Robotics Project
Kalman Filter

Hoang Viet Nguyen, Lukas Leung, Zhuoming Tan

December 12, 2015

Instructor: Professor Ken Basye

1 Introduction

In this project we will use Kalman filter to estimate the motion path of an aircraft. We are writing our program in python.

1.1 Physics model

The following model defines the state estimation:

$$\hat{\mathbf{x}}_t = \mathbf{A}\hat{\mathbf{x}}_{t-1} + \mathbf{B}\mathbf{u}_t + \mathbf{w}_{t-1} \tag{1}$$

Which is

$$\begin{bmatrix} p_x^{(t)} \\ p_y^{(t)} \\ v_x^{(t)} \\ v_y^{(t)} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & t & 0 \\ 0 & 1 & 0 & t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\text{state transition}} \begin{bmatrix} p_x^{(t-1)} \\ p_y^{(t-1)} \\ v_x^{(t-1)} \\ v_y^{(t-1)} \end{bmatrix} + \underbrace{\begin{bmatrix} t^2/2 & 0 \\ 0 & t^2/2 \\ t & 0 \\ 0 & t \end{bmatrix}}_{\text{control matrix}} \begin{bmatrix} a_x^{(t-1)} \\ a_y^{(t-1)} \end{bmatrix} + \text{noise} \tag{2}$$

whose noise $\sim \mathcal{N}(0, \Sigma_x)$. The state transition matrix A and control matrix B are what we use in the program. They are only dependent on t, here representing the time step, so are constant matricies if we run the program

with a fixed time interval setting. And the following defines the observation from the state

$$\mathbf{z}_k = \mathbf{H}\hat{\mathbf{x}}_k + \text{noise} \tag{3}$$

which is

$$\begin{bmatrix} p_x^{(t)} \\ p_y^{(t)} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}}_{\text{estimation model}} \begin{bmatrix} p_x \\ p_y \\ v_x \\ v_y \end{bmatrix} + \text{noise} \tag{4}$$

whose noise $\sim \mathcal{N}(0, \Sigma_z)$. Because our estimation includes both position and velocity, but out measurement only has position, we have an measurement matrix C to translate our measurement to estimation.

1.2 Prediction

$$p_t = \mathbf{A}p_{t-1}\mathbf{A}^T + Q \tag{5}$$

where Q is the convarience matrix of E_x.

1.3 Measure

The Kalman gain would be

$$k_k = p_k\mathbf{H}^T\left(\mathbf{H}p_k\mathbf{H}^T + R\right)^{-1} \tag{6}$$

where R is the convarience matrix of E_z. The system noise covariance matrix:

$$Q = \begin{bmatrix} \sigma_1^2 & & \sigma_1\sigma_3 & \\ & \sigma_2^2 & & \sigma_2\sigma 4 \\ \sigma_3\sigma_1 & & \sigma_3^2 & \\ & \sigma_4\sigma_2 & & \sigma_4^2 \end{bmatrix} \tag{7}$$

has been erased off the zero elements because we have assumed x and y velocities are independent.

2 Program

2.1 Initial Conditions

The arguments we put into the program are explained below, with their default initial input.

$$\texttt{true\_initial\_state} = \begin{bmatrix} p_{x0} \\ p_{y0} \\ v_{x0} \\ v_{y0} \end{bmatrix} = \begin{bmatrix} 10 \\ 10 \\ 0 \\ 0 \end{bmatrix} \tag{8}$$

$$\texttt{initial\_estimation} = \begin{bmatrix} p_{x0} \\ p_{y0} \\ v_{x0} \\ v_{y0} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{9}$$

$$\texttt{acceleration} = \begin{bmatrix} a_{x0} \\ a_{y0} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{10}$$

$$(\texttt{int})\ \texttt{number\_of\_iters} = 10 \tag{11}$$

$$(\texttt{double})\ \texttt{delta\_t} = 1.0 \tag{12}$$

$$\texttt{sig\_acceleration} = \begin{bmatrix} \text{std\_dev}(a_x) & \text{std\_dev}(a_y) \end{bmatrix} = \begin{bmatrix} 0.1 & 0.1 \end{bmatrix} \tag{13}$$

$$\texttt{var\_obs} = \begin{bmatrix} \text{var}(\text{observed\_}P_x) & 0 \\ 0 & \text{var}(\text{observed\_}P_y) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{14}$$

2.2 Process

The program generates the noisy observations by adding random Gaussian noise to the true trajectory.

3 Conclusion

We have plotted within the program with `matplotlib`.