Vision-Based Predictive Robotic Tracking of a Moving Target

Alison E. Hunt and Arthur C. Sanderson

CMU-RI-TR-82-15

Department of Electrical Engineering
The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

January 1982

This work was supported in part by the Westinghouse Electric Corporation and The Robotics Institute, Carnegie Mellon University.

Table of Contents

List of Figures

List of Tables

Abstract

This work represents a more general approach to robotic system design than one based on predefined responses in a controlled environment. An implementation of a vision-based robotic tracking system is presented in which target trajectory predictions enable the robot to track and intercept a moving target. A host microcomputer receives target position information from a vision module, predicts the target's trajectory, and issues tracking commands to the robot controller.

Five predictive algorithms are derived for implementation in the system, including a Kalman and an augmented Kalman filter. The use of one-step as well as absolute and relative n-step predictions is investigated. The "best predictor" algorithm is presented, by which one of the five predictions is selected to be used as the robotic tracking command.

Using data from experimental trials, predictor results are compared and robotic tracking performance and interception success are evaluated for the target both moving and after it comes to rest.

Constraints limiting the applicability of this implementation are discussed and possible improvements and extensions suggested.

Chapter 1
Background and Introduction

Current robotic research recognizes the need for increased flexibility and adaptability in robotic systems. Sophisticated robotic systems must be capable of performing a variety of tasks and of acquiring and interpreting environmental information. The next generation of robots will be equipped with sensors to enable them to respond appropriately to a changing work environment. Robotic senses such as vision, sonar, tactile and proximity sensing have been successfully implemented and continue to be an important concentration of current research on the integration of sensor based systems.

A recent emphasis of robotic research has been the development of vision based robotic control systems. Spurred by industrial requirements, much of this research has focused on the design of automatic assembly line stations. In the standard case, a stationary object, randomly located and oriented, is visually inspected and recognized by a stationary camera. The object is then located and transferred by the robot to a predetermined location [Makhlin 81, Carlisle 81]. Some implementations have extended these results by employing stationary-base line tracking to retrieve parts from a moving conveyor belt [Dawson 79, Ward 79]. Here the constant, known velocity of the belt is used to compute coordinate transformations. The robot then performs its pretaught instructions from the stationary case in a moving frame of reference. In both of these applications the robotic system is performing repetitive tasks on the basis of predefined responses to a predictable environment. The lack of sensory feedback demands strict workpiece requirements and precise control of the moving belt.

Implementations of visual servoing techniques have produced flexible robotic systems which are capable of responding to a changing, unpredictable environment [Hill 79, Weiss 81]. A mobile camera mounted on the manipulator end-effector provides visual feedback. Position and velocity correction terms enable the robot to maintain its position with respect to an object moving at an unknown velocity.

Sensor equipped mobile robots have been used to demonstrate obstacle avoidance and navigation procedures. By using visual [Moravec 80] or ultrasonic [Bauzil 81] input, robot rovers can assess their current surroundings in order to choose a navigational route.

The idea of employing a predictive algorithm to enable a robot to estimate the trajectory and thus intercept a moving object arose as a complement to and extension of these and similar efforts. Open-loop predictive tracking offers a more general approach to robot control

than one based on pretaught instructions superimposed on a moving frame of reference. The need for a predefined environment is eliminated, although by definition the target's trajectory must be predictable. It is only required, however, that there be enough knowledge of target behavior so that an appropriate model may be chosen.

Predictive robotic tracking could be of significant value when used in conjunction with visual servoing. Control dedicated solely to reducing current position and velocity errors between manipulator and target could result in the robot lagging the target. With the incorporation of predictive tracking, the robot could anticipate the target's future position and be waiting there to intercept it.

Modern predictive filtering has been widely used for tracking maneuverable vehicles in navigational and tactical applications. A common approach has been the use of Kalman filtering techniques. The Kalman filter [Kalman 60, Kalman 61] is a linear optimal recursive filter which has as its developmental roots the original concept of least squares estimation by Gauss [Sorenson 70a, Sorenson 70b] and the work of Wiener and Kolmogorov [Sorenson 70b] in the early 1940's dealing with the estimation of random signals. The recursive nature of the Kalman filter, as well as the fact that the problem is stated as a differential rather than an integral equation, makes it especially appropriate for computer implementation.

Kalman filtering has been immensely popular in aerospace applications such as in navigation and guidance. These recursive techniques have been applied in the tracking function of air-traffic control radar [Singer 71]. A predictive Kalman filter was used in the control of the Apollo spacecraft in coasting flight [Battin 70]. Predictions of the spacecraft's orbit were made periodically so that small corrections to the speed and direction could be made if the spacecraft deviated from the intended course.

Kalman filters have been implemented in a broad range of marine applications. They have been used in the tracking of submarines and as guides for scuba divers in undersea exploration [Holdsworth 70]. These filters operate on sonar observations of range and bearing.

1.1. Objectives

Today's evolving robotics technology draws on established results and ongoing research in a variety of related fields. The intent of this project was to demonstrate one possible application of established predictive techniques to vision-based robotic tracking. The primary goal defining the design and implementation of the tracking control system was the successful predictive tracking and ultimate interception by the robot of the moving target. Once this goal was attained for the selected targets, no attempts were made to further optimize the control system.

In the process of realizing the primary goal of successful tracking and interception, a secondary objective arose. Tracking performance was of course a direct function of

prediction success. Five predictive algorithms were designed and implemented in the tracking system. It became an important objective, therefore, to evaluate the predictors' relative performances under experimental conditions in order to improve robotic tracking performance.

1.2. Control System Overview

The robotic tracking system implemented is illustrated in Figure 1-1 as an open-loop digital system. The image of the moving target is sampled and undergoes feature extraction and interpretation to give the estimated current target position. Predictive algorithms use this current estimate and past estimates to predict a future target position. The processor performs a coordinate transformation on the target position prediction to yield the desired robot position. This is sent to the robot's controller where it is asynchronously sampled and used in closed-loop control of the robot joint positions.

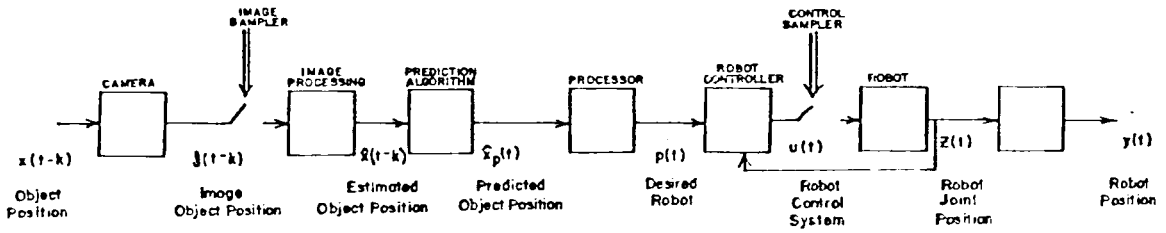

Figure 1-1: Target Tracking Control System

The total system response time is a function of both the image and robot control sampling times. The robotic positioning system accepts updates only after its output has come to a stop, and is therefore unable to synchronize with visual sampling rates.

1.3. Applications

This implementation made concessions to several timing constraints imposed by system components. The outcome of these concessions was to limit tracking experiments to slowly moving targets. It is believed, however, that the limitations of this implementation do not deny its demonstration of the potential usefulness and generality of predictive techniques for robotic tracking procedures.

Similar predictive tracking methods could easily be incorporated into existing robotic assembly stations. A generalized work station would result, eliminating the need for precise conveyor belt control and even enabling the robot to retrieve parts rolling on the belt's

surface. A work station equipped with a tracking robot could retrieve gravity fed parts, bypassing the need for conveyor belt transport altogether.

Suitably refined implementations of this concept might find applications in forms as frivolous as an improved mouse trap or a robotic tennis partner, or in more sober situations such as a robotic rescue operation for astronauts drifting into space [Heer 79] or charting migratory patterns of deep sea animals.

Chapter 2
Theory for Predictive Algorithms

Five predictive algorithms were designed to provide trajectory predictions for the robotic tracking procedure. All five theories assumed the same basic model for the trajectory of the moving target. The assumptions implicit in this model were that the target moved at nearly constant velocity and that its trajectory was approximately linear. Some algorithms rested more heavily on these assumptions than others. One predictor allowed for a parabolic trajectory, while another could account for the effect of deceleration. Ranked by their relative inflexibility to deviations from these assumptions, the five algorithms are:

- a simple two point extrapolator,
- a 1^{st} order linear regression model,
- a 2^{nd} order linear regression model,
- a 1^{st} order two dimensional Kalman filter, and
- an augmented 1^{st} order two dimensional Kalman filter.

2.1. Two Point Extrapolator

A simple but crude predictor was designed for which the trajectory was assumed linear and of constant velocity, with no allowance for deviation in either assumption.

Using the vector

$$\vec{x}_k = \begin{bmatrix} x_k \\ y_k \end{bmatrix}$$

to denote the position of the target at time k, the velocity at time k can be written as

$$\vec{\dot{x}}_k = \vec{x}_k - \vec{x}_{k-1} = \begin{bmatrix} x_k - x_{k-1} \\ y_k - y_{k-1} \end{bmatrix}$$

where the magnitude of the time interval is implicitly assumed to be unity. Using $\vec{\hat{x}}_k$ to denote the estimated or predicted value of $\vec{x}_k$, the position at time $k+1$ is given by the one-step prediction

$$\vec{\hat{x}}_{k+1} = \vec{x}_k + \vec{\dot{x}}_k = \begin{bmatrix} 2x_k - x_{k-1} \\ 2y_k - y_{k-1} \end{bmatrix} \tag{2.1}$$

The predicted position at time $k + n$, $n > 1$, is found similarly. The n-step prediction is written

$$\vec{\hat{x}}_{k+n} = \vec{x}_k + n\vec{\dot{x}}_k = \begin{bmatrix} (n+1)x_k - nx_{k-1} \\ (n+1)y_k - ny_{k-1} \end{bmatrix} \tag{2.2}$$

2.2. First Order Linear Regression Predictor

A second predictor was formulated using linear regression techniques. The best linear fit in the least squares sense to position points $\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_k$ was found at each time k, from which the predicted position at time $k+1$ was determined by extrapolation. This model allows for a linear trajectory with zero mean noise, but again assumes noiseless constant velocity.

Derivations of the equations for least-squares fit can be found in the literature [Bevington 69, Schwartz 75] but are presented here in one dimension for completeness. At time k, the i^{th} position vector $\vec{x}_{k,i}$ is modeled as a linear regression of the form

$$\vec{\hat{x}}_{k,i} = \begin{bmatrix} \hat{x}_{k,i} \\ \hat{y}_{k,i} \end{bmatrix} = \begin{bmatrix} a_k i + b_k \\ \alpha_k i + \beta_k \end{bmatrix}$$

where $i = 1, 2, \ldots, k$. The coefficients a_k and b_k vary with k, the number of data points available. Proceeding in one dimension only and omitting the subscript k, the true value x_i is expressed as the model's estimated value $\hat{x}_i$ plus an error term ε_i:

$$x_i = \hat{x}_i + \varepsilon_i$$

$$= ai + b + \varepsilon_i$$

A sum of squared errors measure of the quality of the fit of the estimates derived from the model to the set of actual data points

$$x_1, x_2, \ldots, x_k$$

can be written as

$$S = \sum_{i=1}^{k} \varepsilon_i^2$$

$$= \sum_{i=1}^{k} (x_i - ai - b)^2$$

The least squares fit is found by choosing the coefficients a and b which minimize S at each value of k. It can be shown that a minimum for S is guaranteed [Schwartz 75]. The minimizing values of a and b are found by setting partial derivatives equal to zero:

$$\frac{\partial S}{\partial a} = -2\sum_{i=1}^{k} i\,(x_i - ai - b) = 0$$

$$\frac{\partial S}{\partial b} = -2\sum_{i=1}^{k} (x_i - ai - b) = 0$$

These equations can be rearranged to yield the pair of simultaneous equations

$$\sum_{i=1}^{k} ix_i = a\sum_{i=1}^{k} i^2 + b\sum_{i=1}^{k} i$$

$$\sum_{i=1}^{k} x_i = a\sum_{i=1}^{k} i + kb$$

Solving for a and b yields

$$a = \frac{\sum ix_i - 1/k\sum x_i\sum i}{\sum i^2 - 1/k(\sum i)^2}$$

$$b = 1/k\left(\sum x_i - a\sum i\right)$$

A similar derivation yields equations for α and β in the y direction.

At each time k the coefficient pairs a_k, α_k and b_k, β_k yielding the best linear fit to data points

$$\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_k$$

are determined. Thus at time k the best (least squares) estimate of $\vec{x}_k$ is given by

$$\underset{\Delta}{\vec{x}}_k = \begin{bmatrix} a_k k + b_k \\ \alpha_k k + \beta_k \end{bmatrix}$$

The value of the next position, $\vec{x}_{k+1}$, is then predicted from a linear extrapolation of this trajectory model:

$$\underset{\sim}{\overset{\Delta}{x}}_{k+1} = \begin{bmatrix} a_k(k+1) + b_k \\ \alpha_k(k+1) + \beta_k \end{bmatrix} \tag{2.3}$$

The n-step prediction is given analogously as

$$\underset{\sim}{\overset{\Delta}{x}}_{k+n} = \begin{bmatrix} a_k(k+n) + b_k \\ \alpha_k(k+n) + \beta_k \end{bmatrix} \tag{2.4}$$

2.3. Second Order Linear Regression

An attempt was made to design a third one-step predictor which would accommodate a parabolic trajectory. This derivation is identical in form to that describing the first order linear regression except that a parabolic rather than linear model is constructed to describe the points along the trajectory.

At time k the i^{th} position vector ($i = 1, 2, ..., k$) is modeled as

$$\underset{\sim}{\overset{\Delta}{x}}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} a_k i^2 + b_k i + c_k \\ \alpha_k i^2 + \beta_k i + \gamma_k \end{bmatrix}$$

where the coefficients again vary with k.

Proceeding in a manner similar to before, the actual value of $\underset{\sim}{\overset{\Delta}{x}}_i$ in the x dimension is written (again omitting k subscripts)

$$x_i = \overset{\Delta}{x}_i + \varepsilon_i$$

$$= ai^2 + bi + c + \varepsilon_i$$

The sum of squared errors to be minimized is then

$$S = \sum_{i=1}^{k} \varepsilon_i^2$$

$$= \sum_{i=1}^{k} \left(x_i - c - bi - ai^2\right)^2$$

Taking partial derivatives and setting equal to zero:

$$\frac{\partial S}{\partial a} = -2 \sum_{i=1}^{k} i^2 \left(x_i - ai^2 - bi - c\right) = 0$$

$$\frac{\partial S}{\partial b} = -2\sum_{i=1}^{k} i\left(x_i - ai^2 - bi - c\right) = 0$$

$$\frac{\partial S}{\partial c} = -2\sum_{i=1}^{k} \left(x_i - ai^2 - bi - c\right) = 0$$

Rearranging produces

$$a\sum i^4 + b\sum i^3 + c\sum i^2 = \sum i^2 x_i$$

$$a\sum i^3 + b\sum i^2 + c\sum i = \sum i x_i$$

$$a\sum i^2 + b\sum i + ic = \sum x_i$$

This set of equations is then solved for the coefficients a, b, and c.

$$a = \frac{\left[1/k\sum i\sum x_i - \sum i x_i\right]\left[1/k\sum i^2\sum i - \sum i^3\right] - \left[1/k\sum i^2\sum x_i - \sum i^2 x_i\right]\left[1/k\sum i\sum i - \sum i^2\right]}{\left[1/k\sum i\sum i^2 - \sum i^3\right]^2 - \left[1/k\sum i^2\sum i^2 - \sum i^4\right]\left[1/k\sum i\sum i - \sum i^2\right]}$$

$$b = \frac{1/k\sum i\sum x_i - \sum i x_i - a\left[1/k\sum i\sum i^2 - \sum i^3\right]}{1/k\sum i\sum i - \sum i^2}$$

$$c = \frac{\sum x_i - b\sum i - a\sum i^2}{k}$$

Expressions for α_k, β_k, and γ_k in the y direction are found in an identical manner.

The coefficient pairs a_k, α_k; b_k, β_k ; and c_k, γ_k thus describe the least squares parabolic fit in the two independent directions at each time k. The estimated current value is

$$\underline{\hat{x}}_k = \begin{bmatrix} \hat{x}_k \\ \hat{y}_k \end{bmatrix} = \begin{bmatrix} a_k k^2 + b_k k + c_k \\ \alpha_k k^2 + \beta_k k + \gamma_k \end{bmatrix}$$

A parabolic extrapolation gives the predicted value of the position at time $k + 1$:

$$\underline{\hat{x}}_{k+1} = \begin{bmatrix} a_k(k+1)^2 + b_k(k+1) + c_k \\ \alpha_k(k+1)^2 + \beta_k(k+1) + \gamma_k \end{bmatrix} \tag{2.5}$$

An n-point extrapolation yields the n-step prediction

$$\underline{\underline{\Delta}}_{\underline{x}_{k+n}} = \begin{bmatrix} a_k(k+n)^2 + b_k(k+n) + c_k \\ \alpha_k(k+n)^2 + \beta_k(k+n) + \gamma_k \end{bmatrix} \tag{2.6}$$

2.4. Kalman Filter

The fourth predictive algorithm employs a first order two dimensional Kalman filter [Kalman 60, Kalman 61] to recursively estimate the next point of the trajectory. The two previous least squares estimators required that the entire past observed data sequence be stored and used in the computation of the new estimate as each new data sample is received. Clearly this method becomes impractical as the length of the data sequence increases. The Kalman filter generalizes the linear minimum mean square error estimation problem by allowing the parameter vector to be the state of a linear dynamic system driven by white noise. This state space approach yields recursive formulas which describe each current estimate as a function only of the previous estimate and the new data sample. Thus only the last estimate must be stored. In addition to eliminating the need for extensive data storage, the Kalman filter is computationally more efficient than non-recursive techniques (*e.g. Weiner filter*) which require inversion of large matrices [Tretter 76].

2.4.1. General Kalman Filter Theory

The one-step prediction Kalman filter equations are derived by assuming a model for the generation of the signal under observation as well as a model for its measurement process. It is typically assumed that the signal vector $\vec{s}(k)$ is the state of an N-dimensional system with a state equation of the form

$$\vec{s}(k+1) = \Phi(k+1,k)\vec{s}(k) + \vec{w}(k) \tag{2.7}$$

where $\Phi(k_1,k_2)$ is an N x N state transition matrix and $\vec{w}(k)$, representing the noise contained in the signal, is a zero-mean N-dimensional vector random process with uncorrelated elements. The noise process thus has the statistics

$$\mathrm{cov}\left[\vec{w}(k),\vec{w}(l)\right] \triangleq E\left[\vec{w}(k)\vec{w}^T(l)\right] = Q(k)\delta_{kl}$$

In observing the signal vector $\vec{s}(k)$ it is assumed that M simultaneous noisy measurements are made at time *k*. The observed data $\vec{z}(k)$ then take the form

$$\vec{z}(k) = H(k)\vec{s}(k) + \vec{n}(k) \tag{2.8}$$

where H(k) is a known M x N matrix and $\vec{n}(k)$ is a white, zero-mean, M-dimensional vector random process and is uncorrelated with the signal noise $\vec{w}(k)$:

$$\mathrm{cov}\left[\vec{n}(k),\vec{n}(l)\right] \triangleq E\left[\vec{n}(k)\vec{n}^T(l)\right] = R(k)\delta_{kl}$$

and

$$\mathrm{cov}\left[\vec{w}(k),\vec{n}(l)\right] = 0$$

The vector $\vec{n}(k)$ represents the noise introduced during measurement. The model of this system is illustrated in figure 2-1.

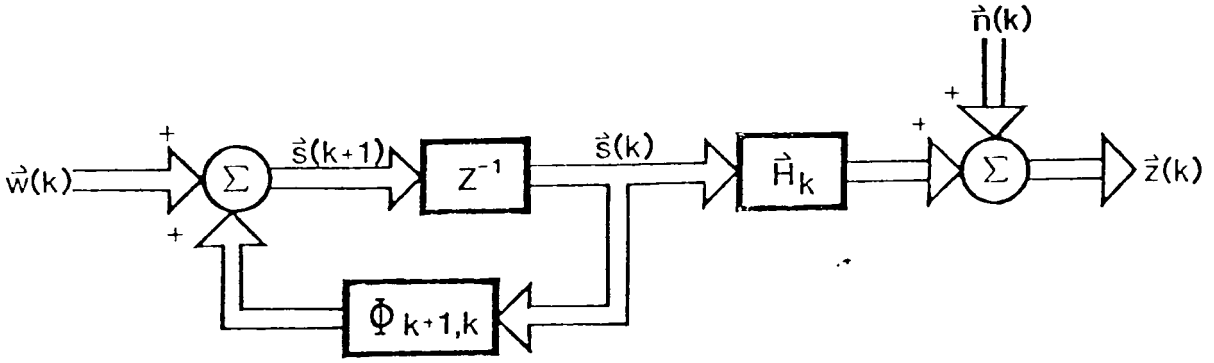

Figure 2-1: Model of dynamic system excited by random noise

Kalman's complete derivation will not be retraced here. The final recursion relations evolve from minimization of the mean-squared error by approaching the problem from the point of view of conditional distributions and expectations. This allows use of the Orthogonality Principle from probability theory [Papoulis 65] which states that when the minimum variance linear estimate $\hat{s}$ is used for s, then the error $\varepsilon = s - \hat{s}$ is orthogonal to the observed data samples x_i. The optimal estimate $\hat{s}$ can therefore be considered as a linear combination of all previous observations, or in other words, as the output of a linear filter whose input is the actual value of the observable random variables.

The three resulting equations fully describe the recursive solution to the one-step prediction problem. The optimum prediction vector is given by the recursion

$$\hat{\vec{s}}(k+1|k) = \Phi_{k+1,k}\,\hat{\vec{s}}(k|k\text{-}1) + G(k+1)\left[\vec{z}(k) - H_k\,\hat{\vec{s}}(k|k\text{-}1)\right] \tag{2.9}$$

where $G(k+1)$ is a time varying gain matrix. Thus the best prediction of $\vec{s}(k+1)$ using current data $\vec{z}(k)$ is found by extrapolating (i.e. multiplication by transition matrix $\Phi_{k+1,k}$) the previous best predicted value and then adding a correction term comprised of the error between the previous prediction and the current observation vector. The optimal estimation is therefore performed by a linear dynamic system of the same form as the model for the signal and observation random process of equation (2.7), as illustrated in Figure 2-2. The model of the message source (without the added noise) is in fact embedded in the filter. Hence for its realization the optimal filter requires only the message source model and the variable gain matrix $G(k+1)$.

An equivalent expression for $\hat{\vec{s}}(k+1|k)$ is written [Schwartz 75]

$$\hat{\vec{s}}(k+1|k) = \Phi_{k+1,k}\,\hat{\vec{s}}(k) \tag{2.10}$$

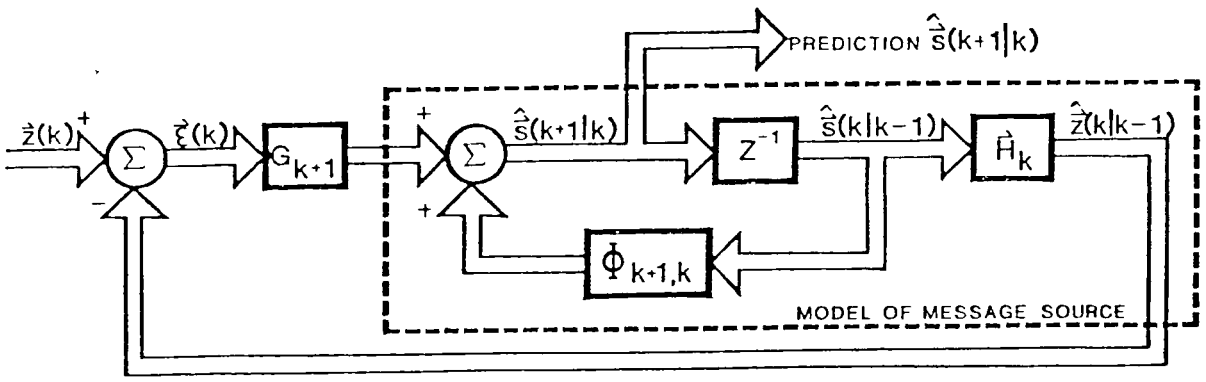

Figure 2-2: Optimum filter based on the one-step prediction algorithm

Thus the prediction $\hat{\vec{s}}(k+1|k)$ at time k is equal to the current estimate extrapolated into the future using the known signal dynamics matrix $\Phi_{k+1,k}$. This expression will be useful during the initialization procedure.

The time varying gain matrix is related recursively to the covariance matrix of the error of the predicted estimate, $V(k+1|k)$. This quantity is defined by

$$V(k+1|k) \triangleq E\left[\left(\vec{s}(k+1) - \hat{\vec{s}}(k+1|k)\right]\left[\vec{s}(k+1) - \hat{\vec{s}}(k+1|k)\right)^T\right] \tag{2.11}$$

The recursive expression for the gain matrix in terms of the covariance matrix is

$$G_{k+1} = \Phi_{k+1,k}\, V(k|k\text{-}1)H_k^T\left[H_k V(k|k\text{-}1)H_k^T + R_k\right]^{-1} \tag{2.12}$$

where R_k is the $M \times M$ additive noise covariance matrix previously defined. The covariance matrix can itself be determined recursively:

$$V(k+1|k) = \left[\Phi_{k+1,k} - G_{k+1}H_k\right]V(k|k\text{-}1)\Phi_{k+1,k}^T + Q_k \tag{2.13}$$

Q_k is the $N \times N$ signal driving-noise covariance matrix as before. Equation (2.13) can be expressed alternatively by the two equations

$$V(k+1|k) = \Phi_{k+1,k}V(k|k)\Phi_{k+1,k}^T + Q_k \tag{2.14}$$

$$V(k|k) = V(k|k\text{-}1) - V(k|k\text{-}1)H_k^T\left[H_k V(k|k\text{-}1)H_k^T + R_k\right]^{-1}H_k V(k|k\text{-}1) \tag{2.15}$$

Expressed in this form $V(k|k)$, the error covariance matrix for the filtered (current) estimate $\hat{\vec{s}}(k|k)$, is readily available. The mean squared errors can thereby be monitored as the recursive estimation proceeds. This form, however, will be used only during the initialization procedure.

Once the system is properly initialized (as explained in the following section), equations (2.9), (2.12), and (2.13) contain the relations necessary to recursively generate the solution to the one-step prediction problem. The prediction operation consists of computing the gain matrix G_{k+1} at time k from equation (2.12) and using it to update the predicted signal vector in

equation (2.9). The next value of the covariance matrix is then calculated [eq (2.13)] in order to find G_{k+2}, and the sequence is repeated.

Stability of the filter will not be discussed except to note that it is guaranteed if the signal process model is time invariant and asymptotically stable.

Although it has been implied in the preceding discussion, it is worth emphasizing that, given the message system with noise sources as assumed at the start, the Kalman filter derived is optimal among the class of all linear filters. If the additional assumption is made that all noise sources are gaussian, then the Kalman filter is optimal among all filters [Anderson 79].

2.4.2. Application of Kalman Filtering to the Robotic Tracking Problem

In the application of the Kalman filter to the trajectory prediction problem, several simplifying assumptions are made [Schwartz 75]. The system input noise $\vec{w}(k)$ and observation noise $\vec{n}(k)$ are assumed to be stationary random processes. Their respective covariance matrices, Q_k and R_k, therefore become independent of time and can be written as Q and R. In addition, the model of the signal and observation process is considered to be a time-invariant dynamical system. The transition matrix $\Phi_{k+1,k}$ and system matrix H_k are therefore represented by the constant matrices Φ and H. Under these constraints, the only dependence on time is contained in the time-varying gain matrix G_{k+1}. The Kalman filter in this form consequently converges to the time-invariant Weiner filter. Although under these constraints it yields the same results as the Weiner filter, the steady-state Kalman filter is still preferred for implementation since it bypasses the matrix spectral factorization required by the Weiner method. In cases where the observed data sequence is long and initial transients small, the time-invariant steady-state Kalman filter is a justifiable simplification which allows economical implementation [Tretter 76].

2.4.2.1. One-Step Prediction

With these assumptions, the signal model and equations (2.9), (2.12), and (2.13) defining the filter become

$$\vec{s}(k+1) = \Phi\vec{s}(k) + \vec{w}(k) \tag{2.16}$$

$$\vec{z}(k) = H\vec{s}(k) + \vec{n}(k) \tag{2.17}$$

$$\overset{\Delta}{\vec{s}}(k+1|k) = \Phi\overset{\Delta}{\vec{s}}(k|k-1) + G(k+1)\left[\vec{z}(k) - H\overset{\Delta}{\vec{s}}(k|k-1)\right] \tag{2.18}$$

$$G(k+1) = \Phi V(k|k-1)H^T\left[HV(k|k-1)H^T + R\right]^{-1} \tag{2.19}$$

$$V(k+1|k) = \left[\Phi - G(k+1)H\right]V(k|k-1)\Phi^T + Q \tag{2.20}$$

In order to implement the filter in the moving target tracking application, it is necessary first

to assume a signal model of the target's two dimensional trajectory. Four signals are to be estimated: the position coordinates $x(k)$, $y(k)$ and the velocity vector components $\overset{\circ}{x}(k)$, $\overset{\circ}{y}(k)$. The state of the system, $\vec{s}(k)$, is thus the four component vector

$$\vec{s}(k) = \begin{bmatrix} s_1(k) \\ s_2(k) \\ s_3(k) \\ s_4(k) \end{bmatrix} = \begin{bmatrix} x(k) \\ \overset{\circ}{x}(k) \\ y(k) \\ \overset{\circ}{y}(k) \end{bmatrix} \tag{2.21}$$

These signals are related by the dynamics of motion to describe the target's trajectory. Under the assumptions of constant velocity the state equations of the signal model are

$$x(k+1) = x(k) + \Delta\overset{\circ}{x}(k) + w_1(k) \tag{2.22}$$
$$\overset{\circ}{x}(k+1) = \overset{\circ}{x}(k) \tag{2.23}$$
$$y(k+1) = y(k) + \Delta\overset{\circ}{y}(k) + w_2(k) \tag{2.24}$$
$$\overset{\circ}{y}(k+1) = \overset{\circ}{y}(k) \tag{2.25}$$

where Δ is the sampling interval. The two maneuver noise terms $w_1(k)$ and $w_2(k)$ account for deviations from the assumed constant velocity trajectory.

In the vector notation of equation (2.16) the transition matrix Φ and the maneuver noise vector $\vec{w}(k)$ are now given by

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

and

$$\vec{w}(k) = \begin{bmatrix} w_1(k) \\ 0 \\ w_2(k) \\ 0 \end{bmatrix}$$

In order to satisfy the assumptions made in the derivation of the predictive filter, $w_1(k)$ and $w_2(k)$ are assumed zero-mean, white, mutually uncorrelated, stationary random variables with respective variances σ_1^2, σ_2^2. The maneuver noise covariance matrix is thus

$$Q \triangleq E\left[\vec{w}(k)\vec{w}^T(k)\right] = \begin{bmatrix} \sigma_1^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_2^2 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Measurements are made only of the x and y positions with additive noise $n_x(k)$, $n_y(k)$ respectively, giving the two dimensional measurement vector

$$\vec{z}(k) = \begin{bmatrix} z_x(k) \\ z_y(k) \end{bmatrix} = \begin{bmatrix} x(k) + n_x(k) \\ y(k) + n_y(k) \end{bmatrix} \tag{2.26}$$

The measurement vector $\vec{z}(k)$ is therefore related to the signal vector $\vec{s}(k)$ by equation (2.17)

$$\vec{z}(k) = H\vec{s}(k) + \vec{n}(k)$$

with the matrix H and vector $\vec{n}(k)$ given by

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

and

$$\vec{n}(k) = \begin{bmatrix} n_x(k) \\ n_y(k) \end{bmatrix}$$

The measurement noises $n_x(k)$ and $n_y(k)$ are assumed zero-mean, white, mutually uncorrelated stationary random variables with respective variances σ_x^2 and σ_y^2, giving the measurement noise covariance matrix

$$R \triangleq E\left[\vec{n}(k)\vec{n}^T(k)\right] = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix}$$

2.4.2.2. N-Step Prediction

The n-step Kalman filter prediction is obtained by a simple linear extrapolation of the one-step prediction. By analogy with equation (2.10) $\hat{\vec{s}}(k+1|k) = \Phi_{k+1,k}\hat{\vec{s}}(k)$ the n-step prediction is written in terms of the one-step as

$$\hat{\vec{s}}(k+n|k) = \Phi_{k+n,k+1}\hat{\vec{s}}(k+1|k) \tag{2.27}$$

where the transition matrix $\Phi_{k+n,k+1}$ is defined

$$\Phi_{k+n,k+1} = \begin{bmatrix} 1 & (n-1)\Delta & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & (n-1)\Delta \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2.4.2.3. Filter Initialization

With the signal and measurement models now established, it remains only to specify the actual values taken by the two noise covariance matrices **Q** and **R** and to specify the values of the filter's initial state. **Q** and **R** are functions only of the variances σ_1^2, σ_2^2, σ_x^2, and σ_y^2, whose specifications are discussed in Appendix A. The initialization procedure follows that outlined by Schwartz and Shaw [Schwartz 75].

The signal estimate at time $k = 2$, $\hat{\vec{s}}(2)$, is computed from the first two measurements $\vec{z}(1)$ and $\vec{z}(2)$ by means of a non-recursive two-point extrapolator identical to the predictor presented in Section 2.1. The four components of the signal at time $k = 2$ are estimated by

$$\hat{s}_1(2) = \hat{x}(2) = z_x(2) \tag{2.28}$$

$$\hat{s}_2(2) = \hat{\dot{x}}(2) = \left[z_x(2) - z_x(1)\right]/\Delta \tag{2.29}$$

$$\hat{s}_3(2) = \hat{y}(2) = z_y(2) \tag{2.30}$$

$$\hat{s}_4(2) = \hat{\dot{y}}(2) = \left[z_y(2) - z_y(1)\right]/\Delta \tag{2.31}$$

The error on this estimate is calculated to be

$$\vec{s}(2) - \hat{\vec{s}}(2) = \begin{bmatrix} x(2) - z_x(2) \\ \dot{x}(2) - \left[z_x(2) - z_x(1)\right]/\Delta \\ y(2) - z_y(2) \\ \dot{y}(2) - \left[z_y(2) - z_y(1)\right]/\Delta \end{bmatrix}$$

Recalling from the constant velocity signal model of equations (2.23) and (2.25)

$$\dot{x}(2) = \dot{x}(1) = \left[x(2) - x(1) - w_1(1)\right]/\Delta$$

$$\dot{y}(2) = \dot{y}(1) = \left[y(2) - y(1) - w_2(1)\right]/\Delta$$

and from the model of additive measurement noise of equation (2.26)

$$z_x(2) = x(2) + n_x(2)$$
$$z_y(2) = y(2) + n_y(2)$$

The error vector therefore reduces to

$$\vec{s}(2) - \overset{\Delta}{\vec{s}}(2) = \begin{bmatrix} -n_x(2) \\ \left[n_x(1) - n_x(2) - w_1(1)\right]/\Delta \\ -n_y(2) \\ \left[n_y(2) - n_y(1) - w_2(1)\right]/\Delta \end{bmatrix}$$

The previously defined covariance matrix of the filtered estimate V(2|2) (equation (2.15)) can now be evaluated element by element to give

$$v(2|2) \triangleq E\left[\left[\vec{s}(2) - \overset{\Delta}{\vec{s}}(2)\right]\left[\vec{s}(2) - \overset{\Delta}{\vec{s}}(2)\right]^T\right]$$

$$= \begin{bmatrix} \sigma_x^2 & \sigma_x^2/\Delta & 0 & 0 \\ \sigma_x^2/\Delta & \left[2\sigma_x^2 + \sigma_1^2\right]/\Delta^2 & 0 & 0 \\ 0 & 0 & \sigma_y^2 & \sigma_y^2/\Delta \\ 0 & 0 & \sigma_y^2/\Delta & \left[2\sigma_y^2 + \sigma_2^2\right]/\Delta^2 \end{bmatrix}$$

The initial value V(3|2) of the covariance matrix of the one-step prediction can now be calculated using equation (2.14) . The predicted signal $\overset{\Delta}{\vec{s}}(3|2)$ is obtained form the estimate $\overset{\Delta}{\vec{s}}(2)$ from equation (2.10). Using equation (2.12) G_4 can be found, and the next prediction $\overset{\Delta}{\vec{s}}(4|3)$ is found by equation (2.9), completing the initialization procedure. Successive predictions are computed by repeated implementation of the three recursive equations (2.9), (2.12), and (2.13).

2.5. Augmented Kalman Filter

2.5.1. One-Step Prediction

The fifth and final predictor results from modifying the first order Kalman filter developed in the previous section in order to more realistically model the effect of acceleration [Schwartz 75]. In the original derivation the signal source was modeled by the four equations (2.22) - (2.25):

$$x(k+1) = x(k) + \Delta\overset{\circ}{x}(k) + w_1(k)$$
$$\overset{\circ}{x}(k+1) = \overset{\circ}{x}(k)$$
$$y(k+1) = y(k) + \Delta\overset{\circ}{y}(k) + w_2(k)$$
$$\overset{\circ}{y}(k+1) = \overset{\circ}{y}(k)$$

The noise terms $w_1(k)$ and $w_2(k)$ represent the effect of random uncorrelated accelerations on the position coordinates x and y. (Note the two acceleration terms are actually $w_1(k)/\Delta^2$ and $w_2(k)/\Delta^2$.) If it is instead assumed that there is some memory associated with these acceleration terms, then they can be represented as remaining correlated over successive time intervals:

$$w(k+1) = \rho w(k) + q(k) \tag{2.32}$$

with $q(k)$ a zero-mean white-noise process with variance σ_q^2. The variance of $w(k)$ is now expressed $\sigma_w^2 = \sigma_q^2/(1-\rho^2)$. The parameter ρ is the degree of correlation between successive accelerations and thus can range from 0 to 1. This new definition of the dynamic behavior of $\vec{w}(k)$ together with the original four signal dynamics equations now completely describe the signal source:

$$x(k+1) = x(k) + \Delta\overset{\circ}{x}(k) + w_1(k) \tag{2.33}$$
$$\overset{\circ}{x}(k+1) = \overset{\circ}{x}(k) \tag{2.34}$$
$$w_1(k+1) = \rho_1 w_1(k) + q_1(k) \tag{2.35}$$
$$y(k+1) = y(k) + \Delta\overset{\circ}{y}(k) + w_2(k) \tag{2.36}$$
$$\overset{\circ}{y}(k+1) = \overset{\circ}{y}(k) \tag{2.37}$$
$$w_2(k+1) = \rho_2 w_2(k) + q_2(k) \tag{2.38}$$

In the original derivation where $\vec{w}(k)$ was a white noise process, the signal vector was represented by the first-order difference equation [eq (2.7)]

$$\vec{s}(k+1) = \Phi\vec{s}(k) + \vec{w}(k)$$

This relationship applies again in the correlated acceleration case through a simple augmentation of the vectors $\vec{s}(k)$ and $\vec{w}(k)$ and the signal dynamics matrix Φ. The augmented signal vector is defined by

$$\vec{s}(k) = \begin{bmatrix} x(k) \\ \overset{\circ}{x}(k) \\ w_1(k) \\ y(k) \\ \overset{\circ}{y}(k) \\ w_2(k) \end{bmatrix} \tag{2.39}$$

and the signal dynamics matrix Φ and the maneuver-noise vector $\vec{w}(k)$ are given by

$$\Phi = \begin{bmatrix} 1 & \Delta & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \rho_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & \rho_2 \end{bmatrix}$$

and

$$\begin{bmatrix} 0 \end{bmatrix}$$

$$\vec{w}(k) = \begin{bmatrix} 0 \\ q_1(k) \\ 0 \\ 0 \\ q_2(k) \end{bmatrix}$$

The maneuver noise covariance matrix is consequently augmented to

$$Q \triangleq E\left[\vec{w}(k)\vec{w}^T(k)\right] = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{q1}^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{q2}^2 \end{bmatrix}$$

As before, the observations are noisy measurements of the two signal components, with the observation vector written

$$\begin{aligned} \vec{z}(k) &= \begin{bmatrix} x(k) + n_1(k) \\ y(k) + n_2(k) \end{bmatrix} \\ &= H\vec{s}(k) + \vec{n}(k) \end{aligned} \tag{2.40}$$

where the matrix **H** is now

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

The measurement noise covariance matrix remains

$$R \triangleq E\left[\vec{n}(k)\vec{n}^T(k)\right] = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix}$$

2.5.2. N-Step Prediction

The n-step prediction is again defined as an extrapolation of the one-step prediction by the relation

$$\hat{\vec{s}}(k + n|k) = \Phi_{k+n,k+1}\hat{\vec{s}}(k + 1|k) \tag{2.41}$$

The augmented transition matrix is given by

$$\Phi_{k+n,k+1} = \begin{bmatrix} 1 & (n-1)\Delta & \sum_{m=1}^{n-1} \rho_1^{m-1} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \rho_1^{n-1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & (n-1)\Delta & \sum_{m=1}^{n-1} \rho_2^{m-1} \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & \rho_2^{n-1} \end{bmatrix}$$

2.5.3. Filter Initialization

The initialization procedure for the augmented Kalman filter is identical to that described for the original filter. The error in the signal estimate at time $k = 2$ is found to be

$$\vec{s}(2) - \overset{\Delta}{\vec{s}}(2) = \begin{bmatrix} -n_x(2) \\ \left[n_x(1) - n_x(2) - w_1(1)\right]/\Delta \\ w_1(2) \\ -n_y(2) \\ \left[n_y(1) - n_y(2) - w_2(1)\right]/\Delta \\ w_2(2) \end{bmatrix}$$

The covariance matrix of the filtered estimate $\mathbf{V}(2|2)$ can then be evaluated element by element yielding

$V(2|2) =$

$$\begin{bmatrix}
\sigma_x^2 & \sigma_x^2/\Delta & 0 & 0 & 0 & 0 \\
\sigma_x^2/\Delta & \left[2\sigma_x^2+\sigma_1^2\right]/\Delta^2 & -\rho_1\sigma_1^2/\Delta & 0 & 0 & 0 \\
0 & -\rho_1\sigma_1^2/\Delta & \sigma_1^2 & 0 & 0 & 0 \\
0 & 0 & 0 & \sigma_y^2 & \sigma_y^2/\Delta & 0 \\
0 & 0 & 0 & \sigma_y^2/\Delta & \left[2\sigma_y^2+\sigma_2^2\right]/\Delta^2 & -\rho_2\sigma_2^2/\Delta \\
0 & 0 & 0 & 0 & -\rho_2\sigma_2^2/\Delta & \sigma_2^2
\end{bmatrix}$$

Chapter 3
Real Time Robotic Tracking Procedure

3.1. Hardware Configuration

The predictive control algorithms for robotic tracking were implemented on existing hardware. Control programs were written in *C* on a DEC Vax 780 system running *Unix* and downloaded by serial line to a DEC PDP 11/23 with 56K bytes available memory. The station 11/23 ran the control program which communicated over 9600 baud serial lines with a M.I.C. VS-100 Vision System and a Unimation Puma 500 robot. Both the vision and the robot systems are stand alone systems accommodating terminal input. Communication between the 11/23 and the two devices was accomplished through interrupt driven terminal emulator i/o routines.

The M.I.C. Vision System consists of a GE 2500 camera with a 16 mm. lens, an image processing unit and a DEC LSI-11 microcomputer. When a request is issued from the 11/23, the vision system thresholds a (256 x 256) video gray-level image into a run length encoded binary image and returns the two dimensional centroid of the object. Because the real time requirements of this application favor minimizing centroid calculation time, the *Count Bits* option was selected rather than the more time consuming *Connectivity Analysis* and *Object Recognition* options. With the *Count Bits* option, the entire field of view is treated as one object for the statistical analysis. There is therefore no need for pattern recognition, but care must be taken to ensure proper threshold and window adjustment so that only the target is within the field of view.

The camera was mounted on a stationary base 140 cm above the plane of the object's trajectory, providing an 80 x 80 cm^2 field of view. At this height, approximately 29.6 units in the camera coordinate system correspond to 1 mm. in the plane of motion, or approximately 300 camera units to 1 cm.

The Unimation Puma 500 is a five axis machine which has as its own dedicated computer controller a DEC LSI-11 running the *VAL* language. Its servo system consists of a microprocessor and power amplifier for each joint. The Puma 500 has a reach of 0.86 meter and can attain tip speeds of up to 1 m/s.

It was anticipated that, in the process of tracking the target, the robot's arm would come between the target and the camera, thus obscuring the view. To avoid this problem a thin

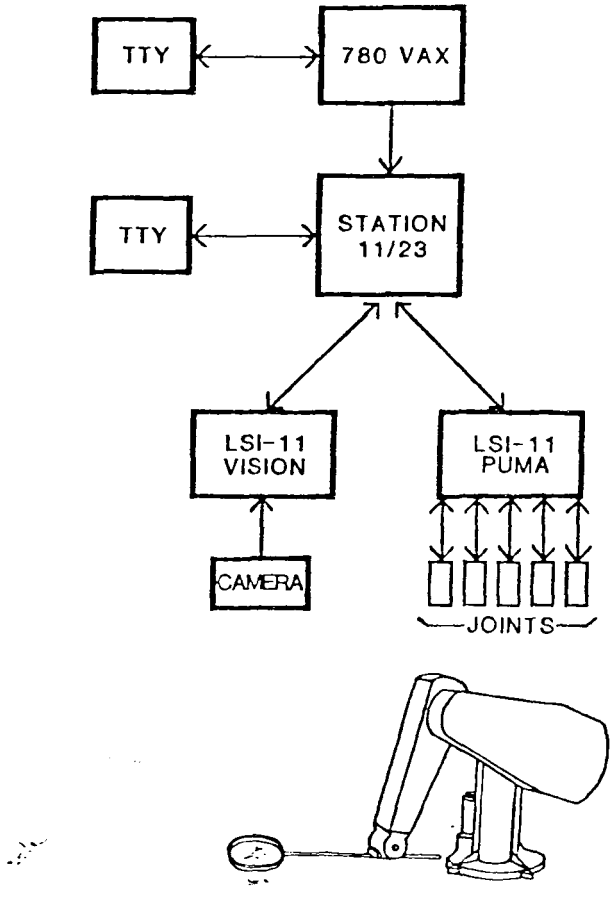

Figure 3-1: Hardware Components

extension was mounted on the end effector, allowing partial but sufficient view of the target even when directly underneath the arm extension. Due to the use of the *Count Bits* option, pictures taken of the partially obscured target contributed more noise to that already inherent in the centroid calculation. A hoop 9.5 cm in diameter was attached to the end of the extension. During tracking procedures the robot arm was instructed to maintain a height so that the hoop hovered closely above the target. When instructed to intercept the target, the hoop was lowered. A successful interception was defined as one in which the lowered hoop encircled the target.

Wind-up toys approximately 5.5 cm in diameter were used as targets for the two-dimensional tracking experiments. Their trajectories were roughly linear, but were very noisy and sometimes biased in one direction. With the given hoop and toy dimensions, there was an allowable error between the centers of the hoop and target of 2 cm. in any direction. Interception was successful, therefore, if the robot's actualization of the predicted target position was less than 600 camera units from the target. The success or failure of interception thus provided a general measure of the predictive algorithm effectiveness.

3.2. Software Control System

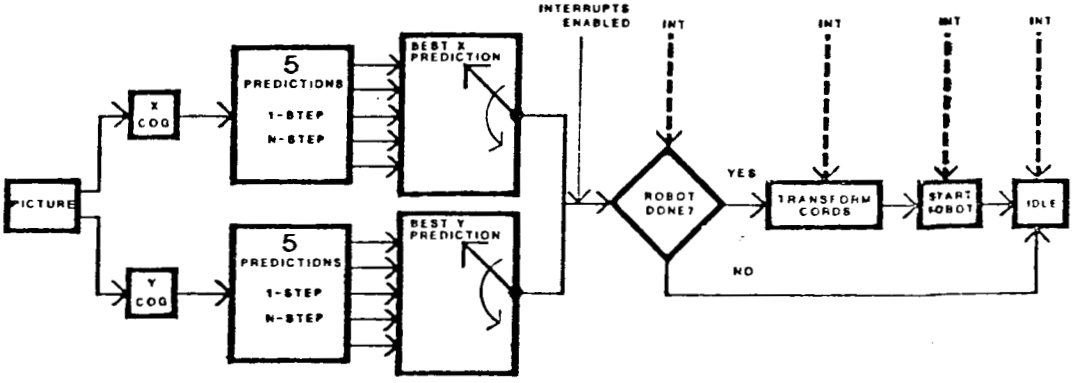

Figure 3-2: Robotic Tracking Control System

The software system implemented for real time tracking is illustrated in Figure 3-2. At each clock interrupt the control loop is reentered and a *Take Picture* command is issued over the serial line to the MIC vision system. The vision system takes a picture of the moving target and returns its two dimensional center of gravity. Using the current and past target positions, or in the case of the Kalman filters, the current position and previous filter output, predictions of the *x* and *y* coordinates of the next position are made by each of the five predictors. Each predictor in addition updates its n-step prediction. On the basis of past predictor performance and predefined prediction step, a 'best' prediction is then selected to be used for tracking. At this point in the flow graph interrupts are re-enabled.

Since all predictions are computed in the camera coordinate system, the prediction to be used for tracking is transformed into the robot's frame of reference. A sequence of *Val* commands to the Puma controller is then initiated over the serial line. These commands send the desired robot position to the Puma and instruct it to begin moving. The control loop is exited without waiting for the robot to signal completion of movement.

At the next clock interrupt, the control loop is reentered, a new picture taken, best prediction made, and clock interrupts re-enabled. If the previous iteration had been interrupted during coordinate transformation or communication with the Puma, execution is simply resumed at that point. Otherwise, the status of the robot arm is examined. If the arm has completed its last *MOVE* instruction and is now at rest, a transformation of this iteration's best prediction is computed and new *MOVE* instructions to the robot initiated. If the arm is still moving from instructions of the previous iteration, the loop exits. The processor then remains idle until the next clock interrupt.

The decision to allow idle time in the control loop was made on the assumption that the positioning instructions to the robot would not be completed in one iteration. Once this series of instructions is begun, the next desired robot position is defined to be the last prediction

made. Rather than continually polling the robot and potentially entering the positioning instruction phase late in the cycle, thus committing the robot to move to the current prediction, if on initial inspection the robot is still busy the control loop exits and waits for its next picture and prediction cycle. When the robot is next polled and found idle, its positioning instructions can be initiated with the updated prediction.

3.3. Timing Considerations

Timing considerations played a significant role in the design of the overall control system and in the decision to employ n-step predictions. The timing constraints imposed by the use of both the M.I.C. vision system and the Puma robot system were considerable but were accepted as given. Although their effect was to limit implementation to the tracking of very slowly moving targets, it was not to the detriment of the generality and potential applicability of this research.

FUNCTION		TIME (MSEC)
TAKE PICTURE		232
RETURN X-Y CENTROID		270
PREDICTIONS:	2-pt Extrapolator	5
	Linear Reg.	11
	2nd Order Reg.	20
	Kalman Filter	54
	Aug. Kalman Filter	128
BEST PREDICTION SELECTION AND COORD. TRANSFORMATION		20
ROBOT COMMANDS		504

Table 3-1: Timing Requirements for Tracking

Typical time requirements for each phase of the tracking procedure are presented in Table 3-1. The computation times required by each predictor are listed individually. The value given for the robot instructions indicates only the time required to send the desired position instruction to the Puma controller and not the robot's transit time.

It is clearly the processes of picture taking and sending commands to the robot that determine the order of magnitude of the sampling time. It was desirable both to use a high picture sampling rate for prediction accuracy and also to pass as many predictions as possible to the robot as desired position instructions. The result of this trade-off was to sample the picture once per second. This sampling interval ensured that the phases of picture taking and predicting were completed before the next interrupt, a necessary condition both for data integrity and also because the recursive filters required the previous output in order to compute the current prediction. Since communication with the Puma required a

major portion of the total loop time and because it was not necessary to move the robot at each prediction, interrupts were enabled after the prediction phase. As a consequence of this choice of sampling interval, the robot in general completed one position command for every two picture and prediction iterations.

3.4. User-Selected Parameters

Design of the robotic tracking algorithm was governed by two principal objectives. The primary goal was the successful predictive tracking and ultimate interception by the robot of the moving target. An important consequence of this aim, the monitoring and comparison of the success of the five predictors, served as the second objective. In order to achieve these goals, flexibility was required in the choice of parameters governing prediction. This was accomplished through two software options available to the user at run-time: the choice of performing a relative or absolute n-step prediction, and the automatic, continuously updated selection of the 'best' predictor.

3.4.1. N - Step Prediction

The n-step predictor option was necessitated by immutable timing constraints. The most significant obstacle influencing performance of the robotic tracking system is the copious timing overhead required for communication with both the vision system and the robot controller. Additional timing expenditure results from the inability of the Puma controller to accept a new *MOVE* instruction unless it has completed the previous one. Because of these timing demands, by the time a robot movement is completed, the prediction upon which it was based is several iterations out of date. When using one-step predictions the robot in fact appears to follow the target rather than anticipate its course.

To compensate for this lag, software options were implemented to permit user selection of the value of the prediction step, n. The first method employs the relative n-step predictors. At each time t, t = 0, 1, 2, ..., this method predicts the position the target will take at time $t + n$. When the choice of predictor step size is appropriate for the time required for robotic interpretation of the prediction, the robot is successful in anticipating the target's trajectory.

The alternate method enabled absolute n-step predictions. Here the target position at time t = n is computed, being continually updated as t approached n. This is accomplished by a succession of n, n-1, n-2, ..., 1 -step predictions and thus is equivalent to a relative (n-t) -step predictor. The purpose of this approach is to reduce the effect of robot transit time by prepositioning the robot with the first few predictions, many seconds before the predefined interception time. The robot's position can then be fine-tuned with each successive iteration as the predictions become increasingly accurate without expending the time required to move the robot large distances.

3.4.2. Current Best Predictor

The second option, the selection of the 'best' predictor at each time t, is valuable in the achievement of both goals. Through its continuously updated selection of best predictor, data are provided to evaluate the relative success of the various predictive algorithms. The obvious benefit, however, is the use of the current best predictor to improve tracking performance.

The 'best' predictor is defined in the following way. The sum of one-step prediction errors squared over the last user-selected M iterations is computed for each predictor at each time t. The x and y component errors of each prediction are computed independently. The prediction components incurring the least error in their respective direction are selected to define the current best prediction vector. This predictor is then used to compute the next n-step prediction to be sent as tracking commands to the robot. The choice of the best prediction vector therefore assumes that the component predictors which have performed best in the recent past are the most likely to perform well in the immediate future.

This algorithm's results are perfect only if the first predictor selected continues to yield the least prediction error throughout the entire experiment. This condition is rarely met. The principal intent of this selection procedure, however, is to provide a simple and fast means by which predictions with consistently gross errors are rejected as robot position instructions. With this intent its main requisite, the best predictor algorithm is expected to yield satisfactory results. It is in the very least a more intuitively satisfying method of predictor selection than one based on arbitrary choices or user prejudices.

3.5. Coordinate Transformation

All predictions are computed in the camera coordinate system. Before being sent as tracking instructions, therefore, the best n-step prediction first must be transformed into its equivalent representation in the robot frame of reference. A set of transformation matrices provides translation, rotation and scaling relations between the two coordinate systems. These relations are illustrated in Figure 3-3.

The original camera coordinate system is represented by the x_c, y_c axes. The desired transformation is to the robot coordinate system, represented by the x_r, y_r axes. With h and k denoting the x and y respective translations, m and n the scale factors, and λ the angle of rotation between the two coordinate systems, the robot system is defined in terms of the camera system by

$$\begin{bmatrix} x_r \\ y_r \end{bmatrix} = \begin{bmatrix} \cos\lambda & \sin\lambda \\ -\sin\lambda & \cos\lambda \end{bmatrix} \begin{bmatrix} m & 0 \\ 0 & n \end{bmatrix} \begin{bmatrix} x_c + h \\ y_c + k \end{bmatrix}$$

The values of the five transformation parameters are defined during program initialization.

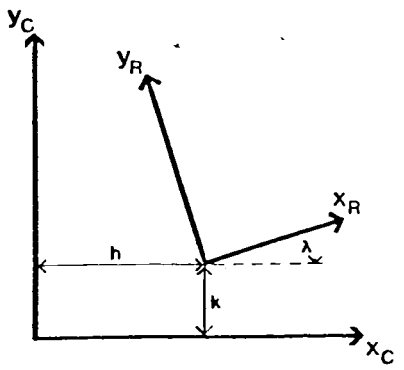

Figure 3-3: Illustration of Camera to Robot Coordinate Transformation

A picture is taken of the location of the robot's end effector in three predefined positions. The resulting three matrix equations of the form given above with x_r, y_r and x_c, y_c known are then solved to give values for λ, m, n, h, and k.

The generality of this transformation provides freedom in the placement of the camera while ensuring consistent inter-experiment results.

Chapter 4
Robotic Tracking Results

Two wind-up toys were used as targets in a total of 15 tracking experiments. Target 1 (toy robot) was chosen for its linear trajectory. The second target (toy turtle) had a frequent though inconsistent parabolic trend, and was selected to demonstrate the effect of such a bias on the linear predictors. Each target had undergone a series of preliminary tracking trials to provide data for the computation of the noise covariance matrices used in both the Kalman and the augmented Kalman filters.

Specific values of the user-selected options were chosen to typify the standard experimental trial. The value of the relative n-step prediction was standardized to N = 3. This step value provided sufficient prediction lead time for the robot's delayed actualization of the predicted position still to anticipate the target's path. The value of M = 2 was chosen as the standard window size over previous iterations used in determining the best predictor. Preliminary experiments showed this value to be a satisfactory compromise between the improved smoothing offered by a larger window and the increased response time of a smaller one. Systematic variations on the standard parameters were implemented in several trials to investigate the effect of the other available options.

In every one of the 15 experimental runs discussed here the robot successfully intercepted the target. Thus half of the first objective is accomplished. Attainment of the remaining goals, the evaluation of tracking performance and the comparison of the five predictors, requires data from predictions made both while the target was moving and after it had come to rest. In every trial where the time of interception was under real time user control[1], therefore, the robot was instructed to 'catch' the target and thus terminate data collection only after the object had been at rest for several seconds. Thus in 10 of the 15 successful interceptions the target was stationary when caught. The success of these stationary interceptions, however, is not trivial. Since all predictions from each iteration were saved, the outcome of a hypothetical interception attempt at any time, target moving or stationary, can be determined a posteriori. Thus no information has been lost. The advantage to this procedure, as stated before, was the collection of data enabling analysis of the predictors' outcomes both with target moving and stopped. This consideration was a relatively minor one during the design stage of the research, and therefore did not influence the choice of predictors implemented. It became

---

[1]That is, in every trial where the relative n-step prediction had been selected. The absolute n-step prediction by definition required that the time of interception be known a priori.

increasingly important, however, as the data collection proceeded due to 'toy fatigue'. After repeated trials the spring mechanisms used to propel each target began to lose their force. Consequently for a given time of interception, a target which was in motion during an early trial may already have come to rest at a later one. It was important, therefore, to ensure that the best predictor and hence the robot would behave appropriately in both situations.

As a result of having broadened the definition of successful interception to include interception of the stationary as well as moving target, data were collected to evaluate the success of the remaining goals. Robotic tracking performance depended on the choice of best predictor and on the ability of the Puma controller and robot to actualize its instructed desired position. Success of the best predictor in turn was a function of both the best predictor algorithm and the outcomes of the five predictors. Modification of the Puma system being beyond the scope of this research, its performance was accepted as given. It therefore remains to examine the individual predictor outcomes, choice of best predictor and ultimate robot tracking motions to achieve the original objectives.

4.1. Predictor Results

The design of four of the five predictors assumed a linear trajectory. The remaining parabolic predictor is linear in the degenerate case. Each predictor would be expected to yield perfect results if the target's path were truly linear, a hypothesis substantiated by preliminary prediction experiments on artificial linear data. The comparison of the five predictors becomes interesting in the experimental situation when the target trajectory deviates from its theoretical linearity. There are two independent criteria to be used as the bases of evaluation of predictor performance. The first criterion is the predictors' relative ability to accommodate additive noise, whether biased or white, in the linear path of the moving target. The second is their relative success in determining when the target comes to rest. It would seem as though these two criteria are contradictory, as will be seen often to be the case.

4.1.1. General Characterization

Prediction data obtained from a representative tracking experiment (target 2, trial 1) are presented in Figures 4-1 through 4-7, pages 31 - 35, 41 - 42. All figure scales are in camera units, where approximately 300 units correspond to 1 cm. In the graphs displaying one dimensional one-step and relative n-step predictions versus time, the true target position at time k as well as the five predictions of the position the target would take at that time are plotted at $t = k$. In the relative n-step case, for example, the predictions plotted at $t = k$ were actually made at time $t = k\text{-}n$, and are a function only of the target positions up to that time. They are plotted at $t = k$ with the k^{th} target position to allow easy visual interpretation of the prediction errors.

The prediction parameters in this example are of the standard form: a 3-step prediction

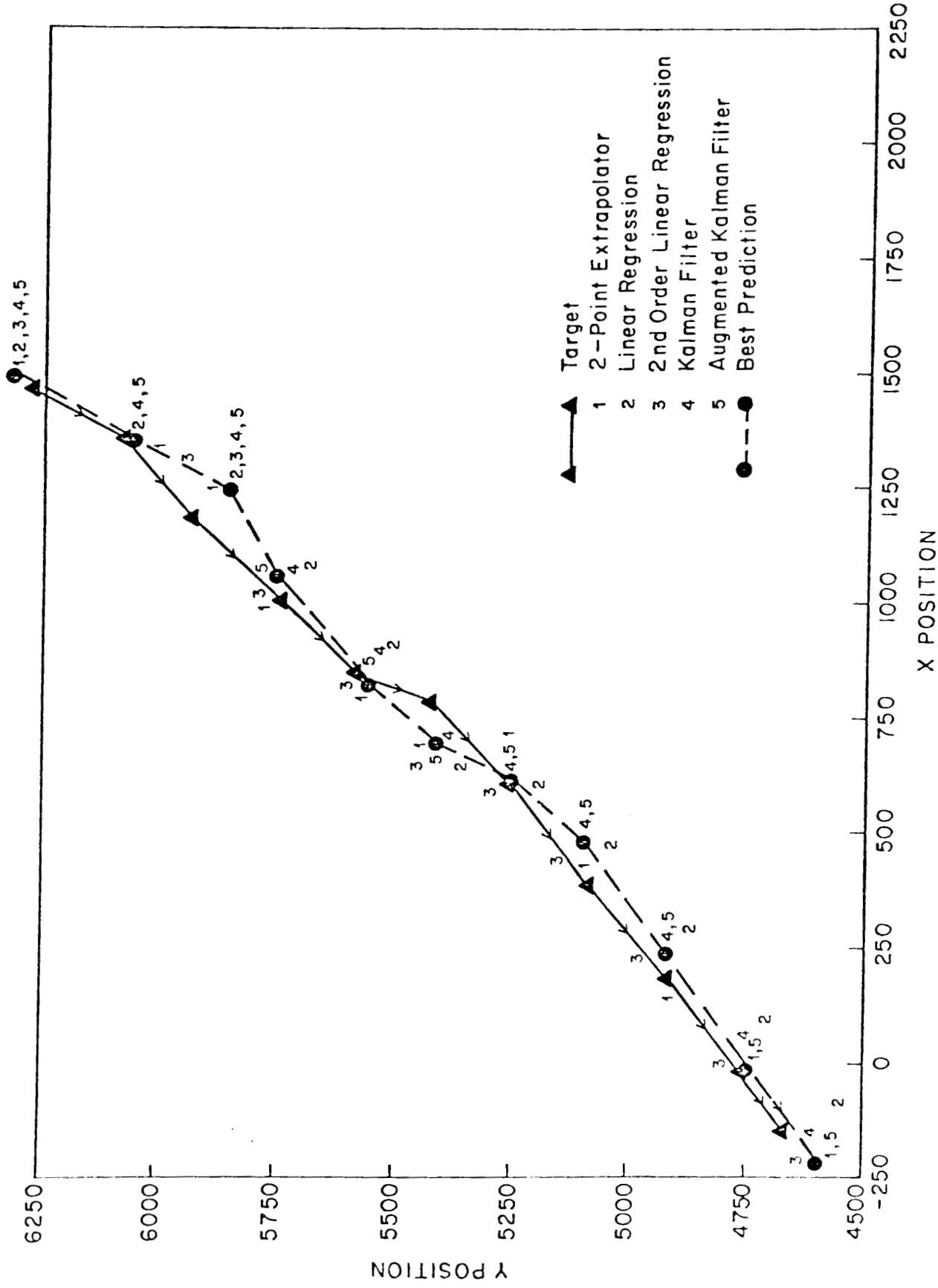

Figure 4-1: X vs Y Relative 1-Step Predictions, Target Moving

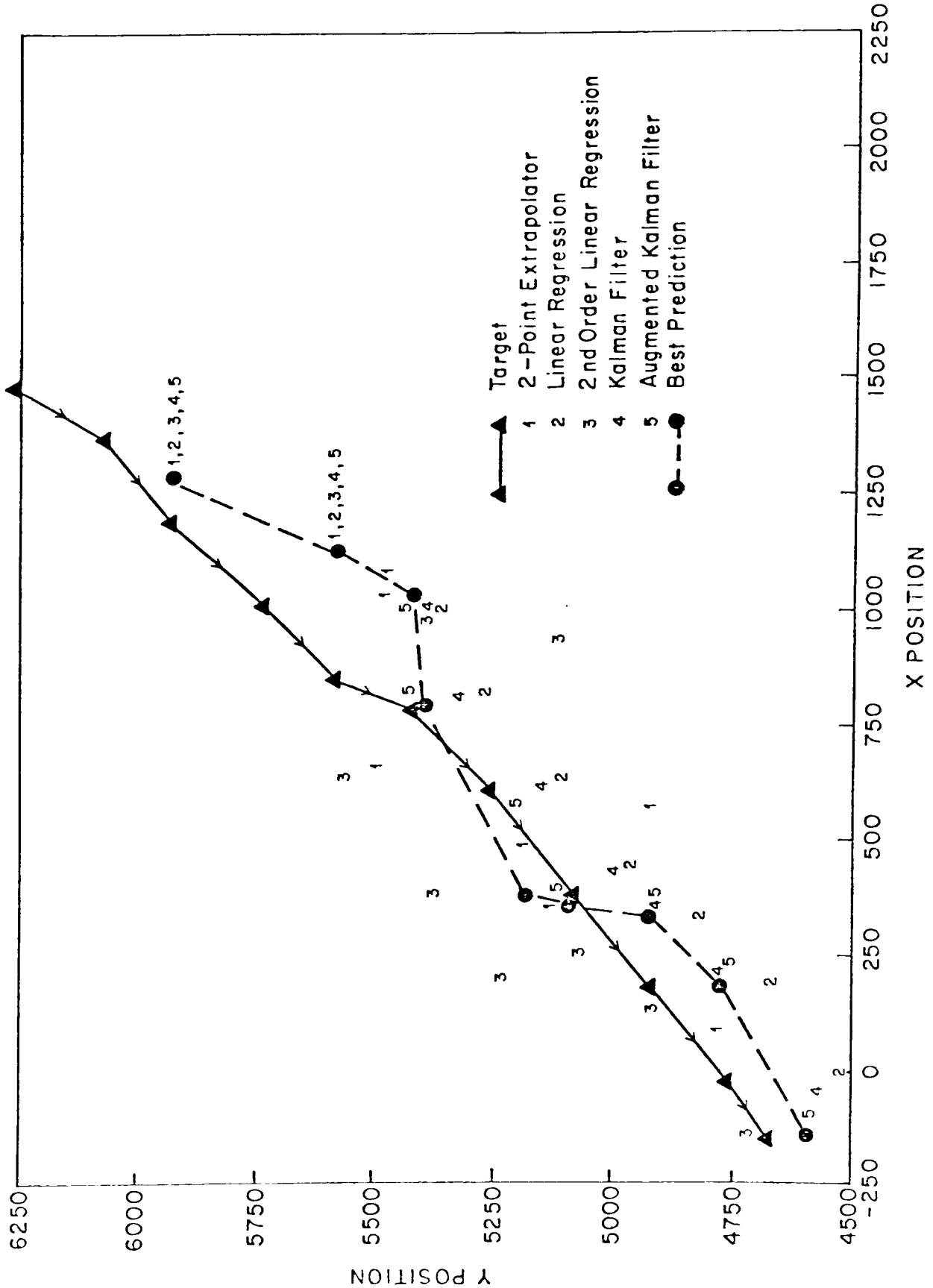

Figure 4-2: X vs Y Relative 3-Step Predictions, Target Moving

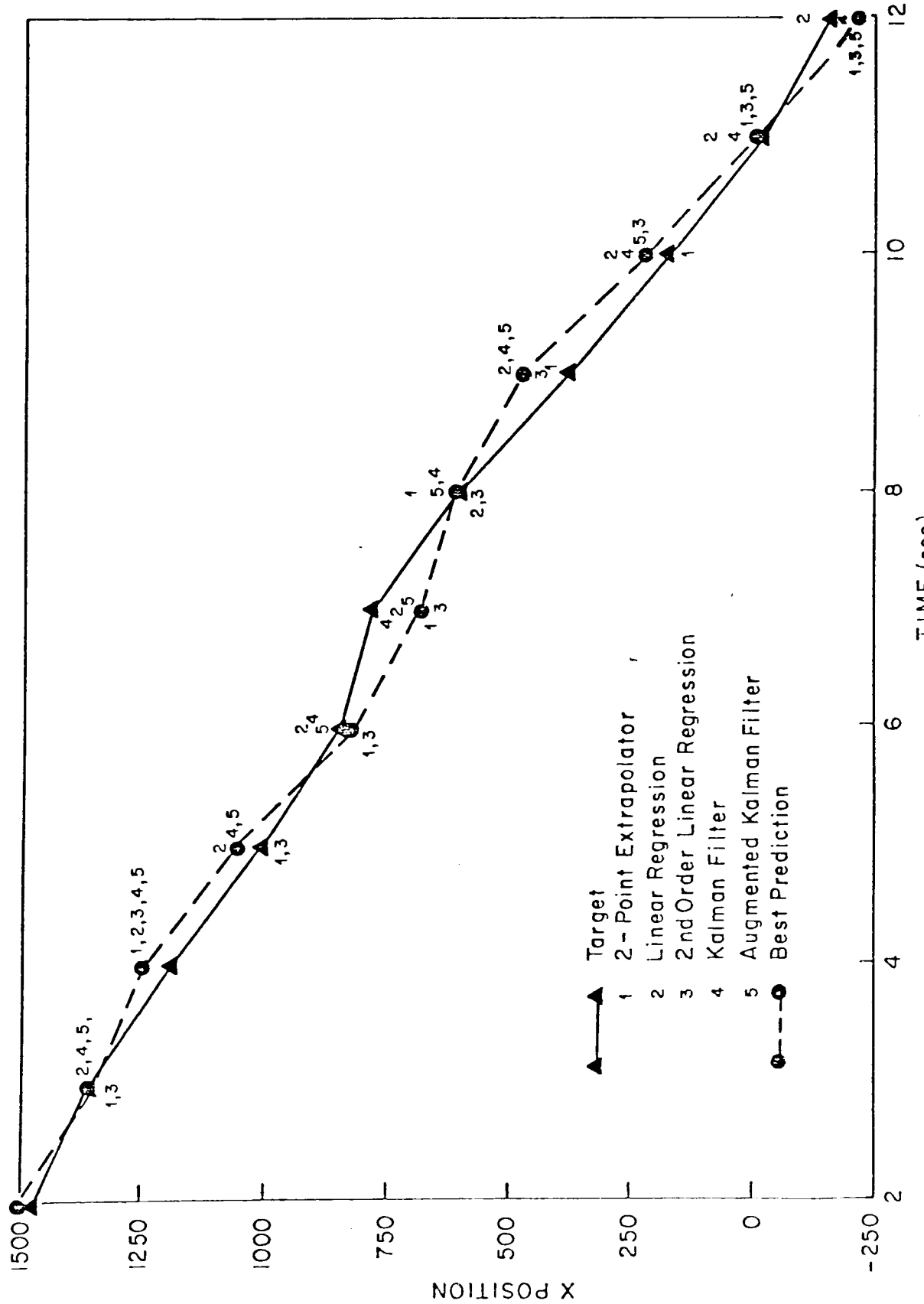

Figure 4-3: X vs T Relative 1-Step Predictions, Target Moving

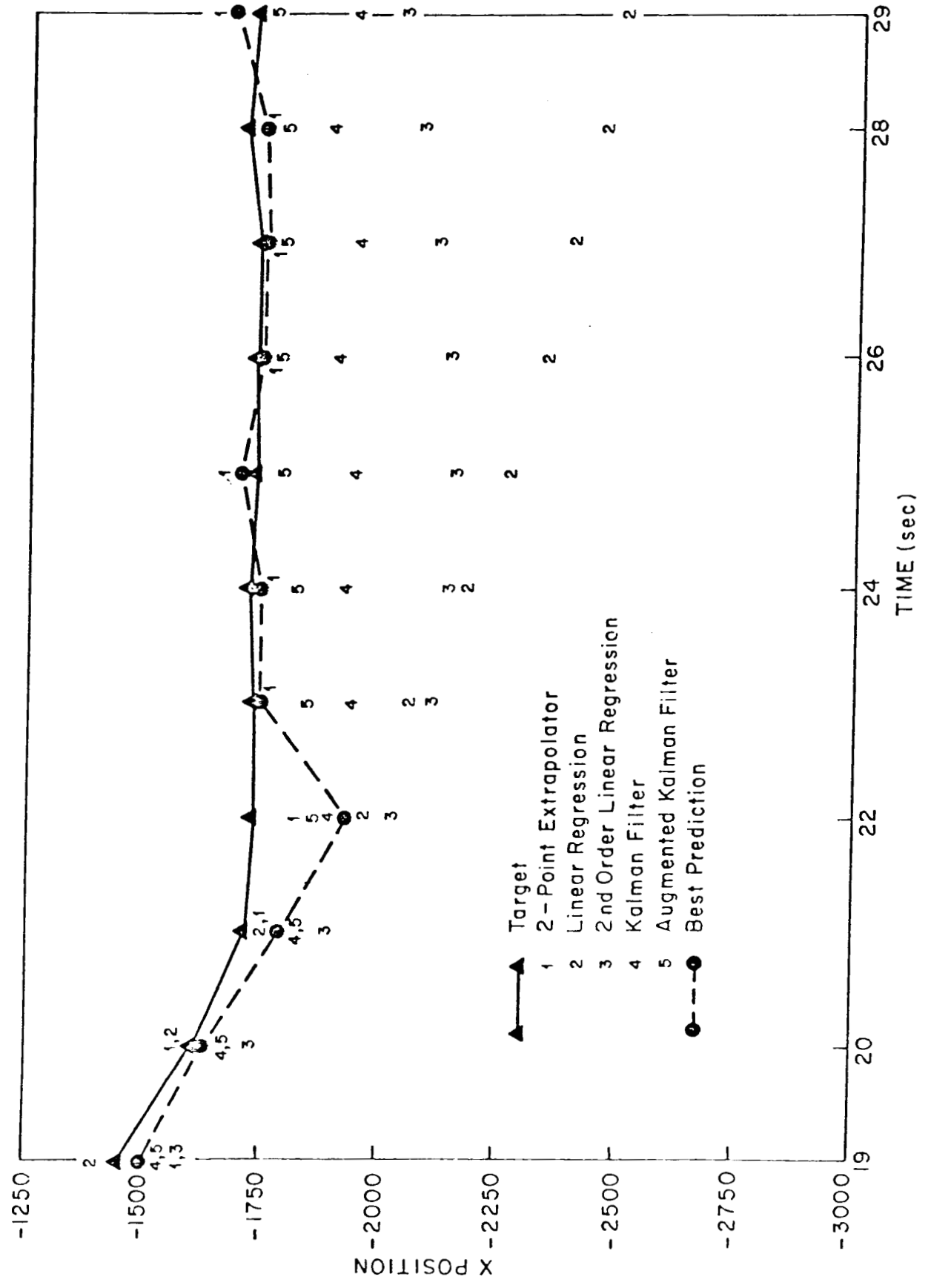

Figure 4-4: X vs T Relative 1-Step Predictions, Target Stopped

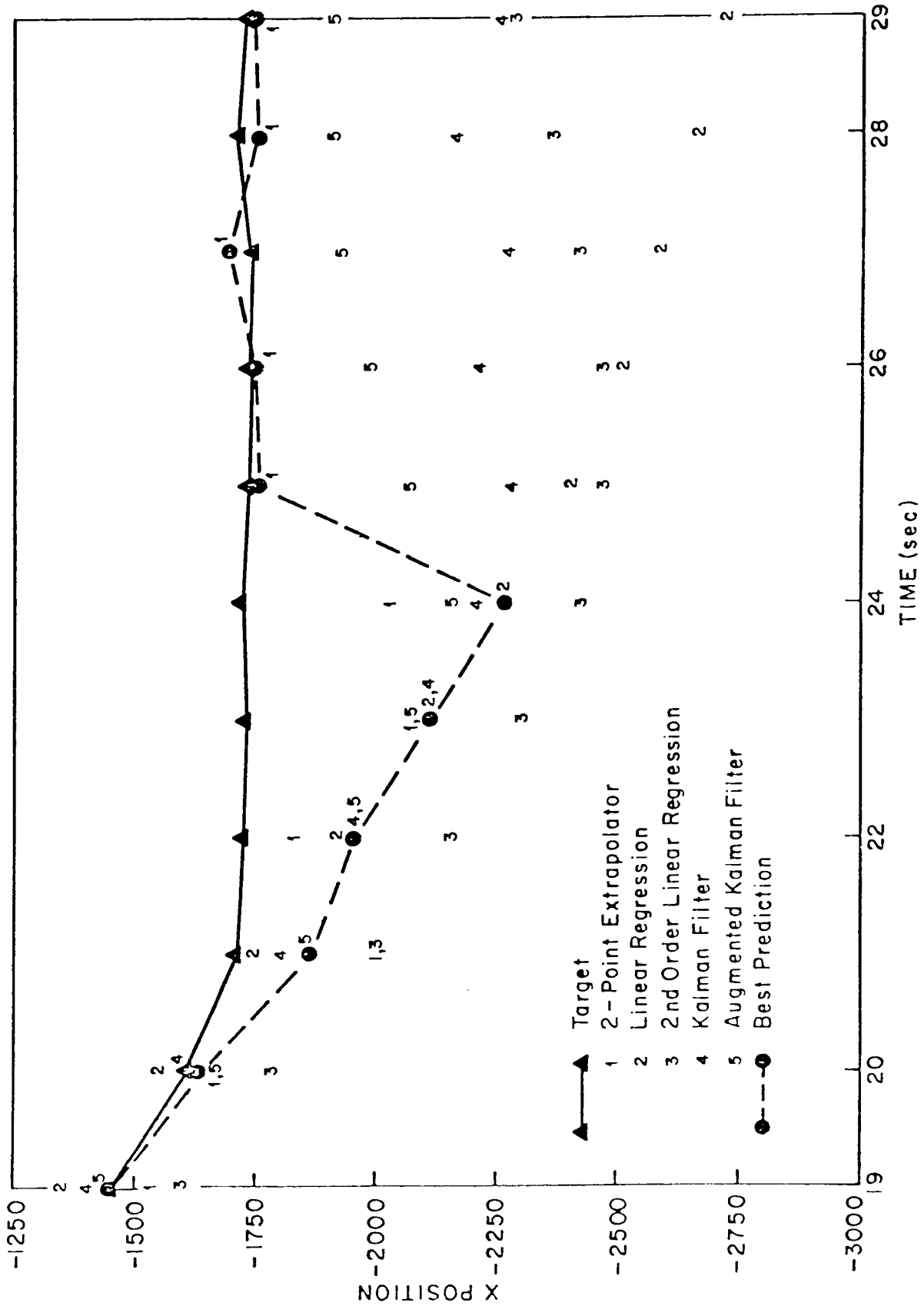

Figure 4-5: X vs T Relative 3-Step Predictions, Target Stopped

governing robot movement and best predictor selection determined by the previous two one-step predictions.

4.1.1.1. One-step Predictions

Figures 4-1 and 4-3 illustrate the one-step predictions of the moving target for every predictor. In general, each predictor is relatively error prone during the initial phase of tracking *(Figure 4-3, t = 4,5)*, but after several iterations the target establishes its linear trend and the predictions become more accurate *(t = 10-12)*.

Since the two-point extrapolator has a very short memory, it responds quickly and radically to deviations from the linear trajectory. It may even predict that the target is moving backwards *(t = 7,8)*. The linear regression predictor has a constantly increasing memory extending back to the first data sample, with each iteration being equally weighted. It is well behaved after several iterations, therefore, if the trajectory is close to linear. If its predictions begin to deviate from the true path because of sudden noise, however, it requires many subsequent iterations to correct itself *(t = 8-11)*. The second-order linear regression model has similar characteristics. When this predictor deviates from the true path it corrects itself more quickly than the first-order model, but subsequently overshoots *(t = 8-11)*. Both the Kalman and augmented Kalman filters allow for the effect of additive noise and have self-adjusting gains to vary the influence of past iterations. Providing the filter parameters were appropriately chosen (as they apparently were in this example), after the first few iterations these predictors are very well behaved. They are slightly disturbed by fluctuations in the signal *(t = 7,8)* but adapt quickly *(t = 9-12)*. The difference between the two Kalman filters is slight in this example, but the augmented filter appears to correct more quickly after being disturbed *(Figure 4-1, lower left)*.

The same basic characteristics of each filter are exhibited more dramatically when the target comes to rest *(Figure 4-4)*. For the same reason that the two-point extrapolator was a noisy predictor with the target moving, it is now the first to respond correctly when the target stops *(t = 21-23)*. Even with the target now still, this predictor is still perturbed by vision noise *(t = 24-29)*, and theoretically never would improve regardless of the number of subsequent iterations. The linear regression predictor, its memory consisting of 21 collinear points, stubbornly resists noise and adapts extremely slowly. As before, the second-order regression model responds more quickly, but once corrected, overshoots. The Kalman filter adapts fairly well, but fails to converge during the time displayed *(t = 26-29)*. The augmented Kalman filter adapts to the stationary target quickly and smoothly. It reflects the end of motion almost as quickly as the two-point extrapolator *(t = 22-24)* but is less affected by camera noise *(t = 26-29)*.

4.1.1.2. Relative N-Step Predictions

Each predictor's relative n-step prediction is simply an extrapolation of its one-step and therefore demonstrates the same characteristic behavior. The n-step predictors merit separate study because one of them, on the basis of past one-step performance, is selected to determine the robot's next position and thus affects overall tracking performance. There are two essential differences in the outcomes of the one-step versus the n-step predictions. Whereas in the one-step case the prediction is based on information current to the previous iteration, the n-step prediction is *n* iterations out-of-date at the time it is applied. The n-step predictors therefore establish noise resistant linear paths and accommodate the stopping of the target at correspondingly later times than do the one-step predictors. The second difference concerns the prediction error. Since the n-step predictions are *n* point extrapolations of the one-step predictions, any error is proportionately larger.

The effect of these differences is significant even in the 3-step prediction case *(Figures 4-2, 4-5 - 4-7, pages 32, 35, 41, and 42)*. Of the predictions for the moving target *(Figures 4-2 and 4-6)* the second-order regression is a particularly good example. Based on the noisy first three points of the trajectory, this predictor models a parabolic path which produces large errors beginning three iterations later *(t = 8-11)*. By the time this predictor begins to settle, the target has come to rest.

The predictions of the stationary target exhibit the same individual predictor characteristics and further illuminate the effects of the 3-step prediction *(Figures 4-5, 4-7)*. The two-point extrapolator and the two Kalman filters out-perform the linear regressors. The two-point extrapolator responds quickly but noisily to the target's stopping while the augmented Kalman filter responds more slowly but also more smoothly. Again there is a correspondingly greater lag in predicting the position of the stationary target. None of the predictors begins to respond to the cessation of motion until four iterations after the fact *(t = 25)*. The individual predictor errors are, as well, proportionately larger than in the one-step case.

4.1.1.3. Absolute N-step Predictions

In trials where the absolute n-step prediction option was selected, the predicted target position at $t = n$ is recomputed at each iteration. Since this is accomplished by using a series of relative n-step predictions where *n* is decremented by one each iteration, the absolute n-step predictions demonstrate the same characteristics as the relative n-step. In the absolute case, however, the effects are significantly larger during the initial iterations when the prediction steps are large, resulting in potentially enormous prediction errors. As the experiment nears completion, however, the prediction steps decrease with resultant smaller position errors.

Data from a second tracking experiment (target 1, trial 6) where the absolute 10-step prediction was selected are presented in Figures 4-11 and 4-12, pages 50 and 51. Figure 4-12 displays the predicted position values in a different form than was used for the relative n-step data. Since in the absolute n-step case all predictions are of the target's final position,

displaying the data in the same manner as in the relative n-step case would have been uninformative. Instead, in this example, all predictions are plotted at the time the prediction was made rather than at the time the position was predicted to occur and should be compared to the final target position for an interpretation of the prediction error.

Even without the benefit of chronological information a general impression of each predictor's performance can be drawn *(Figure 4-11)*. The two-point extrapolator is fairly accurate though widely scattered (noisy). The second-order regression performs poorly with only four predictions within the range of the graph. The remaining predictors, the linear regression model and the two Kalman filters, are relatively consistent and more accurate than the two-point extrapolator. The same results can be seen perhaps more clearly in one dimension displayed against time *(Figure 4-12)*. Here it is apparent that a slight parabolic trend in the initial three data points coupled with the large prediction step caused the second-order regression model to produce enormous prediction errors *(t = 2-4)*. It is also seen that in the y direction, the Kalman filters yield predictions very close to those of the linear regression model. This could indicate that the parameters chosen for these filters did not allow the gain factor to adjust appropriately. Because of the decreasing prediction step size, however, by the later iterations all predictors had stabilized, yielding excellent results.

4.1.2. Quantitative Results of One-step Predictors

Since they were required for the computation of the n-step predictions, the one-step prediction data are available for all trials and serve as a good basis of inter-trial comparison. The normalized mean squared errors for the five one-step predictions were computed for each experimental trial to provide a quantitative measure of predictor performance. This quantity was computed also for the set of predictions chosen by the best predictor algorithm. The two situations of predictions made while the target was moving and after it had stopped were considered separately. The mean squared prediction error in both cases was computed over the first ten iterations after motion had begun or had ceased, as appropriate for the case.

These predictor performance measures were normalized to compensate for target trajectory variations among the trials. The normalization factor was the mean of the smallest prediction error squared at each of the ten iterations. This quantity is a lower bound on the mean squared prediction error given the five available predictors and the ability to selectively implement them. It is therefore identical to the mean squared error of the selected best predictor if the best predictor algorithm were always successful in choosing the next predictor to incur the least error. In addition to compensating for trajectory variations, therefore, the normalization gives a measure for the performance of the best predictor algorithm.

The normalized mean squared prediction errors for each experiment conducted using target 2 are presented in tables 4-1 and 4-2. The starred entries under the five predictors indicate the predictor with the least squared error for that trial. The "chosen best" prediction, the outcome of the best predictor algorithm, is starred when it is less than any of the individual predictors.

TRIAL	2-POINT EXTRAP	1ST ORDER REGRESS	2ND ORDER REGRESS	KALMAN FILTER	AUG KAL FILTER	CHOSEN BEST
TARGET MOVING						
1	3.56	5.55	3.78	3.48	3.10 *	3.27
2	4.22	7.44	3.36	3.44	2.10 *	1.88 *
3	2.46	11.60	2.17 *	5.31	2.96	2.49
4	7.35	3.62	2.98 *	3.12	3.07	3.66
5	2.30 *	7.66	3.06	3.88	2.54	3.59
6	3.67	14.43	2.87 *	5.74	2.93	3.02
7	1.90 *	6.40	2.62	3.23	1.95	2.36
MEAN	**3.64**	**8.10**	**2.98**	**4.03**	**2.66 ***	**2.90**
SD	1.84	3.71	0.51	1.06	0.48	0.67
TARGET STOPPED						
1	1.01 *	292.42	105.33	31.04	5.40	3.71
2	2.30 *	254.80	116.11	20.69	4.72	5.79
5	1.0 *	381.33	165.90	31.71	5.65	4.30
6	2.23 *	168.00	40.48	20.78	4.95	6.13
MEAN	**1.64 ***	**274.14**	**106.96**	**26.06**	**5.18**	**4.98**
SD	0.73	88.44	51.57	6.15	0.42	1.16

Table 4-1: Mean Squared Prediction Error Normalized by True Best - Target 2, X Component

The information presented in the two tables substantiates the general impression of the individual predictors formed from the examination of the graphs. In predicting the path of the moving target, the two-point extrapolator is unremarkable. Although its *x* component in two trials has the smallest mean squared error among the predictors, considering the *x* and *y* components together, its error is on only four occasions less than that of the best predictor. Ranked solely by its mean of the seven trials, the two-point extrapolator is the third best in both directions. The linear regression model performs consistently poorly in both directions. Its squared error is only once less than that of the best predictor (and even then by less than one standard deviation) (*Table 4-1, trial 4*). The mean of the seven trials ranks it last in both coordinates. The second order regression model is apparently very sensitive to the degree to which it has correctly modeled the system. Presumably because there was a parabolic tendency in that direction, the *x* coordinate three times out of the seven performed best among the five predictors, and in four trials better than the chosen best predictor. The overall mean in the *x* direction ranked it second. The *y* component, however, performed poorly, ranking fourth. The *x* component of the Kalman filter was ranked an overall fourth. Its poor performance was presumably due either to an inadequate choice of filter parameters or to a nonlinear trajectory which resulted in the second order regression's success. In the *y* direction, however, the Kalman filter was ranked a close second. In four of the seven trials it performed among best the predictors, five times incurring a smaller error than the chosen best predictor. The augmented Kalman filter performed excellently, being ranked first in both coordinates. Even in the *x* direction where the Kalman filter had failed, the augmented Kalman

TRIAL	2-POINT EXTRAP	1ST ORDER REGRESS	2ND ORDER REGRESS	KALMAN FILTER	AUG KAL FILTER	CHOSEN BEST
TARGET MOVING						
1	1.63	4.75	3.67	1.86	1.40 *	1.37 *
2	9.54	8.35	12.08	8.30	6.06 *	5.90 *
3	6.19	9.76	5.48	3.18 *	3.88	4.72
4	3.15	2.81	3.02	2.09 *	2.10	2.39
5	3.65	3.53	4.57	1.51 *	2.09	2.94
6	7.27	29.79	9.92	5.51 *	6.59	12.57
7	2.95	4.33	2.74	1.85	1.81 *	2.42
MEAN	**4.91**	**9.05**	**5.93**	**3.47**	**3.42 ***	**4.62**
SD	2.83	9.50	3.64	2.53	2.14	3.83
TARGET STOPPED						
1	1.29 *	1581.73	16.57	1.34	1.48	1.71
2	1.88	2526.71	219.06	1.90	1.19 *	2.46
5	1.47 *	14663.60	257.58	1.60	2.25	1.36 *
6	1.16 *	459.87	11.02	1.17	1.71	1.73
MEAN	**1.45 ***	**4807.98**	**126.06**	**1.50**	**1.66**	**1.82**
SD	0.31	6624.50	130.60	0.32	0.45	0.46

Table 4-2: Mean Squared Prediction Error Normalized by True Best - Target 2, Y Component

filter was twice the best among the predictors and in five trials performed better than the chosen best. In the *y* direction it was even more successful, having the smallest error in three of the seven trials.

When the target stopped the predictors' relative performances changed dramatically. The two-point extrapolator was superior to all other predictors in both directions. Considering the *x* and *y* data as one group, in seven of the eight trials it had the least error. It was the only predictor to perform better than the chosen best predictor. In predicting the location of the stationary target both linear regressors performed extremely poorly. Both of the Kalman filters performed well, particularly in the *y* coordinate where the means of their errors were only slightly greater than that of the two-point extrapolator but less than the chosen best predictor's.

4.2. Results of the Best Predictor Algorithm

Of the five predictions computed at each iteration, one must be selected to define the robot's next tracking command. The selection method employed was the 'best predictor' algorithm defined in Section 3.4.2. Because this algorithm is unable to forecast change in predictor performance, it selects the predictor with least error in previous iterations to give the best prediction in the future.

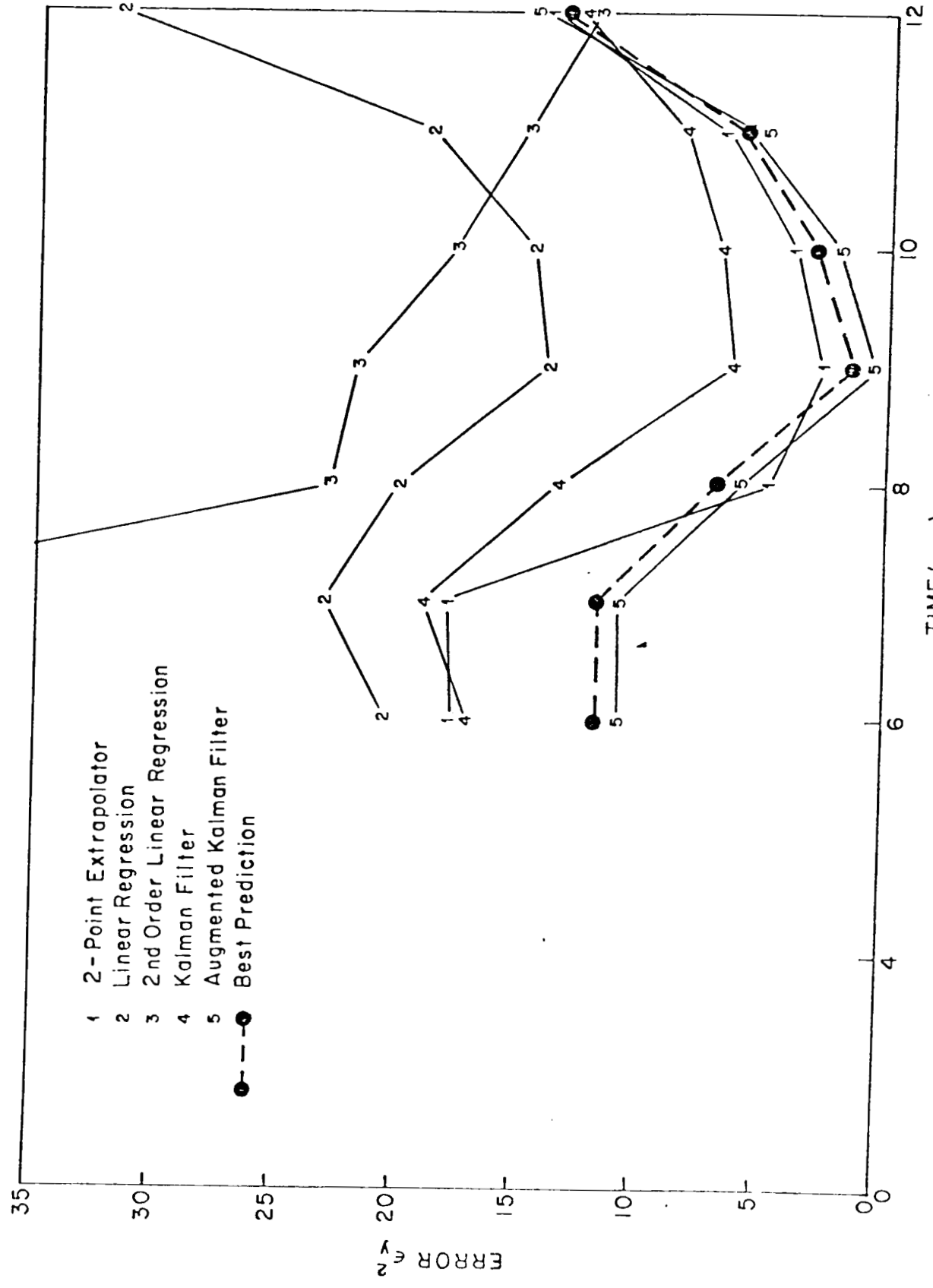

Figure 4-6: Smoothed ϵ_y^2 vs T - Relative 3-Step Prediction, Target Moving

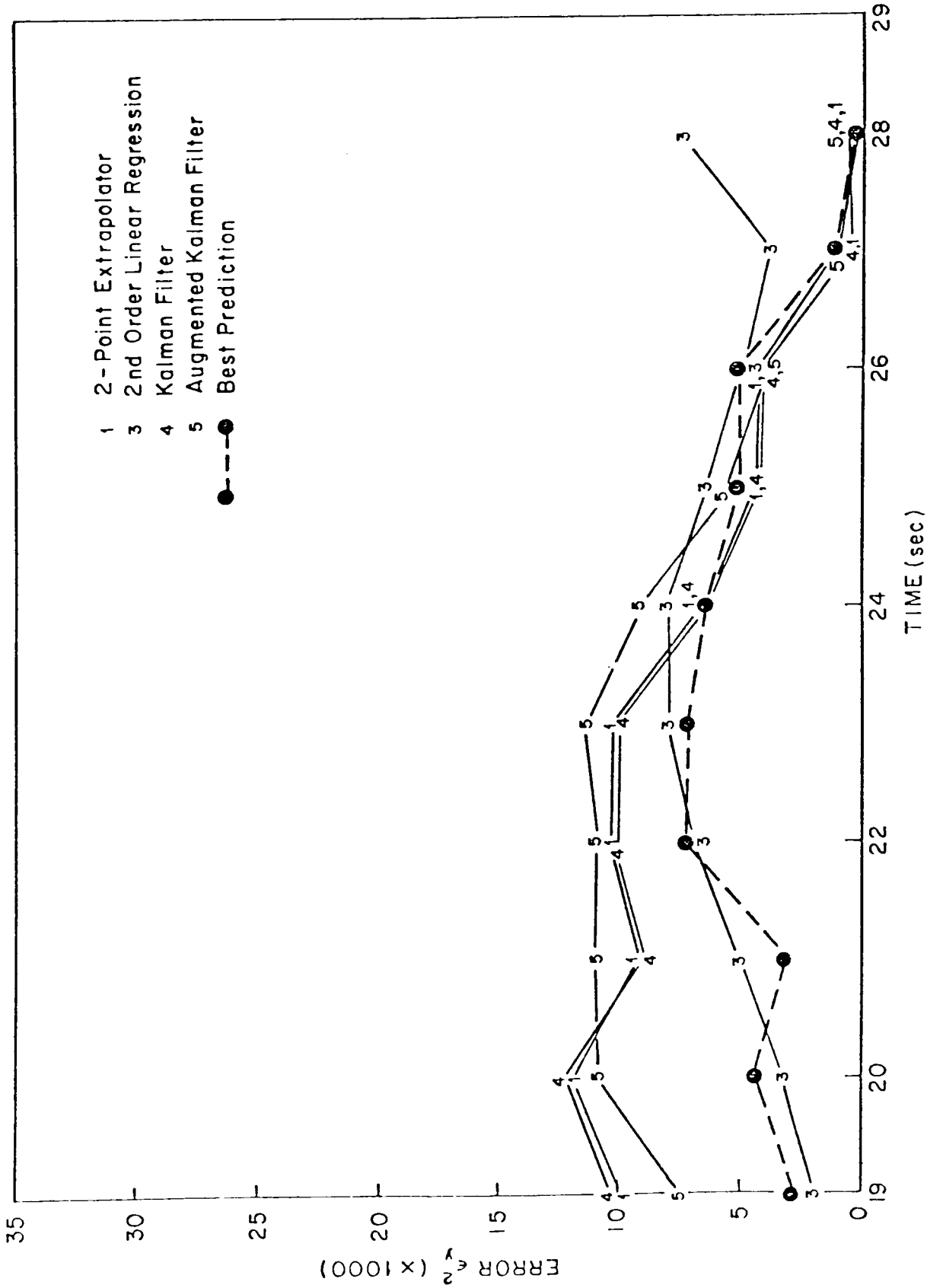

Figure 4-7: Smoothed ϵ_y^2 vs T - Relative 3-Step Prediction, Target Stopped

The best predictor algorithm in general succeeds in selecting if not always the best, one of the best predictions at each iteration. In Figure 4-2 for example, visual inspection shows the trajectory of the coordinates chosen as best predictions to be an improvement over the general trend of the individual predictors. As hypothesized, it is particularly effective in rejecting the results of consistently poor predictors. Again with reference to Figure 4-2, the second order linear regression predictor yields significantly larger errors than the other predictors, and is rejected by the algorithm for this reason. The algorithm's strength, however, is also its weakness. By the time corresponding to the last trajectory point plotted on this graph the second order regression has corrected itself and is now the closest prediction. Its y component is rejected as best prediction, however, on the basis of its past poor performance.

The best predictor algorithm is at its most valuable in accommodating the stopping of the target. As the previous study of the individual predictors' performance indicated, it is typically the case that the predictor which incurs the least error in predicting the moving target's trajectory is rarely the one which best predicts the position of the stationary target. Use of the best predictor algorithm provides an effective latching mechanism. An example of the squared prediction error smoothed by a five point moving window is presented for all predictors in Figures 4-6 and 4-7, pages 41 and 42 (target 2, trial 1, y coordinate). When the target is moving the best predictor measurably outperforms four of the five individual predictors *(Figure 4-6)*. Its performance would have improved marginally only if it had consistently selected the predictions of the augmented Kalman filter. Immediately after the target comes to rest, the best predictor error is less than that of the augmented Kalman filter as well as three other predictors, and is comparable only to the error in the second order regression predictions. Had either the augmented Kalman or the second order regressor been selected throughout the entire experiment instead of the best predictor's selective use of each, tracking performance would have decreased.

The information presented in Tables 4-1 and 4-2 allows a more quantitative examination of the best predictor's performance. The normalization procedure discussed in Section 4.1.2 provides an absolute measure against which to compare the best predictor's results. Considering data from both components and the moving and stationary target separately, the mean squared prediction error of the best predictor is of the same order of magnitude as the "true best" in all trials except one. The four means computed are all less than five times the true best.

The best predictor's performance is best evaluated, however, by comparison with the results of the separate predictors. Again considering the two coordinates independently, the best predictor's error is the least among all predictors in only one trial of fourteen with the target moving and one of eight when stationary. None of the best predictor's means in the four categories is least. Although rarely yielding the least error, the best predictor still gives good results. Of the fourteen moving target trials where the best predictor was only once best, it is in thirteen trials within one standard deviation of the prediction of least error. Three of the best predictor's four means are within one standard deviation of the mean with the least error.

The alternative to implementing an algorithm such as the best predictor algorithm which selectively incorporates results of the five predictors is to choose one predictor to suit all possible situations. This exclusive predictor must yield good results in both components, whether the target is moving or has come to rest. The results of the augmented Kalman filter indicate it may have been a successful candidate for the majority of cases, but before the fact there was no way of knowing this. Had the Kalman filter instead been implemented, the subsequent robotic tracking for both the moving and stationary target would have been excellent in the *y* direction but extremely poor in the *x* direction. Obviously predictions for both directions must be adequate for successful tracking. Assume instead that the second order regression model were chosen on the basis of its excellent *x* component performance with the target moving. The tracking performance when the target stopped would then have been two orders of magnitude worse than that based on the best predictor.

Again it is apparent that the conservative design of the best predictor algorithm involved compromise. The possibility of slightly decreasing robotic tracking error by using the predictions of a single well-behaved predictor was traded for the security of ensuring wild point elimination. The merit of the algorithm was in its generality. Even without prior knowledge of the type of trajectory and which predictor would be best suited for it, the best predictor ensures acceptable if not always the best possible results.

4.3. Tracking Results

The computation of the five separate predictions and the selection of one as best predictor determined the tracking commands used to direct the robot's motion. On the assumption that the coordinate transformation matrix and the internal Puma position control were accurate and consistent, there were two remaining influences on the robot's ultimate trajectory. The first was the obvious effect of the end-effector speed. This was set to a standard value of 85 cm/sec for all experiments. The second factor was the communication time required to send the desired position instructions over the serial line to the robot controller. The time to complete one cycle of the control loop, from the initial *Take Picture* command to the robot's acknowledgment of having reached a new position, was greater than the picture sampling rate. The processes of picture taking and computing predictions were protected against interrupts, and thus finished before the next clock pulse. The remaining time was devoted to the transformation and passing of new position information to the robot. Requiring more time than was available, these processes were typically interrupted and completed during the next iteration. As a consequence, only selected best predictions were actualized by the robot. The selectivity factor was simply whether the robot had completed its previous positioning instructions and was ready to accept new ones.

4.3.1. General Characterization

Two sets of robotic tracking data are presented in Figures 4-8 through 4-12. Figures 4-8 through 4-10 display data from the trial whose relative 3-step predictions were studied in section 4.1.1.2 *(target 2, trial 1)*. The example of absolute n-step tracking data presented in Figures 4-11 through 4-12 corresponds to the prediction data discussed in section 4.1.1.3 *(target 1, trial 6)*.

4.3.1.1. Relative N-Step

The robot's two-dimensional motion in the relative 3-step prediction case is displayed without regard to time in Figure 4-8 on page 46. Its trajectory very closely approximates that of the target. The robot's path was in fact an improvement over the one defined by the best predictor. This was for no reason other than good luck in the coincidence of the better estimates constituting the best predictor path with the robot's ability to accept a new desired position instruction. It was just as likely that the predictions with greater error be actualized. As a general rule the robot interpreted every other prediction. The exception was the second robot position, having been determined by the third rather than second prediction after the first position. This was simply because the distance between the initial robot position and the first desired position was considerably greater than that between adjacent selected positions with a resulting increase in time required for the actual robot motion. This is the only instance when the robot's transit time was of any significance.

An examination of one dimension graphed against time provides further insight into the robot's behavior *(Figure 4-9)*. The robot spends most of its time motionless, waiting for prediction computations and serial line communication before it receives its next desired position command. The actual time spent in transit is relatively insignificant by comparison. In order to maximize the chances of intercepting the target at a random instant, the robot's ideal zig-zag y vs t path should consist of small segments and be centered around the target's trajectory. The length of the segments (that is, the time the robot spends motionless between position updates and the time in transit) is determined given the sampling rate of the complete control loop and the robot speed. The extent to which the zig-zag pattern is centered about the target trajectory is determined by the value of the prediction step. In this example the choice of a 3-step prediction was a good one, resulting in the robot spending equal amounts of time behind and ahead of the target.

As previously discussed, when the target comes to rest, all predictors typically overshoot to varying degrees. The best predictions made of the stationary target are especially well behaved in this example *(Figure 4-10)*. The selection of best predictions interpreted by the robot was again a fortunate one, since the prediction with the greatest error was omitted *(t = 24)*. Because of the design of the control system, the time between robotic position updates remains unchanged despite the small distance traveled.

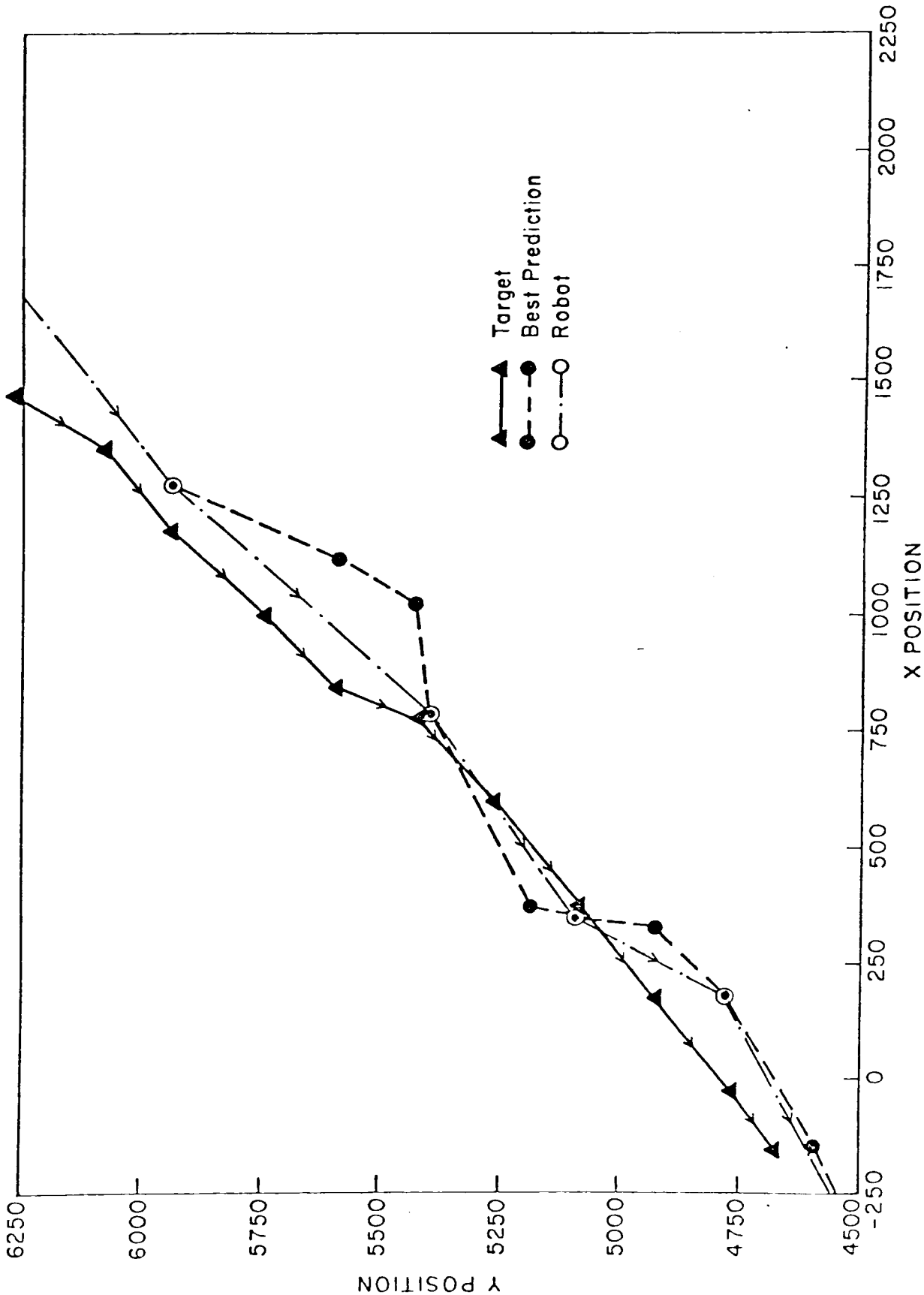

Figure 4-[illegible] Robot Trajectory - Relative 3-Step Prediction, Target Moving

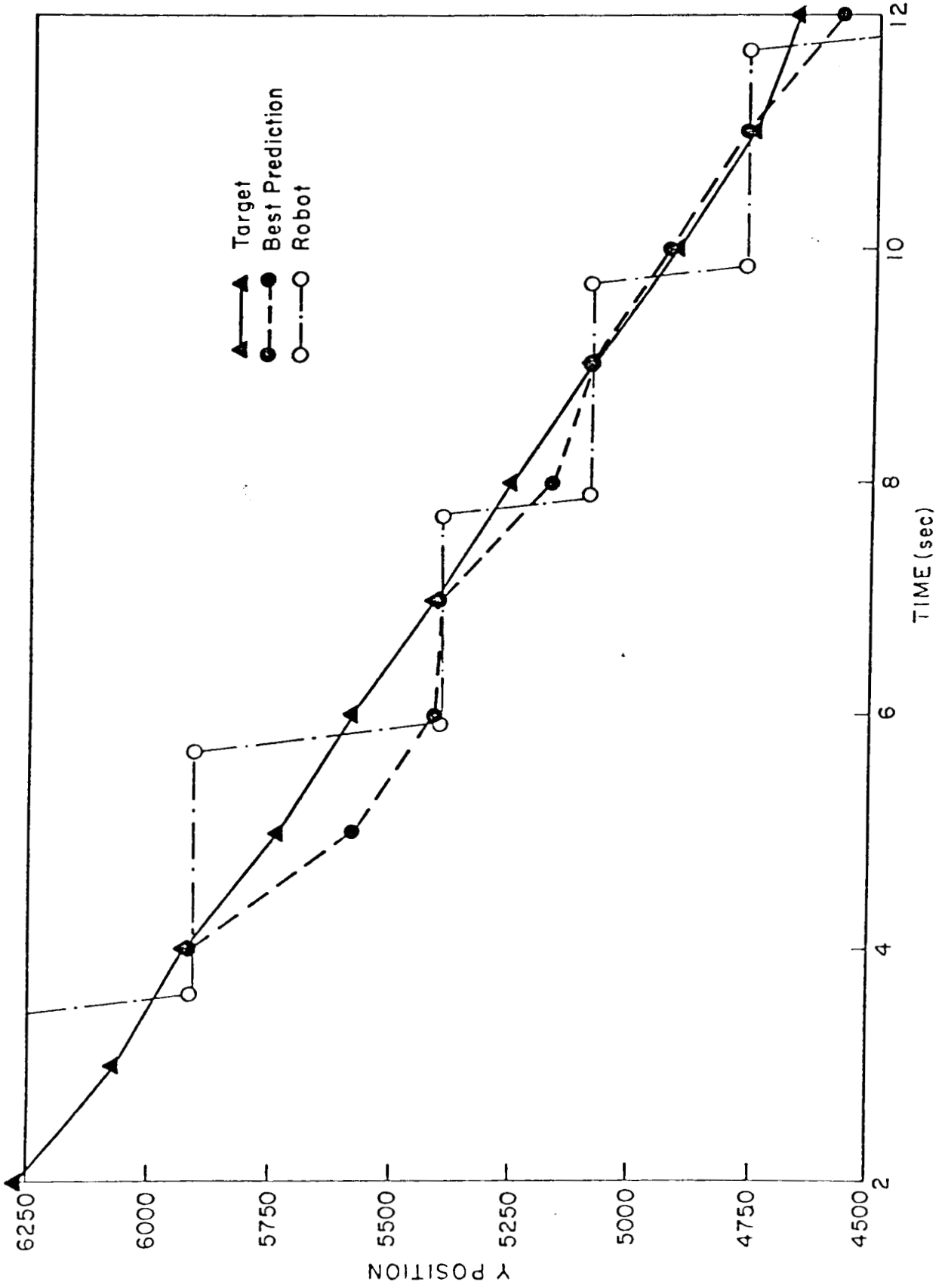

Figure 4-9: Y vs T Robot Trajectory - Relative 3-Step Prediction, Target Moving

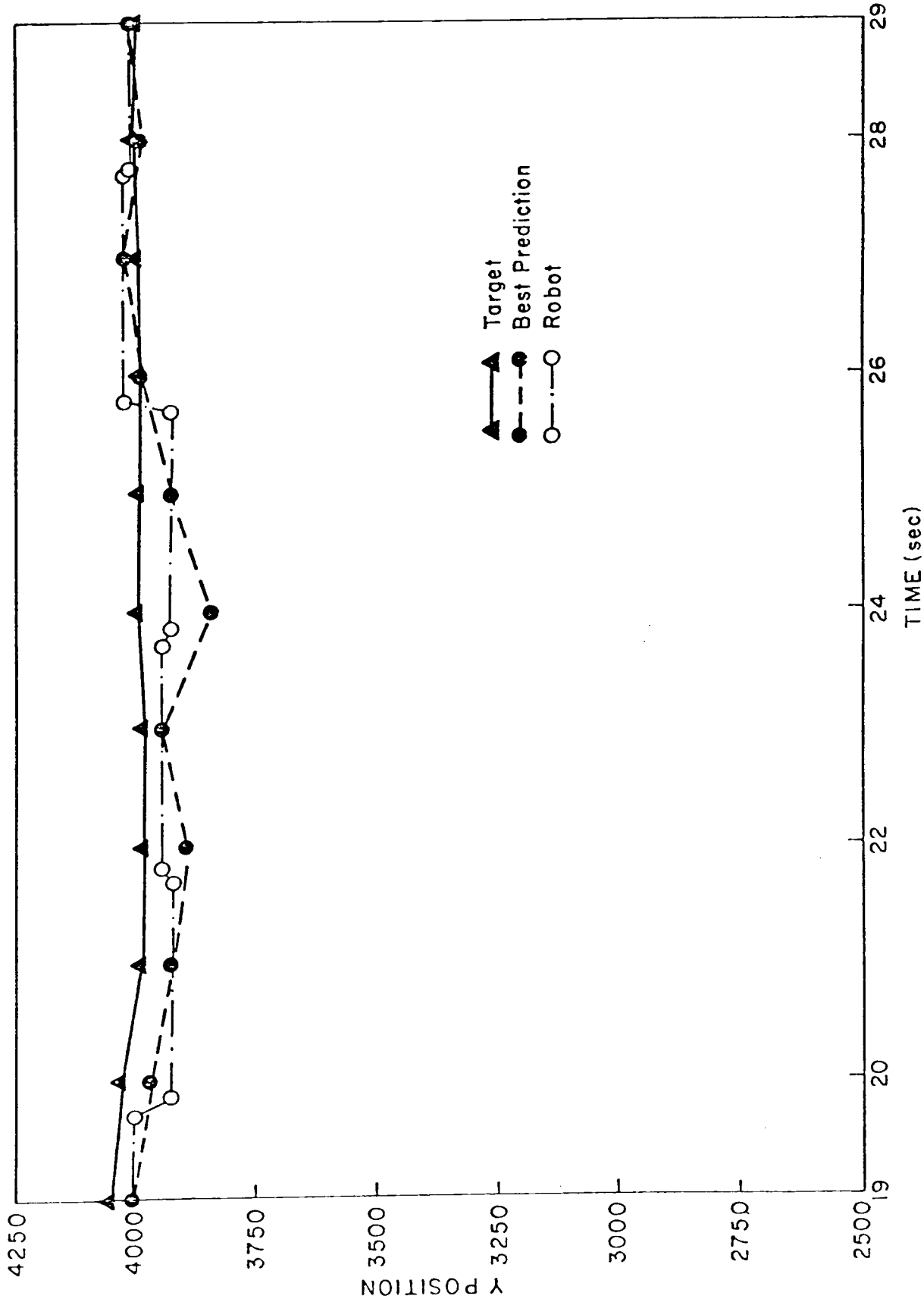

Figure 4-10: Y vs T Robot Trajectory - Relative 3-Step Prediction, Target Stopped

4.3.1.2. Absolute N-Step

An example of the absolute 10-step prediction tracking behavior is illustrated in Figures 4-11 and 4-12. Figure 4-11 presents an enlarged scale view of the robot's two dimensional path. Although the robot had only actualized the position predicted at $t = 7$ by the time it was instructed to intercept the target, it was nevertheless close enough to the final target position to result in a successful interception.

The robot's y trajectory is plotted against time in Figure 4-12. At $t = 3$ the robot reached the predicted final target position made at $t = 1$. It was then instructed to move to the position predicted at $t = 3$. Because of the tendency of the absolute n-step prediction method to be error prone during the initial iterations, the prediction made at $t = 3$ had a very large error, and in fact lies off the graph. When the Puma controller received this prediction as its new desired position, it recognized it to be out of the robot's range and refused the command to move there. By the time the controller was ready to accept another position command, the best predictor path had stabilized and was yielding acceptable predictions. The next two desired positions sent to the robot were successful in fine-tuning the robot's position, resulting in a successful target interception.

The rationale for implementing the absolute n-step prediction had been that by prepositioning the robot at the outset of tracking and then using subsequent more accurate predictions to reduce the original positioning error, the robot's transit time could be reduced. As has been shown, the time required for the actual movement of the robot was insignificant in comparison with the communication time. Not only was the assumed advantage of this method meritless, but also its implementation produced large prediction errors and potentially dangerous robot movements.

4.3.2. Quantitative Tracking Results

The results of one possible measure of robotic tracking performance are summarized for target 2 in Tables 4-3 through 4-6. Data from four experiments in which tracking was based on relative n-step predictions are presented in Tables 4-3 and 4-4, with the moving and stationary target treated as separate cases. The first three trials were based on relative 3-step predictions, while trial 6 used relative 5-step predictions. The quantities tabulated represent the percent of the first 10 seconds after motion begins or ends, as appropriate, that the error between the robot's position and the target is within the given bounds. Since in general every other iteration's prediction was actualized by the robot, in 10 seconds the robot occupied approximately five different positions. Table 4-3 presents the y component alone. Since tracking performance is a function of the simultaneous error in both directions, these data are provided by Table 4-4. The simultaneous error is defined as the maximum of the corresponding x and y component errors.

Data from the trials in which the absolute n-step tracking option was employed are presented in Tables 4-5 and 4-6. The same representation is used with the exception of the

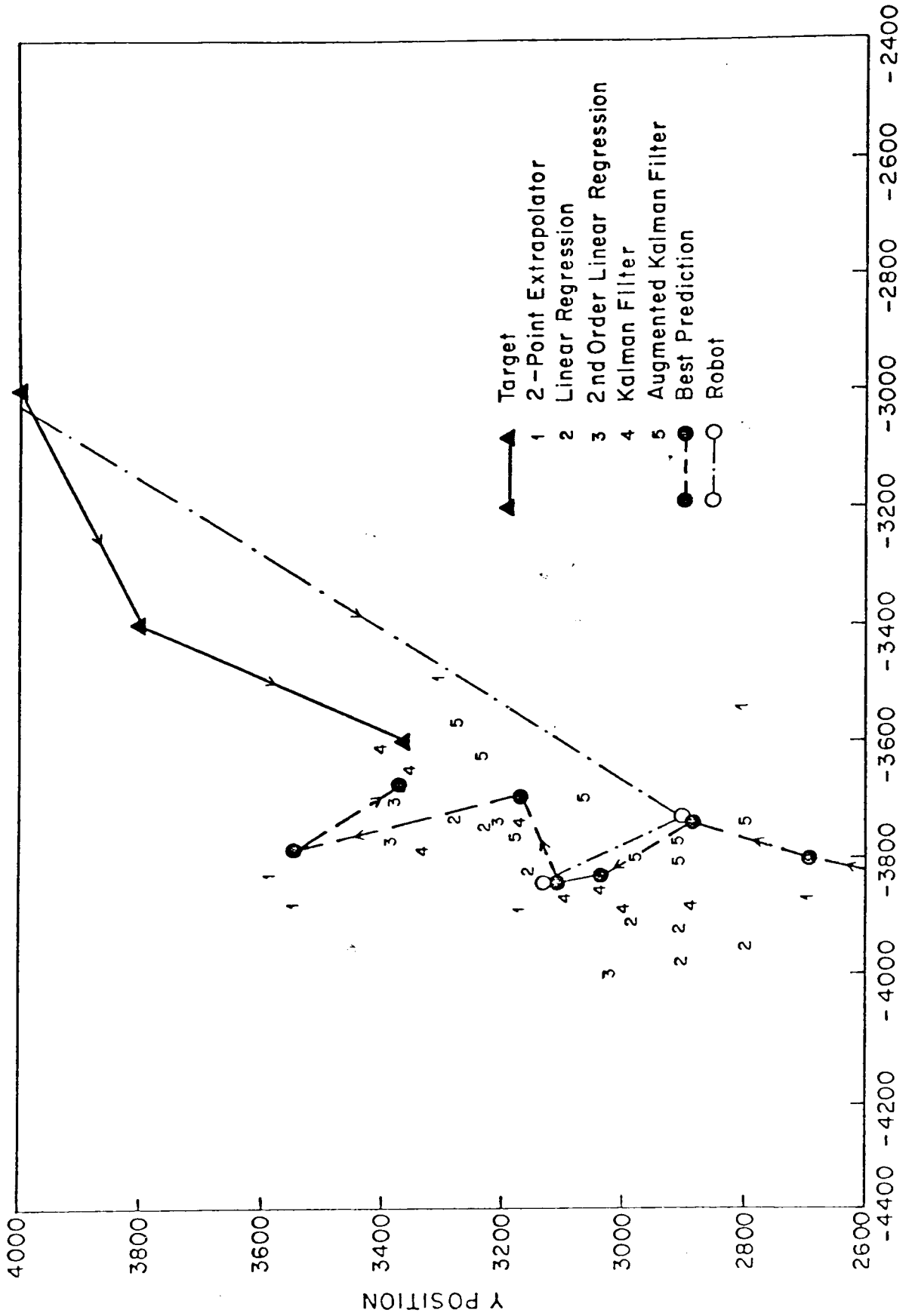

Figure 4-11: X vs Y Robot Trajectory and Absolute 10-Step Predictions

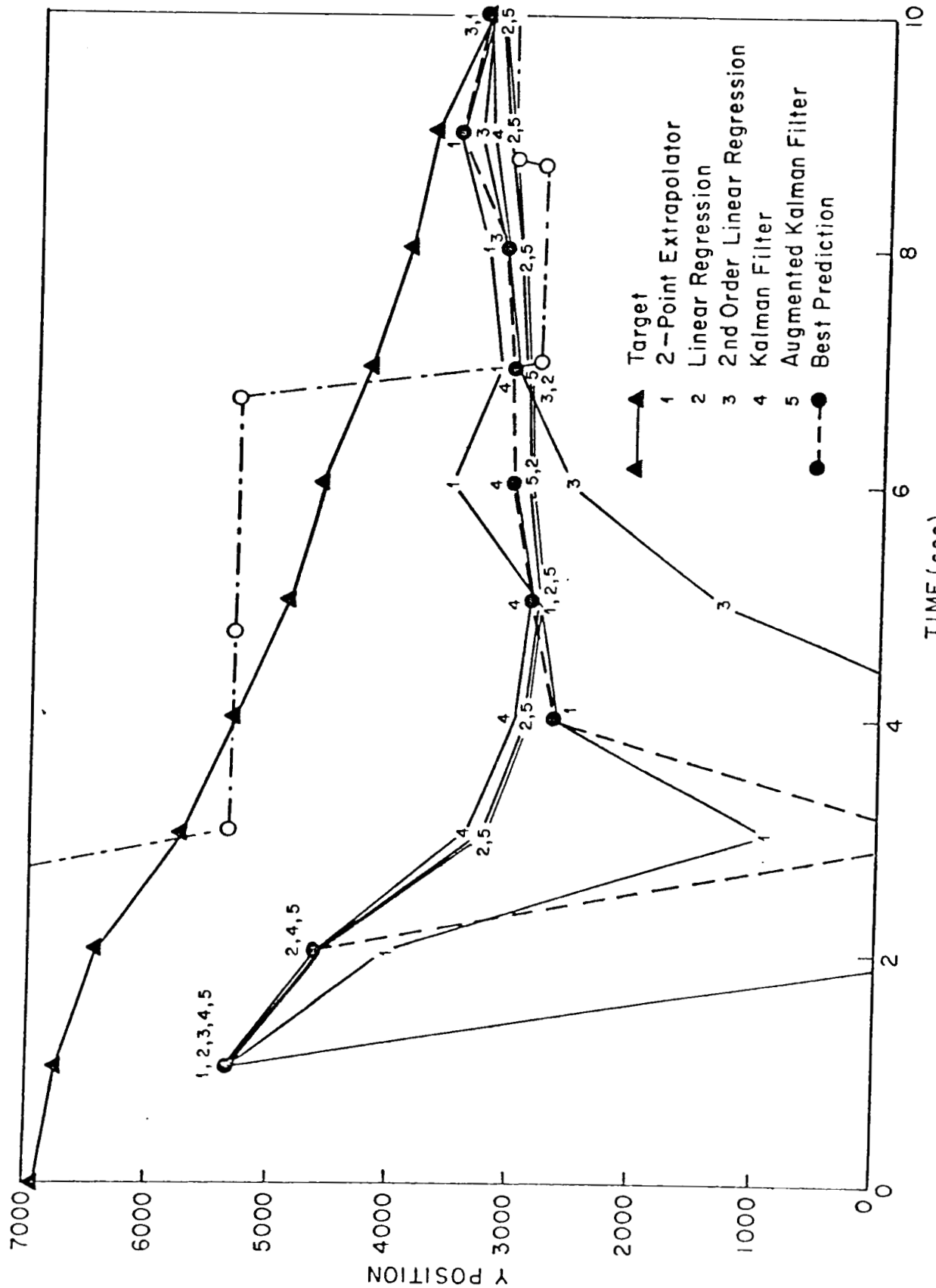

Figure 4-12: Y vs T Robot Trajectory and Absolute 10-Step Predictions

TRIAL	<100	<250	<500	<750	<1000
TARGET MOVING					
1	61	97	100	100	100
2	28	68	100	100	100
5	26	79	97	100	100
6	9	43	65	86	100
TARGET STOPPED					
1	100	100	100	100	100
2	75	100	100	100	100
5	100	100	100	100	100
6	54	78	100	100	100

Table 4-3: Time Percentage of Robot Position Errors, Relative N-step Predictions - Target 2, Y Component

TRIAL	<100	<250	<500	<750	<1000
TARGET MOVING					
1	32	83	100	100	100
2	18	61	100	100	100
5	10	48	97	100	100
6	1	18	39	86	100
TARGET STOPPED					
1	66	75	100	100	100
2	53	54	75	100	100
5	57	59	100	100	100
6	51	52	73	76	100

Table 4-4: Time Percentage of Robot Position Errors, Relative N-Step Predictions - Target 2, Both Components

performance during the first 10 seconds of motion being compared to the 10 seconds prior to interception. Trials 3 and 7 were based on an absolute 20-step prediction. Trial 4 employed an absolute 10-step prediction and is therefore categorized in the tables only by the 10 seconds prior to interception.

In all tables the bounds are given in camera coordinates. For a clearer understanding of their physical significance, recall that approximately 300 units in the camera coordinate system correspond to 1 cm and that a successful interception requires a position error of no more than 2 cm. When both component errors are less than 250 camera units (column 2) the greatest two dimensional error is 353, ensuring interception. If both directional errors are less than 500 (column 3) the largest possible error is 707, implying interception is likely but not certain. When one component error is between 500 and 750 and the other error is known only to be less than 750 (column 4), interception is possible but very unlikely. An error greated than 750 precludes successful interception.

TRIAL	<100	<250	<500	<750	<1000	<2000
FIRST 10 SECONDS						
3	0	0	3	4	4	40
7	0	0	0	0	13	39
LAST 10 SECONDS						
3	0	28	29	74	100	100
4	0	66	100	100	100	100
7	74	77	90	91	100	100

Table 4-5; Time Percentage of Robot Position Errors, Absolute N-Step Predictions - Target 2, Y Component

TRIAL	<100	<250	<500	<750	<1000	<2000
FIRST 10 SECONDS						
3	0	0	0	0	4	31
7	0	0	0	0	13	39
LAST 10 SECONDS						
3	0	28	29	53	100	100
4	0	33	60	88	100	100
7	0	26	72	91	100	100

Table 4-6: Time Percentage of Robot Position Errors, Absolute N-Step Predictions - Target 2, Both Components

It is apparent from the tabulated results that tracking performance varies between corresponding components. The overall tracking performance represents a degradation of the *y* component alone. For the three trials in which relative 3-step predictions were used, overall tracking performance is nonetheless excellent. For more than half the duration of the first 10 seconds with the target either moving or stopped their composite tracking errors are less than 250, and nearly always are less than 500. These clearly are encouraging results, indicating almost certain interception at any time. The 5-step prediction is less well suited to the timing requirements of the control system. Its tracking error of the moving target is within acceptable bounds significantly less often than that of the 3-step prediction. The excellent performance resulting from both step values of the relative prediction option as the target comes to rest can be attributed at least in part to the flexibility offered by the best predictor algorithm.

Study of the predictor results for the absolute n-step option indicated large errors are likely during the initial phase of tracking. This result is confirmed by the tracking results. By the final 10 seconds of tracking the performance has improved, but is still not comparable to the tracking results of the trials where the relative n-step predictions were computed. The performance over the 10 seconds prior to interception was immaterial in the real time situation, where interception success or failure depended only on the error in the last robot

position. The tabulated data are valuable in this form, however, as another indication that the original assumption governing implementation of the absolute n-step prediction method was unfounded.

Chapter 5
Conclusions

Results of the predictive tracking experiments indicate attainment of both research objectives. The target was successfully tracked and intercepted in each experimental trial. Pursuit of the second goal, the evaluation and comparison of predictor performance, provided data which enabled the best predictor algorithm to selectively implement the results of the five predictors.

5.1. Contribution

Direct applications of this implementation are limited because of the system constraints discussed earlier. Perhaps this research's most important contribution is the demonstration of one possible method of predictive robotic tracking with results indicating that it can be effective. This work also illustrates several existing limitations which must be overcome before a more practical implementation is possible.

Results of the predictor comparison may serve as a guide for predictor selection in future implementations. Use of the best predictor algorithm eliminated the need for the selection of one general purpose predictor to be used at each iteration and in every trial. If such a selection were to be made on the basis of the specifications and results presented here it would probably favor implementation of the augmented Kalman filter, provided proper selection of filter parameters could be assured. Before this predictor can be recommended, however, there is an additional factor to consider. To an extent to be defined by the application's specifications, predictor accuracy must be weighed against algorithm execution time [Singer 71]. Although this was not an issue in this implementation because of other more severe timing considerations and the aim of investigating the performance of several predictors simultaneously, in another situation algorithm execution time would likely be of prime importance. The marginal performance improvement offered by the augmented Kalman filter over the simple two-point extrapolator must be considered in conjunction with the order of magnitude difference in execution time.

5.2. Limitations and Suggested Improvements

5.2.1. Software

There are several possible software modifications to this implementation which may have improved tracking performance, but which were not implemented since the original objectives were satisfied. Successful prediction of the stationary target became an important measure of predictor performance during data collection, and yet all predictors had been designed to operate on a moving target. A predictor which assumed no motion may have allowed the best predictor algorithm to more quickly recognize the end of target motion.

The best predictor algorithm, designed to be a simple and fast method of selecting one prediction from the five possible candidates, produced results which were acceptable for this implementation. There were instances when tracking performance would have been slightly improved had only one predictor been used throughout the experiment, but distinguishing between the best and the almost best of the five predictors was in many cases a moot point, since the target likely would have been successfully intercepted with either. The ability of the algorithm to correctly select the best predictor might be improved by incorporating a reliability factor for each predictor into the least squared error requirement. These weighting factors could be a function of individual predictor performance during the entire experiment and would tend to filter out random good predictions in a generally poor predictor.

A potentially serious problem was evident when the absolute n-step predictions were used to direct the robot's motion. Significant tracking errors were likely to result from the large prediction steps during the initial phase of the experiment. These errors could have been avoided by waiting to send the first tracking instruction to the robot until several prediction iterations had been completed.

5.2.2. Hardware

The general applicability of this research was hindered by the timing limitations imposed by the M.I.C. vision module and the Puma robot system. The vision system offers versatile and sophisticated options such as connectivity analysis and blob recognition at the expense of computation time. This implementation required fast computation of the centroid of a single object, for which coarse grid scanning would have sufficed. By selecting the count bits option over connectivity analysis, the picture processing time was reduced from 580 msecs to 216 msecs. Additional savings could have resulted from a parallel instead of serial line connection to the 11/23. To process one picture and return the center of gravity, it was necessary to send and receive approximately 82 characters over the 9600 baud serial line. Use of the available parallel port connection therefore would have reduced total processing time by another 82 msecs as well as eliminating the need for execution of character to integer conversion routines.

The general purpose design of the integral Puma system presented additional timing problems. Because it was not intended to serve primarily as a slave to an external executive, the *Val* language does not offer a specific command which allows the host to specify absolute joint positions. It was necessary to invoke a series of functions to pass absolute joint position information to the robot, resulting in a total of approximately 200 characters sent and received over the serial line for each desired robot position. Additional constraints resulted from the use of terminal emulator serial line i/o routines. Because the Puma was designed to expect input from a terminal, its input queue is apparently very short and consistently overflowed when communication was at the maximum baud rate. It was therefore necessary to introduce a stall loop into the 11/23 output routine, significantly increasing the amount of time spent in communication.

Unimation now markets a Univision system [Unimation 81], which consists of a Puma robot and a M.I.C. vision module connected by a parallel interface. This system would address some of the problems presented here if the 11/23 could maintain its executive position by intercepting the communication between the two devices, performing its prediction computations on the vision data and then issuing its desired robot positions to the robot as vision system emulated commands.

5.3. Suggestions for Future Work

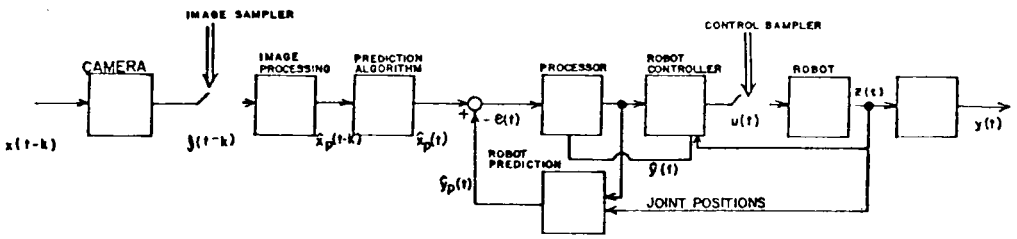

Figure 5-1: Closed-Loop Tracking System

A logical extension of this research is to implement predictive target tracking in a closed loop system as diagramed in Figure 5-1 where the robot's positioning system can be updated regardless of its status. Target predictions together with the simultaneous robot position prediction would define the error signal sent to the processor for coordinate transformation. The robot controller would receive the desired robot position as input, returning the resulting estimated robot position as feedback to the processor. A similar procedure could be implemented using visual-servoing techniques.

In conjunction with the closed loop approach or as a separate study, another possible extension of this research would be to bypass the limitations of the Puma controller by using the computed predictions to drive the robot's joints directly. This would be a major

undertaking, requiring the host to manage all joint transformations and trajectory control, but could result in a significant improvement in tracking performance.

Appendix A
Parameter Selection for the Kalman Filters

Before the Kalman filters can be implemented specific values must be assigned to the four noise variances σ_1^2, σ_2^2, σ_x^2, and σ_y^2. The augmented Kalman filter in addition requires specification of ρ_1 and ρ_2, the correlation between successive accelerations in the x and y directions respectively. These values could be specified probabilistically by assuming the form of the density functions taken by the noise processes [Schwartz 75]. The lack of statistics for the target motion and camera noise made this infeasible. The six parameters were instead determined empirically from preliminary observation data.

A series of experiments were conducted in which camera observation data were collected of each moving target with the robot disabled. Six trials were run with target 1 and four with target 2. An off-line two dimensional search was conducted on these data with the Kalman filter to determine the pair of noise variances in each direction which yielded the least mean squared prediction error over the entire experiment. Because σ_1^2 and σ_x^2 influenced only x and σ_2^2 and σ_y^2 only y, the search could be conducted in two dimensions for each direction independently, rather than requiring a computationally expensive four dimensional search.

To determine the best pair of noise variances in the x direction the data were presented to the filter in a simulation of the real time procedure. One-step predictions and their squared errors were computed at each iteration. The desired output was the mean squared prediction error for the given pair of variances. This procedure was repeatedly followed to give errors for the 81 parameter pairs where σ_1^2 and σ_x^2 took the values .01, .02, .05, .2, .5, 2, 5, 20, and 100.

Each data set was allowed two "yes votes" for the two parameter pairs which produced the least errors and two "no votes" for the two least successful pairs. These votes were plotted on a grid and visually inspected. The yes votes were found to be grouped in a cluster for target 1 and in a separate cluster for target 2. The no votes in general divided into two clusters independent of the target and separate from the two yes clusters. The filter parameter pairs for each target were selected by choosing a representative pair from the yes cluster while requiring that no adjacent grid square contain a no vote.

An identical procedure was followed for the variances in the y direction. The parameters selected for the Kalman filter were

target 1: $\sigma_x^2 = .01$ $\sigma_1^2 = .05$
$\sigma_y^2 = .01$ $\sigma_2^2 = .05$

target 2: $\sigma_x^2 = .01$ $\sigma_1^2 = .05$
$\sigma_y^2 = .02$ $\sigma_2^2 = 20.0$

The augmented Kalman filter required specification of three parameters for each component. These parameters were determined in a similar manner except that a three dimensional search was performed. The variances took the same range of values as in the two dimensional case, and ρ_1 and ρ_2 were varied from 0.0 to 1.0 in increments of 0.2. The same criteria were used to select the best parameter set from the three dimensional grid, with the following results.

target 1: $\sigma_x^2 = .02$ $\sigma_1^2 = .05$ $\rho_1 = .4$
$\sigma_y^2 = 2.0$ $\sigma_2^2 = .02$ $\rho_2 = .2$

target 2: $\sigma_x^2 = .01$ $\sigma_1^2 = .02$ $\rho_1 = .8$
$\sigma_y^2 = .02$ $\sigma_2^2 = .2$ $\rho_2 = 1.0$

References

[Anderson 79] B. Anderson, J. Moore.
Optimal Filtering.
Prentice-Hall, Inc., 1979.

[Ayres 81] R. Ayres, S. Miller.
The Impacts of Industrial Robots.
Technical Report, Carnegie-Mellon Univ., The Robotics Institute, November, 1981.

[Battin 70] R.H. Battin, G.M. Levine.
Application of Kalman Filtering Techniques to the Apollo Program.
In C.T. Leondes (editor), *Theory and Applications of Kalman Fitering*, chapter 14. NATO AGARD, 1970.

[Bauzil 81] G. Bauzil, M. Briot, P. Ribes.
A Navigation Sub-System Using Ultrasonic Sensors for the Mobile Robot Hilare.
In *Proceedings of the 1st International Conference on Robot Vision and Sensory Controls*, pages 47-58". Laboratoire d'Automatique et d'Analyse des Systemes du C.N.R.S., Toulouse, France, April, 1981.

[Bevington 69] P.R. Bevington.
Data Reduction and Error Analysis for the Physical Sciences.
McGraw-Hill, Inc., 1969, chapter 6.

[Carlisle 81] B. Carlisle, S. Roth, J. Gleason, D. McGhie.
The Puma/VS-100 Robot Vision System.
In *Proceedings of the 1st International Conference on Robot Vision and Sensory Controls*, pages 149 - 160. , April, 1981.

[Dawson 79] B.L. Dawson.
Moving Line Applications with a Computer Controlled Robot.
industrial Robots 1:117 - 131, 1979.

[Heer 79] E. Heer.
Robot and Automation Technology Requirements for Space Industrialization.
In *9th International Symposium on Industrial Robots*, pages 107 - 124. SME and RIA, March, 1979.

[Hill 79] J. Hill, W. Park.
Real Time Control of a Robot with a Mobile Camera.
In *9th International Symposium on Industrial Robots*, pages 233 - 246. SME and RIA, March, 1979.

[Holdsworth 70] J. Holdsworth, J. Stolz.
Marine Applications of Kalman Filtering.
In C.T. Leondes (editor), *Theory and Applications of Kalman Filtering*, chapter 17. NATO AGARD, 1970.

[Kalman 60] R.E. Kalman.
A New Approach to Linear Filtering and Prediction Problems.
Trans. ASME, Journal of Basic Engineering 82 D:35 - 45, March, 1960.

[Kalman 61] R.E. Kalman, R.S. Bucy.
New Results in Linear Filtering and Prediction Theory.
Trans. ASME, Journal of Basic Engineering 83:95 - 108, March, 1961.

[M.I.C. 80] *VS-100 Reference Manual*
Machine Intelligence Corp., 1980.

[Makhlin 81] A.G. Makhlin.
Westinghouse Visual Inspection and Industrial Robot Control System.
In *Proceedings of the 1st International Conference on Robot Vision and Sensory Controls*, pages 35 - 46. Robotics Technology Division, Westinghouse Research and Development Center, Pittsburgh, PA, April, 1981.

[Moravec 80] H.P. Moravec.
Obstacle Avoidance and Navigation in the Real World by a Seeing Eye Robot Rover.
Technical Report 3, Carnegie-Mellon University, The Robotics Institute, Sept, 1980.

[Papoulis 65] A. Papoulis.
Probability, Random Variables, and Stochastic Processes.
McGraw-Hill, 1965.

[Schwartz 75] M. Schwartz, L. Shaw.
Signal Processing: Discrete Spectral Analysis, Detection, and Estimation.
McGraw-Hill, Inc., 1975.

[Singer 71] R.A. Singer, K.W. Behnke.
Real-Time Tracking Filter Evaluation and Selection for Tactical Applications.
IEEE Trans. on Aerospace and Electronic Systems 7(1):100 - 110, January, 1971.

[Sorenson 70a] H.W. Sorenson.
Least-Squares Estimation from Gauss to Kalman.
IEEE Spectrum :63 - 68, July, 1970.

[Sorenson 70b] H.W. Sorenson, A.R. Stubberud.
Linear Estimation Theory.
In C.T. Leondes (editor), *Theory and Applications of Kalman Filtering*, chapter 1. NATO AGARD, 1970.

[Tretter 76] S.A. Tretter.
Introduction to Discrete-Time Signal Processing.
John Wiley & Sons, 1976, chapter 14.

[Unimation 80] *User's Guide to VAL*
Unimation Inc., 1980.

[Unimation 81] *Univision Interface Documentation*
Unimation Inc., 1981.

[Ward 79] M.R. Ward, L. Rossol, S.W. Holland, R. Dewar.
Consight: A Practical Vision-Based Robot Guidance System.
In 9^{th} *International Symposium on Industrial Robots*, pages 195 - 211. SME and RIA, March, 1979.

[Weiss 81] L. E. Weiss.
Adaptive Visual Servo Control of Robot: Analysis and Design of an Image-Based System.
June, 1981.
PhD Thesis Proposal, Carnegie-Mellon Univ.