2017 CNL Final project: Push notification via Wifi

> Team 13
> 2017.06.15

Motivation

- When entering a place such as a library or a classroom, we want to get information about that place by internet, such as visiting the website of the course to see if there are some announcements or downloading the class materials.
- If the desired information can pop up to users automatically, it will be very convenient.

Goal

- Create a system that can push different kinds of notifications in different places by Wifi.
    - First, we have to get wifi configuration of users
    - Then, we need to implement Wifi positioning

Programming Languages

- Node js
- Javascript
- HTML
- Python

Equipments

- AP * 3
- Laptop * 1

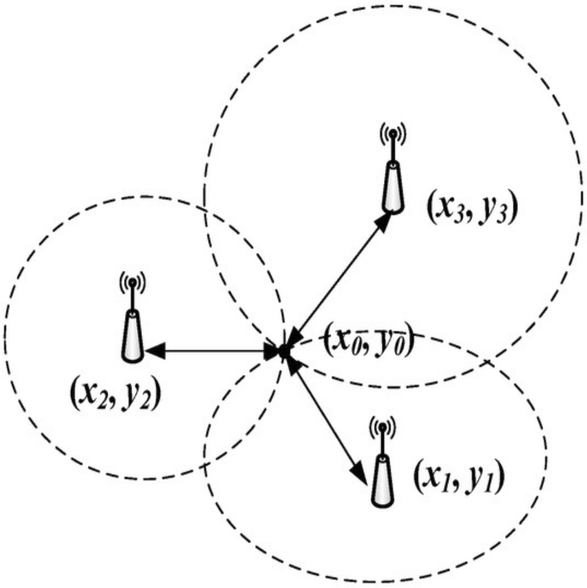
3 Scenarios

Scenario 1

- Assign a website to each AP
- The website assigned to the AP which has the **Smallest Free Space Path Loss (FSPL)** for user will pop up automatically
- FSPL is the loss in signal strength that occurs when an electromagnetic wave travels over a line of sight path in free space
- Why use **FSPL** instead of the largest signal level ?
    - Maybe the largest Wifi signal level doesn't come from the nearest AP.
    - **FSPL (dB) = signal_level_transmitted (dBm) - signal_level_received (dBm)**
    - So, the smallestst FSPL will come from the nearest AP.

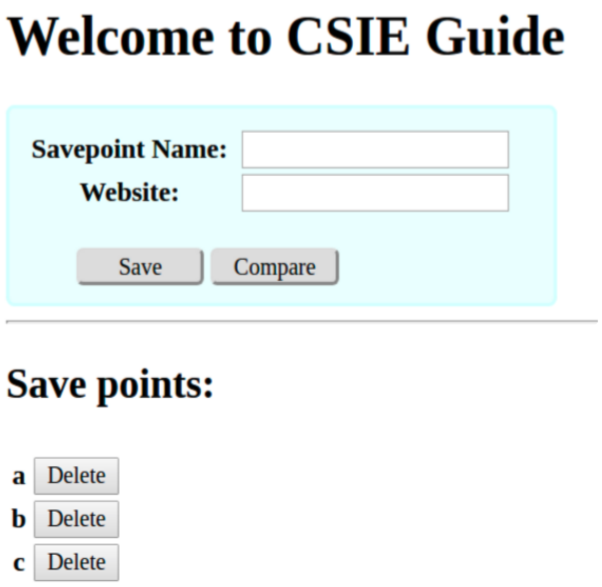
Scenario 2

- Assign a website to each region
- According to the **position** of user, the corresponding website will pop up automatically
- Difficulty : wifi positioning



Scenario 3

- User can manually set the website corresponding to a location.
- That website will pop up when the user enter that location



Welcome to CSIE Guide

Savepoint Name:

Website:

Save points:

a

b

c

Implementation

Implementation for Scenario 1

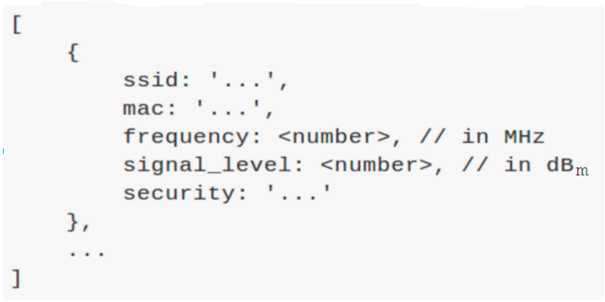
Method

- Scan the wifi information of a user and count FSPL of each AP user detected
- Open the new page corresponding to the AP with the smallest FSPL

Problems and solutions

- Problem
  - We want to implement the service in front-end, but we can’t get the Wifi information of users because of the privacy issue.
- Solution
  - User needs to install npm packages and start the server on their machines
  - Scan wifi information by **node-wifi** npm package (scan 5 times and select the median)

node-wifi



```
[
    {
        ssid: '...',
        mac: '...',
        frequency: <number>, // in MHz
        signal_level: <number>, // in dBm
        security: '...'
    },
    ...
]
```

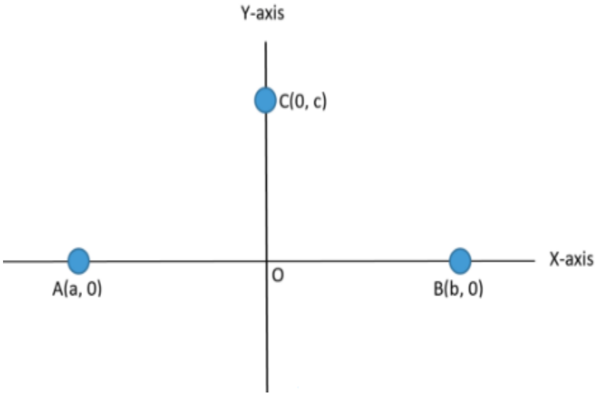
Implementation for Scenario 2

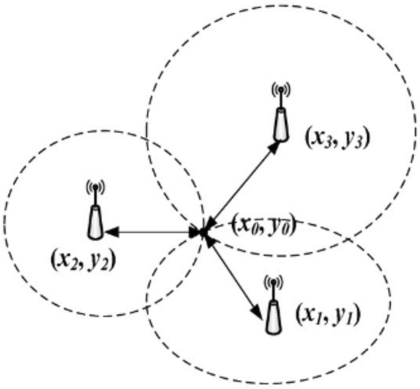
Method 1: Use triangular positioning

- Apply the triangular positioning method to evaluate a user's coordinate by the **calculated distance** between APs to the user

Coordinate

- Assume there are three APs: points A, B, C
- The x-axis is the line between A and B
- The y-axis passes C and is perpendicular to x-axis
- Coordinates of AP : A(a, 0), B(b, 0), C(0, c)

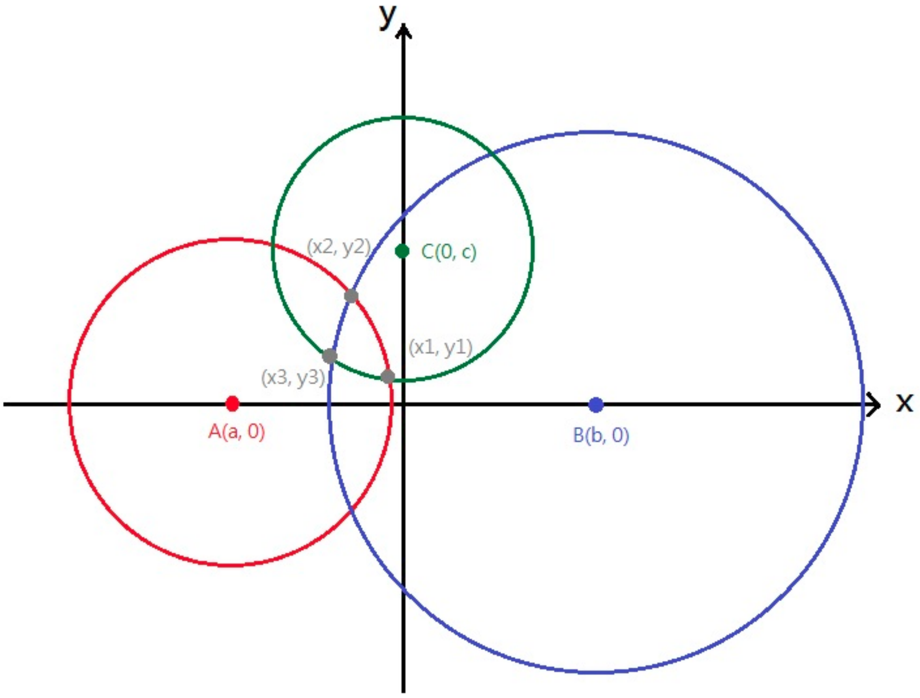




Triangular Positioning

- Given three distances of arbitrary point to A, B, C
- Algorithm Traingular Positioning (a, b, c, d_1, d_2, d_3)
  - Check triangular inequality
    - If triangular inequality doesn't hold, adjust d_1, d_2, d_3
  - Compute (x_1, y_1), (x_2, y_2), (x_3, y_3)
  - $x = \dfrac{x_1 + x_2 + x_3}{3}$
  - $y = \dfrac{y_1 + y_2 + y_3}{3}$
  - return (x, y)
- Why we need to compute 3 times?

    - The distances might be inaccurate.



Distance function

- Ideal distance function
    - $d = \dfrac{10^{0.05 \cdot FSPL - 4.622}}{f}$
    - d: distance (m), f: frequency (MHz), $FSPL$: Free Space Path Loss (dB)
- Our distance function
    - Scan the signal levels at different distances and get a set of $(FSPL, d)$
    - $d = \dfrac{10^{aFSPL+b}}{f} \quad \Rightarrow \quad \log(d \cdot f) = aFSPL + b$
    - Compute a, b by linear regression of $(FSPL, \log(d \cdot f)) \quad \Rightarrow \quad a = 0.05,\ b = 2.447$
    - $d = \dfrac{10^{0.05 \cdot FSPL + 2.477}}{f}$

Problems and solutions

- Problem: Not quite accurate
- Solution: Try Machine Learning methods for wifi positioning

Method 2: Machine Learning (regression)

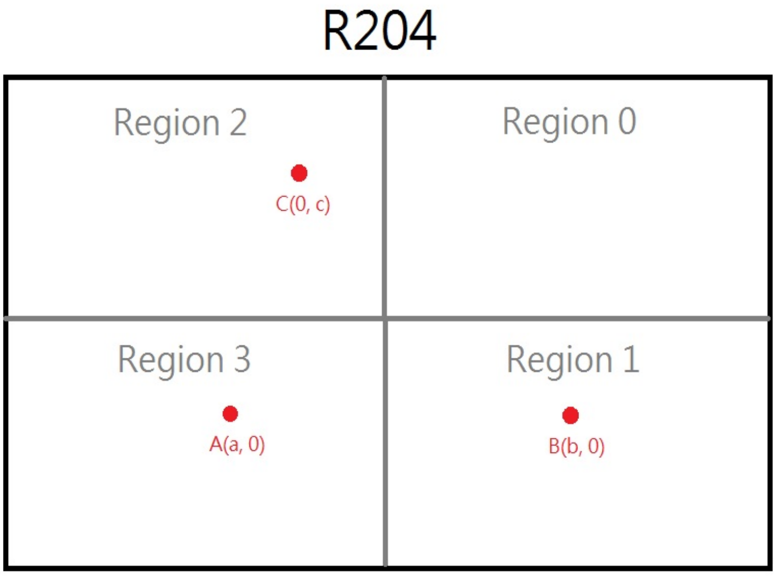
- Training data: **`coordinate x, coordinate y, signal_levels`**
    - 320 for training, 80 for testing
- Use `sklearn.svm.SVR` for training
- Evaluation: RMSE (Root Mean Square Error)

- Coordinate x : $E_{in} = 2.3137 \quad E_{out} = 2.9152$
- Coordinate y : $E_{in} = 1.0991 \quad E_{out} = 1.7591$

- Problem
  - Coordinate x is not accurate enough
- Solution
  - Instead of calculating the coordinate, we determine which region users are in

Method 3: Machine Learning (classification)

- Training data: **`region, signal_levels`**
  - 320 for training, 80 for testing
- Use `sklearn.svm.SVC` for training
- Evaluation: 0/1 classification Error
  - $E_{in} = 0.0469$
  - $E_{out} = 0.325$
- After adjusting boundary and use all data for training, this can achieve about **80%** success rate
- Possible improvements
  - Use more AP for positioning, and more data and features for training
  - For example, we can add humidity, temperature to our features



Implementation for scenario 3

Method

- When user record a location, a savepoint will be created.
  - The savepoint contains the website assigned and the wifi configuration at that location.

- Savepoints will also be stored in users' browsers by using cookies
- When user click compare, the system compares the similarity of current Wifi configuration to all savepoints one by one.
  - $\text{signal(mW)} = 10 \cdot \log_{10}(\text{signal\_level(dBm)})$
  - $$\begin{aligned}\text{Difference} &= \sum(\text{signal\_saved} - \text{signal\_received})^2 \\ &= \sum(10^{\frac{\text{signal\_level\_saved}}{10}} - 10^{\frac{\text{signal\_level\_received}}{10}})^2\end{aligned}$$
  - Open the website corresponding to the savepoint with the smallest difference

Conclusion

We have learned how to

- Scan wifi signal level
- Handle the interaction between back-end server and front-end website
- Deal with async functions in javascript
- Implement triangular positioning algorithm
- Define our own distance function
- Utilize ML methods to solve problems
- Store and load cookies

Reference

- D-Link DIR-615 (https://www.speedguide.net/routers/d-link-dir-615-wireless-300n-router-243)
- A research of wireless location algorithm on Android platform (http://implement.ee.nthu.edu.tw/2011/poster.php?ID=B59)
- WiFi Indoor Positioning System Design Based on the Triangle Positioning Algorithm (http://cbimg.cnki.net/Editor/2016/0215/gxkx/ce75e214-193b-41fb-b2a4-6e670db55230.pdf)
- sklearn.svm.SVR (http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html)
- sklearn.svm.SVC (http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html)
- node-wifi (https://www.npmjs.com/package/node-wifi)
- Async functions (https://developers.google.com/web/fundamentals/getting-started/primers/async-functions)
- Nodejs async series (https://stackoverflow.com/questions/22424592/nodejs-async-series-pass-arguments-to-next-callback/26614972)
- IBM WL.Device.Wifi (https://www.ibm.com/support/knowledgecenter/en/SSHS8R_7.0.0/com.ibm.worklight.apiref.doc/html/refjavascript-client/html/WL.Device.Wifi.html#acquireVisibleAccessPoints)
- Mozilla JavaScript Set (https://developer.mozilla.org/zh-TW/docs/Web/JavaScript/Reference/Global_Objects/Set)

- ECMAScript 6 sets: union, intersection, difference (http://2ality.com/2015/01/es6-set-operations.html)
- Free Space Path Loss (http://www.radio-electronics.com/info/propagation/path-loss/free-space-formula-equation.php)
- Wikipedia dBm (https://en.wikipedia.org/wiki/DBm)

Work distribution

studentId	student name	work load	work
b03902024	鄭筱樺	$\frac{1}{6}$	scan wifi configuration, web server for 3 scenarios, test 3 scenarios, ppt
b03902084	王藝霖	$\frac{1}{6}$	scan wifi configuration, web server for 3 scenarios, test 3 scenarios
b03902080	黃子賢	$\frac{1}{6}$	write ppt and report, test 2 scenarios
b03902026	徐新凱	$\frac{1}{6}$	mathematical formulation of triangular positioning, data collection for training, test 2 scenarios
b03902126	高翊軒	$\frac{1}{6}$	Implement scenario 3 and website
b03902022	于建民	$\frac{1}{6}$	ppt, implement scenario 2 (distance function, ML methods)