Report Date: 18/11/2022
To: ematson@purdue.edu, ahsmith@purdue.edu, lee3450@purdue.edu
From: Team Coyote(Sensors & Network)

- Hyemin Lim (freemini2@cau.ac.kr)
- Nayoun Kim (202010766@live.wsu.ac.kr)
- Jaehui Boo (32192075@dankook.ac.kr)
- Hyeongjun Kim (aa980305@cu.ac.kr)

Summary

The localization algorithm experiment was going on this entire week. First, the experiment environment was set at the conference room located at the 1st floor of KSW. The experiment includes three Raspberry Pi each connected with USB microphone and Heltec ESP32 as an end node. These end node will detect a sound and send a timestamp of its detection time. With those timestamps, the difference between them is put into the code to calculate location.

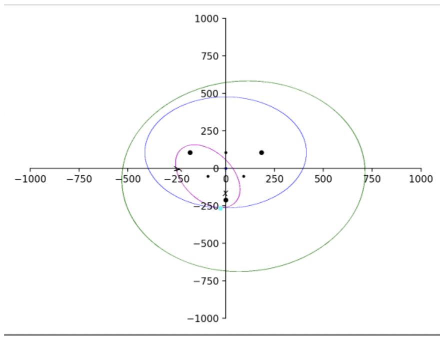

Fig 1. the result of the first experiment

However, the predicted result which is three hyperbolas intersecting each other was not shown as a result. Instead, ellipses are keep showing up. It was because the relationship between ellipses and hyperbolas. Both of them are Conic Section, which comprises ellipse, hyperbola, circle, parabola. There is a value called eccentricity, it means the distance between any given point on a conic section and a focus point divided by perpendicular distance from that point to the nearest directrix.[1] If eccentricity is over 1, it makes hyperbola. If it is between 0 and 1, it makes an ellipse. The team assumed it was the reason.

$$e = \frac{2r}{v\Delta t}$$

Eccentricity is represented in our project as the equation above. r is the half of distance between sensors, v is the speed of sound, and Δt is the time difference of each sensor's signal receival. It has to be over 1, so it can represent a hyperbola as the algorithm intended. To do that, the team did some experiments.

The only element that could be changed was r, the distance between each sensors. If r is too short, the maximum time difference that match the eccentricity condition was too small.(i.e. 0.xxxx milliseconds) And r can't be too long due to constraints of the indoor experiment.

Then to reduce Δt, the team tried to stabilize the delay between microphone detection and the sound source. By recording and detecting certain sound played repeatedly, the average latency of microphone can be calculated. All three of microphones have approximately 36 ms of time delay. However, the difference of delay between microphones were not significant. Therefore, the delay was not the reason of malfunction of algorithm.

For Unity development, mapping coordinates sent by http request from Node js server has been succeeded. There was some issues regarding laptop and it is ongoing issue.

What Coyote Team completed this week:

- Initial Experiment of localization
- change r (distance between sensors) experiment
- Average microphone delay estimation experiment
- Built Node.js server for visualization platform & Sending data to Unity(frontend)
- Mapping sent coordinates on Unity map
- Receive real time Coyote location data using Websocket in Unity
- Organize Unity code, such as turning singleton variables into arrays
- Implement Push Notification code

Things to do by next week

- Increase accuracy of the localization algorithm
- [Unity]Develop real time mapping of coyote coordinates
- [Unity]Add camera touch functionality
- Write paper
- Prepare for the final presentation
- New memory leak problem solution
- Implement Unity code that moves the camera view whenever users touch
- Write Unity code that moves to the mapped camera view whenever real-time mapping is detected
- Test after writing a code that maps the real-time coyote location whenever it is detected

Problems or challenges

- Localization accuracy
- Unity Memory leak

References

[1] Openstax. "Conic sections" Calculus Vol.2. Accessed: Dec, 02, 2022 [online]. Available: https://openstax.org/books/calculus-volume-2/pages/7-5-conic-sections