MATLAB simulation of RSS based channel modelling localization and tracking

A J - component report for

Course - Wireless Mobile Communication
Course code - ECE 3003

Submitted by

SHAMBHAVI AWASTHI (17BEC0084)
VIRAJ CHOKHANY (17BEC0619)

Slot - B1

Submitted to

Proff. BUDHADITYA BHATTACHARYYA. Sir

In partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY (B.TECH)

In

ELECTRONICS AND COMMUNICATION ENGINEERING

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF ELECTRONICS ENGINEERING

NOVEMBER 2020

DECLARATION BY THE CANDIDATE

We hereby declare that the Industrial Internship report entitled **"MATLAB simulation of RSS based channel modelling localization and tracking"** submitted by **Shambhavi Awasthi(17BEC0084) and Viraj Chokhany (17BEC0619)** to Vellore Institute of Technology, Vellore in partial fulfillment of the requirement for the award of the degree of **Bachelor of Technology (B.TECH) in Electronics and Communication Engineering** is a record of bonafide project undertaken by us under the supervision of **Proff. BUDHADITYA BHATTACHARYYA, SENSE , VIT University**. We further declare that the work reported in this report has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Signature of the students

Shambhavi Awasthi
17BEC0084

Viraj Chokhany
17BEC0619

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering

BONAFIDE CERTIFICATE

This is to certify that the Industrial Internship report entitled **"MATLAB simulation of RSS based channel modelling localization and tracking"** submitted by **Shambhavi Awasthi (17BEC0084) and Viraj Chokhany (17BEC0619)** to Vellore Institute of Technology, Vellore in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology(B.TECH) in **Electronics and Communication Engineering** is a record of bonafide Project undertaken by him/her under my supervision. The training fulfills the requirements as per the regulations of this Institute and in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Signature of the Supervisor

SUPERVISOR

Date:

Date:

Internal Examiner(s)

External Examiner(s)

ACKNOWLEDGEMENT

An endeavor over a long period can be successful only with advice and guidance of many well-wishers. We take this opportunity to express my deep sense of gratitude and appreciation for all those who encouraged me to the successful completion of the project.

We would like to thank **VIT University and Professor BUDHADITYA BHATTACHARYYA.** Sir for providing me with a chance to do my project in this organization. We would also like to extend my gratitude to VIT Vellore for providing me with the opportunity and guidance to carry out the project.

We express my deep gratitude towards my parents and peers for providing me with their invaluable moral support without which we would not have been able to complete the project in a timely fashion.

Place : Vellore

Date :

Shambhavi Awasthi
(17BEC0084)

Viraj Chokhany
(17BEC0619)

ABSTRACT:

Wireless communication has evolved significantly, over the past several decades, to meet the ever-growing demand for high data rates over the wireless medium. Systems have been designed for indoor applications and outdoor applications where mobility is a very important aspect of system specification. The propagation model for the wireless channel is characterized by path loss, multipath fading and Doppler spread (or Doppler spectrum). All these characteristics are affected by the physical environment between the transmitter and receiver as well as system dependent parameters such as antenna heights, antenna beam-widths, antenna polarization, and mutual coupling between multiple antennas.This project presents a MATLAB simulation of RSS based channel modelling , localization and tracking. Many factors have to be considered when RSS-based localization applications are designed, starting from selection of the proper propagation model, which has to represent in a relatively accurate way the interaction between the RF signal and the environment. It is demonstrated that the RSS can be used for outdoor localization.

INTRODUCTION AND WORKFLOW:

Determining the location of a target is one of the fundamental functions of sensor networks . Real-time and accurate position information plays a crucial role in a variety of wireless applications, such as logistics, search-and-rescue operations, and location-based services.
Localization in sensor networks is critical for search and rescue. Linear least squares (LLS) estimation is a sub-optimum but low-complexity localization algorithm based on measurements of location-related parameters. Commonly, there are two types of LLS localization algorithms using range measurements; one is based on introducing a dummy variable , and the other is based on the subtraction of the reference measured range. Moreover, their respective weighted LLS (WLLS) algorithms can be adopted to further improve the localization accuracy.
In this project we are using the following setup:

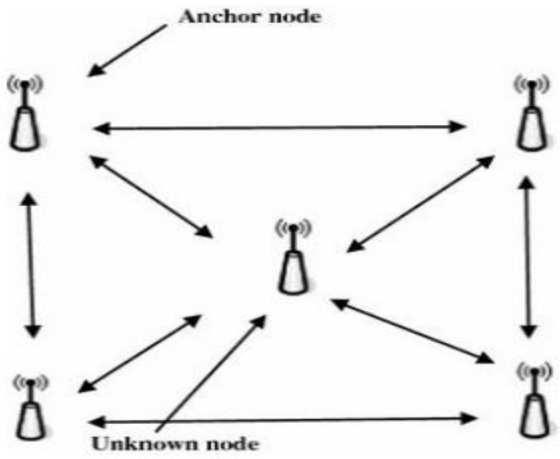

Fig 1. Setup for the project

This contains 4 Anchor nodes with known coordinates which will be referred as Base stations and one unknown node ie. Mobile node which we will be tracking and locating. In WSN, only a limited amount of nodes in the network can know their locations by a variety of positioning mechanisms. These nodes are commonly called as anchor nodes. Anchor nodes are positioned by employing RSS measurements in this proposed method.A Mobile Node thus is the basic Node object with added functionalities of a wireless and mobile node like ability to move within a given topology, ability to receive and transmit signals to and from a wireless channel etc. Range information is commonly adopted in the first step of localization, which can be measured using or received-signal-strength (RSS) estimates. Further estimation functions like LLS and WLS are used to find localization error between actual location and estimated coordinates. As a result of this project we present two outputs:

1. CDF vs Localization error graph for various mobility patterns like random, straight line and square.
2. CDF vs Localization error graph for various estimation functions like LLS and WLS.

ALGORITHM:

1. Received Signal Strength:

The received signal strength (RSS) is the strength of a received signal measured at the receiver's antenna. RSS is determined by the transmission power, the distance between the transmitter and the receiver, and the radio environment. The received signal strength is a measurement that is hard to forge arbitrarily and it is highly correlated to the transmitter's location. RSS indicator indicates the power of the received signal and is a function of distance between the transmitter and receiving device that gets impacted due to various in-path interferences.

2. Linear Least Square:

Linear least squares (LLS) is the least squares approximation of linear functions to data.

3. Weighted Least Square:

Weighted least squares (WLS), also known as weighted linear regression, is a generalization of ordinary least squares and linear regression in which the errors covariance matrix is allowed to be different from an identity matrix. WLS is also a specialization of generalized least squares in which the above matrix is diagonal.

A general explaination of the algorithm used is given below:

we assume that the agent is connected to different anchors, and these anchors are able to measure the range between the agent and anchors via RSS parameters. Let N be the total number of all anchors in the localization network. In the first localization step, the range measurement between the agent and the ith ($i = 1,2,\ldots,N$) anchor is denoted as di . Let $\mathrm{p} = [x\ y]^T$ be the unknown two-dimensional (2-D) coordinate of the agent, which is to be estimated, and let $\mathrm{p} = [x_i\ y_i]^T$ be the known 2-D coordinate of the ith anchor. The error-free range between the agent and the ith anchor is calculated as:

$$d_i = \mathbf{p} - \mathbf{p}_i = \sqrt{(x - x_i)^2 + (y - y_i)^2}$$

The range measurement is modeled as:

$$\hat{d}_i = d_i + n_i$$

where n_i is the ranging error in dˆi, which results RSS estimation disturbance. It is assumed that $\{n_i\}$ are independent Gaussian processes with zero mean and variances .

Obtaining all the range measurements in leads to the following inconsistent equations:

$$(x - x_i)^2 + (y - y_i)^2 = \hat{d}_i^2,\ i = 1,\ 2, \cdots,\ N$$

For RSS-based range measurements, the log-normal model is commonly used to calculate range from path loss, which is given by

$$\overline{P}_r(d) = P_0 - 10\gamma log_{10}\left(\frac{d}{d_0}\right) + V$$

where Pr(d) denotes the average received power in decibels at a distance d, P_0 denotes the received power in decibels at reference distance d_0, γ is the path-loss exponent with typical value between 2 and 6, and V is commonly modeled as a zero-mean Gaussian random variable with variance σ^2, which represents the large-scale fading variations (i.e., shadowing).

$$\sigma^2_{i,\text{CRLB, RSS}} = \left(\frac{(ln10)\sigma_{\text{sh}} d_i}{10\gamma}\right)^2.$$

In the second localization step, we adopt different LLS localization algorithms to convert the nonlinear range measurements in into linear models in p and give a close form location estimate pˆLLS.

We choose the rth anchor as the reference anchor for example without loss of generality. By subtracting the nonlinear expression of the rth anchor in from the rest of the expressions, we have:

$$2(x_i - x_r)x + 2(y_i - y_r)y = d_r^2 - d_i^2 - k_r + k_i$$

where we define $k_i \triangleq x_i^2 + y_i^2$, $i = 1,2,\ldots,N$.

Rewriting in matrix form :

$$\mathbf{A_{II}} = 2\begin{bmatrix} x_1 - x_r & y_1 - y_r \\ \vdots & \vdots \\ x_{r-1} - x_r & y_{r-1} - y_r \\ x_{r+1} - x_r & y_{r+1} - y_r \\ \vdots & \vdots \\ x_N - x_r & y_N - y_r \end{bmatrix}, \mathbf{b_{II}} = \begin{bmatrix} d_r^2 - d_1^2 - k_r + k_1 \\ \vdots \\ d_r^2 - d_{r-1}^2 - k_r + k_{r-1} \\ d_r^2 - d_{r+1}^2 - k_r + k_{r+1} \\ \vdots \\ d_r^2 - d_N^2 - k_r + k_N \end{bmatrix}$$

Thus location estimate is given by:

$$\widehat{\mathbf{p}}_{\mathbf{LLS-II}} = \left(\mathbf{A}_{\mathbf{II}}^T \mathbf{A}_{\mathbf{II}}\right)^{-1} \mathbf{A}_{\mathbf{II}}^T \mathbf{b}_{\mathbf{II}}$$

Third part of the algorithm is WLS estimation function:

$$\widehat{\mathbf{p}}_{\mathbf{WLLS-II}} = \left(\mathbf{A}_{\mathbf{II}}^T \mathbf{C}_{\mathbf{II}}^{-1} \mathbf{A}_{\mathbf{II}}\right)^{-1} \mathbf{A}_{\mathbf{II}}^T \mathbf{C}_{\mathbf{II}}^{-1} \mathbf{b}_{\mathbf{II}},$$

where $\mathbf{C}_{\mathbf{II}}$ is the weighted matrix and its elements can be derived as [13]

$$[\mathbf{C_{II}}]_{ij} = 4\hat{d}_r^2 \sigma_r^2 + 3\sigma_r^4 - \sigma_r^2\left(\sigma_i^2 + \sigma_j^2\right) + \sigma_i^2 \sigma_j^2 + I(i,j)\left(4\hat{d}_i^2 \sigma_i^2 + 2\sigma_i^4\right)$$

with $i,j = 1,2,\ldots,N, i \neq r, j \neq r$, and $I(i,j)$ is an indicator function which is 1 for $i = j$, and is 0 otherwise.

CODE:

1. Function to compute the euclidean distance from source to four anchor nodes:

```
% 17BEC0084 Shambhavi Awasthi
% 17BEC0619 Viraj Chokhany
% wmc project B1 slot

% Function to compute Euclidean distance from the source to 4 anchor nodes
% a_x: x coordinates of the anchors
% a_y: y coordinates of the anchors
% s_x: x coordinate of the source
% s_y: y coordinate of the source
% returns distance vector for 4 distances

function[dist] = compute_dist(a_x, a_y, s_x, s_y)
for i = 1:4;
    dist(i) = eucl_dist(s_x, s_y, a_x(i), a_y(i));
end;
```

2. Function to compute the received signal strength from the path loss model:

```
% 17BEC0084 Shambhavi Awasthi
% 17BEC0619 Viraj Chokhany
% wmc project B1 slot

% Function to compute the received signal strength from the path loss model
% The received signal strength (RSS) is the strength of a received signal
measured at the receiver's antenna.
% for 4 signals transmitted from the source node to the anchor nodes
% for a radio signal travelling from a source node to four anchor nodes
% dist: vector containing distances from source to 4 anchors
% noise: vector of normally distributed random variables that models the
shadowing phenomenon
% noise is computed from noise function
% distance is computed from euclidean distance function
% alpha: path loss exponent
% P_0: reference power at reference distance d_0
% d_0: reference distance

function[rss]=compute_rss(dist, noise, alpha, P_0, d_0)
for i=1:4;
    rss(i) = P_0 - (10*alpha*log10(dist(i)/d_0)) + noise(i);
end;
```

3. Function to compute distances from the source to the anchor nodes using the lognomal path loss model:

```
% 17BEC0084 Shambhavi Awasthi
% 17BEC0619 Viraj Chokhany
% wmc project B1 slot

% function to estimate distances from the source to the anchor nodes using
the lognomal path loss model
% rss: vector of RSS from 4 anchor nodes
% noise: noise values for 4 anchors
% alpha: path loss exponent
% P_0: reference power at reference distance (dB)
% d_0: reference distance (m)
% returns the estimated distance

function[estimated_dist]=estimate_dist(rss, noise, alpha, P_0, d_0)
for i = 1:4;
    estimated_dist(i) = 10^((P_0+noise(i)-rss(i))/(10*alpha))*d_0;
end;
```

4. Function to compute distance between 2 points in 2D plane:

```
% 17BEC0084 Shambhavi Awasthi
% 17BEC0619 Viraj Chokhany
% wmc project B1 slot

% This function computes euclidean distance between two points
% x1, y1 coordinates of the first point
% x2, y2 coordinates of the second point
% return distance between (x1,y1) and (x2,y2)
```

```
function[dist] = eucl_dist(x1, y1, x2, y2)
dist = sqrt((x1-x2)^2+(y1-y2)^2); % distance formula 2D
```

5. Function to generate noise vector:

```
% 17BEC0084 Shambhavi Awasthi
% 17BEC0619 Viraj Chokhany
% wmc project B1 slot

%Generates white noise of secified length using 'randn' function,
% which produces random numbers that follow a Gaussian distribution
%A white noise signal (process) is constituted by a set of independent
and identically distributed random variables.
% length: the length of the noise vector
% mu: mean
% sigma: deviation

function[noise]=set_noise(mu, sigma, length)
noise = sigma*randn(1,length)+mu;
```

6. Function to generate random mobility pattern for source nodes:

```
% 17BEC0084 Shambhavi Awasthi
% 17BEC0619 Viraj Chokhany
% wmc project B1 slot

% function to generate random mobility patterns
% it returns two vectors  Xs and Ys for the source nodes
% nb_locations = 100

function [ s_x, s_y, s ] = random( nb_locations )
    s = 1;
    % generating vector of x coordinates of the source nodes Xs
    s_x = randi([1,999], 1, nb_locations); % generates vector for
nb_locations randomly from 1-999
    % generating vector of y coordinates of the source nodes Ys
    s_y = randi([1,999], 1, nb_locations); % generates vector for
nb_locations randomly from 1-999
end
```

7. Function to generate square mobility pattern for source nodes:

```
% 17BEC0084 Shambhavi Awasthi
% 17BEC0619 Viraj Chokhany
% wmc project B1 slot

% Function to generate square mobility pattern
% It generates points for 1000 source nodes

function [ x, y, s ] = square( )
    x = zeros(1, 200);
    y = zeros(1, 200);
    for i = 1:200;
        if i <= 50;
            x(i) = i * 20;
            y(i) = 20;
        elseif i <= 100;
```

```
            x(i) = 1000;
            y(i) = (i -100) * 20;
        elseif i <= 150;
            x(i) = 1000 - (i - 150) * 20;
            y(i) = 1000;
        elseif i <= 200
            x(i) = 20;
            y(i) = 1000 - (i - 200) * 20;
        end;
    end;
    s = 20;
end
```

8. Function to generate straight line mobility pattern for source nodes:

```
% 17BEC0084 Shambhavi Awasthi
% 17BEC0619 Viraj Chokhany
% wmc project B1 slot

% linear mobility pattern generating function for source nodes
% nb_locations = 200

function [ x, y, s ] = straight_line ( nb_locations )
    x = zeros(1, nb_locations);
    y = zeros(1, nb_locations);
    p = 1000/nb_locations; % 1000 source nodes
    for i = 1:nb_locations;
        x(i) = i * (p); % x-coordinate
        y(i) = i * (p); % y-coordinate
    end;
    s = sqrt(1000 / nb_locations);
end
```

9. Function to compute erro between the actual and estimated coordinates:

```
% 17BEC0084 Shambhavi Awasthi
% 17BEC0619 Viraj Chokhany
% wmc project B1 slot

% function to compute the distance between the actual and estimated
coordinates
% this determines the error of distance
% actual_coord: actual coordinates of source node A(n,2)
% estimated_coord: estimated coord of source node B(n,2)

function[error]=compute_error(actual_coord,estimated_coord)
for i=1:size(actual_coord, 1);
% error is basically the euclidean distance between actual and estimated
coordinates
% euclidean distance is calculated used already defined eucliean function
    error(i) = eucl_dist(actual_coord(i, 1), actual_coord(i,2),
estimated_coord(i,1), estimated_coord(i,2));
end;
```

10. Function to compute the estimated position of a mobile device using LLS:

```
% 17BEC0084 Shambhavi Awasthi
% 17BEC0619 Viraj Chokhany
```

```
% wmc project B1 slot

% Function to compute the estimated position of a mobile device using LLS
% base_stations: coordinates of known base stations, one coordinate per
line
% distances: estimated distance to each base station, one per column
% Returns the estimated x and y coordinates of the mobile device

function x = lls( base_stations, distances )
    n = size(distances, 2); % size of distances for dimension 2
    b = zeros([n-1, 1]); % Preallocate the array
    h = base_stations(2:n, 1:2) * 2; % leaving first coordinate for base
station 1
    ht = transpose(h);
    % Compute vector b
    for i = 2:n;
        b(i-1) = (base_stations(i, 1) ^ 2) + (base_stations(i, 2) ^ 2)-
(distances(i) ^ 2) + (distances(1) ^ 2);
    end
    % Compute the estimated location relative to base_station 1 and
translate in absolute coordinates.
    % ht * h \ ht is equivalent to inv(ht * h) * ht
    x = transpose(((ht * h) \ ht) * b) + base_stations(1, 1:2);
end
```

11. Function to compute the estimated position of a mobile device using WLS:

```
% 17BEC0084 Shambhavi Awasthi
% 17BEC0619 Viraj Chokhany
% wmc project B1 slot

% Function to compute the estimated position of a mobile device using WLS
% base_stations: coordinates of known base stations, one coordinate per
line
% distances: estimated distance to each base station, one per column
% Returns the estimated x and y coordinates of the mobile device

function x = wls(base_stations, distances)
% these steps are similar to LLS
    n = size(distances, 2); % compute alpha
    b = zeros([n-1, 1]); % Preallocate the array
    H = base_stations(2:n, 1:2) * 2;
    % Compute vector b
    for i = 2:n;
        b(i-1) = (base_stations(i, 1) ^ 2) + (base_stations(i, 2) ^ 2)-
(distances(i) ^ 2) + (distances(1) ^ 2);
    end

    % compute WLS
    S = zeros([n-1, n-1]); % allocating (n-1) X (n-1) matrix
    for i = 1:n-1;
        for j = 1:n-1;
            var = distances(1)^4; % Computes the variance for every distance
            if i == j;
                var = distances(1)^4 + distances(i+1)^4; % The variances
for the diagonal of the matrix is different
            end;
            S(i,j) = var;
        end;
```

```
    end;
    x = transpose(((transpose(H) * (S \ H)) \ transpose(H)) * (S \ b));
end
```

12. Test function:

```
% 17BEC0084 Shambhavi Awasthi
% 17BEC0619 Viraj Chokhany
% wmc project B1 slot

% Function to test an estimation function on a specific mobility pattern
% Testing the estimation function passed in parameter as a function handle
with the mobility pattern also specified as a function handle.
% Returns a vector containing the various amount of error in the estimation
of the position

function [ errors ] = test_function( estimation_function,
mobility_pattern,base_stations, mob_pattern_parameters, alpha, mu,
sigma, P_0, d_0 )
    anchors_x = base_stations(:, 1); % x-coordinates for anchors ( known
base stations )
    anchors_y = base_stations(:, 2); % y-coordinates for anchors ( known
base stations )
    [ mobile_x, mobile_y, s ] =
mobility_pattern(mob_pattern_parameters{:}); % from mobility pattern
functions
    nb_positions = size(mobile_x, 2); % assigns number of columns in
mobile_x
    nb_base_stations = size(base_stations, 1); % assigns number of rows
in base stations i.e number of anchor nodes
    estimated_coord = zeros(nb_positions, 2); % assigns number of columns
in nb_positions

    % for each source node compute the distance to the anchors, estimate
rss
    % estimate distance from rss, estimate coordinates using the algorithms

    for i = 1:nb_positions
        %compute distances to the anchors for each source node
        dist = compute_dist(anchors_x, anchors_y, mobile_x(1,i),
mobile_y(1,i));

        %estimate the rss for each source node
        rss = compute_rss(dist, set_noise(mu, sigma, nb_base_stations),
alpha, P_0, d_0);

        %estimate distances to the anchor nodes based on the rss vector
        estimated_dist = estimate_dist(rss, set_noise(mu, sigma,
nb_base_stations), alpha, P_0, d_0);

        %estimate the coordinates using the estimation function
        %i.e LLS and WLS
        estimated_coord(i, :) = estimation_function(base_stations,
estimated_dist);
    end;

    % create matrix of size (n,2) storing actual coordinates for n points
    % horzcat : concatenate vectors horizontally
```

```
    actual_coord = horzcat(mobile_x(:), mobile_y(:));

    %compute error vector
    errors = compute_error(actual_coord, estimated_coord);

end
```

13. Main function:

```
% 17BEC0084 Shambhavi Awasthi
% 17BEC0619 Viraj Chokhany
% wmc project B1 slot

clc
close all

% Defining all the parameters for the functions
% Estimation functions list
estimation_functions = {@lls @wls};

% Mobility patterns
mob_patterns = {@random @straight_line @square};

% Specific parameters relative to each mobility pattern
mob_patterns_parameters = {
    { 100 } % random(nb_locations)
    { 200 } % linear(nb_locations)
    { } % square()
};

% Coordinates of the anchor nodes ie. 4 here in the setup
base_stations = [
    0, 0;
    0, 1000;
    1000, 1000;
    1000, 0
];

% Noise parameters
mu = 0;
sigma = 1;

% Path loss exponent (dB)
alpha = 3;

% Reference power at reference distance (dB)
P_0 = -10;

% Reference distance (m)
d_0 = 1;

% Map each function name to the parameters that we need to pass
% container.Map function maps the value to the keys
% mapping the mobility pattern to its mobility pattern parameters
% return [x,y,s]
mob_patterns_parameters_map = containers.Map(cellfun(@(x) func2str(x),
mob_patterns, 'UniformOutput', false),mob_patterns_parameters);
```

```
% A cell array is a data type with indexed data containers called cells,
where each cell can contain any type of data
errors = cell(size(mob_patterns, 2) * size(estimation_functions, 2));

i = 1;
% finding localization errors
for mob_pattern = mob_patterns;
    for estimation_function = estimation_functions;
        for filter = filters;
            errors{i} = test_function(estimation_function{1},
mob_pattern{1},base_stations,
mob_patterns_parameters_map(func2str(mob_pattern{1})), alpha, mu,
sigma, P_0, d_0);
            i = i + 1;
        end;
    end;
end;

% Graph plotting
% 2 figures containing many subplots
close all
nb_est_func = size(estimation_functions, 2);
nb_mob_patterns = size(mob_patterns, 2);

% Colormap
cc = lines(max(nb_est_func, nb_mob_patterns));

figure('Name', 'Estimation functions - 17BEC0084 17BEC0619')
for i = 1:nb_est_func;
        subplot(nb_est_func ,1,i)
        for k = 1:nb_mob_patterns;
            h = cdfplot(errors{(i -1)+ (k -1) * nb_est_func+j});
            set(h,'LineWidth',1.5, 'DisplayName',
func2str(mob_patterns{k}), 'Color', cc(k, :))
            hold on
        end
        hold off
        xlabel('Localization error')
        ylabel('CDF')
        legend('show')
        axis([0 450 0 1.2])
        title(strcat(func2str(estimation_functions{i}),' estimation
function - 17BEC0084 17BEC619'))
end

figure('Name', 'Mobility patterns - 17BEC0084 17BEC0619')
for i = 1:nb_mob_patterns
    subplot(nb_mob_patterns, 1, i)
    for j = 1:nb_est_func
            h = cdfplot(errors{(i -1) * nb_est_func + (j -1)+ 1});
            set(h,
'LineWidth',1.5,'DisplayName',func2str(estimation_functions{j}),
'Color', cc((j -1)+1, :))
            hold on
    end
    hold off
    xlabel('Localization error')
    ylabel('CDF')
    legend('show')
    axis([0 450 0 1.2])
```

```
    title(strcat(func2str(mob_patterns{i}),' mobility pattern -
17BEC0084 17BEC619'))
end
```

WORKING OF THE CODE:

- Estimation functions are initialised for LLS and WLS in main_loop. Similarly mobility patterns i.e random , straight line and square are also initialised and mapped to respective parameters inputs as follows:
  For random input nb_locations = 200
  For straight line input nb_locations = 100
  For square no input is required.
- Now we input base station coordinates as (known coordinates or anchor nodes) - (0,0) , (0,1000) , (1000,1000) , (1000,0).
- Next we initialise noise parameters for the channel as mean = 0 and deviation = 1 following Gaussian Distribution and for generating white noise.
- Also path loss exponent is 3 dB, reference power at refernce distance of 1m is -10 for the calculation of RSS using lognormal path loss model.
- Using the main_loop.m we now find the localization error for all mobility patterns using both LLS and WLS and the result is stored in errors cells. This localization error is found using test_function.m .
- Test_function takes inputs for estimation functions, mobility patterns, noise parameters stated above and parameters for path loss model to estimate RSS.
- In test_function, we define x coordinates of the 4 anchors and y coordinates of all 4 anchors.
  Anchor_x=[0 0 1000 1000]
  Anchor_y=[0 1000 1000 0]
  We find [x y s] for all mobility patterns and store them column wise as
  [x,y,s for random ; x,y,s for linear ; x,y,s for square]
  Number of mobile node positions = number of x-coordinates
  Number of base station or anchor nodes = 4
  Number of estimated coordinate = number of positions of the mobile node
- Now for all the mobile node positions generated through mobility patterns we find error as follows:
  - Compute distance between 4 anchor nodes and source node using compute_dist.m which takes input for (a_x,a_y) coordinates of anchors and (s_x,s_y) coordinates of source (generated through mobility pattern). This function computes distance using eucl_dist.m thus using 2D distance formula to find distance between both points.
  - Next, we compute RSS for each source node using compute_rss.m which takes above calculated distance, noise array generated from set_noise.m, alpha, reference power and refernce distance for path loss model as input. Compute_rss.m computes the received signal strength using Log normal path loss model formula for all 4 anchor nodes.
  - Now, we find the estimated distance. Here estimate_dist.m is used which takes above calculated RSS, noise array, alpha, reference power and refernce distance for path loss model as input. It returns estimated distances.
  - Last is to find the estimated coordinates using LLS and WLS. Through computation on lls.m and wls.m , we get the estimated coordinates for all

source node positions. In lls.m we generate B and H matrix using above algorithm equations. Similarly for wls.m .

- Concatenating the originally generated mobile source coordinates through mobility patterns we get the actual coordinates.
- Hence we compute error between actual and estimated coordinates using compute_error.m function. Error is computed as euclidean distance between actual and estimated coordinates.
- This error for all mobility patterns and LLS and WLS estimation is returned as a result of test_function.m .

- Now we plot this error for all mobility patters as well as for both estimation functions.
- First plot is for two estimation functions and hence contains 2 subplots.
- Second plot is for various mobility patterns and hence contains 3 subplots.

OUTPUTS AND RESULTS:

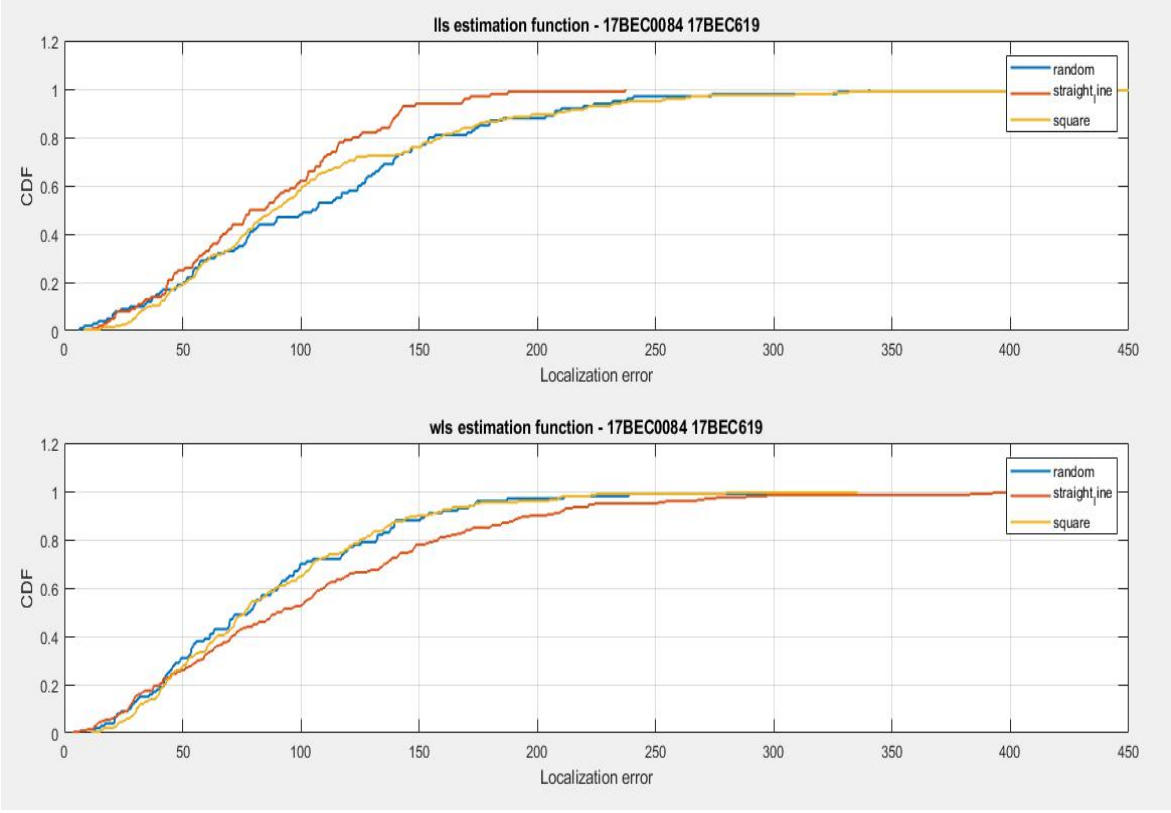

Fig 2: CDF vs Localization error plot for LLS and WLS estimation functions.

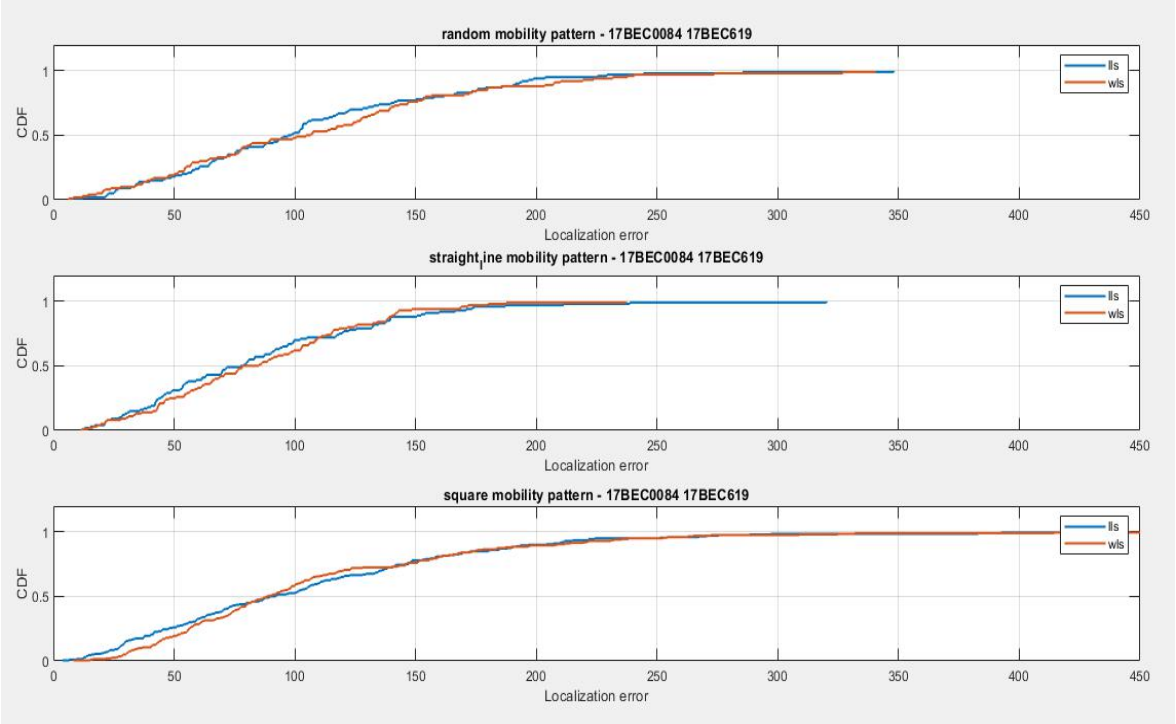

Fig 3: CDF vs Localization error plot for various mobilty patterns for the mobile node.

INFERENCE:

As we can observe from the above graphs:

Fig 2: The WLS estimation function is more accurate than the LLS estimation. The CDF becomes constant after a range and hence the error between the accurate and estimated coordinates is reduced.

Fig 3: From the graphs we can observe LLS and WLS for various mobility patterns. Error is minimum for straight line movement.

FUTURE WORK:

This project can futher be extended for other mobility patterns like circle. Morever filters could be used along with estimation functions i.e LLS and WLS like Kalman filters to get more accurate results.

LINK TO GITHUB REPOSITORY FOR CODE FILES:

https://github.com/ShambhaviAwasthi/WMC-Project---MATLAB-simulation-of-RSS-based-channel-modelling-localization-and-tracking

REFERENCES:

1. Location Estimation Methods - Shahin Farahani, in ZigBee Wireless Networks and Transceivers, 2008

2. Fundamentals of communication networks - Shiwen Mao, in Cognitive Radio Communications and Networks, 2010

3. Optimization of anchor nodes in wireless sensor network - Devendra Kumar Yadav ; Pragyan Mishra ; Sasmita Behera

4. Application of Channel Modeling for Indoor Localization Using TOA and RSS - Ahmad Hatami

5. Analysis of path loss exponent error in ranging and localization of wireless sensor network - Chuan Chin Pu ; Pei Cheng Ooi ; Boon Giin Lee ; Wan-Young Chung