Target Tracking a Non-Linear Target Path Using Kalman Predictive Algorithm

Abstract

Some forms of motion are measured using passive reflective markers attached to points of interest during motion capture. Analysis of the motion with quantitative feedback is desired as soon as possible. The goal of this paper is to present an algorithmic approach that may provide this motion path analysis in real-time. Some of the problems encountered with real-time analysis are: (a) The ability to calculate target centroid data in real-time. (b) The ability to estimate a marker/target position even while occluded. (c) The ability to properly discriminate the markers during the full range of motion. In other words, the goal is to keep the initial target assignments, whether they are target 1, target 2, or target 3, and so on throughout the range of motion.

This paper uses a centroid algorithm that calculates the centroid data, making only one pass through the video picture. Assuming very little knowledge of any of the path motions, and that paths will be non-linear in their motion, two motion models of the Kalman filter are presented in order to create the ability to track and estimate target paths so that occluded targets are properly tracked. A use of the Kalman filter output and the Maximum Likelihood Estimation (MLE) algorithm are presented as a means to maintain proper target discrimination. The results of this paper are that the marker/target centroid data can be calculated with one pass through the data. The Kalman filter noisy acceleration model presents the most optimal solution for target path tracking with the human motion trials analyzed. The Kalman filter, combined with the MLE algorithm, presents a solution to track both occluded markers and properly discriminate the markers throughout the range of analyzed motion. Our research lays a foundation for implementing a real-time target path tracking and discrimination for many application areas for track target movements.

Introduction

The focus of the paper is to provide a robust solution of predicting the path of passive trackers even if the trackers are occluded for example in the study of human movement or activity by video-based analysis. Some techniques use active markers (optic based or RF based, etc.) placed on segments of interest. Others use either passive techniques, such as filming the motion and manually digitizing points of interest, or a more automatic passive means such as passive markers placed on segments of interest where a computer finds the markers of interest. Of these, one of the more popular methods is to attach passive reflective markers to the area that is the subject of the movement of study. This is the technique that this paper is written to, with the intent being to enhance the ability to capture and display information that is desired by the measurement system. The markers used in these particular methods reflect the light that is shined upon the field of motion area and show up in the video as high intensity marks, or targets of interest, such as shoulder, knee, etc. The motion being analyzed is then captured on video. Each video frame in the sequence is analyzed to determine the center of mass for each target by either some manual method, or some signal processing means. Then the target locations are tracked so that some empirical data can be reported based on the movements of the tracked positions. These techniques are computationally time consuming and require the data to be post processed.

For example, consider an analysis of a person's golf swing. Markers are placed on the points of interest such as: Head of golf club; Midway point of golf club; Hand hold of golf club; Shoulders of golfer; and Head of golfer. As the golfer takes a swing, the target markers are shown on the video and tracked, and the motion is analyzed, and empirical data is generated, such as head position during this movement or speed of golf club head. Multiple cameras might be desirable when the motion of interest is complex and the markers attached to the points of interest move in a very non-linear path. Often the target will also become occluded, or the target may move in a curvature path.

This paper investigates a method to calculate a target centroid in one algorithm pass through the video frame and to create a method based on Kalman filtering that can be used to track targets through a video

sequence. The problem to be solved is that of target discrimination. This is the ability to assign a target to a particular segment of movement, such as the right shoulder, and always be able to identify the path of this target even while other targets may cross this path, or in the event that this target may be hidden for a short period. Target tracking presents the most difficult problem. The targets may undergo acceleration and deceleration, their paths may be nonlinear (possibly circular), and they might cross paths with other targets. Many times the tracking algorithm will have no feedback on target locations for short intervals, since some targets may be occluded for several frames of the video sequence. Much of the work in predictive-path analysis using Kalman filter techniques has been written assuming a more linear trajectory, such as ballistic or aircraft target tracking. Maneuvering target prediction is a more complicated area of study and several approaches have been written about [1][2][3]. But it is the effort of this paper to present a multi–target discrimination technique that can be used in the biomechanical motion studies. Three main areas of development that will be addressed in this paper are: centroid calculation, Kalman filtering, and target discrimination. These are described below.

Centroid Calculation

Since this analysis is based on video, the ability to calculate target centroids within software becomes computationally intensive. Normally it will take several passes through the two dimensional (2-D) video picture for an algorithm to find a centroid value. Most of the current approaches might use a 3×3 convolution pass for edge detection, or some other type of algorithm. At a normal National Television System Committee (NTSC) composite video frame rate of 30 Hz, there exist 30 frames, or pictures, of video data in which there may be multiple targets for every second of video. This quickly becomes a time-consuming operation with the standard 2-D approaches. This paper presents an algorithm for going through the video in one iteration, and having the centroid data immediately available, which significantly reduces the processing time.

Kalman Filtering

This is the filtering scheme that will be used to predict the next state of the motion model used. The approach taken is to use a third-order kinematic equation to see if an accurate state prediction can provide a suitable solution to the motion of various targets encountered in these human motion studies.

Target Discrimination

This is the ability to assign a target to a particular segment of movement, such as the right shoulder, and always be able to identify the path of this target even while other targets may cross this path, or in the event that this target may be hidden for a short period. The targets undergo positive and negative accelerations, their paths are nonlinear (possibly circular), and they will cross paths with other targets. The approach for this paper is to be able to correctly discriminate up to nine targets, even when they cross other target paths, or become occluded.

The Kalman filters used in this paper are tested on actual trial data captured by motion measurement software. The company Vicon/Peak [4] offered the following three trials for use in this paper. Figure 1 shows the Walking trial; Figure 2 is the jumping tria. Trial 3 provides more complicated data as shown in Fig. 3, there are three markers attached to a hand-held wand. The wand is being waved in a random manner around and back and forth.

The Centroid Algorithm

The first step in performing this tracking work is to locate all of the markers in a video frame that will later be associated as targets. Then the targets will be assigned a position in order to track and predict the path. Color images (Figure 4) are converted to binary, scanned from top to bottom (Figure 5) and then

thresholded (Figure 8) to provide good results. The target locations are obtained using centroid block diagram (Figure 7).

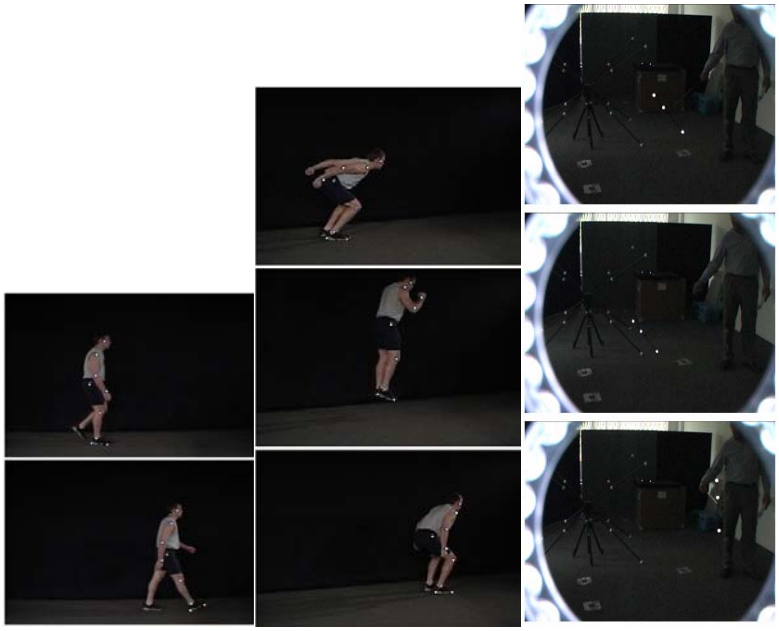
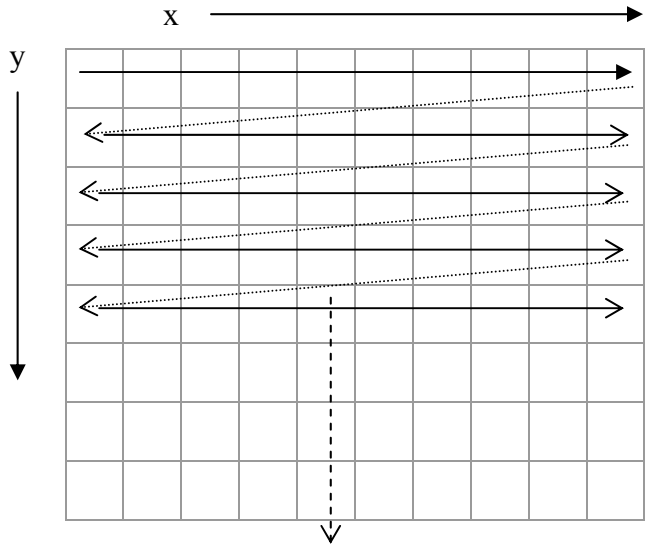


Figure 1 (left): Simple walking example. Figure 2 (middle) Horizontal jump trial. Figure 3 (right) Wand trial.

Figure 2: Sample video picture used.



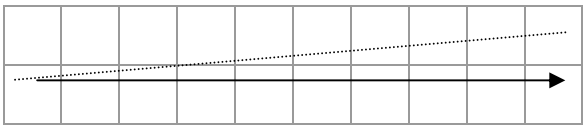


Figure 3: Scanning scheme.

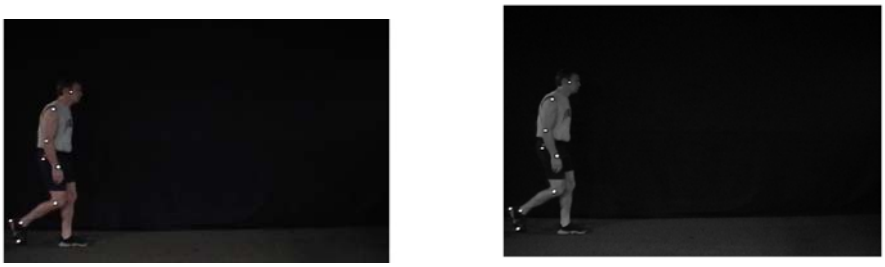


Figure 4: Color and gray-scale image.

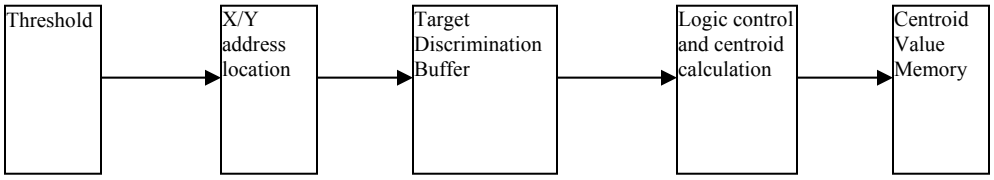


Figure 5: Centroid block diagram.

The centroid algorithm is block diagrammed in Fig. 7 and will be explained in subsequent sections. After a gray-scale image has gone through a threshold filter (Figure 8) it will have the properties as shown in Fig. 8. Note that the only pixels that exist with a intensity value above the black, or zero, background are the markers of interest.

Figure 6: Threshold image.

Once a target threshold is exceeded, its address location is stored in a buffer memory. The address simply comprises two counters: one for *x* (column), one for *y* (row), which are incremented as the pixels are scanned. For example, the addressing scheme is shown in Fig. 9. The addresses that contain all the target pixels are (3,2), (4,2), (3,3), (4,3)

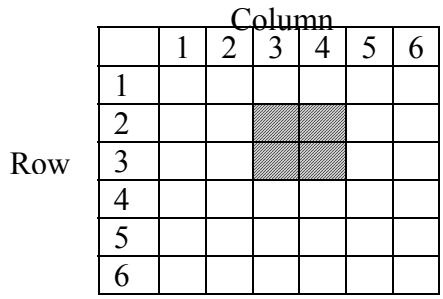


Figure 7: Addressing scheme

. Once a target pixel has exceeded a threshold, that address location is saved to a target buffer. The buffer will add all consecutive pixels that exceed threshold and it will also add the number of pixels that make up the target, see next section for description of centroid calculation. In the horizontal scan, as long as each consecutive pixel is above the threshold, the buffer will continue to sum the pixel data. As soon as a pixel is reached that does not exceed threshold, the summation is ceased and the left side *x* (horizontal) address and right side *y* (vertical) address are used as thresholds that are used for the next scanned row. When the next consecutive row is being scanned, if any pixel exceeds the threshold value and exists between the left and right address locations, then these pixels are summed into the previous target data. At this point, once a target is deemed completed, the summation data is sent to the logic control and calculation block. The centroid is calculated using equation (1).

$$(x_i, y_i) = C_i = \left(\frac{\sum_{k\in\{\text{pixels in target i}\}} x_k}{\sum_{k\in\{\text{pixels in target i}\}} n_k}, \frac{\sum_{k\in\{\text{pixels in target i}\}} y_k}{\sum_{k\in\{\text{pixels in target i}\}} n_k} \right) \quad \textbf{(1)}$$

The block will take the summation data and perform the division of Equation (1). Using the pixel values from Fig. 9,

$$\sum_{k\in\{\text{pixels in target i}\}} x_k = 14, \quad \sum_{k\in\{\text{pixels in target i}\}} y_k = 10, \text{ and } \sum_{k\in\{\text{pixels in target i}\}} n_k = 4$$

will yield the following centroid:

$$\frac{\sum_{k\in\{\text{pixels in target i}\}} x_k}{\sum_{k\in\{\text{pixels in target i}\}} n_k} = \frac{14}{4} = 3.5, \quad \frac{\sum_{k\in\{\text{pixels in target i}\}} y_k}{\sum_{k\in\{\text{pixels in target i}\}} n_k} = \frac{10}{4} = 2.5.$$

For the example shown in Fig. 10, using the algorithm as described above, the pixel location at (2,3) will not be acquired in the target data. Using a technique that can join two targets together to form one target solves this particular error. In the above sample target, once the pixel data at (2,3) is reached, it does not fall into the search parameters of the target row 2 above it, and so a second target is created; but once the next pixel at (3,3) is reached, it now falls into both targets, the summation data from these two separate targets are joined together, and one of the target buffers is erased.

This algorithm, by design, is capable of discriminating targets up to one pixel separation in both the horizontal and vertical directions. Due to shuttering or smearing, there exist several pixels at the target edges that may or may not meet the threshold value. These are pixels that border on the edge of the target. In most cases the target edges dropping in and out present very little error if the number of target pixels is large. If the number of pixels is small, then the error contribution to the calculated centroid data is larger. In order to create a measurement error for use in the Kalman filter, we analyzed this further as below.

Viewing Fig. 15, the centroid is calculated to be (3.5, 4). However, if the two pixels shown in blue were missing due to simply being under the threshold value, then the centroid would still be (3.5, 4). If just the two pixels shown in red were missing, the centroid would be (3.64, 4.21).

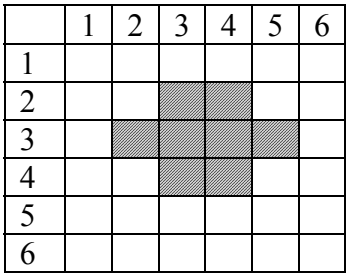


Figure 8: Sample target

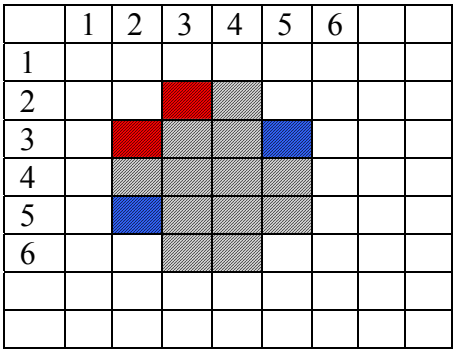


Figure 9: A 16 pixel target size.

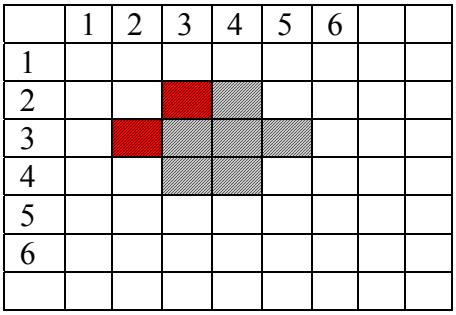


Figure 10: An 8 pixel target size

Viewing Fig. 16 above, with all pixels shown, shows that the centroid is (3.5, 3). With the red pixels missing, the centroid is now (3.8, 3.17). With a smaller number of pixels, the error may be greater in calculating the centroid; however, the error is not on the scale of a whole pixel but on the order of tenths of a pixel. The threshold value was chosen to keep the number of pixels close to a mean of 18-22 pixels in size per target. In tracking the targets within the image field and not in "real" dimension space, the error contribution of the image sensor is on the sub-pixel level as well; therefore, the error contribution chosen for the Kalman measurement error was chosen to be ±0.5 pixels, which should be more than an adequate model of the possible error contribution due to both the centroid algorithm and the camera imaging errors.

The Kalman Filter

The Kalman filter is introduced as the estimator used to estimate the position of the targets being tracked. A discussion of the state-space model form used to model the differential equation for motion is given. The operation of the five step recursive Kalman filter is explained. The acceleration and jerk models used to model the target motions of the system are described. The time step update variable is discussed and a derivation of the standard deviation of movement noise is described. The operation and performance of the Kalman filter is shown on the walking, jumping and wand trials. The Kalman filter, introduced by R.E. Kalman as a solution to predictive problems [5], is used to estimate the target position in this paper. The Kalman filter is a predictive algorithm based on a linear quadratic estimation problem, which is used to estimate the instantaneous state of a linear dynamic system [6]. The state $x(t_o)$ of a system at time t_o is the information at t_o that, together with the input $u(t)$, for $t \geq t_o$, determines uniquely the output y(t) for all $t \geq t_o$ [7]. Because the state of a system is being estimated the model for the system will be a state-space model. A state-space model is simply a representation of a differential equation, and can be shown in the following example for a simple first order differential equation. Equation (2) below, is a first order kinematic equation of motion stating that the x position of a particle at the next sample of time, $k+1$, is the previous location plus the velocity times the time increment:

$$x_{k+1} = x_k + T\dot{x}_k \tag{2}$$

$$\dot{x}_k = \text{Velocity}$$

This may be represented as a state-space, or state equation, model the following form as follows:

$$\begin{bmatrix} x_{k+1} \\ \dot{x}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ \dot{x}_k \end{bmatrix} \tag{3}$$

In this example $\begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}$ represents the system matrix. The vector $\begin{bmatrix} x_k \\ \dot{x}_k \end{bmatrix}$ represents the state vector of the model, and vector $\begin{bmatrix} x_{k+1} \\ \dot{x}_{k+1} \end{bmatrix}$ is the state vector being estimated at the next sample time.

In this paper, the targets being tracked could follow a nonlinear path. In other words, the trajectory and velocities cannot be assumed to be linear or constant. However, because the motion can be modeled by the second- and third-order kinematic linear equations, described in more detail later, the model is represented as a linear equation. Expanding on the state-space model, the Kalman filter is based on the estimation of the state by using the state and the variance of the state; or in other words, how much the state is expected to change. By viewing a simple one-dimensional model of constant velocity target motion, the components of the Kalman filter will be shown.

Equation (2) represents a simple constant velocity model. However, in most tracking scenarios the target undergoes varying velocity, which is manifested as an acceleration component. Equation (4) represents a tracking target undergoing a change in velocity, which in this case is modeled as a noise input.

$$x_{k+1} = x_k + T\dot{x}_k + \frac{T^2}{2}\ddot{x}_k, \quad \frac{T^2}{2}\ddot{x}_k = u_w$$

$$x_{k+1} = x_k + T\dot{x}_k + u_w \tag{4}$$

$$\dot{x}_k = \text{Velocity},\ \ddot{x}_k = \text{Acceleration}$$

In this paper, a random motion is assumed in which the noise $u_w \sim N(0,\sigma_w^2)$. N denotes a Gaussian random variable of mean 0 and variance σ_w^2. Putting equation (3) into state space form results in the following equation:

$$p_{k+1} = Ap_k + Bu_w$$

$$s_k = Cp_k + Du_v \quad \textbf{(5)}$$

Because the variables *x* and *y* are being used for target locations, the variable *p* is used to denote the state variables, and *s* is used to denote the measured position; see Sections 4.5.1 and 4.5.2. The term u_v, which is defined as $u_v \sim N(0,\sigma_v^2)$, is used to denote measurement noise, or in other words the degree to which the measurement is unknown. Therefore, Eq. (5) represents a state equation for a simple one-dimensional noisy acceleration model of target motion. It is this model that the Kalman filter will build upon. The covariance of the state error is calculated and is represented by Σ_e. It is also further defined, that in order for the Kalman filter to be optimum, the Gaussian movement noise and the Gaussian measurement noise must be uncorrelated. It is through recursive operations that the Kalman filter becomes an optimum solution to the prediction of the next step. The Kalman filter becomes optimum because it minimizes the expected squared-error of the state, which is the uncertainty of the state. It operates by calculating a new gain, (*K*), at each measurement update state [8][9] and does not rely on one constant gain as in a steady-state system. The Kalman filter's calculations correspond to either a movement or measurement update, as shown in Table 1 [10]. Table 1 shows the steps in implementing the Kalman filter [11] and the order in which they are calculated in this paper. The terms Σ_e^-, and Σ_e^+ are used to denote the covariance of the state error before the measurement update, (Σ_e^-), and after the measurement update, (Σ_e^+). Steps 1–5 are processed recursively throughout the target tracking trial.

There exist several different models for estimating the movement of the target being estimated. References [12], [13], [14], [15] and [16] provide excellent sources of examining the different motion models. The model in this paper uses the basic kinematics of motion model. Two models as discussed below, were chosen.

This model simply uses noise as the acceleration component driving the movement. The two-dimensional noisy acceleration state space model is given in Eq. (11) below.

$$\begin{bmatrix} p_{x_{k+1}} \\ p_{y_{k+1}} \\ p_{v_{x_{k+1}}} \\ p_{v_{y_{k+1}}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_{x_k} \\ p_{y_k} \\ p_{v_{x_k}} \\ p_{v_{y_k}} \end{bmatrix} + \begin{bmatrix} T^2/2 & 0 \\ 0 & T^2/2 \\ T & 0 \\ 0 & T \end{bmatrix} u_w$$

$$\begin{bmatrix} s_{xk} \\ s_{yk} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_{x_k} \\ p_{y_k} \\ p_{v_{x_k}} \\ p_{v_{y_k}} \end{bmatrix} + u_v \quad \textbf{(6)}$$

Equation (11) is the two-dimensional state-space representation of the second-order motion Eq. (12).

$$x_{k+1} = x_k + T\dot{x}_k + \frac{T^2}{2}\ddot{x}_k \quad \textbf{(7)}$$

Table 1: Kalman filter steps

Measurement Update	1. Compute the Kalman gain.	$K = \frac{\Sigma_e^- C^T}{C\Sigma_e^- C^T + \sigma_v^2}$ **(8)**
Measurement Update	2. Update the error covariance.	$\Sigma_e^+ = (1 - KC)\Sigma_e^-$ **(9)**
Measurement Update	3. Update estimate with the measurement *p*.	$\hat{p}_{k+1} = \hat{p}_k + K(p - C\hat{p}_k)$ **(10)**
Movement Update	4. Project the error covariance ahead.	$\Sigma_e^- = A\Sigma_e^+ A^T + \sigma_w^2$ **(11)**
Movement Update	5. Project the state ahead.	$\hat{p}^- = A\hat{p}^+$ **(12)**

Table 2 provides a definition of the symbolic terms used in this paper.

Table 2: Kalman filter terms

Terms	Definition
A	The movement state matrix of the targets. How the targets will move from one state to the next.
B	The noise matrix state of the input noise. This determines in what manner the noise will affect the target position.
C	The measured position matrix, which of the position matrix states are measured.
u_w	The Gaussian movement noise input.
σ_w	The standard deviation of the movement noise.
σ_w^2	The covariance of the movement noise.
u_v	The Gaussian measurement noise.
σ_v	The standard deviation of the measurement noise.
σ_e^2	The covariance of the measurement noise.
$\hat{p}$	The variables of the state position estimate. $p = [p_{x_k}, p_{y_k}, p_{v_{x_k}}, p_{v_{y_k}}]^T$ for the noisy acceleration model. $p = [p_{x_k}, p_{y_k}, p_{v_{x_k}}, p_{v_{y_k}}, p_{a_{x_k}}, p_{a_{y_k}}]^T$ for noisy jerk model. p_{x_k} = horizontal target position, p_{y_k} = vertical target position, $p_{v_{x_k}}$ = horizontal velocity, $p_{v_{y_k}}$ = vertical velocity, $p_{a_{x_k}}$ = horizontal acceleration, $p_{a_{y_k}}$ = vertical acceleration
p	The only measured output is the x (for 1-dimensional) and y (for 2-dimensional) position of the target
s	The measured position output matrix.

Noisy Jerk Model

This model is using the change of acceleration, or jerk, as the component driving the motion. The two-dimensional noisy jerk state space model is given in Eq. (13).

$$\begin{bmatrix} p_{x_{k+1}} \\ p_{y_{k+1}} \\ p_{v_{x_{k+1}}} \\ p_{v_{y_{k+1}}} \\ p_{a_{x_{k+1}}} \\ p_{a_{y_{k+1}}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & T & 0 & T^2/2 & 0 \\ 0 & 1 & 0 & T & 0 & T^2/2 \\ 0 & 0 & 1 & 0 & T & 0 \\ 0 & 0 & 0 & 1 & 0 & T \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_{x_k} \\ p_{y_k} \\ p_{v_{x_k}} \\ p_{v_{y_k}} \\ p_{a_{x_k}} \\ p_{a_{y_k}} \end{bmatrix} + \begin{bmatrix} T^3/6 & 0 \\ 0 & T^3/6 \\ T^2/2 & 0 \\ 0 & T^2/2 \\ T & 0 \\ 0 & T \end{bmatrix} u_w$$

$$\begin{bmatrix} s_{xk} \\ s_{yk} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_{x_k} \\ p_{y_k} \\ p_{v_{x_k}} \\ p_{v_{y_k}} \\ p_{a_{x_k}} \\ p_{a_{y_k}} \end{bmatrix} + u_v \qquad \textbf{(13)}$$

This equation is derived from the third-order motion Eq (14).

$$x_{k+1} = x_k + T\dot{x}_k + \frac{T^2}{2}\ddot{x}_k + \frac{T^3}{6}\dddot{x}_k \qquad \textbf{(14)}$$

The terms of the above equations are:

x_k = target *x* (horizontal) position, $\dot{x}_k$ = *x* (horizontal) velocity, $\ddot{x}_k$ = *x* (horizontal) acceleration, $\dddot{x}_k$ = *x* (horizontal) jerk

Using the above models and the covariance of the movement noise $\sigma_w^2 \times I$, where *I* denotes the identity matrix, and measurement noise $\sigma_v^2 \times I$, the Kalman filter has all the requirements to implement a recursive solution. The first equation in the Kalman equation (6) depends on an initial covariance estimate Σ_e^-. This initial estimate is assigned a small value, because there is no knowledge of the actual uncertainty of the first estimate. In this paper the value selected is 1.0 times the identity matrix, meaning that the uncertainty of the first target assignment is one pixel. The next equation that requires an initial estimate is Eq. (8). The initial estimate of the position is assigned the initial measured value. With these two values defined, the Kalman filter can then be implemented. Another point of discussion is the value of the sampling interval (*T*) used in the algorithm. The algorithm uses $T = 1$, even though the video frame rate being examined is 30 frames/second, or sample time = 33.3 ms. In this case, the targets are being tracked and estimated within the pixel field of the sensor plane, meaning that the pixel location is what is being used to track the targets. With this relative pixel position, the sample time can simply be normalized to a period of 1. It does not change the performance of the tracking, except that it may alleviate the possibility of some round-off errors that might exist as the Kalman equations are used, and perhaps make real-time implementation more feasible since many numerical multiplies are reduced to a multiply by 1.

If the prediction tracking was done in "real" space, or in other words, if the estimation and location calculations were being performed based on the field of motion distances, then the sample time of 33.3ms would have to be used for this calculation to be correct. As an example, if a target is being tracked and it moved 22 pixels to the right, using a value of $T = 1$ would track and calculate an estimate of that target to

be approximately 22 pixels to the right. If, however, that same 22 pixel movement was in reality 10 feet, in the motion space, to the right in the actual motion space, in order for the estimation to correctly approximate a movement of 10 feet the actual sample time would have to be used in order to show an estimate of 10 feet. However, in most cases all that is desired is to accurately track in the sensor plane, which in this case is the pixel space. This means that in most cases the time (T) can be normalized to 1.

It thus depends on the desired output of the model. If actual distances are not of concern at the output, but rather the ability to track and discriminate a target in a field of view, the sample time T can be normalized to 1. If the actual output distances of and rate of travel distances of the target need to be known, then the actual sample time may be used. In both cases the Kalman filter and discrimination algorithms work just the same.

Selection of Movement-Noise Covariance Matrix

An estimate of the standard deviation of the movement noise could be selected by recursively selecting a value and measuring the error. Based on this experimental approach, an appropriate estimate as to this value may be derived. This paper chooses a few targets and the standard deviation of both the horizontal and vertical accelerations and jerk were calculated in order to get more of an empirical starting point for the selection of the movement standard deviation. A single target was chosen from each of the three trials: walking (Fig. 17), jumping (Fig. 19), and wand trial (Fig. 21). This was performed in order to assign an initial estimate as to the noise expectation driving the motion. Because two models are being utilized, one is driven by a Gaussian "acceleration" component, and the other is driven by a Gaussian "jerk" component. The velocity, acceleration and jerk output plots of these models are shown in the Figs. 18, 20 and 22. Table 3 contains the measured data of the three components, velocity, acceleration, and jerk for the single selected target within each of the three trials.

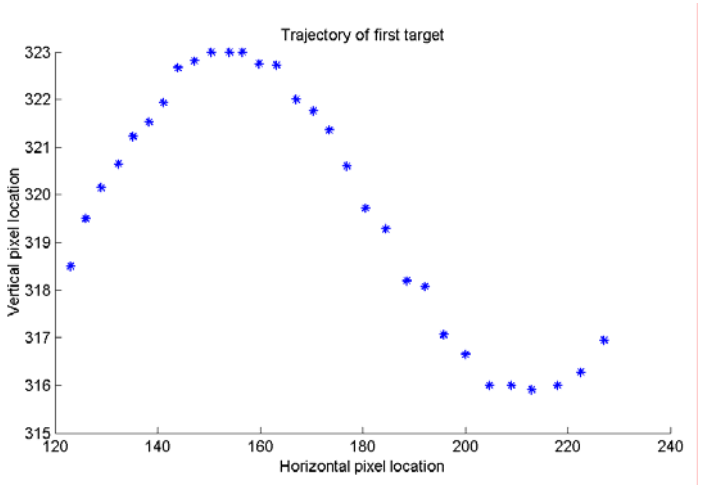


Figure 11: First target trajectory plot of walking trial

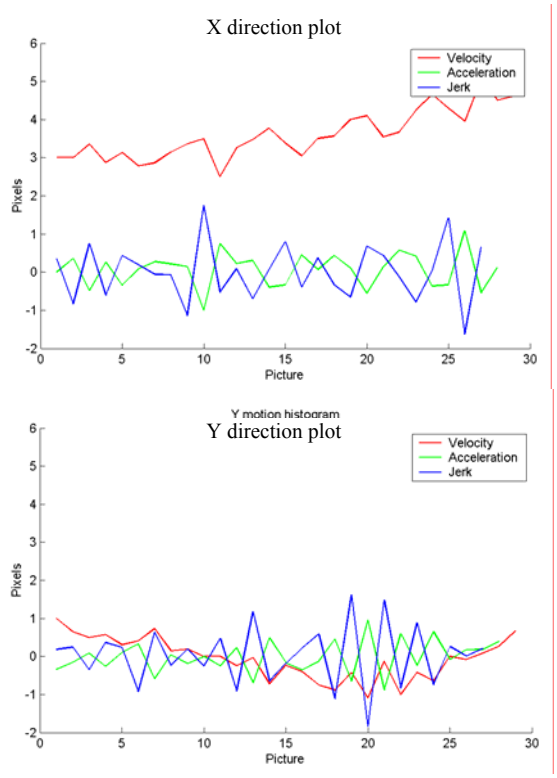


Figure 12: *x* and *y* velocity, acceleration, and jerk plots

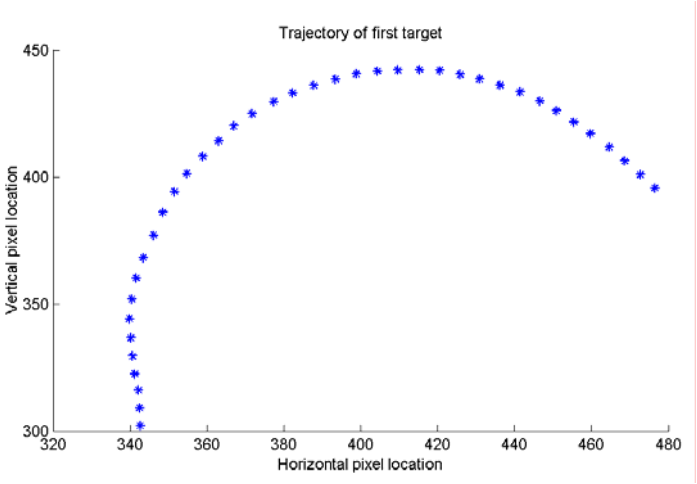


Figure 13: First target trajectory plot of jumping trial

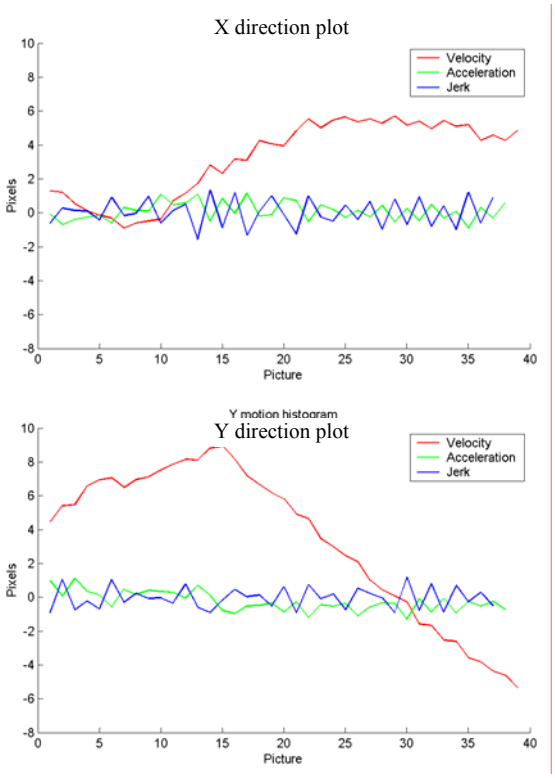


Figure 14: *x* and *y* velocity, acceleration, and jerk plots

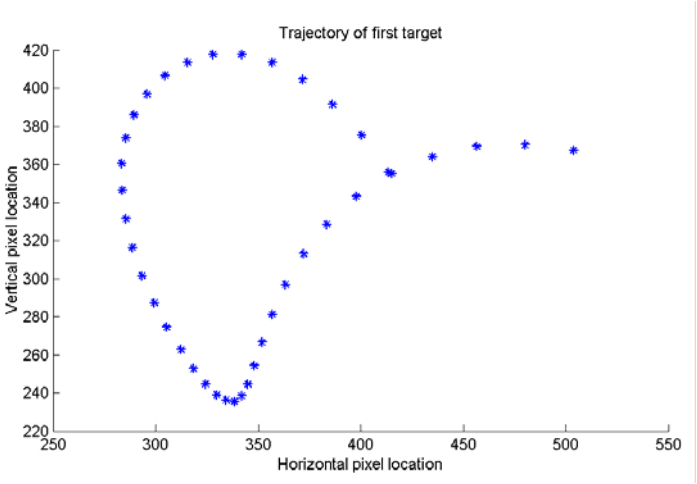


Figure 15: First target trajectory plot of wand trial

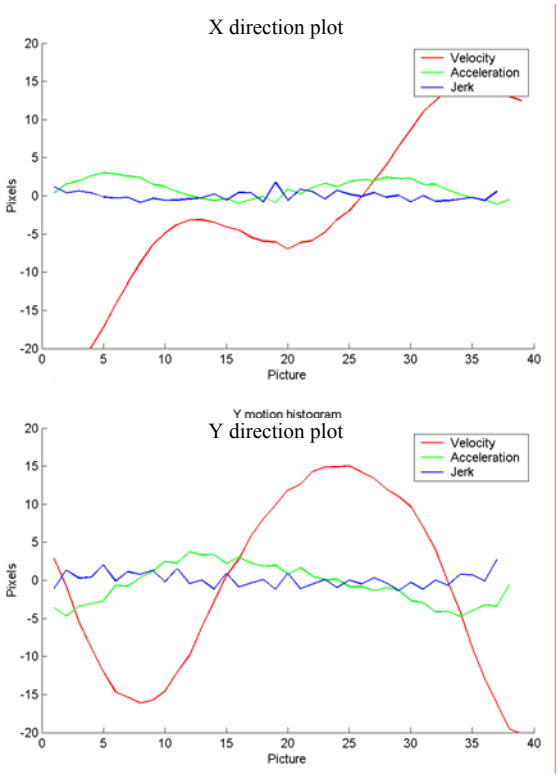


Figure 16: *x* and *y* velocity, acceleration, and jerk plots

Table 3 shows the measured data from these graphs.

Table 3: Dynamic Parameter Data

		Horizontal data	Vertical data
Walking trial	Mean Velocity	3.7667 pixels/time	-0.0167 pixels/time
	σ_v	1.1503 pixels	0.5769 pixels
	Mean Acceleration	0.2031 pixels/time^2	0.0019 pixels/time^2
	σ_a	0.9014 pixels	0.4339 pixels
	Mean Jerk	0.1528 pixels/time^3	0.0261 pixels/time^3
	σ_j	1.0840 pixels	0.7851 pixels
Jumping trial	Mean Velocity	3.5231 pixels/time	2.2305 pixels/time
	σ_v	2.1128 pixels	5.1589 pixels
	Mean Acceleration	0.0972 pixels/time^2	-0.3568 pixels/time^2
	σ_a	0.5334 pixels	0.5139 pixels
	Mean Jerk	-0.0033 pixels/time^3	-0.0398 pixels/time^3
	σ_j	0.8424 pixels	0.6437 pixels
Wand trial	Mean Velocity	-1.9963 pixels/time	-0.8100 pixels/time
	σ_v	11.1543 pixels	11.6927 pixels
	Mean Acceleration	0.9538 pixels/time^2	-0.6066 pixels/time^2
	σ_a	1.2290 pixels	2.5838 pixels
	Mean Jerk	-0.0249 pixels/time^3	0.0814 pixels/time^3

Table 3: Dynamic Parameter Data

		Horizontal data	Vertical data
	σ_j	0.6116 pixels	0.9709 pixels

Using this data as a starting point, the initial σ_a for the noisy acceleration model was chosen from the horizontal walking trial to be 0.90. The walking trial represents the medium value between the three trials and the horizontal value is the greater of the two values and therefore the initial starting value is 0.90. The starting standard deviation for the noisy jerk model was also chosen in the same manner and was selected as 0.85 from the Jumping trial.

Operation of the Kalman Filter

The Kalman filter noisy acceleration state–space model, Eq. (11), operations for all three trials are shown in Figs. 23-25. The error was calculated by measuring the vector distance from the measured position to the estimated position. Equation (15) gives the algorithm used:

$$\text{error}_{\text{distance}} = \sqrt{(x-\hat{x})^2 + (y-\hat{y})^2} \quad \textbf{(15)}$$

This is performed for every target within every video frame. Note how well the estimation of the target is tracking the true target trajectory. Each target that is being tracked has a Kalman filter buffer assigned to it. What this means is that there is a Kalman calculation performed on every target that exists. The second plot within each figure shows the actual tracked target motion within each trial. The true, or measured target is shown as a blue circle, and the predicted, or Kalman estimated position is shown as a red asterisk. Note how well the prediction follows the measured value. Further empirical analysis is examined in subsequent sections. The Kalman filter noisy jerk model, Eq. (13), operation for all three trials is shown below in Figs. 26-28. The σ_w for the noisy acceleration and noisy jerk model, were iteratively chosen to present the best results as far as error is concerned. However, due to the fact that these models operate in a manner that the measurement is "trusted;" in other words, the σ_w may be quite large but will correct to the measured data; this leads to the fact that although the σ_w may be quite large, it will correct to the measurement after the first few iterations. If the measurement noise were made greater, a larger value for σ_w would have a greater effect. In fact using σ_w = 0.900, and σ_v = 0.0833

$$\Sigma_e^- \Rightarrow \begin{bmatrix} 0.7976 & 0 & 0.8447 & 0 \\ 0 & 0.7976 & 0 & 0.8447 \\ 0.8447 & 0 & 1.1698 & 0 \\ 0 & 0.8447 & 0 & 1.1698 \end{bmatrix}, \quad \Sigma_e^+ = \begin{bmatrix} 0.0754 & 0 & 0.0799 & 0 \\ 0 & 0.0754 & 0 & 0.0799 \\ 0.0799 & 0 & 0.3598 & 0 \\ 0 & 0.0799 & 0 & 0.3598 \end{bmatrix}$$

If σ_w = 3.29,

$$\Sigma_e^- = \begin{bmatrix} 4.6874 & 0 & 7.1860 & 0 \\ 0 & 4.6874 & 0 & 7.1860 \\ 7.1860 & 0 & 12.4725 & 0 \\ 0 & 7.1860 & 0 & 12.4725 \end{bmatrix}, \quad \Sigma_e^+ = \begin{bmatrix} 0.0819 & 0 & 0.1255 & 0 \\ 0 & 0.0819 & 0 & 0.1255 \\ 0.1255 & 0 & 1.6484 & 0 \\ 0 & 0.1255 & 0 & 1.6484 \end{bmatrix}$$

This shows that the correction covariance Σ_e^+ is quickly corrected in the measurement update by the Kalman gain, which is calculated using Σ_e^-. The Kalman filter implementation explained in this section utilizes some data that is generated from the discrimination algorithm for its measurement update. The discrimination algorithm is described in greater detail in the next section.

The Kalman filter is used as the optimal estimation algorithm in this paper in order to predict the trajectory path of the targets being measured or tracked. The state-space representation of the second- and third-order kinematic model is utilized as the linear equation to model the non-linear trajectory of the targets. Operation of the Kalman filter occurs in five recursive steps, which involve calculating the optimum gain, state and state covariance of the current step and propagating to the next time step. The first target model chosen to model the trajectory/movement of the targets is a second-order differential model of motion called the noisy-acceleration model, which models the driving input as a Gaussian acceleration noise. The second model is a third-order differential model of motion called the noisy-jerk model, which models the driving input as a Gaussian jerk noise. The sample time of 33.3ms can be normalized to 1. The selection of σ_w was empirically derived so that a reasonable starting value can be selected. Operation of the two models was shown, as well as the very good prediction performance from the Kalman filter in both models. The best results were achieved by using the noisy-acceleration model.

Target Discrimination

Target discrimination is the ability to assign a target to a particular segment of movement, such as the right shoulder, and always be able to identify the path of this target even while other targets may cross this path, or if it is hidden for a short period. This presents the most difficult problem in tracking the target. These targets, as mentioned above, undergo positive and negative accelerations; their paths are non-linear, they may be circular, and they might cross paths with other targets. Many times a target may become obscured and the true target position data may be unavailable for several video frames until the targets become distinguishable again. Much of the work in predictive path analysis using Kalman filter techniques has been written assuming a linear trajectory, such as ballistic or aircraft target tracking [17]. Maneuvering target prediction is a more complicated area of study and several approaches have been documented, but they provide tradeoffs based on target dynamics. In other words, they work well if the type of movement is known beforehand. However, in the area of biomechanical studies presented here, each target may undergo a wide variety of positional maneuvers.

The approach used in this paper to assigning an estimated target position to a measured target position with discrimination between the targets is based on the maximum likelihood estimation (MLE) algorithm. As is shown in Eq. (8), the estimate of the state, the update requires that a target position already be assigned to this estimated position in order to correct the estimate. It is necessary that the target discrimination already be assigned to this position estimate before this step can be performed. The result is that the estimated target position for the next picture is assigned to a target before the update of the position estimate. The target discrimination step is taken right before the Kalman filter equations, except for the first picture, in which the targets are first being assigned. This means that the propagation estimates, $\hat{p}^-$ and Σ_e^- are used for determining which estimated target is assigned to a measured target for the next picture, $\hat{p}_{k+1}$.

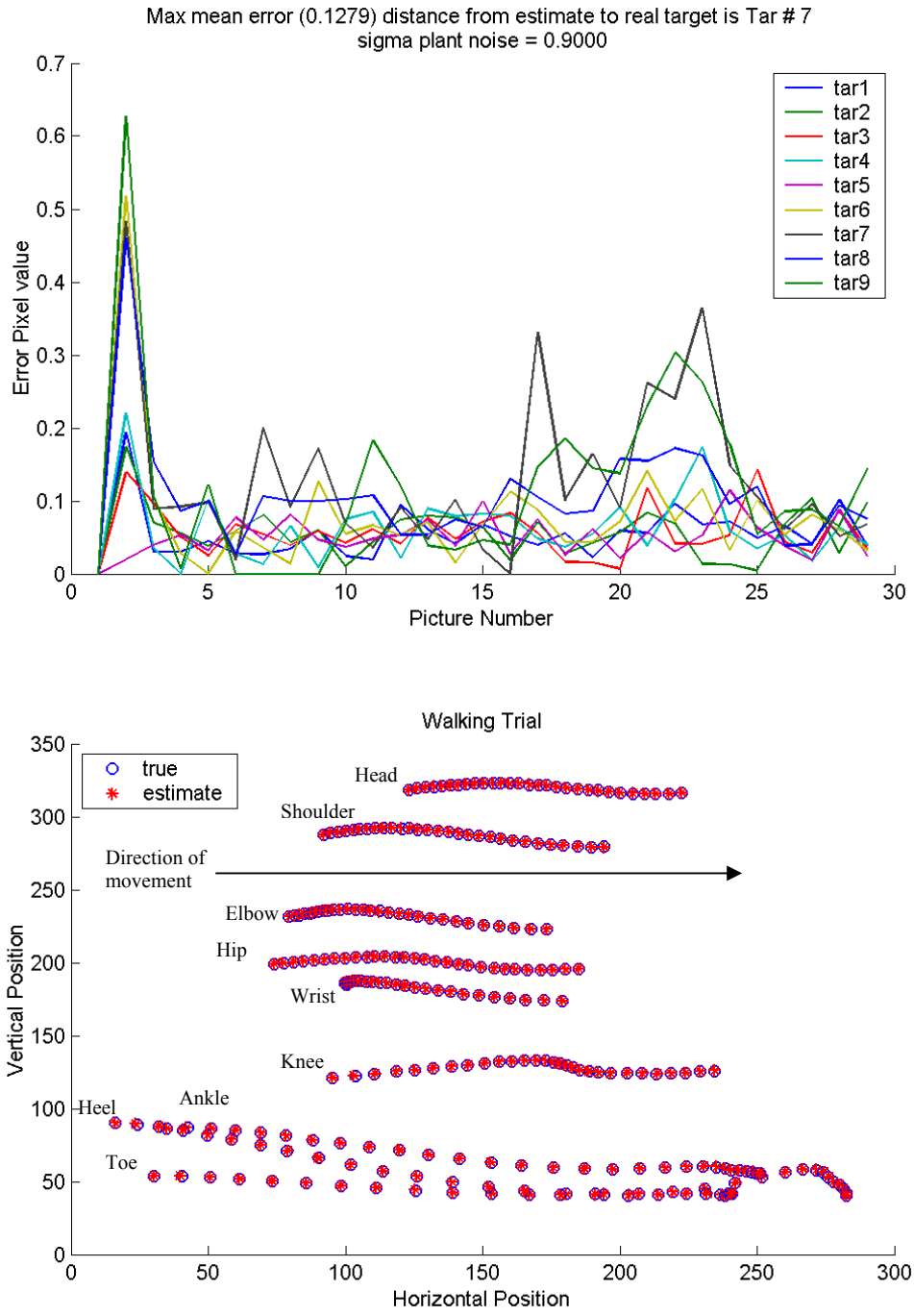


Figure 17: Acceleration walking trial - Error plot and trajectory plot

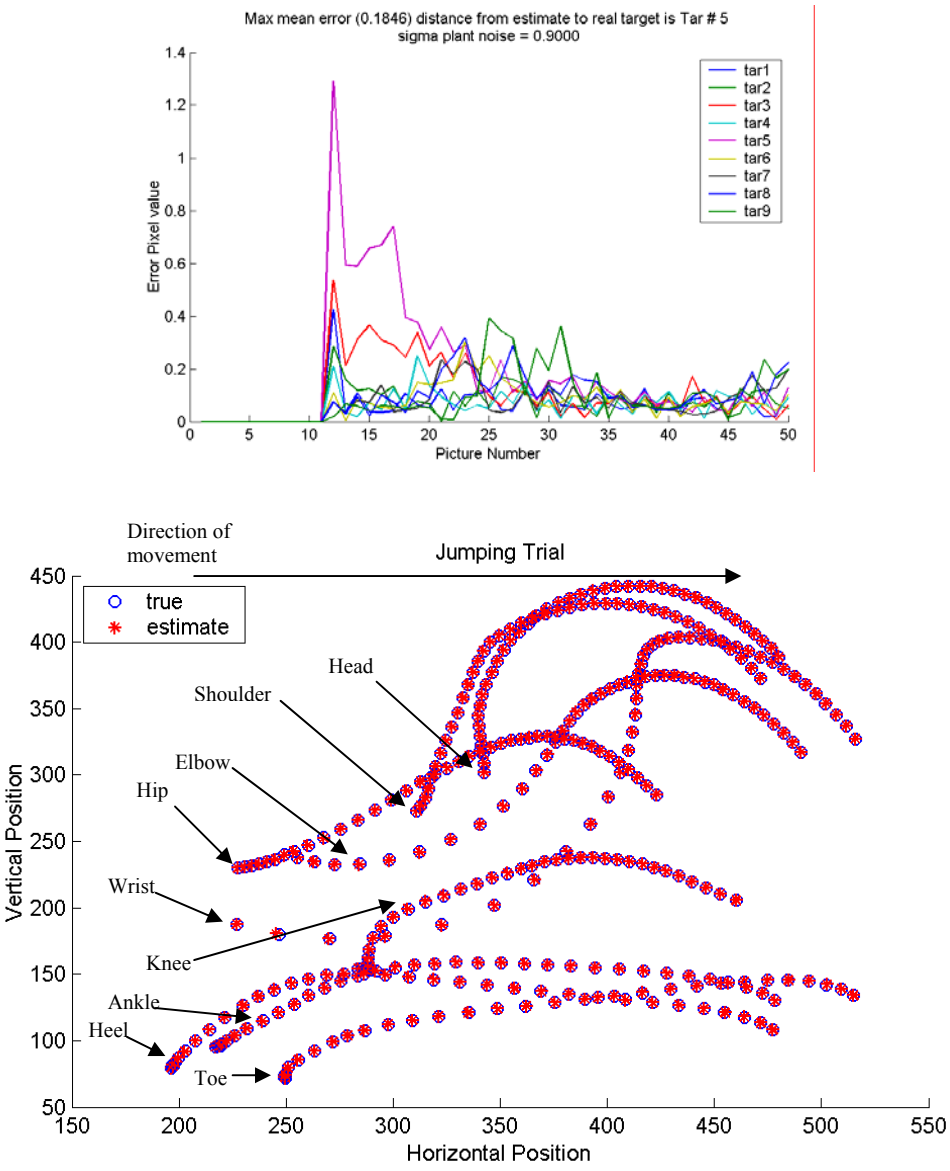


Figure 18: Acceleration jumping trial - Error plot and trajectory plot

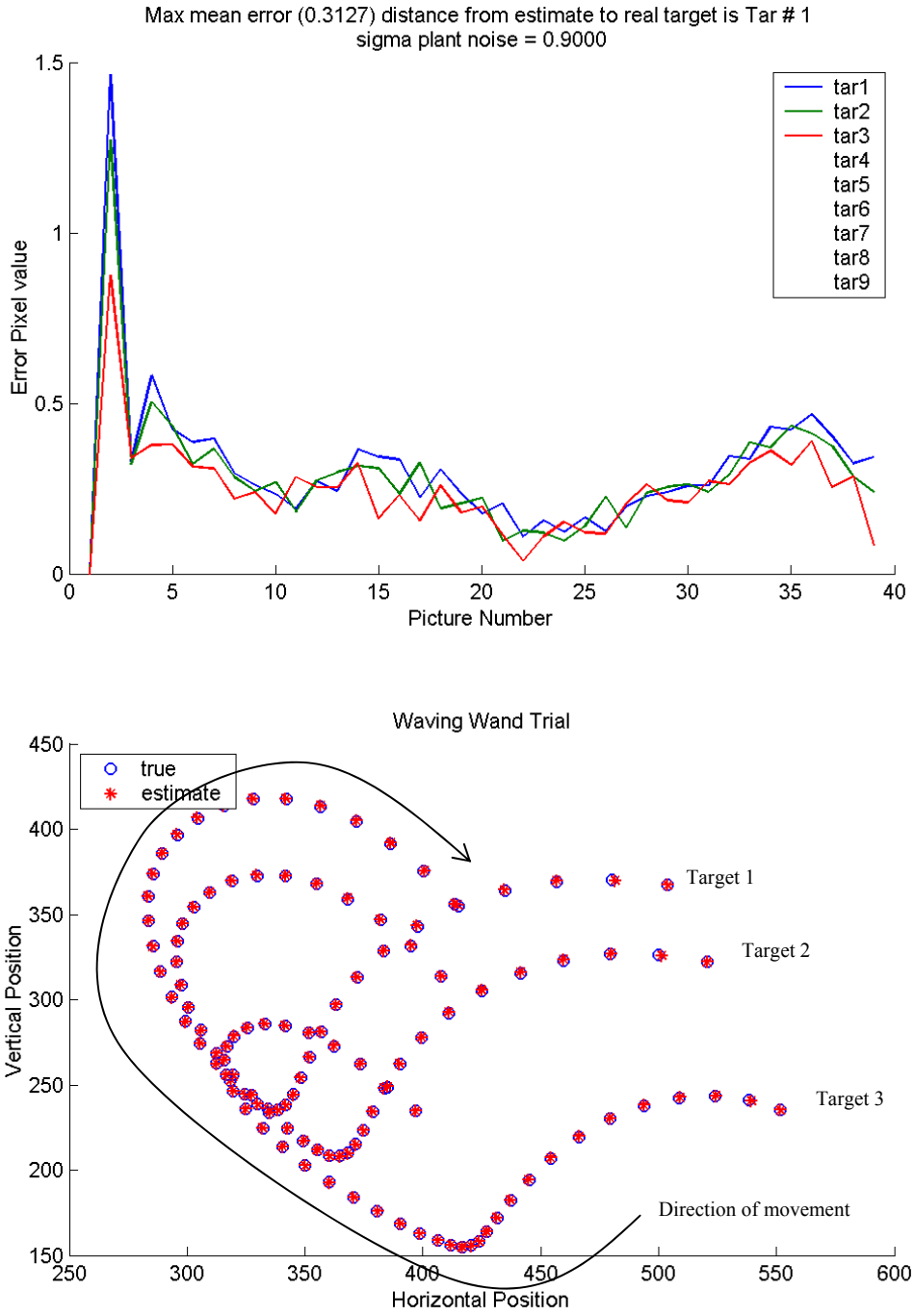


Figure 19: Acceleration wand trial - Error plot and trajectory plot

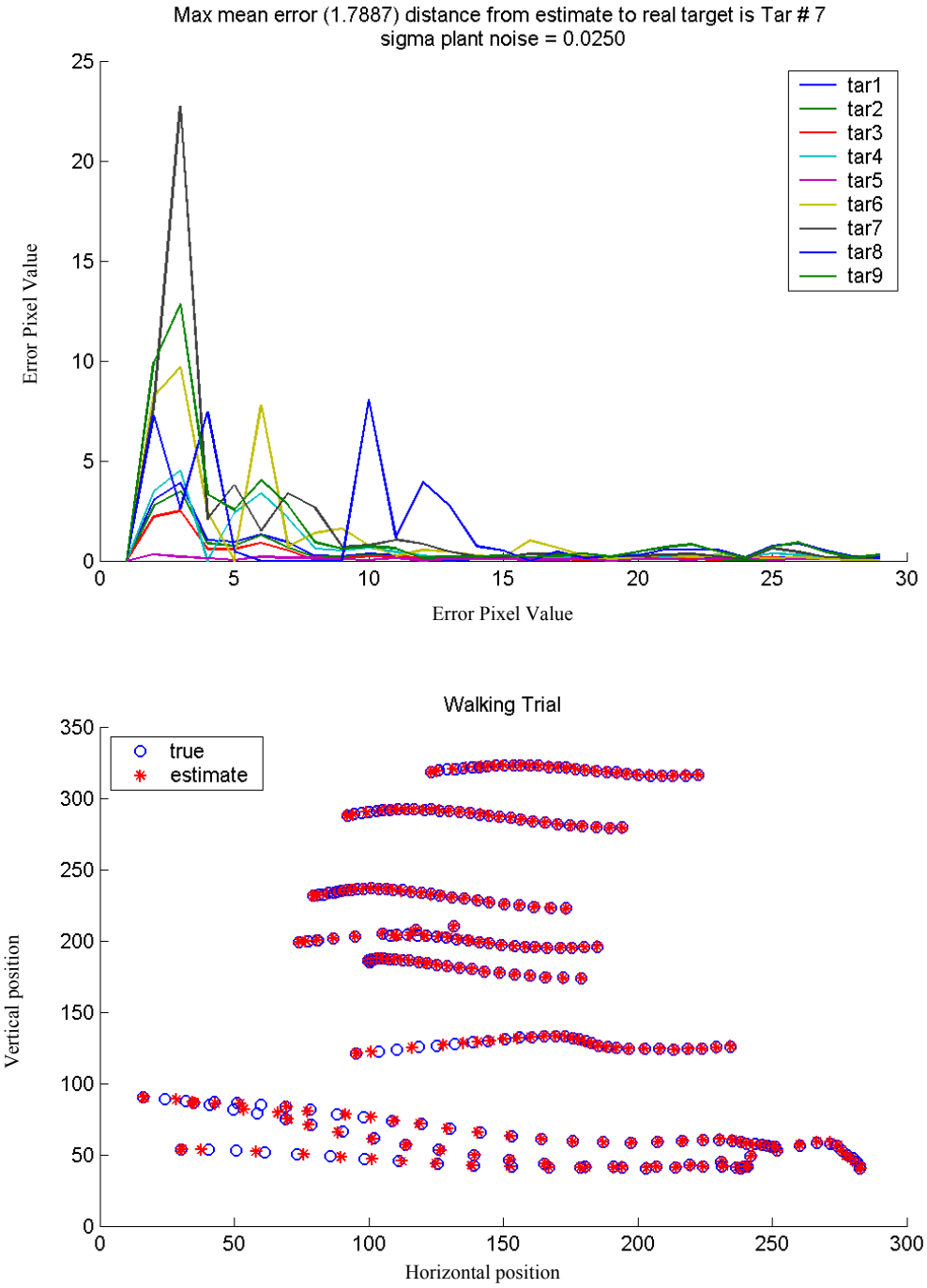


Figure 20: Jerk walking trial - Error plot and trajectory plot

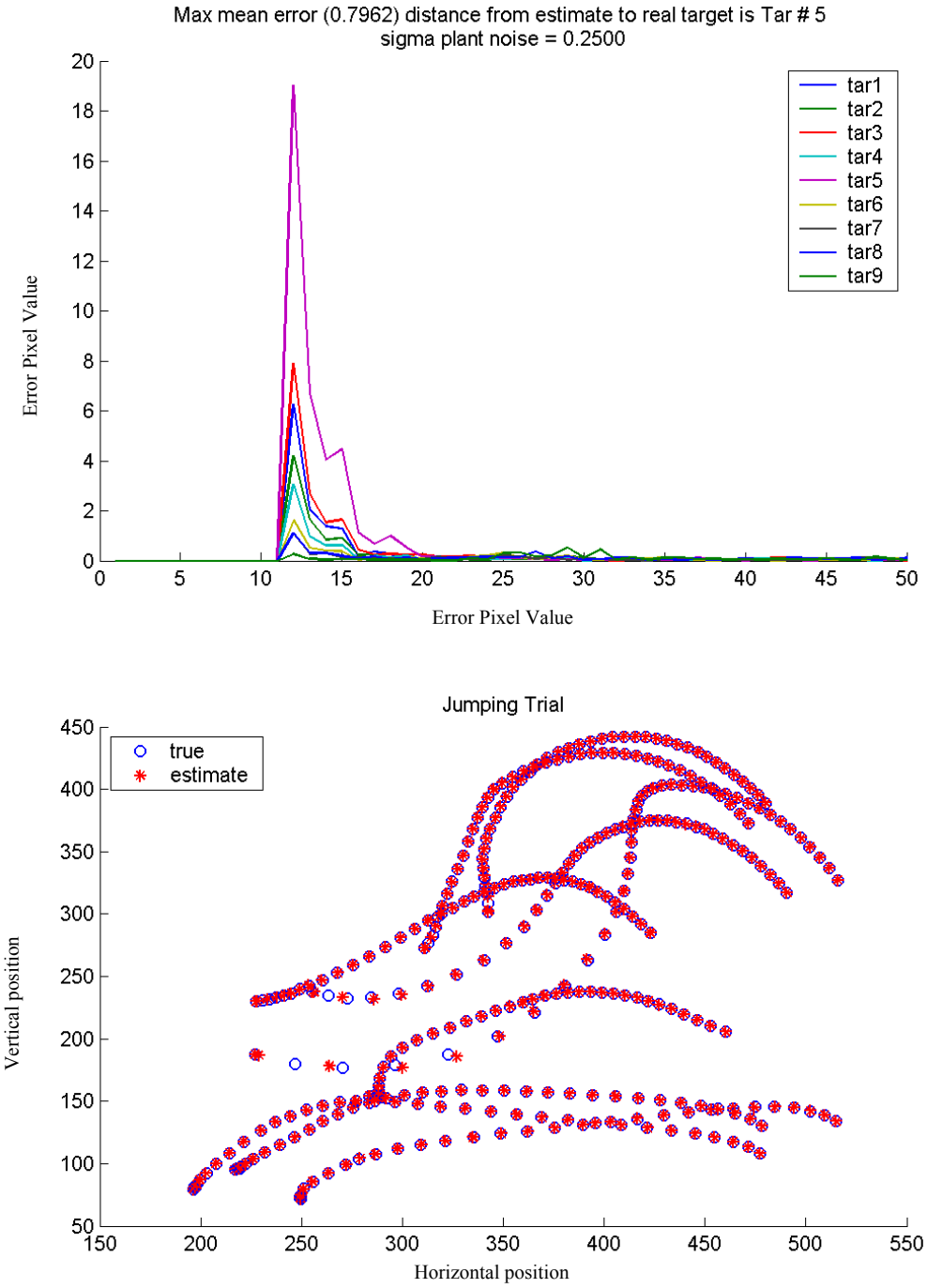


Figure 21: Jerk jumping trial - Error plot and trajectory plot

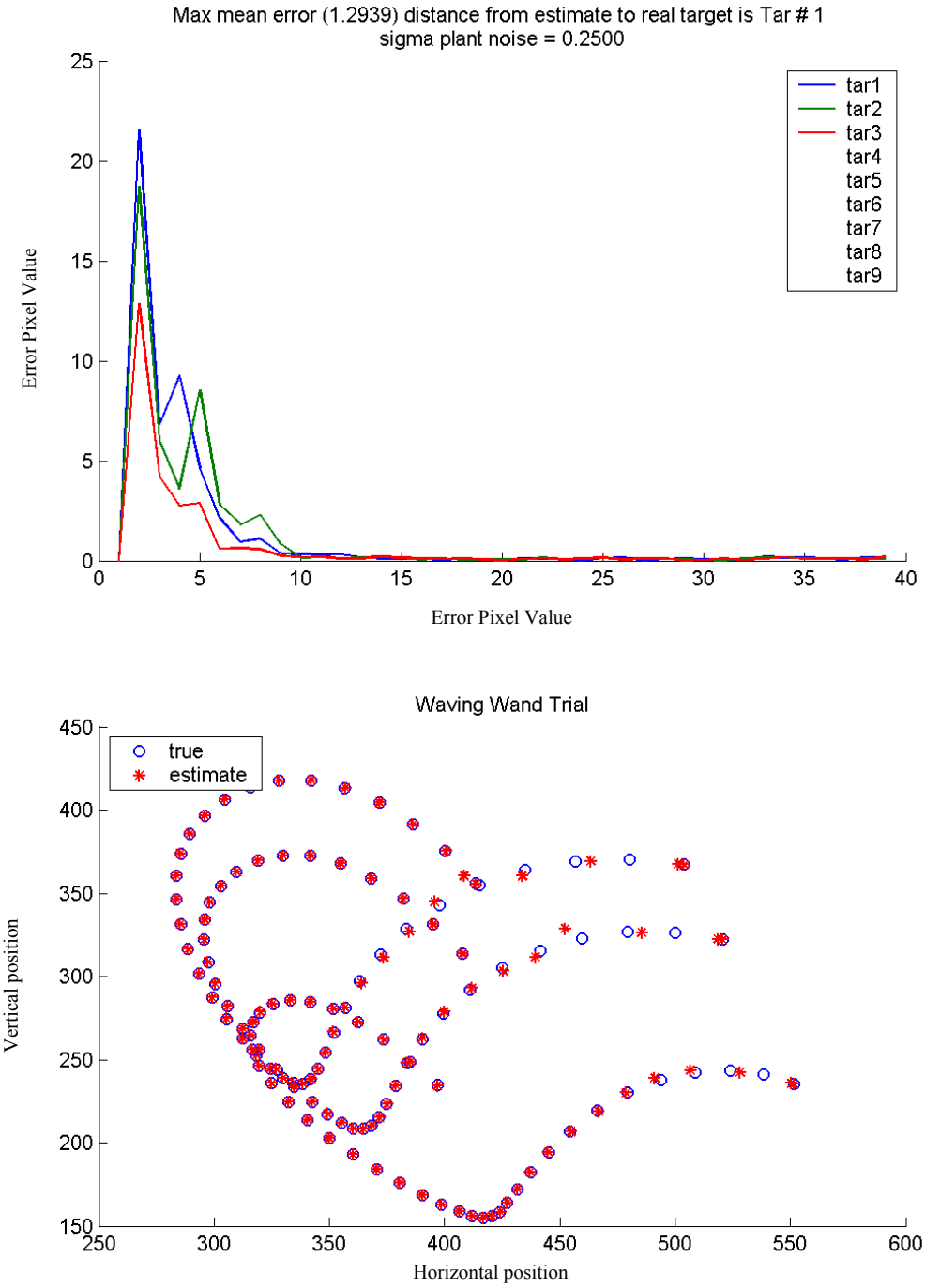


Figure 22: Jerk wand trial - Error plot and trajectory plot

Maximum Likelihood Estimation

The method of target discrimination is based on performing a two-dimensional maximum likelihood estimation on all the measured target positions and estimated positions, and then choosing the maximum likelihood of an estimated target to be assigned to a measured target position. To illustrate, Fig. 29 shows a hypothetical scenario where the measured target values are known and the assignment of the estimated target position is described. In this scenario the two ⊗ symbols represent a position in one frame in time t, and the next time, $t+1$, is where the two ○ and × symbols which represent measured and estimated positions.

As an example, see Fig. 29. In this example, there exist two targets with the trajectory as shown[1]. Note that the estimates for the next two positions of target #1 and #2 ($\times_1$ and $\times_2$) are both adjacent to measured target #1 ($\bigcirc_1$). The question to answer is: Which estimate will be assigned to target #1 and target #2? If a simple radius measure is used such as $r_{1\to 1} = \sqrt{\left(\hat{p}_x - p_{x_1}\right)^2 + \left(\hat{p}_y - p_{y_1}\right)^2}$, and $r_{2\to 1} = \sqrt{\left(\hat{p}_x - p_{x_2}\right)^2 + \left(\hat{p}_y - p_{y_2}\right)^2}$, it is clear to see, in this example, that both $\times_1$ and $\times_2$ are the same distance from $\bigcirc_1$; so the question is still, "which one should be assigned to the measured target $\bigcirc_1$?"

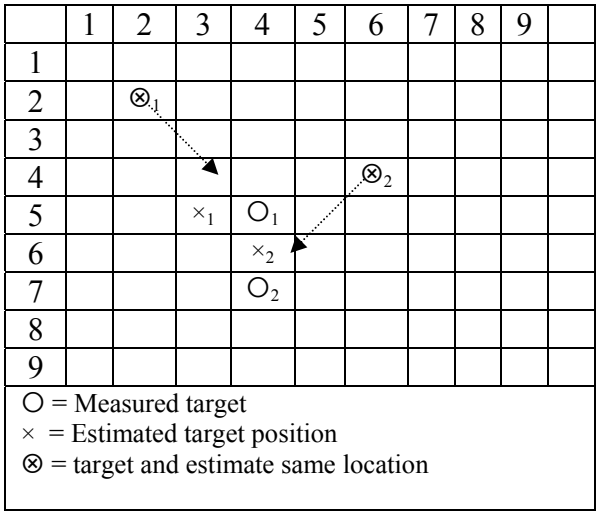


Figure 23: Discrimination example

The approach taken in this paper is to use the MLE algorithm given as [17] [18]:

$$\text{likelihood} = \frac{1}{\sqrt{(2\pi)\left|\Sigma_p\right|}} e^{\left((p-\hat{p})^T(\Sigma_p)^{-1}(p-\hat{p})/2\right)} \quad \textbf{(16)}$$

Equation 16 calculates the likelihood of each prediction/target pair. Then the pairing with maximum likelihood is chosen to be the assignment. Using a simple data example for Fig. 29 above, the data is presented for the purposes of showing the functionality of an MLE calculation:

$$\text{Target 1: } \Sigma_{p_1} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 2 \end{bmatrix}, \text{ Target 2: } \Sigma_{p_2} = \begin{bmatrix} 1 & -0.5 \\ -0.5 & 2 \end{bmatrix}$$

$$\left|\Sigma_{p_1}\right| = \left|\det(\Sigma_{p_1})\right| = 1.75, \quad \left|\Sigma_{p_2}\right| = \left|\det(\Sigma_{p_2})\right| = 1.75$$

$$\text{likelihood}_1 = \frac{1}{\sqrt{(2\pi)|1.75|}} \exp\left(\left(\begin{bmatrix} 4 \\ 5 \end{bmatrix} - \begin{bmatrix} 3 \\ 5 \end{bmatrix}\right)^T \begin{bmatrix} 1 & 0.5 \\ 0.5 & 2 \end{bmatrix}^{-1} \left(\begin{bmatrix} 4 \\ 5 \end{bmatrix} - \begin{bmatrix} 3 \\ 5 \end{bmatrix}\right) / 2\right) = 0.4037$$

$$\text{likelihood}_2 = \frac{1}{\sqrt{(2\pi)|1.75|}} \exp\left(\left(\begin{bmatrix} 4 \\ 5 \end{bmatrix} - \begin{bmatrix} 4 \\ 6 \end{bmatrix}\right)^T \begin{bmatrix} 1 & -0.5 \\ -0.5 & 2 \end{bmatrix}^{-1} \left(\begin{bmatrix} 4 \\ 5 \end{bmatrix} - \begin{bmatrix} 4 \\ 6 \end{bmatrix}\right) / 2\right) = 0.3034$$

The maximum likelihood of which likelihood value is assigned to target O_1 would be likelihood_1, which has the greatest likelihood parameter. Therefore $\times_1$ will be assigned to O_1; consequently $\times_2$ will be assigned to O_2 by using this algorithm. This is how the discrimination algorithm chooses which target estimate to be assigned to which target.

Operation of Algorithm

The algorithm used in this paper operates in the following manner: A measured target is used as the p in Eq. 16 above, and then a likelihood is calculated on all estimated target positions in a picture against this current measured target and is loaded into a matrix. The next measured target is loaded and again a likelihood is calculated for all estimated targets in the picture against this second target, and this is loaded into the matrix, see Table 4.

	Estimated targets				
Measured targets	1	2	3	4	5
1	$\text{likelihood}_{1,1}$	$\text{likelihood}_{1,2}$	$\text{likelihood}_{1,3}$	$\text{likelihood}_{1,4}$	$\text{likelihood}_{1,5}$
2	$\text{likelihood}_{2,1}$	$\text{likelihood}_{2,2}$	$\text{likelihood}_{2,3}$	$\text{likelihood}_{2,4}$	$\text{likelihood}_{2,5}$
3	$\text{likelihood}_{3,1}$	$\text{likelihood}_{3,2}$	$\text{likelihood}_{3,3}$	$\text{likelihood}_{3,4}$	$\text{likelihood}_{3,5}$
4	$\text{likelihood}_{4,1}$	$\text{likelihood}_{4,2}$	$\text{likelihood}_{4,3}$	$\text{likelihood}_{4,4}$	$\text{likelihood}_{4,5}$
5	$\text{likelihood}_{5,1}$	$\text{likelihood}_{5,2}$	$\text{likelihood}_{5,3}$	$\text{likelihood}_{5,4}$	$\text{likelihood}_{5,5}$

Table 4: MLE matrix example

All the likelihood data points are scanned for the maximum value and an assignment is made based on the location of the value. As an example, suppose that $\text{likelihood}_{3,2}$ is the greatest value; this results in target estimate # 2 being assigned to measured target # 3, and so on until all estimated targets have been assigned to a measured target. Once a measured target has been assigned to an estimated target, the measured target row is deleted as well as the estimated target column, so that another likelihood value that was calculated for that target cannot be assigned to another, see Table 5.

	Estimated targets				
Measured targets	1	2	3	4	5
1	$likelihood_{1,1}$	$likelihood_{1,2}$	0	$likelihood_{1,4}$	$likelihood_{1,5}$
2	$likelihood_{2,1}$	$likelihood_{2,2}$	0	$likelihood_{2,4}$	$likelihood_{2,5}$
3	0	0	0	0	0
4	$likelihood_{4,1}$	$likelihood_{4,2}$	0	$likelihood_{4,4}$	$likelihood_{4,5}$
5	$likelihood_{5,1}$	$likelihood_{5,2}$	0	$likelihood_{5,4}$	$likelihood_{5,5}$

Table 5: MLE matrix example continued

Note that all target # 3 likelihood values have been set to zero, so that only those likelihood values associated with a particular measured target can be assigned to that target.

This results in the ability of the algorithm to have an initial assignment of targets and to keep that assignment throughout the trial regardless of where the target is within the video field. Note previously in Chapter 4, that the video is scanned from top to bottom. If a target exists on the head and the hands, with the hands at the side target # 1 is the head. However, once the hands are raised above the head, target # 1 would now become the hands and target # 2 would be the head. The goal is to keep target # 1 as the head and target # 2 the hands regardless of their position. The discrimination part of the algorithm now has the ability to perform this function with all the previous functions being performed. As an example, see Fig. 30, in this figure a green color target is assigned to the fifth target being plotted on the jumping trial. Targets four and five are being shown in the following figures because they represent the most difficult discrimination problem in these trials.

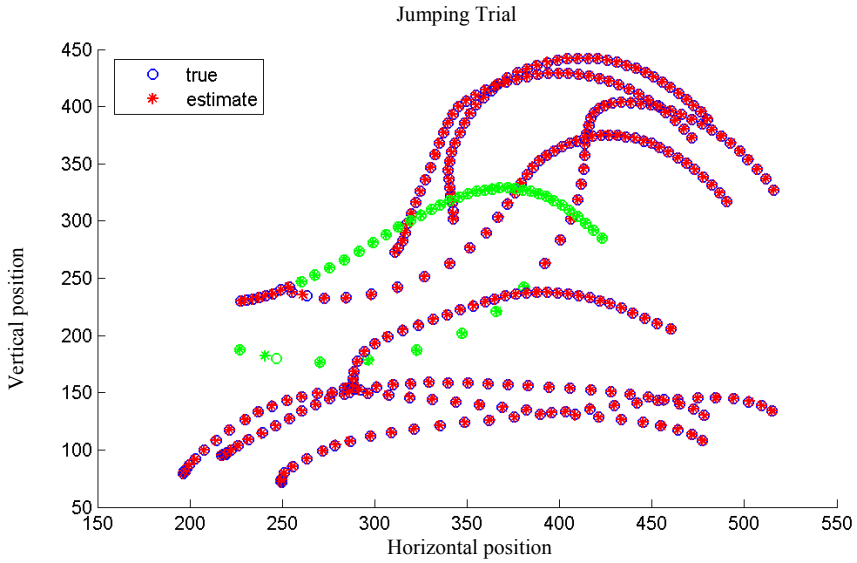


Figure 24: Target # 5 with uncorrected discrimination

Note that for the first eight frames target # 5, wrist (see Fig. 24), is correctly assigned as the fifth target. But on frame # 9 the hand goes above the hip and now the hip is the target # 5. The goal is to make the hand remain target # 5 throughout the trial.

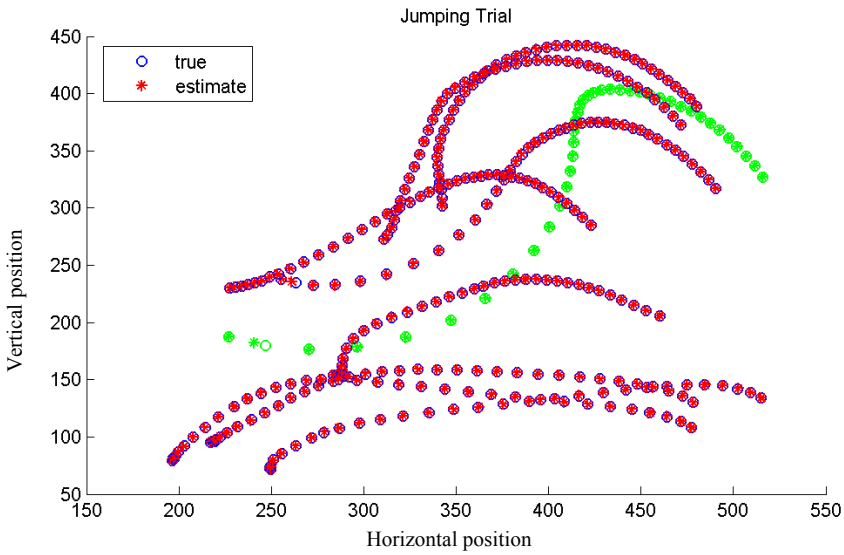


Figure 25: Target # 5 with the correct discrimination applied

Note, in Fig. 31, that with the correction of the discrimination algorithm target, number 5 is always being displayed as target # 5, even when it becomes the third target towards the end of the trial.

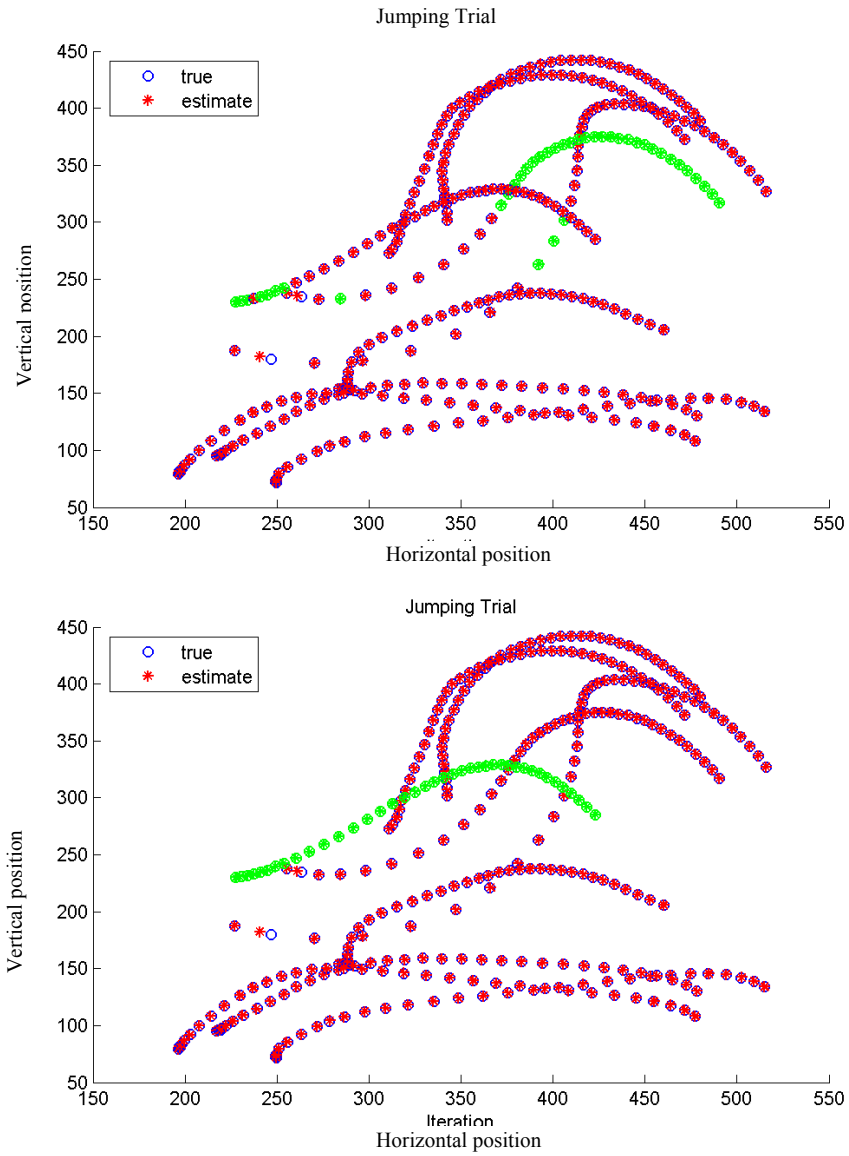


Figure 26: Target # 4 with both uncorrected and corrected discrimination

The same scenario is again shown in Fig. 32, where target # 4 is always shown correctly when the corrected discrimination is applied.

Occluded Targets

Another function being performed by the discrimination MATLAB function is to assign an estimated $\hat{p}$ to a target that has been occluded or hidden. In this manner a hidden target will continue to be estimated until the target comes back into view. This is performed in the following manner. First all likelihood values are assigned as described above; then if there is an likelihood value that is left unassigned, the $\hat{p}$ horizontal and vertical positional value is now assigned to that target as the measured value. The target continues to propagate into the future by using the last calculated values for velocity and/or acceleration. In this manner, an occluded target is still being estimated by its last calculated velocity and acceleration values until it is seen again. Figure 27 below shows a real

example of a target that becomes occluded and then reappears. This example is taken from the walking target trial where the hip is occluded by the arm for a number of frames.

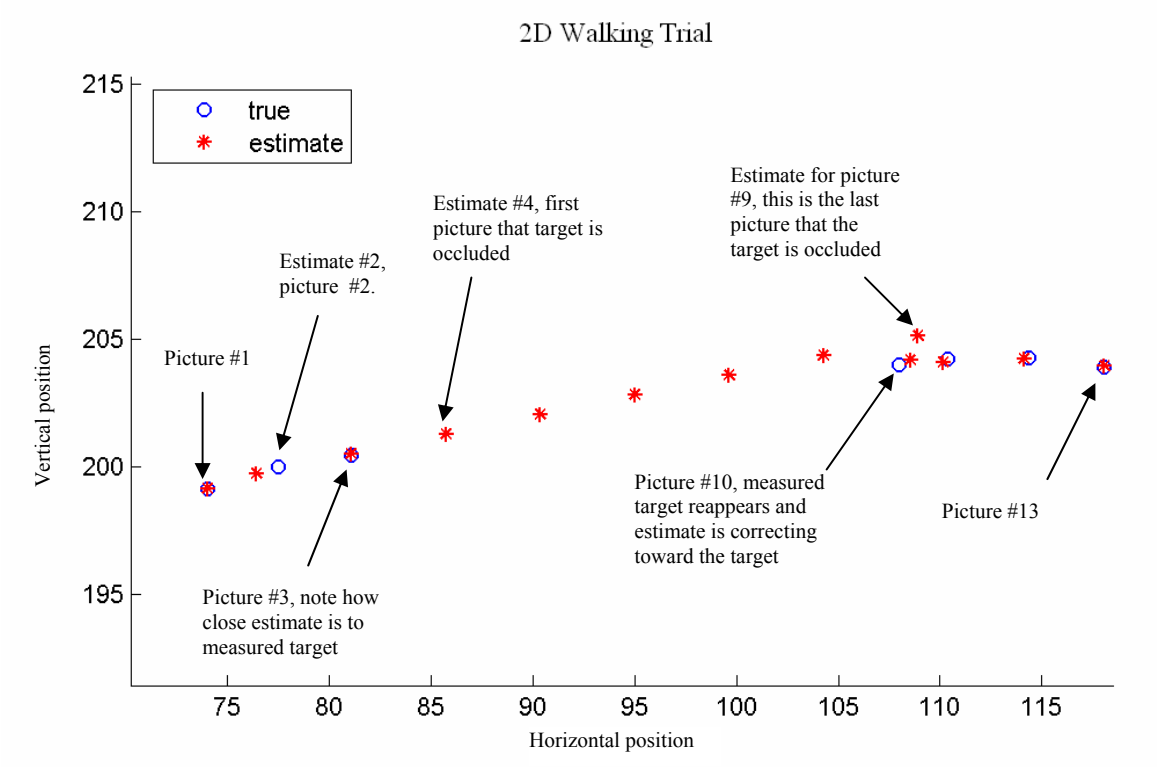


Figure 27: Occluded target example

This is a close up of target number four of the walking target trial, which shows pictures 1-13. The movement begins on the left side and is progressing to the right. In the first picture, the estimate is assigned the target measured target value. Once picture number 4 is calculated, the estimate continues to track the estimate parameters until the target re-appears at picture # 10, at which time the estimate is correcting toward the target.

The result is that targets, which are occluded or hidden, can still be estimated; and once the target reappears, the algorithm will correct to the target once again.

Discussion and Conclusion

Examine the performance output of the two models in Chapter 4, "Kalman filter," Figs. 23-28, and note that the error of the jerk model is greater than the error offered by the acceleration model. Both models had their noise standard deviation, σ_w, chosen for maximum tracking capability by recursively going through the trials with different values chosen for σ_w and measuring the error. Even with an optimum selection of σ_w, the jerk model still results in a greater error. By viewing Figs. 18, 20, and 22, note that the acceleration and jerk both appear to be noise models of the motion being measured. This is actually born out in Figs. 23-28, which plot the performance of the two models. If acceleration has the properties of white Gaussian noise, then the model, which is driven by the noisy acceleration, would be the appropriate model. Therefore, the noisy acceleration model is the most appropriate because the Kalman filter is being

used to estimate the state, which is composed of the position and velocity, both of which are parameters that can be estimated because they are not noise.

If, on the other hand, the noisy jerk model is being used, the noise component of the jerk driving the motion within the Kalman filter is estimating a system that has position, velocity, and acceleration as state variables. If acceleration is itself a white Gaussian noise, the model cannot estimate a Gaussian noise distribution within the state, which therefore leads to the error of the noisy jerk model being greater than the noisy acceleration model.

The human movement being measured in this paper presents a white Gaussian acceleration factor. If the movements within the biomechanical trials were to exhibit an acceleration parameter that did not exhibit a white noise component, it would be expected that the noisy jerk model would perform better than the acceleration model. However, in this paper, the biomechanical motion, which does not contain motion fast enough to model with the jerk as a driving parameter, is best modeled by using the noisy acceleration as the input driving force.

With the proper model chosen, the ability to discriminate the targets throughout the range of motion measurement is possible. Because the centroid algorithm and video sensor are very accurate, the model can tolerate a large spread of values for σ_w, thereby making the system less sensitive to σ_w selection, and therefore easier to tune.

One purpose of this paper was to calculate the centroid of targets within one scan or pass through the video. Another problem was the attempt to use the Kalman predictive algorithm to predict the path of each target being tracked in order to continue to track targets when they are occluded for several frames of video. The third problem addressed was the use of the MLE algorithm as a means to properly discriminate the targets throughout the range of motion; in other words, keeping the initial target assignments, whether they are target 1, target 2, or target 3, and so on throughout the range of motion.

In regard to the first goal of finding centroid within one scan or pass of the video data, the algorithm presented in this paper does calculate a 2-D centroid with only one scan of the video data.The Kalman filter algorithm presented two models, noisy acceleration and noisy jerk, that were used in order to find the most accurate model to be used on these trials. Between the two models the noisy acceleration model was the better motion model used for these human-based biomechanical motions. Occluded targets were predicted and tracked while they were not in the video picture.The use of the MLE algorithm provides a means to properly discriminate the different targets that are being tracked, resulting in targets being tracked throughout the range of motion by the correct assignment given at the beginning of the trial. We conclude that the ability to track and discriminate targets or markers attached to human motion can be reliably accomplished, even though very little information may be known about the motion beforehand.

Future Work

The algorithm worked well enough that the next stage would be an effort to implement this algorithm in hardware for real-time target tracking without relying on a post processing system. Another area of interest is to use this algorithm with multiple cameras, using the prediction and discrimination techniques to create and successfully model three-dimensional models of the human motion being observed in a much faster manner than is currently available. Another possibility of applied study is to use different target identification methods such as limb segmentation, or another form of electrical sensing that might aid in robotic measured systems. This could enable a robotic-based system to more quickly identify and track certain objects or areas of interest within the sensor input parameters. Finally, another area of future work might be using other sensor system inputs for target prediction and discrimination so that a faster system could be developed for another field that has a goal of faster target position calculation and tracking.

Bibliography

[1] M. E. Hawkins, *High Speed Target Tracking Using Kalman Filter and Partial Window Imaging*, Masters Paper, Georgia Institute of Technology, April 2002.

[2] N. Funk, "A Study of the Kalman Filter Applied to Visual Tracking," Project for CMPUT 652, University of Alberta, 2003.

[3] M. Efe, D.P. Atherton, "Adaptive Kalman Filters for Manoeuvring Target Tracking," *Proceedings of 37th IEEE Conference on Decision and Control*, 1998.

[4] Vicon Peak – Colorado, 7388 S. Revere Parkway Suite 901, Centennial, CO 80112, USA, T: +1 303 799 8686, F: +1 303 799 8690, E: info@peakperform.com

[5] R.E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82 (Series D): pp. 35-45, 1960.

[6] M. S. Grewal, and A. P. Andrews, *Kalman Filtering, Theory and Practice Using MATLAB*, 2d ed., New York: Wiley-Interscience, 2001.

[7] Chi–Tsong Chen, *Linear System Theory and Design*, 3d ed., Oxford University Press, 1999.

[8] E. Brookner, *Tracking and Kalman Filtering Made Easy*, New York: Wiley-Interscience, 1998.

[9] G. L. Plett, *ECE5530 Multivariable Control Systems II Course Reader*, University of Colorado at Colorado Springs, 2002.

[10] A. R. Washburn, A Short Introduction to Kalman Filters,
Available: http://diana.or.nps.navy.mil/~washburn/Files/KalmanIntro.pdf, accessed October 30, 2005.

[11] G. Welch, and G Bishop, An Introduction to the Kalman Filter,
Available: http://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf, accessed November 15, 2005.

[12] X. Rong Li, and V. P. Jlokov, "Survey of Maneuvering Target Tracking. Part I: Dynamic Models," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1333–64, October 2003.

[13] X. Rong Li, and V. P. Jlokov, "Survey of Maneuvering Target Tracking. Part II: Ballistic Target Models," *Proceedings of SPIE Conference on Signal and Data Processing of Small Targets*, San Diego, CA, July-August 2001. Paper 4473-63

[14] X. Rong Li, and V. P. Jlokov, "Survey of Maneuvering Target Tracking. Part III: Measurement Models," *Proceedings of SPIE Conference on Signal and Data Processing of Small Targets*, San Diego, CA, July-August 2001. Paper 4473-63.

[15] X. Rong Li, and V. P. Jlokov, "Survey of Maneuvering Target Tracking. Part IV: Decision-Based Models," *Proceedings of SPIE Conference on Signal and Data Processing of Small Targets*, Orlando, FL, April 2002. Paper 4728-60

[16] X. Rong Li, and V. P. Jlokov, "Survey of Maneuvering Target Tracking. Part V: Multiple-Model Methods," *IEEE Transactions on Aerospace and Electronic Systems*: November 2003.

[17] Y. Bar-Shalom, and X. Rong Li, T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*, Theory Algorithms and Software, New York: Wiley-Interscience, 2001.

[18] S.Purcell, Maximum Likelihood Estimation,
Available: http://statgen.iop.kcl.ac.uk/bgim/mle/sslike_1.html, accessed October 29, 2005.