Rhea Kudtarkar
Shiva Nathan
Udit Ranasaria
18-500: ECE Design Experience
Refocus SOW

This week made it clear that our plan cannot rely on hardware getting delivered and built mostly because COVID-19 has disrupted shipping of parts and access to necessary equipment. Therefore, we have redesigned our scope of work to remove reliance on hardware by programming a hardware simulator. This software will simulate tags on a basketball court, generating distances between simulated devices to simulate time taken for signals to travel between devices. The central server will receive (tag_id, anchor_id, timestamp_ns) tuples just as it would have with the real hardware.

Assuming this hardware simulator, we can write real time localization software (RTLS) that implements multilateration to calculate the simulated tags' locations. We will compare the RTLS' calculated locations to the actual hardware-simulated locations to assess the RTLS' accuracy. There are several simple and effective multilateration algorithms that we can implement for an ideal tag system -- that is, a system which assumes perfectly accurate timestamps.

After implementing the ideal algorithms, we will add artificial error into the simulated timestamps, and implement multilateration algorithms that can account for such drift. The localization programmer will not know how drift and inaccuracy is added to the timestamps. We hope that these tests will give a better estimate of the precision of our hardware localization, versus the original benchmarks that we developed which did not account for probability error correction in the software.

After we build a simulator and error-correcting localization software, we will assess the feasibility of implementing a real hardware setup. If we do not feel that we can build our system given the remaining time, materials, and requirements, then we will push hardware out of the scope of this semester's work entirely. Instead, we will develop a sports analytics software stack that builds on the implemented RTLS software. We will also try to build a basketball video game -- different from the RTLS hardware simulator -- that generates data to test the analytics stack.

To recap, we will be pushing hardware out of our scope. Instead, we will build a hardware simulator that interfaces with our localization software. The localization software MVP will assume perfect data, but then we will implement a revised version that can account for inaccurate timestamps. Finally, we will measure and test our performance by comparing simulated locations with our localization calculated locations. This will provide a clear requirement of the required hardware accuracy, which we will use to reassess the feasibility of implementing physical hardware. If we decide that we cannot implement a physical hardware stack within the remaining time and material constraints, then we will instead build analytics software and a "virtual basketball" game to test that software.

Outline

- Conclusion that testing and working on hardware components will be delayed and significantly hindered, so we are shifting our focus away from that but not removing it entirely
- Build a TDoA anchor+tag system simulator to feed timestamped data into the central server
- Write the multi-lateration software to turn the timestamped pulses into actual locations
    - Compare to simulated locations
    - Ensure there is a layer of abstraction between simulator coder and localizer coder to ensure no extra assumptions are made
- Add some natural, random drifts and imperfections todsimulated timestamps and use probabilistic modeling to improve our localization software
    - Use this push and pull to derive the exact drift inaccuracies that can be afforded by our hardware system
- If at this time hardware is more feasible to resume working on, build prototype hardware to achieve those requirements
- Otherwise, drop hardware and build a sports specific analytics ML software stack on the localized data
    - Maybe even make a multiplayer “virtual basketball” game to test our software on so we get non computer generated simulation.