Improved Indoor Localization via WiFi Trilateration

Adrian Brandemuehl
Martin Pandola

TA Daniel Gardner
1 Introduction

1.1 Statement of Purpose

Indoor GPS is inaccurate. Improving it would:

- Help 911 operators find callers accurately.
- Allow developers to improve indoor app experiences.

To improve accuracy we will design a WiFi RSSI trilateration system implemented through a mesh network of nodes.

1.2 Objectives

This project should be able to accurately localize phones and push the location data back to them.

1.2.1 Benefits

- Developers can provide better indoor mapping experiences
- Allows users to receive accurate directions while indoors
- Improves reported e911 locations

1.2.2 Product features

- Per device location broadcast
- LocationServices integration
- Increased indoor location accuracy

1.3 Alternatives

There are several alternative implementations already existing.

<table>
<thead>
<tr>
<th>Method</th>
<th>Downsides</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magnetic field mapping</td>
<td>Entire area has to be pre-mapped</td>
</tr>
<tr>
<td>WiFi signal mapping</td>
<td>Entire area has to be pre-mapped</td>
</tr>
</tbody>
</table>

Our method requires:

- Basic installation (stick and plug in)
- Minimal calibration
- Very little phone battery
2 Design

2.1 Block Diagram

![Block Diagram](image_url)

Figure 1: Block Diagram

2.2 Block Descriptions

(I) **Antenna**: We will design a PCB trace omnidirectional 2.4GHz antenna to capture WiFi packets. Directionality would introduce stronger signals from specific areas, decreasing the overall accuracy of the device.

(II) **WiFi IC**: The WiFi IC will be used to control the antenna. It will be able to advertise as an 802.11g Access Point as well as control a mesh network between the nodes. 802.11g was chosen based on the ESP8266 Tx power and Rx sensitivity for each protocol. The mesh network is required for 3 nodes to be able to do the trilateration calculation.

(III) **Flash**: The flash will be used to store the microcontroller program as well as the configuration values for the nodes. The ESP8266 requires program flash. On boot, the ESP8266 copies the instruction memory from the flash into the internal RAM.
(IV) **Microcontroller:** The microcontroller will be in charge of the trilateration calculations. It will collect the signal strengths and send them to the node with the strongest signal (theoretically the node closest to the user), which will then calculate the user’s position and push that back to the phone.

(V) **Status LEDs:** LED’s will indicate status signals to allow operators to determine it’s status from a distance. Two LED’s will indicate the power status, and mesh network connection status of the node. Since the nodes will be high off of the ground, visual status indication will be important for operators to verify if the module is working.

(VI) **Step-Down Voltage Regulator:** Steps down 5V from a USB input to the 3.3V required to power an ESP8266 (see fig. 1). Given that we are designing an RF application, the power supply must have very little noise that could affect the operation of the antenna. The regulator will be a Linear Drop Out regulator, rather than a switching regulator. Switching regulators introduce extra noise, while providing high efficiency. LDO’s provide low noise power, but have lower efficiency. Since power is provided via a wall socket, efficiency is of little concern, hence we are going to use an LDO based regulator.
## 3 Requirements and Verification

<table>
<thead>
<tr>
<th>Module</th>
<th>Requirement</th>
<th>Verification</th>
<th>Points allocated</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Antenna</strong></td>
<td>Must be able to transmit a signal with 15dBm input power (as specified by the ESP8266 datasheet) that has no more than 92dB signal loss (See Tolerance Analysis).&lt;br&gt;Must have $50\Omega \pm 5\Omega$ impedance (see Tolerance Analysis) at 2.4GHz.&lt;br&gt;Must be omnidirectional: must receive packets from all directions without varying RSSI values more than 10%.</td>
<td>We will test the signal power at 100 meters.&lt;br&gt;We will test the antenna’s impedance using a network analyzer.&lt;br&gt;We will send a node packets from a set distance while rotating the receiving node.</td>
<td>30</td>
</tr>
<tr>
<td><strong>Wireless IC</strong></td>
<td>Must be able to transfer data to other nodes within 100 meters with a bit rate error of less than $10^{-6}$.&lt;br&gt;Must generate a 2.4GHz 802.11g network that Android smartphones can connect to.&lt;br&gt;Must have 1000 packets per second of throughput with a Bit Error Rate of less than $10^{-6}$.</td>
<td>We will test this by placing two nodes in line of sight, 100 meters away from each other. We will send a predetermined stream of data, and compare the received signal to the sent to calculate bit rate error.&lt;br&gt;We will make connections to the WiFi network from multiple types of smartphones.&lt;br&gt;We will test this by placing two nodes in line of sight, 100 meters away from each other. We will send 1000 premade packets and measure how much time it takes to receive the packets to calculate the transfer rate. In addition, we will check how many bits were incorrectly transmitted to determine the Bit Error Rate.</td>
<td>20</td>
</tr>
<tr>
<td>Module</td>
<td>Requirement</td>
<td>Verification</td>
<td>Points allocated</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>Microcontroller</td>
<td>Must be able to perform the trilateration algorithm on 1000 packets per second on each node.</td>
<td>We will create a test dataset of packets and metadata and have the microcontroller perform the algorithm on them, while timing how long it takes. We will test this by changing the configuration values while the system is running and compare the calculated location values with the actual locations.</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Must be able to change persistent configuration values without turning off the system.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flash</td>
<td>Must run reliably at 80MHz.</td>
<td>We will write a program to read and write sequential locations in the flash and time it to determine how many reads and writes we can have per second.</td>
<td>10</td>
</tr>
<tr>
<td>Voltage Converter</td>
<td>Must supply 3.3V ± 0.3V from a USB power source (5V, 1.5A).</td>
<td>We will draw 1.5A from a DC power source set at 5.3V and 4.7V, pass it through the converter, and record the output voltages.</td>
<td>10</td>
</tr>
<tr>
<td>Status LEDs</td>
<td>Must be visible at 15 ft in direct sunlight.</td>
<td>We will create a circuit with the LED and 120Ω resistor, power it with 3.3V, and see if it is visible from 15 ft next to a window.</td>
<td>10</td>
</tr>
</tbody>
</table>
4 Schematics

Figure 2: USB to UART [9]

Figure 3: 5V to 3.3V Regulator [7]
Figure 4: Antenna Impedance Matching [6]
Figure 5: ESP8266 with External SPI Flash and Oscillator [11] [10]
5 Tolerance Analysis

The most integral part of this project is the detection and capture of WiFi packets. The blocks that contribute the most to this functionality are the WiFi IC and the omnidirectional antenna.

The ESP2866 specifies its transmission power for the 802.11g protocol as +17dBm, and its reception sensitivity as -75dBm. Thus, the absolute maximum loss we can tolerate for our system is:

\[
\text{Loss}_{\text{max}} = 17 \text{dBm} + 75 \text{dBm} = 92 \text{dB}
\]

If the antenna is operating outside of this range, our WiFi IC will not receive packets accurately. Since we will be sending packets using the TCP protocol the IC will need request many resends, drastically reducing throughput. Using the equation for free-space path loss, we can find a maximum operating distance.

\[
FSPL(\text{dB}) = 10 \log \left( \frac{4\pi df}{c} \right)^2 = -147.55 + 20 \log(d) + 20 \log(f)
\]

Where \( f \) is the signal frequency in Hertz, \( d \) is the distance from the transmitter in meters, and \( c \) is the speed of light in a vacuum. Since our signal has a frequency of 2.4GHz, we can plug that into our equation to omit the \( f \) variable.

\[
FSPL(\text{dB}) = 40.05 + 20 \log(d)
\]
A plot of this equation is shown in fig. 7. Our figure only shows distance through 100m to illustrate that our distance requirement is valid, but we can use our $\text{Loss}_{\text{max}}$ in our previous equation to calculate the maximum distance before our IC stops functioning.

$$92\text{dB} = 40.05 + 20\log(d_{\text{max}})$$

$$\Rightarrow d_{\text{max}} = 395.82\,\text{m}$$

This is still not a viable distance to expect the IC to be functioning however, due to the fact that our free-space path loss equation assumed vacuum conditions and we assumed maximum power transfer in our antenna which would imply our antenna impedance is exactly matched to the IC output impedance as discussed before.

Figure 7: Signal Loss by Distance at 2.4GHz (Original Plot)

Since free-space path loss will degrade our signal strength, we need to find the distance in which our RSSI loss over 1dB will exceed our required accuracy. Using the same equation
from before we find that the loss right under this requirement is 77dB as illustrated below.

\[
FSPL(70m) = 40.05 + 20 \log(70m) = 76.952 \\
FSPL(78m) = 40.05 + 20 \log(78m) = 77.892
\]

Clearly, this difference is less that 1dB and the distance difference is 8m (our required accuracy). In this calculation we will take into account the loss from the omnidirectionality of our chosen antenna design. As seen in the tables below, our antenna (the Regular IFA) has a maximum loss of -28.9dB in the YZ plane when horizontally polarized. Unfortunately, as our devices will likely be mounted on walls, horizontal polarization is the most likely polarization we will be dealing with. Using this loss from the antenna, combined with the loss of our maximum free-space path loss, we get a loss of:

\[
77dB + 28.9dB = 105.9dB
\]

Since this is outside of our IC’s maximum sensitivity for 802.11g, our accuracy requirement will not be a problem as long as we detect the packet.

<table>
<thead>
<tr>
<th>Plane</th>
<th>Max ∆dB</th>
<th>Plane</th>
<th>Max ∆dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>XY Vertical</td>
<td>-23dB</td>
<td>XY Vertical</td>
<td>-28.5dB</td>
</tr>
<tr>
<td>XY Horizontal</td>
<td>-23.1dB</td>
<td>XY Horizontal</td>
<td>-29.5dB</td>
</tr>
<tr>
<td>XZ Vertical</td>
<td>-8.7dB</td>
<td>XZ Vertical</td>
<td>-21.8dB</td>
</tr>
<tr>
<td>XZ Horizontal</td>
<td>-16.5dB</td>
<td>XZ Horizontal</td>
<td>-19.7dB</td>
</tr>
<tr>
<td>YZ Vertical</td>
<td>-12.4dB</td>
<td>YZ Vertical</td>
<td>-4.3dB</td>
</tr>
<tr>
<td>YZ Horizontal</td>
<td>-28.9dB</td>
<td>YZ Horizontal</td>
<td>-19.2dB</td>
</tr>
</tbody>
</table>

Comparison between regular and meandered inverted F antenna designs.[5][6]

6 Cost and Schedule

6.1 Cost Analysis

6.1.1 Labor

Martin: $40/hour * 2.5 * 10 hours/week * 9 weeks = $9,000
Adrian: $40/hour * 2.5 * 10 hours/week * 9 weeks = $9,000
Total labor cost: $18,000
6.1.2 Parts

<table>
<thead>
<tr>
<th>Price</th>
<th>Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2</td>
<td>WiFi IC + MCU</td>
</tr>
<tr>
<td>$40/3</td>
<td>4 Layer PCBs</td>
</tr>
<tr>
<td>$3</td>
<td>Assorted components</td>
</tr>
<tr>
<td>$0.50</td>
<td>Flash Memory</td>
</tr>
<tr>
<td>$1.50</td>
<td>USB Connector</td>
</tr>
</tbody>
</table>

Total cost per node = $20.
We will need to build 3 nodes for prototyping and 3 for a final design, totaling $122.

6.1.3 Grand Total

Total = Labor + Parts = $18,000 + $63 = $18,122.

6.2 Schedule

<table>
<thead>
<tr>
<th>Week</th>
<th>Adrian</th>
<th>Martin</th>
</tr>
</thead>
<tbody>
<tr>
<td>9/18/16</td>
<td>Design MCU connections schematic</td>
<td>Research and design antenna layout</td>
</tr>
<tr>
<td>9/25/16</td>
<td>Join antenna design</td>
<td>Continue antenna layout design</td>
</tr>
<tr>
<td>10/2/16</td>
<td>Write trilateration algorithm</td>
<td>Finalize PCB design</td>
</tr>
<tr>
<td>10/9/16</td>
<td>Order and fabricate PCB</td>
<td>Verify power and LED modules</td>
</tr>
<tr>
<td>10/16/16</td>
<td>Verify MCU and Flash modules</td>
<td>Verify antenna performance</td>
</tr>
<tr>
<td>10/23/16</td>
<td>Mesh network protocol</td>
<td>Access point code and trilateration integration</td>
</tr>
<tr>
<td>10/30/16</td>
<td>Write configuration code</td>
<td>Write position logging code</td>
</tr>
<tr>
<td>11/6/16</td>
<td>Write Android integration app</td>
<td>Write position log reader code</td>
</tr>
<tr>
<td>11/13/16</td>
<td>Test and debug</td>
<td>Test and Debug</td>
</tr>
<tr>
<td>11/28/16</td>
<td>Present</td>
<td>Present</td>
</tr>
</tbody>
</table>

7 Safety and Ethics

The primary ethical concern in our project is the passive location tracking of devices. The device that we’re building would theoretically be able to passively track devices without user interaction. However, the software that we will be writing will require that users connect to a WiFi network that does not give internet connection. Users will have no incentive to connect to the WiFi network outside of improving their location accuracy. In keeping with the IEEE Code of Ethics clause 9, “to avoid injuring others, their property, reputation, or employment by false or malicious action;”, we shall not invade users privacy without consent [12].
References


