FROM AN IMAGE SEQUENCE TO A RECOGNIZED POLYHEDRAL OBJECT

D. W. Murray, D. A. Castelow and B. F. Buxton

GEC Research Ltd,
Hirst Research Centre,
East Lane, Wembley,
Middlesex, HA9 7PP, UK

ABSTRACT

We describe the combination of several novel algorithms into a system which obtains visual motion from a sequence of images and uses it to recover the 3D geometry and 3D motion of polyhedral objects relative to the sensor. The system goes on to use the recovered geometry to recognize the object from a database, a stage which also resolves the depth/speed scaling ambiguity, resulting in absolute depth and motion recovery. We demonstrate its performance of imagery from a well carpentered CSG model and on real imagery from a simple wooden model.

I INTRODUCTION

The primary aim of research into computational vision, and indeed of that into many other automated sensing techniques, is to give machines the power to perceive the three dimensional nature of the environment in which they will be required to take intelligent action. More often than not in the robot's world, any action will involve movement and so the recovery of three dimensional motion at an early stage of the sensory processing is of key importance to robotics.

The 2D visual motion derived from a sequence of time-varying images is one valuable source of information about the 3D scene and its motion relative to the sensor. At the most basic level, visual motion can be used simply to flag scene motion, but it has long been appreciated, certainly since the work of von Helmholtz (1866), that encoded within it is much more detail about the 3D geometric structure and 3D motion of the scene. The capability to exploit this in the human visual system has been demonstrated in a variety of psychophysical experiments over many years (e.g. Wallach and O'Connell 1953, Johansson 1973, Proffitt and Bertenthal 1984) but it is only recently that computing resources have been sufficient to spur the derivation of detailed computational schemes to do the same — that is, solve the structure-from-motion problem (e.g. Ullmann 1979, Fennema and Thompson 1979, Clocksin 1980, Longuet-Higgins and Prazdny 1980, Thompson and Barnard 1981, Longuet-Higgins 1981, Bruss and Horn 1983, Lawton 1983, Hildreth 1984, Tsai and Huang 1984, Longuet-Higgins 1984, Buxton *et al* 1984, Waxman and Ullman 1985, Waxman and Wohn 1985, Harris *et al* 1986, Westphal and Nagel 1986).

As this lengthy, but far from complete, list of references may suggest, *solve* is too sweeping a word. With the extra degrees of freedom introduced by unconstrained relative motion between camera and viewed object, the problem has proved frustratingly difficult, being a combination of two non-trivial tasks. In the first place, obtaining 2D visual motion from a sequence of images is vulnerable to noise and ultimately to the possibility that the visual motion field might not correspond to the analytical or geometric motion field because of lighting and occlusion effects. Secondly, even when given 'geometric' visual motion, structure-from-motion algorithms are typically ill-conditioned with respect to noise.

In this paper we describe the combination of several vision algorithms into a simple system to obtain the visual motion from an image sequence of a moving polyhedral object, to recover the 3D structure and 3D motion of the object and to go on to use the structure to recognize the object from a database. We demonstrate the algorithms using image sequences of a chipped block, obtained both from a CSG model and from a real model.

Part funded by the Alvey IKBS Directorate as IKBS 013 *Spatio-temporal processing and optical flow for computer vision*

The emphasis here is on the functionality of a complete system, examining whether algorithms which appear successful in isolation are sufficiently robust to accept the corrupted data processed at an earlier stage and are able to provide sufficiently reliable output to drive the next stage in the machine vision processing hierarchy.

We chose to restrict processing to a polyhedral world because the simplifying constraints it affords have lead naturally to more highly developed algorithms in the areas of geometrical matching and segmentation, while not excluding much of the man-made world. In retrospect, we see no reason why the approach should not be extended to embrace simple curved primitives.

An overview of the system, is given in Figure 1, where the bold larger boxes embrace the principal conceptual tasks of (a) early processing and obtaining visual motion, (b) segmentation, (c) obtaining structure from motion and (d) model matching. Box (e) is concerned with model compilation. For the first task we present a new method developed from the technique of Scott (1986); for the second we use a novel pot-pourri of standard image-based techniques; for (c) we use a simple least squares iterative technique; and, finally, for task (d) we have used the matching paradigm of Grimson and Lozano-Pérez (1984,1985) in which extensive geometrical modifications have been made to allow matching to 3D edge segments (Murray and Cook 1986). In the following sections we discuss these tasks in more detail and show encouraging results on synthetic and real imagery.

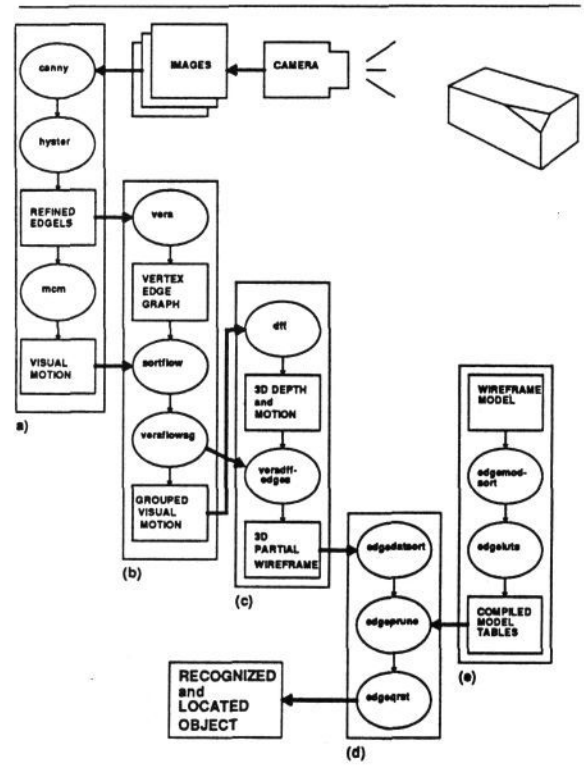


Figure 1. An overview of the system. The larger boxes embrace the major tasks of (a) low-level processing and obtaining visual motion, (b) image and visual motion segmentation (c) structure-from-motion and (d) model matching. Box (e) describes model compilation.

AVC 1987 doi:10.5244/C.1.28

II OBTAINING VISUAL MOTION

It is possible to distinguish two classes of scheme for obtaining visual motion from a sequence of images. At the lower level are the intensity-based methods where local visual motion estimates are derived from local changes in the image irradiance. While these methods typically yield visual motion data at a fairly dense set of points in the image, they supply only partial information about the field. Use of the motion constraint equation (Horn and Schunck 1981) means that information is only obtained about the component of visual motion along the irradiance gradient (the vernier velocity), leading to the well-known aperture problem (Ullmann 1979). At a higher level in the visual processing hierarchy are the token matching or correspondence techniques in which features such as corners are tracked and matched over time. Here, if the matching tokens are highly distinctive, a complete visual motion vector is obtained, but only at a few points in the image. Note that as tokens become less distinctive, ambiguities in matching arise inevitably: a pure gradient scheme can be thought of as matching on quasi-featureless tokens and the most that can be recovered using only local information is the vernier velocity (Ullmann and Hildreth 1983).

Scott has recently introduced a new method of deriving visual motion (Scott 1986) which successfully combines aspects of the two schemes outlined above. He uses a patch of pixel intensities in one frame as the matching token and searches uniformly in a region around that patch in the next frame, computing a matching strength at each position. By determining the principal axes and moments of the matching strength distribution he is able to determine two orthogonal components of the flow together with their associated confidences. If the region is distinctive, say a blob or corner, the strong matches will be tightly grouped and both components of the visual motion are recovered with high confidence. On the other hand, if the intensity patch forms part of an extended edge, the strong matches will have a similarly extended distribution, and only the component of visual motion perpendicular to the edge is returned with high confidence. The crucial point is that the principal axis decomposition *determines* whether the patch of pixels used for matching is a distinctive token or not.

We use Scott's principal axis decomposition method here but, rather than using intensity patches, we match on intensity edgels (elements of intensity edge one pixel long) which are detected using the Canny operator (Canny 1983) and refined using thresholding with hysteresis.

The details of the method are given in the paper by Castelow *et al* (1987) in this conference, and here we show only the pertinent results. Three images are required to compute the flow and Figure 2 shows the central image from the sequence of a chipped block translating vertically upwards in the direction of the y axis (and not rotating). The image sequence was synthesized using a CSG body modeller so that accurate 3D world coordinates could be obtained with which to compare with our computed results which we show later. The block subtends angles to the camera

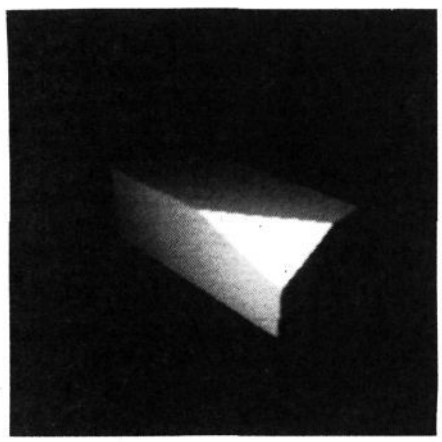


Figure 2. The central frame from a sequence of a moving chipped block. Three frames are used to computed the flow.

of about about 30° vertically and 35° horizontally, and the depth variation from front to back of the block is some 40% of its mean depth. In Figure 3 we show the edgels detected by the Canny operator *after* thresholding with hysteresis, again for the central frame. In Figure 4 is shown the major (higher weight) component of the visual motion computed at the edgel positions. As expected for extended edges, the major components are all nearly perpendicular to their associated edge directions. The magnitude of the motion vectors has been enlarged by a factor of 10 to make them easier to see.

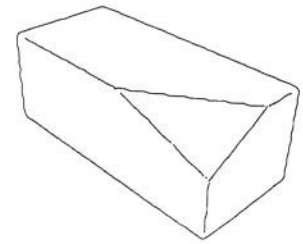


Figure 3.
The edge map for the centre frame after thresholding.

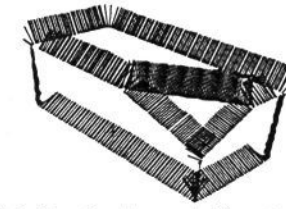


Figure 4. The major components of visual motion. As expected for extended edges, these lie nearly perpendicular to the edge direction.

III SEGMENTATION

If the detailed solutions of the structure-from-motion problem alluded to in the introduction could be extended to several structures in the scene, we would have a relatively low-level scheme for the segmentation of the scene into primitives labelled by geometrical disposition and, perhaps more importantly, motion. This scenario is an attractive one, as visual motion appears to be a powerful cue for segmentation in the human visual system, illustrated by the ease with which we detect an otherwise perfectly camouflaged creature as soon as it *moves* and by simple but compelling experiments with interpenetrating moving random dot patterns.

Attention was given to segmentation using motion in earlier research (Fennema and Thompson 1979, Thompson 1980, Neumann 1980) but there the flow-to-scene reconstructions involved quite severe simplifying assumptions. More recent work has used realistic reconstructions, but appears computationally very expensive, involving in one case the use of Hough transforms (Adiv 1985) and in another global optimization using simulated annealing (Murray and Buxton 1987). Murray and Williams (1986) used a local planar surface mask to detect 3D motion and orientation boundaries, a method which is parallel (Buxton and Williams 1986), but it requires a dense flow field to function

Perhaps in the light of these persistent difficulties, there has been a return to using simpler properties of the visual motion (Thompson and Pong 1987, Spoerri and Ullman 1987). It appears expedient too to assist any segmentation from motion by using simpler 2D image information as much as possible to presort the visual motion into connected groups. This task is not too onerous for images of a polyhedral world.

The segmentation builds up a 2D vertex-edge graph using a recipe which borrows much from techniques described in standard texts such as Ballard and Brown (1982). The entities in the graph obviously include vertices and edges, but also include caps (which 'cover' any unattached terminators of edges, making subseuqent traveral of the graph more uniform), occlusions, and regions. Regions are not invoked in this work but we comment on occlusions later.

The starting point for the construction of the vertex-edge graph is the set of edgels in the centre frame which were used for flow matching. Using their position and orientation they are linked into extended strings as follows:

a) Starting at the strongest unvisited edgel i, links are made to edgels on the left until a break occurs. The leftmost edgel becomes the leftmost in a new string. (Left and right are defined as if looking from the dark to the light side of an edge.)

b)Links are made to the right from i until a break occurs. The rightmost edgel becomes the rightmost in the string.

c)If there are unvisited edgels, continue from (a).

If the linking is proceeding from an edgel i, the orientation of i is classified into one octant bin corresponding to one of the eight pixel neighbours. Suppose that the pixel neighbour pointed to directly by i is labelled C and the two on the dark and light sides of C labelled D and L, respectively. If there is an edgel j in C then to make a link the orientation difference between i and j must be less than some threshold angle. If the threshold is exceeded no link can be made and i becomes the leftmost or rightmost string terminator. However, if there is no edgel in C a search is made in D and L. A potential link exists if the required edgel exists and the orientation difference is less than the threshold angle. If there is only one potential link this is chosen but if two exist the one with the smaller orientation difference is used. Obviously if no potential link exists, i becomes a terminator.

The next step is to find points of high curvature along the *strings*. The local edge orientation as a function of length along the string is convolved with the derivative of a Gaussian (Asada and Brady 1984) and extrema in this signal which exceed some threshold are marked as kinks. No attempt is made to classify the kink shape: the string is simply broken and a putative vertex inserted and links made on both sides to the string ends.

The shape of each smooth string section is determined and, at present, only those edges found to be straight are considered further. If a straight edge has a free terminator, a search is made for a nearby vertex or terminator to which to link. The heuristic search favours large growth forwards but does not exclude smaller sideways and backwards growth. The final stages in constructing the vertex-edge graph involve coalescing collinear linked edges, removing very short dangling edge spurs, capping free terminators and refining vertex positions to the best junction of the edges linked to them.

The result of this sequence of operations on the block are shown in Figure 5. For the simple block image the vertex-edge graph is complete, but the further processing does not assume completeness, rather only that each subgraph of connected entities constitutes a rigid body.

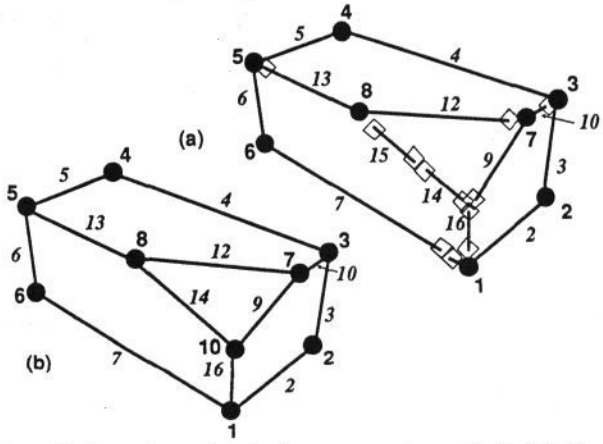


Figure 5. Stages in creating the image vertex-edge graph. In (a) strings have been created and vertices (blobs) inserted at points of high curvature. Free terminators are denoted by diamonds. In (b) the free terminators are linked to nearby vertices and neighbourng straight lines coalesced.

IV OBTAINING 3D STRUCTURE FROM MOTION

Because the visual motion has been computed at edgels which are subsequently linked into extended edges, which in turn are connected by vertices in the image vertex-edge graph, there is a substantial set of constraints restraining the calculation of structure from motion.

The image-wide vertex-edge graph is first split into subgraphs of mutually connected entities. Graph breaks occur at caps and occlusions. The assumptions are then made that, first, each subgraph forms part of a rigid body and, secondly, that straight edges and vertices in the subgraph map onto straight edges and vertices in the world. Thus the scene variables for each subgraph are reduced to the depths Z_i of the terminators of the edges and the motion parameters $\hat{\mathbf{V}}$ and Ω. One cannot of course compute more than the direction of the rectilinear velocity because of the depth/speed scaling ambiguity inherent in visual motion processing: it is impossible to say whether an object is large, far off and travelling quickly or small, near-to and moving slowly. As we shall see, this ambiguity is only removed after matching to a model of known size.

Let the terminator of a straight 3D edge have coordinates

$$\mathbf{R}_i = (X_i\ Y_i\ Z_i)\ .$$

It will be imaged by a camera with a pinhole lens at the origin, optic axis along the positive $\hat{\mathbf{z}}$ axis and image plane $z = -l$, at

$$\mathbf{r}_i = (x_i\ y_i\ -l) = -l\frac{\mathbf{R}_i}{Z_i}\ ,$$

a point which will be a terminator in the vertex-edge graph, ie., a vertex, occlusion or cap.

If the scene point is moving as

$$\dot{\mathbf{R}}_i = \mathbf{V} + \Omega\times\mathbf{R}_i$$

with respect to the camera then the predicted full visual motion $\dot{\mathbf{r}}_i = (\dot{x}_i\ \dot{y}_i)$ at the image point is

$$\dot{\mathbf{r}}_i = -l\frac{\mathbf{V}}{Z_i} + \Omega\times\mathbf{r}_i - \mathbf{r}_i\hat{\mathbf{z}}\cdot\left[\frac{\mathbf{V}}{Z_i} - \frac{\Omega\times\mathbf{r}_i}{l}\right].$$

At an image point $\mathbf{r}$ lying on the straight edge between the two terminators i and j such that

$$\mathbf{r}(\lambda) = \lambda\mathbf{r}_j + (1-\lambda)\mathbf{r}_i \quad (0\le\lambda\le1)\ ,$$

the predicted full visual motion will be

$$\dot{\mathbf{r}}(\lambda) = \lambda\dot{\mathbf{r}}_j + (1-\lambda)\dot{\mathbf{r}}_i\ .$$

However, our motion algorithm computes visual motion at edgel positions, $\mathbf{r}_e$, which will in general not lie *precisely* on the straight line between i and j, but will be scattered either side of it. To determine λ, we approximate it to that appropriate for the nearest point on the line: thus,

$$\lambda = \frac{(\mathbf{r}_e-\mathbf{r}_i)\cdot(\mathbf{r}_j-\mathbf{r}_i)}{(\mathbf{r}_j-\mathbf{r}_i)\cdot(\mathbf{r}_j-\mathbf{r}_i)}\ .$$

Suppose that the motion algorithm computes a component $\mathbf{v}$ of the full motion at the edgel at point $\mathbf{r}_e$. The magnitude of the component of the *predicted* full motion in this direction, i.e. the predicted magnitude of the component itself, is estimated from the scene parameters as

$$s = s(\mathbf{V}, \Omega, Z_i, Z_j, \lambda, \hat{\mathbf{v}}) = \dot{\mathbf{r}}(\lambda)\cdot\hat{\mathbf{v}}\ ,$$

where $\hat{\mathbf{v}}$ is the unit vector in the direction of $\mathbf{v}$.

Hence, writing $\Gamma_i = 1/Z_i$ and $\Gamma_j = 1/Z_j$ for the inverse depths, the expression for the predicted magnitude of the motion component becomes

$$\begin{aligned} s = {} & V_x\{[\Gamma_i(\lambda-1) - \Gamma_j\lambda]l\cos\theta\} \\ & V_y\{[\Gamma_i(\lambda-1) - \Gamma_j\lambda]l\sin\theta\} \\ & V_z\{[\Gamma_i(\lambda-1)f_i - \Gamma_j\lambda f_j]\} \\ & \Omega_x\{[-(\lambda-1)f_iy_i + \lambda f_jy_j + l^2\sin\theta]/l\} \\ & \Omega_y\{[(\lambda-1)f_ix_i - \lambda f_jx_j - l^2\cos\theta]/l\} \\ & \Omega_z\{[(\lambda-1)g_i - \lambda g_j]\}\ , \end{aligned} \qquad \text{(IV.1)}$$

where

$$\cos\theta = \hat{\mathbf{v}}\cdot\hat{\mathbf{x}}\ ,\quad \sin\theta = \hat{\mathbf{v}}\cdot\hat{\mathbf{y}}$$

and

$$f_i = x_i\cos\theta + y_i\sin\theta\ ,\quad f_j = x_j\cos\theta + y_j\sin\theta\ ,$$
$$g_i = y_i\cos\theta - x_i\sin\theta\ ,\quad g_j = y_j\cos\theta - x_j\sin\theta\ .$$

In matrix notation, (IV.1) is

$$[s] = [\mathbf{A}][m]\ ,$$

where $[m]$ is the scene motion vector given by

$$[m]^T = (V_x\ V_y\ V_z\ \Omega_x\ \Omega_y\ \Omega_z)\ .$$

Alternatively, rearranging the expression, we have

$$s = \Gamma_i \{(\lambda-1)[V_x l\cos\theta + V_y l\sin\theta + V_z f_i]\} + \Gamma_j \{-\lambda[V_x l\cos\theta + V_y l\sin\theta + V_z f_i]\} + K(\Omega)$$

where $K(\Omega)$ comprises the angular momentum (last three) terms of equation IV.1. In other words,

$$[s] = [\mathbf{B}][\Gamma]+[K(\Omega)]$$

where $[\Gamma]$ is the vector of reciprocal depths

$$[\Gamma]^T=(\Gamma_1\ \Gamma_2\ \cdots\ \Gamma_d)\ .$$

So, assuming values for the reciprocal depths, we can find the motion parameters by solving

$$[v] = [\mathbf{A}][m]\ , \qquad \text{(IV.2)}$$

where $[v]$ is column vector of measured visual motion components. For an overdetermined system of equations we can solve in the classical least-squares sense using the pseudoinverse by writing

$$[\mathbf{A}^T v] = [\mathbf{A}^T\mathbf{A}][m]$$

and then performing a Gauss-Jordan elimination with pivoting. (Probably a more accurate method would be to Householder's method directly on IV.2.)

Conversely, assuming fixed motion, we can solve for the d reciprocal depths in a least-squares sense from

$$[\mathbf{B}^T][v-K]=[\mathbf{B}^T\mathbf{B}][\Gamma]\ .$$

In this way our algorithm avoids a non-linear optimization problem with non-linear constraints. First the depths are fixed at unity and the motion is computed. The motion is held at these newly computed values and the reciprocal depths computed and so on. Note that all three components of **V** must be allowed to vary in the motion minimization phase because during this the depths are fixed with an arbitrary scale.

In Figure 6 we show views of the reconstructed partial wireframe after 200 iterations. The computed depths will be compared with their real world values after resolution of the depth/speed scaling ambiguity during model matching (see Table 2).

The optimization procedure is found to converge gracefully, but rather slowly, as one might expect for problems where minimizations over variables are split. Another difficulty, as we shall see in the experiments on real imagery, is a softness to coupling between the angular velocity and appropriate components of the translational velocity. The two problems suggest that it may be worthwhile devising a more sophisticated optimization technique, possibly one which could incorporate further constraints, such as the higher likelihood that vertices on a closed loop of edges are coplanar.

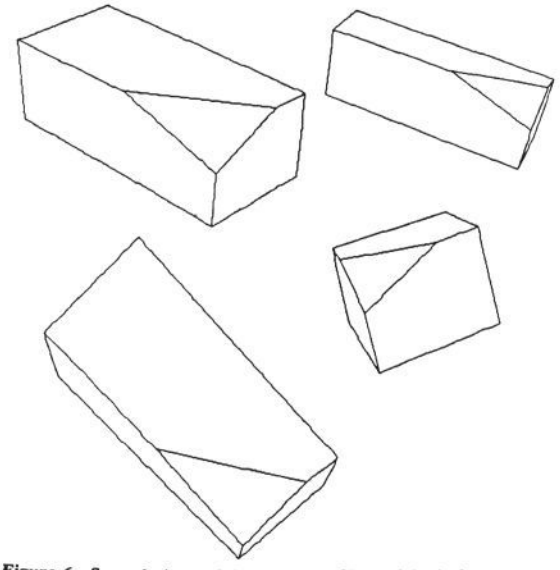


Figure 6. Several views of the computed 3D partial wireframe.

V GEOMETRIC MODEL MATCHING

The partial 3D geometry recovered from the structure-from-motion algorithm is suited to matching to a 3D wireframe model. Murray and Cook (1986) have already described the geometry required to use the generate-and-test algorithm of Grimson and Lozano-Pérez (1985) to match a partial data wireframe with a complete model wireframe. We shall therefore only give a brief résumé here.

The method of Grimson and Lozano-Pérez involves a quasi-exhaustive search of the interpretation space, using simple consistency checks between the geometry of pairs of data and the geometry of the pair of potential matches on the model to establish facts like *if* (edge datum a is matched with model edge i) *then* (edge datum b can/cannot be matched with model edge j). The 'cannot' answers, of which there should be many, are remarkably effective in quenching the otherwise unbridled combinatorial explosion of the matching search space. This process usually generates a small number of feasible interpretations of the data, which must then be tested for *global* geometric consistency by determining the rotation, translation and also, because of the depth/speed ambiguity in our case, *scaling* of the model to fit the data.

A. Generating feasible interpretations

Murray and Cook developed an angle constraint and three directional constraints to impose consistency between the model edges and data edges, based on the body-centred data metrics

$$\hat{\mathbf{e}}_a\cdot\hat{\mathbf{e}}_b \quad \hat{\mathbf{e}}_a\cdot\hat{\mathbf{d}}_{ab} \quad \hat{\mathbf{e}}_b\cdot\hat{\mathbf{d}}_{ab} \quad \hat{\mathbf{e}}_{ab}\cdot\hat{\mathbf{d}}_{ab} \qquad \text{(V.1)}$$

(where $\hat{\mathbf{e}}_a$ and $\hat{\mathbf{e}}_b$ are edge directions, $\hat{\mathbf{d}}_{ab}$ is a unit vector pointing between any two positions on edges a and b, and $\hat{\mathbf{e}}_{ab}$ is a unit vector in the direction of $\hat{\mathbf{e}}_a\times\hat{\mathbf{e}}_b$) and their equivalents on the model. In addition, the constraints determine consistency between the arbitrary but fixed signs of edges on the model and the arbitrary signs given to the data edges before matching. That is, where possible the matcher determines which end of an edge fragment is nearer which end of a model edge. This resolution of the edge fragments' director/vector ambiguities increases the power of the constraints as the search ventures deeper into matching space.

As an example of the use of constraints in Murray and Cook (1986), we outline here the simplest, the angle constraint. It demands that if edge fragments a and b are assigned putatively to model edges i and j then the range of possible angles between sensed fragments must embrace the angle between the edges on the model.

On the model all edges have an arbitrarily chosen, but unambiguous sign. If $\hat{\mathbf{M}}_i$ and $\hat{\mathbf{M}}_j$ are unit vectors in the direction of the model edges, the angle between them is

$$A_{ij} = \cos^{-1}(\hat{\mathbf{M}}_i\cdot\hat{\mathbf{M}}_j)\ .$$

The signs of the data fragments however are initially ambiguous. If we arbitrarily assign a vector $\hat{\mathbf{e}}_a$ to the direction of fragment a, its actual direction (i.e. that consistent with the model) could be $\pm\hat{\mathbf{e}}_a$. To take account of this we maintain a table of the signs of edges, putting them in to five categories. The sign of a can be

$$(U) \quad (+) \quad (-) \quad (+n) \quad (-n)$$

for uncertain, definitely $+\hat{\mathbf{e}}_a$, definitely $-\hat{\mathbf{e}}_a$, and of the same sign as, and different sign from, edge fragment n. With no knowledge of the signs, there are then two possibilities for the sensed angle between fragments a and b. If the fragments have the same sign the angle is

$$\gamma_{ab} = \cos^{-1}(\hat{\mathbf{e}}_a\cdot\hat{\mathbf{e}}_b)\ ,$$

and if different

$$\gamma_{ab}{}^* = \cos^{-1}(-\hat{\mathbf{e}}_a\cdot\hat{\mathbf{e}}_b) = \pi-\gamma_{ab}\ .$$

Including sensing error angles α_a and α_b, for a valid pairing at least one of two logical expressions must be satisfied:

$$l_s = max[(\gamma_{ab}-\alpha_a-\alpha_b),0] \le A_{ij} \le min[(\gamma_{ab}+\alpha_a+\alpha_b),\pi]$$

OR

$$l_d = max[(\gamma_{ab}{}^*-\alpha_a-\alpha_b),0] \le A_{ij} \le min[(\gamma_{ab}{}^*+\alpha_a+\alpha_b),\pi]\ .$$

The subscripts s and d indicate that these are the satisfaction conditions for when fragments a and b have the same or different signs. However, it is important to note that because of measurement uncertainties the logical *OR* above is not exclusive.

There are several output conditions depending on the fragment signs input to the constraint as described by Murray and Cook. Suppose that:

(1)**on entry the signs of a and b are both uncertain (U).** Then, if l_s is true and l_d false, the pairing is valid and the two edge fragments a and b must have the same sign and can be relabelled $(+b)$ and $(+a)$, respectively. Conversely, if l_s is false and l_d true then the pairing is valid and the two edge fragments a and b must have different signs and can be relabelled $(-b)$ and $(-a)$, respectively. If both l_s and l_d are true, nothing is learned about the signs, and if both l_s and l_d are false the pairing is invalid and the search backtracks.

(2)**the sign of one edge fragment is known on entry,** say a is signed $(-)$. Then when l_s is true and l_d false, fragment b must also have sign $(-)$. Conversely, if l_s is false and l_d true, fragment b must be signed $(+)$. If both tests succeed the sign of b remains (U), and if both fail, the pairing is invalid.

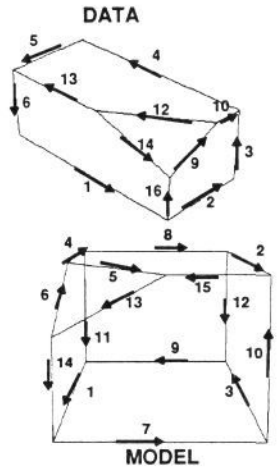
(3)**fragments a and b are signed as $(-n)$ and (U) on entry.** Then, if l_s is true and l_d false, b must be signed $(-n)$, and so on. If a and b are signed $(-n)$ and $(+m)$ on entry then if l_s is true and l_d false then we know that $(+m)=(-n)$, and so on.

(4)**both signs are known absolutely on entry.** If the signs are identical, then for a valid pairing l_s *must* be be true. If different, l_d *must* be true.

Whenever a sign is changed a check is made for any other signs which depend on it and they are updated recursively.

We note that, given *two* edge fragments with uncertain signs, the angle constraint can never determine the absolute sign for each fragment because it involves a product. However, the first two direction constraints of expression (V.1) involve only one edge vector and so can determine the absolute sign of that edge. Thus those direction constraints determine signs and the angle constraint and third direction constraint propagate them.

The results from matching to the block geometry are shown in Figure 7. It turns out in this experiment that there is only one feasible interpretation found during the generation phase, which by inspection of the model and data edge labels and directions is clearly valid. This is formally tested as described below.



Matches		
D	M	+-
1	1	+
2	7	+
3	10	+
4	2	-
5	8	-
6	11	+
9	13	-
10	15	-
12	5	-
13	4	+
14	6	-
16	14	-

Figure 7. The partial wireframe edge labels and directions, the model edge labels and directions, and the single interpretation obtained by the edge matcher. This interpretation is verified by the location algorithm (Section V.B).

B. Testing the feasible interpretations

Because of the use of pairwise constraints, there is no guarantee that a feasible interpretation is consistent with a single global transformation between model and sensor spaces. Therefore each interpretation must be tested by finding the rotation, scaling and translation $([\mathbf{R}], S, \mathbf{t})$ which link model (μ) and sensor (σ) spaces as

$$\sigma = S[\mathbf{R}]\mu + \mathbf{t} .$$

We determine the rotation first, using the quaternion method of Faugeras and Hébert (1983), and then find the scaling and translation simultaneously in a least squares procedure.

In Table 1 we show the results of this test on our block data together with the 'real world' values of $[\mathbf{R}]$ and $\mathbf{t}$ obtained from the CSG model from which the initial images were synthesized.

	-0.72	-0.01	-0.70
$[\mathbf{R}]_{comp}=$	0.32	0.90	-0.32
	0.62	-0.48	-0.65
$\mathbf{t}_{comp}=$	-0.10	-0.22	0.88
$S=$	0.0237		
	-0.71	0.000	-0.71
$[\mathbf{R}]_{CSG}=$	0.35	0.87	-0.35
	0.61	-0.50	-0.61
$\mathbf{t}_{CSG}=$	-0.10	-0.22	0.90

Table 1. The computed rotation, translation and scale. The real world rotation and translation computed from the CSG model are shown for comparison.

If the model data base includes only uniquely shaped objects, the computed scale, S, resolves the depth/speed scaling ambiguity and hence allows absolute location of the object and determination of its speed. The depths and speed delivered by the structure-from-motion algorithm are related to their required values (Z_{comp} and V_{comp}) by

$$Z_{comp} = \frac{Z_{sfm}}{S} \quad , \quad V_{comp} = \frac{V_{sfm}}{S} \quad .$$

In Tables 2 and 3 we compare the computed values of the motion and depths with their real world values from the CSG model.

$\hat{\mathbf{V}}_{comp}=$	-0.03	1.00	-0.00
$V_{comp}=$	0.27		
$\boldsymbol{\Omega}_{comp}=$	-0.00	0.00	0.00
$\hat{\mathbf{V}}_{CSG}=$	0.00	1.00	0.00
$V_{CSG}=$	0.29		
$\boldsymbol{\Omega}_{CSG}=$	0.00	0.00	0.00

Table 2. *The computed velocity direction, speed and angular velocity compared with their real world (CSG) values. The speed is in model units per frame. The velocity direction is in error by only 1.5 °.*

Computed Depths	CSG Depths
35.5	35.4
37.9	37.8
37.3	37.9
39.5	40.0
40.0	40.0
42.4	44.0
48.3	47.4
50.2	51.4
55.8	53.5

Table 3. The computed depths compared with the real world (CSG) values. The depths are listed in order of increasing CSG depth.

VI REAL OBJECTS AND REAL IMAGES

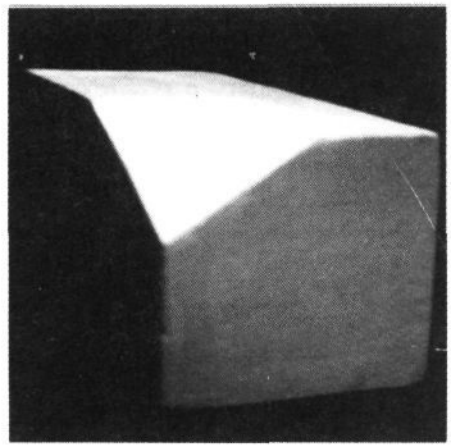
It was noted in the introductory section that structure-from-motion algorithms are typically ill-conditioned — that is, the scene reconstruction is very sensitive to noise in the input data. Noise arises not only because of electrical and optical distortions in the camera, but also because of deficiencies in the scene and image descriptions. To examine the performance of the sequence of algorithms in the presence of significant noise we describe here experiments on real images of a wooden incarnation of the chipped block.

The block itself was roughly carpentered from balsa wood with overall size approximately $164\times75\times58mm^3$ and painted with a light matt paint. The images were captured from a CCD camera with 512×512 rectangular pixels which were resampled in software to produce 512×384 images with square pixels. The centred 384×384 section was used for experiment. No special calibration of the camera was performed. It was assumed that the optic axis passed through the centre of the captured image, that the pixel width was given by the nominal width of the CCD chip of $8.8mm$ divided into 512 pixels and that the focal length of the

lens was the quoted $24mm$, and gave perfect perspective images. In all the experiments the full cone angle subtended by the object to the camera did not exceed 15.7°.

In the first experiment the camera was translated vertically (along positive $\hat{y}$) at $3mm.F^{-1}$, with the object positioned around $400mm$ in front of the camera and illuminated by diffuse natural light. (F^{-1} denotes *per frame*). The centre frame of the sequence is shown in Figure 8 along with the thresholded edgemap. It can be seen that the somewhat rounded edges of the object and lighting and reflectance effects lead to rather meandering edges, an example of noise introduced by incomplete modelling of the scene and image. The computed flow and three stages in the segmentation are shown in Figure 9. In Figure 10 we show several views around the partial wireframe implied by the depth map computed by the structure-from-motion algorithm. The overall structure is excellent, although the top view shows that there is some distortion, reflecting the fact that the loci of error around each depth point are prolate ellipsoids extended along the line of sight.

Despite the distortion, the inclusion of tolerance to angular and positional errors in the matching algorithm made it possible to match all twelve of the observed edges to the model wireframe, as shown in Figure 11. (Note that the carpentered block has different model labelling than the CSG model block.) From the computed scale we derived the estimates for the absolute depth and speed shown in Table 4.



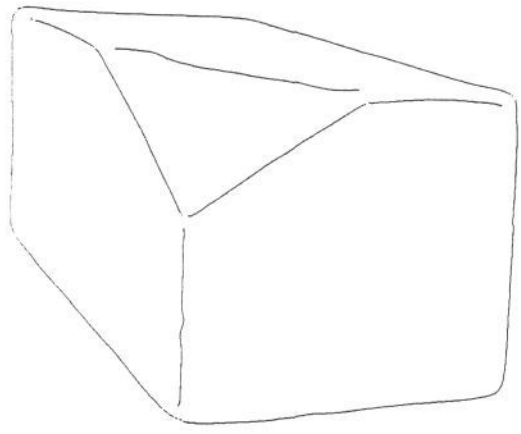
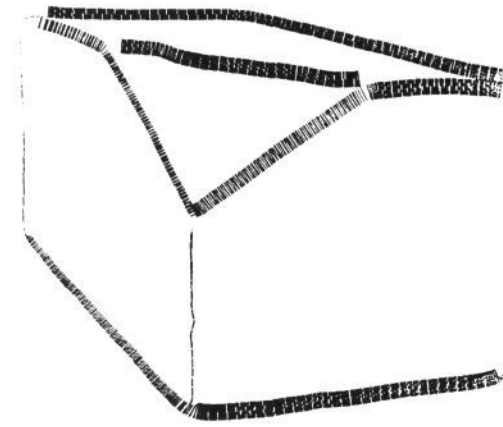


Figure 8. The central image and thresholded edge map for experiment 1.



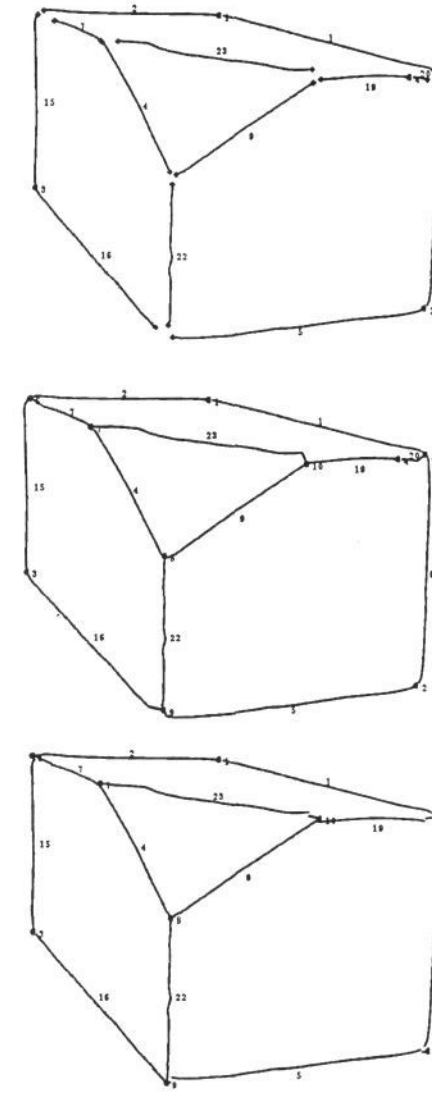


Figure 9. The optic flow computed from the sequence in Figure 8, and three stages in the segmentation of the central edge-map. The effect of recomputing the vertex positions can be seen clearly.

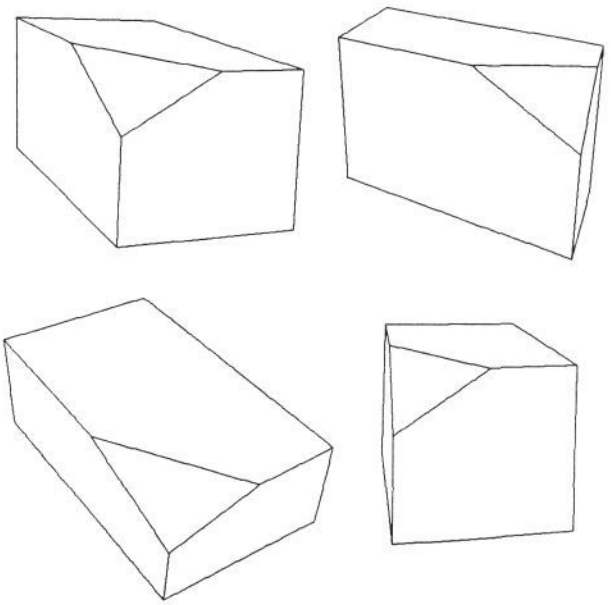
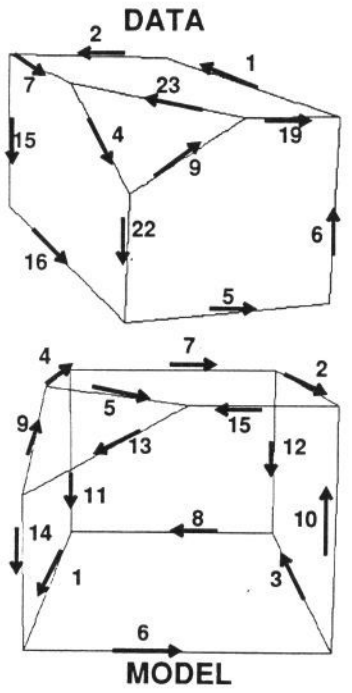


Figure 10. Several views of the 3D partial wireframe. There was no constraint on the angular velocity.



Matches		
D	M	+-
1	2	-
2	7	-
4	9	-
5	6	+
6	10	+
7	4	-
9	13	-
15	11	+
16	1	+
19	15	-
22	14	+
23	5	-

Figure 11. The data to model matching for experiment 1. Only one feasible interpretation exists.

Velocity Direction	+0.058	+0.998	-0.029
Speed	$7.2mm.F^{-1}$		
Angular Velocity	+0.097	-0.008	+0.000
Vertex	Depth *mm*		
5	403		
1	473		
2	407		
9	390		
8	388		
7	424		
6	466		
3	464		
10	395		

Table 4. The computed motion and depths for sequence 1 with no constraint on the angular velocity.

The depth values are good. However, although the translational velocity direction is correct, the translational velocity magnitude is over estimated at $7mm.F^{-1}$. The reason for this may be found in the x-component of angular velocity. Its value of $10^{-2}rad.F^{-1}$ coupled with the depth of around $400mm$ leads to the discrepancy or compensation in the rectilinear velocity of $4mm.F^{-1}$. (In effect, the camera erroneously believes it is panning upwards, tracking the block, and so to recreate the observed visual motion, the block must be translating faster than it really is.)

In the presence of noise, quasi-instantaneous or snapshot processing is particularly soft to this sort of coupling, and it is most troublesome in the early stages of our minimization procedure when the depth variations are small. In many practical circumstances however it will be possible to put bounds in the size of the angular velocity, and in a second analysis of the same sequence we have constrained the angular velocity to be zero. The resulting reconstructed wireframe is shown in Figure 12 and, after matching, the computed scale yields the depth and motion values of Table 5. The speed is now more correctly estimated as $2.9mm.F^{-1}$.

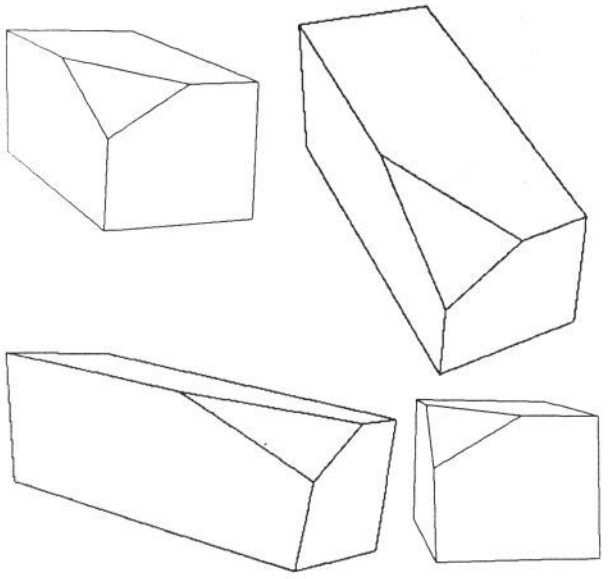


Figure 12. The partial wireframe obtained by the structure-from-motion algorithm using the flow of Figure 9 with the angular velocity constrained to be zero.

Velocity Direction	+0.026	+0.999	-0.004
Speed	$2.9mm.F^{-1}$		
Angular Velocity	0	0	0
Vertex	Depth *mm*		
5	356		
1	526		
2	369		
9	336		
8	324		
7	398		
6	507		
3	500		
10	339		

Table 5. The computed motion and depths for sequence 1 with the angular velocity constrained to be zero. The vertex index is that from the segmentation.

In the second experiment the block was moved away from the camera at $10mm.F^{-1}$: the central frame and edgemap are shown in Figure 13. The computed flow and segmentation are shown in Figure 14. This experiment further highlighted the difficulty introduced by the angular velocity coupling. The structure obtained with no constraint was poor. On constraining the angular velocity to be zero, however, the recovered structure was much improved (Figure 15), although even here it was rather distorted by the vertices nearest the image centre. The depth estimate at these points is more uncertain because they lie nearer the focus of expansion of the flow field and the flow is smaller. It was only possible to match seven of the observed edges to the model wireframe, resulting in rather poor localization and, *inter alia*, an over-estimate of the speed as $15mm.F^{-1}$, as shown in Table 6.

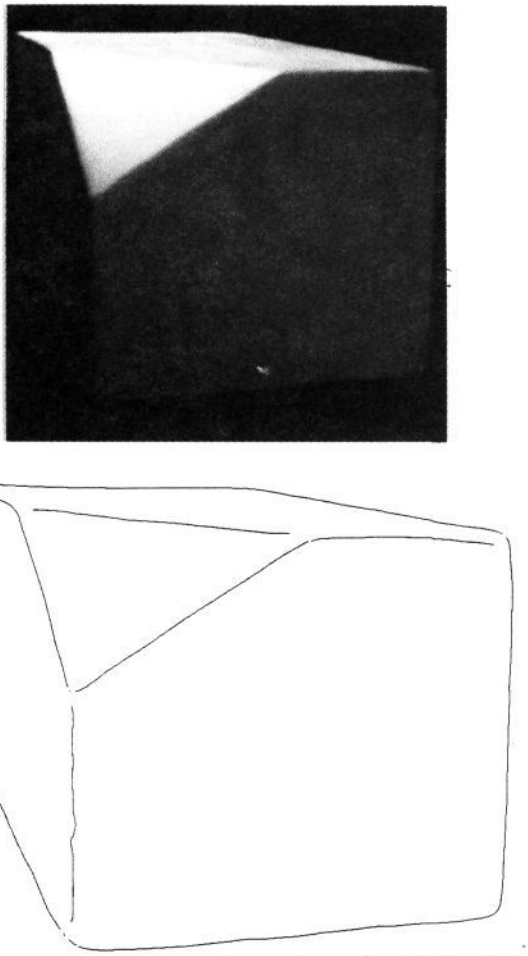
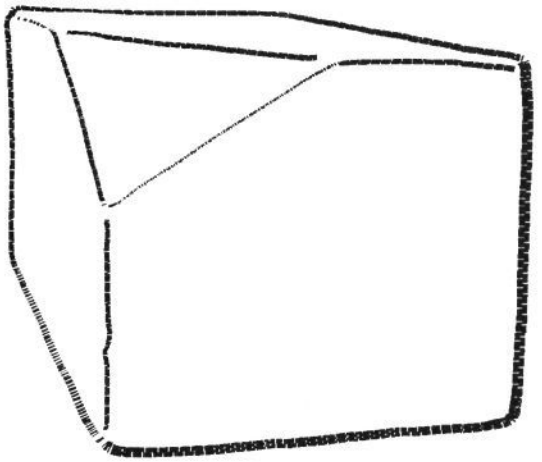


Figure 13. The central image and edgemap for experiment 2. The block is moving away from the camera.



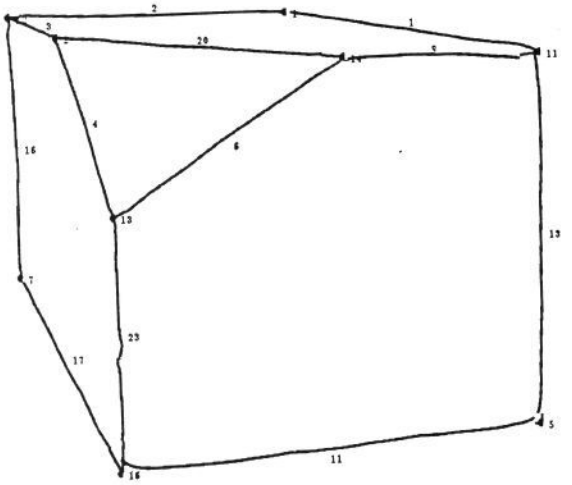


Figure 14. The computed flow and segmentation from the images in Figure 13.

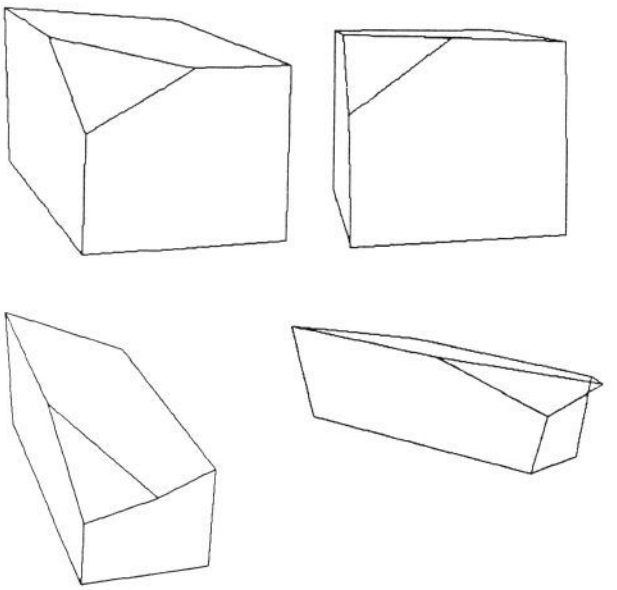


Figure 15. Views of the computed 3D partial wireframe for experiment 2 with the angular velocity constrained to be zero.

Velocity Direction	+0.017	+0.029	+0.999
Speed	$14mm.F^{-1}$		
Angular Velocity	0	0	0
Vertex	Depth *mm*		
11	392		
1	566		
5	407		
16	393		
7	614		
12	639		
2	482		
13	377		
14	353		

Table 6. The computed motion and depths for sequence 2 with the angular velocity constrained to be zero.

VII CONCLUSIONS AND DISCUSSION

Our conclusions can be stated simply. We have a first attempt at a system that can travel up the computational hierarchy of representations from the pixel grey levels in a sequence of 2D images, through 2D edges representations and visual motion to 3D edge representations and scene motion, and finally to symbolic descriptions of those moving 3D edges.

There appears to be a strong principal reason for the ability of the system to obtain quite accurate depth and motion values, and that is the success of the segmentation of the image and visual motion. The segmentation problem is of course greatly eased because of our restricted world domain but there are, nevertheless, outstanding problems with it. We will discuss these briefly now.

The difficulties are captured by the line drawing of Figure 16 which shows one cuboid (B) in the background occluded by another (F) in the foreground. Is cuboid B rigidly attached to or not, and is there a depth discontinuity at point X or not?

The simplest remedy would be to try to detect all 'T' junctions, where one line meets another continuous line, mark them as potential occlusions and break the subgraphs at these points However, even if this could be done with certainty, we run the

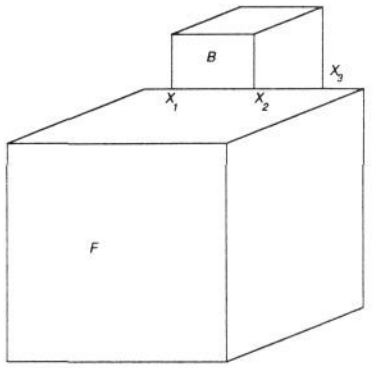


Figure 16. The problem of occlusions.

risk of erroneously splitting up two entities which *together* form an interesting shape for recognition but *apart* are rather dull, and at the same time reducing the size of the set of constraints used in finding structure from motion. A much more attractive way might be to refine our decisions on occlusions by detecting anomalies in the visual motion at the virtual vertex at the image junction of the occluding and occluded lines. A further possibility, computationally more difficult, may be to weaken the assumptions about rigidity at likely occlusions, and then try to fit the entire graph allowing breaks to appear at putative occlusions as an when needed. These issues remain to be addressed in a determined manner.

It is worth returning to the more general importance of segmentation in visual motion processing. If our laboratory is typical, it would seem that there must be many elegant ways of determining visual motion which languish unused simply because it is difficult to find ways of usefully relating the visual motion in the image to its originating structure in the scene, and similarly many structure-from-motion algorithms which are redundant because there is no way of segmenting the geometry-related visual motion they require from the image. If *detailed* 3D structure is required from visual motion it appears that one must be able to link image representations associated with the flow with the 3D geometric representations associated with the scene. In this work we have used a consistent representation of 2D and 3D edges and vertices throughout the processing. This provides a strong set of geometrical constraints which allow us to identify visual motion with the geometrical flow field from physical structures moving in the scene and provide stability to the structure-from-motion computations. In these circumstances we can indeed recover accurate and recognizable 3D geometry and motion.

Inherently more difficult to link up in this way are representations based on surfaces. For surfaces one must detect both boundaries in 3D motion and orientation from the visual motion, which in the absence of other clues about the 3D, implies a simultaneous solution of the segmentation and structure-from-motion problems. Perhaps at the extremes of difficulty are the far less structured environments, for example, natural scenes of trees, where it is difficult to define what we mean by segmentation let alone perform it. Perhaps in these circumstances all we should hope for are qualitative notions of scene depth and motion.

Finally, we comment on temporal stability. There is clearly a limit to the quality of information that one can obtain using 'snapshot' processing. To hope to recover complete and accurate structure and motion of a scene on the basis of a handful of frames, which at video rates would span say 60ms, is rather optimistic and it is gratifying that the algorithms presented here perform as well as they do. Newtonian laws (and snooker players) tell us that we should expect spatio-temporal coherence and that motion can be predicted even through collisions. Figure 17 shows that after only *one* iteration the structure-from-motion algorithm obtains qualitatively sensible results for the depth values for the CSG sequence. Thus, it would seem practicable within our system to perform fewer iterations at each frame, but integrate the partial results over several frames, building up more certain estimates for the motion and structure.

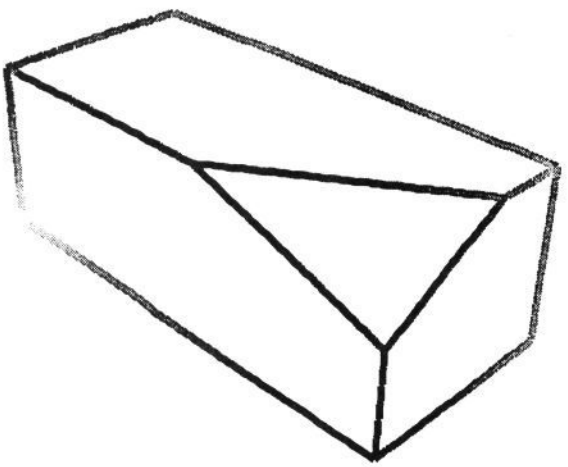


Figure 17. After only one iteration, the structure-from-motion algorithm can achieve qualitatively sensible results. Darker points are nearer the camera. This was for the CSG block with unconstrained motion.

ACKNOWLEDGMENTS

The authors thank Guy Scott, Chris Harris and Steve Maybank for useful comments and AIVRU Sheffield University and IBM UK, GEC Research's Alvey partners on IKBS Project 025, for the use of the Canny edge finding code and the CSG body modeller, respectively. Whilst completing this work, one of us (DWM) was the GEC Visiting Fellow at University College, Oxford, and he wishes to thank the College, the Department of Engineering Science and Professor Mike Brady for providing facilities.

REFERENCES

ADIV G. (1985). *Determining 3-D motion and structure from optical flow generated by several moving objects.* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.PAMI-7, pp.384-401.

ASADA, H and BRADY, J.M. (1984). *The curvature primal sketch.* Proc. Workshop on Computer Vision: Representation and Control, Annapolis, MD, (April 1984) pp.8-17. IEEE Computer Society Press, Silver Spring, MD.

BRUSS, A.R. and HORN, B.K.P. (1983). *Passive Navigation.* Computer Vision, Graphics and Image Processing, Vol.21, pp.3-20.

BUXTON, B.F., BUXTON, H., MURRAY, D.W. and WILLIAMS, N.S. (1984). *3D solutions to the aperture problem.* Proc. 6th European Conf. on Artificial Intelligence, Pisa, 1984, (ed. T. O'Shea) pp.631-640, Elsevier, Amsterdam.

CANNY, J.F. (1983). *Finding edges and lines in images.* AI-TR No. 720, Artificial Intelligence Laboratory, MIT, Cambridge, MA.

CASTELOW, D.A., MURRAY, D.W., SCOTT, G.L. and BUXTON, B.F. (1987) *Matching Canny edgels to compute the principal components of optic flow.* This conference.

CLOCKSIN, W.F. (1980). *Perception of surface slant and edge labels from optical flow: a computational approach.* Perception, Vol.9, pp.253-269.

FAUGERAS, O.D. and HEBERT, M. (1983) *A 3D recognition and positioning algorithm using geometric matching between primitive surfaces.* Proc. Int. Joint Conf. on Artificial Intelligence IJCAI-83. pp. 996-1002.

FENNEMA, C.L. and THOMPSON, W. B. (1979). *Velocity determination in scenes containing several moving objects.* Computer Graphics and Image Processing, Vol.9, pp.301-315.

GRIMSON W.E.L. and LOZANO-PEREZ, T. (1984). *Model-based recognition and localization from sparse range or tactile data.* International Journal of Robotics Research Vol.3 , pp.3-35.

GRIMSON W.E.L. and LOZANO-PEREZ, T. (1985). *Search and sensing strategies for recognition and localization of two and three dimensional objects.* 3rd Int. Symp. of Robotics Research, Gouvieux, France (Oct 1985), pp.81-88

HARRIS, C.G., IBISON, M.C., SPARKES, E.P. and STEPHENS, M. (1986). *Structure and motion from optical flow* Proc. Alvey Vision Conference, Bristol (Sep. 1986). To be published in Image and Vision Computing.

HELMHOLTZ, H. von (1866). *Handbuch der physiologischen Optik.* Voss, Leipzig. Translation by J.P.C. Southall (ed.) (1962). *Physiological Optics.* Dover, New York.

HILDRETH, E.C. (1984). *Computations underlying the measurement of visual motion.* Artificial Intelligence, Vol.23, pp.309-354.

HORN, B.K.P. and SCHUNCK, B.G. (1981). *Determining optical flow.* Artificial Intelligence, Vol.17, pp.185-203.

JOHANSSON, G. (1973). *Visual perception of biological motion and a model for its analysis.* Perception and Psychophysics, Vol.14, pp.201-211.

LAWTON, D.T. (1983). *Processing translational motion sequences.* Computer Vision, Graphics and Image Processing, Vol.22, pp. 116-144.

LONGUET-HIGGINS, H.C. (1981). *A computer algorithm for reconstructing a scene from two projections.* Nature, Vol.293, pp.133-135.

LONGUET-HIGGINS, H.C. (1984). *The visual ambiguity of a moving plane.* Proc. R. Soc. Lond. B, Vol.223, pp.165-175.

LONGUET-HIGGINS, H.C. and PRAZDNY, K. (1980). *The interpretation of a moving retinal image.* Proc. R. Soc. London B, Vol. 208, pp.385-397.

MURRAY, D.W. and BUXTON, B.F. (1987). *Scene segmentation from visual motion using global optimization.* IEEE Transactions on Pattern Analysis and Machine Intelligence Vol. 9 pp.220-228.

MURRAY, D.W. and COOK, D.B. (1986). *Using the orientation of fragmentary 3d edge segments for polyhedral object recognition* Submitted to International Journal of Computer Vision.

MURRAY, D.W. and WILLIAMS, N.S. (1986). *Detecting the image boundaries between optical flow fields from several moving planar facets.* Pattern Recognition Letters Vol.4, pp.87-92.

NEUMANN, B. (1980). *Motion analysis of image sequences for object grouping and reconstruction.* Proc. of Pattern Recognition Conference, Miami, Florida, 1980.

PROFFITT, D.R. and BERTENTHAL, B.I. (1984). *Converging approaches to extracting structure from motion: psychophysical and computational investigations of recovering connectivity from moving point-light displays.* Proc. First Conf. on Artif. Intell. Applications, Denver, Co, Dec 1984, pp. 232-238, IEEE Computer Society Press, Silver Spring, MD.

SCOTT, G.L. (1986). *A single method for extracting both point and edge motions from a pair of images.* Proc. Alvey Vision Conference, Bristol (Sept 1986) and to be published in Image and Vision Computing.

SPOERRI, A. and ULLMAN, S., 1987. *The early detection of motion boundaries* Proceedings of the First Internation Conference on Computer Vision. IEEE Computer Society Press, Washington, DC., 1987, pp209-218.

THOMPSON, W.B. (1980). *Combining motion and contrast for segmentation.* IEEE Transactions on Pattern Analysis and Machine Intelligence Vol.PAMI-2, pp.543-549.

THOMPSON, W.B. and BARNARD, S.T. (1981). *Lower-level estimation and interpretation of visual motion.* IEEE Computer, Vol.14, pp.20-28. THOMPSON, W.B. and PONG, T-C., 1987. *Detecting moving objects* Proceedings of the First Internation Conference on Computer Vision. IEEE Computer Society Press, Washington, DC., 1987, pp201-208.

TSAI, R.Y. and HUANG, T.S. (1984). *Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces.* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.PAMI-6, pp.13-27.

ULLMAN, S. (1979). *The interpretation of visual motion.* MIT Press, Cambridge, MA.

ULLMAN, S. and HILDRETH, E.C. (1983). *The measurement of visual motion.* in *Physical and biological processing of images.* (eds O.J. Braddick and A.C. Sleigh), Springer-Verlag, Berlin.

WAXMAN, A.M. and WOHN, K. (1985). *Contour evolution, neighbourhood deformation and global image flow: planar surfaces in motion.* International Journal of Robotics Research, Vol.4, pp.95-108.

WAXMAN, A.M. and ULLMAN, S. (1985). *Surface structure and 3D motion from image flow kinematics.* International Journal of Robotics Research, Vol.4, pp.72-94.

WESTPHAL, H. and NAGEL H-H. (1986). *Towards the derivation of three-dimensional descriptions of image sequences for nonconvex moving objects.* Computer Vision, Graphics and Image Processing Vol.34, pp.302-320.

WALLACH, H. and O'CONNELL, D. N. (1953). *The kinetic depth effect.* Journal of Experiment Psychology, Vol.4, pp.205-217.