To appear in the *Transactions on Robotics and Automation*

Six Degree-of-Freedom Hand/Eye Visual Tracking with Uncertain Parameters

Nikolaos P. Papanikolopoulos[1], Bradley J. Nelson[2], and Pradeep K. Khosla[3]

Abstract

Algorithms for full 3D robotic visual tracking of moving targets whose motion is 3D and consists of translational and rotational components are presented. The objective of the system is to track selected features on moving objects and to place their projections on the image plane at desired positions by appropriate camera motion. The most important characteristics of the proposed algorithms are the use of a single camera mounted on the end-effector of a robotic manipulator (eye-in-hand configuration), and the fact that these algorithms do not require accurate knowledge of the relative distance of the target object from the camera frame. The detection of motion is based on a cross-correlation technique known as Sum-of-Squares Differences (SSD) algorithm. The camera model used introduces a number of parameters that are estimated on-line, further reducing the algorithms' reliance on precise calibration of the system. An adaptive control algorithm compensates for modeling errors, tracking errors, and unavoidable computational delays which result from time-consuming image processing. Experimental results are presented to verify the efficacy of the proposed algorithms and to highlight the limitations of the approach. These experiments were performed using a multi-robotic system consisting of Puma 560 manipulators.

Keywords: Robotic Visual Servoing, Adaptive Control, Image Motion, Calibration.

1. Department of Computer Science, University of Minnesota, 200 Union St. SE, Minneapolis, MN 55455.
2. The Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA 15213-3890.
3. Department of Electrical and Computer Engineering and The Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA 15213-3891.

1. Introduction

The ability to track moving objects using visual feedback is an important characteristic that many intelligent robotic systems must possess. This ability is often necessary if robotic systems are to inspect, grasp, or assemble objects in environments that are dynamically varying or inherently difficult to calibrate. Workspaces in which objects are transported on conveyer belts or in another moving robot's grasp fall into this category, as well as underwater, toxic, and outer space environments. The main advantages that distinguish visual sensors from other types of sensors and make them particularly useful for these kinds of tasks, are that vision is a non-contact sensing mode that is able to provide information over a relatively large region of a system's workspace. Even so, statically located visual sensors have a very limited working region when one considers the limited depth-of-field and spatial resolution that vision sensors typically possess. However, the working region of a visual sensor such as a CCD camera can be greatly extended if the camera is allowed to move while tracking and observing objects of interest. Simple 2D planar tracking of objects with a camera that can also move in 2D is a well understood and easily implemented task. The tracking of objects with full 3D (six degree-of-freedom) motion using a single camera with full 3D motion, conversely, has proven to be an extremely difficult task to achieve with actual eye-in-hand robotic servoing systems. This paper shows how several cases of full 3D object motion can be tracked and presents experimental results which demonstrate an eye-in-hand system successfully tracking full 3D motion.

The problems that make full 3D eye-in-hand tracking difficult to achieve include the need to compensate for the noise that exists in all images, the time consuming image processing algorithms that must be used, the large amounts of data that must be processed, the nonlinearities inherent in a camera-lens system, and the difficulty in inferring full 3D motion from a series of single 2D images. From a control theory standpoint these problems translate into large delays in the feedback loop, unreliable sensor data, inadequately modeled or difficult to model plants or systems, and poorly-conditioned transformations.

Through proper modeling of the eye-in-hand system, by employing adaptive control techniques, and by using robust and fast image processing routines on commercially available image processing hardware, we have experimentally demonstrated that tracking an object with full 3D motion (both translational and rotational components) by a camera that also possesses full 3D motion capabilities can be achieved. Our experimental tracking system consists of several important characteristics that contribute to the system's usefulness and success. We use a single camera mounted on the end-effector of a six degree-of-freedom manipulator in order to demonstrate that relatively unsophisticated off-the-shelf hardware can be used to solve the 3D tracking problem. Visual mea-

surements are based on a pyramidal Sum-of-Squares Differences (SSD) algorithm [1]. A properly formulated adaptive control algorithm (designed on several simplifying assumptions) uses these measurements to determine the correct input to a cartesian robot controller. Numerical issues related to the strong coupling that exists between the rotational and translational degrees of freedom of the object being tracked are, for the first time, treated in a way that guarantees tracking of the object in the majority of the cases. However, our algorithms for full 3-D tracking fail in several cases that are discussed in the experimental section (one of them is the case of an object that is spinning much faster than the camera system can roll about its optical axis). It should be pointed out that we propose algorithms for an approximation of a very nonlinear problem.

The major differences of our system from similar research efforts [3][4][5][19][20][21] are the use of a single moving camera, the ability to compensate for inaccurate camera parameters and unknown depth (distance from the camera to the target), full 3D tracking ability, the small number of parameters that are estimated on-line, and the integration of the characteristics of the motion detection algorithm into a mathematical model for tracking.

These differences allow the system to be used in environments that are inherently difficult to calibrate, such as underwater, in toxic environments, or in outer space. This paper extends our previous work [14][15][16] in Controlled Active Vision by allowing full 3D tracking of objects, by reducing the number of parameters that are estimated on-line, and by presenting experimental results from tests performed using commercial manipulators. The experiments were performed on the Rapid Assembly System which consists of three Puma 560's. In order to test our approach, a tracking Puma holds a camera and responds to arbitrary full 3D motion of an object held by a second Puma, based solely on the visual feedback provided by the camera mounted on the tracking Puma's end-effector.

We begin by describing the mathematical framework under which our problem is solved. The control, filtering, and estimation strategies are discussed in the next section, followed by a presentation of experimental results and a summary.

2. Modeling of the Visual Servoing Problem

To model the 3-D visual servoing problem, we first assume a pinhole model for the camera with a frame {C} placed at the focal point of the lens. This formulation was presented in [16] and is reviewed here. A feature on an object at $\boldsymbol{P}$[1] with coordinates (X_o,Y_o,Z_o) in the camera frame

1. Bold symbols denote vectors or matrices.

projects onto the camera's image plane at

$$x_i = \frac{fX_o}{Z_o s_x} + x_p \tag{1}$$

$$y_i = \frac{fY_o}{Z_o s_y} + y_p \tag{2}$$

where (x_i,y_i) are the image coordinates of the feature, f is the focal length of the lens, s_x and s_y are the horizontal and vertical dimensions of the pixels on the CCD array, and (x_p,y_p) is the piercing point of the optical axis on the CCD. In addition, it is assumed that $Z_o >> f$. This assumption holds because the maximum focal length of our camera is 16mm, while Z_o is approximately 300mm. Initially, let us assume that the camera moves in a static environment with a translational velocity $\boldsymbol{T}=[\dot{x}_c\ \dot{y}_c\ \dot{z}_c]^{\mathrm{T}}$ and a rotational velocity $\boldsymbol{R}=[\omega_{xc}\ \omega_{yc}\ \omega_{zc}]^{\mathrm{T}}$ with respect to the camera frame {C}. The velocity of point $\boldsymbol{P}$ in the camera frame {C} induced by camera motion is then

$$\frac{d\boldsymbol{P}}{dt} = -\boldsymbol{T} - \boldsymbol{R} \times \boldsymbol{P} \tag{3}$$

If we let $x=(x_i - x_p)$ and $y=(y_i - y_p)$ represent the projection of $\boldsymbol{P}$ on the image plane, then the velocity of the projection of $\boldsymbol{P}$ on the image plane $(\dot{x}, \dot{y})$ is equivalent to the camera induced image motion of $\boldsymbol{P}$ which we represent by (u_c,v_c). By explicitly calculating $\dot{\boldsymbol{P}}$ from (3) and determining the projection of $\dot{\boldsymbol{P}}$ on the image plane using (1) and (2), the camera induced image motion of the point $\boldsymbol{P}$ can be determined to be (assuming, for the moment, that the object is stationary)

$$u_c = -\frac{f\dot{x}_c}{Z_o s_x} + \frac{x\dot{z}_c}{Z_o} + \frac{xys_y\omega_{xc}}{f} - \left(\frac{f}{s_x} + \frac{x^2 s_x}{f}\right)\omega_{yc} + \frac{ys_y}{s_x}\omega_{zc} \tag{4}$$

$$v_c = -\frac{f\dot{y}_c}{Z_o s_y} + \frac{y\dot{z}_c}{Z_o} + \left(\frac{f}{s_y} + \frac{y^2 s_y}{f}\right)\omega_{xc} - \frac{xys_x\omega_{yc}}{f} - \frac{xs_x}{s_y}\omega_{zc} \tag{5}$$

The image motion that can actually be observed on the image plane, however, is due to both the image motion induced by camera motion (u_c,v_c) and the image motion induced by object motion (u_o,v_o). The observed image motion, which we represent by (u,v), can only be determined for successive image frames which are separated in time by a sampling period T, and can be written as

$$u(kT) = u_o(kT) + u_c(kT) \tag{6}$$

$$v(kT) = v_o(kT) + v_c(kT) \tag{7}$$

The above equations do not account for the computational delays that occur when calculating the observed image motion and in determining the camera induced image motion. When we include

these delays in our model, (6) and (7) become

$$u(k) = u_o(k) + q^{-d+1}u_c(k) \tag{8}$$

$$v(k) = v_o(k) + q^{-d+1}v_c(k) \tag{9}$$

where d is the delay factor ($d \in \{1,2,...\}$), q^{-1} is the backward shift operator, and we let $k=kT$ in order to simplify notation without any loss of generality. It can easily be seen that the observed image motion between successive images can also be represented by

$$u(k) = \frac{x(k+1) - x(k)}{T} \tag{10}$$

$$v(k) = \frac{y(k+1) - y(k)}{T} \tag{11}$$

If we substitute $u(k)$ and $v(k)$ in (8) and (9) with their equivalent expressions from (10) and (11), and if we assume model inaccuracies due to neglected accelerations and inaccurate robot control can be represented as white noise, then (8) and (9) can be rewritten as

$$x(k+1) = x(k) + Tq^{-d+1}u_c(k) + Tu_o(k) + v_x(k) \tag{12}$$

$$y(k+1) = y(k) + Tq^{-d+1}v_c(k) + Tv_o(k) + v_y(k) \tag{13}$$

where $v_x(k)$ and $v_y(k)$ are zero-mean, mutually uncorrelated, stationary random variables with variances σ_x^2 and σ_y^2, respectively. Equations (12) and (13) can be written in state-space form as

$$\boldsymbol{x_F}(k+1) = \boldsymbol{A_F}\boldsymbol{x_F}(k) + \boldsymbol{B_F}(k-d+1)\boldsymbol{u}(k-d+1) + \boldsymbol{E_F}\boldsymbol{d_F}(k) + \boldsymbol{H_F}\boldsymbol{v_F}(k) \tag{14}$$

where[1] $\boldsymbol{A_F}=\boldsymbol{H_F}=\mathbf{I}_2$, $\boldsymbol{E_F}=T\mathbf{I}_2$, $\boldsymbol{x_F}(k)\in R^2$, $\boldsymbol{d_F}(k)\in R^2$, $\boldsymbol{u}(k)\in R^6$, and $\boldsymbol{v_F}(k)\in R^2$. The matrix $\boldsymbol{B_F}(k)\in R^{2x6}$ is derived by discretizing (4) and (5) and writing the terms representing the camera-lens geometry separate from the camera motion terms, which are the control inputs of the system. The resulting matrix is

$$\boldsymbol{B_F}(k) = \begin{bmatrix} -\frac{f}{Z_o(k)s_x} & 0 & \frac{x(k)}{Z_o(k)} & \frac{x(k)y(k)s_y}{f} & -\left(\frac{f}{s_x} + \frac{x^2(k)s_x}{f}\right) & \frac{y(k)s_y}{s_x} \\ 0 & -\frac{f}{Z_o(k)s_y} & \frac{y(k)}{Z_o(k)} & \left(\frac{f}{s_y} + \frac{y^2(k)s_y}{f}\right) & -\frac{x(k)y(k)s_x}{f} & -\frac{x(k)s_x}{s_y} \end{bmatrix} \tag{15}$$

The vector $\boldsymbol{x_F}(k)=[x(k)\ y(k)]^{\mathrm{T}}$ is the state vector, $\boldsymbol{u}(k)=[\dot{x}_c\ \dot{y}_c\ \dot{z}_c\omega_{xc}\ \omega_{yc}\omega_{zc}]^{\mathrm{T}}$ is the control input vector, $\boldsymbol{d_F}(k)=[u_o(k)\ v_o(k)]^{\mathrm{T}}$ is the exogenous deterministic disturbances vector, and $\boldsymbol{v_F}(k)=[v_x(k)\ v_y(k)]^{\mathrm{T}}$ is a white noise vector. The measurement vector $\boldsymbol{y_F}(k)=[y_x(k)\ y_y(k)]^{\mathrm{T}}$ for the feature is given by

$$\boldsymbol{y_F}(k) = \boldsymbol{C_F}\boldsymbol{x_F}(k) + \boldsymbol{w_F}(k) \tag{16}$$

where $\boldsymbol{w_F}(k)=[w_x(k)\ w_y(k)]^{\mathrm{T}}$ is a white noise vector $\boldsymbol{w}(k) \sim N(\mathbf{0}, \boldsymbol{W})$ and $\boldsymbol{C_F}=\mathbf{I}_2$. The measurement

1. The symbol $\mathbf{I}_n$ denotes the identity matrix of order n.

$\boldsymbol{y_F}(k)$ is computed using the SSD algorithm described in [14]. In addition, a method for automatic selection of features and on-line evaluation of visual measurements (useful in cases of sudden occlusion of features) is presented in [14].

In order to solve for the manipulator control input, it can be shown that at least three feature points which are not collinear are needed [17]. In other words, less than three feature points do not provide enough measurements in order to reliably compute the manipulator control input. The state space model for M ($M \geq 3$) feature points can be written as

$$\boldsymbol{x}(k+1) \;=\; \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}(k-d+1)\boldsymbol{u}(k-d+1) + \boldsymbol{E}\boldsymbol{d}(k) + \boldsymbol{H}\boldsymbol{v}(k) \tag{17}$$

where $\boldsymbol{A}=\boldsymbol{H}=\mathbf{I}_{2M}$, $\boldsymbol{E}=T\mathbf{I}_{2M}$, $\boldsymbol{x}(k)\in R^{2M}$, $\boldsymbol{d}(k)\in R^{2M}$, $\boldsymbol{u}(k)\in R^{6}$, $\boldsymbol{v}(k)\in R^{2M}$. The matrix $\boldsymbol{B}(k)\in R^{2M\text{x}6}$ and is

$$\boldsymbol{B}(k) \;=\; \begin{bmatrix} \boldsymbol{B_F}^{(1)}(k) \\ \cdots \\ \boldsymbol{B_F}^{(M)}(k) \end{bmatrix} \tag{18}$$

The superscript (j) denotes each of the feature points ($j\in\{1,2,...,M\}$). The vector $\boldsymbol{x}(k)=[x^{(1)}(k)\ y^{(1)}(k)...\ x^{(M)}(k)\ y^{(M)}(k)]^{\mathrm{T}}$ is the new state vector, $\boldsymbol{d}(k)=[u_o{}^{(1)}(k)\ v_o{}^{(1)}(k)...\ u_o{}^{(M)}(k)\ v_o{}^{(M)}(k)]^{\mathrm{T}}$ is the new exogenous deterministic disturbances vector, and $\boldsymbol{v}(k)=[v_x{}^{(1)}(k)\ v_y{}^{(1)}(k)...\ v_x{}^{(M)}(k)\ v_y{}^{(M)}(k)]^{\mathrm{T}}$ is the new white noise vector. The new measurement vector $\boldsymbol{y}(k)=[y_x{}^{(1)}(k)\ y_y{}^{(1)}(k)...\ y_x{}^{(M)}(k)\ y_y{}^{(M)}(k)]^{\mathrm{T}}$ for the feature is given by

$$\boldsymbol{y}(k) \;=\; \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{w}(k) \tag{19}$$

where $\boldsymbol{w}(k)=[w_x{}^{(1)}(k)\ w_y{}^{(1)}(k)...\ w_x{}^{(M)}(k)\ w_y{}^{(M)}(k)]^{\mathrm{T}}$ is a white noise vector $\boldsymbol{w}(k)\sim N(\mathbf{0}, \boldsymbol{W})$ and $\boldsymbol{C}=\mathbf{I}_{2M}$.

We can combine (17) and (19) into a MIMO (Multi-Input Multi-Output) model (it is assumed that $\boldsymbol{d}(k)$ is constant)

$$(1-2q^{-1}+q^{-2})\,\boldsymbol{y}(k) \;=\; \boldsymbol{B}(k-d)\boldsymbol{u}(k-d) - \boldsymbol{B}(k-d-1)\boldsymbol{u}(k-d-1) + \boldsymbol{n}(k) \tag{20}$$

where $\boldsymbol{n}(k)$ is the white noise vector, and corresponds to the measurement noise, modeling errors, and noise introduced by inaccurate robot control. We can assume that $\boldsymbol{B}(k\text{-}d)\approx\boldsymbol{B}(k\text{-}d\text{-}1)$ if camera and object motion between successive samples is sufficiently small. This reduces the complexity of (20), and allows us to rewrite this equation as a MIMO ARX (AutoRegressive with eXternal input) model. This model consists of $2M$ MISO (Multi-Input Single-Output) ARX models. The new model's equation is

$$(1-2q^{-1}+q^{-2})\,\boldsymbol{y}(k) \;=\; \boldsymbol{B}(k-d)\Delta\boldsymbol{u}(k-d) + \boldsymbol{n}(k) \tag{21}$$

where $\Delta \boldsymbol{u}(k\text{-}d)$ is defined as

$$\Delta \boldsymbol{u}(k-d) = \boldsymbol{u}(k-d) - \boldsymbol{u}(k-d-1) \tag{22}$$

It is important to note the need to create a single MIMO model or multiple MISO models for the system, rather than multiple SISO models, due to the strong coupling that exists between translational and rotational motion along and about the camera X and Y axes. Although multiple SISO models are computationally far easier to implement, these models are unable to account for this coupling. This problem has been extensively studied in [14][16] and is related to the accurate computation of individual translational and rotational tracking motions.

In the next section we present the control and estimation techniques for the 3D visual tracking problem.

3. Control and Estimation

The control objective is to move the manipulator holding the camera so that the features on the object being tracked move to some desired positions on the image plane, or that these features remain at some desired positions as the object being tracked moves. Adaptive control techniques can be particularly effective for visually servoing a manipulator that tracks a moving object when the depth of the object with respect to the hand-held vision sensor is not precisely known (the success of tracking does not depend on the accurate knowledge or accurate estimation of the depth). These techniques use the estimated values of the unknown parameters in order to compute the control signal. This approach is called *certainty equivalence adaptive control* [8]. A variety of tracking algorithms can be created depending on the parameter estimation schemes and control laws chosen. The rest of this section is devoted to a description of the control and estimation schemes, and highlights the differences from the techniques reported in [16].

3.1. Selection of an Efficient Control Law

As stated previously, the control objective is to track the motion of certain features on the target and place the projection of these features at some desired positions on the image plane. The tracking of the features' projections is realized by an appropriate motion of the robot-camera system described by the model in (21). A simple control law can be derived by the minimization of a cost function that allows weights to be placed on the positional error of the features, the control signal, and the change in control signal

$$J(k+d) = [\boldsymbol{y}(k+d) - \boldsymbol{y}_D(k+d)]^T \boldsymbol{Q} [\boldsymbol{y}(k+d) - \boldsymbol{y}_D(k+d)] + \boldsymbol{u}^T(k) \boldsymbol{L} \boldsymbol{u}(k) + \Delta \boldsymbol{u}^T(k) \boldsymbol{L}_d \Delta \boldsymbol{u}(k) \tag{23}$$

The vector $\boldsymbol{y}_D(k)$ represents the desired positions of the projections of the M features on the image

plane. This vector is known *a priori* and may be constant or time-varying. In (23), $[\boldsymbol{y}(k+d)-\boldsymbol{y}_D(k+d)]^T\boldsymbol{Q}[\boldsymbol{y}(k+d)-\boldsymbol{y}_D(k+d)]$ represents the cost of the feature or servoing error, $\boldsymbol{u}^T(k)\boldsymbol{L}\boldsymbol{u}(k)$ is the cost of providing a control input, and $\Delta\boldsymbol{u}^T(k)\boldsymbol{L}_d\Delta\boldsymbol{u}(k)$ is the cost of changing the control input. The selection of the weighting matrices $\boldsymbol{L}$, $\boldsymbol{L}_d$, and $\boldsymbol{Q}$ allows one to place more or less emphasis on the control input, the control input change, and the servoing error when attempting to satisfy the control objective. The control law is derived from the minimization of the cost function (23) by taking the derivative of $J(k+d)$ with respect to the vector $\boldsymbol{u}(k)$, setting the derivative of $J(k+d)$ to zero, combining the result with the system model in (21), and solving for $\boldsymbol{u}(k)$ (the procedure can be found in [8][14]). The resulting control law is

$$\boldsymbol{u}(k) = -(\boldsymbol{B}^T(k)\boldsymbol{Q}\boldsymbol{B}(k)+\boldsymbol{L}+\boldsymbol{L}_d)^{-1}\Big[\boldsymbol{B}^T(k)\boldsymbol{Q}\{(d+1)\boldsymbol{y}(k)-\boldsymbol{y}_D(k+d)-d\boldsymbol{y}(k-1)- d\boldsymbol{B}(k-d)\boldsymbol{u}(k-d)+\sum_{m=1}^{d-1}\boldsymbol{B}(k-m)\boldsymbol{u}(k-m)\}-\boldsymbol{L}_d\boldsymbol{u}(k-1)\Big] \tag{24}$$

An important characteristic of this control law is that we impose constraints on the components of required camera tracking motion $\boldsymbol{u}(k)$, which is a constraint existing in the camera frame space, rather than imposing constraints on the image motion induced by camera motion, which is a constraint represented in the image plane space. Thus, we directly control the magnitudes of the control signal and the control signal change. This results in a control law that is more robust and feasible than the one proposed in [5].

It should be noted that the term $\Delta\boldsymbol{u}(k)\boldsymbol{L}_d\Delta\boldsymbol{u}(k)$ in the cost function (23) introduces an integral term into the control law. This term is desirable since our mathematical model (21) has a deterministic disturbance component. However, this term can lead to possible saturation of the control inputs, so it becomes necessary to determine when saturation has occurred in order to turn off the integrator.

The design parameters in our control law include the elements of the matrices $\boldsymbol{L}$, $\boldsymbol{L}_d$, and $\boldsymbol{Q}$. We often set $\boldsymbol{L}=\boldsymbol{0}$ and $\boldsymbol{L}_d\neq\boldsymbol{0}$ in order to achieve a fast and bounded response. The matrix $\boldsymbol{Q}$ must be positive definite, while $\boldsymbol{L}$ and $\boldsymbol{L}_d$ must be positive semi-definite. If the matrix $\boldsymbol{B}(k)$ is full rank, then the matrix $[\boldsymbol{B}^{\mathrm{T}}(k)\boldsymbol{Q}\boldsymbol{B}(k)+\boldsymbol{L}+\boldsymbol{L}_d]$ is invertible. The matrix $\boldsymbol{B}(k)$ loses rank when the M feature points are collinear [5][14]. An extensive study of other conditions which cause a loss of rank in $\boldsymbol{B}(k)$ can be found in [14]. Finally, several researchers [7][9][11][18] in computer vision have studied this problem from the pose estimation perspective.

Unfortunately, no standard procedure exists for choosing the individual elements of $\boldsymbol{L}$, $\boldsymbol{L}_d$, and $\boldsymbol{Q}$. A common technique to employ is the optimization technique [12]. This is a straightforward way

to include the constraints that the robotic device imposes on the control amplitudes in the control law, so that the control signals are bounded and feasible. This also allows us to account for the limited search region of the SSD algorithm, so that control inputs will not cause the manipulator to move the camera beyond the maximum feature displacement of 32 pixels that our vision system can measure between successive frames.

The control law derived previously (24) did not account for noise or inaccuracies in the camera and depth related parameters contained in $\boldsymbol{B}(k)$. Noise and inaccurate parameter values can cause the system to exhibit sluggish and even unstable tracking behavior. When these factors are taken into account, the control objective (23) becomes

$$J(k+d) = E\{ [\boldsymbol{y}(k+d) - \boldsymbol{y}_D(k+d)]^T \boldsymbol{Q} [\boldsymbol{y}(k+d) - \boldsymbol{y}_D(k+d)] + \\ + \boldsymbol{u}^T(k)\boldsymbol{L}\boldsymbol{u}(k) + \Delta\boldsymbol{u}^T(k)\boldsymbol{L}_d\Delta\boldsymbol{u}(k) \,\big|\, F_k \} \tag{25}$$

where the symbol $E\{X\}$ denotes the expected value of the random variable X and F_k is the sigma algebra generated by the past measurements and the past control inputs up to time k. The new control law is (based on the *certainty equivalence principle* [8])

$$\boldsymbol{u}(k) = -(\hat{\boldsymbol{B}}^T(k)\boldsymbol{Q}\hat{\boldsymbol{B}}(k) + \boldsymbol{L} + \boldsymbol{L}_d)^{-1} \Big[\hat{\boldsymbol{B}}^T(k)\boldsymbol{Q}\{ (d+1)\boldsymbol{y}(k) - \boldsymbol{y}_D(k+d) - d\boldsymbol{y}(k-1) - \\ - d\hat{\boldsymbol{B}}(k-d)\boldsymbol{u}(k-d) + \sum_{m=1}^{d-1} \hat{\boldsymbol{B}}(k-m)\boldsymbol{u}(k-m) \} - \boldsymbol{L}_d\boldsymbol{u}(k-1) \Big] \tag{26}$$

where $\hat{\boldsymbol{B}}(k)$ is the estimated value of the matrix $\boldsymbol{B}(k)$. The matrix $\hat{\boldsymbol{B}}(k)$ is dependent on the estimated values of the features' depth $\hat{Z}_o^{(j)}(k)$ ($j \in \{1,2,...,M\}$) and the coordinates of the features' image projections. In particular, the matrix $\hat{\boldsymbol{B}}(k)$ is defined as

$$\hat{\boldsymbol{B}}(k) = \begin{bmatrix} \hat{\boldsymbol{B}}_{\boldsymbol{F}}^{(1)}(k) \\ \dots \\ \hat{\boldsymbol{B}}_{\boldsymbol{F}}^{(M)}(k) \end{bmatrix} \tag{27}$$

where $\hat{\boldsymbol{B}}_{\boldsymbol{F}}^{(j)}(k)$ is given by

$$\hat{\boldsymbol{B}}_{\boldsymbol{F}}^{(j)}(k) = \begin{bmatrix} -\dfrac{f}{\hat{Z}_o^{(j)}(k)s_x} & 0 & \dfrac{x^{(j)}(k)}{\hat{Z}_o^{(j)}(k)} & \dfrac{x^{(j)}(k)y^{(j)}(k)s_y}{f} & -\left(\dfrac{f}{s_x} + \dfrac{(x^{(j)}(k))^2 s_x}{f}\right) & \dfrac{y^{(j)}(k)s_y}{s_x} \\ 0 & -\dfrac{f}{\hat{Z}_o^{(j)}(k)s_y} & \dfrac{y^{(j)}(k)}{\hat{Z}_o^{(j)}(k)} & \left(\dfrac{f}{s_y} + \dfrac{(y^{(j)}(k))^2 s_y}{f}\right) & -\dfrac{x^{(j)}(k)y^{(j)}(k)s_x}{f} & -\dfrac{x^{(j)}(k)s_x}{s_y} \end{bmatrix} \tag{28}$$

3.2. Estimation of Depth Related Parameters

There are several methods which can be used to estimate the depth related parameters that appear in $\hat{\boldsymbol{B}}(k)$. In this section, we describe one technique which we have successfully implemented, and in [14] we describe two other techniques which are related to the following scheme. The estimation scheme that is described is simpler and more effective than the one that we reported in [16]. For the technique used to produce the experimental results given in Section 4, we define $(f/(s_x Z_o^{(j)}(k)))$ as $\zeta_o^{(j)}(k)$ so that (21) can be rewritten as

$$\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k) = 2\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k-1) - \boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k-2) + \zeta_o^{(j)}(k-d)\boldsymbol{B}_{\boldsymbol{F}_t}^{(j)}(k-d)\Delta\boldsymbol{T}(k-d) + \boldsymbol{B}_{\boldsymbol{F}_r}^{(j)}(k-d)\Delta\boldsymbol{R}(k-d) + \boldsymbol{n}_{\boldsymbol{F}}^{(j)}(k) \tag{29}$$

where $\boldsymbol{n}(k) \sim N(\boldsymbol{0}, \boldsymbol{N})$,

$$\boldsymbol{B}_{\boldsymbol{F}_t}^{(j)}(k) = T\begin{bmatrix} -1 & 0 & \dfrac{x^{(j)}(k)s_x}{f} \\ 0 & -\dfrac{s_x}{s_y} & \dfrac{y^{(j)}(k)s_y}{f} \end{bmatrix} \tag{30}$$

$$\boldsymbol{B}_{\boldsymbol{F}_r}^{(j)}(k) = T\begin{bmatrix} \dfrac{x^{(j)}(k)y^{(j)}(k)s_y}{f} & -\left(\dfrac{f}{s_x} + \dfrac{(x^{(j)}(k))^2 s_x}{f}\right) & \dfrac{y^{(j)}(k)s_y}{s_x} \\ \left(\dfrac{f}{s_y} + \dfrac{(y^{(j)}(k))^2 s_y}{f}\right) & -\dfrac{x^{(j)}(k)y^{(j)}(k)s_x}{f} & -\dfrac{x^{(j)}(k)s_x}{s_y} \end{bmatrix} \tag{31}$$

and

$$\Delta\boldsymbol{T}(k) = \boldsymbol{T}(k) - \boldsymbol{T}(k-1) \qquad \Delta\boldsymbol{R}(k) = \boldsymbol{R}(k) - \boldsymbol{R}(k-1) \tag{32}$$

By defining $\boldsymbol{h}_{\boldsymbol{t}}^{(j)}(k)$ and $\boldsymbol{h}_{\boldsymbol{r}}^{(j)}(k)$ as $\boldsymbol{h}_{\boldsymbol{t}}^{(j)}(k) = \boldsymbol{B}_{\boldsymbol{F}_t}^{(j)}(k)\Delta\boldsymbol{T}(k)$ and $\boldsymbol{h}_{\boldsymbol{r}}^{(j)}(k) = \boldsymbol{B}_{\boldsymbol{F}_r}^{(j)}(k)\Delta\boldsymbol{R}(k)$, respectively, equation (29) can be transformed into

$$\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k) = 2\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k-1) - \boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k-2) + \zeta_o^{(j)}(k-d)\boldsymbol{h}_{\boldsymbol{t}}^{(j)}(k-d) + \boldsymbol{h}_{\boldsymbol{r}}^{(j)}(k-d) + \boldsymbol{n}_{\boldsymbol{F}}^{(j)}(k) \tag{33}$$

For the final transformation of (29) we first define the vector $\Delta\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k)$ to be

$$\Delta\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k) = \boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k) - 2\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k-1) + \boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k-2) - \boldsymbol{h}_{\boldsymbol{r}}^{(j)}(k-d) \tag{34}$$

Equation (29) can now be rewritten in a convenient form for estimating the depth related parameter

$$\Delta\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k) = \zeta_o^{(j)}(k-d)\boldsymbol{h}_{\boldsymbol{t}}^{(j)}(k-d) + \boldsymbol{n}_{\boldsymbol{F}}^{(j)}(k) \tag{35}$$

The vector $\Delta \boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k)$ consists of values from past feature measurements and past rotational control inputs, and $\boldsymbol{h}_t^{(j)}(k)$ consists of values from past translational control inputs. Therefore, these two vectors are known, while the scalar $\zeta_o^{(j)}(k)$ is continuously estimated.

It is possible to successfully implement a full 3D eye-in-hand tracking system by assuming that the depth related parameter can be updated by standard recursive estimation equations to be given later, including the parameter update equation

$$\hat{\zeta}_o^{(j)}(k) = \hat{\zeta}_o^{(j)}(k-1) \tag{36}$$

where the superscript ($^-$) denotes the predicted value, the superscript ($^+$) denotes the updated value). The schemes presented in [14] make this assumption, however, a more accurate form for the updated parameter equation (36) can be obtained.

We propose a modified scheme based on the equation

$$Z_o^{(j)}(k+1) \approx Z_o^{(j)}(k) + \Delta Z_{obj}^{(j)}(k) + q^{-d+1}\Delta Z_{cam}^{(j)}(k) \tag{37}$$

where $\Delta Z_{obj}^{(j)}(k)$ represents the change in depth of the target due to its motion and

$$\Delta Z_{cam}^{(j)}(k) = -\{\dot{z}_c(k) + [\omega_x(k) y^{(j)}(k) s_y - \omega_y(k)(x^{(j)}(k) s_x)] \frac{Z_o^{(j)}(k)}{f}\} T \tag{38}$$

This represents the change of depth of the features on the object due solely to motion of the camera and is a known quantity since it is based on past control inputs. The value is derived by substituting the equivalent expression for $X_o^{(j)}(k)$ and $Y_o^{(j)}(k)$ from (1) and (2) into (3). Equation (37) is actually an approximation of the changing depth of the object from the camera, since accelerations of the object and camera are ignored. If the sampling time between images is sufficiently small, however, this approximation is quite reasonable. We can rewrite (37) as

$$Z_o^{(j)}(k) \approx 2Z_o^{(j)}(k-1) - Z_o^{(j)}(k-2) + \Delta Z_{cam}^{(j)}(k-d) - \Delta Z_{cam}^{(j)}(k-d-1) \tag{39}$$

Through algebraic manipulation, (39) becomes

$$\zeta_o^{(j)}(k) = \frac{\zeta_o^{(j)}(k-1)}{2 - \dfrac{\zeta_o^{(j)}(k-1)}{\zeta_o^{(j)}(k-2)} + \zeta_o^{(j)}(k-1)\dfrac{s_x}{f}[\Delta' Z_{cam}^{(j)}(k-d) - \Delta' Z_{cam}^{(j)}(k-d-1)]} \tag{40}$$

where

$$\Delta' Z_{cam}^{(j)}(k) = -\{\dot{z}_c(k) + [\omega_x(k) y^{(j)}(k) s_y - \omega_y(k) x^{(j)}(k) s_x] \frac{1}{s_x \zeta_o^{(j)}(k)}\} T \tag{41}$$

Substituting the estimated values of $\zeta_o^{(j)}(k)$ into (40), the parameter update equation given by (36)

becomes

$$ {}^{-}\zeta_o^{(j)}(k) = \frac{{}^{+}\zeta_o^{(j)}(k-1)}{2 - \frac{{}^{+}\zeta_o^{(j)}(k-1)}{{}^{+}\zeta_o^{(j)}(k-2)} + {}^{+}\zeta_o^{(j)}(k-1)\frac{s_x}{f}\left[{}^{+}\Delta' Z_{cam}^{(j)}(k-d) - {}^{+}\Delta' Z_{cam}^{(j)}(k-d-1)\right]} \tag{42} $$

The term ${}^{+}\Delta' Z_{cam}^{(j)}(k)$ is derived from $\Delta' Z_{cam}^{(j)}(k)$ in (33) by substituting $\zeta_o^{(j)}(k)$ with ${}^{+}\zeta_o^{(j)}(k)$.

The depth related parameter $\zeta_o^{(j)}(k)$ is estimated by the following equations. An initial estimate of $\hat{\zeta}_o^{(j)}(k)$, $\hat{\zeta}_o^{(j)}(0)$ must be given. We also assume that a covariance scalar $p^{(j)}(k)$ which represents

$$ p^{(j)}(k) = E\left\{\left[\zeta_o^{(j)}(k) - \hat{\zeta}_o^{(j)}(k)\right]^2\right\} \tag{43} $$

is initially given and is represented by $p_o = p^{(j)}(0)$. The value of the covariance scalar p_o can be interpreted as a measure of the confidence we have in our estimate of $\zeta_o^{(j)}(0)$. Accurate knowledge of $\zeta_o^{(j)}(0)$ corresponds to a small value for p_o. The term $s^{(j)}(k)$ is a covariance scalar which corresponds to the white noise that characterizes the transition between states and is assumed constant. $\boldsymbol{N}^{(j)}(k)$ is the constant predefined covariance matrix of the gaussian noise vector $\boldsymbol{n}_{\boldsymbol{F}}^{(j)}(k)$. The recursive equations are given by [13]

$$ {}^{-}\hat{\zeta}_o^{(j)}(k) = \frac{{}^{+}\hat{\zeta}_o^{(j)}(k-1)}{2 - \frac{{}^{+}\hat{\zeta}_o^{(j)}(k-1)}{{}^{+}\hat{\zeta}_o^{(j)}(k-2)} + {}^{+}\hat{\zeta}_o^{(j)}(k-1)\frac{s_x}{f}\left[{}^{+}\Delta' Z_{cam}^{(j)}(k-d) - {}^{+}\Delta' Z_{cam}^{(j)}(k-d-1)\right]} \tag{44} $$

$$ {}^{-}p^{(j)}(k) = {}^{+}p^{(j)}(k-1) + s^{(j)}(k-1) \tag{45} $$

$$ {}^{+}p^{(j)}(k) = \left[\left\{{}^{-}p^{(j)}(k-1)\right\}^{-1} + \boldsymbol{h}_{\boldsymbol{t}}^{(j)T}(k-d)\left\{\boldsymbol{N}^{(j)}(k)\right\}^{-1}\boldsymbol{h}_{\boldsymbol{t}}^{(j)}(k-d)\right]^{-1} \tag{46} $$

$$ \boldsymbol{k}^T(k) = {}^{+}p^{(j)}(k)\boldsymbol{h}_{\boldsymbol{t}}^{(j)T}(k-d)\left\{\boldsymbol{N}^{(j)}(k)\right\}^{-1} \tag{47} $$

$$ {}^{+}\hat{\zeta}_o^{(j)}(k) = {}^{-}\hat{\zeta}_o^{(j)}(k) + \boldsymbol{k}^T(k)\left[\Delta\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k) - {}^{-}\hat{\zeta}_o^{(j)}(k)\boldsymbol{h}_{\boldsymbol{t}}^{(j)}(k-d)\right] \tag{48} $$

The depth related parameter $\zeta_o^{(j)}(k)$ is time-varying since the target and the camera move in 3D. The estimation scheme described by equations (44)-(48) can compensate for the time-varying nature of $\zeta_o^{(j)}(k)$, because the scheme was designed under the assumption that the estimated variable undergoes a random change.

This estimation scheme requires the use of one parameter per feature point making it computationally realistic for real-time control. Some researchers, for example [5], propose the use of an adaptive scheme which estimates all of the elements of the matrix $\boldsymbol{B}(k)$ on-line. As reported in [5],

this approach is computationally difficult to perform in real-time. Our proposed approach alleviates these computational problems.

4. Experiments

The ability of the eye-in-hand system to successfully track objects whose motion is fully three dimensional using the vision, control, and estimation algorithms presented in this paper has been experimentally verified. These experiments are extensively described in [14]. The results of one of these experiments are presented in this section. For the experiments, the objective of the eye-in-hand system is to move the camera so that the image projections of the four features on the object being tracked remain at their initial positions on the camera's image plane, though, the system has the capability to allow the coordinates of the features to be placed at some desired coordinates other than their original locations. The objects that are used for tracking include books, blocks, and general items with distinct features. One Puma 560 holds the camera while the other (both Puma 560's are components of the Rapid Assembly System) holds the target.

For the experiments performed, the maximum permissible translational speed of the Puma holding the camera is 10cm/sec (maximum translational speed of the targets is 7cm/sec), and the rotational components are limited to 0.3rad/sec. The target's initial depth is approximately 290mm from the camera frame. The camera's CCD array has pixel dimensions s_x and s_y of 11μm and 13μm, respectively, and the camera lens has a focal length of 16mm. The controller gains are $\boldsymbol{Q}=0.9\boldsymbol{I}_8$, $\boldsymbol{L}=\boldsymbol{0}$, and $\boldsymbol{L_d}=diag\{0.04,0.04,1.0,5x10^5,5x10^5,5x10^5\}$. It has been experimentally determined that the diagonal elements of $\boldsymbol{Q}$, $\boldsymbol{L}$, and $\boldsymbol{L_d}$ can vary by a factor of between 2 and 3 and the system will continue to track successfully. The delay factor d is 2. The vector of the desired coordinates of the features on the image plane, $\boldsymbol{y_D}(k)$, is constant and is set to $\boldsymbol{y_D}(k)=\boldsymbol{y}(0)$. The initial values for the depth related parameters $\hat{\zeta}_o^{(j)}(k)$ and their associated covariance scalars $p^{(j)}(k)$ can be found in [14]. In particular, the initial values for the depth related parameters $\hat{\zeta}_o^{(j)}(k)$ are different from the actual values by a factor of 4.

The four features (one in each image quadrant) on the object being tracked are selected by the user with a mouse, and the tracking quality of the features are then evaluated on-line based on the confidence measures described in [14]. In our experiment, the features are the four corners of the target. If a feature does not satisfy a confidence threshold, the user is asked to select a replacement object feature. In addition, the option of invoking an automatic feature selector is available. A scheme for dealing with suddenly partially occluded targets is also presented in [14].

Figures 1 through 6 show the results from the experiment. The trajectory of the target is shown in

Figure 1 by plotting the difference in the target Puma's end-effector frame at time instant k from the initial end-effector frame at k=0, versus kT. At k=0, the Z axis of the target Puma's end-effector frame is aligned with the optical axis of the camera. Four parameters (one parameter per feature) are estimated during tracking using the estimation scheme described by (44). Figure 2 shows the actual tracking errors recorded during the experiment. The errors were calculated by determining the relative transformation between end-effectors of the tracking and target manipulator and recording the change in this transformation from its initial value at k=0. This error transformation is significant to within the accuracy of the calibration of the Rapid Assembly System, which is less than 1mm. The translational errors are always within 15mm, and rotational errors remain within 6^{o}. The deviations of the four feature coordinates from their initial positions on the image plane are shown in Figures 3 through 6. Deviations from desired X and Y positions never exceed 20 pixels during tracking, with the maximum deviations occurring immediately after the target's direction of motion abruptly changes. The maximum search range of the SSD algorithm presented in [14] is 32 pixels, so the errors in pixel position fall well within the tracking capabilities of the vision system. From the graphs, one can observe that the tracking and feature errors reach their maximum values immediately after the target trajectory changes direction to return to its initial pose. It can be seen that it takes approximately 10 seconds for the destabilizing effects of the change in the target trajectory velocity to be overcome, and the tracking errors to return to their steady-state behavior. The robustness of our algorithms has also been tested by adding artificial noise to the images and the results can be found in [14].

Two interesting observations can be made concerning the system based on the experimental results. First, error in the Z direction, along the optical axis, is relatively large when compared to errors along other directions. This is mainly due to the camera geometry, which makes tracking along the optical axis difficult. Another interesting observation is that an error in yaw appears in Figure 2, even though the target's motion did not have a yaw component. This occurs because of the strong coupling that exists between motion about X (yaw) and along Y. To track an object moving in the Y direction, the camera could move along Y, or it could rotate about X. The same is true for motion along X and about Y (pitch). Numerically, this implies that the matrix $\hat{\boldsymbol{B}}(k)$ is poorly conditioned. Features appropriately positioned on the image plane and an appropriate relative initial position of the camera with respect to the target can reduce the condition number of $\hat{\boldsymbol{B}}(k)$, and decrease errors due to this coupling. Given the camera-lens system used in the experiments, initial target locations in excess of 2m from the camera make the system too poorly conditioned to track reliably. If all feature points are chosen too near the center of the image, $\hat{\boldsymbol{B}}(k)$ can also be too poorly conditioned to allow reliable tracking.

5. Conclusions

Visually tracking objects with full 3D motion by a single camera which also possesses full 3D motion capabilities is an ability many autonomous robotic systems need in order to provide more effective visual input for performing robotic tasks. This ability, however, has proven a difficult one to attain. The reasons that full 3D visual tracking is difficult include the noise characteristics that vision sensors and image processing algorithms possess, the inevitable delays that image processing algorithms introduce into the control loop because of the large amounts of data that must be processed, the nonlinearities inherent in camera-lens and manipulator systems, and the difficulty in inferring full 3D motion from a sequence of 2D images. We have shown that appropriate application of modern control theory combined with recent advances in computer vision theory can solve several cases of the full 3D robotic visual servoing and tracking problem. By properly modeling the eye-in-hand system, by using a unique adaptive control formulation for eye-in-hand parameter estimation, and by incorporating fast and robust computer vision algorithms, a system has been experimentally verified which successfully tracks objects with full 3D motion. Successful tracking occurs despite the strong coupling of control inputs inherent in systems that allow simultaneous translational and rotational tracking. Experimental results have been presented and the limitations of our approach have been discussed. These results demonstrate the robustness of the system to inaccurate initial estimates of system parameters, as well as the ability of the system to track different types of target trajectories. This work also demonstrates that the Controlled Active Vision paradigm [14] can be successfully used to extend the capabilities of eye-in-hand systems where other tracking methods have failed.

6. Acknowledgments

This research was supported by the Defense Advanced Research Projects Agency through ARPA Order Number DAAA-21-89C-0001, by the National Science Foundation through Contract Number IRI-9410003, and by Sandia National Laboratories through Contract Number AL-3021. Nikos Papanikolopoulos was supported by the McKnight Land-Grant Professorship Award program. Brad Nelson was supported by a National Defense Science and Engineering Graduate Fellowship from the U.S. Army Research Office through Grant Number DAAL03-91-G-0272. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the funding agencies.

7. References

[1] P. Anandan, “A Computational Framework and an Algorithm for the Measurement of Visual Motion,” in *International Journal of Computer Vision*, 2(3), pp. 283-310, 1988.

[2] D.H. Ballard and C.M. Brown, *Computer Vision*, Prentice Hall, Inc., Englewood Cliffs, N.J., 1982.

[3] F. Chaumette, P. Rives, and B. Espiau, “Positioning of a Robot with Respect to an Object, Tracking it, and Estimating its Velocity by Visual Servoing,” in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 2248-2253, April, 1991.

[4] E.D. Dickmanns, B. Mysliwetz, and T. Christians, “An Integrated Spatio-Temporal Approach to Automatic Visual Guidance of Autonomous Vehicles,” in *IEEE Trans. on Systems, Man, and Cybernetics*, 20(6), pp. 1273-1284, 1990.

[5] J.T. Feddema and C.S.G. Lee, “Adaptive Image Feature Prediction and Control for Visual Tracking with a Hand-Eye Coordinated Camera,” in *IEEE Trans. on Systems, Man, and Cybernetics*, 20(5), pp. 1172-1183,1990.

[6] J.T. Feddema, C.S.G. Lee, and O.R. Mitchell, “Weighted Selection of Image Features for Resolved Rate Visual Feedback Control,” in *IEEE Trans. on Robotics and Automation*, 7(1), pp. 31-47, 1991.

[7] D. Forsyth, J.L. Mundy, A. Zisserman, C. Coelho, A. Heller, and C. Rothwell, “Invariant Descriptors for 3-D Object Recognition and Pose,” in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(10), pp. 971-991.

[8] G.C. Goodwin and K.S. Sin, *Information and Systems Science Series—Volume 1: Adaptive Filtering, Prediction and Control*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1984.

[9] R.M. Haralick, C. Lee, K. Ottenberg, and M. Nolle, “Analysis and Solutions of the Three Point Perspective Pose Estimation Problem,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 592-598, 1991.

[10] A.J. Koivo and N. Houshangi, “Real-Time Vision Feedback for Servoing of a Robotic Manipulator with Self-Tuning Controller,” in *IEEE Trans. on Systems, Man, and Cybernetics*, 21(1), pp. 134-142,1991.

[11] R. Krishmam, H.J. Sommer III, and P.D. Spidaliere, “Monocular Pose of a Rigid Body Using Point Landmarks”, in *CVGIP: Image Understanding*, 55(2), pp. 307-316, 1992.

[12] F.L. Lewis, *Optimal Control*, John Wiley and Sons, New York, 1986.

[13] P.S. Maybeck, *Stochastic Processes, Estimation, and Control*, Academic Press, London, 1979.

[14] N.P. Papanikolopoulos, “Controlled Active Vision,” Ph.D. Thesis, Department of Electrical and Computer Engineering, Carnegie Mellon University, August, 1992.

[15] N.P. Papanikolopoulos, P.K. Khosla, and T. Kanade, “Vision and Control Techniques for Robotic Visual Tracking,” in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 857-864, April, 1991.

[16] N.P. Papanikolopoulos and P.K. Khosla, “Adaptive Robotic Visual Tracking: Theory and Experiments,” in *IEEE Trans. on Automatic Control*, 38(3), pp.429-445, 1993.

[17] N.P. Papanikolopoulos and P.K. Khosla, “Selection of Features and Evaluation of Visual Measurements for 3-D Robotic Visual Tracking” in *Proc. of the 1993 IEEE International Symposium on Intelligent Contro*l, pp. 320-325, August 25-27, 1993.

[18] C.A. Rothwell, A. Zisserman, C.I. Marinos, D.A. Forsyth, and J.L. Mundy, “Relative Motion and Pose from Arbitrary Plane Curves”, in *Image and Vision Computing*, 10(2), pp. 250-262, 1992.

[19] L.S. Shapiro, H. Wang, and J.M. Brady, “A Matching and Tracking Strategy for Independently Moving Objects,” in *Proc. of the British Machine Vision Conference*, pp. 306-315, 1992.

[20] R. Szeliski and S.B. Kang, “Recovering 3D Shape and Motion from Image Streams Using Nonlinear Least Squares,” in *Journal of Visual Communication and Image Representation*, 5(1), pp. 10-28, 1994.

[21] L.E. Weiss, A.C. Sanderson, and C.P. Neuman, “Dynamic Sensor-Based Control of Robots with Visual Feedback,” in *IEEE Journal of Robotics and Automation* RA-3(5), pp. 404-417, October, 1987.

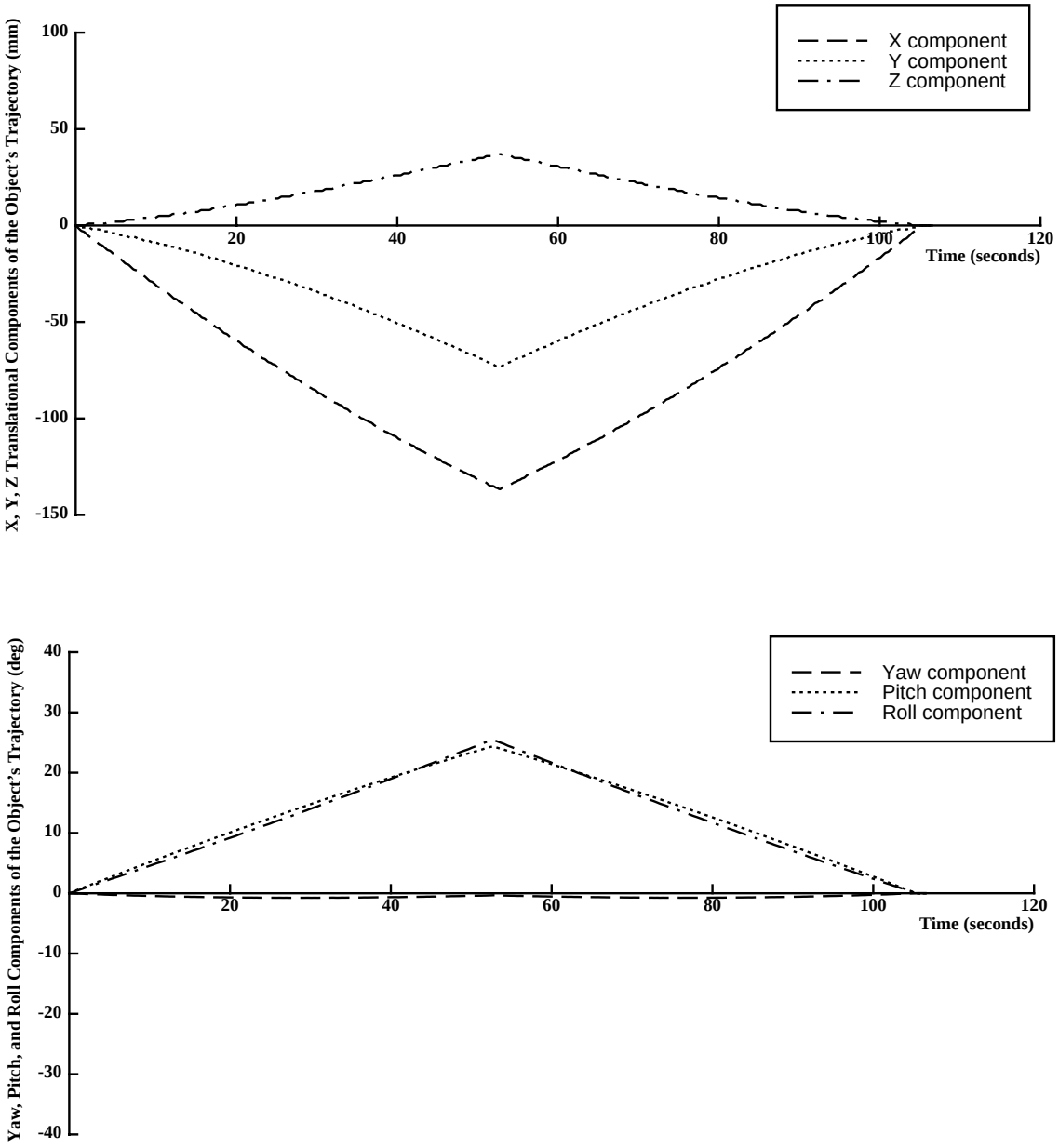

Figure 1. Translational and rotational trajectories of the moving object with respect to its initial pose.

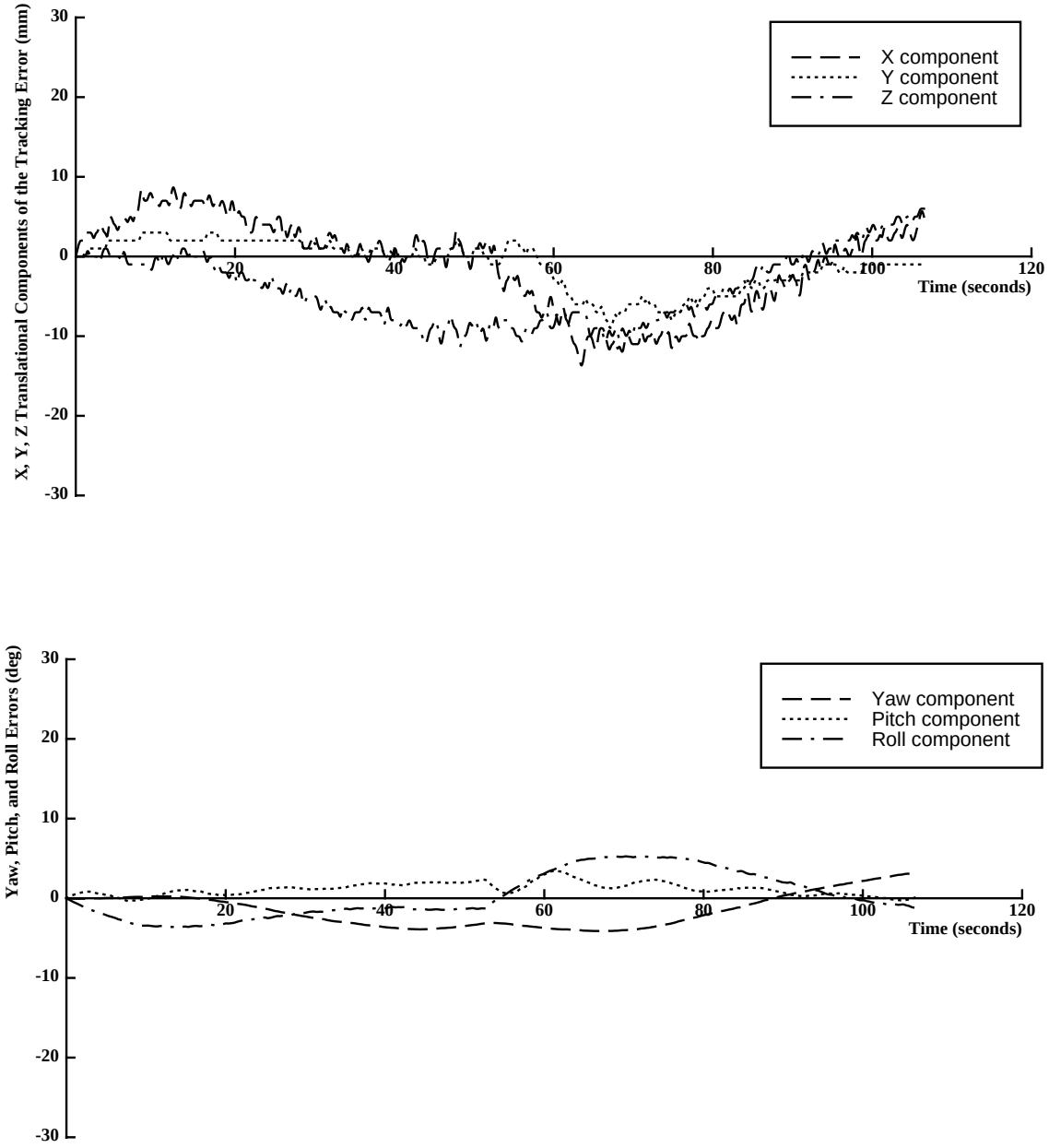

Figure 2. Translational and rotational tracking errors.

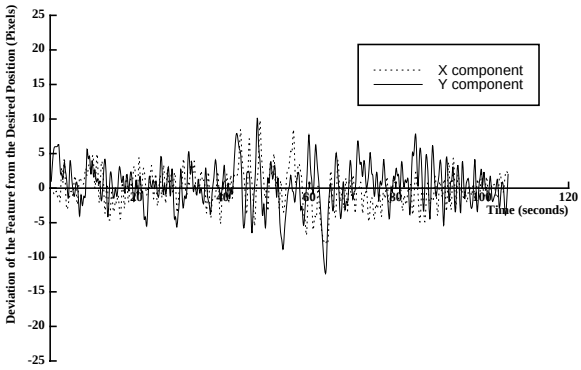

Figure 3. Deviation of feature A from its desired position.

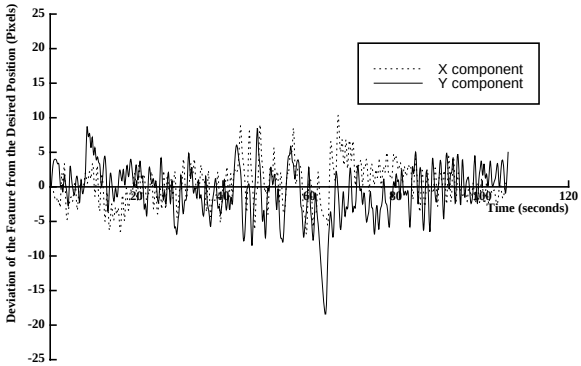

Figure 4. Deviation of feature B from its desired position.

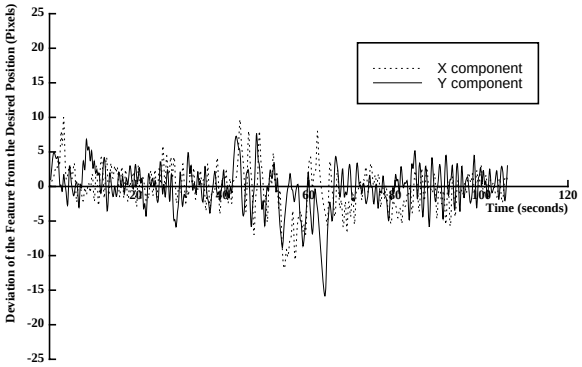

Figure 5. Deviation of feature C from its desired position.

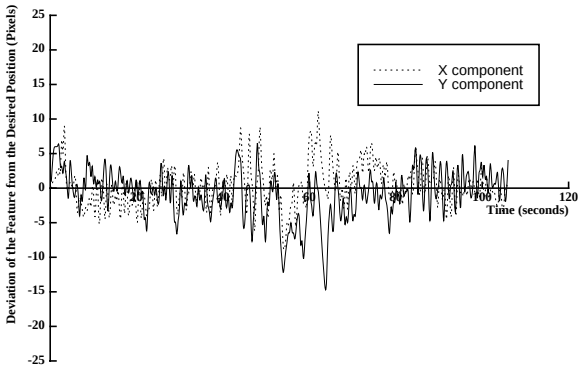

Figure 6. Deviation of feature D from its desired position.