Indoor Positioning Data Exploration

STAT4/510 Wasabees Group

2023-11-30

1. Describe your data
2. Describe the basic variable component of your data
3. Report the various findings you have established so far, with interprtation (include discussion on what you find and add how useful it is to your project objective)
4. Discuss any challenges you encountered, and ways by which you handled these.
5. Introduction

Indoor position systems (IPS) development is an active area of research that can be used in numerous settings. Part of efforts to develop, calibration and model these systems are achieved throught the use of WIFI signals. The following report, describes and characterizes a large data set compiled in a 15 by 36 meter area that contains six (wifi routers) access points, signal strength, various locations and orientation of the devices. The data is subdivided in two sub-sets, one denominated "offline data", which corresponds to various testing devices connected to the network at different locations and orientations, and an "online data", where 60 locations and orientations of the devices were selected at random.

The offline data was collected designing a 1 meter resolution grid, resulting in 166 locations. In each of these locations, the device was oriented starting at 0 degrees inclination and at 45 degrees increments, and the strength signal measured. Furthermore, each combination of location/orientation was sampled 110 times. This grid sampling is intended to be used to calibrate a indoor positioning model. On the other hand, the online data was designed to simulate real-world data, in which locations are not bounded by the 1 meter grid used in the offline data, and were selected at random. This randomization included the orientation of the device and therefore, the online data consists of 60 randomly selected location/orientation combinations sampled 110 times.

More details of the floorplan, and location of online and offline data can be seen in Figure 1.

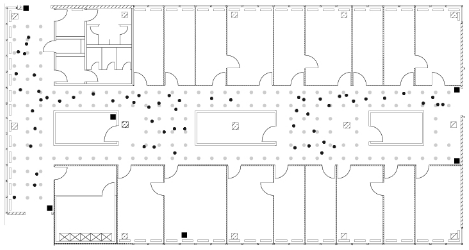

Figure 1: Figure 1: Flooplan location. Access points are squares. Grey dots are offline data locations and black dots are online data locations.

```r
#Read the offline data txt file, with each row in the txt file becoming a text string
raw_offline = readLines("Data/offline.final.trace.txt")
head(raw_offline)
```

```
## [1] "# timestamp=2006-02-11 08:31:58"
## [2] "# usec=250"
## [3] "# minReadings=110"
## [4] "t=1139643118358;id=00:02:2D:21:0F:33;pos=0.0,0.0,0.0;degree=0.0;00:14:bf:b1:97:8a=-38,2437000000
## [5] "t=1139643118744;id=00:02:2D:21:0F:33;pos=0.0,0.0,0.0;degree=0.0;00:14:bf:b1:97:8a=-38,2437000000
## [6] "t=1139643119002;id=00:02:2D:21:0F:33;pos=0.0,0.0,0.0;degree=0.0;00:14:bf:b1:97:8a=-38,2437000000
```

```
#The txt file contains multiple rows starting with # (for example, rows 1, 2, 3) which does not share t

clean_offline <- raw_offline[substr(raw_offline,start=1, stop=1) != "#"]
str(clean_offline)
```

```
##  chr [1:146080] "t=1139643118358;id=00:02:2D:21:0F:33;pos=0.0,0.0,0.0;degree=0.0;00:14:bf:b1:97:8a=-
```

```
#After this code, clean_offline has 146,080 rows, less than the original 151,392 rows
```

```
#In clean_offline, each row now has the same structure:
#Semicolons separate between variables
#Equal signs separate variable names and values
#For the signal readings, readings of the same access point are separated by commas

#The code below took all the separation out of one line (line 1)
line = strsplit(clean_offline[1], "[; , =]") [[1]]
line
```

```
##  [1] "t"                 "1139643118358"     "id"
##  [4] "00:02:2D:21:0F:33" "pos"               "0.0"
##  [7] "0.0"               "0.0"               "degree"
## [10] "0.0"               "00:14:bf:b1:97:8a" "-38"
## [13] "2437000000"        "3"                 "00:14:bf:b1:97:90"
## [16] "-56"               "2427000000"        "3"
## [19] "00:0f:a3:39:e1:c0" "-53"               "2462000000"
## [22] "3"                 "00:14:bf:b1:97:8d" "-65"
## [25] "2442000000"        "3"                 "00:14:bf:b1:97:81"
## [28] "-65"               "2422000000"        "3"
## [31] "00:14:bf:3b:c7:c6" "-66"               "2432000000"
## [34] "3"                 "00:0f:a3:39:dd:cd" "-75"
## [37] "2412000000"        "3"                 "00:0f:a3:39:e0:4b"
## [40] "-78"               "2462000000"        "3"
## [43] "00:0f:a3:39:e2:10" "-87"               "2437000000"
## [46] "3"                 "02:64:fb:68:52:e6" "-88"
## [49] "2447000000"        "1"                 "02:00:42:55:31:00"
## [52] "-84"               "2457000000"        "1"
```

```
#The code below takes the time, id, pos, and degree readings out of "line" and arrange them in a matrix
line[c(2,4,6,7,8,10)] %>%
  matrix(ncol=6,byrow=TRUE) %>% view()

#The code below takes the mac, signal, frequency, and device type reading out of "line" and arrange the
line[-(1:10)] %>%
  matrix(ncol=4, byrow=TRUE) %>% view()
```

```r
#After this, we can combine the two matrices to create one row of data for our ideal data frame.

#The code below attempts to put all the above operations in this code chunk in one function
clean_function = function(x) {
  line = strsplit(x, "[; , =]") [[1]]
  a = line[-(1:10)] %>% matrix(ncol=4, byrow=TRUE)
  b = line[c(2,4,6,7,8,10)] %>% matrix(ncol=6, nrow=nrow(a), byrow=TRUE)
  cbind(b,a)
}

#Test clean_function with row 1 in clean_offline
lapply(clean_offline[1], clean_function) [[1]] %>% view()
```

```r
#This code chunk will use clean_function for all rows in clean_offline and rbind them together
```