Modeling of 3D object shape based on Superquadrics and Z-Buffer Algorithm

Dae-Hyun Kim, D. H. Hyeon, S. H. Lee, J. S. Choi
Dept. of Image Eng., Graduate school of Advanced Imaging Science, Multimedia, and Film, Chung-Ang University
221 Huksuk-Dong, Dongjak-Ku, Seoul 156-756 South Korea
Tel. 82-2-820-5412, Fax. 82-2-814-5404
e-mail : vante77@candy.ee.cau.ac.kr

Abstract : Superquadrics can represent various and complex 3D objects with only some parameters(size, position, deformation etc.). So if we use both superquadrics and deformed superquadrics, we can also represent more realistic 3D objects which are existed in real world. In this paper we use the z-buffer algorithm and stencil buffer together because this is very useful when the superquadric primitives are combined. The fundamental ideas are illustrated with a number of tables and figures.

1. INTRODUCTION

Determining the identity, position and orientation of randomly placed objects in 3D space is important in industrial and navigational robotics. To acquire the aforementioned information reliably, a robust and efficient 3D object representation scheme is demanded. So far, many kinds of 3D object representation schemes havae been proposed. Of many different ways of representing 3D objects, surface-based scheme has been known to be very useful since range information which is inherently topographical description of the 3D surface in the scene is used more often as an input to the computer vision systems. However, as yet, any scheme cannot represent complex and realistic 3D object effectively well[1].

In this paper, we propose a robust and efficient method to represent 3D objects. We exploit CSG (Constructive Solid Geometry) tree whose primitives are represented by superquadrics and their deformations. We define the set operations required for constructing CSG tree efficiently by using the z-buffer algorithm and stencil buffer. Since superquadrics can represent a wide variety of complex 3D primitives, our scheme can very effectively model the complex and realistic 3D objects with a small number of primitives and the set operations.

In previous work, the CSG set operations were defined by implicit function. However if the models were more complex the implicit functions would be more complicated. Also since superquadrics is a object centered coordinate system, when some objects are combined we must consider the reference coordinate system.

In proposed method, we have used the z-buffer algorithm and the stencil buffer. It is very efficient and useful to represent 3D objects because they just compare the depth value of each object. Moreover it is possible to represent non-symmetric object if we deform and combine some superqadrics which are symmetric.

2. DEFINITION OF SUPERQUADRICS AND THEIR DEFORMATIONS

2.1 Definition of Superquadrics

Superquadrics is a family of parametric shapes which are extensions of quadrics and defined mathematically as follows[1][2].

$$X(\eta,\omega) = \begin{Bmatrix} a_1 \cos^{\varepsilon_1}(\eta)\cos^{\varepsilon_2}(\omega) \\ a_2 \cos^{\varepsilon_1}(\eta)\sin^{\varepsilon_2}(\omega) \\ a_3 \sin^{\varepsilon_1}(\eta) \end{Bmatrix} \tag{1}$$

a_1, a_2, a_3 : superquadr ic size in x, y, z

$\varepsilon_1, \varepsilon_2$: squareness in the latitude and longitude

$-\pi/2 \le \eta \le \pi/2, -\pi \le \omega \le \pi$

2.2 Superquadric Inside-Outside Function

Equation (1) is a parametric equation of a superquadric surface. By eliminating parameters η and ω, using equality $\cos^2(\alpha)+\sin^2(\alpha)=1$, we get the following implicit equation

$$\left(\left(\frac{x}{a_1}\right)^{2/\varepsilon_2} + \left(\frac{y}{a_2}\right)^{2/\varepsilon_2}\right)^{\varepsilon_2/\varepsilon_1} + \left(\frac{z}{a_3}\right)^{2/\varepsilon_1} = 1 \tag{2}$$

Based on this implicit equation of the superquadric surface we define the following function

$$F(x,y,z) = \left(\left(\left(\frac{x}{a_1}\right)^{2/\varepsilon_2} + \left(\frac{y}{a_2}\right)^{2/\varepsilon_2}\right)^{\varepsilon_2/\varepsilon_1} + \left(\frac{z}{a_3}\right)^{2/\varepsilon_1}\right)^{\varepsilon_1} \tag{3}$$

We refer to this function as the inside-outside function(or implicit funtion) because it determines where a given point lies relative to the superquadric surface.

2.3 Deformations of Superquadrics

2.3.1 Tapering

Tapering deformation along axis z is

$$X = f_x(z)x, Y = f_y(z)y, Z = z \tag{4}$$

where X, Y, Z are the components of the surface vector **X** the deformed superquadric, f_x and f_y are the tapering functions in the x- and y-axes of the object centered coordinate system, and x, y, z are the components of the original surface vector **x**.

2.3.2 Twisting

Twisting is the deformation of rotating points around a reference axis by angle θ which is the function of the reference coordinate and twisting parameter t_ω.

$$X = x\cos\theta - y\sin\theta,\ Y = x\sin\theta + y\cos\theta,\ Z = z \qquad (5)$$

2.3.3 Bending

Bending is deformed by rotating points on the superquadrics around a bending axis by an angle γ which is a function of bending parameter K.

$$\begin{aligned} X &= x + \cos\alpha\cdot(R-r)\ ,\ Y = y + \cos\alpha\cdot(R-r) \\ Z &= \sin\gamma\cdot(K^{-1} - r) \end{aligned} \qquad (6)$$

2.3.4 Combined Deformation

Deformation can be applied successively. Any combination of three types of deformation is feasible. It is natural to apply in the order of tapering, twisting and bending since one can envision the shape after deformation easily. The results are in Fig.1

tree are boolean operations or rigid motions and leaf nodes are solid primitives. So, the algorithm for CSG traverses the tree and classifies the set-membership.

4.1 Previous work

If F1 and F2 are implicit function of two superquadrics SQ1 and SQ2 respectively then the union, difference and intersection between two superquadrics are defined in (8) and the results are shown in Fig. 2[1].

$$\begin{aligned} SQ_1\cup SQ_2 &= \{(F1(x,y,z)=1\ \&\ F2(x,y,z)\geq 1)\ \text{or}\ (F2(x,y,z)=1\ \&\ F1(x,y,z)\geq 1)\} \\ SQ_1\cap SQ_2 &= \{(F1(x,y,z)=1\ \&\ F2(x,y,z)\leq 1)\ \text{or}\ (F2(x,y,z)=1\ \&\ F1(x,y,z)\leq 1)\} \\ SQ_1 - SQ_2 &= \{(F1(x,y,z)=1\ \&\ F2(x,y,z)\geq 1)\ \text{or}\ (F2(x,y,z)=1\ \&\ F1(x,y,z)\leq 1)\} \end{aligned} \qquad (8)$$

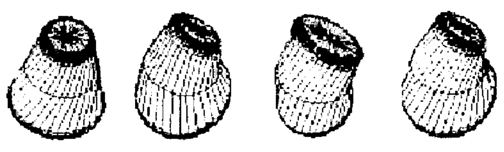
However, since the superquadrics have a object centered coordinate system, when we combine some superquadric primitives, which are differently oriented, translated and deformed as show in Fig. 3, we have to

$$X = D(x) = LO\cdot BE\cdot TW\cdot TA\cdot x =$$

$$\begin{bmatrix} r11 & r12 & r13 & \Delta X \\ r21 & r22 & r23 & \Delta Y \\ r31 & r32 & r33 & \Delta Z \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & \cos\alpha\cdot(R-r) \\ 0 & 1 & 0 & \sin\alpha\cdot(R-r) \\ 0 & 0 & \sin\gamma\cdot(1/k-r) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} f_x(z) & 0 & 0 & 0 \\ 0 & f_y(z) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad (7)$$

LO : rotation&translation ***BE*** : bending ***TW*** : twisting ***TA*** : tapering ***x*** : superquadric surface point

consider the reference coordinate system. Also more complex object is represented more complicated the implicit function is. Therefore the cost will be expensive.



(a) (b) (c) (d)

Fig. 1 Superquadrics generated from a cylinder by combination of deformations

(a) Tapered and twisted cylinder
(b) Tapered and bended cylinder
(c) Twisted and bended cylinder
(d) Tapered, twisted and bended cylinder

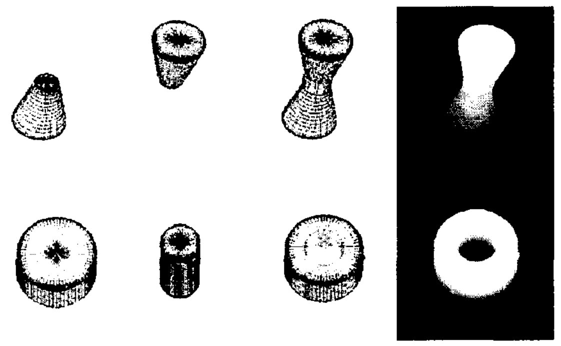


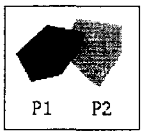
Fig. 2 Combined superquadrics by CSG defined by implicit function

(a) Object 1 (b) Object 2
(c) Union(up) and difference(down)
(d) Depth images

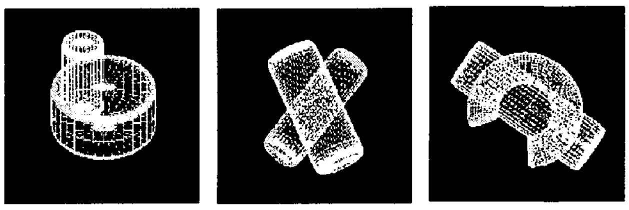
(a) (b) (c) (d)

3. Z-buffer algorithm

The z-buffer algorithm is one of the most commonly used routines. It is simple and easy to implement, and is often found in hardware. The idea behind it is uncomplicated: Assign a z-value to each polygon and then display the one that has the smallest value[5][6][7].



For example, consider these two polygons (P1 and P2). The computer would start with P1 and put its depth value into the buffer. It would do the same for P2. It will then check each overlapping pixel and check to see which one is closer to the viewer, and display the appropriate color.



(a) (b) (c)

Fig. 3 Deformed superquadrics are difficult to combine with implicit function

(a) Translation
(b) Rotation
(c) Deformation

4. SET OPERATION BETWEEN SUPERQUADRICS

CSG is represented by binary tree. Internal nodes of CSG

4.2 Proposed method

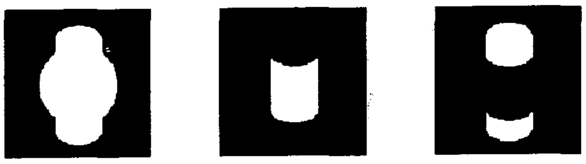
4.2.1 Union

The union of two solids, A and B, means rendering both the parts of A not enclosed by B and the parts of B not enclosed by A. To render the union of two solids, simply requires rendering them in either order using depth testing. The stencil buffer isn't required[3][4][5].

4.2.2 Intersection

The intersection of two solids A and B, means rendering the volume that is enclosed by both A and B. The algorithm to find this intersection can be thought of as the parts of B that are in A's volume, and drawing them, then finding the parts of A that are in B's volume, and drawing them[3][5]. It consists of two passes of the basic interior algorithm described Table 1 and Fig. 4-(b) shows this result.

The final result is a combination of the existing surfaces of the original primitives. No new surfaces are created. Since the visible results of the intersection will consist solely of the front surfaces of the primitives, only the front faces of a volume that are interior to the other need to be rendered.



(a) **(b)** **(c)**

Fig. 4 Interior and Exterior parts of object B
(a) Original image □ Object A
(b) Interior part image □ Object B
(c) Exterior part image

4.2.3 Difference

The difference between solids A and B can be thought of A intersected with the inverse of B. In other words, only draw the parts of A that aren't also parts of B. In terms of our stencil algorithm, this means drawing the front parts of A that are outside of B. In order to create a solid that looks closed, we also have to draw parts of B, specifically, the back part of B in A. It consists of two passes of the basic exterior algorithm described Table 2 and Fig. 4-(c) shows this result.

5. EXPERIMENTAL RESULT

Overall procedures are as follow. First we decide the number of primitives. Next we input the parameters of each superquadrics primitive. Then we set the required operation among the primitives. In last, we combine the primitives using the CSG algorithm that is defined in section 4.

In Fig. 5 and 6, we show the procedure of set operation, which is consisted of two and three primitives respectively. In Fig. 7, we show the various results of modeling. In addition to this, if we shade in the model surface with geometric information which is obtained from superquadrics, we are able to acquire the synthetic range image easily.

6. CONCLUSIONS

In this paper we describe the representation method of 3D object, which used both the 3D geometric information of superquadrics and z-buffer algorithm. In simulation, we can represent various 3D object using superquadrics and for more complex object, we have used the deformed superquadrics and CSG tree that is defined in section 2 and 4 respectively. So we are able to model the complex and realistic 3D object effectively well.

In previous work, they have used the set operation that is defined by superquadric implicit function. The implicit function is more complex when an object is moved, rotated or deformed. Especially for rotation, we have to consider the reference coordinate system because superquadrics has a object centered coordinate system. Therefore there are some restrictions to represent objects.

However, since the z-buffer algorithm compares only depth information of objects and then combines them, we need not consider the movement, rotation and deformation of objects. In addition to this, this proposed method is able to represent the asymmetric object using CSG tree although superquadrics is symmetric.

References

[1] S. C. Hwang, H. S. Yang, "3D object Representation using the CSG tree and Superquadrics", *KITE Journal of Electronic Engineering,* Vol.2, No.1 June 1991

[2] Frank Solina, Ruzena Bajcsy, "Recovery of Parametric Models from Range Images: The Case for Superquadrics with Global Deformations", *IEEE Trans. On Pattern Analysis and Machine Intelligence,* Vol.12, No.2, Feb.,1990

[3] Nigel Stewart, Goeff Leach, "An Improved Z-Buffer CSG Rendering Algorithm", *1998 Eurographics/Siggraph Workshop on Graphics Hardware,* pp.25-30

[4] N. Stewart, G. Leach, S. John, "A single Z-Buffer CSG Rendering Algorithm for Convex Objects", *Siggraph '99*

[5] Tom McReynolds, SGI, "Programming with OpenGL: Advanced Rendering", *Siggraph '96*

[6] Vera B. Anand, "Computer Graphics and Geometric Modeling for Engineers", WILEY

[7] A. Watt, F. Policarpo, "The Computer Image", ADDISON-WESLEY

STEP 1	Turn off and clear the stencil buffer Turn on the depth test Enable face culling and set back face culling Render A

⇩

STEP 2	Set the stencil buffer to increment if the depth test passes Render B Set the stencil buffer to decrement if the depth test passes Cull the front face of B Render B again

⇩

STEP 3	Set the stencil buffer to pass if the stencil value ***isn't zero*** Disable the stencil buffer updates Turn on back face culling; Turn off depth test Render A

Table 1. Basic interior algorithm

STEP 1	Turn off and clear the stencil buffer Turn on the depth test Enable face culling Set back face culling Render A

⇩

STEP 2	Set the stencil buffer to increment if the depth test passes Render B Set the stencil buffer to decrement if the depth test passes Cull the front face of B Render B again

⇩

STEP 3	Set the stencil buffer to pass if the stencil value ***is zero*** Disable the stencil buffer updates Turn on back face culling; Turn off depth test Render A

Table 2. Basic exterior algorithm

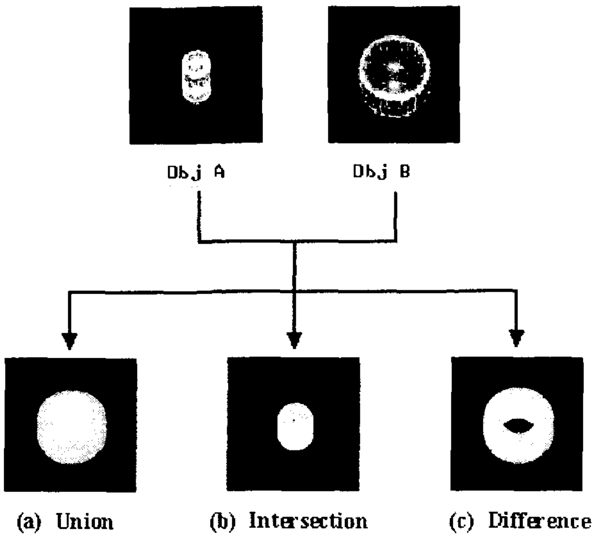


Fig. 5 Set operation between two objects

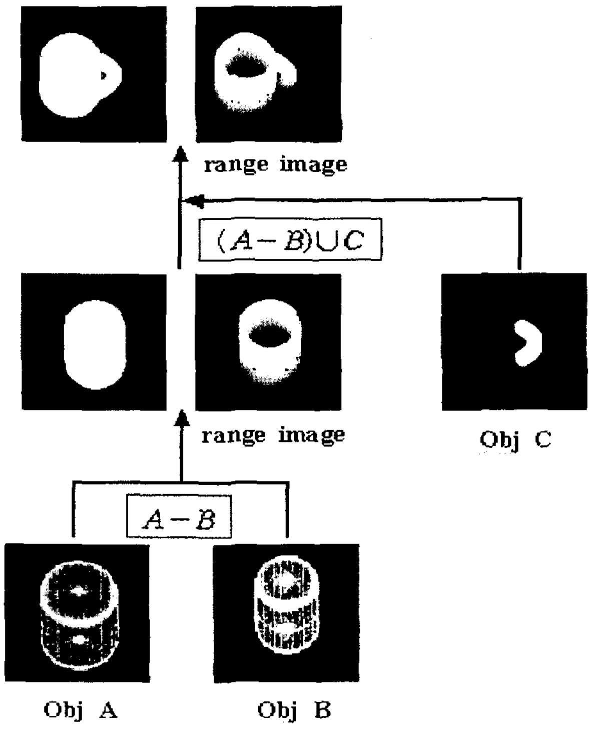


Fig. 6 Set operation between three objects

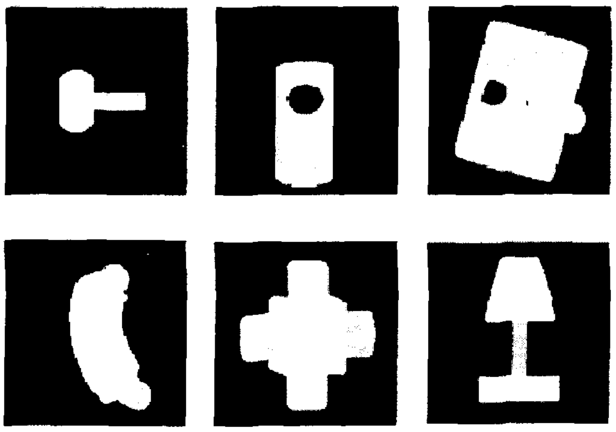


Fig. 8 A variety of combined superquadrircs