# SAFE COMMUNICATION BETWEEN LEAD AND FOLLOWING VEHICLE

Senior Design Team sdmay18-29

## Team Members:


Sang Uk Park
Brad Stiff
Jose Candelario
Yifan Lu
Justin Wheeler
Junho Chun
Zhize Ma


## Faculty Advisor:

Chinmay Hegde

## Client:

Vishal Mahulkar

## Problem Statement

The GPS is the most important components in our project's sensor system. Our problem, was that the GPS itself could not transmit data and we had no control over the data sent, thus we needed to find a tool to help send and receive the relevant information reliably within 300 feet. This problem also tasked us with the responsibility of powering the individual devices efficiently and safely.

## Solution

Our solution to our problem is to use a GPS mounted on the leading vehicle with 900MHz transceivers to effectively transmit the information of latitude, longitude, and time to the following autonomous vehicle.

## Introduction

For our senior design project, our goal is to set up communications between two vehicles, which includes reading, sendings, receiving GPS signal from a leading vehicle to the following one, powering all involving sensors and circuitaries, and formatting output data.

## Design Requirements

**Functional**
- Able to send the precise location data of a lead vehicle to a following vehicle
- Able to transmit data in sufficient range
- Able to communicate sensors information to ROS

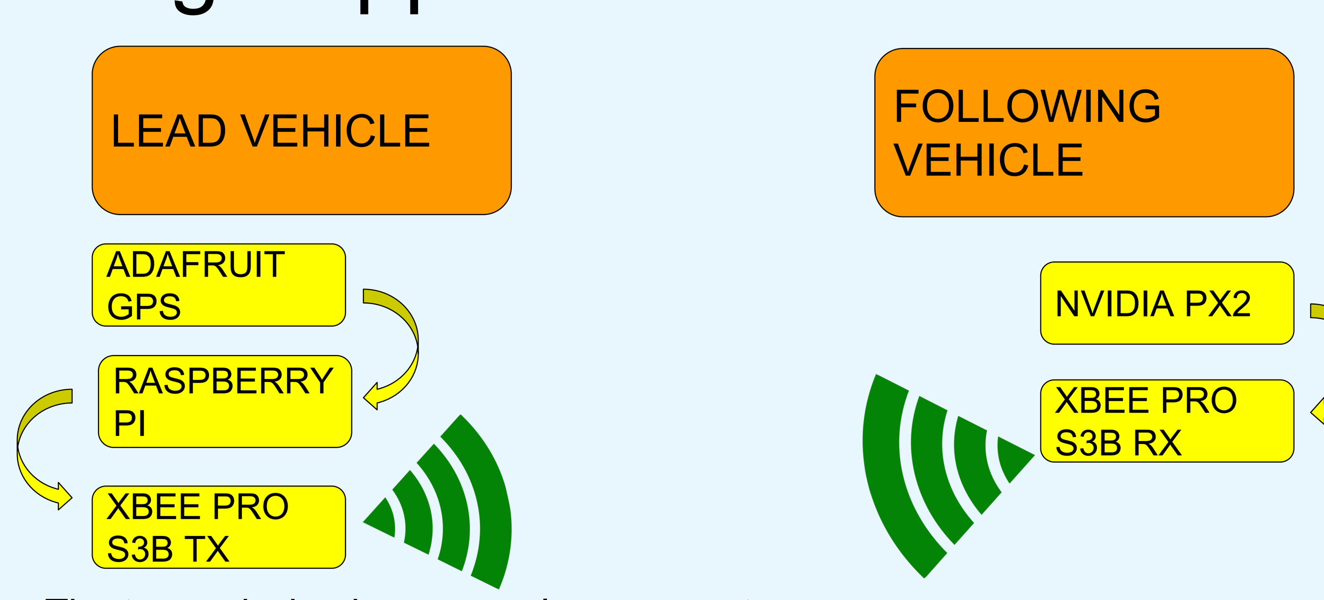
**Non-functional**
- XBee location and PX2 placement
- Hardware and power set up throughout the car

**Engineering**
- More information to send the harder to lower frequency

## Operational Environment

- At least 300 feet distance between vehicles
- Working in any type of environment
- At least 20 mph speed for both vehicles

## Design Approach




The transmission has several components:
- Adafruit GPS
  - Receives information from satellites including latitude, longitude, and time (capable of sending more with slower Hz)
- Raspberry Pi
  - Used to configure the Adafruit GPS as well as package the information with packet loss to be sent over the XBee
- XBee Pro S3B
  - Used to transmit information over long distances with moving vehicles. 900 MHz with 9 miles LOS using our 900 MHz duck antennas
- Nvidia PX2
  - Mounted with ROS (Robot Operating System) can receive information from several sensors to tell the car where it needs to go

Relevant Standards
- IEEE 802.15.4

## Intended Uses & Users

- Interstate Pilot using Driver for Extended Availability
- We can eliminate STOP&GO waves (Phantom Traffic Jam) caused by human driving behavior
  Ex) Increased Highway Capacity, Lower Fuel Consumption, Fewer Accidents
- Vehicle on Demand
- Human do not have any options to take over driving task
  Ex) Different Business Models: Taxi Service, Car Sharing, Autonomous Cargo Vehicle
- Mass Transportation
- Companies need to spend less money on drivers
  Ex) Companies can send multiple cargos together with only one driver

## Technical Details

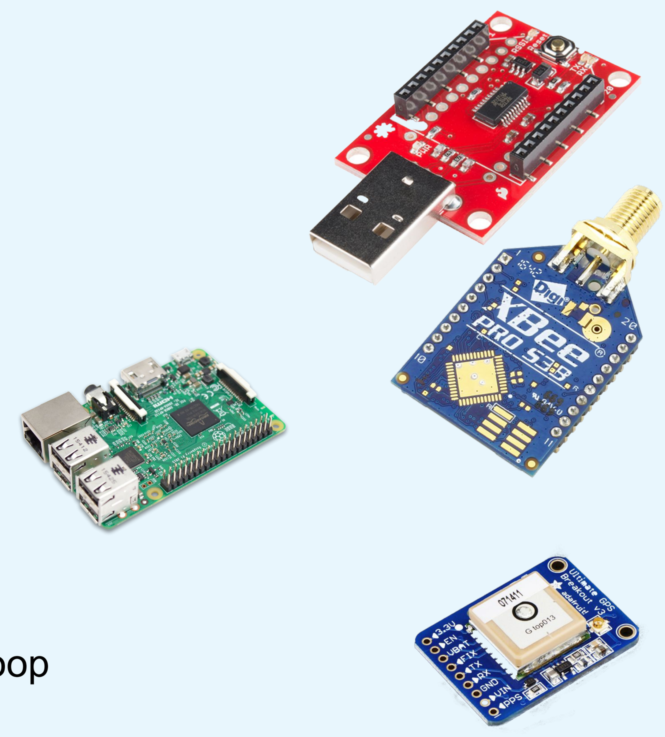
**Technology Modules:**

XBee Pro S3B
- Transceiver Rage: 300 meters
- Baud Rate: 115200 max

Raspberry Pi 3 Model B
- CPU: Quad Cortex A53
- Memory: 1GB SDRAM
- USB Port: 4

Adafruit Ultimate v3 GPS
- Position Accuracy: 1.8 meters
- Baud Rate: 9600 default
- Update Rate: Between 1~10 Hz

Nvidia Drive PX 2
- Sensor Fusion Support: Camera, Lidar, Radar
- Data Collection: Algorithm Development, Hardware Loop



**Software Modules:**

Python
- Version: 2.7.12

ROS Kinetic Kame
- Running on the Nvidia Drive PX 2

Raspbian Stretch OS
- Default operating system for the RPi

Adafruit CircuitPython GPS Module
- Open-source library which allowed interfacing with the GPS

PyCharm Community Edition
- Python IDE by JetBrains

## Testing

**Environment**

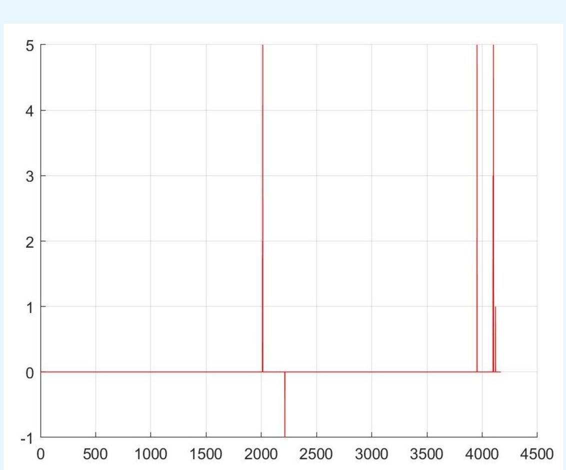
The following conditions were used for testing:
- Highway and urban roads.
- Low speed with many turns and straight high speed streets
- Test was done on a cold day.

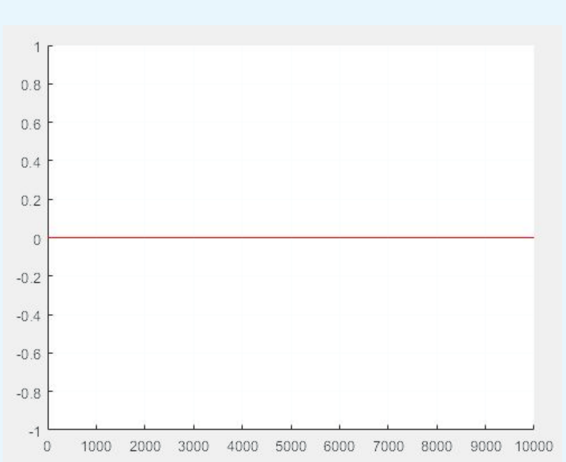
**Strategies**

There were two main tests that we wanted to perform on.

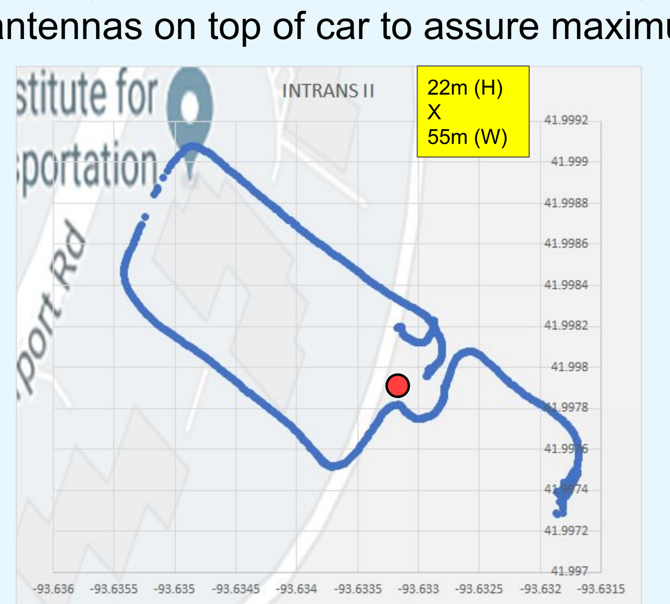
Different Speeds:
- Drove two vehicles at certain speeds
- Increments of 5 mph
- 20 mph to 70 mph
- Below you can find one of the graphs we used as well as one before we switched to the 900 MHz antennas. The difference is that with the old ones, that transmitted at 2.4GHz, big chunks of information was lost. Once we switched to these ones no data was lost when speeding up. Time to transmit also stayed constant compared to before.

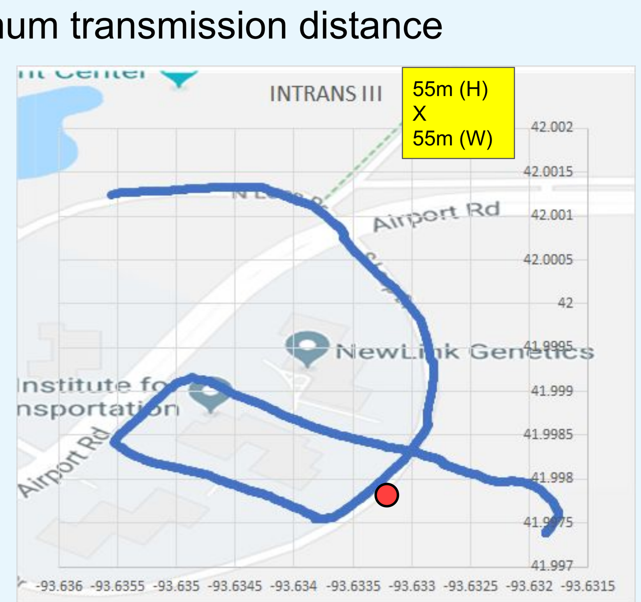
X axis is amount of data received, y axis is bare minimum data lost. We knew that way more than this data was lost but this gave us an idea of how to find some lost data.





Different Distances:
- Had one car stay in place
- Other one went around behind buildings
- Made a circle around a building
- Drove away as far as possible before losing complete information of data sent
- Had antennas on top of car to assure maximum transmission distance




## Resources

**Materials:**

| | | |
|---|---|---|
| Kuman 7" HDMI Display | | **59.98** |
| Raspberry Pi 3 Model B | | **34.50** |
| Samsung 32GB SD Card | | **9.99** |
| XBee Pro S3B | x2 | **78.00** |
| USB Dongle | x2 | **49.90** |
| Antenna | x2 | **16.98** |
| Adapter Cable | x2 | **10.98** |

| | |
|---|---|
| Adafruit Ultimate GPS | **39.95** |
| USB to Serial Cable | **9.95** |
| Antenna | **12.95** |
| Adapter Cable | **3.95** |
| 30 Amp Connectors | **16.47** |
| 6 to 8 Pin PCIe Adapters | **6.99** |
| **Total** | **350.59** |