ELSEVIER

IMAVIS 1086T

Image and Vision Computing 14 (1996) 715–732

image AND vision COMPUTING

Promising research
Vision-based robot positioning using neural networks

Gordon Wells[a,*], Christophe Venaille[b], Carme Torras[a]

[a]*Instituto de Cibernética, Diagonal 647, Barcelona 08028, Spain*
[b]*Thomson Broadcast Systems, Laboratoires Electroniques de Rennes, Avenue de Belle Fontaine, 35510 Cesson-Sévigé, France*

Received 12 June 1995; revised 26 January 1996

Abstract

Most vision-based robot positioning techniques rely on analytical formulations of the relationship between the robot pose and the projected image coordinates of several geometric features of the observed scene. This usually requires that several simple features such as points, lines or circles be visible in the image, which must either be unoccluded in multiple views or else part of a 3D model. Feature-matching algorithms, camera calibration, models of the camera geometry and object feature relationships are also necessary for pose determination. These steps are often computationally intensive and error-prone, and the complexity of the resulting formulations often limits the number of controllable degrees of freedom. We provide a comparative survey of existing visual robot positioning methods, and present a new technique based on neural learning and global image descriptors which overcomes many of these limitations. A feedforward neural network is used to learn the complex implicit relationship between the pose displacements of a 6-dof robot and the observed variations in global descriptors of the image, such as geometric moments and Fourier descriptors. The trained network may then be used to move the robot from arbitrary initial positions to a desired pose with respect to the observed scene. The method is shown to be capable of positioning an industrial robot with respect to a variety of complex objects with an acceptable precision for an industrial inspection application, and could be useful in other real-world tasks such as grasping, assembly and navigation.

Keywords: Visual servoing; Robot control; Neural networks; Image features

1. Introduction

A long-standing goal in the field of robotics has been to endow robots with more sophisticated position control capabilities through the use of vision feedback. In theory, a robot with vision would be able to approach, track and grasp unknown objects in arbitrary locations, assemble parts and avoid unexpected obstacles, even despite deformations and miscalibrations in its mechanical structure. The ideal system can be imagined as that which allows manipulating arbitrary objects in real environments, without having to model the object, the environment or the camera, which needs no calibration, and which can track moving objects in real time.

Present technology still seems far from achieving these goals. Despite the growing volume of published works describing visual control methods and systems, most of them have not evolved far beyond the 'laboratory' problem: for simplified, invariant objects and environments. With the exception of several commercial bin-picking systems, the few existing industrial systems are generally aimed at repetitive tasks in simplified environments, such as welding metal and applying silicone beads.

One of the main reasons for the limitations of many visual control systems is the persistent use of very simple, unrobust vision techniques. Typically, a few points, lines or other geometric object characteristics are used as image features, whose reliable extraction largely depends on the shape of the object and its surroundings, and often requires inserting specially designed visual cues into the scene. Assuming such a set of local features may be obtained, precise positioning can often be achieved. Few geometrical assumptions may be made with regard to arbitrary scenes in real environments, however, where scenes are generally complex, difficult to model, and partially occluded depending on the viewpoint.

* E-mail: wells@ic.upc.es

PII S0262-8856(96)0186-4

To translate geometric features into robot movements, models of the object and the camera are usually required, the camera must be calibrated, and the features in two or more images must be matched in order to estimate the relative coordinates of the robot and the object. These tasks are often burdensome, computationally intensive and prone to estimation errors.

In the present work, a visual positioning method is proposed based on neural learning and global image descriptors, with the aim of overcoming some of these limitations and making an important advancement towards the use of visual robot positioning in real environments.

1.1. Vision-guided robots

The basic task of visual positioning is to control the pose of a robot end-effector using information extracted from images of the robot's workspace. The robot pose is a six-element vector representing the position and orientation of the end-effector in 3D space. In the case of mobile robots, the position and orientation of the robot itself must be controlled. In general, the aim is to achieve a desired pose relative to one or more objects or cues viewed in the image, for the purpose of inspecting, grasping or manipulating them, or else to guide robot navigation through the environment. If the object is stationary, the task is simply one of static positioning. For moving objects, the object must be dynamically tracked so as to intercept or achieve a stable pose relative to it before its manipulation.

Visual control has been studied extensively for industrial robot arms. While 6-dof control is often sought, visual servoing of just two or three degrees of freedom is often sufficient in some applications, and can greatly simplify controller design. Vision-guided control has also been applied to mobile robot navigation.

Visual information may be obtained from one or more cameras. While using a single camera is clearly advantageous in terms of hardware and computational costs, having two or more cameras makes it possible to acquire depth and pose information about unknown objects, through the use of stereo matching algorithms, for example.

The camera or cameras may be mounted either on the robot's end-effector or else at some fixed location in the environment. Arm-mounted cameras have the advantages of proximity to the task being performed, and mobility, allowing attention to be focused as needed so as to avoid occlusion, resolve ambiguity, and increase accuracy. This can be a drawback, however, since the field of view depends on the arm location and orientation, and the focus can change as the camera approaches the workpiece. Arm-mounted cameras also add weight to the manipulator, and are subject to collisions. Externally mounted cameras place fewer constraints on motion planning, and may be oriented so as to maximize their field of view. For large workspaces, pan-tilt mechanisms or multiple cameras may be necessary, at the expense of calibration accuracy. Also, many movement operations may be easily formulated in terms of the image by virtue of the camera moving with the end-effector.

1.2. Previous work

The existing literature on vision-based robot control is quite extensive, with some of the earliest works dating back to the early 1970s. Authors have proposed many working systems and approaches to specific theoretical and technical aspects in related areas of vision processing and control, such as stereo vision, pose and depth estimation, camera models and calibration techniques, image feature selection and processing, trajectory generation, adaptive control, and dynamic stability analysis. A comprehensive literature review may be found in the work by Corke [1].

Despite the significant advances being made in the field, very few real industrial applications of vision-guided robotic systems have so far been reported. Most have been research-oriented and typically only tested in simulation or on toy problems in simplified laboratory environments. Examples include picking up a toy train, inverted pendulum balancing, ball catching, juggling and ping-pong playing. (The latter three, although complex dynamic tasks overall, may be considered as simplified positioning tasks in the sense that they involve only trajectory tracking of the centroid of a moving ball, and its interception with a relatively large paddle or net, as opposed to precise 6D positioning of a gripper relative to a complex scene.) A few industrial applications described are automated inspection [2], welding, sealant application, docking, conveyor-belt picking, part mating and fruit harvesting. References may be found in Corke [1]. Successful results for road-vehicle guidance have been reported by Dickmanns et al. [3] and Masaki et al. [4]. Feddema [5] discusses the current status of commercialization of visual servoing technology and speculates on potential application areas in the near future.

The current lack of application-driven systems is perhaps because vision-guided control still has not been proven sufficiently reliable, robust, useful and cost-effective for many real-world problems, as postulated by Feddema [5]. While cost is destined to decrease as hardware is improved, increased reliability and robustness will depend primarily on a more sophisticated use of vision. Firstly, better techniques must be developed for extracting information useful for robot control from complex, real-world images. Secondly, these methods must be integrated into complete vision and control packages including high-level modules for object recognition, environmental modelling, and task and

motion planning. An excellent overview of current visual servoing technology and open issues may be found in the work by Hagar and Hutchinson [6].

Existing works in vision-guided robot control may be classified in a number of ways. Here, we will discuss the state of the art within the context of three important issues common to all approaches: (1) the type of image feature information used, (2) the visual feedback representation, and (3) the control scheme.

1.2.1. Image features

Practically all works to date have made use of simple geometric features extracted from images: points, lines, circles, squares, region areas, etc. The use of point features appears to be most common, where points correspond to object corners, holes, centroids of objects or regions, or specially designed target cues placed in the scene. Four or more points are generally used, which is the minimum number necessary to uniquely determine the pose of an arbitrary object [7]. Point features are used for pose determination in the works of Abidi and Gonzalez [8], and Mandel and Duffie [9].

The projection variations of a circle pattern were used by Kabuka and Arenas [10] in a robot docking application, and the centroid and diameter of circles was used by Harrell et al. [11] for fruit tracking with an orange-harvesting robot. Vanishing points (intersection of two nearly parallel lines) and line orientations were used by Zhang et al. [12] for robot navigation. Chaumette et al. [13a,13b] and Espiau et al. [13c] derived variations of a tracking method for points, circles and lines. Weiss et al. [14] studied visual control using the center of gravity, area and relative region area of simulated images of polyhedral objects.

Although approaches based on geometric features have produced useful results, it cannot be assumed that a set of simple geometric features can always be reliably obtained from images encountered in many real-world situations. Object shape and texture, occlusion, noise and lighting conditions have a large effect on feature visibility. The use of more global image characteristics would therefore seem like a robust alternative to geometric features. Recently, a number of authors have reported the use of more global types of image features in visual control applications. Optic flow [15] has been applied by Allen et al. [16] and Papanikolopoulos [17]. Corke [18] and Andersson [19] use first-order geometric moments to compute the target centroid. In all these cases, however, only a reduced number of degrees of freedom could be controlled. Bien et al. [13d] and Jang et al. [20] showed how a number of different global image features could be used within a general visual control framework. Listed features included geometric moments, image projections on a line, random transforms, template matching and Fourier transforms. Results are given only for simplified tracking tasks using as features the target centroid and area.

Perhaps the most interesting recent work is that of Nayar et al. [21]. Using Principal Component Analysis, an efficient method was developed for compressing a set of training images into a subspace of eigenvectors using the pixel gray levels directly as image features. Using a B-spline interpolation of the same set of images projected onto this eigenspace, a hypersurface of all possible images of the object is generated, parametrized with respect to their coordinates, which allows determining with good precision the coordinates of images taken from arbitrary locations. The method was demonstrated for complex, real images, although for only three degrees of freedom.

Our work demonstrates how global image encodings such as Fourier descriptors and geometric moments may be applied to complex, real images to effectively position a 6-dof industrial robot.

1.2.2. Visual feedback representation

Robots may be controlled either by Cartesian movement commands or direct joint commands. Consequently, extracted feature information must be converted into one of these two forms of feedback. It is mainly the complexity of obtaining this transformation which has given rise to the wide variety of approaches to vision-based control. Sanderson and Weiss et al. [22] distinguished between position-based and image-based methods, depending on the chosen feedback representation.

In position-based methods, the feature information is used to compute the relative pose of the robot with respect to the target object. Cartesian movement commands are then generated from the pose information. One advantage of this approach is that the problem of vision processing is decoupled from that of manipulator control, allowing each to be studied separately. Another is that the pose estimations may be verified, and the generated trajectories may be checked for collisions or kinematic irregularities. The main disadvantages cited are sensitivity to image noise and computational cost due to the additional image analysis and inverse kinematic calculations required. Another drawback is the need for camera modelling and calibration to obtain the feature projection transformation and the camera-wrist relationship. Inaccuracies in the camera model are often a source of control error. While camera calibration techniques such as that of Tsai and Lenz [23] may be used to obtain the eye-hand relationship, they add to the overall computational complexity of the system and can also introduce inaccuracies.

Researchers have explored a number of different methods for computing the robot pose from feature information. Perhaps the most straightforward is to derive the analytical transformation between the projections of several object points on the image plane and their 3D pose, using the known relationship between the object points themselves, and a projection model of the camera. This problem is the subject of so-called

photogrammetric techniques [24,25]. The works of Abidi and Gonzalez [8], and Mandel and Duffie [9] are typical examples. Similar transformations have been derived for other geometric features such as lines or circles [10].

Another approach to 3D pose determination is to use 2D projection transformations together with depth-estimation techniques. Stereo vision algorithms have been used for visual servoing by Rizzi and Koditschek [26], Andersson [19] and Hagar et al. [27]. In a similar way, depth information may also be derived from sequential views from a single camera, using techniques known as monocular stereo, motion stereo or depth from motion. Examples include the works of Papanikolopoulos [17] and Vernon and Tistarelli [28].

In so-called image-based methods, the observed feature information is used directly as feedback in the manipulator control law; no pose estimation is performed. The approach generally taken is to estimate the matrix relating changes in the camera pose to feature variations in the image plane, often referred to as the feature Jacobian or image Jacobian matrix. Although this relationship is highly non-linear, it is typically linearized within a small range of the current estimation point, and updated continuously as the robot moves through the workspace. The inverse of the feature Jacobian is used in a control law to compute the camera movements needed to reduce the error between the desired and observed image features. The transformation of camera movements to joint commands is obtained by way of the camera-wrist transformation and the manipulator kinematics.

Methods for computing the image Jacobian include empirical solution [13,20,29] and estimation within adaptive control schemes [14,17,30]. While Jacobian-based methods have produced useful results in a number of applications, they impose a similar computation burden to that of pose-estimation methods, since the image Jacobian must be continuously updated and inverted.

An attractive approach is to have a system which *learns* the nonlinear relationship between the observed 2D feature deviations and the robot movements. Skaar et al. [31] developed a method for learning the image Jacobian, by way of least-squares estimation, from several observations of cues along the approach trajectory. The method was successfully applied to a part-mating task. Neural networks have been applied in many areas of robot control, as described by Torras [32]. Hashimoto et al. [33] used a neural network to learn the direct mapping between the image deviations of four feature points and the joint angles of a 6-dof manipulator. A disadvantage to including the inverse kinematics in the mapping is that the learned relationship is pose-dependent, i.e. it only applies for positioning with respect to the target object in a particular location. In the present work, a neural network is used to learn the pose-independent mapping between feature deviations and pose changes based on images sampled from the workspace.

1.2.3. The control scheme

The speed of robot response is often an important consideration, especially in applications in which the robot must track and interact with fast-moving objects. In the past, the rate at which image processing could be performed with existing hardware was a limiting factor in the working speed of most visual robot control systems. Typically, robots were required to stop for several seconds after each movement iteration to allow for the acquisition and processing of each new image. Such systems are frequently referred to as 'static look-and-move' structures, following the classification introduced by Sanderson and Weiss [22]. They distinguished between 'static' and 'dynamic' systems, indicating either asynchronous or synchronous timing of image processing and manipulator control, and between 'look-and-move' and visual-servoing' structures, depending on whether the joints are directly controlled by vision feedback alone, or in parallel with closed-loop joint controllers.

While static look-and-move control is sufficient in some applications, real-time dynamic visual servoing is the focus of much of today's research in visual control. Achieving real-time performance requires consideration of a number of difficult control issues, however, since the robot's own dynamics come into play at high speeds, greatly affecting tracking precision and stability. This is especially true if the joints are controlled by vision feedback alone. Dynamic issues in visual control are discussed by Corke [34]. Robot dynamics are typically highly non-linear due to frictional and inertial effects, thus complicating controller design. Adaptive control schemes are used in some approaches as a means of compensating for unknown and changing system parameters [14,17,30,35]. Neural control has also been successfully applied for on-line learning of unknown robot dynamics and target trajectory prediction by Miller [36] and Cembrano et al. [37]. A common problem is providing visual feedback at a sufficiently high rate to the joint controllers, which generally have very high sampling rates. Image processing rates being significantly slower in most cases, feedback data are often supplemented using trajectory prediction and planning techniques [16,38]. To avoid the additional image analysis time required for pose estimation, direct image-based feedback is often used in visual servoing systems. A common difficulty is choosing the appropriate features to control each joint, since feature-pose relationships are nonlinear and highly coupled. Typical solutions include the use of feature selection schemes [38] and controlling only a reduced number of degrees of freedom. Nevertheless, the advent of faster computing hardware is making high-bandwidth, real-time image processing increasingly feasible.

1.3. Our approach

In this work, we present a neural-network-based visual

positioning system using global image descriptors of real-world images. The main result is to demonstrate how neural networks may be used with global image features such as Fourier descriptors and geometric moments to effectively position a 6-dof robot in a real industrial environment. Neural networks are especially suited for learning complex nonlinear functions for which no explicit analytical formulation is available. Here, a feedforward neural network is used to learn the nonlinear relationship between variations in global descriptors of the images observed from a wrist-mounted camera, and changes in robot pose. The complete transformation is learned directly, with no need for camera calibration, knowledge of the camera's intrinsic parameters, or estimation of other intermediate transformations. Furthermore, the use of global image descriptors makes the method applicable to real-world images in which simple geometric features may be difficult to extract, and eliminates the need for feature matching between the reference and observed images. This represents a novel and more advantageous use of the function-approximation capability of neural networks for visual robot control than in previous neural approaches [33,36], in which only point features were used.

The network is trained using a series of example images taken at known random displacements from a prespecified pose, relative to the scene, at which a desired 'reference' image is observed. It is then used as an image-based controller to generate cartesian movement commands with which to return the robot to the reference pose from any arbitrary initial position. Since the main concern was learning the feature-command relationship for static scenes, dynamic control aspects were not considered; a static look-and-move-scheme was used. However, the method could feasibly be used for dynamic tracking. The present system was developed primarily for an industrial inspection application being developed within the ESPRIT project 'CONNY'. Nevertheless, the techniques employed are applicable to other vision-based control systems.

A first set of experiments involved simulated images containing four point features. This allowed preliminary testing of the neural-network approach, and comparison with previous analytical and neural methods using similar images [33]. Neural networks trained on four point features were used later, in conjunction with a correlation matching method, to enhance the final positioning accuracy following servoing with geometric-moment descriptors.

Subsequently, the use of global image descriptors as network inputs was explored. Fourier descriptors were used to encode the external contour of observed objects, and their variations with respect to those of a reference image were used as network inputs. These tests demonstrated the feasibility of using non-geometric features, and eliminated the need for feature matching between the observed and reference images. Positioning results using an industrial robot are presented.

By encoding images with geometric moments, the method was extended to complex images in which a predominant closed contour may not be present. A series of descriptors based on low and high-order moments was formulated which effectively quantifies the image variations produced by 6-dof robot movements. The variations in the descriptors produced by camera displacements were used to train a feedforward network as before. Results are presented for the industrial robot.

Final positioning accuracy was increased by way of a second positioning step using neural networks trained on four point features, using a correlation method for feature matching. Results from a series of positioning trials using the industrial robot are given.

2. Four point features

2.1. Image generation

For initial experiments, a simulated reference image was generated consisting of four point features. These points can be imagined as four distinct features extracted from the image of a real object, and were chosen based on the assumption that four points suffice for determining the position and orientation of an arbitrary 3D object (see, for example, Horaud et al. [7] Abidi and Gonzalez [8]). The four feature points were arranged to form a square of width 100 mm in the simulated 'scene', as shown in Fig. 1. The reference position chosen was with the camera's viewing axis perpendicular to the center of the square formed by the four points, at a distance of 750 mm.

A software simulator was developed to model the projected images that would be seen from a CCD camera mounted on a robot arm. This 'moveable-camera simulator' allowed the rapid generation of realistic training sets for varying object and camera characteristics, and the subsequent testing of neural networks trained on these data by simulating the movements the robot would make in 3D space to approach the pre-defined reference position and displaying the resulting images. The model used for the simulated camera was the well-known 'pinhole' projection model, as illustrated in Fig. 1. The model equations may be found in Abidi and Gonzalez [8].

2.2. Neural network training

Training examples were generated by choosing 1000 random camera poses within the ranges −50 to 50 mm translation and −3 to 3 degrees rotation for all three coordinate axes (which ensure that the four points are always visible in the resulting images) and generating the

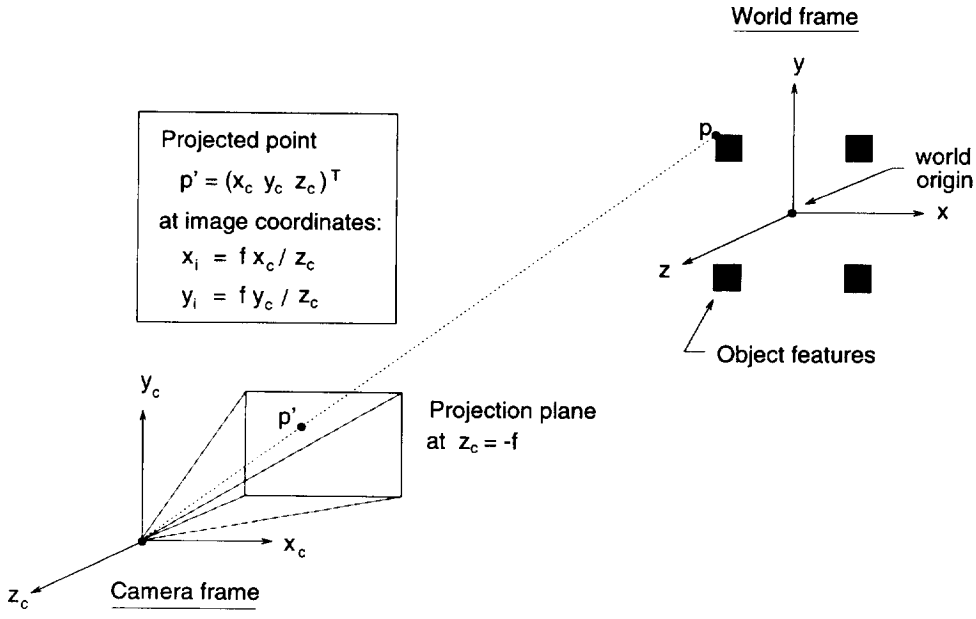

Fig. 1. The pinhole projection camera model used to generate simulated 4-point images.

corresponding images of the four projected feature points using the simulated camera program. The x, y-coordinates in the reference image of the four points were subtracted from those in the observed image to produce 8 feature 'offsets', which were used as the inputs to the neural network. The desired outputs of the neural network were the 6 cartesian coordinates (position and orientation) of the simulated camera, with respect to the reference frame, from which the observed image was viewed. An additional test set of 250 randomly generated images was used for training validation.

These data were used to train a 3-layer backpropagation network with eight inputs, 30 hidden units and six outputs. (An introduction to neural networks, including backpropagation learning, may be found in [13a] and the work by Wells [39]). The network was used with a tanh activation function in the hidden layer, a linear output-layer transfer function, and the delta learning rule. Training was continued until no further decrease in the RMS output error for the test set could be achieved, which occurred after approximately 200,000 training iterations. The final RMS error was 0.18.

2.3. Visual positioning tests

Once trained, the network was used in an iterative visual positioning scheme to test its ability to guide the camera back to the reference position from any given initial position and orientation. This process is illustrated in Fig. 2. The camera is first placed at a randomly selected initial position in the vicinity of the reference position. The feature offsets are computed for the four points in the resulting image and input to the neural network. The network outputs a vector of six coordinates representing its estimate of the current camera position and orientation. The camera is then 'moved' toward the reference position in the negative direction of these coordinates. The new image and feature offsets are computed at this new camera position. The average relative change in the image offsets is then calculated as the absolute value of the difference between the averages of the eight offsets in the current and previous iterations. If this value is below a prespecified convergence threshold,

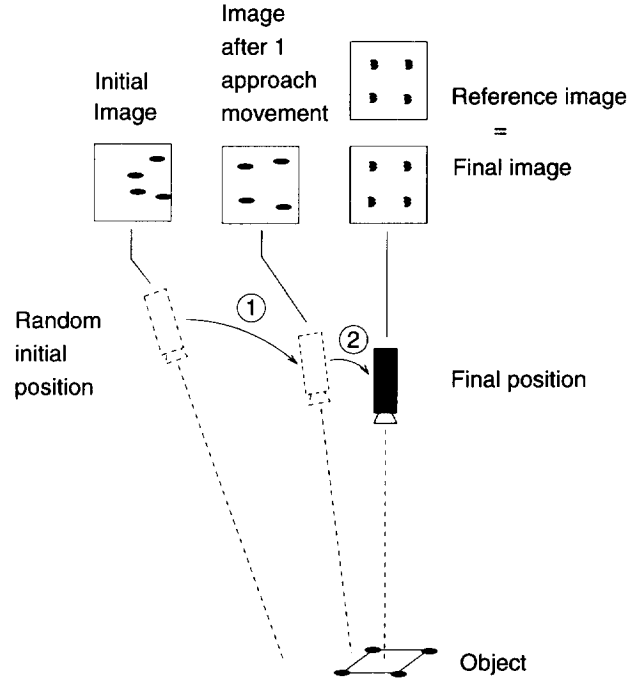

Fig. 2. Visual positioning of a robot-mounted camera using iterative approach movements.

the process is stopped. Otherwise, approach movements are continued until this condition is met.

In initial tests, it was observed that the method often converged to positions in which the observed image very closely resembled the reference image (2D error < 0.5 pixel), but which were somewhat displaced from the reference position (3D translation error ≤ 12 mm, rotation error ≤ 1.5 degrees). Upon closer inspection, it was discovered that many images resulting from translations on either the x or y axis were almost identical to those resulting from rotations around the opposite axis. In fact, it may be demonstrated analytically [9] that multiple solutions exist for pose determination based on four planar points. This ambiguity in the data accounted for the relatively poor training results, and for the significant 3D positioning errors obtained in the tests, a problem which was solved by simply moving one of the four points perpendicularly back away from the plane of the other three by a distance of 100 mm, giving more relative depth to the four features of the simulated 'object'. Using the image data base computed with this new feature arrangement, the network training time was reduced to only 80,000 iterations, with a final RMS output error of 0.03. This improved feature arrangement was maintained to produce the test results shown here.

Generalization test results are shown in Table 1. Column 1 shows results for initial positions within the same range of displacement from the reference position as used for training. Columns 2 and 3 show generalization results for 2 and 3 times this range. Values are averages for 100 random initial positions, and standard deviations are shown alongside. The convergence threshold (average 2D error variation at final iteration) used was 0.001 pixel. The final average 2D positioning error in the image was 0.30 pixel, and the final average 3D positioning error was 1.51 mm translation and 0.11 degrees rotation (averages for the three coordinate axes), after an average of eight movement iterations. For initial positions within a range of displacement from the reference position two and three times larger than that used to generate the training images (columns 2 and 3), the network generalized quite well and was able to achieve the same final positioning accuracy in just a few more approach movements in the case of generalization ratio = 3.

Table 1
Generalization test results for 4-point images

Generalization ratio	1		2		3	
	mean	std	mean	std	mean	std
2D error (pixels)	0.30	0.07	0.30	0.07	0.32	0.08
3D T error (mm)	1.51	0.55	1.50	0.59	1.58	0.63
3D R error (deg)	0.11	0.05	0.10	0.05	0.10	0.05
Movements	8	6	7	5	10	6

A rough comparison may be made between these results and those of other authors based on both analytical and neural methods. Hashimoto et al. [33], with their similar neural method using real four-point images, obtained final 2D positioning errors of 14 pixels for a network trained on large offsets (approximately the same ranges as those used here), which was reduced to seven pixels with a network trained on offsets in a range 20% as large as the first. No 3D error values were given. These relatively large errors could be due in part to the small size of the training sets used (60 examples). Our training set size was chosen so as to include at least the number of possible combinations of three offset values one each of the six axes (one positive, one zero, and one negative value), meaning $3^6 = 729$ examples, which would seem like the minimum number needed to approximate the function reasonably well over the entire workspace. Although the values given in Table 1 are for simulated images, we later obtained the same precision values for real images (see the section concerning Fine Correction Using Point Features). In the case, the average 2D error was also less than 1 pixel, which was the resolution of the vision system.

Quantitative comparison with results of analytical methods is more difficult due to the scarce data provided by their authors. Mandel and Duffie [9] analytically estimated the transformation matrix relating the 2D image coordinates of four feature points with robot pose, but only gave final error values for Tz of 5.43 mm and an average value for rotation of -0.32 degrees. Abidi and Gonzalez [8] described a similar method, which they used to robotically insert a nozzle with a flared rim into a slotted receptacle, correcting the final position through force control during the insertion. Although no final positioning errors were given (neither 2D or 3D), a rough idea of the 3D error may be obtained as the difference between their initial estimate of the desired pose and the subsequently corrected estimate for the single experiment they presented. The differences for the six axes are: $Tx, y, z = 7.1$, 7.1, 8.4 mm, and $Rx, y, z =$ 3.05, 0.0, 2.6 deg. This does not provide any information about the actual 3D error following the second pose correction, however.

3. Fourier descriptors

3.1. Image generation

In this set of experiments, the geometrical image features were substituted with global image descriptors as inputs for the neural network. Images were restricted to simple objects, highly contrasted with respect to the background, for which it was possible to extract an external silhouette. Fourier descriptors were used to encode the shape of the resulting closed contour.

In tests using a real robot/camera system in a real

environment, various simple objects were used, such as a small water valve. The objects were placed on a white translucent table illuminated from below to enhance contrast.

A 6-axis manipulator (GT Productique) with a camera mounted on its end-effector was used to acquire images in the real environment. A control program, implemented on a Sun workstation, digitized images at a rate of 40 ms per image and sent positioning commands to the robot in cartesian coordinates. Inverse kinematic calculations were performed by a low-level controller running on a PC linked to the Sun workstation by a serial port.

3.2. Image processing

Before computing the Fourier descriptors of the real images, several preprocessing operations were performed on each image to extract the external contour of the object. The image was binarized with an adaptive threshold, and connected components were then computed. The biggest object (excluding the background) was identified, and its external contour was extracted.

The shape of the silhouette was encoded by sampling 512 equidistant points along the contour and calculating their Fourier descriptors. Each 2D point U_m may be considered a 2D vector in complex space ($U_m = x_m = jy_m$). The Fourier descriptors are defined as:

$$D_n = \frac{1}{N}\sum_{m=0}^{N-1} U_m \mathrm{e}^{-j2\pi nm/N}, \tag{1}$$

where we take $n = (-16, \ldots, -1, 0, 1, \ldots, 15, 16)$. This defines a list of 33 descriptors which encode the low-frequency content of the shape. Only these descriptors were used since, on the one hand, the neural network required a fixed number of inputs, and on the other, the remaining coefficients correspond to higher frequencies which encode very fine details in the shape, which can often include unwanted image noise.

It may be observed that these coefficients are not invariant to basic geometrical transformations such as translation, rotation and scaling, nor to the initial point chosen on the contour at which to begin the calculation. In pattern recognition applications, it is usually desired to normalize the Fourier coefficients so as to make them invariant to geometric transformations. For visual servoing, however, variance to translation, rotation and scaling must be preserved, and to avoid matching between the observed and reference images, invariance to the starting point on the contour is desired.

Invariance to the starting point may be achieved by observing the effect of shifting the starting point by an arbitrary number of points, k, so as to obtain a new set of points U'_m defined by

$$U'_m = U_{(m+k) \bmod N}. \tag{2}$$

The Fourier descriptors become

$$D'_n = D_n \mathrm{e}^{2j\pi kn/N}. \tag{3}$$

Consequently, the original Fourier descriptors D_n may be recovered by simply multiplying each descriptor D'_n by $\mathrm{e}^{-2j\pi kn/N}$.

Since the phase angle Φ'_1 of the first coefficient D'_1 is equal to

$$\Phi'_1 = \frac{2\pi k}{N}, \tag{4}$$

the original coefficients may be expressed as

$$D_n = D'_n \mathrm{e}^{-jn\Phi'_1}, \tag{5}$$

which are invariant to shifting the starting point by any arbitrary number of points k.

3.3. Neural network training

Training sets of real images were generated for the neural network by choosing 500 random initial poses in the vicinity of the reference position, in this case within the ranges of −25 to 25 mm translation and −15 to 15 degrees rotation for the three coordinate axes. The reference position was chosen arbitrarily so as to have the camera centered perpendicularly above the object's surface, at a distance of approximately 550 mm. The center of rotation was a point centered 550 mm in front of the camera on the optical axis (near the object's surface), which helps maintain the object within the field of view. A translational component is therefore derived from the rotations around the two horizontal axes, making the actual maximum ranges of Tx and Ty equal to $\pm[25 + 550 \cdot \sin(15)] = \pm 167$ mm.

For the observed image at each pose, the closed contour of the object was extracted as explained above, and a set of 512 equidistant points were sampled from the contour to compute the Fourier descriptors. The network inputs were simply the differences between each of the corresponding normalized Fourier coefficients (66 terms for 33 complex coefficients) for the reference and observed images. As with the four-point images, the desired outputs were the six cartesian coordinates of the camera. A set of 500 additional randomly generated images were used as a test set for training validation.

These data were used to train a 3-layer backpropagation network as before, but with 66 inputs, 30 hidden nodes and six outputs. Training was performed for 3000 training-set passes, after which the final RMS error for the test set was 0.03.

3.4. Visual positioning tests

The trained neural networks were tested using the same iterative visual positioning scheme as in the four-point experiments. Due to the slow speed of the serial

Table 2
Final positioning error for 10 servoing trials using a backpropagation network trained with Fourier descriptors

Axis	Valve 32 FD	
	mean	std
Tx (mm)	−0.2	1.7
Ty	−0.6	2.2
Tz	0.1	1.2
Rx (deg)	−0.2	1.8
Ry	0.7	2.1
Rz	0.0	0.9

communication link used in the image-processing setup, however. statistics were computed for just 10 servoing trials. The 10 initial positions were chosen within the same ranges of displacement as used for training. Each trial was stopped after 10 approach movements, after which no further convergence could be observed, although in general five or six movements were sufficient.

The average and standard deviation of the robot displacement along each of the six axes during the final (10th) movement are shown in Table 2. Ideally, both should approach zero, indicating convergence to a stable pose – presumably the reference pose. The absolute mean values for rotation were below 0.7 degrees, and below 0.6 mm for the translation axes. The standard deviation values are used as a measure of the final positioning errors, since they indicate the degree of variation between the final position of different trials.

From Table 2 it is evident that the positioning errors were significantly higher for translation and rotation on the x and y axes than on the z (optical) axis. The error values for Tx and Ty represent a 7–9% error with respect to the full range of possible translational displacements, while the error for Tz represents a 5% error. Keeping in mind that the center of rotation used is displaced 550 mm from the wrist, the actual translational displacements from the reference pose of the camera are somewhat higher than these values for axes x and y, since an additional translational component is derived from the rotational errors around these axes. As for the rotational errors, Rx and Ry represent a 12–14% error with respect to the full range of rotational displacements used, while the error in Rz represents only a 6% error.

The higher positioning errors for axes x and y are believed to be due to ambiguities between the image variations produced by translations on either axis and rotations on the opposite axis – the same problem as discussed above for the simulated four-point images. In the latter case, translational and rotational movements could be disambiguated by simply varying the locations of the points in order to add more relative depth to the image, which would be equivalent to selecting feature points at different depths in the image of a real object. The solution is not so straightforward when global image features such as Fourier descriptors are used, however, since the feature locations are inherent to the image itself and cannot be chosen explicitly. While it is possible that objects of different shape and appearance might produce more unambiguous image variations for rotations and translations on the x and y axes, it was clear for the objects used here that many images taken at different poses were nearly indistinguishable to the eye, as may be seen in the two images of Fig. 3.

Note that rotations and translations on the z (optical) axis produce unique image variations – simple rotation of the object in the image plane and global change of scale, respectively – which cannot be easily confused with the effects of movements on any other axis. Consequently, error values are lower for the movements on the z axis.

An example of visual positioning with respect to a real object (a water valve) is illustrated in the photographs in Fig. 4. The accurate 2D convergence of the observed images onto the reference image is easily seen.

4. Geometric moments

4.1. Motivation

To make the neural visual positioning method applicable to more general classes of images, a more flexible type of image descriptors must be used. Fourier

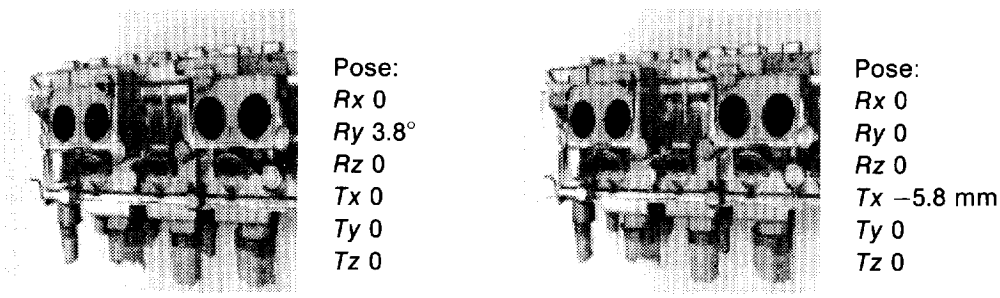

Fig. 3. Example of two nearly identical images taken from two different poses, giving rise to ambiguity in the set of training images.

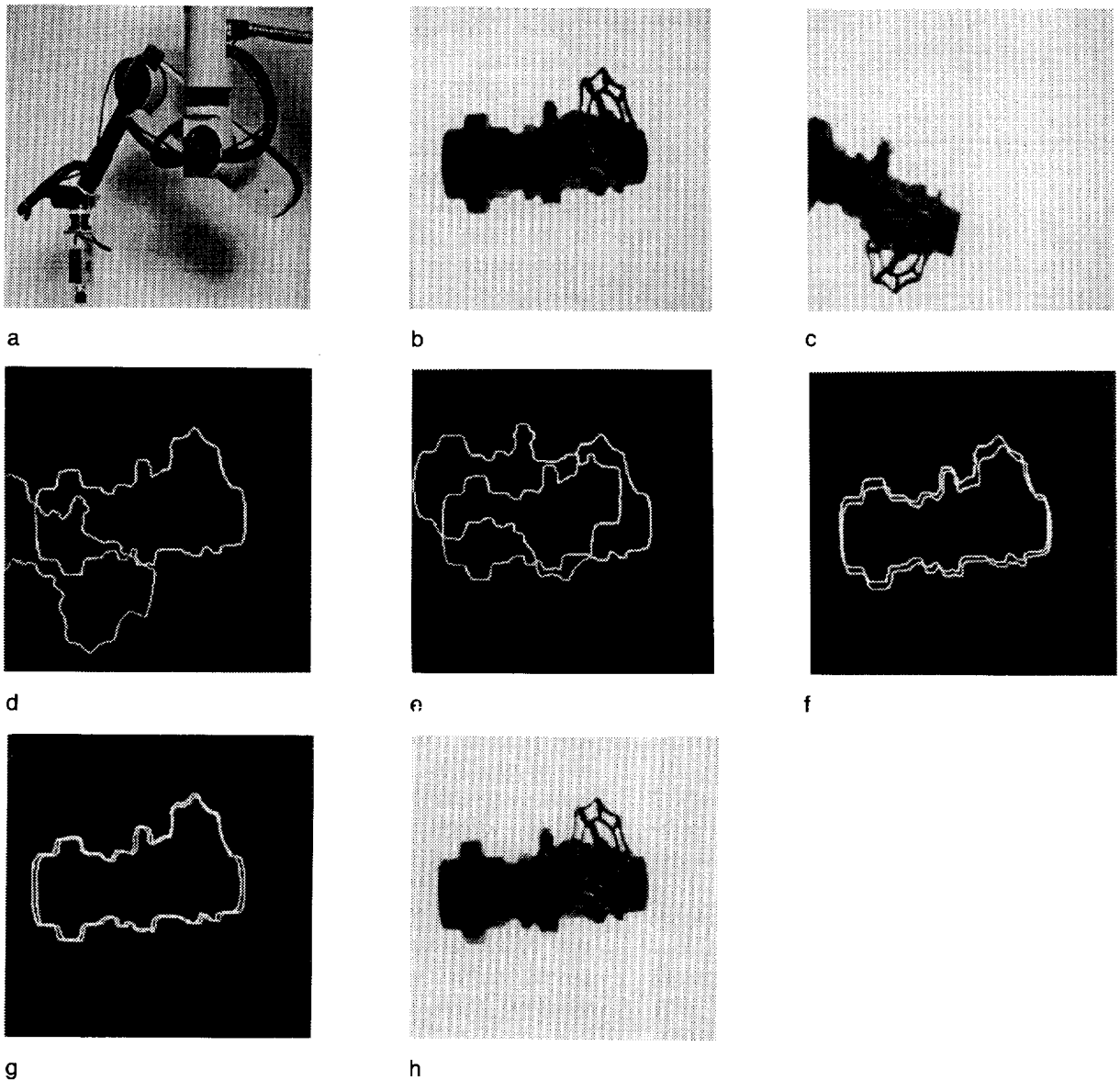

Fig. 4. Visual positioning with respect to a water valve, using Fourier descriptors of the object contour. The sequence shows the reference, initial and final images, and the superimposed contours of the reference and current image at several steps of the positioning process. Note that positioning is possible even when the object lies partly outside the image. (a) Robot and camera, (b) reference image, (c) initial image, (d) initial contours, (e) after 1 movement, (f) after 5 movements, (g) after 7 movements, (h) final image.

descriptors require that the object's external contour be clearly visible and distinguishable from the image background, which is not always the case, as when large objects are viewed at close range. Geometric moments do not have this restriction, since they are used to encode the image as a whole. They are similar to Fourier descriptors, however, in that finer degrees of image detail may be encoded by computing moments of successively higher order. Consequently, they are very compatible with the neural learning approach, since a fixed number of moments may be selected to encode the most important characteristics of the image and used as network inputs. Although computationally intensive, efficient algorithms [40,41] and even specialized hardware [19] are available for fast moment computation.

4.2. Image acquisition

Experiments with geometric moments were first performed using the same set of images of the water valve (with external contour) as used in the Fourier descriptor experiments. This allowed the performances of both methods to be compared. In later tests, close-range images of an automobile cylinder head were used, which provided a set of coarsely textured images with little or no visible background.

The robot/camera system used was the same as before, except that, in this case, two cameras approximately 10 cm apart were used to acquire 'stereo' image pairs. This was found to improve positioning performance by up to 15% on some coordinate axes and to help prevent divergence when positioning is begun from poses relatively far from the reference pose. Due to the sensitivity of geometric moments to variations in pixel grey levels, a controlled lighting arrangement was used in which light was reflected off a white ceiling above the work area, which was enclosed in a black curtain. This ensured contrast and relatively uniform illumination of the object from all directions.

4.3. Preprocessing operations

Before computing geometric moments, several preprocessing operations are performed on the images for filtering and normalization purposes. Images are first processed with a Prewitt filter to obtain the derivatives of the grey-level values. The result is a highly filtered image in which only the feature boundaries are visible and highlighted. The images are then normalized with respect to their mean grey-level value to minimize the effects of any lighting variations between images.

4.4. Geometric moment computation

From the filtered and normalized images, a series of descriptors involving geometric moments is computed. The general formula for geometric image moments is given by

$$m_{ij} = \sum_x \sum_y x^i y^j f(x,y) \tag{6}$$

where m_{ij} is the moment of order ij, x and y are the coordinates of each pixel in the image, and $f(x,y)$ is the grey-level value of the pixel (although here the derivatives of the grey-level values are used). By giving different values to orders i and j, several important characteristics of the image may be encoded. For example, m_{00} is the total 'mass' of the image, and m_{02} and m_{20} are the moments of 'inertia' around the x and y axes, respectively.

While, for pattern recognition applications, moment *invariants* are typically used to recognize image features regardless of the viewing position, for visual servoing it is desired that the moments have a *variant* relationship with respect to the camera pose. In the present work, ten descriptors involving moments were chosen which were believed to best quantify the variations in the object's projection in the image when the camera pose is varied on any of its 6 axes. The first two descriptors are the x and y coordinates of the image centroid, which is clearly variant with camera translation parallel to the image plane:

$$\bar{x} = m_{10}/m_{00} \quad x \text{ coordinate of image centroid.} \tag{7}$$

$$\bar{y} = m_{01}/m_{00} \quad y \text{ coordinate of image centroid.} \tag{8}$$

To represent the rotation of the object in the image plane, the angle of rotation of the principal axis of inertia may be used. This quantity may be derived from the eigenvectors of the matrix

$$\begin{bmatrix} \bar{m}_{20} & -\bar{m}_{11} \\ -\bar{m}_{11} & \bar{m}_{02} \end{bmatrix} \tag{9}$$

where $\bar{m}_{11}$, $\bar{m}_{20}$ and $\bar{m}_{02}$ are *central moments*, defined with respect to the centroid of Eqs. (7) and (8) as:

$$\bar{m}_{ij} = \sum_x \sum_y (x - \bar{x})^i (y - \bar{y})^j f(x,y). \tag{10}$$

The eigenvector v corresponding to the largest eigenvalue λ_1 of matrix (9) gives the direction of the main inertia axis. The elements of v are used as descriptors:

$$v_1 \quad x \text{ component of main inertia axis.} \tag{11}$$

$$v_2 \quad y \text{ component of main inertia axis.} \tag{12}$$

The scaling of the object, due primarily to camera translation along the optical axis, may be quantified by the radii of the major and minor inertia axes. These are derived from the eigenvalues, λ_1 and λ_2, of matrix (9):

$$r_1 = \sqrt{\frac{\lambda_1}{m_{00}}} \quad \text{radius of major inertia axis.} \tag{13}$$

$$r_2 = \sqrt{\frac{\lambda_2}{m_{00}}} \quad \text{radius of minor inertia axis.} \tag{14}$$

Thus, the six expressions given by Eqs. (7), (8) and (11)–(14) are used to quantify the image variations resulting from six possible camera translations and rotations.

Since somewhat unsatisfactory results were obtained in initial servoing experiments using just these six descriptors, the four third-order moments, m_{30}, m_{21}, m_{12} and m_{03}, were added to more sharply characterize the image. The variations in these ten descriptors with respect to a reference image are used as inputs to the neural network, which learns to map them to the applied 6D camera movements.

4.5. Training-set generation

Training data for the neural networks were generated from images taken from the wrist-mounted cameras at a series of random known poses within a limited range of a predefined reference position. The reference pose was arbitrarily chosen at a point centered approximately 550 mm above the object. Random poses were selected within a range of −25 to 25 mm translation from the reference pose, and between −15 and 15 degrees rotation for all coordinate axes, where the center of rotation was a point centered 550 mm in front of the two cameras as before.

Stereo image pairs were taken at 750 poses for both the valve and the cylinder-head scenes. After preprocessing, the 10 geometric-moment descriptors were computed for each image as described above. The differences between the

corresponding descriptors in the observed and reference images were used as network inputs. The desired network outputs were the six relative cartesian coordinates corresponding to the applied movement of the robot from the reference pose to the current pose. Consequently, each training example consisted of a 20-element input vector (10 values for each camera) and the associated 6-element output vector. Sets of 250 additional 'stereo' images of both scenes were used as test sets for training validation.

4.6. Neural network training

The data for each scene was used to train a 3-layer backpropagation network. While, in previous experiments, the number of hidden nodes and training examples were chosen rather arbitrarily, having decided upon geometric moments as the descriptors to be used in the final industrial application called for a more systematic selection of these parameters in order to optimize performance. The hidden-layer size, activation functions, and other training parameters were selected empirically through systematic experimentation, as summarized in Wells and Venaille [42].

Hidden layers of 10, 20, 30, 40, 50 and 60 nodes were tested. A minimum of 30 hidden nodes were found necessary for the valve data, and 50 for the cylinder head. In both cases, reducing the number of nodes by 10 increased the final error for the test set by 2–3% and steadily increased with fewer nodes. Adding more nodes noticibly slowed down training and produced a negligible reduction in the final error.

The minimum training set was determined by comparing training results using 500, 750, 1000, 3000 and 10,000 examples. Training times were unacceptably slow for more than 1000 examples, with no significant improvement in performance for more than 750. The final training error was only slightly higher with 500 examples, but 750 (at least 729) were finally maintained for the reasons explained above for four-point images.

Of the possible combinations of sigmoid, tanh, sine and linear functions, a tanh function in the hidden layer and a linear function in the output layer produced

Table 3
Final positioning errors for 10 servoing trials using neural networks trained with geometric-moment descriptors

Axis	Valve 22 GM		Cylinder head 22 GM	
	mean	std	mean	std
Tx (mm)	0.4	1.6	−1.9	2.9
Ty	−0.3	2.0	−1.4	3.1
Tz	0.4	1.4	−1.0	2.6
Rx (deg)	−0.1	4.0	−0.5	1.8
Ry	0.3	1.3	−0.7	1.7
Rz	0.1	0.1	−0.3	0.7

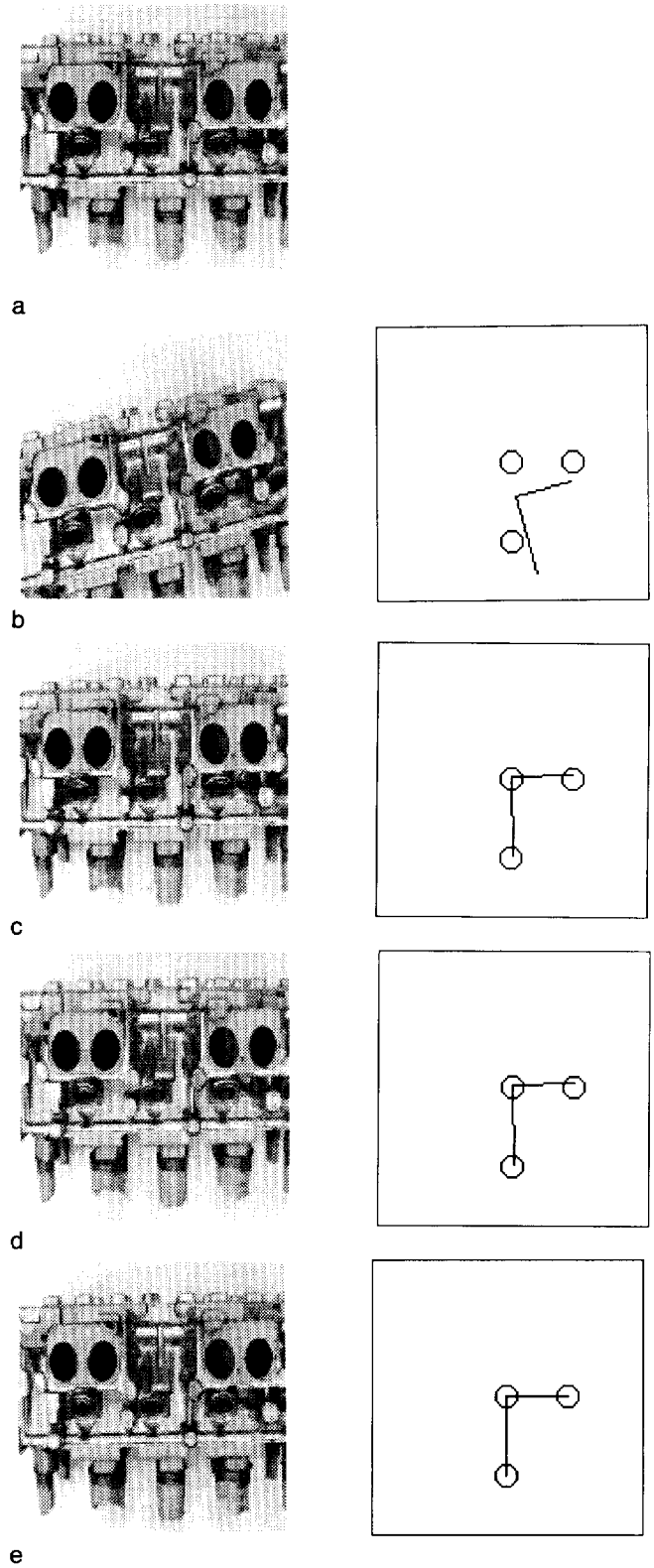

Fig. 5. Visual positioning with respect to an automobile cylinder head, using geometric-moment descriptors of the image. The sequence shows the reference image and current image after three of a series of five movements, along with a 2D representation of the major and minor inertia axes of each image, where the circles correspond to the axes endpoints in the reference image. (a) Reference image, (b) initial image, (c) move 1, (d) move 3, (e) move 5.

the best training results for both sets of data. The 'Quickprop' variant of the backpropagation learning rule was used, which provides automatic adjustment of the learning rate. Training convergence was achieved after 10,200 passes through the training set.

4.7. Visual positioning tests

The trained neural network was tested in a series of 10 positioning trials as before, for both the water valve and the cylinder-head objects. From the results shown in Table 3, it may be seen that positioning accuracy for the water valve is approximately equivalent to that obtained using Fourier descriptors, including the higher error values for movements on axes x and y, presumably for the same reasons as described above.

Errors are somewhat higher, however, for movements Rx using geometric moments (4.0 mm) than for Fourier descriptors (1.8 mm). Apparently, the geometric-moment descriptors used are less able to disambiguate rotational movements around the x axis for this particular object, which is likely due to the fact that these movements are perpendicular to the line of stereo separation in the horizontal plane, and are therefore less disambiguated by the stereo configuration of the cameras, the effect being additionally compounded by the high degree of symmetry of the valve object around the x axis in the image. If the valve had been rotated 90 degrees, the results would likely have been interchanged for axes x and y, but this was not verified. The cylinder head is more asymmetric around both the x and y axes, and the positioning errors for Tx, Ty, Rx and Ry are correspondingly very similar. In both cases, convergence to a stable pose was achieved after an average of 5–7 movements.

The images observed during a typical positioning trial using geometric moment descriptors are shown in Fig. 5. A 2D representation of the position and orientation of the major and minor inertia axes of each image is included to facilitate comparison of each image with the reference image. Graphs of the movements made along each of the 6 robot axes during the positioning trial of Fig. 5 are shown in Fig. 6.

4.8. Additional tests

The final precision achieved with geometric moments represents an average reduction in the positioning error of approximately 90% with respect to the full range of initial pose displacements used. While these results may be judged as quite good, considering the high 'compression ratio' of the image descriptors used, the small size of the images (i.e. low resolution), and the relatively large

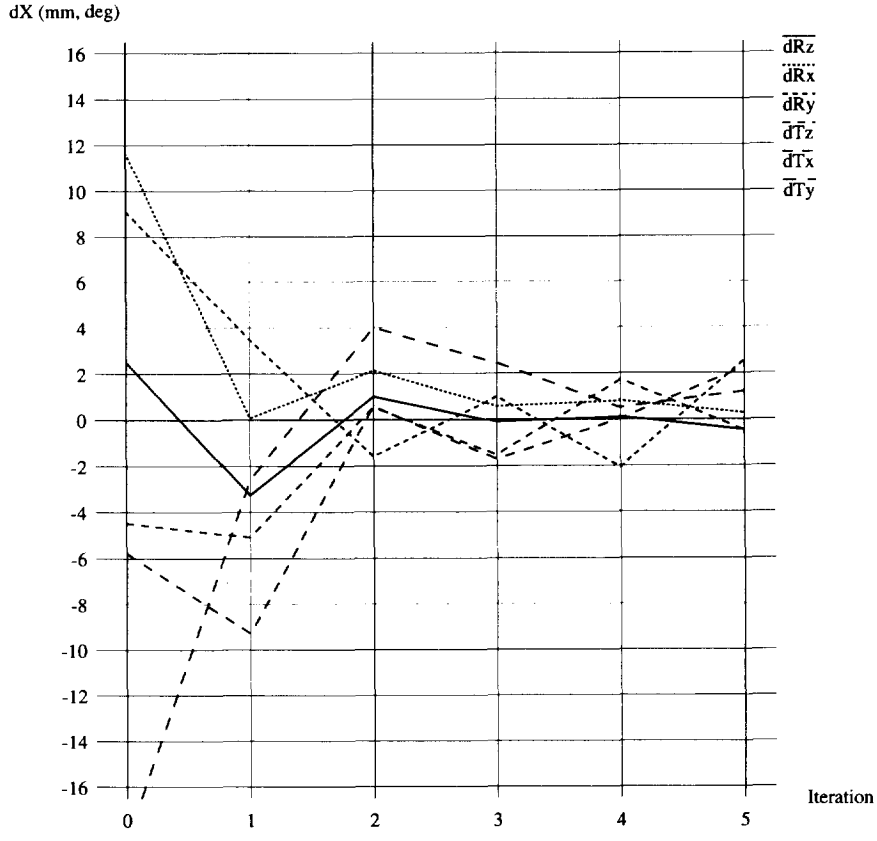

Fig. 6. Movements made along each of the six robot axes during the positioning trial of Fig. 5. ——: dRz; - - - -: dRx; – – –: dRy; – – -: dTz; — —: dTx; — –: dTy.

distance between the camera and the object, it is clear that there is considerable room for improvement.

Several variations were tested in an effort to increase the precision. In one test, the 2nd, 4th and 5th-order moments were added to the image feature vector (14 additional descriptors). This resulted in a 4% training improvement, but little noticeable improvement in positioning performance. Although adding moments of even higher order could be imagined to further enhance precision, the addition of more network inputs makes training increasingly slow.

The benefit of formulating the first six descriptors so as to relate them with physical quantities was verified in another experiment, in which only the two coordinates of the centroid and the moments of orders 2 through 5 were used as network inputs. It can be imagined that the neural network might learn equally well the direct implicit association between variations in the raw moments and pose changes. However, it was found that the precision is from 6–12% better when these specially formulated descriptors are included.

Increasing the stereo basis from 10 to 20 cm was found to have little effect on results. One likely way, though not yet tried, to improve precision could be to train a second neural network using image data sampled at a higher resolution and then perform a second 'fine-positioning' step. This could be done by collecting images within a smaller range of pose displacements, and possibly also zooming in on the object and computing geometric moments for the resulting close-range images. A multi-resolution approach of this type (without zooming in) was used in the neural approach, based on four point features, of Hashimoto et al. [33] with positive results.

5. Fine correction using point features

As seen above, neural-based visual positioning using Fourier descriptors or geometric moments allow positioning to approximately 3 mm translation and 2 degrees rotation of the desired pose. While further refinements in the method may eventually increase the precision attainable with these image descriptors alone, time limitations in the CONNY project required that a reliable method be developed to enhance the precision following the initial positioning step using global descriptors. To this end, a neural fine-correction step was developed based on the four-point-feature method described above. With it, the final precision may be increased to approximately 2 mm translation and 0.1 degree rotation, which were within the performance specifications required for the industrial inspection application being developed by Thomson.

The fine-correction method consists of specifying four feature points in the reference image, and then training a neural network to relate the x, y displacements of these points in the image with arbitrary pose displacements of the camera within the workspace. In operation, a simple correlation method is used to find the matching four points in the current image, their offsets are computed, and the network gives the corresponding pose correction.

The location of the four feature points in the reference image is arbitrary, but should be chosen carefully so as to reduce the possibility of multiple matches. Each of the four points defines the center of a small 'reference window', w, as shown in Fig. 7. In the observed image, four larger 'search areas' are defined, each containing an area of the same size and location as one of the reference windows in the reference image. Then, a 'correlation window', P, of the same dimensions as the reference window is displaced to all possible coordinates m, n within the search area, and the following correlation function is computed at each location:

$$R(m,n) = \frac{\sum_x \sum_y (P_{m,n}(x,y) - \bar{P}_{m,n})(w(x,y) - \bar{w})}{\left(\sum_x \sum_y (P_{m,n}(x,y) - \bar{P}_{m,n})^2 \right)^{1/2}} \quad (15)$$

where $w(x,y)$ and $P(x,y)$ are the pixel grey-level values in the reference and observed images, respectively. The correlation values are centred around the mean grey-level values $\bar{w}$ and $\bar{P}_{m,n}$, which increases the robustness of the method to noise and variations in image intensities.

The coordinates m, n at which the correlation window produces the maximum correlation value, $\max_{m,n} R(m,n)$, is labelled as the observed point which matches the corresponding reference point in the reference image.

The dimensions of the search and correlation windows should be chosen small enough to maximize speed while not compromising matching performance. The images used in these experiments measured 128 × 128 pixels. The reference and correlation windows had dimensions 16 × 16 pixels, and the search area was 32 × 32 pixels in size.

5.1. Fine-correction tests

A series of 15 positioning tests was performed using the network trained with four-point-feature data. Training parameters and performance were similar to the values given above for experiments with four point features. The training set consisted of 750 images, the test set had 250 images, and the range of pose displacements used for training-set generation and testing was 20% of that used for the geometric-moment data: ±5.0 mm translation and ±3.0 degrees rotation.

As seen in the results of Table 4, the fine correction step allows an improvement in positioning accuracy to below 2.2 mm translation for Ty and 0.1 degree rotation on all three axes. The error for Tx is somewhat lower, 1.9 mm, by virtue of the stereo camera configuration, and the accuracy for Tx reaches 0.1 mm. The robot converged to a stable pose in an average of 5.9 movements (std = 6.7). Note that

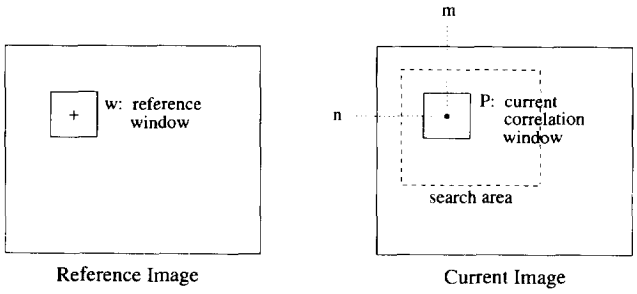

Fig. 7. Windows used in the correlation method.

these values are in good correspondence with those obtained for simulated four-point images (Table 1).

Due to the limited resolution of the images used here, the final error values in Table 4 represent the highest obtainable precision with this method under these experimental conditions. The maximum 2D resolution was 1 pixel, and in all the servoing trials, the final 2D error converged to zero, thus limiting the 3D errors to the values shown here. This would suggest that the precision could be improved even further by increasing the resolution, either by using larger images or by using a multi-resolution approach as described above. This could be the subject of further research.

It must also be pointed out that, although using point features and correlation matching allows a higher positioning accuracy than either Fourier descriptors or geometric moments, it is not entirely feasible to perform visual positioning based on this method alone, since correlation matching requires that the object rotation be less than 10 degrees on all axes. Consequently, this method may only be applied once the positioning error has been sufficiently reduced by some other technique.

Fig. 8 shows the sequence of fine-correction movements performed using four-point features immediately following the positioning sequence with geometric moments shown in Fig. 5. Although differences in the images themselves are barely noticeable to the eye, the small differences are evident in the 2D representation showing the relative coordinates of the four points in the reference and current images.

Graphs of the movements made along each of the 6

Table 4
Final positioning errors for 15 servoing trials using neural network trained with 4-point feature data and correlation matching

Axis	Cylinder head 4-points	
	mean	std
Tx (mm)	0.2	1.9
Ty	0.1	2.2
Tz	0.2	0.1
Rx (deg)	0.0	0.1
Ry	0.0	0.1
Rz	0.0	0.1

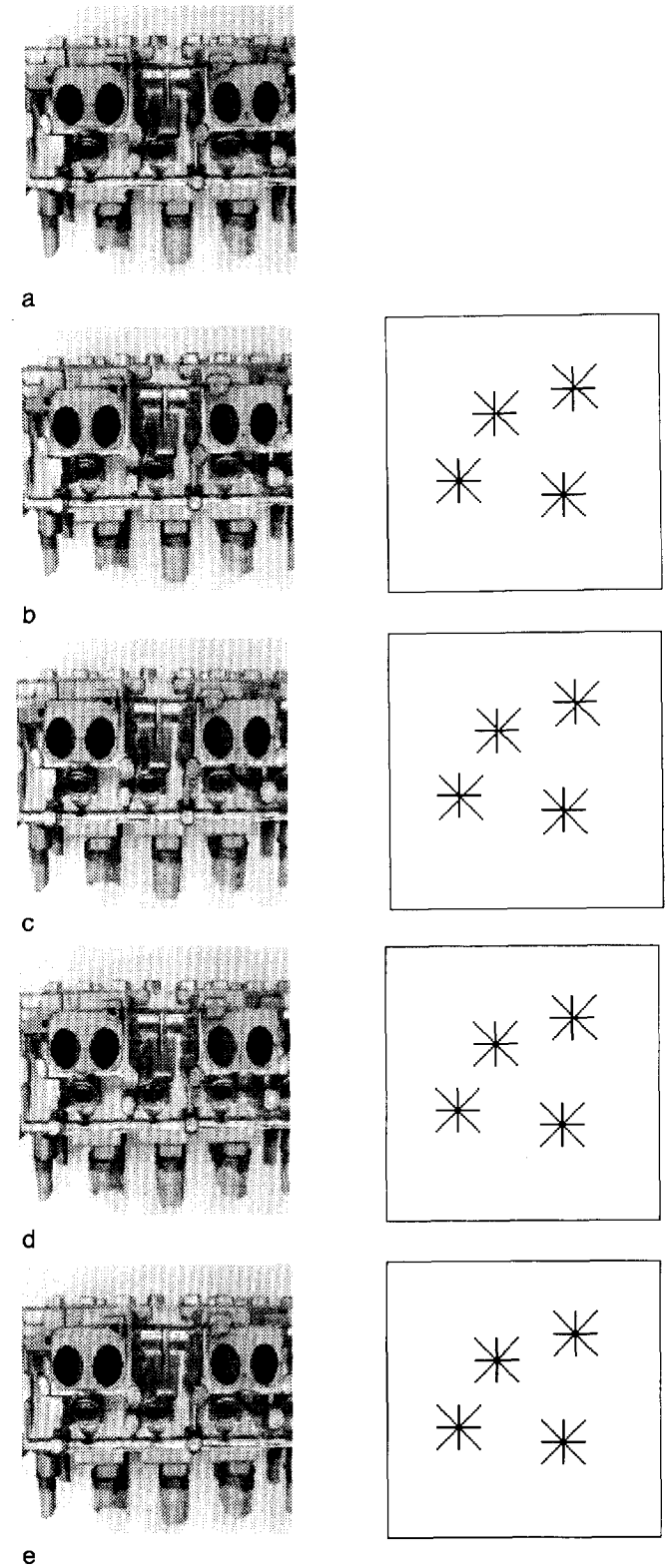

Fig. 8. Fine position correction using 4-point features, following the positioning sequence with geometric moments of Fig. 5. The sequence shows the reference image and current image after three of a series of five movements, along with a 2D representation of the coordinates of the four points in the reference (shown as 'X') and current (shown as '+') images. (a) Reference image, (b) initial image, (c) move 1, (d) move 3, (e) move 5.

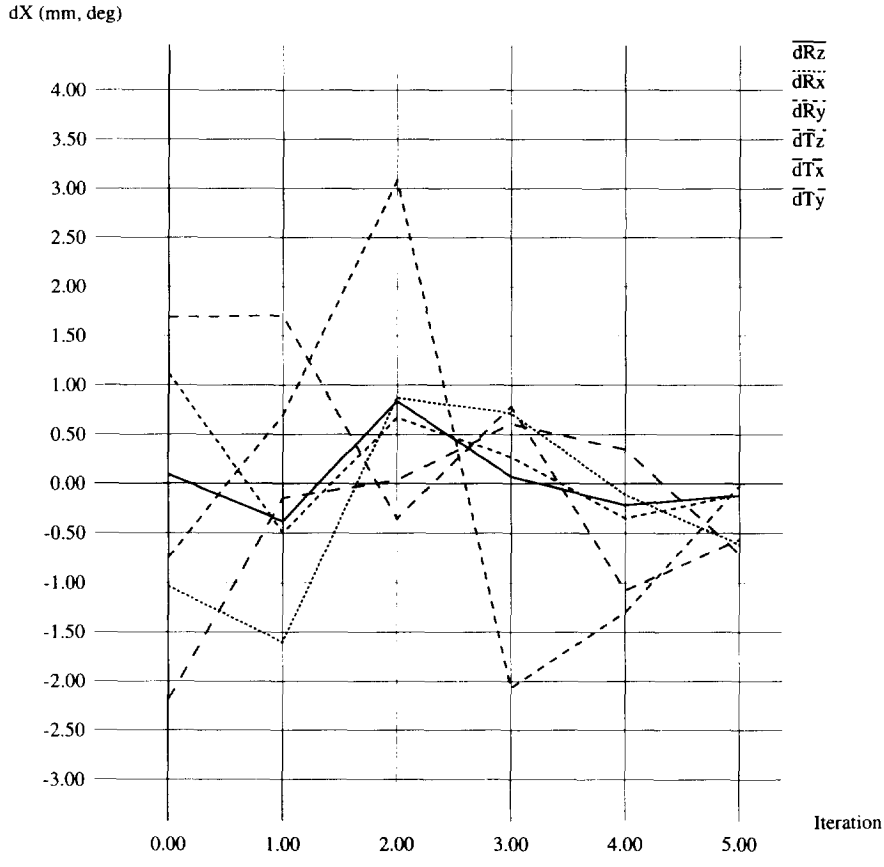

Fig. 9. Movements made along each of the 6 robot axes during the positioning trial of Fig. 8. ——: dRz; - - - -: dRx; – – –: dRy; – – –: dTz; — —: dTx; — –: dTy.

robot axes during the positioning trial of Fig. 5 are shown in Fig. 9.

6. Conclusions

In this work, we have reviewed the current state of the art in vision-based robot positioning, and have described a novel method for visual positioning of a 6-dof manipulator based on neural learning and global image descriptors. The method takes advantage of the approximation capabilities of backpropagation networks to learn the complex mappings between variations in global image encodings, such as geometric moments and Fourier descriptors, and the displacement of the 6-dof robot from a desired pose. By using global image descriptors, the typically problematic steps of extraction and matching of geometric features are eliminated, making it possible to position the robot with respect to arbitrary real scenes. In addition, neural learning makes it possible to obtain the direct mapping between image variations and the robot pose, avoiding explicit camera modelling, calibration and all other intermediate transformations entirely, reducing the computational burden and eliminating possible sources of estimation error. Nor is any explicit pose information required, since the desired pose is specified simply as a reference image, and the neural controller learns the relative displacements of the robot from the pose where this image is initially observed.

Experiments performed so far have shown that it is possible to obtain final positioning errors of approximately 3 mm translation and 2 degrees rotation using geometric moments of Fourier descriptors alone (with the camera approximately 550 mm from the object). These values may be decreased to less than 2.2 mm and 0.1 degree when a subsequent correction step is applied using a neural network trained with four point features in conjunction with a correlation matching procedure. In subsequent tests [43], both the geometric-moment and fine-correction steps of the positioning method have shown to be very robust with respect to different perturbations such as lighting variations, generalization outside the range of training poses, image noise and object defects. The method has proven to be particularly robust to even severe cases of partial occlusion, converging to the desired pose from initial poses in which large portions of the object were not visible, either because they were occluded by other parts of the object, or else fell outside the image boundary. In fact, occlusion is basically unimportant, since the neural network has learned to associate the global 'appearance' of the image (as opposed to locally visible features) with pose over the

entire workspace. Analytic feature-based methods would generally fail in these cases unless provided with a 3D object model or special mechanisms for dealing with missing features.

Although the precision achieved using global descriptors was quite satisfactory for the visual inspection application developed here, further refinements are needed in order to approach the precision provided by local-feature techniques, as demonstrated by the point-based fine-correction method. A significant disadvantage of low-order geometric moments in particular is that they group large amounts of diverse pixel information into single values, thus losing most of the local fine-detail information and resulting in excessive similarity between many of the training images. Including higher-order moments might add needed resolution to the training set, although the number of moments becomes larger as their order is increased. Precision might also be enhanced by means of a 'multi-resolution' approach, in which a second fine-positioning step is performed using a neural network trained on closer-range images within a smaller range of movements.

Further extensions could include improvement of the control scheme to provide continuous motion of the end-effector for applications requiring dynamic tracking.

Acknowledgements

Thomson Broadcast Systems at Rennes and the Instituto de Cibernética are currently developing neural vision-based robot positioning systems under ESPRIT project No. 6715 'Robot Control based on Neural Network Systems', partially supported by the European Community. Research in neural networks and their applications to process control at the Instituto de Cibernética has been partially supported by the research grants CICYT-TIC91-0423 and CICYT-TAP93-0451 of the Spanish Science and Technology Council.

References

[1] P. Corke, Visual control of robot manipulators – a review, in K. Hashimoto (ed.) *Visual Servoing*, World Scientific, Singapore, 1993, pp. 1–31.

[2] C. Venaille, G. Wells and C. Torras, Application of neural networks to image-based control of robot arms, *Proc. 2nd IFAC Symposium on Intelligent Components and Instruments for Control Applications (SICICA), Budapest, Hungary*, 1994, pp. 281–286.

[3] E. Dickmanns, B. Mysliwetz and T. Christains, An integrated spatio-temporal approach to automatic visual guidance of autonomous vehicles, *IEEE Trans. Systems, Man, and Cybernetics* (20) (1990) 1273–1284.

[4] I. Masaki, S. Decker, A. Gupta, B. Horn, H. Lee, D. Martin, C. Sodini, J. White and J. Wyatt, Cost-effective vision systems for intelligent vehicles, *Proc. Int. Symposium on Intelligent Vehicles, Paris, France*, October 1994, pp. 39–43.

[5] J. Feddema, Visual servoing: a technology in search of an application, *IEEE Int. Conf. Robotics and Automation, Notes for Workshop M-5 (Visual Servoing), San Diego, CA*, May 1994.

[6] G. Hagar and S. Hutchinson, Visual servoing: achievements, issues and applications, *IEEE Int. Conf. Robotics and Automation, Notes for Workshop M-5 (Visual Servoing), San Diego, CA*, May 1994.

[7] R. Horaud, B. Conio and O. Leboulleux, An analytic solution for the perspective 4-point problem, *Computer Vision, Graphics and Image Processing*, (44) (1989) 33–44.

[8] M. Abidi and R. Gonzalez, The use of multisensor data for robotic applications, *IEEE Trans. Robotics and Automation, 6*(2) (1990) 159–177.

[9] K. Mandel and N. Duffie, On-line compensation of mobile robot docking errors, *IEEE J. Robotics and Automation, 3*(6) (December 1987).

[10] M. Kabuka and A. Arenas, Position verification of a mobile robot using standard pattern, *IEEE J. Robotics and Automation, 13*(6) (December 1987).

[11] R. Harrell, D. Slaughter and P. Adsit, A fruit-tracking system for robotic harvesting, *Machine Vision and Applications*, (2) (1989) 69–80.

[12] Z. Zhang, R. Weiss and A. Hanson, Automatic calibration and visual servoing for a robot navigation system, *Proc. IEEE Conf. on Robotics and Automation, Atlanta, GA*, May 1993, pp. 14–19.

[13] F. Chaumette, La relation vision-commande: théorie et application à des tâches robotiques, *Ph.D. Thesis*, Université de Rennes I, France, 1990.

[13a] J. Hertz, A. Krogh and R. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Wesley, Redwood, 1991.

[13b] F. Chaumette, P. Rives and B. Espiau, Positioning of a robot with respect to an object, tracking it and estimating its velocity by visual servoing, *Proc. IEEE Int. Conf. on Robotics and Automation, Sacramento, April 1991*, pp. 2248–2253.

[13c] B. Espiau, F. Chaumette and P. Rives, A new approach to visual servoing in robotics, *IEEE Trans. on Robotics and Automation, June 1992.*

[13d] Z. Bien, W. Jang and J. Park, Characterization and use of feature-Jacobian matrix for visual servoing, in K. Hashimoto (ed.), *Visual Servoing*, World Scientific, Singapore, 1993, pp. 317–363.

[14] L. Weiss, A. Sanderson and C. Neuman, Dynamic sensor-based control of robots with visual feedback, *IEEE J. Robotics and Automation, 3*(5) (October 1987) 404–417.

[15] K. Horn and B. Schunck, Determining optical flow, *Artificial Intelligence*, (17) (1981) 185–203.

[16] P. Allen, B. Timcenko and P. Michelman, Hand-eye coordination for robotic tracking and grasping, in K. Hashimoto (ed.), *Visual Servoing*, World Scientific, Singapore, 1993, pp. 33–70.

[17] N. Papanikolopoulos, Adaptive control, visual servoing and controlled active vision, *IEEE Conf. Robotics and Automation, Notes for Workshop M-5 (Visual Servoing), San Diego, CA*, May 1994.

[18] P. Corke, Video-rate robot visual servoing, in K. Hashimoto (ed.), *Visual Servoing*, World Scientific, Singapore, 1993, pp. 257–284.

[19] R. Andersson, Real-time expert system to control a robot ping-pong player, *PhD Thesis*, University of Pennsylvania, June 1987.

[20] W. Jang and Z. Bien, Feature-based visual servoing of an eye-in-hand robot with improved tracking performance, *Proc. IEEE Conf. on Robotics and Automation, Sacramento, April 1991*, pp. 2254–2260.

[21] S. Nayar, H. Murase and S. Nene, Learning, positioning and tracking visual appearance, *Proc. IEEE Conf. on Robotics and Automation, San Diego, CA*, May 1994, pp. 3237–3244.

[22] A. Sanderson and L. Weiss, Image-based visual servo control using relational graph error signals, *Proc. IEEE* (1980) 1074–1077.

[23] R. Tsai and R. Lenz, A new technique for fully autonomous and efficient 3D robotics hand/eye calibration, *IEEE Trans. Robotics and Automation, 5*(3) (June 1989) 345–359.

[24] G. Stockman and S. Chen, Detecting the pose of rigid objects: a comparison of paradigms, *Optical and Digital Pattern Recognition, 754* (1987) 107–115.

[25] Y. Hung, P. Yeh and D. Harwood, Passive ranging to known planar point sets, *Proc. IEEE Conf. Robotics and Automation, St. Louis, MI,* March 1985, pp. 80–85.

[26] A. Rizzi and D. Koditschek, A dynamic sensor for robot juggling, in K. Hashimoto (ed.), *Visual Servoing*, World Scientific, Singapore, 1993, pp. 229–256.

[27] G. Hagar, W. Chang and A. Morse, Robot feedback control based on stereo vision: towards calibration-free hand-eye coordination, *Proc. IEEE Int. Conf. Robotics and Automation, San Diego, CA,* May 1994, pp. 2850–2856.

[28] D. Vernon and M. Tistarelli, Using camera motion to estimate range for robotic parts manipulation, *IEEE Trans. Robotics and Automation, 6*(5) (October 1990) 509–521.

[29] K. Hashimoto, T. Kimoto, T. Ebine and H. Kimura, Manipulator control with image-based visual servo, *Proc. IEEE Int. Conf. Robotics and Automation*, 1991, pp. 2267–2272.

[30] J. Feddema and C. Lee, Adaptive image feature prediction and control for visual tracking with a hand-eye coordinated camera, *IEEE Trans. Systems, Man and Cybernetics, 20*(5) (1990) 1172–1183.

[31] S. Skaar, W. Brockman and W. Jang, Three-dimensional camera space manipulation, *Int. J. Robotics Research, 9*(4) (August 1990).

[32] C. Torras, Neural learning for robot control, *Proc. 11th Euro. Conf. on Artificial Intelligence (ECAI '94), Amsterdam, Netherlands*, August 1994, pp. 814–819.

[33] H. Hashimoto, T. Kubota, M. Kudou and F. Harashima, Self-organizing visual servo system based on neural networks, *IEEE Control Systems* (April 1992).

[34] P. Corke, Dynamics of visual control, *IEEE Int. Conf. on Robotics and Automation, Notes for Workshop M-5 (Visual Servoing), San Diego, CA,* May 1994.

[35] B. Ghosh, Nonlinear estimation schemes for visual servoing, *IEEE Int. Conf. Robotics and Automation, Notes for Workshop M-5 (Visual Servoing), San Diego, CA,* May 1994.

[36] W. Miller, Sensor-based control of robotic manipulators using a general learning algorithm, *IEEE Trans. Robotics and Automation, 3*(2) (April 1987).

[37] G. Cembrano, C.Torras and G. Wells, Neural networks for robot control, *Proc. IFAC Symposium on Artificial Intelligence in Real-time Control – AIRTC'94*, Pergamon Press, Oxford, 1994.

[38] J. Feddema, C. Lee and R. Mitchell, Feature-based visual servoing of robotic systems, in K. Hashimoto (ed.), *Visual Servoing*, World Scientific, Singapore, 1993, pp. 105–138.

[39] G. Wells, An introduction to neural networks, in L. Boullart, A. Krijgsman and R. Vingerhoeds (eds.), *Application of Artificial Intelligence in Process Control*, Pergamon Press, Oxford, 1992.

[40] B. Li, A new computation of geometric moments, *Pattern Recognition, 26*(1) (1993) 109–113.

[41] X. Jiang and H. Bunke, Simple and fast computation of moments, *Pattern Recognition, 24*(8) (1991) 801–806.

[42] G. Wells and C. Venaille, Inverse task Jacobian module: optimal neural network training with geometric moments and correlation, *Technical Report*, CONNY (ESPRIT III, No. 6715), No SG1-WP-CIB-06.94/1 Instituto de Cibernética, Barcelona, June 1994.

[43] C. Venaille, P. Adam, J. Thévenon, G. Wells and E. Lesaint, Industrial inspection robot, *Technical Report*, CONNY (ESPRIT III, No. 6715), Thomson Broadcast Systems, Rennes, France, March 1995.