Visual Tracking and Control using Lie Algebras

T. Drummond
Department of Engineering
University of Cambridge
Cambridge, UK CB2 1PZ
twd20@eng.cam.ac.uk

R. Cipolla
Department of Engineering
University of Cambridge
Cambridge, UK CB2 1PZ
cipolla@eng.cam.ac.uk

Abstract

A novel approach to visual servoing is presented, which takes advantage of the structure of the Lie algebra of affine transformations. The aim of this project is to use feedback from a visual sensor to guide a robot arm to a target position. The sensor is placed in the end effector of the robot, the 'camera-in-hand' approach, and thus provides direct feedback of the robot motion relative to the target scene via observed transformations of the scene. These scene transformations are obtained by measuring the affine deformations of a target planar contour, captured by use of an active contour, or snake. Deformations of the snake are constrained using the Lie groups of affine and projective transformations. Properties of the Lie algebra of affine transformations are exploited to integrate observed deformations to the target contour which can be compensated with appropriate robot motion using a non-linear control structure. These techniques have been implemented using a video camera to control a 5 DoF robot arm. Experiments with this implementation are presented, together with a discussion of the results.

1 Introduction

The use of real-time video information for robotic guidance is increasingly becoming a more attractive proposition. Advances in the power and availability of image processing capabilities have made possible the tracking of complex features, such as surface contours [1], at frame or field rate on standard workstations. This has enabled visual servoing of sufficient accuracy that many useful tasks may now be accomplished.

Here we present an approach which takes advantage of the structure of the Lie algebra of affine transformations to provide an accurate, efficient and stable servoing system. This approach provides a substantial advantage over traditional image-based visual servoing techniques due to the manner in which the structure of the two-dimensional affine transformation group is used to implicitly embed three-dimensional knowledge within an image based system.

The applications which motivate this work are tasks such as welding of automotive or ship parts. These tasks are characterised by the need to accurately place a tool onto a workpiece which may be inaccurately located in relation to the robot. The use of vision to assist in solving these tasks is attractive because of the capability to respond accurately and rapidly to errors in part placement that are difficult to detect by other means.

1.1 Background

The approach proposed here, makes use of image based visual servoing [2, 3], in which the control loop is closed in the two-dimensional image domain rather than in the three-dimensional workspace. This is by contrast to systems which explicitly use the three dimensional nature of the world [4, 5] (often referred to as position-based visual servoing systems).

The two dimensional approach works in the image domain, and computes error measurements directly, such as the (x,y) location of target points [3]. The system then attempts to move so as to minimise this observed error typically using a Jacobian which relates robot motions to changes in observables.

There are two natural locations in which to mount a visual sensor [6], namely static with respect to the world, and static relative to the robot. Our system uses the latter approach with a single camera mounted in the robot's end effector. This constrains the visual servoing problem since the sensor directly measures the relationship between the robot and workpiece. There is the additional benefit of scaling; as the robot approaches the target position, the size of features on the workpiece grows in the image plane.

This work uses the principle of teaching by showing, in which the supervisor shows the system the correct location by placing the robot in the desired relative pose to the target. The system then learns this pose by storing sufficient reference information, captured whilst in that pose, to characterise the three-dimensional relationship between the end effector and the workpiece. This is accomplished by observing one or more contours located on the surface of the workpiece. By recording the view of these contours from the correct target position, any observed deviations from this view can be detected and corrected for by appropriate robot motion.

Contours form a particularly useful feature to track for visual servoing for a number of reasons. Firstly, advances have been made in the robust tracking of contours such that they can be tracked reliably amongst clutter [7]. Secondly, an accurate measure of image motion can be obtained from contours due to the large number of measurements that can be made by integration of normal velocities around the contour [8]. Finally, a planar contour can be readily constrained to undergo affine or projective deformations. The affine deformation group, GA(2), has six dimensions and thus theoretically (and in practice) provides enough information to guide a robot through space. Since such motions can be described by the group of rigid three-dimensional motions, E(3), also having six dimensions. The use of the affine/projective constraint separates this method from previous work using deformable models such as [9] which relies on a more traditional dynamical model of active contours.

The robot control system is based on a Jacobian between the robot motions and the generators of the group of deformations of the contour. Properties of the Lie Algebra of this group are exploited to provide a consistent representation for integrating general affine (or other group) deformations to the contour. This approach allows a single Jacobian, computed once near the target location, to be used across a large range of perturbations.

1.2 System Overview

The system comprises two separate threads which operate concurrently within a workstation which receives a live video feed from the robot, and communicates directly to the robot controller, as illustrated in Figure 1.

1) The live video feed is delivered to the affine snake tracker module which computes a series of local transformations describing the deformation of the contour of interest. Velocities in transformation space are computed and maintained in order to assist in tracking rapid motions.

2) The local transformations computed by the affine snake are integrated using the Lie algebra of affine transformations to obtain an accurate measure of the the total transformation describing the current position of the contour.

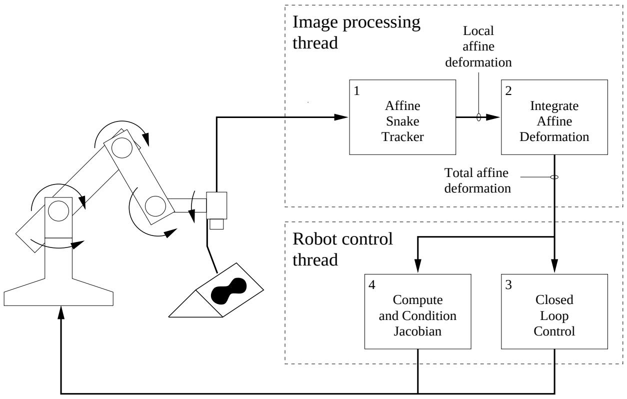

Figure 1: System architecture

3) The integral of affine transformation is passed to the robot control module which uses an affine-to-robot Jacobian, together with a non-linear control law to compute robot motions.

4) The robot control thread also contains a calibration module which computes and conditions the Jacobian by correlating trial motions of the robot with integrated deformations of the contour of interest.

2 Lie Groups and Affine Snakes

A Lie group is a group which locally has the topology of $\mathbb{R}^n$ everywhere (a more precise definition may be found in [10, 11]). There are a number of groups which are interesting for the purposes of this work and which can be defined by their action on $\mathbb{R}^2$ or $\mathbb{R}^3$. These are the groups E(3) (Euclidean transformations in three dimensions), GA(2) (general affine transformations in two dimensions) and P(2) (projective transformations in two dimensions). The dimension n of each group corresponds to the number of independent ways that a small (infinitesimal) transformation can be made. For E(3) and GA(2) this is 6,whereas P(2) has 8 dimensions.

GA(2) is the group of all linear transformations on two dimensional space. This describes the transformations that a planar image can undergo when viewed under weak perspective from a camera moving in three dimensions. The six dimensions of the group are commonly broken down into the differential invariants. These transformations can be represented by matrices in homogeneous co-ordinates. Those corresponding to pure transformations in each of the six modes of deformation, parameterised by α are:

$$M_1 = \begin{pmatrix} 1 & 0 & \alpha \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad M_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & \alpha \\ 0 & 0 & 1 \end{pmatrix}$$
$$M_3 = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} M_4 = \begin{pmatrix} e^{\alpha} & 0 & 0 \\ 0 & e^{\alpha} & 0 \\ 0 & 0 & 1 \end{pmatrix}$$
$$M_5 = \begin{pmatrix} e^{\alpha} & 0 & 0 \\ 0 & e^{-\alpha} & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad M_6 = \begin{pmatrix} \cosh\alpha & \sinh\alpha & 0 \\ \sinh\alpha & \cosh\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad (1)$$

P(2) is the group of all transformations on two dimensional projective space. This group has 8 dimensions which contain the 6 dimensions of GA(2) as a subgroup and include two additional dimensions which produce warping in the image. P(2) describes the transformations of a planar image under strong perspective.

2.1 Vector Fields and Lie Derivatives

The matrices in Equation (1) each describe a continuous one dimensional family of transformations on $\mathbb{R}^2$, parameterised by α. Thus for each matrix, for each α, a point (x, y) is mapped to some point (x', y'). Setting α to zero generates the identity transformation: $(x', y') = (x, y)$. Differentiating with respect to α and evaluating at $\alpha = 0$ creates a vector field:

$$L_i = \left. \frac{\mathrm{d}M_i(\alpha)\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}}{\mathrm{d}\alpha} \right|_{\alpha=0} \qquad (1 \le i \le 6) \qquad (2)$$

Since differentiation is a linear operation, writing

$$G_i = \frac{\mathrm{d}M_i(\alpha)}{\mathrm{d}\alpha} \qquad \text{gives:} \qquad L_i = G_i\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \qquad (3)$$

The matrices G_i are referred to as generators of the Lie Group and form a basis for the Lie Algebra discussed in more detail in Section 3. For GA(2), the generators are:

$$G_1 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} G_2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \quad G_3 = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$
$$G_4 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} G_5 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix} G_6 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \qquad (4)$$

The vector fields, L_i, are used to compute the affine transformation which describes the deformation of a contour in the robot's view. This is achieved through the use of affine snakes. The vector fields generated by the above matrices are:

$$L_1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} L_2 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad L_3 = \begin{pmatrix} -y \\ x \end{pmatrix}$$
$$L_4 = \begin{pmatrix} x \\ y \end{pmatrix} L_5 = \begin{pmatrix} x \\ -y \end{pmatrix} L_6 = \begin{pmatrix} y \\ x \end{pmatrix} \qquad (5)$$

2.2 Affine Snakes

In the system presented here, active contours (snakes) are used to track contours on the workpiece. These snakes are closed polygons with between 16 and 1024 vertices and are initialised by hand either inside or outside the contour of interest. The snake then expands out (or contracts) until strong edges are located in the image. Once the snake has locked on to these edges, it becomes constrained to undergo only deformations within some transformation group of interest.

This is achieved by searching for the contour in the image along the normal to the snake tangent at each node. This gives the measurement space one degree of freedom per node on the snake. The measurement is then projected down onto the subspace defined by the transformation group of interest using singular value decomposition to produce a least squares fit. The transformation group subspace is identified by computing the optical flow created by each generator of the transformation group at each node of the snake, in a direction normal to the tangent of the snake at that node.

The snake tracks the contour using an estimate of the velocity of deformation of the contour in general affine transformation space. This is updated using the measured deviation between the prediction and observed contour to provide a reliable estimate of transformation velocity (see Figure 2). The velocity estimate is then combined with the observed deviation to predict the position of the contour at the next time step:

$$U_i^{t_2} = E_i^{t_1} + (t_2 - t_1)V_i^{t_1} \qquad (6)$$
$$V_i^{t_2} = V_i^{t_1} + \alpha_i E_i^{t_2}/(t_2 - t_1) \qquad (7)$$

where α_i are coupling constants chosen so as to damp oscillatory behaviour in the snake and $t_2 - t_1$ is the time elapsed in frames since the previous observation. This allows the system to cope with missing or intermittent video frames with graceful rather than catastrophic failure.

The weak perspective assumption used to compute the affine deformation of the contour does not entirely hold in the situation presented here. This means that if only affine deformations are used to track the contour, the tracker cannot deform to properly match the shape of the observed contour. Consequently, the range of deformations is extended to the full projective group in such a way that the affine component of the observed deformation is left intact. This independence is achieved by computing the warp vector fields relative to the centroid of the snake. The two additional vector fields which provide this compensation are:

$$L_7 = \begin{pmatrix} x'y' \\ y'^2 \end{pmatrix} \; L_8 = \begin{pmatrix} x'^2 \\ x'y' \end{pmatrix} \qquad (8)$$

where x' and y' are relative to the centroid.

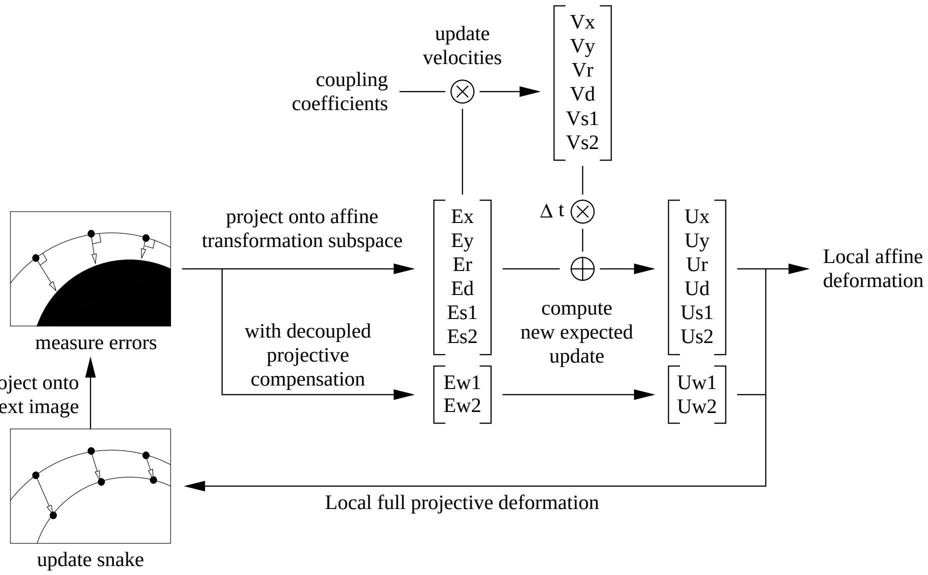

Figure 2: Affine snake system

3 Lie Algebras and Affine Integration

There is a natural representation for affine transformations in terms of matrices in homogeneous co-ordinates, such as those shown in Equation (1). However, an alternative is to use a co-ordinate system to represent small transformations near the identity. In this co-ordinate system, the axes correspond to the different modes of deformation and affine transformations are specified as a weighted sum of the group generators added to the identity. This leads naturally to a local vector space representation for infinitesimal transformations, in which an affine transformation matrix, A can be obtained from a vector, $\mathcal{A}$ by the exponential map:

$$A = e^{\sum_i \mathcal{A}_i G_i} \tag{9}$$

$$\text{where} \quad e^X = I + X + \frac{1}{2}X^2 + \frac{1}{6}X^3 + \ldots$$

For small $\mathcal{A}_i$ this can be approximated by the linear term and the G_i form a basis for a vector space, known as a Lie Algebra. Formally, a Lie Algebra is a vector space together with a bilinear anti-symmetric operator, the Lie Bracket, satisfying the Jacobi identity:

$$[A,[B,C]] + [B,[C,A]] + [C,[A,B]] = 0 \tag{10}$$

Where a Lie Algebra is obtained from a group in the manner identified above, the Lie Bracket is defined by the commutator of the generators:

$$[\mathcal{A}, \mathcal{B}] = \mathcal{C} \tag{11}$$

where $\mathcal{C}$ is defined by

$$\sum_{i,j} \mathcal{A}_i \mathcal{B}_j (G_i G_j - G_j G_i) = \sum_k \mathcal{C}_k G_k \tag{12}$$

The commutation relations for GA(2) are shown in Table 1. The infinitesimal representation of the Lie Algebra can be extended by considering the exponential map for finite transformations. This defines a mapping from the Lie Algebra onto the group, thus providing a convenient way of representing affine transformations as vectors which, in this scenario, can be used to drive the robot control system. Because higher order terms are incorporated into the vector space by this method, it is no longer possible to naïvely add vectors together to obtain a vector representing the composite transformation. A new addition law must be found which preserves the non-commutativity of matrix multiplication so that the sum of two vectors is the vector

	G_1	G_2	G_3	G_4	G_5	G_6
G_1	0	0	$-G_2$	$-G_1$	$-G_1$	$-G_2$
G_2	0	0	G_1	$-G_2$	G_2	$-G_1$
G_3	G_2	$-G_1$	0	0	$2G_6$	$-2G_5$
G_4	G_1	G_2	0	0	$2G_5$	0
G_5	G_1	$-G_2$	$-2G_6$	$-2G_5$	0	$-2G_3$
G_6	G_2	G_1	$2G_5$	0	$2G_3$	0

Table 1: Commutation relations for GA(2):

(a) With Correction Terms (b) No Correction

Figure 3: Effect of correction terms on integral

representing the true product of the transformations.

$$\begin{aligned} \text{Thus} \quad & \mathcal{A} = \mathcal{B} + \mathcal{C} \\ \text{implies} \quad & e^{\sum_i \mathcal{A}_i G_i} = e^{\sum_i \mathcal{B}_i G_i} e^{\sum_i \mathcal{C}_i G_i} \end{aligned}$$

writing $A = \sum_i \mathcal{A}_i G_i$, (B,C similarly) gives

$$\begin{aligned} & I + A + \tfrac{1}{2}A^2 + \dots \\ & \quad = (I + B + \tfrac{1}{2}B^2 + \dots)(I + C + \tfrac{1}{2}C^2 + \dots) \end{aligned}$$

this is solved by setting

$$A = B + C + \tfrac{1}{2}[B, C] + \tfrac{1}{12}[C - B, [B, C]] + \dots \tag{13}$$

This expression carries over into the Lie Algebra, replacing the matrix commutator with the Lie Bracket. This result is important because the correction terms provide a method of consistently adding together vectors which represent group transformations.

This can then be used to integrate a series of affine deformations, so that the integral faithfully represents the total deformation. Figure 3 illustrates the importance of including these correction terms in the integral to track a complex series of transformations.

One of the key advantages of this representation (as opposed to parameterising in terms of the elements of the matrix in homogeneous co-ordinates, for example) is that straight lines through the origin of the vector space are geodesics in the manifold of the Lie Group. It is this property that allows a Jacobian computed in one place to operate correctly across a large range of perturbations.

4 Robot Control

The robot control thread takes the integral of affine transformation in the form of a 6-vector and uses a Jacobian to generate robot motions. This Jacobian is computed from a series of trial robot motions performed in the vicinity of the target location. Each of the six possible robot motions produces in a vector of

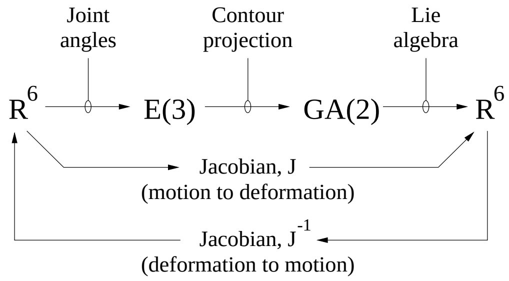

Figure 4: Computing the control Jacobian

integrated deformation, resulting in a 6×6 motion-to-affine Jacobian. This is inverted using SVD to compute the affine-to-motion Jacobian (see Figure 4).

This inverse Jacobian can then be used to compute compensatory motion from the observed integral of affine deformation. The control law is then simply:

$$\Delta R_j = -J_{ij}^{-1} A_i \tag{14}$$

where ΔR is the desired change in robot joint angles, J^{-1} is the affine-to-motion Jacobian and A is the integrated total affine transformation.

5 Results and Discussion

This approach has been implemented in the lab using a SCORBOT ER VII 5 DoF robot arm with a monochrome video camera connected to an SGI O2 workstation (see Figure 7). The workstation is able to track up to 256 snake nodes at video frame rate and control the robot with a cycle time of $0.5 - 1.5$ seconds. The viewing distance when in the target position was 200mm. Because the robot has only five degrees of freedom, the sixth was synthesised as rotation about the optical axis of the camera.

Three sets of experiments were conducted, in which first the part was perturbed, secondly the robot was perturbed and finally closed loop tracking was tested.

The first experiment aimed to test the range of possible perturbations of the workpiece from target position. The experiment was conducted by moving the robot back away from the target position to a preset starting position. The part was then perturbed and the robot asked to servo back to the target location. The translational perturbation was limited by the requirement to keep the contour within the image boundary, resulting in a maximum of approximately 200mm translational perturbation in x, y and z. The maximum rotational perturbation of the angle of inclination of the plane was limited to 20^o about a reference angle of $\theta = 40^o$ with a limit of 40^o about the horizontal axis perpendicular to it (see Figure 5).

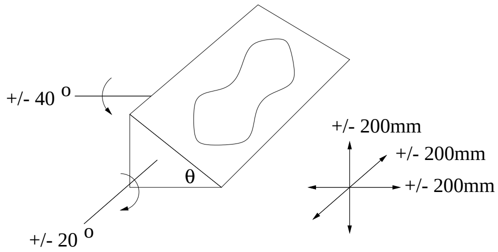

Figure 5: Range of stable perturbations

This experiment has been repeated using the range of contours shown in Figure 6. Contours 1–9 in this set represent an evolving series in which only contours 8 and 9 were sufficiently degenerate to prevent stable servoing. Contours 10 and 11 are almost triangular, and therefore exhibit degeneracy with respect to strong perspective distortions but are still stable under general affine tracking. Contours 1 and 12 provided the most stable results. The reference angle, θ, was also varied and the system was found to servo successfully with $10^o \leq \theta \leq 70^o$ although the range of permissible perturbations was limited at the extreme ends of this range.

The second experiment aimed to test the accuracy of placement of the robot under visual servoing. The workpiece was left fixed, and the robot asked to servo back to the target position from a series of random starting positions. The accuracy of positioning (1 standard deviation) of the camera in this experiment was ± 0.65mm in x and y and 0.3mm in z with 0.15^o in both pitch and roll. The maximum error measured at the tool tip over a series of runs was 1mm, with almost all errors being less than 0.5mm. This is of higher accuracy than the camera positioning due to correlations in the position and rotation errors.

The convergence rate was also computed, with the mean time to convergence being 2.5 cycles after the robot has reached its non-linear control zone.

Finally, closed loop tracking was tested (see Figure 7) which aimed to test the range of acceptable perturbations that can be tracked gradually under closed loop. In this mode substantial perturbations extending to approximately twice those shown in Figure 5 were successfully tracked.

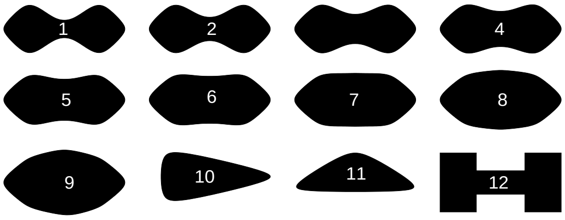

Figure 6: Range of contours used

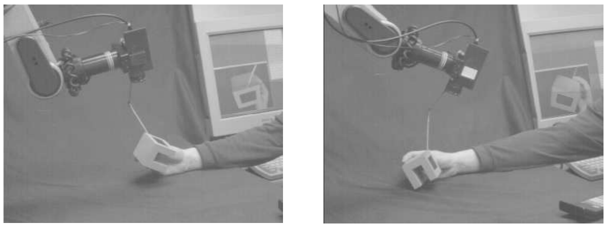

Figure 7: Closed loop visual servoing

Acknowledgements

This work was supported by an EC (ESPRIT) grant no. LTR26247 (VIGOR) and by an EPSRC grant no. K84202.

References

[1] M. Kass, A. Witkin, and D. Terzopoulous. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[2] A.C. Sanderson, L.E. Weiss, and C.P. Neumann. Dynamic sensor based control of robots with visual feedback. *IEEE Journal of Robotics and Automation*, 3:404–417, 1987.

[3] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE T-Robotics and Automation*, 8(3), 1992.

[4] R. Basri, E. Rivlin, and I. Shimshoni. Visual homing: Surfing on the epipoles. In *Proceedings of International Conference on Computer Vision (ICCV '98)*, pages 863–869, 1998.

[5] W.J. Wilson, C.C. Williams Hulls, and G.S. Bell. Relative end-effector control using cartesian position based visual servoing. *IEEE T-Robotics and Automation*, 12(5):684–696, 1996.

[6] S. Hutchinson, G.D. Hager, and P.I. Corke. A tutorial on visual servo control. *IEEE T-Robotics and Automation*, 12(5):651–670, 1996.

[7] M.A. Isard and A. Blake. Visual tracking by stoachastic propagation of conditional density. In *Proceedings of the 4th European Conference on Computer Vision*, pages 343–356, 1996.

[8] R. Cipolla and A. Blake. Image divergence and deformation from closed curves. *International Journal of Robotics Research*, 16(1):77–96, 1997.

[9] P.A. Couvignon, N.P. Papanikolopoulos, M. Sullivan, and P.K. Khosla. The use of active deformable models in model-based robotic visual servoing. *Journal of Intelligent and Robotic Systems*, 17(2):195–221, 1996.

[10] V.S. Varadarajan. *Lie Groups, Lie Algebras and Their Representations.* Number 102 in Graduate Texts in Mathematics. Springer-Verlag, 1974.

[11] D.H. Sattinger and O.L. Weaver. *Lie groups and algebras with applications to physics, geometry, and mechanics.* Number 61 in Applied Mathematical Sciences. Springer-Verlag, 1986.