5

Designing and Building Controllers for 3D Visual Servoing Applications under a Modular Scheme

M. Bachiller, C. Cerrada, J. A. Cerrada
Escuela Técnica Superior de Ingeniería Informática, UNED
Spain

1. Introduction

Industrial Robots are designed for tasks such as grasping, welding or painting in where working conditions are well settled. However, if the working conditions change, those robots may not be able to work properly. So, robots require sensor- based control to perform complex operations and to react to changes in the environment. The achievement of such complex applications needs the integration of several research areas in vision and control such as visual matching, visual tracking and visual servoing (Petersson et al. 2002). Information about the actual system and its environment can be obtained from a great variety of sensors. Vision is probably the sensor that provides the greatest and richest sensing information but the processing of such information becomes complicated. Nevertheless, computer vision has been improved a lot in the last years and it is being frequently used in robotics systems, although serious limitations appear in real time applications due to the time necessary for image processing.

The use of computer vision as a feedback transducer strongly affects the closed loop dynamics of the overall system. Latency is the most significant dynamic characteristic of vision transducers and it has many sources, including transport delay of pixels from the camera to vision system, image processing algorithms, control algorithms and communications with the robot. This delay can cause instability in visual closed loop systems. To achieve fast response and high control accuracy the design of a specific visual feedback controller is required.

Visual servoing is the result of merging several techniques in different fields including image processing, kinematics, dynamics, control theory and real time computing. The task in visual servoing is to control a robot to manipulate its environment with the help of vision as a feedback sensor. An excellent overview of the main issues in visual servoing is given in (Nelson & Papanikolopoulos, 1998).

In this work we present a complete form of designing visual servoing controllers for robotic systems built over a modular conception. The designing method is particularly applied to industrial manipulators equipped with a camera mounted on its end effector, known as eye in hand configuration. The modular conception means that the overall system is composed of a set of independent modules, or subsystems, that are put together, configuring a modular system. In this kind of systems any module can be replaced by other one with the same functionality and the visual controller computation procedure will not change. The goal of any visual servoing

Source: Industrial Robotics: Programming, Simulation and Applicationl, ISBN 3-86611-286-6, pp. 702, ARS/plV, Germany, December 2006, Edited by: Low Kin Huat

system is the same independently of how it is constructed: to control the robot's end effector pose relatively to the target object pose. But the main advantage of our consideration is that if a module has to be changed for any reason it will not be necessary to replace the others or to redesign the full platform, but just to compute a new controller.
This scheme allows us to deal with the problem of controller design from a more generic point of view. We propose a new strategy for designing the control subsystem, which presents advantages as for calculation time, simplicity and estimation errors of object position. One purpose is to demonstrate the possibility of getting competitive results in real time performance with respect to more closed solutions shown in other works and other control configurations, while maintaining the advantage of easy system adaptability.
The remainder of this paper is structured as follows. Firstly, in section 2, we review the relevant fundamentals of visual control arquitectures. In sections 3 and 4, we describe the proposed design methodology and then we analyze the performance of the designed experimental platform to deduce the mathematical models of the involved components. In section 5, we design various control strategies used when the object is static and moving, respectively. After that, we carry out a study of designed controllers in section 6, and we present and discuss some experimental results obtained in the platform in section 7. Finally, conclusions and new tendencies of our work are stated at section 8.

2. Overview of visual control architectures

There are several ways of implementing control schemes when vision is used as a part of the robot position control system. To better understand the different architectures combining control and vision it is previously necessary to analyze how the robotic control system works by itself. A typical robot position control system is shown in figure 1. A system like this is characterized by using a position sensor for each arm joint (Craig, 1989) while the control objective is the Cartesian position of the robot's end effector.
Let $\underline{X}_{ref}$ be the vector denoting the desired Cartesian robot position. For a given reference position $\underline{X}_{ref}$ the corresponding angular reference vector $\underline{\theta}_{ref}$ is obtained by evaluating the inverse kinematics transform of the robot. These joint reference values are the inputs to the joint position controllers of the robot, and there are as many joint controllers as joint drives the robot has. Each controller uses its position sensor to feedback the actual joint angle in an inner control loop. The actual Cartesian position $\underline{X}$ is obtained from the actual joint values $\underline{\theta}$ by evaluating the direct kinematics transform, which depends on the robot geometry.

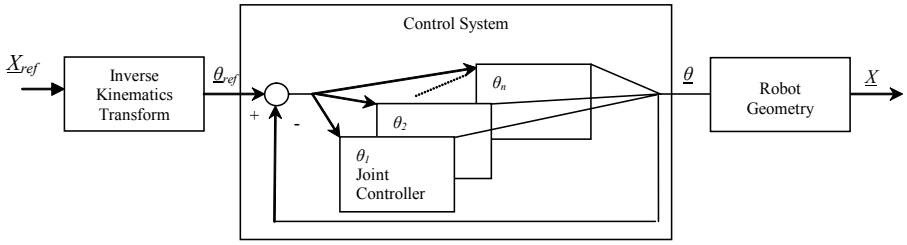


Fig. 1. Block diagram of a robotic control system.

Notice that this scheme represents an open loop control strategy for the Cartesian position because some permanent error can exist between the desired robot position $\underline{X}_{ref}$ and the actual robot position $\underline{X}$. External feedback is necessary to close an outer loop in order to compensate

such error. Visual sensors represent an excellent option to measure and feedback actual robot position in this loop and help to reduce positioning errors. The only drawback of using vision as position sensor is the big amount of computing time required to obtain it.
Earliest applications combining computer vision and robot control where implemented under an open loop scheme, known as **static look and move** architecture. Figure 2 shows a control structure of this type that works by sequentially executing three independent steps (Weiss, 1984). The step by step sequence is repeated until the precision required by the task is achieved. Therefore, the number of necessary iterations depends on the particular task. Let us remark that the nature of this robot positioning process is completely iterative and sequential.

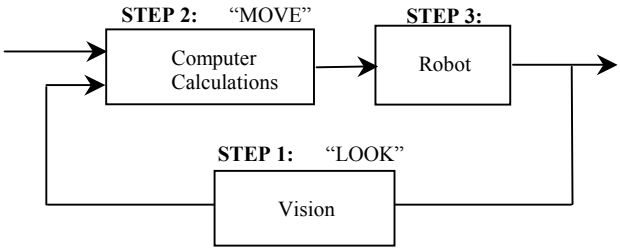


Fig. 2. Static look and move control architecture.

To implement the three steps in parallel in a dynamic control system it is necessary to close the loop by using visual information in the feedback, such as the figure 3 shows. This way of functioning is known as **visual feedback** and the different types of systems that implement it are generically classified as visual servoing architectures.
The main advantage of visual feedback is that it offers much faster dynamic responses than static look and move but, on the other hand, several design problems appear. First, a specific controller is required to compensate the dynamics of the plant, in this case the robot. Secondly, unavoidable delays are introduced in the feedback signal generation, what provokes system instabilities. Nevertheless, the most adequate visual control scheme for robot tasks carried out in unstructured environments is visual servoing because it allows to the robot reacting in real time to unexpected changes in the environment.

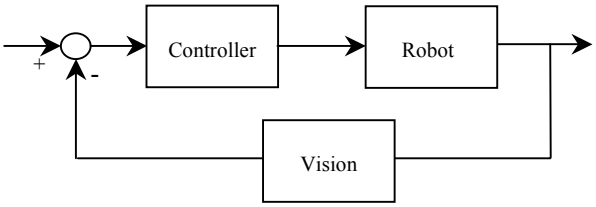


Fig. 3. Generic visual feedback control scheme.

Visual servoing systems typically use one of the two following camera configurations: end effector mounted, often called an eye in hand configuration, or fixed in the workspace. Statically located visual sensors have a very limited working region when taking into account the limited depth of field and spatial resolution that vision sensors typically possess. The working region of a visual sensor can be greatly extended if the camera is allowed to move while tracking the object of interest.
Apart from camera configuration, visual servoing architectures can be differentiated

attending to two main criteria. First, the way of inputting the signal generated by the controller to the robotic system and, secondly, the nature of the visual information supplied by the vision system, i.e. the nature of the controlled variable.
Concerning the first criterion, visual servoing architectures for controlling manipulators can be classified into two fundamental categories: dynamic look and move structures and visual servo structures. When the control architecture is hierarchical and uses the vision system to calculate the set of inputs to the joint level controller, making use of inner joint feedback loops to stabilize the robot, it is referred to as a dynamic look and move structure. A very important advantage of this structure is that it separates the kinematics singularities of the mechanism from the visual controller, allowing the robot to be considered as an ideal Cartesian motion device. In contrast, visual servo structure eliminates the robot controller replacing it with a visual controller in such a way that it is used to compute directly the joints inputs, the loop being the only thing used to stabilize the mechanism.
Concerning the controlled variable, visual servoing systems are classified into two groups: image based control systems and position based control systems. In an image based control system, the error variable is computed in the 2D-image space. This approach eventually reduces the computational delay, eliminates the necessity of image interpretation and eliminates errors due to sensor modeling and camera calibration. However, they require online computation of the image Jacobian. Unfortunately, this quantity inherently depends on the distance from the camera to the target, which in a monocular system is particularly difficult to calculate. Many systems utilize a constant image Jacobian which is computationally efficient, but valid only over a small region of the task space [Hutchinson et al 1996]. On the other hand, in a position based control system, the error variable is computed in the 3D Cartesian space. The main advantage of the last approach is that position of the camera trajectory is controlled directly in the Cartesian space.
Figures 4, 5, 6 and 7 show the schemas of the four possible structures concerning to the combination of the two mentioned criteria, for an eye in hand camera configuration. Notice that visual servo structures require the direct access to the robot operating system in order to govern the joint variables directly, whereas dynamic look and move structures consider the robot as an independent system and can handle it as a black box.
There has been a significant amount of research activity on image based control methods (Weiss et al., 1987), (Feddema & Mitchell, 1989), (Chaumette et al., 1991), (Espiau et al., 1992), (Khosla et al., 1993), (Corke, 1993) whereas there have been only a few researchers working on position based control methods (Koivo & Houshangi, 1991), (Allen et al., 1993), (Wilson et al., 1996), (Vargas et al., 1999). This tendency can be justified because image based systems usually reduce computation delays and eliminate errors due to sensor modeling and camera calibration processes.

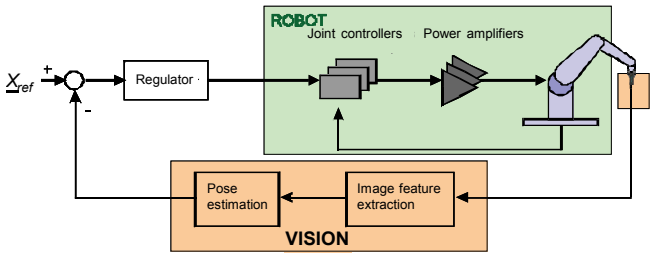


Fig. 4. Dynamic position-based look-and-move structure.

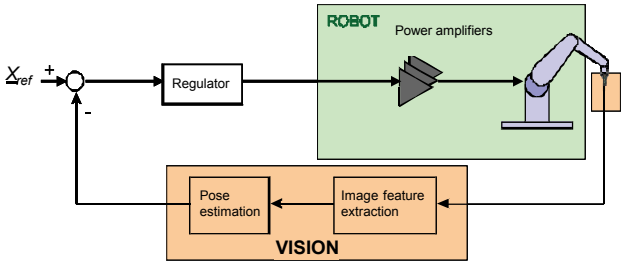


Fig. 5. Position-based visual servo structure.

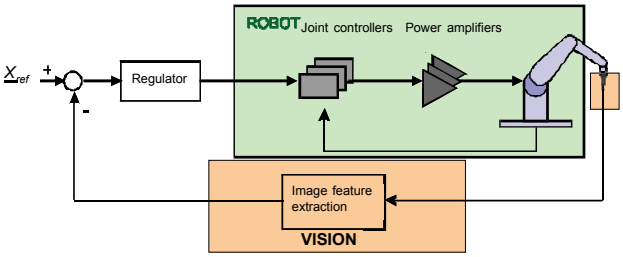


Fig. 6. Dynamic image-based look-and-move structure.

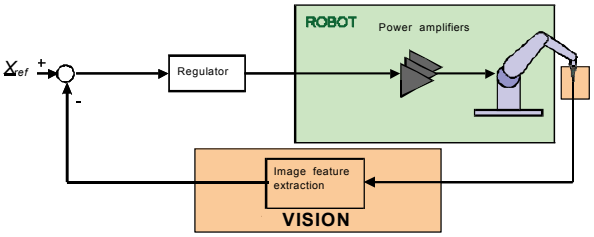


Fig. 7. Image-based visual servo structure.

More recent works (Malis 1999) consider a new visual servo structure that combines the advantages of image based and position based control schemes and avoids some of their disadvantages. This approach, known as 2½D, is characterized by considering the controlled variable composed by two parts, one being represented in the Cartesian space and the other in the image plane. It does not need any 3D model of the object or any precise camera calibration either. However, the method is more sensitive to the existing noise in the image.
Some final considerations can be made concerning the adequacy of the considered visual control architectures to designing and building systems from a modular conception. Having this idea in mind, the most appropriate selection seems to be taking into account position based systems and dynamic look and move structures. Several reasons can be pointed out to think so:

- Position based methods allow a direct and more natural specification of the desired trajectories for the end effector in Cartesian coordinates.
- Many robots have an interface for accepting Cartesian velocity or incremental position commands. In that sense they can be exchanged and all of them can be considered as black boxes.
- The visual controller design of visual servo structures is very complex due to the plant being nonlinear and highly coupled.

3. Modular scheme for visual servoing controllers design

The main idea of the work presented in this paper is to consider the overall visual servoing system formed by a set of independent modules or subsystems connected in a specific way. Taking into account the considerations exposed in the previous section, we have applied this idea to the specific case of designing a position based dynamic look and move system built with an industrial manipulator and an off-the-shelf camera mounted on its end effector. This scheme allows us to deal with the problem of controller design from a more generic point of view. We propose a new strategy for designing the control subsystem, which presents advantages as for calculation time, simplicity and estimation errors of object position. Notice that depending on the type of task we will only have to change the control subsystem specifically designed for this task.

3.1 Preliminary design considerations

First of all, we have used the eye in hand camera configuration in our design proposal keeping in mind its versatility, but this choice does not represent any restriction to the method´s generality. On the other hand, and taking into account that all the studied visual servoing architectures are formed practically by the same functional blocks, we consider that a visual servoing system can be considered as the integration of at least the following set of components (see figure 8):

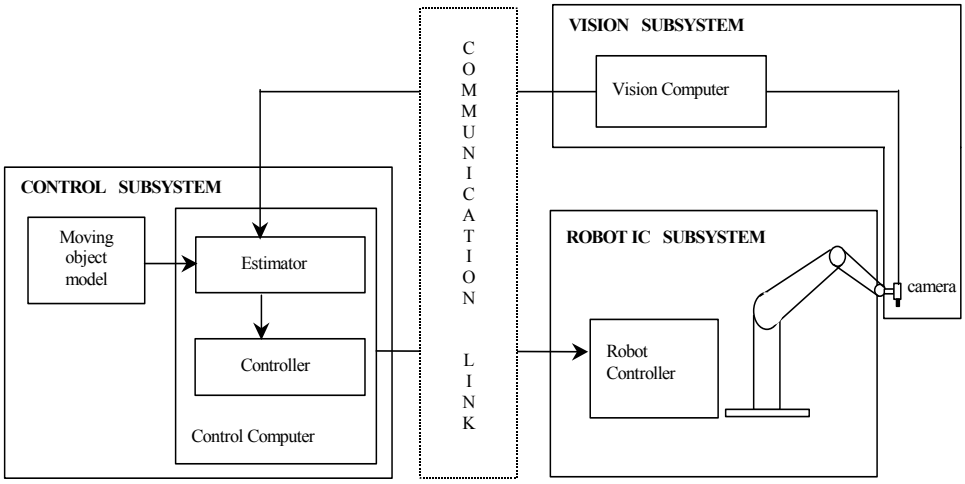


Fig. 8. Subsystems composition of a generic visual servoing system.

1. **Robotic Subsystem**: The robotic subsystem can be any industrial manipulator capable of changing its trajectory in real time by means of specific commands received through a serial line (ALTER line). This command line allows introducing new reference values every T_r milliseconds.
2. **Vision Subsystem:** A camera, the specific hardware and software for image digitalization, and image processing can constitute this subsystem. Both hardware and software can be installed in a vision computer. The function of this subsystem is to measure the actual position of the target object with respect to the robot's end effector in a Cartesian coordinate system every T_v milliseconds.

3. **Control Subsystem**: An estimator (optional) and a controller can form the control subsystem. In principle, this subsystem should be implemented over a specific control computer, but it can also be included in the vision computer, in the robot controller operating system, or both.
4. **Communications**: Communication links between vision and control subsystems and also between control and robotic subsystems form this module.

Notice that the overall system is a multirate system in nature: the sampling interval of the manipulator is T_r while the vision subsystem sampling period is T_v, T_v being larger than T_r. To simplify the study from the control point of view, the vision period can be adjusted to be n times the robot period, n being an integer value, i.e.:

$$T_v = n.T_r \tag{1}$$

For notation concerns, we will use z to represent the z-transform of a system sampled with a period T_v and $\hat{z}$ to the z-transform of a system sampled with a period T_r.

3.2 Fundamentals of the design methodology

The procedure followed to design and evaluate the controllers for the selected visual servoing system can be divided in three well diferenciated phases. The purpose of the first phase is to obtain a mathematical model of the overall system. We call *system model construction* to this phase and it is decomposed into three conceptually different stages: *subsystems modeling*, *system blocks diagram construction* and *identification*. The second phase is properly devoted to design the control subsystem and we call it *control subsystem design*. This design will depend on the type of task to be performed by the system. So, this is the phase where different control techniques can be explored and analyzed. Finally, designed controllers need to be evaluated in order to analyze and compare the performance achieved by each one. All this work is made by means of tests and simulations in the third phase generically called *test results and performance evaluation.*

Apart from the intrinsic value that the comparative study and the performance analysis have by themselves, one of the objectives of the last phase could be to selecting the best controller for each considered task with the purpose of implementing it in the experimental platform. And that is precisely the way we have actuated in this work.

In summary, once the fundamentals and organization of the methodology has been established, the three following sections will explain them more in detail. After that, one additional section will show the real results obtained with the selected controllers built on the experimental platform.

4. First phase: Model System Construction

As it has been mentioned, we are considering a visual servoing system with position based control, dynamic look and move structure and eye in hand configuration. The first phase is to obtain a valid mathematical model for the plant to be governed. In general, to get that model we are using techniques based in conventional control theory, what means that we are going to work with the transfer function of each subsystem and then to build a block diagram of the complete system. In this section we describe the three stages in which we have implemented this phase and the possible alternatives that can be taken into account.

4.1 Subsystems Modeling

We use the architecture shown in Figure 9 which represent to the considered visual servoing system associated to our platform. The four subsystems described in the previous section can be clearly distinguished in this diagram. Nevertheless, several considerations must be taken into account before getting a valid model for each subsystem. First at all, the control subsystem is the objective of the design and consequently it will be considered in the next section. A block named $\underline{R}(z)$ will represent it. With respect to the communication links, they can be considered from the dynamic point of view as small delays that can be integrated in the vision or in the robot time periods. Therefore, no dynamic models are required for them. In summary, it is only necessary to have accurate dynamic models for the sensor used (the vision subsystem) and for the plant to be controlled (the robotic subsystem). These two models are described in next paragraphs.

Vision Subsystem Model

Vision subsystem is in charge of providing a measurement of the 3D object position every sampling period. We have implemented a fast algorithm to estimate object position, based on the simplification of a generic camera calibration model described in (Gonzalez, 1998). This model efficiently compensates non-negligible radial distortion effects introduced by standard lenses, and it has been widely tested in complete 3D object pose measurement. Nevertheless, when object orientation estimation is not required the total on-line computation time can be highly reduced, allowing to achieve smaller T_v values.

Figure 9 illustrates the basic geometry of the considered camera model and robotic environment. This figure also shows the two-point pattern attached to the moving object which is used to estimate its 3D position. Taking away the method for correcting distortion, three main coordinate systems are involved in the model used in this eye in hand configuration:

- $({}^wO, {}^wX, {}^wY, {}^wZ)$: 3D world coordinate system. It is assumed that it coincides with the intrinsic coordinate system associated to the robot, i.e. with the robot coordinate system.
- $({}^cO, {}^cX, {}^cY, {}^cZ)$: 3D camera coordinate system. It is assumed that it is located at the robot's end-effector, i.e., at the actual robot position $\underline{p}_r$.
- $({}^fO, {}^fX, {}^fY)$: 2D image coordinate system. It is the coordinate system of the frame memory where the digitized image is stored and processed.

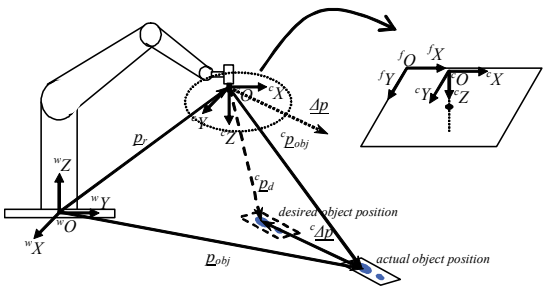


Fig. 9. Basic geometry of the robotic and vision subsystems in the considered configuration.

Let (x_i, y_i, z_i) be the world coordinates of a given point $\underline{p}_i$ located anywhere in the camera workspace, $({}^cx_i, {}^cy_i, {}^cz_i)$ be the camera coordinates of this point, and $({}^fx_i, {}^fy_i)$ be the coordinates of the projection of the point in the image coordinate system. The last two coordinates are related to each other by a (3x4) transformation matrix A which includes the perspective projection elements and the scale factors, as well as the relative translation and rotation between the two coordinate

systems when they exist. In short, they are related by the expression:

$$\begin{bmatrix} \omega\,{}^{f}x_i \\ \omega\,{}^{f}y_i \\ \omega \end{bmatrix} = A \cdot {}^{c}\underline{p}_i = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \cdot \begin{bmatrix} {}^{c}x_i \\ {}^{c}y_i \\ {}^{c}z_i \\ 1 \end{bmatrix} \tag{2}$$

where the 12 parameters of the A matrix are estimated in the off-line calibration process.
To estimate the object position we take advantage of the known two-point geometry pattern. Both points are black circles in a plane being D the constant distance between them, and being one circle double the size of the other. We assume that the pattern plane always remains perpendicular to the camera optical axis, i.e., the two points are at the same ${}^{c}z$ coordinate. Let $({}^{c}x_1, {}^{c}y_1, {}^{c}z_1)$ and $({}^{c}x_2, {}^{c}y_2, {}^{c}z_2)$ be the camera coordinates of the big point pattern and the small one respectively in a given position, and let $({}^{f}x_1, {}^{f}y_1)$ and $({}^{f}x_2, {}^{f}y_2)$ be their corresponding coordinates in the projected image. We will identify the actual object position in camera coordinates with the biggest point position, i.e., ${}^{c}\underline{p}_{obj} = ({}^{c}x_1, {}^{c}y_1, {}^{c}z_1)$. Let ${}^{c}\underline{p}_d$ be the desired object position in camera coordinates, i.e. the reference position of object with respect to the camera, which remains constant during the experiment. The actual object position can be determined by solving the next system with five equations and five unknowns:

$$\begin{cases} (a_{11} - a_{31} \cdot {}^{f}x_1) \cdot {}^{c}x_1 + (a_{12} - a_{32} \cdot {}^{f}x_1) \cdot {}^{c}y_1 = -(a_{13} - a_{33} \cdot {}^{f}x_1) \cdot {}^{c}z_1 + (a_{34} \cdot {}^{f}x_1 - a_{14}) \\ (a_{21} - a_{31} \cdot {}^{f}y_1) \cdot {}^{c}x_1 + (a_{22} - a_{32} \cdot {}^{f}y_1) \cdot {}^{c}y_1 = -(a_{23} - a_{33} \cdot {}^{f}y_1) \cdot {}^{c}z_1 + (a_{34} \cdot {}^{f}y_1 - a_{24}) \\ (a_{11} - a_{31} \cdot {}^{f}x_2) \cdot {}^{c}x_2 + (a_{12} - a_{32} \cdot {}^{f}x_2) \cdot {}^{c}y_2 = -(a_{13} - a_{33} \cdot {}^{f}x_2) \cdot {}^{c}z_1 + (a_{34} \cdot {}^{f}x_2 - a_{14}) \\ (a_{21} - a_{31} \cdot {}^{f}y_2) \cdot {}^{c}x_2 + (a_{22} - a_{32} \cdot {}^{f}y_2) \cdot {}^{c}y_2 = -(a_{23} - a_{33} \cdot {}^{f}y_2) \cdot {}^{c}z_1 + (a_{34} \cdot {}^{f}y_2 - a_{24}) \\ ({}^{c}x_1 - {}^{c}x_2)^2 + ({}^{c}y_1 - {}^{c}y_2)^2 = D^2 \end{cases} \tag{3}$$

Error vector ${}^{c}\Delta\underline{p} = ({}^{c}\Delta x, {}^{c}\Delta y, {}^{c}\Delta z)$ is then computed by subtracting the actual position to the reference position, ${}^{c}\Delta\underline{p} = {}^{c}\underline{p}_d - {}^{c}\underline{p}_{obj}$. This vector is equal but opposite to the robot incremental movement $\Delta\underline{p}$ required to cancel such as error: $\Delta\underline{p} = -\,{}^{c}\Delta\underline{p} = {}^{c}\underline{p}_{obj} - {}^{c}\underline{p}_d$. From figure 1bis it can be seen that ${}^{c}\underline{p}_{obj}$ is the object position in the camera system and it coincides with $(\underline{p}_{obj} - \underline{p}_r)$ in the world system, i.e. $\Delta\underline{p} = \underline{p}_{obj} - \underline{p}_r - {}^{c}\underline{p}_d$. Therefore, the vision subsystem provides every sampling period a triplet of values $\Delta\underline{p} = (\Delta x, \Delta y, \Delta z)$ in the world coordinate system representing the position increment that the robot has to be moved to make the target object be at the right position with respect to the camera. To reach those values, every sampling period the vision subsystem must acquire an image of the pattern, digitize it, process it to obtain $({}^{f}x_1, {}^{f}y_1)$ and $({}^{f}x_2, {}^{f}y_2)$, evaluate and solve the system of equations (3) to obtain ${}^{c}\underline{p}_{obj}$, and finally compute $\Delta\underline{p} = {}^{c}\underline{p}_{obj} - {}^{c}\underline{p}_d$. Because of the required computation time, this information is not available until the next sampling instant. In that sense, when the overall system is working with T_v period, the vision subsystem can be considered as a pure delay. So its transfer functions matrix is:

$$V(z) = diag\,(z^{-1}, z^{-1}, z^{-1}) \tag{4}$$

Figure 10 shows the representative block of the vision subsystem. This diagram also considers a noise signal $\underline{r}_s$ which can be produced, for example, because of bad illumination conditions or just in the digitizing process.

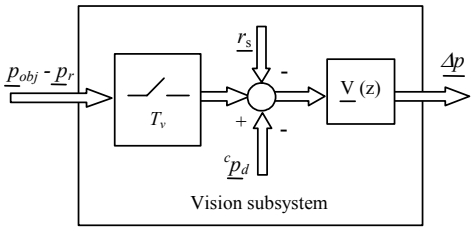


Fig. 10. Representative block of the vision subsystem.

Robotic Subsystem Model

When the manipulator works through the ALTER line it behaves as a decoupled multivariable system. It can be considered as composed of six independent loops (three of them concerning the Cartesian position and the other three concerning the orientation). As it has been mentioned above, in this work we are only taking into account the three loops relative to the Cartesian position. Thus, the manipulator with its actuators and their current feedback loops can be considered as a Cartesian servo device.

In principle, the robotic subsystem can be modeled as an integrator for each Cartesian coordinate: its output is the actual robot position that is equal to the previous one plus the incremental position input signal. From the experimental responses obtained when exciting the used robot with step signals for all the coordinates we have observed that they do not behave as pure integrators, and that some additional dynamic features appear. In order to model this behavior, we consider that the transfer function for each Cartesian coordinate is a second order system with a pole at $z = 1$ (representing the integrator) and another one at $z = \beta_i$ which determines the characteristics of the transient response of the robotic subsystem. So, the transfer functions matrix of the robotic subsystem is:

$$\underline{G}(\hat{z}) = diag\left(\frac{n_x}{(\hat{z}-1)(\hat{z}-\beta_x)}, \frac{n_y}{(\hat{z}-1)(\hat{z}-\beta_y)}, \frac{n_z}{(\hat{z}-1)(\hat{z}-\beta_z)} \right) \tag{5}$$

Figure 11 shows the representative block of the robotic subsystem. The sampling time is T_r and $\underline{e}$ is a noise signal that represents small errors due to data acquisition, which will be used in the identification process. That noise is modeled as a sequence of independent and identically distributed random variables with zero mean.

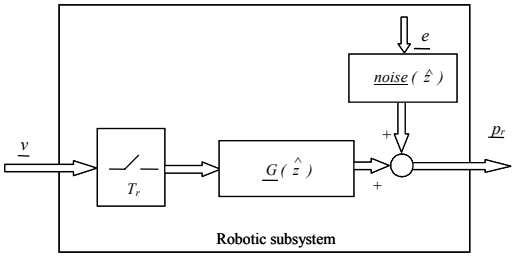


Fig. 11. Representative block for the robot subsystem.

Notice that the input signal to this block (or control signal) can be considered as a velocity reference ($\underline{v}$) since the robotic subsystem must make the specified displacement in T_r milliseconds.

4.2 System´s Block Diagram Construction

If we join all the subsystems in accordance with the position based dynamic look and move structure we obtain a block diagram as shown in figure 12, where the control signal $\underline{v}$ is a velocity reference. For that reason we will refer to this configuration as velocity control scheme.

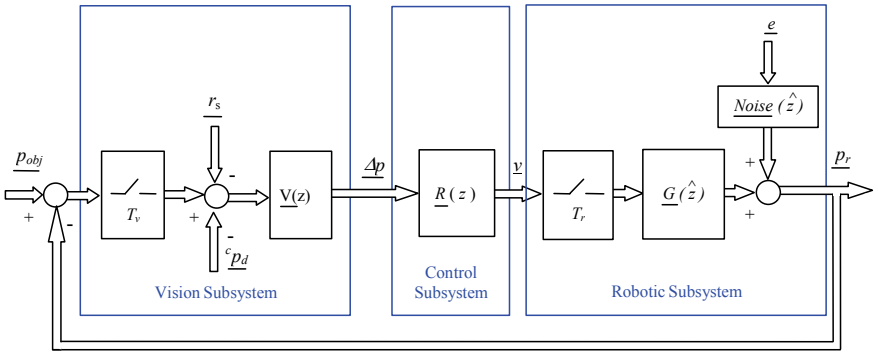


Fig. 12. Block diagram for velocity control.

Using the equation (1) and doing some manipulations this block diagram can be converted into the equivalent one shown in figure 13, where $\underline{T}(\hat{z})$ is given by:

$$\underline{T}(\hat{z}) = diag\left(\frac{1-\hat{z}^{-n}}{1-\hat{z}^{-1}}, \frac{1-\hat{z}^{-n}}{1-\hat{z}^{-1}}, \frac{1-\hat{z}^{-n}}{1-\hat{z}^{-1}}\right) \tag{6}$$

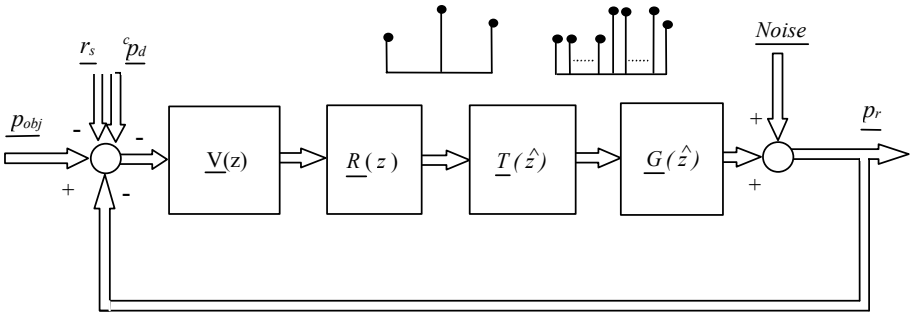


Fig. 13. Equivalent representation of the block diagram for velocity control.

At this point it is important to remark that previous block diagrams correspond to multirate control systems, because two different sampling periods appear in all the representations. In order to design the controllers at the slowest sampling rate, we should have the transfer functions matrix of the robotic subsystem corresponding to a sampling period T_v. However, we know these transfer functions sampled with period T_r. To obtain the desired transfer functions the following two step procedure can be applied: (1) Estimate the continuous transfer functions matrix from the discrete transfer functions matrix sampled with period T_r. (2) Calculate the discrete transfer functions matrix by sampling the continuous transfer functions matrix with period T_v. Applying this procedure to $\underline{G}(\hat{z})$, we obtain $\underline{G}(z)$. Then the

block diagram of figure 13 can be converted into a more generic block diagram as shown in figure 14. Notice that T_v is the sampling period of this new diagram.

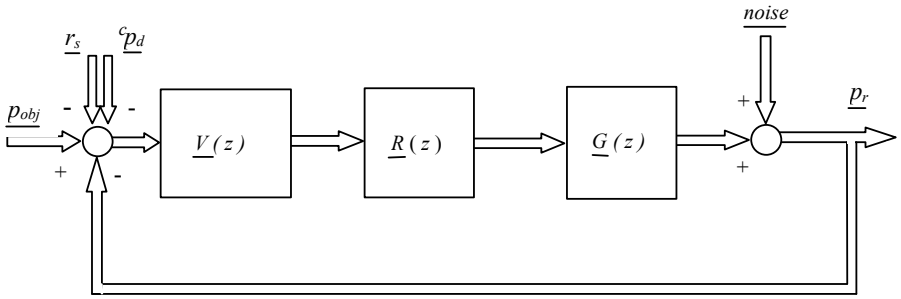


Fig. 14. Generic block diagram of the system.

4.3 Identification

Once a model has been selected to represent each subsystem, an identification process is required to estimate the unknown parameters. In general terms, an identification experiment can be performed by exciting the system (using some kind of input such as step, sinusoid or random signals) and observing the input and the output over a time interval. These signals are normally registered in a computer for information processing. Some statistically based method can be used to estimate the model unknown parameters such as the coefficients of the difference equation. The model obtained is then tested to verify whether it is an appropriate representation for the system. If that is not the case, some more complex model must be considered, its parameters estimated and the new model validated. Concerning our experimental platform the identification process can be summarized in the following comments.

As it has been exposed in the modeling section the vision subsystem behaves as a pure delay. Therefore no unknown parameter must be estimated here. The robotic subsystem is working through its ALTER line. This command allows us to introduce new reference values each 16 milliseconds, which means that in this platform $T_r = 16\ milliseconds$. It has already been mentioned that we are considering a second order linear system to model each coordinate of the robotic subsystem. To identify the respective parameters we have excited each Cartesian coordinate of the robot with different input signals (steps, ramps and parabolas) through its ALTER line. In order to take into account the possible errors produced in the data acquisition process, an autoregressive moving average with exogenous model (ARMAX) has been used to fit the robotic subsystem parameters, due to this model a random term is included that can be interpreted as noise. Parameters are finally identified by minimizing the estimated square error function using an iterative Gauss-Newton algorithm. The obtained transfer functions matrix is given by:

$$\underline{G}(\hat{z}) = diag\left(\frac{0.6325}{(\hat{z}-1)(\hat{z}-0.3810)}, \frac{0.6840}{(\hat{z}-1)(\hat{z}-0.3194)}, \frac{0.7081}{(\hat{z}-1)(\hat{z}-0.2953)} \right) \tag{7}$$

Before starting the control subsystem design, and as it has been mentioned in the previous section, it is first necessary to estimate the continuous transfer functions matrix from the equation

(7). To this aim, we have used the method described in (Feliu, 1986) The obtained $\underline{G}(s)$ is:

$$\underline{G}(s) = diag\left(\frac{-35.689\,s + 3851.45}{s^2 + 60.309\,s}, \frac{-42.634\,s + 4485.437}{s^2 + 71.4102\,s}, \frac{-45.992\,s + 4791.679}{s^2 + 76.298\,s}\right) \quad (8)$$

The resulting continuous transfer functions matrix is then sampled with a T_v period using a zero order hold. Considering *n=10* in the equation (1) and applying this discretization procedure to the transfer functions matrix of robotic subsystem, the following $\underline{G}(z)$ is obtained:

$$\underline{G}(z) = diag\left(\frac{8.5673z + 1.6499}{z(z-1)}, \frac{8.5734z + 1.4765}{z(z-1)}, \frac{8.6224z + 1.459}{z(z-1)}\right) \quad (9)$$

5. Second phase: Control Subsystem Design

In order to design the control subsystem ($\underline{R}(z)$) and to study the behavior of the full system, it is necessary to define the task that the robot has to do. It is evident that the controller will depend on the type of task to perform and the physical constraints of the robot. In general, the problem of robotic visual tracking/servoing is defined as in (Papanikolopoulos, 1992): "*Move the manipulator (with the camera mounted on its end effector) in such a way that the projection of a moving or static object is always at the desired location in the image*". This states the main objective of the control subsystem, but some differences appear if the object moves or remains static.

In that sense, we have considered and studied two different tasks. In the first one, denoted as task 1, the objective is to pose the robot with respect to a static object. The specifications in this case are to reach the reference with no oscillations and within a short period of time during the movement of the robot end effector when a step input is applied. Provided that this step input can be of great amplitude, it is very important to study the Cartesian acceleration value in order to avoid the saturation of some motors. The second task, denoted as task 2, consists of tracking a moving object. Specifications now are that the steady state error becomes zero in a minimum setting, in such a way that the robot will track perfectly and quickly the object when it follows a ramp, parabolic or other type of input.

In the following paragraphs the two types of tasks will be analyzed. Several controllers will be designed and tested. Some comparative observations will be added at the end.

5.1 Task 1: Tracking of a static object

In this instance, the objective is to move the robot, with the camera mounted on its end effector, in such a way that the projection of a static object is in the desired location ${}^f\underline{p}_d$ of the image (see figure 15).

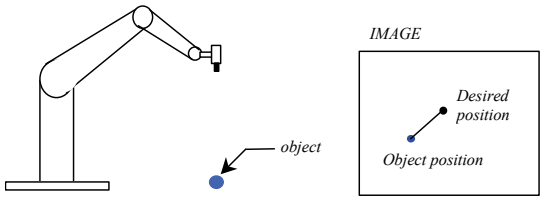


Fig. 15. Task 1 symbolic drawing.

For this task we are going to study various conventional and new methods to design the visual controller. System performance will be evaluated afterwards.

Root locus based controller

The first controller considered can be designed via root locus analysis. These controllers have the smallest performance but they also produce the smallest values of the Cartesian acceleration reference. All the transfer functions of the matrix $\underline{G}(z)$ have one pole at $z = 1$ in the open-loop chain (they are type 1 systems), so the steady state error is zero for a unit step input in all of them. Then a proportional controller is enough to control the robotic subsystem from the point of view of the steady state performance. The controller gains are chosen so that all the closed-loop poles will be real, with the intention of avoiding oscillations while obtaining the fastest possible response. The obtained controllers are given by:

$$\underline{R}(z) = diag(0.0214, \quad 0.0220 \quad 0.0221) \tag{10}$$

Pole placement controller

The second tested controller is designed by applying the pole placement methodology (Norman, 1995) that has already been used in other previous works like (Corke, 1993), (Vargas et al., 1999). In order to choose the closed-loop poles, an optimization process which minimizes the overshoot of the robot output waveform considering that the settling time would be smaller than a given n_s. The best results have been obtained with $n_s = 6$ samples. The closed loop poles for each Cartesian coordinate are located respectively at $p_1 = 0.1607$, $p_2 = -0.0366$ and $p_3 = -0.0001$. In this case, the settling interval is less than the value obtained with the proportional controller.

Optimal controller

We have also taken into account optimal control techniques to design the visual controller. In our case it has been applied as follows. Let us consider the block diagram represented in figure 16. It can be obtained from the general block diagram of figure 14 always that the transfer functions given by $\underline{G}(z)$ be linear and stable. In figure, $\underline{p}_{obj}$ represents the object position after deducing the fixed reference value of ${}^{c}\underline{p}_{d}$. Vision and robotic subsystems have been grouped into a single block whose transfer function matrix can be expressed as:

$$\underline{VG}(m,z) = \underline{V}(m,z) \cdot \underline{G}(m,z) \tag{11}$$

where modified z-transform notation is used as normal in this type of approach. Finally, $\underline{W}(z)$ has the following expression:

$$\underline{W}(z) = \frac{\underline{R}(z)}{1 + \underline{R}(z)\underline{VG}(z,m)} \tag{12}$$

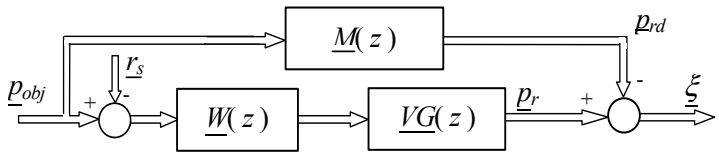


Fig. 16. Scheme used for optimal control.

The goal of the optimal control approach is to find the transfer function matrix $\underline{W}(z)$ so that it minimizes a function of the error vector $\underline{\xi}$, more specifically the integral squared error

(ISE). The error vector usually considered is $\underline{\xi} = \underline{p}_r - \underline{p}_{obj}$, the difference between the input applied and the output achieved. Obviously, the final solution will depend on the type of input chosen. For the task 1 under consideration we propose to modify the applied input by means of a smoothing transfer function matrix $\underline{M}(z)$ as indicated in figure 16. This modified input, denoted as $\underline{p}_{rd}$, tries to be a more realistic reference to be compared with the robot response than the applied input itself. Notice that high-valued step responses are not physically feasible for the robotic subsystem because they involve high torques that could surpass the maximum allowed values, provoking saturation effects in the manipulator actuators. Therefore we have tested the following transfer function matrix:

$$\underline{M}(z) = diag\left(\frac{1-a_x}{(z-a_x)\cdot z}, \frac{1-a_y}{(z-a_y)\cdot z}, \frac{1-a_z}{(z-a_z)\cdot z} \right) \tag{13}$$

where the (a_x, a_y, a_z) values are chosen to obtain the best output behavior with respect to speediness and cancellation of oscillations under the existing constrains. In the simulations the best value was equal to 0.3 for the three Cartesian coordinates. In the experimental test the same values are kept. Controller obtained is:

$$\underline{R}(z) = diag\left(\frac{0.0446z^3 + 0.0435z^2}{z^3 + 0.9849z^2 + 0.5180z + 0.0717}, \frac{0.0441z^3 + 0.0449z^2}{z^3 + 0.9777z^2 + 0.5163z + 0.0663}, \frac{0.0437z^3 + 0.0451z^2}{z^3 + 0.9754z^2 + 0.5159z + 0.0644} \right) \tag{4}$$

Deadbeat controller

Another design possibility is to use a deadbeat controller, which produces the minimum settling interval. The computation of this controller for the three coordinates offers the following transfer functions matrix:

$$\underline{R}(z) = diag\left(\frac{0.0979z^2}{z^2 + z + 0.1615}, \frac{0.0995z^2}{z^2 + z + 0.1469}, \frac{0.0995z^2}{z^2 + z + 0.1419} \right) \tag{15}$$

5.2 Task 2: Tracking of a moving object

Now the robot has to track a moving object as fast and precisely as possible. The task of the control subsystem is to keep the relative position between the end effector and the object by generating a robot movement in order to compensate the movement of the object (see figure 17).

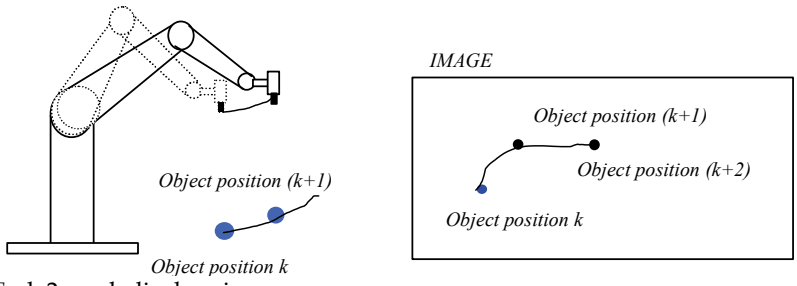


Fig. 17. Task 2 symbolic drawing.

Due to the delay introduced by the vision subsystem, in the majority of the reported experimental systems some type of estimator is used to determine the future position of the object from previous visual information (Houshangi, 1990), (Wang & Varshney, 1991),

(Westmore & Wilson, 1991), (Corke, 1993). Nevertheless, we will demonstrate in this work that it is possible to eliminate the estimator while achieving performances similar to those obtained using an estimator, just by selecting and tuning an adequate control subsystem. Therefore, we consider two types of control subsystems in this task. An estimator and a controller form the first one, and it will be called *control subsystem with estimator*. Only a controller forms the second one, and it will be called *control subsystem without estimator*. Both are analyzed in the following sections.

5.2.1 Control subsystem with estimator

The one sample delay introduced by the vision subsystem leads to a delayed sensing of the target position. Making some kind of prediction to estimate the object position one sample in advance comes out like an interesting strategy to cancel the pure delay transfer function of the vision subsystem. There are different techniques to obtain the prediction of the future position, velocity and acceleration of the object. (Brown et al., 1992) compare a set of methods for velocity estimation. We have studied several types of estimator (Bachiller, 2003), such as first and second order linear regression predictors, Kalman filter based on the assumptions constant acceleration target motion and, finally, an auto-regressive discrete time model. These estimators have been used successfully for position estimation. However, they produce considerable errors when evaluating tracking performance for sinusoidal or triangular profiles of the target motion.

Once the estimator has been selected it is still necessary to design the controller in order to complete the control subsystem (see figure 8). We have examined some techniques described in the previous task but now the pure delay transfer functions of the vision subsystems are not considered because of the existence of the estimator.

Pole placement controller

For each system a pole placement compensator is designed to place the closed loop poles at $p_{1,2}$=*-0.1532±0.7618i* for X Cartesian coordinate, $p_{1,2}$=*-0.1532±0.7618i* for Y Cartesian coordinate and $p_{1,2}$=*-0.1532±0.7618i* for Z Cartesian coordinate. The method used to place these poles is the same that has been presented in task 1.

Optimal controller

The procedure used to obtain the optimal controller in this case is the same exposed for task 1. The only difference is that for task 2 no smoothing transfer function matrix $\underline{M}(z)$ is required. The obtained controller is:

$$\underline{R}(z) = diag\left(\frac{0.181 z^3 - 0.2182 z^2 + 0.0497 z}{z^3 - 1.2661 z^2 + 0.175 z + 0.0911}, \frac{0.1817 z^3 - 0.2181 z^2 + 0.0491 z}{z^3 - 1.2804 z^2 + 0.1998 z + 0.0806}, \frac{0.181 z^3 - 0.2169 z^2 + 0.0486 z}{z^3 - 1.2852 z^2 + 0.2081 z + 0.0771} \right) \quad (16)$$

5.2.2 Control subsystem without estimator

In this work we propose a new control strategy consisting of eliminating the estimator while designing a more efficient controller. We pretend to show that similar and even better performance can be achieved by using this strategy, while some other advantages are obtained. For example, no time is consumed in evaluating the estimators and consequently the overall execution time can be reduced. Error produced in the estimation process is also eliminated. In practice, this strategy will allow us to reduce the vision period (T_v) and in addition to measure visual information more exactly.

In order to get a good tracking performance, it is necessary to cancel the steady state error in the least possible number of samples bearing in mind the torque capabilities. For this reason

we propose to design the control subsystem as a deadbeat controller system, following the conventional methodology for type of controller. Imposing as design specification that the steady state tracking error to ramp reference must be cancelled in a minimum number of samples, the transfer function matrix obtained is shown in table 1.
Notice that these transfer functions have poles outside the unit circle. In order to guarantee the stability of the output signal, we propose a modification of the design criterion to obtain the deadbeat controller in the way that the steady state tracking error must be zero in a finite number of samples. When applying the Truxal method with these specifications a higher number of unknowns than equations appear. A minimization process with restrictions is required to solve this system of equations. Specifically we have imposed the additional constraint that all the poles of the regulator have to be inside the unit circle. Table II presents the controllers finally obtained.

Minimum number of samples	$\underline{R}(z) = diag\left(\frac{0.3094z^3 - 0.2116z^2}{z^3 + z^2 - 1.651z - 0.349}, \frac{0.3094z^3 - 0.2116z^2}{z^3 + z^2 - 1.651z - 0.349}, \frac{0.3094z^3 - 0.2116z^2}{z^3 + z^2 - 1.651z - 0.349} \right)$
Finite number of samples	$\underline{R}(z) = diag\left(\frac{0.1954z^4 + 0.016z^3 - 0.1128z^2}{z^4 + z^3 - 0.6737z^2 - 1.1332z - 0.1931}, \frac{0.1954z^4 + 0.016z^3 - 0.1128z^2}{z^4 + z^3 - 0.6737z^2 - 1.1332z - 0.1931}, \frac{0.1954z^4 + 0.016z^3 - 0.1128z^2}{z^4 + z^3 - 0.6737z^2 - 1.1332z - 0.1931} \right)$

Table 1. Controllers obtained without estimator in task 2 .

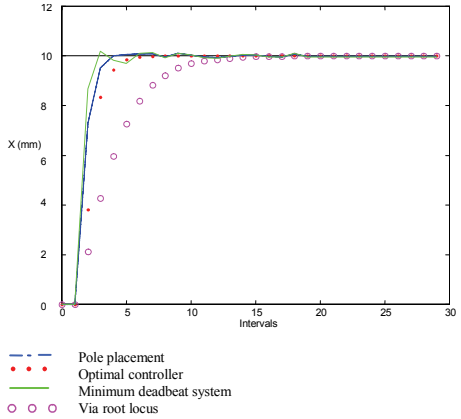
6. Third phase: Test results and performance evaluation

The next step in our plan is to evaluate the different strategies of control in order to select the most efficient controller. In this section we show how performance evaluation has been carried out in our work. We have used simulation software from MATLAB to carry out the different simulations. The following paragraphs summarize the main results obtained from simulation of the designed controllers. In general terms, plots showing the system response to the same input with the different controllers are used to compare controllers' performance.

6.1 Evaluation of the Task 1 controllers

Figure 18 shows the response associated to the X coordinate of the robotic subsystem when the input is a step of 10 millimeters. Notice that the proportional controller response is the slowest one. The three others show a similar rapidness. In order to evaluate performance some criterion must be established. The best way of doing so is to compute a set of parameters from the output signal. These parameters should give quantitative measurements to compare the behavior of the system for the different controllers used. In the present work we have chosen two significant parameters: n_s (magnitude of the settling interval) and r_a (maximum value of the Cartesian acceleration reference).
Table 2 shows the values corresponding to the controllers for the X coordinate when an input step of 10 mm is applied. No relevant differences have been observed in the two other coordinates, so they are not included here. It is noticeable that concerning the settling interval the deadbeat controller seems to be the best strategy. Nevertheless the value of Cartesian acceleration is the highest in all these solutions. The minimum value of this

parameter occurs with a conventional controller, but the settling time is excessive. A trade off is achieved with the optimal controller. Using this controller the steady state error becomes zero in 6 samples, besides the maximum Cartesian acceleration is reduced to half. It means that it can be considered the best strategy for this task.



	n_s	r_a (mm/sec²)
Classic controller	12	13.375
Pole Placement	5	53.2255
Optimal controller	6	27.8702
Dead Beat	4	61.1710

Table 2: Performance evaluation of the designed controllers for the task 1

Fig. 18. Step responses of X coordinate.

6.2 Evaluation of the Task 2 controllers

In order to evaluate performance in this task it is also necessary to consider some normalized trajectory of the mobile object. In this paper we have tested several types of object movements to evaluate the behavior of the different designed controllers. First, we have considered the target object moving with a constant velocity in one of the Cartesian axis (v_x = 62.5 mm/sec). In the second place, we have considered the target tracing a circular path of 100 mm radius, with an angular velocity of 5.2 revolutions per minute (r/min). Finally, a triangular target motion has been considered to study the behavior of the system concerning the settling time after each peak.

Figure 19 shows the tracking error of the X coordinate after deducing the fixed reference value of ${}^c\underline{p}_{dx}$, i.e. $(\underline{p}_{obj} - \underline{p}_r)_x$, when the object is moving along a circular trajectory for different combinations of estimators and controllers. Notice that the tracking error presents an initial oscillation due to the acceleration at start-up, followed by smooth tracking behavior once the controller compensates for the initial disturbances. The conclusion is that the control subsystem without estimator presents the best results for the starting up movement as well as for the tracking movement. Nevertheless, the control subsystem composed of a Kalman filter and an optimal controller presents similar results.

Once the trajectory has been chosen, tracking performance can be measured in terms of bandwidth of the closed loop system. While high bandwidth is desirable to reduce error, it can also lead to a system that is sensitive to noise and non-modeled dynamics. Other commonly used performance metrics such as settling time, overshoot and tracking errors are more appropriate. In this work, we propose to consider a quantitative criterion valid for

comparing the tracking responses corresponding to the different controllers under the same trajectory profile conditions. This parameter is the magnitude of the square root of the mean square error for each Cartesian coordinate measured in millimeters. This parameter allows for comparing tracking performance in terms of trajectory fitting. However, it would be convenient to have another criterion to compare all the control strategies. As in the task 1 case, it can be done by taking into account the values of the Cartesian acceleration reference. Notice that sometimes, due to torque capability constraints, it is necessary to limit the magnitude of the Cartesian acceleration reference so that they remain low enough.

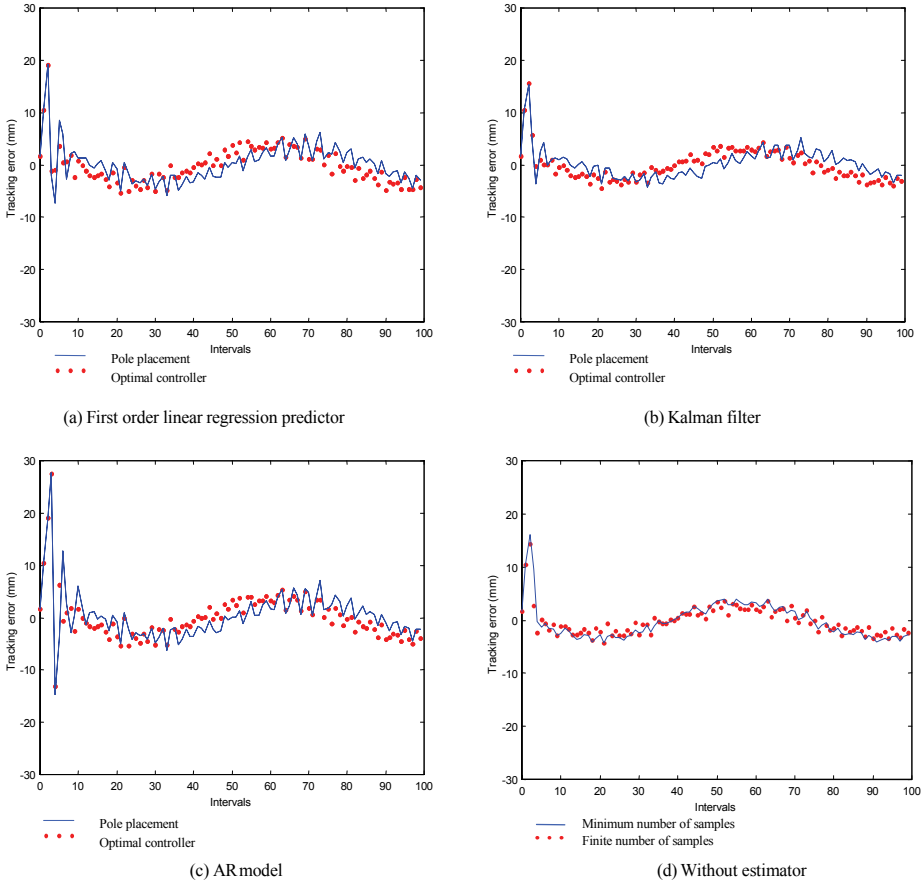


Fig. 19. X coordinate tracking error when evolving through a circular trajectory.

Table 3 shows the values obtained for the X coordinate under three different trajectories for all the control strategies, with and without estimator, considered in the previous section for this task. Table 4 shows the values of the Cartesian acceleration reference corresponding to each case. No relevant differences have been observed in the two other coordinates, so they are not included here. Analyzing the results of these tables it can be observed that, in

general, the deadbeat controller without estimator presents the best results as for tracking error in practically all the trajectory profiles just as for values of the Cartesian acceleration. Nevertheless, the control subsystem composed of the Kalman filter and an optimal controller also gives good results for tracking error although its values of Cartesian acceleration are higher.

		Ramp	**Sinusoid**	**Triangular Profile**
Pole Placement	1st L.R.	4.5895	3.6214	5.1655
	Kalman	4.4088	3.1421	5.8296
	AR	5.6858	4.8866	5.6705
Optimal	1st L.R.	2.6057	3.5623	4.7244
	Kalman	2.5803	3.3114	5.517
	AR	4.0926	4.7006	5.1734
Minimum deadbeat system		2.4259	2.5945	4.6648
Finite deadbeat system		2.5678	3.2612	5.3475

Table 3. Root of the mean square tracking error (in mm) for several controllers.

		Ramp	**Sinusoid**	**Triangular Profile**
Pole Placement	1st L.R.	225.8460	216.0358	232.7056
	Kalman	166.4976	109.3901	190.8991
	AR	424.6092	461.6932	284.0734
Optimal	1st L.R.	218.6440	209.1466	256.7252
	Kalman	161.1881	100.9431	180.5773
	AR	367.0614	356.5014	275.3864
Minimum deadbeat system		186.8742	105.0531	194.4872
Finite deadbeat system		118.0194	82.6676	126.0523

Table 4. Maximum Cartesian acceleration (mm/sec^2) for several controllers.

In conclusion, if we consider both criteria, the finite deadbeat system is the more suitable strategy for this task. This study demonstrates that the control subsystem without estimator produces competitive results besides having a number of advantages: (1) No estimator, neither more nor less complicated, is required and (2) the control subsystem is simple and of easy implementation.

7. Experimental platform validation

This final step of this work is devoted to verifying experimentally the results obtained previously by simulations, with the purpose of demonstrating its design capabilities. On the other hand, real results obtained in this step will allow comparing our methodology with other published approaches. In any case, a real platform is required to carry out such experimentation.

We have developed a visual feedback control system consisting of a Staübli RX90 manipulator, a PC, a camera and a MATROX image processing hardware. The vision computer is connected to the robotic subsystem through a RS 232 series line working to 19200 bauds and it calculates the position increment each 160 msec. However, it would be possible to reduce the value of the vision period using a more powerful image processing hardware or improving the real time image processing software. The

manipulator trajectory is controlled via the Unimate controller's Alter line that requires path control updates every 16 msec. A photograph of the experimental platform is shown in figure 20.
In all the experiments presented in this work, we have kept the reference for the Z-depth, i.e., the vertical distance from the camera to the target, at 330 mm. Furthermore, to achieve a particular motion, the object has been grasped by the end effector of a second robot, a Staubly RX60 manipulator. This allows defining any trajectory in the space as required in the second task. It also allows keeping the two-point target at constant orientation during the trajectory evolution. Camera calibration and inverse calibration algorithms have been specifically developed for the camera and environment, and are described in (Gonzalez, 1998). Anyway, the vision model used for these experiments has been explained in section 4 of this paper.

Fig. 20. Experimental setup.

The experimentation carried out over this platform will be shown and commented in the next paragraphs. In general, although all the presented controllers have been tested on the platform, only the results corresponding to the best control strategy selected in the previous step will be shown and studied here.

7.1 Experimental results of Task 1

In the static object case we have demonstrated with simulations that the best control strategy was the optimal controller. This regulator has been implemented on the platform and the final pose of the robotic manipulator with respect to the target object for a step input is achieved in 6 samples (0.96 sec.). Besides it produces small values of the Cartesian acceleration reference avoiding saturation of joint motors, contrary to the deadbeat controller. Figure 21 shows the plot of the error relative to the X coordinate registered by the vision subsystem when the input is a step of 40 mm.

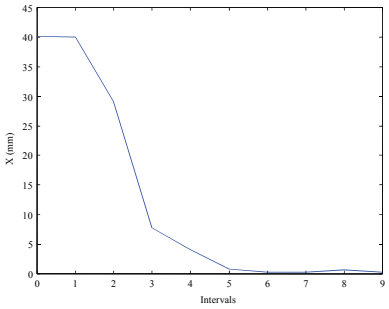
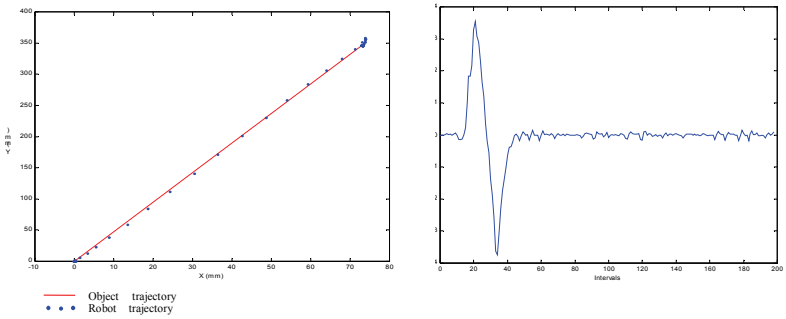


Fig. 21. Error registered by the vision subsystem when the input is a step of 40 mm.

7.2 Experimental results of Task 2

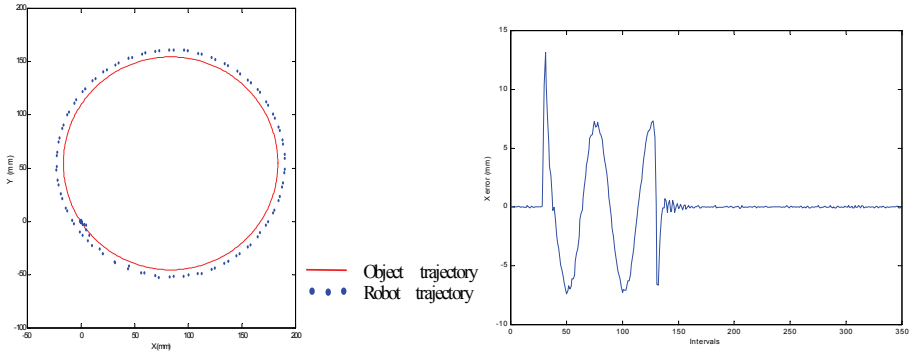
We have accomplished multiple experimental runs for tracking the target object following unknown translation motions into the perpendicular plane of the camera. During these experiments, we have tested the system using mainly linear and curved trajectories for the target.

The first type of experiments was conducted using an object moving along a linear trajectory. The system was able to track successfully the target moving at 185 mm/sec. Figure 22 shows a plot of this experiment relative to the X coordinate of the movement. The next set of experiments was performed making the target follow a circular path with a radius of 150 mm. This path produces a smooth variation of the velocity curve. The system was able to track the object moving with a circular speed of 7.5 radians/min exhibiting a tracking error of ±17 pixels, equivalent to ±7 mm at a depth of 330mm. These results are shown in figure 23. Finally, figure 24 shows an example of a trajectory in the 3D space. The object is moving along a spiral path composed by a circular movement in the XY plane and a linear movement in the Z direction.



(a) Object and robot linear trajectory (b) X coordinate tracking error

Fig. 22. Measured tracking performance for target moving along a linear trajectory with the finite deadbeat system.



(a) Object and robot circular trajectory (b) X coordinate tracking error

Fig. 23. Measured tracking performance for target moving along a circular trajectory with the finite deadbeat system.

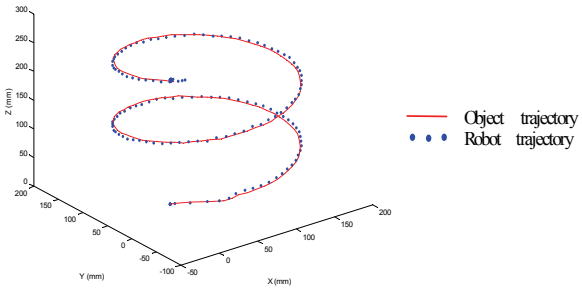


Fig. 24. Object and robot spiral trajectory.

8. Conclusions and new tendencies

This article has presented a complete working plan for designing and performance evaluation of position based dynamic look and move systems using a 6 DOF industrial manipulator and a camera mounted on its end effector. The robot used works using the ALTER line facility, allowing a direct and uncoupled control in Cartesian coordinates.

In the first stage, the dynamic models proposed for the robotic and vision subsystems have been presented, identified and validated by experimental tests. Special attention has been paid to describe the vision model, which has been optimized to cope only with the three-dimensional Cartesian position measurement, in order to reduce the total on-line computation time. The overall block diagram has also been discussed. It corresponds to a multirate control system scheme and a method to calculate its equivalent control scheme at the vision sampling period which has been proposed and applied.

The design of diverse control strategies has been considered in a second stage. Two different control tasks have been taken into account, concerning respectively the tracking of a static object and of a moving one. In both cases, several controllers have been designed and a comparative study of the different behaviors of the overall system has been presented. It is particularly remarkable the optimal control proposal made for task 1. It has been designed so that the possible saturation effects in the manipulator actuators are taken into account by introducing a smoothing transfer function, which is estimated in the optimization process. Simulations of all the considered controllers have been carried out. Selection of the most efficient control strategy for each task has been possible using this strategy. The selected controllers have been implemented and validated on an experimental platform. Let us remark that these controllers are very simple, and consequently more than satisfactory for real time implementation, whereas they achieve high performance functionality.

Finally this work has shown that, following an adequate design methodology like the one presented, a well-designed controller without using estimator can be more efficient than the combination of the estimator plus a classical controller in the considered tracking problem. Specifically, it has been demonstrated that the control subsystem without estimator improves the behavior of the entire system output as for tracking error and value of Cartesian acceleration. The results presented in this paper demonstrate the effectiveness of the designed position based dynamic look and move system to track both static and moving objects.

The visual servoing structure used in this work has the characteristic of being integrated by independent subsystems with its corresponding advantages. Its principal disadvantage is that the torque vector acting over the robot's motors does not appear explicitly and, therefore, it is not possible to know when a referenced movement can saturate any of the motors. For this reason, future experiments will focus on aspects as the study of different schemes that could maintain the exerted torque small and, in any case, under control.
Only one camera has been used in this work, considering that the information supplied such vision subsystem does not introduce uncertainties because of its simplicity. However, when the robot is operating in an unstructured environment, such sensor information and such approach are not always satisfactory. The problem increases when three-dimensional information must be managed, like in the multiple cameras case.
From this point of view, other visual servoing classification not considered in this work should be taken into account, having in mind the number of sensors used. Following cases could be distinguished: (1) monocular visual servoing that uses one camera which can be a fixed camera (fixed camera configuration) or mounted at the end effector the robot (eye-in-hand configuration); (2) multi camera vision system where multiple cameras placed in the work-space are used to get the task specific information. The monocular visual servoing is the simplest solution; however, the depth information of the work-space is lost. This information can be calculated when the vision system uses multiple cameras but not always is possible to extract all the 6-DOF information. Additional difficulties appear when the extrinsic camera parameters (position and orientation of the camera within the work-space) are inexactly obtained due to uncertainties in the camera calibration process.
Motivated by this, the focus in visual servoing has shifted towards 3D problems and compensating the uncertainties. Saedan & Ang (Saedan & Ang, 2002) design a PI controller based 3D pose servoing scheme. This system gives accurate position and orientation control for stacionary target applications, but was unable to avoid error fluctuations for tracking problems. Sahin & Zergeroglu (Sahin & Zergeroglu, 2005) use an adaptative controller based 3D position servoing. This controller is designed to compensate the uncertainties associated with the mechanical parameters of the robot manipulador and intrisic parameters of the cameras. However, this work only presents simulation results to illustrate the performance of the proposed controller. Another solution (Malis, 2004) presents a new visual servoing scheme which is invariant to changes in camera intrinsic parameters. This work shows how to position a camera, with respect to non-planar object, even if the intrinsic parameters change.

9. Acknowledgements

This work has been supported by the Spanish Government under contract through the DPI2005-03769 project, and by the Autonomous Community of Madrid under the RoboCity2030 project, referenced as S-0505/DPI/0176.

10. References

Allen, P. K.; Timcenko, A.; Yoshimi, B. & Michelman, P. (1993). Automated tracking and grasping of a moving object with a robotic hand-eye system, *IEEE Transactions on Robotics and Automation*, Vol. 9, No. 2, 1993, pp. 152-165.

Bachiller, M; Cerrada, J. A. & Cerrada, C. (2003). A Modular Scheme for Controller Design and Performance Evaluation in 3D Visual Servoing, *Journal of Intelligent & Robotic*

Systems, Vol. 36, No. 3, Marzo 2003, pp. 235-264.

Brown, R.; Schneider, S. & Mulligan, M. M. (1992). Analysis of algorithms for velocity estimation from discrete position versus time data, *IEEE Transactions Industrial Electronics*, Vol. 39, No. 1, 1992, pp. 11-19.

Corke, P. I. (1993). Visual control of robot manipulators –A review in Visual Servoing, K. Hashimoto, Ed. Singapore: World Scientific.

Chaumette, F.; Rives, P. & Espiau, B. (1991). Positioning of a robot with respect to an object, tracking it and estimating its velocity by visual servoing, *Proceedings of IEEE Int. Conf. on Robotics and Automation*, 1991, pp. 2248-2253.

Craig, J. J. (1989). Introduction to robotics. Mechanics and control. *Addison-Wesley Publishing Company.*

Espiau, B.; Chaumette, F. & Rives, P. (1992). A new approach to visual servoing in robotics, *IEEE Transactions on Robotics and Automation*, Vol 8, 1992, pp. 313-326.

Feddema, J. T. & Mitchell, O. R. (1989). Vision guided servoing with feature based trajectory generation, *IEEE Transactions on Robotics and Automation*, Vol 5, 1989, pp. 691-700.

Feliu, V. (1986). A Transformation Algorithm for Estimating System Laplace Transform from Sampled Data, *IEEE Transactions on Systems, Man, and Cybernetics*, 1986, pp 168-173.

Gangloff, J. A.; Mathelin, M. & Abba, G. (1998). 6 dof high speed dynamic visual servoing using gpc controllers, *Proceedings of the 1998 IEEE Int. Conf. on Robotics and Automation*, 1998, pp. 2008-2013.

Gonzalez, Y. (1998). Aportaciones a la calibración de cámaras con distorsión geométrica. Aplicación al control con realimentación visual, *PhD thesis* UNED, 1998.

Hashimoto, K.; Ebine, T. & Kimura, H. (1996). Visual servoing with hand eye manipulator optimal control approach, *IEEE Transactions on Robotics and Automation*, 1996, pp. 766-774.

Houshangi, N. (1990). Control of robotic manipulator to grasp a moving target using vision, *In Proceedings of. IEEE Int. Conf. Robotics and Automation*, 1990, pp. 604-609.

Khosla, P. K.; Papanikolopoulos, N. P. & Kanade, T. (1993). Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision, *IEEE Transactions on Robotics and Automation*, Vol. 9, 1993, pp. 14-35.

Koivo, A. J. & Houshangi, N. (1991). Real time vision feedback for servoing robotic manipulator with self tuning controller, *IEEE Transactions on Systems, Man, Cybernetics*, Vol. 1, No. 1, 1991, pp. 134-141.

Malis, E.; Chaumette, F. & Boudet, S. (1999). 2-1/2-d visual servoing. *IEEE Transactions on Robotics and Automation*, Vol. 15, No. 2, April, 1999, pp. 1352-1359.

Malis, E. (2004). Improving vision-based control using efficient second-order minimization techniques. *IEEE Int. Conf. on Robotics and Automation*, 2004.

Nelson, B. & Papanikolopoulos, N. (1998). Special Issue on Visual Servoing, *IEEE Robotics and Automation Magazine,* Vol. 5, No. 4, December, 1998.

Norman, S. N. (1992). Control Systems Engineering. Addison-Wesley, 1995.

Papanikolopoulos, N. P. (1992). Controlled Active Vision, *PhD thesis, Carnegie Mellon University*, 1992.

Petersson, L.; Jensfelt, P.; Tell, D.; Strandberg, M.; Kragic, D. & Christensen, H. I. (2002). System integration for real-world manipulation task. *Proceedings of IEEE Int. Conf. Robotics and Automation*, 2002, pp. 2500-2505.

Saedan, M. & Ang, M. H. (2002). 3D Vision-Based Control On An Industrial Robot. *Proceedings of IASTED International Conference on Robotics and Applications,* Florida, USA, November, 2002, pp. 19-22.

SaHin, H. T. & Zergeroglu, E. (2005). Adaptative Visual Servo Control of Robot Manipulators via Composite Camera Inputs. *Fifth International Workshop on Robot Motion and Control ROMOCO05,* Dymaczewo-Poland, June, 2005, pp.219-224.

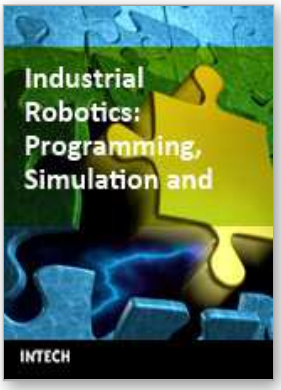
Vargas, M.; Rubio, F. R. & Malpesa, A. R. (1999). Pose-estimation and control in a 3D visual servoing system, *IFAC,* 1999, pp. 317-323.

Wang, T. C. & Varshney, P. K. (1991) A measurement preprocessing approach for target tracking, *IFAC*, Vol. 7, 1991, pp. 199-202.

Weiss, L. (1984). Dynamic Visual Servo Control of Robots: an Adaptative Image Based Approach, *PhD Thesis, Carnegie Mellon University*, 1984.

Westmore, D. B. & Wilson, W. J. (1991). Direct dynamic control of a robot using an end point mounted camera and Kalman filter position estimation, *Proceedings of IEEE Int. Conf. Robotics and Automation*, 1991, pp. 2376-2384.

Wilson, W.; Williams, C. C. & Bell, G. S. (1996). Relative end-effector control using Cartesian position based visual servoing, *IEEE Transactions on Robotics and Automation*, Vol. 12, No. 5, 1996, pp. 684-696.



Industrial Robotics: Programming, Simulation and Applications
Edited by Low Kin Huat

ISBN 3-86611-286-6
Hard cover, 702 pages
Publisher Pro Literatur Verlag, Germany / ARS, Austria
Published online 01, December, 2006
Published in print edition December, 2006

This book covers a wide range of topics relating to advanced industrial robotics, sensors and automation technologies. Although being highly technical and complex in nature, the papers presented in this book represent some of the latest cutting edge technologies and advancements in industrial robotics technology. This book covers topics such as networking, properties of manipulators, forward and inverse robot arm kinematics, motion path-planning, machine vision and many other practical topics too numerous to list here. The authors and editor of this book wish to inspire people, especially young ones, to get involved with robotic and mechatronic engineering technology and to develop new and exciting practical applications, perhaps using the ideas and concepts presented herein.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821