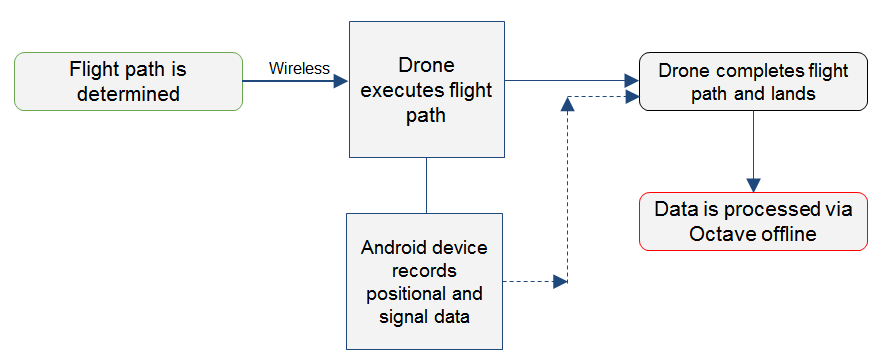
**Abstract**

This project seeks to develop and test a low-cost, easy to use approach to signal mapping. The proposed objective will be accomplished by mounting an Android mobile device on a 3DR Solo drone. Data will be collected via a signal strength measurement app, outputted to a .csv file. The program Octave will be used to place the data points into three dimensional space. At this point, a custom written interpolation algorithm uses the nearest neighbor node signal strength theory to fill in the space where no data is collected. All of this data is then placed onto a 3D graph, for user end visualization. The applications of this project are fourfold. Firstly, this approach will allow for the identification of signal leakage beyond designated broadcasting bands. Secondly, this approach will allow an optimization of signal output based on nearby topography. Thirdly and fourthly, this approach will allow for ease of troubleshooting, as well as general diagnostics.



**Figure 1:** Simplified diagram of the various components and their interactions

**Introduction**

In today’s world, wireless connections are everywhere. However, there are almost no tools that produce an easy to use graphical display. In the United States, the FCC allows certain areas of the spectrum for certain purposes. Cable companies have one section, while the military has its own section. Signal leakage is when signals spill over from their designated band into adjacent bands. This can cause interference and disruption of other signals [51]. In a world of growing wireless connections, signal leakage is becoming more and more of an issue [6]. The approach and techniques proposed in this paper would assist in determining signal variability, strength, and leakage.

Beyond this, there is a general lack of portable and efficient spectrum mapping tools. This project proposes an approach and several techniques that when coupled together can greatly alleviate this issue.

**Objective**

**Create a 3D signal quality map using a drone as a mobile receiver carrier. This will be accomplished through the use of a 3DR Solo drone (receiver carrier), mobile Android device (signal recorder), and Octave (graphical program with custom written interpolation algorithms).**

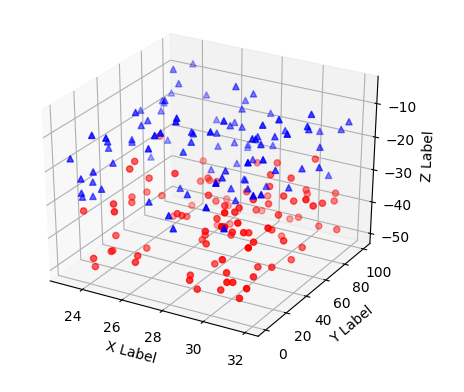
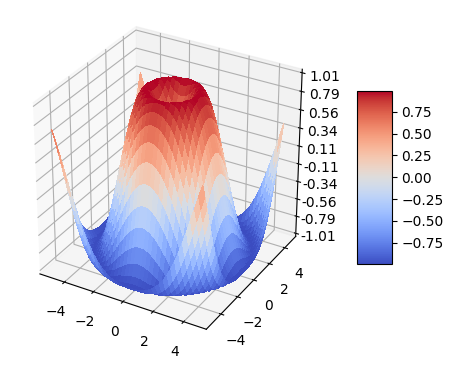
**Implementation**

A mobile receiver in the form of an android phone will be attached to a 3DR Solo drone. This drone will be released into the area of interest and follow a predetermined, uploaded flight plan. Upon completion of the flight, the signal data (comma separated value file) will be downloaded for offline analysis.

The data analysis program will have two purposes. The first purpose is to use a custom written interpolation algorithm to fill in the space where no data was collected. Since wireless signals follow predetermined mathematical formulae, the interpolated data will have a reasonably high degree of accuracy. The second purpose is to produce a graphical output to display the data at a given point in time.

**Materials, Equipment and Programs**

* Drone (3DR Solo)
* Mobile receiver (Android device)
* Data collection software (Network Signal Info app)
* Data processing program (Python/Octave)
* Computer with wireless connectivity



**Figure 4:** Sample graphical output of raw data with coloring by signal strength. [2]

**Figure 5:** Sample graphical output of the set of both interpolated and actual data [2]

**Interpolation Algorithms**

In the field of spectrum mapping and networks, there are three major interpolation algorithms: nearest neighbor, inverse distance weighting, and natural neighbor. Each has specific applications, but is usually used for varying degrees of accuracy (previously mentioned in increasing order).

**Nearest Neighbor**

The point of interest is given the value of the next closest data point. In figure 3, P0 would be given the value of S1, because that is the closest data point.

**Inverse Distance Weighting**

The point of interest is given a value based on the distance for nearby data points. The value of P0 would be determined using the following equation:

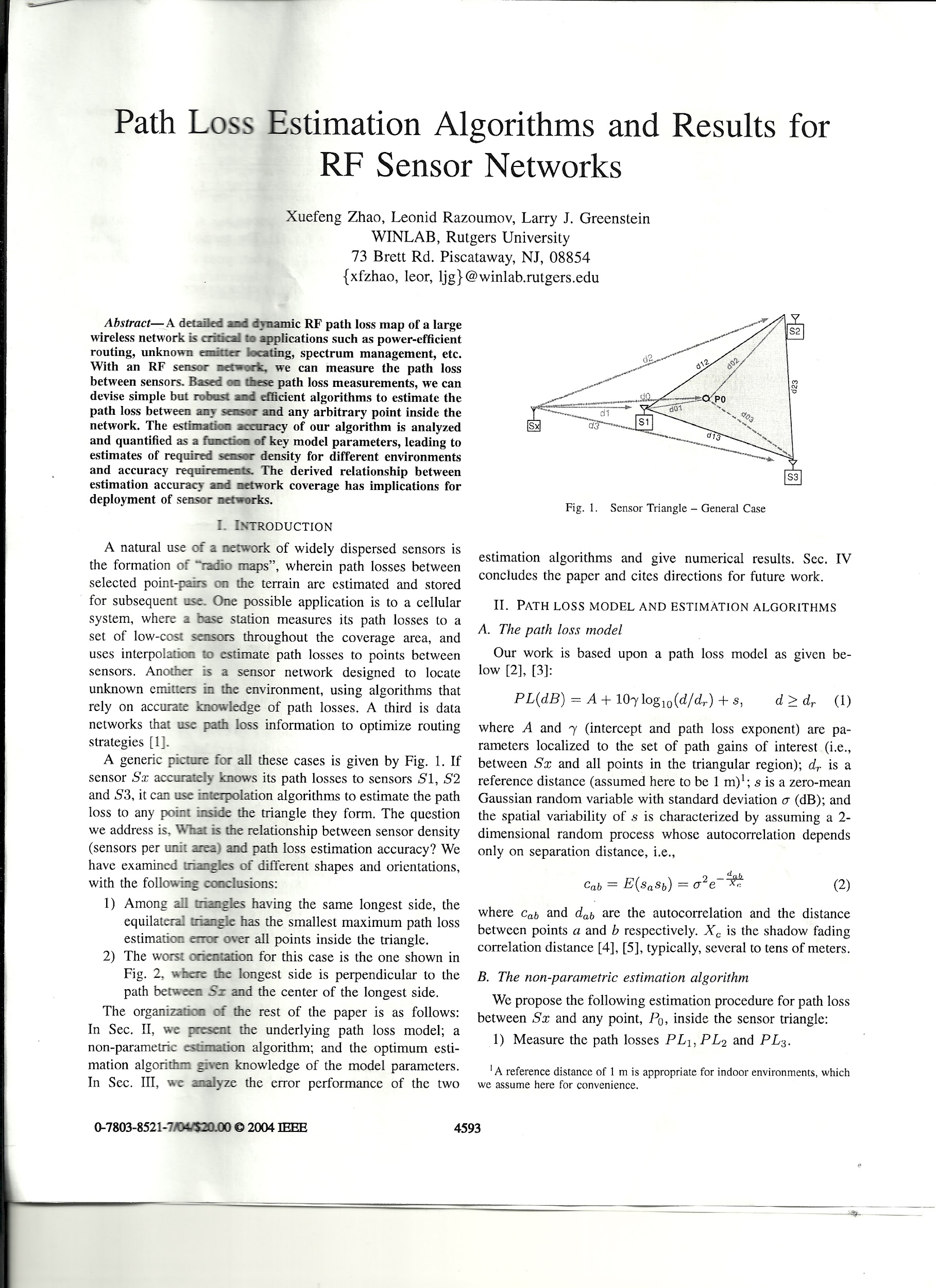
**Natural Neighbor**

The point of interest is given a value based on the area of overlap between nearby data points. It is functionally similar to IDW, but with areas instead of distances.

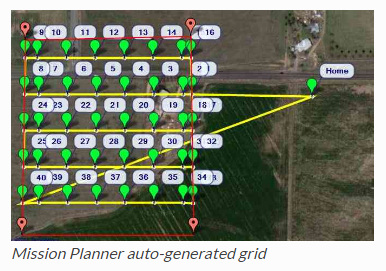
**Algorithm Selection**

A modified nearest neighbor algorithm was selected for several reasons. The primary reason was lack of computing power. Python itself offers limited analysis of large quantities of data, but the computer used for processing also had limited processing power. The nearest neighbor algorithm was modified to use the least computing power because of this.

Of note, the points used by the algorithm were randomly generated. These points were then compared to the original data set, and then given the value of the closest point. Both the original data and generated values were then combined into one data set, which was used for the graphical output.

****

**Figure 3:** Diagram of sensor triangle (possible configuration of data points in 2D space) [62]



**Figure 2:** Mission planner auto grid [43]

**Graphical Output**

At the most basic level, the output file is a three dimensional graph with different colors representing signal strength. Specifically, red is a stronger signal while blue is weaker.

However, through the usage of the custom written interpolation algorithm (modified nearest neighbor), a smooth three dimensional surface can be created. Given more computing power, it would be extremely easy to generate additional points via the interpolation algorithm, thus creating a smoother shape.

**Flight Pathing**

Fight pathing is critical for optimal data collection. This can be accomplished by hand though it is tedious and often ineffective. There is a similar issue in the field of agriculture, specifically when it comes to placing seeds. The seeds must be as close as possible, but still have enough space between them to allow for watering and growth. Similarly, data needs to be logged at certain points in the area of interest- spaced out as far as possible to save time, but close enough to each other so that the interpolation algorithm is accurate.

Using the program MissionPlanner, a grid (called “auto grid”) was generated for the area of interest. These way points were then stored in a mission file, which is uploaded to the drone via a wireless. This flight plan is then executed by connecting to the drone (via its own wireless network) and running the appropriate flight sequences.

**Applications**

Aside from generating a visual representation of the spectrum and signal strength in an area of interest, this project has four major applications.

* **Signal Leakage Identification**
  + In the United States, the FCC allows certain areas of the spectrum for certain purposes. Cable companies have one section, while the military has its own section. Signal leakage is when signals spill over from their designated band into adjacent bands.
  + This can cause interference and disruption of other signals [51]. In a world of growing wireless connections, signal leakage is becoming more and more of an issue [6].
  + The approach and techniques proposed in this paper would assist in determining signal variability, strength, and leakage.
* **Optimization of Signal Output**
  + Signal output is not evenly distributed, nor does it need to be.
  + Using the graphical output, one could easily redesign the signal source to be stronger in areas of interest.
* **Troubleshooting and general diagnostics**
  + A graphical representation of the spectrum would greatly assist users in pointing out issues and consequently fixing them.
  + The entire system is both easy to use and portable, further adding to its usefulness.

**Data Analysis**

1. Data from the .csv log file is read into Python.
2. This data is then placed onto a three dimensional graph and shown to the user. High signal strength is represented as **red** and low signal strength is represented as **blue**.
3. The space between data is filled in by generating additional data points through a custom written interpolation algorithm.
4. Currently, there is no way to represent time on the graph (five variables on three axes presents an issue), but if developed for commercial use, this could be remedied by continually generating graphs for the user.

In its current state, the program for data analysis is written in Python 3.6 . However, it is in the process of being ported over into Octave. Octave is a free version of MATLAB and is prevalent in many engineering fields. As such, an Octave version of the program is both easier to modify and offers more data manipulation options.

**Conclusions**

The approach taken in this experiment proved to be feasible. Although limits on computing power produced a less than optimal result, there is a clear proof of concept.

Of note, the custom written interpolation algorithm (modified nearest neighbor) sacrifices accuracy for speed and computing power. Consequently, the graphical output and any models produced from it would lack significant accuracy. Future iterations should opt to use the optimal algorithm for the appropriate available computing power.