EE793 Target Tracking: Computer Exercise 1

Due: 12.03.2013, 23:59

1 Bayesian State Estimation

This first computer exercise aims that you implement three basic state estimation algorithms and observe the related quantities that was illustrated in class.

a) Download the true target data from the web address
`http://www.eee.metu.edu.tr/~umut/ee793/files/trueTarget.mat`.
When you load this mat file in Matlab, it is going to load into your workspace a 3×151 size matrix named trueTarget ($k = 0, \ldots, 150$). The first, second and third rows of this matrix contain the time stamps (i.e., the time values to which the data belongs), true x-positions x_k and true y-positions y_k of the target you will consider in the following parts. Plot the $\mathsf{x} - \mathsf{y}$ positions and observe the true target trajectory.

b) We now define our state vector as $x_k = \left[\begin{array}{cccc} \mathsf{x}_k & \mathsf{y}_k & \mathsf{v}_k^{\mathsf{x}} & \mathsf{v}_k^{\mathsf{y}} \end{array}\right]^{\mathrm{T}}$ where the variables $\mathsf{v}_k^{\mathsf{x}}$ and $\mathsf{v}_k^{\mathsf{y}}$ represent the velocity of the target along the x and y axes respectively. We assume that the target state evolves according to a nearly constant velocity model which is given below.

$$x_k = \underbrace{\left[\begin{array}{cc} I_2 & TI_2 \\ 0_2 & I_2 \end{array}\right]}_{\triangleq A} x_{k-1} + \underbrace{\left[\begin{array}{c} \frac{T^2}{2} I_2 \\ TI_2 \end{array}\right]}_{\triangleq B} w_k \tag{1}$$

where I_2 and 0_2 are identity and zero matrices of size 2×2; $T = 1$s is the sampling period and

$$w_k \sim \mathcal{N}(w_k; 0, I_2). \tag{2}$$

Suppose that $x_0 \sim \mathcal{N}(x_0; \bar{x}_0, P_0)$ where

$$\bar{x}_0 = \left[\begin{array}{cccc} 1000\text{m} & 1000\text{m} & 0\text{m/s} & 0\text{m/s} \end{array}\right]^{\mathrm{T}}, \tag{3}$$

$$P_0 = \operatorname{diag}\left[\begin{array}{cccc} 100^2\text{m}^2 & 100^2\text{m}^2 & 10^2\text{m}^2/\text{s}^2 & 10^2\text{m}^2/\text{s}^2 \end{array}\right]. \tag{4}$$

Consider that we obtain noisy measurements $\tilde{\mathsf{x}}_k$, $\tilde{\mathsf{y}}_k$ of x_k, y_k as follows.

$$z_k \triangleq \left[\begin{array}{c} \tilde{\mathsf{x}}_k \\ \tilde{\mathsf{y}}_k \end{array}\right] = \left[\begin{array}{c} \mathsf{x}_k \\ \mathsf{y}_k \end{array}\right] + \left[\begin{array}{c} v_k^{\mathsf{x}} \\ v_k^{\mathsf{y}} \end{array}\right] \tag{5}$$

where z_k is our measurement vector and

$$\left[\begin{array}{c} v_k^{\mathsf{x}} \\ v_k^{\mathsf{y}} \end{array}\right] \sim \mathcal{N}\left(\left[\begin{array}{c} v_k^{\mathsf{x}} \\ v_k^{\mathsf{y}} \end{array}\right]; \left[\begin{array}{c} 0 \\ 0 \end{array}\right], \left[\begin{array}{cc} \sigma_{\mathsf{x}}^2 & 0 \\ 0 & \sigma_{\mathsf{y}}^2 \end{array}\right]\right) \tag{6}$$

where $\sigma_{\mathsf{x}} = \sigma_{\mathsf{y}} = 100$m.

Now generate the noisy measurements of the target and observe the noisy measurements plotted on top of the true target positions you obtained in part (a).

Hint: In order to generate Gaussian random variables with specified mean μ and variance σ^2 in Matlab, you can use the command `randn(rowsize,columnsize)` which generates a matrix (with specified number of rows and columns) of zero-mean and unity variance random variables as follows:

$$>> v = \sigma\texttt{*randn(rowsize,columnsize)} + \mu; \tag{7}$$

c) Now, implement a Kalman filter using the models and the initial state parameters described in part (b). The Kalman filter is supposed to take the noisy measurements $\{z_k\}_{k=1}^{150}$ as input in order to obtain the estimates $\{\hat{x}_{k|k}\}_{k=1}^{150}$, predictions $\{\hat{x}_{k|k-1}\}_{k=1}^{150}$ and the measurement predictions $\{\hat{z}_{k|k-1}\}_{k=1}^{150}$. Plot the estimated target trajectory on top of the plot you obtained in part (b). Observe also the position estimation ($\hat{x}_{k|k}$) and prediction ($\hat{x}_{k|k-1}$) errors obtained for the Kalman filter i.e., plot

$$e_{k|k} \triangleq \sqrt{(\mathsf{x}_k - \hat{\mathsf{x}}_{k|k})^2 + (\mathsf{y}_k - \hat{\mathsf{y}}_{k|k})^2} \tag{8}$$

$$e_{k|k-1} \triangleq \sqrt{(\mathsf{x}_k - \hat{\mathsf{x}}_{k|k-1})^2 + (\mathsf{y}_k - \hat{\mathsf{y}}_{k|k-1})^2} \tag{9}$$

with respect to the discrete time index k. Which one of the errors $e_{k|k}$ or $e_{k|k-1}$ is larger most of the times, why? Calculate the RMS estimation and prediction errors given below.

$$\text{RMS}_{k|k} \triangleq \sqrt{\frac{1}{150}\sum_{k=1}^{150} e_{k|k}^2} \tag{10}$$

$$\text{RMS}_{k|k-1} \triangleq \sqrt{\frac{1}{150}\sum_{k=1}^{150} e_{k|k-1}^2} \tag{11}$$

Hint: When there is a gain matrix B multiplying the process noise as in (1), the prediction update for the covariance in the Kalman filter becomes $P_{k|k-1} = AP_{k-1|k-1}A^{\mathrm{T}} + BQB^{\mathrm{T}}$.

d) In this part, in order to gain more intuition about how a Kalman filter works, you are required to run your Kalman filter on exactly the same measurements that you obtained in part (b) but with different measurement noise and process noise covariances. In other words, you will use the Kalman filter under model mismatch.

- First fix the process noise covariance you use in Kalman filter in part (c) and then increase and decrease the measurement noise covariance you use in the Kalman filter 100 times (Note that the measurement noise covariance you use while generating your measurement noise should remain the same as in part (b) and only the covariance R you use in the filter should change.) Plot the estimated target trajectory, the true target trajectory and the measurements on the same plot for each case.
- This time fix the measurement noise covariance you use in Kalman filter in part (c) and then increase and decrease the process noise covariance you use in the Kalman filter 100 times. Plot the estimated target trajectory, the true target trajectory and the measurements on the same plot for each case.

Comment on your results.

e) Now assume that you have a sensor at the origin and you measure (instead of x and y positions) the range i.e., $r_k \triangleq \sqrt{\mathsf{x}_k^2 + \mathsf{y}_k^2}$ and bearing $\theta_k \triangleq \arctan(\mathsf{y}_k/\mathsf{x}_k)$ of the target with zero-mean Gaussian measurement noises with standard deviations $\sigma_r = 100$m and $\sigma_\theta = 5$degrees for the range and the bearing respectively. In other words,

$$z_k \triangleq \begin{bmatrix} \tilde{r}_k \\ \tilde{\theta}_k \end{bmatrix} = \begin{bmatrix} r_k \\ \theta_k \end{bmatrix} + \begin{bmatrix} v_k^r \\ v_k^\theta \end{bmatrix} \tag{12}$$

where z_k is our measurement vector and

$$\begin{bmatrix} v_k^r \\ v_k^\theta \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} v_k^r \\ v_k^\theta \end{bmatrix}; \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_\theta^2 \end{bmatrix}\right). \tag{13}$$

Generate your measurement data and observe the measured noisy range and bearing values of the target.

f) Now implement EKF and UKF which both use the same noisy measurements obtained in part (e) in order to estimate the state x_k. Note that since only the measurement equation is nonlinear, only measurement update requires a treatment different than Kalman filter (You can take $\pi^{(0)} = 1/9$ for UKF.). Observe the position estimation and prediction errors for the filters that you have implemented on the same plot. Calculate the RMS estimation and prediction errors and comment on the performance of the filters.