Quadrics-based matching technique for 3D object recognition

M Hanmandlu*, C Rangaiah* and K K Biswas†

We propose a new method for the recognition of objects from range data by matching the features of the observed surfaces in the scene with the features of the model surfaces. The surfaces are represented as quadrics in which planes are treated as a special case. The parameters of quadrics are converted into features which include Euler parameters accounting for the orientation of the surface. The matching process consists of finding a transformation of the model surfaces and devising a measure of consistency, such that the mean square error between the feature space is a minimum. The matching involves a tree search procedure with backtracking whenever error exceeds the limit.

Keywords: object recognition, feature matching, tree search, backtracking

In the context of robot vision, we are interested in methods which provide quantitative information with regard to recognition and location of both curved and planar objects. Model-based 3D object recognition methods have been used successfully in robot vision. These methods can be categorized into two main classes, depending on whether they use intensity data or range data. The range data related methods have an edge over the others in the sense that they are robust to occlusion and viewpoint dependency and follow simple rules of transformation.

We start with some important contributions related to object recognition using intensity data:

In the Hough feature-based method, an analysis of cluster patterns in Hough space with respect to collinearities yields Hough nets. Based on Hough nets, attributed graph representations of polyhedral objects and 3D wireframe models can be constructed. Image and model graphs are compared to identify objects in the scene.

Majumdar et al. uses a CAD modeller to generate characteristic parameters corresponding to the stable positions of the objects. The unknown stable position of an object is determined by comparing its characteristic parameters with those which are stored.

The method of Chien and Agarwal recognizes 3D objects from a given silhouette. In this, the recognition task is performed by matching 3D features of models with the 2D feature points of an unknown object.

There has been a growing trend towards object recognition using range data in recent years. Some of these techniques are briefly discussed below.

Faugeras and Hebert considered geometrical primitives mainly for planes to match the scene primitives with those of the models. The problems with this technique are the undirected matching and restricted use of planar patches only.

Bolles et al. used scene edge and surface data to predict circular arcs and hence cylindrical features for relating them to models. The key limitation with their method is that only large planar patches could be detected, and recognition and positioning were attempted by cashing in only those features. Oshima and Shirai also used stripe data for object recognition. Their segmentation approach is based on region growing, and matching is restricted to an algorithmically selected kernel which satisfies certain region properties, and which has some relationships with other regions. The kernel is matched against each scene learned to obtain a good match. Selection of a suitable kernel is not all that easy in some situations.

Grimson and Perez make use of sparse tactile sensor data on 3D positions and surface normals to identify and locate from known objects modelled as polyhedra. The difficulty with tactile sensing is that it might disturb the various objects in the process of making measurements.

The TINA vision system of Sheffield University AI & Vision Research is a stereo-based 3D object recognition and location system. The 3D features (viz. straight lines and circular arcs) were matched to the 3D wireframe models using a local feature focus technique.
Working at the pixel level entails the normal problems associated with stereo vision.

In the range-based method of Fan et al.11 an attributed graph is created with nodes representing the surface patches and arcs representing adjacency. Object models are represented as multiple graphs for objects as seen in topologically distinct viewpoints. Thus, one needs to know multiple object models corresponding to topologically distinct viewpoints.

Fisher17 uses 3D surface data in the form of curvature class and frames rules for aggregating surface patches into surface clusters corresponding to distinct objects. The model formulation incorporates image property and relationship evidence from curvature class and structural associations, which were then used in matching. The drawback with this method is that it only uses curvature classes for all of its recognition and location.

In the line of range-based methods, we propose a new model-based vision system for curved object recognition and location following the framework of Faugeras and Hebert7. The present work differs from other work1,2,3,4 in both the representation and the matching process. Instead of just using planar patches we use quadric as primitives for representing segmented scene surfaces. The primitives of the scene are then matched with similar primitives of the object in the model library by converting all the primitives into features which consist of Euler parameters, eigenvalues and translation parameters. We also include planar surfaces as a special case of quadrics. The matching involves determination of a transformation which rotates and translates the model features to be compared with the scene features. As shown below, our segmentation deals with the delineation of surface boundaries, and matching requires only quadric parameters.

As compared to curvature class representation12, the quadrics representation is less sensitive to noise13 and encompasses a wider class of surfaces, but requires a good CAD modeller in the proposed model-based recognition.

The application of quadrics is deemed to be an alternative to superquadrics for the representation of range data, with the former leading to surface modelling and the latter to volumetric modelling by way of part-based models15,16. Representation by superquadrics requires perfect segmentation, with a view to collecting data on all surfaces belonging to primitive parts for constructing models. There is an ambiguity associated with these models, in the sense that the same shape can be described by different sets of parameters. However, we do not need any matching process in this case, as an object here is recognized by way of identifying its constituent primitive models. Modelling becomes difficult if the parts have cavities in them, or are occluded. These problems can be overcome by employing quadrics with accruing computational simplicity in parameter estimation.

SEGMENTATION

Range data provides an important source of 3D shape information. A scene which may contain several surfaces needs to be segmented for the purpose of generating the scene primitives. Image segmentation techniques fall into two main classes: edge detection-based and region growing or clustering-based. The kind of segmentation approach adopted generally paves the way for a particular matching technique. Since we are treating all the surfaces as quadrics, segmentation by edge detection is found to be appropriate. The region growing method is fraught with the problem of identifying the type of surfaces, as done in Faugeras and Hebert3 for making a distinction between planes and quadrics.

There are a number of techniques available in the literature for edge-based segmentation of range data17-19. We briefly mention the technique followed by Wani and Biswas19.

Though the stripe data makes segmentation somewhat simpler7, our starting point here is with the range data which is first sliced to create equidensity contours (EDC) at uniform depth ranges. At each depth value, zi (i = 1, 2, ..., m, m being the number of slices), the range data is clustered in groups of z + Δz, where Δz takes on values 1, 2, 3, 4. The resulting slices have different pixel densities. The sliced data is processed to retain only the pixels along the outer rim of the slice, yielding a single pixel thick EDC for each of the selected range depths. The pixel data for each range is maintained in separate linked lists.

The next step involves the determination of the critical points which are of three types corresponding to three types of edges:

Type-1 critical points: these are the end points or terminating points of various EDCs. At least two neighbouring pixels of such points would belong to the background, thereby indicating the presence of step edges. These may also belong to occluding edges which separate the object from the background.

Type-2 critical points: these are the points where the EDCs exhibit a significant and sudden change in direction, and indicate roof edges where two visible object surfaces meet at an angle.

Type-3 critical points: these are also the end points of the contours, with the difference that all their neighbouring pixels belong to the object, and imply the presence of discontinuities in the EDCs, as well as suggest the presence of semi-step edges. Moreover, they form part of the edges where two surfaces appear to meet, one surface occluding the other.

One only has to check for the presence of background pixels to distinguish between Type-1 and Type-3 critical points, whereas a gradient has to be computed at each point on the contour to detect sudden changes which lead to type-2 critical points.

The formation of edges is done by employing some logical masks. We operate upon the critical points of type-1 by 3 x 3 logical masks, as shown in Figure 1, to yield other points on the step edges. In Figure 1, CP and CC refer to the current point under consideration. If, for any of these masks, BG and OB turn out to be background and object pixels, respectively, as determined by their z values, then the point under test is confirmed to be a part of the step edge.

To obtain roof edges we apply 3 x 5 logical masks, shown in Figure 1, to type-2 critical points. Pixels A1, A2 and B1, B2 are the neighbours of the opposite sides of CP. Pixels D1, D2 are assigned to roof edge if the
the edge boundaries represent the various surfaces involved. The next section describes how these surfaces are parameterized for the purpose of matching.

The proposed segmentation is not based on detecting surface curvature regions, it does not locate pixels by thresholding the magnitudes of principal curvatures", and the patches are not segmented using orientation discontinuities, only the depth discontinuities. Thus the segmentation is considerably simpler than those used elsewhere, and is mainly motivated by the desire to use quadric representation for the surfaces rather than curvature-based representations. For a case where a surface changes smoothly from convex to concave, the curvature change must necessarily be computed using the information on the normals of surface", thus requiring additional computations. However, this feature has not yet been incorporated in our segmentation method, though this can be added with advantage. In that case, we can still follow the same quadratics representation, for it offers a sound matching technique (discussed below).

**SURFACE REPRESENTATIONS**

A curved surface can be represented by the quadratic equation:

$$ax^2 + by^2 + cz^2 + dxy + exz + fyz + gx + hy + ix + j = 0$$ (1)

In matrix form, this becomes:

$$\begin{bmatrix} x & y & z \end{bmatrix} \begin{bmatrix} a & d/2 & e/2 \\ d/2 & b & f/2 \\ e/2 & f/2 & c \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} g \\ h \\ i \end{bmatrix} + j = 0$$ (2)

or

$$YD + EY + j = 0$$ (3)

When an object is being considered, it would have a number of surfaces. The object would then be described by the coefficient sets \((a_i, b_i, \ldots, j_i; i = 1, \ldots, N)\). The classification of quadratic surfaces can be based directly on the coefficients of the quadratic equation, and on a set of indices computed from the coefficients. In addition to classifying curved surfaces in terms of indices, the latter method includes the points and planes as special cases of quadratic surfaces. These indices are invariant to rotation, translation and size of a surface patch, and hence are called invariants. We make use of these invariants in the present work.

Fixing \(j = -1\) in (1), the other unknowns denoted as \(q = [a, b, c, d, e, f, g, h, i]^T\) can be evaluated by solving the matrix equation:

$$Aq = B$$ (4)

where:

$$A = \begin{bmatrix} x_1^2 & y_1^2 & z_1^2 & x_1y_1 & x_1z_1 & y_1z_1 & x_1 & y_1 & z_1 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
x_n^2 & y_n^2 & z_n^2 & x_ny_n & x_nz_n & y_nz_n & x_n & y_n & z_n \end{bmatrix}$$

$$B = [1, 1, \ldots, 1]^T$$
where \( n \) is the number of data points \( (x_i, y_i, z_i) \).

The least squares solution for \( q \) is given by:

\[
q = (A' A)^{-1} A' B
\]  
\( (5) \)

We can also work out the position and orientation of a surface from (1). When we translate the surface origin to the global coordinate system origin by changing variables to \( x', y', z' \) such that:

\[
x' = x + x_0 \quad y' = y + y_0 \quad z' = z + z_0
\]  
\( (6) \)

Then \( (x_0, y_0, z_0) \) gives the location of the surface origin with respect to global system origin. In other words, this is the translation of the surface origin to the global system origin. As a result of this translation, (3) becomes:

\[
Y' D Y'' + k = 0
\]  
\( (7) \)

where \( Y' = [x', y', z']' \) and \( k \) is related to \( j \) in (1) by:

\[
k = j = a x_0 + b y_0 + c z_0 + d x_0 y_0 + e x_0 z_0 + f y_0 z_0
\]

We call \( k \) the translation parameter, as it is due to translation. Here, \( (x_0, y_0, z_0) \) is obtained by solving three equations which result from the comparison of coefficients of the two quadratic equations (3) and (7).

Similarly, when we rotate the coordinate axes, at some rotation the cross product terms in matrix \( D \) go to zero, indicating the situation that the rotated axes are normal to the principal planes of the quadratic surface. Hence the new axes give the orientation information, and can be thought of as a rotation matrix. As the old axes are orthogonal, the new axes are also orthogonal. Since rotation of coordinate axes is equivalent to changing the basis vectors, we need an orthonormal transformation \( P \) to transform \( D \) in (2) into a diagonal form. Let:

\[
\tilde{Y} = P Y \quad \text{or} \quad Y = P^{-1} \tilde{Y}
\]  
\( (8) \)

In view of (8), (3) becomes:

\[
\tilde{Y}' (P^{-1} D P^{-1}) \tilde{Y}' + E P^{-1} \tilde{Y}' + j = 0
\]

\[
\tilde{Y}' (P D P') \tilde{Y}' + E P' \tilde{Y}' + j = 0 (\cdots P^{-1} = P)
\]

Let:

\[
P D P' = \hat{D}
\]  
\( (9) \)

Since \( D \) is symmetric and \( \hat{D} \) is diagonal with its elements representing eigenvalues \( \mu, \) we can rewrite (9) as:

\[
D P = P \hat{D}
\]

This gives us the following eigenvector, eigenvalue relation:

\[
D P_i = \mu_i P_i \quad i = 1, 2, 3
\]  
\( (10) \)

with \( P = [P_1, P_2, P_3] \).

Thus the transformation marix \( P \) consists of eigenvectors as its columns corresponding to eigenvalues of \( D. \) \( P \) therefore gives the orientation as the surface normals are directed along the eigenvectors.

The eigenvalues are associated with the length of the principal axis. If the eigenvalues are unique, the three axes are unique. If two of the eigenvalues are equal, we evidently have a surface of revolution about the third axis. The axes perpendicular to this axis of revolution are not unique; any two perpendicular axes that are both perpendicular to the axes of revolution will suffice. Finally, if all the eigenvalues are equal, the quadric is a sphere; any set of mutually orthogonal axes then form a set of principal axes.

**MATCHING PROCESS**

The problem of object recognition in a 3D scene can be looked upon as one of searching for a consistent matching between scene elements and model elements, both of which are represented as quadrics. The search space is enormous, and to control the combinatorial explosion, qualitative description of surface obtained from the parameters of a quadric may be used to limit the matching between like-surfaced elements on the lines of Grimson23, in which geometrical constraints are imposed on both the scene and model elements.

Let \( G \) and \( G' \) be the description of scene and model elements such that \( G \) is represented by \( (S_i, U_i; i = 1, \ldots, N) \), where \( S_i \) is the type of surface and \( U_i \) is its associated parameters. The parameters under consideration are the orientation \( P \) of surface, matrix \( D \) and translation parameter \( k \). Similarly, \( G' \) is represented by \( (S'_j, U'_j; j = 1, \ldots, N') \). The matching process consists of finding transformation \( J \) which checks whether \( S_i \) is the same as \( S'_j \) before mapping \( U'_j \) of \( G' \) on to its corresponding \( U_i \) of \( G \).

The recognition algorithm here is based on a measure of consistency defined by:

\[
g(M) = \text{Min} \sum_i \| U_i - J(U'_j) \|^2
\]  
\( (11) \)

subject to the condition \( S_i = S'_j \), where \( J(U'_j) \) is the transformed primitive of model elements \( G' \).

**Estimation of transformation**

The transformation being referred to here consists of rotation \( R \) and translation \( T \) with respect to some initial position and orientation. The rotation is made with respect to an axis of the reference system, and is to be followed by the translation. We need to rotate the orientation of the model to match with that of the scene. Similarly, we need to translate the model surface origin to match with that of the scene. Ignoring the coupling between rotation and translation, the equations representing these two operations are given by:

\[
P_i = R P'_i R'
\]  
\( (12) \)

From (7):

\[
k_i = T D; T + k_i
\]  
\( (13) \)

where subscript \( i \) refers to \( i \)th surface. The parameter...
set for the scene primitive is given by:

\[ U_i = \{ P_i, D_i, k_i \} \quad (14) \]

Correspondingly, for the model we have:

\[ U_i' = \{ P_i', D_i', k_i' \} \quad (15) \]

In view of (12) and (13) the measure becomes:

\[ g(M) = \sum_i \| P_i - R P_i' R^t \|^2 + w \sum_k k_i - k_i' - T^t D_i' T \|^2 \quad (16) \]

where \( w \) is a weighting factor. For the calculation of \( R \) and \( T \) we have to minimize the sums:

\[ \sum_i \| P_i - R P_i' R^t \|^2 \quad \text{and} \quad \sum_k \| k_i - k_i' - T^t D_i' T \|^2 \]

respectively.

However, before proceeding further we introduce here the concept of quaternions and Euler parameters.

### Quaternions and Euler Parameters

A quaternion \( \alpha \) is defined as complex number:

\[ \alpha = a_0 + a_1 i + a_2 j + a_3 k \quad (17) \]

for real parameters \( a_i (i = 0, 1, 2, 3) \) and axes \( (i, j, k) \).

It can be viewed as a linear combination of a scalar \( a_0 \) and a spatial vector \( a \) such that:

\[ \alpha = a_0 + a \quad (18) \]

The quaternion \( \alpha \) is also represented as parameters \( (a_0, a) \) which yield Euler parameters when \( \| \alpha \| = 1 \).

The conjugate of \( \alpha \):

\[ \alpha = a_0 - a \quad (19) \]

If \( \alpha \) and \( \beta \) are two quaternions, their multiplication \((\times)\) is defined as:

\[ \alpha(\beta) = (a_0 + a)(b_0 + b) = a_0 b_0 + a_0 b + b_0 a + a(b) \]

Separating scalar and vector products we can rewrite it as:

\[ \begin{bmatrix} c_0 \\ c \end{bmatrix} = \begin{bmatrix} a_0 & -a' \\ a & a_0 I + a \end{bmatrix} \begin{bmatrix} b_0 \\ b \end{bmatrix} = \begin{bmatrix} b_0 & -b' \\ b & b_0 I - b \end{bmatrix} \begin{bmatrix} a_0 \\ a \end{bmatrix} \]

\[ = \alpha \cdot \beta = \beta \cdot \alpha \quad (21) \]

where the signs \( \cdot \) and \( \cdot \) correspond to the signs \( \cdot \) and \( \cdot \) attached to \( a \) and \( b \), respectively.

and \( \cdot \) on \( a \) defines the matrix:

\[ \hat{a} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \quad (23) \]

### Estimation of Rotation

Computation of the rotation matrix \( R \) from (16) is a difficult task, as it is a nonlinear function of rotation angle \( \theta \) about an arbitrary axis. To obtain \( R \) by simplifying the minimization of \( g(M) \), we use quaternions which represent a finite rotation of an axis \( \nu \) in space by an angle \( \theta \). Following Chou and Kamel we have:

\[ R = (\cos \theta) I + (1 - \cos \theta) \nu \nu' + \sin \theta \nu \]

In terms of Euler parameters \((e_i, e)\) which are unit quaternions, (24) can be expressed as:

\[ R = (e_0^2 - e' e) I + 2(e e' + e e) \]

where \( \nu \) is related to Euler parameters through:

\[ \nu = \pm e'/\sqrt{(e' e)} \quad \text{and} \quad e_0 = \cos \theta/2 \]

with:

\[ e = (e_i, e, e_i') \]

Conversely, given a rotation matrix \( R \), it is also possible to obtain its Euler parameters.

Now let us look at minimizing the measure \( g(M) \) in (16). Let the first term be denoted by \( F_r \). Thus:

\[ F_r = \sum_i \| P_i - R P_i' R^t \|^2 \quad (26) \]

Multiplying both sides by the unit norm \( \| R \|^2 \):

\[ \| R \|^2 F_r = \sum_i \| P_i R - R P_i' R^t \|^2 \]

or

\[ F_r = \sum_i \| P_i - R P_i' R^t \|^2 \quad (27) \]

Transforming (27) to an equivalent problem invoking the corresponding Euler parameters, we obtain:

\[ F_r = \sum_i \| L_i(x) l_i - L_i(x) l_i' \|^2 \quad (28) \]

where \( L_i, L_i' \) are the Euler parameters associated with the rotational matrices \( P_i, R, P_i' \), respectively. Here we are treating \( P_i, R_i \) as rotational matrices, as mentioned earlier. Applying (22) in (28) we obtain:

\[ F_r = \sum_i \| L_i^t l_i - L_i^t l_i' \|^2 \]

or

\[ F_r = \sum_i \| L_i l_i - L_i l_i' \|^2 \quad (29) \]
where:

\[ L_i = I_i - I; \]

This has to be minimized subject to the condition \( \|I_i\| = 1 \). In view of this, the above equation becomes:

\[
Fr = \sum i_r L_i L_i' F_r' = I_r C L_i' \tag{30}
\]

where:

\[ C = \sum L_i L_i' \text{ (a symmetric matrix)} \]

The solution to the minimization problem is reduced to finding the eigenvalues of \( C \). The minimum value of \( F_r \) is given by the smallest eigenvalue of \( C \). Thus our error measure of rotation is:

\[ e_r = \mu_{\text{min}}(C) \tag{31} \]

Further, \( I_r \) is given by the eigenvector corresponding to this \( \mu_{\text{min}} \). The components of \( I_r \) which represent Euler parameters can be used to find \( R \) from (25). Note from (30) that we need at least two initial matches which must not be coaxial or coplanar for estimating the rotation.

Now, we have our parameter set corresponding to \( P \) and \( P' \) in terms of Euler parameters \( I \) and \( I' \). In the next subsection we obtain the remaining parameters of \( U \), and \( U' \) in modified form.

**Estimation of translation**

To estimate the translation \( T \) we have to minimize the second term denoted by:

\[ F_t = \sum \|k_i - k_i' - T' D_i' T\|^2 \tag{32} \]

If we make use of transformed relations (13), we get a simplified form for \( F_t \). Thus, with \( PT = \tilde{T} = [t_1, t_2, t_3]' \) (13) becomes:

\[ k_i - k_i' - \tilde{D}_i \tilde{T} = 0 \tag{33} \]

where \( \tilde{D}_i \) is diagonal. Accordingly, since \( \tilde{T} \) is unknown \( \tilde{F}_t \) can be written as:

\[ \tilde{F}_t = \sum \|k_i - k_i' - \tilde{D}_i \tilde{T} \|^2 \tag{34} \]

Let there be \( N \) pairs of scene and model elements over which this minimization of \( \tilde{F}_t \) is desired. Then defining:

\[ d_i = k_i - k_i' \tag{35} \]

\[ W_i = [\tilde{D}_{i1}, \tilde{D}_{i2}, \ldots, \tilde{D}_{i3}]' \]

with \( \tilde{D}_i = [\tilde{D}_{i1}, \tilde{D}_{i2}, \tilde{D}_{i3}] = \text{Diagonal elements of } \tilde{D}_i \)

will lead to:

\[ \hat{F}_t = \|d - \tilde{W} \tilde{T} \|^2 \tag{36} \]

where \( d = [d_1, \ldots, d_N]' \) is an \( N \times 1 \) vector, and \( \tilde{W} \) is a \( N \times 3 \) matrix. The minimization now reduces to solving (36). Applying the least squares method to (36) will yield the solution:

\[ \tilde{T}^2 = (\tilde{W}' \tilde{W})^{-1} \tilde{W}' \tilde{d} \tag{37} \]

The resulting error is given by:

\[ e_t = d'(d - \tilde{W} \tilde{T}^2) \tag{38} \]

The translation vector is obtained as:

\[ T = \pm \frac{P'}{\sqrt{e_t}} \tag{39} \]

Since the sign of \( T \) cannot be fixed by this solution, we turn to (32) again. Expanding the terms in (32), we obtain:

\[ F_t = \sum \|k_i - k_i' - (a'i t_1 + b'i t_2 + c'i t_3 + d'i t_1 t_2 + e'i t_1 t_3 + f'i t_2 t_3)\|^2 \tag{40} \]

If the minimization is over \( N \) pairs, \( F_t \) can be rewritten as:

\[ F_t = \|d - \tilde{W}s\|^2 \tag{41} \]

where \( d = [d_1, \ldots, d_N] \) is an \( N \times 1 \) vector, and \( \tilde{W} = [\tilde{W}_1, \tilde{W}_2, \ldots, \tilde{W}_s]' \) is an \( N \times 6 \) matrix, \( s = [s_1, s_2, \ldots, s_6]' \) is \( [t_1, t_2, t_3, t_1 t_2, t_1 t_3, t_2 t_3]' \) with \( d_i = k_i - k_i' \), and \( \tilde{W}_i = [a_i, b_i, c_i, d_i, e_i, f_i]' \).

The minimization by least square methods yields \( s \) from:

\[ s = (\tilde{W}' \tilde{W})^{-1} \tilde{W}' \tilde{d} \tag{42} \]

and error from:

\[ e_t = d'(d - \tilde{W}s) \tag{43} \]

From (42):

\[ t_1 = \pm \sqrt{s_1} \tag{44} \]

so that:

\[ t_2 = s_2/t_1 \tag{45} \]

\[ t_3 = s_3/t_1 \tag{46} \]

The sign of \( t_1 \) is first chosen arbitrarily. Next it is checked whether \( \text{sgn} (t_2, t_3) = \text{sgn} (s_{23}) \) for a chosen sign of \( t_1 \). If it is not so, the sign of \( t_1 \) is reversed (similarly for the signs of \( t_2 \) and \( t_3 \)). For a correct solution we require six matches of primitive elements in (42) for estimating \( s \), whereas we require only three matches of primitive elements if we use (37) such that the matches must not be coaxial or coplanar. Hence for matching scene elements with the model elements, we use the error measure for translation (38), thus saving computational burden during the matching process. But to find the final location of the identified object in the scene, we must use (43) in place of (38). Thus, for an intermediate level of matching the translational components, we have the original parameters \( D_i \) and \( D_i' \).

Vol 10 No 9 November 1992
reformulated as $\tilde{D}_i$ and $\tilde{D}_i'$ respectively. In view of this, we can express the parameter sets (14) and (15) as:

$$U_i = \{i, \tilde{D}_i, k_i\}$$

$$U_i' = \{i, \tilde{D}_i', k_i\}$$

The overall error for matching becomes:

$$\hat{g}(M_i) = e_i + w\hat{e}_i$$

(47)

where $M_i$ is referred to as the matching of the $i$th surface of the scene with the $i$th surface of the model.

However, to find the final location of the identified object in the scene, the error to be considered is:

$$g(M_i) = e_i + w\hat{e}_i$$

(48)

**SEARCH STRATEGY**

The proposed matching procedure is a tree search with backtracking whenever error $\hat{g}(M)$ exceeds the specified limit $g_{max}$ at some level. To start with, we label the scene surfaces by $i = 1, \ldots, N$ and model surfaces by $j = 1, \ldots, N'$ with the restriction $N' > N$, where each label corresponds to a quadric. This restriction is necessary from the fact that a model contains information about all its surfaces, whereas a scene need not. If the quadrics are classified, then the matching can be made between elements of the same shape in the scene and the model.

The matching algorithm proceeds by considering one of the scene surfaces at a time for the purpose of matching with any of the model surfaces, and selecting that model surface which after rotation and translation yields minimum error measure. To avoid indeterminacy in the tree search procedure, an exhaustive search is made for the first three levels. It may be noted that we require at least two matches of primitive elements for the estimation of rotation, and three matches of primitive elements for the estimation of translation, with the matches satisfying the condition as stated above.

After obtaining partial matching for three levels, the search is made over the next levels taking each partial matching at a time selecting label $j$ which has not been selected so far for matching with the label $i$. The selection is made such that $\hat{g}(M_{ij})$ is minimum at level $i$. Each time an error measure is checked to see whether it is within the specified limit. If the error exceeds this limit, the algorithm backtracks to eliminate label $j$ which yields the largest error at level $i$, otherwise the

---

**Figure 2. Segmentation at various stage.**

(a) Raw image; (b) equidepth contours; (c) roof edges; (d) semi-step edges; (e) step edges; (f) segmented range image

**Figure 3. Segmented range image of object-1 with labels.**

(a) step edge; (b) roof edge; (c) semi-step edge

*Image and Vision Computing*
search proceeds in the forward direction. The process stops when \( N \) surfaces of the scene find the closest match with \( N \) surfaces of the model.

Out of several solutions obtained, the best one is selected as the final matching.

The output of the matching is \( M \), which contains the primitives of the scene identified in the model, and the associated optimal transformation gives the position and orientation of the object in the scene, and error measure \( g(M) \) gives the error involved in the matching process.

RESULTS OF A CASE STUDY

To illustrate the method developed in this paper, we have taken up a composite object consisting of a small cylinder fixed on a large cylinder such that they share the same principal axis. The 3D information about the object is obtained through range measurement shown in Figure 2. By slicing the map at fixed increment ranges and retaining only the pixels at the outer rim of the slice, we obtain Figure 2b. Figure 2c shows the roof edges obtained by applying logical masks on type-2 critical points. Similarly, Figures 2d and e showing semi-step-edges and step-edges, are the result of application of logical masks on type-3 and type-1 critical points, respectively. The final segmented range image is shown in Figure 2f, which indicates boundaries of various regions. Figure 3 shows the different types of edges.

The segmented image is stored as a series of linked lists of the connected pixels. The range data enclosed within the edge boundaries is collected to serve as patches, and is then modelled as quadrics.

![Figure 4. Surface details of models. (a) Object model-1; (b) Object model-2: 1: front surface of bottom cylinder; 2: backsurface of bottom cylinder; 3: front surface of top cylinder; 4: back surface of top cylinder; 5: bottom plane; 6: middle plane; 7: top plane](image)

### Table 1. Coefficients of quadrics representing two typical surfaces

<table>
<thead>
<tr>
<th>Coefficients</th>
<th>Model (surface no. 1)</th>
<th>Scene (surface no. 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a )</td>
<td>-0.00177</td>
<td>-0.00047</td>
</tr>
<tr>
<td>( b )</td>
<td>-0.00173</td>
<td>-0.00023</td>
</tr>
<tr>
<td>( c )</td>
<td>-0.00062</td>
<td>-0.00031</td>
</tr>
<tr>
<td>( d )</td>
<td>0.00151</td>
<td>0.00018</td>
</tr>
<tr>
<td>( e )</td>
<td>0.000000</td>
<td>0.000064</td>
</tr>
<tr>
<td>( f )</td>
<td>0.000000</td>
<td>0.000026</td>
</tr>
<tr>
<td>( g )</td>
<td>-0.03553</td>
<td>-0.01079</td>
</tr>
<tr>
<td>( h )</td>
<td>0.03376</td>
<td>0.02261</td>
</tr>
<tr>
<td>( i )</td>
<td>-0.008719</td>
<td>0.003980</td>
</tr>
</tbody>
</table>

### Table 2. Translated origins

<table>
<thead>
<tr>
<th>Coordinates</th>
<th>Model</th>
<th>Scene</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_o )</td>
<td>28.2901</td>
<td>55.1278</td>
</tr>
<tr>
<td>( y_o )</td>
<td>34.5985</td>
<td>91.6682</td>
</tr>
<tr>
<td>( z_o )</td>
<td>30.0005</td>
<td>161.7581</td>
</tr>
</tbody>
</table>

### Table 3. D matrices

<table>
<thead>
<tr>
<th>Model</th>
<th>Scene</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0000</td>
<td>-3.7057</td>
</tr>
<tr>
<td>-3.7057</td>
<td>9.8325</td>
</tr>
<tr>
<td>-0.13E-05</td>
<td>-0.81E-06</td>
</tr>
</tbody>
</table>

### Table 4. Eigenvalues, eigenvectors and Euler parameters

<table>
<thead>
<tr>
<th>Eigenvalues</th>
<th>Scene</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.349</td>
<td>11.181</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Eigenvectors</th>
<th>Scene</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.3420</td>
<td>0.7079</td>
</tr>
<tr>
<td>-0.9597</td>
<td>0.2577</td>
</tr>
<tr>
<td>0.0000</td>
<td>-0.8576</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Euler parameters</th>
<th>Scene</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.4511</td>
<td>-0.7634</td>
</tr>
</tbody>
</table>

### Table 5. Invariants for classification

<table>
<thead>
<tr>
<th>( I )</th>
<th>19.48</th>
<th>17.80</th>
</tr>
</thead>
<tbody>
<tr>
<td>( J )</td>
<td>-7.68</td>
<td>-52.05</td>
</tr>
<tr>
<td>( K )</td>
<td>1.36</td>
<td>-1.54</td>
</tr>
<tr>
<td>( A )</td>
<td>8.96E+02</td>
<td>8.13</td>
</tr>
<tr>
<td>( A' )</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>( A'' )</td>
<td>-8.99E+03</td>
<td>-4.25E+05</td>
</tr>
<tr>
<td>( A''' )</td>
<td>-5.66E+03</td>
<td>1.36E+05</td>
</tr>
</tbody>
</table>
To carry out the matching process, a CAD model is built up whose surfaces are modelled through quadrics. It may be mentioned that each cylinder is assumed to be made up of a front and a back surface, as one can only see half of the cylinder at any viewpoint. As such, no symmetry property is used in the matching process, and no constraint is placed on how an object is formed. Thus the model consists of seven surfaces, as shown in Figure 4a. The quadric representation of two typical surfaces, one belonging to the model and the other to the scene, is given in Table 1.

Using the coefficients shown in Table 1, the translated (surface) origins are obtained as given in Table 2. The normalized coefficients are then used to construct D matrices given in Table 3.

From the D matrix, we obtain eigenvalues and eigenvectors, the latter being converted into Euler parameters. Table 4 shows the eigenvalues, eigenvectors and Euler parameters for the above typical surfaces. The eigenvalues refer to the elements of transformed D matrix, and the eigenvectors give the orientation of the surface, whereas Euler parameters give the axis-angle representation of the same. The model surface has a rotation about one of its axes as borne out by the model eigenvalues, and the scene surface has no rotation as indicated by its eigenvalues; its axes are therefore unique.

Using the original coefficients of the quadratic equation, we have attempted the classification of quadrics based on Besl and Jain. However, this did not prove to be very conclusive. Then we moved on to the classification based on a set of invariants computed from D matrices. The invariants for the surfaces under consideration are as given in Table 5.

### Table 6. Feature data of the model and scene

<table>
<thead>
<tr>
<th>Euler parameters</th>
<th>Eigenvector</th>
<th>Translation parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.4811</td>
<td>-0.7634</td>
<td>0.0963</td>
</tr>
<tr>
<td>0.5028</td>
<td>0.1281</td>
<td>0.4593</td>
</tr>
<tr>
<td>0.3941</td>
<td>-0.7975</td>
<td>-0.0933</td>
</tr>
<tr>
<td>0.5644</td>
<td>-0.8737</td>
<td>0.1679</td>
</tr>
<tr>
<td>0.9848</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>0.3960</td>
<td>-0.0836</td>
<td>-0.0195</td>
</tr>
<tr>
<td>0.3887</td>
<td>-0.1141</td>
<td>-0.0216</td>
</tr>
<tr>
<td>Scene</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.6926</td>
<td>0.0364</td>
<td>0.3687</td>
</tr>
<tr>
<td>0.7003</td>
<td>-0.6564</td>
<td>-0.6762</td>
</tr>
<tr>
<td>0.5054</td>
<td>0.0900</td>
<td>0.2091</td>
</tr>
<tr>
<td>0.6446</td>
<td>-0.9706</td>
<td>0.0684</td>
</tr>
</tbody>
</table>

The Euler parameters, eigenvalues and translation parameters serve as features for the surfaces in the matching process. The feature data of the model and the scene are given in Table 6.

We have made an exhaustive search for the first three levels, and identified a few partial solutions with $g(M)$ as the measure of consistency which consists of $e$ and $\theta$. In the initial phase, we have used a weighting factor $w$ of 2. The results for 3-levels are presented in Table 7.

Based on these initial selected solutions, the search is extended to the fourth level, where we have used $w$ of 6 to give more weight to the translation error. Since we have four visible surfaces, our search concludes at the

### Table 7. Partial solutions

<table>
<thead>
<tr>
<th>Solution</th>
<th>Partial matching</th>
<th>Error measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(1, 2, 4)</td>
<td>1.35 → Selected</td>
</tr>
<tr>
<td>2</td>
<td>(1, 3, 7)</td>
<td>2.63</td>
</tr>
<tr>
<td>3</td>
<td>(1, 6, 7)</td>
<td>9.07</td>
</tr>
<tr>
<td>4</td>
<td>(2, 1, 3)</td>
<td>1.70 → Selected</td>
</tr>
<tr>
<td>5</td>
<td>(2, 5, 3)</td>
<td>3.15</td>
</tr>
<tr>
<td>6</td>
<td>(2, 6, 5)</td>
<td>7.29</td>
</tr>
<tr>
<td>7</td>
<td>(3, 1, 6)</td>
<td>1.73 → Selected</td>
</tr>
<tr>
<td>8</td>
<td>(3, 2, 7)</td>
<td>2.98</td>
</tr>
<tr>
<td>9</td>
<td>(3, 6, 7)</td>
<td>8.15</td>
</tr>
</tbody>
</table>

### Figure 5. Segemented range image of object-2
Table 9. Results of final matching

<table>
<thead>
<tr>
<th>Solution</th>
<th>Matching</th>
<th>Error measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(2, 6, 3, 8, 9)</td>
<td>2.23 → Final solution</td>
</tr>
<tr>
<td>2</td>
<td>(3, 4, 6, 8, 9)</td>
<td>8.15</td>
</tr>
<tr>
<td>3</td>
<td>(4, 8, 6, 8, 10)</td>
<td>17.85</td>
</tr>
<tr>
<td>4</td>
<td>(6, 7, 3, 9)</td>
<td>10.76</td>
</tr>
<tr>
<td>5</td>
<td>(8, 6, 3, 4, 10)</td>
<td>31.34</td>
</tr>
<tr>
<td>6</td>
<td>(7, 6, 5, 3, 10)</td>
<td>25.81</td>
</tr>
</tbody>
</table>

fourth level. At the end of fourth level, the matching results appear as in Table 8.

Table 8 shows that the final solution selected eventually consists of visible surfaces 1, 3, 7, 6 of the model, viz bottom front cylinder surface, top front cylinder surface, top plane and middle plane, respectively, as identified in the model surfaces.

We have carried out similar studies on another object model comprising six planar surfaces with a solid hemisphere and a hemispherical groove on one of its planar surfaces, which form curved surfaces as shown in Figure 4b. The segmented range image of the corresponding object is shown in Figure 5. The final results of matching are summarized in Table 9.

In the matching process, we have treated all surfaces as quadrilaterals for simplicity of illustration and for want of robust technique for the classification of quadrilaterals, as the invariants used above for this purpose were found to yield erroneous results in a few cases. However, if the shape of the surface is also taken into account, we could carry out the matching of like surfaces through parallel processing, thus limiting the search space to a manageable one.

CONCLUSIONS

A new method has been presented for the recognition of curved and planar surfaces of 3D objects by matching the scene and model surface primitives. Knowledge of the object model in the form of quadrilaterals has been assumed. It has been shown that by converting orientation and position information into Euler parameters, eigenvalues and translation parameters, the matching algorithm not only establishes the identity of the object, but also provides as an important by-product the rotation and translation components of the object so often required for robot manipulation.

Segmentation and representation of range data may not be meaningful in the presence of noise. For this we have to determine the segmentation and model (quadratic) parameter values that maximize the likelihood of the data.7

Sometimes there could be indeterminations in the estimation of transformation if the eigenvalues of both the model and the scene surfaces are non-unique, giving rise to ill-conditioning in the matrices involved. This situation arises when the match is between shapes with aligned coordinate axes or between swept surfaces. When this happens one could choose an alternate coordinate system for the model, since the scene surfaces being view-dependent should not be modified.

If the quadrilaterals can be classified, the matching can be carried out between the scene and model surfaces having the same shape, thus incorporating a parallel processing feature into the matching. We could not incorporate this feature in the present work as the invariants employed were found to yield erroneous results during classification of some surfaces. Computational simplification during the matching is possible if we also use relationship evidence and property values in the correspondence generation between model and scene surfaces, as Fisher12.

Further work is currently being conducted to study the relative merits of quadrilaterals over other forms of surface representations. In this context, we mention that quadrilaterals representation coupled with a good CAD modeller is a viable alternative to volumetric representation using superquadrics, and that the curvature class representation is complementary to it, as these two representations together can eliminate the need for a modeller because it is easier to construct global information akin to volumetric modelling through the principles of curvature rather than with the quadratic parameters.

ACKNOWLEDGEMENTS

We would like to record our thanks to Prof. M. Brady and Dr. A. Blake of Oxford University, U.K. and to the anonymous reviewers for providing valuable suggestions to the authors in improving the manuscript. Our thanks are also due to Mr. Paul Francis of the Department of Computer Science and Engineering IIT Delhi, for his help in the CAD modelling.

REFERENCES


19 Wani, M and Biswas, K K 'A parallel algorithm for range image segmentation', submitted to IEEE


