Presented at the European Robotics and and Intelligent Systems Conference, EURISCON '91, June 23-28, 1991, Corfu, GREECE

REAL-TIME LQG ROBOTIC VISUAL TRACKING *

N. P. Papanikolopoulos and P. K. Khosla

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

Abstract

In this paper, a modified LQG technique is proposed for the solution of the robotic visual tracking problem (eye-in-hand configuration). The problem of robotic visual tracking is formulated as a problem of combining control with computer vision. A cross-correlation method provides the object's motion measurements which are used *to* update the system's measurement vector. These measurements arc fed to a discrete steady state Kalman filter that calculates the estimated values of the system's states and of the exogenous disturbances. Then, a discrete LQG controller computes the desired motion of the robotic system. Experimental results are presented to show the effectiveness of the approach.

1. Introduction

One of the most desirable characteristics of a robotic manipulator is its flexibility. Flexible robots can quickly adapt to the evolving requirements of an unknown task, and can properly react to sudden changes in the environment. Flexibility and adaptability can be achieved by incorporating vision and generally, sensory information in the feedback loop. This sensory information enhances the robot's capability by continuously updating the robot's view (or model) of the world and the task. The completeness and accuracy of this view depends on the existence of a framework for the integration of sensory information with the other components of a robotic system.

Research in computer vision has traditionally emphasized the paradigm of image understanding which focuses on the static analysis of one image. Recently, more emphasis has been given to the dynamic analysis of a sequence of images [1]. This sequence of images is produced either by a moving camera that captures views of a static environment, or by a static camera that captures views of a moving object. A characteristic example of the second category is the motion analysis area where vision information is used for tracking moving objects [2, 3, 4]. Little research [5, 6] has been conducted in using vision information in the dynamic feedback loop (moving camera and target). Particularly, in the motion analysis research, Roach and Aggarwal

*This research was supported by the Defense Advanced Research Projects Agency, through ARPA Order Number DAAA-21-89C-0001 and by Innovision Inc.

[7] have presented a scheme for tracking rigid convex polyhedra. Their scheme was based on image segmentation which is time-consuming. A stereo system for tracking known 3-D targets was presented by Gennery [4]. Wallace and Mitchell [3] have used complex Fourier series for obtaining a solution to the same problem. Hunt and Sanderson [8] have presented algorithms for visual tracking based on mathematical prediction of the position of the object's centroid. Lee and Wohn [9] have used image differencing techniques to track a moving object.

This paper addresses some of the issues associated with the use of vision information in the dynamic feedback loop. In particular, we deal with the problem of robotic visual tracking of a moving target (eye-in-hand configuration). To achieve our tracking objective, we combine computer vision techniques, for detection of motion, with a simple LQG control strategy. A cross-correlation technique (SSD optical flow) is used for computing the vector of discrete displacements in real-time. The paper is organized as follows: In Section 2, we review the definition of the optical flow and present methods for computation of the vector of discrete displacements. The mathematical formulation of the visual tracking problem is described in Section 3. The LQG controller in conjunction with the Cartesian robotic control schemes are discussed in Section 4. The hardware configuration of our experimental testbed (DD Arm II) and some experimental results are presented in Section 5. Finally, in Section 6, the paper is summarized.

2. Optical How

This Section will present an outline of our vision techniques in order to illustrate their characteristics (noise, computational complexity, quantization errors). We assume a pinhole camera model with a frame R_s attached to it. In addition, we assume a perspective projection and the focal length to be unity. A point **P** with coordinates (X_s, Y_s, Z_s) in R_s projects onto a point **p** in the image plane with image coordinates (x, y) given by:

$$x = X_s / Z_s \quad and \quad y = Y_s / Z_s. \tag{1}$$

Let us assume that the camera moves in a static environment with a translational velocity $\mathbf{T} = (T_x, T_y, T_z)^T$ and with an angular velocity $\mathbf{R} = (R_x, R_y, R_z)^T$ with respect to the camera frame R_s. The optical flow equations are given by:

$$u = [\, x\frac{T_z}{Z_s} - \frac{T_x}{Z_s}\,] + [\, x y R_x - (1 + x^2) R_y + y R_z \,] \tag{2}$$

$$v = [\, y\frac{T_z}{Z_s} - \frac{T_y}{Z_s}\,] + [\, (1 + y^2) R_x - x y R_y - x R_z \,] \tag{3}$$

where $u = \dot{x}$ and $v = \dot{y}$. Values u and v are also known as the optical flow measurements. Instead of a static object and a moving camera, if we were to assume a static camera and a moving object then we would obtain the same result as in equations (2) and (3) except for a sign reversal. The computation of u and v has been the focus of much research and many algorithms have been proposed [10, 11]. For accuracy reasons, we use a modified version of the matching-based technique [11] also known as the Sum-of-Squared Differences (SSD) optical flow. For every point $\mathbf{p}_A = (x_A, y_A)$ in image A, we want to find the point $\mathbf{p}_B = (x_A + u, y_A + v)$ to which the point $\mathbf{p}_A$ moves in image B. It is assumed that the intensity values in the neighborhood L of $\mathbf{p}_A$ remain almost constant over time, that the point $\mathbf{p}_B$ is within an area S of $\mathbf{p}_A$, and that velocities are normalized by time T to get the displacements. Thus, for the point $\mathbf{p}_A$ the SSD algorithm selects the displacement $\mathbf{d} = (u, v)$ that minimizes the SSD measure:

$$e(\mathbf{p}_A, \mathbf{d}) = \sum_{m,n \in N} [I_A(x_A + m, y_A + n) - I_B(x_A + m + u, y_A + n + v)]^2 \tag{4}$$

where $u, v \in S$, N is an area around the pixel we are interested in, and I_A, I_B are the intensity functions in images **A** and B, respectively. The different values of the **SSD** measure create a surface called the *SSD* **surface.** The accuracy of the measurements of the displacement vector can be improved by using multiple windows. The selection of the best measurements is based on the confidence measure of each window. Efficient confidence measures for the selection of the most accurate measurements are described in [12]. The next step of our algorithm involves the use of these measurements in the visual tracking process. These measurements should be transformed into control commands to the robotic system. Thus, a mathematical model for this transformation must be developed. In the next Section, we present a mathematical model for the visual tracking problem.

3. Mathematical Model For The 2-D Visual Tracking Of An Object

Consider a target that moves in a plane with a feature, located at a point **P,** that we want to track. The projection of this point on the image plane is the point **p.** Consider also a neighborhood S_w of **p** in the image plane. The problem of **2-D** visual tracking of a single feature point can be defined as: "find the camera translation (T_x, T_y) with respect to the camera frame that keeps S_w stationary in an area S_o around the origin of the image frame". It is assumed that at initialization of the tracking process, the area S_w is brought to the origin of the image frame, and that the plane of motion is vertical to the optical axis of the camera. The problem of visual tracking of a single feature point can also be defined as [2]: "find the camera rotation (R_x, R_y) with respect to the camera frame that keeps S_w stationary in an area S_o around the origin of the image frame". Assume that the optical flow of the point **p** at the instant of time kT is $(u(kT), v(kT))$ where T is the **time** between two consecutive frames. It can be shown that at time $(k+1)T$, the optical flow **is:**

$$u((k+1)T) \approx u(kT) + u_c(kT) \quad , \quad v((k+1)T) \approx v(kT) + v_c(kT) \tag{5}$$

where $u_c(kT), v_c(kT)$ are the components **of** the optical flow induced by the tracking motion of the camera. Equations (5) are **based** on the assumption that the optical flow induced by motion of the feature does not change in the time interval T. Therefore, T should be as **small** as possible. To keep the notation simple and without any **loss** of generality, equations (5) will be used with k and $(k+1)$ instead of kT and $(k+1)T$ respectively. If the camera tracks the feature point with translation $T_x(k)$ and $T_y(k)$ with respect to the camera frame, then the optical flow that is generated by the motion of the camera with $T_x(k)$ and $T_y(k)$ is:

$$u_c(k) = -\frac{T_x(k)}{Z_s} \, , \; v_c(k) = -\frac{T_y(k)}{Z_s} . \tag{6}$$

We assume that for 2-D visual tracking the depth Z_s remains constant. The same model can be used for keeping the feature point stationary in an area S_r different from the origin.

Consider a target that moves in a plane which **is** vertical to the optical axis of the camera. The projection of the target on the image plane is the area S_w in the image plane. The problem of **2-D** visual tracking of a single object can be defined as: "find the camera translation (T_x, T_y) and rotation (R_z) with respect to the camera frame that keeps S_w stationary". It is assumed that the target rotates around an axis Z which at $k=0$ coincides with the optical axis of the camera. The mathematical model of this problem in state-space form is (a formal derivation is given in [13]):

$$\mathbf{x}(k+1) = \mathbf{A}\,\mathbf{x}(k) + \mathbf{B}\,\mathbf{u}_c(k) + \mathbf{E}\,\mathbf{d}(k) + \mathbf{H}\,\mathbf{v}(k) \tag{7}$$

where $\mathbf{A} = \mathbf{H} = \mathbf{I}_3$**, $\mathbf{B} = \mathbf{E} = T\,\mathbf{I}_3$, $\mathbf{x}(k) \in R^3$, $\mathbf{u}_c(k) \in R^3$, $\mathbf{d}(k) \in R^3$, and $\mathbf{v}(k) \in R^3$. The vector

---

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

$x(k)=(x(k), y(k), \theta(k))^T$ is the state vector, $\mathbf{u}_c(k)=(u_c(k), v_c(k), R_z(k))^T$ is the control input vector, $\mathbf{d}(k)=(u(k), v(k), \omega(k))^T$ is the exogenous disturbances vector, and $\mathbf{v}(k)=(v_1(k), v_2(k), v_3(k))^T$ is the white noise vector. $x(k)$, $y(k)$, $\theta(k)$ are now the X, Y and roll component of the tracking error, respectively. The measurement vector $\mathbf{y}(k)=(y_1(k), y_2(k), y_3(k))^T$ is given by:

$$\mathbf{y}(k)=\boldsymbol{C}x(k)+\mathbf{w}(k) \tag{8}$$

where $\mathbf{w}(k)=(w_1(k), w_2(k), w_3(k))^T$ is a white noise vector ($\mathbf{w}(k)\sim N(0,\mathbf{W})$) and $\boldsymbol{C}=\mathbf{I}_3$. The measurement vector is obtained by using the **SSD** algorithm. First, the tracking error of the projections of the two different feature points on the image plane is computed. Then, an algebraic system of four equations (two tracking error equations per point) is formulated. The solution of the system is the X, **Y** and roll component of the tracking error. If the projections of the two feature points on the image plane are not the same, it is guaranteed that the system of equations has a solution. It is assumed that each one of these features at time $t=0$ is located at its desired position. The LQG control strategy that keeps the target stationary is discussed in detail in the next Section.

4. LQG Controller

A useful control technique for this type of problem is the **LQG** (Linear Quadratic Gaussian) control scheme. Neglecting for the time being the white noise terms of our system, we will consider the more general problem of determining the matrices $\boldsymbol{G}$ and $\boldsymbol{G}_d$ in the linear control law:

$$\mathbf{u}_c(k)=-\boldsymbol{G}\mathbf{x}(k)-\mathbf{G}_d\mathbf{d}(k). \tag{9}$$

The reason it is necessary to separate the exogenous variables from the process state vector $\mathbf{x}(k)$, rather than deal directly with the metastate vector $\mathbf{x}_M^T(k)=(\mathbf{x}^T(k), \mathbf{d}^T(k))$, is that in developing the theory for the design of the gain matrix we assume that the underlying process is controllable. If we try to create a new system with metastate vector $\mathbf{x}_M^T(k)=(\mathbf{x}^T(k), \mathbf{d}^T(k))$, we can show, by using the controllability matrix, that the new system is uncontrollable. Thus, we should work with the form of the system (Eq. (7) and (8)) that was developed in Section **3.** One can observe that the matrices **E** and **B** are equal, so equations (7) and (8) can be rewritten as:

$$x(k+1)=\boldsymbol{A}x(k)+\mathbf{B}u_o(k)+\mathbf{H}\mathbf{v}(k) \tag{10}$$

$$\mathbf{y}(k)=\boldsymbol{C}\mathbf{x}(k)+\mathbf{w}(k) \tag{11}$$

where $\mathbf{u}_o(k)=\mathbf{u}_c(k)+\mathbf{d}(k)$. **A** performance criterion that can be minimized for the selection of the optimum gain **matrix** $\boldsymbol{G}$ is:

$$J=\sum_{k=0}^{\infty}[\mathbf{x}^T(k)\boldsymbol{Q}\mathbf{x}(k)+\mathbf{u}_o^T(k)\mathbf{R}\mathbf{u}_o(k)] \tag{12}$$

where $Q=Q^T\geq 0$ where $\mathbf{R}=R^T>0$. The performance criterion contains a quadratic form in the state vector $\mathbf{x}(k)$ plus a second quadratic form in the vector $\mathbf{u}_o(k)$. Physically, the first quadratic form represents a penalty **for** the tracking error and the second corresponds to a modified *cost* of control. The performance criterion is minimized by selecting an appropriate gain matrix **G.** Taking into consideration the white noise terms of our system, the controller becomes an LQG controller and the control law is:

$$\mathbf{u}_o(k)=-\boldsymbol{G}\hat{\mathbf{x}}(k) \tag{13}$$

where $\hat{\mathbf{x}}(k)$ is the estimated value of the state vector $\mathbf{x}(k)$. Thus, $\mathbf{u}_c(k)$ is given by:

$$\mathbf{u}_c(k)=-\mathbf{G}\hat{\mathbf{x}}(k)-\hat{\mathbf{d}}(k) \tag{14}$$

whew $\hat{\mathbf{d}}(k)$ is the estimated value of the disturbance vector $\mathbf{d}(k)$. The performance criterion now is the expected value of J. The optimal control gain matrix $\mathbf{G}$ is $\mathbf{G}=(\mathbf{B}^T\mathbf{P}\,\mathbf{B}+\mathbf{R})^{-1}\mathbf{B}^T\mathbf{P}\,\mathbf{A}$ with $\mathbf{P}$ being the unique symmetric positive definite solution of the matrix algebraic Ricatti equation:

$$\mathbf{A}^T[\mathbf{P}-\mathbf{P}\,\mathbf{B}\,(\mathbf{B}^T\mathbf{P}\mathbf{B}+\mathbf{R})^{-1}\mathbf{B}^T\mathbf{P}]\,\mathbf{A}+\mathbf{Q}=\mathbf{P}. \tag{15}$$

The design parameters are the elements of the matrices $\mathbf{Q},\mathbf{R}$. By selecting these, one can obtain the desired gain matrix $\mathbf{G}$. There is no standard procedure for the selection of the elements of these matrices. One technique [14] is the optimization approach. The next step in our algorithm is the computation of the vectors $\hat{\mathbf{x}}(k)$ and $\hat{\mathbf{d}}(k)$. We design an observer for the estimation of the metastate vector $\mathbf{x}_M(k)$. The state-space model of equations (7)-(8) can be rewritten as:

$$\mathbf{x}_M(k+1)=\mathbf{A}_M\ \mathbf{x}_M(k)+\mathbf{B}_M\mathbf{u}_c(k)+\mathbf{H}_M\mathbf{v}_M(k) \tag{16}$$

$$\mathbf{w} \tag{17}$$

$$\mathbf{A}_M=\begin{bmatrix}1&0&0&T&0&0\\0&1&0&0&T&0\\0&0&1&0&0&T\\0&0&0&1&0&0\\0&0&0&0&1&0\\0&0&0&0&0&1\end{bmatrix},\quad \mathbf{B}_M=\begin{bmatrix}T&0&0\\0&T&0\\0&0&T\\0&0&0\\0&0&0\\0&0&0\end{bmatrix}.$$

and

$$\mathbf{C}_M=\begin{bmatrix}1&0&0&0&0&0\\0&1&0&0&0&0\\0&0&1&0&0&0\end{bmatrix},\quad \mathbf{H}_M^T=\begin{bmatrix}0&0&0&T&0&0\\0&0&0&0&T&0\\0&0&0&0&0&T\end{bmatrix}.$$

As it was mentioned before, the measurement vector consists of the measured translational components of the tracking error $x(k)$, $y(k)$ and of the roll component of it, $\theta(k)$. A steady state Kalman filter [15] can be designed for the estimation of the metastate vector $\mathbf{x}_M(k)$. The assumptions are that $\mathbf{Q}_e=E[\mathbf{v}_M(k)\mathbf{v}_M^T(k)]$, $\mathbf{R}_e=E[\mathbf{w}(k)\mathbf{w}^T(k)]$, $\mathbf{x}_{Mo}=E[\mathbf{x}_M(0)]$ and $E[\mathbf{v}_M(k)\mathbf{w}^T(j)]=0$ for all k,j. The state update equation is:

$$\hat{\mathbf{x}}_M(k+1)=\mathbf{A}_M\hat{\mathbf{x}}_M(k)+\mathbf{B}_M\ \mathbf{u}_c(k)+\mathbf{K}_e(\mathbf{y}(k)-\mathbf{C}_M\hat{\mathbf{x}}(k)) \tag{18}$$

where

$$\mathbf{K}_e=\mathbf{A}_M\ \mathbf{P}_e\mathbf{C}_M^T[\mathbf{C}_M\mathbf{P}_e\mathbf{C}_M^T+\mathbf{R}_e]^{-1} \tag{19}$$

and $\mathbf{P}_e$ satisfies the matrix algebraic Ricatti equation

$$\mathbf{A}_M\ [\mathbf{I}-\mathbf{P}_e\mathbf{C}_M^T(\mathbf{C}_M\mathbf{P}_e\mathbf{C}_M^T+\mathbf{R}_e)^{-1}\mathbf{C}_M]\,\mathbf{P}_e\mathbf{A}^T{}_M+\mathbf{H}_M\mathbf{Q}_e\mathbf{H}_M^T=\mathbf{P}_e\ . \tag{20}$$

The time-invariant steady state Kalman filter can be implemented easily and does not require a large number of calculations. In addition to the steady state Kalman filter, we use the time-varying discrete Kalman filter which constantly updates the Kalman gain matrix $\mathbf{K}_e$. This improves the performance of our observer but it is computationally more expensive than the time-invariant Kalman filter. The state update equation of the new state observer is the same as (18) while the other equations are:

$$\mathbf{K}_e(k)=\mathbf{A}_M\mathbf{P}_e(k)\mathbf{C}_M^T[\mathbf{C}_M\mathbf{P}_e(k)\mathbf{C}_M^T+\mathbf{R}_e]^{-1} \tag{21}$$

$$\mathbf{P}_e(k+1)=[\mathbf{A}_M-\mathbf{K}_e(k)\mathbf{C}_M]\,\mathbf{P}_e(k)\,\mathbf{A}^T{}_M+\mathbf{H}_M\mathbf{Q}_e\mathbf{H}_M^T \tag{22}$$

where $\hat{\mathbf{x}}_{Mo} = E[\mathbf{x}_M(0)]$ and $\mathbf{P}_e(0) = E[(\mathbf{x}_M(0) - \hat{\mathbf{x}}_{Mo})(\mathbf{x}_M(0) - \hat{\mathbf{x}}_{Mo})^T]$. The performance of the observer depends on the selection of the $\mathbf{Q}_e$ and $\mathbf{R}_e$ matrices. We should mention that the white noise model is only an approximation to the actual noise model of the camera. Thus, the selection of the **Q,** and **R,** matrices is done empirically and a search for the best set of noise variances is conducted. The initialization of the vectors $\hat{\mathbf{x}}(k)$ and $\hat{\mathbf{d}}(k)$ is given by:

$$\hat{\mathbf{x}}(1) = \mathbf{y}(1) \;, \hat{\mathbf{d}}(1) = T^{-1}(\mathbf{y}(1) - \mathbf{0}) . \tag{23}$$

The next step of our algorithm is the calculation of the triple $(T_x(k), T_y(k), R_z(k))$. The calculation of the $T_x(k)$ and $T_y(k)$ is done by using equations (6) which require the knowledge of the depth Z_s. *R, (k)* is given directly as the computed control signal. The knowledge of the depth Z_s can be acquired in two ways. The first way is direct computation by a range sensor or by stereo techniques [1]. The use of stereo for the recovery of the depth is a difficult procedure because it requires the solution of the correspondence problem. A more effective strategy that requires the **use** of only one visual sensor is to use adaptive control techniques. The control law is based on the estimated on-line values of the model's parameters that depend on the depth. More details about our adaptive control schemes can be found in [16]. After the computation of the translational $\mathbf{T}(k)$ and rotational $\mathbf{R}(k)$ velocity vectors with respect to the camera frame R_s, we transform them to the end-effector frame R, with the **use** of the transformation ^eT_s. The transformed signals are fed to the robot controller. We experimented with a Cartesian **PD** robot control scheme with gravity compensation. The mathematical model of the robot's dynamics is:

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \tau \tag{24}$$

where **q** is the vector of the joint variables of the robotic arm, **D** is the inertial acceleration related matrix, **c** is the nonlinear Coriolis and centrifugal torque vector, **g** is the gravitational torque vector and τ is the generalized torque vector. The model is nonlinear and coupled. The PD control scheme assumes that all velocities in the dynamics equations are zero. This implies that $\dot{\mathbf{q}} = \dot{\mathbf{J}} = \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) = 0$ ($\mathbf{J}(\mathbf{q})$ is the manipulator's Jacobian).Thus, the actuators' torque vector τ is given by:

$$\tau = \mathbf{J}^T(\mathbf{q})\mathbf{F} + \mathbf{g}(\mathbf{q}) \tag{25}$$

$$\mathbf{F} = \mathbf{K_p}\Delta\mathbf{x} + \mathbf{K_v}[\dot{\mathbf{x}}_\mathbf{d} - \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}\,] + \ddot{\mathbf{x}}_\mathbf{d} \tag{26}$$

where **F** is the generalized force vector, $\Delta\mathbf{x}^T = (\Delta\mathbf{x}_\mathbf{p}^T, \Delta\mathbf{x}_\mathbf{o}^T)$ is the position and orientation error vector, and $\mathbf{K_p}$ and $\mathbf{K_v}$ are diagonal gain matrices. The subscript d denotes the desired quantities. In our experiments, $\ddot{\mathbf{x}}_\mathbf{d}$ is selected to be zero.

5. Hardware And Some Experimental Results

A number of experiments were performed on the CMU DD *Arm* II robotic system. This robotic system consists of: a) a Sun 3/260 host system on a VME bus, b) Multiple Ironics M68020 boards, c) a Mercury 32000 Floating Point Unit, d) an IDAS/150 image processing system, d) a Panasonic industrial CCD color camera, Model GP-CD1H, e) *six* Texas Instrument TMS320 DSP processors, each controlling one joint of the CMU DD Arm II system, f) sensors such as a tactile sensor and a force sensor, and g) a *six* degrees of freedom joystick. The IDAS/150 contains a Heurikon 68030 board as the controller of the vision module and two floating point boards, each one with computational power of 20 Mflops. The software is organized around **3** processes: a) **Vision process** which does all the image processing calculations and has a period of 150 ms, b) **Interpolation process** which reads the data from the vision system, interpolates the data and sends the reference signals to the robot Cartesian controller, and c) **Robot controller process** which drives the robot and has a period of **3.33** ms. During the experiments, the camera is mounted on the end-effector and has a focal length of 7.5mm. The objects **(books,** toys, pencils) are moving on a plane (average depth $Z_s = 680mm$). The user, by moving the mouse around, proposes to the system some of the object's features that he is interested in. Then, the system evaluates on-line the quality of the measurements, based on the confidence measures described in [13]. Currently, four features are used and the size of the attached windows is **10x10.** The experimental results are plotted in Fig. **1** and 2 where the dotdashed trajectories correspond to the trajectories of the center **of mass** of the moving objects. The vector Mez_P represents the position of the end-effector in the world frame. The simple **PD** produces oscillations around the desired trajectory. In this example, along with the translational motion, the object performs a rotational motion around an axis that **passes** through the center of **mass** of the object. Even with noisy measurements, the LQG **seems** to perform well. This becomes obvious, when one reduces the number of the windows which are used (increased noise in the measurements), and the LQG controller continues to keep the target at the desired position.

6. Conclusions

In this paper, we considered a LQG approach to the robotic visual tracking problem (eye-in-hand configuration). We formulated the problem as a control and vision problem and dis-*cussed* the issues related with this formulation. **A** cross-correlation technique **(SSD** Optical Flow) was used to provide accurate measurements **of** the object's motion parameters. ***An*** LQG regulator in conjunction with Cartesian robotic controllers were studied as possible solutions to the robotic visual tracking problem. The vision and control techniques were tested on a real robotic environment, the CMU DD **Arm** II. Experimental results show that the methods are quite accurate, robust and promising. Their most important characteristic is that they can be implemented in real-time. **Future** research efforts should be focused on the extension of the techniques to the **3-D** robotic visual tracking problem, the explicit use of the target model in the whole mathematical formulation, the solution of the problem of vanishing features, and finally, the direct integration of the robot dynamics in the feedback loop.

References

1. L Matthies, R. Szeliski, and T. Kanade, "Kalman filter-based algorithms for estimating depth from image sequence", Tech. report 88-1, Carnegie Mellon University, The Robotics Institute, 1988.

2. D. Tsakiris, "Visual tracking strategies", Master's thesis, Department of Electrical Engineering, University of Maryland, 1988.

3. T.P. Wallace and O.R. Mitchell, "Analysis of three-dimensional movement using Fourier descriptors", *IEEE Trans. PAMI,* Vol. 2, No. 6,1980, pp. 583-588.

4. D.B. Gennery, "Tracking known three-dimensional objects", *Proc. AAAI 2nd Natl. Conf.* on *AI,* 1982, pp. 13-17.

5. J.T Feddema, C.S.G. Lee, and O.R. Mitchell "Automatic selection of image features for visual servoing of a robot manipulator", *Proc. of the IEEE Intern. Conf.* on *Robotics and Automation,* May 1989, pp. 832-837.

6. L.E Weiss, A.C. Sanderson, and C.P. Neuman, "Dynamic sensor-based control of robots with visual feedback", *IEEE Journal* of *Robotics and Automation,* Vol. RA-3, No. 5, October 1987, pp. 404-417.

7. J.W Roach and J.K Aggarwal, "Computer tracking of objects moving in space", *IEEE Trans. PAMI,* Vol. 1, No. 2, 1979, pp. 127-135.

8. A.E. Hunt and A.C Sanderson, 'Vision-based predictive tracking of a moving target", Tech. report CMU-RI-TR-82-15, Carnegie Mellon University, The Robotics Institute, January 1982.

9. S.W. Lee and K. Wohn, "Tracking moving objects by a mobile camera", Tech. report MS-CIS-88-97, Department of Computer and Information Science, University of Pennsylvania, November 1988.

10. B.K.P Horn and B.G. Schunck, "Determining optical flow", *Artificial Intelligence,* Vol. 17,1981, pp. 185-204.

11. P. Anandan, "Measuring visual motion from image sequences", Tech. report COINS-TR-87-21, COINS Department, University of Massachusetts, 1987.

12. N. Papanikolopoulos., P. K. Khosla, and T. Kanade, 'Vision and control techniques for robotic visual tracking", *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 1991, pp. 857-864.

13. N. Papanikolopoulos, P. Khosla, and T. Kanade, "Robotic visual tracking: Theory and experiments", Tech. report ,Carnegie Mellon University, The Robotics Institute, 1990.

14. F.L. Lewis, *Optimal control*, JohnWiley & Sons, New York, 1986.

15. A. Gelb, *Applied optimal estimation*, MIT Press, Cambridge, 1974.

16. N. Papanikolopoulos, P. Khosla, and T. Kanade, *Adaptive robotic visual tracking*, Accepted to the American Control Conference, 1991.

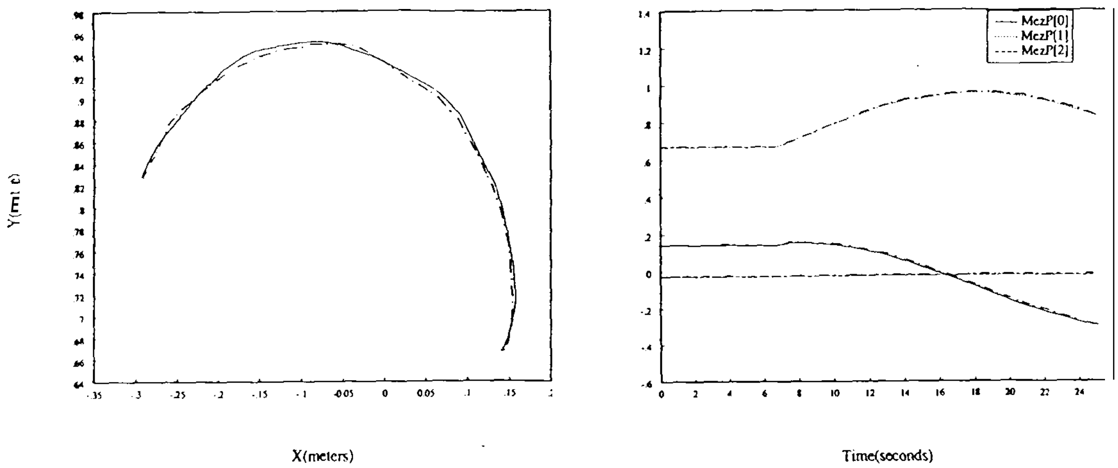

Figure 1: LQG controller in conjunction with a Cartesian PD robotic controller with gravity compensation.

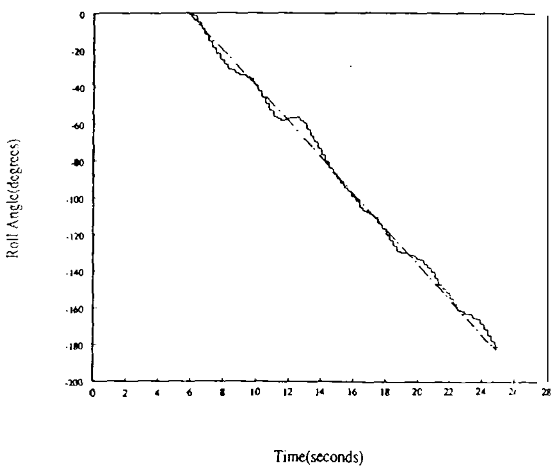

Figure 2: Roll trajectory of the robot end-effector and the object in the previous example.