ECE 445 Team 39 Final Project Report

1. Motivation

1.1 Problem and Solution:

Problem statement:

The project that we decided to modify and work on is the project 'Electronically Enhanced Blind Probing Cane' (Team 18) from Spring 2019. The existing solution is an electronically supplemented traditional walking cane, using LIDAR sensors to detect obstacles and a haptic feedback system using a bracelet peripheral that is used to alert the user when an object is within 1.5 metres from the user. The cane communicates with the user by varying the intensity of vibrations in a bracelet that the user wears as he/she approaches an obstacle. Thus, the scope of the problem that we are trying to solve is: coming up with a solution that improves the walking experience of users with visual impairments, allowing them to better navigate their path around environments that have obstacles. We need to do so by addressing the following challenges:

1) Giving the user a more intelligible form of information about the distance to the detected obstacle rather than just vibrational feedback, which will give the user a much better idea of how far ahead an obstacle is because he can intuitively think of it in terms of how far ahead he needs to walk rather than the degree to which his bracelet has increased in vibrations.
2) The existing project solution will not be able to detect moisture or water and this is definitely a big challenge because if the user steps on water without knowing it and continues to walk on wet surfaces, he/she could seriously injure himself/herself.
3) The existing solution is not equipped to deal with rain and dirt which could cause issues in the practice.
4) The existing solution uses sensors that are bulky and expensive, and it requires the use of a specialized cane that must be carried around.

Proposed solution:
In our new implementation, we aim to fundamentally redesign the cane implementation by coming up with a system which consists of a mounting structure (analogous to a sandal without its sole that uses velcro-based straps to attach to the user's right shoe) that can be placed on top of a shoe such that it contains an ultrasonic sensor (Sensor 1) facing forward which informs the user of how many steps away an obstacle is (by converting the raw distance to the obstacle to a personalized step distance metric based on user's height), from the direct line of the user's gait. Our solution will also alert the user if the surface that the user is walking on is wet, with the help of electrodes that are connected to a fluid detector IC. In the current implementation, the device will only need to be placed on the right shoe and not the left. We use three additional

ultrasonic sensors, one (Sensor 2) to provide the signal to the microcontroller when the foot is flat on the ground, two (Sensors 3 and 4) to track the left foot's position so that it isn't detected as as an obstacle by Sensor 1. We will use a plastic encasing on all the sensors and components to ensure that they aren't affected by rain or dirt.

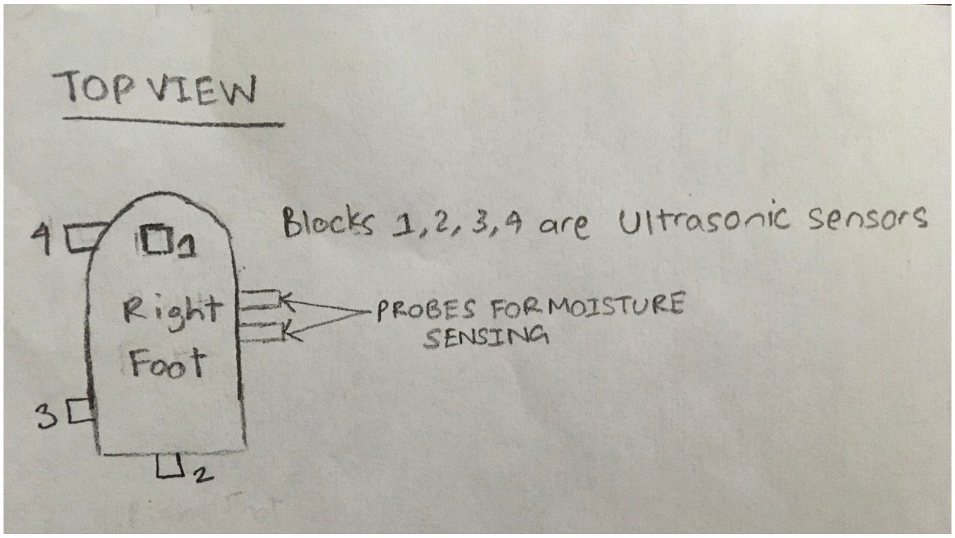
We address all of the challenges mentioned in the problem statement in the following way:
1) The user is given a step-count which will give him/her a much better idea of how far ahead an obstacle is because he can intuitively think of it in terms of how far ahead he needs to walk to encounter the obstacle rather than relying on his ability to interpret changing vibrational sensations.
2) Our device uses a moisture sensor that will alert the user if he/she steps on water and with this new information, the user can slow his gait and be more conscious while walking. This could help prevent a major accident.
3)By encasing all the wires, the top exposed parts of the sensors and the microcontroller in plastic encasings, we ensure that our device can operate in rain and dirt.
4) Our device uses the inexpensive, light, and small ultrasonic sensors which cost a fraction of the amount that lidar sensors cost. Additionally, our device can be used as an attachment to a regular shoe and does not require an entire other device such as a cane to be carried everywhere the user goes.

1.3 High-level requirements:

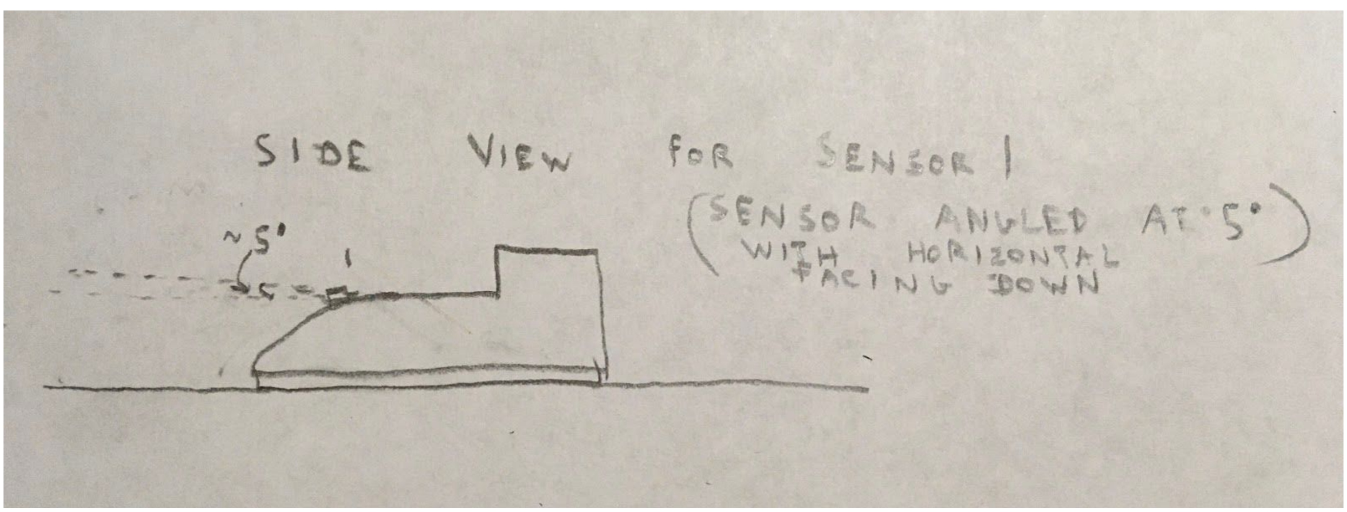
1) The system must be able to detect obstacles to an accuracy of 70% that are up to a range of 1.5 m from the user and must be able to provide accurate audio feedback (about the distance of the obstacle in units of number of steps) to the user when the said obstacle is encountered.
2) The system must be able to detect if the surface that the user is walking on is wet. The failure rate should not be more than 10% and the false positive rate should not be more than 10%.
3) The product, which is integrated on the user's shoe, must not hinder the capacity of the user to walk freely in any manner.

1. Visual Aid:

Layout of sensors as seen from a top view

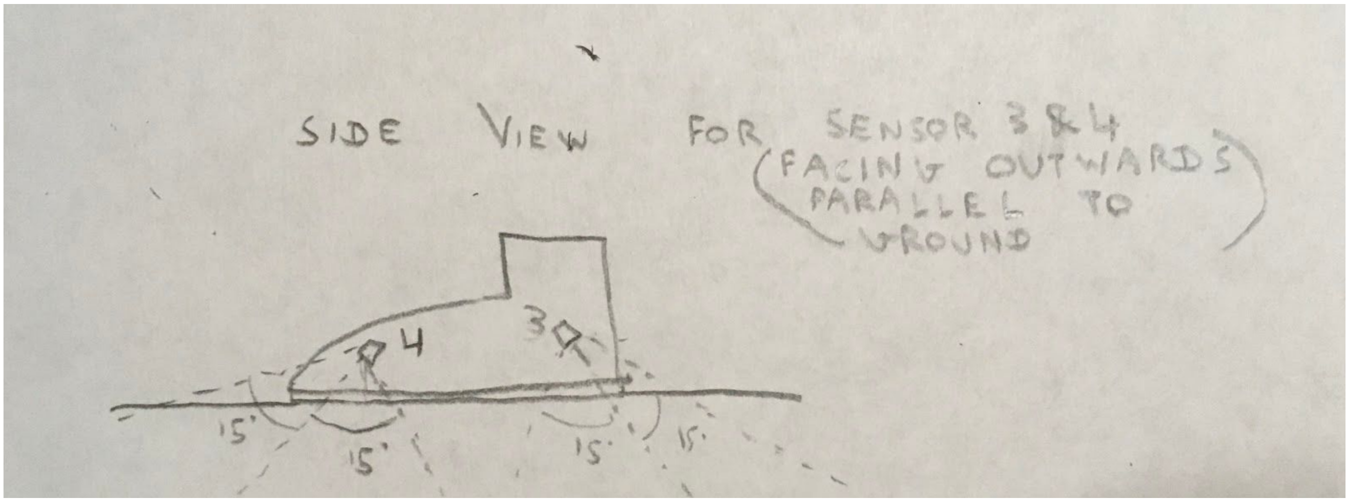


Side view profile: Looking at sensor 1.



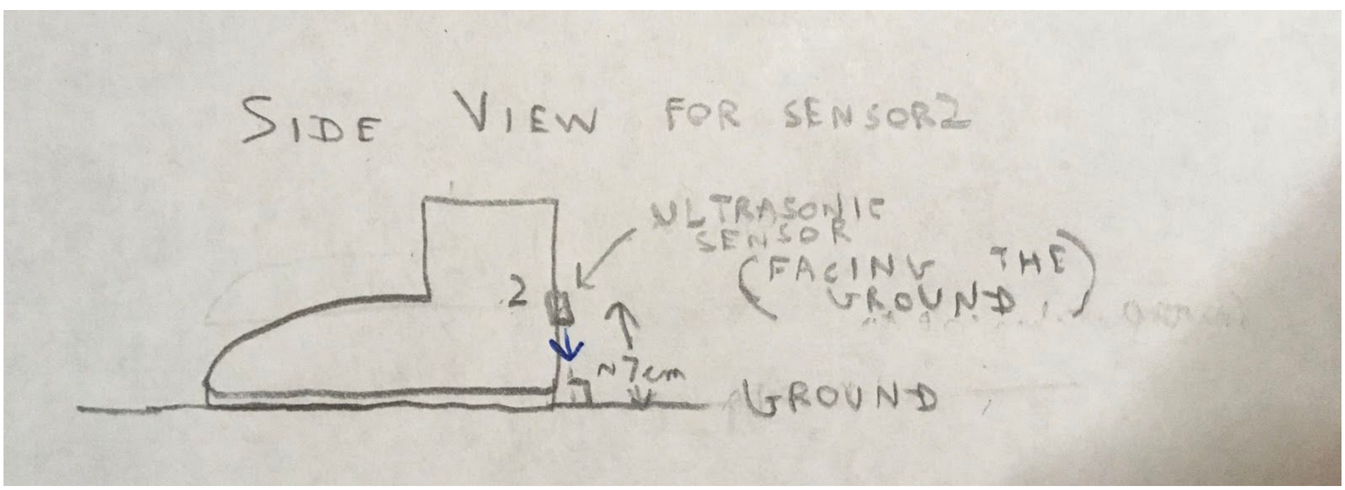
This sensor is responsible for detecting obstacles that are upto 1.5 m away. It will be at a height of 7cm from the ground angled downwards from the horizontal by 5°.

Side view profile: Looking at sensor 3 and 4.



This sensor is responsible for detecting the moving left leg so that the obstacle data provided by sensor 1 is treated as noise whenever the left leg enters its sensor's field of view. These sensors will be placed at a height of 7 cm.

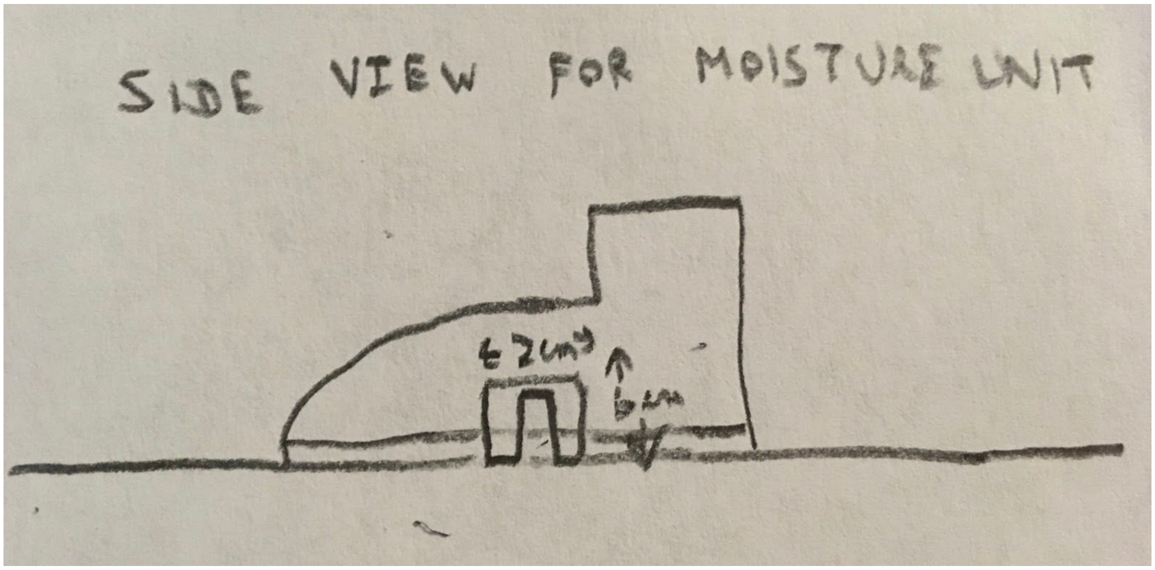
Side view profile: Looking at sensor 2



This sensor faces the ground, and is at a height of 7 cm. When a person walks, this sensor will detect an obstacle at 7cm (the ground) only when the shoe is flat (which could be less than a second)

Its sole purpose is to make sure that the obstacle data provided by sensor 1 is used only when the entire shoe is flat on the ground.

Side view profile: Looking at moisture sensors and electrodes



This moisture unit consists of a moisture detection IC and a pair of electrodes of the dimensions ~(2cm x 6cm) that will be placed at a height of 6 cm on the side of the shoe. Their purpose is to detect the presence of moisture so as to alert the user that he has stepped into water.

2. Design

Figure 1. Block Diagram

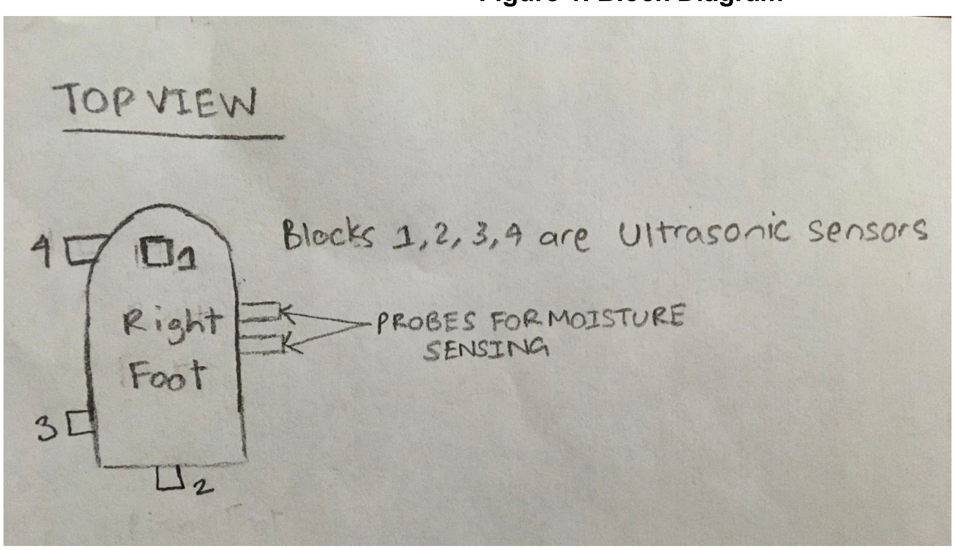


Figure 2. Physical Design

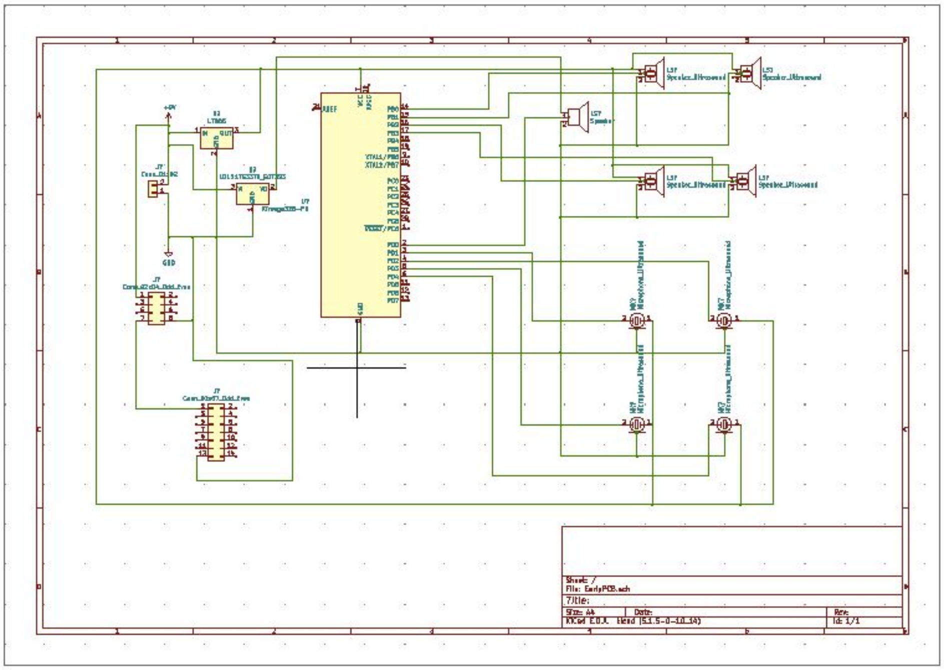


Figure 3. Circuit Schematic

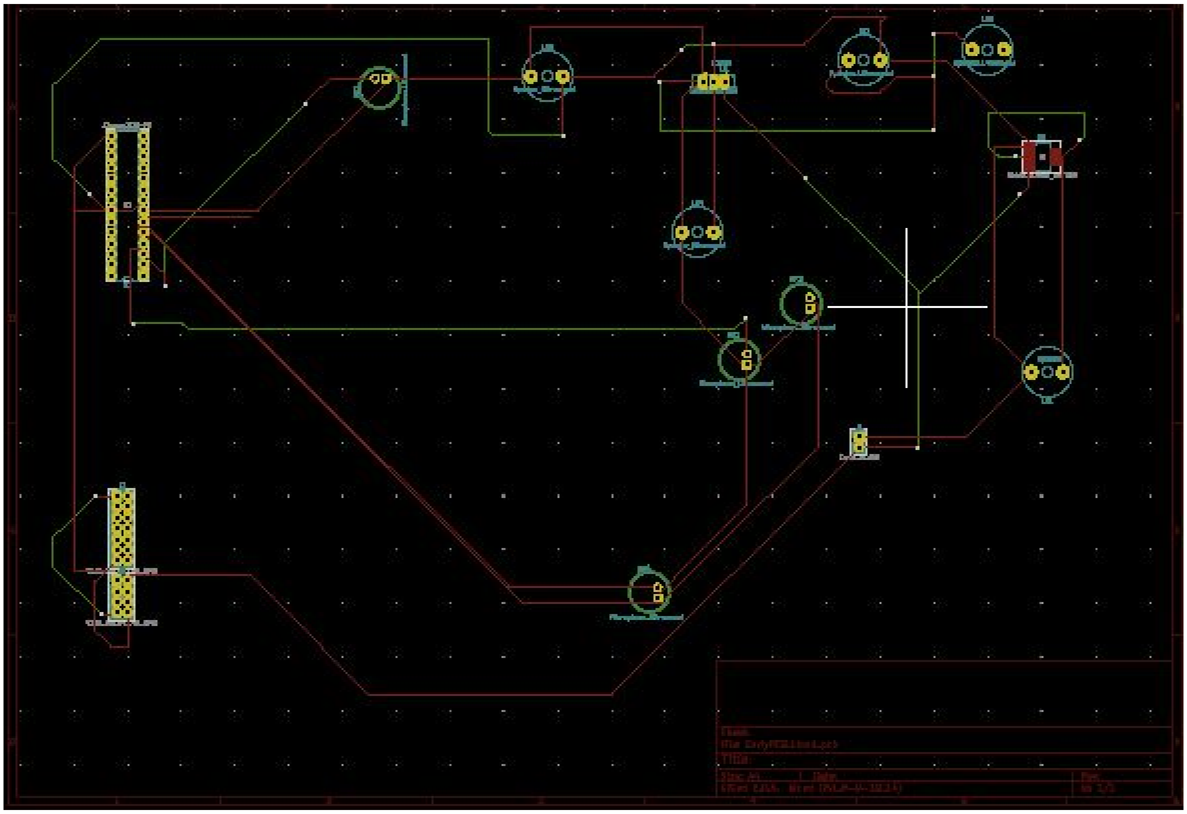
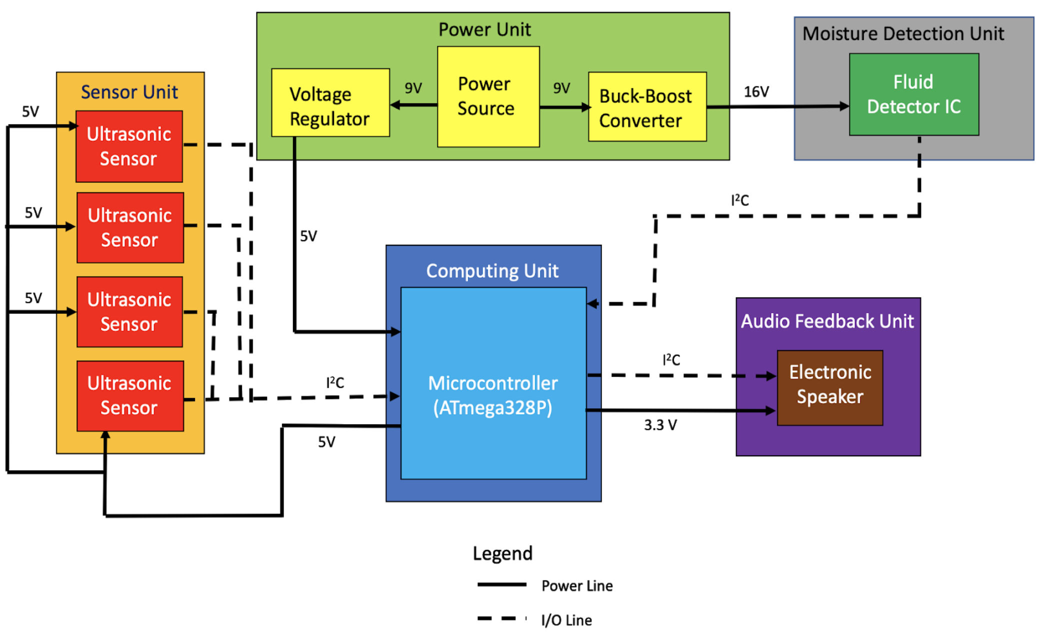


Figure 4. PCB Layout



2. Implementation

1.1 Filter implementation for sensor unit:

Based on the design review of our second project, one of the important feedbacks that we received was that ultrasonic sensors are prone to noisy readings, and that we must have a filtering mechanism to smoothen the output of the ultrasonic sensors in our sensor unit. Thus, we have decided to design, implement and simulate the filtering mechanism of our sensor unit as a part of the implementation section for our final report.

In terms of the filter design, we have decided to use a **moving-average filter** for our filtering implementation. We first looked at different filter designs like the general low-pass filter, median filter and Gaussian filter, however, we decided to go ahead with the moving-average filter because it is easy to understand and use and it has been described as an optimal filter in reducing random noise from measurements and it also retains a sharp step response [12]. The moving average filter computes an average of a number of previously occurring points (M) for each point in the input signal as follows:

$$y[n] = \frac{1}{M}\sum_{k=0}^{M} x[n-k]$$ [13]

Using the above expression, it can also be seen that this operation can be performed simply by convolving x[n] with a length M filter h[n] as follows:

$$\{h[i]\}_{i=0}^{M-1} = \frac{1}{M}$$ [14]

The implication of using this filter is that for each point in the input sequence, it computes an average of M previously occurring points and including the current point, and assigns this average to be the filtered output for the current point.

Thus, we now have the sequence h[n] which we can use as the filter in our convolution process, however, an important point to note is that we cannot simply convolve x[n] and h[n] as they currently stand. This is because x[n], which is the output sequence (of sensor measurements) produced by the ultrasonic sensor, is of indefinite length, which can potentially go on to infinity. In other words, since x[n] is a stream of data that is continually generated by the ultrasonic sensor while it is in operation, we do not know the length of this sequence in advance in order to compute the convolution as presented above.

Thus, we will use a technique called **block-convolution**, specifically the **overlap-add implementation** of block convolution to compute the convolution operation. Block-convolution represents the overall convolution of x[n] and h[n] as a sum of smaller convolutions which are computed by convolving h[n] and L-length locks of x[n] (smaller length blocks of x[n], where L is an arbitrarily chosen integer) [13]. Hence, in essence, what is being done is the sequence x[n] is being broken down into smaller blocks of length L and then the convolution of these blocks and the filter h[n] is being performed as follows:

$$y[n] = \sum_{m=0}^{+\infty} y_m[n-mL]$$

where:

$$y_m[n] = x_m[n] * h[n]$$
$$x_m[n] = x[n + mL] \qquad 0 \leq n \leq L - 1$$

Another important point to note about the overlap-add implementation is that the last M-1 elements and the first M-1 elements of consecutive convolutions ($y_m[n]$) are added.

In this implementation, we are continually performing the convolution operation on the output sequence of the ultrasonic sensor and the filter. Thus, we want to make this convolution operation as efficient as possible. Let us look at the time complexity of the convolution operation if this was simply performed in the sequence (time) domain:

In order to compute one block of the output convolution, we require $O(LM)$ multiply-add operations where L is the number of elements in the ultrasonic sensor output block and M is the number of elements in the filter. We can do better in order to reduce the complexity in computing one block of output elements by using a technique called the **Fast Fourier Transform**. The Fast Fourier Transform has complexity $O(NlogN)$, so in order to utilize the Fast Fourier Transform to compute convolutions:

1) We first zero-pad the two sequences to length L+M-1 which is the length of the linear convolution (N) which can be done in constant time.
2) Transform x[n] and h[n] to frequency domain using FFT which takes total $2NlogN$ time, assuming we have access to only one FFT core at a time and we perform the FFTs serially. A parallelized implementation could bring the time to $NlogN$, but for this discussion we will stick with the serial implementation.
3) Perform multiplication in the frequency domain which requires N operations.
4) Compute Inverse Fast Fourier Transform to transform from frequency domain back to time domain which takes $NlogN$ time.

Thus, in total, the convolution operation takes total $3NlogN + N$ time which is a substantial reduction from the non-optimized quadratic time.

Below is a generalized pseudocode implementation of the filtering aspect of our project:

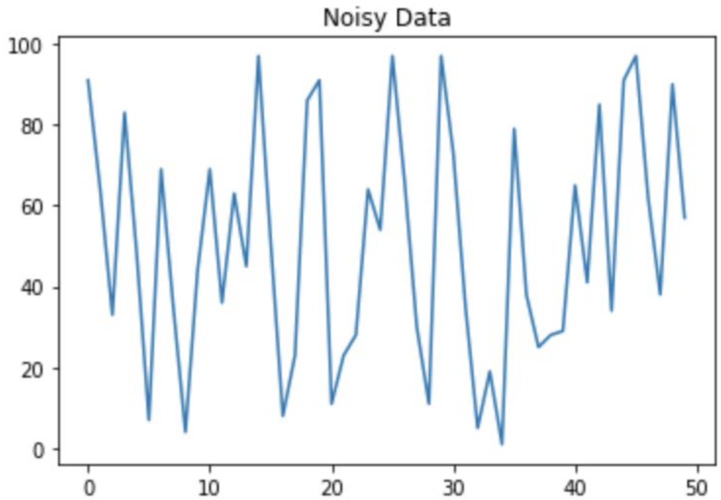
```
L = 100 #size of the block (needed for block convolution) we have arbitrarily chosen L = 256
M = 32  #size of the filter (arbitrarily chosen)
N = L+M-1 #length of the linear convolution

#we are assuming that the the sensor output data x is streaming and already stored in memory

#the filter h is also instantiated and located in memory
h = (1/M)*ones(M)

y_prev = zeros(N) #contains the output convolution of the previous block step (initialized to zeros)
y_cur = zeros(N) #contains the output convolution of the current block step (initialized to zeros)

for m in range(len(x)/L): #iterate through the different block steps
    x_m = x[(m*L):(m+1)*L]  #perform blocking on the sensor output (form L-length block)
    xzp = x_m.append(zeros(M-1))  #zero pad x to be of length L+M-1 (length of linear convolution)
    hzp = h.append(zeros(L-1))    #zero pad h to be of length L+M-1 (length of linear convolution)
    X = FFT(xzp)  #compute the fft of xzp
    H = FFT(hzp)  #compute the fft of hzp
    for i in range(N):  #perform multiplication of X and H in frequency domain
        Y[i] = X[i]*H[i]
    y_cur = IFFT(Y)  #perfom IFFT to obtain convolved output in time domain
    for j in range(M-1): #add the last M-1 elements of y_prev and first M-1 elements of y_cur
        y_cur[j] += y_prev[N-(M-1)+j]
    y_prev = y_cur
```

In order to simulate the effect of using the moving-average filter on a finite input array of noisy data, we wrote a python script that takes this more generalized pseudo-code and simplifies certain aspects of the code [14]. For example we use a python function np.convolve which performs convolution and since we are testing with a finite input array we do not need to use block convolution. We have included the python script below:

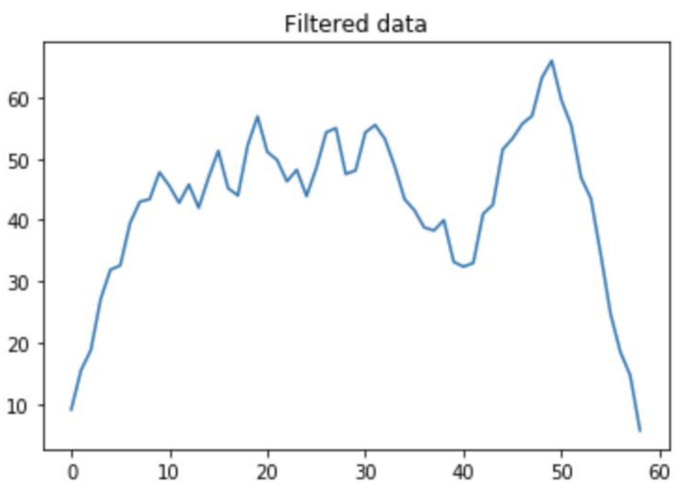
```
In [12]:
import numpy as np

x = np.random.randint(low=1, high=100, size=50)
N = 10
h = (1/N)*np.ones((N,))
y = np.convolve(x_n,h)
y
```

```
Out[12]: array([ 9.1, 15.5, 18.8, 27.1, 31.9, 32.6, 39.5, 43. , 43.4, 47.8, 45.6,
        42.8, 45.8, 42. , 46.9, 51.3, 45.2, 44. , 52.2, 56.9, 51.1, 49.8,
        46.3, 48.2, 43.9, 48.5, 54.3, 55. , 47.5, 48.1, 54.3, 55.5, 53.2,
        48.7, 43.4, 41.6, 38.8, 38.3, 40. , 33.2, 32.4, 33. , 41. , 42.5,
        51.5, 53.3, 55.7, 57. , 63.2, 66. , 59.5, 55.4, 46.9, 43.5, 34.4,
        24.7, 18.5, 14.7,  5.7])
```

Below is the graph of the simulated noisy data:



Below is a graph of how the noisy data looks like after filtering:



As can be seen from the above plot, the filtering operation smoothes out the noisy data and comes up with a much cleaner version of the sensor output that can be further processed to perform the necessary operations in our system like obstacle detection.

Furthermore as we have discussed above, in the actual implementation of our project, we would likely make use of a specialized FFT core architecture to perform the convolution operation, which if used, would dramatically reduce the computation time of our convolution operation as the number of elements in our sensor output scales.

1.2 Programming aspect to obtain sensor information:

The code blocks contain comments explaining how the code works. This would be what we plan to use for our first iteration before debugging.

```
// 1,2,3,4 correspond to ultrasonic sensors as mentioned in the visual aid.
const int t_P1 = 2; //define and setting the pin values
const int e_P1 = 3;
const int t_P2 = 4; //define and setting the pin values
const int e_P2 = 5;
const int t_P3 = 6; //define and setting the pin values
const int e_P3 = 7;
const int t_P4 = 8; //define and setting the pin values
const int e_P4 = 9;

//Just to clarify, sensor 4 is the sensor placed closer to the toes
//and sensor 3 is the one further away...refer to visual aid above
// Each of these numbers corresponds to IO pins on the ATmega chip.
// generally, t_P stands for trig_pin and e_P stands for echo_pin
// Since there are 4 sensors operating, we need 2 variables for each ultrasonic sensor
float t,d d1,d2,d3,d4,h,c ,s;
//t is the time here and d1,d2,d3,d4 are the distance. h is th4e height.
//c is the conversion factor and s is the step size
void setup() {
  pinMode(t_P1, OUTPUT);
  pinMode(e_P1, INPUT);
  pinMode(t_P2, OUTPUT);
  pinMode(e_P2, INPUT);
  pinMode(t_P3, OUTPUT);
  pinMode(e_P3, INPUT);
  pinMode(t_P4, OUTPUT);
  pinMode(e_P4, INPUT);
  Serial.begin(9600); //9600 here is the baudrate
}
```

```
void loop() {
  //Here we reset the trigpin
  digitalWrite(t_P1, LOW);
  digitalWrite(t_P2, LOW);
  digitalWrite(t_P3, LOW);
  digitalWrite(t_P4, LOW);
  delayMicroseconds(2);

  //Here we are basically firing the pulse. The high state is maintained for 10 micro-
  digitalWrite(t_P1, HIGH);
  digitalWrite(t_P2, HIGH);
  digitalWrite(t_P3, HIGH);
  digitalWrite(t_P4, HIGH);
  delayMicroseconds(10);
  digitalWrite(t_P1, LOW);
  digitalWrite(t_P2, LOW);
  digitalWrite(t_P3, LOW);
  digitalWrite(t_P4, LOW);

  t1 = pulseIn(e_P1, HIGH);
  t2 = pulseIn(e_P2, HIGH);
  t3 = pulseIn(e_P3, HIGH);
  t4 = pulseIn(e_P4, HIGH);
  d1 = (t*.034)/2; //0.340 represents the speed of sound in cm/microsecond
  d2 = (t*.034)/2;
  d3 = (t*.034)/2;
  d4 = (t*.034)/2;
```

```
  t4 = pulseIn(e_P4, HIGH);
  d1 = (t*.034)/2; //0.340 represents the speed of sound in cm/microsecond
  d2 = (t*.034)/2;
  d3 = (t*.034)/2;
  d4 = (t*.034)/2;
  h = 175; //corresponds to 175 cm height
  c = 0.415; // conversion factor of height to gait.
  s = h*c;

 if ((d2<7.35)&&(d2>6.65)&&(d4>d3)){
 // the condition above is met only when the foot is flat and the right shoe is ahead of
 Serial.print("Dist: ");
 Serial.println(d1); //This code is to print the value of d on the console
 Serial.print("Step size: ");
 Serial.println(d1/s); //This code is to print the number of steps
 }

 }
```

3. Conclusion:

3.1 Summary of Implementations:

For the summary of implementation, the first implementation aspect of our project that we worked on was the implementation and discussion of the filtering module for our sensor unit. Based on the feedback that we received from our design document review, one key area that we needed to address was the possibility of noisy measurements from the ultrasonic sensors in our sensor unit.

We came up with a moving average filter implementation in order to perform the filtering of the output data from our ultrasonic sensors. We first surveyed some popular filter choices like the generalized low pass filter and median filter but we decided to go ahead with the moving average filter because it is simple to understand and implement.

In the first section of the implementation details for our filtering module, we do a brief overview and summary of signal processing theory and talk about the math behind our filter implementation. We point out that since the ultrasonic sensor outputs a constant stream of data, we do not know the size of the input sequence beforehand. Despite this uncertainty, we can make use of the block convolution (specifically overlap-add implementation of block convolution) to break up the sensor output into blocks which can then be processed separately.

Next, we also point out that since the convolution operation which is being performed continually in our filtering module is time intensive with a time complexity of quadratic time. Hence we introduce the notion of performing filtering using the Fast Fourier Transform.

We next illustrate the time complexity that would be achieved when using an FFT implementation and we then go on to present a pseudocode implementation of the filter using this fast FFT approach.

We then also simplified this logic for the purposes of simulating our filter on a fixed size of randomly generated noisy data. We also showed how the moving average filter manages to smoothen out the noisy measurements to produce a more “smooth” output data which can be processed further down the pipeline of our system.

The development of the schematic was relatively complicated as compared to the development of the schematic for the initial project. We had relatively little knowledge about some of the components that we planned to use in our system which made it harder to organize and layout the PCB. We carefully considered the various components in our system and weighed out the pros and cons for every component and thereby made very careful choices. However, we would have liked to find cheaper alternatives to implement the moisture detection IC as it was the most expensive component in our system. Since the current situation made it impossible to test our design on a real printed circuit board, we cannot be completely confident in the functionality of our system design. There will always be external factors that would adversely affect our system such as acoustically soft surfaces and noisy data from the ultrasonic sensors, but with thorough testing and development, we are quietly confident that our system will perform at an optimal level.

3.2 Unknowns and Uncertainties:

Acoustically soft substances, or in general any substance that is either too small (less than 5mm by 5mm by 5mm) or has a bad sound reflection coefficient will affect the performance of our ultrasonic sensors. Additionally, output noise will also affect our sensors. We are implementing filtration algorithms and might even potentially use more ultrasonic sensors to corroborate sensor 1 data . Despite all this, given the circumstances, the extent to which our design is accurate in detecting obstacles will only be known during the testing phase. Based on our research, the algorithm that we are planning to implement is the moving average filter. We anticipate that it will significantly reduce noise in our output signals and help more accurately predict how far an obstacle is.

Since the mounting structure is placed on top of the user’s shoe, it remains to be seen if it is a source of discomfort to the user or not. It is also possible for the mounting structure to vibrate as and when the user walks which may cause some disturbances and nose in the ultrasonic sensor output

We have still not received or tested our PCB, and while we are confident that it will work, there is no way that we can be 100% sure even before we start testing. Errors may arise when we are soldering components onto the board. Even if the design may work theoretically, it remains to be seen if other external factors have an adverse effect on the performance of our system.

3.3 Ethical Considerations:

Since we will be using lithium batteries in our project, a safety concern associated with using lithium batteries is called “thermal runaway” which causes the battery to overheat and leads to operational failure. Thermal runaway can also cause the battery to heat up to a point that it can catch a fire, thus we need to be cognizant of the fire hazards that are associated with the use of lithium batteries in our project. In the case of a fire, we will make sure that we follow the protocols and safety measures that were discussed in the safety training online lab module which was introduced at the start of the course.

In our project, we will be using ultrasonic sensors for obstacle detection. We reviewed the safety guidelines and precautions enlisted by Omron Industrial Automation [5]. As stipulated by these guidelines, we will ensure that the product and the associated ultrasonic sensors are not used at an operational temperature of greater than 70^{o} C, which is the upper limit of operational temperature for ultrasonic sensors as mentioned in the article “*Ultrasonic Sensors Knowledge (Part 4): Influences on Measurement Accuracy*” [6]. We will also ensure that we do not use these ultrasonic sensors near any air nozzles (which contain multiple frequency components), that have been found to negatively impact the operation of ultrasonic sensors. In order to protect the ultrasonic sensors from water droplets, we will be using plastic encasing to protect the ultrasonic sensors. Furthermore, we do not plan to use the ultrasonic sensors in low temperatures less than $0^{o}C$, because the vinyl cables associated with the ultrasonic sensors are found to bend and break in these conditions.

Moreover, exposing the PCB to water or rain could cause a short-circuit and lead to operational failure, so we will make sure that we do not expose the PCB to water and we thus plan to protect the PCB using a plastic encasing similar to the ultrasonic sensor. In addition, we will also always make certain that we will be using the different electronic components in our system within their operational usage limits including but not limited to voltage, current, temperature and humidity limits.

We have also carefully reviewed the IEEE [3] as well as the ACM Code of Ethics [4] and in the following section of our Project Proposal, we will briefly discuss some of the codes that are relevant to our project and how we plan to go about upholding and abiding by these guidelines to the best of our abilities. Starting with points 1. and 9. of the IEEE Code of Ethics [3] and point 1.2 of the ACM Code of Ethics [4], we will ensure that in designing, implementing, experimentation and testing of our product, we will hold the safety, health and well-being of the general public to the highest order and we will ensure that if and when someone tests or uses our product we will do so only after rigorous and thorough testing, and only if we deem it completely safe to use to the best of our knowledge.

In compliance with point 2. of the IEEE Code of Ethics [3] and point 3. of the ACM Code of Ethics [4], we will ensure that all of the results, estimates and decisions that we present over the course of the development of our project, will be based on data that we collect during the design, implementation and testing phase. All the estimates and results that we present and all the decisions that we make will always be backed by genuine data, that we either obtain from trusted external sources or through the data that we ourselves collect, understand and analyze. We vow to not fabricate results, and we promise to be as open, honest and trustworthy as we possibly can.

We have taken up this project because we are genuinely intrigued by the scope of our project, the impact that technology can have in order to make a difference in people’s lives and to build up our technical knowledge and competence. We vow that the actions of our group will always be guided by honest and genuine intentions, and in this spirit, we vow to abide by point 5 of the IEEE Code of Ethics [3] and points 1.5 and 2.2 of the ACM Code of Ethics [4].

Our group is committed to treating all people fairly and we vow to not discriminate based on including but not limited to race, gender, sexual orientation, age, religion, disability and nationality. We will always celebrate diversity and inclusivity in the work that we do and will fully abide by point 8. of the IEEE Code of Ethics [3] and point 1.4 of the ACM Code of Ethics [4].

We will always hold the privacy of our potential users to the highest regard and we will ensure that the project that we build is in no way collecting unauthorized data from the users. In our project we will be collecting information about the user's height in order to estimate the stride length of the user, but this will be done only on conditional terms if the user wishes to provide this data. We will have a default height parameter that we will manually enter into our product in the case that the user wishes to not provide this data. We also vow that the user will 'own' this provided data in all forms and we shall not disclose this information to any third-party sources. We will also completely abide by point 1.6 in the ACM Code of Ethics [4].

Lastly, throughout the semester we will be reviewing previously published literature and will be analyzing their results, observations and conclusions in order to guide our project. We vow to properly credit other people's work and cite this work in an appropriate format in the reports that we submit. We will hold to highest regard, point 7 of the IEEE Code of Ethics [3] and point 1.5 of the ACM Code of Ethics [4].

3.4 Future Work /Improvements:

Based on how well our solution fared against the previous solution in the areas of range, cost, feedback, usability, adaptability, durability, we concluded that our solution needs to be more adaptable and durable.

Additionally, we will need to keep refining our noise filtration techniques once we start testing. Many advanced filtration algorithms exist and newer ones are being designed every few years. By experimenting with a variety of filtration algorithms, we can gauge what works best in what circumstance. Additionally, we can more accurately detect obstacles by using multiple sensors to corroborate each other's outputs.

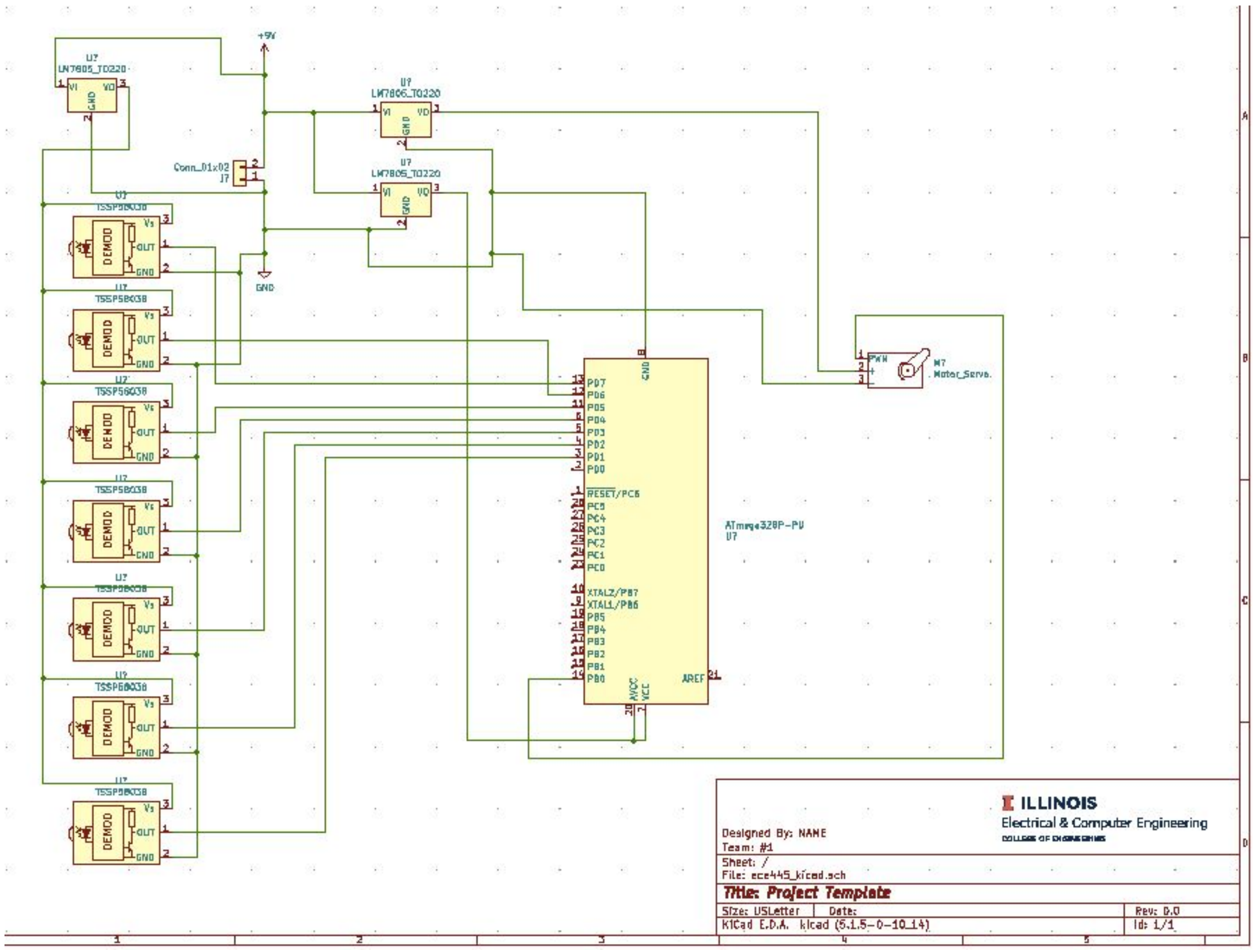
Exhaustive testing will help us make our solution more adaptable because as newer and newer shortcomings come to light, we will modify our solution to make it better and better. Similarly, durability can be improved by trying out multiple materials for the mounting structure and analysing the extent to which wear and tear occurs over time.

We can come up with encasings for our system that meets IP Code standards for waterproofing. Whether our current method of detecting moisture is the best possible or not remains to be seen, We must assume that there is a way to detect moisture more accurately but at a lower cost. Coming up with an alternative approach to detect wetness of surfaces would save cost (Fluid Detector LM1830 chips cost $25 each).

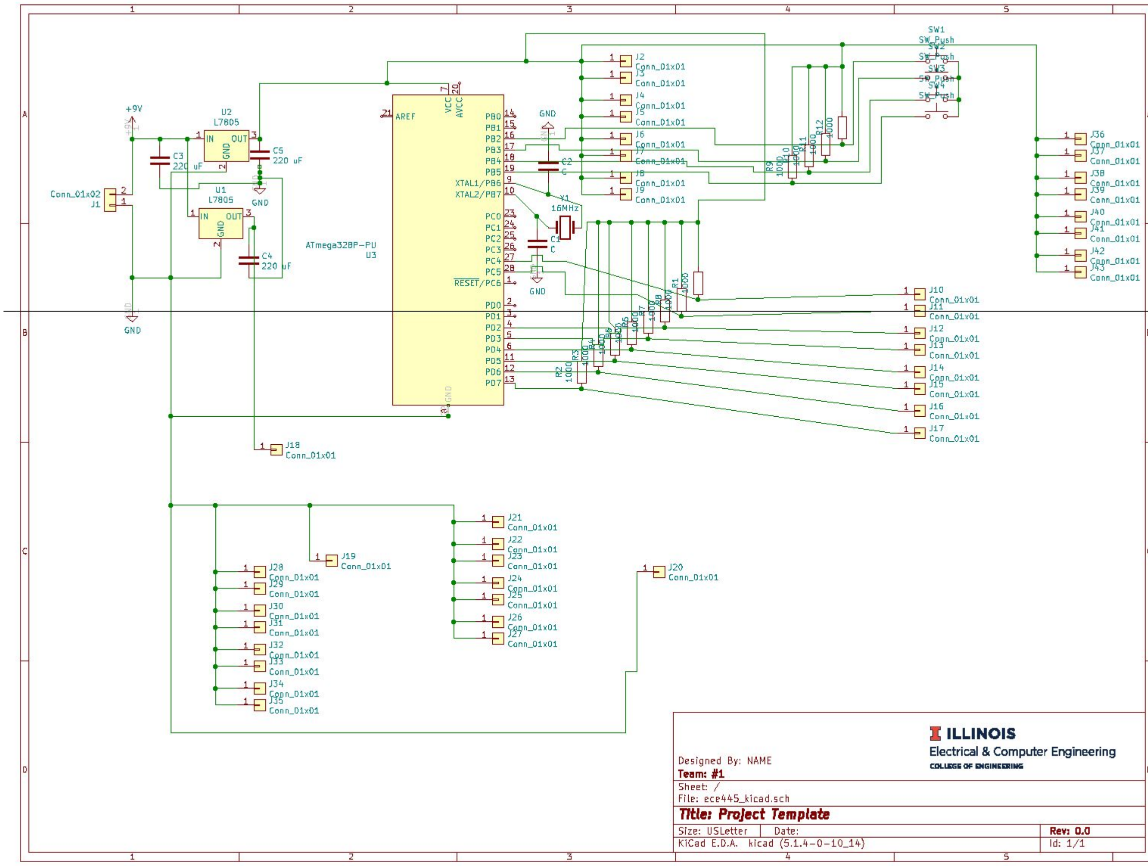
While the range of our sensor was better than the previous solution, coming up with a method to improve our long range obstacle detection capabilities (current solution accurate upto 2.5m at max) is definitely a goal that we should strive to meet. We may be able to amplify attenuated reflected sound waves, and this will help us in more ways than one. Overall, our solution accuracy will increase and we might be able to mitigate the risk of not detecting acoustically soft substances.

4. Extra Credit

During the first design review, the PCB design schematic that we submitted was the following:



After the design review, we spoke to the TAs, did additional research on various aspects of the pcb design and made various changes. The updated PCB design is below



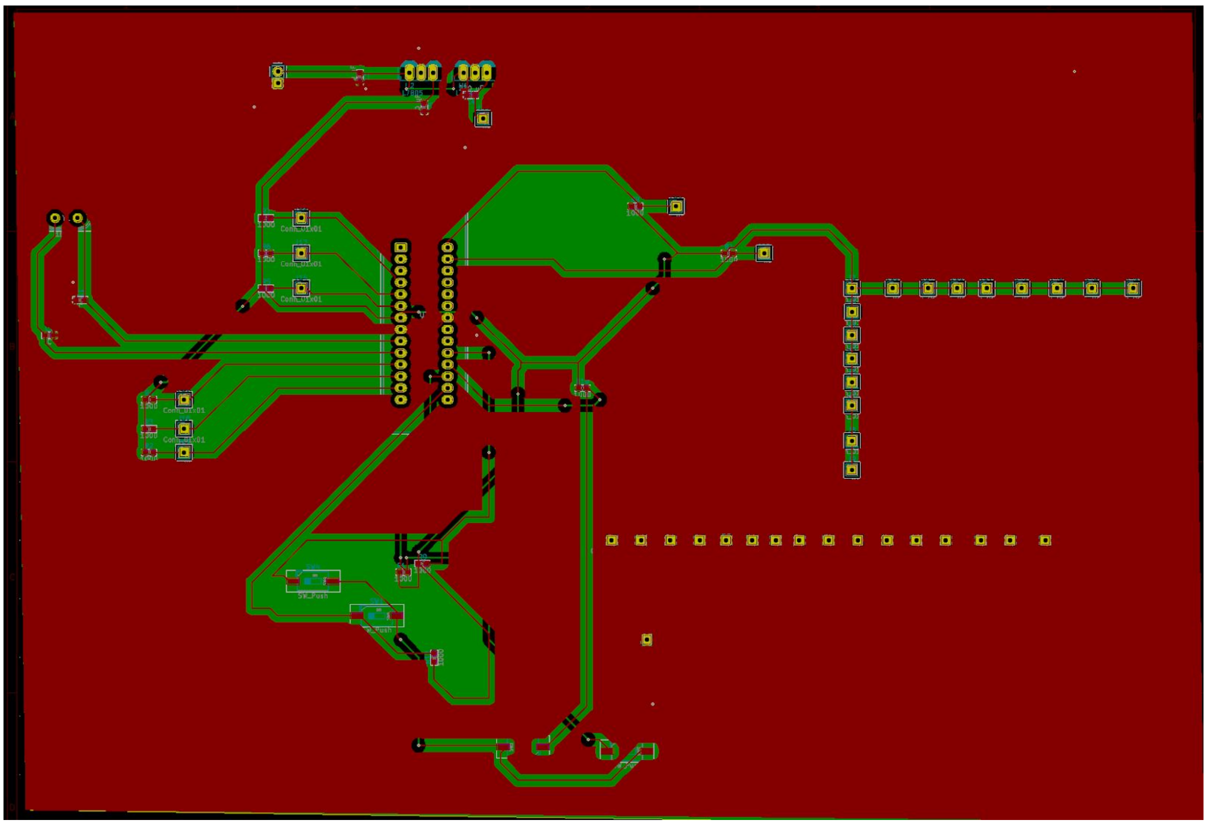
We have made significant changes since then in order to make sure that if our PCB was manufactured, that it would work seamlessly. Our first PCB design draft did not have the following components:

1. Output filter capacitors: Since our first PCB design did not have capacitors placed at the output of the voltage regulator, the voltage ripples in the circuit would have damaged our sensors.
2. Pull-up resistors: In the first design, we directly connected the ultrasonic sensor output pins to the corresponding pins on the Atmega chip. This was a mistake. We needed to implement pull-up resistors in order to make sure that circuit worked as intended. We confirmed our design with a TA during office hours to make sure.
3. 16 MHz crystal oscillator: The ATmega P IC chip works without an external oscillator but after extensive consultation with TAs, and external research, we found that it is highly recommended that an external 16MHz crystal oscillator be used for optimal performance.
4. Switches : In the first design, we did not implement the knob aspect. In the second iteration, we implemented the knob using 4 push button switches, which combined would represent $2^4 = 16$ options. We also added LEDs so that we can externally check if at any given point in time, a switch is on or off.

5. Connectors: In the first design, we had not added enough connectors. These are important because they would enable us to probe different points on the circuit apart from just being able to supply power and transmit output messages.

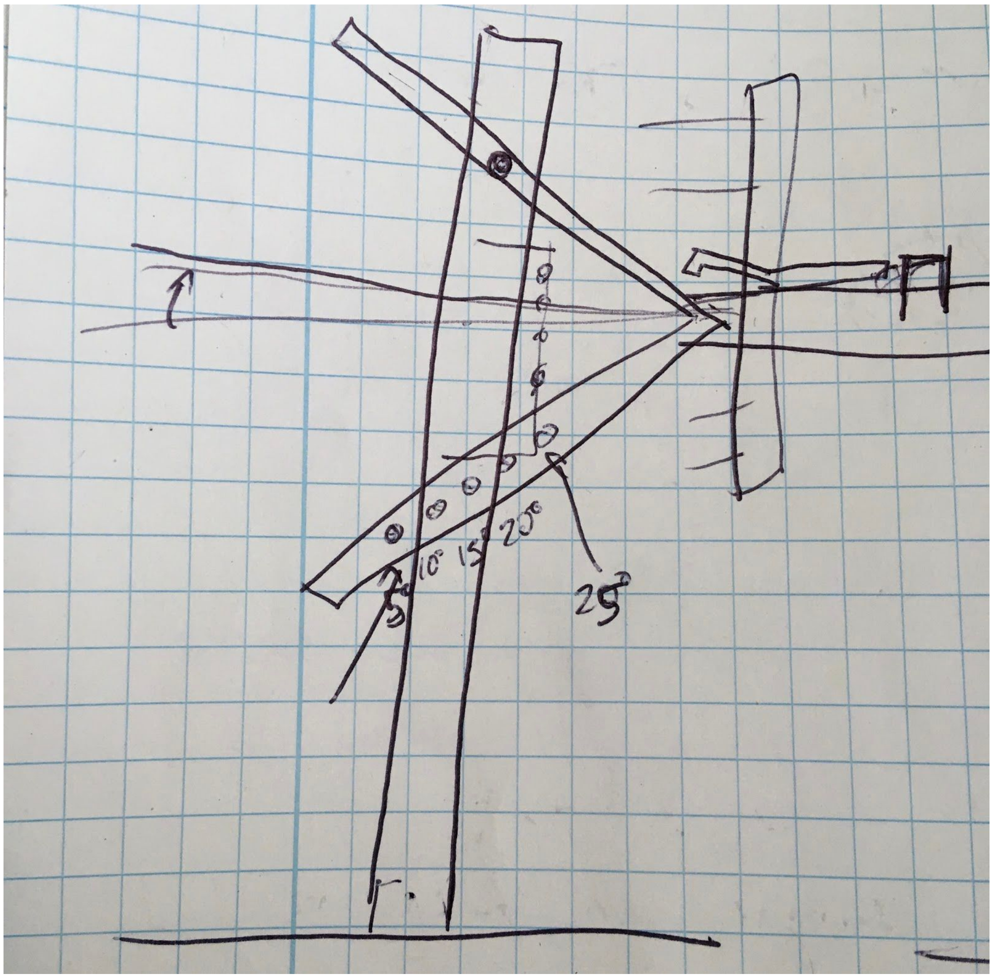
Once we finished this schematic, we verified it with a TA during office hours and after analysing our PCB design, he confirmed that everything seemed to be in order.

PCB layout



Machine shop conversations:

Even after the design review, we spoke regularly to the machine shop and had finalized our design with them. They had already confirmed the dimensions of all of our components and the drawings below illustrate this.

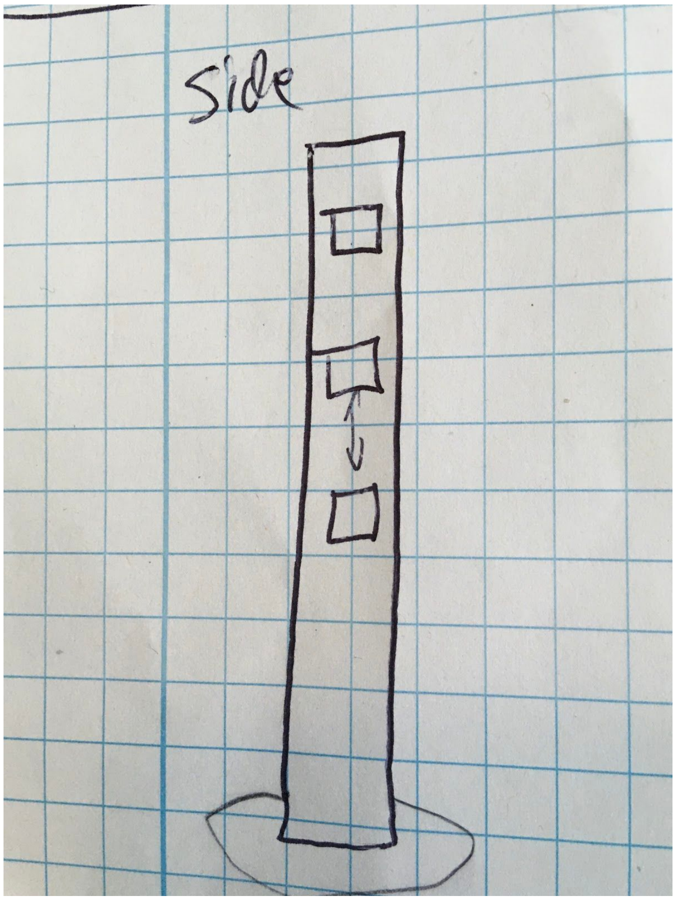


The tallest stick-like structure represents the wooden structure that contained different slots into which the V-shaped structure would lock. By choosing the slot appropriately, the launch angle could be controlled. This was part of the original setup.

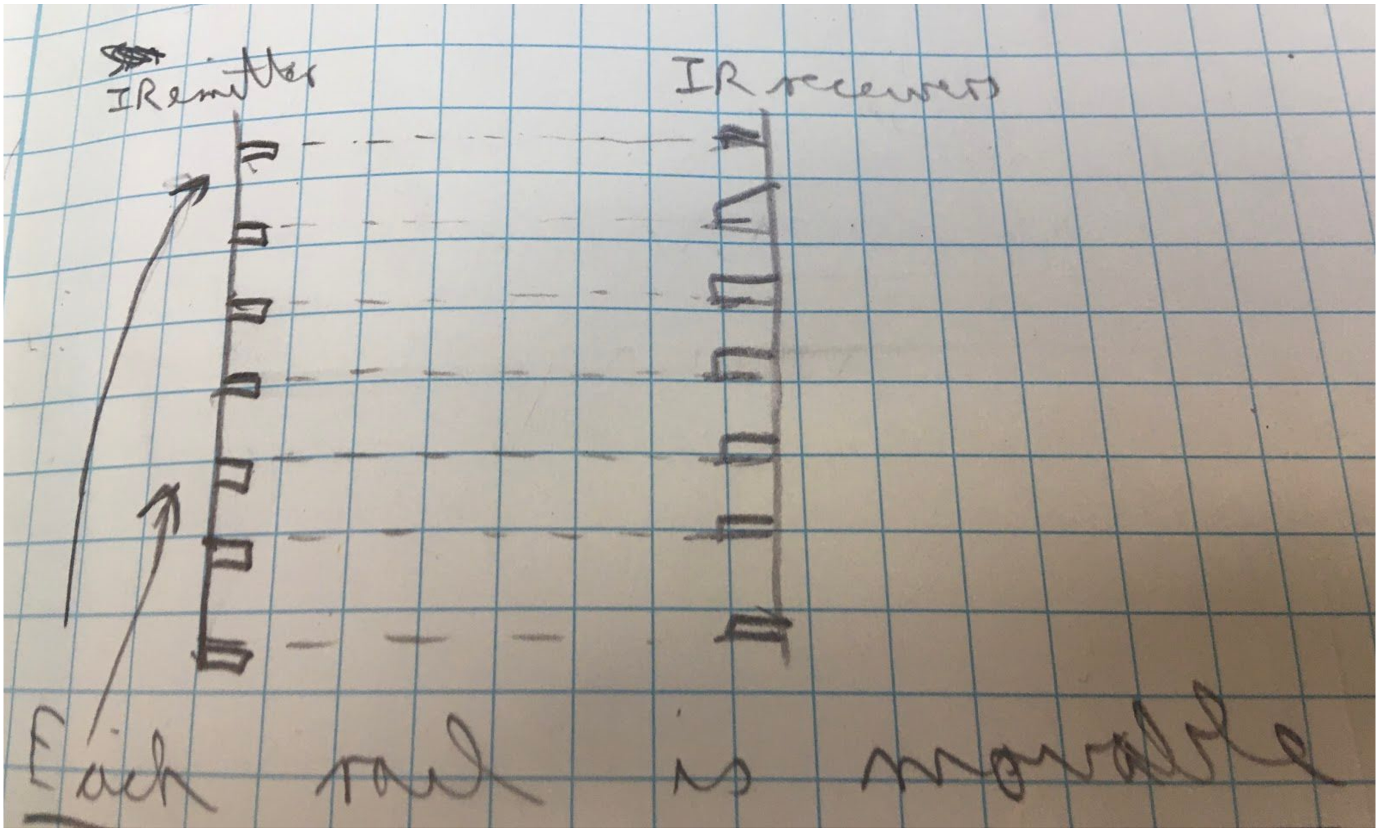
The rod to the right represents one of the bars that would contain 7 moveable rails on which IR sensor emitters can be placed, and the other side would contain a second rod (not shown in the above image), which would contain a similar rod with 7 moveable rails, each of which would hold one IR receiver .

These rods would be designed by the machine shop, and connected to each other such that their point of support would rest on the horizontal wooden structure as shown above.

The side profile of a given rod is shown below



Both these rods would be on either side of the bow-like structure that launches the bat-bot, and they would rest on the bow itself.



The diagram above shows how the emitters and receivers face each other (on either side of the bow-like structure described earlier). Since the rails were designed to be movable on either side, we could ensure through exhaustive testing that the IR emitter and receiver were exactly in line with each other. From a programming stand-point, we would just power on each emitter individually and then calibrate each receiver such that it was able to pick up the signal from the corresponding emitter. Once we had calibrated all the sensor positions, we would be ready to place the bat-bot in position and start testing.

We had accomplished the following after the design review, and before the spring break, and we were ready to start the next phase of our project

References :

[1] S. Y. Kim, "Usability and Design Guidelines of Smart Canes for Users with Visual Impairments," *International Journal of Design*, 2013.

[2] M. Dolan, "How to Measure Stride Length," wikiHow, 12-Mar-2020. [Online]. Available: https://www.wikihow.com/Measure-Stride-Length. [Accessed: 04-Apr-2020].

[3] “IEEE Code of Ethics,” *IEEE*. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 13-Feb-2020].

[4] “The Code affirms an obligation of computing professionals to use their skills for the benefit of society.,” *Code of Ethics*. [Online]. Available: https://www.acm.org/code-of-ethics. [Accessed: 13-Feb-2020].

[5] “Safety Precautions of Ultrasonic Sensors Cautions for Ultrasonic Sensors: OMRON Industrial Automation,” *OMRON*. [Online]. Available: http://www.ia.omron.com/product/cautions/50/safety_precautions.html. [Accessed: 04-Apr-2020].

[6] Fuchs, “Ultrasonic Sensors Knowledge (Part 4): Influences on Measurement Accuracy,” *Pepperl Fuchs*, 20-Jul-2018. [Online]. Available: https://www.pepperl-fuchs.com/usa/en/25518.htm. [Accessed: 04-Apr-2020].

[7] Shawn and Shawn, “Types of Distance Sensor and how to select one?,” *Seeed Enables Your Hardware Innovation Needs*, 01-Oct-1969. [Online]. Available: https://www.seeedstudio.com/blog/2019/12/23/distance-sensors-types-and-selection-guide/. [Accessed: 04-Apr-2020].

[8] “ECE 445 Spring 2019 Team 18 OptiCane Final Paper.”

[9] Adafruit Industries, “Adafruit VL53L0X Time of Flight Distance Sensor - ~30 to 1000mm,” adafruit industries blog RSS. [Online]. Available: https://www.adafruit.com/product/3317. [Accessed: 18-Apr-2020].

[10] Adafruit Industries, “HC-SR04 Ultrasonic Sonar Distance Sensor 2 x 10K resistors,” adafruit industries blog RSS. [Online]. Available: https://www.adafruit.com/product/3942. [Accessed: 18-Apr-2020].

[11] “IP Rating Chart,” DSMT.com. [Online]. Available: http://www.dsmt.com/resources/ip-rating-chart/. [Accessed: 18-Apr-2020].

[12]“The Scientist and Engineer's Guide to Digital Signal Processing Moving Average Filters Available: https://www.analog.com/media/en/technical-documentation/dsp-book/dsp_book_Ch15.pdf.”

[13] “Block Convolution using Overlap Add Method,” *Block Convolution using Overlap Add Method - File Exchange - MATLAB Central*. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/41338-block-convolution-using-overlap-add-method. [Accessed: 09-May-2020].

[14] “Moving average or running mean,” *Stack Overflow*, 01-Aug-1962. [Online]. Available: https://stackoverflow.com/questions/13728392/moving-average-or-running-mean. [Accessed: 09-May-2020].