Kalman approach for position prediction in an undersampled tracking problem

Wernher von Braun

December 13, 2014

Abstract

Idea : Use a Kalman filter as a predictor, with the prediction loop running at a rate faster than the correction loop (aka update loop) so waypoints can be extrapolated from measurements even though we don't receive data.

1 Kalman Filter summary

1.1 Prediction steps

The first step is to predict the a priori state estimate

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k \hat{\mathbf{x}}_{k-1|k-1} \tag{1}$$

Where $\hat{\mathbf{x}}_{k|k-1}$ is the estimate of the state vector at iteration k, with iteration k not included. $\mathbf{F}_k$ is the state propagation matrix for iteration k and $\hat{\mathbf{x}}_{k-1|k-1}$ is the estimate of the state vector at iteration $k-1$, with iteration $k-1$ included. Notice that in our case, the command u is modeled as noise that's why it's absent from the equation.
The second step is to predict the a priori estimate covariance

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^{\mathrm{T}} + \mathbf{Q}_k \tag{2}$$

$\mathbf{P}_{k|k-1}$ is the parameter covariance matrix at iteration k, with iteration k not included. $\mathbf{Q}_k$ is the process covariance matrix (aka state propagation estimate error).

1.2 Update steps

In order to compute the optimal Kalman gain $\mathbf{K}_k$, we first need to compute the measurement residual $\tilde{\mathbf{y}}_k$ and the residual covariance $\mathbf{S}_k$

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k|k-1} \tag{3}$$

$$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k \tag{4}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_k^{-1} \tag{5}$$

Where $\mathbf{z}_k$ is the measurement. $\mathbf{H}_k$ is the design matrix that maps the state $\mathbf{x}$ into the measurement space and is defined such that $\mathbf{z} = \mathbf{H}\mathbf{x}$. $\mathbf{R}_k$ is the measurement covariance matrix
The last two equations can be compressed into a more compact equation

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k)^{-1} \tag{6}$$

Once we have the optimal Kalman gain, we correct (update) the estimates with measurement data. The updated a posteriori state estimate is therefore given by

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{y}}_k \tag{7}$$

And the updated a posteriori estimate covariance is given by

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1} \tag{8}$$

$\mathbf{I}$ being of course the identity matrix

1.3 Checklist

- Choice of state vector components $\mathbf{x}$
- Choice of measurements $\mathbf{z}$
- Description of measurement noise $\mathbf{R}$
- Choice of design matrix $\mathbf{H}$
- Choice of state propagation matrix $\mathbf{F}$ and its associated covariance matrix $\mathbf{Q}$
- Initial setup conditions

2 Analysis of the problem

2.1 State vector choice

By definition, the state of a deterministic dynamic system is the smallest vector that fully sums up the past of the system. Therefore, knowledge of the state should allow theoretically to predict the future outputs of the deterministic system.

Our quadrotor tracking problem can be simplified into a $2D$ position prediction problem. So we can take as state

$$\mathbf{x} = \begin{pmatrix} x \\ y \\ \theta \\ \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} \tag{9}$$

Where x, y, θ are the positions and angle that define an object in a $2D$ plane and $\dot{x}$, $\dot{y}$, $\dot{\theta}$ the associated velocities (linear and angular. Refering to classical mechanics equations, position and velocity should allow us to predict the system's behaviour.

2.2 Measurement considerations

The measurement z is considered to be the data received from the other quadrotor. This mainly contains positions, velocities and heading

$$\mathbf{z} = \begin{pmatrix} x \\ y \\ \theta \\ \dot{x} \\ \dot{y} \end{pmatrix} \tag{10}$$

To get the angular velocity $\dot{\theta}$ for the state, we would need to keep the precedent heading θ_{k-1}. So we can make the measurement vector

$$\mathbf{z_k} = \begin{pmatrix} x_k \\ y_k \\ \theta_k \\ \dot{x}_k \\ \dot{y}_k \\ \theta_{k-1} \end{pmatrix} \tag{11}$$

Then, state space and measurement space can be linked through a design matrix $\mathbf{H}$ such that $\mathbf{z} = \mathbf{H}\mathbf{x}$ with

$$\mathbf{H^{-1}} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{1}{\Delta t} & 0 & 0 & -\frac{1}{\Delta t} \end{pmatrix} \iff \mathbf{H} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -\Delta t \end{pmatrix} \tag{12}$$

With Δt the time step at which the loop is ran.
Otherwise, the error of the measurement could be modeled with a gaussian profile considering the GPS noise, but we can first suppose that the measurement is perfect and try to implement our Kalman predictor with no measurement noise. So we fix

$$\mathbf{R} = \mathbf{0} \tag{13}$$

2.3 State propagation

As we want to predict the position of the targeted quadrotor between two measurements of its actual position, we will consider a constant velocity model. Between two measurements, we have to solve a simple differential equation to extrapolate positions from the last measurement data, this can be done using the Euler method which yields

$$\begin{cases} x_k = x_{k-1} + \dot{x}_{k-1}\Delta t \\ y_k = y_{k-1} + \dot{y}_{k-1}\Delta t \\ \theta_k = \theta_{k-1} + \dot{\theta}_{k-1}\Delta t \end{cases} \tag{14}$$

Therefore, state propagation $\mathbf{F}$ would be

$$\mathbf{F} = \begin{pmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{15}$$

As we assumed a constant velocity model between measurements, a change in velocity causes errors. This can be taken into account through the state progragation covariance matrix $\mathbf{Q}$ that would depend on the maximal accelerations that the quadrotor is able to perform. Once again, using the Euler method, we can get the errors through linear integration of acceleration values. To fit $\sim 99.9\%$ of the case, we consider that the maximal acceleration cases cover 4σ, so

$$\begin{cases} 4\sigma_x = \frac{1}{2}\ddot{x}_{max}\Delta t^2 \\ 4\sigma_\theta = \frac{1}{2}\ddot{\theta}_{max}\Delta t^2 \\ 4\sigma_{\dot{x}} = \ddot{x}_{max}\Delta t \\ 4\sigma_{\dot{\theta}} = \ddot{\theta}_{max}\Delta t \end{cases} \Leftrightarrow \begin{cases} \sigma_x = \frac{1}{8}\ddot{x}_{max}\Delta t^2 \\ \sigma_\theta = \frac{1}{8}\ddot{\theta}_{max}\Delta t^2 \\ \sigma_{\dot{x}} = \frac{1}{4}\ddot{x}_{max}\Delta t \\ \sigma_{\dot{\theta}} = \frac{1}{4}\ddot{\theta}_{max}\Delta t \end{cases} \tag{16}$$

We suppose that $\ddot{x}_{max} = \ddot{y}_{max}$ so $\sigma_x = \sigma_y$ and $\sigma_{\dot{x}} = \sigma_{\dot{y}}$. The resulting state propagation error

matrix is given by

$$\begin{aligned}\mathbf{Q} &= \begin{pmatrix} \sigma_x & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_x & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_\theta & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{\dot{x}} & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{\dot{x}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{\dot{\theta}} \end{pmatrix} \\ &= \begin{pmatrix} \frac{1}{8}\ddot{x}_{max}\Delta t^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{8}\ddot{x}_{max}\Delta t^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{8}\ddot{\theta}_{max}\Delta t^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{4}\ddot{x}_{max}\Delta t & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{4}\ddot{x}_{max}\Delta t & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{4}\ddot{\theta}_{max}\Delta t \end{pmatrix} \end{aligned} \tag{17}$$

To set this, we only need two parameters: $\ddot{x}_{max}$ and $\ddot{\theta}_{max}$.

3 Implementation

```
initialisation;
while true do
    compute the a priori state estimate;
    compute the a priori estimate covariance;
    if new measurement taken then
        compute optimal Kalman gain;
        correct the a posteriori state estimate;
        correct the a posteriori estimate covariance;
    else
        go on;
    end
    return current state;
end
```

Algorithm 1: Pseudocode for Kalman predictor implementation

4 Results & Discussion

Appendices

A Notation index

- $\hat{\mathbf{x}}$ State vector estimate
- $\mathbf{F}$ State propagation matrix
- $\mathbf{Q}$ Process covariance matrix (State propagation error)
- $\mathbf{z}$ Measurement
- $\tilde{\mathbf{y}}$ Measurement residual
- $\mathbf{R}$ Measurement covariance matrix
- $\mathbf{K}$ Kalman gain
- $\mathbf{H}$ Design matrix
- $\mathbf{P}$ Parameter covariance matrix
- $\mathbf{S}$ Residual covariance matrix