3-D OBJECT RECOGNITION THROUGH IMPLICIT MODEL MATCHING

KENNETH M. DAWSON-HOWE and DAVID VERNON

Department of Computer Science
Trinity College, Dublin 2, Ireland

A new approach to object recognition is presented, in which secondary representations of 3-D models are synthesized/derived (in various forms) and subsequently compared in order to invoke views of models, tune model pose and verify recognition hypotheses. The use of these secondary representations allows complex models (e.g. surface-based or volumetric models) to be compared implicitly (rather than explicitly comparing the component primitives of the models). This in turn overcomes the problem of the stability of the model primitives, and provides independence between the complex 3-D representations and the recognition strategy (i.e. the invocation, matching and verification techniques). The secondary representations employed are Extended Gaussian Images, directional histograms, needle diagrams, depth maps and boundary curvature signatures.

The technique is demonstrated using models, of reasonably complex objects, derived from actively sensed range data.

Keywords: Object recognition, 3-D modelling, active sensing, extended Gaussian images, histograms, needle diagrams, depth maps.

1. INTRODUCTION

A large number of the current approaches (see Refs. 1–4 for surveys) to 3-D object recognition address the problem by comparing 3-D object models (i.e. comparing 3-D models extracted from image data with 3-D object models which are known *a priori*). These comparisons are typically performed directly, between the basic component primitives of the object models (e.g. edges, or surfaces). Unfortunately, this requires that these primitives should be stable (i.e. be represented in the same way) regardless of viewing direction, noise, and possible occlusion, and this in turn, places serious constraints on the domain of objects which can be reliably, and efficently, recognized. In other words, it seems to be generally considered that "the matching problem will not be solved until an adequate representation is defined".[3]

Consider, for example, the use of surface information, which provides an extremely powerful representation for use within object recognition strategies, due to the level of abstraction and the richness of the data. Most researchers who have addressed the direct comparison of surfaces (e.g. see Refs. 5, 6, 10 and 11) have encountered problems due to stability of the primitives. Note that Ref. 11 documents a comparable system (to the one documented in this paper), based on the use of segmented surface primitives which are matched using an interpretation tree. However, they point out in a previous paper[10] that "the surface fitting problem is a difficult one".

International Journal of Pattern Recognition and Artificial Intelligence Vol. 9 No. 6 (1995) 959–990

This paper presents a new method of comparing 3-D models which, although working with a surface description, overcomes the need for the segmentation of range data into separate surfaces, and hence overcomes the problem of their (i.e. the surface's) instability. It has been tested extensively using both actively sensed range data, and range data computed from passively sensed images using a depth from camera motion algorithm. The tests using actively sensed range data are documented in this paper and the other tests (the data for which was significantly more noisy), which are only commented upon in this paper, are documented in detail in Refs. 12 and 13.

The problem being addressed is that of the automatic recognition of three-dimensional non-articulate rigid objects (i.e. objects which contain no parts that may move relative to one another). In this context that problem is equated to the matching of a view of a solitary 3-D object model (i.e. which has been segmented from all other objects in some scene), the *viewed* model, with *known* object models in order to recognise the view of the object model observed (Note that the terms *viewed* and *known* when being used in these contexts will appear in italics throughout the paper). Recognition is taken to imply determination of the position and orientation of a *known* object model which best matches the *viewed* model, along with an associated measure of reliability.

In order to overcome the need to segment a surface based representation of an object into a number of reasonably sized surfaces, it is necessary to compare "complete" object models. Object models, though, can only be compared directly (i.e. without requiring the comparison of model primitives) using scalar transform descriptors (where scalar transform descriptors are taken to be representations, in terms of a number of scalar values, of entire objects/shapes), and these, at present, are either relevant only to the 2-D shape domain or are just too simple to be applied in the general 3-D domain. This has led to the current dependance on the comparison of model components. In order to allow the direct comparison of complex 3-D models, it is therefore necessary to develop new, rich, scalar transform descriptors, and that task is addressed in this paper, within the framework of a complete recognition system.

The approach taken is, in some ways, similar to those of Ikeuchi[7] and Krishnapuram *et al.*,[14] in that it addresses object recognition in terms of a number of subproblems (i.e. the separate identification of object orientation and object position). It also draws on the work of Horn[8,9] as the Extended Gaussian Image, and the sampled unit sphere itself, are employed in order, respectively, to allow the distribution of visible surface normals to be considered, and to provide a sampling of the potential values of orientation.

Additionally, a technique, comparable to that of Faugeras[15,16] and Henderson,[17] of generating "3-point seed" surfaces from dense range maps is employed. Segmentation of the resultant surface models is addressed in a fashion similar to that of Han.[18]

The paper addresses, in turn, the representations to be used, the methods of obtaining those representations for both *known* and *viewed* objects, the model

invocation strategy, the model matching strategy, the method of hypothesis verification, the testing which was performed and finally some brief conclusions are made.

2. INTRODUCTION TO THE REPRESENTATIONS

Within the approach presented here, are two distinct classes of object representation:

A. The first class of representation details 3-D objects so that they may be theoretically manipulated and considered from any viewpoint. The representation used here was a simple planar-surface based model similar to that of Roberts.[19] However, due to the use of secondary representations, the recognition strategy is independant of the type of representation used to detail the objects.
B. The second class of representation is derived from the first and, hence, the various different representations used from this class may be regarded as secondary representations of the detailed object models. They may also be regarded as complex scalar transform descriptors, as each representation is composed of a fixed-size array of scalars. These representations depend only on certain object paramaters (e.g. orientation), and are used to guide the recognition system efficiently through the potentially exhaustive search for the correct position and orientation of the correct model. The secondary representations used are directional histograms, needle diagrams, depth maps and boundary curvature histograms and are presented, in turn, in the subsections which follow.

2.1. Directional Histogram

The concept of the "Directional Histogram" was developed for the approach presented herein, and is simply the notion of mapping a single component of the visible 3-D orientations (i.e. the surface normals) of a model to a 1-dimensional histogram, where those components are defined about the coordinate axes of the viewing camera model.

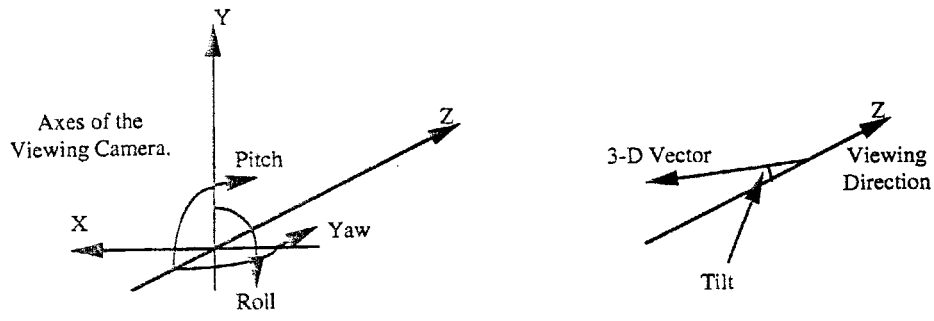


Fig. 1. Definitions of roll, pitch, yaw and tilt about the coordinate axes of the viewing camera. Note that tilt is defined only with respect to the focal axis and hence is independent of roll.

The various components of the orientations used are roll, pitch, yaw and tilt. Roll, pitch and yaw are defined respectively as rotations about Z, X, and Y coordinate axes of the camera model. Tilt is defined as π less the angle between the orientation vector and the focal axis (i.e. the Z-axis of the camera model). See Fig. 1. The directional histogram sums *a component* of the surface normals.

For example in the case of a yaw directional histogram the component of orientation which is employed is that which is in the Y plane (see Fig. 2). The histogram cells range from $-90°$ to $+90°$, that being the possible range of visible surface normals, and typically employ a resolution of around $\frac{1}{4}°$. The histogram is a sum of all the surface normals of the visible cells of the object model (x_i, y_i, z_i) where the angle/cell mapped to in the histogram is defined by $tan^{-1}\frac{z_i}{x_i}$ and the weight is defined by $\sqrt{x_i^2 + z_i^2}$. Roll, pitch and tilt histograms are defined in similar fashions and

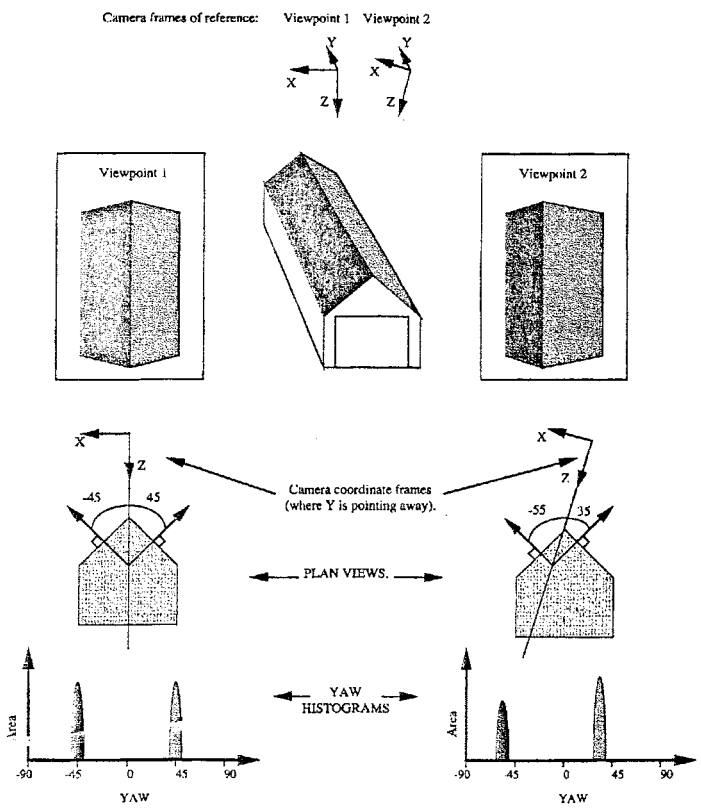


Fig. 2. Example of yaw directional histograms. These two yaw histograms of two views of a garage-like object are a simple example of how directional histograms work. The visible surface areas of the views of the object are mapped to the histograms at their respective yaw angles (which are defined with respect to the focal axes of the camera). In order to aid understanding the sections of the histograms which are mapped from given surfaces are shaded in the grey-level of the surfaces. Notice the shift in the histograms, which is due to the slightly different values of yaw of the two viewpoints.

examples are shown in Figs. 3 to 5. In the case of roll, the visible orientations range from $-180°$ to $+180°$, the angle is defined by $tan^{-1}\frac{x_i}{y_i}$ and the weight is defined by $\sqrt{x_i^2+y_i^2}$. In the case of pitch the visible orientations range from $-90°$ to $+90°$, the angle is defined by $tan^{-1}\frac{z_i}{y_i}$ and the weight is defined by $\sqrt{z_i^2+y_i^2}$. In the case of tilt the visible orientations range from $0°$ to $+90°$, the angle is defined by $|tan^{-1}\frac{z_i}{x_i}|$ and the weight is defined by $\sqrt{x_i^2+z_i^2}$.

Using these histograms the various components of 3-D orientations (i.e. roll, pitch, yaw and tilt) can be considered (and manipulated) separately.

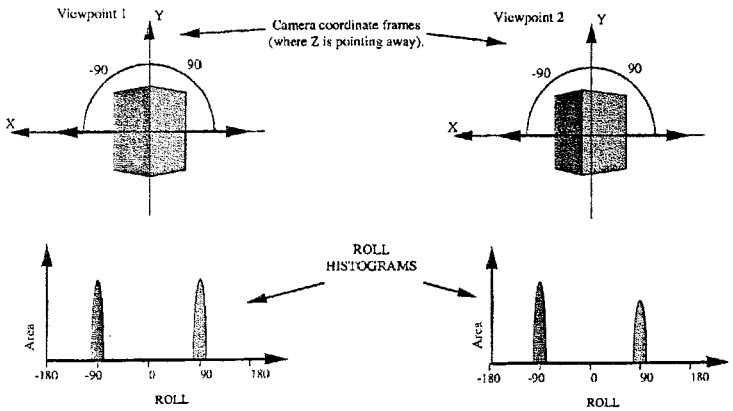


Fig. 3. Example of roll directional histograms (derived from the two views of the garage-like object shown in Fig. 2). While the roll components (i.e. angles) of the orientations for the two views remain the same, the areas mapped change slightly (due to the change in yaw relative to the camera). Again, in order to aid understanding, the sections of the histograms which are mapped from the given surfaces are shaded in the grey-level of the surfaces.

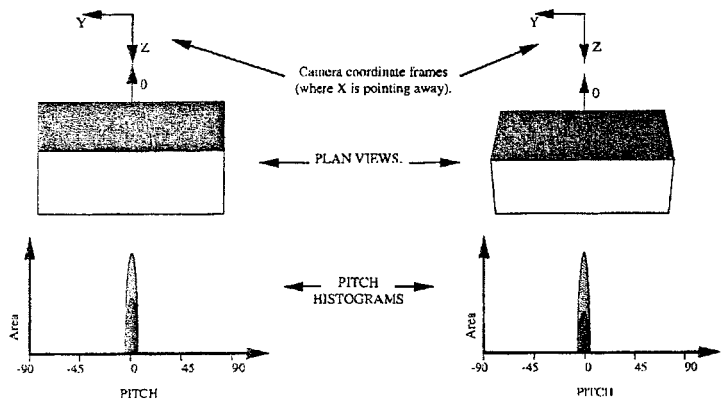


Fig. 4. Example of pitch directional histograms (derived from the two views of the garage-like object shown in Fig. 2). They are similar to the previously shown roll histograms in that the components of the orientations for the two views remain the same although the areas mapped from each individual surface changes slightly (due to the changes in yaw relative to the camera).

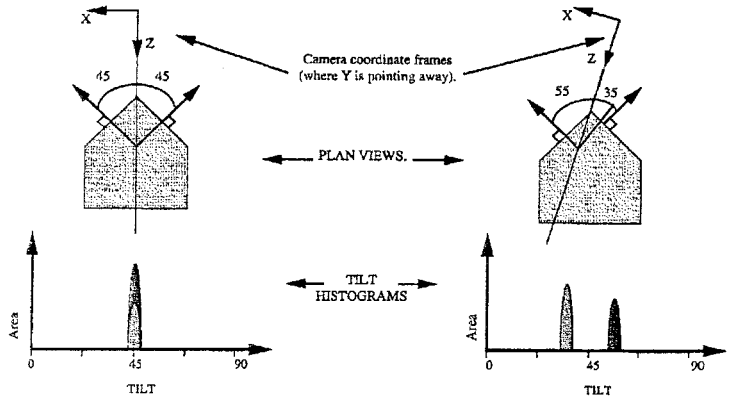


Fig. 5. Example of tilt directional histograms (derived from the two views of the garage-like object shown in Fig. 2). Notice how, for the first view, the two orientations result in the same value of tilt, and in the second view how the values change. Although the diagram looks similar, these histograms differ significantly from the yaw histograms shown in Fig. 2.

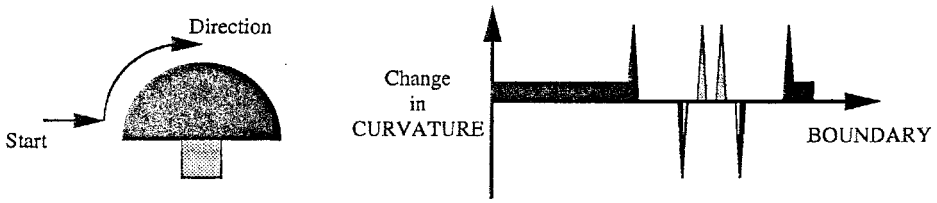


Fig. 6. Example boundary curvature signature. A view of a mushroom-like object is shown, along with it's boundary rate of change of a curvature signature.

2.2. Needle Diagrams

Needle Diagrams are iconic representations of object models from a given viewpoint with every point representing the object's local three-dimensional orientation (rather than the scene illuminance/reflectance).

2.3. Depth Maps

Depth Maps are also iconic representations of an object models from a given viewpoint, with every point in the array representing the distance of the object (as viewed at that point) from the viewpoint (i.e. the focal point of a camera).

2.4. Boundary Curvature Signatures

By following the silhouette of an object we may calculate a measure of its boundary's curvature. That measure may then be used to calculate the rate of change of curvature for all points on the boundary, providing a description of the boundary which is invariant to roll (e.g. see Fig. 6).

3. OBTAINING OBJECT MODELS

Known object models were defined, in terms of planar surfaces, using a simple Computer-Aided Design (CAD) specification technique. This was felt to be preferable to the alternative of building the models from multiple views, as it provided well-defined (noise-free) models which could be manipulated more efficiently. Examples of the models specified using this technique are shown in Fig. 7 and these models, in fact, comprise the database of *known* objects which were used during testing.

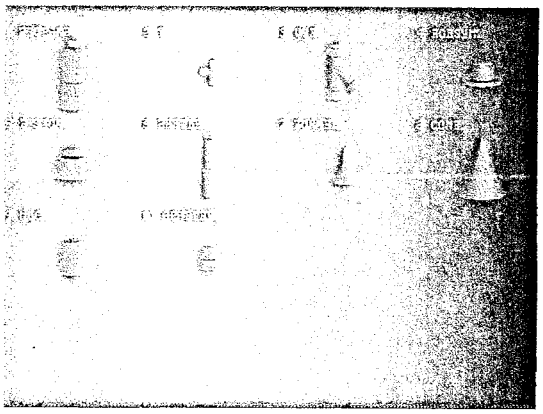


Fig. 7. CAD models.

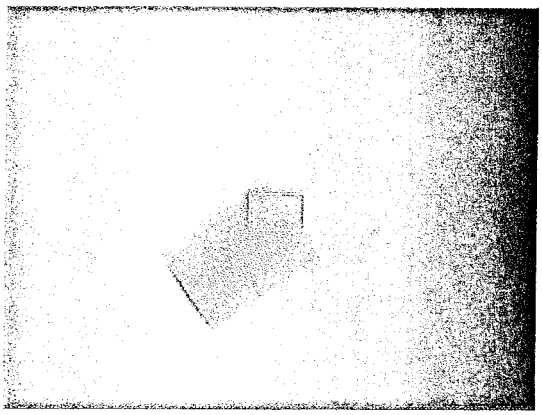


Fig. 8. A range image of a coffee cup in which brightness is inversely proportional to the depth (i.e. the closer a point is, the brighter it is). The rectangular region which is marked is used in the examples which follow, in order to show in a more detailed fashion, the effects of the various processing operations which are employed.

The views of objects to be recognised were supplied as range data, by the Pattern Recognition and Image Processing Lab. of Michigan State University. They were produced using a Technical Arts 100X scanner (commonly known as the 'White scanner') and scan conversion software which was originally developed by Paul Besl (which provides the depth information in terms of (x, y, z) coordinates). A sample range image is shown in Fig. 8.

Viewed models had to built from this range data, and this was done in three steps:

1. Surface interpolation.
2. Surface smoothing.
3. Model segmentation.

Each of these processes is described, in turn, in the sections which follow.

3.1. Surface Interpolation From Laser Range Data

Faugeras[15,16] and Henderson[17] describe a method of building surface representations from well-sampled point data (e.g. range data). "3-point seed" surface planes are generated using any combination of three points which are within the sampling distance of each other. These seed surfaces are subsequently merged into larger surfaces.

The method for object recognition detailed in this paper does not require that the surfaces be bounded by perceptually relevant edges (e.g. object boundaries, or lines of high curvature). Because of this, it was sufficient to represent the range data in terms of 3-point seed surfaces (e.g. see Fig. 9), overcoming the difficulties of representing objects which cannot be readily segmented into stable surface representations.

The use of very simply extracted surface primitives does not in any way detract from techniques for extracting larger scale surfaces — and in fact the recognition technique would work equally well with larger scale primitives.

3.2. Smoothing Surface Orientations

Direct interpolation between range points, however, must take into consideration two issues which are well known from the intensity image domain (see Ref. 21)

1. Signal noise: In other words fluctuations in the signal due to random effects.
2. Sensor accuracy: Any depth sensor will only produce results to a given resolution. If that (quantisation) resolution is small with respect to the sampling resolution of the sensor then surfaces which are interpolated between adjacent range points could easily be seriously affected (in terms of their orientation). This is not an unusual affect, and actually occurred in the range data which was used supplied. For example, consider Fig. 10, in which a surface interpolation of part of the coffee cup range image is shown. It is evident that there is a stepping effect in the range data of the curved surface of the coffee cup.

In order to overcome these problems it is necessary to smooth the signal in some way, and as it was intended to use the surface information from the model, the surfaces themselves were smoothed on the basis of their orientations.

A local smoothing filter $S_{w_{center},w_{adjacent}}(x, y, z)$ for the normals of the 3-point seed surfaces (where (x, y, z) represents the normal vector of any 3-point seed surface) was defined as follows, and is graphically shown in Fig. 11. w_{center} and $w_{adjacent}$ are weighting for the center/current surface and the adjacent surfaces, respectively.

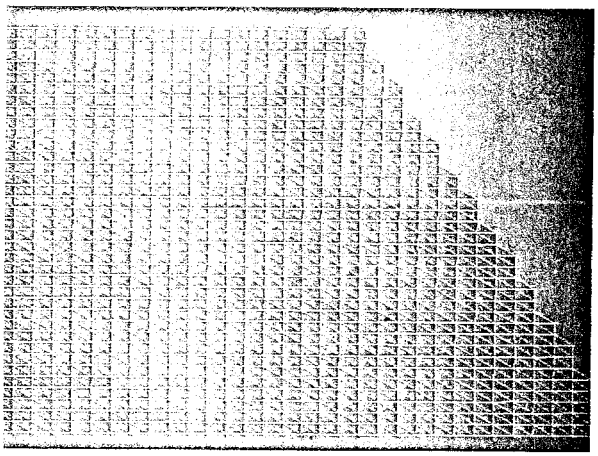


Fig. 9. The 3-point seed surface connections for the marked region of the previously shown range image. Each vertice represents a point from the range image.

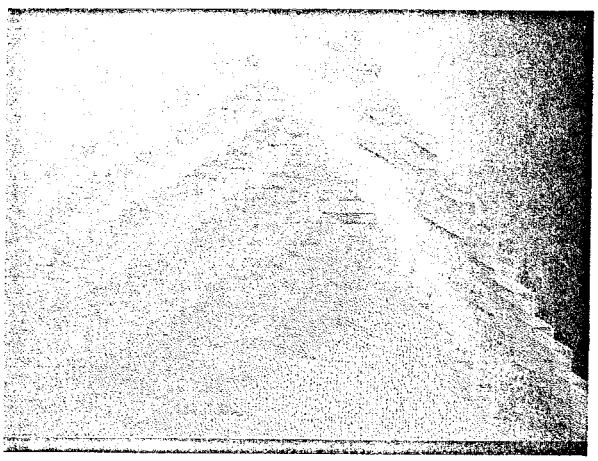


Fig. 10. Rendering of the 3-point seed surfaces where the grey-level is inversely proportional to the angle of tilt. Notice the step-like effect which is caused by the signal quantization.

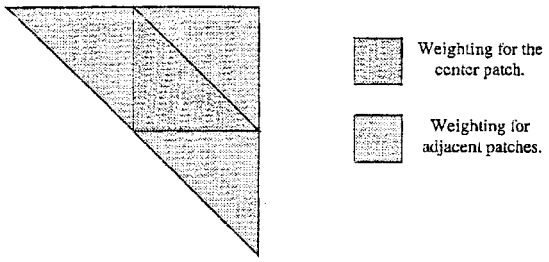


Fig. 11. A local filter for smoothing the orientations of 3-point seed surfaces.

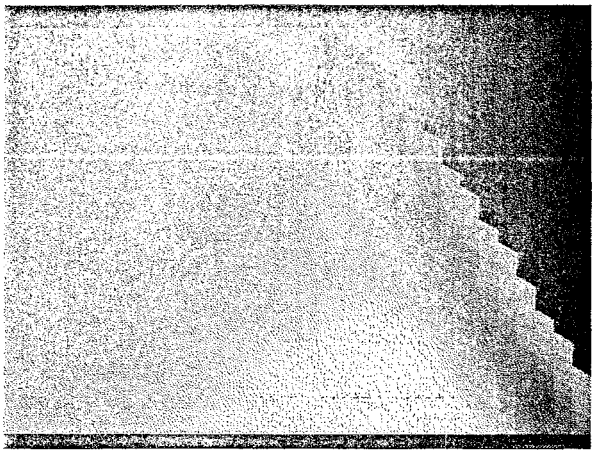


Fig. 12. Rendering of the 3-point seed surfaces after smoothing the model on the basis of the surface orientations.

Given a surface a with surface normal vector (x_a, y_a, z_a) and three adjacent surfaces b, c and d with surface normal vectors (x_b, y_b, z_b), (x_c, y_c, z_c) and (x_d, y_d, z_d), the smoothed normal vector for surface a $(x_{smooth}, y_{smooth}, z_{smooth})$ is defined as:

$$S_{w_{center}, w_{adjacent}}(x_a, y_a, z_a) = \left(\frac{x_{average}}{V_l}, \frac{y_{average}}{V_l}, \frac{z_{average}}{V_l} \right) \tag{1}$$

where

$$x_{average} = x_a.w_{center} + (x_a + x_b + x_c).w_{adjacent}$$
$$y_{average} = y_a.w_{center} + (y_a + y_b + y_c).w_{adjacent}$$
$$z_{average} = z_a.w_{center} + (z_a + z_b + z_c).w_{adjacent}$$

and

$$V_l = Average\ vector\ length = \sqrt{x_{average}{}^2 + y_{average}{}^2 + z_{average}{}^2} \tag{2}$$

In order to propagate the effects of this local filtering operation, the filter is applied to the surface model iteratively. For example the surface model shown

previously in Fig. 10 when smoothed with $S_{1,2}$ for 8 iterations results in the surface model rendered in Fig. 12. The values of w_{center} and $w_{adjacent}$ (i.e. 1 and 2, respectively) were chosen through experimentation.

3.3. Object Segmentation

The technique of implicit model matching, as detailed in this paper, requires that *viewed* models represent only single objects. Hence, if scenes with multiple objects are to be considered the scene model must be segmented into separate object models. Boundaries between viewed 3-D surfaces may be classified as *convex, concave* and *jump boundaries* (see Han *et al.*[18]) and these classifications may be used when segmenting surfaces into objects. In this research a very simple approach of segmenting only at *jump boundaries* has been employed (as these may be detected quite easily in data which has not been segmented into surfaces). For example in Fig. 13 this simple type of segmentation allows the *Wye* piping joint to be segmented from the other pipes, but does not allow the other two pipes to be segmented correctly.

More complex segmentation techniques such as those of Han *et al.* should be employed, although it should be noted that those techniques may be extended even further, by making use of characteristics of the surfaces (such as major axis in the case of curved surfaces) which are seperated by the boundaries. For example, the major axes of the cylindrical parts of the pestle (shown in Fig. 14) which are broken into separate regions by the simple segmentation technique used herein, are the same and hence the surfaces are more than likely to belong to the same object.

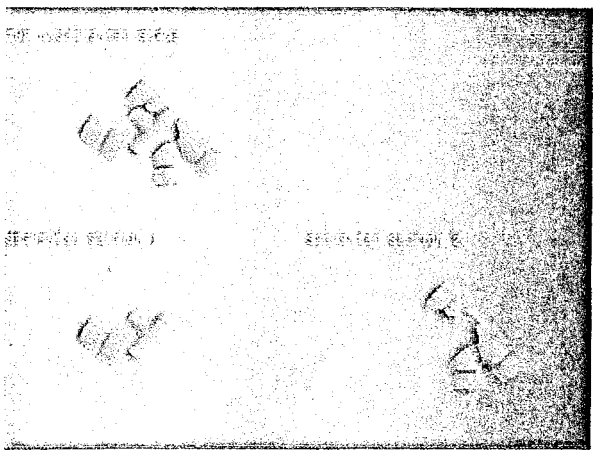


Fig. 13. Segmentation of objects in a complex scene.

4. MODEL INVOCATION

Model invocation is the selection of *known* models with which to attempt matching (of the *viewed* model). However, with the approach presented in this paper,

matching is performed between views of object models. Hence, rather than just invoke object models, it is necessary to invoke approximate views of models. (Note that only approximate views of the *known* object models need be invoked as the matching operations both calculate the model position and fine tune the model orientation of the invoked *known* model views). This inherently addresses model invocation (in the strict sense of the term) as, if no view of a particular model is invoked then the model itself is not invoked.

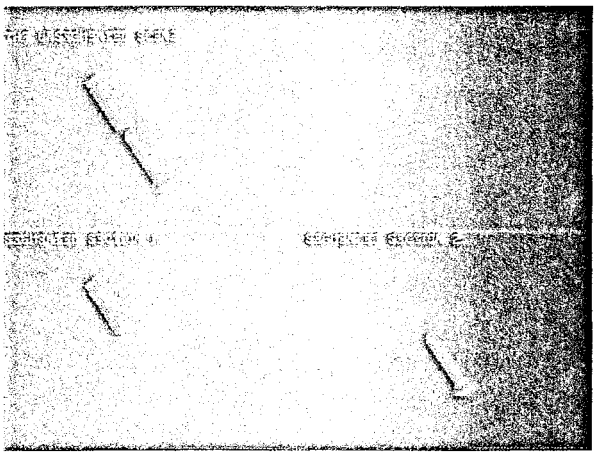


Fig. 14. Mistaken segmentation of a single object.

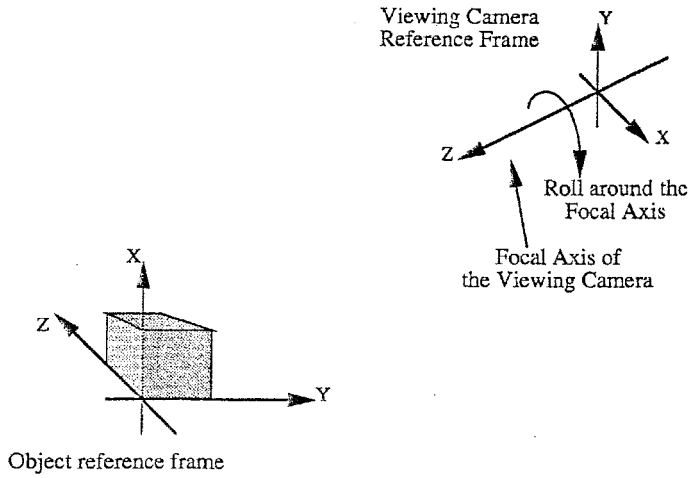


Fig. 15. Viewing angle geometry.

Model view invocation is performed, herein, by determining possible orientations from which each *known* model could be viewed (in order to result in a view similar to that of the *viewed* model). The focal axis of the viewing device/camera with

respect to the *known* model's frame of reference is first determined, and subsequently possible values for the roll of the camera with respect to it's own frame of reference (i.e. around the focal axis) are calculated (see Fig. 15).

4.1. Cameras

It has been stated that the task of model invocation is to determine orientations from which each *known* model could be viewed, so that the resultant view might be similar to that of the *viewed* model. This is an important point, as it is indicative of the fact that, rather than compute transformations which theoretically map *known* models to the same 3-D space as the *viewed* model, the task of model invocation and model matching in this method is to determine potential orientations from which to view the *known* models (i.e. a virtual camera model), with respect to the individual *known* models frames of reference (much as Goad does in Ref. 22).

The *viewed* model should have a camera model of some sort associated with it (i.e. that which represents the original viewing device). The camera model used with *known* models is similar, in that it uses the same internal parameters, although it is defined with respect to a different coordinate frame (i.e. that of the *known* object model).

This use of camera models does not cause any restrictions on the technique and it is necessary due to the use of iconic representations (i.e. depth maps and needle diagrams) during the matching process.

4.2. Determining Potential Orientations of the Focal Axis

In order to determine potential orientations of the camera with respect to a *known* model's frame of reference, a sample of all possible orientations is used. This sampling of orientation space is defined by the tesselations of a unit (Gaussian) sphere.

Using each of the sphere tesselations surface normals as possible orientations of the camera focal axis, directional histograms of the tilts visible from the *known* model are determined. These directional histograms (after being smoothed) are compared with a directional histogram of tilt determined from the *viewed* model, resulting in a degree-of-fit for each possible orientation.

A degree-of-fit is calculated using normalised cross correlation for each invocation and gives a measure of likelihood that the model at an orientation within the range defined by the relevant sphere tessalation is that which must be recognised. The orientations which possibly contain a match are defined by those tessalations of the sphere which contain local maximum values for the degree-of-fit of tilt (as they will change slowly from tessalation to tessalation, as long as the visible view of the model changes slowly between tessalations), and the likelihood is defined by the degree-of-fit.

An example of the comparison of tilts from each possible orientation is shown in Fig. 16. The *known* model is considered from all possible viewpoints as defined by each tesselation of the sphere, and each tesselation of the sphere is shown encoding the degree-of-fit determined through the comparison of tilt histograms.

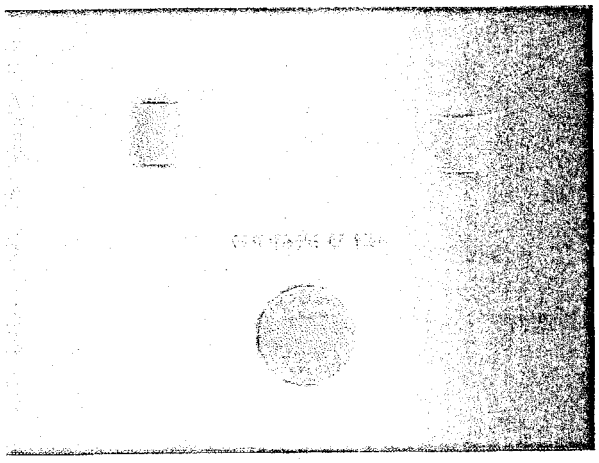


Fig. 16. The comparison of tilt histograms. The *known* model (top right) is considered from each possible viewpoint, as defined by the tesselated sphere — and the grey-level of each sphere tesselation shown is proportional to the degree-of-fit found between the tilt histogram determined from the *viewed* model (top left) and that determined from each view of the *known* model. Notice the band of tesselations around the sphere. These represent equivalent views of the *known* cylinder model which correspond to the *viewed* model of the cylinder.

4.3. Determining Possible Values of Roll

Having determined potential orientations for the camera focal axis, potential values for roll, around that focal axis, must be calculated. This is accomplished by comparing (using normalised cross correlation) the directional histogram of roll defined by the *viewed* model with that derived from the *known* model in an arbitrary roll, as viewed using the previously determined focal axis. The comparison is performed at all possible values of roll (where the resolution of the search for roll is defined by the resolution of the directional histograms used), of the theoretical model, simply rotating (i.e. shifting in a circular fashion) the roll histogram of the theoretical model in order to define each possible roll. Smoothing of the histograms before comparison again provides a smoothly changing degree-of-fit, and once again the local maxima define the potential matches. An example of the comparison of roll histograms is given in Fig. 17.

The result of this operation is the determination of viewing orientation frames of reference, defined with respect to *known* models' frames of reference, which may provide views of those *known* models, which are similar to the *viewed* model.

5. MODEL MATCHING STRATEGY

Model invocation, in this method, supplies potential orientation frames of reference for *known* models. These, however, are only approximations as the orientation space is quite coarsely sampled. One task of model matching is, then, to fine-tune

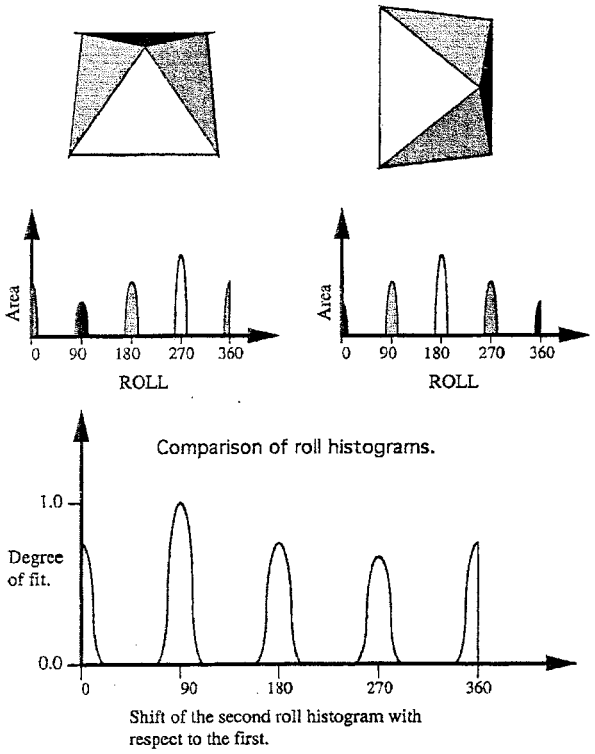


Fig. 17. Example: comparison of roll histograms for two views of an object, taken using the same focal axis but with the roll of the camera around that axis is altered.

these orientation frames and to determine the position of the viewing camera relative to the *known* model.

Model matching consists of a number of steps, during which the orientation frames will either be accepted (and their updated versions passed to the model verification stage) or rejected. Firstly, the approximate position of the object with respect to the virtual camera is calculated. This is done first, as only visible surfaces are considered in the next step, which is the fine-tuning of object orientation. Object position is then fine-tuned, followed by object depth and finally object position is re-tuned (as the tuning of object depth will affect the object position tuning), this time to sub-pixel accuracy. Each of these steps is discussed in turn.

5.1. Determining Approximate Object Position

In order to allow the position of the object to be fine-tuned (and in order to allow the object to be viewed during tuning operations in general), the position of the virtual camera (see Sec. 4.1) with respect to the *known* model must be estimated. To do this, the position of the *viewed* model with respect to its viewing camera may be employed (see Fig. 18). The imaged centroid of the *viewed* model, and an

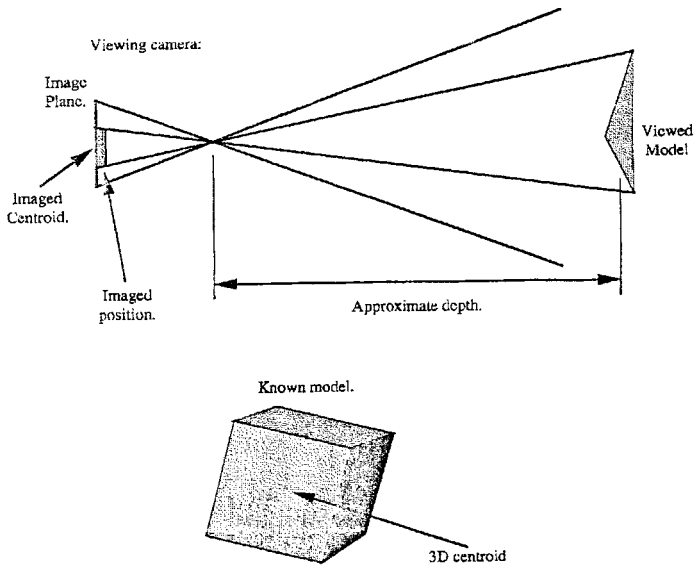


Fig. 18. The geometrical features used when determining approximate object position.

approximate measure of the distance of the *viewed* model from the viewing camera are both very easily computed. The position of the camera which views the *known* model is then approximated by placing the camera in a position relative to the *known* model's 3-D centroid, such that the centroid is at the correct approximate distance from the camera (as determined from the *viewed* model), and is imaged by the camera in the same position as the *viewed* model's imaged centroid.

5.2. Fine-Tuning Object Orientation

Fine-tuning object orientation is performed in a similar way to the approximate determination of object roll. Pitch, yaw and roll histograms derived from the theoretical model are compared to those derived from the *viewed* model. The differences between them indicate the amount by which the orientation may best be fine-tuned (e.g. see Fig. 19). This is done within a limited range of angles (the range being defined by the sampling of the sphere which was used for sampling orientation space during the model invocation stage), and at reasonably high angular resolutions (i.e. $\frac{1}{4}^{\circ}$), in order to allow tuning to be accurate.

Additionally it should be pointed out that the individual fine-tuning operations affect not only the current directional histogram (e.g. roll), but also, in a small way the other directional histograms (e.g. pitch, yaw). Hence the fine-tuning is performed by calculating a specific directional histogram, tuning the relevant component of orientation, and then moving on to the next directional histogram. This is done until the amount of tuning for all components of orientation falls below the required accuracy, or until the total tuning on any of the orientations exceeds the range allowed (in which case the invoked model view is rejected).

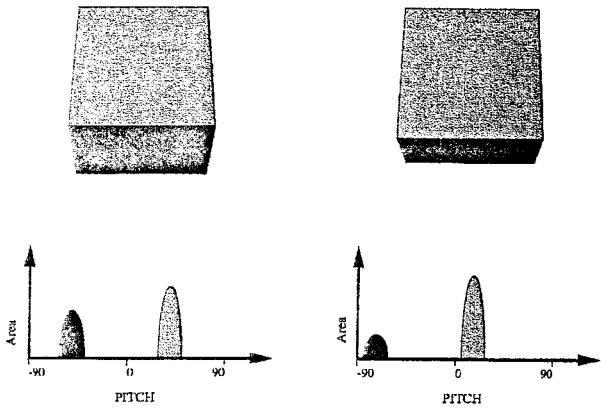


Fig. 19. An example of the fine-tuning of orientation that is required. The two views of the cube model shown have slightly different pitch values, and this is reflected in their directional histograms.

5.3. Fine-Tuning Viewed Object Position

Fine-tuning of the viewed object position may be formally defined as tuning of the relative position of the viewing camera with respect to the *known* models reference frame, by translation in the plane which is orthogonal to the focal axis of the camera (i.e. in the directions parallel to the image plane).

This operation is performed using a template matching technique in which each needle diagram of the *known* model is compared, using a normalised correlation mechanism, with a needle diagram of the *viewed* model. The position of the template which returns the highest correlation is taken to be the optimal position for the *known* model. The needle diagrams are rendered (as are the depth maps used in the next section) using a standard hidden surface removal algorithm.

5.3.1. Correlation of needle diagrams

The standard method of comparing iconic representations is normalised cross-correlation, but it is defined only with respect to scalars. For the comparison of needle diagrams 3-D vectors must be compared and that correlation NV for each possible position of the template (m, n) is defined as follows

$$NV(m,n) = \frac{\sum_i \sum_j f(viewed(i,j))(\pi - angle(viewed(i,j), known(i-m)(j-n)))}{\sum_i \sum_j f(viewed(i,j)) * \pi} \tag{3}$$

where $viewed(i,j)$ and $known(i,j)$ are the 3-D orientation vectors from the *viewed* and *known* needle diagrams respectively,

$$f(vector) = 1 \ (if \ the \ vector \ is \ defined)$$
$$= 0 \ (if \ the \ vector \ is \ NULL)$$

and

$$angle(vector1, vector2) = The\ angle\ between\ the\ two\ vectors. \quad (4)$$

In order to allow for possible occlusions of the *viewed* model, we consider only points in the *viewed* models needle diagram $viewed(i,j)$ which have object needles, using the function $f()$.

π is the maximum difference between any two 3-D vectors, and *angle*() is a function which returns the angle between the two vectors/needles passed, or 0 should one of the vectors be undefined. Hence for the comparison of any two vectors the top half of the fraction will be increased by a value between 0 and π and the bottom half of the fraction will be increased by π.

This definition results in a maximum value of 1 for $NV(m,n)$ if the needle diagrams are the same, and a minimum value of 0 if they are exact opposites.

5.3.2. Efficient template matching

In order to make this operation more efficient, the needle diagrams are first compared at lower resolutions. These comparisons at lower resolutions use only a simple comparison of the needle diagrams $S(m,n)$ which is a measure of the number of needles in the *viewed* models needle diagram which have a corresponding needle (of any orientation) in the *known* models needle diagram.

$$S(m,n) = \frac{\sum_i \sum_j f(viewed(i,j)) * f(known(i,j)}{\sum_i \sum_j f(viewed(i,j))} \quad (5)$$

where the definitions previously indicated apply.

Only positions of the template for which $S(m,n)$ is greater than 0.75 are considered at higher resolutions.

5.4. Fine-Tuning Object Depth

Fine-tuning the distance between the *known* model and its viewing camera is done by direct comparison of the depth maps generated from both the *viewed* model, and the *known* model (in its determined pose). The Depth Change Required (or DCR; note this is the optimal displacement of the model, and not an error term) is defined as follows:

$$DCR = \frac{\sum_i \sum_j f(viewed(i,j)) * f(known(i,j)) * (viewed(i,j) - known(i,j))}{\sum_i \sum_j f(viewed(i,j)) * f(known(i,j))} \quad (6)$$

where $viewed(i,j)$ and $known(i,j)$ are the depths from the *viewed* and *known* depth maps respectively, and

$$f(vector) = 1\ (if\ the\ depth\ is\ defined)$$
$$= 0\ (if\ the\ depth\ is\ unknown)$$

This depth change is directly applied as a translation to the pose of the camera which views the *known* model, in a direction defined by the focal axis of the camera.

Due to perspective effects this operation will have effects on the depth map rendered and so is applied iteratively until the DCR falls below an acceptable level (e.g. 1 mm along the Z axis).

5.5. Fine-Tuning Object Position to Sub-Pixel Accuracy

The final section of fine-tuning is again of the viewed object position. This is for two reasons: Firstly due to possible effects of tuning object depth, and secondly in order to tune the position to an extremely high degree of accuracy.

This sub-pixel tuning of the viewed object position is accomplished using a combination of template matching, normalised correlation and quadratic modelling techniques.

The best position may be determined to pixel accuracy using the template matching and correlation technique as previously described in Sec. 5.3. In order to determine the position to sub-pixel accuracy the normalised correlations around the best position (as determined to pixel accuracy) are used and are modelled as quadratics in two orthognal directions (i.e. parallel to the two image axes). See Ref. 12 for further details.

5.6. Dealing with Only One Visible Surface

The determination of the roll of the viewing camera with respect to a *known* model inherently assumes the visibility of surfaces of more than a single orientation. If only one orientation is present from the visible surfaces there is an ambiguity with respect to rolling around the orientation vector of the visible surfaces. This problem, however, is simpler than the situation where surfaces of multiple orientations are visible, as it reduces to a 2-D shape recognition task.

5.6.1. Detecting that only one orientation is visible

Detecting whether surfaces with multiple or single orientations are visible may be done by considering the Extended Gaussian Image[8,9] of the *viewed* model. The distribution of orientations may be calculated as a standard deviation and if that deviation is greater than a small angle (e.g. 11°) then multiple surfaces must be regarded as being present. See Fig. 20.

5.6.2. Determining surface 'roll' for single orientations

Somewhat surprisingly, the only parts of the algorithm (being presented here) for recognising objects which are affected by this special case are the determination of possible rolls, and the fine-tuning of those possible rolls.

In this situation, where surface orientation information can no longer be of assistance, possible object roll may be determined (and fine-tuned) by using the rate of change of boundary curvature. This rate of change of curvature is independent of the roll, and may be calculated quite easily if an approximation to the camera position is known. The approximation to camera position can be calculated quite

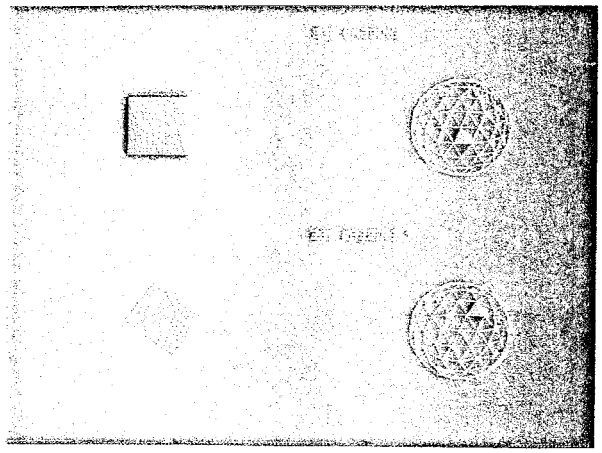


Fig. 20. Two views of a cube, and EGIs with the surface orientations of each view mapped to them. The distribution of the surface orientations mapped to EGI indicate whether the view contains a single main orientation or multiple orientations.

easily (as before), but prior to instead of after the roll calculation. The object may then be drawn, and its boundary traced, deriving a measure of its curvature and, after smoothing the curvature to remove noise, the rate of change of the curvature may be calculated.

The signature of the rate of change of curvature of known models may then be compared to one calculated from the *viewed* model, in the same way as the directional histograms of roll were compared, in order to determine possible rolls (e.g. see Fig. 21). In this situation however, the possible rolls determined are not in terms of degrees, but rather are in terms of boundary features, so care must be taken when calculating the amount of roll. It is also important to realise that the boundaries will most probably be of different sizes and so must be scaled to a fixed size of the boundary curvature signature before comparison.

6. MODEL VERIFICATION

The normalised correlation comparison of needle diagrams, between the *viewed* model and *known* models in determined poses, as used when fine-tuning object position (see Sec. 5.3), gives a degree-of-fit which represents all aspects of object position and orientation. This degree-of-fit, then, may be regarded as an overall degree-of-fit between the two models (in the given poses).

The pose of the *known* model associated with the best degree-of-fit, given that the degree-of-fit is over a high threshold is taken to represent the *viewed* model. Search for this best degree-of-fit is continued until all invoked possibilities are considered.

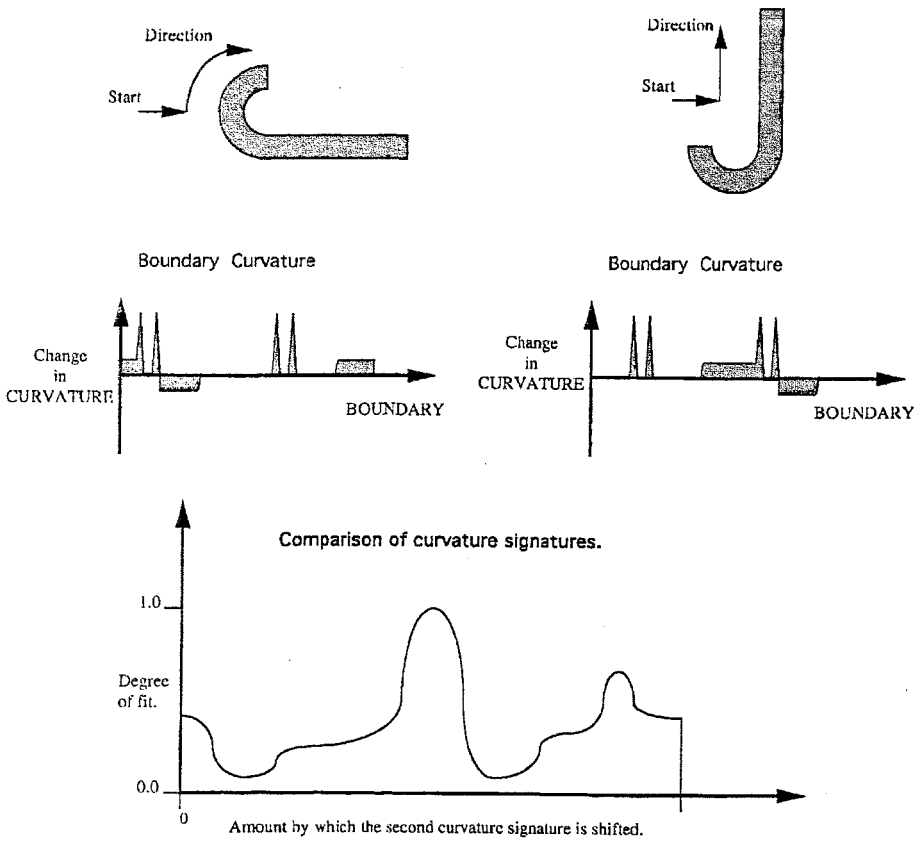


Fig. 21. Example: comparison of boundary curvature.

7. TESTING

In this section the capabilities of the approach are demonstrated on models derived from range images of single objects and then the issues involved with extending the technique to scenes containing multiple objects are addressed.

Testing through experimentation proves only the applicability of the method in the instances which are tested. It also suggests the aptness of the method in other similar circumstances, although the validity of the implication may often be related to the degree to which test instances are selected due to their particular suitability. The instances presented in this chapter were taken from a database of range images which was created at the Pattern Recognition and Image Processing Lab. of Michigan State University (MSU). This database consisted of range images of objects which were selected by those at MSU, and hence the objects were in no way chosen because of their suitability.

Unfortunately, for the objects for which range images were supplied, no CAD data was available (and the physical objects were in MSU). Hence models of the

objects present in the range images had to be roughly approximated on the basis of the range data supplied. This was done visually, by rendering an image of each of the objects from the range data, and then calculating the approximate shape and size of the objects. The range images were labelled by MSU and this facilitated the task of estimating shape.

This method of deriving the CAD models results in the models being only approximations to the actual objects, and this in turn makes the task of recognition more difficult. The database of models which was developed is this way is shown in Fig. 7.

7.1. Recognition of Objects in Scenes with One Object

The technique of Implicit Model Matching presented in this paper is intended for the recognition of single objects (whether they be presented separately or whether they have been segmented from all other objects in a complex scene). The main body of testing of this approach is, then, with single objects. In Figs. 22–24 an example of recognition of a piston ring-like object is shown in a number of steps. The database of *known* objects was comprised of the ten objects which are shown in Fig. 7.

Figures 25 through to 33 then show examples of the best matches determined for various views of different objects, and each of these figures 25–33 contain the following: In the *top left quadrant*, a rendering of the needle diagram of the *viewed* model in which the grey-level is proportional to tilt. In the *top right quadrant*, Tilt, Roll, Pitch and Yaw Directional histograms which are derived from the *viewed* model. In the *bottom left quadrant*, a rendering of the needle diagram of the *known*

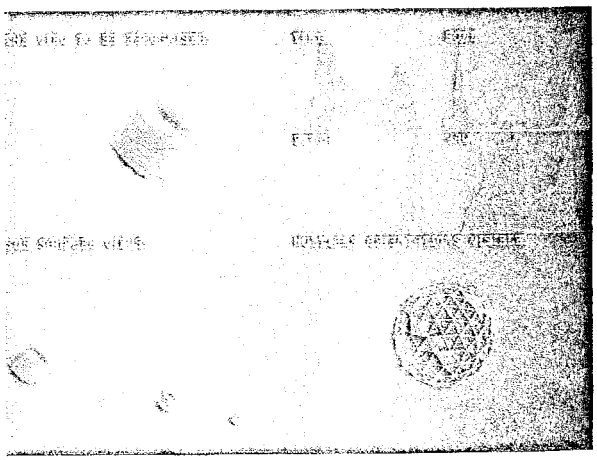


Fig. 22. A range image of a piston rendered using tilt, along with various directional histograms, sub-sampled views (which are used when fine-tuning object position) and an EGI, all of which were derived from the view shown.

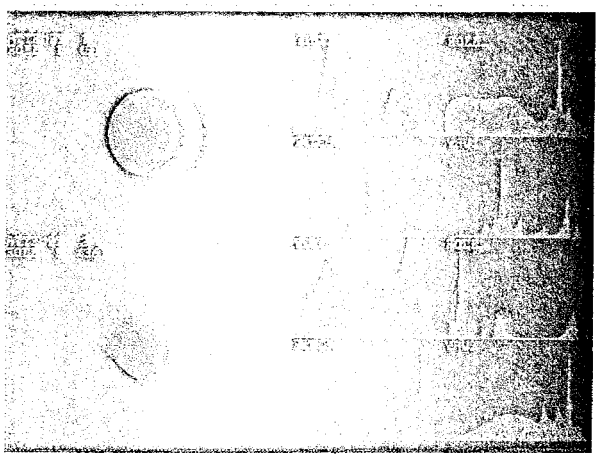


Fig. 23. Two orientations of a CAD model of a piston which might represent the viewed model shown in the previous figure. These were determined solely on the basis of tilt and roll directional histograms.

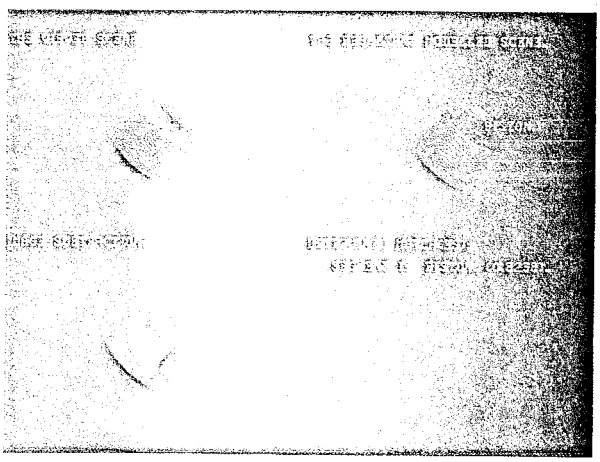


Fig. 24. The recognised piston, together with the subtraction of the two needle diagrams and the degree of fit which was associated with the match.

model in the determined pose which best matched the *viewed* model, together with the degree-of-fit determined and finally in the *bottom right quadrant*, a subtraction of the rendered needle diagrams of the *viewed* model and the best matching *known* model in its determined pose, is shown.

The best measure-of-fit for each *known* model when compared with each *viewed* model are listed in Fig. 35.

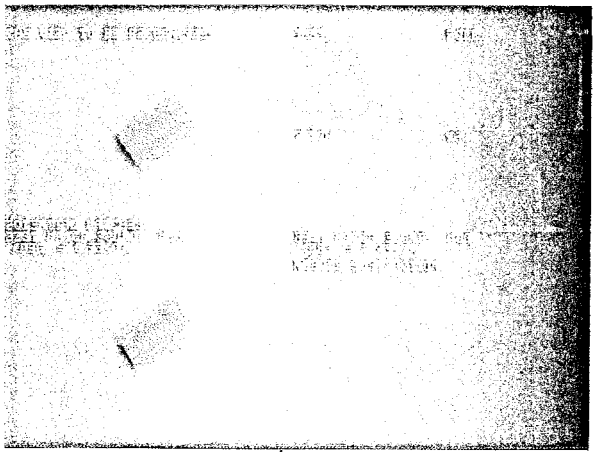


Fig. 25. Recognition of a coffee cup.

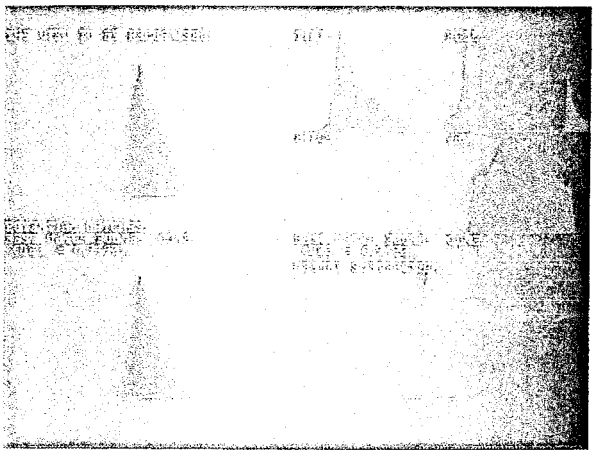


Fig. 26. Recognition of a cone.

7.2. Recognition of Correctly Segmented Objects

The segmentation technique employed is detailed in Sec. 3.3. Consider the *Wye* piping junction which is segmented correctly (as "segmented Sec. 2") in Fig. 13. When the recognition strategy is applied to that section, the resultant best match is as shown in Fig. 34.

We note, however, that the orientation of the pipe is slightly incorrect. Consider the subtraction of needle diagrams shown in the bottom right quadrant. The section pipe which points downwards may be observed to be at a slightly incorrect orientation. The reason for this due to the loss of information from the occluded section of the pipe, and there are two ways in which this might be overcome:

1. *Object model reconstruction.* It might be possible to reconstruct the occluded surfaces on the basis of the surfaces which are visible, using a technique such as that of Fisher[6,23]
2. *Considering complex models in terms of sub-parts.* It could also be possible to consider complex objects such as a *Wye* piping joint as comprising a number of sub-parts (e.g. in this case two sections of a pipe). This would significantly reduce the amount of information encoded in the directional histograms, and should result in directional histograms for each sub-part which are "clearer" (i.e. easier to match with directional histograms from *known* objects).

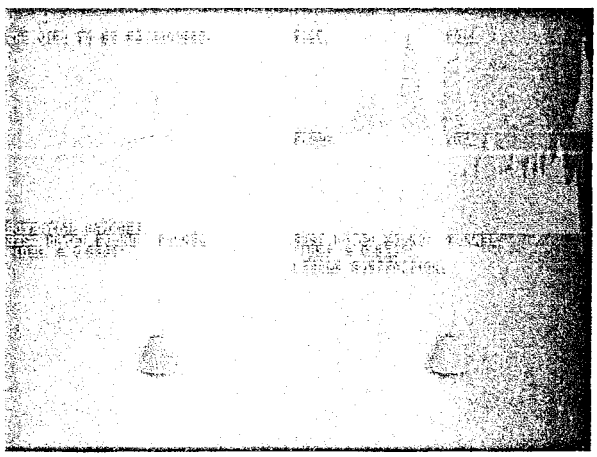


Fig. 27. Recognition of a funnel.

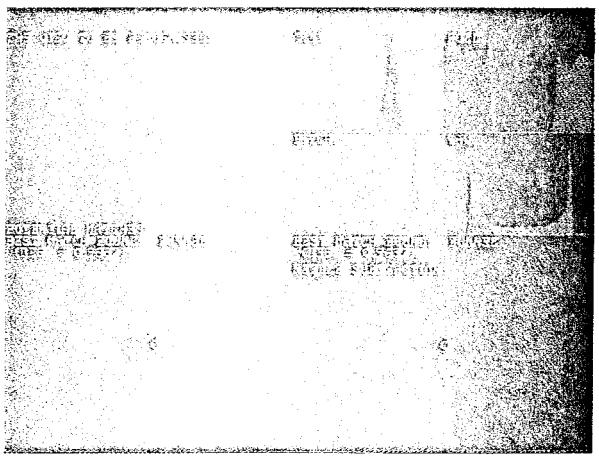


Fig. 28. Recognition of another view of the funnel. Notice the serious effect the missing parts problem has caused with respect to the amount of the viewed model which is actually presented.

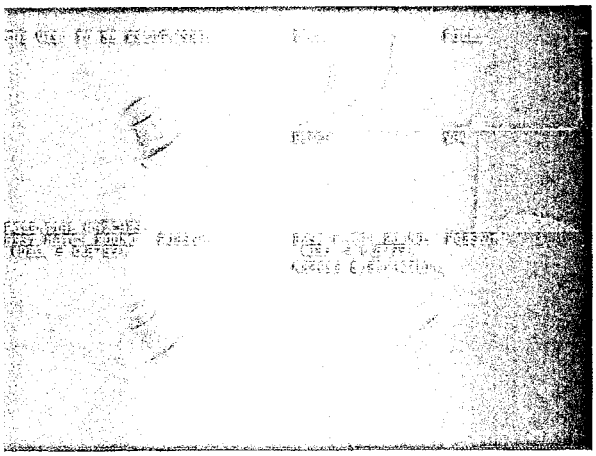


Fig. 29. Recognition of an industrial part.

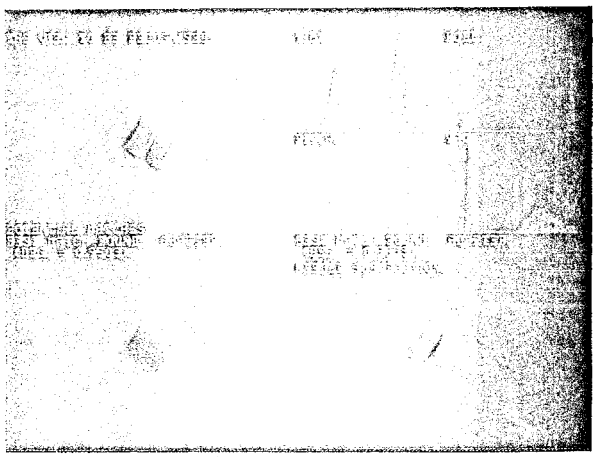


Fig. 30. Recognition of an adapter

8. CONCLUSIONS

The examples demonstrate the ability of the object recognition strategy developed in this paper, to identify the correct model and its pose in scenes which can be segmented so that only single objects are considered.

Further testing of the algorithm was performed employing 3-D models derived using a depth from camera motion algorithm (i.e. a passive technique for the determination of depth at significant intensity discontinuities in images followed by a simple planar interpolation).[13] The models determined ranged quite significantly in terms of the amount of noise present, and the object recognition technique was found to degrade quite gracefully as the noise increased.

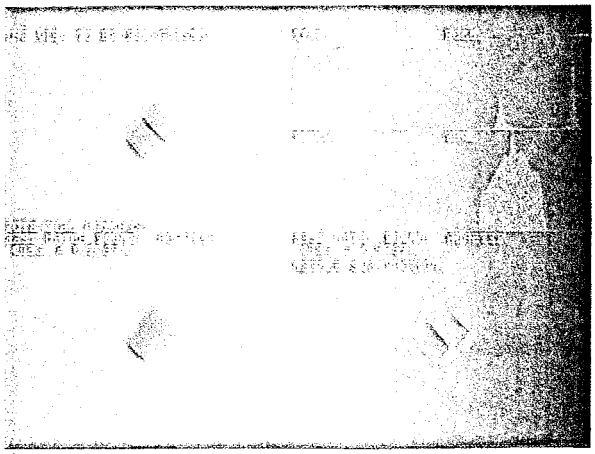


Fig. 31. Recognition of another view of the adapter. This view is being considered as a single object, although it actually breaks into two segments if the segmentation technique is applied to it.

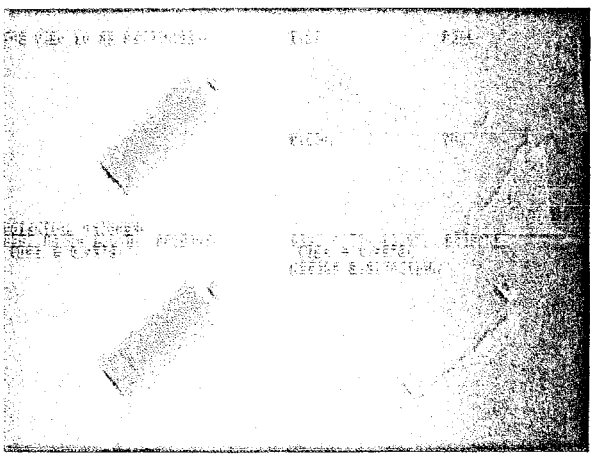


Fig. 32. Recognition of a cylinder for holding propane.

The time for the algorithm to be executed on a single T800 transputer was dependent on the complexity of the models used (much of the time being spent rendering needle diagrams and depth maps) and, in the experiments detailed in this report, ranged from 30 minutes up to 3 hours. However, recognition time was not even considered when developing the implementation, and it is clear (to the authors) that a significantly more efficient implementation is possible.

Recognised instances of several different objects were shown in this paper. However, it is important to also consider the discriminatory ability of the approach (i.e. how well each of the other *known* models matches a given *viewed* model). In

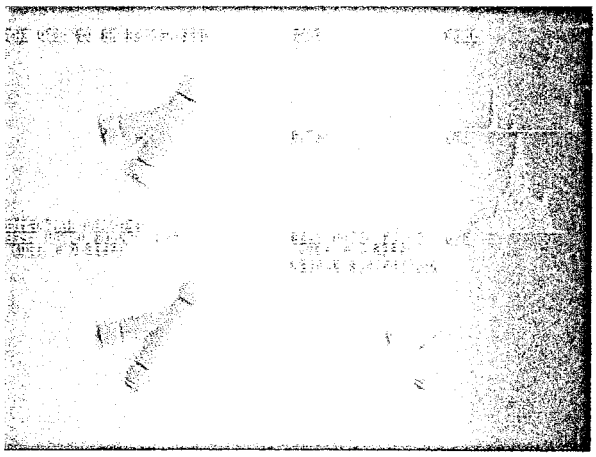


Fig. 33. Recognition of a *Wye* piping joint.

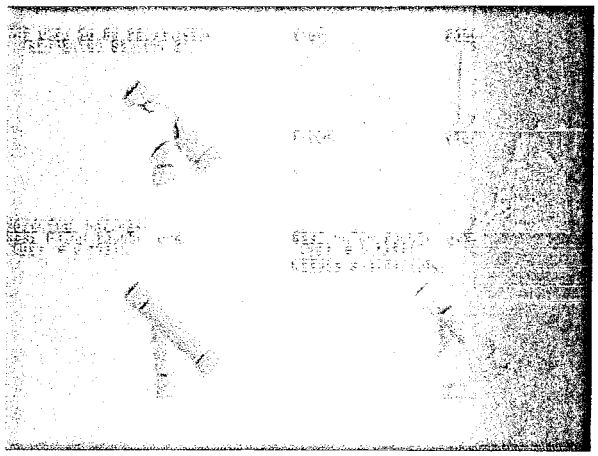


Fig. 34. Recognition of a *Wye* piping joint, which was segmented from scene containing multiple objects. The information shown is of the same type as that shown in Figs. 25 to 33.

order to do this a table of the best degree-of-fit between each *known* model and each *viewed* model was built up and is shown in Fig. 35.

Although each *viewed* model is matched with the correct *known* model, it is interesting to note that several other of the *known* models result in reasonably high measures-of-fit. Consider, for example, the 0.8805 degree-of-fit between the "propane" known model and the view of the coffee mug. Although the "propane" model is significantly larger than the view of the coffee mug a high degree-of-fit is determined due to a part of the "propane" model (i.e. part of the cylinder) being similar to the part of the coffee mug observed (as both objects are basically cylindrical). This is also due to the definition of the correlation measure between

Scene	Figure	Known models and best associated degrees-of-fit when compared with the scene model.										Recognised model
		Piston	Mug	Cone	Funnel	Robsym	Adapter	Propane	Wye	Pestle	T	
Piston	24	0.8238	0.5836	0.6246	0.2692	0.4934	0.4565	0.7230	0.4357	0.1752	0.1630	Piston √
Coffee mug	25	0.5719	0.9340	0.8087	0.3009	0.5514	0.5046	0.8805	0.5470	0.1754	0.1975	Mug √
Cone	26	0.6040	0.6466	0.9492	0.4749	0.3870	0.2827	0.8220	0.5953	0.2840	0.1503	Cone √
Funnel	27	0.6772	0.5785	0.7149	0.8864	0.6190	0.5551	0.6275	0.7640	0.6980	0.3809	Funnel √
Funnel	28	0.7962	0.9340	0.8264	0.9637	0.7599	0.7898	0.8260	0.7772	0.3030	0.3746	Funnel √
Robsym	29	0.5825	0.5877	0.5413	0.3351	0.8759	0.5464	0.6077	0.5071	0.2157	0.2077	Robsym √
Adapter	30	0.8185	0.8592	0.7917	0.5981	0.7798	0.9503	0.8973	0.5961	0.2596	0.2589	Adapter √
Adapter	31	0.6855	0.8813	0.8170	0.6940	0.6905	0.9069	0.8073	0.6831	0.2859	0.3561	Adapter √
Propane cylinder	32	0.4509	0.5019	0.5932	0.1912	0.3549	0.2389	0.9378	0.4251	0.1788	0.0979	Propane √
Wye piping joint	33	0.5102	0.3992	0.5639	0.1894	0.4040	0.2620	0.5085	0.8339	0.1581	0.1506	Wye √
Wye piping joint	34	0.5527	0.3196	0.5579	0.1867	0.3387	0.1830	0.5811	0.7634	0.0792	0.0291	Wye √

Fig. 35. Table of results, of the best degree-of-fit determined for each *known* model when compared with each *viewed* model.

needle diagrams (which takes no account of the regions of the *known* model which do not correspond to any part of the *viewed* model).

In fact, several of the ten objects used in these experiments are quite similar (i.e. are cylindrical), and hence by recognising these objects the testing provides strong support for the approaches' ability to discriminate between similar objects.

Additionally, considering the recognised *Wye* piping joint in Fig. 34 it is evident that the ability of the approach to function when data is missing from a *viewed* object (either by occlusion or by incorrect segmentation) is related to whether or not the remaining data characterises the object sufficiently well. This seems a reasonable constraint, as the ability to recognise is thence directly related to the amount (i.e. relative to being completely visible), and quality, of the *viewed* data.

Finally it must be emphasized that the object recognition technique presented compares 3-D models only on the basis of comparisons of representations which are synthesized/derived from the 3-D models. Some of these representations, in particular needle diagrams and depth maps, are based on the original image frame of reference, and hence bare an obvious correspondence with the original scene data, although they are generated by considering 3-D surface-based models. It is this interplay, using secondary representations, between the *viewed* data and the *known* CAD models which provides the technique with much of its function and robustness.

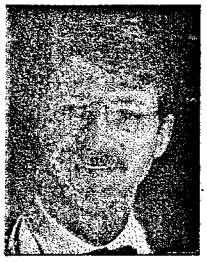
REFERENCES

1. P. J. Besl and R. C. Jain, "Three-dimensional object recognition", *ACM Comput. Surv.* **17**, 1 (1985) 75–145.
2. R. T. Chin and C. R. Dyer, "Model based recognition in robot vision", *ACM Comput. Surv.* **18**, 1 (1986) 67–108.
3. J. P. Brady, N. Namdhakumar, and J. K. Aggarwal, "Recent progress in object recognition from range data", *Image and Vision Comput.* **7**, 4 (1989) 295–307.
4. P. Suetens, P. Fua, and A. J. Hanson, "Computational strategies for object recognition", *ACM Comput. Surv.* **24**, 1 (1992) 5–61.
5. O. D. Faugeras, M. Herbert, and M. Pauchon, "Segmentation of range data into planar and quadratic patches", *Proc. Conf. on Computer Vision and Pattern Recognition (CVPR '83)*, Washington DC, 1983, pp. 8–13.
6. R. B. Fisher, *From Surfaces to Objects*, John Wiley, Chicester, 1989.
7. K. Ikeuchi, "Generating an interpretation tree from a CAD model for 3D-object recognition in bin-picking tasks", *Int. J. Computer Vision* **1**, 2 (1987) 145–165.
8. B. K. P. Horn and K. Ikeuchi, "Picking parts out of a bin", MIT Artificial Intelligence Laboratory, Memo No. 746, October 1983.
9. B. K. P. Horn, "Extended Gaussian Images", *Proc. IEEE* , **72**, 12 (1984) 1671–1686.
10. P. J. Flynn and A. K. Jain, "Surface classification: Hypothesis testing and parameter estimation", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition '88*, Ann Arbor, 1988, pp. 261–267.
11. P. J. Flynn and A. K. Jain, "BONSAI: 3-D object recognition using constrained search", *IEEE Trans. Pattern Analysis and Machine Intelligence* **13**, 10 (1991) 1066–1175.

12. K. Dawson, "Three-dimensional object recognition through implicit model matching", Ph.D. thesis, Department of Computer Science, University of Dublin, Trinity College, Dublin, Ireland, May 1991.
13. K. Dawson and D. Vernon, "3-D Object recognition using passively sensed range data", *Proc. ECCV '92* , G. Sandini ed., LNCS-Series vol. 588, Springer-Verlag 1992, pp. 806–814.
14. R. Krishnapuram and D. Casesent, "Determination of three-dimensional object location and orientation from range images", *IEEE Trans. Pattern Anal. and Machine Intell.* **11**, 11 (1989) 1140–1157.
15. O. D. Faugeras, "Conversion algorithms between 3D shape representations", *Fundamentals in Computer Vision*, ed. O. D. Faugeras, Cambridge University Press, 1983, pp. 305–314.
16. O. D. Faugeras, and M. Herbert, "The representation, recognition and locating of 3-D objects", *Int. J. Robotics Res.* **5**, 3 (1986) 27–52.
17. T. C. Henderson, "Efficient 3-D object representations for industrial vision systems", *IEEE Trans. Pattern Anal. Machine Intell.* **PAMI5**, 6 (1983) 609–618.
18. J. H. Han and R. A. Volz, "Region grouping from a range image", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition '88*, Ann Arbor, 1988, pp. 241–248.
19. L. G. Roberts, "Machine perception of three-dimensional solids", *Optical and Electro-Optical Information Processing*, ed. J. T. Teppett *et al.*, MIT Press, Cambridge, MA, 1965, pp. 159–197.
20. A. P. Reeves and R. W. Taylor, "Identification of three-dimensional objects using range information", *IEEE Trans. Pattern Analysis Machine Intell.* **PAMI-11**, 4 (1989) 403–410.
21. P. J. Besl, "Geometric signal processing", *Analysis and Interpretation of Range Images*, eds. R. C. Jain and A. K. Jain (Springer-Verlag, New York, 1990), pp. 141–205.
22. C. Goad, "Special purpose automatic programming for 3D model-based vision", *Proc. Image Understanding Workshop*, Arlington, VA, DARPA, Science Applications, McLean, VA, June 1983, pp. 94–104.
23. R. B. Fisher, "Using surfaces to recognise partially obscured objects", *Proc. 8th Int. Joint Conf. on Pattern Recognition*, Karlsruhe, West Germany, Aug. 1983, pp. 989–995.

Received 9 December 1992; revised 12 January 1994.

Kenneth M. Dawson-Howe received his Bachelors degree in computer science from the University of Dublin, Trinity College in 1987. He then went to work for Fuji Bank in Tokyo as a system analyst, and returned at the end of 1988 to the University of Dublin to take up a research position in the area of computer vision. He was awarded his Doctorate in computer science in 1991, and then joined the Department of Computer Science as a lecturer. His research areas include object recognition, camera calibration, analysis of security videos, and mobile robotics. He is currently actively working in image compression and automatic diagnosis in medical images.



David Vernon began his professional career as a software engineer with Westinghouse Electric Inc. where he worked from 1979 to 1981. He left Westinghouse to pursue post-graduate research and completed his Ph.D. at Trinity College Dublin in 1985. In 1983, he was appointed as a Lecturer in the Department of Computer Science in Trinity College. From 1991 to 1993, he worked as Scientific officer in the European Commission and from 1994 to 1995 he led a wide ranging initiative in Trinity College on the development of a comprehensive policy on information systems.

He is the author and editor of four books on computer vision: *Machine Vision; Parallel Computer Vision; Computer Vision: Craft, Engineering, and Science; and Data Fusion Applications.* He has published over forty papers in the fields of computer vision and computer science.

Dr Vernon is a Chartered Engineer and a Fellow of Trinity College, Dublin Ireland.