Sheet 6
Topic: Extended Kalman Filter II
Submission deadline: Fri 08.06.2007, 11:00 a.m. (before class)

Introduction

This exercise deals with the prediction step of the Extended Kalman Filter (EKF). A pose estimate in time step $t$ is represented by a Gaussian with mean $\mu_t$ and a covariance matrix $\Sigma_t$. These are the corresponding update equations:

$$
\mu_t := g(\mu_{t-1}, u_t) \\
\Sigma_t := G \Sigma G^T + VMV^T
$$

Exercise 1:

For the Kalman filter, you will need an implementation of a 3x3 matrix for Java. In the supplied source framework, you will find a class called CarmenMatrix. Complete the stubs for the methods\texttt{transpose, add} and \texttt{mult}. Test your functions by computing

$$
A = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}, \quad B = A^T, \quad C = A + A^T, \quad D = AA^T
$$

Add the corresponding lines to the method \texttt{RobotControl.TestMatrix()}. Verify the results by hand.

Exercise 2:

Complete the stub methods \texttt{StateJacobianG} and \texttt{MotionNoiseJacobianV}.

Note that this basically means that you have to implement the transformations you have calculated last week: from the accumulated odometry readings ($s_{t-1} = (x_{t-1}, y_{t-1}, \theta_{t-1})$ and $s_t = (x_t, y_t, \theta_t)$) to the control vector

$$
u_t = \begin{pmatrix} \delta_{rot1}^t \\ \delta_{trans}^t \\ \delta_{rot2}^t \end{pmatrix} = transform(s_{t-1}, s_t)
$$
as well as the state and motion noise Jacobians

\[ G_t = \frac{\partial g(\mu_{t-1}, u_t)}{\partial \mu_{t-1}} \quad \text{and} \quad V_t = \frac{\partial g(\mu_{t-1}, u_t)}{\partial u_t}. \]

**Exercise 3:**

Complete the method `VisualizationPanel.applyKalmanFilter`. This can be divided into the initialization of `Mu` and `Sigma`, and the actual prediction step

\[
\begin{align*}
\mu_t &:= g(\mu_{t-1}, u_t) \\
\Sigma_t &:= G\Sigma G^T + VMV^T
\end{align*}
\]

With the source code, you will find a log file called `sheet5.log`. It contains the two movements for which you already computed \( G, V, Mu \) and `Sigma` by hand. Verify that your program reports the same values (Hint: use `CarmenMatrix.Display()` to print the content of a matrix).

The correct values of last week were:

\[
\begin{align*}
\mu_0 &= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \quad \Sigma_0 = \begin{pmatrix} 0.1 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.1 \end{pmatrix} \\
\mu_1 &= \begin{pmatrix} 2.950 \\ 0.520 \\ 0.349 \end{pmatrix}, \quad \Sigma_1 = \begin{pmatrix} 0.372 & -0.122 & -0.056 \\ -0.122 & 1.044 & 0.317 \\ -0.056 & 0.317 & 0.138 \end{pmatrix} \\
\mu_2 &= \begin{pmatrix} 12.950 \\ 0.520 \\ -0.175 \end{pmatrix}, \quad \Sigma_2 = \begin{pmatrix} 0.622 & -0.682 & -0.056 \\ -0.682 & 21.963 & 1.774 \\ -0.056 & 1.774 & 0.176 \end{pmatrix}
\end{align*}
\]

**Exercise 4:**

Guess what initial state uncertainty \( \Sigma_0 \) and noise introduced by the motion \( M \) could have led to the following figures. Try to find values that produce a similar plot (use logfile `fr_sim2.log`). Write down your findings (per trajectory: a (possible) explanation of the observed expansion and a guess of the corresponding matrices).
Figure 1: Four Kalman trajectories, produced with different settings for the state and motion noise.