# Abstract

In this analysis we are trying to determine how to improve predictability with a location system using similar access points (APs). Initially our dataset of signal strength data is used to map the general structure of a 15m by 36m space at the University of Mannheim, we then use another dataset to predict our location within that space. We are changing out signal strength data from two different APs, the primary AP that was used for testing and an alternative that was dropped as a duplicate, that are situated physically close to one another, to determine which provides better predictive capability. In the end we find that the alternative AP that was dropped from the original dataset provides improved predictive capability. From the signal strength characteristics between the two APs, it is found that the alternative AP provides a different strength profile which could be an explanation of the improved predictive capability.

# Introduction

Locating and tracking items and people has been a fundamental problem for many organizations. Commercial industries wish to track shipments and goods, people want to use navigation systems when travelling in automobiles, and militaries want to be able locate and monitor the movement of soldiers and vehicles on the battlefield. The desire is to perform this tracking in as near real time as possible. The emergence of wireless technologies has enabled real time positioning in many different environments.

The Global Positioning System (GPS) provides nearly world-wide coverage via satellites. GPS systems are commonly used for outdoor tracking systems such as vehicle navigation, surveying, aviation and many other outdoor tracking uses. Depending on the type of receiver, GPS has an accuracy of 4.9m down to the millimeter level [1] . GPS depends on the ability of the receiver to get a “fix” from at least three different satellites. Getting a “fix” means the receiver is in view of the satellite and receiving information. The more satellites the receiver can “see”, the better the tracking and positioning information is going to be. However, GPS does not work well indoors due to not being able to receive satellite information, resulting in poor accuracy.

Radio Frequency identification (RFID) is a technology also capable of providing location and tracking information. RFID depends on a tag being present on the object being tracked and some type of receiver being able to read the data from the tag. There three types of tags in RFID systems, active, semi-active and passive. Active and semi-active tags are powered by batteries and typically provide information across a greater distance than passive tags. Passive tags rely on the reader to provide the power and have a much more limited range. One approach used in locating robots indoors used an RFID grid of passive tags [2]. The passive tags are embedded in walls, floors or any immovable object. The area where the robot’s location is tracked is then laid down like a grid of tags with unique identifiers. As the robot passes the tags, its location can be derived.

Indoor positioning systems have grown in scope and capabilities with advances in wireless technologies. Indoor positioning systems used by commercial and research institutions allow for tracking of people and objects in an indoor environment such as hospitals, warehouses and factories. The proliferation of wireless local area networks (LANs) means indoor positioning systems can utilize the Wi-Fi signals used in network access points to track and locate people and equipment that are accessing the network [3]. The scale of the wireless LANs provides a much larger area for tracking than may be feasible using the above RFID scheme. Wireless LANs by nature provide an excellent system for location positioning. Wireless LANs have fixed access points whose locations are known. As clients roam within the wireless LAN, they receive signals from different fixed access point.

The data set we are working with is publicly available from the Community Resource for Archiving Wireless Data at Dartmouth (CRAWDAD). It contains two separate sets of data. The first set is referred to as “offline” data is a reference set. It was captured at the University of Manheim using a handheld device on a grid of 166 points, each 1 meter apart on a single floor of a build. [3]. The second data set is referred to as the “online” data set. It contains measurements taken at 60 random locations an orientation and used for testing any model derived from the offline data set.

Table 1 below shows the fields captured during each measurement.

Table - Fields in raw data set

|  |  |
| --- | --- |
| Field Name | Field Contents |
| t | Timestamp of measurement |
| id | MAC address of scan device |
| Pos | (X,Y,Z) coordinates |
| Degree | Orientation of the device in degrees |
| MAC | MAC address of responding device,  Signal strength in dBm  Channel frequency  Device mode – 1=ad-hoc, 3=AP |

Once the data has been sorted and cleaned, fields present in Table 2 are the fields used for the analysis.

Table - Fields in clean data set

|  |  |
| --- | --- |
| Field Name | Contents |
| Time | Time stamp of measurement |
| PosX | X coordinate of measurement |
| PosY | Y coordinate of measurement |
| PosZ | Z coordinate of measurement |
| Orientation | Orientation of device in degrees |
| MAC | MAC address of responding device |
| Channel | Wi-Fi channel during measurement |
| Type | Type of device, 1=ad-hoc, 3=AP |
| Scanmac | MAC address of the measuring device |

# Background

In this paper we are analyzing an indoor positioning system which uses a Wi-Fi network infrastructure to determine its position relative to the access points (AP) in the network. Signal strength from each access point is measured using the Receive Signal Strength(RSS). In our study, the measurements are provided as negative numbers with a smaller number corresponding to a weaker value. We can assume that these measurements are in dBm. dBm is a measurement of power for radio waves. Decibels is unit for comparing the ratio between two power levels where . When the denominator of the equation, *P2*  is 1 milliwatt, we refer to this measurement as dBm [4]. In this sense dBm is measuring the power of the received signal to 1 milliwatt on a logarithmic scale. Other standard types of measurements using decibels is dBV, where the reference (*P2*) is to 1 volt.

Measuring signal strength in dBm is different than using the often cited RSSI, Received Signal Strength Indicator. RSSI is a relative measurement by the radio manufacturer of signal strength. One manufacture may rate signal strength from 0-100 where another may use 0-255. This makes RSSI a poor choice to determine signal strength, and the absolute power measurement of dBm used in this study the proper measurement.

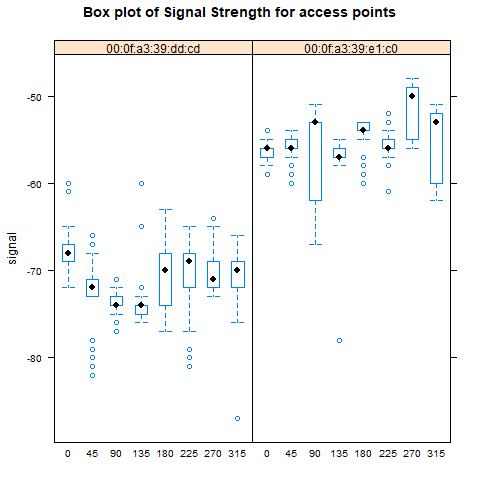
The level of received signal strength is dependent on several characteristics of the receiving device. The design of the radio itself is on important factor. The design of the antenna is perhaps the most important factor in good radio design. The shape, placement and material the antenna is made of all go into designing a good antenna. Antenna design is a complex subject and we will not attempt to get into any of the specifics other than to mention its importance in signal strength. Placement of the antenna in the receiving device, be it a phone, tablet or other portable computing device, is also critical. Complex computing devices have multiple sources of electronic “noise” in them which can interfere with the receive strength of an antenna. CPU’s, high speed busses, touch sensors, and rotating media are examples of components that contribute to a noisy environment. Isolating these noise sources from the antenna can be achieved by physical placement in the system or, if moving a component is not possible, placing a Faraday Cage over a noisy component to isolate the device.

# Method

We will be analyzing which of the two access points should be used in the dataset, either MAC address 00:0F:A3:39:E1:C0 or 00:0F:A3:39:DD:CD. For simplicity we will refer to 00:0F:A3:39:E1:C0 as primary\_AP and 00:0F:A3:39:DD:CD as alternate\_AP. In the original signal analysis address the alternate\_AP was dropped as a duplicate to the primary\_AP, we want to determine whether this was the correct choice. We want to keep the assumption of only using the 6 access points, so we will be strictly doing analysis of how the alternate\_AP acts as a replacement for the primary\_AP. Another thing to keep in mind is our use of signal strength and infrastructure of access points is for the purposes of acting as a positioning system. There are other characteristics that would normally be considered for Wi-Fi networks such as channel overlap and signal loss for data transmission that are considered to be beyond the scope of this paper.

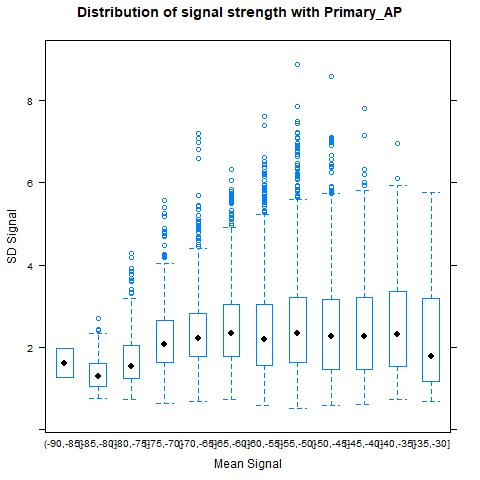
Initially we want to compare the strength of the signal distribution with primary\_AP included then we will run the same calculation with the alternate\_AP. We first plot the distribution of the signal strength for the two APs and see a significant drop of about 20 dBm between the primary\_AP and the alternate\_AP. This behavior can be seen in Figure 1, where there is a drop in the signal strength with about another 5 dBm drop based on the orientation of the observation.

Figure 1:Box plots for signal strength



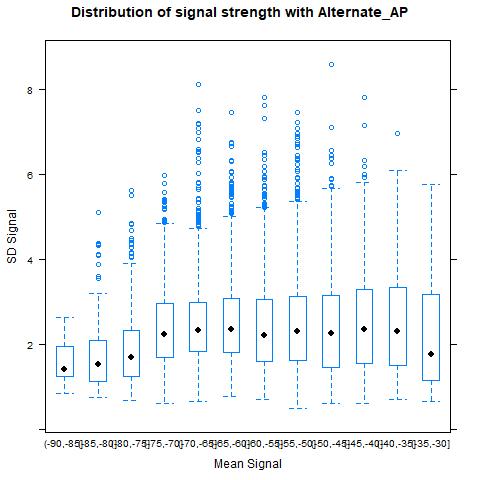
In Figure 2 we have the distribution of the signal strength for the total dataset, including primary\_AP. Toward the righthand of the x-axis we have higher strength signals that we are able to get from the AP. We find that there is a lot more variance among the higher strength signals.

Figure 2:Signal Distribution of dataset including Primary\_AP



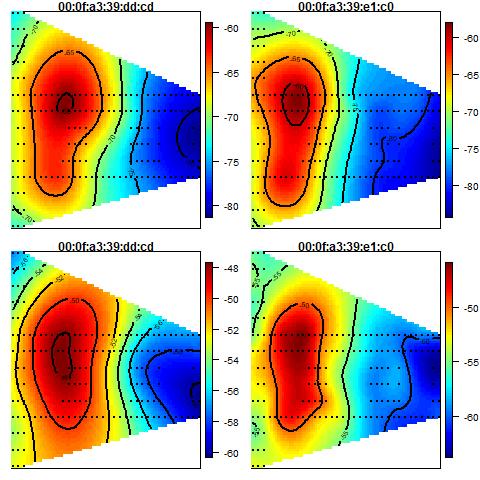
Comparing these initial results, we see in Figure 3 that there is greater variability with the lower strength signals than before. Overall, with the inclusion of the alternate\_AP we find more uniform variability across the entire range of strengths. Normally we would want to have stronger signals for Wi-Fi access points, when talking about data transfer, but since we are concerned with other data in relation to the APs, we are not certain how this will impact the accuracy of our predictions.

Figure 3:Signal Distribution of dataset including alternate\_AP



Here we are comparing the strength of the two signals through the space that the samples were taken. Figure 4 shows the median strength in dBm, with dark red being areas with strongest signal, deep blue with the weakest and the other shades being a gradient between the two extremes. The center of the dark red areas is most likely where the APs are deployed so it is easy to see why one of the APs was removed as a duplicate. The same AP is represented twice to show the difference in signal strength for different orientation one at 0O and one at 135O. Primary\_AP seems to have weaker signal strength at the extremes of the measurable area, but we will need to measure the Sum of Square errors to see what AP helps with the model.

Figure 4:Median Signal Strength distribution for the two APs



To measure the effectiveness of the different APs on the k-nearest neighbor model, we are going to graph the values for the sum of squared errors for all of the values of k from 0 to 20. From there we will take the best value of k for each dataset and get the sum of squared errors for our final comparison.

In Figure 5 we see the optimal number for k is about 7 with a sum of squared errors of 1045. Where in Figure 6 we see there is an optimal value for k of about 4 with a sum of squared errors of 960. Overall it appears that including the alternate\_AP instead of the primary\_AP provides better means of prediction in k-nearest neighbors calculations.

Figure 5:Plot of SSE for primary\_AP

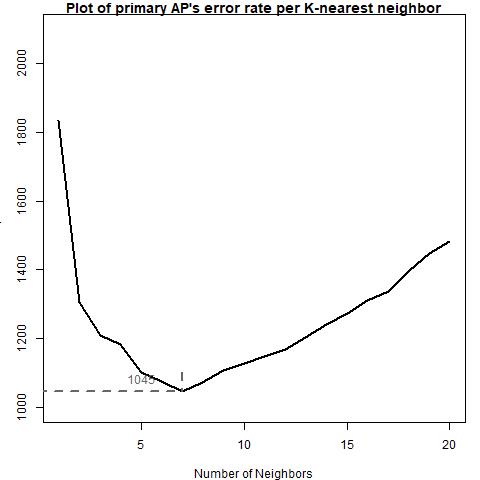
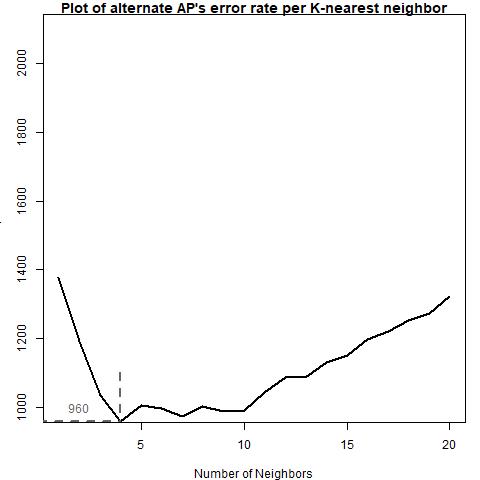


Figure 6:Plot of SSE for alternate\_AP



# Results

In our analysis of the primary\_AP and alternate\_AP we look at the influence of the alternate\_AP on the signal characteristics on the sample of entire dataset. We compared the characteristics of the signal strength with the APs alone and how they change the distribution of the variance for the entire AP infrastructure.

In the end it was found that the alternate\_AP is more effective in providing support for prediction with k-nearest neighbors. It is our hypothesis that improved predictive accuracy for the alternate\_AP is due to the more uniform distribution of variance across all signal strengths for the AP infrastructure. The alternate\_AP added a new signal strength profile compared to what a lot of the other existing APs provided across all different orientations.

# Future Work

With the existing dataset additional work can be done to produce an improved heatmap that lays out all of the access points in a single plot. The hope is to shows the relative hot spots for each AP and how other APs compensate for weaknesses in the signal coverage. At the beginning of our analysis we stated we were only going to be considering use of 6 APs for this exercise and there are a couple of reasons to keep the number of APs to a minimum, reduced IT infrastructure, reduced IT administration, reduced signal overlap, and reduced possibility of signal bleed. Depending on the environment some or all of these concerns could impact the decision of how many APs are installed for such a location system. With a heatmap showing total signal coverage we would have improved insight into what the optimal number of APs is along with their placement.

The network presented in this data is a site survey which uses fixed access points with set indoor configuration of walls and doors. If anything were to happen in the environment changing the signal strength at all the fixed locations, the calibration portion of the creating the offline data set needs to be performed all over again. Any changes in internal structure of the room such as temporary walls, cubicle movement or even changes in the physical location of access points would cause errors in locating objects or people. This problem is known as *concept drift* [5]. Performing this task is time consuming and potentially expensive. An adaptive system that can tolerate a certain level of change in the environment would be worthwhile exercise.

# References

|  |  |
| --- | --- |
| [1] | National Oceanic and Atmospheric Administration, "https://www.gps.gov/systems/gps/performance/accuracy/#how-accurate," Official US government information about GPS, 5 December 2017. [Online]. Available: https://www.gps.gov/systems/gps/performance/accuracy/#how-accurate. [Accessed 6 October 2018]. |
| [2] | J. A. G. M. D. L. V. M. P. Ricardo Tesoriero, "Tracking Autonomous Entities using RFID Technology," *IEEE Transactions on Consumer Electroncis,* vol. 55, no. 2, pp. 650-655, 2009. |
| [3] | D. N. a. D. T. Lang, "Chapter 1 - Predicting Location via Indoor Positioning Systems - Introduction," in *Data Science in R - A Case Studies Approach to Computational Reasoning and Problem Solving*, Boca Rotan, CRC Press, 2015, pp. 3-4. |
| [4] | In-tech Magazine, "db vs dBm," Internationl Society of Automation, 1 November 2002. [Online]. Available: https://www.isa.org/standards-publications/isa-publications/intech-magazine/2002/november/db-vs-dbm/. [Accessed 6 October 2018]. |
| [5] | T.-C. W. S. S. Mustafa Abdat, "Survey on Indoor Wireless Positioning Techniques: Towards Adaptive Systems," in *International Conference on Distributed Framework for Multimedia Applications (DFmA)*, Penang, 2010. |

# Appendix

R Code

library(lattice)

library(fields)

options(digits = 2)

setwd('c:/Users/Steven/Dropbox/School/MSDS 7333 Quantifying the World/Session 6/')

#Function to round orientation

roundOrientation = function(angles) {

refs = seq(0, by = 45, length = 9)

q = sapply(angles, function(o) which.min(abs(o - refs)))

c(refs[1:8], 0)[q]

}

#Function to process each line of the input file

processLine = function(x)

{

tokens = strsplit(x, "[;=,]")[[1]]

if (length(tokens) == 10)

return(NULL)

tmp = matrix(tokens[ - (1:10) ], , 4, byrow = TRUE)

cbind(matrix(tokens[c(2, 4, 6:8, 10)], nrow(tmp), 6,

byrow = TRUE), tmp)

}

#function to read in data

readData = function(filename = 'Data/offline.final.trace.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)

}

#Load offline data into variable

offline = readData()

#Variable that is all 7 APs

subMacs = names(sort(table(offline$mac), decreasing = TRUE))[1:7]

#New column with X-Y string

offline$posXY = paste(offline$posX, offline$posY, sep = "-")

#Variable with data grouped by X-Y position, angle, and MAC

byLocAngleAP = with(offline,

by(offline, list(posXY, angle, mac),

function(x) x))

#Add additional information to X-Y data including mean, median, sd for signal

signalSummary =

lapply(byLocAngleAP,

function(oneLoc) {

ans = oneLoc[1, ]

ans$medSignal = median(oneLoc$signal)

ans$avgSignal = mean(oneLoc$signal)

ans$num = length(oneLoc$signal)

ans$sdSignal = sd(oneLoc$signal)

ans$iqrSignal = IQR(oneLoc$signal)

ans

})

#transform signal summary into dataframe

offlineSummary = do.call("rbind", signalSummary)

#Set dataframes for alternate and primary APs

primary\_AP = "00:0f:a3:39:e1:c0"

alternate\_AP = "00:0f:a3:39:dd:cd"

primary\_AP\_set = offlineSummary[ offlineSummary$mac != alternate\_AP, ]

alternate\_AP\_set = offlineSummary[ offlineSummary$mac != primary\_AP, ]

#Save jpeg of signal strength boxplot for primary

jpeg(file = "Geo\_BoxplotSignalSDByAvg\_primary.jpeg")

oldPar = par(mar = c(3.1, 3, 1, 1))

breaks = seq(-90, -30, by = 5)

bwplot(sdSignal ~ cut(avgSignal, breaks = breaks),

data = primary\_AP\_set,

main = 'Distribution of signal strength with Primary\_AP',

subset = (mac != "00:0f:a3:39:dd:cd"),

xlab = "Mean Signal", ylab = "SD Signal")

par(oldPar)

dev.off()

#Save jpeg of signal strength boxplot for alternate dataset

jpeg(file = "Geo\_BoxplotSignalSDByAvg\_alternate.jpeg")

oldPar = par(mar = c(3.1, 3, 1, 1))

breaks = seq(-90, -30, by = 5)

bwplot(sdSignal ~ cut(avgSignal, breaks = breaks),

data = alternate\_AP\_set,

main = 'Distribution of signal strength with Alternate\_AP',

subset = (mac != "00:0f:a3:39:e1:c0"),

xlab = "Mean Signal", ylab = "SD Signal")

par(oldPar)

dev.off()

#boxplot comparing the signal strength by angles

jpeg(file = "Geo\_BoxplotSignalByMacAngle.jpeg")

oldPar = par(mar = c(3.1, 3, 1, 1))

library(lattice)

bwplot(signal ~ factor(angle) | mac, data = offline,

main = 'Box plot of Signal Strength for access points',

subset = posX == 2 & posY == 12

& (mac == "00:0f:a3:39:dd:cd" | mac == "00:0f:a3:39:e1:c0"),

layout = c(2,1))

par(oldPar)

dev.off()

#summary(offline$signal)

#boxplot comparing the signal strength by angles

jpeg(file = "Geo\_DensitySignalByMacAngle.jpeg")

oldPar = par(mar = c(3.1, 3, 1, 1))

densityplot( ~ signal | mac + factor(angle), data = offline,

main = 'Density plot of Signal Strength for access points',

subset = posX == 24 & posY == 4 &

(mac == "00:0f:a3:39:dd:cd" | mac == "00:0f:a3:39:e1:c0"),

bw = 0.5, plot.points = FALSE)

par(oldPar)

dev.off()

surfaceSS = function(data, mac, angle = 45, main\_title ) {

require(fields)

oneAPAngle = data[ data$mac == mac & data$angle == angle, ]

smoothSS = Tps(oneAPAngle[, c("posX","posY")],

oneAPAngle$avgSignal)

vizSmooth = predictSurface(smoothSS)

plot.surface(vizSmooth, type = "C", main = main\_title,

xlab = "", ylab = "", xaxt = "n", yaxt = "n")

points(oneAPAngle$posX, oneAPAngle$posY, pch=19, cex = 0.5)

}

#boxplot comparing the signal strength by angles

jpeg(file = "GET\_Signal\_Strength.jpeg")

parCur = par(mfrow = c(2,2), mar = rep(1, 4))

mapply(surfaceSS, mac = subMacs[ rep(c(2, 1), each = 2) ],

angle = rep(c(0, 135), 2), main\_title = subMacs[ rep(c(2, 1), 2) ],

data = list(data = offlineSummary))

par(parCur)

dev.off()

#offlineSummary = subset(offlineSummary, mac != subMacs[2])

#Set location for access points

AP = matrix( c( 7.5, 6.3, 7, 6, 2.5, -.8, 12.8, -2.8,

1, 14, 33.5, 9.3, 33.5, 2.8),

ncol = 2, byrow = TRUE,

dimnames = list(subMacs, c("x", "y") ))

diffs = offlineSummary[ , c("posX", "posY")] -

AP[ offlineSummary$mac, ]

offlineSummary$dist = sqrt(diffs[ , 1]^2 + diffs[ , 2]^2)

#Plot of signal strength from access point

jpeg(file="Geo\_ScatterSignalDist.jpeg",height = 1000, width = 500)

oldPar = par(mar = c(3.1, 3.1, 1, 1))

library(lattice)

xyplot(signal ~ dist | factor(mac) + factor(angle),

data = offlineSummary,

subset = (mac == "00:0f:a3:39:dd:cd" | mac == "00:0f:a3:39:e1:c0"),

main='Signal Strength for distance from access point',

pch = 19, cex = 0.3,

xlab ="distance")

par(oldPar)

dev.off()

#Load online data

macs = unique(offlineSummary$mac)

online = readData("Data/online.final.trace.txt", subMacs = macs)

online$posXY = paste(online$posX, online$posY, sep = "-")

length(unique(online$posXY))

tabonlineXYA = table(online$posXY, online$angle)

tabonlineXYA[1:6, ]

keepVars = c("posXY", "posX","posY", "orientation", "angle")

byLoc = with(online,

by(online, list(posXY),

function(x) {

ans = x[1, keepVars]

avgSS = tapply(x$signal, x$mac, mean)

y = matrix(avgSS, nrow = 1, ncol = 7,

dimnames = list(ans$posXY, names(avgSS)))

cbind(ans, y)

}))

onlineSummary = do.call("rbind", byLoc)

# dim(onlineSummary)

# names(onlineSummary)

reshapeSS = function(data, varSignal = "signal",

keepVars = c("posXY", "posX","posY")) {

byLocation =

with(data, by(data, list(posXY),

function(x) {

ans = x[1, keepVars]

avgSS = tapply(x[ , varSignal ], x$mac, mean)

y = matrix(avgSS, nrow = 1, ncol = 6,

dimnames = list(ans$posXY,

names(avgSS)))

cbind(ans, y)

}))

newDataSS = do.call("rbind", byLocation)

return(newDataSS)

}

#

selectTrain = function(angleNewObs, signals = NULL, m = 1){

# m is the number of angles to keep between 1 and 5

refs = seq(0, by = 45, length = 8)

nearestAngle = roundOrientation(angleNewObs)

if (m %% 2 == 1)

angles = seq(-45 \* (m - 1) /2, 45 \* (m - 1) /2, length = m)

else {

m = m + 1

angles = seq(-45 \* (m - 1) /2, 45 \* (m - 1) /2, length = m)

if (sign(angleNewObs - nearestAngle) > -1)

angles = angles[ -1 ]

else

angles = angles[ -m ]

}

angles = angles + nearestAngle

angles[angles < 0] = angles[ angles < 0 ] + 360

angles[angles > 360] = angles[ angles > 360 ] - 360

angles = sort(angles)

offlineSubset = signals[ signals$angle %in% angles, ]

reshapeSS(offlineSubset, varSignal = "avgSignal")

}

findNN = function(newSignal, trainSubset) {

diffs = apply(trainSubset[ , 4:9], 1,

function(x) x - newSignal)

dists = apply(diffs, 2, function(x) sqrt(sum(x^2)) )

closest = order(dists)

return(trainSubset[closest, 1:3 ])

}

predXY = function(newSignals, newAngles, trainData,

numAngles = 1, k = 3){

closeXY = list(length = nrow(newSignals))

for (i in 1:nrow(newSignals)) {

trainSS = selectTrain(newAngles[i], trainData, m = numAngles)

closeXY[[i]] =

findNN(newSignal = as.numeric(newSignals[i, ]), trainSS)

}

estXY = lapply(closeXY,

function(x) sapply(x[ , 2:3],

function(x) mean(x[1:k])))

estXY = do.call("rbind", estXY)

return(estXY)

}

floorErrorMap = function(estXY, actualXY, trainPoints = NULL, AP = NULL,plottitle){

plot(0, 0, xlim = c(0, 35), ylim = c(-3, 15), type = "n",main=plottitle,

xlab = "", ylab = "", axes = FALSE)

box()

if ( !is.null(AP) ) points(AP, pch = 15)

if ( !is.null(trainPoints) )

points(trainPoints, pch = 19, col="grey", cex = 0.6)

points(x = actualXY[, 1], y = actualXY[, 2],

pch = 19, cex = 0.8 )

points(x = estXY[, 1], y = estXY[, 2],

pch = 8, cex = 0.8 )

segments(x0 = estXY[, 1], y0 = estXY[, 2],

x1 = actualXY[, 1], y1 = actualXY[ , 2],

lwd = 2, col = "red")

}

calcError =

function(estXY, actualXY)

sum( rowSums( (estXY - actualXY)^2) )

estXYk3\_primary = predXY(newSignals = onlineSummary[ , (macs[!(macs %in% alternate\_AP)])],

newAngles = onlineSummary[ , 4],

primary\_AP\_set, numAngles = 3, k = 3)

estXYk1\_primary = predXY(newSignals = onlineSummary[ , (macs[!(macs %in% alternate\_AP)])],

newAngles = onlineSummary[ , 4],

primary\_AP\_set, numAngles = 3, k = 1)

trainPoints\_primary = offlineSummary[ offlineSummary$angle == 0 &

offlineSummary$mac == primary\_AP ,

c("posX", "posY")]

jpeg(file="GEO\_FloorPlanK3Errors\_primary.jpeg")

oldPar = par(mar = c(1, 1, 1, 1))

floorErrorMap(estXYk3\_primary, onlineSummary[ , c("posX","posY")],

trainPoints = trainPoints\_primary, AP = AP, plottitle='3-neighbors error for primary AP')

par(oldPar)

dev.off()

jpeg(file="GEO\_FloorPlanK1Errors\_primary.jpeg")

oldPar = par(mar = c(1, 1, 1, 1))

floorErrorMap(estXYk1\_primary, onlineSummary[ , c("posX","posY")],

trainPoints = trainPoints\_primary, AP = AP, plottitle='1-neighbor error for primary AP')

par(oldPar)

dev.off()

actualXY = onlineSummary[ , c("posX", "posY")]

sapply(list(estXYk1\_primary, estXYk3\_primary), calcError, actualXY)

#Block of code to compute alternate k-neighbors values

###############

estXYk3\_alternate = predXY(newSignals = onlineSummary[ , (macs[!(macs %in% primary\_AP)])],

newAngles = onlineSummary[ , 4],

alternate\_AP\_set, numAngles = 3, k = 3)

estXYk1\_alternate = predXY(newSignals = onlineSummary[ , (macs[!(macs %in% primary\_AP)])],

newAngles = onlineSummary[ , 4],

alternate\_AP\_set, numAngles = 3, k = 1)

trainPoints\_alternate = offlineSummary[ offlineSummary$angle == 0 &

offlineSummary$mac == primary\_AP ,

c("posX", "posY")]

jpeg(file="GEO\_FloorPlanK3Errors\_alternate.jpeg")

oldPar = par(mar = c(1, 1, 1, 1))

floorErrorMap(estXYk3\_alternate, onlineSummary[ , c("posX","posY")],

trainPoints = trainPoints\_alternate, AP = AP, plottitle='3-neighbors error for alternate AP')

par(oldPar)

dev.off()

jpeg(file="GEO\_FloorPlanK1Errors\_alternate.jpeg")

oldPar = par(mar = c(1, 1, 1, 1))

floorErrorMap(estXYk1\_alternate, onlineSummary[ , c("posX","posY")],

trainPoints = trainPoints\_alternate, AP = AP, plottitle='1-neighbor error for alternate AP')

par(oldPar)

dev.off()

sapply(list(estXYk1\_alternate, estXYk3\_alternate), calcError, actualXY)

###############

v = 11

permuteLocs = sample(unique(offlineSummary$posXY))

permuteLocs = matrix(permuteLocs, ncol = v,

nrow = floor(length(permuteLocs)/v))

onlineFold = subset(offlineSummary, posXY %in% permuteLocs[ , 1])

reshapeSS = function(data, varSignal = "signal",

keepVars = c("posXY", "posX","posY"),

sampleAngle = FALSE,

refs = seq(0, 315, by = 45)) {

byLocation =

with(data, by(data, list(posXY),

function(x) {

if (sampleAngle) {

x = x[x$angle == sample(refs, size = 1), ]}

ans = x[1, keepVars]

avgSS = tapply(x[ , varSignal ], x$mac, mean)

y = matrix(avgSS, nrow = 1, ncol = 6,

dimnames = list(ans$posXY,

names(avgSS)))

cbind(ans, y)

}))

newDataSS = do.call("rbind", byLocation)

return(newDataSS)

}

offline\_primary = offline[ offline$mac != alternate\_AP, ]

keepVars = c("posXY", "posX","posY", "orientation", "angle")

onlineCVSummary = reshapeSS(offline\_primary, keepVars = keepVars,

sampleAngle = TRUE)

onlineFold = subset(onlineCVSummary,

posXY %in% permuteLocs[ , 1])

offlineFold = subset(offlineSummary[offlineSummary$mac!=alternate\_AP,],

posXY %in% permuteLocs[ , -1])

estFold = predXY(newSignals = onlineFold[ , 6:11],

newAngles = onlineFold[ , 4],

offlineFold, numAngles = 3, k = 3)

actualFold = onlineFold[ , c("posX", "posY")]

calcError(estFold, actualFold)

K = 20

err = rep(0, K)

for (j in 1:v) {

onlineFold = subset(onlineCVSummary,

posXY %in% permuteLocs[ , j])

offlineFold = subset(offlineSummary[offlineSummary$mac!=alternate\_AP,],

posXY %in% permuteLocs[ , -j])

actualFold = onlineFold[ , c("posX", "posY")]

for (k in 1:K) {

estFold = predXY(newSignals = onlineFold[ , 6:11],

newAngles = onlineFold[ , 4],

offlineFold, numAngles = 3, k = k)

err[k] = err[k] + calcError(estFold, actualFold)

}

}

jpeg(file = "Geo\_CVChoiceOfK\_primary.jpeg")

oldPar = par(mar = c(4, 3, 1, 1))

plot(y = err, x = (1:K), type = "l", lwd= 2,

main="Plot of primary AP's error rate per K-nearest neighbor" ,

ylim = c(1000, 2100),

xlab = "Number of Neighbors",

ylab = "Sum of Square Errors")

rmseMin = min(err)

kMin = which(err == rmseMin)[1]

segments(x0 = 0, x1 = kMin, y0 = rmseMin, col = gray(0.4),

lty = 2, lwd = 2)

segments(x0 = kMin, x1 = kMin, y0 = 1100, y1 = rmseMin,

col = grey(0.4), lty = 2, lwd = 2)

#mtext(kMin, side = 1, line = 1, at = kMin, col = grey(0.4))

text(x = kMin - 2, y = rmseMin + 40,

label = as.character(round(rmseMin)), col = grey(0.4))

par(oldPar)

dev.off()

#Calculate k-neighbors = 5

estXYk\_primary = predXY(newSignals = onlineSummary[ , (macs[!(macs %in% alternate\_AP)])],

newAngles = onlineSummary[ , 4],

offlineSummary[offlineSummary$mac!=alternate\_AP,], numAngles = 3, k = which.min(err))

calcError(estXYk\_primary, actualXY)

cat('The optimal number of neighbors is ',which.min(err))

offline\_alternate = offline[ offline$mac != primary\_AP, ]

keepVars = c("posXY", "posX","posY", "orientation", "angle")

onlineCVSummary = reshapeSS(offline\_alternate, keepVars = keepVars,

sampleAngle = TRUE)

onlineFold = subset(onlineCVSummary,

posXY %in% permuteLocs[ , 1])

offlineFold = subset(offlineSummary[offlineSummary$mac!=primary\_AP,],

posXY %in% permuteLocs[ , -1])

estFold = predXY(newSignals = onlineFold[ , 6:11],

newAngles = onlineFold[ , 4],

offlineFold, numAngles = 3, k = 3)

actualFold = onlineFold[ , c("posX", "posY")]

calcError(estFold, actualFold)

K = 20

err = rep(0, K)

for (j in 1:v) {

onlineFold = subset(onlineCVSummary,

posXY %in% permuteLocs[ , j])

offlineFold = subset(offlineSummary[offlineSummary$mac!=primary\_AP,],

posXY %in% permuteLocs[ , -j])

actualFold = onlineFold[ , c("posX", "posY")]

for (k in 1:K) {

estFold = predXY(newSignals = onlineFold[ , 6:11],

newAngles = onlineFold[ , 4],

offlineFold, numAngles = 3, k = k)

err[k] = err[k] + calcError(estFold, actualFold)

}

}

jpeg(file = "Geo\_CVChoiceOfK\_alternate.jpeg")

oldPar = par(mar = c(4, 3, 1, 1))

plot(y = err, x = (1:K), type = "l", lwd= 2,

main="Plot of alternate AP's error rate per K-nearest neighbor" ,

ylim = c(1000, 2100),

xlab = "Number of Neighbors",

ylab = "Sum of Square Errors")

rmseMin = min(err)

kMin = which(err == rmseMin)[1]

segments(x0 = 0, x1 = kMin, y0 = rmseMin, col = gray(0.4),

lty = 2, lwd = 2)

segments(x0 = kMin, x1 = kMin, y0 = 1100, y1 = rmseMin,

col = grey(0.4), lty = 2, lwd = 2)

#mtext(kMin, side = 1, line = 1, at = kMin, col = grey(0.4))

text(x = kMin - 2, y = rmseMin + 40,

label = as.character(round(rmseMin)), col = grey(0.4))

par(oldPar)

dev.off()

#Calculate k-neighbors = 5

estXYk\_alternate = predXY(newSignals = onlineSummary[ , (macs[!(macs %in% primary\_AP)])],

newAngles = onlineSummary[ , 4],

offlineSummary[offlineSummary$mac!=primary\_AP,], numAngles = 3, k = which.min(err))

calcError(estXYk\_alternate, actualXY)

cat('The optimal number of neighbors is ',which.min(err))