Uncalibrated Visual Servoing

Mike Spratling Roberto Cipolla
Department of Engineering,
University of Cambridge,
Cambridge. CB2 1PZ.
mws20@eng.cam.ac.uk cipolla@eng.cam.ac.uk

Abstract

Visual servoing is a process to enable a robot to position a camera with respect to known landmarks using the visual data obtained by the camera itself to guide camera motion. A solution is described which requires very little *a priori* information freeing it from being specific to a particular configuration of robot and camera. The solution is based on closed loop control together with deliberate perturbations of the trajectory to provide calibration movements for refining that trajectory. Results from experiments in simulation and on a physical robot arm (camera-in-hand configuration) are presented.

1 Introduction

Visual servoing is a process by which the appearance of landmarks is used to control the positioning of the camera with respect to the world. The camera is thus the sensor for a control scheme, in which the position of the camera itself is the object of control. The objective is for a robot to position the camera in a specific 'target' pose (defined at initialisation) with respect to one or more landmarks. The robot commences servoing at a 'start' location from which these landmarks are visible.

Visual servoing is potentially useful for situations where a robot must orientate itself with respect to the world; for navigation purposes with a mobile robot, and for grasping and insertion operations with a robot arm. Several applications for tracking or grasping moving objects are described in the literature [1] but only application to static targets is considered here.

2 Previous Work

Visual servoing techniques are classified in terms of the parameters supplied to the control strategy [1, 2]. Image-based control is exercised in response to the image features directly. Position-based control is exercised in response the estimated pose of the camera which is recovered from the image features (figure 1).

Image-based control simplifies the process significantly and is thus commonly used (*e.g.* [3, 4, 5, 6, 7, 8]). By defining an error signal in terms of image features

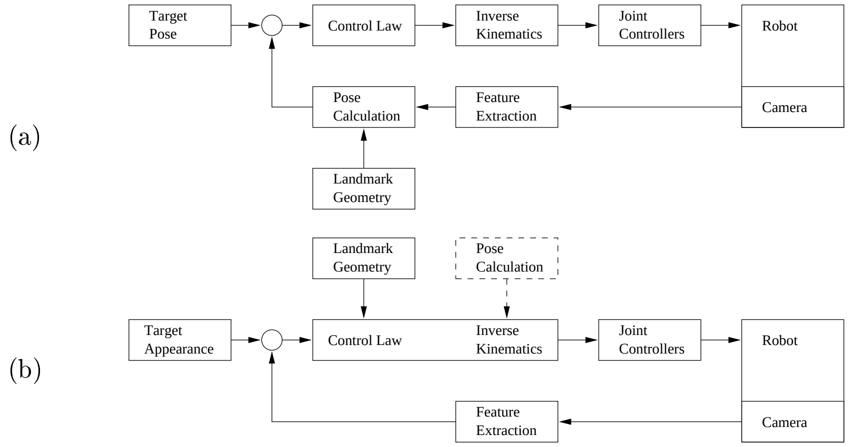


Figure 1: Standard Visual Servoing Methods (a) Position-Based, (b) Image-Based

a control law can be defined, *e.g.* [6] uses:

$$\delta\vec{\beta}(t) = -\left(\mathbf{L}^T(t)\mathbf{Q}\mathbf{L}(t) + \mathbf{R}\right)^{-1}\mathbf{L}^T(t)\mathbf{Q}\left(\mathbf{x}(t) - \mathbf{x}_{target}(t+1)\right) \tag{1}$$

where: $\delta\vec{\beta}$ is the required change in camera pose,
$\mathbf{Q}$ and $\mathbf{R}$ are control weighting matrices,
$\mathbf{L}$ is the interaction matrix,
$\mathbf{x}$ is the image coordinates of landmark points,
$\mathbf{x}_{target}$ is the desired image coordinates of landmark points.

Essentially this control law uses a matrix $\mathbf{L}$, called the interaction matrix, to relate the error in image coordinates of landmark points, $\mathbf{x}$, to the required change in camera pose, $\vec{\beta}$. Some inverse kinematics is then required to convert this required change in pose into actuator commands.

The interaction matrix is pose specific. It is often predefined for a particular pose and type of landmark (such as plane surfaces, cylinders, points, *etc.*), or is extracted from deliberate movements of the camera. Once defined the interaction matrix can be kept constant if the start location is close to the target location (in which case repeated application will converge to the target). Alternatively the interaction matrix is updated as the camera moves by approximating how the change in pose will affect the interaction matrix [4, 6, 7] (which requires the recovery of some 3D structure).

3 Proposed Method

Image-based visual servoing is preferable since it is faster, simpler, and offers the possibility of significant implementation non-specificity compared to position-based methods. However, most current visual servoing methods do not take advantage of the possibility to extract the relation between image features and robot motion at run-time and instead pre-define the interaction matrix for particular landmarks and use robot specific kinematics to achieve the desired change in pose. The aim of this work was to develop a visual servoing technique free from such implementation specific details.

More generally than in the previous section an interaction matrix can be defined to relate visual appearance, $\vec{\omega}$, directly to motor actions, $\vec{\alpha}$:

$$\delta\vec{\omega} = \mathbf{L}\delta\vec{\alpha} \tag{2}$$

This is an image-motor rather than an image-pose interaction matrix as is used in equation 1. It results in the simplified visual servoing scheme shown in figure 2 and described below.

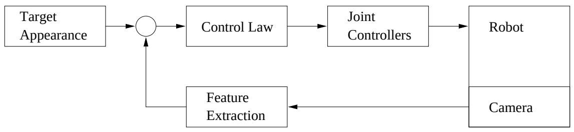


Figure 2: Proposed Visual Servoing Method

3.1 Image Features

If it is assumed that the viewing geometry is such that the formation of the image of a landmark can be modelled by weak perspective [9], then two images of a planar surface, ${}^{1}\mathbf{x}$ (at pose 1) and ${}^{2}\mathbf{x}$ (at pose 2), can be related by an affine transformation:

$${}^{2}\mathbf{x} = {}^{2}_{1}\mathbf{A}\ {}^{1}\mathbf{x}$$

Where ${}^{2}_{1}\mathbf{A}$ is the matrix of the affine transformation from pose 1 to pose 2. The parameters of the affine transformation can be converted into the differential invariants of the image displacement or velocity field [10]:

$${}^{2}_{1}\mathbf{A} = \frac{1}{2}\begin{pmatrix} div\mathbf{v} + def_1\mathbf{v} + 1 & def_2\mathbf{v} - curl\mathbf{v} & t_x \\ def_2\mathbf{v} + curl\mathbf{v} & div\mathbf{v} - def_1\mathbf{v} + 1 & t_y \end{pmatrix}$$

Which are used in preference to the affine parameters as they are related to the scene structure in a more intuitive manner: $div\mathbf{v}$ is the divergence or scale change, $curl\mathbf{v}$ is the curl or image plane rotation, and $def_1\mathbf{v}$ and $def_2\mathbf{v}$ are the two components of deformation or pure shear. A vector can thus be derived to represent the change in image appearance from pose 1 to pose 2:
${}^{2}_{1}\vec{\omega} = \begin{pmatrix} div\mathbf{v} & curl\mathbf{v} & def_1\mathbf{v} & def_2\mathbf{v} & t_x & t_y \end{pmatrix}^T$. Each landmark contour thus provides 6 parameters for the required change in image appearance to control the pose of the camera in 6 degree of freedom (d.o.f.) space. Planar landmarks have been chosen since they are common, easy to find, and sufficient information can be extracted without any requirement to match corresponding points (see section 4) or to know the contour shape.

Under weak perspective there is a pose ambiguity such that two distinct poses share the same visual appearance. This is due to there being no depth information to distinguish which points on the contour are furthest from the camera: Consider a hemisphere centred on the camera such that each point on the the surface of the hemisphere defines the landmark position and orientation then points at equal depth but on opposite sides of the hemisphere will define views sharing the same visual appearance under weak perspective.

3.2 Algorithm

With the definition given in equation 2, the interaction matrix becomes robot as well as pose specific, as it includes a local approximation of the inverse kinematics.

The image-motor Jacobian, is a local, linear approximation, to this interaction matrix. It can be estimated by measuring the change in visual appearance during deliberate movements of the robot along each of its d.o.f.s:

$$\mathbf{L}_{ij} \approx \frac{\Delta\omega_i}{\Delta\alpha_j} = \frac{\left({}^{current}_{target}\omega_i\right) - \left({}^{perturb}_{target}\omega_i\right)}{\left({}^{current}\alpha_j\right) - \left({}^{perturb}\alpha_j\right)}$$

for $1 < i < 6n$ (where n is the number of landmarks),
and $1 < j < m$ (where m is the number robot actuators),
where $\Delta\vec{\omega}$ is the measured change in image appearance,
$\Delta\vec{\alpha}$ is the change in robot actuator states.

Having performed these calibration movements the required motion can be calculated from the (pseudo-)inverse, $\mathbf{L}^{+}$, of $\mathbf{L}$:

$$\Delta\vec{\alpha}^{reqd} \approx \mathbf{L}^{+}\left[\left({}^{current}_{target}\vec{\omega}\right) - \left({}^{target}_{target}\vec{\omega}\right)\right] = \mathbf{L}^{+}\left({}^{current}_{target}\vec{\omega}\right)$$

It is possible to reduce the number of calibration movements required by using fixation. The motion axes which achieve fixation do not then need to be included in the Jacobian. Despite the reduction in the number of calibration movements thus achieved updating the interaction matrix as pose changes is costly and in some applications could interfere with the task. Motion towards the target would also be fitful and inelegant. Avoiding separate calibration phases entirely would seem to be desirable. Updating the image-motor interaction matrix analytically is more difficult than with an image-pose interaction matrix and the knowledge of object pose and robot kinematics that would be required to achieve this would defeat the aims of this research. Some information about the required modification to the interaction matrix can be obtained by comparing the measured change in appearance after each motion with that predicted by equation 2 [11], but the refinement thus produced can be only approximate since the error along a single trajectory is measured while the robot will have more than one degree of freedom.

The algorithm proposed here does integrate the calibration movements with the motion towards the target, to form a continuous trajectory. As the camera moves the change in appearance of the landmarks is measured. The motion is then perturbed by a deliberate change in speed of one axis and the effect this has on the appearance of the landmarks, in comparison with the effect expected from the previous (unperturbed) trajectory is used to calculate a Jacobian, **Lj**, for this motion axis, j, individually:

$$\mathbf{Lj}_i \approx \frac{\Delta\omega_i}{\Delta\alpha_j} = \frac{\left[\left({}^{current}_{target}\omega_i\right) - \left({}^{perturb}_{target}\omega_i\right)\right] - \left[\left({}^{previous}_{target}\omega_i\right) - \left({}^{current}_{target}\omega_i\right)\right]}{\left({}^{current}\alpha_j\right) - \left({}^{perturb}\alpha_j\right)}$$

for $1 < i < 6n$ (where n is the number of landmarks).

Hence, an estimate is made of the required motion along this axis:

$$\Delta\alpha_j^{reqd} \approx \mathbf{Lj}^{+}\left({}^{current}_{target}\vec{\omega}\right)$$

and this estimate is used to update the trajectory. The process is then be repeated for each robot motion axis in turn. And the entire cycle iterated until the target appearance is achieved.

The continuous motion of the camera envisioned in the above description has in practice been implemented with the movements split into discrete steps. The maximum permissible movement each iteration is limited to attempt to approximate closed-loop control, and since the Jacobians are only locally valid (and are also only approximate). Such limits can be set adaptively: If the landmarks are stationary a large change in appearance when making a perturbation indicates that motion should be slowed along that axis, while a small change in appearance indicates that the limit on movement should be increased.

4 Implementation

The algorithm described was developed and tested in simulation (using Matlab) and with a physical robot arm (a Scorbot ER-VII) with a CCD camera mounted in the end-effector. Both setups used monocular vision. Only static control is considered; dynamics are ignored in the simulations and a separate dedicated robot controller executes the positioning commands sent to the physical robot.

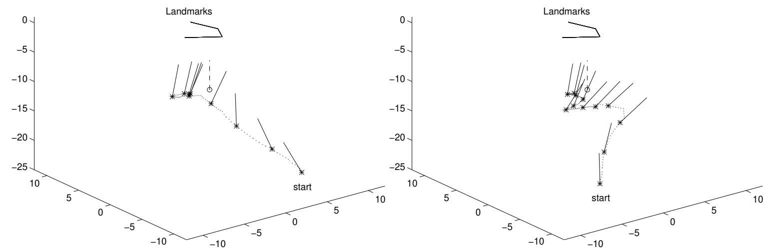
The simulation enables the camera to move with 6 d.o.f.s. The simulation places no restrictions on valid poses unlike a physical robot arm which can only reach certain configurations. Also, each motion axis is aligned with a camera axis. The type of robot simulated is thus similar in configuration to a flying insect such as a bee. It provides an experiment of navigation for a mobile robot rather than a simulation of the robot arm. Landmarks are defined as coplanar points, which are matched (perfectly) across images, and imaged under perspective projection. The camera field of view is not bounded and hence there are no problems with landmarks being lost from view.

The robot arm had 5 d.o.f.s. The dedicated robot controller allowed control in terms of both cartesian coordinates and joint angles. The pose of the robot affects the camera motion induced by a particular motion axis in both cases. In the simulations the affine transformations were calculated via the image coordinates of corresponding landmark points. Such distinguished points are rarely available in real images. However, the affine transformation can be estimated from circular moments of corresponding contours [12] (this estimate is only accurate for small image distortions but has proved sufficiently correct for use here despite the large differences in camera pose). Such a method was implemented on B-splines fitted to the landmark contours. Active B-spline contours, or snakes [13], constrained to deform affinely, were used for tracking the landmarks on the image [10]. This method, although not requiring point correspondences, does still require contour correspondences. Affine invariant signatures can be used for contour matching [14], but have not been included in this implementation.

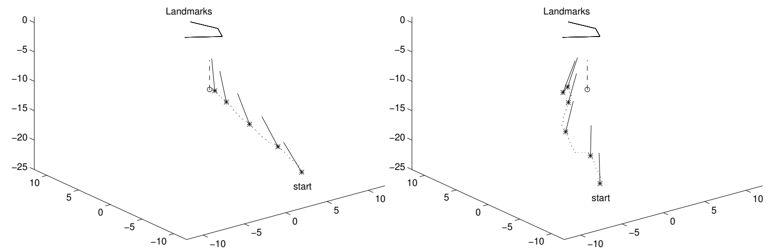
Fixation is easily achieved with the simulated robot where motion axes are aligned to provide pure rotations about the image plane x and y axes and the required angle of rotation is purely a function of the image position of the fixation point. For the robot arm fixation is not such a simple task as the camera motions induced by robot axes change with pose. However, the wrist roll axis (that nearest to the end-effector) induces image motion along the x-axis while the wrist pitch joint (the next nearest to the end-effector) induces image motion along the x-axis and/or the y-axis dependent on pose. By measuring, in the image, the movement of the fixation point induced by these axes the required rotations can be approximated and fixation achieved after a few iterations.

5 Results

Figure 3 shows some examples of the trajectories achieved using the method described above in simulation, using a single landmark, with and without fixation (*i.e.* to control 4 or 6 d.o.f.s). The trajectories generated without fixation are less direct than those achieved with fixation. Without fixation translational motions are sometimes used to attempt to correct errors caused by incorrect orientation. Such problems do not occur when fixation is used as the best orientation is always achieved before the translational motion is calculated. The initial trajectory for each experiment was along the optical axis in a sense determined by $div\mathbf{v}$ (the isotropic expansion or contraction of the contour which provides an indication of whether the camera needs to move towards or away from the landmark).



(a)

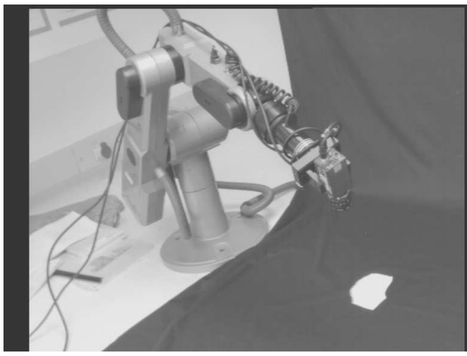


(b)

Figure 3: Examples of camera movement achieved in simulation (a) without fixation movements (b) with fixation movements. The optical centre of the camera is marked with a '$*$' and its direction of view is the direction of the line. Target pose is at coordinates (0,0,-10), marked with a 'o', facing vertically

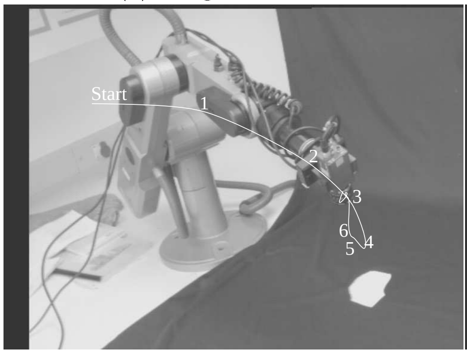
The robot does not have an axis of motion which will always produce camera translation along the optical axes, but the algorithm will work for any random initial movement; it will just be slower to reach the target. With all experiments using the robot fixation was performed at each step, since unlike in the simulation the camera field of view is bounded and fixation is useful to prevent losing sight of the landmarks. The robot has successfully servoed using one, two and three landmark contours using both joint and cartesian control of the robot. Results are shown in figures 4 and 5.

The algorithm is essentially an optimisation to null the affine transformation using gradient descent. Iteratively calculating the full Jacobian also provides a

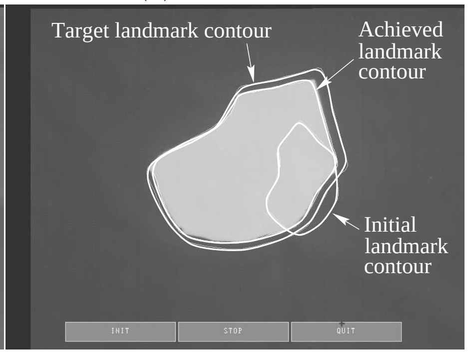


(a) Target Position

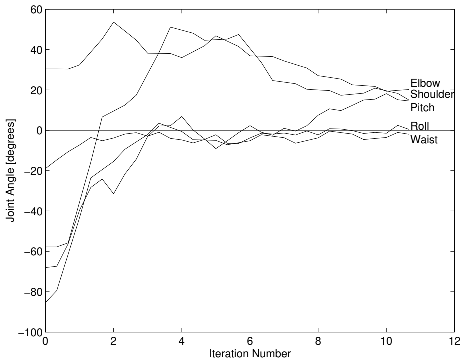
(b) Start Position



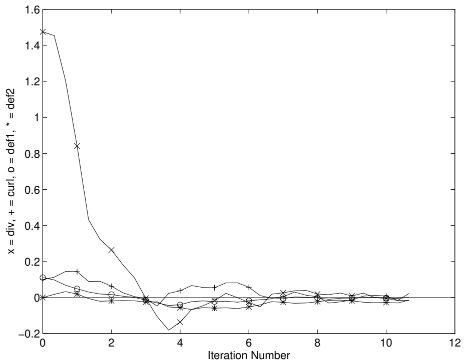
(c) Achieved Position



(d) Landmark Appearance

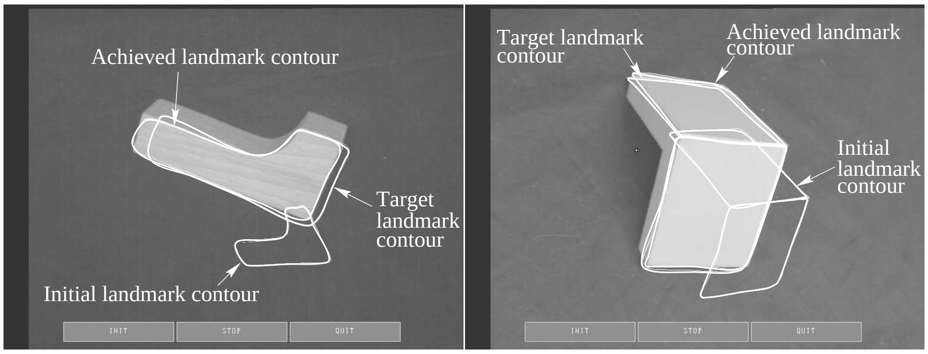


(e) Change in Robot Position Error



(f) Change in Image Appearance Error

Figure 4: Result from an experiment using a 5 d.o.f. robot arm with the camera in the end-effector servoing using one landmark contour. The configuration of the robot is shown at (a) the target position, (b) the start position, and (c) the position achieved after 10 iterations of the algorithm (the path taken is drawn with numbers indicating camera position at each iteration). (d) compares the landmark contour appearances at each of these configurations. The change in the positioning error with time is shown in terms of (e) the difference between the target robot joint angles and those at each iteration, and (f) the change in the parameters for the required change in image appearance. The direct path translation of the camera between start and target locations was approximately 450mm, and the longest dimension of the landmark is 126mm.



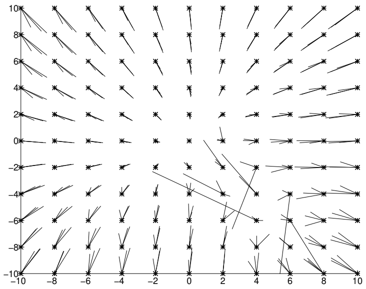
(a) (b)

Figure 5: Further results from experiments using the 5 d.o.f. robot arm using (a) one landmark contour, and (b) two landmark contours. The number of iterations of the algorithm used to generate these results was (a) 5 iterations, (b) 24 iterations. The direct path translation of the camera between start and target locations was approximately (a) 500mm, (b) 250mm, and the longest dimension of each block is approximately 100mm.

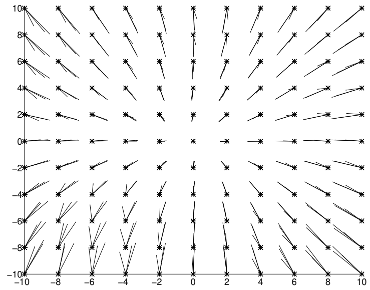
gradient descent method but one where the trajectory along the steepest gradient is found. Figure 6 is a plan view of the situation depicted in figure 3 to illustrate the trajectories calculated throughout the volume in which visual servoing takes place. The algorithm will succeed if the trajectories predicted always reduce the affine parameters, without becoming stuck in a local minima where no further gradient descent can occur but which is not at the target. The pose ambiguity under weak perspective (section 3.1) is such a local minima, however, under full perspective the pose ambiguity does not exist. The artifacts introduced into the affine parameters by perspective effects are small but near the pose ambiguity become dominant inducing large erroneous movements to be predicted (figure 6 (a)). The pose ambiguity is thus a repeller rather than a local minimum under full perspective. The pose ambiguity is removed if a second non-coplanar landmark is available (figure 6 (b)). More than one landmark is thus desirable not only to increase the information available but to suppress erroneous movements caused near the pose ambiguity. For the robot implementation affine contour tracking is used causing the effective camera model to be closer to weak perspective, but no local minimum has been found in practice when servoing on a single contour.

6 Discussion

This paper has presented a novel, and yet very simple, algorithm for visual servoing. By constraining the type of landmarks used to be planar contours a method has been developed which requires no *a priori* information about the landmark geometry or robot configuration, and does not require distinguished points, the recovery of any 3D structure, proprioceptive measurements, nor a kinematic model of the robot, all of which represent improvements over previous methods. Instead all necessary information is extracted at run-time. The inelegance of separate cal-



(a)



(b)

Figure 6: The x and y components of movements calculated from each point marked with '$*$' for 3 depths in the z-axis (each line points in the direction and has a length proportional to the magnitude of the movement). The movements were estimated using the full Jacobian for (a) one landmark, and (b) two landmarks. The target location is at (0,0) so ideally all movement vectors should point towards the centre. The first landmark has a pose ambiguity under weak perspective in the lower right quadrant, the second landmark introduced in (b) has a pose ambiguity under weak perspective in the upper left quadrant.

ibration movements is avoided by using a closed-loop control scheme relying on perturbations of the trajectory to update the estimate of the required trajectory.

The lack of *a priori* knowledge required about the camera/robot system makes this algorithm very general as has been demonstrated by its application to two entirely different robot morphologies. A reduction in the number of controlled axes, and an increase in performance, is achieved by implementing fixation movements at the expense of making the method more robot specific. Further improvement in performance gained from robot specific knowledge comes from identifying an axis of motion which can induce motion along the camera optical axis.

The method has been developed to apply to the situation where the target location is not close to the start location, and hence when a single interaction matrix is not sufficient. Although only static targets have been considered the algorithm has been slightly extended, and tested in simulation, to servo relative to targets in motion.

The method can be slow and seldom provides a direct path to the target location. Some improvement to the method might be achieved if more tightly closed-loop control was exercised. Ideally the image processing to calculate the Jacobian, would be performed while the robot was moving, enabling continuous updating of the estimated trajectory (alternatively smaller steps could be taken). This would avoid jerky motion, produce finer control, and enable trajectory errors to be corrected earlier. However, large movement steps are need to increase signal-to-noise ratio because snake tracking and the calculation of the affine transformation are inexact when using real image data.

Acknowledgements

Nick Hollinghurst and Jun Sato donated some software used in this project. Financial support, for the first author, was provided by a Schiff Scholarship.

References

[1] P. I. Corke. 'Visual Control of Robot Manipulators - A Review'. In K. Hashimoto, editor, *Visual Servoing.* World Scientific, 1993.

[2] A.C. Sanderson and L.E. Weiss. 'Adaptive Visual Servo Control of Robots'. In A. Pugh, editor, *Robot Vision.* IFS Publications Ltd., 1983.

[3] F. Chaumette. 'Visual Servoing Using Image Features Defined Upon Geometrical Primitives'. In *Proc. 33rd Conf on Decision and Control*, pages 3782–3787. IEEE, 1994.

[4] C. Colombo and J. L. Crowley. 'Uncalibrated Visual Tasks via Linear Interaction'. In Burnard Buxton and Roberto Cipolla, editors, *ECCV'96 Proceedings of the 4th European Conference on Computer Vision*, pages 583–592, 1996.

[5] B. Espiau, F. Chaumette, and P. Rives. 'New Approach to Visual Servoing in Robotics'. *IEEE Trans. Robotics and Automation*, 8(3), 1992.

[6] B. Nelson and P.K. Khosla. 'Integrating Sensor Placement and Visual Tracking'. In *Proc. 3rd Int. Symp. on Experimental Robotics*, 1993.

[7] C.E. Smith, S.A. Brandt, and N.P. Papanikolopoulos. 'Robotic Exploration Under the Controlled Active Vision Framework'. In *Proc. 33rd Conf on Decision and Control*, pages 3796–3801. IEEE, 1994.

[8] L.E. Weiss, A.C. Sanderson, and C.P. Neuman. 'Dynamic Sensor-based Control of Robots with Visual Feedback'. *IEEE Trans. Robotics and Automation*, 3(5):404–417, 1987.

[9] L.G. Roberts. 'Machine Perception of Three-Dimensional Solids'. In J.T. Tippet, editor, *Optical and Electro-Optical Information Processing.* MIT Press, 1965.

[10] R. Cipolla and A. Blake. Surface orientation and time to contact from image divergence and deformation. In G. Sandini, editor, *Proc. 2nd European Conference on Computer Vision*, pages 187–202. Springer–Verlag, 1992.

[11] M. Jägersand and R. Nelson. 'Adaptive Differential Visual Feedback for Uncalibrated Hand-Eye Coordination and Motor Control'. Technical Report TR 579, Department of Computer Science, University of Rochester, 1994.

[12] J. Sato and R. Cipolla. 'Image Registration Using Multi-Scale Texture Moments'. *Image and Vision Computing*, 13(5):341–353, 1995.

[13] A. Blake, R. Curwen, and A. Zisserman. 'A Framework for Spatio-Temporal Control in the Tracking of Visual Contours'. *Int. Journal of Computer Vision*, 11(2):127–145, 1993.

[14] J. Sato and R. Cipolla. 'Affine Integral Invariants and Matching of Curves'. In *Proc. International Conference on Pattern Recognition*, August 1996.