Problem 1: Coordinates and transformations

The two-dimensional pose of a robot with respect to a global coordinate frame is (x, y, θ) where (x, y) denotes its position in the plane and θ its orientation. The homogeneous transformation matrix that represents a pose with respect to the origin of the global coordinate system is

$$\begin{bmatrix} \cos\theta & -\sin\theta & x \\ \sin\theta & \cos\theta & y \\ 0 & 0 & 1 \end{bmatrix}$$

This matrix maps homogeneous coordinates relative to the pose (x, y, θ) to homogeneous global coordinates.

a) While at pose (x_1, y_1, θ_1), the robot senses a landmark ℓ at position (ℓ_x, ℓ_y) with respect to its local frame. Use a homogeneous transformation matrix to calculate the coordinates of the landmark with respect to the global frame.

b) Now imagine that you are given the landmark's coordinates (l_{gx}, l_{gy}) with respect to the global frame. Calculate the coordinates that the robot will sense in its local frame.

c) The robot moves to a new pose (x_2, y_2, θ_2) with respect to the global frame. Find the homogeneous transformation matrix that represents the new pose (x_2, y_2, θ_2) with respect to the previous pose (x_1, y_1, θ_1). In other words, the transformation matrix should map homogeneous coordinates with respect to the second pose to homogeneous coordinates with respect to the first pose. Hint: Write your answer as a product of homogeneous transformation matrices.

d) The robot is at pose (x_2, y_2, θ_2). Where is the landmark $\ell = (\ell_x, \ell_y)$ (in the robot's first frame) with respect to the robot's local frame now?

SOLUTION:

a) To find the coordinates of the landmark in the global frame, we first convert it to homogeneous coordinates with respect to the pose $\boldsymbol{x}_1$, multiply it by the transformation matrix that maps pose one coordinates to global coordinates, and then convert the homogeneous global coordinates back to standard coordinates. The coordinates of the landmark (ℓ_x^g, ℓ_y^g) with respect to the global frame are

$$\begin{bmatrix} \ell_x^g \\ \ell_y^g \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & x_1 \\ \sin\theta_1 & \cos\theta_1 & y_1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \ell_x \\ \ell_y \\ 1 \end{bmatrix}$$

b) Given the global landmark coordinates (ℓ_x^g, ℓ_y^g), we can invert the result from part (a) to obtain the landmark coordinates (ℓ_x, ℓ_y) with respect to the robot pose as

$$\begin{bmatrix} \ell_x \\ \ell_y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & x_1 \\ \sin\theta_1 & \cos\theta_1 & y_1 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} \ell_x^g \\ \ell_y^g \\ 1 \end{bmatrix}$$

c) Let A_1 and A_2 denote the transformation matrices corresponding to the first and second

robot poses, respectively. These matrices transform coordinates as follows:

$$A_1 : \text{pose } 1 \mapsto \text{global}$$
$$A_2 : \text{pose } 2 \mapsto \text{global}$$

To map homogeneous coordinates relative to pose two to homogeneous coordinates relative to pose one, we can use the product

$$A_1^{-1} A_2 : \text{pose } 2 \mapsto \text{pose } 1$$

d) To obtain the position of the landmark with respect to the second pose, we can use the transformation matrix $A_2^{-1} A_1$ that maps homogeneous coordinates relative to the first pose to homogeneous coordinates relative to the second pose (this is the inverse of the map in part (b)). Therefore, the coordinates (ℓ_x^2, ℓ_y^2) of the landmark with respect to the second pose is

$$\begin{bmatrix} \ell_x^2 \\ \ell_y^2 \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & x_2 \\ \sin\theta_2 & \cos\theta_2 & y_2 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & x_1 \\ \sin\theta_1 & \cos\theta_1 & y_1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \ell_x \\ \ell_y \\ 1 \end{bmatrix}$$

Problem 2: Sensing

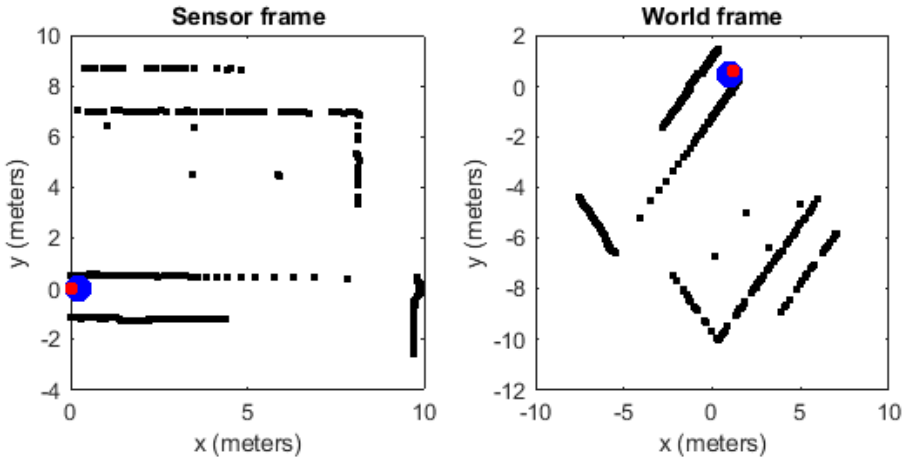
A robot is located at the pose $(x, y, \theta) = (1, 0.5, \pi/4)$. Its laser range finder is mounted on the robot with pose $(x_s, y_s, \theta_s) = (0.2, 0, \pi)$ with respect to the robot's frame of reference. The distance measurements of one laser scan can be found in the file `laserscan.dat`. The first distance measurement is taken at the angle $-\pi/2$ in the frame of reference of the laser range finder, the last distance measurement it taken at the angle $\pi/2$. In other words, the field of view of the sensor is π. All neighboring measurements are in equal angular distance.

a) Plot all laser endpoints in the frame of reference of the laser range finder.

b) The provided scan exhibits an unexpected property. Identify it and suggest an explanation.

c) Use homogeneous transformation matrices to compute and plot the center of the robot, the center of the laser range finder, and all laser endpoints in world coordinates.

SOLUTION: The plots below show the sensor measurements (black), robot position (blue), and sensor position (red) in the sensor frame and world frame. It appears as the laser can, at times, see through the walls, which should not be possible. This can indeed happen if the "wall" is actually a semitransparent obstacle, such as a grid, fence, or glass.




```
clear; clc;

% robot pose relative to the world frame
xr = [1; 0.5; pi/4];

% sensor pose relative to the robot frame
xs = [0.2; 0; pi];

% load the data
z = importdata('../data/laserscan.dat');

% number of measurements
m = length(z);
```

```
14
15 % angles of the measurement relative to the sensor
16 thetaz = linspace(-pi/2,pi/2,m);
17
18 % coordinates of the measurement relative to the sensor
19 xz = z .* cos(thetaz);
20 yz = z .* sin(thetaz);
21
22 % transformation from robot frame to world frame
23 A = [cos(xr(3)) -sin(xr(3)) xr(1); sin(xr(3)) cos(xr(3)) xr(2); 0 0
       1];
24
25 % transformation from sensor frame to robot frame
26 B = [cos(xs(3)) -sin(xs(3)) xs(1); sin(xs(3)) cos(xs(3)) xs(2); 0 0
       1];
27
28 % transform measurements to world frame
29 meas_world = A*B*[xz; yz; ones(1,m)];
30
31 % convert homogeneous coordinates to standard coordinates
32 xw = meas_world(1,:);
33 yw = meas_world(2,:);
34
35 % sensor coordinates in the world frame
36 s = A*[xs(1); xs(2); 1];
37 s = s(1:2)/s(3);
38
39 % robot coordinates in the sensor frame
40 r = inv(B)*inv(A)*[xr(1); xr(2); 1];
41 r = r(1:2)/r(3);
42
43 figure(1);
44
45 % plot the measurements, robot, and sensor in the sensor frame
46 subplot(1,2,1);
47 plot(xz,yz,'k.','MarkerSize',10); hold on;
48 plot(r(1),r(2),'b.','MarkerSize',40);
49 plot(0,0,'r.','MarkerSize',20);
50 xlabel('x (meters)');
51 ylabel('y (meters)');
52 title('Sensor frame');
53 axis square
54
55 % plot the measurements, robot, and sensor in the world frame
56 subplot(1,2,2);
57 plot(xw,yw,'k.','MarkerSize',10); hold on;
58 plot(xr(1),xr(2),'b.','MarkerSize',40);
59 plot(s(1),s(2),'r.','MarkerSize',20);
60 xlabel('x (meters)');
61 ylabel('y (meters)');
62 title('World frame');
63 axis square
```

Problem 3: Distance sensor

In this problem, you try to locate your friend using cell phone signals. Suppose that in the map of Oxford, the campus of the Miami University is located at $m_0 = (10, 8)$ and your friend's home is at $m_1 = (6, 3)$. You have access to the data received by two cell towers, which are located at the positions $x_0 = (12, 4)$ and $x_1 = (5, 7)$, respectively. The distance between your friend's cell phone and the towers can be computed from the intensities of your friend's cell phone signals. These distance measurements are disturbed by independent zero-mean Gaussian noise with variances $\sigma_0^2 = 1$ for the first tower and $\sigma_1^2 = 1.5$ for the second tower. You receive the distance measurements $d_0 = 3.9$ and $d_1 = 4.5$ from the two towers.

a) Is your friend more likely to be at home or at the university? Explain your calculations.

b) Implement a function that generates a 3D-plot of the likelihood $p(z \mid m)$ over all locations m in the vicinity of the towers. Mark m_0, m_1, x_0, and x_1 in the plot. Is the likelihood function that you plotted a probability density function? Explain your answer.

c) Now, suppose you have prior knowledge about your friend's habits which suggests that your friend currently is at home with probability $p(\text{home}) = 0.7$, at the university with $p(\text{university}) = 0.3$, and at any other place with $p(\text{other}) = 0$. Use this prior knowledge to recalculate the likelihoods in part (a).

Problem 4: Distance–bearing sensor

Assume you have a robot equipped with a sensor capable of measuring the distance and bearing to landmarks. The sensor furthermore provides you with the identity of the observed landmarks.

A sensor measurement $z = (z_r, z_\theta)$ is composed of the measured distance z_r and the measured bearing z_θ to the landmark ℓ. Both the range and the bearing measurements are subject to zero-mean Gaussian noise with variances σ_r^2 and σ_θ^2, respectively. The range and the bearing measurements are independent of each other.

A sensor model $p(z \mid x, \ell)$ models the probability of a measurement z of landmark ℓ observed by the robot from pose x. Design a sensor model for this type of sensor, and explain your model.

SOLUTION: We want to design a sensor model $p(z \mid x, \ell)$ to calculate the probability of a measurement z given a robot pose x and landmark pose ℓ. A sensor measurement $z = (z_r, z_\theta)$ consists of a distance z_r and angle z_θ measurement for the landmark. Both measurements are subject to Gaussian noise with variances σ_r^2 and σ_θ^2, respectively. The robot pose is given by $x = (x_x, x_y, x_\theta)$, the landmark position by $\ell = (\ell_x, \ell_y)$. Range and bearing measurements are independent, so

$$p(z \mid x, \ell) = p(z_r, z_\theta \mid x, \ell) = p(z_r \mid x, \ell)\, p(z_\theta \mid x, \ell)$$

To evaluate our sensor model, we need to evaluate the probability density of a normal distribution at the measurement z_i, with the expected measurement μ_i as the mean and the standard deviation σ_i^2.

$$p(z_i \mid x, \ell) = \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{(z_i - \mu_i)^2}{2\sigma_i^2}\right)$$

- **Range measurement.** For the range, the expected distance to the landmark is

  $$\mu_r = \sqrt{(\ell_x - x_x)^2 + (\ell_y - x_y)^2}$$

  and the variance is σ_r^2. Therefore, the probability of receiving a distance measurement z_r given the robot pose x and landmark position ℓ is

  $$p(z_r \mid x, \ell) = \frac{1}{\sqrt{2\pi\sigma_r^2}} \exp\left(-\frac{(z_r - \mu_r)^2}{2\sigma_r^2}\right)$$

- **Bearing measurement.** For the bearing, the expected angle to the landmark is

  $$\mu_\theta = \mathrm{atan2}(\ell_y - x_y, \ell_x - x_x) - x_\theta$$

  and the variance is σ_θ^2. We cannot simply subtract the expected and measured bearing because of the discontinuity of the angle representation between $(-\pi, \pi]$. The shortest angular difference between z_θ and μ_θ is

  $$|\Delta z_\theta| = \min\{|z_\theta - \mu_\theta|,\ 2\pi - |z_\theta - \mu_\theta|\}$$

  The probability of receiving a bearing measurement z_θ given the robot pose x and landmark position ℓ is

  $$p(z_\theta \mid x, \ell) = \frac{1}{\sqrt{2\pi\sigma_\theta^2}} \exp\left(-\frac{(z_\theta - \mu_\theta)^2}{2\sigma_\theta^2}\right)$$

Problem 5: Differential drive motion model

a) Write a function that implements the forward kinematics for a differential drive robot. The function should take as inputs the pose (x, y, θ) of the robot before applying the control action, the distances (d_l, d_r) traveled by the left and right wheels, and parameter d that is half the distance between the wheels. The output of the function should be the pose of the robot (x', y', θ') after applying the control action.

Hint: Make sure to take into account both straight-line motion and curved motion.

b) Write a function that implements the backward kinematics for a differential drive robot. The function should take as inputs the pose (x, y, θ) of the robot before applying the control action, the position of the robot (x', y') after applying the control action, and parameter d that is half the distance between the wheels. The output of the function should be the distances (d_l, d_r) traveled by the left and right wheels.

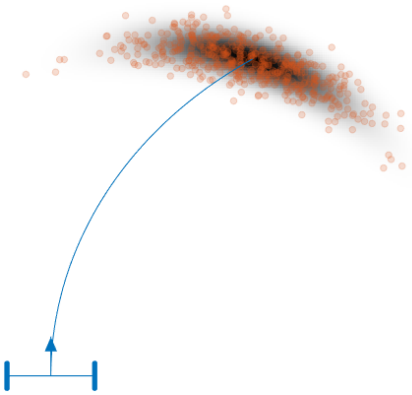
Hint: Make sure to take into account both straight-line motion and curved motion.

c) Now consider a differential drive robot with initial pose $(x, y, \theta) = (0, 0, \pi/2)$ and whose distance between the wheels is $2d = 1$. Suppose the actuation noise $(\varepsilon_l, \varepsilon_r)$ are independent Gaussian random variables with zero mean and standard deviations $\sigma_l = 0.3$ and $\sigma_r = 0.1$. Since we are only interested in the final robot position (and not orientation), we can ignore the final rotation. Suppose the robot measures the wheel distances $(d_l, d_r) = (3, 2)$.

Show all of the following on a single plot.

- Use your function from part (a) to plot 500 samples of future robot positions.
- Use your function from part (b) to plot the probability of the robot being at position (x', y') after executing the control action.
- Plot the initial pose of the robot.
- Plot the nominal trajectory of the robot, which is its motion without noise (this should be the arc of a circle).

Hint: Your plot should look similar to the following:



SOLUTION:

```
function [xp,yp,thp] = differential_drive_forward_model(x,y,th,dl,dr,d
    )

    if dl == dr
        % straight-line motion
        xp  = x + dl*cos(th);
        yp  = y + dl*sin(th);
        thp = th;
    else
        % circular motion
        R   = d*(dr+dl)/(dr-dl);
        dth = (dr-dl)/(2*d);
        thp = th + dth;
        xp  = x + R*(sin(thp) - sin(th));
        yp  = y - R*(cos(thp) - cos(th));
    end
end
```

```
function [dl,dr] = differential_drive_backward_model(xp,yp,x,y,th,d)

    % differences in each state coordinate
    dx = xp - x;
    dy = yp - y;

    if dy*cos(th) == dx*sin(th)

        % straight-line motion
        if sin(th) ~= 0
            dl = dy/sin(th);
            dr = dy/sin(th);
        else
            dl = dx/cos(th);
            dr = dx/cos(th);
        end
    else
        R   = 0.5*(dx^2+dy^2)/(dy*cos(th) - dx*sin(th));
        dth = 2*atan((dy-dx*tan(th/2))/(dx+dy*tan(th/2))) - th;

        dl = dth*(R - d);
        dr = dth*(R + d);
    end
end
```

```
clear; clc; close all;

% actuation noise standard deviations
sigmal  = 0.3;
sigmar  = 0.1;

% initial pose
```

```
8  x  = 0;
9  y  = 0;
10 th = pi/2;
11
12 % (half of the) distance between wheels
13 d = 0.5;
14
15 % wheel distances from odometry
16 dl = 3;
17 dr = 2;
18
19 % circle under deterministic actuation
20 R   = d*(dr+dl)/(dr-dl);
21 dth = (dr-dl)/(2*d);
22 xc  = x - R*sin(th);
23 yc  = y + R*cos(th);
24
25 % grid size
26 nx = 300;
27 ny = 300;
28
29 % grid
30 xp = linspace(-0.5,2.5,nx);
31 yp = linspace(-.5,3,ny);
32
33 % probability
34 p = zeros(nx,ny);
35
36 % samples
37 samples = zeros(2,500);
38
39 f = figure;
40
41 axis([min(xp) max(xp) min(yp) max(yp)]); hold on;
42 axis square;
43 set(gca,'XColor','none','YColor','none','color','none');
44
45 % sample future poses using the forward motion model
46 for i = 1:500
47
48     % sample the actuation noise
49     eps1 = normrnd(0,sigmal);
50     eps2 = normrnd(0,sigmar);
51
52     % perturbed wheel distances
53     dlhat = dl + eps1;
54     drhat = dr + eps2;
55
56     % next state using perturbed wheel distances
57     [sampled_xp, sampled_yp] = differential_drive_forward_model(x,y,th
           ,dlhat,drhat,d);
58
```

```
    samples(:,i) = [sampled_xp; sampled_yp];
end

% probability of future poses using the backward motion model
for i = 1:nx
    for j = 1:ny

        % perturbed odometry
        [dlhat, drhat] = differential_drive_backward_model(xp(i),yp(j)
            ,x,y,th,d);

        % actuation noise
        epsl = dlhat - dl;
        epsr = drhat - dr;

        % probability
        p(i,j) = normpdf(epsl,0,sigmal)*normpdf(epsr,0,sigmar);
    end
end

% normalize the density
p = p/sum(p(:));

imagesc(xp,yp,1-p'); colormap gray; axis xy;

% plot the sampled states
s = scatter(samples(1,:),samples(2,:),'filled','MarkerFaceColor','#
    D95319','MarkerEdgeColor','#D95319','MarkerFaceAlpha',0.2,'
    MarkerEdgeAlpha',0.2);
s.SizeData = 10;

% draw the prior robot pose
robot(x,y,th);

% nominal trajectory
phi = linspace(th,th+dth,1000);
plot(xc + R*sin(phi), yc-R*cos(phi), 'Color','#0072BD');

exportgraphics(f,'../figures/models.png');
```

Problem 6: Differential drive locomotion

A differential drive robot starts at pose $(x, y, \theta) = (1, 2, \pi/2)$. It has to move to the pose $(x', y', \theta') = (1.5, 2.0, \pi/2)$. The movement of the vehicle is described by the distances of the left and right wheels, d_l and d_r.

a) What is the minimal number of steering commands (d_l, d_r) needed to guide the vehicle to the desired target location?

b) What is the length of the shortest trajectory under this constraint?

c) Which sequence of steering commands guides the robot on the shortest trajectory to the desired location if an arbitrary number of steering commands can be used, and what is the length of this trajectory?

Problem 7: Odometry-based motion model

In this problem, you will implement an odometry-based motion model for a planar mobile robot.

a) Write a function that implements the odometry-based motion model. Your function should take the following three arguments:

$$x_t = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \qquad u_t = \begin{bmatrix} \delta_{r_1} \\ \delta_{r_2} \\ \delta_t \end{bmatrix} \qquad \alpha = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{bmatrix}$$

where x_t is the state of the robot before moving, u_t is the odometry reading obtained from the sensor measurements, and α are the noise parameters of the motion model. The actuation noise parameters are Gaussian random variables with the following distributions:

$$\begin{aligned} \varepsilon_{r_1} &\sim \mathcal{N}(0, \alpha_1 |\delta_{r_1}| + \alpha_2 \delta_t) \\ \varepsilon_{r_2} &\sim \mathcal{N}(0, \alpha_1 |\delta_{r_2}| + \alpha_2 \delta_t) \\ \varepsilon_t &\sim \mathcal{N}(0, \alpha_3 \delta_t + \alpha_4 (|\delta_{r_1}| + |\delta_{r_2}|)) \end{aligned}$$

The return value of the function should be the new pose x_{t+1} of the robot predicted by the model.

b) If you evaluate your motion model over and over again with the same starting position, odometry reading, and noise values, what is the result you would expect?

c) Evaluate your motion model 5000 times for the following values:

$$x_t = \begin{bmatrix} 2.0 \\ 4.0 \\ 0.0 \end{bmatrix} \qquad u_t = \begin{bmatrix} \pi/2 \\ 0.0 \\ 1.0 \end{bmatrix} \qquad \alpha = \begin{bmatrix} 0.1 \\ 0.1 \\ 0.01 \\ 0.01 \end{bmatrix}$$

Plot the resulting (x, y) positions for each of the 5000 evaluations in a single plot.

Problem 8: Velocity-based motion model

Consider a robot which moves on a circular trajectory with noise-free constant translational and rotational velocities, v and ω. The current pose of the robot is (x, y, θ).

a) Derive the following expression for the center of the circle (x_c, y_c).

$$\begin{bmatrix} x_c \\ y_c \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \frac{v}{\omega} \begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix}$$

b) Now consider the situation where we are given a start pose (x, y, θ) and an end pose (x', y', θ') connected by a circular movement. Prove that, for some real scalar μ, the center of the circle can be expressed as

$$\begin{bmatrix} x_c \\ y_c \end{bmatrix} = \frac{1}{2} \begin{bmatrix} x + x' \\ y + y' \end{bmatrix} + \mu \begin{bmatrix} y - y' \\ x' - x \end{bmatrix}$$

Hint: Consider the line ℓ that connects the center of the circle and the half-way point P between (x, y) and (x', y'). How are ℓ and P related to the two terms in the equation?

c) Show that the value of μ is given by

$$\mu = \frac{1}{2} \cdot \frac{(x - x')\cos\theta + (y - y')\sin\theta}{(y - y')\cos\theta - (x - x')\sin\theta}$$

Hint: μ can be calculated by using the fact that the line described in part (b) and the line from (x_c, y_c) to (x, y) intersect at (x_c, y_c).

Problem 9: Linear least squares

Consider a robot that uses a distance sensor to measure the relative distances to landmarks in both the x and y directions. The positions of the landmarks in the two-dimensional plane are

$$\ell_1 = \begin{bmatrix} 2 \\ 6 \end{bmatrix} \qquad \ell_2 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \qquad \ell_3 = \begin{bmatrix} 10 \\ 2 \end{bmatrix} \qquad \ell_4 = \begin{bmatrix} 7 \\ 8 \end{bmatrix}$$

The measured positions (relative to the robot) of the landmark locations are

$$z_1 = \begin{bmatrix} -0.57 \\ 2.39 \end{bmatrix} \qquad z_2 = \begin{bmatrix} -4.86 \\ -3.93 \end{bmatrix} \qquad z_3 = \begin{bmatrix} 5.17 \\ -1.91 \end{bmatrix} \qquad z_4 = \begin{bmatrix} 2.17 \\ 3.75 \end{bmatrix}$$

a) Find the least squares estimate of the robot position.

b) Now suppose we know that the covariance matrix for the noise measurements is given by the block-diagonal matrix

$$R = \text{diag}(R_1, R_2, R_3, R_4)$$

where

$$R_1 = \begin{bmatrix} 1 & 0 \\ 0 & 0.3 \end{bmatrix} \qquad R_2 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.5 \end{bmatrix} \qquad R_3 = \begin{bmatrix} 0.8 & 0 \\ 0 & 2 \end{bmatrix} \qquad R_4 = \begin{bmatrix} 0.5 & 0 \\ 0 & 1.5 \end{bmatrix}$$

Find the weighted least squares estimate of the robot position and its covariance matrix. Also, find the covariance matrix of the non-weighted least squares estimate from part (a).

c) On a single figure, plot the landmarks and both estimates of the position. Also, draw the probability ellipse corresponding to a 50% confidence level around each estimate using the MATLAB function `drawprobellipse(xhat,C,0.5)`, where `xhat` is the estimate and `C` is its covariance matrix. Which estimate (weighted or not) is the better estimate, and why?

SOLUTION:

a) Let (x, y) denote the (unknown) position of the robot. The measurement model for the j^{th} landmark is then

$$\begin{bmatrix} z_j^x \\ z_j^y \end{bmatrix} = \begin{bmatrix} \ell_j^x - x + r_j^x \\ \ell_j^y - y + r_j^y \end{bmatrix}$$

where (r_j^x, r_j^y) is the measurement noise. Collecting all of the measurements into matrix form and rearranging, we have that

$$0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ \vdots & \vdots \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} \ell_1^x - z_1^x \\ \ell_1^y - z_1^y \\ \vdots \\ \ell_4^x - z_4^x \\ \ell_4^y - z_4^y \end{bmatrix} + \begin{bmatrix} r_1^x \\ r_1^y \\ \vdots \\ r_4^x \\ r_4^y \end{bmatrix}$$

This is in the form of a general linear least squares problem $0 = Ax - b + r$, which has the solution

$$\hat{x}_{\text{LS}} = (A^{\mathsf{T}} A)^{-1} A^{\mathsf{T}} b = \begin{bmatrix} 4.2725 \\ 3.9250 \end{bmatrix}$$

b) Given the noise covariance matrices, we can now compute the covariance matrix associated with the least squares estimate from part (a), which is given by

$$C_{\text{LS}} = (A^{\mathsf{T}}A)^{-1}A^{\mathsf{T}}RA(A^{\mathsf{T}}A)^{-\mathsf{T}} = \begin{bmatrix} 0.1500 & 0 \\ 0 & 0.2688 \end{bmatrix}$$

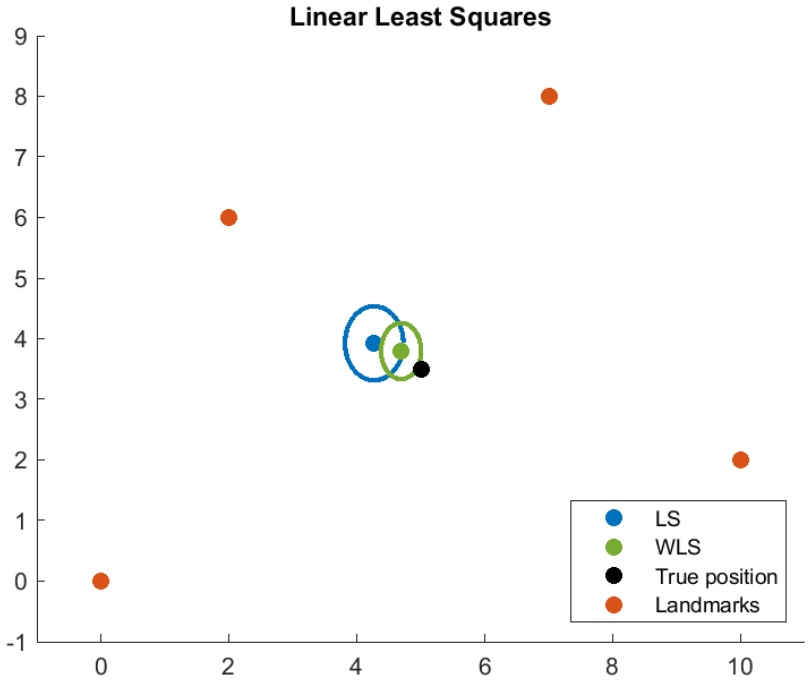
Furthermore, the weighted least squares estimate and its associated covariance matrix are given by

$$\hat{x}_{\text{WLS}} = (A^{\mathsf{T}}R^{-1}A)^{-1}A^{\mathsf{T}}R^{-1}b = \begin{bmatrix} 4.6925 \\ 3.7972 \end{bmatrix}$$

and

$$C_{\text{WLS}} = (A^{\mathsf{T}}R^{-1}A)^{-1} = \begin{bmatrix} 0.0702 & 0 \\ 0 & 0.1538 \end{bmatrix}$$

c) The following figure plots the landmarks and both LS and WLS estimates with their associated probability ellipses. Also shown is the true position $(5, 3.5)$ of the robot. The code that produces the plot is given below. Weighted least squares produces a better estimate than standard LS since it takes into account the accuracy of each sensor. We can see this in the analysis since the covariance matrix for WLS is smaller than that of LS. In the plot, the probability ellipse for WLS is smaller than that of LS indicating that we are more confident of our estimate from WLS.




```
clear; clc; close all;

% landmark positions
landmarks(1).position = [ 2; 6];
landmarks(2).position = [ 0; 0];
```

```
landmarks(3).position = [10; 2];
landmarks(4).position = [ 7; 8];

% landmark covariances
landmarks(1).covariance = [1 0; 0 0.3];
landmarks(2).covariance = [0.1 0; 0 0.5];
landmarks(3).covariance = [0.8 0; 0 2];
landmarks(4).covariance = [0.5 0; 0 1.5];

% number of landmarks
m = length(landmarks);

% true position (used only for plotting)
x = [5; 3.5];

% measurements
z = [-0.57; 2.39; -4.86; -3.93; 5.17; -1.91; 2.17; 3.75];

% linear least squares: minimize |Ax-b|^2
A = repmat(eye(2),m,1);
l = reshape([landmarks(:).position],2*m,1);
b = l - z;
R = blkdiag(landmarks(:).covariance);

% pseudoinverse of A
pinvA  = (A'*A)\A';

% position estimate
x_LS  = pinvA*b;
x_WLS = (A'/R*A)\A'/R*b;

% covariance
C_LS  = pinvA*R*pinvA';
C_WLS = inv(A'/R*A);

% confidence level for probability ellipses
alpha = 0.5;

figure; hold on;

% plot the LS estimate and its probability ellipse
plot(x_LS(1),x_LS(2),'.','Color','#0072BD','MarkerSize',25);
drawprobellipse(x_LS,C_LS,alpha,'#0072BD');

% plot the WLS estimate and its probability ellipse
plot(x_WLS(1),x_WLS(2),'.','Color','#77AC30','MarkerSize',25);
drawprobellipse(x_WLS,C_WLS,alpha,'#77AC30');

% plot the true position
plot(x(1),x(2),'k.','MarkerSize',25);

% plot the landmarks
```

```
58 for i = 1:m
59     plot(landmarks(i).position(1),landmarks(i).position(2),'.','
           Color','#D95319','MarkerSize',25);
60 end
61
62 title('Linear Least Squares');
63
64 xlim([min(l(1:2:end))-1,max(l(1:2:end))+1]);
65 ylim([min(l(2:2:end))-1,max(l(2:2:end))+1]);
66
67 legend('LS','','WLS','','True position','Landmarks','Location','
       Southeast');
```

Problem 10: Nonlinear least squares

Write a function that solves a given nonlinear least squares problem. Your function should take as inputs the residual function r, its Jacobian function J, an initial condition x_0, and a tolerance ε. The output should be an approximate solution to the nonlinear least squares problem.

Now consider a robot that uses a distance sensor to measure the range to landmarks. The positions of the landmarks in the two-dimensional plane are

$$\ell_1 = \begin{bmatrix} 1.5 \\ 1.7 \end{bmatrix} \qquad \ell_2 = \begin{bmatrix} 1.5 \\ 2.0 \end{bmatrix} \qquad \ell_3 = \begin{bmatrix} 1.9 \\ 2.5 \end{bmatrix} \qquad \ell_4 = \begin{bmatrix} 2.1 \\ 1.8 \end{bmatrix} \qquad z_5 = \begin{bmatrix} 2.5 \\ 1.7 \end{bmatrix}$$

The measured range (relative to the robot) of the landmark locations are

$$z_1 = 0.86 \qquad z_2 = 1.12 \qquad z_3 = 1.75 \qquad z_4 = 1.36 \qquad z_5 = 1.66$$

Use your function to estimate the position (x, y) of the robot given the range measurements to the known landmark locations.

- You will need to write functions to compute the residual and its Jacobian.
- You may use the Gauss–Newton algorithm and/or Levenberg–Marquardt algorithm.
- You may need to use a scheme to adaptively choose the step size.

SOLUTION: The following MATLAB function implements the Levenberg–Marquardt algorithm with an adaptive stepsize for the nonlinear least squares problem. For computational efficiency, the linear system is solved using the backslash operator \, not matrix inversion `inv()`.

```
function [x,trajectory] = LeastSquares(r,J,W,x0,tol,alpha)

% initialization
x = x0;
err = 2*tol;
lambda = 1;

trajectory = {x0};

% repeat until the relative error is less than the tolerance
while err > tol

    % Jacobian of the residual evaluated at the current iterate
    Jk = J(x);

    % residual evaluated at the current iterate
    rk = r(x);

    % store Jk'*Jk so that we only compute it once
    JJ = Jk'*W*Jk;

    % linear system
```

```
    A = JJ + lambda*diag(diag(JJ));
    b = Jk'*W*rk;

    % solve the linear system of equations
    dx = A\b;

    % update the stepsize parameter
    if norm(r(x-dx)) < norm(r(x))
        lambda = alpha*lambda;

        % update the iterate
        x = x - dx;
    else
        lambda = lambda/alpha;
    end

    % compute the error
    err = norm(dx);

    % store the trajectory
    trajectory{end+1} = x;
end

end
```

The residual is

$$r(x,y) = \sum_j z_j - \left\| \begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} \ell_{j,x} \\ \ell_{j,y} \end{bmatrix} \right\|$$

and its Jacobian is

$$J(x,y) = \begin{bmatrix} \frac{\ell_{1,x} - x}{\sqrt{(\ell_{1,x}-x)^2 + (\ell_{1,y}-y)^2}} & \frac{\ell_{1,y} - y}{\sqrt{(\ell_{1,x}-x)^2 + (\ell_{1,y}-y)^2}} \\ \vdots & \vdots \\ \frac{\ell_{m,x} - x}{\sqrt{(\ell_{m,x}-x)^2 + (\ell_{m,y}-y)^2}} & \frac{\ell_{m,y} - y}{\sqrt{(\ell_{m,x}-x)^2 + (\ell_{m,y}-y)^2}} \end{bmatrix}$$

Problem 11: Bayes rule

Suppose you are a witness to a nighttime hit-and-run accident involving a taxi in Cincinnati. All taxi cars in Cincinnati are blue or green. You swear under oath that the taxi was blue. Extensive testing shows that, under the dim lighting conditions, discrimination between blue and green is 75% reliable.

a) Given your statement as a witness and given that 9 out of 10 taxis in Cincinnati are green, what is the probability of the taxi being blue?

b) Is there a significant change if 7 out of 10 taxis in Cincinnati are green?

c) Suppose now that there is a second witness who swears that the taxi is green. Unfortunately he is color blind, so he has only a 50% chance of being right. How would this change the estimate from part (b)?

SOLUTION:

a) Let the random variable x denote the actual color of the taxi, and let the random variable y denote the color observed by the witness. Both random variables are either green or blue. Our goal is to compute the probability that the taxi is blue given the prior distribution on taxis and the statement of the witness, that is, $p(x = \text{blue} \mid y = \text{blue})$.

Since discriminating between the two colors is 75% reliable, we have the following:

$$p(y = \text{green} \mid x = \text{green}) = 0.75 \qquad p(y = \text{blue} \mid x = \text{green}) = 0.25$$
$$p(y = \text{blue} \mid x = \text{blue}) = 0.75 \qquad p(y = \text{green} \mid x = \text{blue}) = 0.25$$

where the right two probabilities come from the fact that the total probability must be one. Since we know that 9 out of 10 taxis in Cincinnati are green, we have the prior

$$p(x = \text{green}) = 0.9 \quad \text{and} \quad p(x = \text{blue}) = 0.1$$

Applying Bayes rule, we have that

$$p(x = \text{blue} \mid y = \text{blue}) = \frac{p(y = \text{blue} \mid x = \text{blue})\, p(x = \text{blue})}{p(y = \text{blue})}$$

To compute the denominator, we can use the law of total probability,

$$\begin{aligned} &p(y = \text{blue}) \\ &= p(y = \text{blue} \mid x = \text{blue})\, p(x = \text{blue}) + p(y = \text{blue} \mid x = \text{green})\, p(x = \text{green}) \\ &= (0.75)(0.1) + (0.25)(0.9) = 0.3 \end{aligned}$$

Substituting all of the probabilities into Bayes rule give

$$p(x = \text{blue} \mid y = \text{blue}) = \frac{p(y = \text{blue} \mid x = \text{blue})\, p(x = \text{blue})}{p(y = \text{blue})} = \frac{(0.75)(0.1)}{0.3} = 0.25$$

Therefore, given the statement of the witness and the prior distribution of the taxi colors, there is a 25% probability that the taxi is blue (so the witness is most likely wrong).

b) If 7 out of 10 taxis are green, then the prior distribution changes to

$$p(x = \text{green}) = 0.7 \qquad \text{and} \qquad p(x = \text{blue}) = 0.3$$

Substituting these new values into Bayes rule, we have that

$$p(x = \text{blue} \mid y = \text{blue}) = \frac{(0.75)(0.3)}{(0.75)(0.3) + (0.25)(0.7)} = 0.5625$$

In contrast to part (a), it is now more likely that the witness is correct.

c) Since the second witness has only a 50% change of being right, we are not expecting to gain any additional information on the color of the taxi. We can show this as follows. Let the random variable z denote the color observed by the second witness, which is either green or blue. The probability that we want to compute is then $p(x = \text{blue} \mid y = \text{blue}, z = \text{green})$. From Bayes rule with multiple measurements, this is equal to

$$\eta\, p(y = \text{blue} \mid x = \text{blue})\, p(z = \text{green} \mid x = \text{blue})\, p(x = \text{blue})$$

for some constant η. The two cases are:

$$p(x = \text{blue} \mid y = \text{blue}, z = \text{green}) = \eta\, (0.75)(0.50)(0.30)$$
$$p(x = \text{green} \mid y = \text{blue}, z = \text{green}) = \eta\, (0.25)(0.50)(0.70)$$

Since these probabilities must sum to one, the normalizing constant is $\eta = 5$. The probability that the taxi is blue given the observations of both witnesses is then 56.25%, which is the same as in part (b). Therefore, the second witness did not provide any information about the color of the taxi.

Problem 12: Bayes filter

A vacuum cleaning robot is equipped with a cleaning unit to clean the floor along with a sensor to detect whether the floor is clean or dirty. Neither the cleaning unit nor the sensor are perfect. From previous experience you know that the robot succeeds in cleaning a dirty floor with a probability of

$$p(x' = \text{clean} \mid x = \text{dirty}, u = \text{vacuum-clean}) = 0.7$$

where x' is the state of the floor after having vacuum-cleaned, u is the control action, and x is the state of the floor before performing the action. The probability that the sensor indicates that the floor is clean although it is dirty is $p(z = \text{clean} \mid x = \text{dirty}) = 0.3$, and the probability that the sensor correctly detects a clean floor is given by $p(z = \text{clean} \mid x = \text{clean}) = 0.9$. Unfortunately, you have no knowledge about the current state of the floor. However, after cleaning the floor the sensor of the robot indicates that the floor is clean.

a) Compute the probability that the floor is still dirty after the robot has vacuum-cleaned it. Use an appropriate prior distribution and justify your choice.

b) Which prior gives you a lower bound for that probability?

SOLUTION:

a) We want to compute the probability $p(x' = \text{dirty} \mid z = \text{clean}, u = \text{vacuum-clean})$, which is the belief after applying the vacuum-clean control action and observing that the floor is clean. The prior belief $\text{bel}(x)$ is unknown. Since the total belief must sum to one, we have that

$$\text{bel}(x = \text{clean}) = q \qquad \text{and} \qquad \text{bel}(x = \text{dirty}) = 1 - q$$

for some $q \in [0, 1]$. From the actuation update rule of the Bayes filter, the belief of the state x' after applying the vacuum-clean action is

$$\text{bel}(x') = \sum_x p(x' \mid x, u)\, \text{bel}(x)$$

which has the two values

$$\begin{aligned}\text{bel}(x' = \text{clean}) = {} & p(x' = \text{clean} \mid x = \text{clean}, u = \text{vacuum-clean})\, \text{bel}(x = \text{clean}) \\ & + p(x' = \text{clean} \mid x = \text{dirty}, u = \text{vacuum-clean})\, \text{bel}(x = \text{dirty})\end{aligned}$$

and

$$\begin{aligned}\text{bel}(x' = \text{dirty}) = {} & p(x' = \text{dirty} \mid x = \text{clean}, u = \text{vacuum-clean})\, \text{bel}(x = \text{clean}) \\ & + p(x' = \text{dirty} \mid x = \text{dirty}, u = \text{vacuum-clean})\, \text{bel}(x = \text{dirty})\end{aligned}$$

Using the prior belief and assuming that a clean floor remains clean after vacuum-cleaning, these are given by

$$\begin{aligned}\text{bel}(x' = \text{clean}) &= q + 0.7\,(1 - q) \\ \text{bel}(x' = \text{dirty}) &= 0.3\,(1 - q)\end{aligned}$$

As a check, we can verify that these probabilities sum to one as they should.
We can now use perception update rule of the Bayes filter, which gives that the belief of the state x'' after measuring that the floor is clean is

$$\mathrm{bel}(x'') = \eta\, p(z \mid x'')\, \mathrm{bel}(x')$$

The two probabilities are

$$\mathrm{bel}(x'' = \mathrm{clean}) = \eta\, p(z = \mathrm{clean} \mid x'' = \mathrm{clean})\, \mathrm{bel}(x' = \mathrm{clean}) = \eta\,(0.9)\big(q + 0.7\,(1-q)\big)$$

and

$$\mathrm{bel}(x'' = \mathrm{dirty}) = \eta\, p(z = \mathrm{clean} \mid x'' = \mathrm{dirty})\, \mathrm{bel}(x' = \mathrm{dirty}) = \eta\,(0.3)\big(0.3\,(1-q)\big)$$

Since these probabilities must sum to one, the normalizing coefficient is $\eta = 50/(9(q+4))$. We can then find the probability that the floor is still dirty after the robot has vacuum-cleaned it and the sensor measured that the floor is clean by substituting in different priors.

- If we have no prior knowledge, then $q = 0.5$ and the probability that the floor is dirty is approximately 5.56%.
- If we know that the floor is dirty, then $q = 0$ and the probability that the floor is dirty is 12.5%.

b) Intuitively, we expect the best case prior to be that the floor is clean. Indeed, the choice $q = 1$ minimizes the probability that the floor is dirty, in which case there is zero probability that the floor is dirty after having applied the vacuum-clean action.

Problem 13: Counting model

A robot applies the so-called simple counting approach to build a grid map of a 1D environment consisting of the cells c_0, c_1, c_2, c_3. While standing in cell c_0, the robot integrates four measurements z_{t_0}, z_{t_1}, z_{t_2}, z_{t_3}. After integrating these measurements, the resulting belief of the robot with regards to the occupancy of the four cells is $b_0 = 0$, $b_1 = 1/4$, $b_2 = 2/3$, $b_3 = 1$. Given that the first three measurements are $z_{t_0} = 1$, $z_{t_1} = 2$, $z_{t_2} = 3$, compute the value of the last measurement z_{t_3}.

Problem 14: Occupancy mapping

A robot has to build an occupancy grid map consisting of cells $c_0, \ldots, c_n$ of a simple one-dimensional environment using a sequence of measurements from a range sensor. The robot is located at the left-most cell c_0, and it is equipped with a distance sensor oriented towards the last cell c_n.

Assume that every grid cell with a distance (based on its coordinate) smaller than the measured distance is occupied with probability $p = 0.3$, and every cell behind the measured distance is occupied with probability $p = 0.6$. Do not update the belief of any cell located more than 20 cm behind the measured distance. Assume a prior belief of $p = 0.5$.

The cell coordinates span from 0 to 200 cm (both endpoints included) in increments of 10 cm. The measurements (in cm) are as follows:

101	82	91	112	99	151	96	85	99	105

Write a program to calculate the resulting occupancy grid map using the inverse sensor model, and plot the belief of each cell as a function of its position.

Problem 15: Occupancy mapping

Prove that in the occupancy grid mapping framework the occupancy value of a grid cell $P(m_j \mid x_{1:t}, z_{1:t})$ is independent of the order in which the measurements are integrated.

Problem 16: Extended Kalman filter

The extended Kalman filter (EKF) is an implementation of the Bayes Filter in which the sensor and measurement noise are Gaussian and the sensor reading and motion update functions are nonlinear. In this problem, you will derive and implement the EKF for a differential drive robot with state (x, y, θ) operating in the two-dimensional plane with a range-only sensor and odometry readings.

a) The Bayes filter processes three probability density functions:

$$p(x_t \mid u_t, x_{t-1}) \qquad p(z_t \mid x_t) \qquad \text{bel}(x_t)$$

State the normal distributions of the EKF which correspond to these probabilities.

b) Briefly explain each component of the EKF, that is, μ_t, Σ_t, g, G_t, h, H_t, Q_t, R_t, K_t, and why they are needed. What are the similarities and differences between the Kalman filter and the EKF?

c) Find a closed-form expression for the Jacobian matrix G_t of the noise-free odometry-based motion model of a differential drive robot.

d) Write a function that implements the prediction step of the EKF using the Jacobian G_t from part (c). Your function should take as inputs the mean and covariance of the belief mean, the odometry, and the noise covariance matrix Q_t. The output of the function should be the updated belief mean and covariance.

e) Find a closed-form expression for the Jacobian matrix H_t of the noise-free measurement model of a range-only sensor.

f) Write a function that implements the correction step of the EKF using your Jacobian H_t from part (e). Your function should take as inputs the mean and covariance of the belief, the measurement, and the sensor covariance matrix R_t.