Real-time tracking of complex structures with on-line camera calibration

T. Drummond and R. Cipolla
{twd20|cipolla}@eng.cam.ac.uk

Department of Engineering
University of Cambridge
Cambridge CB2 1PZ, UK

Abstract

This paper presents a novel three-dimensional model-based tracking system which has been incorporated into a visual servoing system. The tracking system combines modern graphical rendering technology with constrained active contour tracking techniques to create wire-frame-snakes. It operates in real time at video frame rate (25 Hz) and is based on an internal CAD model of the object to be tracked. This model is rendered using a binary space partition tree to perform hidden line removal and the visible features are identified on-line at each frame and are tracked in the video feed. The tracking system has been extended to incorporate real-time on-line calibration and tracking of internal camera parameters. Results from on-line calibration and visual servoing experiments are presented.

1 Introduction

The tracking of rigid three-dimensional objects is useful for numerous applications, including motion analysis, surveillance and robotic control tasks.

This paper tackles two problems. The first is the accurate tracking of a known three-dimensional object in the field of view of a camera. The output of this tracker is a continuously updated estimate of the pose of the object being viewed. The second is the tracking of the internal camera parameters themselves. The resulting tracking system has been used to close the loop in a robot control system to guide a robotic arm to a previously taught target location relative to a workpiece.

This work has been motivated by tasks such as the robotic manufacture of cars and ships. These tasks are characterised by the need to accurately position a tool on the workpiece which may be inaccurately located with respect to the camera. The examples presented in this paper come from the domain of the welding of ship parts.

Because a video signal contains a very large amount of data, it is important to extract only a small amount of salient information, if real-time frame (or field) rate performance is to be achieved [1]. This observation leads to the notion of *feature based* tracking [2]. Image contours are a particularly powerful feature and have been used by many succesful tracking systems [3, 4, 5, 6].

A similar approach to that presented here has been taken by the RAPiD system [4] which is also based on tracking visible features of a three-dimensional model. The wire-frame-snake system presented here differs two important ways from the RAPiD approach, namely that the visible features are computed on-line (rather than pre-computed for what is essentially a viewsphere) and that the edge features to be tracked are dynamically reweighted in real-time based on their saliency. The first aspect permits the tracking of complex three-dimensional structures, while the second improves robustness and precision of the tracker.

Another important approach to tracking is the CONDENSATION algorithm [5] which obtains extremely high robustness by tracking a discretely sampled probability distribution of hypotheses from frame to frame. A significant difference in this work lies in the trade-off between robustness and precision. Whereas CONDENSATION typically takes a large number of samples with a comparatively small number of edge measurements per sample, the wire-frame-snake tracking system uses a large number of measurements in order to obtain high precision. Typically, 400 edge measurements are made per image frame.

The use of visual feedback ouput by such tracking systems for robotic control is increasingly becoming an attractive proposition. A distinction is often made [7] between *image-based* [8] and *position-based* [9] visual servoing. The approach presented here projects the action of three-dimensional motion into the image where it is fitted to image measurements.

2 Theoretical framework

The approach proposed here for tracking a known 3-dimensional structure is based upon maintaining an estimate of the camera projection matrix, P, in the co-ordinate system of the structure. This projection matrix is represented as the product of a matrix of internal camera parameters:

$$K = \begin{bmatrix} f_u & s & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

and a Euclidean projection matrix representing the position and orientation of the camera relative to the target structure:

$$E = \begin{bmatrix} R & t \end{bmatrix} \qquad \text{with } RR^T = I \text{ and } |R| = 1 \tag{2}$$

The projective co-ordinates of an image feature are then given by

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = P \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{3}$$

with the acutal image co-ordinates given by

$$\begin{pmatrix} \tilde{u} \\ \tilde{v} \end{pmatrix} = \begin{pmatrix} u/w \\ v/w \end{pmatrix} \tag{4}$$

Rigid motions of the camera relative to the target structure between consecutive video frames can then be represented by right multiplication of the projection matrix by a Euclidean transformation of the form:

$$M = \begin{bmatrix} R & t \\ {\scriptstyle 0\,0\,0} & {\scriptstyle 1} \end{bmatrix} \tag{5}$$

These M, form a 4×4 matrix representation of the group SE(3), which is a 6-dimensional Lie Group. The generators of this group are typically taken to be translations in the x, y and z directions and rotations about the x, y and z axes, represented by the following matrices:

$$G_1 = \begin{bmatrix} 0&0&0&1 \\ 0&0&0&0 \\ 0&0&0&0 \\ 0&0&0&0 \end{bmatrix}, G_2 = \begin{bmatrix} 0&0&0&0 \\ 0&0&0&1 \\ 0&0&0&0 \\ 0&0&0&0 \end{bmatrix}, G_3 = \begin{bmatrix} 0&0&0&0 \\ 0&0&0&0 \\ 0&0&0&1 \\ 0&0&0&0 \end{bmatrix}, \tag{6}$$
$$G_4 = \begin{bmatrix} 0&0&0&0 \\ 0&0&1&0 \\ 0&-1&0&0 \\ 0&0&0&0 \end{bmatrix}, G_5 = \begin{bmatrix} 0&0&-1&0 \\ 0&0&0&0 \\ 1&0&0&0 \\ 0&0&0&0 \end{bmatrix}, G_6 = \begin{bmatrix} 0&1&0&0 \\ -1&0&0&0 \\ 0&0&0&0 \\ 0&0&0&0 \end{bmatrix}$$

These generators form a basis for the vector space (the Lie algebra) of derivatives of SE(3) at the identity. Consequently, the partial derivative of projective image co-ordinates under the ith generating motion can be computed as:

$$\begin{pmatrix} u' \\ v' \\ w' \end{pmatrix} = PG_i \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{7}$$

with

$$L_i = \begin{pmatrix} \tilde{u}' \\ \tilde{v}' \end{pmatrix} = \begin{pmatrix} \frac{u'}{w} + \frac{uw'}{w^2} \\ \frac{v'}{w} + \frac{vw'}{w^2} \end{pmatrix} \tag{8}$$

giving the motion in true image co-ordinates. A least squares approach can then be used to fit the observed motion of image features between adjacent frames. This process is detailed in Section 3.3.

In a similar manner, the motion of features in the image due to the change of camera parameters can be computed and these motion fields incorporated into the least squares solution. This extension is discussed in Section 4.

2.1 Tracking edges

An important aspect of the approach presented here is the decision to track the edges of the model (which appear as intensity discontinuities in the video feed). Edges are strong features that can be reliably found in the image because they have a significant spatial extent. Furthermore, this means that a number of measurements can be made along each edge, and thus they may be accurately localised within an image.

This approach also takes advantage of the aperture problem (that the component of motion of an edge, tangent to itself, is not observable locally). This problem actually yields an enormous benefit since the search for intensity discontinuities in the video image can be limited to a one dimensional path that lies along the edge normal, $\hat{n}$ (see Figure 1) and thus has linear complexity in the search range, rather than quadratic. This benefit is what makes it possible to track complex structures in real time on a standard workstation without additional hardware. The normal component of the motion fields, L_i are then also computed (as $L_i \cdot \hat{n}$).

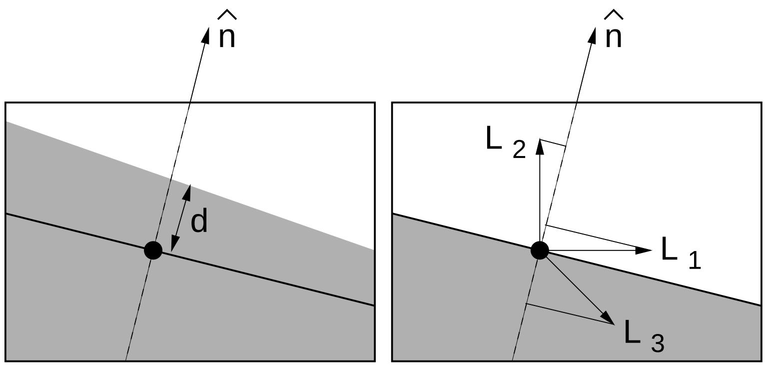

Figure 1: Computing the normal component of the motion

3 Tracking system

The three-dimensional tracking system makes use of constrained snake technology [6] to the follow edges of the workpiece that are visible in the video image. One novel aspect of this work is the use of a real-time hidden-line-removal rendering system (using binary space partition trees [10]) to dynamically determine the visible features of the model in real-time. This technique allows accuarate frame rate tracking of complex structures such as the ship part shown in Figure 2.

Figure 3 shows system operation. At each cycle, the system renders the expected view of the object (a) using its current estimate of the projection matrix, P. The visible edges are identified and tracking nodes are assigned at regular intervals in image co-ordinates along these edges (b). The edge normal is then searched in the video feed for a nearby edge (c). Typically $m \approx 400$ nodes are assigned and measurements made in this way. The system then projects this m-dimensional measurement vector onto the 6-dimensional subspace corresponding to Euclidean transformations (d) using the least squares approach described in Section 3.3 to give the motion, M. The Euclidean part of the projection matrix, E is then updated by right multiplication with this transformation (e). Finally, the new projection matrix P is obtained by multiplying the camera parameters K with the updated Euclidean matrix to give a new current estimate of the local position (f). The system then loops back to step (a).

3.1 Rendering the model

In order to accurately render a CAD model of a complex structure such as the one shown in Figure 2 at frame rate, an advanced rendering technique such as the use of binary space partition trees is needed [10]. This approach represents the object as a tree, in which each node contains the equation of a plane in the model, together with a list of edges and convex polygons in that plane. Each plane partitions 3-dimensional space into the plane and the two open regions either side of the plane. The two branches of the tree represent those parts of the model that fall into these two volumes. Thus the tree recursively partitions space into small

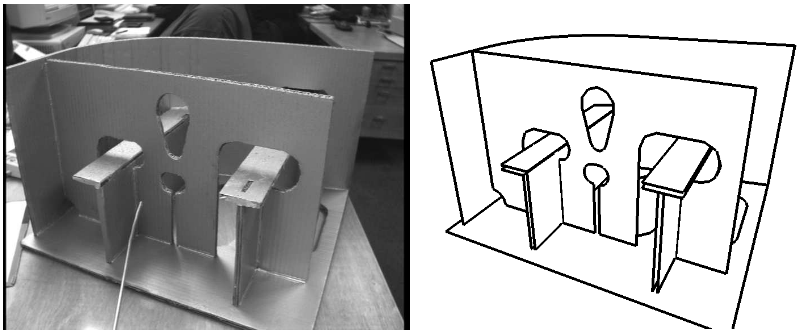

Figure 2: Image and CAD model of ship part

regions which, in the limit, contain no remaining model features. The rendering takes place by performing an *in-order* scan of the tree, where at each node, the viewpoint is tested to see if it lies in front, or behind the plane. When this is determined, those features lying closer to the camera are rendered first, then the plane itself, and finally, the more distant features. The use of a stencil buffer prevents over-writing of nearer features by more distant ones and also provides a layer map when the rendering is complete. The ship part contains 12 planes, but since 8 of these (corresponding to the T and L beams) are split into two parts by a vertical plane partition, there are 20 nodes in the tree.

3.2 Locating edges

Once rendering is complete, the layer map is used to locate the visible parts of each edge by comparing the assigned layer of the plane for each edge in the model with the layer in the stencil buffer at a series of points along that edge. Where the depths agree, trackers are assigned to search for the nearest edge in the video feed along the edge normal (see Figure 4).

The result of this process is a set of trackers with known position in the model co-ordinate system, with computed edge normals and the distance along those

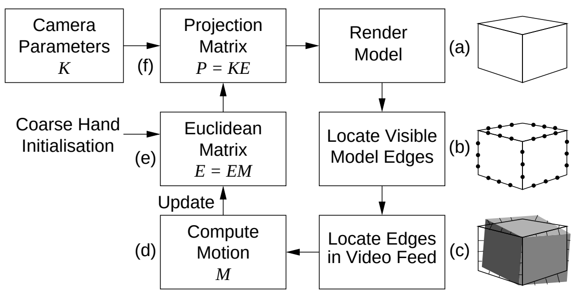

Figure 3: Tracking system operation

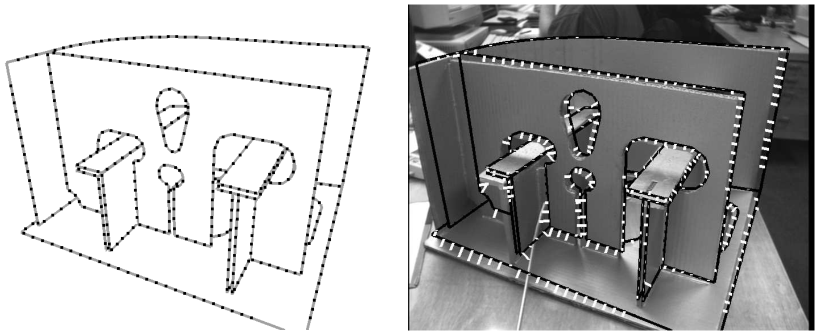

Figure 4: Tracking nodes assigned and distances measured

normals to the nearest image edge. Grouping these distances together provides an m-dimensional measurement vector.

3.3 Computing the motion

Step (d) in the process involves the projection of the measurement vector onto the subspace defined by the Euclidean transformation group. The action of each of the generators of SE(3) on the tracking nodes in image co-ordinates can be found by computing PG_i and applying this to the homogeneous co-ordinates of the node in 3-space. This can be projected to give a vector, L_i^ξ describing the image motion of the ξth node for the ith generator of Euclidean motion of the object. $L_i^\xi \cdot \hat{n}^\xi$ then describes the magnitude of the edge normal motion that would be observed in the image at each node for each group generator. These can be considered as a set of m-dimensional vectors which describe the motion in the image for each mode of Euclidean transformation. The system then projects the m-vector corresponding to the measured distances to the observed edges onto the subspace spanned by the transformation vectors. This provides a solution to finding the geometric transformation of the part which best fits the observed edge positions, minimising the square error between the transformed edge position and the actual edge position (in pixels). This process is performed as follows:

$$O_i = \sum_\xi d^\xi (L_i^\xi \cdot \hat{n}^\xi) \tag{9}$$

$$C_{ij} = \sum_\xi (L\xi_i \cdot \hat{n}^\xi)(L_j^\xi \cdot \hat{n}^\xi) \tag{10}$$

$$\alpha_i = C_{ij}^{-1} O_j \tag{11}$$

(with Einstein summation convention over latin indices). The α_i then contain the quantity of each mode of Euclidean motion that has been observed. The final step is to compute the actual motion of the model and apply it to the matrix E in (2). This is done by using the linear approximation to the exponential map connecting α_i with SE(3).

$$E_{t+1} = \text{Euclideanise}(E_t(I + \textstyle\sum_i \alpha_i G_i)) \tag{12}$$

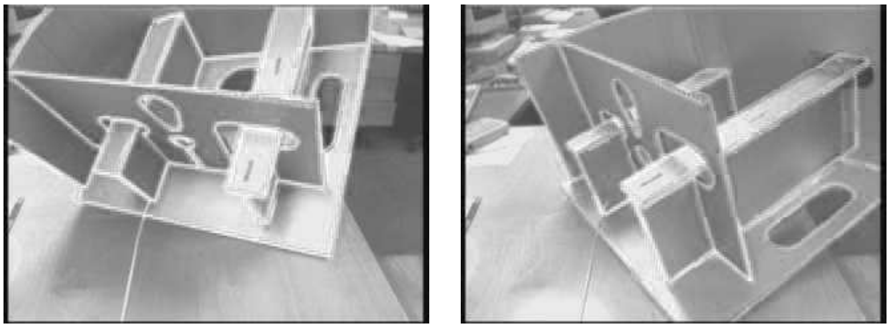

Figure 5: Frames from tracking sequence

Because the motion is typically small, this approximation is accurate, although the matrix E must be coerced back into true Euclidean form after being updated.

This section has described the basic version of the tracking system. This system performs well over a wide range of configurations (see Figure 5) and is extremely robust to occlusion (see Figure 6). One key advantage of this approach is that it is possible to extend it to include more complex situations such as tracking camera parameters in addition to object motion.

4 On-line camera calibration

The internal characteristics of a pinhole camera can be described with five parameters. These are the focal length, aspect ratio, u and v co-ordinates of the principal point and the skew [11]. The matrix of camera parameters is shown in (1). In practice, the skew of the camera is known to be zero and here we enforce that condition. Thus there are just four parameters to be modelled.

For each of these parameters, there is an associated vector field, just as for motion in space. The vector fields corresponding to the camera parameters can be easily described in terms of the $\binom{u}{v}$ co-ordinates in the image plane. This creates

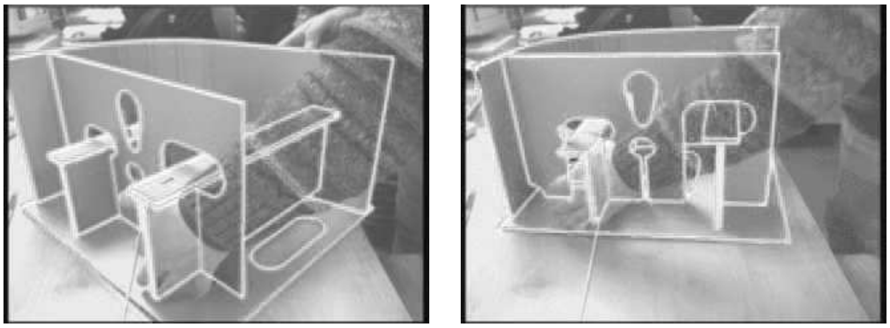

Figure 6: Frames from tracking sequence with occlusion

four new vector fields, $L_i, 7 < i < 10$. These are added to the vector fields already used for tracking in the system which then fits a least squares solution in ten dimensions instead of six. The resulting system is then able to dynamically track the camera parameters in addition to the motion of the target and is able, for example, to distinguish between motion towards the target and a zoom.

4.1 Results

Some experiments to calculate the accuracy of camera calibration calculated in this way were performed.

4.1.1 Stability with respect to image noise

Firstly, the configuration of the camera and a calibration grid were kept fixed and the calibration calculated on a series of runs in order to assess the impact of image noise on the calibration. An 8.5 mm lens was used for this experiment. The mean and standard deviation of the focal length ($f = \sqrt{f_u f_v}$), aspect ratio($a = f_u/f_v$) and principal point (u_0, v_0) over these runs were computed. The results were $f = 785.55 \pm 0.05$ pixels, $a = 0.950055 \pm 0.00004$ and $(u_0, v_0) = (336.4 \pm 0.2, 278.8 \pm 0.5)$ pixels. In all cases the standard deviation was $O(10^{-4})$ times the characteristic scale (for u_0 and v_0 this is the focal length).

4.1.2 Stability with respect to configuration

Secondly, a series of runs were performed in which the configuration was varied in order to provide an estimate of the accuracy of the calibration measurements. A 16mm lens was used for this experiment, for which the results were $f = 1442.2 \pm 7.1$ pixels, $a = 0.94995 \pm 0.00067$ pixels and $(u_0, v_0) = (355 \pm 10, 282 \pm 24)$ pixels. The standard deviation values obtained in this experiment were all less than 1% of the characteristic scale (with the exception of v_0 which was slightly larger). This compares well with standard calibration techniques [11] which was tested with images captured from the sequences and also generated errors of O(1%).

5 Visual servoing system

The visual servoing system takes the Euclidean matrix, E as output from the tracking system and uses this within a non-linear control law to provide feedback to servo the robot to a stored target position. These are learned by acquiring the Euclidean matrix with the robot placed in the target position by the supervisor. The inverse of this target matrix, E_t^{-1}, is easily computed and the product of this with the current position matrix yields the transformation from the target position to the current position.

$$T = EE_t^{-1} \tag{13}$$

The translation and rotation vectors that must be applied to the robot are then easily extracted from this representation. (here $i, j, k = 1,2,3$):

$$t_i = T_{i4} \tag{14}$$

$$r'_i = \frac{1}{2} \sum_{jk} \epsilon_{ijk} T_{jk}$$

$$r_i = \frac{r'_i \sin^{-1}(|r'|)}{|r'|} \tag{15}$$

The vectors t and r are then multiplied by a gain factor and sent to the robot as end effector translantion and rotation velocities. The gain is dependent on the magnitudes of t and r so that small velocities are damped to obtain higher precision, while large errors in position may be responded to quickly. A maximum velocity clamp is also applied for safety reasons and to prevent possible instabilities due to latency.

5.1 Results

The tracking system and visual servoing system have been tested in a number of experiments to assess their performance both quantitatively and qualitatively. These experiments were conducted with an SGI O2 workstation (225 MHz) controlling a Mitsubishi RV-E2 robot.

5.1.1 Stability of the tracker with respect to image noise

The stability of the tracker with a stationary structure was measured to assess the effect of image noise on the tracker. The standard deviation of position and rotation as measured from the Euclidean matrix were measured over a run of 100 frames. From a viewing distance of 30cm, the apparant r.m.s. translational motion was found to be 0.03mm with the r.m.s. rotation being 0.015 degrees.

5.1.2 Accuracy of positioning

The accuracy of positioning the robot was measured with two experiments. Firstly, the ship part was held fixed and the robot asked to home to a given position from a number of different starting points. When the robot had ceased to move, the program was terminated and the robots position queried. The standard deviation of these positions was computed and the r.m.s. translational motion was 0.08mm with the r.m.s. rotation being 0.06 degrees.

The second accuracy experiment was performed by positioning the ship part on an accurate turntable. The part was turned through fifteen degrees in one degree rotations and the robot asked to return to the target position each time. Again, the position of the robot was queried and a circle was fitted to the data. The residual error was computed and found to give an r.m.s. positional error of 0.12mm per measurement (allowing for the three degrees of freedom absorbed into fitting the circle).

6 Conclusion

This paper has presented an extensible framework for real-time three-dimensional tracking of complex structures. The system has been implemented and been shown to exhibit sufficient accuracy for many useful tasks, such as robot control. The formulation used is extensible, as has been demonstrated by the incorporation of on-line camera calibration which provides real-time performance and yields accuracy comparable to existing techniques.

Acknowledgements

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

References

[1] C. Harris. Geometry from visual motion. In A. Blake, editor, *Active Vision*, chapter 16, pages 263–284. MIT Press, 1992.

[2] G. Hager, G. Grunwald, and K. Toyama. Feature-based visual servoing and its application to telerobotics. In V. Graefe, editor, *Intelligent Robotic Systems*. Elsevier, 1995.

[3] D. Terzopoulos and R. Szeliski. Tracking with Kalman snakes. In A. Blake, editor, *Active Vision*, chapter 1, pages 3–20. MIT Press, 1992.

[4] C. Harris. Tracking with rigid models. In A. Blake, editor, *Active Vision*, chapter 4, pages 59–73. MIT Press, 1992.

[5] M. Isard and A. Blake. CONDENSATION - conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

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

[7] 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.

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

[9] 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.

[10] M. Paterson and F. Yao. Efficient binary space partitions for hidden surface removal and solid modeling. *Discrete and Computational Geometry*, 5(5):485–503, 1990.

[11] O.D. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, 1993.