBELT

Obstacle Detection Device for the Visually Impaired

Carlos Bautista | Steven Rodriguez

Advisor: Professor Ramesh Karri

Smartbelt Senior Thesis

Contents

Contact Information

Carlos Bautista
cbautista1002@gmail.com
646 – 267 – 4763

Steven Rodriguez
srodriguez511@gmail.com
347 – 403 – 6677

Abstract

The Smartbelt is designed to provide a greater degree of mobility for the visually impaired. It is a hands-free, wearable device which detects obstacles around the user. Through the use of ultrasonic signals, the sensors on the Smartbelt can determine if there are obstacles nearby in three dimensions and provide the appropriate vibration feedback to the user. Integrated circuits are used throughout the system to control the sensors, interpret their readings, and provide proper vibrations.

Introduction

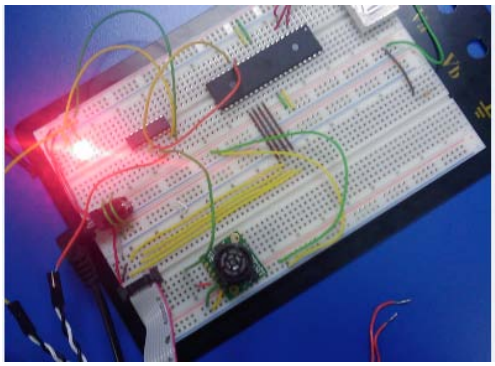
Considering the fact that sight is just one of a human's five senses, it is pretty astonishing to think that a huge percentage, 80 to 90 percent, of the information we assimilate from the world around us is gathered through sight. Even human to human communication is based more on sight and body language than on voice and words. With this in mind, it may be difficult to understand how it is to live without sight. One's ability to communicate, navigate, learn, etc. is greatly affected when sight is not available.

Advances in assistive technology for the visually impaired have been plentiful in certain areas while few and far between in other areas. For example, a visually impaired person's ability to take in information has been greatly improved through brail, audio and advances in computer software. But their ability to navigate through dense indoor and outdoor environments is still limited to the "White Cane" or guide dogs. A hands-free, wearable and technologically advanced device appropriate for the 21st Century is overdue.

The Smartbelt is a design that expands on the already existing assistive methods designed for the visually impaired. Similar to the "White Cane", the Smartbelt's goal is to provide the user with the ability to detect objects within their path.

Development

The project began in the Fall 2008 Semester in the Embedded Systems class taught by professor Karri. Students were asked to pick a sensor of their choice and learn how to interface it with a microcontroller. Professor Karri sparked the interest of an assistive technology for the blind. The idea seemed practical and simple enough so it was decided that a Max Sonar based sensor would be a great sensor to understand as a introduction to a future project.



At the end of the semester, we were successfully able to talk to the sensor using an ATMEL AVR and completed that project. Our next step was deciding where this could be efficiently implemented for a visually impaired person. Initially it was decided that we could put the sensors on the walking cane itself, but discovered that this was already in existence in Europe. Much of the already existing technology for the blind is expensive and complicated to use. We decided that it would be much more efficient to have a hands free device and set forth to achieve the following goals in our project:

Project Goals:

1. **Hands free device**
2. **Vibration feedback alert**
3. **Cost efficient**
4. **Visually hidden and simple**
5. **Reliable**
6. **Detecting forward/backward facing objects**
7. **Gap detection**

Initial Design

From the goals we set at the beginning of the project, we decided that the most simplest and practical applications would be to implement the detection system onto a wearable belt. A sensor would be placed in both the front and rear portions of the belt with the microcontroller along the backside. As a feedback system, we would use a simple vibrator such as those in your cellular phone to indicate an object. One would be placed at both the front and rear portions of the belt. We were successful in creating our initial design.

As can be seen in the picture, the device is wearable and the sensor barely noticeable. LED's would indicate to us, weather the system has power.

Testing and Alteration

Although we had created our first prototype, we were strong believers in improvement. Because the device is highly practicle and extremely usable, it was decided that we would take it to the next step. With assistance from Chris Alvarez of the BEST center, we were able to arrange a meeting with the Hellen Keller Services for the Blind. We brought the device to their office and they were excited about the opportunity to work with us on this great project. They tested our device and gave us great feedback. Highlighted below were the important things we felt we needed to accomplish in our next prototype:

1. No longer a need for a rear sensor as they will always walk forward
2. Definitely need gap detection
3. Vibration feedback is a great idea

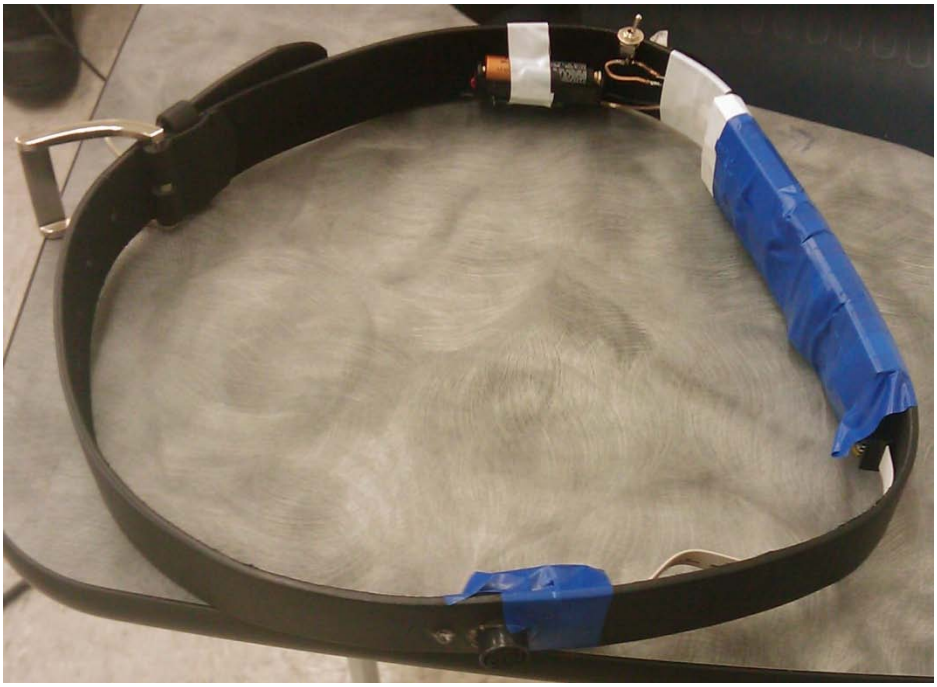
4. Hide the inner wiring from the user
5. Better fit for the user
6. Detection from 3 feet and beyond

With these ideas and our own personal ambitions, we set forth to improve the device for our next meeting.

Work in Progress

For reliability reasons, we moved away from soldering and decided to place a mini breadboard on our new design. This would eliminate the possibility of shorting wires and running long cables around the belt which the users felt were a safety issue. It was much simpler to implement the system the second time around as we knew how to connect everything and the breadboard made it much simpler and efficient.

The biggest addition we added was the change from detection of an object under 2 feet to, detecting objects between 40 inches and 80 inches. We realized that the smart belt is a great addition the smart cane and would be able to warn users of objects just beyond their reach as well as objects not on the floor. In addition, we added ranges of vibration feedback. As the user gets closer to the object the vibrations will increase in intensity and speed. These are easily adjusted through for loops depending on the user's preference.



Gap Detection Enhancement

The final addition to our project is the gap detection system unit. We will have additional sensor placed onto the front portion of the belt which would upon startup determine the users average distance from the ground. Whenever this value is off by a preset amount of inches, eg. 5, we will alert the user that there is either a gap or an incline ahead of them. We are currently working on this part of the system now and will have this available for our second video shoot with the Hellen Keller Services.

Below is a sample of the code:

```
if(countSetup <= 10){
                    sumDist += final;
                    countSetup++;
                    averageDist = sumDist / countSetup;
                    minDist = averageDist - 5;
                    maxDist = averageDist + 5;
                }
                else{
                    //test cases
                    if(final < minDist || final > maxDist ){ //vibrate if
                        PORTB = (1 << PB0) | (1 << PB1) | (1 << PB2);
                        _delay_ms(180);
                        PORTB = (0 << PB0) | (0 << PB1) | (1 << PB2);
                        _delay_ms(150);

                    }
                }
```

Issues During Design

As we designed the system we faced several hurdles that caused much distress. Initially we did not realize the ATMEGA32 shipped at an internal 1MHZ clock. This makes serial transmission impossible as the error rate is too high since the clock speed is not fast enough to cycle. We learned, through much research and datasheet documentation, that the clock fuse bits should be changed to 8Mhz.

Another major issue was soldering the wires onto the device. We did not want to damage any of the circuits and applying large amounts of heat can often cause circuits to fail. It took much testing and re-soldering to often find that wires weren't touching or were holding a very unstable connection. Most of these problems were eliminated with the use of a breadboard.

The biggest issue was designing a system that is implementable onto a belt. It took various design changes to get the sensors onto a location that looked hidden and was still reliable.

Design Specifications

The Smartbelt system consists of ultrasonic sonar sensors, vibration motors, an ATMEGA32 microcontroller, a MAX232 converter and a 16:1 multiplexer. The ATMEGA32 is the controlling unit and receives serial transmission from the sensors. The sensors use the RS232 serial protocol and the ATMEGA32 uses the USART serial protocol. Due to this incompatibility, a MAX232 is used to convert the different voltage levels. However, the ATMEGA32 is only capable of receiving a single serial transmission at a time. Therefore a multiplexer is used to alternate between the sensors, reading one sensor at a time. This can be adjusted and up to 16 sensors can be used per multiplexer. As a source of notification to the user, when any obstacle is detected within a range encoded onto the ATMEGA32, a vibrator alerts the user of the upcoming obstacle. These vibrators are placed next to each sensor which allows the user to have a pinpoint feedback as to where the object is located.

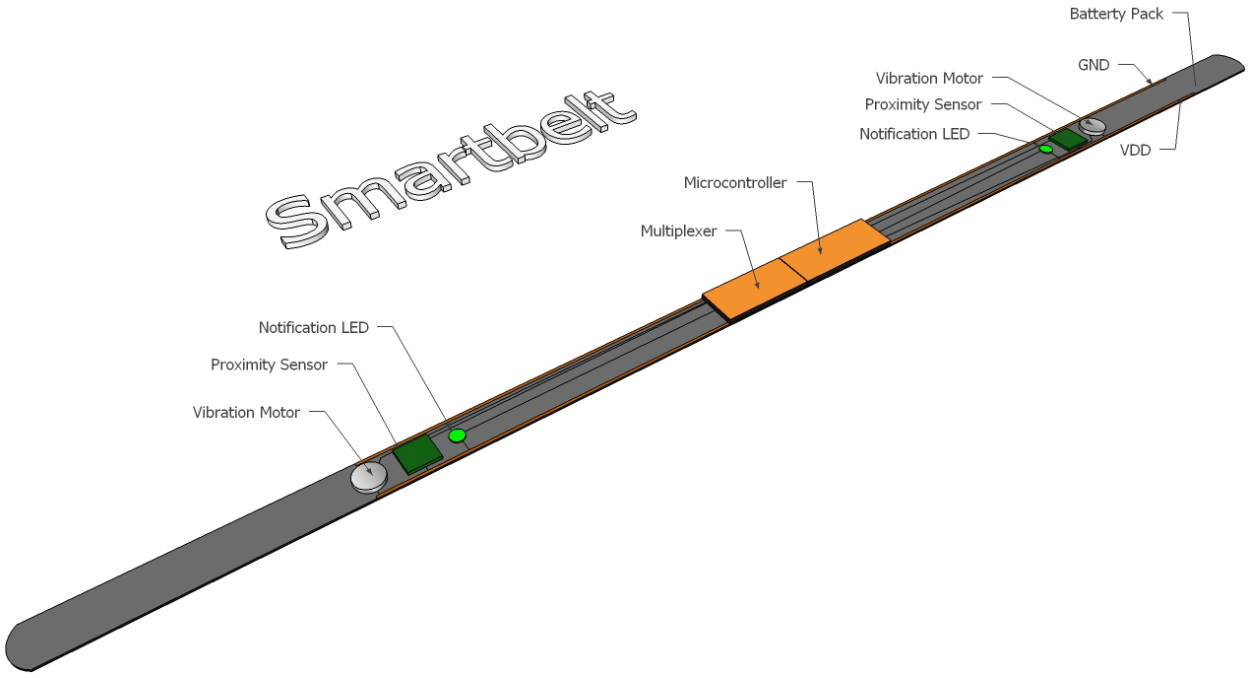


Figure 1: Smartbelt Diagram

LV-MaxSonar

The MaxSonar detects an object's distance using ultrasonic sonar waves and outputs the distance in three different modes simultaneously: Analog Voltage, Pulse Width, and Serial Transmission/Receive. As can be seen here there are three necessary pins for our design, GND, VDD (2.5 – 5.5 V), and TX. The mode of serial operation is RS232. The sensor will output in serial mode, ASCII R followed by 3 successive ASCII bits from 0 – 255, representing the range in inches the object is from the sensor and finally a carriage return. It operates at 9600 bps.

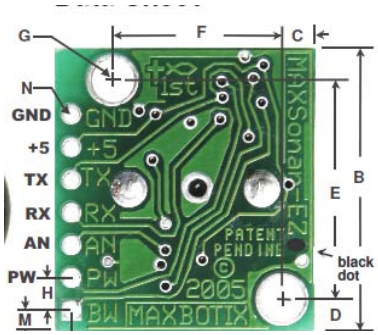


Figure 2: Maxbotix LV MaxSonar Diagram

ATMEGA32

The ATMEGA32 has a USART – Universal Synchronous Asynchronous Receive Transmission. It is a type of serial protocol that works in a smaller voltage range than RS232. This made it necessary to use a MAX232 chip which will translate the USART to RS232 voltage levels and vice versa. By default, the ATMEGA32 ships with an internal frequency of 1MHZ. At these frequencies, 9600 bps serial communication has an error rate of 7% which is too high to properly communicate with the device. Therefore it was necessary to change the internal frequency to 8MHZ where 9600bps has a .2% error rate. This was done by changing CKSEL fuse bits to 0100 using AVR Studio. The UBBR register handles the baud rate. UCSRA, UCSRB, and UCSRC are the USART registers. A and B are mostly status while C is the control register. However, it shares the same I/O address as UBBR. If UCSEL is set to 0, the UBBR register is changed and if set to 1 UCSRC can be changed. UCSRB contains the TX and RX enable bits and UCSRC contains the bits responsible for setting transmission to 8 bits with 1 stop bit and no parity. Port B on the ATMEGA32 was set to act as an output which releases 5 volts in each pin. These pins are where the vibrators and the select lines to the multiplexer are connected.

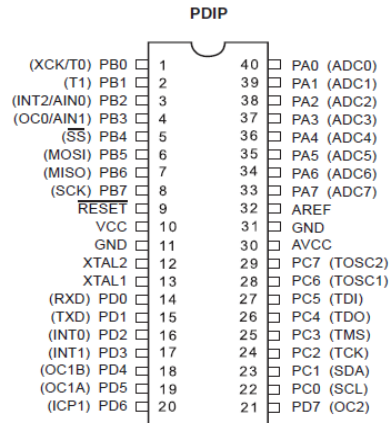


Figure 3: Atmel AVR ATMEGA32 Pinout

16-1 CD74HC4067 Multiplexer

The multiplexer is capable of channeling serial signals. Pin 1, is the output, which is the result of choosing which signal the program will check from the sensors. The enable bit, pin 15, is active low and it is grounded. Pins 10, 11, 14 and 13 represent S0, S1, S2 and S3 respectively. The two sensors are connected to I1 and I3. The code is designed to alternate between the two sensors by counting to a certain number and setting the select signals of I1 and I3 at different times.

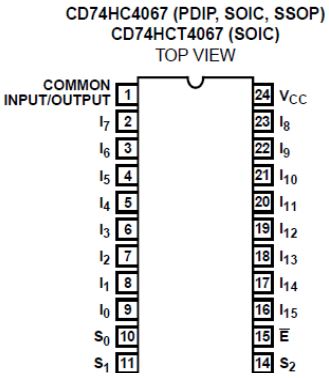


Figure 4: Texas Instruments 16-1 Multiplexer Pinout

Algorithm Design

The algorithm design for the Smartbelt is simple and efficient. The appropriate registers and setup modes for the ATMEGA are activated first. Data is received until an "R" character is sent by the sensor. Once the program receives this signal, it knows the next 3 characters represent the range and therefore it will continue to receive 3 more characters. If needed, this value is also transmitted out from the ATMEGA for debugging purposes and can be seen on any serial interface such as HyperTerminal. Finally, the pre-defined range detection system takes the number it received and checks if it is within the 'danger' zone. When an object is within the danger zone, the vibrator is activated.

```
//USART Program
//The ATMEGA32 will receive a serial transmission and send it
//Frequency - 8MHZ Operating at 9600BPS
//PB0 - VIBRATOR
//PB1 - INDICATOR LED
//PB2 - POWER LED
//PB3 - S0 - MUX (PIN 10)
//PB4 - S1 - MUX (PIN 11)
//PB5 - S2 - MUX (PIN 14)
//PB6 - S3 - MUX (PIN 13)
//FRONT SENSOR - I1 (PIN 8) of MUX
#define F_CPU 8000000UL
#include <util/delay.h>
#include <avr/io.h>
#include <stdlib.h>

void USART_Init(unsigned int baud); //Init Baud
void USART_Transmit(unsigned char data); //Transmission
unsigned char USART_Receive(void); //Receive
int convert(char);

int main(){
```

```
        unsigned char data2; //Hold the data we send/receive
        unsigned char storage[3];
        int hun = 0, ten = 0, one = 0, final = 0;
        int base = 48;
        DDRB = 0b11111111;

        USART_Init(51); //Initilize to 9600bps
        for(;;){

    //I1 is 1000 (S0 S1 S2 S3) - PB3 is high
                PORTB = (1 << PB2) | (1 << PB3) | (0 << PB4) | (0 << PB5) | (0 << PB6);
                data2 = USART_Receive();
                USART_Transmit(data2);
                if(data2 == 0x52){ //if data = R
                        //save the distanc
                        data2 = USART_Receive();
                        storage[0] = data2;
                        data2 = USART_Receive();
                        storage[1] = data2;
                        data2 = USART_Receive();
                        storage[2] = data2;
                        USART_Transmit(storage[0]); //output the distance
                        USART_Transmit(storage[1]);
                        USART_Transmit(storage[2]);
                        //convert to a number, makes it easier to use
                        hun = convert(storage[0]);
                        ten = convert(storage[1]);
                        one = convert(storage[2]);
                        //make it integer total value
                        hun = (hun - base) * 100;
                        ten = (ten - base) * 10;
                        one = one - base;
                        final = hun + ten + one; //final value should match storage

                        //test cases
                        if(final >= 40 && final <= 80){ //vibrate if
                                if(final >= 70 && final < 80){
                                        for(int i = 0; i < 1; i++){
                                                PORTB = (1 << PB0) | (1 << PB1) | (1 << PB2);
                                                _delay_ms(150);
                                                PORTB = (0 << PB0) | (0 << PB1) | (1 << PB2);
```

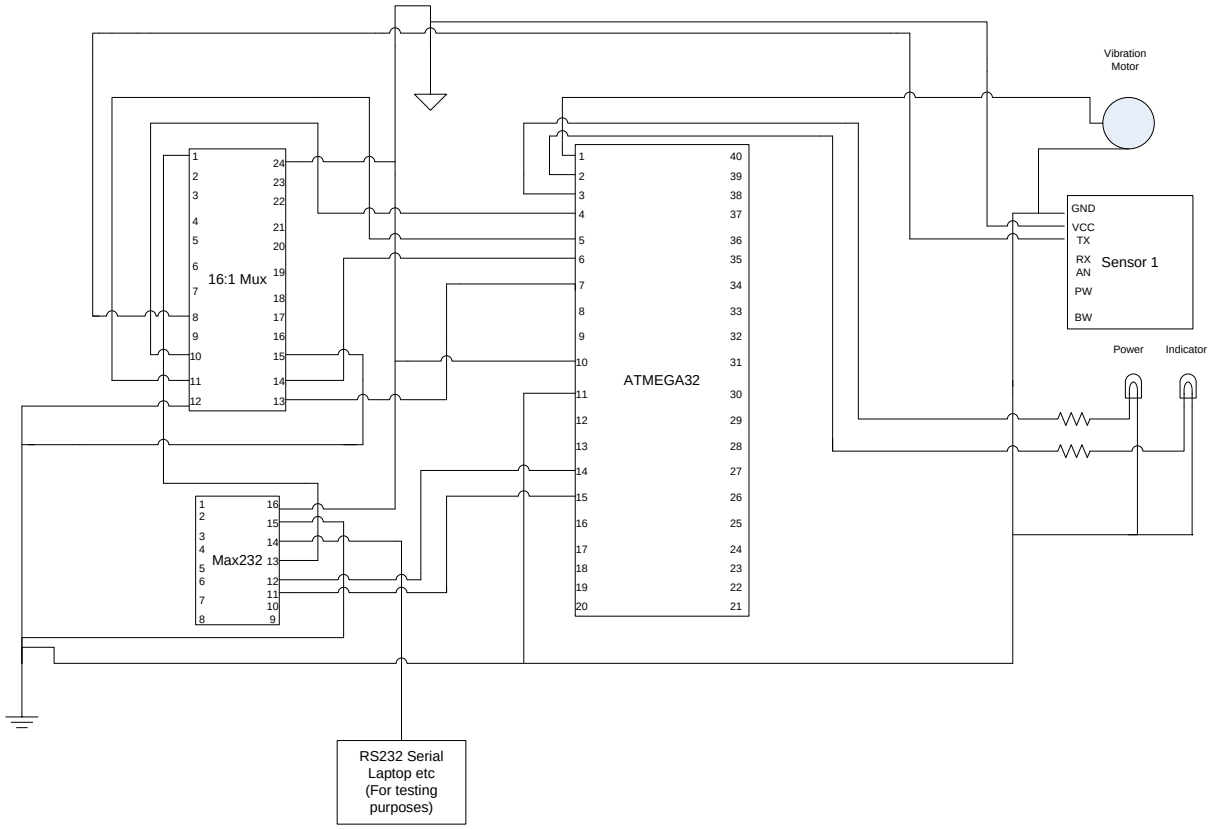
```
                                        _delay_ms(230);
                                }
                        }
                        else if(final >= 60 && final < 70){
                                for(int i = 0; i < 1; i++){
                                        PORTB = (1 << PB0) | (1 << PB1) | (1 << PB2);
                                        _delay_ms(180);
                                        PORTB = (0 << PB0) | (0 << PB1) | (1 << PB2);
                                        _delay_ms(150);
                                }
                        }
                        else if(final >= 50 && final < 60){
                                for(int i = 0; i < 1; i++){
                                        PORTB = (1 << PB0) | (1 << PB1) | (1 << PB2);
                                        _delay_ms(180);
                                        PORTB = (0 << PB0) | (0 << PB1) | (1 << PB2);
                                        _delay_ms(125);
                                }
                        }
                        else{
                                for(int i = 0; i < 1; i++){
                                        PORTB = (1 << PB0) | (1 << PB1) | (1 << PB2);
                                        _delay_ms(150);
                                        PORTB = (0 << PB0) | (0 << PB1) | (1 << PB2);
                                        _delay_ms(100);
                                }
                        }
                }
        }

    }
    return 0;
}
void USART_Init( unsigned int baud )
{
    UBRRH = (baud >> 8); ///Set baud rate
    UBRRL = baud;
    UCSRB = (1<<RXEN)|(1<<TXEN); //Enable RX and TX
    UCSRC = (1<<URSEL)  | (3<<UCSZ0) ; //8 data bits . 1 stop bit.
}
void USART_Transmit( unsigned char data )
```

```
{
        while ( !( UCSRA & (1<<UDRE)) ); // Wait for empty transmit buffer
        UDR = data; //Put data into buffer, sends the data
 }
unsigned char USART_Receive( void )
{
        while ( !(UCSRA & (1<<RXC)) ); //Wait for data to be received
        return UDR; //Get and return received data from buffer
}

//convert the ascii chars to dec
int convert(char achar){
  if(achar == '0')
        return 48;
  else if(achar == '1')
        return 49;
  else if(achar == '2')
        return 50;
  else if(achar == '3')
        return 51;
  else if(achar == '4')
        return 52;
  else if(achar == '5')
        return 53;
  else if(achar == '6')
        return 54;
  else if(achar == '7')
        return 55;
  else if(achar == '8')
        return 56;
  else
        return 57;
}
```

Schematic



Cost Analysis

Prototype System:

Unit	Quantity	Cost	Total Cost
Belt	1	$5.00	$5.00
Sensor	2	$25.00	$50.00
ATMEGA32	1	$8.80	$8.80
MAX232	1	$1.00	$1.00
CD74HC4067 MUX	1	$1.00	$1.00
Vibrator	2	$7.95	$15.90
LED	2	$0.50	$1.00
Breadboard	3	--	$10.00
Total Cost			**$92.70**

As you can see, our system is fairly cost efficient and affordable for an everyday user.

Benefits for Target Audience

There are 20 million Americans who are documented as having experienced exceptional vision loss. Nearly one million of those are legally blind, ten percent of which have absolutely no vision. This is an unfortunately large number, in any regard. The Smartbelt can improve the everyday life of these individuals.

A human's ability to safely navigate is highly dependent on sight. Any device that can aid in this process performs a critical function. The Smartbelt cannot substitute sight but it can emulate one important detail that sight provides us with – obstacle detection. Being able to detect where an obstacle is, relative to the user, increases a visually impaired person's ability to navigate safely. It performs this all while not overwhelming any other sense. The second most important sense to an individual, who is visually impaired, hearing, is left unaffected letting the user employ their ears for other important notifications. Also, because the device is wearable it leaves the user with the ability to use their hands freely.

Although the visually impaired are currently used to other methods of obstacle detection, they can easily learn to use the Smartbelt. After training with the Smartbelt, sensing and interpreting the vibrations can become second nature.

System Comparisons | Smartbelt Features

Currently the favored methods of obstacle detection are walking sticks and guide dogs. These methods pose several issues and concerns.

- × Walking sticks mainly detect obstacles that lay on the ground leaving the user the task of figuring out the height of the object.
- × Walking sticks also occupy one of the user's hands – a highly unfavorable characteristic when trying to navigate.
- × Walking sticks may accidently hit other people while the user it trying to navigate.
- × Walking sticks have a limited range – their length depends on the user's height although longer sticks may be used. Their range is still limited to just a couple of feet.
- × Walking sticks may be difficult to store in public environments, especially if they're not foldable/collapsible.
- × Walking sticks require a constant back and forth waving motion. Although the user can get used to this, they can still become physically tired – especially those with further physical disabilities or the very young and elderly.
- × Walking sticks are a highly visible symbol of blindness. Although this is usually a positive characteristic, under certain conditions a visually impaired person may not want to be identified as such.
- × Guide dogs require lots of maintenance. Like any other pet, they must be fed and looked after.
- × Guide dogs are not a suitable option for those with allergies.

The Smartbelt does away with many of these issues, all while providing the aforementioned benefits (previous section).

- ✓ Smartbelt's detect obstacles in three dimensions, up to 21 feet, without occupying the user's hands.
- ✓ Smartbelt's are small and are worn directly on the body without ever affecting any nearby pedestrians. Its small form factor makes it very discrete and unnoticeable.
- ✓ Smartbelts do not require constant user intervention for proper operation allowing the user to concentrate on other activities. This also saves the user energy expenditure.
- ✓ Smartbelts need not be taken off when not in use. They can simply be turned off.
- ✓ Smartbelts are virtually maintenance free. The only requirement is that the battery needs to be recharged / replaced.
- ✓ Smartbelts are highly scalable, configurable and customizable providing the opportunity for several price points.
    - o Many sensors can be placed onto the Smartbelt: as little as one sensor or as many as the belt can hold.
    - o Potential user can have the option of selecting different configurations. This allows for the user to only pay for what they need.
    - o As the Smartbelt is worn on the body, it can become a fashion statement. Different materials and textures can be used for the belt allowing the user to display their personal taste.

Future Steps

We are working with Poly's incubator to have the device patented and used in the marketplace. Additionally, we are continuing to work with the Helen Keller Services to determine appropriate needs for the device and constant improvements. The system is so versatile that it can be used on any part of the body and it is even possible to integrate an entire system along different articles of clothing. In future additions, we plan to use smaller chips to make the system more reliable and hidden as well as cost efficient. The arduino based platform is one such solution.