RBE595 - Homework 1

Keith Chester

Due date: January 15, 2024

In this assignment we are provided with a proposed system of a drone tracked by a series of motion capture cameras. We are asked to formulate the system dynamics and Kalman Filter for this system, and then provided a set of measurements from the system for testing out implementation.

CSV files are provided that provide measurements - some in velocity, others direct position.

The system dynamics are given as:

$$\mathbf{x} = \begin{bmatrix} p \\ \dot{p} \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \tag{1}$$

...and inputs in the form of

$$\mathbf{u} = m\ddot{p} \tag{2}$$

...wherein m is our drone mass, at 27 grams. We are instructed to assume that all noise is additive Gaussian noise with 0 mean and a diagonal covariance matrix since the form of $\Sigma = \sigma^2 I$. The noise has $\sigma = 0.05$ m for low noise, $\sigma = 0.20$ m for high noise, and $\sigma = 0.05$ m/s for velocity noise.

Task 1

In this task we are asked to determine the matricies used in the process and measurement models, ignoring the noise matricies.

First we start with our state and known equations for our state and motion:

$$\mathbf{x} = \begin{bmatrix} p \\ \dot{p} \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \tag{3}$$

...and we know the state model for motion is:

$$\begin{aligned} x &= x + \frac{dx}{dt} + \frac{dx}{dt^2} \\ y &= y + \frac{dy}{dt} + \frac{dy}{dt^2} \\ z &= z + \frac{dz}{dt} + \frac{dz}{dt^2} \\ \dot{x} &= \dot{x} + \frac{d\dot{x}}{dt} \\ \dot{y} &= \dot{y} + \frac{d\dot{y}}{dt} \\ \dot{z} &= \dot{z} + \frac{d\dot{z}}{dt} \end{aligned} \tag{4}$$

...which we can also express in discrete terms:

$$\begin{aligned} x &= x + \dot{x}\Delta t + \frac{1}{2}\ddot{x}\Delta t^2 \\ y &= y + \dot{y}\Delta t + \frac{1}{2}\ddot{y}\Delta t^2 \\ z &= z + \dot{z}\Delta t + \frac{1}{2}\ddot{z}\Delta t^2 \\ \dot{x} &= \dot{x} + \ddot{x}\Delta t \\ \dot{y} &= \dot{y} + \ddot{y}\Delta t \\ \dot{z} &= \dot{z} + \ddot{z}\Delta t \end{aligned} \tag{5}$$

We wish to find the matricies in the following equation:

$$\dot{x} = \mathbf{F}\mathbf{x} + \mathbf{G}\mathbf{u} + E\eta \tag{6}$$

Since we are ignoring process noise, we can drop $E\eta$ from the equation. Solving for the **F** matrix:

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

...this is derived from an understanding of our state equations above.
For our **G** matrix, we first look at our **u** state equation:

$$\mathbf{u} = m\ddot{p} \tag{8}$$

...thus $\ddot{p}$ can be isolated:

$$\ddot{p} = \frac{\mathbf{u}}{m} \tag{9}$$

...so our G equation would be:

$$\mathbf{G} = \begin{bmatrix} \frac{\Delta t^2}{2} & 0 & 0 \\ 0 & \frac{\Delta t^2}{2} & 0 \\ 0 & 0 & \frac{\Delta t^2}{2} \\ \frac{\Delta t}{m} & 0 & 0 \\ 0 & \frac{\Delta t}{m} & 0 \\ 0 & 0 & \frac{\Delta t}{m} \end{bmatrix} \tag{10}$$

...thus leaving us with the following:

$$\ddot{x} = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} + \begin{bmatrix} \frac{\Delta t^2}{2} & 0 & 0 \\ 0 & \frac{\Delta t^2}{2} & 0 \\ 0 & 0 & \frac{\Delta t^2}{2} \\ \frac{\Delta t}{m} & 0 & 0 \\ 0 & \frac{\Delta t}{m} & 0 \\ 0 & 0 & \frac{\Delta t}{m} \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} \tag{11}$$

Now we need to solve for the covariance extrapolation equation, given by:

$$\mathbf{P}_{n+1,n} = \mathbf{F}\mathbf{P}_{n,n}\mathbf{F}^T + \mathbf{Q} \tag{12}$$

...wherein **Q** is the process noise covariance matrix, $\mathbf{P}_{n,n}$ is the covariance matrix at time n, and $\mathbf{P}_{n+1,n}$ is the covariance matrix at time $n+1$. **F** is the state transition matrix, which we have already solved for.

First, we must specify **P**:

$$\mathbf{P} = \begin{bmatrix} p_x & p_{xy} & p_{xz} & p_{x\dot{x}} & p_{x\dot{y}} & p_{x\dot{z}} \\ p_{yx} & p_y & p_{yz} & p_{y\dot{x}} & p_{y\dot{y}} & p_{y\dot{z}} \\ p_{zx} & p_{zy} & p_z & p_{z\dot{x}} & p_{z\dot{y}} & p_{z\dot{z}} \\ p_{\dot{x}x} & p_{\dot{x}y} & p_{\dot{x}z} & p_{\dot{x}} & p_{\dot{x}\dot{y}} & p_{\dot{x}\dot{z}} \\ p_{\dot{y}x} & p_{\dot{y}y} & p_{\dot{y}z} & p_{\dot{y}x} & p_{\dot{y}} & p_{\dot{y}\dot{z}} \\ p_{\dot{z}x} & p_{\dot{z}y} & p_{\dot{z}z} & p_{\dot{z}x} & p_{\dot{z}y} & p_{\dot{z}} \end{bmatrix} \tag{13}$$

...wherein:

- p_x is the variance of the position in the x direction
- p_y is the variance of the position in the y direction
- p_z is the variance of the position in the z direction

...for the initial use of the filter we will assume the $6x6$ identity matrix $\mathbf{I}$.
We then need to solve for the Q matrix. The Q matrix for our state is given by:

$$\mathbf{Q} = \begin{bmatrix} \sigma_x^2 & \sigma_{xy}^2 & \sigma_{xz}^2 & \sigma_{x\dot{x}}^2 & \sigma_{x\dot{y}}^2 & \sigma_{x\dot{z}}^2 \\ \sigma_{yx}^2 & \sigma_y^2 & \sigma_{yz}^2 & \sigma_{y\dot{x}}^2 & \sigma_{y\dot{y}}^2 & \sigma_{y\dot{z}}^2 \\ \sigma_{zx}^2 & \sigma_{zy}^2 & \sigma_z^2 & \sigma_{z\dot{x}}^2 & \sigma_{z\dot{y}}^2 & \sigma_{z\dot{z}}^2 \\ \sigma_{\dot{x}x}^2 & \sigma_{\dot{x}y}^2 & \sigma_{\dot{x}z}^2 & \sigma_{\dot{x}}^2 & \sigma_{\dot{x}\dot{y}}^2 & \sigma_{\dot{x}\dot{z}}^2 \\ \sigma_{\dot{y}x}^2 & \sigma_{\dot{y}y}^2 & \sigma_{\dot{y}z}^2 & \sigma_{\dot{y}x}^2 & \sigma_{\dot{y}}^2 & \sigma_{\dot{y}\dot{z}}^2 \\ \sigma_{\dot{z}x}^2 & \sigma_{\dot{z}y}^2 & \sigma_{\dot{z}z}^2 & \sigma_{\dot{z}x}^2 & \sigma_{\dot{z}y}^2 & \sigma_{\dot{z}}^2 \end{bmatrix} \tag{14}$$

...which, since there is no correlation between variables, we can 0 out most of this:

$$\mathbf{Q} = \begin{bmatrix} \sigma_x^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_y^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_z^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{\dot{x}}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{\dot{y}}^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{\dot{z}}^2 \end{bmatrix} \tag{15}$$

which would be $\sigma^2\mathbf{I}$.
We now look at our measurement equation:

$$\mathbf{z}_n = \mathbf{H}\mathbf{x}_n + \mathbf{v}_n \tag{16}$$

...wherein $\mathbf{z}_n$ is the measurement vector at time time n, $\mathbf{H}$ is the observation matrix, $\mathbf{x}_n$ is the state vector at time n, and $\mathbf{v}_n$ is the random noise vector at time n.

$$\mathbf{z}_n = \begin{bmatrix} x_{n,measured} \\ y_{n,measured} \end{bmatrix} = \mathbf{H} \begin{bmatrix} x_n \\ y_n \\ z_n \\ \dot{x}_n \\ \dot{y}_n \\ \dot{z}_n \end{bmatrix} \tag{17}$$

...where $\mathbf{H}$ changes based on whether we are measuring position or velocity. For position:

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{18}$$

...and for velocity:

$$\mathbf{H} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{19}$$

For $\mathbf{v}_n$ we will assume no noise, so $\mathbf{0}$ and thus ignore the term.

The Kalman Gain is given by:

$$K_n = P_{n,n-1}\mathbf{H}^T(\mathbf{H}P_{n,n-1}\mathbf{H}^T + \mathbf{R})^{-1} \tag{20}$$

For $\mathbf{R}_n$, we need to estimate the noise of the measuring sensor within reason. For the given sensor I chose a value of $1e^{-3}$ m, or 0.1 cm for σ thus $\sigma^2 = 1e-6$. Thus I model the $\mathbf{R}_n$ as:

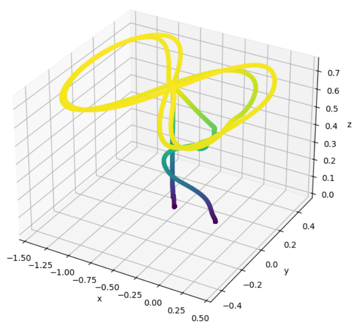
$$\mathbf{R} = \begin{bmatrix} 1e^{-6} & 0 & 0 \\ 0 & 1e^{-6} & 0 \\ 0 & 0 & 1e^{-6} \end{bmatrix} \tag{21}$$

...which is the last matrix needed to perform the operation.

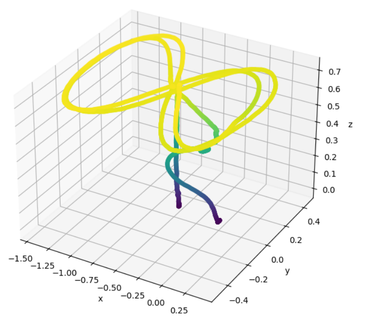
Task 2

In this section, we are tasked with reading in the aforementioned system's CSV files of recordings from the motion capture recordings. Some of these are straight readings of the position or the velocity of the drone; others introduce a low or high amount of noise.

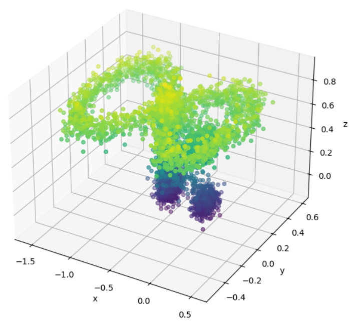
The implementation of our Kalman filter using the math from Task 1 is found in the attached **task2.py**; I recommend looking there for the coded implementation. Below are the produced graphs of the results of the Kalman filter on the provided data.



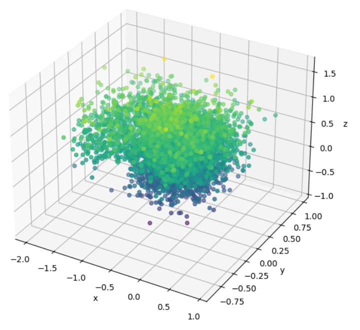
(a) Direct MoCap Position



(b) Direct MoCap Velocity



(c) Low Noise



(d) High Noise

Figure 1: Kalman Filter results

Here we see that as noise is added, the precise trajectory tracking becomes cloudier, with the Kalman filter attempting to simplify and smooth the trajectories.