# Localization Using a Microphone Network

Group 102

Eric Moringe – erimo668
David Wiman – davwi279

## I Introduction

The purpose of this lab was to estimate the location of and track a small autonomous car emitting a known sound pulse roughly twice every second. This was done by placing four microphones around the area that the car was within and examining at what time each of the microphones picked up the sound pulses at. This is an example of a passive SONAR network. Since the purpose of the lab was to estimate the car's position in space, the equations used were multiplied by the speed of sound at sea level to get the answers in distance, rather than time. Since the microphones' positions were known with high accuracy, the position of the car should be possible to determine fairly well. Throughout the report, the words sensor and microphone as well as the words car and target will be used synonymously.

## II Sensor Calibration

To determine the characteristics of the individual microphones, a calibration run was performed. All eight microphones (two sets of four) were place equally far away from the car as it stood still and emitted sound pulses. The sound pulse that was used was a bi-directional chirp ranging from 600 to 1400 Hz and the time between pulses was roughly 0.5 s. This pulse was used since it is relatively easy to pinpoint in time. The different arrival times could then be compared to the average time of arrival (TOA) for each pulse and the bias and variance was computed. The empirical probability density functions of the measurement noise in each microphone can be seen in Figure 1. The computed variances were used to construct variance matrices used in later tasks.

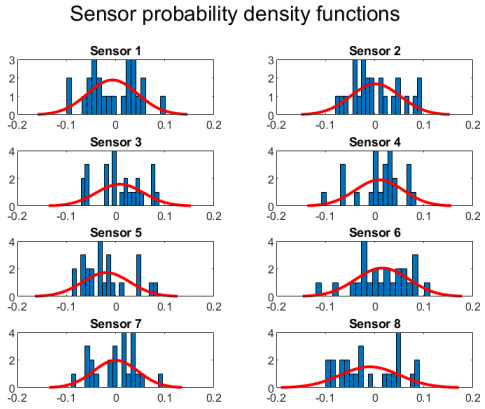


Figure 1: Sensor probability density functions

## III Signal Modeling

In this part, two different signal models were chosen. The first one was a TOA model with known sensor positions $p_k$ but with unknown pulse time $r_0$, see Equation (1). The other one was a TDOA model (time difference of arrival) were the pulse times have been eliminated by looking at the difference between arrival times at different microphones, see Equation (2). Microphone 1 was treated as a reference.

$$\begin{aligned} &y_k = ||x - p_k|| + r_0 + e_k, \\ &k = 1, 2, 3, 4 \ , \ e_k \sim \mathcal{N}(0, R_k) \end{aligned} \tag{1}$$

$$\begin{aligned} &y_k = \ ||x - p_k|| - ||x - p_1|| + e_k - e_1, \\ &k = 2, 3, 4 \ , \ e_k \sim \mathcal{N}(0, R_k), \ e_1 \sim \mathcal{N}(0, R_1) \end{aligned} \tag{2}$$

The Equations (1) and (2) above can also be found in Section IX.3 as an m-file.

For model 1, the variance matrix $P_1$ had the individual variance of each sensor along the diagonal,

$$P_1 = \begin{bmatrix} R_1 & 0 & 0 & 0 \\ 0 & R_2 & 0 & 0 \\ 0 & 0 & R_3 & 0 \\ 0 & 0 & 0 & R_4 \end{bmatrix}. \tag{3}$$

For model 2, the variance matrix $P_2$ had the sum of sensor 1's variance and another sensor's variance along the diagonal as well as the variance of sensor 1 everywhere else,

$$P_2 = \begin{bmatrix} R_1 + R_2 & R_1 & R_1 \\ R_1 & R_1 + R_3 & R_1 \\ R_1 & R_1 & R_1 + R_4 \end{bmatrix} \tag{4}$$

# IV Experiments

Since it was suspected that the placement of the microphones could effect the result, two sets of four microphones each in two different configurations were used for the experiment, hence the eight microphones mentioned in Section II. Microphone 1 through 4 in Figure 1 belong to configuration 1 and microphones 5 through 8 belong to configuration 2. The two configurations can be seen in Figures 2 and 3. Notice however that both sets of microphones are numbered 1 through 4. The label T1 indicates the starting position of the car.

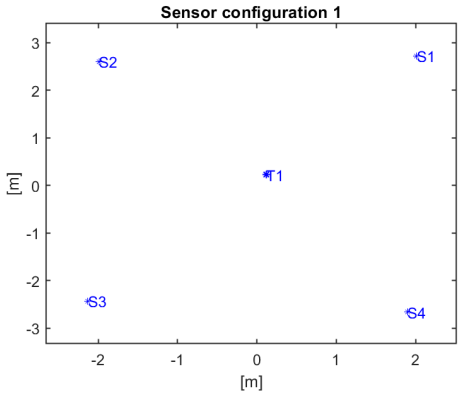


Figure 2: Sensor configuration 1.

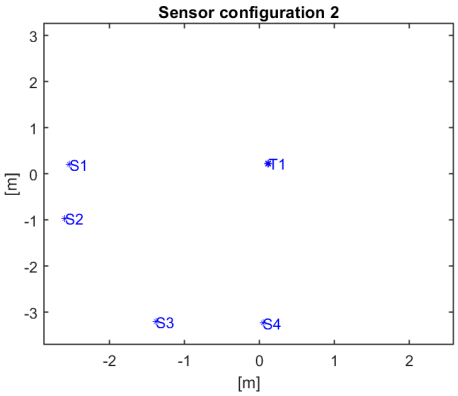


Figure 3: Sensor configuration 2.

Configuration 1 was chosen as it was expected to be the most suitable for the task since it was equally spaced and well distributed across the target area. Configuration 2 was more focused on a certain corner of the target area and was expected to have worse performance over all. Two fundamentally different configurations were chosen in order to see varying behavior.

# V Configuration Analysis

To test the hypothesis in Section IV and determine which configuration should be used moving forward, a test was performed using model 1 and the nonlinear least squares (NLS) loss function,

$$V(x) = (y - h(x))^T R^{-1} (y - h(x)). \tag{5}$$

The loss function was computed across a grid covering all possible position of the car with a resolution of 10 cm using both microphone configurations and the results were compared. The surface plots of the loss function can be seen in Figures 4 and 5.

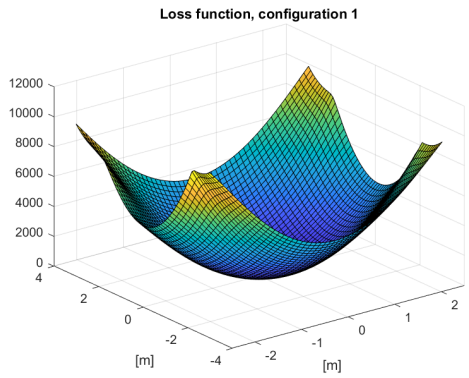


Figure 4: NLS loss function across a grid for configuration 1.

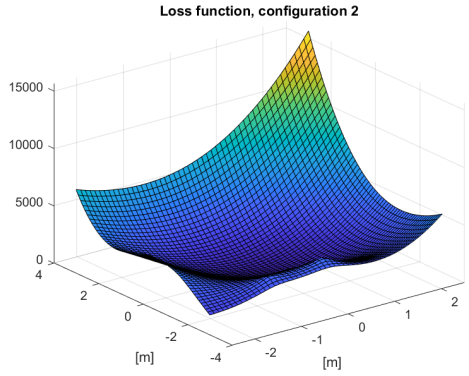


Figure 5: NLS loss function across a grid for configuration 2.

As predicted, configuration 2 is very accurate (small value for the loss function) in the corner where the microphones are located but severely worse when the car is far away. Model 1 performs at its best in the middle of the grid and worse at the corners. However, it does not perform as poorly as model 2. It was decided to use configuration 1 moving forward.

# VI Localization

To estimate the position of the car as it drove along a predetermined trajectory in the shape of a figure 8, two different methods were used, from now on called method 1 and method 2. The car drove two laps in order to get enough usable data.

1. Signal model 1 (TOA) was used together with NLS using a Gauss-Newton search.
2. Signal model 2 (TDOA) was used together with NLS.

For method 1, the measurements that had been collected could be used directly, but for method 2, a new measurement matrix made up of the differences in TOA for three pairs of sensors, 1-2, 1-3 and 1-4 had to be constructed. Sensor one was the reference sensor.

To initialize the parameter $r_0$ for the Gauss-Newton search in method 1, the earliest TOA (converted to distance) for any sensor was used.

The results can be seen together with the ground truth (red path) in Figures 6 and 7. The results from method 2 were the better of the two.

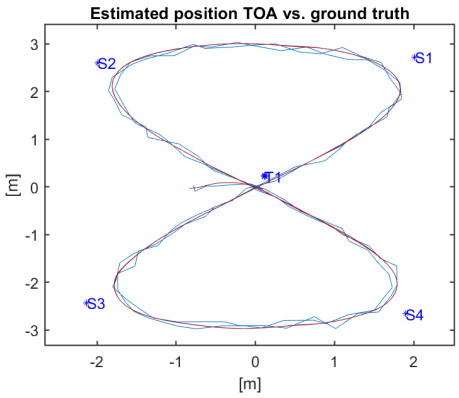


Figure 6: Estimated position for method 1, TOA vs. ground truth.

# VII Tracking

To track the car, two motion models were used together with two measurement models. The first motion model was a constant velocity model (CV2D) and the second one was a coordinated

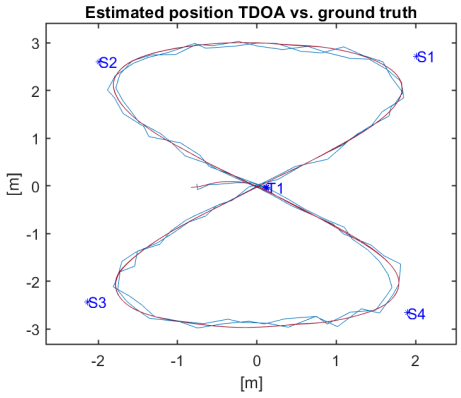


Figure 7: Estimated position for method 2, TDOA vs. ground truth.

turn model (CTCV2D), both taken from the SigSys toolbox in Matlab. The matrix representation of the CV2D-model can be seen in Equation (6). The CTCV2D-model cannot be represented on matrix form since all parts of the model are updated independently. However, the formulas for these updates can be seen in Equation (7).

$$\bar{x}(t+T) = \begin{bmatrix} I_n & TI_n \\ 0_n & I_n \end{bmatrix} \bar{x} + \begin{bmatrix} \frac{T^2}{2} I_n \\ TI_n \end{bmatrix} \bar{w} \tag{6}$$

$$\begin{cases} x(t+T) = x + \frac{v_x}{\omega} sin(\omega T) - \frac{v_y}{\omega}(1 - cos(\omega T)) + \frac{T^2}{2} w_x \\ y(t+T) = y + \frac{v_x}{\omega}(1 - cos(\omega T)) + \frac{v_y}{\omega} sin(\omega T) + \frac{T^2}{2} w_y \\ v_x(t+T) = v_x cos(\omega T) - v_y sin(\omega T) + T w_{v_x} \\ v_y(t+T) = v_x sin(\omega T) + v_y cos(\omega T) + T w_{v_y} \\ \omega(t+T) = \omega + T w_\omega \end{cases} \tag{7}$$

Since the motion models are written in the general form with varying factors in front of the different noise variables $w$, we need to modify which covariance matrix is used in the code, since it otherwise uses the covariance matrix for the case without the factors. This is done by doing the following general matrix multiplication: $Q = G \cdot q \cdot G^T$. Here Q is the new covariance matrix that fits in the toolbox and is used within the model in the EKF later, G is the matrix with the factors in front of the different noise variables $w$ and q is a tuning parameter of suitable size.

For the CV2D-model we have

$$G_{CV2D} = \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix} \tag{8}$$

and

$$q_{CV2D} = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.01 \end{bmatrix}. \tag{9}$$

With $T = 0.5$ this yields the covariance matrix

$$Q_{CV2D} = 10^{-3}. \begin{bmatrix} 0.15625 & 0 & 0.6250 & 0 \\ 0 & 0.15625 & 0 & 0.625 \\ 0.625 & 0 & 2.5 & 0 \\ 0 & 0.625 & 0 & 2.5 \end{bmatrix}. \tag{10}$$

This gives us that our noise for the CV2D-model follows the distribution

$$\bar{w} \sim \mathcal{N}(0, Q_{CV2D}). \tag{11}$$

For the CTCV2D-model we have

$$G_{CTCV2D} = \begin{bmatrix} \frac{T^2}{2} & 0 & 0 \\ 0 & \frac{T^2}{2} & 0 \\ T & 0 & 0 \\ 0 & T & 0 \\ 0 & 0 & T \end{bmatrix} \tag{12}$$

and

$$q_{CTCV2D} = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0.01 \end{bmatrix}. \tag{13}$$

With $T = 0.5$ this yields the covariance matrix

$$Q_{CTCV2D} = 10^{-3}. \begin{bmatrix} 0.15625 & 0 & 0.6250 & 0 & 0 \\ 0 & 0.15625 & 0 & 0.625 & 0 \\ 0.625 & 0 & 2.5 & 0 & 0 \\ 0 & 0.625 & 0 & 2.5 & 0 \\ 0 & 0 & 0 & 0 & 2.5 \end{bmatrix}. \tag{14}$$

This gives us that our noise for the CTCV2D-model follows the distribution

$$\begin{bmatrix} w_x \\ w_y \\ w_{v_x} \\ w_{v_y} \\ w_\omega \end{bmatrix} \sim \mathcal{N}(0, Q_{CTCV2D}). \tag{15}$$

All the process noise in both models, meaning $q_{CV2D}$ and $q_{CTCV2D}$ above, was initially assumed to be uncorrelated and have the variance $0.1^2$. This was a crude simplification but since the results were satisfactory, the assumptions were not altered. The size of the tuning parameter q in both cases is determined by the matrix G describing the noise. Since $G_{CV2D}$ has two columns, $q_{CV2D}$ has to be 2x2 in size and the same goes for $G_{CTCV2D}$ which has three columns and thus require $q_{CTCV2D}$ to be 3x3 in size.

The first measurement model that was tested had the following form,

$$y_k = \hat{p}_k + e_k, \; e \sim \mathcal{N}(0, R), \tag{16}$$

where $\hat{p}_k$ is the estimated position according to method 2 in Section VI. The second measurement model was the previously described TDOA model. In all four cases, an extended Kalman filter was used for the estimation. All the results can be seen in Figures 8–11.

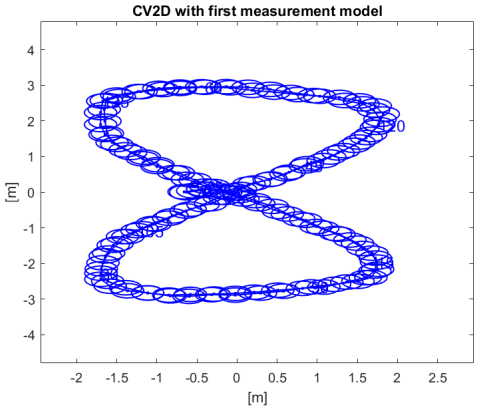


Figure 8: Tracking the car using the CV2D motion model and the first measurement model. 90% confidence interval.

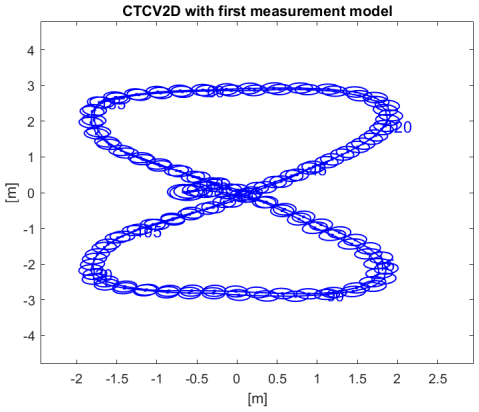


Figure 9: Tracking the car using the CTCV2D motion model and the first measurement model. 90% confidence interval.

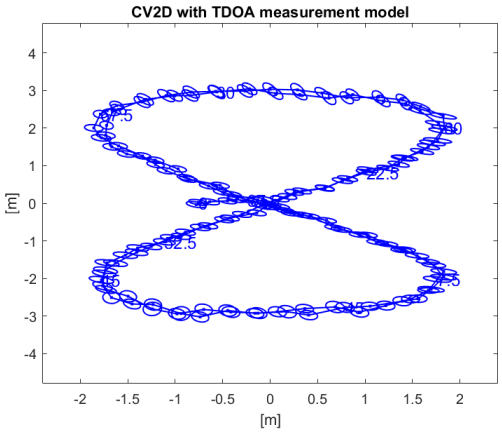


Figure 10: Tracking the car using the CV2D motion model and the TDOA measurement model. 90% confidence interval.

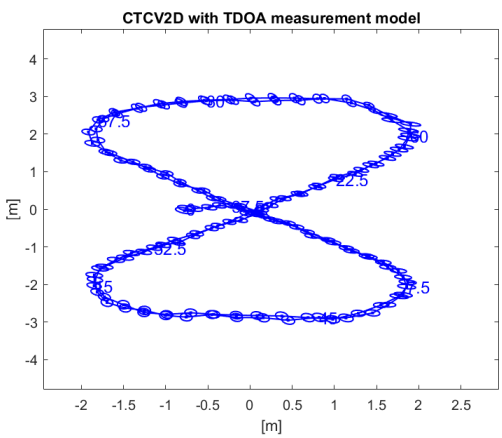


Figure 11: Tracking the car using the CTCV2D motion model and the TDOA measurement model. 90% confidence interval.

Firstly, it can be seen that the CTCV2D motion model gives better results than CV2D for both measurement models. Secondly, the TDOA measurement model is better than the first measurement model for both motion models. Ergo, the best results comes from using CTCV2D and TDOA (Figure 11). Conversely, the most inaccurate results comes from CV2D and the first measurement model (Figure 8).

# VIII Discussion

To examine the robustness of the tracking methods, one method was chosen and the position of the microphones were disturbed in order to empirically evaluate the sensitivity. The chosen method used the CV2D motion model with the TDOA measurement model (Figure 10) and an extended Kalman filter. After testing, it could be concluded that when normally-distributed random numbers were added to the microphones' positions, the method failed to track the target properly when the variance was greater than 0.4 m, see Figures 12 and 13.

Figures 12 and 13 were based on randomly drawn numbers and will of course differ from one run to another. However, after several test, these seem to be representative results.

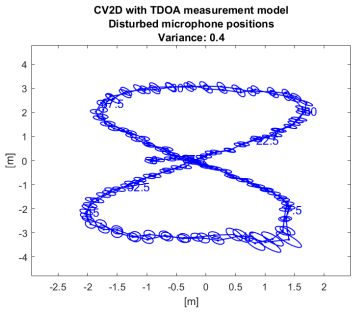


Figure 12: Tracking the car using the CV2D motion model and the TDOA measurement model. Disturbed microphone positions with a variance of 0.4 m.

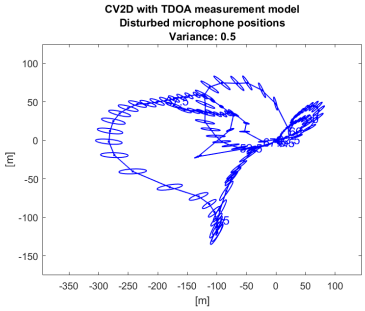


Figure 13: Tracking the car using the CV2D motion model and the TDOA measurement model. Disturbed microphone positions with a variance of 0.5 m.

# IX Matlab Code

## IX.1 main.m

```
clear
clc

plot_on = 0;
run('startup');
run('C:\Users\David Wiman\OneDrive\Dokument\MATLAB\TSRT14\initSigSys.m')

%% Calibration

filename = 'calibration';
run('SFlabRun');
run('task1');

%% Live run

filename = 'grupp7';
run('SFlabRun');
run('task2');
run('task3');
run('task4');
run('task5b');
run('task5d');
run('task6a');
run('task6b');
run('task7');
```

## IX.2 task1.m

```
% Allocate space
measurement_error = zeros(8,24);
mean_value = zeros(1,24);

% Turn tphat from time to distance
tphat_dist = tphat*343; % Speed of sound at sea level

% Calculate the mean across all sensors per measurement
for index = 1:length(tphat_dist)
    mean_value(index) = mean(tphat_dist(:,index));
end

% Calculate measurement error vectors, we assume the mean is the true time
% of arrival
for sensor = channels
        measurement_error(sensor,:) = tphat_dist(sensor,:) - mean_value;
end

% Calculate histogram, etc.
```

```
for index = channels
    if plot_on
        figure(index);
        histfit(measurement_error(index,:), 20);
    end
    bias(index) = mean(measurement_error(index,:));
    variance(index) = var(measurement_error(index,:));
end
```

## IX.3 task2.m

```
% h-function for toa
h_tdoa1 = inline('[sqrt((x(1,:)-th(1)).^2+(x(2,:)-th(2)).^2) + x(3,:) ;
    sqrt((x(1,:)-th(3)).^2+(x(2,:)-th(4)).^2)+ x(3,:) ; sqrt((x(1,:)-th(5))
    .^2+(x(2,:)-th(6)).^2)+ x(3,:) ; sqrt((x(1,:)-th(7)).^2+(x(2,:)-th(8))
    .^2)+ x(3,:)]', 't', 'x', 'u', 'th');

% h-function for tdoa2
h_tdoa2 = inline('[sqrt((x(1,:)-th(1)).^2+(x(2,:)-th(2)).^2) - sqrt((x(1,:)
    -th(3)).^2+(x(2,:)-th(4)).^2) ; sqrt((x(1,:)-th(1)).^2+(x(2,:)-th(2))
    .^2) - sqrt((x(1,:)-th(5)).^2+(x(2,:)-th(6)).^2) ; sqrt((x(1,:)-th(1))
    .^2+(x(2,:)-th(2)).^2) - sqrt((x(1,:)-th(7)).^2+(x(2,:)-th(8)).^2)]', '
    t', 'x', 'u', 'th');
```

## IX.4 task3.m

```
% Set constants and create target_start_pos, data from lab session
nr_targets = 1;
nr_sensors = 4;
target_start_pos = [position(1,2), position(1,3), position(1,1)];

% Square, setup 1
sensor_setup_1_pos = [2.0133, 2.7216, -1.9919, 2.6033, -2.1295, -2.4340,
    1.8988, -2.6645];

% L-shaped, setup 2
sensor_setup_2_pos = [-2.5393, 0.2055, -2.6009, -0.9686, -1.3848, -3.2034,
    0.0435, -3.2329];

% Calculate pe for setup 1 tdoa2
pe_1 = variance(1)*ones(3,3);
pe_1 = pe_1 + diag([variance(2), variance(3), variance(4)]);

% Calculate pe for setup 2 tdoa2
pe_2 = variance(5)*ones(3,3);
pe_2 = pe_2 + diag([variance(6), variance(7), variance(8)]);

% Create sensor network for setup 1, tdoa1
s_setup_1_tdoa1 = sensormod(h_tdoa1, [3, 0, 4, 8]);
s_setup_1_tdoa1.x0 = target_start_pos;
s_setup_1_tdoa1.th = sensor_setup_1_pos;
```

```
s_setup_1_tdoa1.pe = diag(variance(1:4));
R_setup_1_tdoa1 = diag(variance(1:4));
figure(31)
plot(s_setup_1_tdoa1)

% Create sensor network for setup 2, tdoa1
s_setup_2_tdoa1 = sensormod(h_tdoa1, [3, 0, 4, 8]);
s_setup_2_tdoa1.x0 = target_start_pos;
s_setup_2_tdoa1.th = sensor_setup_2_pos;
s_setup_2_tdoa1.pe = diag(variance(5:8));
R_setup_2_tdoa1 = diag(variance(5:8));
figure(32)
plot(s_setup_2_tdoa1)

% Create sensor network for setup 1, tdoa2
s_setup_1_tdoa2 = sensormod(h_tdoa2, [2, 0, 3, 8]);
s_setup_1_tdoa2.x0 = target_start_pos(1:2);
s_setup_1_tdoa2.th = sensor_setup_1_pos;
s_setup_1_tdoa2.pe = pe_1;

% Create sensor network for setup 2, tdoa2
s_setup_2_tdoa2 = sensormod(h_tdoa2, [2, 0, 3, 8]);
s_setup_2_tdoa2.x0 = target_start_pos(1:2);
s_setup_2_tdoa2.th = sensor_setup_2_pos;
s_setup_2_tdoa2.pe = pe_2;
```

## IX.5 task4.m

```
% Compute V across a grid of possible car positions, -3 to 3 by -2.5 to 2.5
    meters

% Allocate space and create temporary sensormod objects
V_setup_1_tdoa1 = zeros(61, 51);
V_setup_2_tdoa1 = zeros(61, 51);
s_setup_1_tdoa1_tmp = s_setup_1_tdoa1;
s_setup_2_tdoa1_tmp = s_setup_2_tdoa1;

% V for setup 1
y = simulate(s_setup_1_tdoa1_tmp, 1);
for x_index = 1:61
    for y_index = 1:51
        s_setup_1_tdoa1_tmp.x0 = [(x_index/10)-3; (y_index/10)-2.5;
            position(1,1)];
        h = h_tdoa1(0, s_setup_1_tdoa1_tmp.x0, 0, s_setup_1_tdoa1_tmp.th);

        R = R_setup_1_tdoa1;

        V_setup_1_tdoa1(x_index, y_index) = (y.y'-h)'*inv(R)*(y.y'-h);
    end
end
```

```
21
22 % V for setup 2
23 y = simulate(s_setup_2_tdoa1_tmp, 1);
24 for x_index = 1:61
25     for y_index = 1:51
26         s_setup_2_tdoa1_tmp.x0 = [(x_index/10)-3; (y_index/10)-2.5;
               position(1,1)];
27         h = h_tdoa1(0, s_setup_2_tdoa1_tmp.x0, 0, s_setup_2_tdoa1_tmp.th);
28
29         R = R_setup_2_tdoa1;
30
31         V_setup_2_tdoa1(x_index, y_index) = (y.y'-h)'*inv(R)*(y.y'-h);
32     end
33 end
34
35 % Plot the results
36 figure(41)
37 surf([-2.5:0.1:2.5]', [-3:0.1:3], V_setup_1_tdoa1);
38
39 figure(42)
40 surf([-2.5:0.1:2.5]', [-3:0.1:3], V_setup_2_tdoa1);
```

## IX.6 task5b.m

```
1 % Allocate space and create temporary sensormod objects
2 x_hat_vector_tdoa1 = zeros(2, length(tphat));
3 s_setup_1_tdoa1_tmp = s_setup_1_tdoa1;
4
5 % Convert tphat to distance instead of time
6 tphat_dist = tphat*343;
7
8 % Create a SIG object of the measured time of arrivals
9 y = sig(tphat_dist(1:4,:)', 2);
10
11 % Set the target start position and a probable first r0 (earliest TOA for
      the first pulse)
12 s_setup_1_tdoa1_tmp.x0 = [position(1,2) ; position(1,3) ; min(tphat_dist
      (1:4,1))];
13
14 for time_index = 1:length(tphat_dist)-1
15     x_hat_tdoa1 = estimate(s_setup_1_tdoa1_tmp, y(time_index,:), 'thmask',
          zeros(8,1)); % esitmate target position from sensor setup and
          measurements
16     x_hat_vector_tdoa1(:,time_index) = x_hat_tdoa1.x0(1:2); % store
          measurements for ploting
17     s_setup_1_tdoa1_tmp.x0 = [x_hat_tdoa1.x0(1:2) ;  min(tphat_dist(1:4,
          time_index+1))]; % update the target position and pulse time
18 end
19
20 % We think we need to update the target position and pulse time since
```

```
% "esitmate" will performe a Gauss-Newton search in a small box around x0.
% If the true position is outside of this search-box, the best guess will
% be somewhere along the edge of the box where the loss functions is
% minimized. Moving the search-box along with
% each estimated position ensures we are sufficiently close, both in
% position and time, to the next position that it will be within the
% search-box.

% Plot the target path together with sensor positions
figure(52)
plot(x_hat_vector_tdoa1(1,:), x_hat_vector_tdoa1(2,:))
hold on
plot(s_setup_1_tdoa1)
```

## IX.7 task5d.m

```
% Allocate space
x_hat_vector_tdoa2 = zeros(2, length(tphat));
tphat_dist_diff = zeros(3, length(tphat));

% Convert tphat to distance instead of time
tphat_dist = tphat(1:4,:)*343;

% Construct pairwise differnces, sensor 1 is reference
for time_index = 1:length(tphat_dist)
    tphat_dist_diff(1,time_index) = tphat_dist(1,time_index) - tphat_dist
        (2,time_index);
    tphat_dist_diff(2,time_index) = tphat_dist(1,time_index) - tphat_dist
        (3,time_index);
    tphat_dist_diff(3,time_index) = tphat_dist(1,time_index) - tphat_dist
        (4,time_index);
end

% Create a SIG object of the measured time differences of arrival
y_diff = sig(tphat_dist_diff', 2);

for time_index = 1:length(tphat_dist)-1
    x_hat_tdoa2 = estimate(s_setup_1_tdoa2, y_diff(time_index,:), 'thmask',
        zeros(8,1)); % esitmate target position from sensor setup and
        measurements
    x_hat_vector_tdoa2(:,time_index) = x_hat_tdoa2.x0; % store measurements
        for ploting
    s_setup_1_tdoa2.x0 = x_hat_tdoa2.x0; % update the target position and
        pulse time
end

% Plot the target path together with sensor positions
figure(54)
plot(x_hat_vector_tdoa2(1,:), x_hat_vector_tdoa2(2,:))
hold on
```

```
plot(s_setup_1_tdoa2)
```

## IX.8 task6a.m

```
% Motion models
motion_model_1 = exmotion('cv2d'); % ca2d gives almost identical results
motion_model_2 = exmotion('ctcv2d');

% Measurement model
h_1 = inline('[x(1,:) ; x(2,:)]', 't', 'x', 'u', 'th');
measurement_model_1 = sensormod(h_1, [4, 0, 2, 0]);
measurement_model_1.pe = 0.1^2*eye(2); % low on this means we trust the
    measurement model
measurement_model_1.px0 = eye(4);
measurement_model_1.x0 = [x_hat_vector_tdoa2(:,1) ; 0 ; 0];

% Motion models with measurement models
motion_model_1_1 = addsensor(motion_model_1, measurement_model_1);
motion_model_2_1 = addsensor(motion_model_2, measurement_model_1);

% Tuning parameter
q_1 = 0.1^2*eye(2);
q_2 = 0.1^2*eye(3);

T = 0.5; % Since fs from exmotion is 2
G_1 = kron([T^2/2 ; T], eye(2)); % CV2D noise model
G_2 = G_1;
G_2(5,3) = T; % CTCV2D noise model

motion_model_1_1.pv = G_1*q_1*G_1';
motion_model_2_1.pv = G_2*q_2*G_2';

% Create a SIG object of the artificial measurements
y = sig(x_hat_vector_tdoa2');

% Kalman filter
x_hat_kalman_1 = ekf(motion_model_1_1, y);
x_hat_kalman_2 = ekf(motion_model_2_1, y);

% Plotting Kalman estimates
figure(61)
xplot2(x_hat_kalman_1, y.y, 'conf', 90)
figure(62)
xplot2(x_hat_kalman_2, y.y, 'conf', 90)
```

## IX.9 task6b.m

```
% Measurement model
measurement_model_2 = s_setup_1_tdoa2;

% Motion models with measurement models
```

```
motion_model_1_2 = addsensor(motion_model_1, measurement_model_2);
motion_model_2_2 = addsensor(motion_model_2, measurement_model_2);

motion_model_2_2.x0 = [0 ; 0 ; 0 ; 0 ; 0];

% Kalman filter
x_hat_kalman_1_tdoa2 = ekf(motion_model_1_2, y_diff);
x_hat_kalman_2_tdoa2 = ekf(motion_model_2_2, y_diff);

% Plotting Kalman filter estimates
figure(63)
xplot2(x_hat_kalman_1_tdoa2, y_diff.y, 'conf', 90)
figure(64)
xplot2(x_hat_kalman_2_tdoa2, y_diff.y, 'conf', 90)
```

## IX.10 task7.m

```
% Motion models
motion_model_7 = exmotion('cv2d');

% Measurement model
measurement_model_7 = s_setup_1_tdoa2;
amp = 0.5;
measurement_model_7.th = measurement_model_7.th + amp*randn(8,1); % At 0.7
    the estimation fail

% Motion models with measurement models
motion_model_7 = addsensor(motion_model_7, measurement_model_7);

% Particle filter
x_hat_7 = ekf(motion_model_7, y_diff);

% Plotting particle filter estimates
figure(71)
xplot2(x_hat_7, y_diff.y, 'conf', 90)

% Plotting true and disturbed sensor position
figure(72)
plot(s_setup_1_tdoa2)
hold on
plot(motion_model_7.th(1), motion_model_7.th(2), 'd')
plot(motion_model_7.th(3), motion_model_7.th(4), 'd')
plot(motion_model_7.th(5), motion_model_7.th(6), 'd')
plot(motion_model_7.th(7), motion_model_7.th(8), 'd')
```

# Review

# Localization using a Microphone Network

Group 496's peer review on group 102's report

## I Result repeatability

The specific position of each sensor would be good to have in order to recreate a similar result to the one given in the report. Saying that they where placed around the trajectory of the target can be very relative even if there's a figure over the setup.

At the beginning there are four microphones but then it's eight. It can be seen as a description over each model but then this should be clarified.

## II Solutions to the tasks

The solutions to the tasks are all presented clearly with only a few small proposed additions.

### II.1 Task 1 – Sensor Calibration

It might be beneficial to add the formulas for the bias and the covariance in task 1 to the report. Other than that

### II.2 Task 2 – Signal Modeling

Each model was presented in a pedagogical and structured way. It was easy to understand how the equations was used and what each component in the where. The information about the chirp feels a bit out of place, consider moving it either to section **II Sensor Calibration** och **IV Experiments** since this is preprocessed and not something that is modeled.

### II.3 Task 3 – Experiments

There is a clear description of the configuration together with compatible figures.

### II.4 Task 4 – Configuration Analysis

Referring back to the earlier hypothesis is a nice way to draw conclusions about the configurations. It shows that you have thought it trough.

### II.5 Task 5 – Localization

Why the models is called "b" and "d" may not be understandable to all readers since it is a reference to the suggested methods in the compendium. Also, having these similar letters is a bit confusing. Consider renaming them to something more distinguishable and intuitive.

In the presentation of the result as well as the figure texts for figure 6 and 7, the comparisons are made between "model 1" and "model 2" instead of the methods "b" and "d". They are of course correlated but it seems as if though it is the localization methods that are being compared in the task, and therefore they should be referenced, and not the sensor models.

### II.6 Task 6 – Tracking

The results and comparisons are clear and easy to follow with a distinct conclusion.

### II.7 Task 7 – Discussion

The group has obviously done a lot of testing to find the limits of the model and clearly presents the results with data and figures.

## III Conclusions

All conclusions in the report follow clearly from the data and are easy to understand. It is hard to find rival conclusions.

## IV Strengths of the lab report

The report has a clear thread throughout the text and the actions taken are well grounded in the previous statements and conclusions. All conclusions emerge naturally from the results and the reasoning is easy to follow.

# V General improvements

The text is very well written but there are a few small improvements that can be made.

In the beginning of the text the tense changes between past and present, e.g. "We are interested in the position in space, so we multiplied the equations...". Custom would be to keep to the past tense all throughout the text.

Another language based improvement is that the authors refer to them selves as "we" several times which is unadvised in a report. The report is written in an active voice whereas most reports are written in a passive voice.

# Rebuttal

The position of the microphones will not be explained further since there are already figures describing their positions and the number of microphones used have been more thoroughly explained.

We did not feel the need to present the formulas for bias and variance since we assume that the reader has taken a course in statistics.

We moved the explanation of the chirp pulse to Section II as recommended.

Further more, we changed the names of the tracking methods from b and d to 1 and 2.

Lastly, we change the entire report to use the past tense and removed the use of "we" and "our".

# Statement of Work

We, Eric Moringe and David Wiman, hereby guarantee that all authors of this document have jointly and equally contributed to the execution of this lab and the writing of this report. Furthermore, all authors stand by the report as written and we are all able to explain its contents.