1. *Introduction and Background:*

Radar means "radio detection and ranging," at the same time, which means finding targets by radio and determining their location in space. For this reason, radar is also known as "radio location". Radar is an electronic device that uses electromagnetic waves to detect targets. Radar emits electromagnetic wave to illuminate the target and receive its echo, obtain the distance, range rate (radial velocity), azimuth, altitude and other information from the target to the electromagnetic wave launching point.

Why do we need radar? Radar came into being because during the war between Britain and Germany during world war I, Britain needed a kind of radar that could detect metal objects in the air to help search for German aircraft in the anti-air war. During the second world war, radar technologies such as surface-to-air, air-to-ground (search) bombing, air-to-air (interception) fire control and identification friend or foe were developed.

2. *Briefly Experimental Design:*

By combining Hardware (Raspberry Pi + Matrix Voice) and Software (Python in Linux), at the same time, generating our own pulse train, to simulate how exactly a radar works.

3. *Materials and Methods:*

This experiment included two parts: Hardware and Software. For Hardware, we use the Raspberry Pi with Matrix Voice to receive the sound or our pulse train. For Software, we use Python in Linux system to do the coding step in order to let our Radar works well.

First, we want to generate our own pulse train with PRI 0.15s, PW 2.0ms, and fc 5.0kHz. Then we play our own pulse train through our pc. Next, we decide one suitable distance to record this pulse train using the Matrix Board. After recording these data, then we gate them for SNR gain. According to different distance, we can detect different consequences of our pulse train. Thus, we can estimate the distance through our data. Then, in order to observe them exactly, we also add envelope detection. At the same time, we have tried FM chirp to add in our pulse train. Finally, we are trying to add Cross Correlation or Matched Filter.

Therefore, during this experiment, we have several steps to achieve our goal.

We generate the pulse train, record the pulse train using the Matrix Board, gating the recording for SNR gain, and estimate the distance, adding envelope detection and do the FM chirp.

4. *Experimental Results:*

Without Filter and Cross-correlation

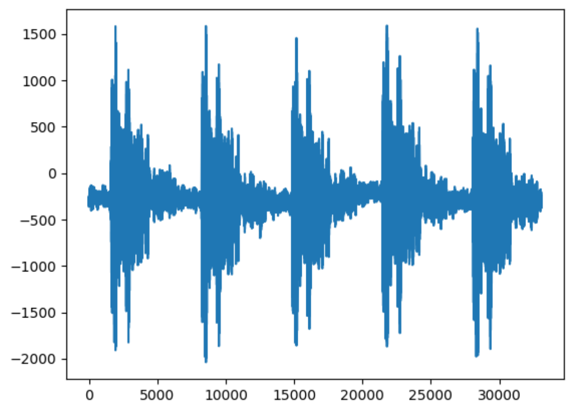


Fig1. Pulse Train

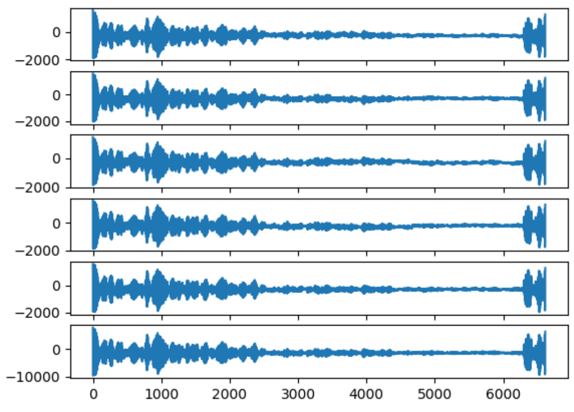


Fig2. Gating

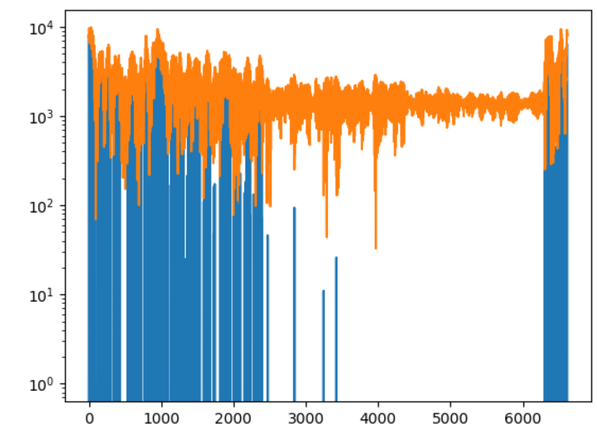


Fig3. Envelope Detection

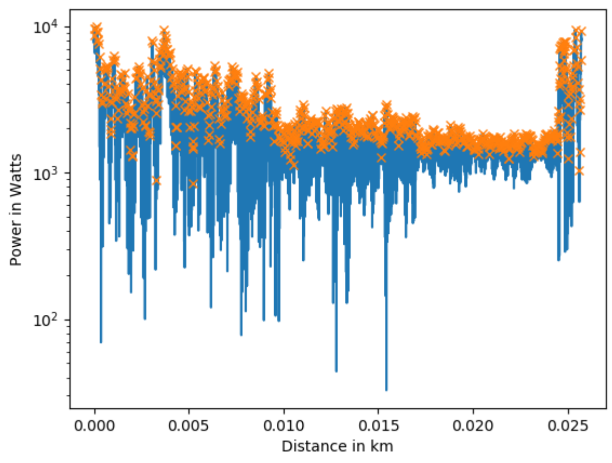


Fig4. Find peaks to determine the distance

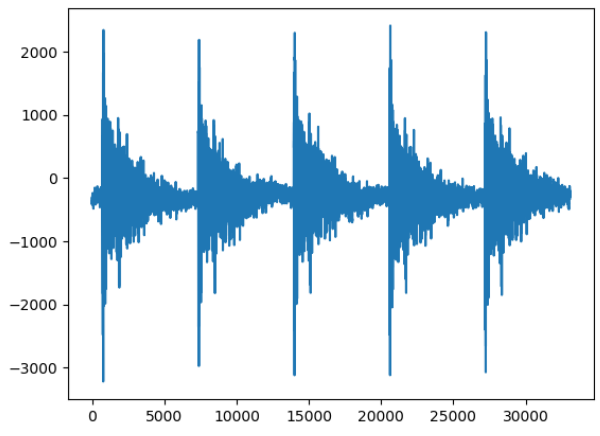


Fig5. Adding chirp with Pulse train

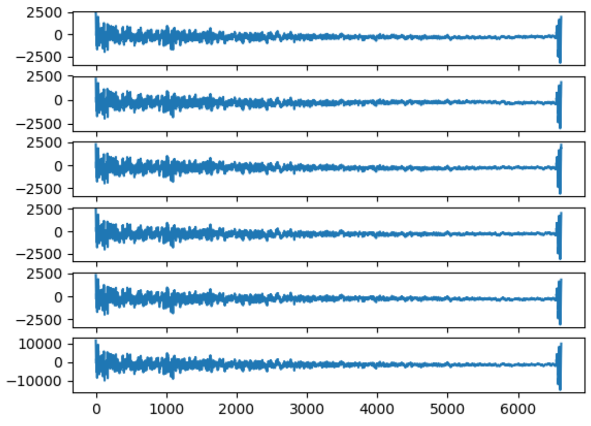


Fig6. Gating chirp

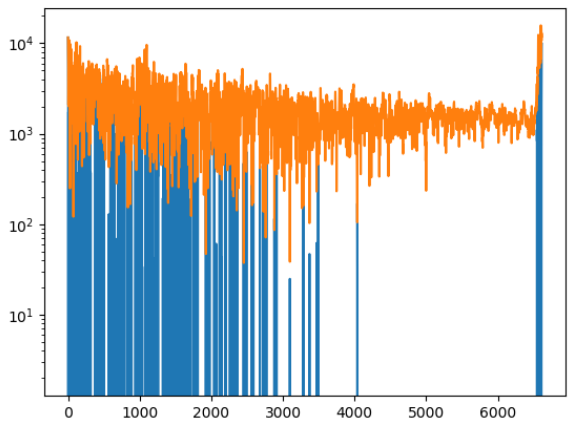


Fig7. Envelope detection of chirp

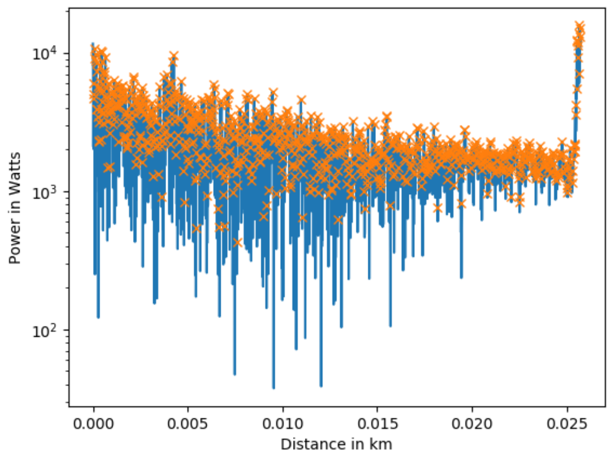


Fig8. Find peaks of chirp to determine the distance

5. *Discussions:*

According to our experimental results, we have achieved one goal that using Matrix Voice to detect signal, collect data, then analyze data to determine the distance. First, we want to collect more than 5 observations. According to Fig1 and Fig5, there are 5 observations to show our generated pulse train which is received by Matrix Voice. Figure 1 is our initial pulse train, but Figure 5 is added chirp in pulse train. For chirped pulse train, we set up this frequency of chirp: from 4000 Hz to 6000Hz. In order to measure the long-distance and time-preserving resolution, radars need a short period of impulse wave but a continuous transmission signal. The chirped signal (Figure 5) can retain the characteristics of continuous signal and pulse at the same time, so it looked more linear to observe.

For Figure 2 and Figure 6, one is gating our pulse train and another is gating chirped pulse train. We show 6 number of rows for each gating. 1st – 5th rows are our 5 observations by using generated pulse train. The last row, the 6th row is the summary of all five rows. Why do we want to do this? After adding them together, they will show us a more apparent observation. At the same time, using chirped pulse train looks more linear than pulse train without chirp.

Signal's value has become positive through hilbert envelop detection. Because the square of its real number and the square of its imaginary number add up to take the root. In this way we can clearly see the change in signal amplitude or energy. This makes it easier to observe the threshold of the image. Then you can use the find peaks function to easily find the highest peak. The sampling point corresponding to the x axis is the sampling point we need to find.

In order to get a clearer sampling point of what we want. We set a kernel frequency from 4000 to 6000, and read the array of unit kernel. The fc is 5000hz. We performed cross correlation between enveloped filtered(3800-6200) gating signal and kernel, and got the highest peak we wanted. When the two most similar signals pass, it produces the largest peak. Its corresponding x-axis value is the sampling point we want.

Next, we decide to add envelope detection. Envelope detection is electronic circuit with high frequency signal as the input signal and provides the envelope of the original signal. By observing the envelope detection of pulse train, X-axis gyroscope means samples. Figure 3 and Figure 6 (chirped) help us explain more.

Finally, find peaks to determine the distance. Figure 4 and Figure 8 help us understand how radar works well. Finding the peak value is in order to see X-axis gyroscope. X-axis gyroscope in Figure 4 and Figure 8 mean Distance in km. How can we get this distance? In other words, why can we use distance in km as our X-axis gyroscope? The reason is that we use one radar equation. The equation is:

$$R(unambiguous) = \frac{1}{2} * \left[343\left(\frac{m}{s}\right) * t(s)\right]$$

By observing Figure 4 and Figure 8, we can easily notice that the distance is 0.0055 km. That means the distance is 5.5 m.

6. ***Improve our results:***

In order to improve our detected results, we finally decide to add Filter and Cross-Correlation. We set up the bandpass filter: from 3800 Hz to 6200 Hz. Now, we will show these new improved results.

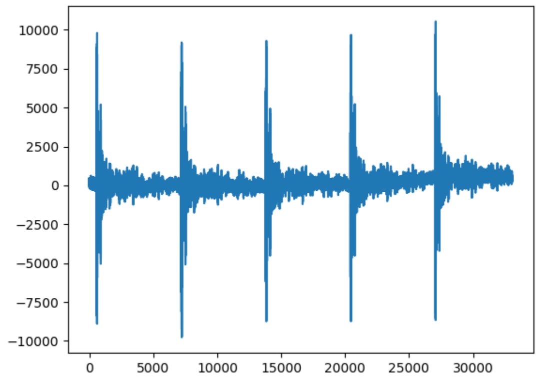


Figure 9. Pulse Train

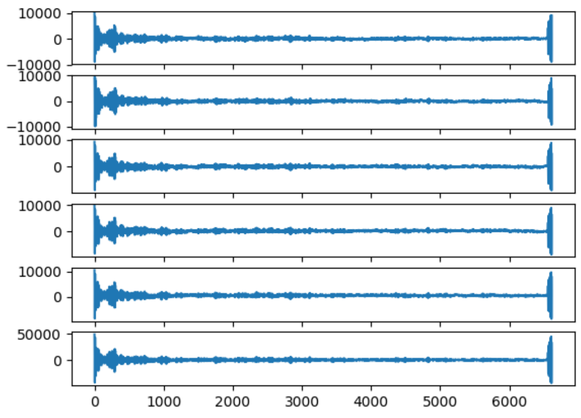


Figure 10. Gating

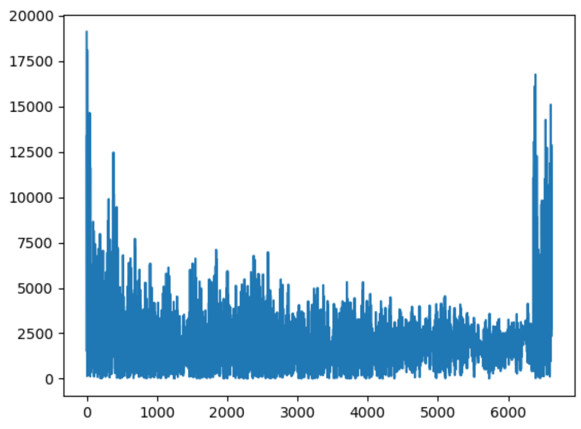


Figure 11. Get only positively of our pulse train

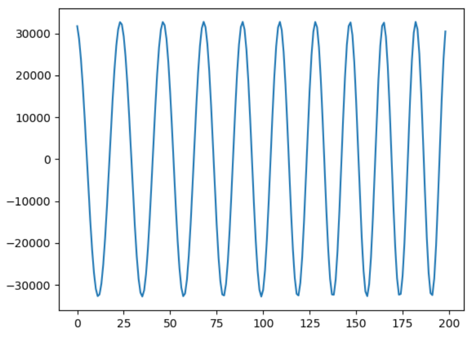


Figure 12. Adding chirp (set up frequency)

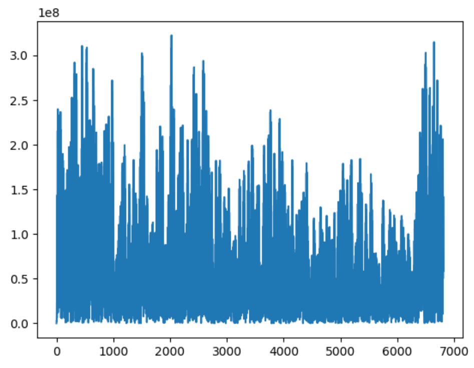


Figure 13. Adding Cross – correlation

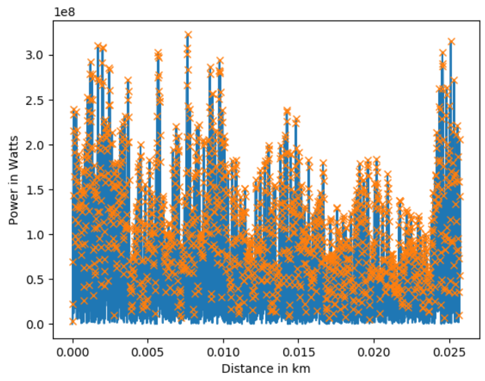


Figure 14. Find peaks to determine distance
(Distance is 7.5 meters which is also approximately actual distance)

7. *Summary:*

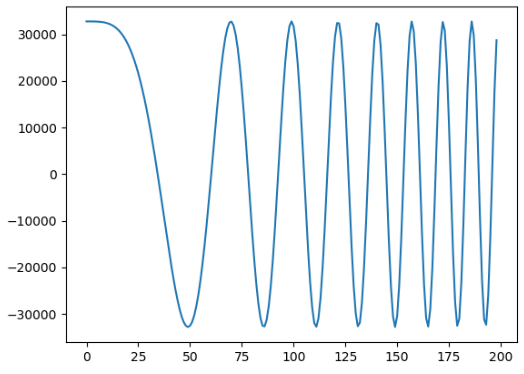
After finishing this experiment, we have learned a lot of things. First of all, communicating is very important in class or in one group. Why do we mention it? For example, sometimes we want to express our thought in the class, but we always express it wrongly. In fact, bad communicating with others causes lots of problems. Therefore, we start to learn to communicate with others in order to improve our spoken skills.

For technical skills, we have learned a lot of Radar knowledge. At the same time, we have abilities to deal with some coding issues now. Before taking this Radar & Sonar course, we are a little bit afraid of writing code, especially using Python. Because this is our first time to try writing Python code, we think it’s very difficult to write code. However, we have abilities to write Python code after spending lots of time on it. Of course, we also meet many questions during writing code. For example, we don’t know how to connect WiFi for our Raspberry Pi, how to roll gating, and how to build chirp. Fortunately, we can ask help from Professor or discuss these questions on Slack.

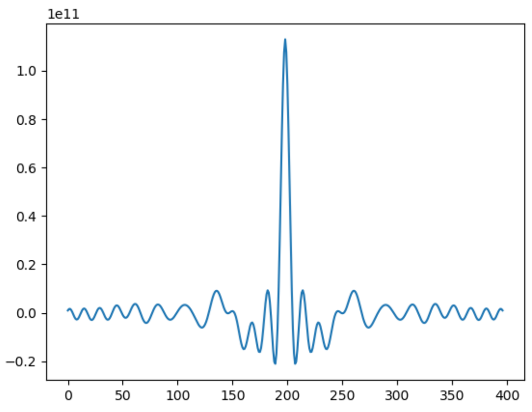
8. *Citations and files:*

Generate and write pulse train wav file
Pythoncollectdata.py
RadarSimv1.py
XcorrlndexIntroduction.ipynb

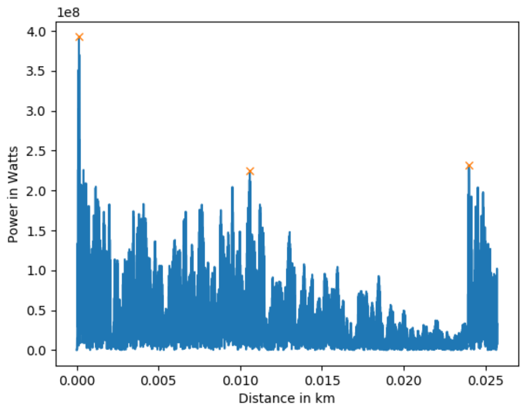
Finally, after demo and presentation, we decide to add some much more helpful pictures that we detect now. (we think it is more likely correct after demo).



FM chirp (0 – 8kHz)



Cross – correlation (Kernel itself)



8 m (distance from sound)