A Multiple Target Range and Range-Rate Tracker Using an Extended Kalman Filter and a Multilayered Association Scheme

A thesis

submitted by

Leah Uftring

In partial fufillment of the degree requirements

for the degree of

Master of Science

in

Electrical Engineering

Tufts University

May 2008

Advisor: Dr. Eric Miller

Abstract

In many applications, it is necessary to track an unknown number of targets with a remote sensor. Several applications for which radar is used to sense the environment include anti-aircraft warfare, tracking of space debris, and missile defense. If wideband radar is used, one can receive multiple returns from scatterers on a single object in one pulse. In the particular problem considered, it is necessary to track multiple scatterers on an unknown number of objects, whose motion includes a nonlinear rotational velocity component, with a generic wideband radar model. The measurements used are scatterer range and range-rate (in analogy to Doppler) and are nonlinear functions of the scatterer position and velocity, parameters in the state-space model. The tracker design uses an Extended Kalman Filter to model the nonlinear dynamics and measurement models and a multi-pronged association scheme to determine the scene and track its evolution over time.

Contents

List of Figures

List of Tables

A Multiple Target Range and Range-Rate Tracker Using an Extended Kalman Filter and a Multilayered Association Scheme

Chapter 1

Introduction

Multiple target trackers (MTT) are often used in many applications including air defense, ground target tracking, and missile defense [1]. In this situation, there are multiple objects in the scene providing multiple returns to the sensor. A variety of sensors, including IR, sonar, ground-based radar, and airbourne radar, can be used in these applications [2, 3, 4]. MTT have two portions: an association algorithm to assign new measurements to current information and a filter to track the objects [1, 2, 4].

The filter component models the object dynamics and the measurements in terms of those dynamics [3]. Since we may have some idea of the objects' motion, but this motion can not be known exactly, a small amount of noise is added to the model. This added noise is called process noise [1]. A second type of noise is added to the measurements. Since an actual sensor is making the measurements, there will be some error in the measurements, and this additional measurement noise in the model compensates for this occurrence [4]. The modeled object dynamics consist of the state and the function that propagates this state into the future, the state transition function. The measurement function maps the state to the measurements.

If both the state transition function and the measurement function are linear functions of the state, we can use a linear filter. The optimal linear filter, for additive Gaussian noise, is the Kalman filter [1, 5]. If either the measurement function or the state transition function, or both, are nonlinear, then a nonlinear filter must be used [1, 5]. Several types of nonlinear filtering techniques exist [5]. The easiest to use are suboptimal, but the optimal filters are difficult to use and computationally complex. However, the challenge of tracking objects with nonlinear dynamics and with nonlinear relationships to the measurements still exist, and this problem must be tackled.

Extended Kalman Filters (EKF) are often used to handle these nonlinear problems. Some examples in which EKFs have been used to track nonlinear object motion are: accelerating targets [6], satellite trajectory estimation [7], turning civilian aircraft [8], and road target tracking [9]. Another common type of nonlinear filter is the particle filter [5]. There is a large difference in computational complexity between these two types of filters. The particle filter carries with it a much higher computional burden than the EKF. While particle filters have sometimes been favored when dealing with nonlinear problems [10, 11], the EKF can provide comparable, if not better, results, especially when computational complexity is taken into account [12, 13, 14]. Other types of nonlinear filtering techniques exist, such as the Gaussian Sum Filter or the Unscented Kalman Filter(UKF). However, these filters do not provide the same ease of use as the EKF, and, in the case of the UKF, may again demand a greater computational complexity [5].

The second component of a MTT is the data association algorithm [4]. The association portion of the tracker determines how new information is mapped to current and old information [2]. Association algorithms consist of a cost function and an assignment method. The cost function is the equation which

determines how likely particular assignments are, and the assignment method determines which assignments are allowed. Standard cost function are generally related to distance or a similar metric. Three types of assignment methods are common. These are nearest neighbor, global nearest neighbor, and joint probabilistic data association. The best assignment method to use depends on the particular application [4, 15]. Nearest neighbor is good to use when there is a high probability of missed detections. Global nearest neighbor is good to use when there is a low probability of both missed detections and false alarms. Joint probabilistic data association is good to use when both the probability of false alarms and missed detections are high. The applications of these assignment methods is done using an optimal assignment algorithm. Several different techniques to do this are the Munkres Algorithm, the JVC algorithm, and the Auction algorithm. The easiest to implement is the Munkres algorithm [3].

1.1 Contributions

In this work, we use an EKF to estimate the dynamics of objects with a nonlinear rotational velocity component. We use a generic radar model for our measurements. The measurements used are range and range-rate and will be nonlinear functions of the filter state.

Since our measurements are range and range-rate and we want to find states in a Cartesian frame, we have an unobservable problem [3]. Much of the work that exists when dealing with range and range-rate measurements also includes a measurement of angle, which makes the problem observable [16, 17, 18, 19]. Another interesting part of our problem is the dynamics of our targets. Existing work tends to model rotational object motion as a maneuver

[8, 20, 21]. Existing work also tends to treat objects as point targets which will only give one return [7, 8, 9, 20]. Although we use a generic sensor model for our measurements, we use a generic wideband sensor model. If the radar has a high range resolution resulting from large bandwidth, it is called a wideband radar. In this case, one can receive multiple returns from a single object in one pulse. The parts of the object which generate these returns are called scatterers.

This work makes three contributions. First, it deals with modeling the nonlinear rotational component as an explicit part of the object dynamics. This is different from the previous considerations which treat targets with rotational motion as undergoing a maneuver and not as a constant portion of the object's dynamics. Second, we consider an unobservable problem in order to see how well things can be done in this case. The third major contribution is that it deals with measurements from a wideband sensor.

1.2 Outline of the Thesis

This thesis is organized the following way. Chapter 2 focuses on the technical background in designing a tracker. The three components are the state space model, the filter used, and the association scheme. Different options are discussed for each component. The next chapter describes the design of the tracker: the state space model used, the details of the filter equations, and the association algorithm used. The experiments are discussed in Chapter 4. We ran experiments on the filter and the association algorithm separately. Then we put these together for our tracker and repeated the experiments. This chapter shows our results. Chapter 5 wraps up the thesis with our conclusions and suggestions for future work.

Chapter 2

Filtering and Association Techniques

A multiple target tracker needs to be designed to track objects over time. While the details of a tracker design depend on its particular application, each generally consists of two major components: an association scheme and a filtering method, along with an appropriate dynamic, or state space, model of the target.

2.1 State Space Models

The modeling of nonlinear and/or maneuvering object dynamics is by no means novel. State space models are dependant on type, coordinate system, and reference frame. Three types of nonlinear state space models are continuous time, discrete time, and continuous-discrete time [22]. The continuous time model is

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{f}(t, \mathbf{x}(t)) + \mathbf{G}(t, \mathbf{x}(t))\mathbf{w}(t) \\ \mathbf{z}(t) &= \mathbf{h}(\mathbf{x}(t), t) + \mathbf{v}(t) \end{aligned} \tag{2.1}$$

where the former equation is the state dynamic model, $\dot{\mathbf{x}}(t)$ is the vector of state space derivatives with respect to time; $\mathbf{f}(t, \mathbf{x}(t))$ is the nonlinear function that maps the current state, $\mathbf{x}(t)$, and current time, t, to its derivatives; $\mathbf{w}(t)$ is the process noise vector as a function of time; and $\mathbf{G}(t, \mathbf{x}(t))$ is the matrix which maps the process noise into each component of the state vector derivatives. Then the components of the measurement equation are: $\mathbf{z}(t)$ is the current measurement; $h(\mathbf{x}(t))$ is the nonlinear function that maps the current state and time to the current measurement; and $\mathbf{v}(t)$ is the measurement noise.

The discrete time model is

$$\begin{aligned} \mathbf{x}(k+1) &= \mathbf{f}(k, \mathbf{x}(k)) + \mathbf{G}(k)\mathbf{w}(k) \\ \mathbf{z}(k) &= \mathbf{h}(k, \mathbf{x}(k)) + \mathbf{v}(k) \end{aligned} \tag{2.2}$$

where $\mathbf{x}(k+1)$ is the state at time $k+1$; the nonlinear function which maps the current state, $\mathbf{x}(k)$, is $\mathbf{f}_k(k, \mathbf{x}(k))$; the process noise is $\mathbf{w}(k)$; and the matrix which maps the process noise to the next state is $\mathbf{G}(k)$. The measurement noise at state k is $\mathbf{v}(k)$, and the nonlinear function that maps the current state to the current measurement, $\mathbf{z}(k)$, is $\mathbf{h}(k, \mathbf{x}(k))$.

The third formulation, the continuous-discrete time formulation, is:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{f}(t, \mathbf{x}(t)) + \mathbf{G}(t, \mathbf{x}(t))\mathbf{w}(t) \\ \mathbf{z}(k) &= \mathbf{h}(k, \mathbf{x}(k)) + \mathbf{v}(k) \end{aligned} \tag{2.3}$$

where the continuous time quantities are defined as in the continuous time formulation and the discrete time quantities are defined as in the discrete time formulation. This model is especially helpful when taking measurements of the object at discrete sample times and the continuous time motion of the object

is known. In each of these set of equations, the control input, $\mathbf{u}(k)$, has been left out since it will not be applicable in this particular case.

In our applications, while our targets are spinning throughout all measurement times and do not undergo a manuever, considering the work that has been done for targets undergoing a circular manuevering is a good place to start. An extensive survey of point target manuevering models is given in [20]. The two most applicable models are the coordinated turn and the circular manuever. The coordinated turn model descibes the position and velocity of the object during its circular turn in Cartesian coordinates. The circular manuever model places its polar coordinate system at the center of the manuever. Since we wish to model our states in Cartesian coordinates, but also need to include a rotational motion component that is body-centric, both of these models are relevent to designing the state space model we develop for our problem in Chapter 3.1.

The coordinated turn model uses a Cartesian coordinate system in the sensor reference frame. This continuous time model gives [20]:

$$\dot{x}(t) = v(t)\cos(\phi(t)) \tag{2.4}$$

$$\dot{y}(t) = v(t)\sin(\phi(t)) \tag{2.5}$$

$$\dot{v}(t) = a_t(t) \tag{2.6}$$

$$\dot{\phi}(t) = \frac{a_n(t)}{v(t)} \tag{2.7}$$

$$\tag{2.8}$$

where $\dot{x}(t)$ is the derivative of the state position in x with respect to time, $\dot{y}(t)$ is the derivative of the state position in y with respect to time, $v(t)$ is the velocity vector, ϕ is the heading angle, a_t is the tangential acceleration and a_n

is the normal acceleration. For circular, constant-speed motion, $a_n \neq 0$ and $a_t = 0$.

The circular manuever is best modeled from the center of the manuever [20]. This discrete time model is maneuver centered and assumes the center of the maneuver is known. This allows one to calculate r, the radius of the maneuver, from the measurement, along with ϕ, the current angle of the object in the maneuver circle, and $\dot{\phi}$ which maps the linear velocity and the rotational velocity together. This gives, for a sample time T, [21]:

$$r(k+1) = r(k) \tag{2.9}$$

$$\phi(k+1) = \phi(k) + \dot{\phi}(k)T \tag{2.10}$$

$$\dot{\phi}(k+1) = \dot{\phi}(k). \tag{2.11}$$

Measurements are generally given in the sensor coordinate frame of range, azimuth, elevation, and range-rate [23]. When such sensor coordinates are used for the measurements and measurement noise, but the target state is not, this is known as tracking in mixed coordinates [23]. Since our measurements, range and range-rate, are fixed to be in the sensor coordinate system, but our state will not be, we will be tracking in mixed coordinates.

2.2 Filtering Options

Several types of filters exist for nonlinear object tracking. Two common types are the Extended Kalman Filter (EKF) and the particle filter. However, these two types differ greatly in computational burden with the particle filter being of a much greater complexity. While particle filters are sometimes favored when dealing with nonlinear problems [10, 11], the EKF can provide

comparable, if not better, results, especially when computational complexity is taken into account[12, 13, 14]. Since it is desired that we have a real-time tracker, the EKF was the filter model chosen. In order to properly describe the EKF, its linear basis, the Kalman Filter(KF), must first be described.

2.2.1 Kalman Filter

The Kalman Filter is an optimal filter for linear problems. It can take on two forms: a continuous time form, and a discrete time form. Since the measurements or observations to be used, range and range-rate, are discrete, our description will focus on the discrete time model.

Based on the state space model, we have a state vector $\mathbf{x}(k)$ which denotes the state of the object at time $t = kT$ where k is the sample number and T is the time between samples. The linear function which links $\mathbf{x}(k+1)$ to $\mathbf{x}(k)$ is $\mathbf{F}(k)$, to the system input $\mathbf{u}(k)$ is $\mathbf{G}(k)$ and which links $\mathbf{z}(k+1)$ to $\mathbf{x}(k+1)$ is $\mathbf{H}(k+1)$. When dealing with real systems, some system noise, $\mathbf{v}(k)$, is present along with some measurement noise, $\mathbf{w}(k)$. Both the system noise, also called process noise, and the measurement noise are modeled as Gaussian distributions with zero mean and a chosen standard deviation. From [1], the true state evolution can then be defined as follows:

$$\mathbf{x}(k+1) = \mathbf{F}(k)\mathbf{x}(k) + \mathbf{G}(k)\mathbf{u}(k) + \mathbf{v}(k) \tag{2.12}$$

$$\mathbf{z}(k+1) = \mathbf{H}(k+1)\mathbf{x}(k+1) + \mathbf{w}(k+1). \tag{2.13}$$

With this linear dynamic system model, the discrete-time KF filter estimation equations can be developed and are given in [1]. The prediction portion of the KF, state prediction, state covariance prediction, and measurement prediction,

is given as, respectively,

$$\hat{\mathbf{x}}(k+1|k) = \mathbf{F}(k)\hat{\mathbf{x}}(k|k) + \mathbf{G}(k)\mathbf{u}(k) \tag{2.14}$$

$$\mathbf{P}(k+1|k) = \mathbf{F}(k)\mathbf{P}(k|k)\mathbf{F}(k)^T + \mathbf{Q}(k) \tag{2.15}$$

$$\hat{\mathbf{z}}(k+1|k) = \mathbf{H}(k+1)\hat{\mathbf{x}}(k+1|k), \tag{2.16}$$

where $\hat{\mathbf{x}}$ and $\hat{\mathbf{z}}$ denote the estimates of the state and the measurement, respectively. The matrix $\mathbf{Q}$ maps the estimate of process noise into the state prediction covariance. The notation $k+1|k$ means the state at $k+1$ given the data through time k. The measurement residual can then be calculated as

$$\nu(k+1) = \mathbf{z}(k+1) - \hat{\mathbf{z}}(k+1|k). \tag{2.17}$$

From the measurement residual, one can develop the residual covariance,

$$\mathbf{S}(k+1) = \mathbf{R}(k+1) + \mathbf{H}(k+1)\mathbf{P}(k+1|k)\mathbf{H}(k+1)^T \tag{2.18}$$

where the matrix $\mathbf{R}$ maps the estimate of measurement noise into the residual covariance, and the filter gain is

$$\mathbf{W}(k+1) = \mathbf{P}(k+1|k)\mathbf{H}(k+1)\mathbf{S}(k+1)^{-1}. \tag{2.19}$$

We can now determine the updated state and covariance estimates. The updated state estimate is

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + \mathbf{W}(k+1)\nu(k+1), \tag{2.20}$$

and the updated state covariance is

$$\mathbf{P}(k+1|k+1) = \mathbf{P}(k+1|k) - \mathbf{W}(k+1)\mathbf{S}(k+1)\mathbf{W}(k+1)^{-1}. \tag{2.21}$$

2.2.2 Extended Kalman Filter

From this discrete time linear formation of the KF, the discrete time nonlinear formation of the EKF is based. The development given here is based on [1]. For the state space model for the EKF, the linear equations $\mathbf{F}(k)$ and $\mathbf{G}(k)$ are replaced by one nonlinear function $\mathbf{f}(k, \mathbf{x}(k), u(k))$ and $\mathbf{H}(k)$ is replaced by $\mathbf{h}(k+1, \mathbf{x}(k+1), u(k+1))$,

$$\mathbf{x}(k+1) = \mathbf{f}[k, \mathbf{x}(k), u(k)] \tag{2.22}$$

$$\mathbf{z}(k+1) = \mathbf{h}[k+1, \mathbf{x}(k+1)]. \tag{2.23}$$

To create the state transition matrix, $\mathbf{F}(k)$, necessary to calculating the state prediction covariance, one calculates the Jacobian of $\mathbf{f}[k, \mathbf{x}(k), u(k)]$,$\mathbf{F}(k)$. Analogously, the Jacobian of $\mathbf{h}[k+1, \mathbf{x}(k+1)]$, $\mathbf{H}(k+1)$, models the measurement matrix used to calculate the residual covariance and filter gain. The filter state and measurement prediction equations then become

$$\hat{\mathbf{x}}(k+1|k) = \mathbf{f}[k, \hat{\mathbf{x}}(k), \mathbf{u}(k)] \tag{2.24}$$

$$\hat{\mathbf{z}}(k+1|k) = \mathbf{h}[k+1, \hat{\mathbf{x}}(k+1|k)], \tag{2.25}$$

and the state covariance prediction is

$$\mathbf{P}(k+1|k) = \mathbf{F}(k)\mathbf{P}(k|k)\mathbf{F}(k)^T + \mathbf{Q}(k), \tag{2.26}$$

where

$$\mathbf{F}_k(k) = \left.\frac{\partial \mathbf{f}[k, \mathbf{x}(k), \mathbf{u}(k)]}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}(k|k)}. \tag{2.27}$$

Then, analogously, the residual covariance is calculated as

$$\mathbf{S}(k+1) = \mathbf{R}(k+1) + \mathbf{H}(k+1)\mathbf{P}(k+1|k)\mathbf{H}(k+1)^T \tag{2.28}$$

and

$$\mathbf{H}(k) = \left.\frac{\partial \mathbf{h}[k+1, \mathbf{x}(k+1|k)]}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}(k+1|k)}. \tag{2.29}$$

The rest of the filter equations remain the same with $\mathbf{H}_{k+1}(k+1)$ again replacing $\mathbf{H}(k+1)$ in the filter gain equation.

2.3 Association Techniques

Association techniques have several main parts. First and foremost is the cost function or the metric upon which to base assignment. The cost function is the equation which measures how likely new information is to belong to old information. In this case, old information is a filter track or a set of previously linked together measurements, and new information is the set of measurements at the next measurement time. For the remainder of this discussion, the set of old information will be referred to as a set of tracks and the set of new information is the set of measurements. Next is the type of assignment that is allowed. If new measurements can be assigned to more than one track, then this type of assignment is called Nearest Neighbor (NN) assignment [4, 15]. If new measurements can be assigned to only one track, then this type of assignment is called global nearest neighbor (GNN) assignment [4, 15]. If a weighted sum of the measurements, based on their probability of occurrence, is

assigned to each track, then this type of assignment is called joint probabilistic data association (JPDA) [4, 15]. The third part decides how assignments are maintained across time. In this work, we restrict ourselves to the basic multiple target tracker (MTT) single time assignment. Another type of assignment over time is multiple hypothesis tracking (MHT). The details of MHT are presented in [2, 24].

Two common, and straightforward to implement, metrics are distance and statistical distance [2, 4]. Statistical distance can be defined as a weighted difference between two vectors. For a vector $\mathbf{x}$, $\mathbf{x} = [x_1 \ x_2]^T$, and a vector $\mathbf{y}$, $\mathbf{y} = [y_1 \ y_2]^T$, where $\mathbf{x}$ and $\mathbf{y}$ are both zero mean with a common covariance, $\mathbf{R}$, we can define

$$d_{statistical} = (\mathbf{x} - \mathbf{y})^T \mathbf{R}^{-1} (\mathbf{x} - \mathbf{y}). \tag{2.30}$$

The statistical distance, or its square root, may also be referred to as the Mahalanobis distance [25, 26]. Now that potential cost functions have been discussed, we need to consider the types of assignment possible. The discussion on assignment type ignores the possibilities of false alarms or missed detections and also the topic of track maintenance [24, 27, 28].

2.3.1 Nearest Neighbor

The nearest neighbor (NN) assignment allows a new measurement to be assigned to more than one track. It is generally used with either a distance or statistical distance cost function, hence the name nearest neighbor. For a set of tracks, t_i, for $i = 1 \ldots N$, each has a corresponding measurement prediction $\hat{\mathbf{z}}_i(k+1|\hat{\mathbf{x}}(k))$. This measurement prediction is the measurement estimate at the next measurement time, $k+1$, based on the data and state estimates though time k. There is also a set of measurements at time $k+1$,

$\mathbf{z}_j(k+1)$, for $j = 1 \ldots N$. Since we are assuming there are no false alarms or missed detections, there will be N measurements at each measurement time. We can define an $N \times N$ matrix $\mathbf{A}$ called the cost matrix. The elements of the cost matrix, a_{ij}, are created by applying the cost function to the measurement prediction corresponding to track t_i, $\hat{\mathbf{z}}_i(k+1|\hat{\mathbf{x}}(k))$, and the measurement $\mathbf{z}_j(k+1)$. For example, if there were 5 tracks and 5 measurements, then the row $[a_{31}\ a_{32}\ a_{33}\ a_{34}\ a_{35}]$ expresses the value of the cost function using the measurement prediction for track 3, $\hat{\mathbf{z}}_3(k+1|\hat{\mathbf{x}}(k))$ and each of the 5 measurements. Since we are using distance or statistical distance as the cost function, for each row in the cost matrix we want to minimize the distance between the current track and the new measurements. In other words, we want to pick the element in the row with the smallest value to determine which measurement to assign to the current track. Since assignment is done by comparing each track to all measurements individually, one measurement may be assigned to more than one track. More detail on the probability density functions behind NN assignment can be found in [29].

2.3.2 Global Nearest Neighbor

The global nearest neighbor (GNN) assignment allows each new measurement to be assigned to only one existing track. As with the nearest neighbor assignment, it is generally used with a cost function based on relative distance. Again, for a set of tracks, t_i for $i = 1 \ldots N$, where each has a corresponding measurement prediction, $\hat{\mathbf{z}}_i(k+1|\hat{\mathbf{x}}(k))$, and a set of measurements, $\mathbf{z}_j(k+1)$ for $j = 1 \ldots N$, we wish to find the $N \times N$ cost matrix, $\mathbf{A}$, where the elements of $\mathbf{A}$, a_{ij}, are the values of the cost function evaluated using the measurement prediction $\hat{\mathbf{z}}_i(k+1|\hat{\mathbf{x}}(k))$ and measurement $\mathbf{z}_j(k+1)$. Since only one mea-

surement can be assigned to each track, the cost matrix assignments need to be minimized over the entire set of assignments, instead of just for each row. This can be expressed by the equation [30]:

$$S = \sum_{i=1}^{n} a_{ip_i} \tag{2.31}$$

where the elements of p_i are unique. For example [30], if the cost matrix is

$$A = \begin{bmatrix} 7 & 5 & 11.2 \\ 5 & 4 & 1 \\ 9.3 & 3 & 2 \end{bmatrix} \tag{2.32}$$

and only one measurement can be assigned to one track, then there are 6 possible permutations, p_i: $\{(1,2,3),\ (1,3,2),\ (2,1,3),\ (2,3,1),\ (3,1,2),\ (3,2,1)\}$. The first set indicates that measurement 1, $\mathbf{z}_1(k+1)$, is assigned to measurement prediction 1, $\hat{\mathbf{z}}_1(k+1|\hat{\mathbf{x}}(k))$, measurement 2, $\mathbf{z}_2(k+1)$, is assigned to measurement prediction 2, $\hat{\mathbf{z}}_2(k+1|\hat{\mathbf{x}}(k))$, and measurement 3, $\mathbf{z}_3(k+1)$, is a assigned to measurement prediction 3, $\hat{\mathbf{z}}_3(k+1|\hat{\mathbf{x}}(k))$. These six permutations give, respectively, values of their sum, S: $\{13.0,\ 11.0\ 12.0\ 15.0\ 19.2\ 24.5\}$. Therefore, the solution to mimimize the sum is permutation 2, $(1,3,2)$, giving a sum of 11.0. The GNN algorithm is generally considered superior to the NN algorithm [4, 31].

2.3.3 Joint Probabilistic Data Association

The joint probabilistic data association (JPDA) algorithm allows each track to be assigned to a weighted sum of the new measurements. It can also be used with a cost function based on relative distance. For a set of tracks, t_i for $i = 1 \ldots N$, where each has a corresponding measurement prediction,

$\hat{\mathbf{z}}_i(k+1|\hat{\mathbf{x}}(k))$, and a set of measurements, $\mathbf{z}_j(k+1)$ for $j = 1 \ldots N$, we can define an $N \times N$ matrix $\mathbf{A}$ of all the possible a_{ij} assignments. We can then define a matrix $\mathbf{B}$ with elements b_{ij}

$$b_{ij} = \frac{a_{ij}}{\sum_{j=1}^{N} a_{ij}}. \tag{2.33}$$

The $\mathbf{B}$ matrix is a matrix of probabilities that express how likely each track is to be assigned to each of the measurements. Using the weights in $\mathbf{B}$, we can assign to each track a measurement, $\mathbf{znew}_i(k+1)$, which is a weighted sum of the measurements, $\mathbf{z}_j(k+1)$,

$$\mathbf{znew}_i(k+1) = \sum_{j=1}^{N} b_{ij} \cdot \mathbf{z}_j(k+1) \tag{2.34}$$

and the measurement $\mathbf{znew}_i(k+1)$ is assigned to track t_i. In other words, the JPDA algorithm updates each track by combining all available measurements weighted by their probability of association [15].

Chapter 3

Model, Filter, and Association Scheme Development

Now that we have outlined the technical background, we can focus on this particular application. To begin this work, the scene used first needs to be detailed. Although the problem of tracking an unknown number of targets with wideband radar generally arises for three-dimensional space or air surveillance, a two-dimensional frame was used herein. This is not a wholly unsubstantiated simplification if consideration is given to the type of measurements used. When using both range and Doppler measurements, a two-dimensional frame, relative to one chosen object in the scene, is often used. While relative measurements were not used in this work, the extension could be made.

Now that basic environment has been described, we need to describe the targets used and the type of motion they will undergo. The targets used are squares with each corner of the box representing a scatterer that gives a measurement return. The observed scene will have two of these squares, and therefore eight measurements for each measurement time. The sensor is placed at the origin in Cartesian coordinates. It is assumed that no measurements

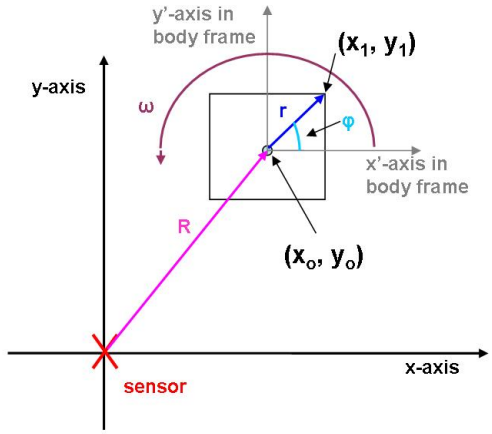

Figure 3.1: The figure above illustrates both the body-centered frame and the object in the Cartesian frame.

are missed for any time and there are no false alarms. The targets will have both translational velocity and rotational velocity but no acceleration. While space targets do have an acceleration term due to gravity, this motion component can generally only be observed over long measurement times, and so the simplification is not without justification.

3.1 State Space Model

The approach taken combines the ideas behind the coordinated turn and the circular maneuver. The nonlinear rotational component is viewed as being in a body-centered reference frame analogous to the maneuver-centered frame. The states used are r, the body moment arm, ω, the turn rate in radians per second, and ϕ, the initial angle of the scatterer on the body. Figure 3.1 illustrates this frame.

Then, similar to the coordinated turn model, the linear translational component can be mapped into x and y components. However, if we break $\mathbf{v}(t)$ into its x and y components prior to this mapping, we can remove the nonlinear

portion of the equation.

$$\dot{x}(t) = v_x \tag{3.1}$$

$$\dot{y}(t) = v_y. \tag{3.2}$$

This then becomes a basic linear state space. So now we have a linear model for the object as a whole and a nonlinear model for spinning motion of the scatterers. Previous models have used two filters to separate this motion [32, 33], but we want to have one filter. If we add these two models together, we will have modeled both the linear and nonlinear portions of the dynamics. Since we are using a two-dimensional frame with targets represented as squares whose corners are scatterers, we can define the target dynamics with the following equations. If (x_0, y_0) is the target's centroid location, (x_1, y_1) is the specific scatterer location, and (v_x, v_y) is the target translational motion in the x and y directions, respectively, then

$$x_0(t) = x_0(0) + v_x(t)t \tag{3.3}$$

$$y_0(t) = y_0(0) + v_y(t)t \tag{3.4}$$

$$x_1(t) = x_0(t) + r\cos(\omega t + \phi) \tag{3.5}$$

$$y_1(t) = y_0(t) + r\sin(\omega t + \phi) \tag{3.6}$$

where $(x_0(0), y_0(0))$ is the target's initial centroid location. Then the continuous time state is

$$\mathbf{x} = [x_0\ y_0\ v_x\ v_y\ \omega\ x_1\ y_1\ r\ \phi]^T \tag{3.7}$$

with

$$\frac{d\mathbf{x}}{dt} = [v_x \ v_y \ 0 \ 0 \ 0 \ (v_x - r\sin(\omega t + \phi)) \ (v_y + r\cos(\omega t + \phi)) \ 0 \ 0]^T . \tag{3.8}$$

Since we can determine the dynamics exactly, but our measurements are discrete, it is best to use a continuous-discrete nonlinear model. We now need to consider the measurement portion on the state space model. Then, for R equal to the range from the radar to the scatterer:

$$R(k+1, \hat{\mathbf{x}}(t+T|t)) = \sqrt{x_1^2 + y_1^2}, \tag{3.9}$$

and RR equal to the range-rate of each scatterer relative to the radar:

$$\begin{aligned} RR(k+1, \hat{\mathbf{x}}(t+T|t)) &= (x_1^2 + y_1^2)^{-1/2}(x_1 \frac{dx_1}{dt} + y_1 \frac{dy_1}{dt}) && (3.10) \\ \frac{dx_1}{dt} &= v_x - \omega r \sin(\omega t + \phi) && (3.11) \\ \frac{dy_1}{dt} &= v_y + \omega r \cos(\omega t + \phi), && (3.12) \end{aligned}$$

the measurement function is:

$$\mathbf{h}(k+1, \hat{\mathbf{x}}(t+T|t)) = [R(k+1, \hat{\mathbf{x}}(t+T|t)) \ RR(k+1, \hat{\mathbf{x}}(t+T|t))]. \tag{3.13}$$

Since we are trying to determine the x and y components from range and range-rate without angle, we have an unobservable problem. We can see this from the following figures. Figure 3.2 shows two circles. The first is at *Range* 1 and the second is at *Range* 2. As one can see, there are multiple combinations of x and y which could give either *Range* 1 or *Range* 2. Without knowing angle, there is no way of knowing how the range maps into the x and y components. From this figure, one can also see that the difference between *Range* 1

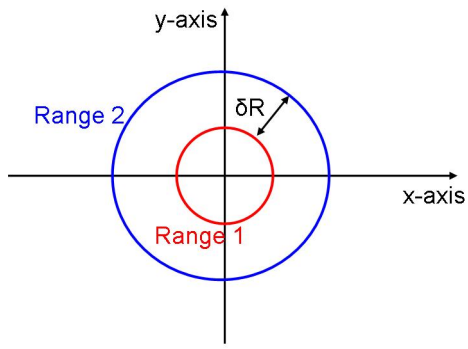

Figure 3.2: The figure above illustrates range and range-rate measurements in a Cartesian coordinate system.

and *Range* 2 is δR. The range-rate is δR. If, as in this case, δR is positive, then the range is increasing and the circle is expanding. If the range-rate were negative, then the range would be decreasing, the circle contracting, and *Range* 2 would be a smaller circle than *Range* 1. This could be seen quantitatively by considering the Fisher Information Matrix. For more information on this topic please see [3]. Now that the dynamics of the target and the measurements in terms of those dynamics have been defined, we can consider the filter model.

3.2 EKF Details

Since we are using a continuous-discrete state space model, it will be best to model this same time component in our filter. In this case, the state prediction equation becomes

$$\frac{d\hat{\mathbf{x}}}{dt} = \mathbf{f}\left[t, \hat{\mathbf{x}}(t)\right] \tag{3.14}$$

and

$$f_1\left(t,\ \hat{\mathbf{x}}(t)\right) \quad = \quad v_x \tag{3.15}$$

$$f_2\left(t,\ \hat{\mathbf{x}}(t)\right) = v_y \tag{3.16}$$

$$f_6\left(t,\ \hat{\mathbf{x}}(t)\right) = v_x - \omega r \sin(\omega t + \phi) \tag{3.17}$$

$$f_7\left(t,\ \hat{\mathbf{x}}(t)\right) = v_y + \omega r \cos(\omega t + \phi) \tag{3.18}$$

and

$$f_3\left(t,\ \hat{\mathbf{x}}(t)\right) = f_4\left(t,\ \hat{\mathbf{x}}(t)\right) = f_5\left(t,\ \hat{\mathbf{x}}(t)\right) = f_8\left(t,\ \hat{\mathbf{x}}(t)\right) = f_9\left(t,\ \hat{\mathbf{x}}(t)\right) = 0. \tag{3.19}$$

The continuous-discrete version of the EKF is derived and discussed in [1, 3, 22], and the final relevant filter equations will be paraphrased here. Given that the above state prediction equation is an ordinary differential equation, we can use a numerical method such as the Runge-Kutta Method to solve for $\hat{\mathbf{x}}(t + \delta t)$ when $t = kT$ and $\delta t = T$ for k equal to an integer and T equal to the sample time.

To find the Jacobian $\mathbf{F}_k(t, \hat{\mathbf{x}}(t))$, again with $t = kT$, we must take partial derivatives of each function in $\mathbf{f}[t, \hat{\mathbf{x}}(t)]$ listed in Equations 3.12 and 3.13. This gives

$$\mathbf{F}_t\left(t,\ \hat{\mathbf{x}}(t)\right) = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & \frac{\partial f_6}{\partial \omega} & 0 & 0 & \frac{\partial f_6}{\partial r} & \frac{\partial f_6}{\partial \phi} \\ 0 & 0 & 0 & 1 & \frac{\partial f_7}{\partial \omega} & 0 & 0 & \frac{\partial f_7}{\partial r} & \frac{\partial f_7}{\partial \phi} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.20}$$

where

$$\frac{\partial f_6}{\partial \omega} = -r\sin(\omega t + \phi) - t\omega r\cos(\omega t + \phi) \tag{3.21}$$

$$\frac{\partial f_6}{\partial r} = -\omega\sin(\omega t + \phi) \tag{3.22}$$

$$\frac{\partial f_6}{\partial \phi} = -\omega r\cos(\omega t + \phi) \tag{3.23}$$

$$\frac{\partial f_7}{\partial \omega} = r\cos(\omega t + \phi) - t\omega r\sin(\omega t + \phi) \tag{3.24}$$

$$\frac{\partial f_7}{\partial r} = \omega\cos(\omega t + \phi) \tag{3.25}$$

$$\frac{\partial f_7}{\partial \phi} = -\omega r\sin(\omega t + \phi). \tag{3.26}$$

We then propagate this linear approximation forward to $t = (k+1)\cdot T$, and, in doing so, effectively discretize it, by finding the matrix exponential,

$$\mathbf{F}_k(k, \hat{\mathbf{x}}(k)) = \exp(\mathbf{F}_t(t, \hat{\mathbf{x}}(t))\cdot T). \tag{3.27}$$

Now that we have linearized and discretized the state prediction covariance, we should consider the linearization of the measurement function $\mathbf{h}(k+1, \hat{\mathbf{x}}(t+T|t))$. As stated previously, this is done by finding the Jacobian $\mathbf{H}_{k+1}(k+1, \hat{\mathbf{x}}(t+T|t))$, just as we found $\mathbf{F}_t(t, \hat{\mathbf{x}}(t))$, where

$$\mathbf{h}(k+1, \hat{\mathbf{x}}(t+T|t)) = [R(k+1, \hat{\mathbf{x}}(t+T|t))\ RR(k+1, \hat{\mathbf{x}}(t+T|t))] \tag{3.28}$$

with

$$h_1(k+1, \hat{\mathbf{x}}(t+T|t)) = R(k+1, \hat{\mathbf{x}}(t+T|t)) \tag{3.29}$$

$$h_2(k+1, \hat{\mathbf{x}}(t+T|t)) = RR(k+1, \hat{\mathbf{x}}(t+T|t)). \tag{3.30}$$

This gives:

$$\mathbf{H}(k+1, \hat{\mathbf{x}}(t+T|t)) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \frac{\partial h_1}{\partial x_1} & \frac{\partial h_1}{\partial y_1} & 0 & 0 \\ 0 & 0 & \frac{\partial h_2}{\partial v_x} & \frac{\partial h_2}{v_y} & \frac{\partial h_2}{\partial \omega} & \frac{\partial h_2}{\partial x_1} & \frac{\partial h_2}{\partial y_1} & \frac{\partial h_2}{\partial r} & \frac{\partial h_2}{\partial \phi} \end{bmatrix} \tag{3.31}$$

where

$$\frac{\partial h_1}{\partial x_1} = x_1(x_1^2 + y_1^2)^{-1/2} \tag{3.32}$$

$$\frac{\partial h_1}{\partial y_1} = y_1(x_1^2 + y_1^2)^{-1/2} \tag{3.33}$$

$$\frac{\partial h_2}{\partial v_x} = x_1(x_1^2 + y_1^2)^{-1/2} \tag{3.34}$$

$$\frac{\partial h_2}{\partial v_y} = y_1(x_1^2 + y_1^2)^{-1/2} \tag{3.35}$$

and

$$\frac{\partial h_2}{\partial \omega} = (x_1^2 + y_1^2)^{-1/2} \cdot$$
$$[-x_1 r \sin(\omega t + \phi) - x_1 \omega r t \cos(\omega t + \phi) + y_1 r \cos(\omega t + \phi) - y_1 \omega r t \sin(\omega t + \phi)] \tag{3.36}$$

$$\begin{aligned} \frac{\partial h_2}{\partial x_1} = x_1(x_1^2 + y_1^2)^{-3/2} \cdot [-x_1 v_x + x_1 \omega r \sin(\omega t + \phi) - y_1 v_y - \\ y_1 \omega r \cos(\omega t + \phi) + \left[(x_1^2 + y_1^2)^{-1/2}(v_x - \omega r \sin(\omega t + \phi))\right] \end{aligned} \tag{3.37}$$

$$\begin{aligned} \frac{\partial h_2}{\partial y_1} = y_1(x_1^2 + y_1^2)^{-3/2} \cdot [-x_1 v_x + x_1 \omega r \sin(\omega t + \phi) - y_1 v_y - \\ y_1 \omega r \cos(\omega t + \phi) + \left[(x_1^2 + y_1^2)^{-1/2}(v_y + \omega r \cos(\omega t + \phi))\right] \end{aligned} \tag{3.38}$$

$$\frac{\partial h_2}{\partial r} = (x_1^2 + y_1^2)^{-1/2} \cdot [-x_1 \omega \sin(\omega t + \phi) + y_1 \omega \cos(\omega t + \phi)] \tag{3.39}$$

$$\frac{\partial h_2}{\partial \phi} = (x_1^2 + y_1^2)^{-1/2} \cdot [-x_1 \omega \cos(\omega t + \phi) - y_1 \omega \sin(\omega t + \phi)] \tag{3.40}$$

3.2.1 Initialization

Now that the filter equations have been outlined, we need to consider its initialization. The first thing to note is that we will collect filter data for a specified length of time. We assume that this time length is of long enough duration to capture one period of the object's spinning motion. The details of object determination and scatterer-to-scatterer assignment are covered in the next section. The linear velocity calculation is similar to that outlined in [34]. The linear velocity is approximated by:

$$v_x = \frac{x_0(t) - x_0(t-T)}{T} \tag{3.41}$$

$$v_y = \frac{y_0(t) - y_0(t-T)}{T} \tag{3.42}$$

and the rotational velocity, ω, by:

$$F_1 = fft(RR_1) \tag{3.43}$$

$$F_2 = fft(RR_2) \tag{3.44}$$

$$F_3 = fft(RR_3) \tag{3.45}$$

$$F_4 = fft(RR_4) \tag{3.46}$$

and then ω_1, ω_2, ω_3, and ω_4 are equal to the frequencies producing the largest peaks in F_1, F_2, F_3, and F_4 for RR_1, RR_2, RR_3, and RR_4 equal to the set of range-rate values corresponding to scatterers 1,2,3, and 4, respectively. The value used for ω will be the mean of ω_1, ω_2, ω_3, and ω_4.

This leaves x_0, y_0, x_1, y_1, r, and ϕ still to be calculated. Since we know the range for each of the four scatterers for all collected times, and which object they belong to, then we can determine the range of the center for all collected

times, R as the mean of R_1, R_2, R_3, R_4.

$$R = \frac{1}{4}\sum_{i=1}^{4} R_i \tag{3.47}$$

By assuming that we know the radar line of sight angle, $\theta(t)$, just prior to starting the filter, we can determine x_0 and y_0 along with x_1 and y_1. This assumption is equivalent to starting the wideband tracker from a previously known position obtained from a narrowband or search radar and will give us a good estimate of the Cartesian position to start the filter on [35].

$$x_0(t) = R(t)\cos(\theta(t)) \tag{3.48}$$

$$y_0(t) = R(t)\sin(\theta(t)) \tag{3.49}$$

Since both the time between samples, T, and the relative difference between the radar line of sight to the center and to each of the scatterers will be small, we can also find

$$x_0(t-T) = R(t-T)\cos(\theta(t)) \tag{3.50}$$

$$y_0(t-T) = R(t-T)\sin(\theta(t)) \tag{3.51}$$

$$x_1(t) = R_1(t)\cos(\theta(t)) \tag{3.52}$$

$$y_1(t) = R_1(t)\sin(\theta(t)), \tag{3.53}$$

and the calculations for scatterers 2, 3, and 4 are similar. We can see this easily from Figure 3.3. Since the angle to the center of the object is θ, given the small size of the object, this same angle can also be used to map the ranges for the four scatterers into its x and y components, x_1 and y_1. Then the two

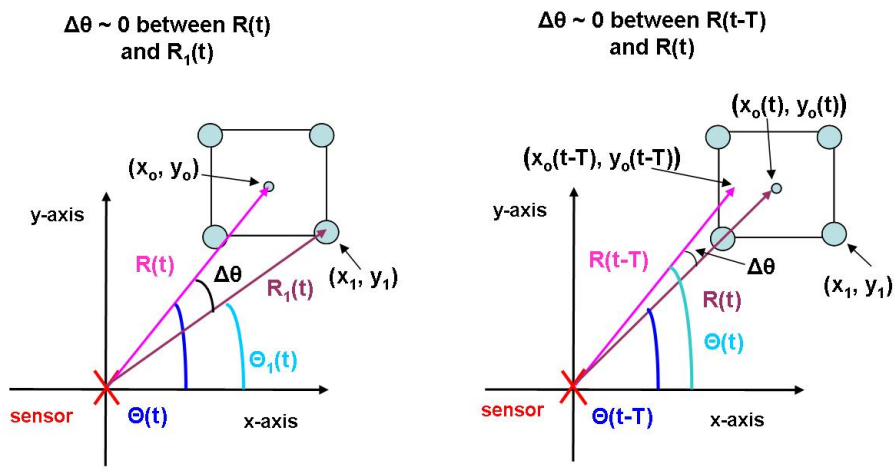

Figure 3.3: The figure above illustrates how x_0, y_0, x_1, and y_1 are initialized.

remaining state parameters, r and ϕ, can easily be determined as

$$r = \frac{1}{4}\sum_{i=1}^{4} max\,|R_i - R| \tag{3.54}$$

$$error_i = (R_1 - R) - r\cos(\omega t + \phi_i) \tag{3.55}$$

for $\phi_i = 0\ to\ 2\pi\ in\ steps\ of\ 0.1$

$$\phi = min(error_i). \tag{3.56}$$

3.3 Association Method Employed

The association scheme was broken up into two parts. The first part deals with association in the filter initialization step, scatterer-to-scatterer assignment, and the second part handles the measurment-to-track association occurring during a filter run. For our association algorithm, we assume that there are no missed detections and no false alarms. The reader is directed to [27]

for a discussion on how to handle false alarms and [28] for a discussion on handling missed detections.

3.3.1 Initial Association

As stated, the initialization portion of the tracker collects data for a specified length of time before beginning the filter. We assume the collection time observes at least one period of the object's rotation. We also assume that the objects are separated in range for the duration of this data collection. With this set of measurements, the first task involves determining the objects in the scene. This determination uses a binning or clustering algorithm with defined thresholds to find objects. While a better algorithm might be found by basing the object determination on the rigid body characteristics of the objects, this algorithm would be complex and time intensive and unlikely to be substantially better than a clustering algorithm with our assumptions. The steps of the clustering algorithm, similar to methods for establishing groups in group tracking [35], are as follows:

1. Define a range threshold value, T_R, and a range-rate threshold value, T_{rate}.

2. Sort the N measurements $\mathbf{z}_i$ by ascending range order so that $\mathbf{z}_1$ has the smallest range value, $\mathbf{z}_1(1)$, and $\mathbf{z}_N$ the largest, $\mathbf{z}_N(1)$.

3. Take $\mathbf{z}_1$ as belonging to Object 1, and test the next closest measurement $\mathbf{z}_2$ to see if the range difference, δR_{12}, is less than or equal to the range threshold: $\delta R_{12} \leq T_R$. If yes, then assign this new measurement as belonging to Object 1 also. If no, then start a new object, Object 2, with $\mathbf{z}_2$.

4. Now check to see if $\mathbf{z}_3$ also belongs to Object 1. For this we need to find δR_{23}. This illustrates the general comparison term we are using, $\delta R_{i,i+1} = \mathbf{z}_{i+1}(1) - \mathbf{z}_i(1)$, for threshold comparison.

5. Since the objects are arranged in range order, once $\delta R_{i,i+1}$ exceeds the threshold we can declare a new object without considering $i + 2 \ldots N$.

6. Repeat Step 3 for the general case until all scatterers have been assigned to an object.

7. Check that the differences in range-rate, $\delta RR_{j,j+1}$, for all scatterers $j = 1 \ldots M - 1$ on an object satisfy $\delta RR_{j,j+1} \leq T_{rate}$.

The final step of checking to make sure the grouped scatterers are within the range-rate bound, T_{rate}, should help to separate objects that are at the same range, if their range-rates are sufficiently separated. However, in the context of our problem, it is unlikely that this range-rate thresholding would be able to aid in distinquishing between scatterers on different objects, and so we included the assumption that the objects are separated in range. Furthermore, since the clustering algorithm will be repeated at each time, it is essential that the objects do not cross in range between measurement times or the algorithm would need to be expanded to deal with this possibility.

Now at each time we have a set of objects and the scatterers assigned to each object. We assume that the objects maintain range order for the data collection time, so we have effectively associated objects to one another for the entire initialization time. However, within each object we are unsure of which measurements go with each scatterer. In other words, we now need to perform a scatterer-to-scatterer assignment for each object.

To do this scatterer-to-scatterer assignment, we will consider the distance between each current scatterer's measurement vector and the measurement vector at the next time. This is not a strict distance function since the measurement vector is in $[units\ units/sec]$. Now that we have the cost function defined, we need to consider the assignment methods that could be used. Since we know based on our assumptions that each scatterer will produce a measurement at the next time, the GNN method will be the best to use, and the NN and JPDA algorithms would not be as useful. Since we are using a distance measure, and we want to assign the current scatterers to closest scatterers in range and range-rate space at the next time, we wish to minimize our total cost. The Munkres algorithm is an optimal assignment algorithm for cost matrix minimization and can aid with this problem [2, 30]. Instead of having to try every permutation of p_i, as in Section 2.3.2, that will minimize the total cost, the Munkres algorithm provides a calculation method [2, 30].

3.3.2 Measurement-to-Track Association

The measurement-to-track assignment is the last portion of the assignment scheme that needs to be discussed. Since we know the noise variance in the measurements we can use the statistical or Mahalanobis distance as the cost function to construct the cost matrix. This equation was given in Section 2.3 as Equation 2.23. Again, we can use the GNN type of assignment using the Munkres algorithm to implement it.

Chapter 4

Experiments

We choose to perform several experiments in order to look at the two aspects of our tracker: the filter and the association technique. The filter experiments consist of two filter runs with one object and two different noise levels. This will show how robust the filter is to noise. We will use the initialization method outlined in Section 3.2.1 and perfect scatterer-to-track association while the filter is running.

4.1 Filter Experiments

To determine how the filter performs we will look at the filter runs for one object with two different levels of noise on the measurements. We assume the noise is Gaussian distributed with zero mean in both the range and range-rate components. For both noise cases, 5 *sec* worth of data was collected for initialization.

4.1.1 Low Noise Case

The measurement noise, in both range and range-rate, was assumed to be Gaussian distributed with zero mean. For the low noise case, $\sigma_R = 0.005\ units$ and $\sigma_{RR} = 0.0025\ units/sec$. If the object was exo-atmospheric and approximately 100km away from the sensor, this percent error would map to $\sigma_R \approx 12m$ and $\sigma_{RR} \approx 6m/s$. The object parameters were

$$x_0(0) = 30\ units \tag{4.1}$$

$$y_0(0) = 30\ units \tag{4.2}$$

$$v_x = -1\ unit/sec \tag{4.3}$$

$$v_y = -1\ unit/sec \tag{4.4}$$

$$\omega = \pi\ rad/sec \tag{4.5}$$

$$r = 2\ units. \tag{4.6}$$

With this object defined, we can see the reason why the FFT of the range-rate was used to find an estimate of ω in the initialization of the filter. The FFT of the range for Scatterer 1, the Scatterer located at $\phi = 45\ degrees$, is shown in Figure 4.1. The FFT of the range-rate for Scatterer 1 is shown in Figure 4.2. As one can see, the FFT of the range-rate shows the rotational component more clearly than the FFT of the range. We could have also used the difference in range between Scatterer 1 and the center of the object. The initial covariance matrix, $\mathbf{P}_{init}$, the process noise matrix, $\mathbf{Q}$, and the measurement noise matrix, $\mathbf{R}$, assumed uncorrelated errors and so their diagonal values were

$$diag(P_{init}) = [5\ 5\ 10\ 10\ \pi\ 10\ 10\ 3\ \pi/4] \tag{4.7}$$

$$diag(Q) = [1\ 1\ 0.01\ 0.01\ 0.001\ 1\ 1\ 0.05\ 0.1] \tag{4.8}$$

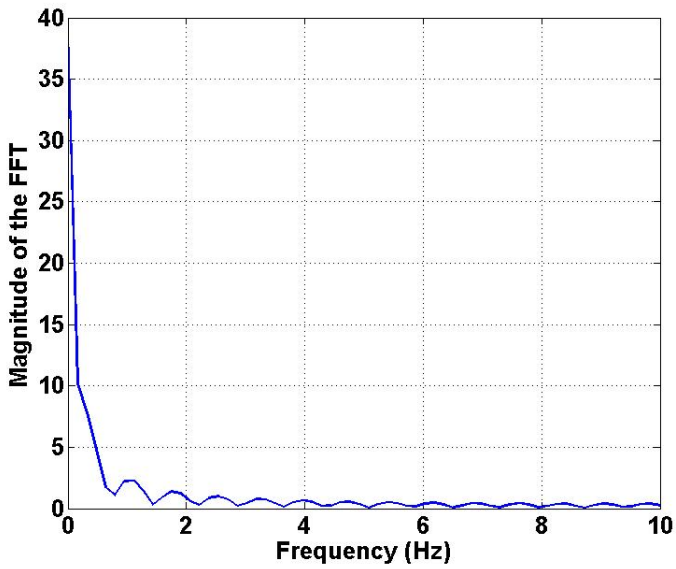

Figure 4.1: Magnitude of the FFT of Scatterer 1's range measurements.

$$diag(R) \quad = \quad 1 \cdot 10^{-4}[0.25\ 0.0625]. \tag{4.9}$$

The sample time, T, was 0.05 sec corresponding to a sampling rate of 20 Hz. Figure 4.3 shows the range of the object over time, and Figure 4.4 below shows the range and range-rate measurements for all four scatterers. As all four scatterers have generally identical results, we will concentrate on the results for Scatterer 1, which is the scatterer at $\phi = \pi/4$. To get a brief understanding of how the filter performs, we can compare the true scatterer movement in the xy-coordinate frame to the filter's estimate of this motion. This is shown in Figure 4.5. One can see qualitatively that the filter is performing rather well. Now we can look at how well the filter estimates each state. The results are shown in Figures 4.6 - 4.14.

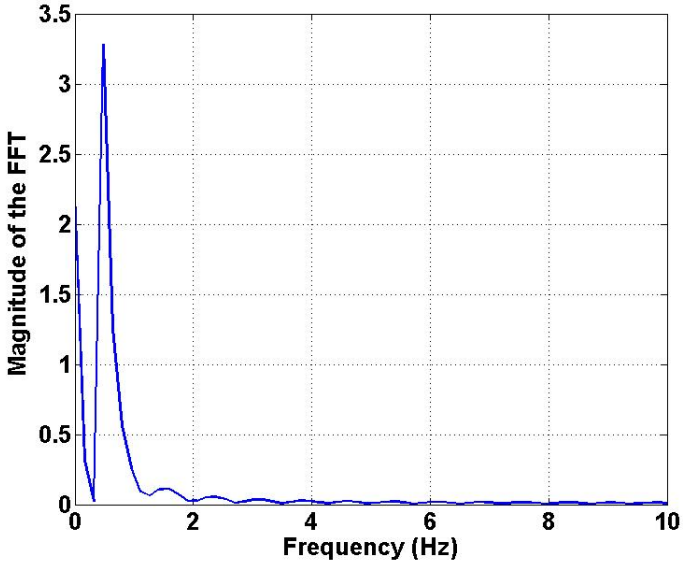

Figure 4.2: Magnitude of the FFT of Scatterer 1's range-rate measurements.

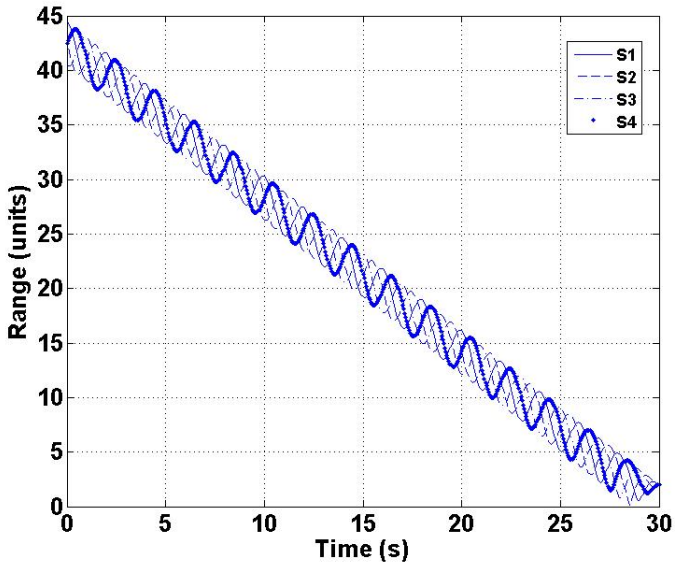

Figure 4.3: Range vs. time for Object 1, low noise case.

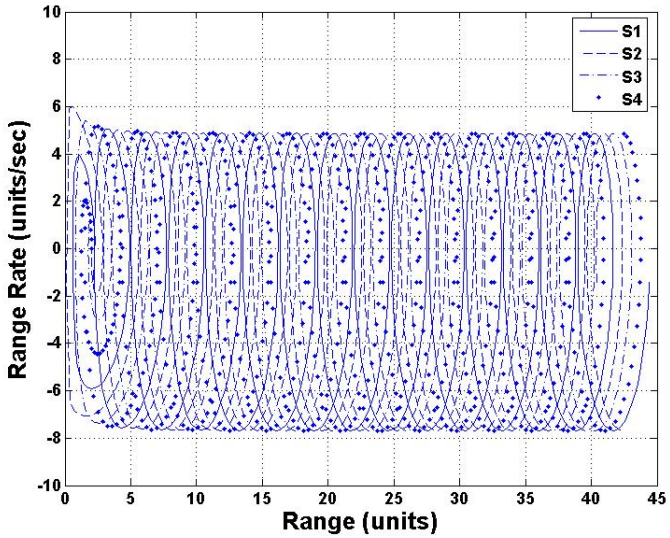

Figure 4.4: Range and range-rate measurements for all times for Object 1 for the low noise case.

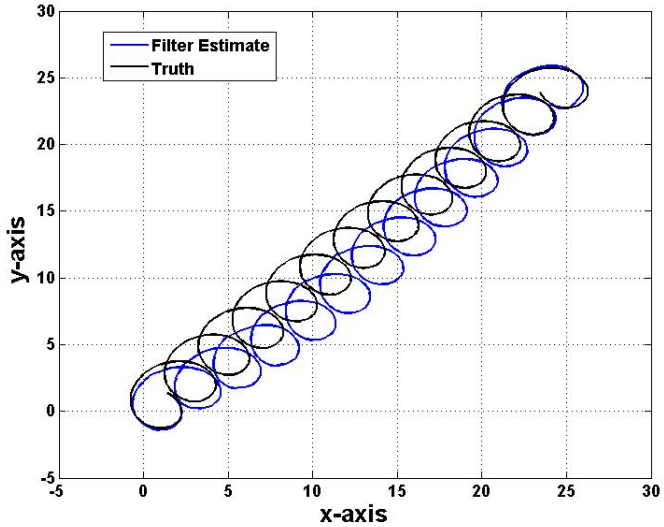

Figure 4.5: This figure shows how the filter's motion estimate for Scatterer 1 compares with truth for Scatterer 1 for the low noise case.

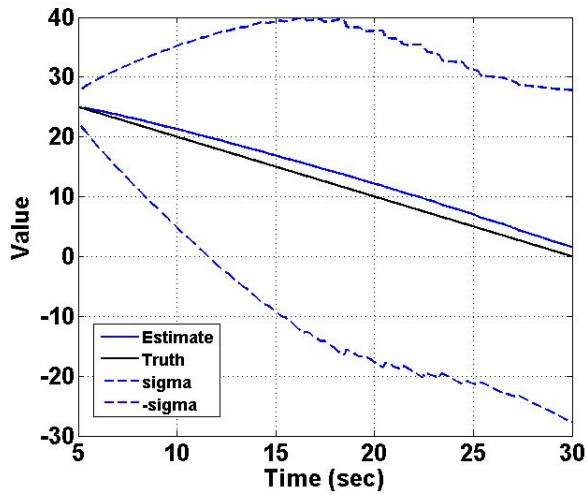

Figure 4.6: This figure shows the filter's estimate of x_0, truth, and $\pm$ one standard deviation for the low noise case.

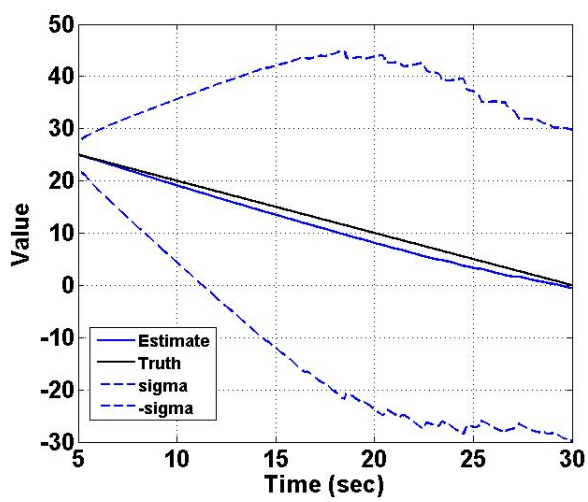

Figure 4.7: This figure shows the filter's estimate of y_0, truth, and $\pm$ one standard deviation for the low noise case.

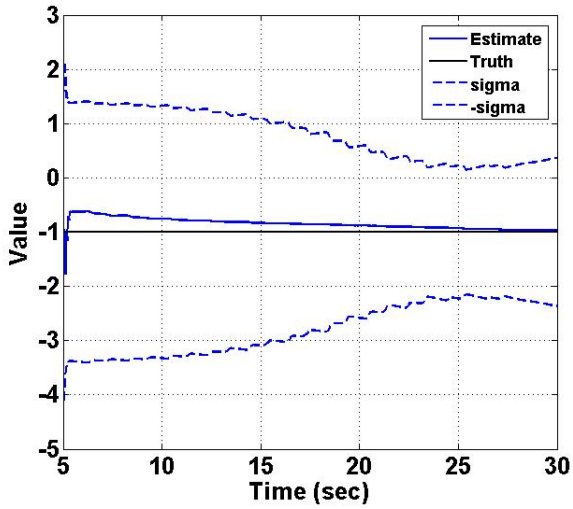

Figure 4.8: This figure shows the filter's estimate of v_x, truth, and $\pm$ one standard deviation for the low noise case.

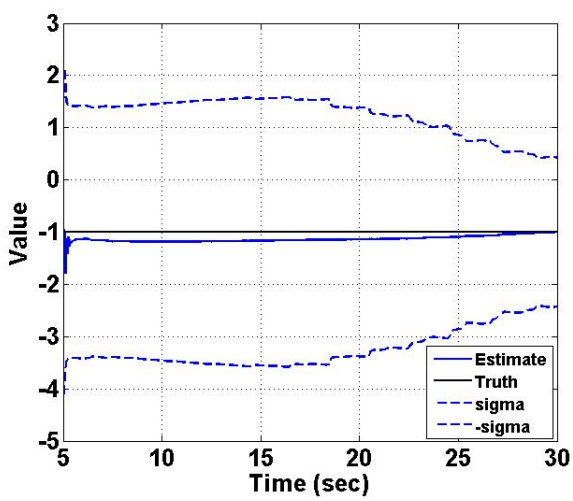

Figure 4.9: This figure shows the filter's estimate of v_y, truth, and $\pm$ one standard deviation for the low noise case.

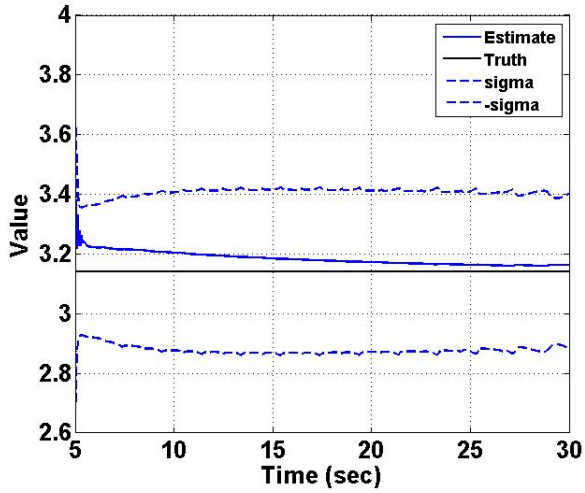

Figure 4.10: This figure shows the filter's estimate of ω, truth, and $\pm$ one standard deviation for the low noise case.

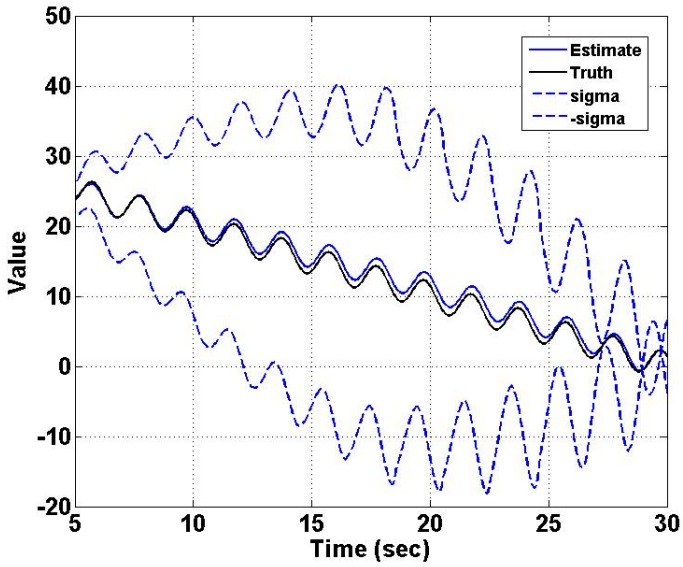

Figure 4.11: This figure shows the filter's estimate of x_1, truth, and $\pm$ one standard deviation for the low noise case.

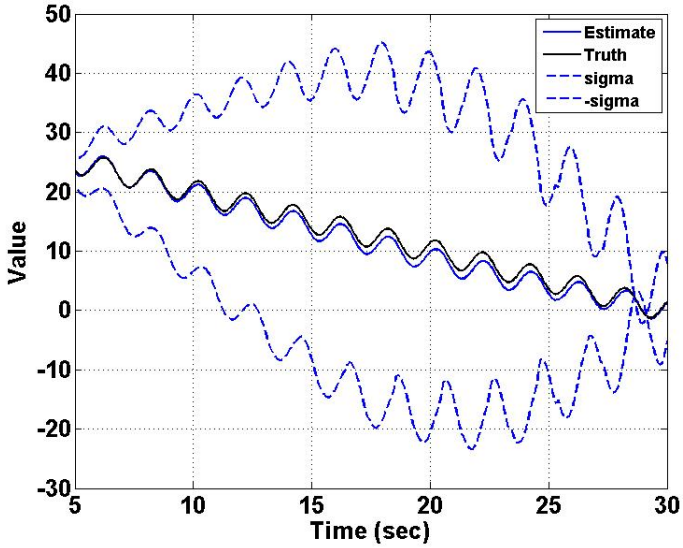

Figure 4.12: This figure shows the filter's estimate of y_1, truth, and $\pm$ one standard deviation for the low noise case.

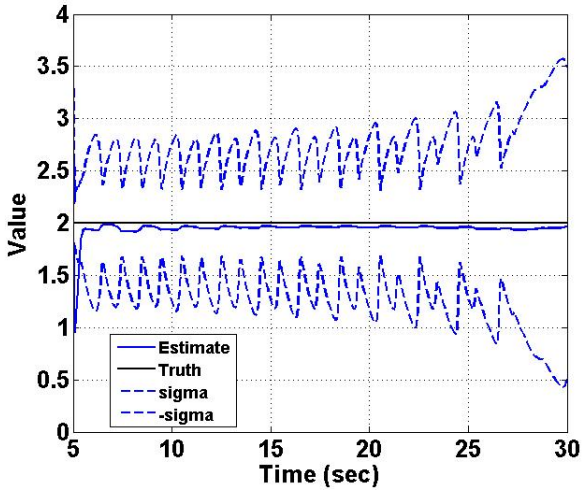

Figure 4.13: This figure shows the filter's estimate of r, truth, and $\pm$ one standard deviation for the low noise case.

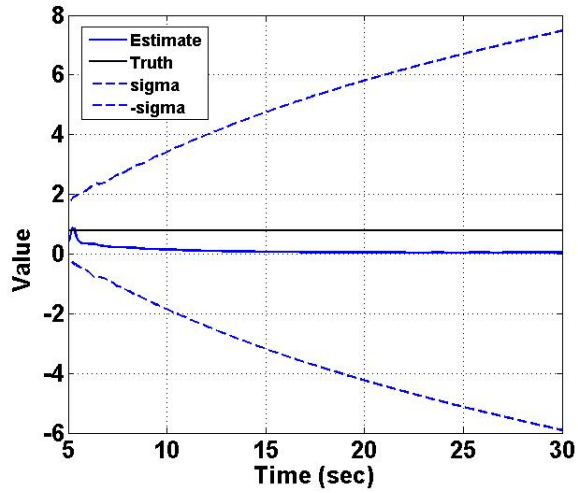

Figure 4.14: This figure shows the filter's estimate of ϕ, truth, and $\pm$ one standard deviation for the low noise case.

4.1.2 High Noise Case

Now for the high noise case. Again we will use Object 1 as described in Equations 4.1-4.6 with a sample rate of 20 Hz. In this high noise case, the measurement noise was again Gaussian with zero mean, and the standard deviations of these distributions, in range and range-rate, are $\sigma_R = 0.1$ $units$ and $\sigma_{RR} = 0.05$ $units/sec$. If the object was exo-atmospheric and approximately 100km away from the sensor, this percent error would map to $\sigma_R \approx 240$ m and $\sigma_{RR} \approx 120$ m/s. The initial covariance, P_{init}, and the process noise matrix, $\mathbf{Q}$, are the same as in Equation 4.2. The new measurement covariance matrix, $\mathbf{R}$, is $diag(R) = [0.0225\ 0.0056]$. The measurements for all four scatterers are shown in Figure 4.16, but as with the low noise case, we will concentrate on only Scatterer 1. The range over time for all four scatterers is shown in Figure 4.15. As one can see in comparing Figures 4.3 and 4.17, the filter's ability to correctly estimate the scatterer motion degrades with increasing noise as would be expected. Figures 4.18-26 show the filter results for each one of the states.

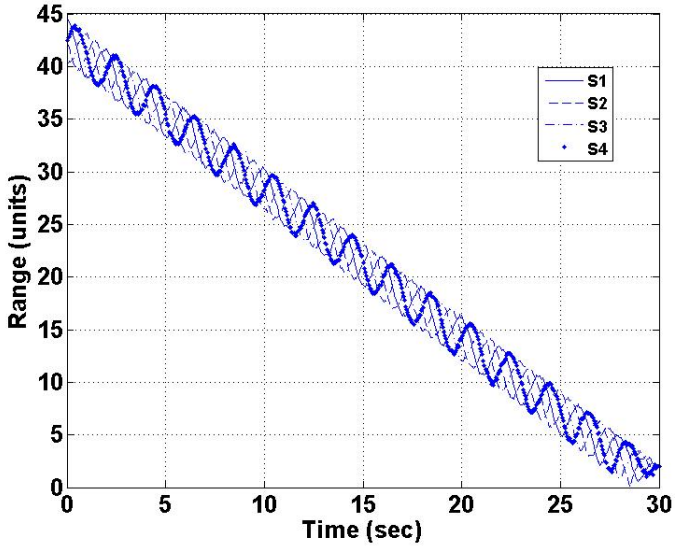

Figure 4.15: Range measurements vs. time for all four scatterers for the high noise case.

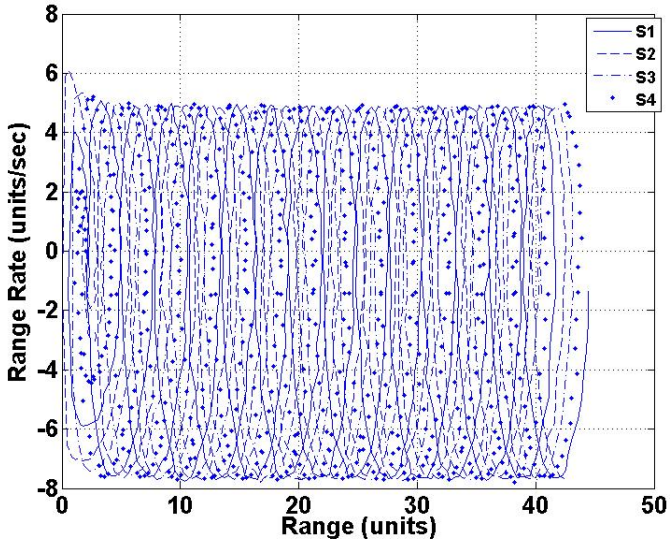

Figure 4.16: Range and range-rate measurements for all four scatterers for the high noise case.

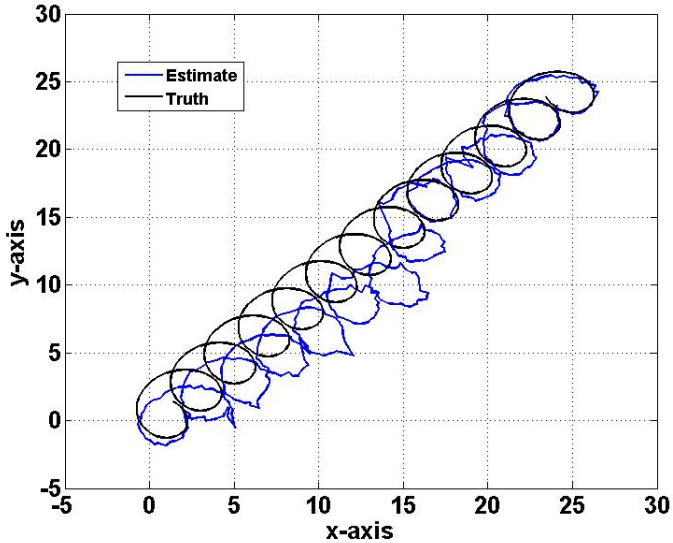

Figure 4.17: Filter estimate compared to truth for Scatterer 1's motion for the high noise case.

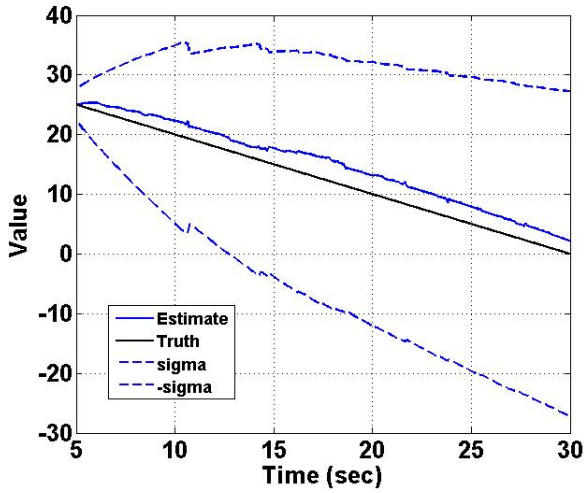

Figure 4.18: This figure shows the filter's estimate of x_0, truth, and $\pm$ one standard deviation for the high noise case.

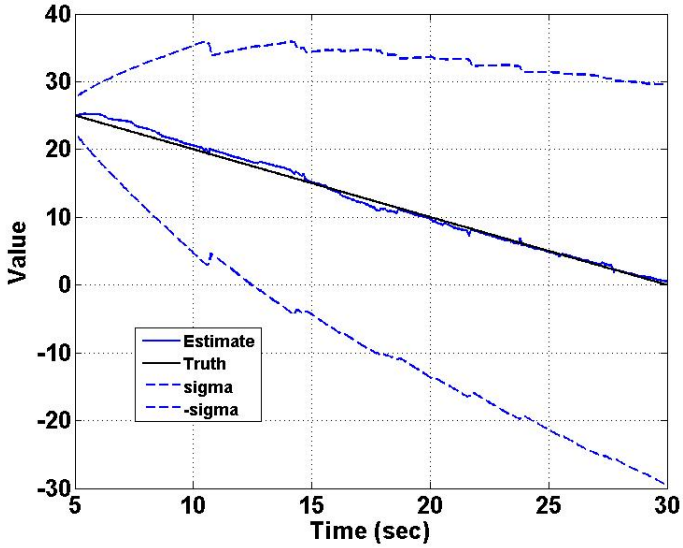

Figure 4.19: This figure shows the filter's estimate of y_0, truth, and $\pm$ one standard deviation for the high noise case.

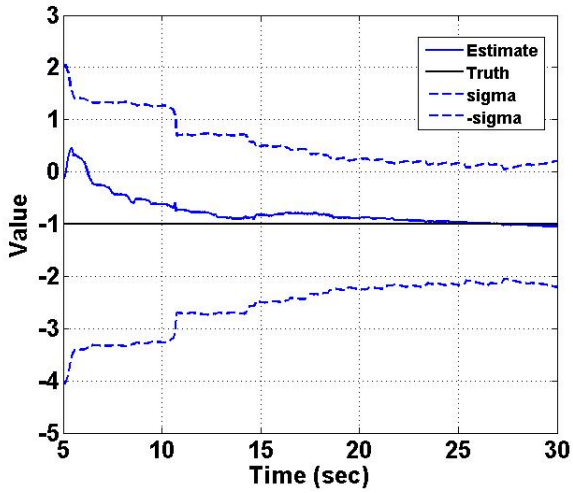

Figure 4.20: This figure shows the filter's estimate of v_x, truth, and $\pm$ one standard deviation for the high noise case.

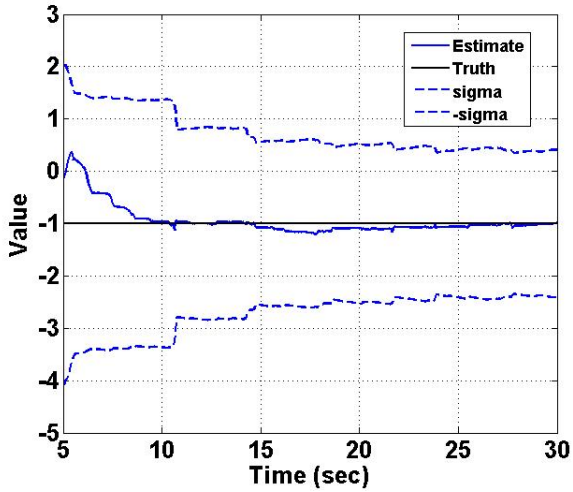

Figure 4.21: This figure shows the filter's estimate of v_y, truth, and $\pm$ one standard deviation for the high noise case.

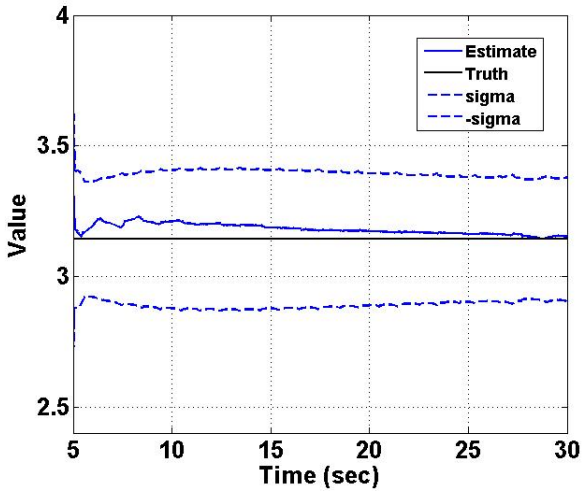

Figure 4.22: This figure shows the filter's estimate of ω, truth, and $\pm$ one standard deviation for the high noise case.

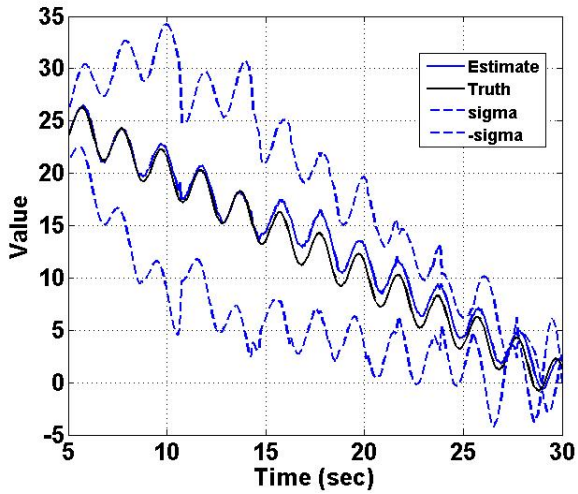

Figure 4.23: This figure shows the filter's estimate of x_1, truth, and $\pm$ one standard deviation for the high noise case.

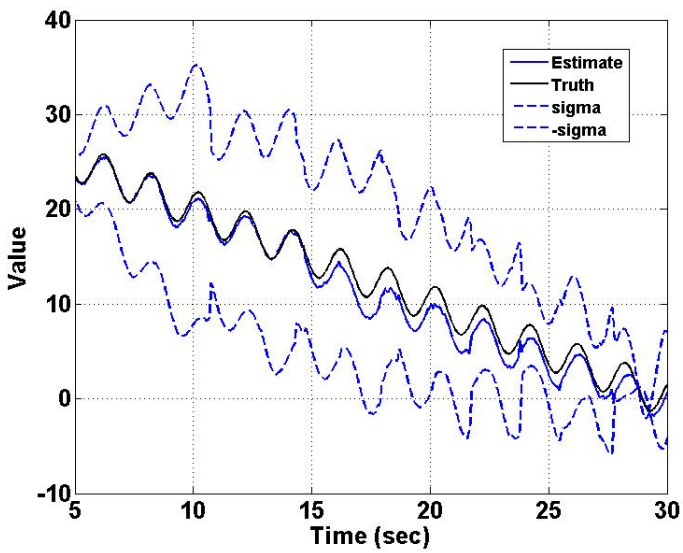

Figure 4.24: This figure shows the filter's estimate of y_1, truth, and $\pm$ one standard deviation for the high noise case.

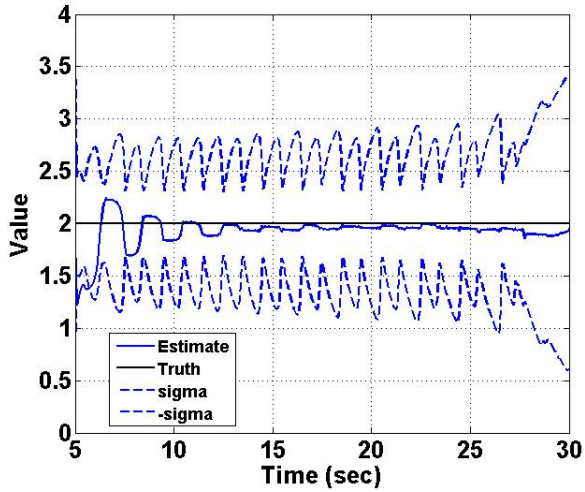

Figure 4.25: This figure shows the filter's estimate of r, truth, and $\pm$ one standard deviation for the high noise case.

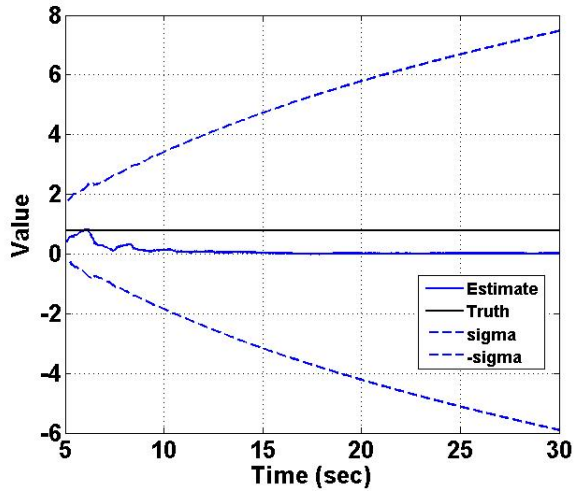

Figure 4.26: This figure shows the filter's estimate of ϕ, truth, and $\pm$ one standard deviation for the high noise case.

We can see from the standard deviations of the covariance results that x_0 and y_0 are unobservable since the covariance is not converging as measurements are added. While the covariances of x_1 and y_1 appear to be converging, this is likely due to the fact that the objects, and therefore the targets (the scatterers), are moving towards the sensor. These two states are also unobservable. From the shape of the covariance of ϕ, it appears that this state is also unobservable.

4.2 Association Technique Experiments

4.2.1 Binning Algorithm

The results of this clustering algorithm are shown in Figure 4.28. The test case used had two objects in the scene with Object 1 defined in Equations 4.1-4.6 in Section 4.1.1 and Object 2 defined as

$$x_0(0) = 75\ units \tag{4.10}$$

$$y_0(0) = 50\ units \tag{4.11}$$

$$v_x = -2\ units/sec \tag{4.12}$$

$$v_y = -1\ units/sec \tag{4.13}$$

$$\omega = 2\pi\ rad/sec \tag{4.14}$$

$$r = \sqrt{2}\ units. \tag{4.15}$$

The high noise case from Section 4.1.2 was used to noise up the measurements. The scene to use for initialization is shown in Figure 4.27. A sample rate of 20 Hz was again used, and 5 sec worth of data was collected for initialization.

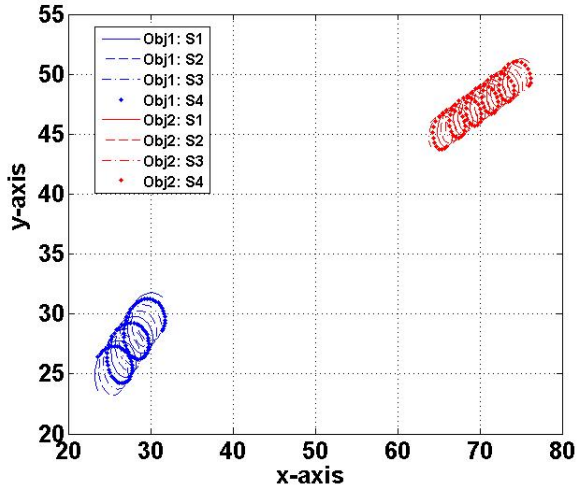

Figure 4.27: This figure shows the scene used to illustrate the binning algorithm.

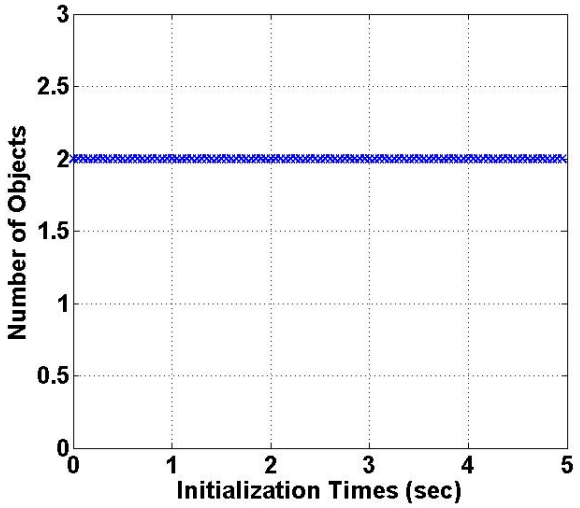

Figure 4.28: This figure shows the binning results for the initialization portion of the association algorithm for the high noise case.

4.2.2 Scatterer-to-Scatterer Assignment

The next portion of the assignment algorithm to be considered is the scatterer-to-scatterer assignment. For Object 1 as defined in Equations 4.1-4.6, Table 4.1 shows the percent of correct results for this assignment. The high percentages along the diagonal show that the distance cost function with the GNN assignment via the Munkres algorithm is working well.

4.2.3 Measurement-to-Track Assignment

As with the scatterer-to-scatterer association, to evaluate the measurement-to-track algorithm, we need to look at the percentage of assignments between scatterers. Table 4.2 shows these results for the high noise case for Object 1 defined as in Equations 4.1-4.6 and Object 2 defined as in Equations 4.10-4.15. These results have been organized so that correct assignments are made along the diagonal. As we can see from the high percentages along the diagonal, the use of statistical distance as the cost function with the GNN assignment via the Munkres algorithm is working well.

4.3 Tracker Experiments

Now that we have seen how the filter and the association method work separately, we need to see how well they work together. Based on the fact that both seem to be working well separately, it is likely that using them together, as a tracker, we will get good results. As with the filter performance results, we will only look at the results for Scatterer 1 on Object 1 and Scatterer 1 on Object 2.

Label	M1	M2	M3	M4	M5	M6	M7	M8
S1	100	0	0	0	0	0	0	0
S2	0	100	0	0	0	0	0	0
S3	0	0	100	0	0	0	0	0
S4	0	0	0	100	0	0	0	0
S5	0	0	0	0	100	0	0	0
S6	0	0	0	0	0	100	0	0
S7	0	0	0	0	0	0	100	0
S8	0	0	0	0	0	0	0	100

Table 4.1: Table of Scatterer-to-Scatterer Assignment Percentages for High Noise Case

Label	M1	M2	M3	M4	M5	M6	M7	M8
T1	99.8	0	0	0.1996	0	0	0	0
T2	0.1996	99.8	0	0	0	0	0	0
T3	0	0.1996	99.6	0.1996	0	0	0	0
T4	0	0	0.3992	99.6	0	0	0	0
T5	0	0	0	0	99.6	0	0.1996	0.1996
T6	0	0	0	0	0.3992	99.6	0	0
T7	0	0	0	0	0	0.3992	99.4	0.1996
T8	0	0	0	0	0	0	0.3992	99.6

Table 4.2: Table of Measurement-to-Track Assignment Percentages for High Noise Case

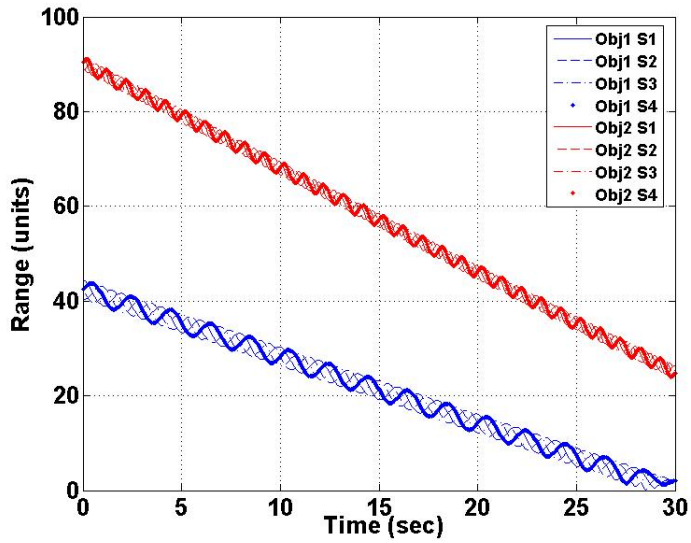

Figure 4.29: Range measurements vs. time for all four scatterers for both objects.

4.3.1 Low Noise Case

We first consider the low noise case. Object 1 and Object 2 are the same as defined in Equations 4.1-4.6 and 4.10-4.15, and the low noise case is the same as in Section 4.1.1. The range over time and the measurements for all times are shown in Figures 4.29 and 4.30, respectively. Again, to get a quick feel for how the filter is performing, we can look at the true scene and the tracker's estimate of the scene together. We can see from the quick view that the tracker is performing well for Object 1. It seems to capture the dynamics of Object 2 well, but there is an offset from the actual location. To get a more complete picture, we can look at the tracker performance for each state individually. We will consider each state for both Objects in turn.

For Object 1, the tracker starts off well for the center location in the x-coordinate, x_0. The estimate then departs from truth by a slight bias, and this bias is held close to constant for most of the remaining tracking time. As expected, since x_0 is an unobservable state from range and range-rate

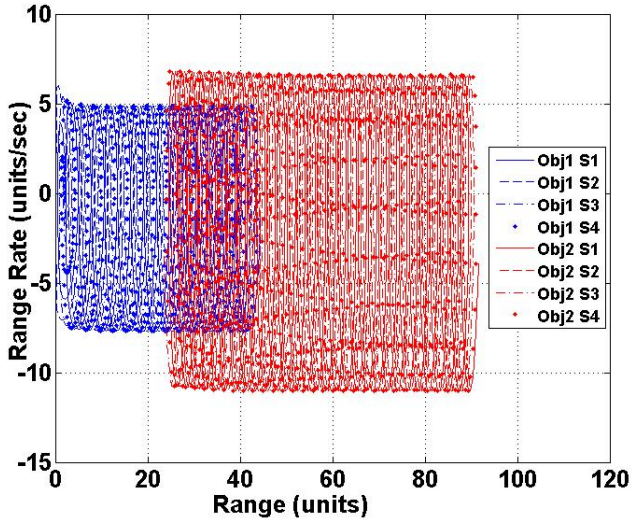

Figure 4.30: Range and range-rate measurements for all four scatterers for both objects for the low noise case.

measurements, the covariance is diverging for most of the run. As the estimate improves slightly, the covariance contracts slightly, but then begins to diverge again. For Object 2, the tracker has an almost immediate bias in its estimate of x_0 and the covariance is either diverging or remaining constant at a large value for this tracker run. The results for y_0 for Objects 1 and 2 are fairly similar. In both cases, the diverging covariance improves slightly as the estimates improve, although the estimate for Object 2 starts off poor and improves, and the estimate for Object 1 starts out close to truth, starts to move away, and then is able to regain an estimate at almost truth. The next results are those for the linear velocity components: v_x and v_y. The tracker estimate for v_x for Object 1 has trouble initially, but quickly begins to approach the true value. The covariance begins to converge as the estimate improves. The tracker estimate for v_x for Object 2 also has trouble initially but quickly converges to truth. The covariance has more trouble converging but begins to as the estimate gets close to truth. The covariance for v_y for both Objects 1 and 2 looks similar: as

the measurement time increases, the covariance converges. This convergence coincides with an improved estimate in v_y for both objects. The results for ω are also very similar for both objects. Initially the estimate is off, but as the number of measurements increases, the estimate converges to truth. The covariances are initially large but converge quickly. The estimates for x_1 are very different for the two objects. For Object 1, the estimate of x_1 initially follows truth well, moves away slightly, and then begins to follow truth again. The covariance is initially small and then diverges. The covariance begins to decrease as the object approaches the sensor at the origin. In fact, at the end of the measurement times, when the covariances decrease to almost zero, this is because the center of the object is almost exactly at the origin. One can see this by checking the final locations of x_0 and y_0 for Object 1. The results for y_1 for Object 1 are similar. For Object 2, the estimate of x_1 is off by an almost constant bias amount and the covariance diverges. The results for y_1 for Object 2 are similar. The estimate of r is very similar for both Objects 1 and 2, although the covariances are different. The covariance of r for Object 1 is fairly settled until the object reaches the origin, the location of the sensor. Then the covariance begins to diverge. For Object 2, the covariance settles quickly and maintains this performance. The final state to consider is ϕ. The tracker results are very similar for both objects. The tracker estimate quickly converges to a value at a small off-set from truth and the covariance diverges.

4.3.2 High Noise Case

Since things look promising for the low noise case, we need to see how things look for the high noise case. Object 1 and Object 2 are the same as defined in Equations 4.1-4.6 and 4.3, and the high noise case is the same as

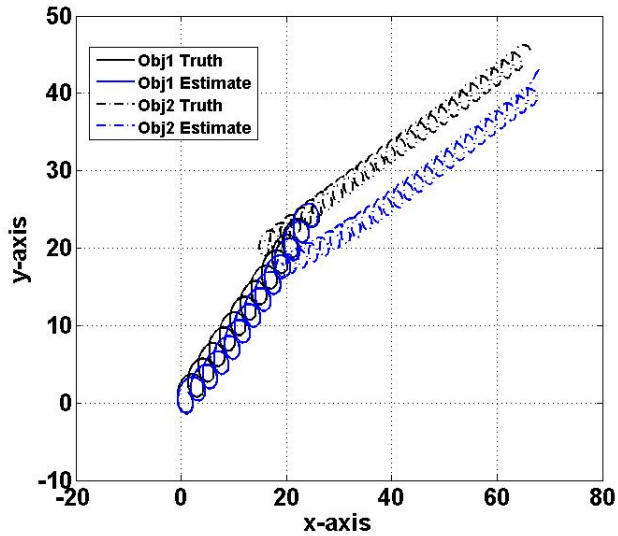

Figure 4.31: Tracker estimate compared to truth for Scatterer 1's motion for both objects.

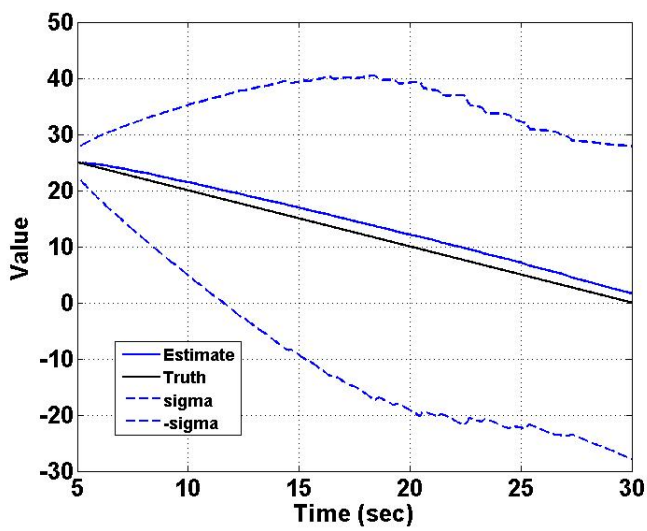

Figure 4.32: This figure shows the tracker's estimate for Object 1 of x_0, truth, and $\pm$ one standard deviation for the low noise case.

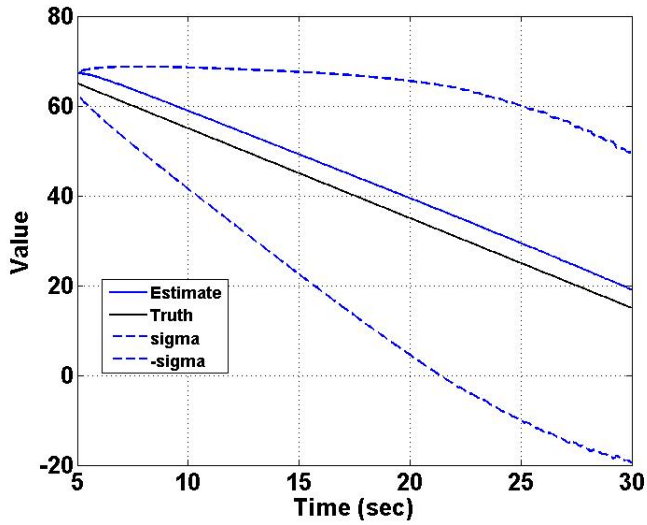

Figure 4.33: This figure shows the tracker's estimate for Object 2 of x_0, truth, and $\pm$ one standard deviation for the low noise case.

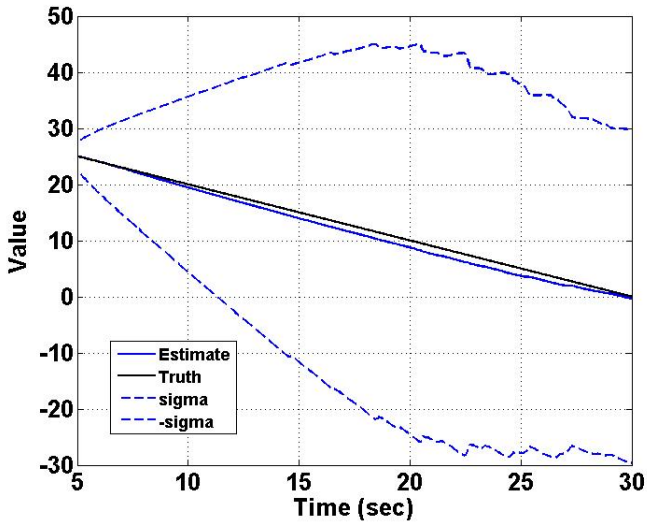

Figure 4.34: This figure shows the tracker's estimate for Object 1 of y_0, truth, and $\pm$ one standard deviation for the low noise case.

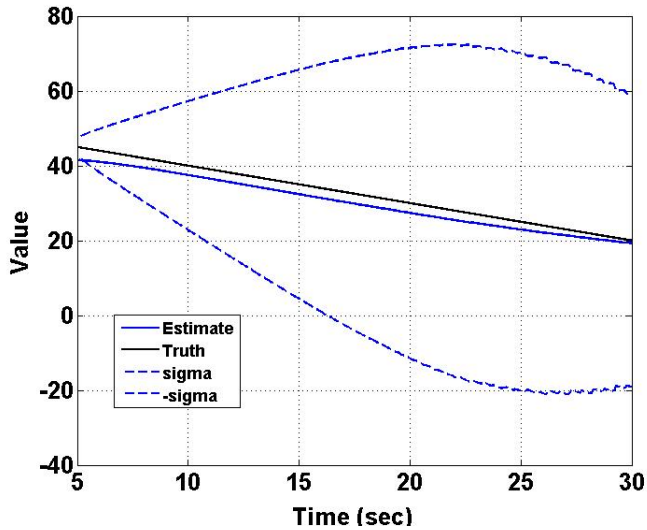

Figure 4.35: This figure shows the tracker's estimate for Object 2 of y_0, truth, and $\pm$ one standard deviation for the low noise case.

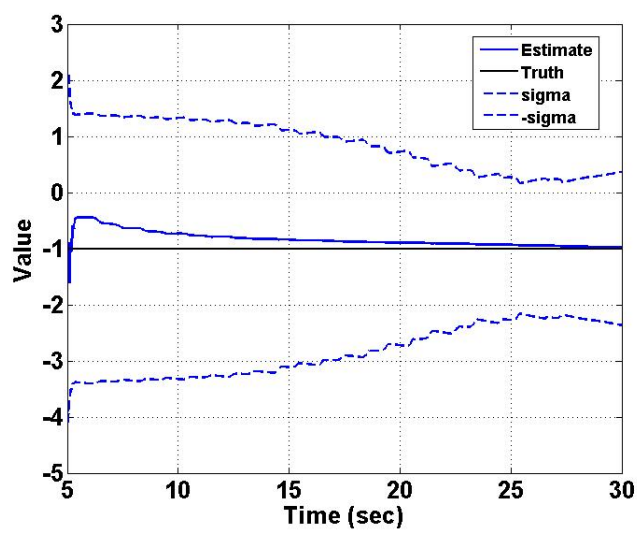

Figure 4.36: This figure shows the tracker's estimate for Object 1 of v_x, truth, and $\pm$ one standard deviation for the low noise case.

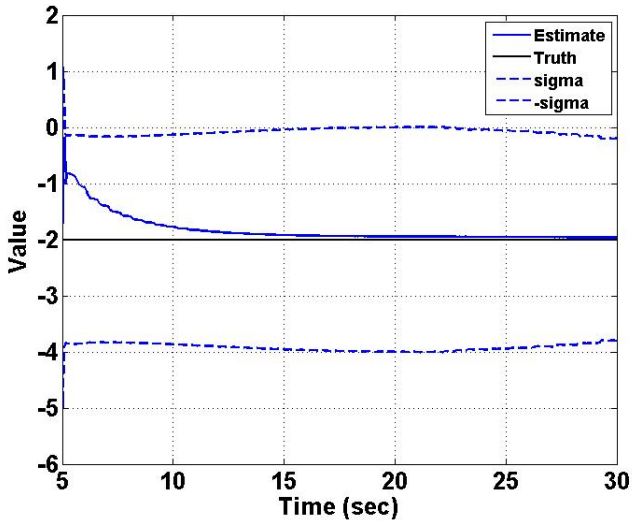

Figure 4.37: This figure shows the tracker's estimate for Object 2 of v_x, truth, and $\pm$ one standard deviation for the low noise case.

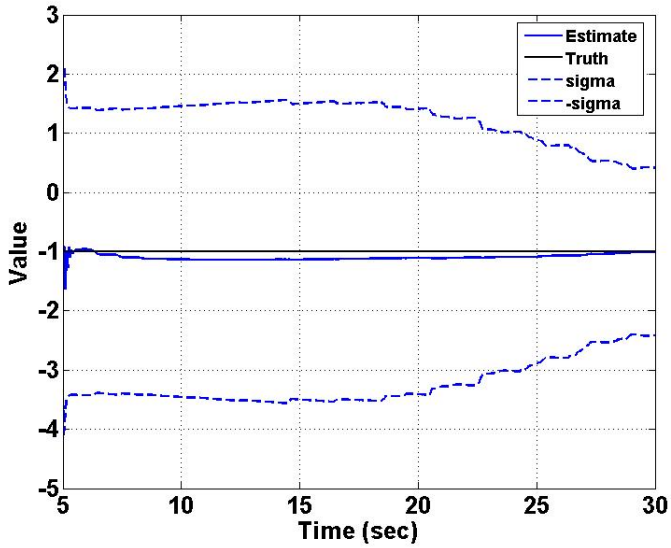

Figure 4.38: This figure shows the tracker's estimate for Object 1 of v_y, truth, and $\pm$ one standard deviation for the low noise case.

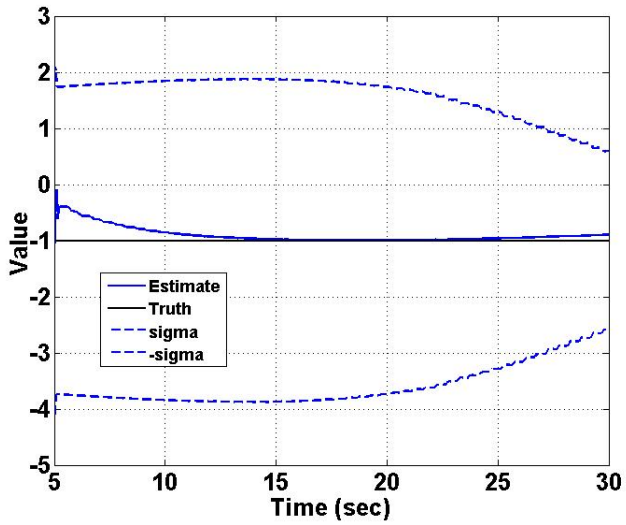

Figure 4.39: This figure shows the tracker's estimate for Object 2 of v_y, truth, and $\pm$ one standard deviation for the low noise case.

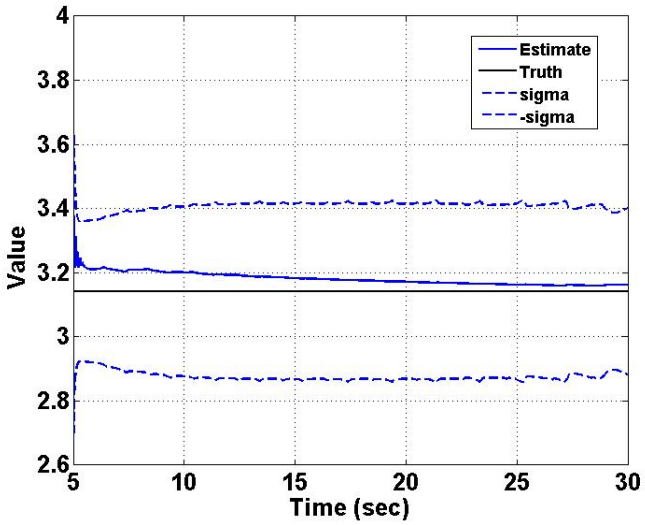

Figure 4.40: This figure shows the tracker's estimate for Object 1 of ω, truth, and $\pm$ one standard deviation for the low noise case.

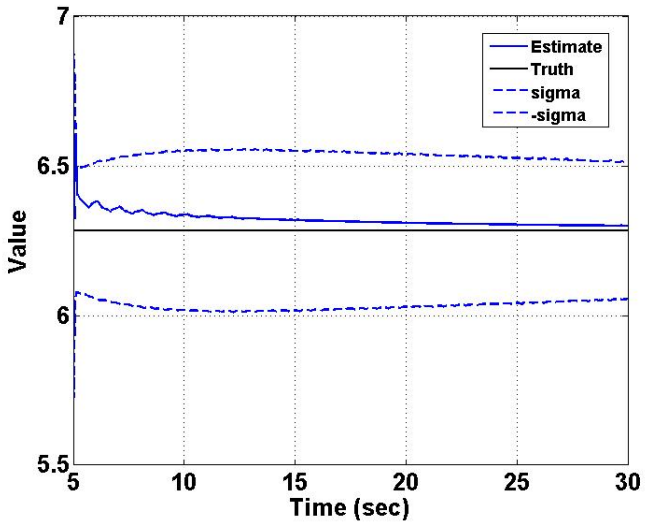

Figure 4.41: This figure shows the tracker's estimate for Object 2 of ω, truth, and $\pm$ one standard deviation for the low noise case.

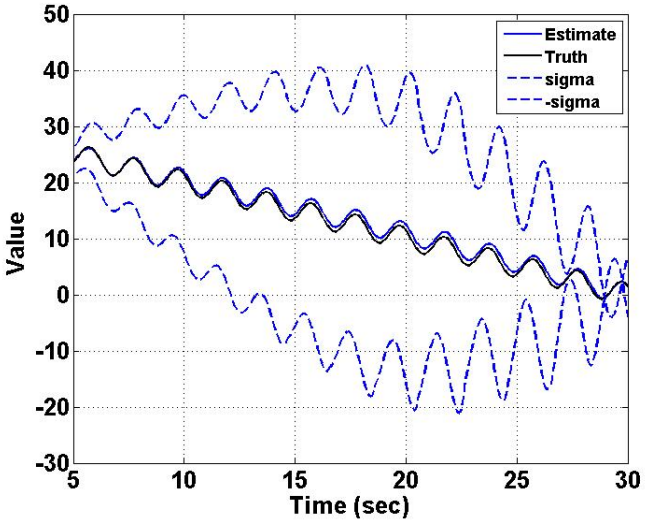

Figure 4.42: This figure shows the tracker's estimate for Object 1 of x_1, truth, and $\pm$ one standard deviation for the low noise case.

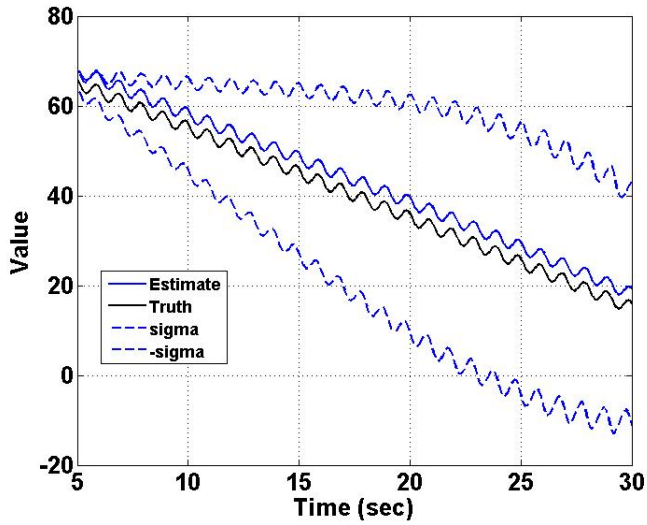

Figure 4.43: This figure shows the tracker's estimate for Object 2 of x_1, truth, and $\pm$ one standard deviation for the low noise case.

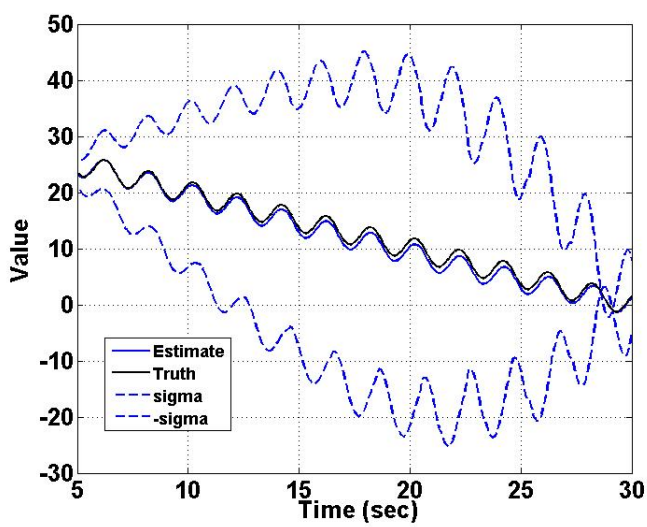

Figure 4.44: This figure shows the tracker's estimate for Object 1 of y_1, truth, and $\pm$ one standard deviation for the low noise case.

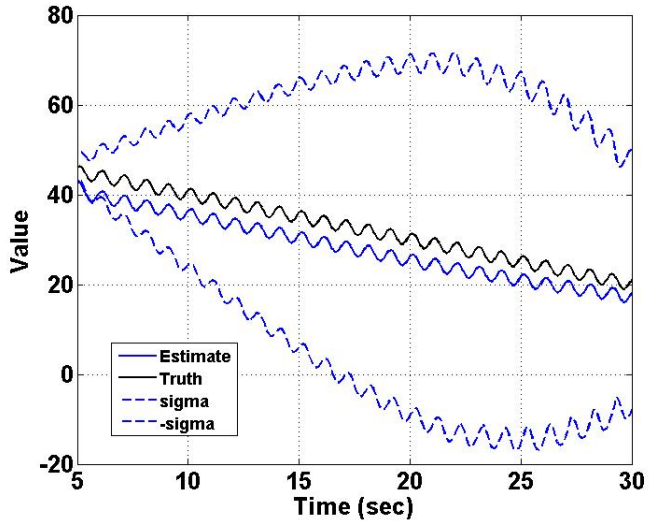

Figure 4.45: This figure shows the tracker's estimate for Object 2 of y_1, truth, and $\pm$ one standard deviation for the low noise case.

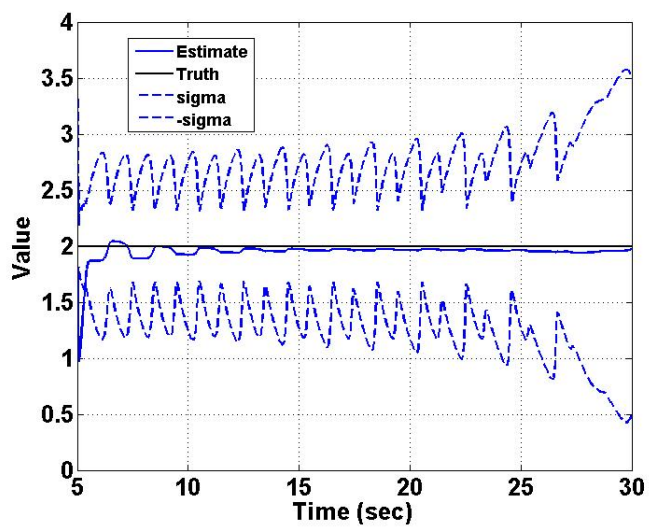

Figure 4.46: This figure shows the tracker's estimate for Object 1 of r, truth, and $\pm$ one standard deviation for the high noise case.

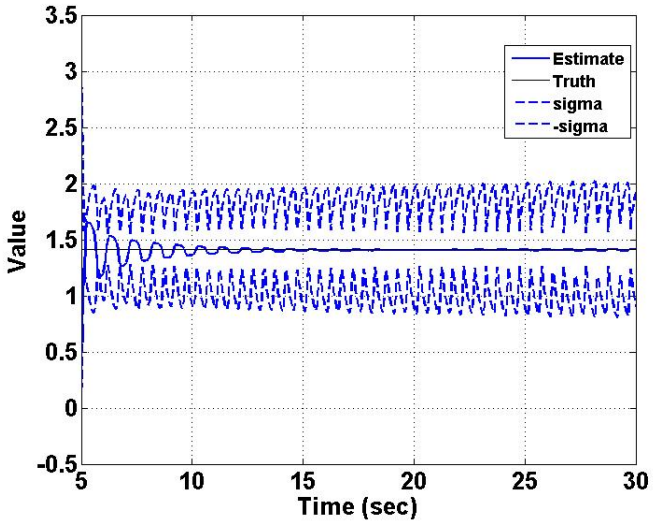

Figure 4.47: This figure shows the tracker's estimate for Object 2 of r, truth, and $\pm$ one standard deviation for the low noise case.

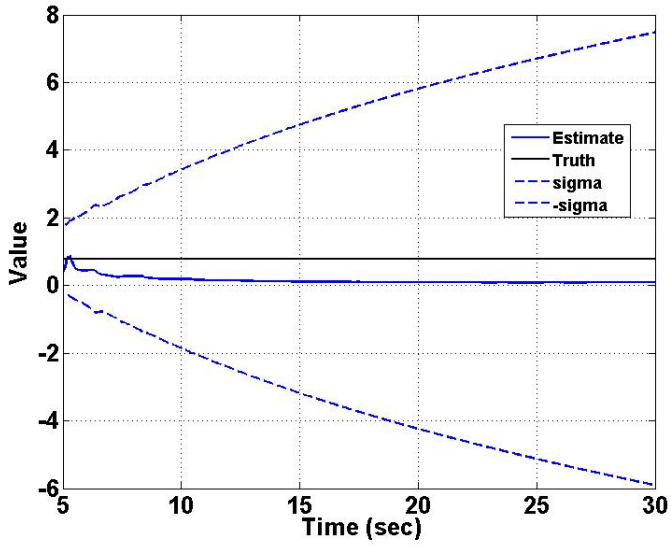

Figure 4.48: This figure shows the tracker's estimate for Object 1 of ϕ, truth, and $\pm$ one standard deviation for the low noise case.

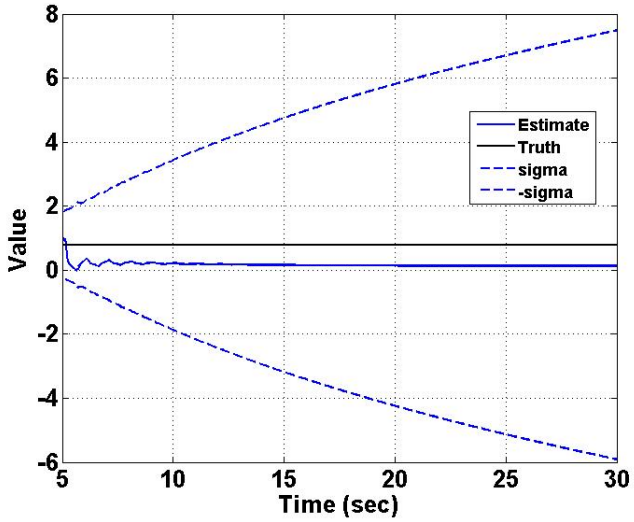

Figure 4.49: This figure shows the tracker's estimate for Object 2 of ϕ, truth, and $\pm$ one standard deviation for the low noise case.

in Section 4.1.2. The range over time and the measurements for all times are shown in Figures 4.50 and 4.51, respectively. Again, to get a quick feel for how the filter is performing, we can look at the true scene and the tracker's estimate of the scene together. We can see that the tracker starts out well for both objects. Then the performance begins to degrade for both objects. As more measurements are added, the tracker is able to recover and begins to get more accurate estimates. We can see this most obviously for Object 2. Now we can look at the actual states to get a feel for what is happening.

For Object 1, the estimate of x_0 starts out well, diverges slightly, and then begins to converge to truth and the covariance diverges. For Object 2, the estimate of x_0 starts out at an off-set, gets closer to truth, and then begins to diverge slightly. The covariance for x_0 for Object 2 diverges. The estimate for y_0 for Object 1 starts off well, diverges a bit, and then converges back to truth. The covariance diverges. For Object 2, the estimate of y_0 starts off poor but then converges to truth while the covariance diverges. The estimate

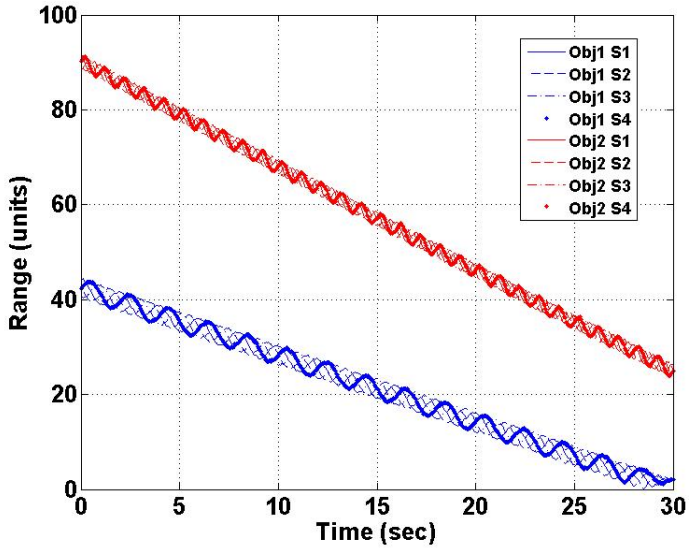

Figure 4.50: Range measurements vs. time for all four scatterers for both Objects 1 and 2.

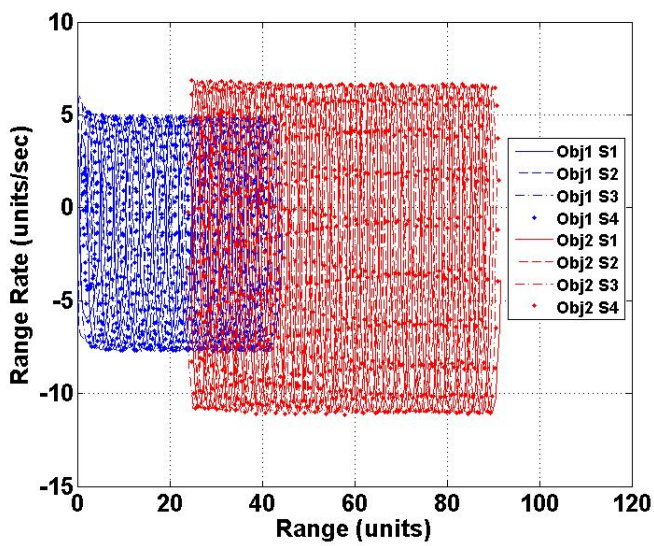

Figure 4.51: Range and range-rate measurements for all four scatterers for both objects for the high noise case.

of v_x converges quickly to truth and the covariance also converges for Object 1. For Object 2, the estimate of v_x converges to truth and the covariance also converges. The results for v_y for Objects 1 and 2 are similar: the estimates and the covariance converge. The estimate of ω for Object 1 converges as does the covariance. For Object 2, the estimate and covariance of ω also converge. The next state to consider is x_1 for Objects 1 and 2. For Object 1, the estimate starts out well, but begins to diverge away from truth. As the number of measurements increases, the estimate seems to slightly improve. The covariance initially diverges but begins to converge as the object approaches the sensor. Once the object is almost on top of the sensor, the covariance converges to almost zero. The results for y_1 for Object 1 are similar. For Object 2, the estimate initially starts off at an off-set but begins to converge to truth. When this happens, the covariance, which was initially diverging, begins to converge. As the estimate then begins to diverge again, the covariance again begins to diverge. The results for y_1 for Object 2 are similar. For Object 1, the tracker estimate of r converges to truth. The covariance initially converges, but as the object gets closer to the sensor, the covariance for this state diverges. For Object 2, the tracker estimate of r converges and the covariance also settles. The results for ϕ are similar for Objects 1 and 2. In both cases, the tracker estimate for ϕ converges to a value at an off-set from truth. This off-set is larger than the off-set in the low noise case. And in both cases, the covariance diverges.

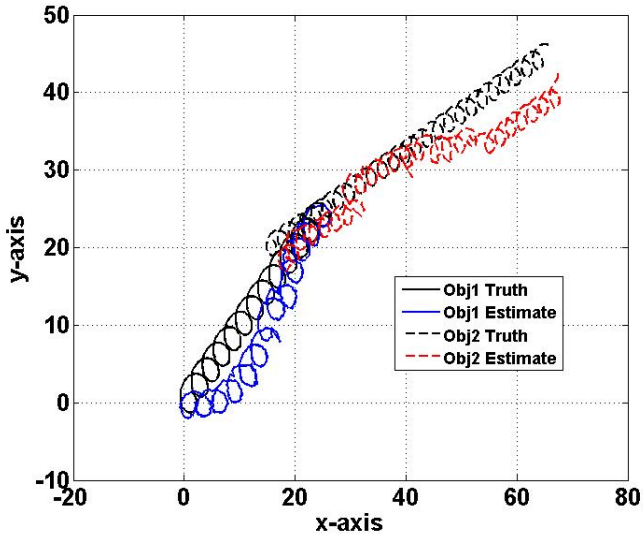

Figure 4.52: Tracker estimate compared to truth for Scatterer 1's motion for Objects 1 and 2.

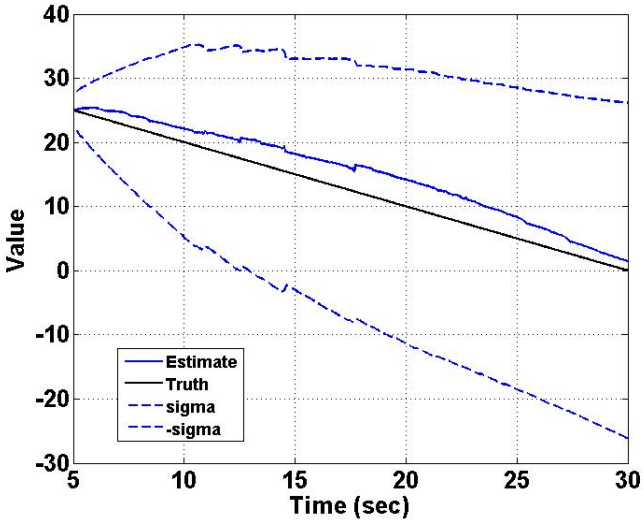

Figure 4.53: This figure shows the tracker's estimate for Object 1 of x_0, truth, and $\pm$ one standard deviation for the high noise case.

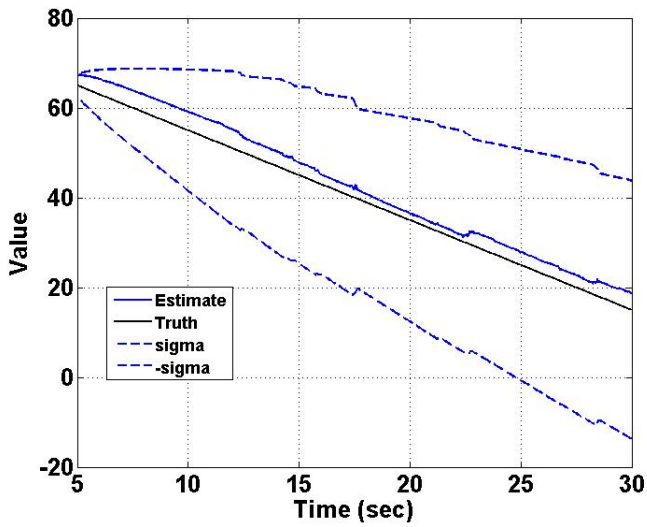

Figure 4.54: This figure shows the tracker's estimate for Object 2 of x_0, truth, and $\pm$ one standard deviation for the high noise case.

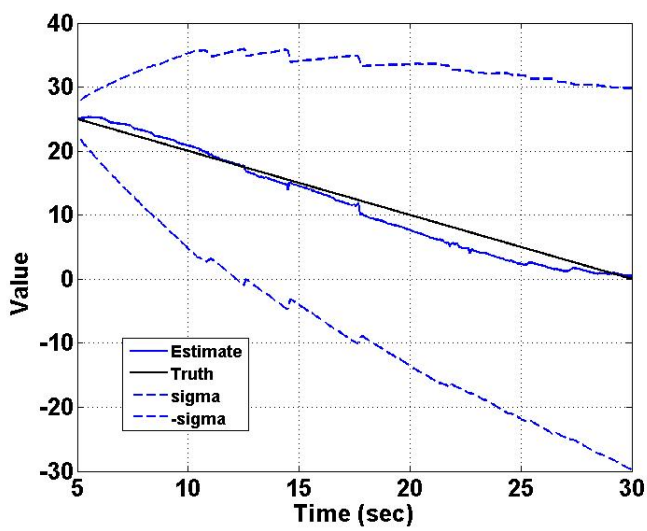

Figure 4.55: This figure shows the tracker's estimate for Object 1 of y_0, truth, and $\pm$ one standard deviation for the high noise case.

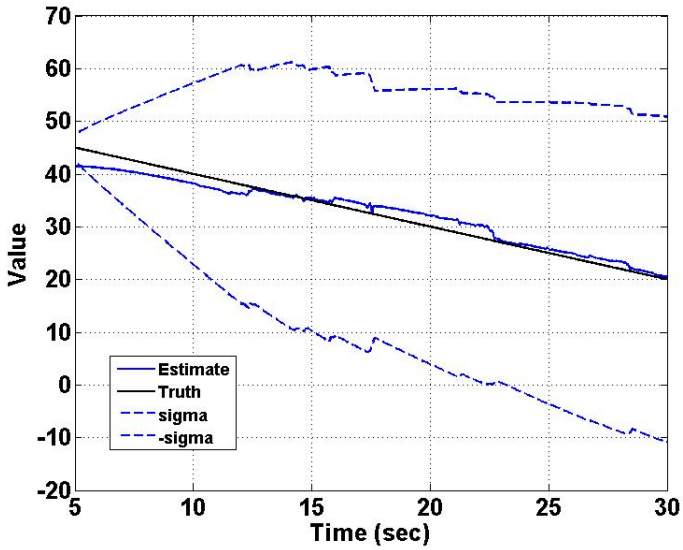

Figure 4.56: This figure shows the tracker's estimate for Object 2 of y_0, truth, and $\pm$ one standard deviation for the high noise case.

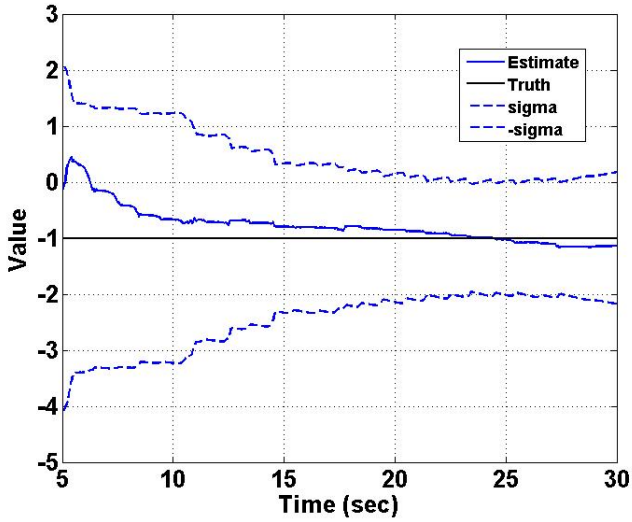

Figure 4.57: This figure shows the tracker's estimate for Object 1 of v_x, truth, and $\pm$ one standard deviation for the high noise case.

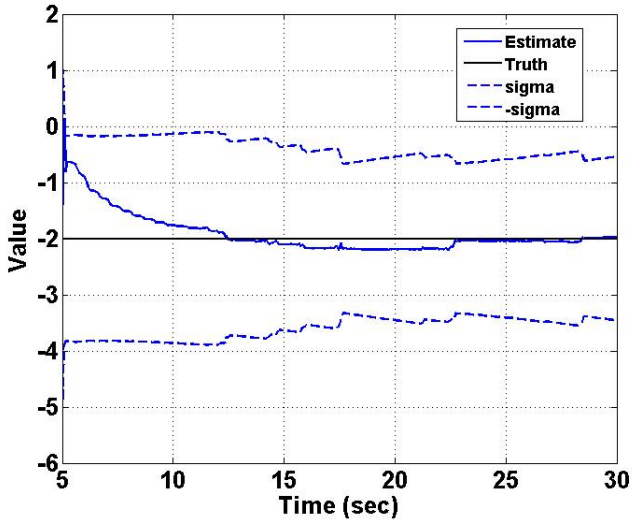

Figure 4.58: This figure shows the tracker's estimate for Object 2 of v_x, truth, and $\pm$ one standard deviation for the high noise case.

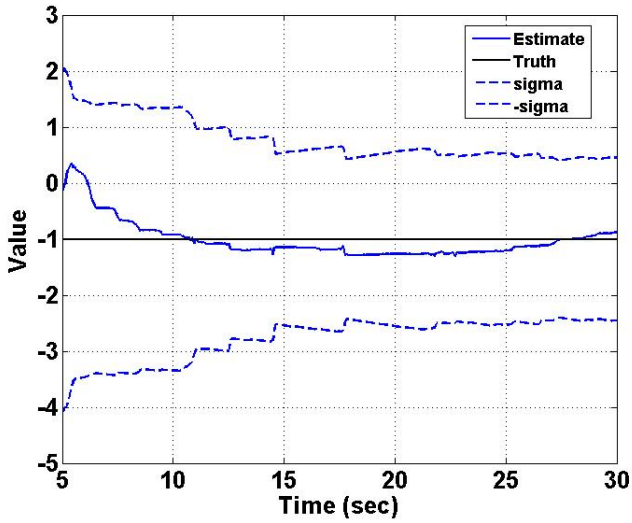

Figure 4.59: This figure shows the tracker's estimate for Object 1 of v_y, truth, and $\pm$ one standard deviation for the high noise case.

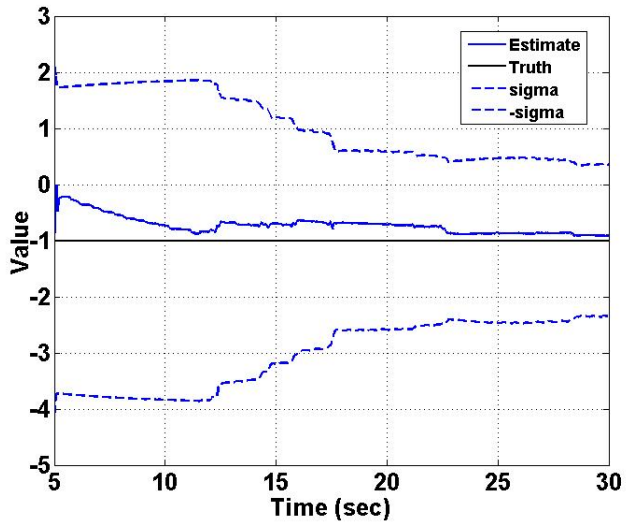

Figure 4.60: This figure shows the tracker's estimate for Object 2 of v_y, truth, and $\pm$ one standard deviation for the high noise case.

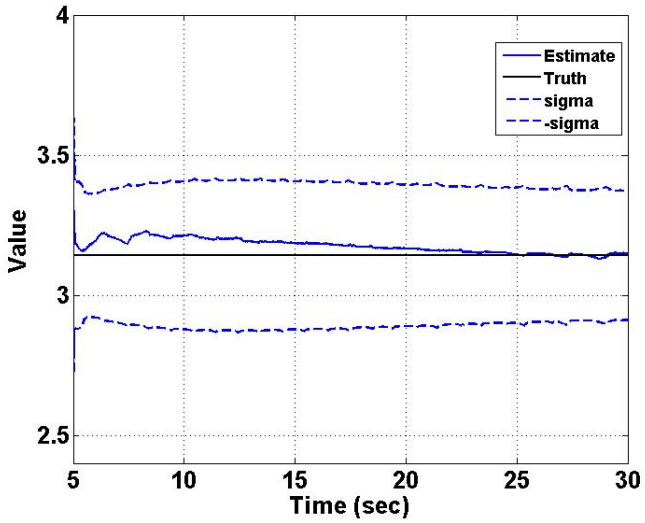

Figure 4.61: This figure shows the tracker's estimate for Object 1 of ω, truth, and $\pm$ one standard deviation for the high noise case.

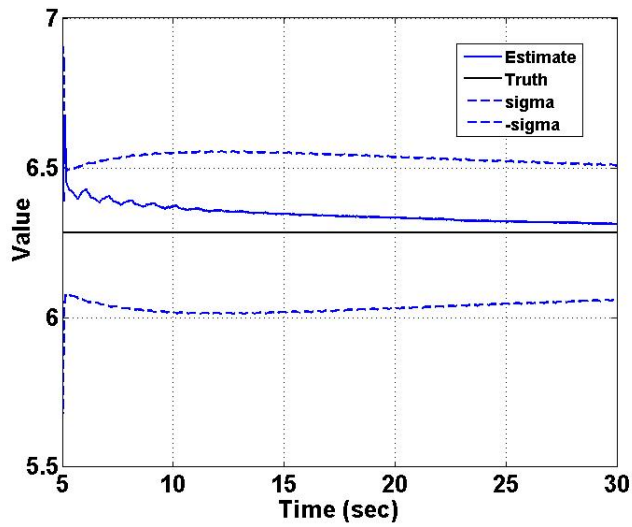

Figure 4.62: This figure shows the tracker's estimate for Object 2 of ω, truth, and $\pm$ one standard deviation for the high noise case.

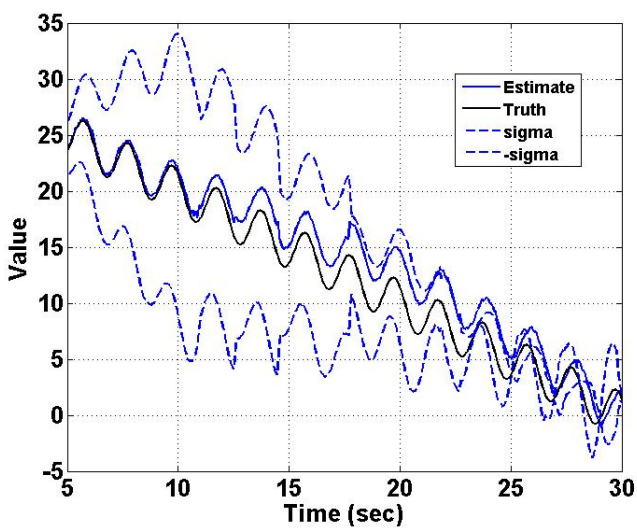

Figure 4.63: This figure shows the tracker's estimate for Object 1 of x_1, truth, and $\pm$ one standard deviation for the high noise case.

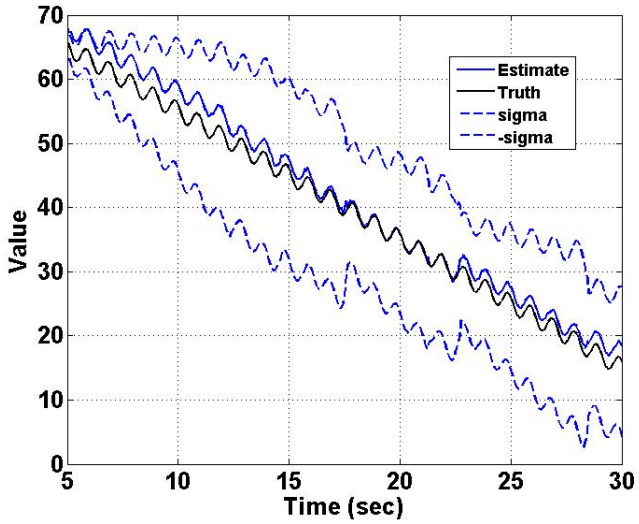

Figure 4.64: This figure shows the tracker's estimate for Object 2 of x_1, truth, and $\pm$ one standard deviation for the high noise case.

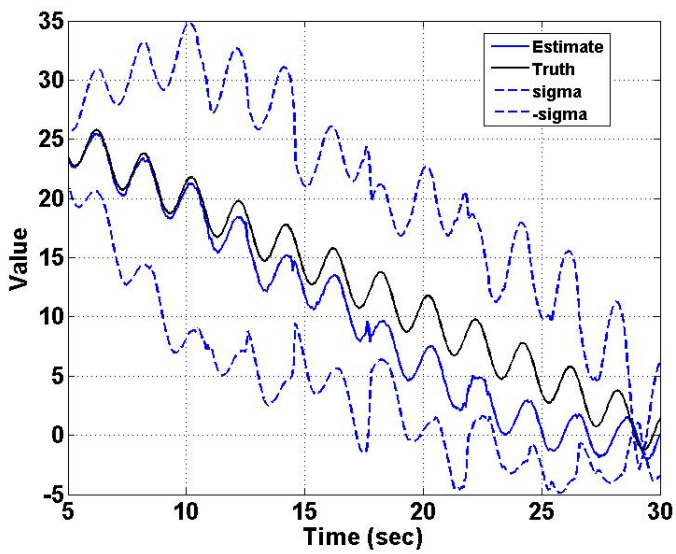

Figure 4.65: This figure shows the tracker's estimate for Object 1 of y_1, truth, and $\pm$ one standard deviation for the high noise case.

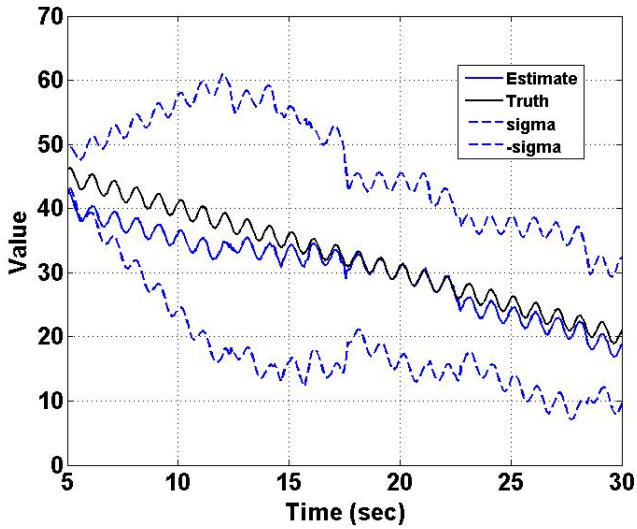

Figure 4.66: This figure shows the tracker's estimate for Object 2 of y_1, truth, and $\pm$ one standard deviation for the high noise case.

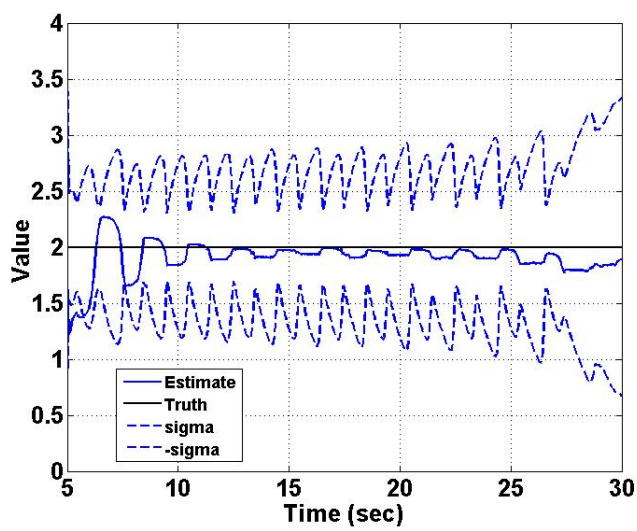

Figure 4.67: This figure shows the tracker's estimate for Object 1 of r, truth, and $\pm$ one standard deviation for the high noise case.

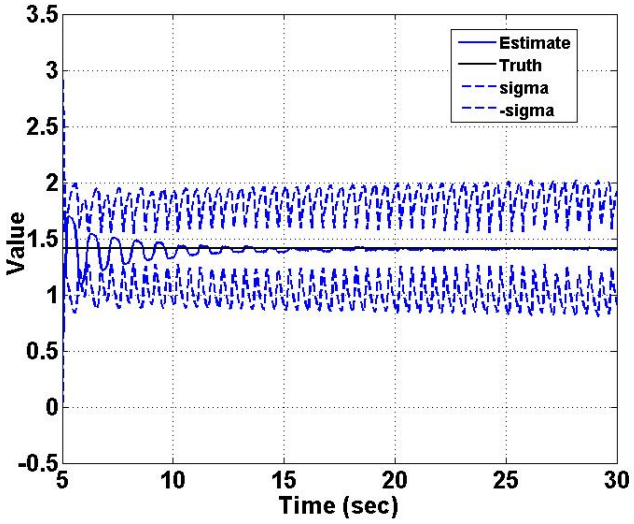

Figure 4.68: This figure shows the tracker's estimate for Object 2 of r, truth, and $\pm$ one standard deviation for the high noise case.

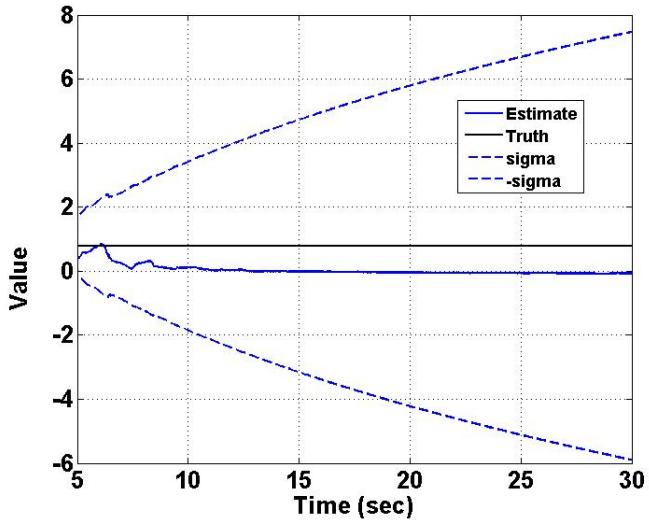

Figure 4.69: This figure shows the tracker's estimate for Object 1 of ϕ, truth, and $\pm$ one standard deviation for the high noise case.

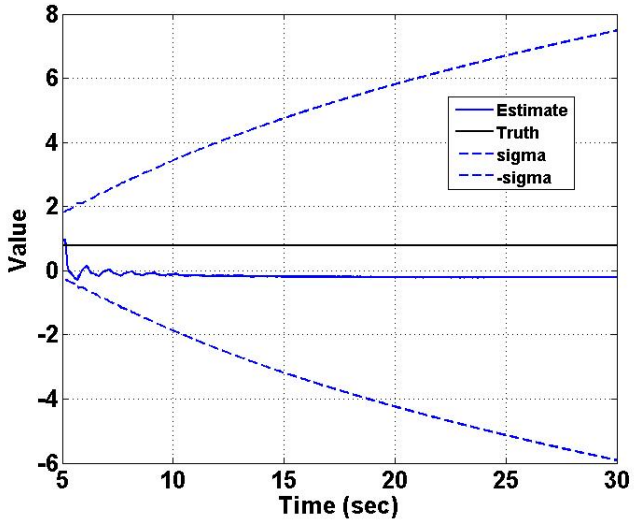

Figure 4.70: This figure shows the tracker's estimate for Object 2 of ϕ, truth, and $\pm$ one standard deviation for the high noise case.

Chapter 5

Conclusions and Future Work

This work set out to build a multiple target tracker when the measurements are range and range-rate. The sensor, although modeled to be generic, is providing wideband returns of the objects. This means there will be multiple returns corresponding to one object. These returns are coming from scatterers located at various places on the object. The objects' motion includes both a linear and nonlinear component. Since the measurements are range and range-rate, the relationship between the measurements and the filter state is nonlinear. To handle the nonlinearity in both the objects' motion and in the measurements, we used an EKF. To deal with the association problem, we used two distance-based cost functions with the GNN method of assignment and the Munkres algorithm. Since we want Cartesian states from range and range-rate measurements, we have an unobservable problem so the covariance values will overestimate the error in the tracker states. The tracker will be able to track the objects as long as the initial state is close to truth.

The experiments section in Chapter 4 showed how the filter, association method, and tracker all perform. The filter runs were considered for one particular object scene at two different noise cases. The filter used the initialization

portion of the association algorithm, the binning algorithm and scatterer-to-scatterer assignment, but assumed perfect association for the measurement-to-track assignment. The filter experiments were performed for two noise cases: a low noise case and a high noise case. The results for the low noise case are very close to truth. The accuracy in the estimate decreases as the noise increases to the high noise case. The three parts of the association algorithm were considered for the high noise case since this would be the more challenging of the two cases. The binning algorithm results, scatterer-to-scatterer assignment, and measurement-to-track assignment all show excellent performance for the high noise case.

When we put the filter and association scheme together to create the tracker, we again considered both the high and low test cases. Based on the filter's performance for both the high and low noise cases, and the association algorithm's performance for the high noise case, we expect the tracker results for the low noise case to perform well and the results from the high noise case to be adequate. This was indeed the result obtained. The tracker performed well for the low noise case, but performance decreased in accuracy for the high noise case. Comparing the filter and association results for the high noise case, it appears that the filter is the limiting component of the tracker.

5.1 Future Work

While the tracker performance looks good so far, there are many more tests that need to be done. These tests could look at other noise levels and test cases. First of all, only two runs were performed. To truly understand the capabilites and limitation of the filter, Monte Carlo runs would need to be performed. These runs could consider several different parameters of the

object dynamics and/or number or type of objects considered and also the noise level. It would be interesting to find the noise level which renders the tracker results useless.

We could also consider future work in the state space model. In this work, we assumed that the object was spinning in the plane of the reference frame. However, this assumption is likely to not always hold. An interesting extension would be to consider this case and include another angle to account for the projection on the plane of the reference frame. This extension would lead to the introduction of more nonlinearities. Since our filter was built in 2D, another interesting extension would be to look at a model for the filter in 3D. This model could included both a spinning and precessing component.

In this work, we only considered the results from one sensor. It would be interesting to see how, or if, things improve if multiple sensors were used. This could lead to an improved estimate of the objects' dynamics and location. If the number of radars used was increased to three, the angle from the objects to each sensor would be able to be determined, to some degree of accurracy, and the unobervability of the problem would disappear.

Another place for future work would be to investigate any possible improvements from using a different filter or association technique. We could consider either the UKF or the particle filter. Both filters will increase the computational complexity, but it would be interesting to see if this leads to an increase in tracker accurracy. A new association technique could look at using the estimate of the centroid position as the criteria for assignment to an object and then JPDA algorithm could be used to update each scatterer. This is likely to improve the estimate of the centroid position and object velocity, but decrease the ability to tell the exact object location of the scatterer. A challenge to the current association algorithm would be to include false alarms

and missed detections. Since it is unlikely to get perfect detection all the time, this extension would be quite useful.

A final situation that could be considered for future work is when objects cross in range. To handle this problem, we could expand the MTT to be an MHT or build an algorithm to determine these crossings and coast through this situation.

Bibliography

[1] Bar-Shalom, Y., Li, X. R., and Kirubarajan, T. Estimation with Applications to Tracking and Navigation. New York: John Wiley and Sons, Inc, 2001.

[2] Blackman, S. Multiple-Target Tracking with Radar Applications. USA: Artech House, 1986.

[3] Jazwinski, A. Stochastic Processes and Filtering Theory. New York: Academic Press, 1970. Vol. 64 of Mathematics in Science and Engineering. Ed. Richard Bellman.

[4] Blackman, S. and Popoli, R. Design and Analysis of Modern Tracking Systems. Boston: Artech House, 1999.

[5] Ristic, B., Arulampalam, S., and Gordon, N. Beyond the Kalman Filter: Particle Filters for Tracking Applications. Boston: Artech House, 2004.

[6] Bugallo, M., Xu, S., and Djuric, P. "'Performance Comparison of EKF and particle filtering methods for maneuvering targets."' Digital Signal Processing. 17, (2007):774-786.

[7] Erwin, R., Santillo, M., and Bernstein, D. "'Spacecraft Trajectory Estimation Using a Sampled-Data Ex-

tended Kalman Filter with Range-Only Measurements"'. Proceedings of the 45th IEEE Conference on Decision and Control. (Dec. 2006):3144-3149.

[8] Gustafsson, G. and Isaksson, A. "'Best Choice of Coordinate System for Tracking Coordinated Turns"'. Proceedings of the 35th Conference on Decision and Control. (Dec. 1996):3145-3150.

[9] Yang, C., Bakich, M., and Blasch, E. "'Nonlinear Constrained Tracking of Targets on Roads"'. 7th International Conference on Information Fusion. (2005):235-242.

[10] Cui, N., Hong, L., and Layne, J. "'A comparison of nonlinear filtering approaches with an application to ground target tracking"'. Signal Processing. 85, (2005):1469-1492.

[11] Zhao, Z., Chen, H., Chen, G., Kwan, C., and Li,X. "'Comparison of Several Ballistic Target Tracking Filters"'. Proceedings of the 2006 American Control Conference. (June 2006): 2197-2202.

[12] Farina, A., Ristic, B., and Benvevuti, D. "'Tracking a Ballistic Target: Comparison of Several Nonlinear Filters"'. IEEE Transactions on Aerospace and Electronic Systems. 38, (July 2002): 854-867.

[13] Ristic, B. and Arulampalam, M.S. "'Tracking a manoeuvring target using angle-only measurements: algorithms and performance"'. Signal Processing. 83, (2003):1223-1238.

[14] Mallick, M. and Arulampalam, S. "'Comparison of Nonlinear Filtering Algorithms in Ground Moving Target Indicator (GMTI) Tracking"'. Signal and Data Processing of Small Targets. (2003). Ed. Oliver E. Drummond. Proceedings of SPIE. 5204, (2003): 630-647.

[15] Bar-Shalom, Y., Kirubarajan, T., and Lin, X. "'Probabilistic Data Association Techniques for Target Tracking with Applications to Sonar, Radar, and EO Sensors"'. IEEE Aerospace and Electronic Systems Magazine. 28, (August 2005): 37-56.

[16] Bar-Shalom, Y. "'Negative Correlation and Optimal Tracking with Doppler Measurements"'. IEEE Transactions on Aerospace and Electronic Systems. 37, (July 2001): 1117-1120.

[17] Pulford, G. and Evans, R. "'A Multipath Data Association Tracker for Over-the-Horizon Radar"'. IEEE Transactions on Aerospace and Electronic Systems. 34, (Oct. 1998): 1165-1183.

[18] Hilands, T. and Thomopoulos, S. "'Detection and Range/Doppler Estimation for Colocated Sensors"'. IEEE Transactions on Aerospace and Electronic Systems. 33, (July 1997): 825-834.

[19] Petsios, M., Alivizatos, E., and Uzunoglu, N. "'Manoeuvering target tracking using multiple bistatic range and range-rate measurements"'. Signal Processing. 87, (2007):665-686.

[20] Li, X. and Jilkov, V. "'Survey of Maneuvering Target Tracking. Part I: Dynamic Models"'.

IEEE Transactions on Aerospace and Electronic Systems. 39, (Oct. 2003):1333-1364.

[21] Roecker, J. and McGillem, C. "'Target Tracking in Maneuver-Centered Coordinates"'. IEEE Transactions on Aerospace and Electronic Systems. 25, (Nov. 1989): 836-843.

[22] Li, X. and Jilkov, V. "'A Survey of Maneuvering Target Tracking: Approximation Techniques for Nonlinear Filtering"'. Proceedings of 2004 SPIE Conference on Signal and Data Processing of Small Targets. (April 2004): 537-550.

[23] Li, X. and Jilkov, V. "'A Survey of Maneuvering Target Tracking - Part III: Measurement Models"'. Proceedings of SPIE Conference on Signal and Data Processing of Small Targets. (July - Aug. 2001): 1-24.

[24] Blackman, S. "'Multiple Hypothesis Tracking For Multiple Target Tracking"'. IEEE Aerospace and Electronic Systems Magazine, Part 2: Tutorials. 19, (Jan. 2004): 5-18.

[25] Raju, G.V.S. and Wang, H. "'Sensor Data Fusion Using Mahalanobis Distance and Single Linkage Algorithm"'. IEEE International Conference. (Oct. 1994): 2605-2610.

[26] Chen, J., Leung, H., Lo, T., Litva, J., and Blanchette, M. "'A Modified Probabilisitic Data Association Filter in a Real Clutter Environment"'. IEEE Transactions on Aerospace and Electronic Systems. 32, (Jan. 1996): 300-313.

[27] Nahi, N. "'Optimal Recursive Estimation With Uncertain Observation"'. IEEE Transactions on Information Theory. 15, (July 1969): 457-462.

[28] Farina, A., Ristic, B., and Timmoneri, L. "'Cramer-Rao Bound for Nonlinear Filtering With $P_d < 1$ and Its Application to Target Tracking"'. IEEE Transactions on Signal Processing. 50, (Aug. 2002): 1916-1924.

[29] Li, X. "'The PDF of Nearest Neighbor Measurement and a Probabilistic Nearest Neighbor Filter for Tracking in Clutter"'. Proceedings of the 32nd Conference on Decision and Control. (Dec. 1993): 918-923.

[30] Bourgeois, F. and Lassalle, J. "'An Extension of the Munkres Algorithm for the Assignment Problem to Rectangular Matrices."' Communications of the ACM. 14, (Dec. 1971): 802-804.

[31] Leung, H., Hu, Z., and Blanchette, M. "'Evaluation of Multiple Radar Target Trackers in Stressful Environments"'. IEEE Transactions on Aerospace and Electronic Systems". 35, (April 1999):663-674.

[32] Alouani, A.T., Xia, P., Rice, T.R., and Blair, W.D. "'Two-Stage Kalman Estimate For Tracking Maneuvering Targets"'. IEEE International Conference on Systems, Man, and Cybernetics. 2, (Oct. 1991):761-766.

[33] Algrain, M., and Saniie, J. "'Interlaced Kalman Filtering of 3-D Angular Motion Based on Euler's Nonlinear Equations"'. IEEE Transactions on Aerospace and Electronic Systems. 30, (Jan. 1994):175-185.

[34] Yang, C., Qu, J., Mao, S., and Li, S. "'A Initialization Method for Group Tracking"'. Proceedings of the IEEE National Aerospace and Electronics Conference. 1, (May 1995):303-308.

[35] Shyu, H., Lin, Y., Yang, J., and Jinchi, H. "'The Group Tracking of Targets on Sea Surface by 2-D Search Radar"'. IEEE International Radar Conference. (1995):329-333.