Track Filter Design: A MATLAB Implementation

Jacob Epifano, Michael Giuliano
Section 1

July 19, 2018

1 Introduction

1.1 Background

Track filtering, also referred to as state estimation, is "the process in which measurements from a single object are used to estimate the kinematic states of that object" [1]. Track filtering is an important component in a weapon system because every component, sensor or otherwise, that is contained in the weapon system suffers from some form of noise. Commonly, the cumulative noise of all components is experienced equally across the range of measurements, data, and operating conditions and appears random in nature. In others, a form of mechanical, electrical, or extraneous bias introduced into the system will cause the data to sway in a particular direction. Identification of all forms of error and selection of the appropriate filtering mechanism are critical to the success of a weapon control system due to the filter's ability to take inherent system error into account. The filter will serve as a means to more accurately process sensor data and give and return the data with a minimized error [1]. Such a powerful form of prediction has the potential to not only be applied to weapon control systems but in areas of finance, robotics, and more.

1.2 Kalman Filter

Kalman filtering is a recursive algorithm that allows for the optimization of "filter gains" over the course of three different steps. Prior to any calculation, the kinematic state is initialized according to the first measurement taken since the filter has no means of extrapolating from a single point. Once a second measurement is taken, a predictive algorithm estimates the future state. Next, gains are selected based upon an optimization criterion. In many respects, the gain is the crucial component in a well designed filter and determines how much mathematical weight to place on the current measurement. In the case of the Kalman filter, this means minimizing the summation

of the position and velocity covariance. Upon gain selection, the filter updates the current state using the newest measurement and the weighted average of the residual between the measured and predicted new states. Kalman says that given a set of optimized gains, the kinematic state can be predicted withing a prescribed range of uncertainty. The largest caveat with Kalman filtering is that it assumes that process noise contained in measurements are Gaussian in nature. If such an assumption is incorrectly made, a bias will be introduced in the filter estimates and cause the state error to not be contained with the target bounds of the covariance matrix.

The MATLAB filter was implemented over 500 Monte Carlo runs and used discrete white noise acceleration in the process noise matrix to describe the potential error. The main-loop implementation for the Kalman filter is shown below: [1]

Algorithm 1 Kalman Filter

for all k **do**
$x(k+1|k) \leftarrow \phi x(k)$
$meas(k+1|k) \leftarrow Hx(k+1) + n(k+1)$
$\nu(k+1|k) \leftarrow meas(k+1) - Hx(k+1|k)$
$P(k+1|k) \leftarrow \phi P(k|k)\phi' + Gw(k)G'$
$K \leftarrow P(k+1|k)H'Q^{-1}$
$P \leftarrow (I - KH)P(I - KH)' + KRK'$
$x(k|k) \leftarrow x(k+1|k) + K\nu$
end for

1.3 Optimal Reduced State Estimator (ORSE)

ORSE filters are similar to Kalman in that they are recursive optimization algorithms. They are differentiated by the fact that ORSE filters do not assume noise to be Gaussian but rather as deterministic. In this way, random and bias errors are calculated separately and together contribute to the overall target model error. ORSE optimizes the mean squared error of the estimate, taking into account position and velocity covariance as well as each of their biases. [2] Another main difference between ORSE and Kalman is the inclusion of the D, or lag, matrix. "D is the matrix that provides an estimate of the filter lag at the current time. When the state estimates are updated, D is adjusted to reflect the filter lag at that time," [2]. It obtains its values using the G matrix which describes the growth target model error over time and can be considered a normalized error due to acceleration. In practice, the filter gains for ORSE are greater in numeric value for a given model error because it must account for a consistent error.

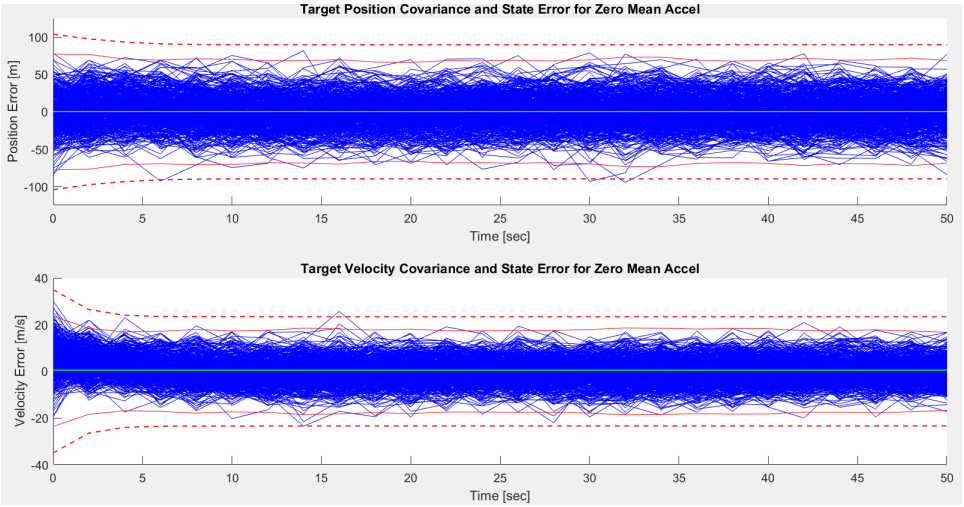
The MATLAB implementation includes calculations for the random process error matrix M, bias and filter lag matrix D, and the covariance matrix S that uses D and M: all of which are distinct features of ORSE. Lambda was chosen to represent the maximum acceleration of the target and is a fixed one dimensional constant. The main-loop implementation for the ORSE 1D filter is shown below:

Algorithm 2 ORSE Filter

for all k **do**
$x(k+1|k) \leftarrow \phi x(k)$
$meas(k+1|k) \leftarrow Hx(k+1) + n(k+1)$
$\nu(k+1|k) \leftarrow meas(k+1) - Hx(k+1|k)$
$M \leftarrow \phi M \phi'$
$D \leftarrow \phi D + G$
$S \leftarrow M + D\lambda^2 D'$
$K \leftarrow SH'(HSH' + R)^{-}1$
$x(k|k) \leftarrow x(k+1|k) + K\nu$
$M \leftarrow (I - KH)M(I - KH)' + KRK'$
$D \leftarrow (I - KH)D$
end for

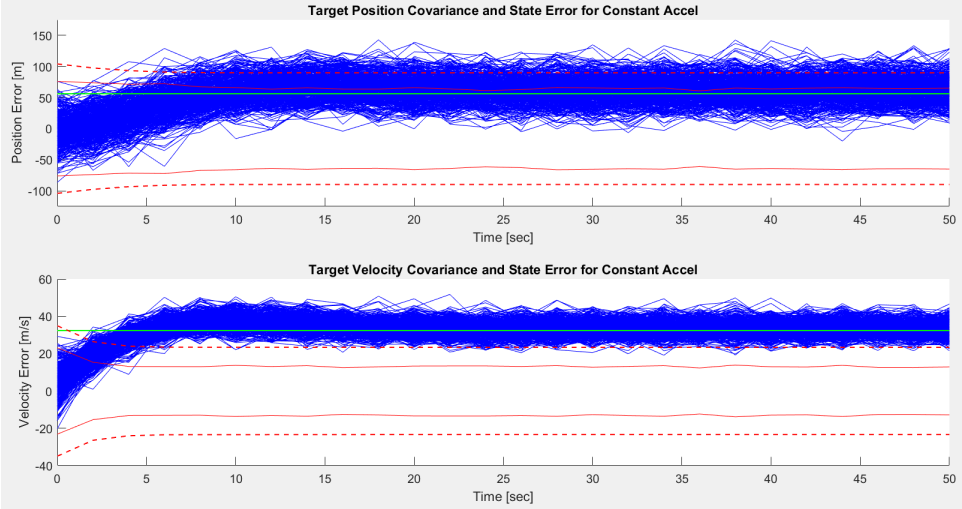
2 Results and Discussion

2.1 Kalman: Zero Mean Acceleration



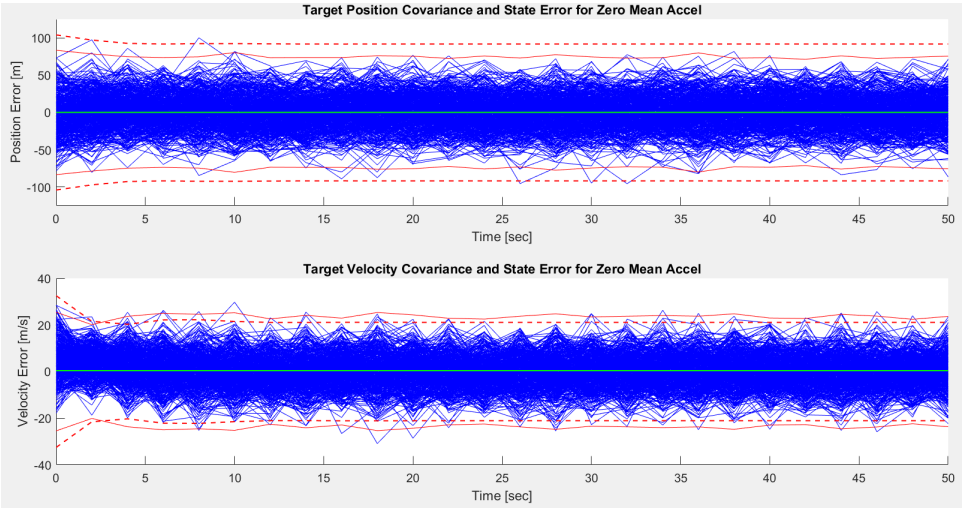
The estimation errors are contained within the 3 standard deviation bounds of the covariance matrix. This behavior is expected because the generated measurement noise fits Kalman's assumption of being Gaussian in nature.

2.2 Kalman: Constant Acceleration



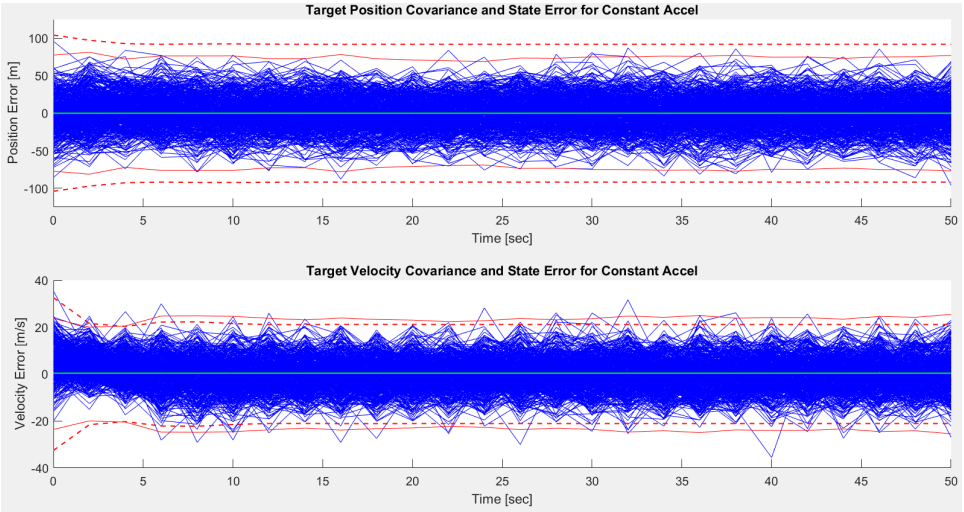
Errors are NOT contained which is also expected because Kalman assumes zero mean Gaussian acceleration. The acceleration that is generated for this plot is constant and not Gaussian.

2.3 ORSE: Zero Mean Acceleration



The target error is contained because the model assumptions are aligned with the true target. characteristics.

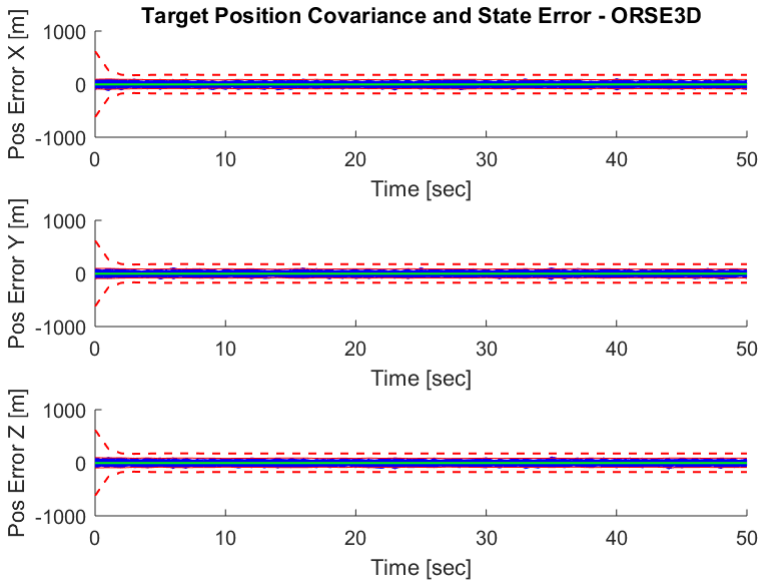
2.4 ORSE: Constant Acceleration

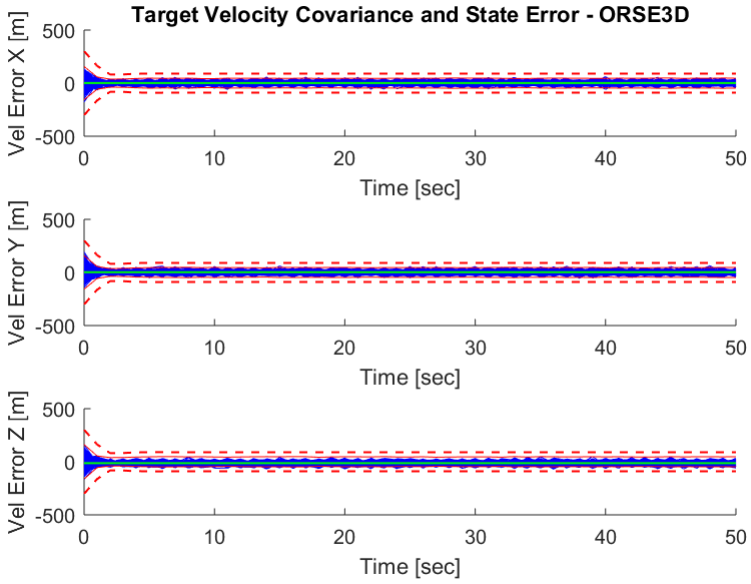


Errors are contained because ORSE does not make the same assumptions as Kalman, thus has the ability to contain constant accelerations.

2.5 ORSE3D

The ORSE filter functionality was extended from 1D to 3D functionality. This involved expanding the 1D matrices to their proper sizes for 3D calculations. Procedurally, the algorithm follows the same format and overall methods for calculation. Lambda was chosen to represent the maximum acceleration of the target experienced in each of the three dimensions. This takes into account the acceleration due gravity in the z-plane with the addition of a smaller random amount in all three dimensions. Accommodations were made in the storing of vectors for use with plotting, and the resulting plots are shown below:





The filter error is constrained in all dimensions for both position and velocity error.

3 Conclusion

The successful implementation of Kalman and ORSE filters requires knowledge of the kinematic state of the object from measurements, as well as the type and magnitude of error in the system. In a real-life combat engagement scenario where the truth trajectory of an object is unknown, the designer is beholden to a set of assumptions regarding the error of sensor measurements. If the noise of a system is not random in nature, Kalman cannot accurately account for the drift that occurs in the residual state error. ORSE filtering is more robust in its implementation for its ability to account a consistent error into the model. Thus, its resulting state error safely resides within the bounds set by the target state covariance. Extending the filter to three dimensions requires redefining the kinematic and filter matrices to store and process each of the spacial vectors together while displaying their outputs separately.

4 References

1. G. Bock, Introduction to Track Filtering, in *Introduction to Weapon Systems.*
2. G. Bock, Track Filtering in the Weapon System, in *Introduction to Weapon Systems.*