Temporal Registration for Assembly*

S. S. Ravela T. J. Schnackertz Jr. R. A. Grupen R. S. Weiss
E. M. Riseman A. R. Hanson

Department of Computer Science, University of Massachusetts at Amherst

Abstract

The role of visual registration in executing assembly tasks is examined. A framework to execute assembly operations is presented in which assembly tasks are executed using a finite state machine that supervises the interaction between registration and control components. Temporal registration is achieved visually using a camera pair by combining image based feature tracking and weak camera-object calibration. A peg-in-hole insertion operation is demonstrated.

1 Introduction

Automated mechanical assembly operations can be decomposed broadly in to *task planning* and *task execution*. Our model for automated assembly begins with an assembly plan [15, 16] and constructs a contact state adjacency graph [5] (critical points). This graph relates discernible contact states as landmarks en-route to the goal state and sorts them based on the ambiguity of the sensor data they will produce. A sequence of these contact states must be achieved within associated tolerances for successful assembly. A critical point defines a *fits* relationship that is best represented as poses and associated tolerances. A robust assembly plan establishes a *fits* relationship between bodies by designating a sequence of feature relationships in pose space and their associated sensory signature. Sensory information includes force and pose in general, however, in this paper we are concerned with pose exclusively.

The sequence of sub-goals of relative poses output by the planning component can be compiled into a stochastic finite state machine (FSM) [11]. A state within this machine is a discernible assembly sub-goal. An arc in the supervisory FSM represents particular controllers that can be activated to effect a transition. In order to perform the assembly a path through the appropriate sub-goal states must be traversed using valid controllers available from the current state. The FSM supervisor represents an executable control strategy of the compiled plan, is largely device independent, and in addition, coordinates a sequence of valid local control actions that achieve global objectives while preserving responsiveness and suppressing run-time disturbances.

Task execution, begins with an FSM and the associated controllers. These controllers are constructed to suppress relative pose space errors to the next subgoal. In order to provide pose space errors (control signals), two mechanisms must be available. The first is a method to identify visual features in the assembly plan that unambiguously define body poses and that are sensitive to relevant pose errors. The second is a method to maintain temporal registration over part motion. Control can then be viewed as establishing and/or maintaining feature relationships on the image plane.

In this paper we focus on the temporal registration of part models using a camera pair as the sensor. Registration provides a visual basis for achieving a subset of critical points in an assembly. Assuming initial correspondence between a part model and visual data[1] temporal registration is maintained in two steps; First a weak perspective affine transform ($\mathcal{A}$) [13] is employed that expresses the relation between visual data and the object model. In this formulation, no information about the camera parameters or relative pose is required, resulting in a very weakly calibrated mechanism. The $\mathcal{A}$ transform can be computed given a minimum of four non-coplanar model data correspondences. The second step in maintaining registration is tracking of image features of model points [19]. The tracking technique employed in this paper is relatively insensitive to illumination changes and fully compensates for rigid 2D motion. Tracked locations are used to update the $\mathcal{A}$ transform thus, registering parts temporally.

In a peg-in-hole insertion task demonstrated in this paper we construct motor controllers that are driven by pose errors obtained from registered model points. Using the FSM supervisor we then demonstrate inser-

*The authors received support from ARPA and TACOM under contract DAAE07-91-C-R035 and NSF under grants IRI-9208920, IRI-9116297 and CDA-8922572.

[1] Experiments with part identification have been carried out with the model matching schemes presented in [2, 3].

tion to the point where object features become invisible. In a general assembly task, the visual component would cover a part of the critical points (subject to visibility) following which a different sensory modality such as contact force information [5, 18] must be employed.

The remainder of this paper is organized as follows. Section 2 looks at related literature. Section 3 describes the registration component. Section 4 details the peg-in-hole experiment. Conclusions are presented in section 5

2 Related Work

Temporal registration has been addressed by several researchers [6, 10, 17, 22, 23, 24]. Dickinson et. al. [6] use an aspect prediction graph together with a network of active contours introduced in [14]. Active contours are usually purely gradient based and can be sensitive to undesirable local edge maxima. Uenohara and Kanade [22] use normalized cross-correlation for tracking and combine it with pose estimation. Both Gennery and Walters [10, 24] employ a Kalman filter for predictive pose and edge based tracking. Lowe uses line data as image features with a weighted least squares fit to the model parameters. Matching itself is achieved via a best first search using Bayesian theory to measure the probabilities feature matches.

Tracking which is a central component in temporal registration has been addressed by using lines [17, 4, 20], edges [10, 24] including edge contours [6, 14] and intensity [22, 12, 21] including optic flow [1]. For example Crowley [4] used a set of parameterized line tokens which were matched using the Mahalanobis distance with predicted feature vectors. Sawhney [20] extended this approach to triples of lines, grouped under the shallow structure assumption under affine transformations. These techniques however are slow. Other model-based tracking such as [12] uses a hierarchy of features to represent a model. Constraints on the state of the feature are propagated down the hierarchy and at the lowest level tracking is accomplished using convolutions (edges) or SSD methods [1]. Hager does not explicitly address the issue of mismatches as is done by Shi and Tomasi [21]. Affine feature dissimilarity over multiple frames is used to identify good features to track.

Nelson et. al., examine the use of visual servoing [18] for assembly and propose three methods to reduce the calibration problem.

3 Registration Components

As specified in section 1, the $\mathcal{A}$ transform is computed from a minimum of four model-data correspondences. Over each subsequent image frame, part model coordinates are projected into each camera pair and are tracked. Tracked locations are used to update the $\mathcal{A}$ transform. The $\mathcal{A}$ transform is generally invertible and local 3D information of objects into any part's model frame can be reconstructed. Further, this matrix captures the epipolar constraint between two cameras which is exploited to reduce search during tracking.

Initial model-data correspondences are also used to capture feature templates from the image around current data locations. In this section we revisit the formulation of the $\mathcal{A}$ matrix(see [13]) which is a weak perspective affine approximation of the transformation between the object or world coordinates and the camera-pair. Then we present the algorithm for tracking templates in the image plane, that compensates 2D rigid motion.

3.1 Weak Perspective Affine Transform: $\mathcal{A}$ Matrix

Under the assumption of weak perspective, the projection of a world point in homogeneous coordinates is an orthographic projection to a plane at constant depth from the image plane and parallel to it, times a constant scale factor. This transformation is expressed as:

$$p_i = \begin{pmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \end{pmatrix} \cdot p_w = \mathcal{P} \cdot p_w$$

Points away from the depth k (at which $\mathcal{A}$ is constructed) experience a projection error that is proportional to the displacement from k. The *weak perspective* projection is a linear approximation to full perspective projection. This is a reasonable assumption for objects whose extent is small compared to their depth from the camera. For two cameras roughly parallel to each other viewing a common scene of interest, the weak perspective transform can be extended to represent a combined projection of p_w to each camera as:

$$(p_1 \quad p_2)^T = \begin{pmatrix} \mathcal{P}_1 \\ \mathcal{P}_2 \end{pmatrix} \cdot p_w = \mathcal{A} \cdot p_w$$

$\mathcal{A}$ is 4×4 affine and transforms arbitrary model coordinates into image projections. This matrix can be estimated from a minimum of 4 non-coplanar observations. Further, this matrix is generally non-singular (unless the cameras overlap each other) and is therefore invertible. Thus $p_w = \mathcal{A}^{-1} \cdot (p_1 \quad p_2)^T$. Errors in reconstruction due to incorrect image coordinates linearly scale along the perceived object coordinate frame [13]. The matrix $\mathcal{A}^{-1}$ also encodes the epipolar constraint. Since p_w is of the form $(x\ y\ z\ 1)^T$ the last row of $\mathcal{A}^{-1}$ represents the epipolar constraint, i.e.;

$$\begin{pmatrix} a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix} \cdot (p_1 \ p_2)^T = 1$$

This constraint is imposed during tracking in that tracked locations in the left camera are used to limit the search in the right camera to the epipolar line. In the absence of more observers, the choice of camera to impose the epipolar constraint is arbitrary.

Theoretically, assumptions stated in this section must be satisfied strongly to obtain accurate reconstructions. In practice, significant deviations from the assumptions still yield good results around the center of the coordinate frame used to compute $\mathcal{A}$. Thus, $\mathcal{A}$ is a local approximation that needs to be constantly updated. This observation in fact is exploited in the peg-in-hole insertion example described in the next section. As the peg gets closer to the hole, the error estimates between the peg and hole poses (expressed in the peg frame) get more accurate and under a scheduled gain, good servoing results can be achieved.

3.2 Feature Tracking

The tracking component localizes a set of feature templates in a newly acquired image within search windows around hypothesized 2D feature locations. Feature location hypotheses can be obtained in many ways. Coarse motion estimates when available (either from the arm kinematics or coarse motions on a mobile platform or from an assumption of an underlying motion model), can be used to project predicted object feature locations in the next image. If no such estimates are available, current locations can be used as hypotheses in the next image. The search window size includes the maximum displacement between the feature location hypothesis to it's actual location, and varies similarly. Corresponding to the case when there are motion estimates the search window size limits the permissible estimation error in motion. In the latter case, this size limits the amount of image motion that can occur from frame to frame.

The basic algorithm for matching templates to image patches is a combination of normalized cross-correlation and steerable filters. The normalized cross-correlation measure [7] assumes that the surfaces in the environment are Lambertian, that they can be locally approximated by a plane, and that the illumination incident on the surfaces can be locally approximated by a constant. Under these assumptions the correlation measure is normalized in that it is independent of the illumination incident on the surface. However, good experimental results have been obtained with this measure on surfaces that are only weakly Lambertian (see [7] for a derivation and [8] for experiments with this measure).

However, normalized cross-correlation degrades when there is a relative rotation between the templates and image patches [19]. To compensate for 2D rotations it is sufficient to note that linear shift invariance in polar space is equivalent to rotational invariance in Cartesian space and we formulate an equivalent correlation expression in polar space.

A feature template is defined as a pair $\langle \tau, \theta_t \rangle$ where τ is an image patch centered over a dominant image edge and $\theta_t \epsilon [-\pi, \pi]$ is the phase of the maximum response of a steered Gaussian derivative filter [9] with the edge at the patch center. Templates are then localized within a search window γ in a new image as follows:

1. Spatial gradients and their orientations are computed by filtering γ with steerable Gaussian derivative filters and suppressing non-maximal edges within the search window.

2. Each local maximal edge location (i, j) in γ is a potential candidate for the new location of the template, and normalized correlation in polar space is used to identify the best match.

The advantage of using steerable filters is that they can be represented as a set of basis filters from which an arbitrary orientation of a template can be estimated [9]. For edges, first derivatives of Gaussian masks can be used. Their performance is better than that of box filters, for example, when there are non-step edges. The correlation in polar space can be performed efficiently (no space allocation) by simply re-sampling the template in the sampling order originating at θ_t and a match candidate location at θ_i(the orientation response at response at (i, j)). While polar correlation compensates for any changes in orientation of the feature there is, however, an issue of sampling and interpolation accuracy when going from Cartesian coordinates of the image to the polar coordinates under which normalized correlation is performed. Accuracy is traded for speed to a certain degree in the real-time applications investigated, and sampling is performed without interpolation . Some other properties in addition to 2D rotation invariance up to discretization are described in [19].

4 Example: Peg-in-hole Insertion

A schematic of our platform for visual servoing is included in Figure 1 and the task is to insert a square peg into a square hole. The manipulator has four degrees of freedom, representing 3D Cartesian translations and rotation about the vertical.

Three controllers namely, $C0$, $C1$ and $C2$ are constructed. $C0$ is designed to achieve and maintain a reference offset from the assembly to guarantee that no collision will occur while $C1$ orients and aligns the peg to the hole. Finally, $C2$ visually servos the aligned peg into the hole. The Cartesian error signals that drive the controllers are subspaces of 4D pose errors between the peg and the hole i.e. $(\delta x, \delta y, \delta z, \delta\theta)$. These errors are obtained from peg and hole centroid locations along with an estimate of the hole surface normal. Controller $C0$ is driven by $(0, 0, z_d - \delta z, 0)$ [2],

[2] z_d is the desired height at which alignment must be performed -currently a user defined parameter.

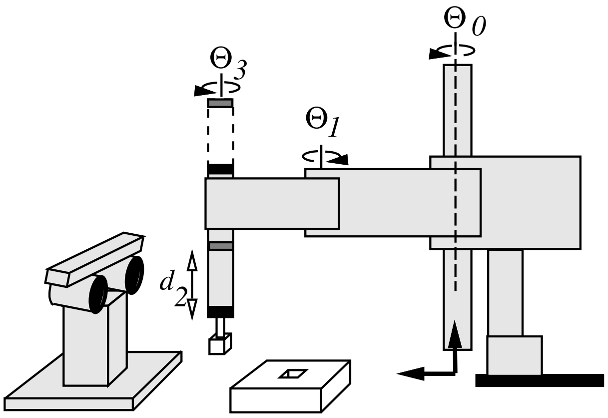

Figure 1: Schematic of the Visual Servoing Platform

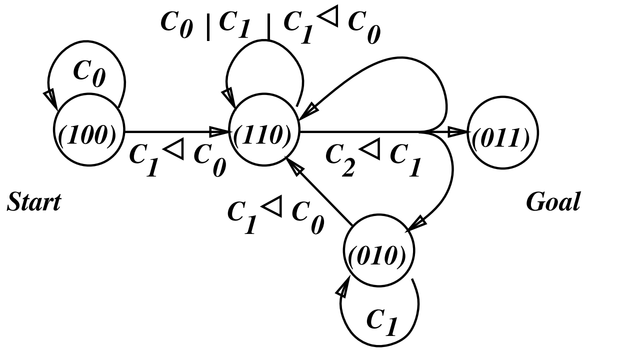

Figure 2: Safe, device independent strategy for peg-in-hole.

$C1$ by $(\delta x, \delta y, 0, \delta\theta)$ and $C2$ by$(0, 0, \delta z, 0)$. Updated 3D peg and hole centroids are computed from local re-constructions of features (model points) on the peg and hole.

The strategy to perform the insertion is captured in the finite state machine described in figure 2. States are encoded as a vector of predicates asserted by controllers. The vector (p_0, p_1, p_2) reflect the status of controllers $C0, C1$ and $C2$ respectively. The assertion p_i =TRUE(1) states that C_i is within some task specific neighborhood of equilibrium. Control compositions $C_i \triangleleft C_j$ (read C_i subject to constraints on C_j) assert p_j and execute C_i, but only up to constraints imposed by p_j. In this manner, we can guarantee that C_i will not undo states achieved in C_j. Moreover, the FSM supervisor can incorporate constraints such as: *the peg cannot be inserted until it has been aligned*$(p_0 \vee p_1)$. The resulting strategy for this case is $\{(C_1 \triangleleft C_0), (C_2 \triangleleft C_1)\}$ or *align at the reference offset, followed by insert while remaining aligned.* The final goal state(011) is achieved when the peg is aligned and pushed to the critical height (the point after which features cannot be seen).

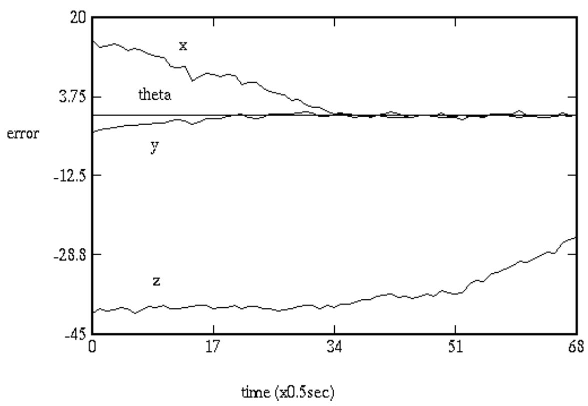

Figure 4: The $x, y, z,$ *and* θ errors vs. time.

Figure 3 presents a stereo snapshots of an insertion run. Initial correspondences between the peg model and the image pair are specified. The image features on the peg are represented by isolated white filled rectangles on the corners of the peg. Hole features are specified in the image. Given the peg model-data correspondences, $\mathcal{A}$ and $\mathcal{A}^{-1}$ are computed. Model points of tracked peg features are reconstructed and the peg centroid is computed. Note that the hole is assumed stationary in this example so its features are not tracked. The user marked hole feature locations are always used to compute the corresponding 3D points in the peg frame using $\mathcal{A}^{-1}$. The hole centroid is inferred and the projections of the centroids to either camera is shown in the image sequence as filled patches within rectangles. The 4D error vector is computed using the peg and hole centroids and the vertical estimated from tracked peg features.

Figure 4 reports the time history of geometric errors estimated using forward kinematics. The insertion stops when the peg is just inserted into the hole.

5 Conclusions and Future Work

The task execution level of a guided mechanical assembly operation is a tight interaction between temporally registered data and controllers. This interaction is supervised by an FSM that by construction captures global task constraints. A camera pair is used to temporally register objects over 3D motion. Temporal registration is also central to scene annotation and general manipulation. While the registration example described in this paper employed a tracker that runs at $6Hz$ for 8 points, we have recently achieved frame rate tracking that has not yet been integrated into the visual servoing system.

There are several extensions that are being considered to the existing work. Ongoing work involves inte-

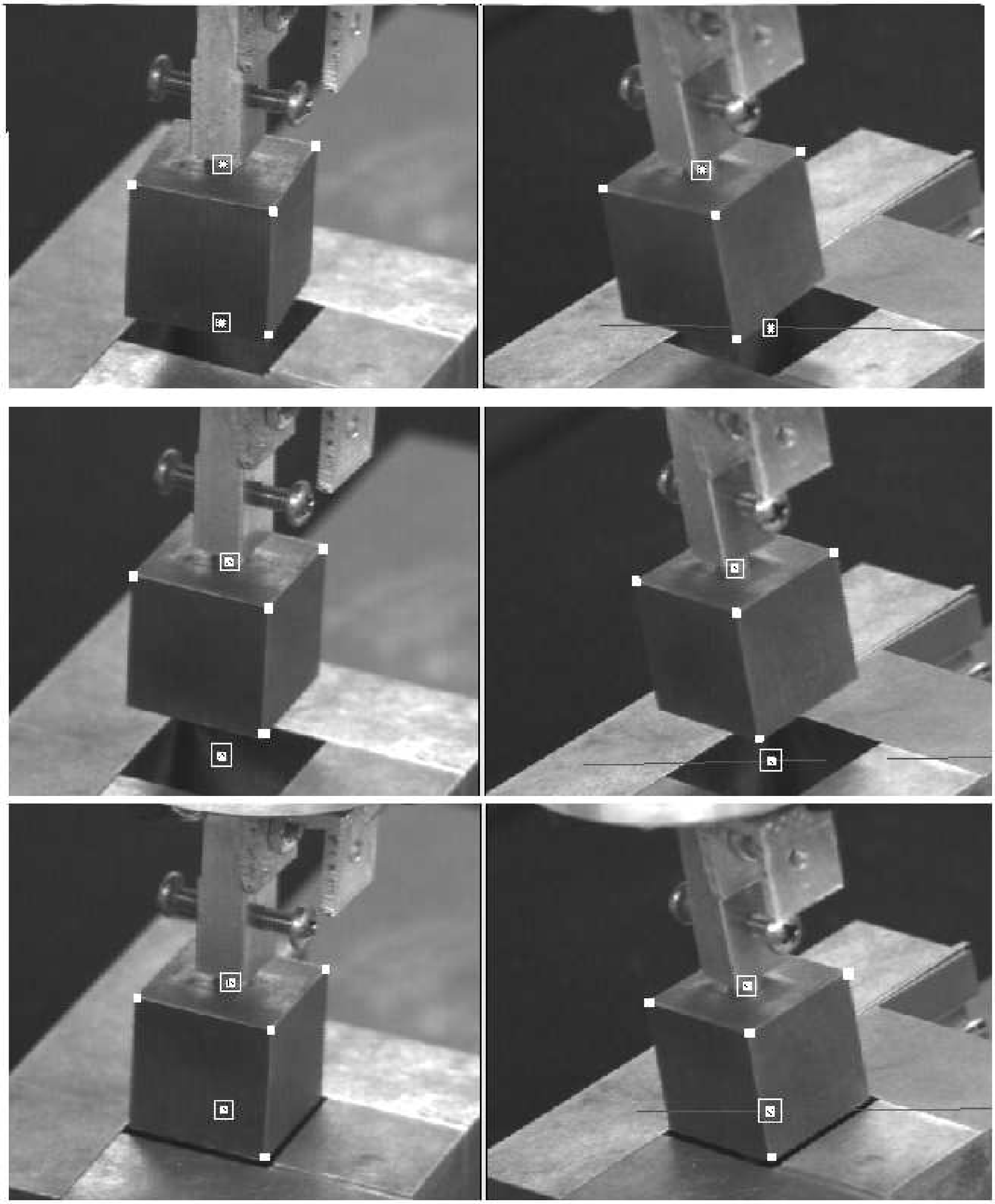

Figure 3: A sequence of stereo image pairs illustrating the features used to guide the insertion.

gration of the assembly planner, a part identifier and an FSM generator with the task execution system described in this paper. An issue not considered in this application is the traversal of distinct aspects as the object moves across a camera. It is shown elsewhere that with *a priori* aspect tables registration across distinct aspects is possible in real-time.

Acknowledgment

The authors wish to thank Manfred Huber at the Laboratory for Perceptual Robotics for designing the Finite State Supervisor. Thanks are also due to Kamal Souccar for hardware and software support.

References

[1] Anandan, P., "A Computational Framework and an Algorithm for the Measurement of Visual Motion", *Int. J. Comput. Vision*, 2:283-310, 1989.

[2] Beveridge, J. R., Weiss, R. S. and Riseman E. M., "Combinatorial optimization applied to 2d model matching subject to fixed and variable scale transformations" *Proc. IEEE Int. Conf. Patt. Recog.*, pp. 18–23, June, 1990,

[3] Beveridge, J. R. and Riseman, E. "Optimal Geometric Model Matching under Full 3D Perspective", *Proc. 2nd CAD-Based Vision Workshop* pp. 54-63, 1994, Pittsburgh, PA

[4] Crowley, J. L. and Stelmaszyk, P., "Measurement and integration of 3-D structures by tracking edge lines", *Proc. European Conf. on Comput. Vision*, pp. 269-280, 1990.

[5] Dakin, G., "Fine-Motion Planning for Robotic Assembly in Local Contact Space", *Ph.D. Dissertation*, University of Massachusetts at Amherst, 1994.

[6] Dickinson, S. J., Jasiobedzki, P., Olofsson, G. and Christensen Henrik I., " Qualitative Tracking of 3-D Objects using Active Contour Networks", *Proc. of Comput. Vision and Patt. Recognition*, pp.812-817, June 1994, Seattle, Washington

[7] Fennema, C. L., "Interweaving Reason, Action and Perception", *COINS TR91-56*, Dept. of Computer Science, Univ. of Massachusetts, Amherst, 1991.

[8] Fennema, C. L., "Finding Landmark Features Under a Broad Range of Lighting Conditions", *Proc. SPIE Intelligent Robots and Computer Vision X11: Algorithms and Techniques*, 2055:181-191, Boston, MA, Sept. 1993.

[9] Freeman, W. T. and Adelson, E. H., "The Design and Use of Steerable Filters", *IEEE Trans. Patt. Anal. Machine Intell.*, 13(9):891-906, Sept., 1991.

[10] Gennery, D., "Tracking known Three-Dimensional objects", *Int. J. of Comput. Vision*, 7(3):243-270, 1992.

[11] Grupen, R., Huber, M., Coelho, J,. and Souccar, K. "Distributed Control Representation for Manipulation Tasks," *IEEE Expert, Special Issue on Intelligent Robotic Systems, Ed. R. Arkin*, 10(2):9-14, 1994.

[12] Hager, G. D., "Real-Time feature tracking and projective invariance as a basis for hand-eye coordination.", *Proc. Comput. Vision Patt. Recognition*, pp. 533-539, 1994.

[13] Hollinghurst, N. J., and Cipolla, R., "Uncalibrated Stereo Hand-Eye Coordination", *CUED/F-INFENG/TR 126*, Department of Engineering, University of Cambridge, Cambridge, England CB2 1PZ, 1993.

[14] Kass., M., Witkin, A. and Terzopolous, D, "Snakes: Active contour models", *Int. J. Comput. Vision*, 1(4):321-331, 1988.

[15] Liu, Y. and Popplestone, R. J., "Planning for Assembly from Solid Models", *Proceedings of the IEEE Conference on Robotics and Automation*, May, 1989, Scottsdale, AZ.

[16] Liu, Y. and Popplestone, R. J., "Symmetry Groups in Analysis of Assembly Kinematics", *Robotics and Automat. 91*, 572–577, April 1991.

[17] Lowe, D. G., "Robust Model-based Motion Tracking Through the Integration of Search and Estimation",

[18] Nelson, B., Papanikolopoulos, N. P., and Khosla, P. K., "Visual Servoing for Robotic Assembly", *in "Visual Servoing", Hashimoto, K., (ed.), World Scientific*, 1994.

[19] Ravela, S., Draper, B., Lim, J. and Weiss, R., "Adaptive Tracking and Model Registration Across Distinct Aspect", *Technical Report, CMPSCI TR95-21*, Department of Computer Science, University of Massachusetts, Amherst

[20] Sawhney H. and Hanson A., "Tracking, Detection and 3D representation of potential obstacles using affine constraints", *Proc. Img. Understanding Wkshp.*, pp. 1009-1017, San Diego California, January 1992.

[21] Shi, J. and Tomasi, C., "Good features to track", *Proc. Comput. Vision Patt. Recognition*, pp. 593-600, 1994.

[22] Uenohara, M., and Kanade, T., "Vision-Based Object Registration for Real-Time Image Overlay", *(In Press) Jrnl. Comput. Bio. and Med.*

[23] Verghese, G., Gale, K. L., and Dyer, C. R.,"Real-time motion tracking of three dimensional objects", *Proc. IEEE conf. Robotics and Automation*, pp. 1998-2003, 1990.

[24] Walters, W. J.,"Visual Servo Control of Robots using Kalman Filter Estimates of Pose Relative To Work-pieces", *in "Visual Servoing", Hashimoto, K., (ed.), World Scientific*, 1994.