グラフィクスとCAD　42－5
(1989. 12. 14)

On Identifying Symmetry of a Three-Dimensional Object from its Octree

Predrag Minovic, Seiji Ishikawa and Kiyoshi Kato

Kyushu Institute of Technology,
Faculty of Engineering

ABSTRACT　An algorithm for identifying symmetry of a three-dimensional object given by its octree is presented and the symmetry degree, a measure of object symmetry, is proposed. The algorithm is based on traversal of the octree obtained by the principal axis transform of an input octree. An object can be at an arbitrary position and with arbitrary orientation within the octree space and all types of symmetry, *i.e.* bilateral, axial and point symmetry can be identified. The operation of the algorithm is illustrated using some synthetic test objects. The results, comprising an identified symmetry type and the corresponding symmetry degree, were satisfactory.

オクトツリーを用いた3次元物体の対称性の認識について

プレドラッグ・ミノヴィッチ　石川聖二　加藤清史

九州工業大学　工学部

あらまし　オクトツリーで表現された3次元物体の、対称性を認識するアルゴリズムを提案し、また物体の対称の程度を現わす対称度を定義する。本アルゴリズムは、対象物体が与える一般のオクトツリー表現から、主軸変換によって得られる主オクトツリー上のノードを順に辿ることによって、その物体の対称性の有無を調べる。物体の位置・方向は任意で、すべての対称のタイプ、即ち面対称・軸対称・点対称が認識される。合成データを用いた例によって、アルゴリズムの動作が示されている。認識された対称のタイプおよび対称度の値とも、満足のいく結果が得られた。

1. INTRODUCTION

Symmetry is one of the most important features of objects. Identifying symmetry is an interesting problem in itself in computational geometry, but the notion of symmetry can be useful in other fields as well. Among possible applications an obvious one is in data compression, since for representing symmetrical object it is enough to keep a model of its symmetrical half and the information about the position of the plane or the axis of symmetry. In computer vision, symmetry can be used as a special object's feature which would reduce the number of candidates for matching in the library. A more advanced application can be automatic correction of an "almost symmetrical" object to a symmetrical one. This may be useful, for example, in computer vision for object models obtained from camera or range images, in CAD systems for object models built on the basis of digitized line drawings, or in computer graphics for models of biomedical organs obtained by computer tomography.

In spite of the importance and usefulness of the notion of symmetry, there have been, to the best of our knowledge, no techniques reported for identifying symmetry of three-dimensional (3D) objects, although there were some efforts in the case of 2D images. One of the first results was reported in [1]. It is rather limited in scope because an image symmetry can be detected along 4 directions only (horizontal, vertical, the left and the right diagonal). Quadtrees were used for image representation and it was shown that they facilitate efficient checking for symmetry, but the method has almost no practical importance since an image has to be aligned with the quadtree space. The same authors, however, used this idea for data compression, obtaining so-called "minimal quadtrees" [10]. Kanade and Kender developed a method for detecting ordinary and skewed symmetry of line drawings [8]. Skewed symmetry was used as a cue for the recognition of 3D objects from one view [9]. Slightly more general approach was reported by Attalah [2]. His system can identify the axes of ordinary symmetry of a figure which consists of a collection of finite number of points, segments, circles, *etc.* In [5], a method which deals with an arbitrary 2D image and is able to detect both ordinary and skewed symmetry was presented, but its time complexity can be excessive (heuristic search for the axes of symmetry which satisfy the predefined analytic constraint) and it is not fully automatic because the symmetry evaluator which checks symmetry for a supposed axis is not reliable. An interesting research was recently done by Marola. He considered "almost symmetric" planar images and introduced the so-called coefficient (measure) of symmetry [11]. The results have been used for object detection and location [12].

Symmetry is also a subject of an active research in psychophysics and the theory of perception. There have been some attempts to quantitatively describe symmetry and several symmetry measures were proposed [18, 17], but no attempts to locate symmetry axes in an analytical way are reported (it is interesting that a special device called symmetrograph was constructed to locate symmetry axes mechanically [18]).

Thus, even in 2D it seems that there is not one standard technique for symmetry identification, and it is not clear how to extend available methods to a 3D case. In this paper, we claim that our procedure is capable of identifying symmetry of an arbitrary 3D object without imposing any constraints on its shape, position and orientation. It is only assumed that an object is represented by its octree (for the reasons explained below). Bilateral, axial and point symmetry can be identified and the level of symmetry is expressed (measured) by means of the so-called symmetry degree.

This paper is organized as follows: after the review of some facts about principal axes, and a short overview of octree data structure in section 2, detailed explanation of the proposed algorithm is given in section 3. Results are presented and discussed in section 4, followed by conclusion and ideas about further work in section 5.

2. TOWARDS THE SOLUTION

Two issues have turned out to be crucial for the solution of the symmetry identification problem: strong analytic conditions for symmetry and a general object representation scheme which can support all required computations. These topics are briefly discussed in the sequel.

2.1. An analytic conditions for symmetry

First, a short explanation of the notion of principal axes is given, because our analytic condition is based on them. Consider an object in the N-dimensional space. Principal axes can be defined [7] as a set of N vectors which satisfy the following conditions:

a. they point in the direction of the maximum variance of data;

b. they are mutually orthogonal.

Principal axes are actually eigenvectors of the so-called covariance matrix. Each eigenvalue is equal to the variance of data along a corresponding eigenvector. Principal axes have a common intersection point at the object's centroid, but their orientation is undefined. Since covariance matrix is symmetrical, the eigenvalue problem associated with it always has real solutions.

On the other hand, principal axes have the following important relations with the object symmetry [15]:

Theorem 1: Any plane of symmetry of a body is perpendicular to a principal axis.

Theorem 2: Any axis of symmetry of a body is a principal axis.

These facts can be used as necessary conditions for an axis (a plane) to represent an axis (a plane) of symmetry of a given object. But these conditions are not sufficient, thus symmetry has to be verified or rejected by direct examination. For doing this, it will be convenient to use the coordinate system placed at an object's centroid with principal axes as coordinate axes. In that case, symmetry has to be examined with respect to the coordinate axes and the coordinate planes. Point symmetry with respect to the origin (*i.e.* the centroid) can also be examined.

Aligning an object from an arbitrary coordinate system to the one described above is known as a principal axis transform [7] (sometimes referred to as Hotelling, eigenvector or discrete Karhunen-Loeve transform). It has important property of being invariant to the rotation and translation of an object.

The proposed condition for symmetry is selective enough because in general case there will be 7 types of symmetry to check for: bilateral, with respect to 3 coordinate planes; axial, with respect to 3 coordinate axes and point symmetry with respect to the origin (*i.e.* the centroid). It will be shown later that some hints (for which type of symmetry to check first) can be employed to speed up this search.

2.2. Object representation

In the case N=3, the covariance matrix is the matrix of inertia, comprising central moments of order two. Hence, the chosen object representation scheme has to support the following operations: computing of moments, linear transformation (translation and rotation) and symmetry evaluation. It will be shown that the octree data structure satisfies all these requirements.

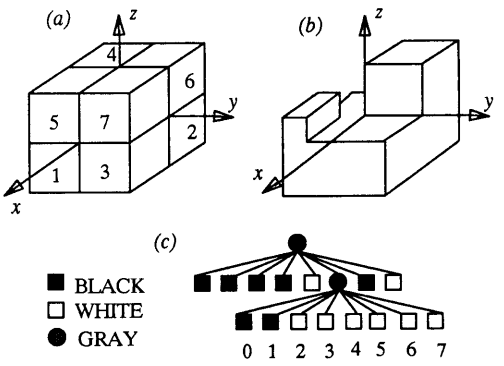


Figure 1. An object and its octree. (*a*) Octant labels (octant 0 is occluded). (*b*) An object, and (*c*) its octree.

An octree is a hierarchical data structure which represents a given object as a juxtaposition of cubes of different sizes, obtained by recursive and regular decomposition of the 3D space (Fig. 1). Cubes are organized in a tree of degree 8. Decomposition is applied until either all cubes are completely inside or outside the object or the predefined depth of recursion (called octree

resolution) is reached. Octree cubes (sometimes called nodes or octants) are classified as BLACK and WHITE (representing presence and absence of the object, respectively; these two are referred to as leaf nodes), and GRAY (they intersect the boundary the object, and they are further decomposed). Potential GRAY nodes at the resolution level cannot be further decomposed. They are marked as BLACK or WHITE according to a certain "leaf criterion". Octree nodes at the resolution level are called voxels.

Octree encoding methods can be divided into pointer and pointer-less schemes. The former are based on a hierarchical nature of the octree, while the latter take advantage of the fact that each octree node has either 8 or zero sons. The choice of the encoding scheme is influenced by an actual application.

Octrees are direct extension of quadtrees in 3D; both structures are based on the coherence of data they represent. Octrees have been used for various applications in computer graphics and pattern recognition. Recent surveys are available: [13, 14, 3]. Octrees are attractive because: (*a*) any object can be represented, regardless of its shape; (*b*) calculations of mass, moments and set operations are extremely simplified and (*c*) an octree can be constructed automatically from 2D camera views, range data, cross sections, other representation schemes, *etc.* Disadvantages include (*a*) explosion of the number of octree nodes for high resolution models and (*b*) quantization error as a result of the application of a leaf criterion.

Regarding the requirements stated before it can be noted that octrees support computation of moments and symmetry evaluation very well. Linear transformations are not immune to a quantization error, but it does not affect their performance significantly.

3. ALGORITHM OVERVIEW

Under the assumption that for a given object its octree representation is available, an algorithm for symmetry identification can be formulated in three steps as follows:

Step 1: Compute the parameters of the principal axis transform.

Step 2: Perform the principal axis transform of the input octree, and build the "principal octree".

Step 3: Traverse the principal octree and judge the symmetry by computing corresponding symmetry degrees for bilateral, axial and point symmetry.

These steps are explained in more detail below. Estimation of the execution time is also included.

3.1. Parameters of the principal axis transform

The mass and the centroid of an object can be expressed by the means of moments of order zero and one, respectively. Moments of order $(p+q+r)$ for $p, q, r \in N_0$ (N_0 is the set of non-negative integers) are given by

$$M_{pqr} = \iiint_S x^p y^q z^r \rho(x, y, z)\, dx\, dy\, dz \qquad (1)$$

where S is the whole octree space and ρ is a density function assumed to be piecewise continuous (thus bounded) and equal to zero except at a finite part of S. The mass and the coordinates of the centroid are given by

$$m = M_{000},$$

$$(x_c, y_c, z_c) = \left(\frac{M_{100}}{M_{000}}, \frac{M_{010}}{M_{000}}, \frac{M_{001}}{M_{000}} \right)$$

Now, central moments of order $(p+q+r)$ for $p, q, r \in N_0$ are defined as:

$$\mu_{pqr} = \iiint_S (x-x_c)^p (y-y_c)^q (z-z_c)^r \cdot \rho(x, y, z)\, dx\, dy\, dz \qquad (2)$$

where S and ρ have the same meaning as before. Note that in the case of octrees, integration in equations (1) and (2) becomes summation over all BLACK octree nodes (since $\rho=0$ for WHITE nodes). Function ρ becomes the mass of a particular cube, with point (x, y, z) being its center. This computation is rather straightforward (but see [4] for some slight improvements). As mentioned earlier, in the 3D space a covariance matrix is the matrix of inertia which is given by the means of central moments of order two:

$$I = \begin{bmatrix} \mu_{020}+\mu_{002} & -\mu_{110} & -\mu_{101} \\ -\mu_{110} & \mu_{200}+\mu_{002} & -\mu_{011} \\ -\mu_{101} & -\mu_{011} & \mu_{200}+\mu_{020} \end{bmatrix}$$

Now, the eigenvalue problem for matrix I is solved. Suppose that obtained eigenvectors $\mathbf{e}_1$, $\mathbf{e}_2$ and $\mathbf{e}_3$ are given by their unit components e_{ij} (i, j = 1, 2, 3). Then the transformation matrix for the principal axis transform can be expressed by

$$T = Trans(-x_c, -y_c, -z_c) \cdot Rot$$

where *Trans* represents translation to the centroid, and *Rot* is a rotational component defined by

$$Rot = \begin{bmatrix} e_{11} & e_{12} & e_{13} & 0 \\ e_{21} & e_{22} & e_{23} & 0 \\ e_{31} & e_{32} & e_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Note that the eigenvalue problem for matrix I always has solutions, thus matrix T always exists. However eigenvectors are uniquely defined (up to their direction) only if all eigenvalues are distinct. If 2 (3) eigenvalues are equal, directions of 2 (3) eigenvectors are not unique (but condition of mutual orthogonality holds). These degenerative cases correspond to appearance of multiple axes and/or planes of symmetry. They are not further elaborated here.

3.2. Octree transformation

In this step the input octree is transformed using transformation matrix T defined above. The problem with an octree transformation is that, save a few exceptions (translation for a vector with integer components, rotation by multiples of 90°), a quantization error is introduced again. Thus, a leaf criterion has to be chosen carefully to prevent changes in object's mass and shape. Precision can be improved by increasing the resolution of the output octree, but the number of its nodes will increase too. Note that for small objects such resolution increase is mandatory, in order to preserve their shape. Regardless of a transform, the most precise octree is kept all the time, and all transformations are computed with respect to it.

Among several methods (although similar in nature) reported for octree transformation, the method of Weng and Ahuja has been adopted [16]. The leaf criterion is formulated so that a voxel of the output octree is marked BLACK if and only if its center lies on or inside some transformed BLACK node of the input octree. This scheme tends to keep the object's mass unchanged. The *average* execution time is reported to be $O(rK)$, where K is the the number of nodes of the input octree and r is the resolution of the output octree.

We shell refer to the transformed octree as the *principal octree* of an object.

3.3. Symmetry evaluation

The particular type of symmetry is judged according to the corresponding *symmetry degree*, a measure of symmetry which we define as the normalized size of the symmetric subset (core) of the object (this measure is known as "self-overlap" [18]). In other words symmetry degree (sd) is the ratio:

$$sd = \frac{m_{sym}}{m}$$

where m_{sym} is the mass of the symmetric subset of the object, and m is the object mass. It is clear that in a general case $0 \leq sd \leq 1$. The value m_{sym} is computed from the principal octree.

In [1] a recursive method for identifying axial symmetry of an image from its quadtree with respect to coordinate axes was developed. It is based on a notion of classes of symmetry and it can be extended to deal with octrees. However, instead of doing that, we developed a simpler, direct method based on octree traversal and location of symmetric "brothers". Two octree nodes are symmetric brothers, *e.g.*, with respect to the z-axis if they are (a) of the same size, and (b) if the center of one of them is $C_1(x, y, z)$ then the center of the other one is $C_2(-x, -y, z)$. Instead of comparing coordinates of centers it is enough to compare position of nodes in the

octree using symmetry mapping tables. In the case of axial symmetry with respect to z-axis (see Fig. 1a) the corresponding table is given below;

d	0	1	2	3	4	5	6	7
$s(d)$	3	2	1	0	7	6	5	4

Pairs of symmetric octants (*e.g.*, 0 and 3, 1 and 2, 4 and 7, 5 and 6) are simultaneously traversed, with node d of one of them compared with node $s(d)$ of the other one. Beginning with $m_{sym} = 0$, comparison is done for all pairs of symmetric brothers using the following rules;

a. if at least one node is WHITE do nothing (proceed with next pair of nodes),

b. if at least one node is BLACK increase m_{sym} for double the mass of the other node, and

c. if both nodes are GRAY traverse their sons using the table above, and compare them using these rules.

Note that in our approach sd is always greater then zero, because only lines and planes passing through the centroid are considered, but for the same reasons it is not necessarily the maximum sd for a given object (for maximum sd lines and planes passing off the centroid have to be considered too).

It was mentioned earlier that eigenvalues of the inertia matrix represent variance of data along the corresponding eigenvectors. Hence, equality of two eigenvalues may indicate axial symmetry, equality of all three eigenvalues may indicate point symmetry and distinct eigenvalues may indicate bilateral symmetry. These relations can be used as hints for which type of symmetry to check first. However, as noted before (in Section 3.1.), degenerative cases (2 or 3 eigenvalues equal) correspond to multiple axes and/or planes of symmetry. At the present state of the algorithm, when they appear indicated type of symmetry can be identified but, in a general case, multiple symmetry cannot.

3.4. Execution Time

Average execution time for the algorithm is

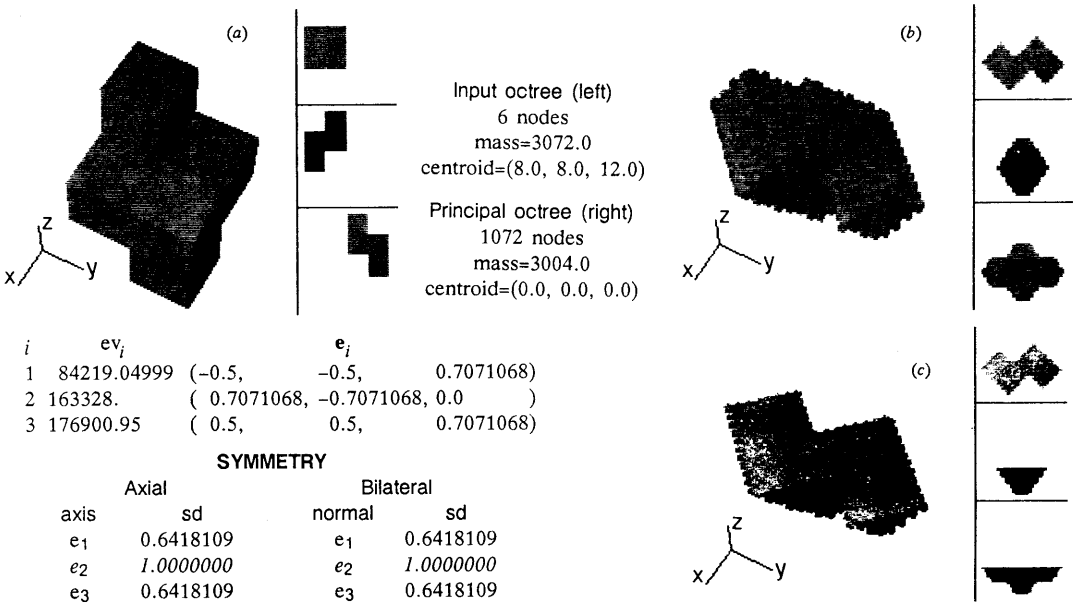
$$O\left(\left(2 + r_2 + c \cdot 4^{(r_2 - r_1)}\right) K\right)$$

where r_1 and r_2 are the resolutions of the input octree and the principal octree, respectively, K is the number of nodes of the input octree, and c is the number of symmetry evaluations in step 3 (up to 7). Three factors in the term multiplied by K reflect time requirements for steps 1, 2 and 3, respectively. In step 1, one octree traversal is done to compute the object's mass and centroid, and another one to compute central moments. Time for step 2 is given in section 3.2. In step 3, for each symmetry evaluation one traversal of the principal octree is done, and its number of nodes is proportional to its surface. The measure given here assumes an arbitrary position of the object, *i.e.* such that the number of octree nodes before and after the transform is almost unchanged. Taking in account that typically $r_2 - r_1 \leq 2$, and the nature of operations in step 2 (rotation) and step 3 (only traversal and comparison), the dominant component is the time necessary for step 2.

4. IMPLEMENTATION AND RESULTS

The algorithm was implemented in *C* on a workstation. Octrees are encoded using a pointer-less scheme known as the linear octree [6]. An octree viewer was implemented using back-to-front painters algorithm. Among several tests performed with synthetic objects, four results are given here.

In the first three tests, octree resolutions were r_1=3 and r_2=6. Note that because these objects belong to the class of small objects (only several nodes at r_1=3) r_2 must be set greater than r_1 (valid results were obtained with r_2=4 too, but higher resolution was chosen to improve display quality). The program report consists of the eigenvalue problem solution (eigenvectors originate at the centroid of an input object) followed by computed symmetry degrees for each symmetry type. Complete results of test 1 are shown in Fig. 2. Three different types of symmetry were identified with sd=1. In test 2, one asymmetric object was processed (Fig. 3). The highest reported sd is 0.679. The object in Fig. 4a was used in both tests 3 and 4. In test 3, one axis and two planes of symmetry were identified (Fig. 4b). Note that this object has four planes of sym-



i	ev_i	$\mathbf{e}_i$
1	84219.04999	(-0.5, -0.5, 0.7071068)
2	163328.	(0.7071068, -0.7071068, 0.0)
3	176900.95	(0.5, 0.5, 0.7071068)

SYMMETRY

Axial		Bilateral	
axis	sd	normal	sd
e_1	0.6418109	e_1	0.6418109
e_2	*1.0000000*	e_2	*1.0000000*
e_3	0.6418109	e_3	0.6418109

Point: *sd=1.000000*

Fig. 2d.

Fig. 2. Results of test 1. (*a*) An input, and (*b*) the principal octree with accompanied data. (*c*) Symmetric half with respect to bilateral symmetry. Each octree (*a*–*c*) is shown in perspective (left) and three orthographic projections (right; top to bottom along *z*, *y* and *x*-axis). (*d*) The program report (ev—eigenvalues, **e**—eigenvectors; the highest *sd*'s are given in italics).

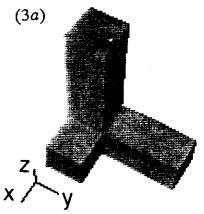


Fig. 3. Results of test 2. (*a*) An input octree, and (*b*) the program report.

i	ev_i	$\mathbf{e}_i$
1	151097.1779	(-0.0693103, -0.3754182, 0.9242604)
2	382665.3812	(-0.3260622, 0.884128, 0.3346657)
3	459188.2192	(0.942804, 0.2781706 0.1836888)

SYMMETRY

Axial		Bilateral	
axis	sd	normal	sd
e_1	0.3633315	e_1	0.4119620
e_2	0.3275573	e_2	0.4320850
e_3	0.2839575	e_3	*0.6791503*

Point: sd=0.227501

Fig. 3b.

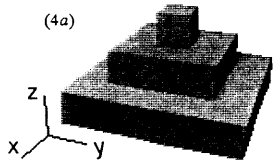


Fig. 4. Results of tests 3 and 4. (*a*) The object used in both tests. (*b*) The program report for test 3 (no quantization error). (*c*) The program report for test 4 (input octree rotated first).

i	ev_i	$\mathbf{e}_i$
1	72715.88309	(1, 0, 0)
2	72715.88309	(0, 1, 0)
3	120288.	(0, 0, 1)

SYMMETRY

Axial		Bilateral	
axis	sd	normal	sd
e_1	0.6142857	e_1	*1.0000000*
e_2	0.6142857	e_2	*1.0000000*
e_3	*1.0000000*	e_3	0.6142857

Point: sd=0.614286

Fig. 4b.

i	ev_i	$\mathbf{e}_i$
1	71600.85098	(0.8940015, -0.4480626, 0.0011503)
2	72431.20813	(0.448062, 0.8940022, 0.0007697)
3	119731.9865	(-0.0013733, -0.0001727, 0.999999)

SYMMETRY

Axial		Bilateral	
axis	sd	normal	sd
e_1	0.5972034	e_1	*0.9490302*
e_2	0.5972034	e_2	*0.9490302*
e_3	*0.9598557*	e_3	0.6134416

Point: sd=0.604420

Fig. 4c.

metry. In test 4 the influence of noise was simulated. The same object was first rotated for 20° around the z-axis and the result was recorded in the octree with r=6, which contains a quantization error. The symmetry identification algorithm was applied to the transformed octree (not shown), and the principal octree was computed with the same resolution. Thus in this case r_1=r_2=6. Obtained *sd*'s and principal axes contain a small error (Fig. 4c).

Note the excellent performance of the algorithm for synthetic objects without noise (tests 1, 3). The simulation of noise in the input octree gave quite satisfactory results with only slight error in *sd* and position of axes (test 4). Similar type of noise can be expected for octrees of real objects constructed using, *e.g.*, camera images. In such environments a certain threshold *th* for *sd* has to be established. Symmetry is claimed if $sd \geq th$. As noted before, the symmetry degree described here is not the maximum one for the given object, which can be considered as deficiency if such *sd* is needed, but this is actually a merit which facilitate classification of asymmetric objects (test 2). At present the algorithm does not support identification of *all* axes and planes of symmetry of an object (test 3).

5. CONCLUSION

An algorithm for identifying symmetry of an arbitrary 3D object is presented and implemented. To the best of our knowledge this is the first such algorithm reported so far. The only prerequisite is that the object is given by its octree. The algorithm has been verified by selected examples using synthetic objects.

Identification of all axes and planes of symmetry remains to be solved. To do this, the relations between a symmetry of cross-sections of an octree with positions of multiple planes of symmetry are studied now. The algorithm will be also tested with octrees of real objects. It can be easily adjusted to deal with 2D images represented by quadtrees.

Symmetry is a very important feature of objects, and the ability to identify this feature will find numerous applications in computer vision, intelligent CAD systems and in some other related fields.

REFERENCES

[1] N. Alexandridis and A. Klinger, Picture decomposition, tree data structures and identifying directional symmetries as node combination, *CGIP* **8**, 1978, 43–77.

[2] M. J. Attalah, On symmetry detection, *IEEE Trans. Computer* **C-34**, 1985, 663–666.

[3] H. H. Chen and T. S. Huang, A survey of construction and manipulation of octrees, *CVGIP* **43**, 1988, 409–431.

[4] C. H. Chien and J. K. Aggarwal, Identification of three dimensional objects from multiple views using quadtrees/octrees, *CVGIP* **36**, 1986, 256–273.

[5] S. A. Friedberg, Finding axes of skewed symmetry, *CVGIP* **34**, 1986, 138–155.

[6] I. Gargantini, Linear octtrees for fast processing of 3D objects, *CGIP* **20**, 1982, 365–374.

[7] R. C. Gonzales and P. Wintz, *Digital picture processing*, chap. 3, Addison-Wesley, Reading, Mass., 1987.

[8] T. Kanade and J. R. Kender, Mapping image properties into shape constraints, *Proc. IEEE Workshop Pic. Data Descript. Managem.*, 1980, 130–135.

[9] T. Kanade, Recovery of the three-dimensional shape of an object from a single view, *Artificial Intelligence* **17**, 1981, 409–460.

[10] A. Klinger et al., Minimal quadtrees, *Proc. 7th Intl. Conf. Patt. Rec.*, 1984, 814–816.

[11] G. Marola, On the detection of the axes of symmetry of symmetric and almost symmetric planar images, *IEEE Trans. PAMI* **11**, 1989, 104–108.

[12] G. Marola, Using symmetry for detecting and locating objects in a picture, *CVGIP* **46**, 1989, 179–195.

[13] H. Samet and R. E. Webber, Hierarchical data structures and algorithms for computer graphics, Part I: Fundamentals, *IEEE CG&A* **8**(3), May 1988, 48–68.

[14] ——, Hierarchical data structures and algorithms for computer graphics, Part II: Applications, *IEEE CG&A* **8**(4), July 1988, 59–75.

[15] K. R. Symon, *Mechanics*, chap. 10, Addison-Wesley, Reading, Mass., 1965.

[16] J. Weng and N. Ahuja, Octrees of objects in arbitrary motion, *CVGIP*, **39**, 1987, 167–185.

[17] E. Yodogawa, Symmetropy, an entropy-like measure of visual symmetry, *Perception & Psychophysics*, **32**(3), 1982, 230–240.

[18] L. Zusne, Measures of symmetry, *Perception & Psychophysics*, **9**(3B), 1971, 363–366.

盛光印刷所