1. Formal Description of the experiment

Key	Value
Date of experiment	11.02-12.02
Experiment team	3 RCSE students, 1 supervisor

The aims of the experiment are:

1. Check the functionality of experimental software system: **GPS_Tracker**, **GPS_Frontend**, **GPS_Android**.

- Check the functional properties of the system.
- System stability, performance and usability measurements in real case scenarios.

2. Optimization algorithms evaluation for UAVs (having Wi-Fi AP) layout optimization.

- Evaluation of correctness of provided optimized positions by the algorithms.
- Measurement of stability, performance, and usability of the algorithms.

Main information

The main purpose of the experiment is to optimize the location of AP's such a way that throughput and RSS of UE's connected to those AP is maximized.

For that we experiment with the following way:

1. APs located in the space without any obstacles. They are surrounded by UEs.
2. UEs connected to APs and evaluate RSS and throughput via specially designed software.
3. Stored measured records by UEs are sent/copied to the central server.
4. An operator uses the provided interface to analyze the measurements and run optimization algorithms to find out the best positions for APs.
5. After the next optimal positions for APs are found, access points moved to the optimized points.
6. An experiment is going to step 3 and repeated until no significant improvement for APs positions will be observed.

The experiment repeated three times with different initial APs and UE positions.

Test sets:

1. Suboptimal (APs in between clusters)
2. Near-optimal (APs in clusters according to K-Means.)
3. Uniform (in the area)

For each test case, we expect that:

1. Minimization of the distance between UEs and APs will lead to RSS gain and throughput increase.
2. The interference effect may be visible – in the suboptimal placements, it should decrease the throughput;

Glossary

Terminology	Description
UE	User Equipment - a mobile terminal which transmits data via the radio link. Each UEs runs different Android OS version (4.0+)
AP	Access Point - a Wi-Fi access point running by a laptop with an external Wi-Fi adapter. Each laptop has a side application to support throughput measurement.
CnC	A central server where the system core is running. All measurement analysis and UI interactions perform here.
UAV	Unmanned Aerial Vehicle - a vehicle having AP on the board.
RSS	Received Signal Strength - the estimated value of signal power for the radio signal. The greater the RSS value, the stronger the signal.

2. Experiment requirement

Hardware Equipment

- 6 cellphones with Android OS (version 5.0 and above). A dedicated GPS and Wi-Fi modules.
- 3 external Wi-Fi Adapters with AP mode capable.
  - Model: AWUS036NEH
  - Antenna's height for CnC: 17.2 cm
  - Antenna's height for AP: 11 cm
- 3 Laptops.
- One ruler.
- Optionally, 6-10 carton bags for hardware safety against the weather.

Software Equipment

For UEs

- Android-enabled smartphone (version 5.0 and above).
  - `Magic Perf` installed.
- `GPS_Android` installed.

For Command Center (CnC)

- Host OS
  - Any Debian-based OS (Debian, Ubuntu, etc.).
  - `VirtualBox` + `VirtualBox Extention Pack` latest version.
  - `Python3` + `Ansible` installed.
  - `openssh-server`.
- The virtual CnC machine
  - `dnsmasq`, `hostapd` installed.
  - `openssh-server` installed.
  - `docker` and `docker-compose` installed.
    - MongoDB container.
    - RabbitMQ container.

        * Mosquitto MQTT broker container.
        * Python3 container.
        * nginx container.
        * nodeJS container.

For APs

- Host OS
  - Any Debian-based OS (Debian, Ubuntu, etc.).
  - `dnsmasq`, `hostapd` installed.
  - `iperf3` installed.
  - `openssh-server` installed.
  - `docker` and `docker-compose` installed.
    * An FTP server container (e.g. VSFTPd).

3. Experiment preparation procedures

In this section, we specify the steps to be performed before the main part of the experiment.

These steps influence experiment parameters. We aim to take into account environmental conditions and to minimize noise components.

Steps

Step	Description
Prepare required hardware and software	Because the experiment would be complicated from the control point of view, this is required to prepare and automate as much as possible
Pre-install GPS_Android on UEs	Before the experiment, the software must be on Android phones already to ensure there is no incompatibility
Check the weather	Specific weather condition (such as snow or rain) may damage equipment and disrupt experiment result
Check the RSS attenuation	This parameter will influence the area used to locate APs and UEs
Measure the experiment area and locate APs on the field	In case if RSS is high enough there is no reason to use a smaller area because the radio link quality would remain the same

4. Experiment description

General perspective

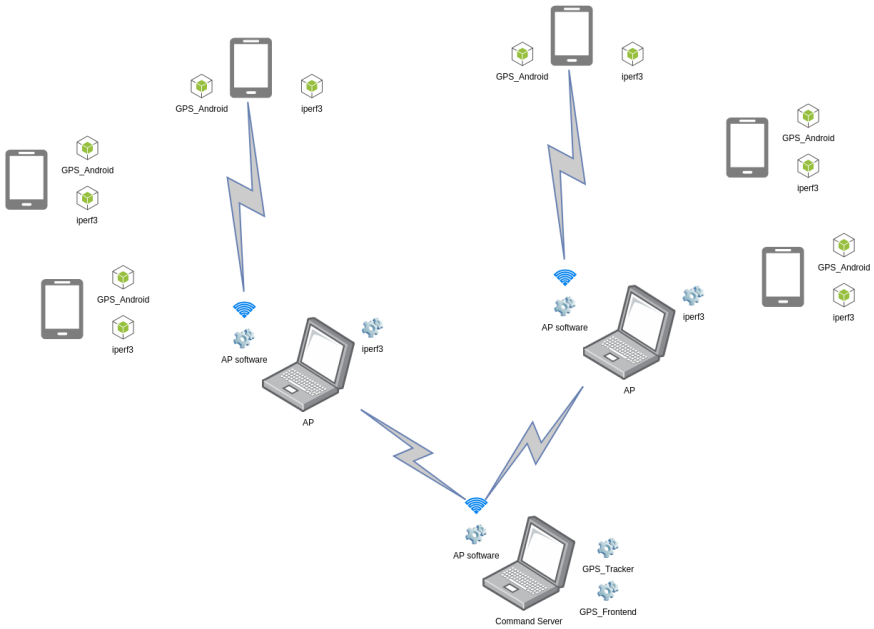


Figure 1: Main perspective of the experiment

From the general system engineering perspective, the experiment is a set of wireless-connected nodes (via Wi-Fi protocol) which measures the receiving signal strength and measure the throughput of the link to upload and download.

All connections are wireless on each bearer:

- UE <-> AP.
- AP <-> CnC.

There investigated one problem: the experimental network bandwidth between UE and AP measurements with `iperf3` showed about **30 MBit/s** speed rate. On the contrary, the speed on the bearer UE <- AP -CnC> showed about **12-15 MBit/s** speed rate. There is markedly seen a drop in speed rate, probably, because of transmission on the radio channel two-times. The problem not in the Wi-Fi itself, but the radio link is not as reliable, as a cord one, so the active bandwidth measurement part must be located as close to the APs as possible - in our case, the server-side `iperf3` is located in APs.

Each UE has two programs on board:

- `GPS_Android` - special software designed to perform RSS and link measurements linked to GPS coordinates.
- `Magic iperf` - a network bandwidth measurement software. Provides a user-friendly interface to client-side app `iperf` on Android phones.

Each APs has two Wi-Fi adapters. Since we use laptops to run APs, they expected to have one already included, thus one external extra required for each AP. The internal Wi-Fi adapter connects to the CnC provided Wi-Fi network, the external Wi-Fi adapter creates the access point named "**ap**" for the UEs.

The CnC uses one external AP to create the access point named "**cnc**".

Deployment Diagram

Table 4: The main deployed components.

Component	Description
Android smartphone group (UEs)	A set of smartphones running Android OS (version 5.0+) with dedicated Wi-Fi and installed software.
Access points (APs)	The computers running a Wi-Fi AP software for the UEs connection. Pass traffic through to CnC and take part in RSS and link quality measurement.
Command Center (CnC)	A computing node running the optimization software. Should be provided with sufficient hardware resources.

From the deployment view, we are using a complicated combination of software and hardware.

We prefer to run AP and designed software separately in a virtual machine. That helps to automate development, testing, and maintenance routines.

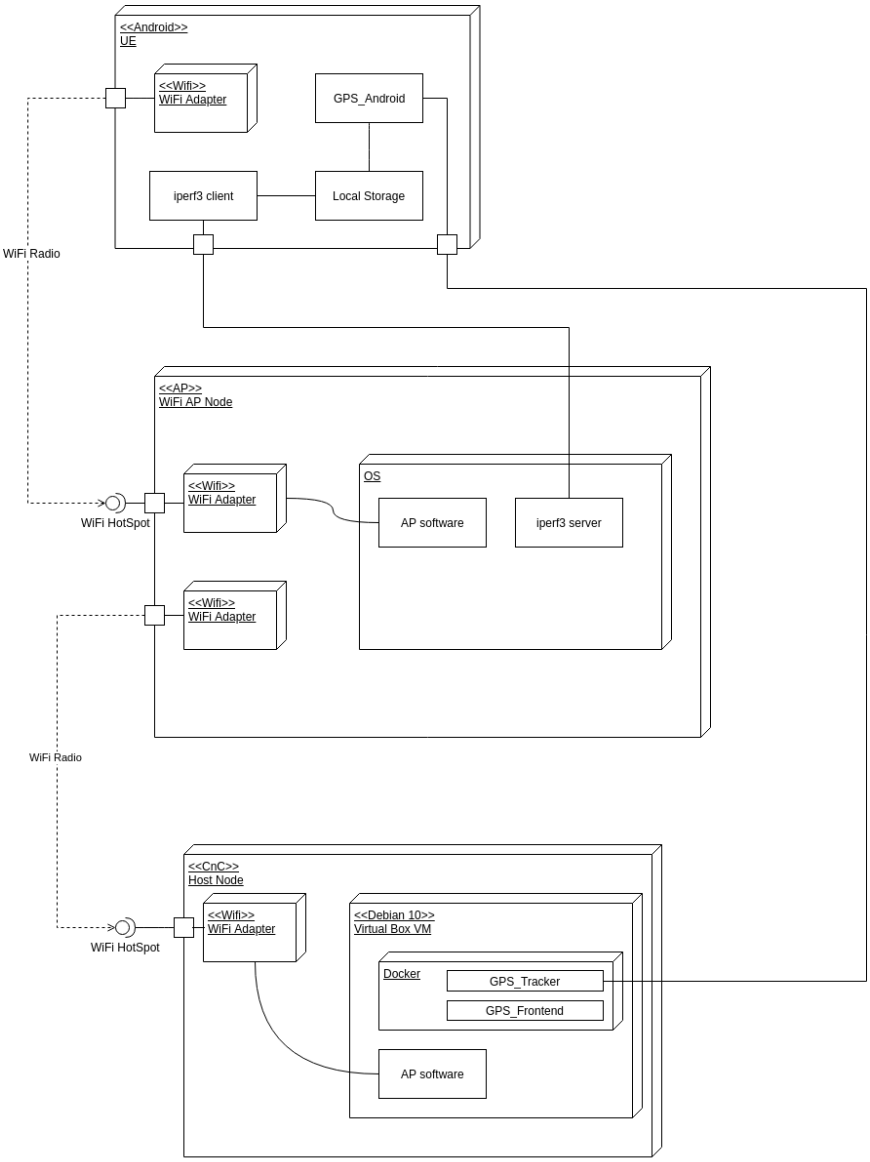


Figure 2: Deployment Diagram

Command center deployment (CnC)

The CnC software runs on Debian OS in VirtualBox virtual machine. `GPS_Tracker` `GPS_Frontend` is designed to run in containers. For the experiment, the virtual machine has **docker** and **docker-compose** installed to run these containers. We don't consider much performance decrease as long as enough hardware resources provided for the CnC virtual machine.

The AP software consists of two packages:

- `hostapd` - software to manage and run Wi-Fi access points.
- `dnsmasq` - DNS/DHCP server, to provide an IP address, routing and DNS information via DHCP protocol.

The AP software starts in the virtual machine. To access the physical external Wi-Fi adapter the hardware pass-through from the host machine to the virtual machine via hypervisor is used.

For easy-to-run configuration and deployment of the CnC, there are provided an **Ansible** script and a **Vagrantfile**. See their requirements to use.

Access Points deployment (APs)

Unlike CnC, physical nodes run the AP software and server-side `iperf3` app.

Network Diagram

The APs subnet has identical settings. These `Wi-Fi HotSpot Network` has an internal DHCP server to provide dynamic addresses for connected UEs. To prevent the network IP addresses collisions and simplify routing, the APs perform masquerading (SNAT/DNAT) on the output interface (the internal interface used to connect to CnC). On the one side, it cannot access the UEs directly from CnC, but the UEs will always reach CnC as long as DHCP sets the default gateway IP address.

In `Wi-Fi CnC Network` installed another DHCP server. It used to reply to connecting APs with dynamic addresses. Because the subnet used in this network is different from the internal Wi-Fi adapter's network in the APs, there is no network collision.

Finally, the UEs can access the static CnC address `192.168.20.1` as well as its local AP's gateway address `192.168.10.1`.

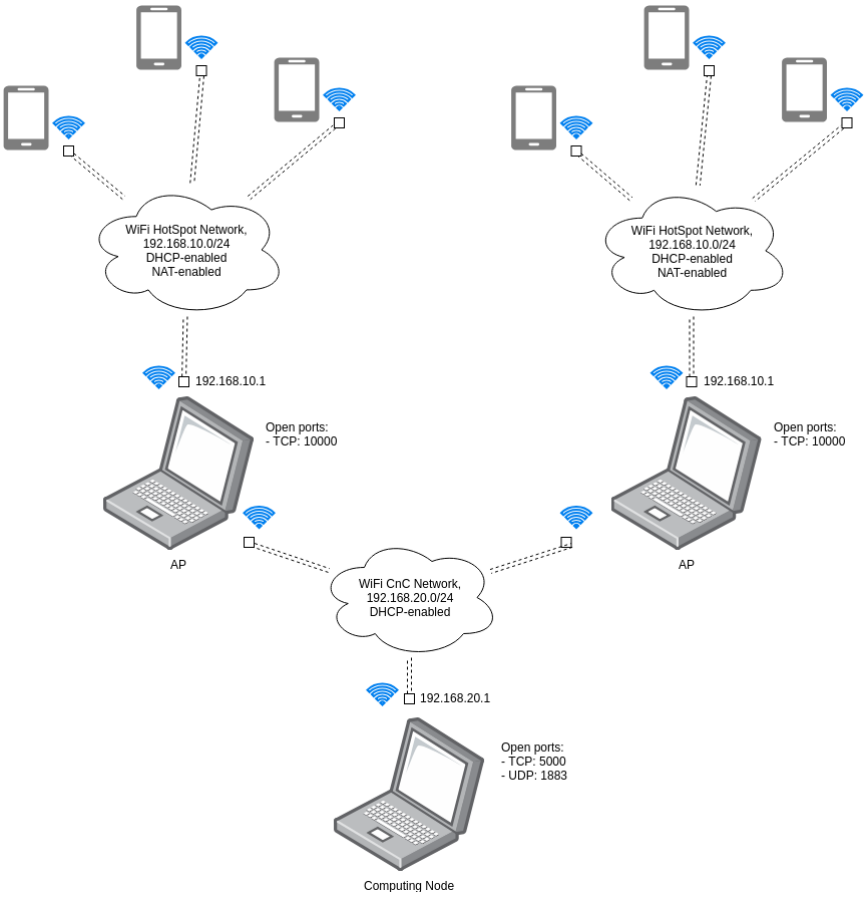


Figure 3: Network Diagram

Experiment Steps

The following steps will be executed for each described experimental case:

Table 5: Steps for one experimental case.

No	Step	Description
1	Initialize network connectivity between APs and CnCs	Ensure that these nodes are available in the network by the ICMP protocol
2	Run the software for the experiment	Startup GPS_Tracker, GPS_Frontend
3	Place the APs and UEs according to an experiment case	There are specific predefined positions for each element on the experiment area.
4	Measure RSS, Link quality for the initial layout	Measurements are done via GPS_Android that sends the result to GPS_Tracker

No	Step	Description
5	Run the APs location optimization for each algorithm in GPS_Tracker	Each optimization algorithm can produce different probable positions for the same UE positions and measurements.
6	Move the APs to the optimized positions	It is expected that new positions for APs would increase our network efficiency.
7	Repeat RSS and Link quality measurements for the optimized APs positions	1-3 interactions for optimization per each case.

Case-1. Sub-optimal layout

In the first case **Access Points(APs)** are placed in the ground so that both of them are in the center of the corresponding quarter of 25x25 meters and far from all UEs to approximately 25 meters.

UEs are left in two groups of three filling the other two quarters respectively.

At the same time, they are not too close to each other in order to not have interference and absolutely the same conditions.

Finally, **CnC** is set in the middle of the whole 50x50 meters an experimental field, so that 2 APs and 2 groups of UEs mentioned above are on the same distance.

This case expects to perform in the best signal quality and transmission rate.

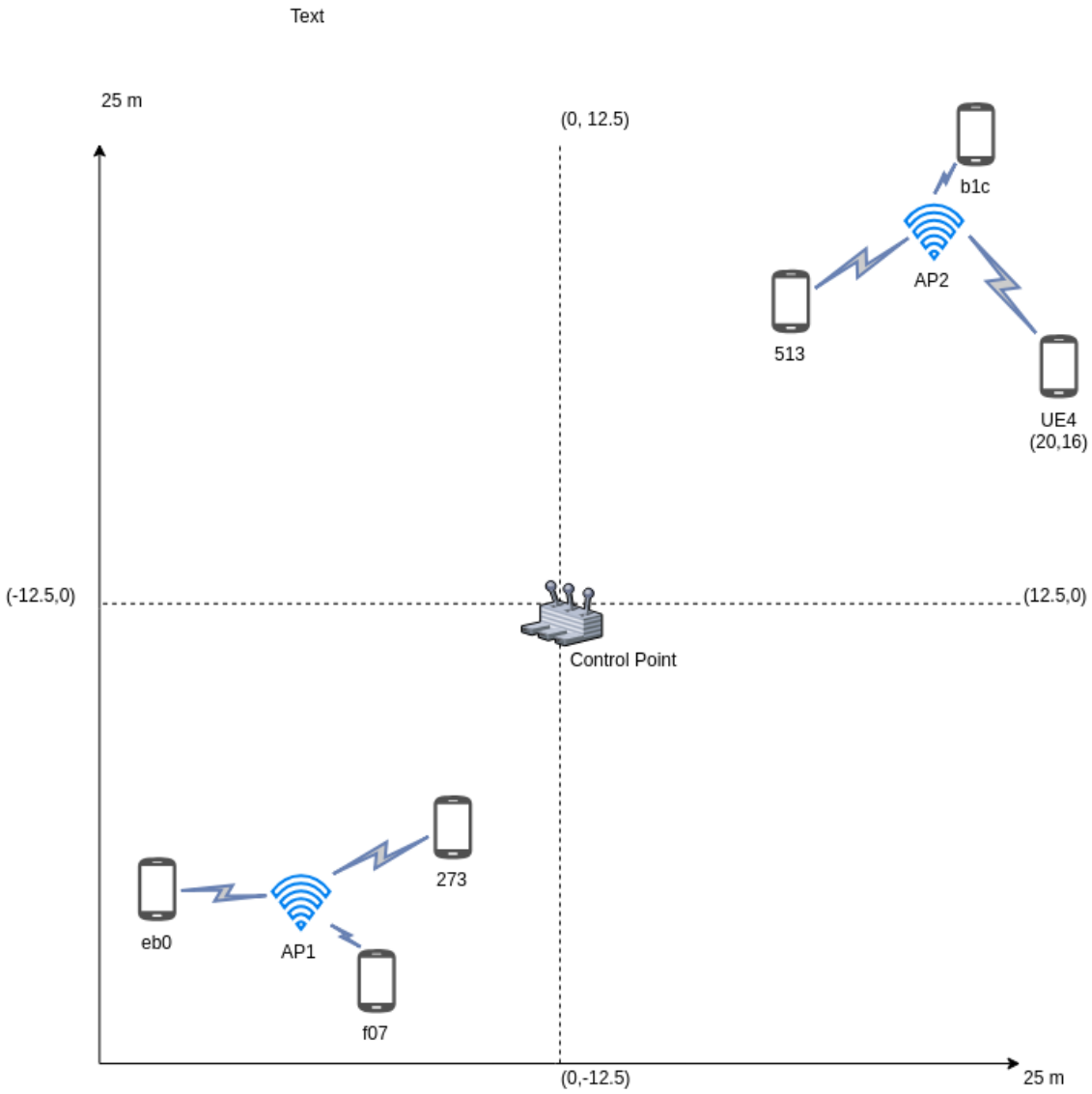


Figure 4: Near-optimal layout example

Case-2. Uniform layout

In the second case, we want to put the APs at the same distance and line from the CnC.

Here, both **APs** is set right on the line between the quarter of UEs and quarter where UE was alone.

Signal quality and transmission rate in this configuration expected to decrease compared to the previous case.

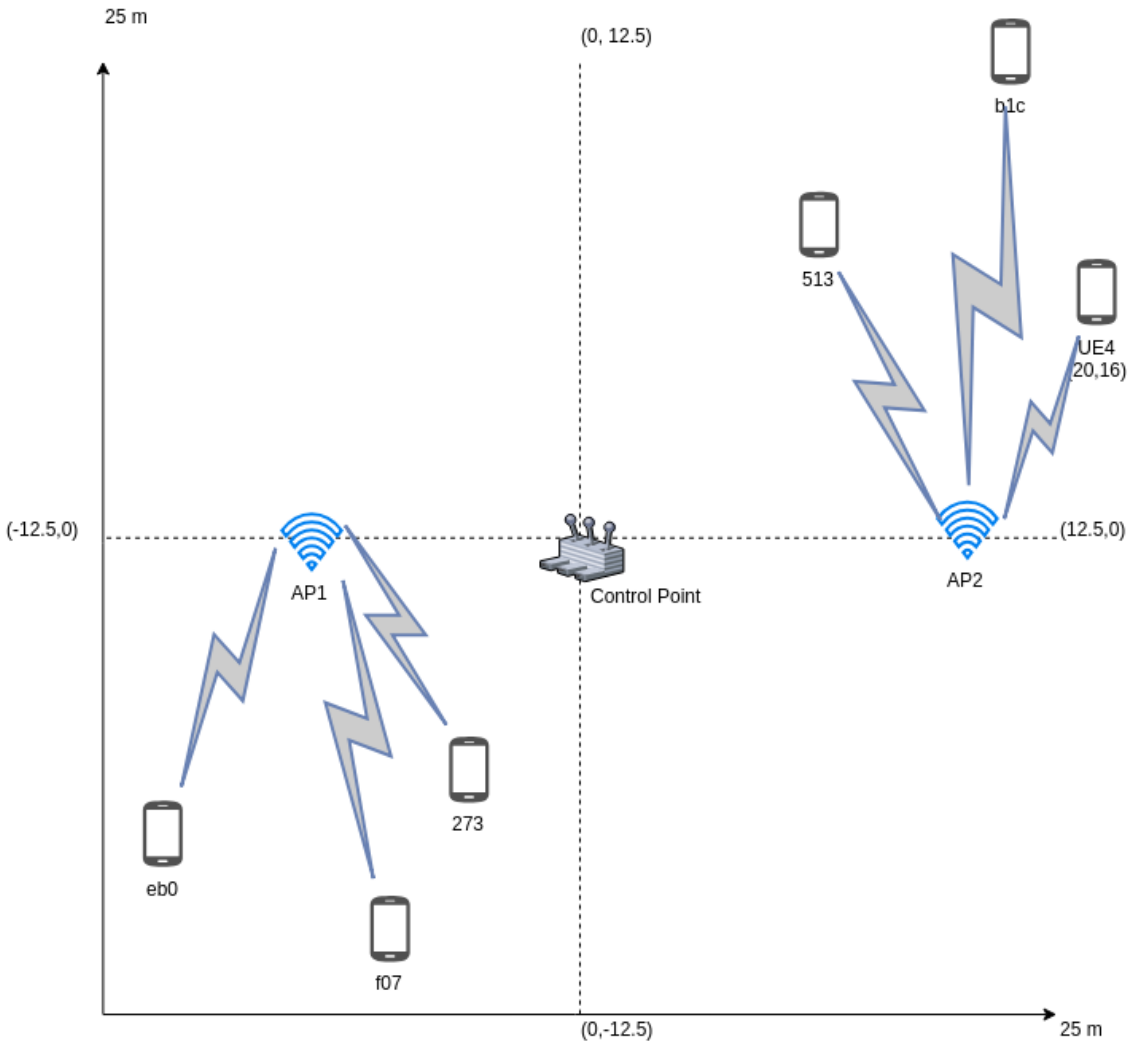


Figure 5: Uniform case layout example

Case-3. Near-optimal layout

In the third case, the only changed things are the position of **APs**.

Each of them now is put in the middle of each group of 3 UEs, therefore they have the same distance to centers of each cluster.

Consequently, RSS and link transmission rate expected to increase

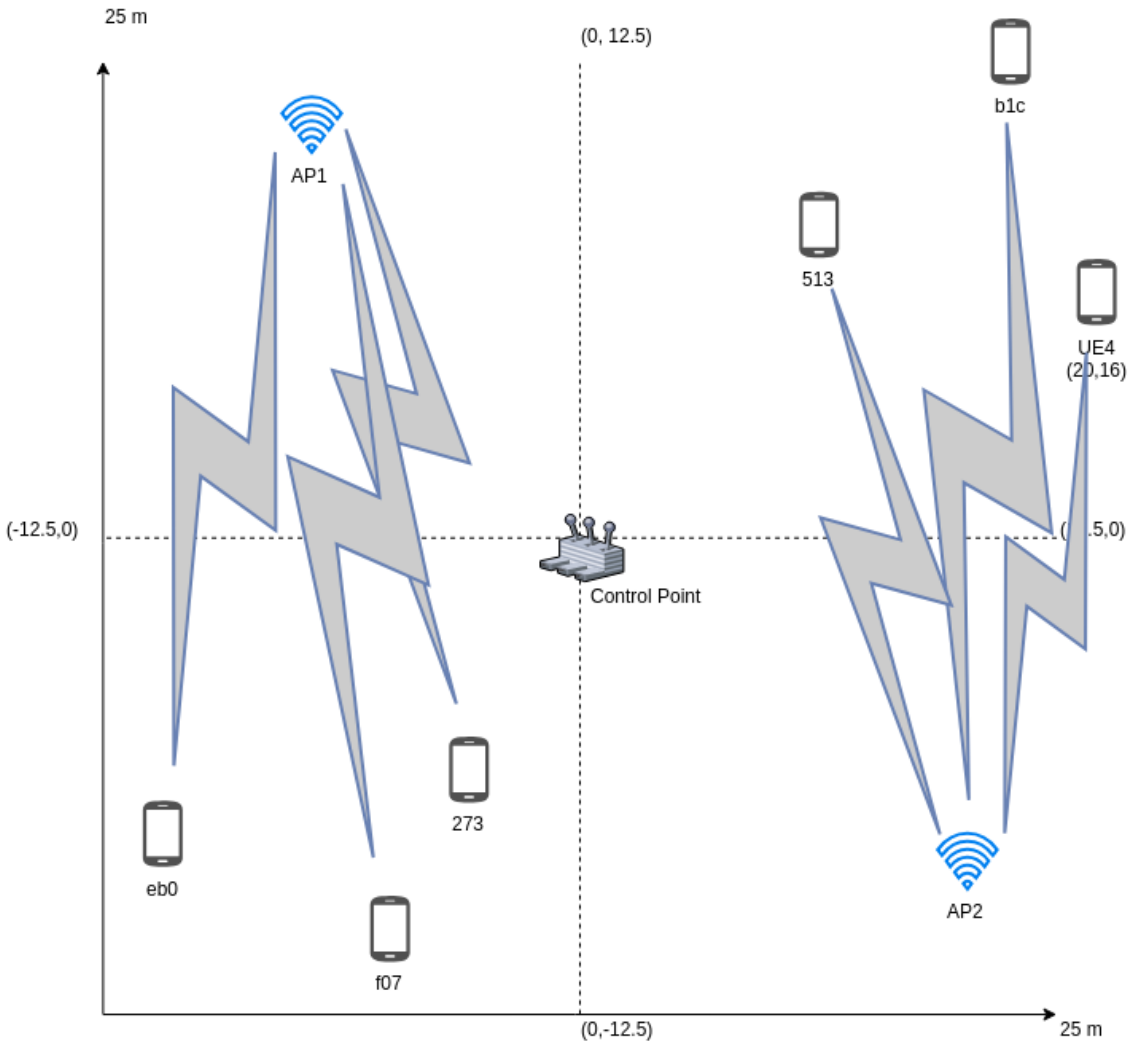


Figure 6: Near-optimal layout example

Experimental Phase

Experiment-1. 12.02.2020

The first attempt took place on 12.02.2020.

The aim was to involve as many UEs as possible. This is partly a reason for experiment day selection. Eventually, **6 UEs** took part ranging by the Android version from 4 to 9.

Besides, the installation of apps and binding Wi-Fi to the necessary access point, a detailed **journal of the Wi-Fi** connection information enabled. This allowed monitoring RSS value on each UE.

Weather conditions

- no precipitation
- cloudy sky
- a thin layer of snow on the ground

Procedure

All items of the experiment placed on the carton boxes on the ground, so that they are a bit raised from the ground.

Figure 7: Layout of CnC server

Case 1

Suboptimal scheme was chosen to begin with: UEs' connection to one AP was made successfully.

The second AP was connected to the other 3 UEs with more effort. The reason for it was in the Wi-Fi module issues of AP.

After some time of data collection, it was clear the second AP was not sending data to CnC at all, so we decided to locate devices according to the near-optimal scheme.

Case 2

This helped to increase the RSS level a bit from -82 and -84 to -76 and -80, data still was collected only from the first AP.

Besides, the graphical view in CnC showed 4 UEs close to each other (having approximately the same GPS coordinate), whereas the other 2 were detected much further.

Then we figured out that the second AP stopped working at some time. Restart and reconnection of UEs did not help to obtain data of that 3 UEs in CnC.

Case 3

It made no sense to try without data collection on CnC.

Outcome

The first attempt of the experiment showed:

- Further development of GPS_Tracker and GPS_Android to fix bugs required.
- Tests of placement algorithms performing, but due to problem with message communication, we were unable to test if the optimized positions would lead to better signal conditions.

As a result, we decided to:

- fix the second AP failure reasons
- figure out the source of 2 point-outliers in the plot

Experiment-1. 22.02.2020

The second attempt took place on 22.02.2020.

This time the aim was to check how **updates** in different parts of the system behave in real condition:

- a new way to estimate load speed in-app
- possibility to read app logs right on UE
- improved front-end part

The goal did not include a full-scale experiment with as many devices involved, so there were 1 CnC, 1 AP, and 3 UEs with the same settings from the previous experiment but updated app.

Weather conditions

- no precipitation
- clear sky
- no snow
- strong winds

Procedure

Only one case (near-optimal) was performed.

Figure 8: The initial CnC server position

All the UEs connections made successfully:

- deviceId assigned
- UE coordinates displayed

During the first half of the experiment time, after pressing 'push once' **upload/download speeds** were estimated and displayed, although the values seemed to be not realistic (300 000 kBit/s). The values were much smaller when CnC was not connected to the Internet.

Later, pressing 'push once' did not cause speed re-estimation, and logs contained error messages about issues related to the MQTT.

Outcome

The second attempt was also not successful. We hadn't managed to solve the problem in telemetry data sending, some design proposals can be a solution for the next iteration.

As a result, we decided to:

- add logging to APs as well
- implement direct HTTP requests

Experiment 3. 07.02.2020

Took place on 27.02.2020

This time the aim was to check how **2 major updates** of the Android app behave.

The *first* one concerned the usage of HTTP requests instead of MQTT. The reason for it is that all previously detected issues related to MQTT in one way or another.

The *second* update was about complete refactoring of the code. It included not only start following MVVM architectural pattern but also removing of redundant 'Connect' button, as well as real-time interaction with the app in the way that coordinates updates the display when 'Push continuously' is enabled.

Procedure

Due to the bad weather (heavy snowfall), we decided to experiment indoor (Mensa has enough space inside). 1 CnC, 1 AP, and 3 UEs took part.

All UEs can connect successfully:

- 'Push once' button pressed
- deviceId assigned
- UE coordinates displayed

Tips

As for 'Push continuously', it should be known in advance the coordinates updated on the display **only in the case of moving to some minimal delta** (10 centimeters). This is insured based on GPS values passed by the Android device.

To make sure the connection is still alive, and the values transferred, it makes sense to check logging messages in logs/log.txt

Outcome

All in all, the system finally started working as expected, and the meaningful set of data collected:

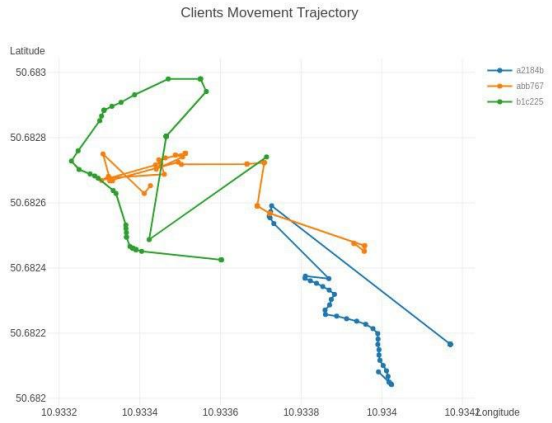


Figure 9: Movement trajectory of connected phones

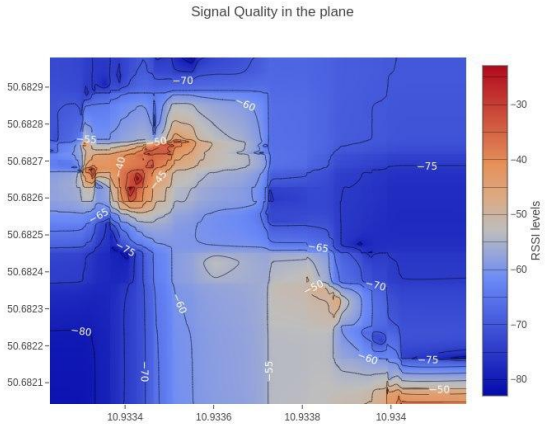


Figure 10: Signal quality map

Figure 11: Signal quality changes

Experiment 4. 03.03.2020

It took place on 03.03.2020. The weather condition is appropriate for running experiments, despite there was rain.

For the fourth attempt, we implement message sending only via HTTP. The main aim of the experiment is to try to place the UEs in three cases (sub-optimal, uniform, near-optimal). Phones forced to push their channel quality metric to the server for each case. On the server-side, GPS_Tracker tries to find optimal positions for UAVs supposing there are 2 AP is required.

We decided to reduce the area layout to 25x25 meters.

Results

THe HTTP protocol helps us to receive messages more reliable, although there was a problem, e.g. the far from AP a UE is the less probable the message receiving. Tests in experiment 3 showed the reliability of sending over HTTP. However, we decided to continue our experiment.

We found out that network speed measurement was not reliable, uplink tests throw the timeout exception in case of larger distance between a UE and used AP.

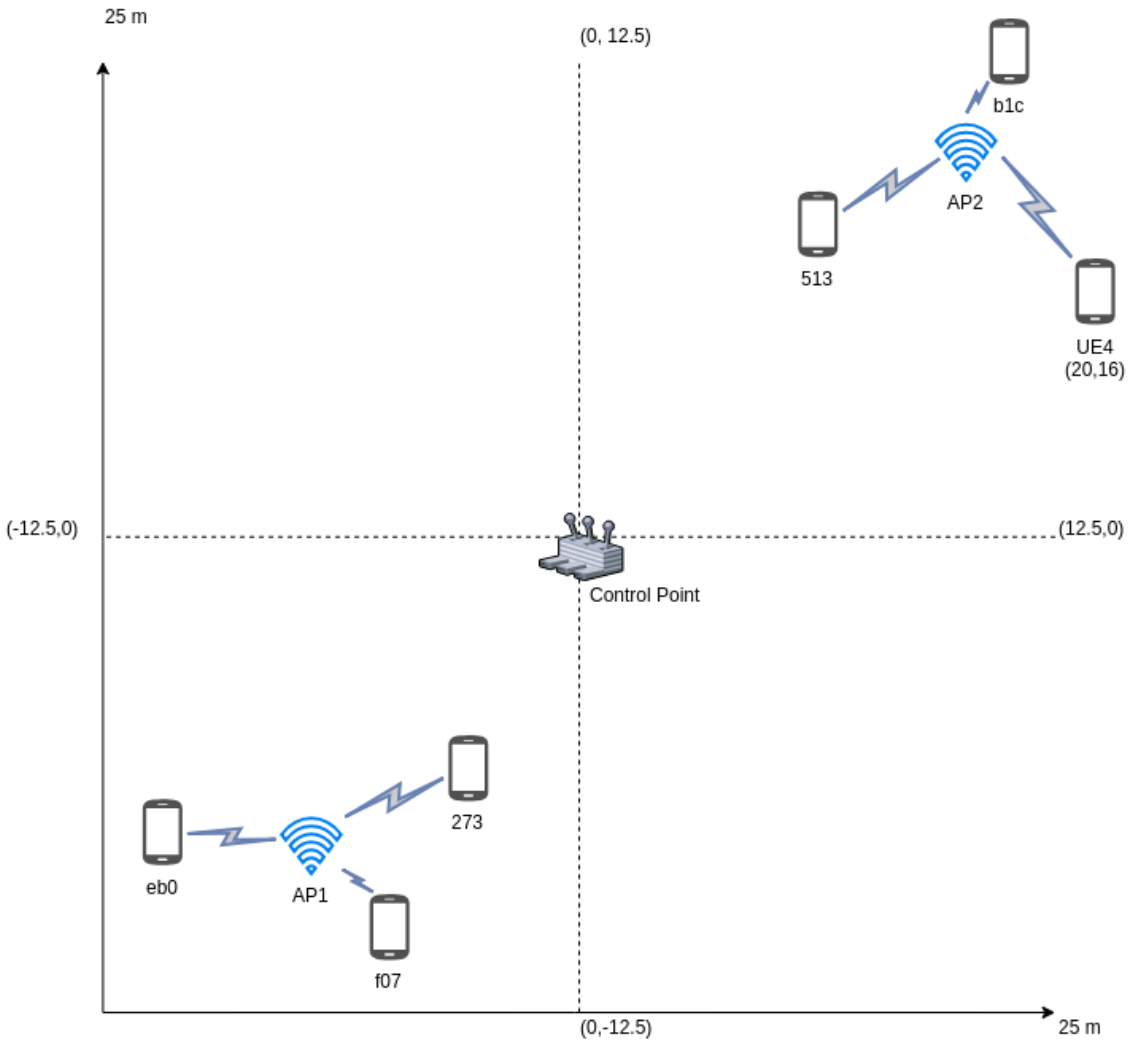


Figure 12: Experiment 4: Initial Layout

The measurements showed the following coordinates for the 6 UEs:

Table 6: Initial positions for UEs

Unique ID	Coordinates
f072f812f48ce468	(50.4056203, 10.5626057)
eb0b54819c69cf0c	(50.4056230, 10.5626126)
27349a2cde6592df	(50.4056445, 10.5625865)
51336504999bc1ca	(50.40567855, 10.5625238)
b1c225280d0ed13f	(50.4056924, 10.5624943)

The experiment is divided into four parts:

- Before 11:05 - suboptimal case
- 11:05 - 11:08 - uniform case
- 11:08 - 11:12 - near-optimal case
- 11:12 - 11:15 suboptimal to compare to the suggested positions

Suboptimal case

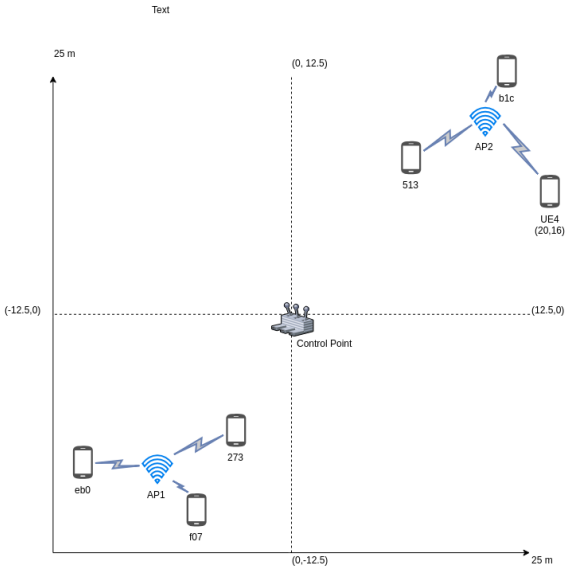


Figure 13: Suboptimal Layout

The first case is the most profitable from the signal quality point of view. The APs are implicitly located at the same distance from the connected UEs. The level of interference is low.

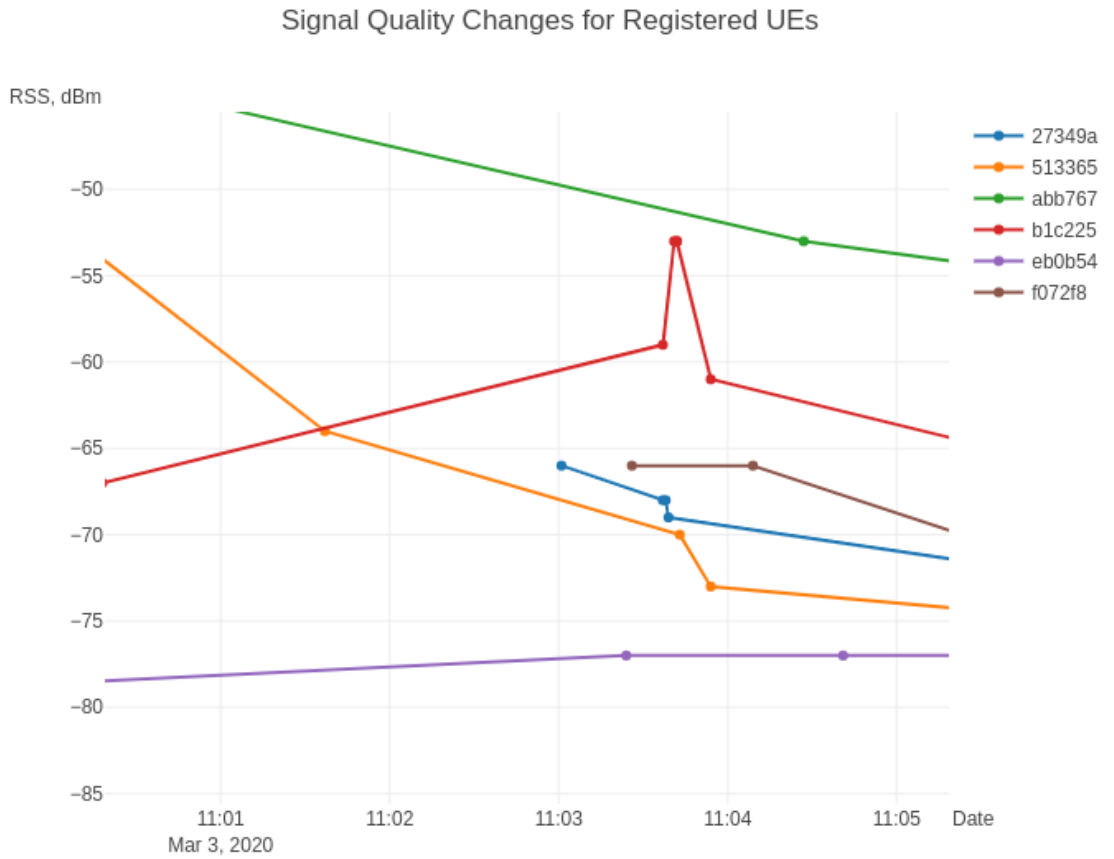
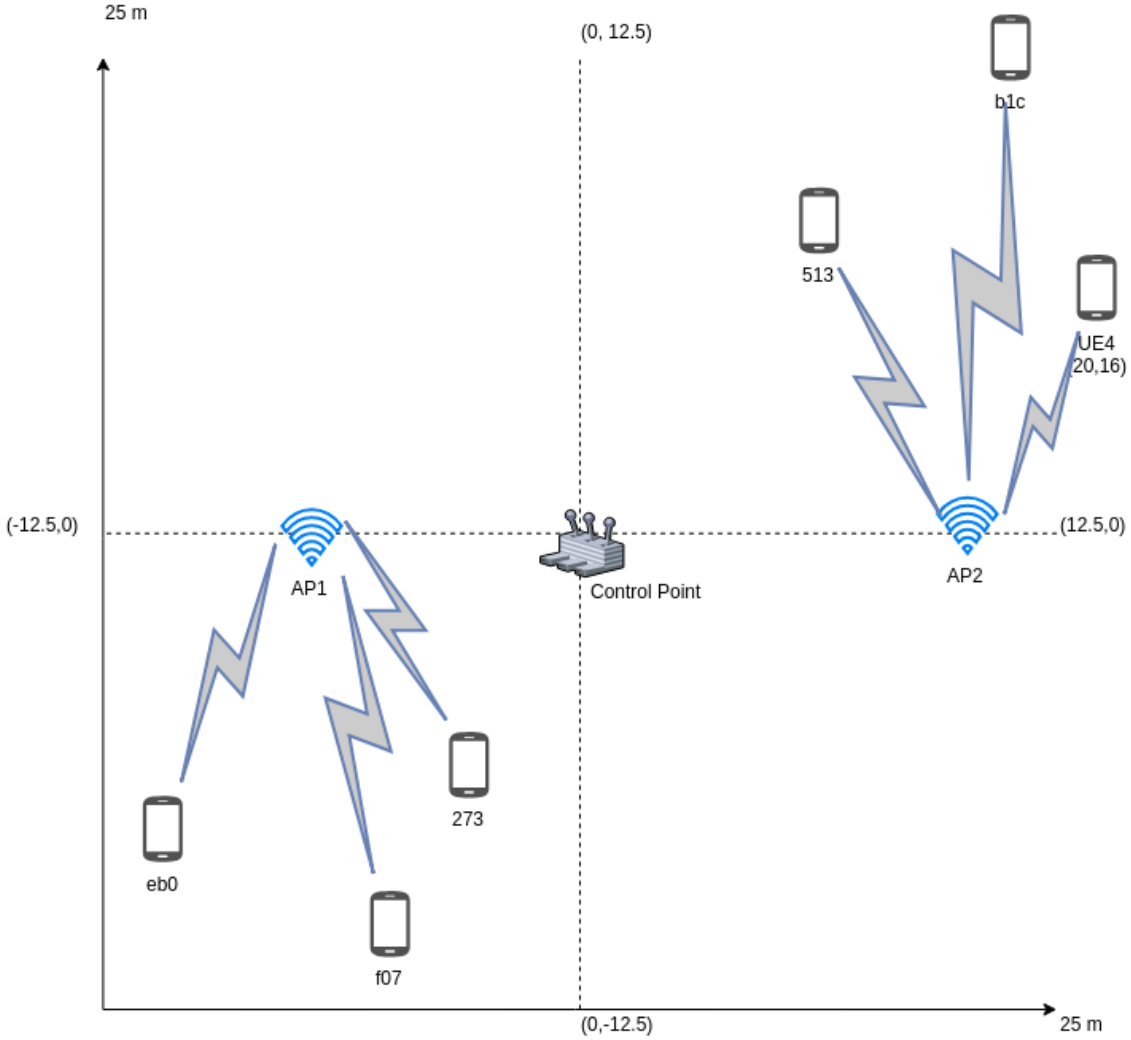


Figure 14: Signal Quality changes in Suboptimal case

Despite the APs were located close to UEs, the RSS level varies markedly. However, only in this case the speed and signal quality measurement were the most stable among all cases.

Uniform case

In this case, the APs are located with equal distance from the CnC on the same line.



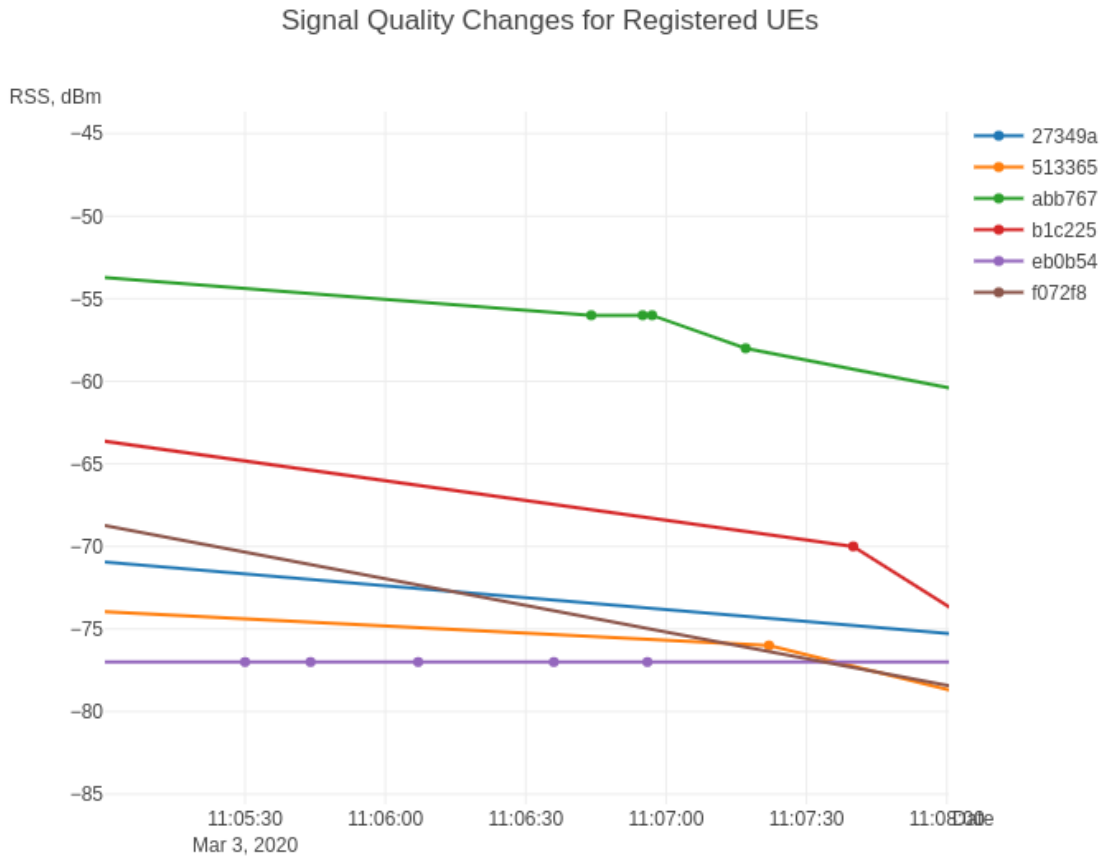


Figure 15: Signal Quality changes in Uniform case

Link measurement became more smooth, but the speed test failed sometimes.

Near-optimal case

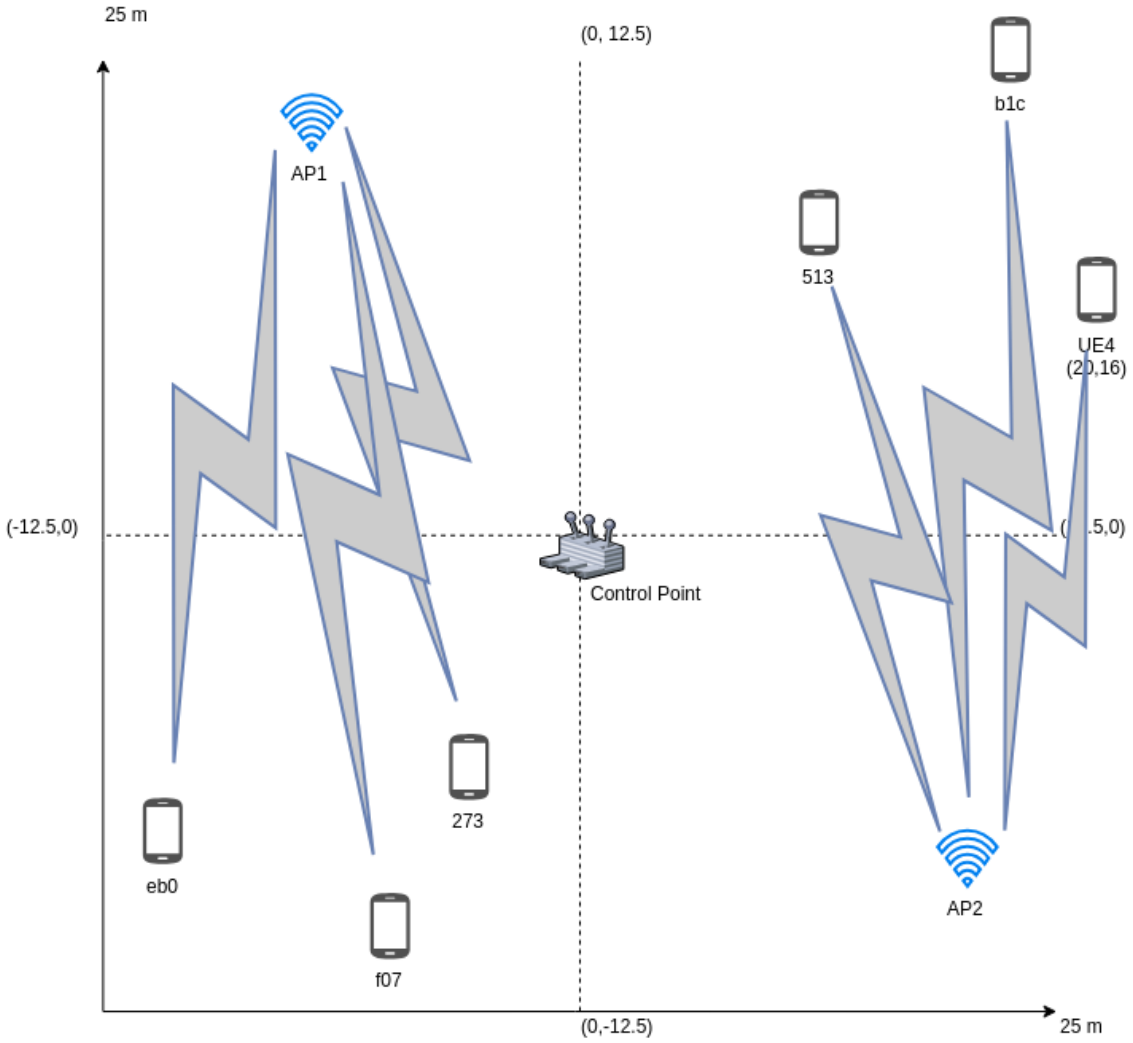


Figure 16: Suboptimal Layout

The third case simulates the situation where the APs are placed uniformly far from centers of UEs' clusters.

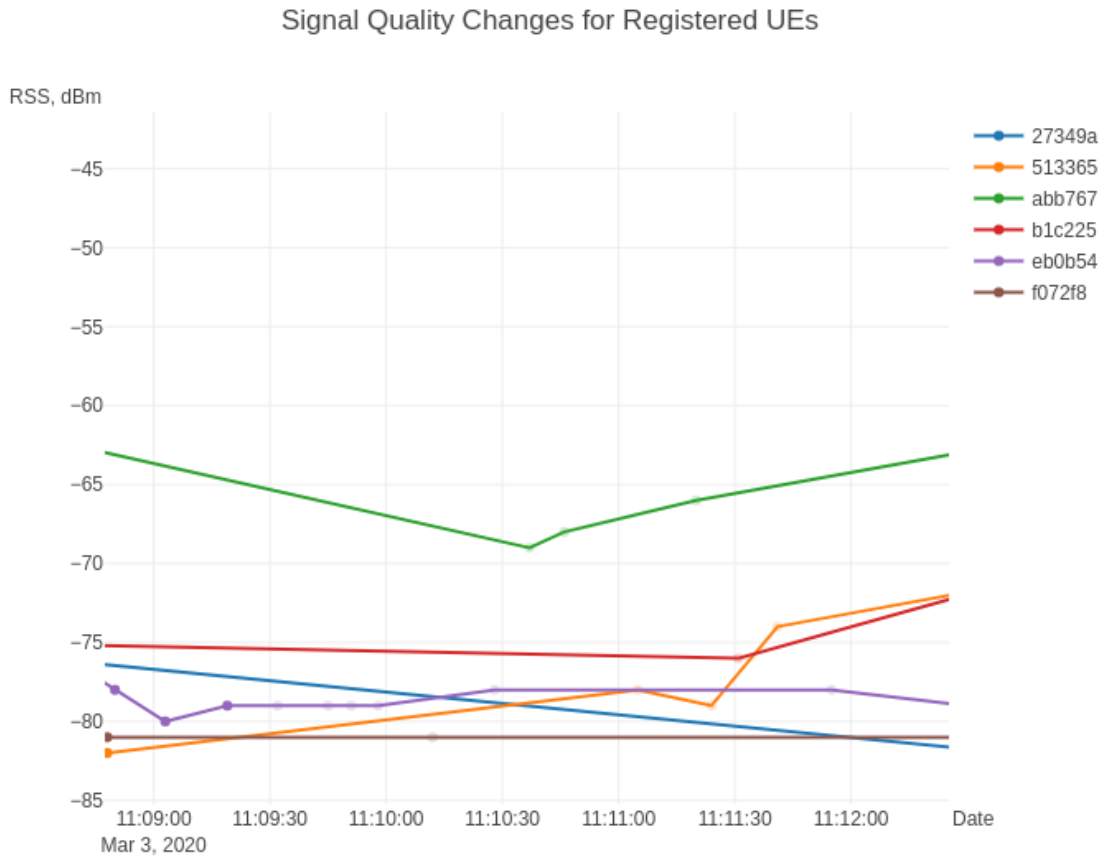


Figure 17: Signal Quality changes in Near-optimal case

Comparison between Suboptimal case, and the suggested positions

To check the validity of the UAVs layout optimization algorithm, we place the APs in a Suboptimal case position.

The given coordinates for **APs**:

AP	Coordinates
AP1	(50.4056741, 10.5625129)
AP2	(50.4056339, 10.5625975)

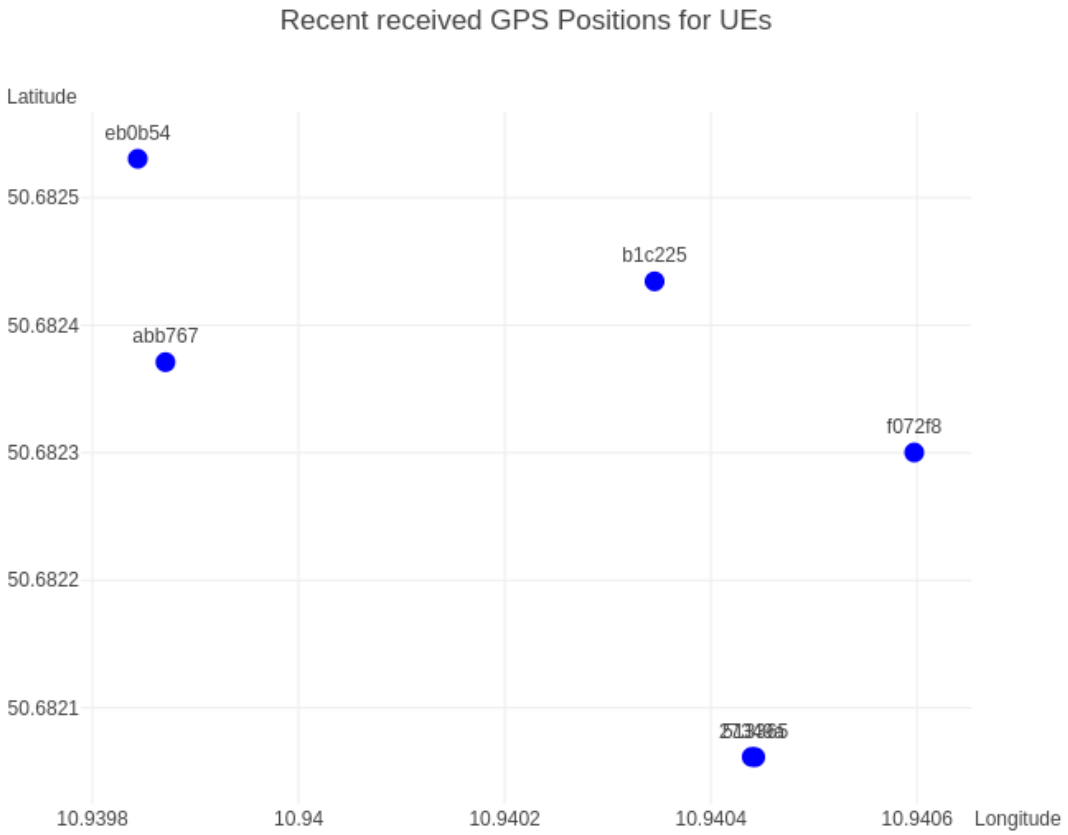


Figure 18: The APs coordinates used to optimize UAVs positions

The coordinates for **27349a2cde6592df** and **51336504999bc1ca** overlap.

The most recent received coordinates for UEs used to schedule an optimization task with the following parameters:

- Number of clusters: 2
- Estimation method: “clustering”

Using of “simplex” method against two clusters is not possible, because that method for 2 clusters cannot converge, the produced result is unreliable.

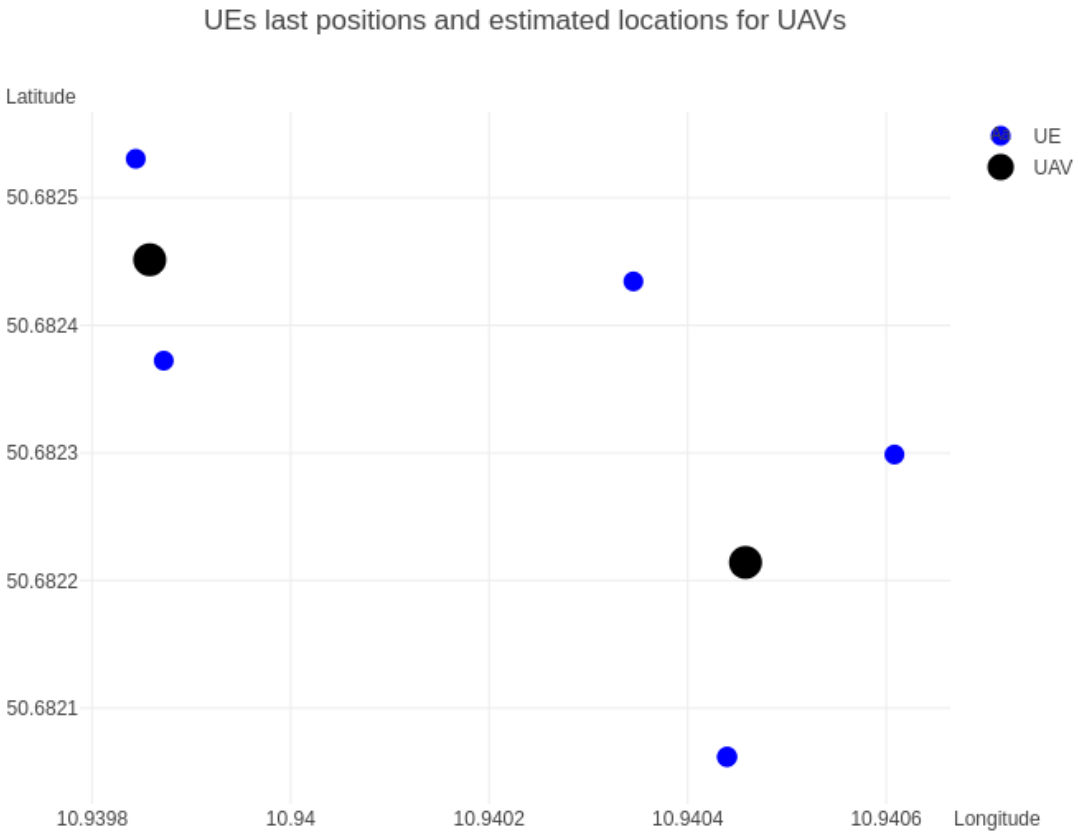


Figure 19: Black dots represent suggested points as the most optimal position for UAVs

The suggested optimal coordinates for **APs**:

AP	Real coordinates	Optimal coordinates
AP1	(50.4056741, 10.5625129)	(50.68221, 10.94046)
AP2	(50.4056339, 10.5625975)	(50.68245, 10.93986)

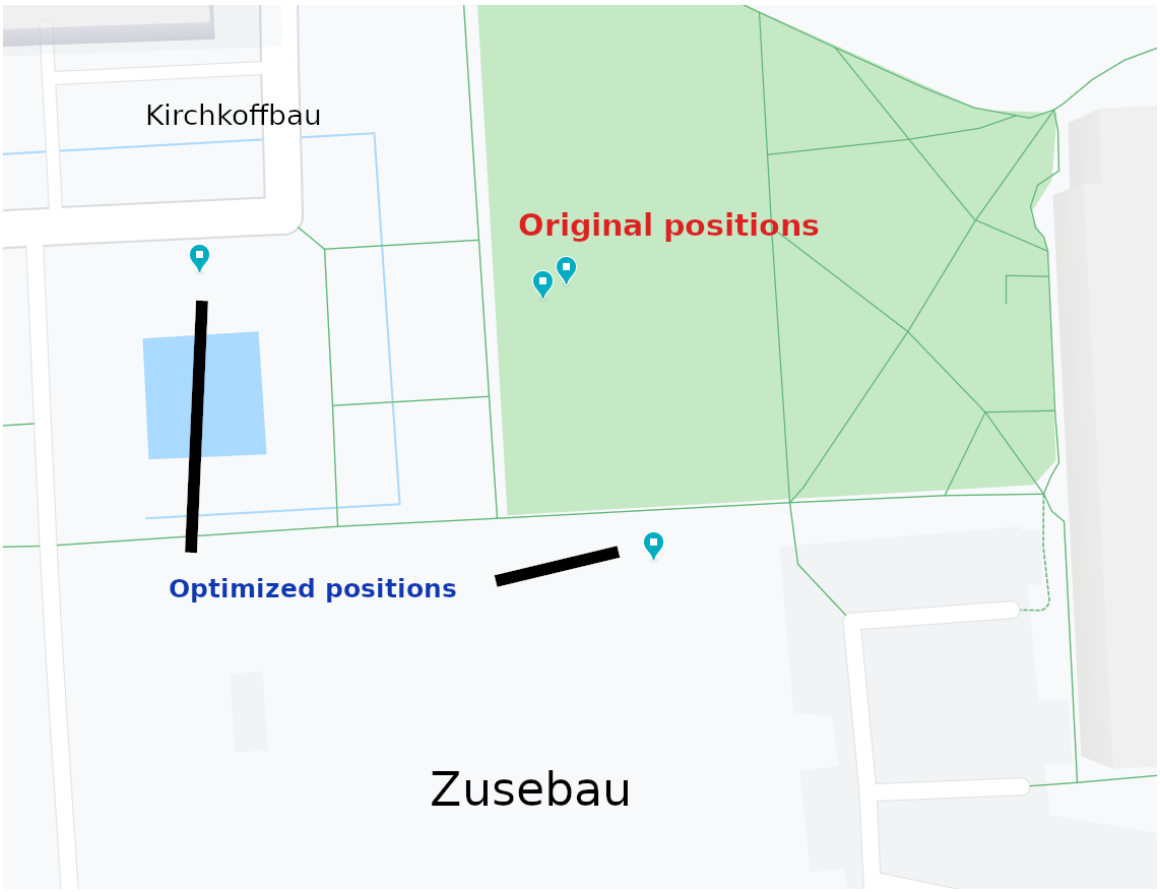


Figure 20: Original and Optimized Coordinates: A map with tags. Clearly seen the suggested "optimal" position can drop out connected clients because of a large distance.

Figure 21: Original and Optimized Coordinates for UAvs: A picture from satellite

"Clustering" algorithm performs simple K-means cluster calculation based on GPS coordinates for UEs. The result provides insights about that layout optimization algorithms relying solely on the coordinate input data can result in a biased solution.

Pictures of experiment

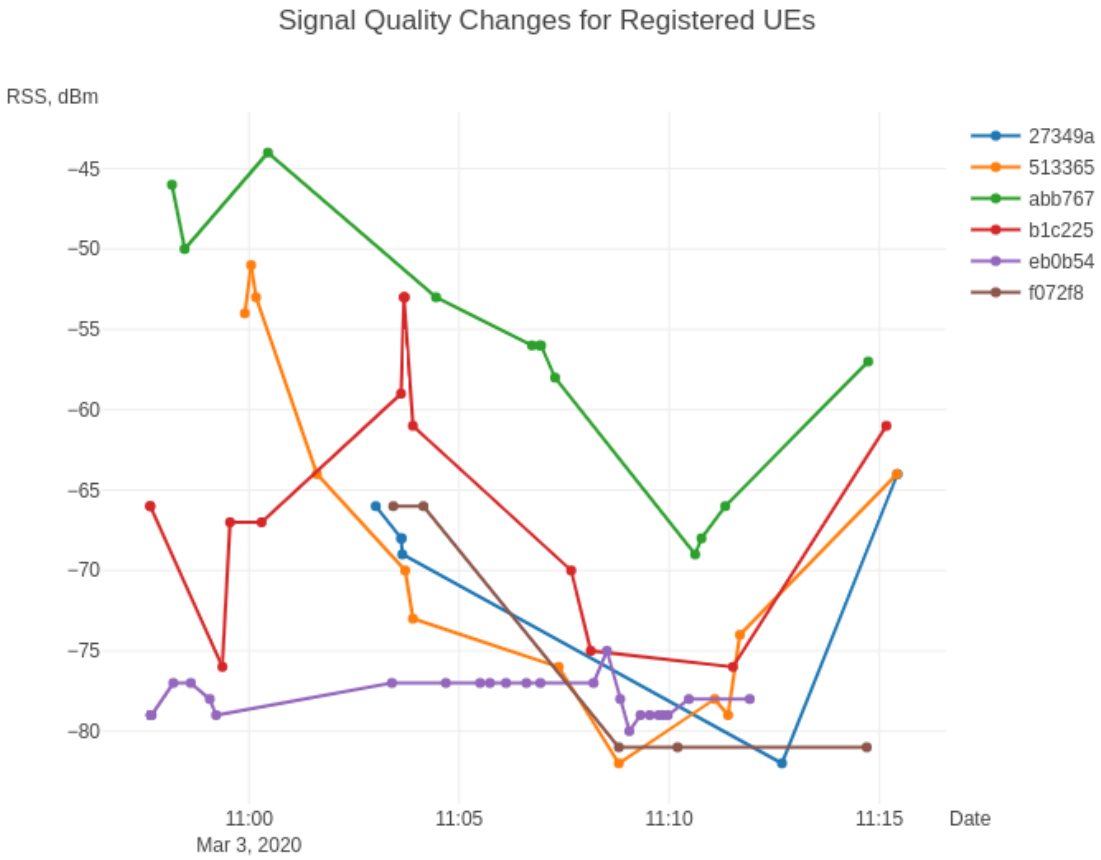


Figure 22: Signal Quality level changes during the experiment

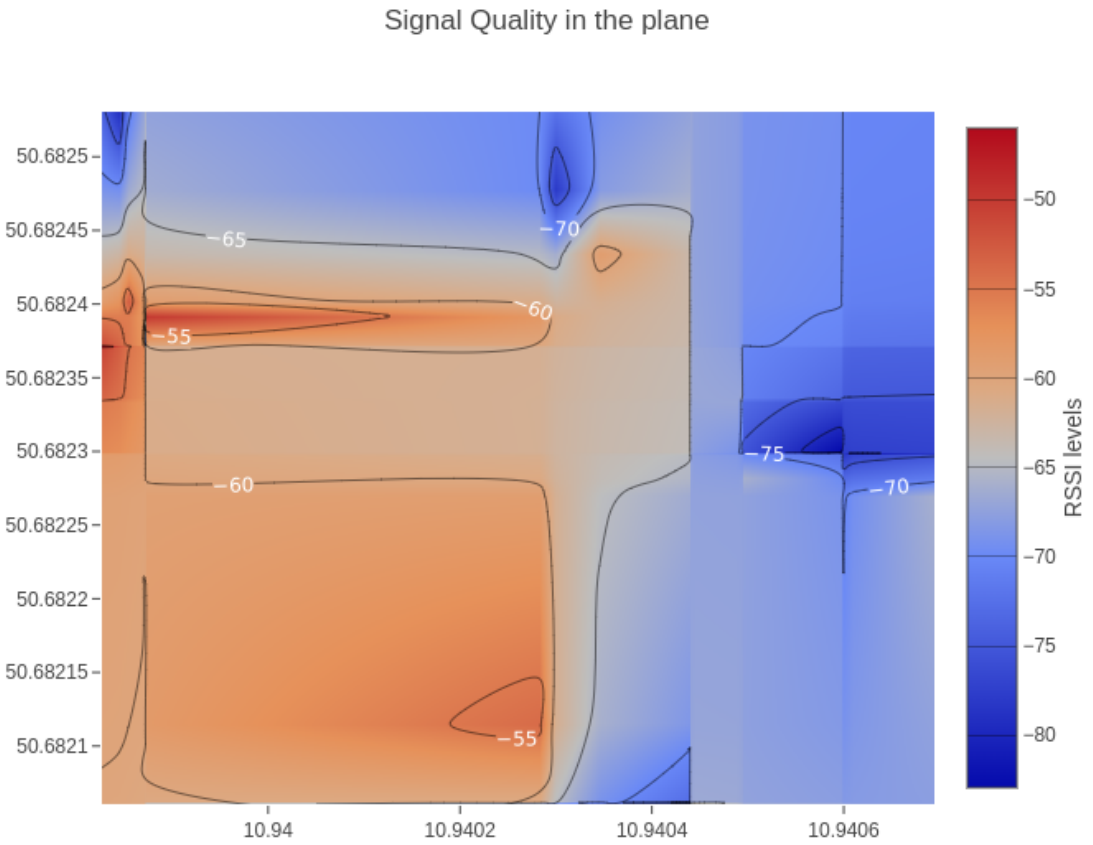


Figure 23: Signal Heatmap obtained during the experiment

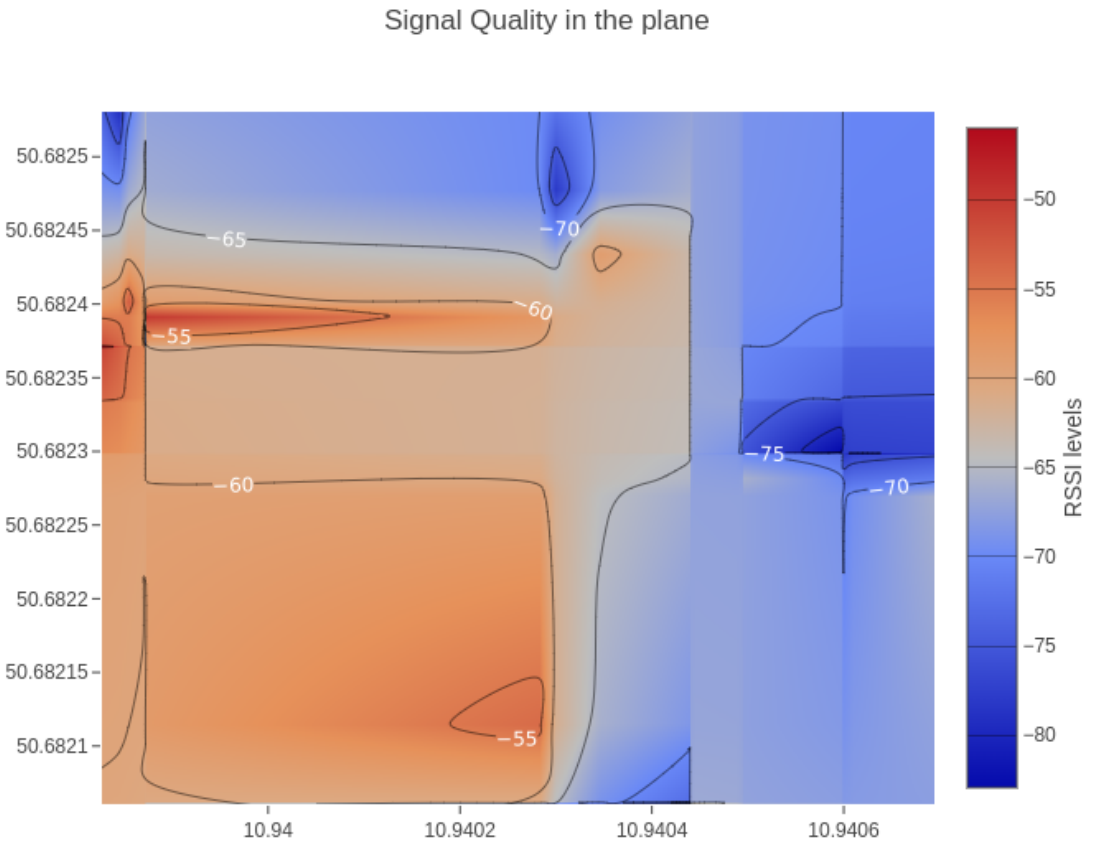


Figure 24: Signal Heatmap obtained during the experiment

Figure 25: Layout Problem discussion

Outcome

Finally, we have managed to test the framework and estimate an optimized position for UAVs which were represented by WiFi access points. The results showed that the framework, as well as optimization algorithms, should be redesigned and evaluated on updated data.

Final Results

We have completed 4 experimental attempts. We encountered problems each attempt, mostly they cover:

- Wi-Fi connection: radio link may suffer serious collisions that drop up data transmission.
- Overcomplicated architecture: the initial proposal architecture is quite sophisticated, but it didn't match the given functional requirement. Further, we figured out a possible simplified solution that gave us an opportunity to run experiments correctly, however, there are still gaps to fill in.
- Optimization algorithms: higher accuracy requires additional data to supply and more advanced algorithms to perform. Even so, there should be a strict evaluation of the results they produce if that is correct.

Despite these problems, we can conclude that the developed framework can be applied for performing UAVs layout optimization, the current version has enough capabilities to provide access to investigate and analyze the measured data. The algorithms included in the current version of the framework have restrictions, simplified and produce biased results, but the contrary has an ability to be highly modified to obtain more reliable results. The extensibility of the platform gives the opportunity to extend the current set of optimization algorithms in a simple way.

Experiment report

Sections

Usage

To produce the full doc in latex:

```
pandoc *.md -H disable_float.tex -o experiment_report.pdf
```