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

Nikolaos P. Papanikolopoulos[*] Brad Nelson[**] Pradeep K. Khosla[**]

[*]Department of Computer Science
University of Minnesota
200 Union St. S.E.
Minneapolis, MN 55455

[**]The Robotics Institute
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213-3891

## Abstract

*Algorithms for 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. This fact makes these algorithms particularly useful in environments that are difficult to calibrate. 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. These experiments were performed using a multi-robotic system consisting of Puma 560 manipulators.*

## 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 [2][6]. 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 extremely difficult to achieve with actual eye-in-hand robotic servoing systems. This paper shows how 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 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 theoretic 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 measurements are based on a pyramidal Sum-of-Squares Differences (SSD) optical flow algorithm that is fast and robust in the presence of noise. A properly formulated adaptive control algorithm 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.

The major differences of our system from similar research efforts [4][5][6] 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 [9][10][11][12] 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 using commercial manipulators. The experiments were performed on the Troikabot, a multi-robotic system of three Puma 560's that operates under the Chimera 3.0 real-time operating system. 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 first describe 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 the Visual Servoing Problem

To model the 3-D visual tracking problem, we assume a pinhole camera model with a coordinate frame at the focal point of the lens. A feature on an object with coordinates $(X_o,Y_o,Z_o)$ in the camera frame projects onto the camera's image plane at $(x,y)$. The manipulator holding the camera provides camera motion by moving the camera frame along its X,Y, and Z axes. The eye-in-hand visual tracking system can be written in state-space form as

$$\begin{aligned}\boldsymbol{x_F}(k+1) &= \boldsymbol{A_F}(k)\boldsymbol{x_F}(k) + \boldsymbol{B_F}(k-d+1)\boldsymbol{u}(k-d+1) + \\ &\quad + \boldsymbol{E_F}(k)\boldsymbol{d_F}(k) + \boldsymbol{H_F}(k)\boldsymbol{v_F}(k)\end{aligned} \qquad (1)$$

where $\boldsymbol{A_F}(k)=\boldsymbol{H_F}(k)=\mathbf{I}_2$, $\boldsymbol{E_F}(k)=T\mathbf{I}_2$, $\boldsymbol{x_F}(k)\in R^2$, $\boldsymbol{d_F}(k)\in R^2$, $\boldsymbol{u}(k)\in R^6$, and $\boldsymbol{v_F}(k)\in R^2$. The matrix $\boldsymbol{B_F}(k)\in R^{2\text{x}6}$ is ($f$ is the focal length and $s_x$, $s_y$ are the camera's pixel dimensions)

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

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 vector representing possible control inputs, $\boldsymbol{d_F}(k)=[u_o(k)\ v_o(k)]^{\mathrm{T}}$ is the exogenous deterministic disturbances vector, and $\boldsymbol{v_F}(k)=[v_x(k)\ v_y(k)]^{\mathrm{T}}$ is a white noise vector, where $v_x(k)$ and $v_y(k)$ are zero-mean, mutually uncorrelated, random variables with variances $\sigma_x^2$ and $\sigma_y^2$, respectively, thus $v_x(k) \sim N(0,\sigma_x^2)$ and $v_y(k) \sim N(0,\sigma_y^2)$. The measurement vector $\boldsymbol{y_F}(k)=[y_x(k)\ \ y_y(k)]^{\mathrm{T}}$ for the feature is given by

$$\boldsymbol{y_F}(k) = \boldsymbol{C_F}\boldsymbol{x_F}(k) + \boldsymbol{w_F}(k) \qquad (1)$$

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 $\boldsymbol{y_F}(k)$ is computed using the SSD algorithm described in [9].

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 [3]. The state space model for $M$ ($M \geq 3$) feature points can be written as

$$\begin{aligned}\boldsymbol{x}(k+1) &= \boldsymbol{A}(k)\boldsymbol{x}(k) + \boldsymbol{B}(k-d+1)\boldsymbol{u}(k-d+1) + \\ &\quad + \boldsymbol{E}(k)\boldsymbol{d}(k) + \boldsymbol{H}(k)\boldsymbol{v}(k)\end{aligned} \qquad (2)$$

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

$$\boldsymbol{B}(k) = \begin{bmatrix} \boldsymbol{B_F}^{(1)}(k) \\ \dots \\ \boldsymbol{B_F}^{(M)}(k) \end{bmatrix} \qquad (5)$$

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

$$\boldsymbol{y}(k) = \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{w}(k) \qquad (4)$$

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 (2) and (4) into a MIMO (Multi-Input Multi-Output) model

$$\begin{aligned}(1-2q^{-1}+q^{-2})\boldsymbol{y}(k) &= \\ &= \boldsymbol{B}(k-d)\boldsymbol{u}(k-d) - \boldsymbol{B}(k-d-1)\boldsymbol{u}(k-d-1) + \boldsymbol{n}(k)\end{aligned} \qquad (5)$$

where $\boldsymbol{n}(k)$ is the white noise vector, and corresponds to the measurement noise, modeling errors, and to 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 motions between successive samples are sufficiently small. This reduces the complexity of (5), 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

$$(1-2q^{-1}+q^{-2})\boldsymbol{y}(k) = \boldsymbol{B}(k-d)\Delta\boldsymbol{u}(k-d) + \boldsymbol{n}(k) \qquad (8)$$

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

$$\Delta\boldsymbol{u}(k-d) = \boldsymbol{u}(k-d) - \boldsymbol{u}(k-d-1) \qquad (9)$$

A single MIMO model or multiple MISO models must be used 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.

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. Adaptive control techniques are 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. These techniques are also effective for recovering the translational and rotational components of the moving object's velocity by using the estimated values of the system's parameters to calculate the object's velocity. A variety of tracking algorithms can be created depending on the parameter estimation schemes and control laws chosen.

## 3.1. Selection of an Efficient Control Law

Tracking the features' projections is realized by an appropriate motion of the robot-camera system described by the model in (8). 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

$$J(k+d) = [\boldsymbol{y}(k+d) - \boldsymbol{y}_D(k+d)]^T \boldsymbol{Q}\, [\boldsymbol{y}(k+d) - \boldsymbol{y}_D(k+d)] + \\ + \boldsymbol{u}^T(k)\boldsymbol{L}\boldsymbol{u}(k) + \Delta\boldsymbol{u}^T(k)\boldsymbol{L}_d\Delta\boldsymbol{u}(k) \qquad (10)$$

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 (10), $[\boldsymbol{y}(k+d) - \boldsymbol{y}_D(k+d)]^T \boldsymbol{Q}\, [\boldsymbol{y}(k+d) - \boldsymbol{y}_D(k+d)]$ 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. Later in this section we will discuss techniques for selecting the elements of these matrices.

The control law is derived from the minimization of the cost function (10) by taking the derivative of $J(k+d)$ with respect to the vector $\boldsymbol{u}(k)$ and combining the result with the system model in (8). The resulting control law is

$$\boldsymbol{u}(k) = -\left(\boldsymbol{B}^T(k)\boldsymbol{Q}\boldsymbol{B}(k) + \boldsymbol{L} + \boldsymbol{L}_d\right)^{-1} \times \\ \times \left[\boldsymbol{B}^T(k)\boldsymbol{Q}\left\{(d+1)\boldsymbol{y}(k) - \boldsymbol{y}_D(k+d) - d\boldsymbol{y}(k-1) - d\boldsymbol{B}(k-d)\boldsymbol{u}(k-d) \right.\right. \\ \left.\left. + \sum_{m=1}^{d-1} \boldsymbol{B}(k-m)\boldsymbol{u}(k-m)\right\} - \boldsymbol{L}_d\boldsymbol{u}(k-1)\right] \qquad (11)$$

An important characteristic of this control law is that we impose constraints on the components of required camera tracking motion $\boldsymbol{u}(k)$. Thus, we directly influence the magnitudes of the control signal and the control signal change, rather than the magnitude of the optical flow.

The expression $\Delta\boldsymbol{u}(k)\boldsymbol{L}_d\Delta\boldsymbol{u}(k)$ in (10) introduces an integral term into the control law (11). This term is desirable since our mathematical model (8) 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 $[\boldsymbol{B}^T(k)\boldsymbol{Q}\boldsymbol{B}(k) + \boldsymbol{L} + \boldsymbol{L}_d]$ is invertible. The matrix $\boldsymbol{B}(k)$ loses rank when the $M$ feature points are collinear [5][11]. An extensive study of other conditions which cause a loss of rank in $\boldsymbol{B}(k)$ can be found in [11].

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 [7]. The control law derived previously (11) did not account for noise or inaccuracies in the camera and depth related parameters contained in $B(k)$. 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 (10) becomes

$$J(k+d) = E\{\, [\boldsymbol{y}(k+d) - \boldsymbol{y}_D(k+d)]^T \boldsymbol{Q}\, [\boldsymbol{y}(k+d) - \boldsymbol{y}_D(k+d)] + \\ + \boldsymbol{u}^T(k)\boldsymbol{L}\boldsymbol{u}(k) + \Delta\boldsymbol{u}^T(k)\boldsymbol{L}_d\Delta\boldsymbol{u}(k) \,|\, F_k \,\} \qquad (12)$$

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

$$\boldsymbol{u}(k) = -\left(\hat{\boldsymbol{B}}^T(k)\boldsymbol{Q}\hat{\boldsymbol{B}}(k) + \boldsymbol{L} + \boldsymbol{L}_d\right)^{-1} \times \\ \times \left[\hat{\boldsymbol{B}}^T(k)\boldsymbol{Q}\left\{(d+1)\boldsymbol{y}(k) - \boldsymbol{y}_D(k+d) - d\boldsymbol{y}(k-1) - d\hat{\boldsymbol{B}}(k-d)\boldsymbol{u}(k-d) \right.\right. \\ \left.\left. + \sum_{m=1}^{d-1} \hat{\boldsymbol{B}}(k-m)\boldsymbol{u}(k-m)\right\} - \boldsymbol{L}_d\boldsymbol{u}(k-1)\right] \qquad (13)$$

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

$$\hat{\boldsymbol{B}}(k) = \begin{bmatrix} \hat{\boldsymbol{B}}_{\boldsymbol{F}}^{(1)}(k) \\ \cdots \\ \hat{\boldsymbol{B}}_{\boldsymbol{F}}^{(M)}(k) \end{bmatrix} \qquad (14)$$

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

$$\hat{\boldsymbol{B}}_{\boldsymbol{F}}^{(j)}(k) = \begin{bmatrix} -\dfrac{f}{\hat{Z}_o^{(j)}(k)s_x} & 0 & \dfrac{x^{(j)}(k)}{\hat{Z}_o^{(j)}(k)} & \dfrac{x^{(j)}(k)y^{(j)}(k)s_y}{f} & -\left(\dfrac{f}{s_x} + \dfrac{x^{(j)}(k)^2 s_x}{f}\right) & \dfrac{y^{(j)}(k)s_y}{s_x} \\ 0 & -\dfrac{f}{\hat{Z}_o^{(j)}(k)s_y} & \dfrac{y^{(j)}(k)}{\hat{Z}_o^{(j)}(k)} & \left(\dfrac{f}{s_y} + \dfrac{y^{(j)}(k)^2 s_y}{f}\right) & -\dfrac{x^{(j)}(k)y^{(j)}(k)s_x}{f} & -\dfrac{x^{(j)}(k)s_x}{s_y} \end{bmatrix} \qquad (15)$$

## 3.2. Estimation of Depth Related Parameters

Several methods can be used to estimate the depth related parameters that appear in $\hat{\boldsymbol{B}}(k)$. For the technique used to produce the experimental results given in Section 4 we define $\left(f/\left(s_x Z_o^{(j)}(k)\right)\right)$ as $\zeta_o^{(j)}(k)$ so that (8) can be rewritten as

$$\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k) = 2\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k-1) - \boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k-2) + \zeta_o^{(j)}(k-d)\boldsymbol{B}_{\boldsymbol{F}_t}^{(j)}(k-d) \\ \Delta\boldsymbol{T}(k-d) + \boldsymbol{B}_{\boldsymbol{F}_r}^{(j)}(k-d)\Delta\boldsymbol{R}(k-d) + \boldsymbol{n}_{\boldsymbol{F}}^{(j)}(k) \qquad (16)$$

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

$$\boldsymbol{B}_{\boldsymbol{F}_t}^{(j)}(k) = T\begin{bmatrix} -1 & 0 & \dfrac{x^{(j)}(k)s_x}{f} \\ 0 & -\dfrac{s_x}{s_y} & \dfrac{y^{(j)}(k)s_x}{f} \end{bmatrix} \qquad (17)$$

$$\boldsymbol{B}_{\boldsymbol{F}_r}^{(j)}(k) = T\begin{bmatrix} \dfrac{x^{(j)}(k)y^{(j)}(k)s_y}{f} & -\left(\dfrac{f}{s_x} + \dfrac{x^{(j)}(k)^2 s_x}{f}\right) & \dfrac{y^{(j)}(k)s_y}{s_x} \\ \left(\dfrac{f}{s_y} + \dfrac{y^{(j)}(k)^2 s_y}{f}\right) & -\dfrac{x^{(j)}(k)y^{(j)}(k)s_x}{f} & -\dfrac{x^{(j)}(k)s_x}{s_y} \end{bmatrix} \qquad (18)$$

and

$$\Delta\boldsymbol{T}(k) = \boldsymbol{T}(k) - \boldsymbol{T}(k-1) \qquad \Delta\boldsymbol{R}(k) = \boldsymbol{R}(k) - \boldsymbol{R}(k-1) \tag{19}$$

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 (16) can be transformed into

$$\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k) = 2\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k-1) - \boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k-2) + \\ + \zeta_o^{(j)}(k-d)\boldsymbol{h}_{\boldsymbol{t}}^{(j)}(k-d) + \boldsymbol{h}_{\boldsymbol{r}}^{(j)}(k-d) + \boldsymbol{n}_{\boldsymbol{F}}^{(j)}(k) \tag{20}$$

For the final transformation of (20) we first define the vector $\Delta\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k)$ to be

$$\Delta\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k) = \boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k) - 2\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k-1) + \boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k-2) - \boldsymbol{h}_{\boldsymbol{r}}^{(j)}(k-d) \tag{21}$$

Equation (20) can now be rewritten in a convenient form for estimating the depth related parameter

$$\Delta\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k) = \zeta_o^{(j)}(k-d)\boldsymbol{h}_{\boldsymbol{t}}^{(j)}(k-d) + \boldsymbol{n}_{\boldsymbol{F}}^{(j)}(k) \tag{22}$$

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.

The depth related parameter can be updated by standard recursive estimation equations. The parameter update equation used is

$${}^{-}\hat{\zeta}_o^{(j)}(k) = {}^{+}\hat{\zeta}_o^{(j)}(k-1) \tag{23}$$

where the superscript ($^-$) denotes the predicted value, the superscript ($^+$) denotes the updated value. A more accurate form for the updated parameter equation (23) can be obtained. We propose a modified scheme based on the equation

$$Z_o^{(j)}(k+1) \approx Z_o^{(j)}(k) + \Delta Z_{obj}^{(j)}(k) + q^{-d+1}\Delta Z_{cam}^{(j)}(k) \tag{24}$$

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

$$\Delta Z_{cam}^{(j)}(k) = -\left\{\dot{z}_c(k) + \left[\omega_x(k)y^{(j)}(k)s_y - \omega_y(k)x^{(j)}(k)s_x\right]\frac{Z_o^{(j)}(k)}{f}\right\}T \tag{25}$$

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. Equation (24) 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 (24) as

$$Z_o^{(j)}(k) \approx 2Z_o^{(j)}(k-1) - Z_o^{(j)}(k-2) + \Delta Z_{cam}^{(j)}(k-d) - \Delta Z_{cam}^{(j)}(k-d-1) \tag{26}$$

Through algebraic manipulation, (26) becomes

$$\zeta_o^{(j)}(k) = \frac{\zeta_o^{(j)}(k-1)}{2 - \dfrac{\zeta_o^{(j)}(k-1)}{\zeta_o^{(j)}(k-2)} + \zeta_o^{(j)}(k-1)\dfrac{s_x}{f}\left[\Delta' Z_{cam}^{(j)}(k-d) - \Delta' Z_{cam}^{(j)}(k-d-1)\right]} \tag{27}$$

where

$$\Delta' Z_{cam}^{(j)}(k) = -\left\{\dot{z}_c(k) + \left[\omega_x(k)y^{(j)}(k)s_y - \omega_y(k)\left(x^{(j)}(k)s_x\right)\right]\frac{1}{s_x\zeta_o^{(j)}(k)}\right\}T \tag{28}$$

Substituting the estimated values of $\zeta_o^{(j)}(k)$ into (27), the parameter update equation given by (23) becomes

$${}^{-}\zeta_o^{(j)}(k) = \frac{{}^{+}\zeta_o^{(j)}(k-1)}{2 - \dfrac{{}^{+}\zeta_o^{(j)}(k-1)}{{}^{+}\zeta_o^{(j)}(k-2)} + {}^{+}\zeta_o^{(j)}(k-1)\dfrac{s_x}{f}\left[{}^{+}\Delta' Z_{cam}^{(j)}(k-d) - {}^{+}\Delta' Z_{cam}^{(j)}(k-d-1)\right]} \tag{29}$$

The term ${}^{+}\Delta' Z_{cam}^{(j)}(k)$ is derived from $\Delta' Z_{cam}^{(j)}(k)$ in (20) by substituting $\zeta_o^{(j)}(k)$ with ${}^{+}\zeta_o^{(j)}(k)$. 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

$$p^{(j)}(k) = E\left\{\left[\zeta_o^{(j)}(k) - \hat{\zeta}_o^{(j)}(k)\right]^2\right\} \tag{30}$$

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 [8]

$${}^{-}\hat{\zeta}_o^{(j)}(k) = \frac{{}^{+}\hat{\zeta}_o^{(j)}(k-1)}{2 - \dfrac{{}^{+}\hat{\zeta}_o^{(j)}(k-1)}{{}^{+}\hat{\zeta}_o^{(j)}(k-2)} + {}^{+}\hat{\zeta}_o^{(j)}(k-1)\dfrac{s_x}{f}\left[{}^{+}\Delta' Z_{cam}^{(j)}(k-d) - {}^{+}\Delta' Z_{cam}^{(j)}(k-d-1)\right]} \tag{31}$$

$${}^{-}p^{(j)}(k) = {}^{+}p^{(j)}(k-1) + s^{(j)}(k-1) \tag{32}$$

$${}^{+}p^{(j)}(k) = \left[\{{}^{-}p^{(j)}(k)\}^{-1} + \boldsymbol{h}_{\boldsymbol{t}}^{(j)T}(k-d)\{\boldsymbol{N}^{(j)}(k)\}^{-1}\boldsymbol{h}_{\boldsymbol{t}}^{(j)}(k-d)\right]^{-1} \tag{33}$$

$$\boldsymbol{k}^T(k) = {}^{+}p^{(j)}(k)\boldsymbol{h}_{\boldsymbol{t}}^{(j)T}(k-d)\{\boldsymbol{N}^{(j)}(k)\}^{-1} \tag{34}$$

$${}^{+}\hat{\zeta}_o^{(j)}(k) = {}^{-}\hat{\zeta}_o^{(j)}(k) + \boldsymbol{k}^T(k)\left[\Delta\boldsymbol{y}_{\boldsymbol{F}}^{(j)}(k) - {}^{-}\hat{\zeta}_o^{(j)}(k)\boldsymbol{h}_{\boldsymbol{t}}^{(j)}(k-d)\right] \tag{35}$$

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 (31)-(35) 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. Experimental Results

The results of one experiment using the control and estimation schemes described previously 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. The objects that are used for tracking include books, blocks, and general items with distinct features.

For the experiments performed, the target's initial depth is

290mm from the camera frame. The 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 four features 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 [9]. If a feature does not satisfy a confidence threshold, the user is asked to select a replacement object feature.

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 (31)-(35). 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 within the accuracy of the calibration of the Troikabot, 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 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. The graphs show that approximately 10 seconds are needed for the destabilizing effects of the change in the target trajectory velocity to be overcome.

Two interesting observations can be made concerning the system based on the experimental results. First, error in Z, along the optical axis, is relatively large when compared to errors along other directions due to the camera geometry. 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, 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. We have shown that appropriate application of modern control theory combined with recent advances in computer vision theory can solve the full 3D 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. Experimental results demonstrate the ability of the system to successfully track objects moving with six degrees of freedom. This work also demonstrates that the Controlled Active Vision paradigm [10] 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 U.S. Army Research Office through Grant Number DAAL03-91-G-0272 and by Sandia National Laboratories through Contract Number AC-3752D.

## 7. References

[1] P. Anandan, "Measuring Visual Motion from Image Sequences," Technical Report COINS-TR-87-21, COINS Department, University of Massachusetts, 1987.

[2] P.I. Corke, "Video-Rate Visual Servoing for Robots," in *Lecture Notes in Control and Information Science*, eds. V. Hayward and O. Khatib, pp. 429-451, Springer-Verlag, 1989.

[3] P.A. Couvignou, N.P. Papanikolopoulos, and P.K. Khosla, "Hand-eye Robotic Visual Servoing Around Moving Objects Using Active Deformable Models" in *Proc. of the 1992 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS-92)*, pp. 1855-1862, July 7-10, 1992.

[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] 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.

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

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

[9] 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.

[10] N.P. Papanikolopoulos, P.K. Khosla, and T. Kanade, "Adaptive Robotic Visual Tracking," in *Proc. of the American Control Confer-*

*ence*, pp. 962-967, June, 1991.
[11] N.P. Papanikolopoulos and P.K. Khosla, “Robotic Visual Servoing Around a Static Target: An Example of Controlled Active Vision,” in *Proc. of the 1992 American Control Conference*, pp. 1489-1494, 1992.
[12] 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.

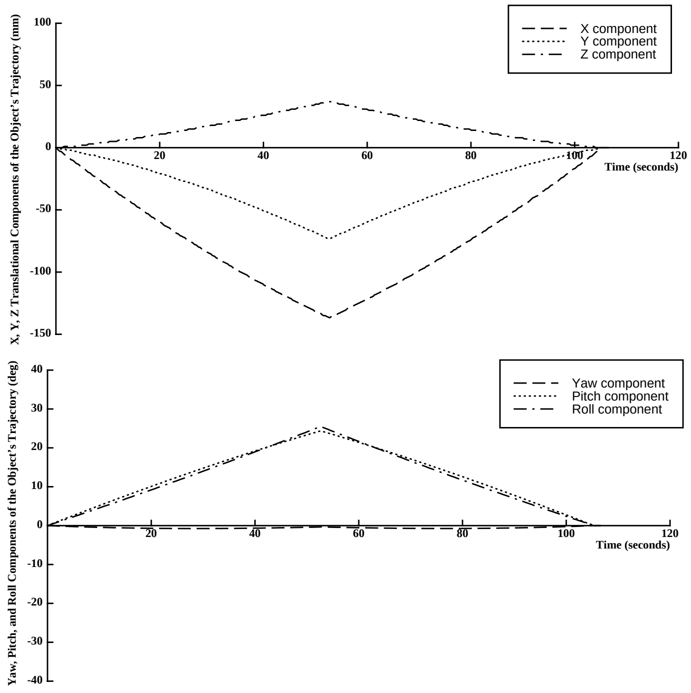

Figure 1. Translational and rotational trajectories of the moving object with respect to its initial pose.

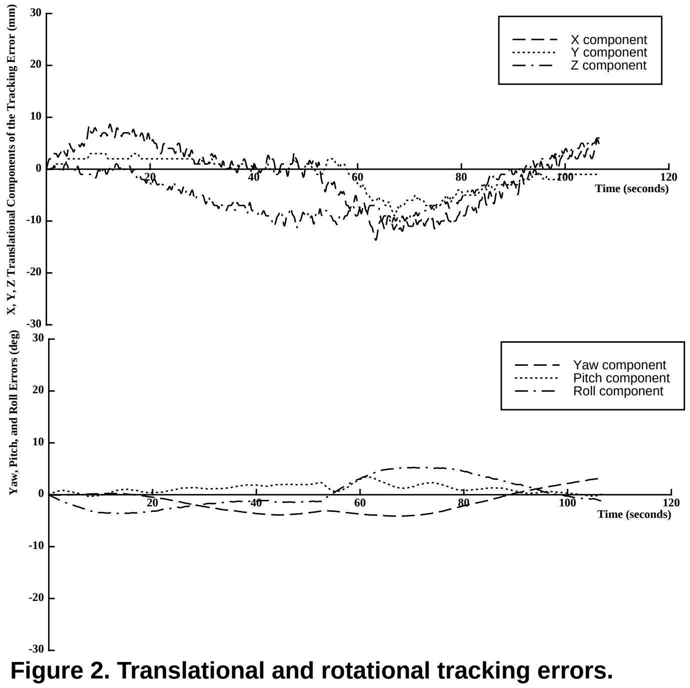

Figure 2. Translational and rotational tracking errors.

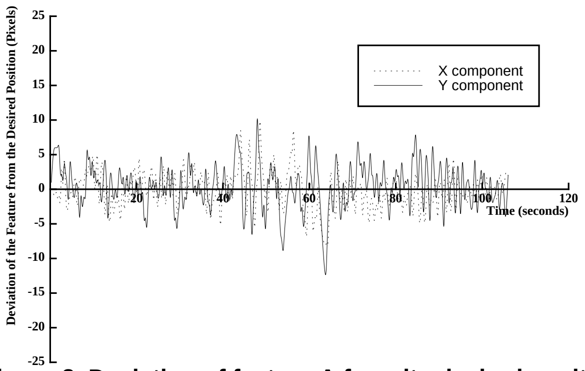

Figure 3. Deviation of feature A from its desired position.

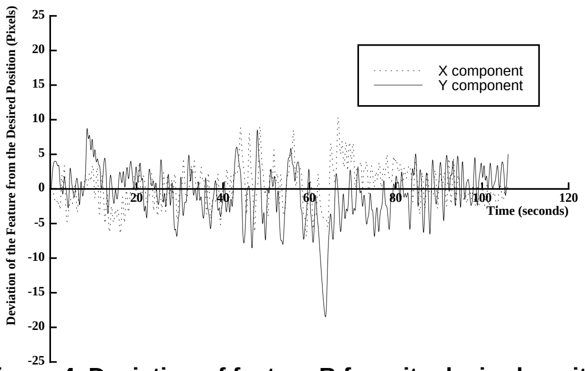

Figure 4. Deviation of feature B from its desired position.

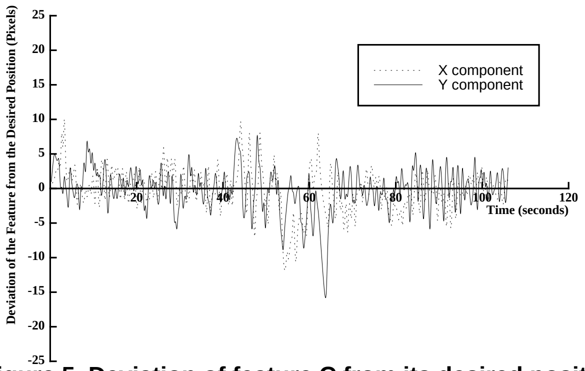

Figure 5. Deviation of feature C from its desired position.

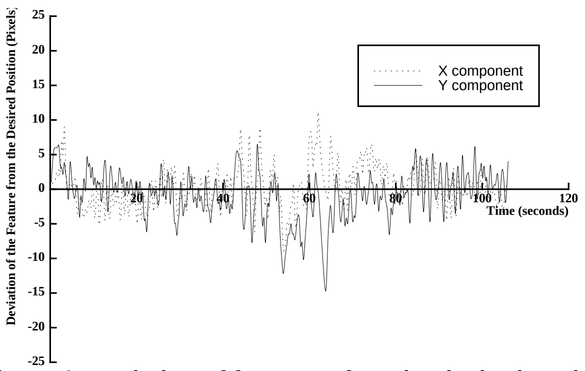

Figure 6. Deviation of feature D from its desired position.