Real-Time Indoor Location Positioning

Justin Howard, Paul Adams, and Stuart Miller

September 8, 2020

1 Introduction

Real Time Location Systems (RTLS) that are capable of tracking business assets and people in special circumstances, are a popular area of research. Warehouse distribution and delivery services have grown more rapidly due to the COVID-19 pandemic, and with that growth, we can expect the relative importance of tracking assets at scale to increase. In this case study, we assess whether it is possible to predict locations based on the measurement of WiFi signals from fixed access points (AP)[1]. Additionally, two APs are placed in close proximity, we assess the impact of using both of these APs. The location sensing system analyzed in this case study is located in an indoor office environment on a single floor of a building.

2 Methods

2.1 Data

2.1.1 Data Collection

The data was collected in an indoor office environment on a single floor with 7 fixed WiFi APs. The WiFi APs are identifiable by their media-access-control (MAC) addresses. It is shown by Nolan and Lang (2015) that two of these APs (`00:0f:a3:39:e1:c0` and `00:0f:a3:39:dd:cd`) are positioned in close proximity. The experiment area was split into a grid 166 measurement points, which were spaced 1 meter apart. The points of measurement are referred to as `posX` and `posY`. Using a handheld WiFi scanning device, WiFi signal strength measurements were collected at each of the 166 locations. Additionally, the handheld WiFi scanning device was rotated to 8 angles (45°increments from 0) at each measurement location. The resulting dataset contains approximately one million measurements.

2.1.2 Data Cleaning

The data are provided as measurement logs with the MAC address of the scanning device, measurement position, measurement orientation, AP MAC address, AP signal strength, and timestamp on a single line. We extracted the data by reading each line and tokenizing each unit of information. The data was initially organized in a long format data frame with time, position X, Y, and Z, orientation, AP MAC address, signal strength, and channel as the columns. Once the data was organized, we were able to determine that several columns were not necessary. Specifically, we removed position Z and the scan MAC address because these

[1] This analysis was conducted with the R programming language (R Core Team (2013)).

were fixed throughout the experiment and channel because each AP used a single channel throughout the experiment. We also found that several MAC addresses were coded as *adhoc* (`type=1`). These were removed since we are only interested in positioning based on the fixed APs (`type=3`). Additionally, we determined that there some noise present in the orientation measurement. The experiment specified that only 8 angles were considered (each 45°increment from 0). All orientation values were rouned to the nearest 45°increment. Any missing signal values were filled with -100, imputing it as a very low strength signal.

2.2 Modeling Method

We chose to model the position as a function of the WiFi singal strength with k-nearest neighbors (k-NN). Intuitively, a k-NN model will predict a new position as the average positions of the closest k data points in the training data based on the WiFi signals. Since the features are microwaves, which can pass through physical barriers, we used euclidean distance to determine the closest signals. To use a k-NN, the value of k most be estimated based on the training data. We used 5-fold cross validation to determine the value of k, selecting the value of k that produced the lowest error based on root meean squared error (RMSE).

An advantage of k-NN is that it is non-parametric, thus it not necessary to assess whether the data exhibit certain distributional properties. However, a potential disadvantage of k-NN is the computational complexity, which is given by (1) for large N. This time complexity means a k-NN would not be suitable for modeling position in large areas as the number measurement points would be large.

$$O\left(n\right) \sim \frac{n^2}{2} \tag{1}$$

2.3 Model Validation

(Describe how you estimated your error and found the best fit ASSUMING you CANNOT USE THE ONLINE DATA.)

kold cross validation

3 Results

As shown in Fig. 1, we found that `k` of 4 or 5 minimized the RMSE of all three models. Based the CV results, 4 was chosen as the value of `k` for the `k`-NN model.

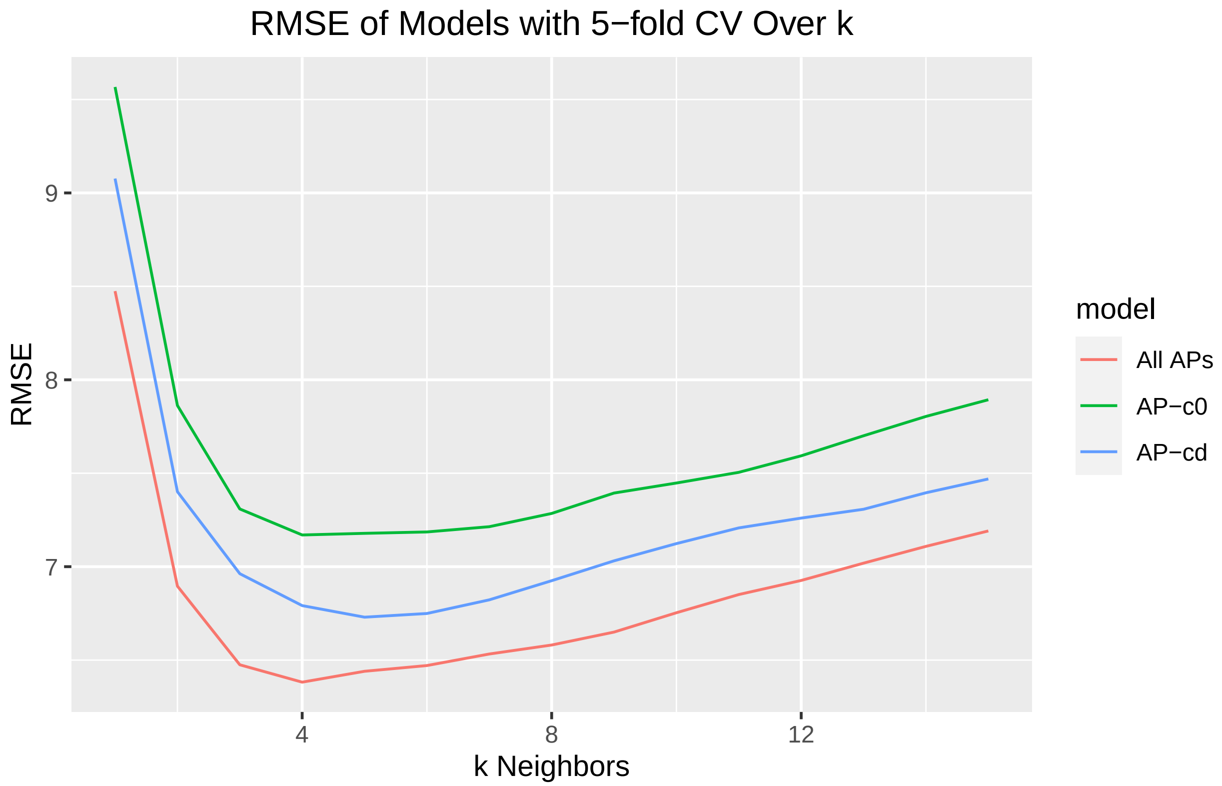

Figure 1: The mean RMSE of 5−fold cross validation for k−NN models using all APs (All APs), all APs excluding
and all APs excluding 00:0f:a3:39:e1:c0 (AP−cd) for k values 1 through 15.

What about using them both?

- There are 2 macIDs located at the same position.

- Does one give better performance than the other?
- (00:0f:a3:39:e1:c0, 00:0f:a3:39:dd:cd)

Use c0, then use cd, than use cd and c0

a plot with all three models

AP Feature Set	Mean Squared Error
Excluding `00:0f:a3:39:dd:cd`	XX
Excluding `00:0f:a3:39:e1:c0`	XX
All APs	XX

4 Conclusion

(What is the drawback (if any of using this method to real-time locate an object)?)

KNN is slow. calculation scales as n^2/2. Considers the whole dataset.

Describe a method that may be an improvement based on your perceived drawbacks.

Linear regression, assuming the assumptions are met

- Variance of the residuals is constant
- Normally distributed residuals
- No outlier + leverage points

Given that electromagnetic signals fall off as $1/r^2$, we could potentonially use polynomial regression.

A Code

The code used to produce the analysis in shown below.

```
#' Process a single line of the data files.
#'
#' @description
#' Produces a list of matricies from a data line.
#' The columns are as follows:
#' time, scanMAC, positionX, positionY, positionZ,
#' orientation, MAC, signalRSSI, channel, router_type
#'
#' @param x A file line
#'
processLine = function(x)
{
  # tokenize line on delimiters ;=,
  tokens = strsplit(x, "[;=,]")[[1]]
  # return null when there are no measurements
  if (length(tokens) == 10)
    return(NULL)
  # get matrix of measured RSSI
  tmp = matrix(tokens[ - (1:10) ], , 4, byrow = TRUE)
  # add handheld device data and return resulting matrix
  cbind(matrix(tokens[c(2, 4, 6:8, 10)], nrow(tmp), 6,
               byrow = TRUE), tmp)
}

#' Round the measurement angle to the nearest 45 deg.
#'
#' @param angles a vector of angles; expected 0 - 360
#'
roundOrientation = function(angles)
  {
  # create a sequence of reference angles
  # 0, 45, 90, ... , 315
  refs = seq(0, by = 45, length  = 9)
  # round angles to the closest reference value
  q = sapply(angles, function(o) which.min(abs(o - refs)))
  c(refs[1:8], 0)[q]
}
```

```
#' Load data files for this case study.
#'
#' @description
#' Produces a data.frame with the following columns
#' "time", "posX", "posY", "orientation", "mac", "signal",
#' "rawTime", "angle"
#'
#' Only regular access points are kept (router_type==3).
#' Time is converted from milliseconds to seconds.
#'
#'
#' @param filename The path to the data file
#' @param subMacs A vector of MAC addresses to use in the data
#'
readData =
  function(filename = './offline.data.txt',
           subMacs = c("00:0f:a3:39:e1:c0", "00:0f:a3:39:dd:cd", "00:14:bf:b1:97:8a",
                       "00:14:bf:3b:c7:c6", "00:14:bf:b1:97:90", "00:14:bf:b1:97:8d",
                       "00:14:bf:b1:97:81"))
  {
    txt <- readLines(filename)
    lines <- txt[ substr(txt, 1, 1) != "#" ]
    tmp <- lapply(lines, processLine)
    offline <- as.data.frame(do.call("rbind", tmp),
                             stringsAsFactors= FALSE)

    names(offline) <- c("time", "scanMac",
                        "posX", "posY", "posZ", "orientation",
                        "mac", "signal", "channel", "type")

    # keep only signals from access points
    offline <- offline[ offline$type == "3", ]

    # drop scanMac, posZ, channel, and type - no info in them
    dropVars <- c("scanMac", "posZ", "channel", "type")
    offline <- offline[ , !( names(offline) %in% dropVars ) ]

    # drop more unwanted access points
    offline <- offline[ offline$mac %in% subMacs, ]

    # convert numeric values
    numVars <- c("time", "posX", "posY", "orientation", "signal")
    offline[ numVars ] <- lapply(offline[ numVars ], as.numeric)
```

```
  # convert time to POSIX
  offline$rawTime <- offline$time
  offline$time <- offline$time/1000
  class(offline$time) <- c("POSIXt", "POSIXct")

  # round orientations to nearest 45
  offline$angle = roundOrientation(offline$orientation)

  return(offline)
}
```

References

Nolan, D. and Lang, D. T. (2015). *Data Science in R A Case Studies Approach to Computational Reasoning and Problem Solving.* CRC Press.

R Core Team (2013). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria.