UC San Diego
JACOBS SCHOOL OF ENGINEERING

PROJECT SPECIFICATION

For Collard Lizard Tracking

Joey Schnecker, Hassan Ahmad, Ryan Margraf, Vinit Saah

WES-207 Capstone Project

Project Charter

Project Overview

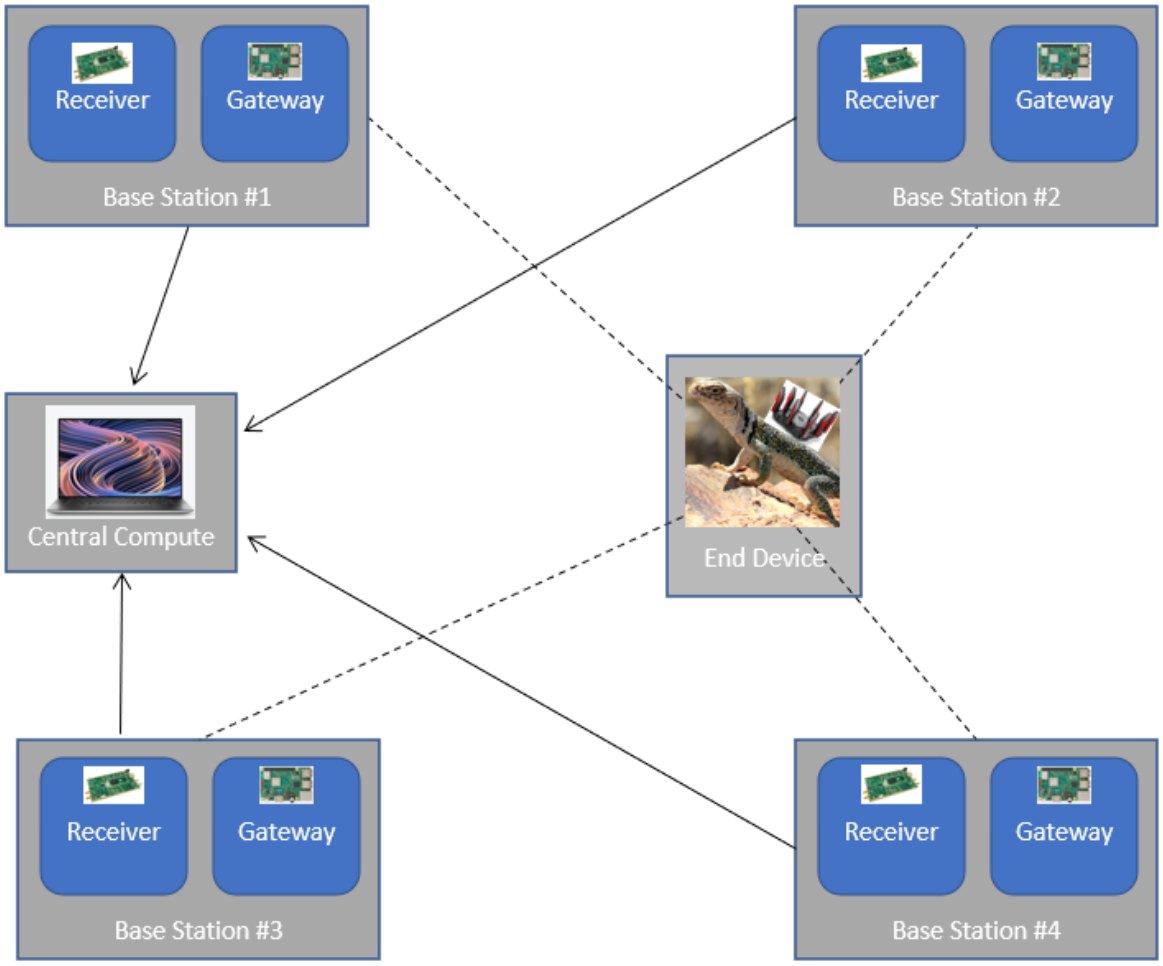
This project aims to build a system that would be used to track the position of collared lizards in Arkansas. The plan is to work on two subsystems and combine them in a final demo. Firstly, we will set up a lab bench with 2 USRP radios, and a single transmit antenna. The two USRP radios would be externally phase-synced initially using a common signal generator feeding two receivers and finally using GPSDOs. We will program the USRP radios to receive data from the transmitter, calculate the arrival time, and then send that information to a host computer. In parallel, we will develop a software application that takes the times of arrival as its input and estimates a position for the transmitter based on TDOA localization algorithms. For the final demo, we plan to connect the two subsystems to demonstrate the localization of a transmitter in a lab environment.

From a networking point of view, we will refer to the components of our lizard tracking system as follows:

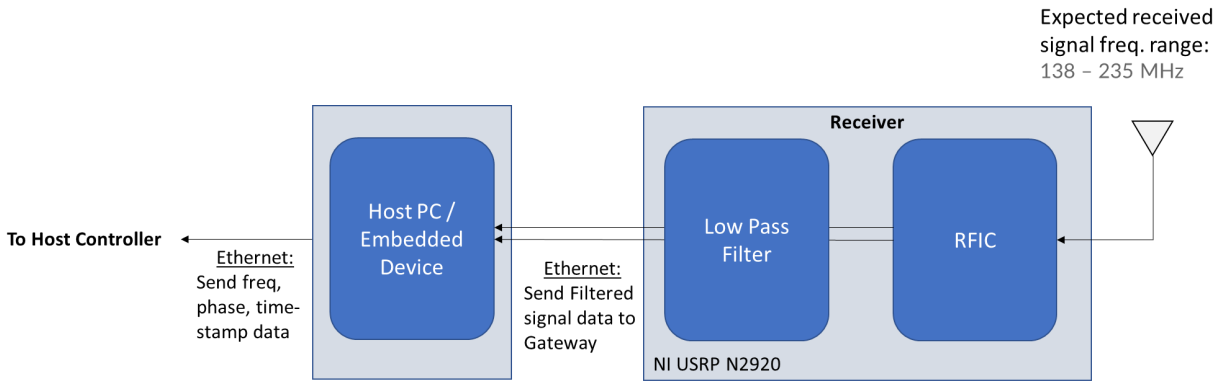
The Transmitter is the **End Device**.
The USRPs are the **Receivers**.
The Host Machines (Laptop or embedded device) are the **Gateways**.
The central computer controlling all the Host Machines is the **Host Controller**.

The diagram below shows how the tracking system components will interface.

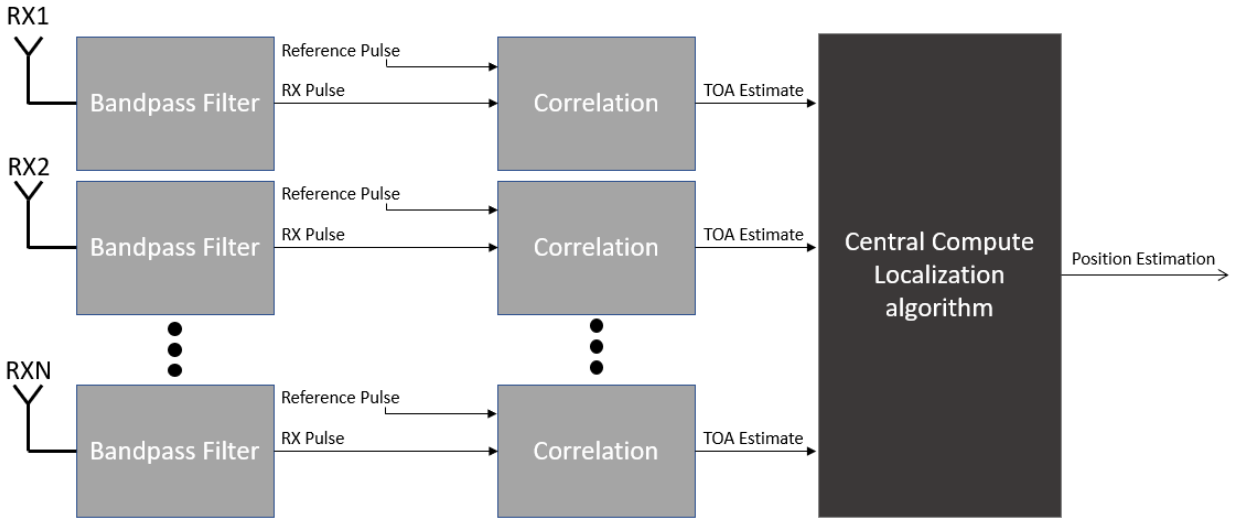
Tracking Network Block Diagram



Base Station Block Diagram



DSP Block Diagram



N >= 4 Base Stations needed for Position Calculation

Project Approach

Overall, the project is to be completed in 10 Weeks of time frame. This project aims to achieve its goals by dividing it into four sets of problems, solving them independently, and threading them together for the final demonstration. The four problem sets are establishing a test bench, solving the DSP, simulating the Localization and UI display of the result.
The problem sets are to be referred through the following TAGS.

TB : Test Bench
DSP : Digital Signal Processing
LOC : Localization
UI : User interface design
NET : Networking

TB_1: Interface a single computer with each USRP

- The first step is to set up the development environment by installing the USRP Hardware Driver (UHD) on Linux machines, (UBUNTU 20.04). Create a test app to initialize the USRP device, and set and get the sampling rate, center frequency, bandwidth, and antenna.
- Store the received samples with timestamp initialized to zero.

TB_2: Interface a single USRP with an external clock source

- Be able to interface with USRP using an external clock source and re-run the USRP test app.

TB_3: Phase synchronize two USRPs with a Signal Generator.

- Connect the USRP through a common clock source. The clock source could be a signal/waveform generator, or a GPSDO(if available).
- The AWG (Arbitrary Waveform Generator) outputs 10 MHz Sine/Square wave. Use tees to split the clock signal over and check output on oscilloscope over two channels to see the output waveforms are in-phase.
- Confirm the output from USRPs on two channels using the same oscilloscope.
- Through GPSDO, each USRP would lock into the GPS signal. We can target PPS for timing source and clock for 10MHz source. Use an oscilloscope to see whether the clocks and timing source are in sync.

TB_4: Test the TX radio and verify we can receive its signal on USRP.

- Run simple lab tests to make sure a USRP is receiving good data from the sample TX radio.
- Use an UHD test app to store the received signal on the host machine.
- Use MATLAB script to read the stored file and plot the time series and power spectrum to identify the pulses and identify the central frequency.

TB_5: Network the machines together

- Set a router to interface the Host and Host controller. Verify that all the networked machines are pingable.
- Copy the received signals from USRPS on host to the host controller.

DSP_1: MATLAB simulation to receive the transmitted waveforms.

- Use MATLAB simulation to create a transmitted pulse at around central frequency.
- Use MATLAB to simulate the received signal with AWGN.
- Use MATLAB to design a filter the desired frequency range and cross-correlate the signal and identify the central frequency.
- Plot the time series of the filtered signal and power spectrum to identify the central frequency.

DSP_2: C++ app using UHD to filter and cross-correlate the received signal.

- Use the filter coefficients from MATLAB in the UHD app to filter the received signal and store in the host with time stamp, receiver coordinates and IDs.
- Once able to do DSP over stored RF samples, use a multithreaded programming to receive and filter simultaneously and store the result for post processing.

DSP_3: Use Angle of arrival to fine tune the received signal timestamp.

- If time permits, evaluate the angle of arrival to improve the timing arrival estimates of the pulses, (Not in MVP)

DSP_4: Use GPSDO to synchronize the receivers and repeat the DSP.

- If USRP B200 arrives with GPSDO, perform the receiver synchronization using clock source and PPS.

LOC_1: Write a script to generate example data.

- Write a MATLAB script that simulates a single transmitter and 2-4 receivers.
- The script should output timestamps that can be used in localization applications.
- Should write the timestamps into a text file.

LOC_2: Write a script to estimate transmitter location.

- Write a MATLAB script that takes in 2-4 timestamps and outputs an estimated location or area based.
- Should read the timestamps from a text file and start by using data generated from the script in task 1.
- Create a plot showing the estimated location vs the known transmitter location.

LOC_3: Test Localization on real data from lab

- Get data from real tests in the lab and test the localization algorithm on real world data.

LOC_4: Port Critical Code to C++ application

- Port Localization math to run in C++.
- Port Reading from text file to C++.
- Write position estimation to a text file.

LOC_5: Demonstration of working system

- Demonstrate working localization algorithm.
- Get time-stamp data from real world lab measurements.
- Run a localization algorithm in C++ and write position estimation to a text file.
- Read the position estimations and create a MATLAB plot showing the results.

UI_1: Display the localized algorithm data as heat map or equivalent graphical representation.

- Add a bare minimum UI display of the localized data.
- The interface would feature an X-Y coordinate system with intersections of radius info for each receiver to pinpoint the locations of each transmitter.
- The UI would indicate the position of the transmitter and would refresh with updated information as the TDOA calculations are completed.

Minimum Viable Product

For this project the minimum viable product is to demonstrate the localization of a single transmitter in a lab setting. The MVP should have at least 2 USRPs synced directly with a signal generator. Ideally we would have a lab testbench with 4 receivers, but we may or may not have the necessary hardware in time, so the MVP is based on using the 2 USRPs we currently have.

First, we will get two SDRs synchronized in the lab with a signal generator or if possible GPSDOs. We can then work on DSP algorithms that use phase information to obtain accurate time of arrival measurements. The MVP deliverable for this portion of the project is to have each USRP receive the radio signal from a sample transmitter and output an estimated time of arrival for each SDR.

Next we need to set up a working test environment which includes a local network so that each base station can communicate their measurements with a central computer. In a production system the networking should be some form a low power wireless network, but this is out of scope for a single quarter. Instead the MVP will create a local network for all machines using a wireless router in the lab.

In parallel to this we will develop a localization simulation. In a production system, there will be a central computer that receives all the time of arrival data from each base station/gateway. The central computer calculates the position of the transmitter using time-difference of arrival algorithms. For the minimum viable product, we must simulate the production system. The MVP requirement for this task is to have a simulation that takes in 2-4 time of arrival measurements and produces a position estimation along with some GUI visualization for demonstration purposes.

The deliverable for the full MVP is to combine the 3 tasks above into a single demonstration in the lab.

Constraints, Risk, and Feasibility

There are multiple constraints to consider for the successful implementation of this project. Namely, sourcing the selected equipment will affect the viability of validating features for the project. Additionally, the task of synchronization between multiple receivers (minimum 2, maximum 4) could require a fairly complex solution, especially at the final ‘deployment’ stage where the receivers are not directly wired to one another if GPSDOs are not received in a timely manner. Another challenge in terms of feasibility is determining the method of sending multiple receiver data to one machine (laptop or embedded device) to consolidate into the final positioning data. USRP gives the option to timestamp the samples, but this would require fairly synced receivers running internal clocks starting at almost the same time and value.

For this lizard-tracking application, the USRP B200 transceivers were selected for the detection of the collared transmitters. Sourcing the selected USRP B200 transceivers in a timely manner and having enough time to interface with them for the project plays a big role in how much can be reasonably achieved within a 10-week period. To bypass this constraint, our team has acquired similar USRP 2920 transceivers readily available in the lab to enable driver development for reading the RX data and filtering/post-processing. These units share most of the driver functionality with the USRP B200 which is based on UHD (USRP Hardware Drivers). As such, we can begin developing the RX post-processing pipeline with the 2920 unit and apply the drivers or general learnings easily to the B200 units once they are received.

Synchronization between the receivers will be crucial to getting accurate time/distance approximations of the transmitter. Based on the initial discussion about this project, the receivers would need to be synchronized wirelessly. Based on timeline constraints, implementing a wireless synchronization may prove to be difficult to achieve along with the other requirements. For our project scope, we are opting to first achieve synchronization with a wired 10MHz sync clock signal from a signal generator. This would simplify our project implementation and allow us to focus our efforts on the Signal Processing and Localization problems present. As project development continues, we will explore wireless synchronization using GPSDO for the USRP B200 transceivers.

Our tracking system will be synthesizing each of the USRP transceiver's RX data into the final position data of the transmitter attached to the lizards. We would need to determine the method of bringing the RX data from multiple USRPs to a single processor where the position data can be calculated algorithmically. The final in-the-field implementation would have some wireless network to accomplish the data transfer from the USRP transceivers to a separate Server-role machine (laptop or embedded device), but to keep our project scope within the 10-week timeframe we will set up a wired ethernet network switch or WIFI routers for the USRP's host (Gateways) and the Host controller ''Server' machine where the localization calculations would be performed at a later point as a Minimum Viable Product. If time permits, we will explore a real-time network implementation to accomplish this task.

Group Management

There is no designated leader in our group. Thus, decisions will be made by consensus when possible, or a majority vote if necessary. We will use Slack to communicate. We plan to have weekly zoom meetings on Wednesday nights and in person meetings on weekends as required.

We will try to keep each other up to date if falling behind schedule so we can help each other or shuffle priorities if necessary. The people responsible for each milestone/deliverable are listed in the Project Milestones and Schedule section of this document. We would be using the GitHub Project management tool/ Atlassian JIRA tool to track progress, create issues, and establish milestones.

The Atlassian JIRA (Free Version) is used to track the project. The dashboard link is.
https://capstone-wes207.atlassian.net/jira/software/projects/CLL/boards/1/roadmap

Project Development

The main roles for this project are DSP engineer, and Localization engineers. The DSP engineers will be responsible for programming the USRP radios to detect the transmitted signal and produce a time of arrival estimation. The Localization engineers will be responsible for developing the application that takes in the times of arrival and calculates an estimated location of the transmitter. These are the main two roles so there will be 2 people assigned as DSP engineers and 2 people assigned as localization engineers. There are several other smaller roles that include Networking engineer, test engineer, Github documentation tracker, and code quality tracker. We plan to split the roles as shown in the table below, but the roles are flexible enough that we can adjust who is working on what as the task difficulties become clearer.

Role	**Team Member**	**Role Description**
DSP Engineer	Vinit, Hassan	Develop all DSP Algorithms for USRP devices.

Localization Engineer	Ryan, Joey	MATLAB Simulation and C++ implementation of localization algorithms.
Networking Engineer	Hassan, Joey	Network the USRP/clients to the Host Laptop Running the localization code.
Quality Assurance	Vinit, Joey	Check code submissions to ensure guidelines are followed, documentation is created, and unit tests are made.
Testing Engineer Hardware	Joey, Ryan, Hassan	Test the physical hardware in the lab for functionality.
Unit Test Engineer Software	all	Each person shall sufficiently test the code he writes

The main hardware we will need are 4 USRPs, preferably with GPSDOs. We are waiting on a donation of B200s, but in the meantime have borrowed some N210s. The computers for development will run Ubuntu 20.04 or a VM running the same to provide a consistent environment with the Up Cores currently being used. The code for the USRPs will be written in C++, though initial simulations will be done in MATLAB. We will also need a router to network the computers together.

Code documentation will be done in the code files themselves. More general project documentation will be done in separate documents. Each team member is responsible for documenting the work they complete, and the quality assurance team members will hold everyone accountable.

Project Milestones and Schedule

Milestones:

- Complete the Project Specification
- Presentations #1, #2 and #3
- Setup computers to interface with USRPs
- Report on How fast we need to sample to perform Localization.
- Synchronize two USRPs in the lab with signal generators.
- Networking Gateways and Server
- Demonstrate ability to save raw USRP RX data into a file.
- Finalize Packet structure for data sent from Gateways to Host Controller PC
- Demonstration of MVP working with Localization all completed in MATLAB.
- Final Demonstration with localization ported to C++.

The roadmap generated from Atlassian JIRA tool is as follows along with the JIRA tracking IDs.

Week 1 (April 14)

- ALL: Write project specification document

- ALL: Figure out how to connect to USRP
- ALL: Get example code to run on USRP
- TB_1 : Interface a single computer with each USRP
- LOC: Research localization algorithms

Week 2 (April 21)

- ALL: Prepare for in-person presentation #1
- TB_2: Interface a single USRP with an external clock source
- TB_3: Phase synchronize two USRPs with a Signal Generator.
- LOC_1: Write a script to generate example data.
- DSP/All: Research/define how accurate of localization we can achieve with our hardware.
- DSP: Examine and document characteristics of test transmit signal.

Week 3 (April 28)

- TB_4: Test the TX radio and verify we can receive its signal on USRP.
- DSP_1: MATLAB simulation to receive the transmitted waveforms.
- LOC_2: Write a script to estimate transmitter location.
- TB_5: Network the machines together
- NET: Finalize packet format for data sent from gateway to host laptop

Week 4 (May 5)

- DSP_2: C++ app using UHD to filter and cross-correlate the received signal.
- DSP: Experiment to determine optimal sampling rate.
- LOC_3: Test Localization on real data from lab

Week 5 (May 12)

- LOC_4: Port Critical Code to C++ application
- UI_1: Display the localized algorithm data as heat map or equivalent graphical representation.

Week 6 (May 19)

- ALL: Prepare for in-person presentation #2
- LOC_5: Demonstration of working system

Week 7 (May 26)

- DSP_3: Use Angle of arrival to fine tune the received signal timestamp (Not MVP).
- DSP_4: Use GPSDO to synchronize the receivers and repeat the DSP (Not MVP).
- LOC: Improve localization estimates by using averaging.

Week 8 (June 2)

- Video Recording

Week 9 (June 9)

- Final Project Demo