**Estimator**:

Estimators are designed to predict each state in a system, if that state information is not directly observable. Often, measurements are taken that contain indirect information about a state. For example, often accelerometer measurements are used in navigation to take measurements of acceleration, which can then be integrated twice to get position over time. Naturally, there is some noise and inaccuracy in this process, but provided the system is fully observable, a Kalman filter can be implemented on a linear system to get observation of state variables from measurement data.

Kalman filters are a simple form of estimator that works on linear systems of the form:

We note that the Kalman filter assumes no user input related to the measurement. Because user input is fully known, we don’t wish to include it in our states that we are estimating. As a result, our Kalman Filter will not try to estimate acceleration, only position and velocity. Otherwise, the Kalman filter model is sufficient for this caravan problem.

A Kalman filter operates in two steps.

**A Priori Measurements:**

In the first, the kalman filter performs an ‘a priori’ estimate, through which it produces an estimate of the state at the current discrete time step based only on the previous time step:

Here, the P matrix is the covariance matrix of the system’s states, while Q is a measure of the system’s state noise. Discrete-time process noise can be modeled as follows:

Where T is the period between each measurement, here defined as 0.1s, and is the power spectral density (PSD) of noise for each component. For our system, we used relatively low process noise with a PSD of 1/10 for position, and 1/20 for velocity. We also note that the values of Fk and Gk are the discrete-time system dynamics, which are covered in another section of this paper, rather than continuous-time dynamics.

**A Posteriori Measurements:**

Next, the Kalman filter takes new measurements into the system and incorporates them. The Kalman filter creates a model for measurements and compares it to the actual measurement data it receives, and weights it’s a priori estimate with the new information to create a new, a posteriori measurement.

**Measurement:**

**A Posteriori Update:**

The A posteriori formula can take many forms, and the form above is known as the Joseph’s form. The reason it is used in lieu of other forms is that the Joseph’s form equation copes well with sparse transition matrices that may lead to low precision inversion of the Pzz term. As a result, the Joseph’s form works best for this system, which has some very sparse matrices.

The final result, xk, is the most accurate version of the system states based on both a model of how these states grow, and measurements that relate to those states.

**Different Measurements at Different Times:**

For our system, two measurements were taken: position of the first vehicle, and range between each vehicle in the caravan. The former was taken once a second, while the latter was taken ten times each second. As a result, we wish to combine both measurements into the Kalman filter as these data come. How do we accommodate this in our Kalman filter system? There are two ways to accommodate measurements taken at different times in a Kalman filter: The first is to change the measurement noise covariance matrix R such that it assumes an infinite potential variance on measurement data for measurements that aren’t present at this time step. The other method is to have a varied-length measurement input Z, and to change the size of the matrix Hk at each step depending on the size of the Z vector. This latter method was chosen for this Kalman filter, as it worked better for this implementation with the matrix sparcity as it was. In instances where the Z vector didn’t include the position measurement, the top row of the H vector was clipped off.

**Discussion:**

The Kalman filter was run both with and without control input to ensure that it was working. In both cases, the filter’s error between estimated state and true state was driven to zero within two minutes. A plot of the estimation error and variances of estimated state positions is given below.

As can be seen, the estimation error is driven to zero quickly, and remains relatively stable throughout the system. In a few rare instances, a noisy measurement of vehicle position bounced the error up for a second or two, but these errors were quickly accommodated by the Kalman filter.

The error of each state was interesting. The kalman filter trusted its estimate of each vehicle position in the caravan less than the vehicle in front of it. This makes sense, given that only the first vehicle has actual position measurements, and subsequent position estimates have more and more noise (driven by range values). This intuitive image of how the Kalman Filter was able to estimate vehicle positions and velocities helped to confirm that the Kalman filter was operating exactly as expected.