Exercise 2 – Odometry, Dead Reckoning and Error Predictions

© Björn Åstrand, Ola Bengtsson

1 Introduction

A crucial thing for any mobile system is to know how the system responds to different inputs, e.g., how the vehicle's motion affects the state variables (x, y, Θ). To fully understand this, it is also crucial to know how the uncertainties of these variables change due to the same motion, i.e., how the wheel motion affects the state variables. Therefore, this exercise focuses on *odometry* (encoder values that are transformed into linear movements) and *dead reckoning* (angles and speed sampled with a specific time interval), which are then integrated during the time interval, giving a linear motion). The exercise also focuses on the uncertainty caused by the latest movement and how they are transformed to the state variables, i.e. error estimation/prediction.

The exercise consists of data from two robots, one with a differential drive (the Khepera mini-robot) and another with a driving wheel in the front of the vehicle (the Snowhite robot, i.e., a three-wheeled robot), see figure 1 and 2.

1.1 Plotting the uncertainty ellipses (in two dimensions)

To plot the uncertainty ellipses (1 standard deviation), you can use the function `plot_uncertainty()`, which is defined and works as follows:

```
function plot_uncertainty(X, C, dim1, dim2)
```

Where X is a 3x1 vector containing the means (i.e. the state variables (x, y, Θ) of the robot), C is the 3x3 covariance matrix of the state variables and dim1, and dim2 is the dimensions that should be plotted (typically if X = (x, y, Θ)^T. You want to plot the uncertainties in x and y then `dim1 = 1` and `dim2 = 2`). The function plots the ellipse in the figure that is currently active.

1.2 Calculating partial derivatives

You can calculate the partial derivatives by hand, but it's recommended to use a tool for that. In Matlab, you can use the function `jacobian()` to calculate partial derivatives:

```
syms x y z
```

```
jacobian([x*y*z,y^2,x + z],[x,y,z])
```

gives $\begin{pmatrix} yx & xz & xy \\ 0 & 2y & 0 \\ 1 & 0 & 1 \end{pmatrix}$

You can also use Wolfram Alpha [3] to calculate the derivatives. Try “d/dx x + vcos(a)Tcos(t+(vsin(a)T/2/L))”

2 Odometry with a differential drive robot

In this part of the exercise, we will use two different *odometry* based motion models to make the *error prediction* of the robot motion.

The first dataset (which you find in the file 'khepera_circle.dat' –write `ENC = load('khepera.txt');` in your Matlab script `ex2_khepera.m`) is collected using the Khepera mini robot and consists of the left and right encoder values. Khepera uses incremental encoders with 600 pulses/revolution. The wheel diameter, *D*, is 15.3mm and the wheelbase (distance between the two wheels), *B*, is 53mm. See Figure 1, which shows a schematic picture of the Khepera robot. Use the Matlab script '`khepera.m`' and fill in the missing parts in the text, i.e., parts looking like; `%MM_PER_PULSE = ??; %.`

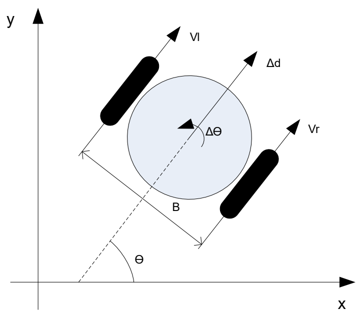

Figure 1 Khepera mini robot

2.1 Uncertainty parameters: forward direction (Δd) and heading (ΔΘ) of the robot

In this part of the exercise, the uncertainty parameters are, besides the state parameters, distance driven, *Δd*, and heading change, *Δθ*. The model for position update will look like this:

$$X_k = f(X_{k-1}, U_k) = X_{k-1} + \begin{bmatrix} \Delta d \cos\left(\theta_{k-1} + \frac{\Delta\theta}{2}\right) \\ \Delta d \sin\left(\theta_{k-1} + \frac{\Delta\theta}{2}\right) \\ \Delta\theta \end{bmatrix} = \begin{bmatrix} x_{k-1} + \Delta d \cos\left(\theta_{k-1} + \frac{\Delta\theta}{2}\right) \\ y_{k-1} + \Delta d \sin\left(\theta_{k-1} + \frac{\Delta\theta}{2}\right) \\ \theta_{k-1} + \Delta\theta \end{bmatrix} \quad (1)$$

To calculate the covariance, use the *law of error propagation,* i.e. the partial derivatives in square multiplied with the corresponding covariances:

$$\Sigma_{X_k} = J_{X_{k-1}} \Sigma_{X_{k-1}} J_{X_{k-1}}^T + J_{\Delta d\Delta\theta} \Sigma_{\Delta d\Delta\theta} J_{\Delta d\Delta\theta}^T \quad (2)$$

There

$$J_{X_{k-1}} = \begin{bmatrix} 1 & 0 & -\Delta d \sin(\theta_{k-1} + \frac{\Delta\theta}{2}) \\ 0 & 1 & \Delta d \cos(\theta_{k-1} + \frac{\Delta\theta}{2}) \\ 0 & 0 & 1 \end{bmatrix} \quad (3)$$

$$J_{\Delta d\Delta\theta} = \begin{bmatrix} \cos(\theta_{k-1} + \frac{\Delta\theta}{2}) & -\frac{\Delta d}{2} \sin(\theta_{k-1} + \frac{\Delta\theta}{2}) \\ \sin(\theta_{k-1} + \frac{\Delta\theta}{2}) & \frac{\Delta d}{2} \cos(\theta_{k-1} + \frac{\Delta\theta}{2}) \\ 0 & 1 \end{bmatrix} \quad (4)$$

$$\Sigma_{\Delta d\Delta\theta} = \begin{bmatrix} \sigma_{\Delta d}^2 & 0 \\ 0 & \sigma_{\Delta\theta}^2 \end{bmatrix} \quad (5)$$

From Wang paper [1] (section 4), we got (assume no correlation between wheels) that:

$$\sigma_{\Delta d}^2 = \frac{\sigma_r^2 + \sigma_l^2}{4} \quad (6)$$

$$\sigma_{\Delta\theta}^2 = \frac{\sigma_r^2+\sigma_l^2}{L^2} \tag{7}$$

2.1.1 Task 1

Calculate the changes in the forward direction (Δd) and heading (ΔΘ) of the robot, i.e., the changes expressed in the robot coordinate system caused by the latest movement. Also, calculate the variances (co-variances) of Δd and Δθ. What uncertainties do you assume?

2.1.2 Task 2

Calculate the new state variables of the robot under the assumption that the robot moves according to a circular trajectory (at this point, you could skip the compensation term given in Wang 1988 [1])? At the same time, calculate the covariance matrix of these new positions? You can assume that the robot always starts in origin and with a heading of 90º, i.e. (x0, y0, θ0) = (0, 0, 90*π/180).

2.1.3 Task 3

Run the entire sequence of encoder but also incorporate the compensation term given in the paper Wang [1]? Are the estimated state variables and covariance matrices the same? If you read the encoder values less often, i.e. if you read them 2, 5 or maybe ten times as seldom, what will happen to the state variables? When do the state variables start to differ – and why? (Hint: Check, e.g., the Δd and Δθ values).

2.1.4 Task 4

How are the covariance matrices evolving during the run (plot the entire run together with the calculated covariance matrices)? You don't have the true values of the state variables (a common problem in the mobile robot community) – but still, is the uncertainties realistic?

2.2 Uncertainty parameters: distance driven by the left and right wheel (Δr, Δl)

Here we got new uncertainty parameters, distance driven by the left wheel, Δl, and distance moved by the right wheel, Δr, and the wheelbase. The model for position update will look like this:

$$X_k = f(X_{k-1}, U_k) = X_{k-1} + \begin{bmatrix} \Delta d \cos\left(\theta_{k-1} + \frac{\Delta\theta}{2}\right) \\ \Delta d \sin\left(\theta_{k-1} + \frac{\Delta\theta}{2}\right) \\ \Delta\theta \end{bmatrix} = \begin{bmatrix} x_{k-1} + \frac{\Delta R+\Delta L}{2}\cos\left(\theta_{k-1} + \frac{\Delta R-\Delta L}{2b}\right) \\ y_{k-1} + \frac{\Delta R+\Delta L}{2}\sin\left(\theta_{k-1} + \frac{\Delta R-\Delta L}{2b}\right) \\ \theta_{k-1} + \frac{\Delta R-\Delta L}{b} \end{bmatrix} \tag{8}$$

And you use this model in tasks 5 and 6. To calculate the covariance, use the *law of error propagation,* i.e. the partial derivatives in square multiplied with the corresponding covariances:

$$\Sigma_{\mathrm{X_k}} = J_{X_{k-1}}\Sigma_{X_{k-1}}J_{X_{k-1}}^T + J_{\Delta r\Delta l}\Sigma_{\Delta r\Delta l}J_{\Delta r\Delta}^T + + J_b\Sigma_b J_b^T \tag{9}$$

There

$$J_{X_{k-1}} = \begin{bmatrix} 1 & 0 & -\Delta d \sin(\theta_{k-1} + \frac{\Delta\theta}{2}) \\ 0 & 1 & \Delta d \cos(\theta_{k-1} + \frac{\Delta\theta}{2}) \\ 0 & 0 & 1 \end{bmatrix} \tag{10}$$

$$J_{\Delta r\Delta l} = \mathit{See\ textbook\ page}\ 189\ (\mathit{equ}\ 5.11) \mathit{or\ the\ slides\ from\ lecture}\ 5$$

$$\Sigma_{\Delta r\Delta l} = \mathit{See\ textbook\ page}\ 189\ (\mathit{equ}\ 5.8) \mathit{or\ the\ slides\ from\ lecture}\ 5$$

$$J_b = \mathit{See\ Exercise}\ 3\ \mathit{and}\ 4\ \mathit{in\ the}\ \text{Error propagation}\ \mathit{material\ sent\ out}$$

$$\Sigma_b = \mathit{See\ Exercise}\ 3\ \mathit{and}\ 4\ \mathit{in\ the}\ \text{Error propagation}\ \mathit{material\ sent\ out}$$

2.2.1 Task 5

Once more, do the same run but use worse known values of the wheel diameter and the wheelbase, e.g., D = 14mm and WB = 45mm? How does it affect the estimated state variables? Plot the trajectories in the same plot as the other trajectories! What assumptions on these errors would you make?

2.2.2 Task 6

Repeat the experiment in task 5 but use the data file 'khepera.txt'? Plot the trajectories in the same plot as the other trajectories, i.e. different wheelbase and wheel diameter!

3 Dead-reckoning using a Three-Wheeled vehicle (Snowhite)

In this part of the exercise, we will use a *dead-reckoning* based motion model to make the *error prediction* of the robot motion.

The third dataset (which you find in the file 'snowhite.txt' –write `DATA = load('snowhite.txt');` in your Matlab script `ex2_Snowhite.m`) is collected using the Snowhite robot and contain the speed and angle of the steering/driving wheel and the ground truth positions in each time step (the sampling period is 50ms). The data file thus looks as follows:

v(0) α(0) $\text{True}_X(0)$ $\text{True}_Y(0)$ $\text{True}_\theta(0)$

v(1) α(1) $\text{True}_X(1)$ $\text{True}_Y(1)$ $\text{True}_\theta(1)$

…

v(n) α(n) $\text{True}_X(n)$ $\text{True}_Y(n)$ $\text{True}_\theta(n)$

The wheelbase (*L*, distance between the front and rear wheel axis) is 680mm. See Figure 2, which shows a schematic picture of the Snowhite robot. From the velocities {v(0), v(1), …, v(n)} it is possible to calculate the change in the robot's forward and rotational displacements (multiplying the velocities with the sample period, T, gives the movement of the robot), which become:

$$\Delta d = v \cdot \cos(\alpha) \cdot T \qquad (11)$$

$$\Delta\theta = \frac{v \cdot \sin(\alpha) \cdot T}{L} \qquad (12)$$

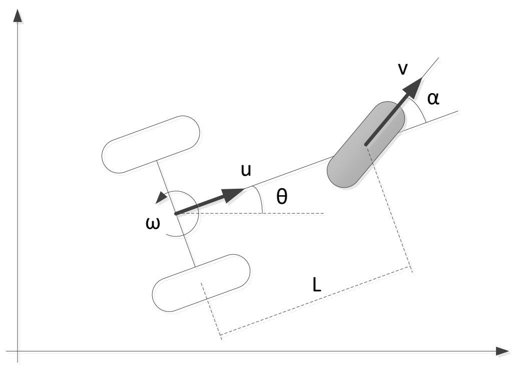

Figure 2 Schematic model of the Snowhite robot. The robot has a steering/driving wheel in the front and two passive fixed wheels in the rear.

We update the motion model since we use a steer-drive robot instead of a differential drive robot. The new uncertainty parameters are the steering angle, α, speed, V, and sampling time T.

$$X_k = f(X_{k-1}, U_k) = X_{k-1} + \begin{bmatrix} \Delta d \cos\left(\theta_{k-1} + \frac{\Delta\theta}{2}\right) \\ \Delta d \sin\left(\theta_{k-1} + \frac{\Delta\theta}{2}\right) \\ \Delta\theta \end{bmatrix} = \begin{bmatrix} x_{k-1} + v\cos(\alpha)\,\mathrm{T}\cos\left(\theta_{k-1} + \frac{v\sin(\alpha)T}{2L}\right) \\ y_{k-1} + v\cos(\alpha)\,T\,\sin\left(\theta_{k-1} + \frac{v\sin(\alpha)T}{2L}\right) \\ \theta_{k-1} + \frac{v\sin(\alpha)T}{L} \end{bmatrix} \quad (13)$$

To calculate the covariance, use the *law of error propagation,* i.e. the partial derivatives in square multiplied with the corresponding covariances:

$$\Sigma_{X_k} = J_{X_{k-1}} \Sigma_{X_{k-1}} J_{X_{k-1}}^T + J_{v\alpha T} \Sigma_{v\alpha T} J_{v\alpha T}^T \quad (14)$$

For guidance, see Exercise 4 in *Calculate Jacobian matrixes* available in Blackboard.

3.1 Task 7

Calculate the new state variables of the robot? Also, calculate the covariance matrix of these new positions? (To do this, you have to derive the Jacobian matrices w.r.t. the uncertain parameters. Follow the example given in the Wang paper [1] or chapter 5 in the textbook [2].) The robot always starts in the first ground truth position and with the first ground truth heading, i.e. $(x_0, y_0, \theta_0) = (\mathrm{True}_X(0), \mathrm{True}_Y(0), \mathrm{True}_\theta(0))$.

3.2 Task 8

Run the entire sequence of speeds and steering angles and compare the estimated state variables to the ground truth values. Also, compare the error in the state variable estimates, i.e., the difference between the estimated state variable and the ground truth state variables, to the standard deviations of the estimated variances (square root of the diagonal elements in the covariance matrix). What errors do you assume in the steering angle and the speed? Make sure the estimated standard deviations (the uncertainty of the estimated state variables) stay close to the error of the state variables.

3.3 Task 9

Don't forget to plot the error ellipses along the path taken by the robot? How long (distance and time) is the path?

4 The report

Answer all tasks 1-9 above, elaborate on figures, include ALL necessary equations (equation 1-2, 8-9, 13-14) with motivated assumptions (e.g. equations 9 and 14 need to have the Jacobian matrixes etc. similar to equation 2 and the corresponding equations 3-7). Use Equation Editor. You should also interpret your results, i.e., handing in a plot without explanations is not enough.

5 References

[1] C. M. Wang, Location Estimation and Uncertainty Analysis for Mobile Robots, IEEE International Conference on Robotics and Automation, 24-29 April 1988, pp. 1231-1235.

https://doi.org/10.1109/ROBOT.1988.12229

[2] Siegwart, R. & Nourbakhsh, I. R. (2004). Introduction to Autonomous Mobile Robots. Massachusetts: The MIT Press.

[3] d/dx x + vcos(a)Tcos(t+(vsin(a)T/2/L)) - Wolfram|Alpha (wolframalpha.com)