Technische Universität Dortmund
Department of Biochemical and Chemical Engineering
Chair of Process Dynamics and Operations
Prof. Dr. Sebastian Engell

DEVELOPMENT OF LOCAL POSITIONING SYSTEM FOR A PIPE-LESS PLANT

Automation & Robotics
Group Project SS18

Group Members:

Abdulrahman Abouelkhair(4511328)
Medhini Rajagopal Balamurugan(4511328)
Stefan Rottstegge(4511328)
Stephan Vette(4511328)

Supervisors:

Afaq Ahmad
Marina Rantanen-Modéer

Abstract

Summary. Note that the abstract heading is unnumbered, it should remain so. To remove heading numbering use:

```
\section*{}
```

Contents

List of Figures

List of Tables

1 Introduction

Add your name to the file name

2 Pipeless Plant

2.1 Existing setup

2.2 Problems with the Existing Setup

..
zb

- Fish eye
- Sunlight..

3 Selection Process

About the 4 techniques..

3.1 Triangulation

Summary

Implementation

Pro and con

..

3.2 Pattern Recognition

Summary

Implementation

Pro and con

..

3.3 RFID

Summary

Implementation

Pro and con

..

3.4 Map-Based Localization

Summary

Implementation

Pro and con

..
example:

Col1	Col2	Col2	Col3
1	6	87837	787
2	7	78	5415
3	545	778	7507
4	545	18744	7560
5	88	788	6344

Table 1: Should be a caption

4 Theoretical Background

4.1 Radio Frequency Identification

4.2 Trilateration[1]

Trilateration is a method to compute the intersecting point of three circles/spheres. For this, it is necessary to know the three center of the circles/spheres plus their corresponding radii. The basic idea is to use the description of sphere.[2]

$$r^2 = (x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2 \tag{1}$$

where ($P_n = (x_n, y_n, z_n)$) is the center of the sphere. To use equation (1) for the 2D indoor localization on a floor, a few assumption can be made. First of all, the z-component of all spheres can be neglected. Another assumption is that we define the origin of the first circle as the center of the coordinate system, the second along the x-axis with an distance (d) and the third shifted in x- (i) and y-direction (j).

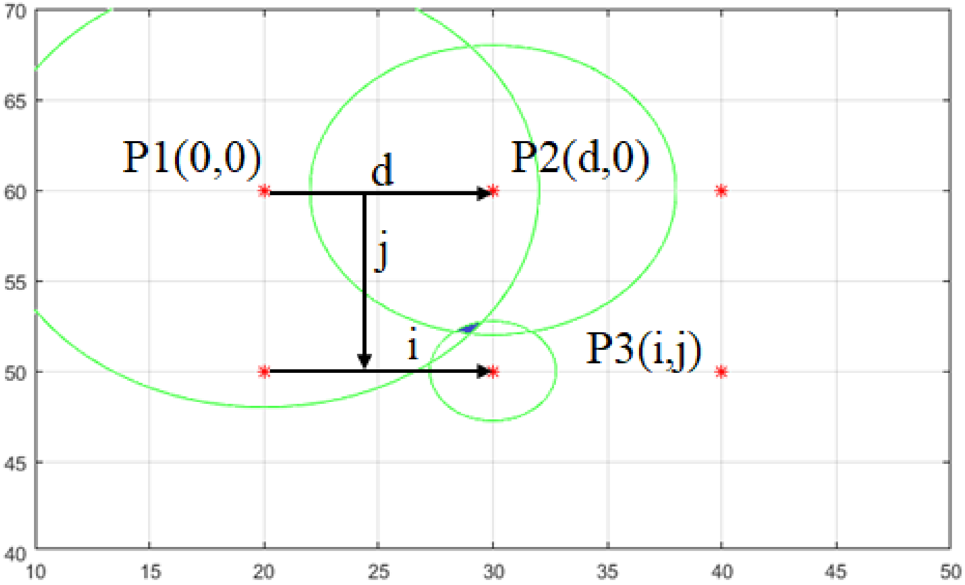


Figure 1: Overview Trilateration

With known positions of the center of the circles d, i and j can be computed in the following

[1] Stephan

[2] Indoor Robot Positioning using an Enhanced Trilateration Algorithm

way: [3]

$$d = |P_2 - P_1| \tag{2}$$

$$e_x = \frac{1}{d}(P_2 - P_1) \tag{3}$$

$$a_x = P_3 - P_1 \tag{4}$$

$$i = e_x \cdot a_x \tag{5}$$

$$a_y = (P_3 - P_1) - i * e_x \tag{6}$$

$$e_y = \frac{a_y}{|a_y|} \tag{7}$$

$$j = e_y \cdot a_x \tag{8}$$

After knowing the these values, the relative distance in x- and y-direction can be computed with the help of 1 and the center of the circles P_1(0,0), P_2(0,d) and P_3(i,j) as follows:

$$x_t = \frac{r_1^2 - r_2^2 + d^2}{2 * d} \tag{9}$$

$$y_t = \frac{r_1^2 - r_3^2 + i^2 + j^2}{2 * j} - i * \left(\frac{x_t}{j}\right) \tag{10}$$

The absolute position of the intersection point is computed in following way:

$$P = P_1 + e_x * x_t + e_y * y_t \tag{11}$$

4.3 ...

[3] Indoor Robot Positioning using an Enhanced Trilateration Algorithm

5 Simulation[4]

Before building up rea

5.1 Emulator

- based on the paper -¿ ref and datasheet
-

5.2 RSSI Measurements with real HW

- Measurement setup
- results -¿ table
- consequences

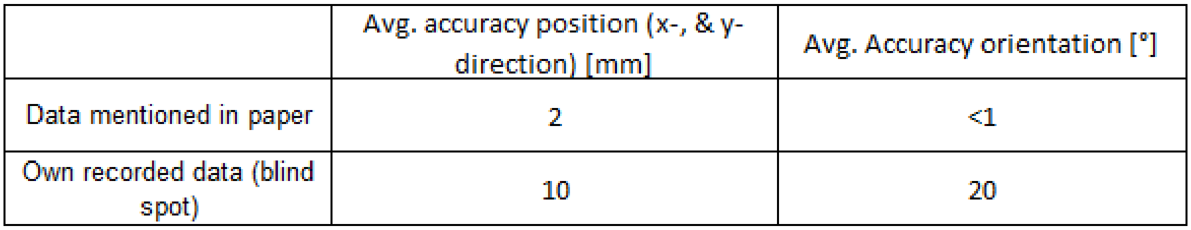
5.3 Simulation with emulated data

-

5.4 Results

After

Table 2: -XX-



	Avg. accuracy position (x-, & y-direction) [mm]	Avg. Accuracy orientation [°]
Data mentioned in paper	2	<1
Own recorded data (blind spot)	10	20

[4] Stephan

6 Implementation

7 Hardware[5]

7.1 Communication (Abdul and/or Stefan)

7.2 Initialization procedure (Stephan and Stefan)

7.2.1 Recording and filtering data (Stefan)

7.2.2 Analysing data (Stefan)

7.2.3 Estimation of position and orientation (Stephan)

7.3 Results

7.4 Improvements

[5] Abdul and Stephan

8 Conclusion

conclude..

9 Future Work

...

10 References

..

11 Appendixes

11.1 Emulator RFID data (Matlab)

```
%% ------------------------------------------------------------------------
% Description:  Emulator, which creates txt file like the reader
%               RSSI related to the real measurements
%               For the Initialization procedure, turn around 360°
% Date:         12.06.2018
% Created by:   Stephan Vette
% -------------------------------------------------------------------------
%% RFID signal emulator
clear all
clc
close all
% Initializing
l1 = 100;   % length of the plant, x  [cm]
l2 = l1;   % width of the plant, y   [cm]
d1 = 10;    % distance between tags     [cm]
d2 = 0;     % distance last tag <-> boarder [cm]
r1 = 14;    % radius of the reading range of every tag
r2 = [r1, 9.75, 9.0, 8.0, 7.0, 6.0, 5.8, 5.5, 5.3, 5.1, 5.0, 4.7, 4.5, 4.3, 4.2, 4.0, 3.5, 2.75, 0]; % distances at certain RSSI
r4 = [0, 1, 2, 3, 4, 5, 4, 3, 2, 1, 0, 1, 2, 3, 4, 5, 6, 7, 7]; % array with the different RSSI values

r3 = 33/2;  % radius of the robot
d3 = 10;    % distance between origin robot and origin antenna [cm]

angle1 = 45; % angle between the measurement points in the init procedure

gamma1 = deg2rad(22.5);  % Start orientation of robot [rad]
robStart = [22.5, 51.5]; % Start position of robot in x, y [cm]

robSpeed = 0.1;        % Speed robot [m/s]
cycleT = 100;         % Cycletime in [ms]

```

```
mode = 1;    % mode=1: tracking all available tags, which are nonzero
             % mode=2: tracking only changes in the RSSI signals
mode_hex = 0;    % activate or deactivate hex ID

% For the name of the txt file
measuementeNumber = num2str(11); % Number of measurement
% Two possibilities for the content of the txt file
% 1. Without filtering. Exactly like the reader creates data
% text0 = '<\r>';
% text1 = 'OK';
% text2 = 'SCAN:+UID=';
% text3 = '+RSSI=';

% 2. Filtered data. Without unusable information.
text0 = ' ';
text1 = ' ';
text2 = ' ';
text3 = ' ';
%% Error check
if mod(l1/d1,1)~=0
    error('Length of platform not dividable by distance between tags');
elseif mod(l2/d1,1)~=0
    error('Length of platform not dividable by distance between tags');
end

%% Computing position of antenna
numTagsX = (l1-2*d2)/d1 +1;
numTagsY = (l2-2*d2)/d1 +1;
numTags = numTagsX * numTagsY;
antPos = robStart + d3 * [cos(gamma1), sin(gamma1)];

%% Display the setup, write important information into a seperate txt file
d1_str = num2str(d1);
l1_str = num2str(l1);
l2_str = num2str(l2);
numTags_str = num2str(numTags);

msg0 = ['Your plane is ',l1_str,'cm x ',l2_str,'cm.'];
```

```
msg1 = ['You chose a distance of ',d1_str, 'cm and need ',numTags_str,' Tags!'];
disp(msg0);
disp(msg1);
nameTxt = ['NumTags',measuementeNumber,'.txt'];
fileNumTags = fopen(nameTxt,'w');
fprintf(fileNumTags,'%6d\n',numTags);    % Write the number of tags in file
fprintf(fileNumTags,'%6d\n',l1);         % Write the size of the plant in file
fprintf(fileNumTags,'%6.4f\n',gamma1);       % Write the starting angle
fprintf(fileNumTags,'%6d\n',robStart(1));      % Write the starting pos
fprintf(fileNumTags,'%6d\n',robStart(2));      % Write the starting pos
fclose(fileNumTags);


%% Drawing environment
figure(1)
x1 = [0 l1 l1 0 0];
y1 = [0 0 l2 l2 0];
plot(x1, y1,'LineWidth',2)
xlim([-5 (l1+5)]);
ylim([-5 (l2+5)]);
hold on

% Position of the tags
ID = 1:numTags;
[Tagx,Tagy] = meshgrid(d2:d1:l1-d2,d2:d1:l2-d2);
plot(Tagx,Tagy,'r*')
% Circles
radiipl = ones(numTagsX,1)*r1;
for k=1:numTagsX
    tempx = Tagx(1:end,k);
    tempy = Tagy(1:end,k);
    temppos = horzcat(tempx,tempy);
    viscircles(temppos,radiipl,'Color','k','LineStyle',':','LineWidth',0.25);
end
robX = robStart(1);
```

```
robY = robStart(2);
plot(robX,robY,'bO','LineWidth',3);
plot(robX,robY,'r:');
viscircles([robX,robY],r3,'Color','k','LineWidth',0.25);
plot(antPos(1),antPos(2),'bs');
xlabel('Length platform in cm')
ylabel('Width platform in cm')
title({'Position and reading range of tags';'Start-, endpoint and path of the robot'});
hold off
pause(1)

%% Animation and loggin
xUpdateAnt = antPos(1);
yUpdateAnt = antPos(2);
deltaR = deg2rad(angle1);       % A new measurement after every XX°
% Txt file name
name = ['Meas_StartingProc_like_reader_real_data',measuementeNumber,'.txt'];
fileID = fopen(name,'w');

% Data stored in variables
dataRSSI = zeros(8,numTags);
streamDataRSSI = zeros(1,numTags);
streamDataRSSIold = zeros(1,numTags);
timeStep = 1;   % current measurement step

% antPos = robStart + d3 * [cos(gamma1), sin(gamma1)];
figure(2)
for l=0:360/angle1
    deltaR_temp = deltaR * l;
    xUpdateAnt = robStart(1) + d3 * cos(gamma1 + deltaR_temp);
    yUpdateAnt = robStart(2) + d3 * sin(gamma1 + deltaR_temp);
    plot(x1, y1,'LineWidth',2)
    hold on
    xlim([-5 (l1+5)]);
    ylim([-5 (l2+5)]);
    [Tagx,Tagy] = meshgrid(d2:d1:l1-d2,d2:d1:l2-d2);
    plot(Tagx,Tagy,'r*')
    plot(robX,robY,'bO','LineWidth',1);
    plot(robX,robY,'r:');
    plot(xUpdateAnt,yUpdateAnt,'bs');
    xlim([-5 (l1+5)]);
    ylim([-5 (l2+5)]);
    viscircles([robX,robY],r3,'Color','b','LineWidth',0.5);
        for k=1:numTagsX
            tempx = Tagx(1:end,k);
            tempy = Tagy(1:end,k);
            temppos = horzcat(tempx,tempy);
            viscircles(temppos,radiipl,'Color','k','LineStyle',':','LineWidth',0.25);
        end
    hold off

    % Creating measurements
    antPosnew=[xUpdateAnt,yUpdateAnt];
    for m = 1:numTags                          % m = current number of tag
        m_str = num2str(m);
        tempTag=[Tagx(m),Tagy(m)];
        tempD = pdist([antPosnew; tempTag] ,'euclidean');

        % Display if tag is in range or not
        if tempD > r1
                streamDataRSSI(m) = 0;
                if (streamDataRSSI(m) ~= streamDataRSSIold(m)) && mode == 2
                    if mode_hex == 1
                        fprintf(fileID,'%d %s%s%s%d%s\n',l*angle1,text2,dec2hex(m, 16),text3,k(end),text0);
                    elseif mode_hex == 0
                        fprintf(fileID,'%d %s%d%s%d%s\n',l*angle1,text2,m,text3,k(end),text0);
                    end
                     fprintf(' %d %d %1d\n',l*angle1,m,'0');
                end
        elseif tempD <= r1
                 % disp(['Label ',m_str,' in range!!!!!!!!!!!!!']);
                 % Relation distance <-> RSSI
                 k_temp = find(r2>=tempD);
                 k = r4(k_temp);
                 dataRSSI(timeStep,m) = k(end);
                 streamDataRSSI(m) = k(end);
                 if (streamDataRSSI(m) ~= streamDataRSSIold(m)) && mode == 2
```

```
                    if mode_hex == 1
                        fprintf(fileID,'%d %s%s%s%d%s\n',l*angle1,
                           text2,dec2hex(m, 16),text3,k(end),text0);
                    elseif mode_hex == 0
                        fprintf(fileID,'%d %s%d%s%d%s\n',l*angle1,
                           text2,m,text3,k(end),text0);
                    end
                    fprintf(' %d %d %8d,\n',l*angle1,m,k(end));
                elseif mode == 1
                    if mode_hex == 1
                        fprintf(fileID,'%d %s%s%s%d%s\n',l*angle1,
                           text2,dec2hex(m, 16),text3,k(end),text0);
                    elseif mode_hex == 0
                        fprintf(fileID,'%d %s%d%s%d%s\n',l*angle1,
                           text2,m,text3,k(end),text0);
                    end
                    fprintf(' %d %d %1d,\n',l*angle1,m,k(end));
                end
        end
    end
    streamDataRSSIold = streamDataRSSI;
    pause(cycleT/1000)
    timeStep = timeStep + 1;
end
savefig('Figure2.fig');
fclose(fileID);

%% Results
% figure(3)   % plot for the max value of every tag
% dataRSSInoT = reshape(max(dataRSSI),[numTagsX,numTagsY]);
% plot3(Tagx,Tagy,dataRSSInoT,'*');
% xlabel('Length platform in cm')
% ylabel('Width platform in cm')
% title('Max RSSI signal of every tag')

figure(4)   % plot of the RSSI signal which are non zero vs. time
dataRSSIsum = sum(dataRSSI);
IDclear = find(dataRSSIsum ~= 0);
IDstr = string(IDclear);
dataRSSIclear = dataRSSI;
dataRSSIclear( :, all( ~any( dataRSSI ), 1 ) ) = []; % and columns
plot(dataRSSIclear);
```

```
xlabel('Measurement points')
ylabel('RSSI')
ylim([0 360/angle1])
legend(IDstr,'FontSize',6);
title('RSSI Signal of every non zero tag')
```