**ECE496 Implementation Plan**

| **Project Title** | Physics-based machine learning models for indoor wireless localization |
| --- | --- |
| **Project Number** | 2021415 |
| **Supervisor** | Costas Sarris |
| **Administrator** | Tome Kosteski |
| **Name of students** | Prerna Anand  John Adrian Ambrad  Hiranya Maharaja  Deeksha Tewari |
| **Date of submission** | 23rd November, 2021 |

# **Table of Contents**

[Table of Contents](#_kqt4taqm01pe)

[Possible Solutions and Design Alternatives](#_p746m7c9r7uw)

[Technical Design and Implementation](#_27yibsqw17hb)

[Test and Verification](#_qobpmgtgefcd)

[RayTracer Data](#_qyyuafi820hr)

[Raspberry Pi](#_qiimwlo7p15t)

[Predictive Algorithm](#_jlygf44x9var)

[Required Apendices](#_whkd5kfrvhjr)

[Appendix A - System Requirements](#_fyulum8yu0u)

[Appendix B - Risk Assessment](#_5z66dv9sdlzc)

[Appendix C - Gantt Chart](#_84nw8hgfex58)

**Summary of Project Status and Changes**

The team has been working on data collection using a Raspberry Pi and the RayTracer provided by the supervisor, as well as researching potential machine learning models.

We were successfully able to collect data using the RayTracer, but encountered two issues with the raspberry pi:

1. The first issue we faced was with using a second Raspberry pi as a receiver. We realized that we need a power source to connect to the second raspberry pi at all times, which might not be possible as there are limited power sources in the corridors. An easier method to collect data would be using the received signal strength using our laptops/mobile phones, which would eliminate the need for a Raspberry Pi receiver.
2. The second issue was having our raspberry pi data not match our simulated data. This means that we need to review our raspberry pi configurations via openWRT to ensure that they are emitting power within a certain range, as earlier teams did when configuring the transmitters. We will also need to ensure that we have set up the receiver points correctly (i.e. laptop and mobile devices).

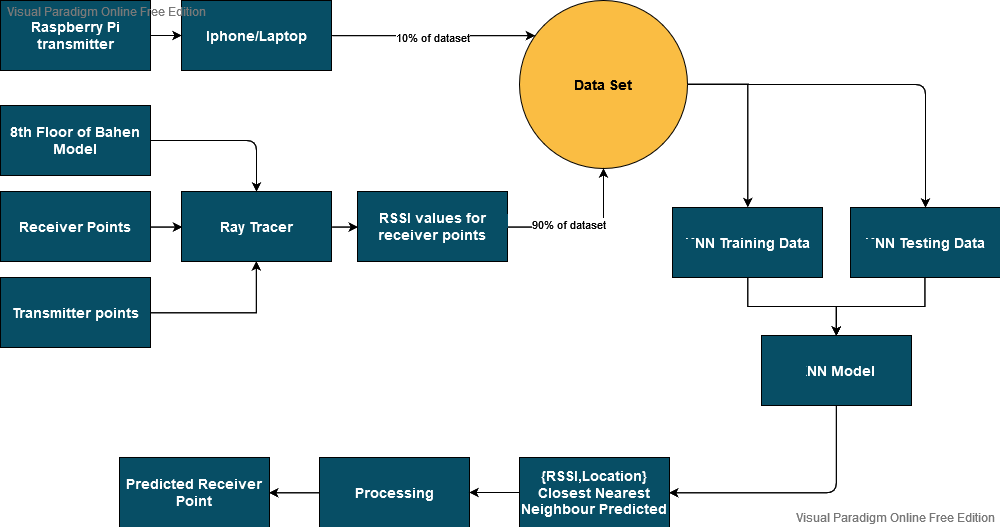
Alongside the data collection and research, we are currently working on training the scikit learn library’s Nearest-Neighbours (NN) machine learning model using the simulation data and collecting more data points using the Raspberry Pis. After we finish collecting Raspberry Pi data, we will augment the Ray Tracer data to match the collected data.

# **Possible Solutions and Design Alternatives**

As our primary solution, we have decided to use an unsupervised learning model. We have chosen to implement a NN Model from the scikit-learn library as our initial design. This method utilizes transfer learning, such that the pre-trained model from scikit-learn will be trained to fit our own data and output a set of possible neighbors of the target point. We will predict the location of the target point by finding the closest intersection between the nearest neighbors that were found using the model. However, since we cannot tune the hyperparameters of this model, we will be moving on to building a custom Nearest Neighbour model once we find a pattern of learning with the pre-trained model. This will allow us to fine tune our model to improve the prediction accuracy.

As an alternate design, we will be building a supervised learning model with a manually labeled dataset. We have chosen to implement a custom Artificial Neural Network model (ANN). The ANN model will benefit from the advantage of having labels along with the input so that the model can validate its own results during training, which could give an accurate prediction. However, the reason we did not choose this model as our primary solution is due to the fact that training supervised model will take more time, and is more dependent on the collected data points than an unsupervised model, which means we may lose out on test accuracy if the model is unable to generalize as well as its unsupervised counterpart.

# **Technical Design and Implementation**

****

**Figure 1: System Level Overview**

To effectively design, create, and test the NN model we are creating a dataset consisting of both measured and simulated data. Referring to figure 1, we get an overview of the workflow to collect data, generate the NN model, and generate predictions. To generate the simulated data, we will be using the provided ray tracer software to create a table of RSSI strengths corresponding to each transmitter. We will be pre-entering receiver and transmitter locations to the model, and note the following:

1. We will be using three transmitter/source points to have enough signal strength across the Bahen 8th floor to collect our data points.
2. 2700 receiver locations were measured from the Blender file spaced 0.5ft apart and entered into the simulation. This will be the training set. For testing, we are planning to pick points randomly distributed across the space to see how the trained model predicts locations.

The simulated dataset will account for 90% of the training dataset. The remaining 10% we will be using data measured from the 8th floor of Bahen using raspberry pi transmitters. Using our phones/laptop as receiver points, we will measure the data collected at regularly spaced intervals (spaced 0.5ft apart) using open source software like NetSpot (Windows, MacOS) and Airport Utility (iOS).

The NN model will train on RSSI values and their associated locations. It will take in values of the RSSI cost and give the [x,y] coordinates of the four closest receiver positions. We will then process them by taking the average of all these values, to find the predicted point.

# **Test and Verification**

## **RayTracer Data**

We were given an ideal value of 5 for the number of reflections by the supervisor. To verify the number of reflections is correct, we ran a simulation for the range 1-5 number of reflections. After getting the 5 csv files with the received signal strength for different numbers of reflections, we ran a convergent analysis on the data to find the ideal value. The ideal value obtained from the analysis was 5, the same as the one suggested by the supervisor.

## **Raspberry Pi**

In order to validate the raspberry pi data we plan to follow the specifications outlined by previous teams. These specifications include the range of wifi power output (range of RSSI values) on the raspberry pi and configurations of the open source openWRT software. We will also analyze the general trend of the raspberry pi data and compare it with the simulated ray tracer data to ensure that both are following the same trendline.

## **Predictive Algorithm**

The predictive algorithm will be tested by using a test dataset. This set will be samples that the algorithm has never encountered in order to give a measure of its ability to generalize to novel situations. We will then directly measure its accuracy by seeing how far its predictions were from the actual coordinates of the target.

| **ID** | **Project Requirement** | **Description** |
| --- | --- | --- |
| 1 | Location in x-y coordinates of a static target given a WiFi signal strength (RSS) | **Test:** Direct measurement |
| 2 | Build a transmitter with the Raspberry Pi | **Test:** Direct Measurement |
| 3 | Data collection can only use RSS (received signal strength) | **Test:** Direct Measurement |
| 4 | Map of Bahen Building | **Review of Design:** Through all trials design will use map of bahen |
| 5 | Raspberry Pi Router | **Review of Design:** Through all trials design will use raspberry pi |
| 6 | Predictions should be less than 4 metres from the actual target | **Test:** Direct Measurement. |

# **Required Apendices**

## **Appendix A - System Requirements**

| **ID** | **Project Requirement** | **Description** |
| --- | --- | --- |
| 1 | Location in x-y coordinates of a static target given a WiFi signal strength (RSS) | **Functional Requirement:** this is the output specification of the project. |
| 2 | Create a transmitter with the Raspberry Pi | **Functional Requirement:** the Raspberry Pi needs to be able to transmit a WiFi signal as that will be fed into our design as the source access point from which we determine the RSS at the target. |
| 3 | Data collection can only use RSS (received signal strength) | **Constraint:** Other methods are not feasible given their large expense and training requirements. |
| 4 | Map of Bahen Building | **Constraint:** Other buildings have thicker walls which will interfere with the collection of accurate data. |
| 5 | Raspberry Pi Router | **Constraint:** We cannot run the risk of disrupting UofT routers and need to test using our own. |
| 6 | Predictions should be less than 4 metres from the actual target | **Objective:** this would be an improvement on the previous teams’ margin of error. |

## **Appendix B - Risk Assessment**

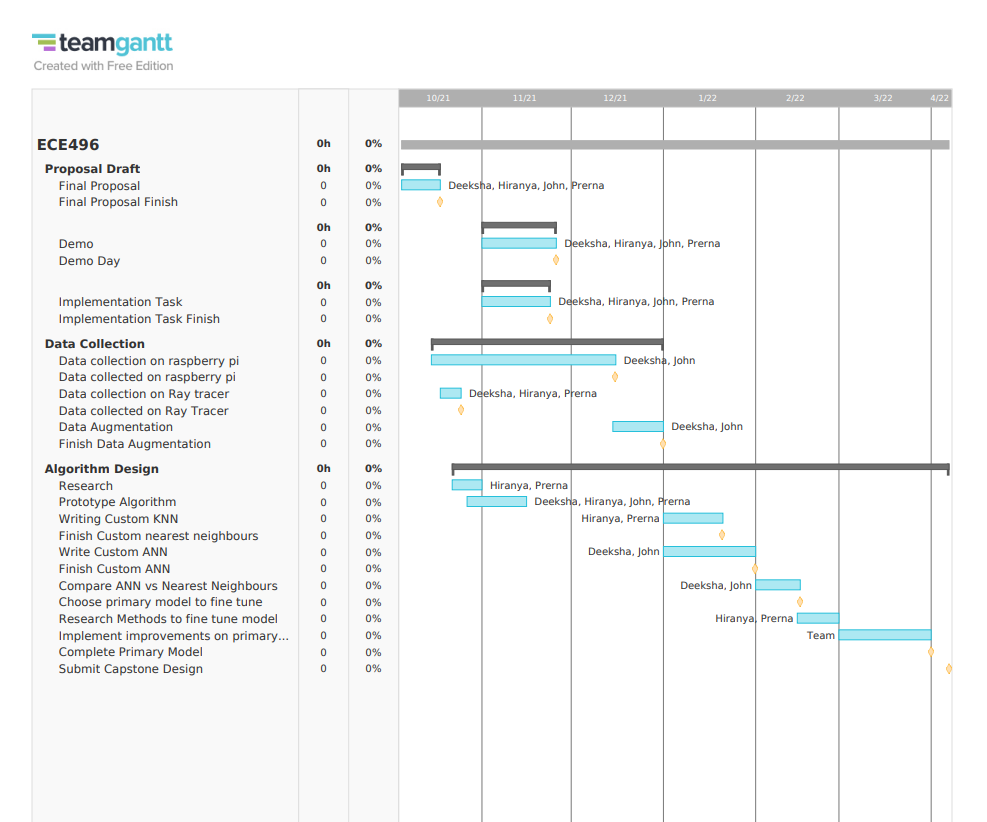
Two critical risks to the project are:

1. The Ray tracer data and Simulation data might have a significant offset
2. The model we choose might not provide us with the accuracy we are aiming for, thus rendering our objective obsolete.

We plan to mitigate the first risk by first ensuring that we are correctly configuring the openWRT software and are only producing the RSSI values in a certain range. Based on the values of the Raspberry Pi, we will add the offset to the RayTracer data to ensure the data matches and is in the same range.

The second issue can be resolved with time management. Instead of working serially on the development of the artificial intelligence models, we plan to parallelise the workflow. In pairs of two, the team will research and prototype different models. The data collected in the next two months will serve as a guide to give us preliminary accuracy values that we can use to evaluate any future data processing and evaluate the efficacy of the model in solving our particular localisation problem. We can then settle on a model and perform training, testing, and validation. We have included a range of accuracies to ensure that even if we are unable to meet the desired objective, providing even an incremental improvement on the previous team’s work would meet the requirements of the project.

## **Appendix C - Gantt Chart**

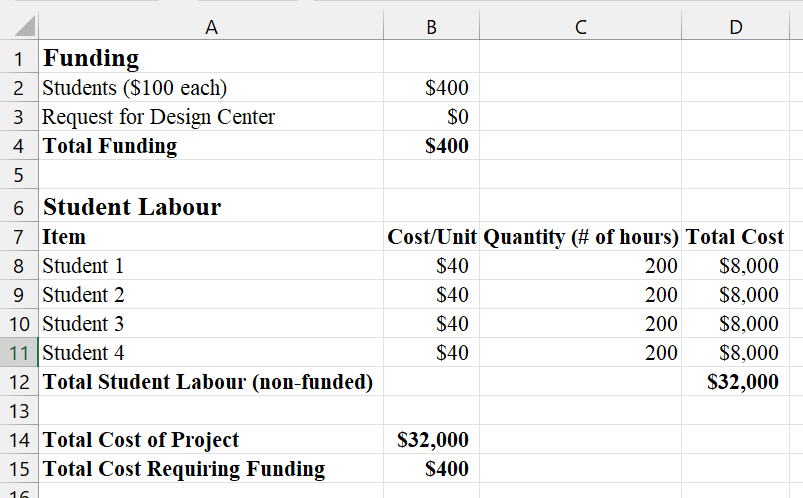
****

Team Gantt Chart displays how we are distributing tasks amongst team members. Each month has an important milestone that the team will review periodically as it moves forward with the project. The team members will collaborate on large, time intensive tasks in the later months.

**Appendix D - Financial Plan**

The cost of this project is $0. We were provided with all the required hardware by the supervisor.

**Budget Table**

****