TARGET TRACKING: IMPLEMENTING THE KALMAN FILTER

Prakhar Agarwal, David Federman, Kevin Ge, Tzu-Han Jan, Calvin Jones, Yihua Lou, Joshua Ma, Shyam Modi, Kevin Shi, Vanessa Tan, Melissa Zhang

Advisor: Randy Heuer
Assistant: Zack Vogel

ABSTRACT

Target tracking is often complicated by the measurement noise. The noise must be "filtered" out in order to predict the true path of a moving target. In this study of linear filtering, the Kalman filter, a recursive linear filtering model, was used to estimate tracks. Various situations were examined, including maneuvering targets, multiple radars, multiple targets, and collision avoidance. Based on the results, the Kalman filter was successful in smoothing random deviations from the true path of the targets, improving in its ability to predict the path of each target as more measurements from the tracker were processed.

INTRODUCTION

Often the data measured by tracking devices are not exact and must be first filtered to improve the estimates. This occurs due to a factor called "noise". There are two types of noise; the measurement noise is caused by inaccuracies in the tracking device, and state noise is caused by turbulence or human error and other environmental factors. Noise results from multiple factors: atmospheric interference, impreciseness of the radar's measurements, turbulence affecting the target's movement, and human inability to navigate in a perfectly straight line. Measurements taken at irregular time intervals also complicate the estimation process. Data-filtering target trackers are utilized in many situations to provide an estimate of the position and velocity of a target at the time of measurement. They are commonly utilized in air traffic control, navigation, and collision avoidance systems. Such fields have always sought improved accuracy in their predictions. Many different filtering methods have been proposed. For example, early alpha-beta trackers assumed constant errors and non-accelerating targets (1).

The Kalman Filter is a recursive process used to filter random inaccuracies in measurements to predict the most likely position and velocity (or any dimension based on position and time) of a moving target based on real-time position coordinate feeds. It considers the probability of a target's position and then updates its "belief" in the location of the target. (2). The advent and growth of computer science in the last half century allowed for application of the recursive linear filtering solution presented in Rudolf Emil Kalman's 1960 paper (3). In this paper, Kalman described his new approach to linear filtering, a series of recursive equations that seek to minimize error by decreasing the covariance, increasing accuracy of the filter's prediction as each position coordinate is provided by target trackers such as radars.

Rudolf Emil Kalman received his bachelor's and master's degree in electrical engineering from the Massachusetts Institute of Technology, and his doctorate in Science from Columbia University. He was Research Mathematician at Research Institute for Advanced

Study, Professor at Stanford University, Graduate Research Professor and Director at the Center for Mathematical System Theory at the University of Florida, Gainesville, and has also worked for Mathematical System Theory at the Swiss Federal Institute of Technology. Kalman has received many awards and belongs to multiple societies in mathematics, the sciences, and engineering (4).

The Kalman filter is a variant of Bayesian filters. Bayesian filters are utilized for their excellent ability to hone in on the true track of the target as more noisy input data is supplied. However, the Kalman Filter is used in most modern target tracking systems because of its computational efficiency (5). First of all, the filter computes without storing past data. This simplicity allows a single personal computer to track upwards of thousands of targets at once. The recursive formulas produce more confident predictions, valuing future points less heavily as compared to the experience gained (abstractly called the "Kalman Gain") from successfully decreasing the magnitude of error in its predictions. Additionally, the filter adapts to varying measurement time intervals and is able to provide error estimates.

This project implemented the original version of the Kalman filter in a Java program that processed simulated data. The equations provide estimates which can be plotted to visually depict a straighter, more accurate path of the target. Noisy measurements were simulated from true tracks. Those measurements were run through the filter, and the filter's predictions were compared to the true track to determine accuracy and success of the filter.

The Model

The Kalman Filter is modeled by utilizing a linear algebra approach using matrices (Equations 1 and 2).

$$X_{k+1} = \Phi X_k + q_k \tag{1}$$

$$Y_k = H X_k + r_k \tag{2}$$

Equation 1 shows that X_{k+1} is a matrix representing the updated state and X_k is a vector representing the current state, which contains position and velocity vectors.

$$\Phi = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

Matrix Φ is a state transformation matrix that relates the state of one time step to the next. This recursive process includes the use of Δt to update the new location and velocity of the target. The variable *k* represents the time step. In our case, Φ is represented by this $4x4$ matrix in Equation 3.

$$q_k - \begin{bmatrix} x_{error1} \\ \dot{x}_{error1} \\ y_{error1} \\ \dot{y}_{error1} \end{bmatrix} \tag{4}$$

The q_k vector is the process noise (Equation 4). In other words, it is the noise due to uncertainty in the transition.

$$Y_k = \begin{bmatrix} x \\ y \end{bmatrix} \tag{5}$$

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{6}$$

$$r_k = \begin{bmatrix} x_{error2} \\ y_{error2} \end{bmatrix} \tag{7}$$

The Y_k matrix (Equation 5) in Equation 2 is a representation of the current measurement in a $2x1$ matrix. The H matrix is a transformation matrix to turn X_k's $4x1$ matrix into a $2x1$ (Equation 6). To do this, the H matrix must be 2x4. The r_k vector is a $2x1$ measurement noise vector (Equation 7). Measurement noise can be described as the error caused by the inaccuracy of our instruments.

The Filter

The Kalman Filter is a set of equations that provides a method to estimate the state of a process. This series of equations consist of two steps: Predict and Correct. These two steps of predict and correct are used recursively. In real time, the raw data would be added to the filter during the correct step. After the current data points are received, the correct step is used to estimate the state and its variances.

Predict:

$$\hat{X}_{k+1|k} = \Phi \hat{X}_{k|k} \tag{8}$$

$$P_{k+1|k} = \Phi P_{k|k} \Phi^T + Q \tag{9}$$

The notation of $\hat{X}_{k+1|k}$ means "$\hat{X}_{k+1}$ given the state of $\hat{X}_k$". $\hat{X}_{k+1|k}$ is the prediction of the X vector at time step $k+1$ given the information known at time step k. Equation 8 projects the next predicted state given the previous state.

The P matrix in Equation 9 is the state covariance matrix representing the covariance of the (x, y) position coordinates and the covariance of the $(\dot{x}, \dot{y})$ velocity vectors. The Q matrix in Equation 9 is a covariance matrix of the process noise. Equation 9 creates a predicted state matrix that factors in process noise and Δt. Note that the predicted state covariance matrix can never be smaller than the current state covariance matrix. During the predict step, we are accumulating error every time we take a raw data measurement.

Correct:

$$K_k = P_{k|k-1}H^T\left[HP_{k|k-1}H^T + R\right]^{-1} \tag{10}$$

$$\hat{X}_{k|k} = \hat{X}_{k|k-1} + K_k\left[Y_k - H\hat{X}_{k|k-1}\right] \tag{11}$$

$$P_{k|k} = [I - K_k H]P_{k|k-1} \tag{12}$$

Equation 10 computes the Kalman Gain, which is the gain or weighting factor for the residual. The Kalman Gain is bounded by R and is only as accurate as the P matrix that is in Equation 10. The R matrix is a covariance matrix of the measurement noise (Equation 13). The shape of the covariance matrix would be an oval.

$$R = \begin{bmatrix} \sigma_x^2 & \sigma_x\sigma_y \\ \sigma_x\sigma_y & \sigma_y^2 \end{bmatrix} \tag{13}$$

Equation 11 updates the state by employing the Kalman Gain and the measurement. The Kalman Gain is used as a weighting factor for the residual ($Y_k - HX_{k|k-1}$), the difference between the actual and predicted measurements, to more accurately correct the predicted state. As time goes on, the measurement is weighted less due to the effect of the gain.

Equation 12 updates the state covariance matrix ($P_{k|k}$) based on the Kalman gain and the predicted state covariance matrix ($P_{k|k-1}$). I is an identity matrix. Note that the updated state covariance matrix can never be larger than the previous state covariance matrix, which means that the estimate gets more accurate.

In effect, the model should become more and more accurate (within the limit of the Q and R matrices) since the model is taking account of previous data to estimate the velocity and position in the future.

SCENARIOS

Basic Kalman Filter

In this scenario, Kalman filter was used to determine the velocity and position of an aerial body with constant velocity within radar range. The filter uses raw Cartesian data to predict the actual position and velocity of the target object, filtering out error caused by noise. The predicted path of the aerial body was then compared with the true path of the target object by calculating the residuals and then creating a residual plot (Figure 1). Residuals are the distance between the predicted and true values. A residual plot is a visual representation of the residuals, which was used to efficiently illustrate the accuracy of the predictions of the filter.

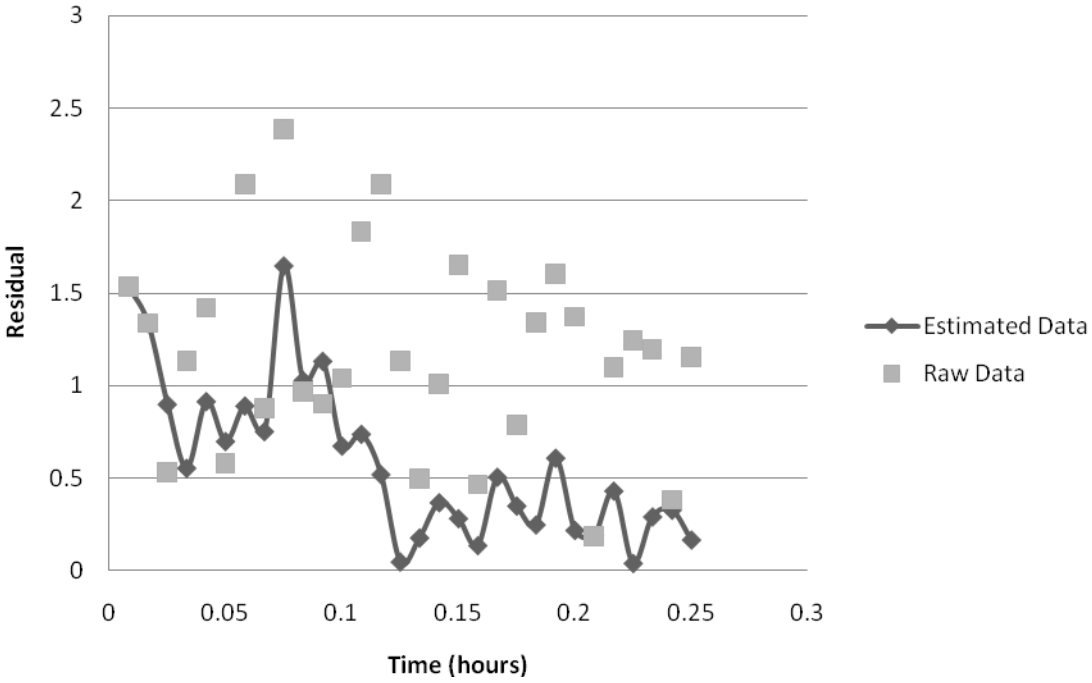


Figure 1 Basic Kalman Filter: Cartesian Coordinates

The overall trend of the Kalman filter estimates is that they become more and more accurate as time passes. The predicted path of the target object had lower residual values compared to the raw data path, illustrating that the performance of the filter improves upon the raw data.

Tracking With Polar Coordinates

This task involved receiving radar data in a more realistic manner, as range, r, and bearing angle, α (Figure 2).

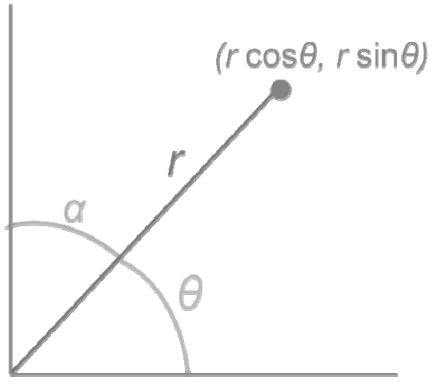


Figure 2 The relationship between the polar and Cartesian coordinates

First, the coordinates given as (r, α) were converted to Cartesian coordinates (x,y) to be used in the previously constructed filter (Equations 14 and 15).

$$y = r\sin\alpha \tag{14}$$

$$x = r\cos\alpha \tag{15}$$

Next, the measurement error matrix, R (Equation 20), needed to be adjusted since the data was measured as range and bearing. Only the measurement standard deviations in range and bearing, σ_r and σ_α, are known. To properly apply the filter, Equations 16 through 19 calculated σ_x^2, σ_{xy}^2, and σ_y^2 from σ_r and σ_α (6).

$$\theta = \frac{\pi}{2} - \alpha \tag{16}$$

$$\sigma_x^2 = \sigma_r^2\cos^2\theta + r^2\sin^2\theta\sigma_\alpha^2 \tag{17}$$

$$\sigma_y^2 = \sigma_r^2\sin^2\theta + r^2\cos^2\theta\sigma_\alpha^2 \tag{18}$$

$$\sigma_{xy}^2 = \frac{1}{2}\sin^2 2\theta[\sigma_r^2 - r^2\sigma_\alpha^2] \tag{19}$$

$$R = \begin{bmatrix} \sigma_x^2 & \sigma_{xy}^2 \\ \sigma_{xy}^2 & \sigma_y^2 \end{bmatrix} \tag{20}$$

Conceptually, the transformations of the variances are necessary (Equations 17 through 19), because the variables x and y are no longer independent, and therefore variance between the values, σ_{xy}^2, is not zero. The ellipse (Figure 3) represents a region of high probability for the true value of a measured data point, taking into account measurement noise. In the Cartesian plane, the ellipse is rotated because errors in x and y are not independent. In other words, the covariance in r and α each affect both the covariance in x and y.

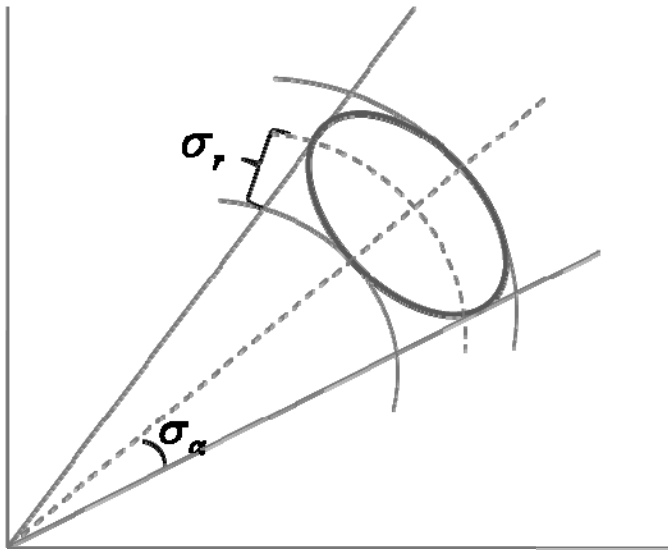


Figure 3 The error ellipse reflects the degree of error in radar measurements

After the coordinates and measurement error matrix are transformed, the data has the proper format to work with the remainder of the code and provide estimates of the target's location. Figure 4 shows the residuals of the filtered data to be consistently lower than those of the measured.

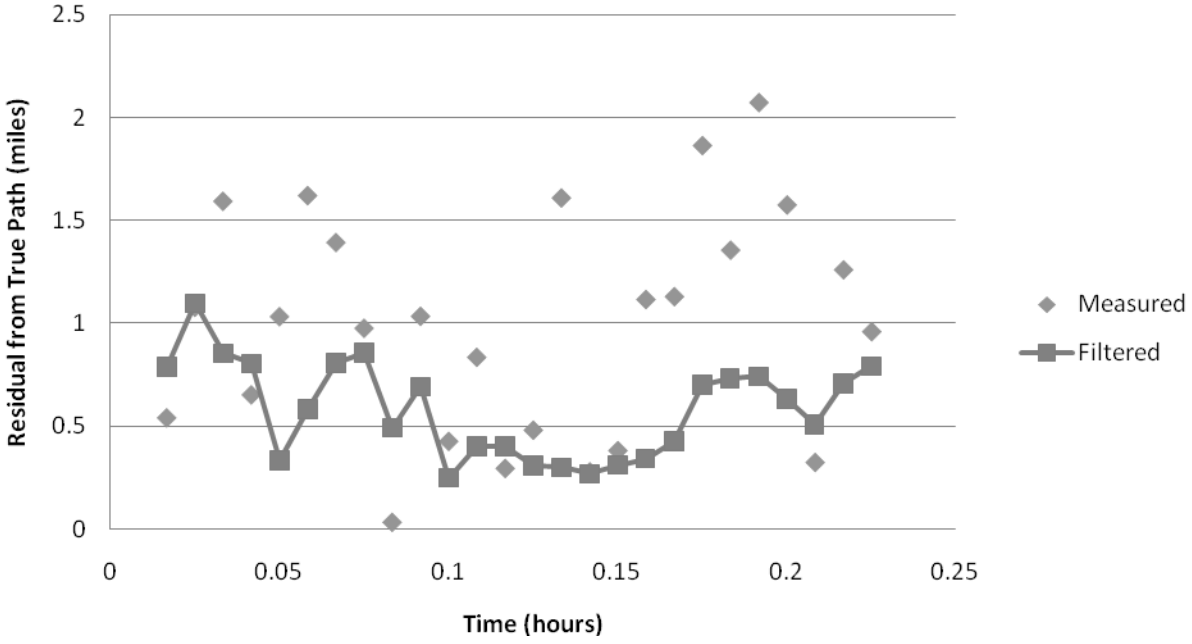


Figure 4 Tracking with Polar Coordinates: Residual Plot

Tracking With Multiple Radars

The next scenario involved tracking a target with two radars, both with known coordinates on a Cartesian plane. While traveling, the target moved out of the range of the first radar, and a second had to be used. The data from the two radars was combined and processed in time order.

The implementation of multiple radars did not involve significant changes to the source code. The data was still recorded as range and bearing of the target relative to the radar. The only change was that each data set had an associated radar number depending on which radar it came from as well. Knowing which radar each data point came from, the point's coordinates could then be translated to a common coordinate plane. This was done by converting to Cartesian coordinates as in the previous scenario and then adding the coordinates of the proper radar as offset.

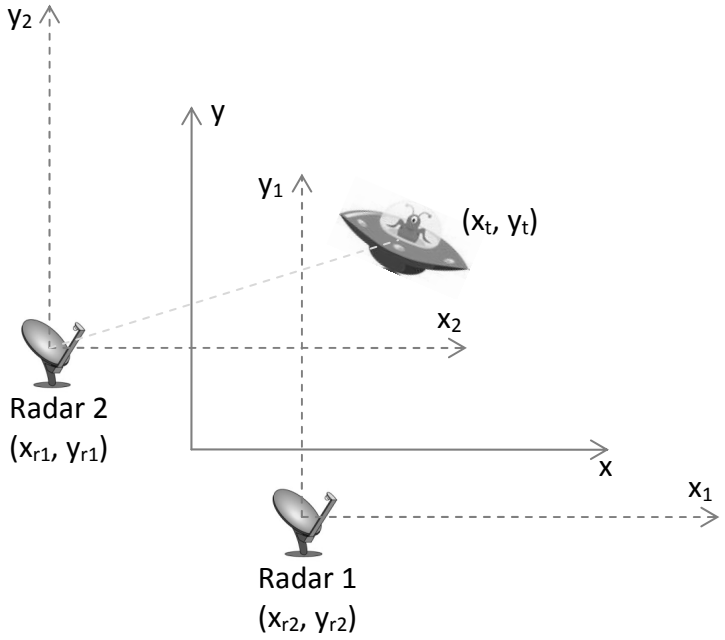


Figure 5 The common coordinate system with those of the two radars

Figure 5 shows that the new coordinates (x_t',y_t') are found by shifting the measured coordinate of the target (x_t,y_t) by the coordinates of the radar that measured the data (x_r, y_r) using Equation 21.

$$(x_t', y_t') = (x_t + x_r, y_t + y_r) \tag{21}$$

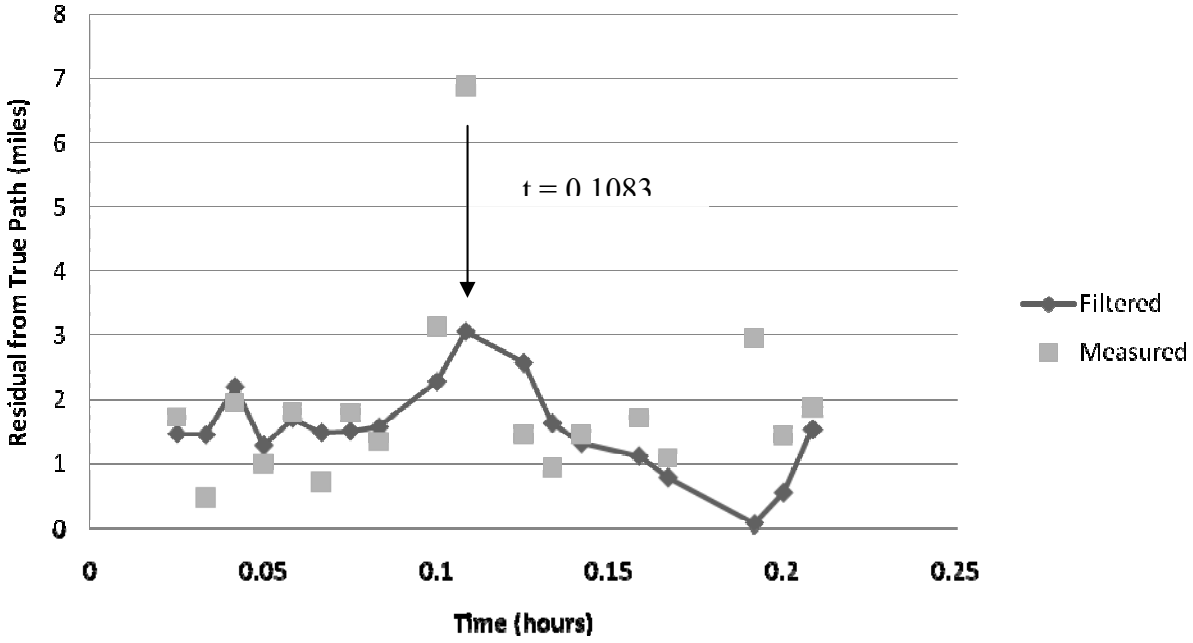


Figure 6 Tracking with Multiple Radars: Residual Plot

There is a significant increase in the measurement residual when the radar sending data switches at t = 0.1083 hours (Figure 6). In comparison, the increase is less dramatic in the filtered data at the same time step. This is because the measurement noise ellipses from the two radars overlap (Figure 7). The resulting measurement error, represented by their intersection, is significantly reduced.

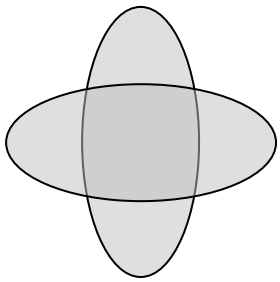
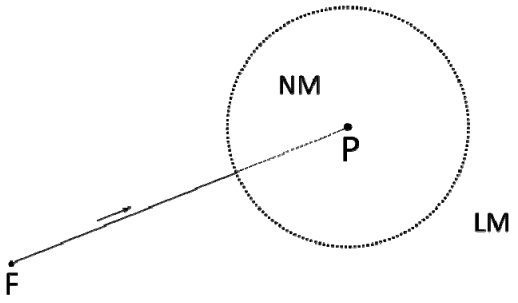


Figure 7 Intersection of the error ellipses of the two radars

Target Maneuvering

This scenario involves a target that maneuvers in sharp turns and multiple linear paths, presenting a problem to our existing program. The existing filter attempts to model the target's path as a straight course, incorrectly tracking the path of a maneuvering target. Our solution to this scenario involves the use of residual gating to identify the time-steps where the target maneuvers.

Two gates are specified, the NM gate (no maneuver) and the LM gate (likely maneuver), shown in Figure 8. Upon the introduction of a data point whose residual falls far outside a given threshold and into the LM gate, our program flags an alert and waits to see if consecutive data points fall into the LM gate as well. Upon reaching two consecutive alerts, the program re-initializes the filter with a new set of starting parameters and conditions.



F = previously filtered point
P = predicted point
NM = no-change gate
LM = re-initialize gate

Figure 8 Residual gates

However, simply comparing the distance between the predicted point and the measured point is insufficient due to state uncertainty, as described in our state covariance matrix. Thus, in order to find the relative distance from the two points, "distance" must be measured in terms of relative probability, in units of standard deviations. By taking only the position elements of our state covariance matrix, we can construct a matrix transform to create the C matrix and its inverse, from which an equation that calculates the "distance" between our predicted and measured points in standard deviations can be formulated (Equations 22 through 24) (7).

$$(n\sigma)^2 = r^T C^{-1} r \tag{22}$$

$$r = [Y_k - H X_{k|k-1}] \tag{23}$$

$$C^{-1} = \begin{bmatrix} C_{xx}^{-1} & C_{xy}^{-1} \\ C_{yx}^{-1} & C_{yy}^{-1} \end{bmatrix} \tag{24}$$

At each time-step, we can calculate the residual in our correct procedure, enabling our filter to effectively detect sharp maneuvers. Our implementation re-initializes the filter upon detecting two consecutive data points with residuals over four standard deviations, which provides a certainty better than one in a thousand.

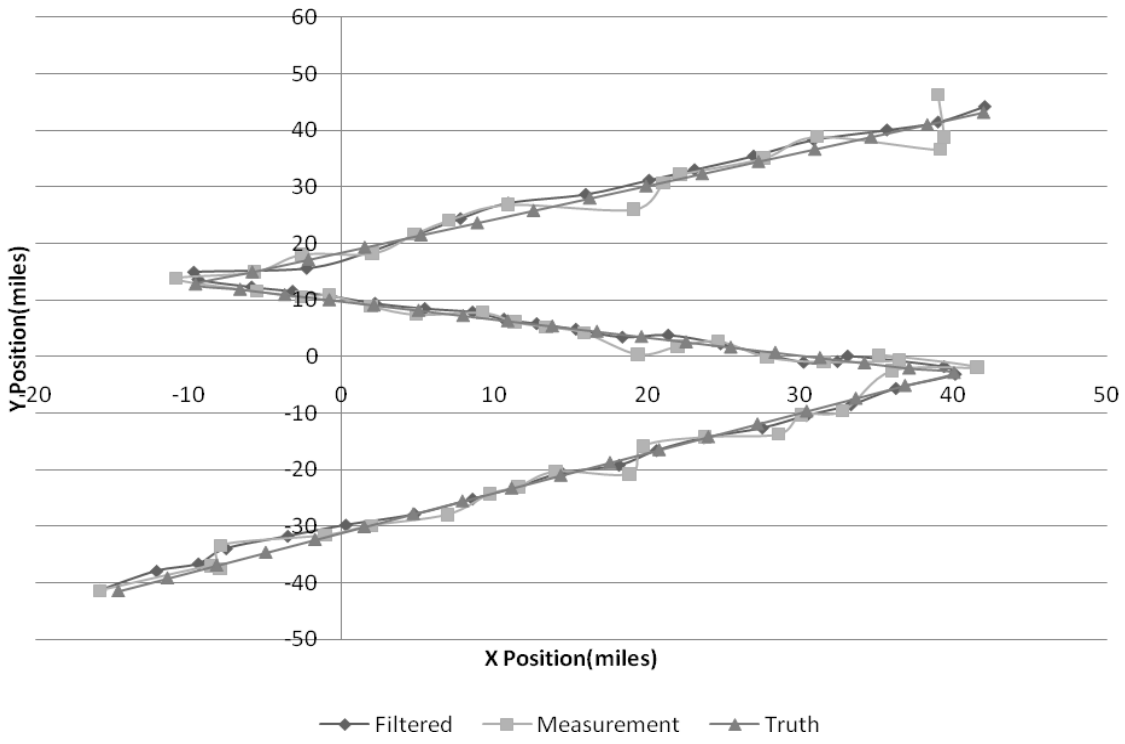


Figure 9 Path of Maneuvering Target

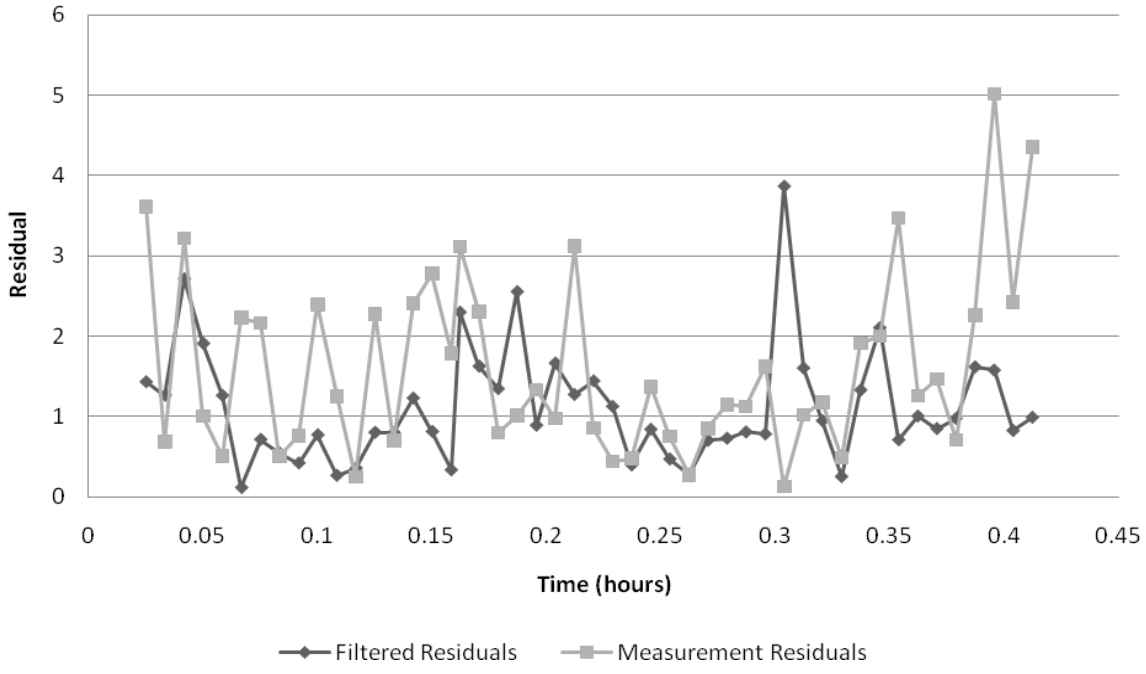


Figure 10 Target Maneuvering: Residual Plot

As demonstrated in Figures 9 and 10, our program recovers very fast from maneuvering tactics, shown by the filter's low residual even after each maneuver.

Target Interception

Though the Kalman filter can be used to predict the path of a moving target, the applications of the filter can also be useful in calculating the path of interception. To do so requires first calculating the position and velocity of the target, projecting its path, and then computing the angle of interception for the designated course.

In Figure 11, an idealized diagram of the problem is illustrated. By using the Law of Cosines (Equation 25), the time of intersection can be calculated, from which angle γ can be calculated.

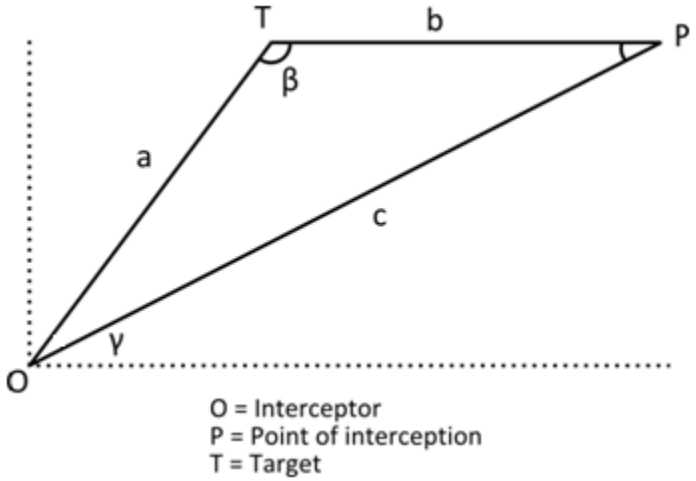


Figure 11 Diagram of target interception course

$$c^2 = a^2 + b^2 - 2ab\cos\beta \tag{25}$$

$$v_1 = \sqrt{\dot{x}^2 + \dot{y}^2} \tag{26}$$

$$v_2 = 630 \tag{27}$$

$$\vec{a} = \sqrt{x^2 + y^2} \tag{28}$$

$$\vec{b} = v_1 t \tag{29}$$

$$\vec{c} = v_2 t \tag{30}$$

Angle β is never directly given, but $\cos\beta$ can be calculated using the components of vectors $\vec{a}$ and $\vec{c}$ (Equation 31).

$$\cos\beta = \frac{\dot{x}x + \dot{y}y}{\sqrt{\dot{x}^2 + \dot{y}^2}\sqrt{x^2 + y^2}} \tag{31}$$

Substituting a, b, c and $\cos\beta$ in the equation of the Law of Cosines with their calculated equivalents gives Equation 32, which can be put in quadratic form (Equation 33).

$$t^2 630^2 = (x^2 + y^2) + v_1{}^2 t^2 - 2(x^2 + y^2)(v_1 t)\cos\beta \tag{32}$$

$$t^2(630^2 - v_1{}^2) + 2t(x^2 + y^2)(v_1)\cos\beta - (x^2 + y^2) = 0 \tag{33}$$

Solving for t using the quadratic equation returns two roots; however, only one is positive if the path of interception is possible. Now, bearing angle γ can simply be calculated by projecting the y-change and x-change of the interceptor and taking the inverse tangent of the ratio between the two (Equations 34 through 36).

$$\Delta x = -y + \dot{y}t \tag{34}$$

$$\Delta y = -y + \dot{x}t \tag{35}$$

$$\gamma = \tan^{-1}\frac{\Delta y}{\Delta x} \tag{36}$$

As shown below in Figure 12, the interceptor successfully follows the interception path of the target, ending with a successful interception.

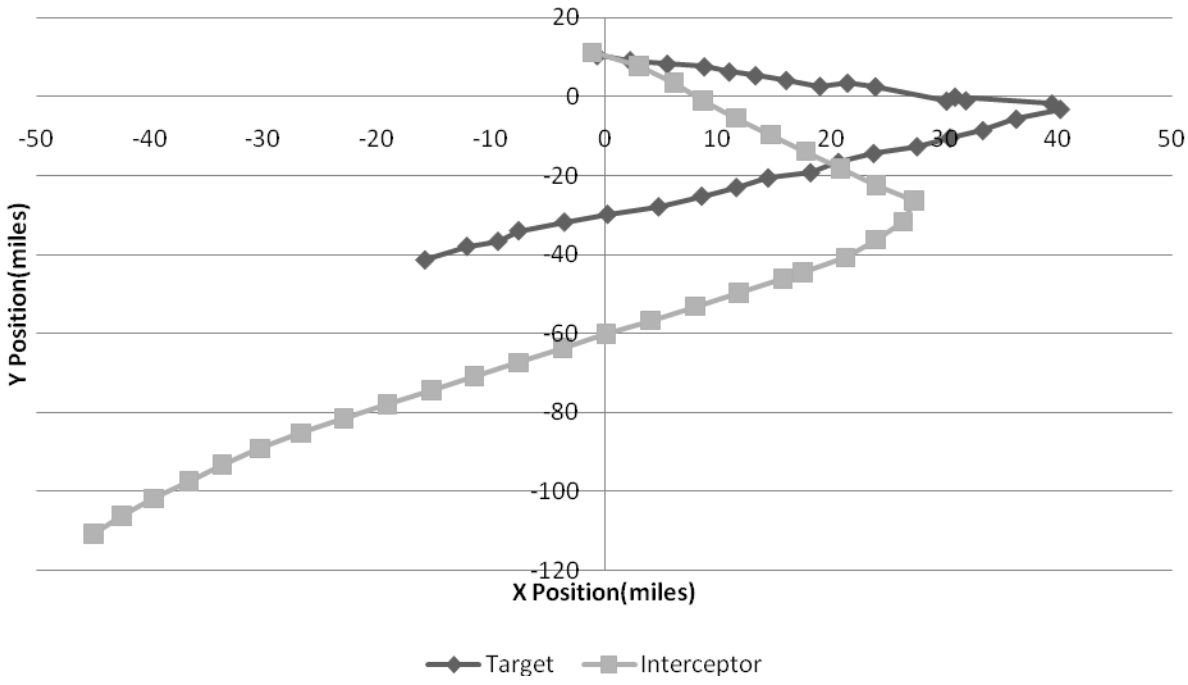


Figure 12 Paths of Interceptor and Target

Tracking Multiple Targets

The Kalman Filter can be used to track the position of multiple targets. To do this, an object-oriented approach was used: a plane class was created containing the iterate method and all the data associated with the plane. Two instances of the plane class were created upon running the program, one for each target being tracked. The main class read in the data and called the iterate method in each of the plane instances, passing the data to them as parameters. The new data points were then retrieved from both planes and printed out.

This object-oriented approach greatly simplified the problem of managing data from both targets and also allowed for more targets to be added and tracked, as new plane instances could easily be created.

Figure 13 shows the two planes being tracked as they travel in a line.

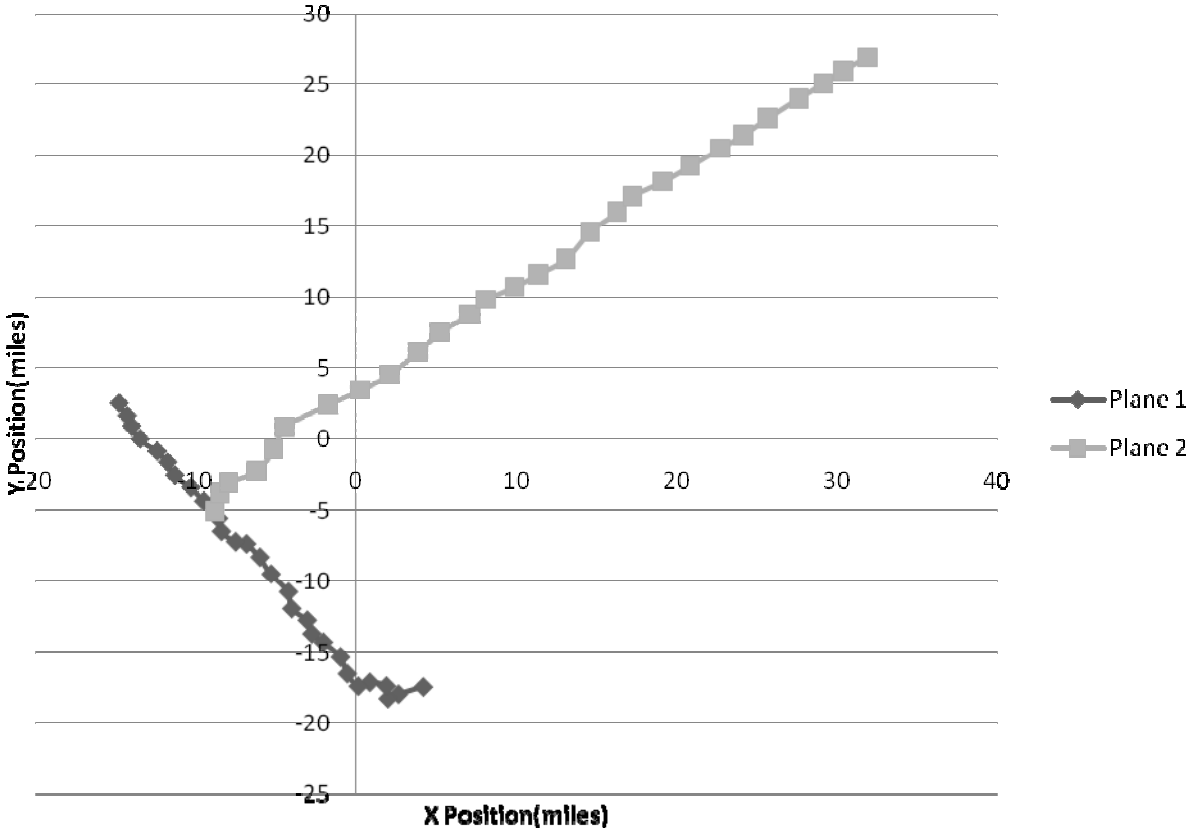


Figure 13 Paths of Multiple Targets

Collision Avoidance Systems

One of the applications of multiple-target tracking is for collision avoidance systems. Planes flying at high speeds often cannot see each other in time to communicate and maneuver to avoid a collision. Thus, tracking the position of planes and alerting them in advance when their trajectories would lead to possible future collisions is vital in air traffic control.

Let x_1, y_1, xv_1, and yv_1 be the x-position, y-position, x-velocity, and y-velocity of the first plane, respectively, and x_2, y_2, xv_2, and yv_2 the second plane. Equation 37 was used after each iteration to check the distance between the planes.

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{37}$$

If Equation 37 gave a value of less than 12 miles, the paths of the planes were then extrapolated to check if they would pass within one mile of each other. A time step of τ = 0.0005 hours was used inside a while loop to project the paths.

$$x_1 = x_1 + xv_1 * \tau \tag{38}$$

$$y_1 = y_1 + yv_1 * \tau \tag{39}$$

$$x_2 = x_2 + xv_2 * \tau \tag{40}$$

$$y_2 = y_2 + yv_2 * \tau \tag{41}$$

The projected positions of the two planes were updated (Equations 38 through 41). Equation 37 was then used to check if the updated distance was less than 1 mile. If so, a message was sent to both pilots alerting them of the projected point of collision. If the new projected distance was found to be greater than the previous projected distance, meaning the planes were traveling away from each other, the loop ended, as the pilots were not in danger. Otherwise, the positions were updated once more using the above equations.

Figure 14 shows the point at which the distance between the planes is less than twelve miles, as well as the projected path and collision point. The filter alerts the planes to the possible collision at that time.

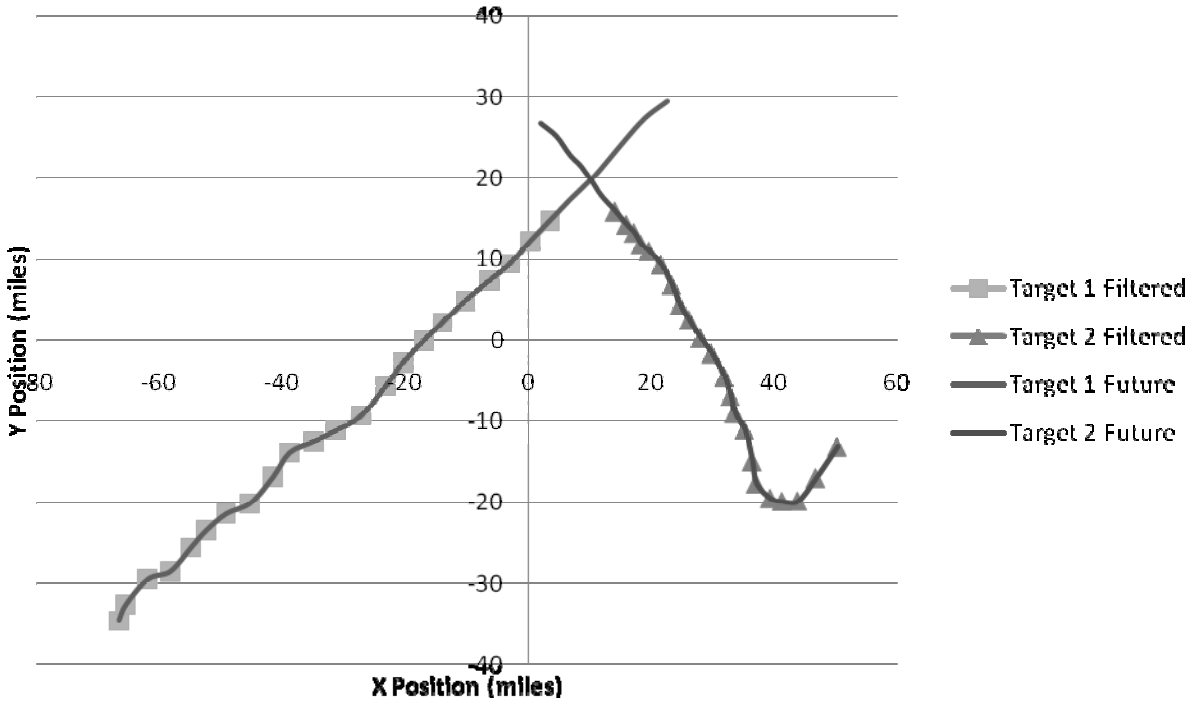


Figure 14 Paths of Multiple Targets with Collision Alert

CONCLUSION

The results obtained from the implementation of a Kalman filter in the tested scenarios clearly display that the Kalman filter is an effective tool. Whether used to track a single target, to intercept the target, or to avoid aircraft collisions, the filter maintained a degree of accuracy significantly higher than that of the raw measurements. The average residuals, in fact, dropped up to forty percent in many cases.

There are many real life applications that are reflective of the scenarios. The ability to track multiple targets and predict collisions between them lends the filter to use in air traffic control. The algorithm could also be used to filter out environmental factors and aid aircrafts for a computer automated autopilot. Additionally, it is applicable in dynamic positioning by which sea vessels take into account wind and sea currents while maintaining a steady position. The Kalman filter is even useful for fields such as macroeconomics, where noise in the data can be reduced to determine the overall market trend.

Possibly the most noteworthy aspect of the Kalman filter is its ability to help track an object with progressively increasing accuracy while not using a significant amount a computer's memory. All the calculations are based solely off data from the current point and the one just

before it. On the other hand, data-modeling techniques such as fitting a least squares regression line requires all of the past data (5).

As the Kalman filter progresses through its stages of predicting and correcting, it is able to keep track of its error and gradually reduce it. The adaptive nature of the algorithm, coupled with its low memory requirements, make the Kalman filter a valuable tool in a myriad of fields.

REFERENCES

(1) Microstar Laboratories. Engine speed monitoring: the alpha-beta filter. [Internet]. [cited 2009 Jul 22]. Available from: http://www.mstarlabs.com/control/engspeed.html

(2) Fox D, Hightower J, Liao L, Schulz D, Borriello G. Bayesian filters for location estimation. IEEE Pervasive Computing, IEEE Computer Society Press; [Internet]. [cited 2009 Jul 22] 2(3): 24-33. Available from: http://www.seattle.intel-research.net/pubs/fox2003bayesian.pdf

(3) Kalman RE. A new approach to linear filtering and prediction problems. Journal of Basic Engineering. 1960; [Internet]. [cited 2009 Jul 22] 82(D): 35-45. Available from: http://noodle.med.yale.edu/~gliu/research/jc2002/kalman1960.pdf

(4) Welch G, Bishop G. [updated 2007 Dec 21] Rudolf Emil Kalman. The Kalman filter. [cited 2009 Jul 22] Available from: http://www.cs.unc.edu/~welch/kalman/kalmanBio.html

(5) Welch G, Bishop G. [updated 2006 Jul 24] An introduction to the Kalman filter. The Kalman filter. [cited 2009 Jul 22] Available from: http://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf

(6) Blackman SS. Multiple-Target Tracking with Radar Applications. Norwood: Artech House. 26-27. 1986.

(7) Atwood B. [updated 2003 Aug] Covariance and GLAST. Stanford University. [cited 2009 Jul 24] Available from: http://www.glast.slac.stanford.edu/software/anagroup/WBA072003-Covariance.pdf