Exercise 3 and 4 – Matching observations to maps, Perception – Sensor fusion (Kalman filter)

1 Introduction

Robot navigation requires that a robot can localize itself. In exercise 2, we use dead reckoning and odometry for localization. Still, as we saw in Exercise 2, the errors in the dead reckoning grow without bounds, which means they are reliable over short distances only. Therefore the position of the Robot needs correction (calibration) relative to the environment in which the Robot is currently active.

The exercise consists of laser data collected by the Snowhite Robot, i.e. the three-wheeled Robot used in exercise 2.

1.1 Plotting the Snowhite (and the laser data at a given location)

To plot a schematic image, you can use the function `plot_threewheeled_laser()` (the two rear wheels, its coordinate axis, the front wheel, the heading axis of the front wheel, the position of the laser sensor and (optional) the actual sensor readings at the given pose).

The function is defined and works as follows:

```
plot_threewheeled_laser(X, wdr, wbr, wtr, SteerA, wdf, wtf, L, alfa,
beta, gamma, ANGLES, MEAS, view_meas)
```

For an explanation to the parameters, `X, wdr, wbr, wtr, SteerA, wdf, wtf` and `L`, see exercise 2. The new parameters are `alfa` = distance along with the robots x-axis between the robots centre position and the centre position of the sensor (660mm), `beta` = distance along with the robots y-axis between the robots centre position and the centre position of the sensor (0mm), `gamma` = the angular difference between the x-axis of the sensor and the x-axis of the Robot (-π/2 radians). ANGLES and MEAS are [Nx1] vectors containing the laser data (angles [radians] and distances [mm]), and `view_meas` is a boolean value saying whether or not to plot the laser data.

1.2 Difference between two angles

One issue that can generate errors that is quite hard to debug is the difference between two angles. Therefore, a function `AngDifference(A1, A2)` calculates the difference correctly. The input is A1 [radians] and A2 [radians]. A1 and A2 can be row-vectors [Nx1] or scalars [1x1]. The function keeps the answer within the boundary [-π:π]. There is a built in function in Matlab that called angdiff that produce the same result if `angdiff(A2, A1)`.

2 Matching laser data to a set of line segments (the Cox scan matching algorithm)

Use the Matlab script' `main.m`', which loads all laser data (angles and measured distances) together with the control input values (velocities and steering angles) and the reference model REF_MODEL consisting of a line segment. The script predicts the position based on the control inputs (plotted as blue dots in the figure), which are plotted together with the ground truth positions (plotted as black dots) and where the laser data are perceived (red circles).

2.1 Task 1

Run the script and make sure you know what it does? Also, make sure you understand how the laser works (e.g., plotting some laser scans)?

2.2 Task 2

Complete the script to predict the uncertainty of the dead reckoning position (which you did in exercise 2)?

2.3 Task 3

Write the function Cox_LineFit(..) (the Cox scan matching algorithm is described in the “Blanche - …” paper by Cox [1] and pseudocode is available in appendix A.) in a separate m-file so it can be called in the following way:

```
[dx,dy,da,C] = Cox_LineFit(angs, meas, [X(kk-1) Y(kk-1) A(kk-1)]',
LINEMODEL,[alfa beta gamma]);
```

2.4 Task 4

Call your function from the main script and update your position estimates using the match results (dx, dy, da and C)? Along the path, plot the errors and the estimated error covariances?

2.5 Task 5 - This is exercise 4

To see how the use of a Kalman filter affects the Robot's position estimates, you should simulate the position fixes, i.e. for the moment, not use the position fixes given by the scan matching algorithm (i.e. given by the function `Cox_LineFit()`)? To simulate position fixes, add a zero-mean Gaussian distributed noise (generated in Matlab by the function `randn`) to the ground truth positions in those positions where the matching occurs, according to Equations 1 – 2?

$$\begin{aligned}\hat{x}_{pf} &= x_{true}(i) + \varepsilon_x \\ \hat{y}_{pf} &= y_{true}(i) + \varepsilon_y \\ \hat{\theta}_{pf} &= \theta_{true}(i) + \varepsilon_\theta\end{aligned} \tag{1}$$

In Equation 1, the errors ε_x, ε_y, ε_Θ are uncorrelated. They have zero mean and Gaussian distributions with standard deviations σ_x, σ_y, σ_Θ, which gives the covariance matrix (representing the uncertainty in the position fix) according to Equation 2.

$$\Sigma_{pf}(i) = \begin{pmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_\theta^2 \end{pmatrix} \tag{2}$$

The Robot's position estimate $\hat{X}(i)$ and covariance matrix $\Sigma_{\hat{X}}(i)$, at the time *i*, are updated by the proper weighting of the position fix, i.e. utilizing a Kalman filter, according to Equations 3 and 4.

$$\hat{X}^+(i) = \Sigma_{pf}(i)\left(\Sigma_{pf}(i) + \Sigma_{\hat{X}}(i)\right)^{-1}\hat{X}^-(i) + \Sigma_{\hat{X}}(i)\left(\Sigma_{pf}(i) + \Sigma_{\hat{X}}(i)\right)^{-1}\hat{X}_{pf}(i) \tag{3}$$

$$\Sigma_{\hat{X}}^+(i) = \left(\Sigma_{pf}^{-1}(i) + \Sigma_{\hat{X}}^{-1}(i)\right)^{-1} \tag{4}$$

Run the entire sequence with encoder twice with different values of the error noise (i.e. vary the standard deviations (variances)), e.g. in the first run, use $\sigma_x = \sigma_y = 10$ mm and $\sigma_\Theta = 1°$? In the second run, instead of using $\sigma_x = \sigma_y = 100$ mm and $\sigma_\Theta = 3°$, do you see any difference in the estimated positions?

2.6 Task 6 - This is exercise 4

In previous tasks, we used the dead reckoning position or the position from the matching algorithm to update the Robot's pose. Use a Kalman filter to fuse the position estimate from the dead reckoning with positions estimates from the scan matching algorithm? It is the same as in chapter 2.5 (Task 5), but using the position fixes given by the scan matching algorithm (task 4) instead of simulating them.

3 The Report

You shall write a report using the guidelines in Appendix B, i.e. a paper format. Answer all questions, elaborate on figures, include all necessary equations with motivated assumptions. You should also interpret your results, i.e., handing in a plot without explanations is not enough.

4 References

[1] I. J. Cox, Blanche - An Experiment in Guidance and Navigation of an Autonomous Robot Vehicle, IEEE Transactions on Robotics and Automation, Volume 7, No. 2, pp. 193-204, April 1991. https://doi.org/10.1109/70.75902

[2] P. S. Maybeck, The Kalman Filter: An Introduction to Concepts, Autonomous Robot Vehicles, I. J. Cox and G. T. Wilfong, Eds., New York: Springer-Verlag. Kalman Filtering Book by Peter Maybeck

5 Appendix

5.1 Appendix A The Cox scan matching algorithm

For all details, refer to the Cox paper [1].

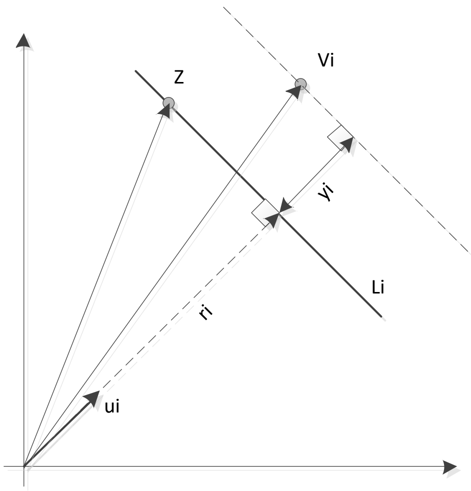
5.1.1 Step 0 – Calculate all unit vectors of all lines

- Calculate all unit vectors, u_i, off all lines, L_i, by calculation the normal to the line, moving it to origo and giving it unit length.
- Calculate distance to line (from origin)

  $r_i = u_i \bullet z$

  their z is any point on the line (select endpoint), and • is the dot-product. In Matlab, it will look like this:

```
ri = dot(ui,Li)
```



5.1.2 The loop

5.1.2.1 Step 1 – translate and rotate data points

- Sensor values to Cartesian coordinates (sensor coordinates)

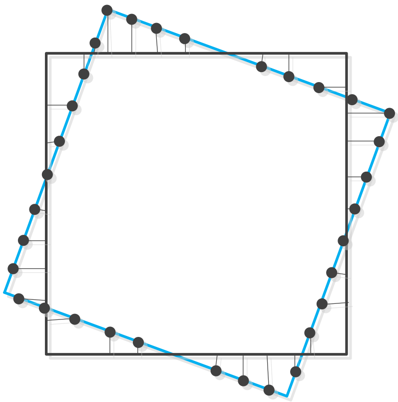
```
x = r*cos(alfa);
y = r*sin(alfa);
```

- Sensor coordinates to robot coordinates

```
R = [cos(Sa) -sin(Sa) Sx; sin(Sa) cos(Sa) Sy;0 0 1]
Xs = R*[x y 1]^T
```

- Robot coordinates to world coordinates

```
R = [cos(θ) -sin(θ) Xr(1); sin(θ) cos(θ)
Xr(2);0 0 1]
Xw = R*[Xs(1) Xs(2) 1]^T
```



5.1.2.2 Step 2 – Find the target for data points

Assign points to the line, i.e. points to closest lines. This means finding `min(yi)` for all lines L_i and assigning point v_i to line L_i. `yi = ri - dot(ui,vi)`

Reject all outliers!

5.1.2.3 Step 3 – Set up linear equation system

Set up the linear equation system and find `b = (dx, dy, da)` from least square. Vm are the position of the robot.

```
xi1 = ui1;
xi2 = ui2;
xi3 = ui*[0 -1;1 0]*(vi-vm);
A = [xi1 xi2 xi3];
B = inv(A^TA)A^T*Y
```

Calculate the variance

`S2 =(y-Ab)(y-Ab)/(n-4)` there `n = max(size(A));`

5.1.2.4 Step 4 - Add latest contribution to the overall congruence

B = [dx dy dθ]

```
ddx = ddx + dx;
ddy = ddy + dy;
ddθ = ddθ + dθ;
```

5.1.2.5 Step 5 – Check if the process has converged

Stop if the changes is below a threshold.

```
if (sqrt(B(1)^2 + B(2)^2) < 5 )&&(abs(B(3) <0.1*pi/180) ),
% stop loop
else
% return to Step 1
end
```

5.2 Exercise report guidelines

Guidelines for writing technical reports

Ola Bengtsson, Björn Åstrand

Abstract

You should summarize the most important results of this exercise and write them in the abstract. The abstract should also include a short description of the exercise's area and why this is important. The abstract should be 10-15 sentences long approximately.

1. Introduction

It would help if you described the area covered in the exercise. The description should be such that any person with appropriate technical knowledge should understand where and when to use the theory and methods described in this report. You might want to present and briefly describe work related to the work presented in the next section, Theory / Method. This means that you e.g. should refer to literature that gives a more profound presentation of the related work [Person and Person 2000]. More information can be found in [Person et al. 2001]. You should, e.g. add the limitations in the experiments and under what assumptions the experiments were done. The introduction part should be one half to one page long.

You need to writhe the report as a scientific report. Therefore, the introduction should include a short description of the rest of the report, i.e. a brief description of what follows in the report.

2. Theory / Method

The experiments should be replicable, which means you should describe them in such a way that any person with appropriate technical knowledge should be able to replicate your experiments by following your description in Sections 2 and 3. This means that Section 2 should include all necessary theories, e.g. mathematical equations, references to necessary papers, illustrative figures etc. (Sections 2 and 3 should be the central core of the report.)

It is crucial to get an exemplary structure of Section 2, why the use of sub-sections (i.e. Section 2.1, 2.2 and Section 2.2.1, 2.2.2 etc.) might be appropriate.

You must add all equations that you use in your code. For example:

- Exercise 1: transformation between degrees/minutes to meters, speed and heading calculation.
- Exercise 2: Error propagation law, kinematic models (for both robots and different uncertainty parameters)
- Exercise 3: Cox algorithm and transformation equations from sensor frame to global frame. Cox uncertainty.
- Exercise 4: EKF equations.

(Note: above is not a complete list)

3. Experiments/results

It would help if you described the conducted experiments so that any person with appropriate technical knowledge can repeat the experiments. It is essential to clearly explain the purpose of each experiment, show the results in illustrative ways and explain the results to the reader. In the case of many experiments, describe each experiment in a separate sub-section, i.e. Section 3.1, 3.2, 3.3, …, 3. N.

4. Conclusions and further improvements

It would be best if you made a clear conclusion on the exercise, what you have learnt etc. It is also appreciated if you write suggestions on improving the exercise for another year.

References

[Person and Person 2000] Person, O.; Person, T. (2000), Title of the paper that I am referring to, Which Journal or Proceedings the Paper was Published, In which town the paper was published, What year the paper was published, In what volume the paper was published, In which pages the paper was published.

[Person et al. 2001] Person, O.; Person, T.; Person, T. Person, F. (2001), Title of the paper that I am referring to, Which Journal or Proceedings the Paper was Published, In which town the paper was published, What year the paper was published, In what volume the paper was published, In which pages the paper was published.