

Modeling of Rocks and Ornamental Garden Stones

by

Christopher J. Ellefson

B.S., Massachusetts Institute of Technology, 1995

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

THE FACULTY OF GRADUATE STUDIES

(Department of Computer Science)

We accept this thesis as conforming
to the required standard

The University of British Columbia

April 1997

© Christopher J. Ellefson, 1997

Abstract

Objects in nature are irregular in shape and the human eye is good at identifying regularities and symmetries when it does not expect them. Modeling these natural objects to make them look realistic can often be difficult and time-consuming. An example of these objects are rocks from a Chinese garden. These rocks have many large holes in them. Traditional modeling techniques do not lend themselves well to creating objects with complex and irregular shape. The holes in these rocks are also too large to be handled with bump or displacement maps.

This thesis discusses a technique to model ornamental rocks and objects with many large holes in them. A 3D material comprised of holes is made using an appropriate distribution for the holes. A stochastic process is used in creating the holes. Information gathered from pictures of real ornamental rocks is used for the distributions of the random variables used to create the holes. This is combined with the base shape of the object using constructive solid geometry. The result is a new object resembling the base shape but with many holes in it. The new shape created is converted to a format which

is usable by most rendering systems, in this case a series of triangles. This is done using a three-dimensional grid as an intermediate structure and then outputting triangles. The new object can then be imported into a rendering program.

We produced models and images from an implementation and they show that the method results in realistic looking images. These models are used in a virtual reconstruction of the Yuan Ming Yuan, a former imperial garden in China.

Contents

Abstract	iii
Contents	v
List of Tables	ix
List of Figures	xi
Acknowledgements	xiii
1 Introduction	1
1.1 Modeling of Natural Phenomena	1
1.2 Rocks	2
1.3 The Garden Project	4
1.4 Objectives	7
2 Related Work	9
2.1 Modeling of Natural Phenomena	9
2.2 2D stone textures	10

2.3	Texturing	13
2.4	Constructive Solid Geometry	14
3	Methodology	17
3.1	Overview	17
3.2	Base Shape	18
3.3	3D Material	18
3.4	Combining Base Shape with Material	19
3.4.1	Why not traditional CSG	19
3.4.2	Voxelization	21
3.4.3	Implementation	21
3.4.4	Optimizations	23
3.5	Polygonalization	26
3.5.1	Marching Cubes	26
3.5.2	Implementation	27
3.5.3	Modifications	29
3.6	Output Format	31
3.6.1	Implementation	31
4	Distribution of holes	33
4.1	Irregularities in Natural Phenomena	33
4.2	Stochastic Processes	34
4.3	Acquiring Distributions	35
4.4	Chi-Squared Test	35
4.5	Location of holes	39

4.6	Diameter	40
5	Results	47
5.1	Overview	47
5.2	Basic CSG operations	47
5.3	More than just a bump map	48
5.4	Distribution of holes	52
5.5	Level of detail	52
5.6	Speed and Size	59
5.7	Realistic looking rocks	59
5.8	Incorporating the Rocks into the Garden	62
6	Conclusion	64
6.1	Future work	64
	Bibliography	66

List of Tables

4.1	Distribution and density data for rock 1	44
4.2	Chi squared values for the diameters	44
5.1	Size and resolution of a modeled rock	55
5.2	Time depending on the number of holes	60
5.3	Time depending on the number of surfaces in the base shape .	60
5.4	Time depending on the size of the voxel set	60

List of Figures

1.1	Photographs of ornamental rocks from a Chinese garden	3
1.2	A painting of one of the views of Yuan Ming Yuan	5
1.3	A painting of an ornamental rock	6
2.1	Previous rock models	12
2.2	A CSG tree	15
3.1	Combining base shape with 3D material	20
3.2	A voxel	22
3.3	Testing for inside or outside	24
3.4	Special cases for ray intersection	25
3.5	A voxelized hole	25
3.6	14 basic patterns of cubes	28
3.7	Marching Cubes problem	30
3.8	New triangles	30
4.1	Sampled rock 1	36
4.2	Sampled rock 2	37

4.3	Sampled rock 3	37
4.4	Bins for centers of holes	39
4.5	Locations of holes	41
4.6	Holes projected onto a plane	42
4.7	Diameters, $f_3(D)$	45
4.8	Distribution functions, $F_3(D)$	46
5.1	A sphere subtracted from a cube	48
5.2	A sphere subtracted from a face	49
5.3	A dodecahedron subtracted from a cube	49
5.4	Many spheres subtract from a cube	50
5.5	No texturing	51
5.6	Texturing	53
5.7	Distribution of holes	54
5.8	Different resolutions of same rock	56
5.9	Different resolutions of same rock	57
5.10	Different resolutions of same rock with texturing	58
5.11	Comparison of real rock and virtual rock	61
5.12	A face rock	62
5.13	Another rock	63
5.14	A rock in the Yuan Ming Yuan	63

Acknowledgements

I would first like to thank my advisor Alain Forunier for all his invaluable help, inspirational talks, and the freedom he gave me. I could not have done it without him.

I would like to thank my second reader, Jack Snoeyink, for his useful comments and being so flexible with his time schedule.

I would like to thank Lifeng Wang and David Botta who are working on the Garden Project. Their work on the garden project provided a lot of the inspiration toward my thesis topic. Their help with obtaining photographs, and working with Alias was invaluable.

I would like to thank Marcelo Walter, for proof reading my thesis and helping with all sorts of stuff. I'd also like to thank to other the Imager Lab members, Paul Lalonde, Kevin Coughlan, Gene Lee, Roger Tam, and Rob Walker for helping out with LaTeX, Matlab and other random things.

I would like to thank my parents, for the strong foundation they gave me and the help my Dad gave me with proof reading.

I would like to thank all people in non-CS activities I have been involved with which have helped relieve stress and made my two years at UBC such a

wonderful experience. These include the Phi Deltis of BC Alpha, the Grizzlies Bearkeepers, the Japan Exchange Club, and Kogido.

Last but certainly not least I would like to thank Ruth Lim, my girlfriend who has given me support, put up with a long distance relationship and have given me modeavation to finish on time.

CHRISTOPHER J. ELLEFSON

The University of British Columbia

April 1997

Chapter 1

Introduction

1.1 Modeling of Natural Phenomena

With computer modeling, objects produced by natural phenomena are some of the most difficult types of objects to model. These are objects which are not man-made or artificially produced. They are things one finds in nature, such as trees, water, animals, and rocks. Computer modeling generally uses geometric shapes, or mathematical equations to represent objects. These shapes are usually somewhat regular or symmetrical. Objects such as tables and chairs can easily be represented with simple geometric shapes, such as spheres, cylinders or planes, but natural phenomena are much more difficult to model. They have many more irregularities in them, lack of symmetry, and have non-repeating patterns. The human eye is good at detecting regularities, symmetries and repeating patterns. When looking an artificial or man-made objects such as a chair it expects to see symmetry and patterns. When looking at natural phe-

nomena the human eye does not expect to see such patterns. These patterns and symmetries are quickly picked out and make the object not look natural. Because of this the human eye is more critical when it comes to looking at things from nature. This makes the task of modeling natural phenomena more difficult.

Natural phenomena encompass a wide variety of things which may be very different from each other. A tree is different from a rock, which is different from snow. Different types of things need to be approached with completely different ways of modeling because they are so different. At the same time the approach taken cannot be too specific. It would be inefficient to use a different modeling approach for every different object modeled. The modeling approach must be applicable to different natural phenomena which share certain common characteristics. Clouds and fire, for example, are dissimilar but they share certain characteristics such as they are both gaseous and semi-transparent, thus allowing the use of similar techniques to model them.

1.2 Rocks

In Chinese gardens, landscape architects have used rocks and stones with large irregular holes. The rocks are used as natural sculptures or are placed together to form embankments. Examples of such rocks can be seen in Figure 1.1. These stones and rocks are natural objects that can be interesting to model. What makes them particularly interesting is the fact they have so many large holes in them. These holes are not just surface features they exist throughout

the volume of the whole stone and often go all the way through. Surface modeling techniques cannot be applied because of the three dimensional nature of the holes. These holes produce extremely irregular shapes and do not have repeating patterns. Since holes may go all the way through the object, they produce complex topologies. These topologies add to the difficulty of the modeling task. There are also some simplifying factors that make rocks easier to model. A rock can be represented as one unit and it does not have separate pieces or any moving parts. The computer modeling of stones and rocks can take advantage of this simplicity.



Figure 1.1: Photographs of ornamental rocks from a Chinese garden [muse93]

There are different ways to make these shapes using traditional computer graphics modeling techniques. One could start out with a simple base

shape, such as an ellipsoid, and then deform it as if it were clay, until a good rock shape is arrived at. This technique may well end up taking a great deal of time and often requires a person with a great deal of artistic ability. Other techniques often used to produce irregularities in an image are bump or displacement maps. However the holes in the ornamental rocks found in a Chinese garden are too large to be handled by bump or displacement maps. These are surface techniques, which do not deal with the whole volume.

The holes of the rocks in a Chinese garden can be large and often have a significant impact on the over all shape of the rock. This characteristic suggests that the modeling of these holes in a rock should be done at the modeling stage and not at the rendering stage when many other techniques such as bump maps or 3D textures are often applied. The approach for the modeling of the rocks discussed in this thesis uses a 3D material comprised of holes that is applied to a base shape of a rock. When combined together they will produce a new object: the rock with holes in it.

1.3 The Garden Project

The rocks and stones created with this method are being used in the Garden Project [wang97]. whose goal is to make a virtual reconstruction of the Yuan Ming Yuan (Garden of Centered Wisdom). This was the garden of the Emperor of China, and is considered to be one of the greatest gardens to ever exist. Covering 780 acres the garden consisted of hills, trees, lakes, buildings, and filled with imperial art and treasures collected from throughout China. In

1860 Yuan Ming Yuan was destroyed by being burnt to the ground by British and French troops at the end of the Second Opium War. What we have left are paintings, maps, and stories that show and tell us what the garden used to appear. Examples of these paintings can be seen in Figure 1.2 and Figure 1.3.

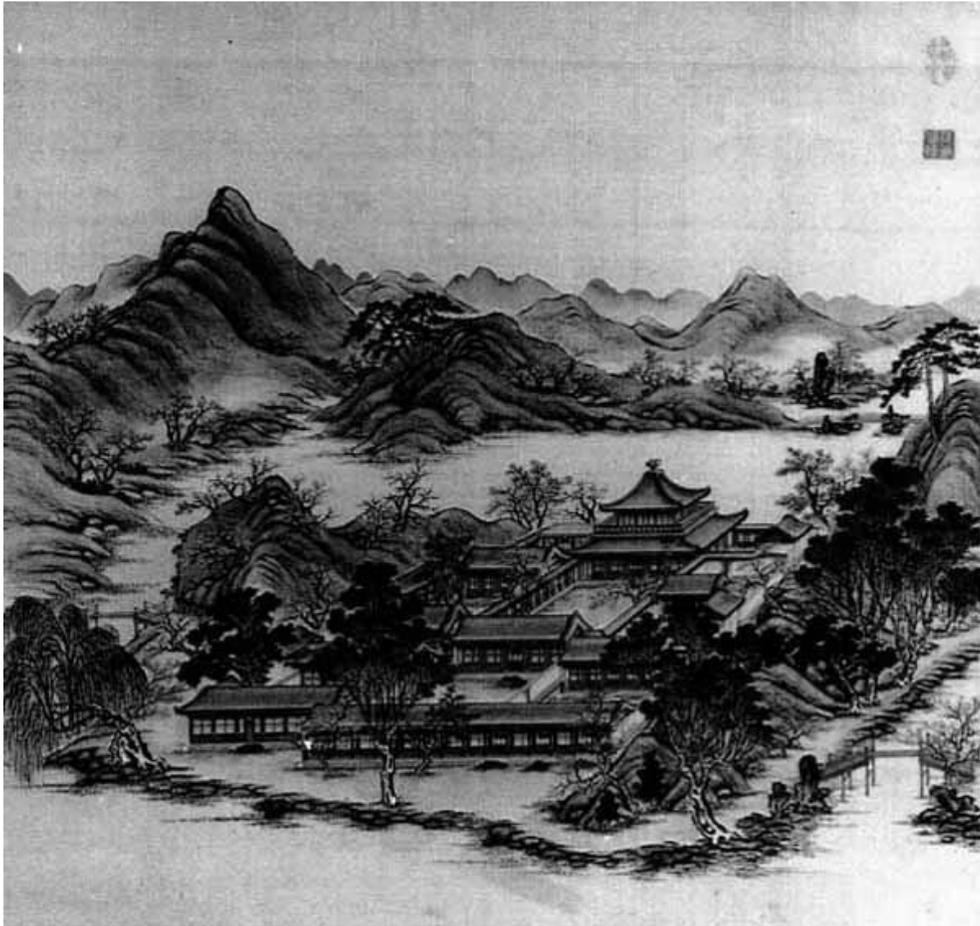


Figure 1.2: A painting of one of the views of Yuan Ming Yuan

One type of object which is found frequently in these gardens is rocks and stones. In Chinese gardens stones are often set as miniature mountains



Figure 1.3: A painting of an ornamental rock
[hung85]

separating various parts of the garden or along the banks of ponds or lakes. These rocks are usually hand-picked and placed to enhance the feel of the garden. Some of the most beautiful rocks which are found are placed on pedestals and used as natural sculptures. The most prized rocks are found in lakes and made of a soft material with large holes eroded away. These are the rocks whose modeling is described in this thesis.

1.4 Objectives

The main goal of this thesis is to present, from a computer graphics point of view, a modeling technique which can be used to model rocks and stones from a Chinese garden. This technique should ideally be extendible to the modeling of other objects with holes in them. The end result is a model which can be used to produce a realistic-looking picture of a rock. The geological properties of the rock, such as what minerals it is made of and how it was formed are not of concern here unless they have direct relevance to the rock's appearance. The modeling technique must also be designed so that the rocks can be used with the rest of the Garden Project. The ornamental rocks will be placed in garden scenes along with many other objects such as trees and buildings. The representation of the final object must fairly universal so it can be incorporated into a standard rendering package. The desired property is the ability to control the level of detail. A rock, as with any natural phenomena, might need an arbitrary level of display detail depending on how close to the virtual viewer it is in the picture.

Chapter 2

Related Work

2.1 Modeling of Natural Phenomena

There has been much work done on modeling of natural phenomena [four89]. Natural phenomena encompass a wide variety of modeling techniques. Some modeling techniques are specific to a given area and some are general modeling techniques which can be applied to different areas.

Plants and trees is one category where a great deal of work has been done where many different techniques have been used. These techniques include Oppenheimer's use of fractals, [oppe86], Prusinkiewicz's work with L-systems, [prus86, prus88, prus93] Reeves and Blau's use of particle systems [reev83], De Reffy's botanical model [reff88], and Weber and Penn's work using statistics of trees [webe95].

Modeling of gaseous phenomena such as clouds, fire, steam or fog is another area in which a lot of work has been done. Some of the earliest work

on clouds was done by Reeves [reev83] with the use of particle systems. Volume models can be seen in the work of Kajiya and Van Herzen [kaji84], Ebert and Parent [eber90], and Rushmeier [rush95]. Various lighting models include work by Max [max86], Rushmeier [rush87] and Nishita [nish96]. Work in specific areas of gaseous phenomena, such as flames, has been done by Chiba [chib94] and Cerimele [ceri91]. Recently, hybrid of previous methods have been use by Stam and Fiume which produce some good results [stam93].

Work has been done with the modeling of other natural phenomena. This includes water and waves, with work from Fournier and Reeves [four86] and Peachey [peac86]. Optical effects of water has be done by Watt [watt90], and Nishita and Nakamae [nish94]. Visualization of lightning has been been done by Reed and Wyvill [reed94]. Seashells have been modeled by Pickover [pick89] and Fowler etc al. [fowl92].

An essential characteristic of plants is that they have obvious structure. an essential characteristic of fluids is that they move. Stones neither have obvious structure nor do they move. What can be learned from these various modeling techniques is the use of random elements and some rendering methods.

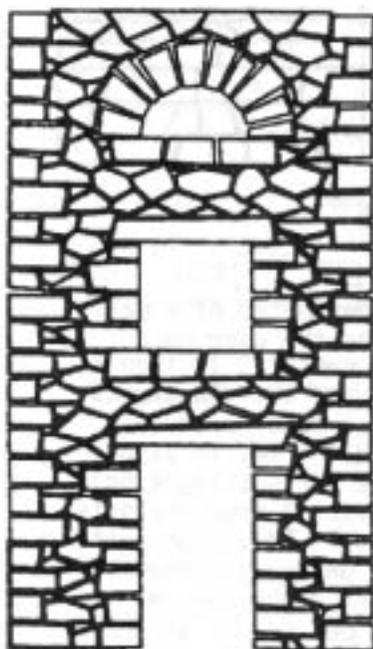
2.2 2D stone textures

Most of the previous work in the modeling of rocks and stones used 2D texture maps. Early work on stone walls includes Yessios' work for Architectural presentation drawings[yess79]. He used stochastic processes for generating

patterns for stone walls and pavements. He used algorithms called stone-by-stone, joint-by-joint and, stone-and-joint. These algorithms position stone shapes on a surface producing 2D, black and white drawings of stone walls. Figure 2.1 (a) shows one of Yessios' rock walls.

Since then many techniques have been added to texturing to improve the appearance of 3D images. Color and depth have been added to texturing, producing more realistic effects such as shadows and the appearance of roughness of the surface. Yessios' work on stone walls was expanded by Miyata many years later [miya90]. Miyata adds color, height, and roughness to the stones surface. A basic joint pattern, which is the shape of an individual stone, is created using a node and link representation. For each stone, a stone texture is created and then clipped to fit inside the area of the stone. The stone texture is created using a fractal subdivision of triangles. This process results in a reasonably realistic image of a stone wall. Figure 2.1 (b) shows one of Miyaya's rock walls.

These methods for generating stones are not applicable for the stones from a Chinese garden because they place 2D textures upon the surface of an object. Even though the use of bump or displacement maps adds a third dimension, they are still mapped to a surface which might result in stretching and deformation of the texture. The unique characteristic of the Chinese garden rocks is the large holes which requires a technique that takes into account the 3D characteristics of the rocks.



(a) stone wall by Yessios [yess79]



(b) A stone wall by Miyata [miya90]

Figure 2.1: Previous rock models

2.3 Texturing

The previous methods discussed for modeling rocks and stones are designed to deal with stone walls and a 2D texture or bump-map pasted on a 3D surface. One problem with this is the deformation of a texture when it is mapped to a different space. There are methods which get around this problem, such as the cellular textures described by Fleischer [fle95]. These cellular textures are designed to deal with surfaces that are composed of small pieces such as thorns or scales and thus are not applicable to such rocks. Advanced displacement map models, such as Pedersen's work on flow fields for displacement mapping [pede96], can produce holes on the surface of the object but cannot produce large holes which go all the way through the objects, such as the ones seen in the ornamental rocks in a Chinese garden.

Three dimensional texture maps is an area in which a reasonable amount of research has been done. Some of the earliest work with 3D textures was from Gardner who used them for the modeling of clouds [gard84]. Peachey used 3D textures as a way of removing some of the parameterization problems associated with 2D texturing. These 3D textures are mainly based on a band-limited noise function [peac85]. The same year Perlin used solid textures as part of his rendering package [perl85]. More advanced three dimensional texturing methods have been developed which require texturing throughout a region, not just at the surface. Perlin uses hyper-textures which used density functions to describe the three dimensional objects [perl89]. Work on solid noise synthesis was done also by Lewis [lewi89]. Solid textures have been

used in modeling of natural phenomena such as fur, by Kaijya [kaji89], and gaseous phenomenon by Ebert and Parent [eber90]. Filtering these textures is a challenge and Buchanan described some filtering methods for three dimensional textures [buch91]. Worley uses a cellular texture basis function for 3D texturing [worl96].

Some of these texturing techniques will be used to enhance the final appearance of the rock but do not suffice as the main modeling technique. Some of these methods can produce objects with complex shape and topology but do not give enough control over the final appearance of the object. These texturing techniques handle the finer detail but not detail as large as some of the holes in the ornamental rocks.

2.4 Constructive Solid Geometry

Constructive solid geometry (CSG) is an area of modeling that has been around for many years and can describe some objects efficiently [ricc73, fole90]. CSG represents objects by taking simple primitives and combining them through the use of boolean expressions such as union and intersection. It then produces a tree with the primitives as the leaves and the boolean operations as the nodes. The new object represented by the tree structure is then generally ray traced. An Example of a CSG tree can be seen in Figure 2.4. Work has been done with more advanced ways to combine objects other than simple boolean expressions, such as union and intersection, and work has also been done to deal with more complicated shapes. Wyvill incorporates free-form deformation

Chapter 3

Methodology

3.1 Overview

This section discusses a modeling technique to deal with objects with many large holes. The modeling technique starts with a base shape of the object without holes. The method will then create a 3D material, which will consist of a distribution of holes throughout a region of 3D space. The base shape will be combined with the material. This combination will occur while the objects are in a voxel representation. The result of the combination will be a new object which has the shape of the base shape with the holes in it. This new object will be converted to be represented by a set of polygons, and can then be used by any renderer.

3.2 Base Shape

The *base shape* is the basic shape of the object before the holes are applied. This shape can be any closed object. A closed object is one where given a point in 3D space it is not ambiguous whether the point is inside or outside of the object. In the implementation of this thesis, objects are represented by polygons in Inventor file format [wern94]. Polygons were chosen because they can represent a wide variety of shapes and are relatively simple to implement. Inventor file format was used because it is a fairly standard format on the SGI platform. The program could be easily extended to handle other object representations or file formats. The base shape of the rocks was created in Alias by deforming a sphere. This object is then polygonalised, and then saved in Inventor file format.

3.3 3D Material

A 3D material comprised of many holes is used the creation of the virtual rocks. This material is represented as a group of objects which have location, shape and size. Each one of these objects is a hole. A hole can be any closed object that can be ray traced, just like the base shape. In practice spheres are most often used. They are quick, simple and are quite similar to the shape of the holes in the real rocks. The program can also use polygonal objects as holes which can be imported the same way as the base shape. It could easily be expanded to handle other shapes or representations, such as parametric

surfaces. These holes can be placed at random throughout the material. Parameters such as number of holes, average size, and different distribution functions can be given to create different material. Different distribution curves can be used to produce varied effects on the appearance of the rocks. The holes can also be individually placed in the material with a specific location and size. Different distribution curves can be used to produce varied effects on the appearance of the rocks. The distribution functions will be discussed in more detail in Chapter 4.

3.4 Combining Base Shape with Material

Once both the base shape and the material are created they are combined together to produce a new object. This new object will be in the shape of what the final stone will look like.

3.4.1 Why not traditional CSG

A rock could be represented with Constructive Solid Geometry tree. CSG takes simple objects and combines them through the use of boolean expressions. It then produces a tree with the simple objects as the leaves and the boolean operations as the nodes. There are problems with this for the modeling of the rocks. The tree which would subtract the union of the holes from the base shape. This combination would be one layer deep and would be very wide. However, this would be a large and inefficient data structure. The main problem with traditional CSG is that the object is implicit and can not be

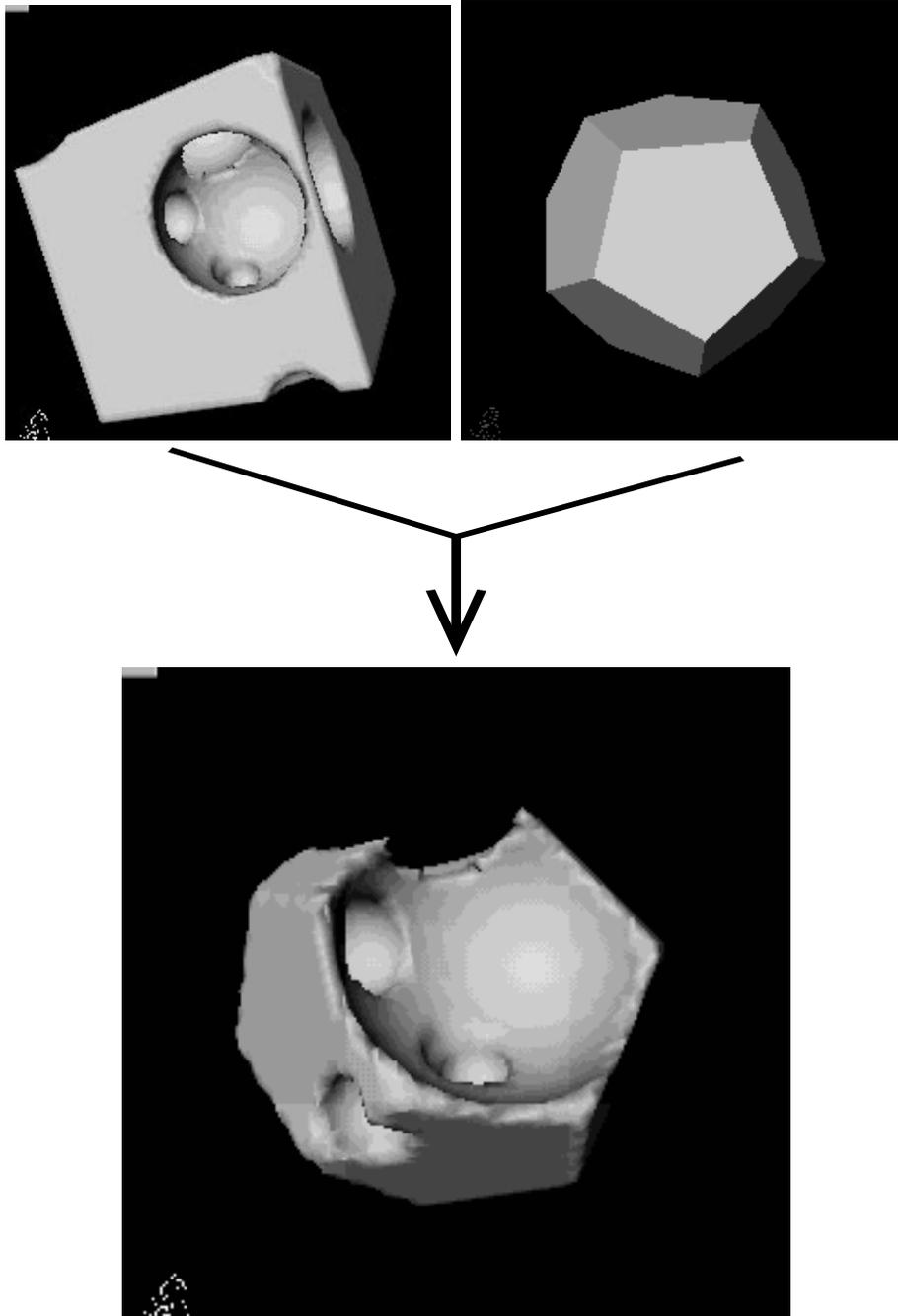


Figure 3.1: Combining base shape with 3D material

used by all types of renderers. The object needs to be an explicit form which is universal to different types of renderers.

3.4.2 Voxelization

Voxelization was the method that was chosen to handle the CSG operations. There are many reasons voxels were chosen. First, a voxel set will have constant size regardless of the complexity of the object. It would not matter then how many holes there were in a given object. Second, voxelization is simple, only local information for each voxel is required. It does not have to deal with all the different possible combinations of surfaces. Another approach which could have been used to get surfaces would be to do CSG operations on the surfaces themselves [laid86] but this would require many special cases, would be order dependent, and could require unnecessarily high resolution in some cases.

Voxelization also has the feature which allows the resolution to vary. When viewing an object up close, more detail is required to get a good looking picture. If the object is in the distance, higher resolution will require more rendering time. The resolution or size of the final object will vary depending on the size of the voxel set.

3.4.3 Implementation

The voxels are stored in a 3D array, and are evenly sized and spaced. Each voxel contains a bit called *inside* which is true if the center of the voxel is inside

the object and false if it is outside. There is also a three-dimensional vector whose range is between 0 and 1 in each direction, X Y and Z. This is the scaled distance that the closest surface is to the center of the voxel. The distance of 1 means the surface is at least one voxel length away. Figure 3.2 shows a two-dimensional example with four voxels. The arrows show the distance vectors of the voxels inside the object. The voxels are all initialized to be inside with

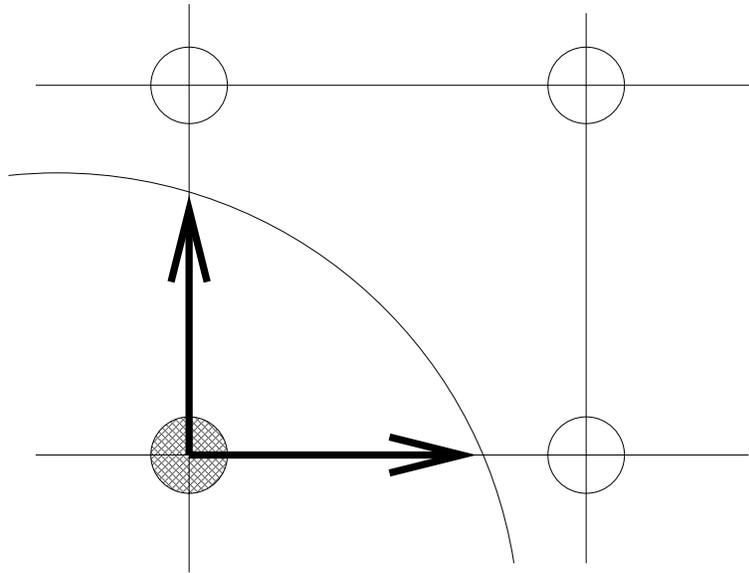


Figure 3.2: A voxel

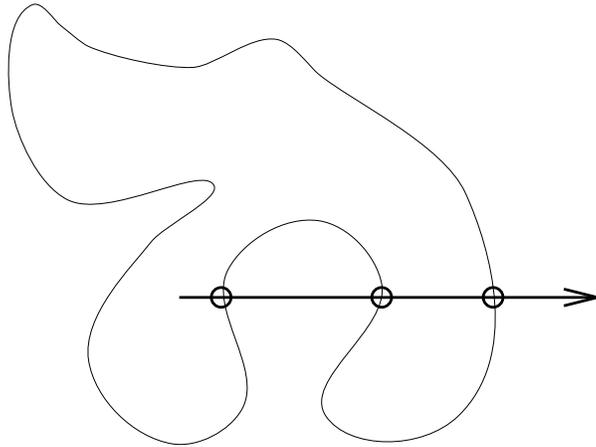
distance 1 in every direction. Each voxel is tested with its position compared to all the holes and the base shape. This is order-independent for both the voxels and the objects. For each voxel, the algorithm tests if the voxel is inside the base shape, and set the *inside* bit accordingly. The test for inside is done by shooting a ray from the center of the voxel and counting the number of intersections with the base shape. If the number of intersections is even, it is outside. If odd then it is inside. Figure 3.3 shows the inside/outside test.

Special cases where the ray hits the edge a surface are checked for. If a ray has two hits in the same location, as seen in Figure 3.4, the test will be redone with the ray moved slightly.

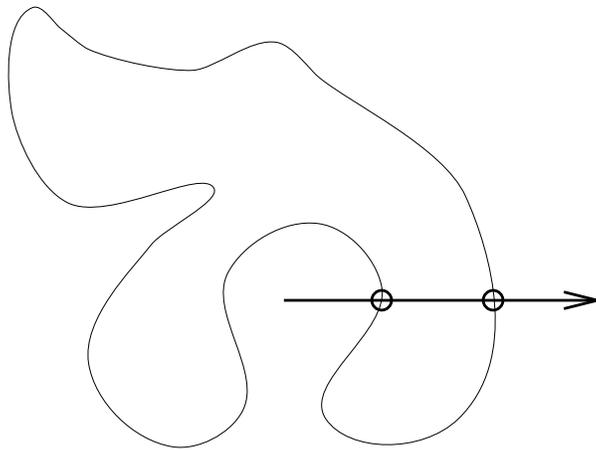
The distances to the closest surface in the X, Y and Z directions are computed. If the distance is within one voxel width then the distance vector is set accordingly. A similar process is done on the voxels with all the holes, except that the test for being inside or outside for each object is reversed. Inside the hole is outside the object therefore setting *inside* is reversed. A hole can only be subtracted away from the voxels. Therefore a voxel can only be changed from inside to outside and not the other way around. The distance vector of a voxel can only be decreased, because the closest surface is sought. If the distance of a new hole is greater than the current distance, the new hole already outside the object. After these operations have been performed the voxels will represent the shape of the final object. For references to the techniques used here see Graphics Gems [glas90, arvo91, heck94].

3.4.4 Optimizations

There are operations which can be done to speed up the voxelization process. With the method described in the previous subsection, each voxel performs an inside test and three ray distance tests for every surface on the base shape and all the holes. Many of these are unnecessary. The distances only need to be checked if there is a surface within a voxel width. There is only a surface close to a voxel if its inside bit has the opposite value as the voxel next to



(a) 3 intersections, inside



(b) 2 intersections, outside

Figure 3.3: Testing for inside or outside

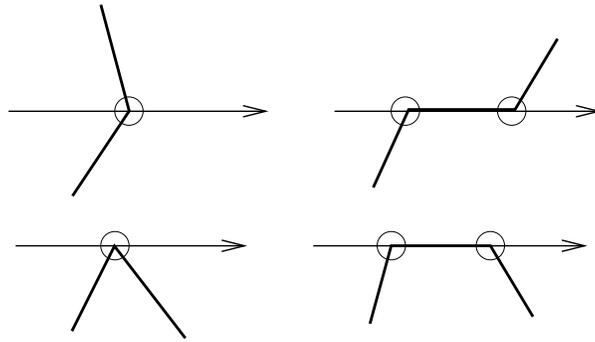


Figure 3.4: Special cases for ray intersection

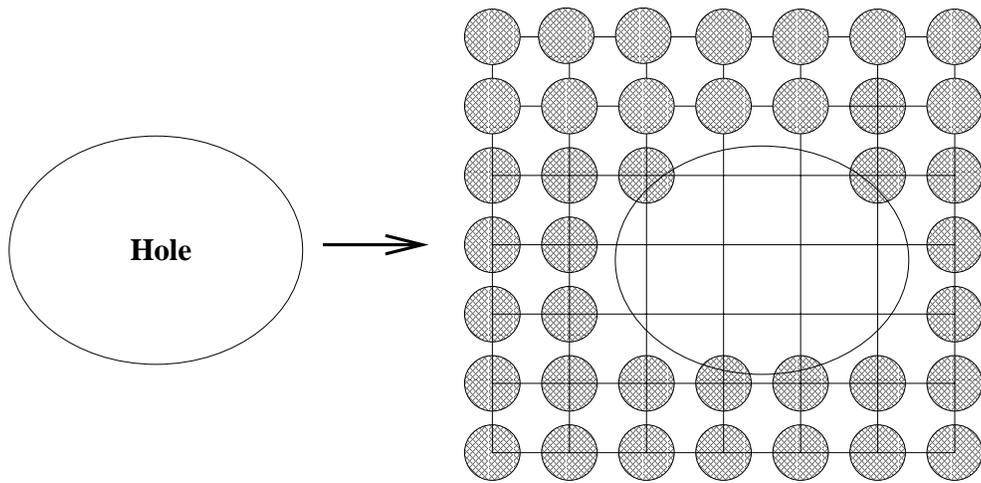


Figure 3.5: A voxelized hole

it. Once a voxel is outside it can not change to become an inside voxel again. To take advantage of these conditions the inside testing is done with all the voxels before the distance testing is done. Once a voxel has switched to become an outside voxel it is not tested with any of the other objects. The distance testing on the voxels are only done if the adjacent voxel in the direction of testing has the opposite inside value [jone96]. These improvements speed up the code by a factor of two to three.

3.5 Polygonalization

After the object is created in voxel representation it needs to be converted into a representation which is more universal. Polygons were chosen. They can be handled by any renderer and most graphics hardware is designed for polygons. With the modeling of rocks there is no need to have information regarding animation because rocks don't move. Often when more complicated representations are used for modeling it is for the movement of the object. Rocks can be represented simply without any additional information. There is also an efficient method for converting from volume data to polygons, which we describe next.

3.5.1 Marching Cubes

Marching Cubes [bloo87, lore87] is the algorithm used to convert the object from voxels to polygons. Marching Cubes uses groups of eight voxels which form a cube to produce polygons describing the surface within that volume.

This is done for each cube of voxels in the volume. There are 8 voxels in a cube and each voxel can be inside or outside. Therefore, for each cube of voxels, there are 256 configurations possible. This number can be reduced dramatically using symmetries. One is the complementary case, where all the voxel inside values are reversed. The triangles will be the same with only the front side reversed. The other reduction can be made when cubes can be the same with a rotation so they have the same configuration. These symmetries will reduce the number of configurations to 14 basic patterns as seen in Figure 3.6. One or more triangles will be created, in correspondence to the base pattern and the rotation.

3.5.2 Implementation

The voxels are stored in a 3D array. A cube of voxels can be formed by taking all combinations of x or $x+1$, y or $y+1$, and z or $z+1$, where the position of the voxel in the array is (x,y,z) . The configuration is stored as a one byte number where each bit is the inside value of a voxel in the cube. This configuration number is used in comparison with the 14 base patterns and their inverses. Rotations are done by passing the configuration number through a rotation function. This is a pre-computed table for the new location of each bit for the 24 different possible rotations. The configuration is tested against the base patterns, and their inverse for all rotations until a match is found. Once a matching pattern is found triangles are created. The three points of a triangle will always lie on the edge of the cube, directly between two adjacent voxels.

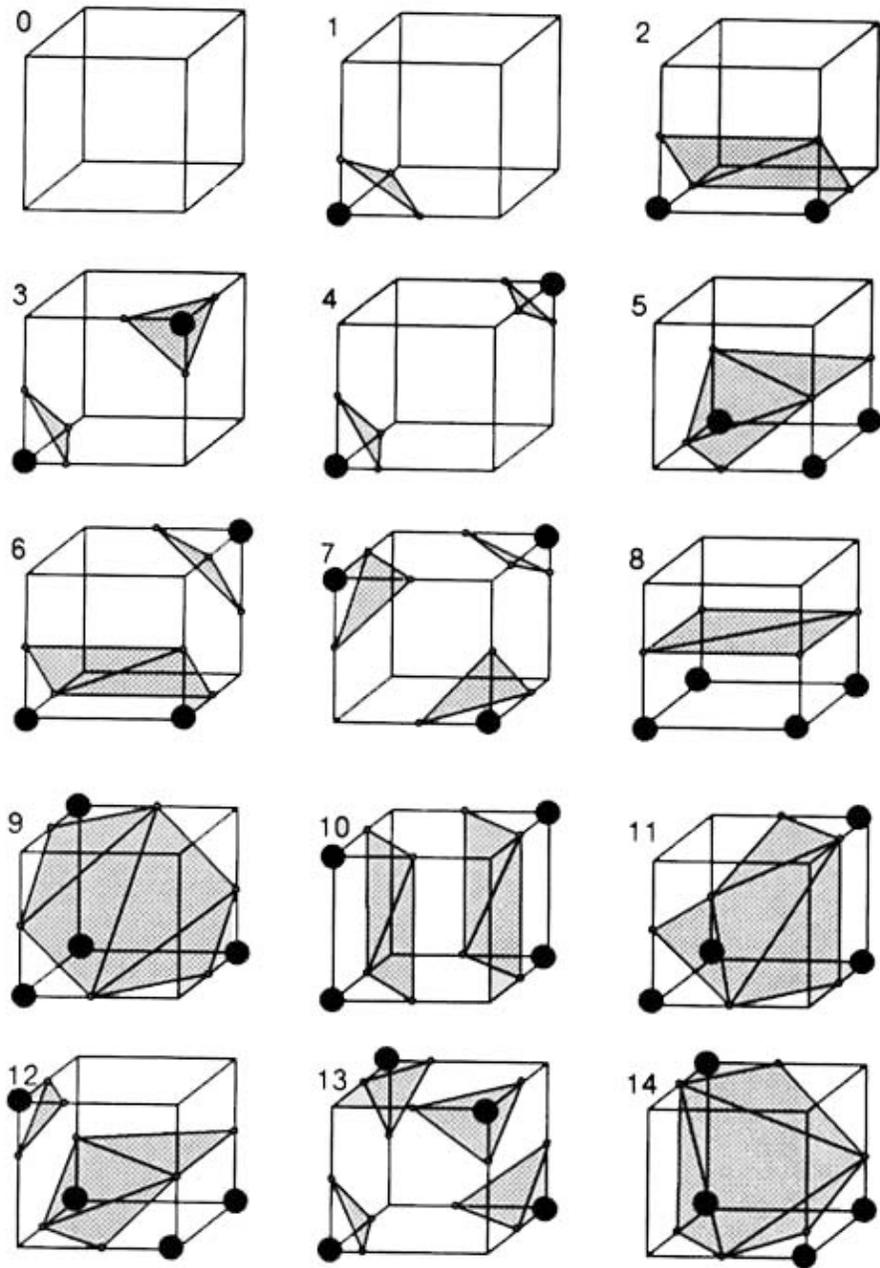


Figure 3.6: 14 basic patterns of cubes
[lore87]

The location of the point is calculated by taking the location of the lower voxel and an offset being one direction of the scaled distance vector stored in the voxel. These points are calculated before any rotation happens, then a rotation function is applied to them. This function is also a pre-computed table for the 12 edges and the 24 rotations. After this algorithm is run on the whole voxel set, the result will be a set of triangles. These triangles are independent of each other and do not have any connection information, or specific order.

3.5.3 Modifications

There were a few modifications made to the original Marching Cubes algorithm. There are a few combinations of configurations which resulted in holes in the surface, as in Figure 3.7. This is because the Marching Cubes algorithm only deals with local information. It is ambiguous whether the two triangles formed from pattern 3 come from the same surface or two different surfaces. Two triangles have been added to the base patterns 3 and 6 from Figure 3.6. These new triangles can be seen in Figure 3.8. This solves the problem of a hole appearing but causes a problem if they are supposed to be two different surfaces. The added triangles from a thin bridge between the two surfaces which is not supposed to be there. The flaw caused by a hole in the surface is also visually more apparent than the flaw caused by a bridge.

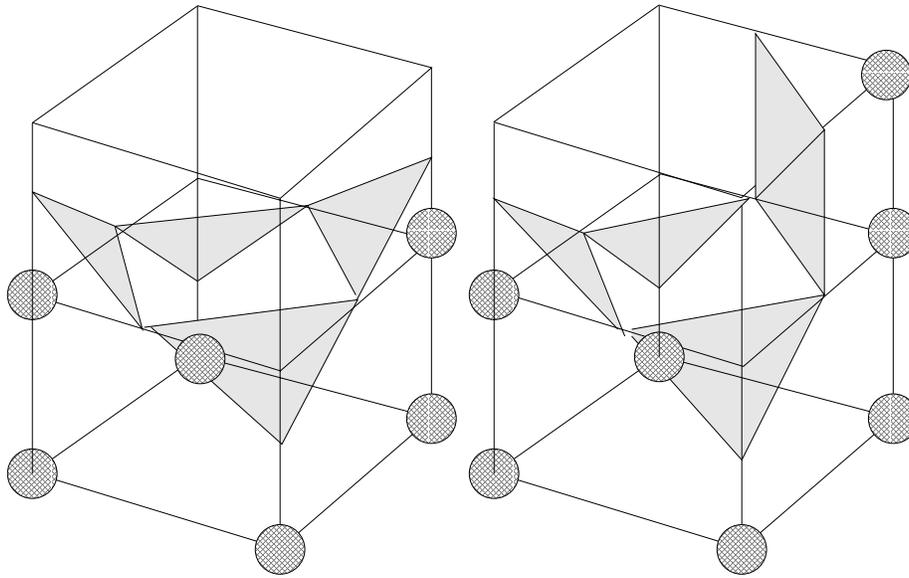
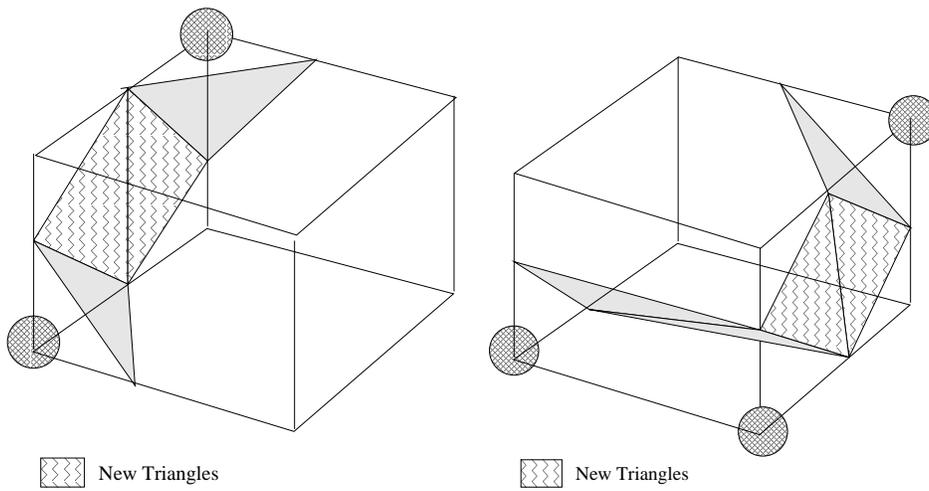


Figure 3.7: Marching Cubes problem



(a) pattern 3

(b) pattern 6

Figure 3.8: New triangles

3.6 Output Format

Inventor File format [wern94] was the format chosen for the output the objects. All the vertices are stored in an indexed list. The triangles are represented as a set of three index numbers corresponding to a list of vertices. The normal to the triangles are also stored in an indexed list and the vertices index to the normal.

3.6.1 Implementation

The vertices of the triangles are indexed using hashing. The coordinates of the vertices are used as the key for the hashing function. The number stored in the hash table is the index number, which is the order in which the vertices are inserted. The vertices is put into an array with the array index corresponding to the one in the hash table. The normal to the triangle associated with the vertices is also stored in an array. When a vertice is added more than once, which happens with adjacent triangles, the normals are summed, then averaged at the end. Triangles are also tested to determine if their three vertices are co-linear. If they are co-linear they are discarded.

Chapter 4

Distribution of holes

4.1 Irregularities in Natural Phenomena

One of the most difficult aspects of modeling natural phenomena is that they contain irregularities. If one has a chair (or some other manufactured object) another chair could look exactly like the first one. When making ten chairs, making nine copies of the original chair will suffice. With natural phenomena this does not work. If one wants ten maple trees one cannot just make one tree and copy it. No two trees are exactly alike. Each tree has a slightly different location, length, and width for each branch, etc. While each tree must be different they must also have a degree of similarity. If the differences are too great they won't look like the same type of tree or even a tree at all.

Stochastic modeling must be used in circumstances such as these [four80]. Stochastic modeling is just the use of randomness in the modeling. With the ornamental rocks from a Chinese garden the primary distinguishing character-

istic of the rocks are their holes. The size, frequency and location of the holes in the rock are the primary aspect of what make each rock unique. Therefore we purpose to model these characteristics with stochastic processes.

4.2 Stochastic Processes

The three aspects of the holes in the rocks which are going to vary are the diameter of a hole, location of the center of a hole, and the number of holes per unit volume. These holes are assumed to be spherical. The number of holes and a scaling factor for the diameters are set by the user. This control lets the user determine how large the rock is. This leaves the location and the diameter of the holes as the random variables in the modeling. When using random variables, their distribution must be taken into account. A distribution function for a random variable describes how that variable behaves. Each random variable X has a cumulative distribution function F where for each sample x :

$$F(x) = P(X \leq x).$$

This function lies in the interval $[0,1]$ and is non-decreasing. An example of a simple distribution function is the uniform distribution

$$F(x) = x$$

where $0 \leq x \leq 1$.

A uniform random variable has an equal probability of being located anywhere in the range 0 to 1. For the parameters of the holes of the rocks, distribution

functions will be determined and then used to produce the modeled rocks.

4.3 Acquiring Distributions

Real rocks from a Chinese garden and their holes are three dimensional and ideally three dimensional data would be gathered. This is not realistic to do because it would be too expensive requiring a CAT scan, MRI or some other similar process. Another way to collect data would be to cut a plane into a real stone and measure the holes. Ornamental stones are not available to be cut up. What is more easily available are pictures that are two dimensional. Pictures of rocks from a Chinese garden are used to collect statistical data on the characteristics of the holes. The location and size of the holes, are determined by a user from the picture. The user draws lines segments on the picture with an interactive program as estimates of the holes major axis. These lines are used to determine the center and the diameter of the holes. Data was collected from rocks 1, 2, and 3 which can be seen in Figure 4.1, Figure 4.2, and Figure 4.3.

4.4 Chi-Squared Test

The chi-squared test is used to help determine similarities between distributions. The test compares the probability density functions of two distributions. A probability density function of a random variable X shows its frequency of occurrence. The density function $f(x)$ is the derivative of the distribution

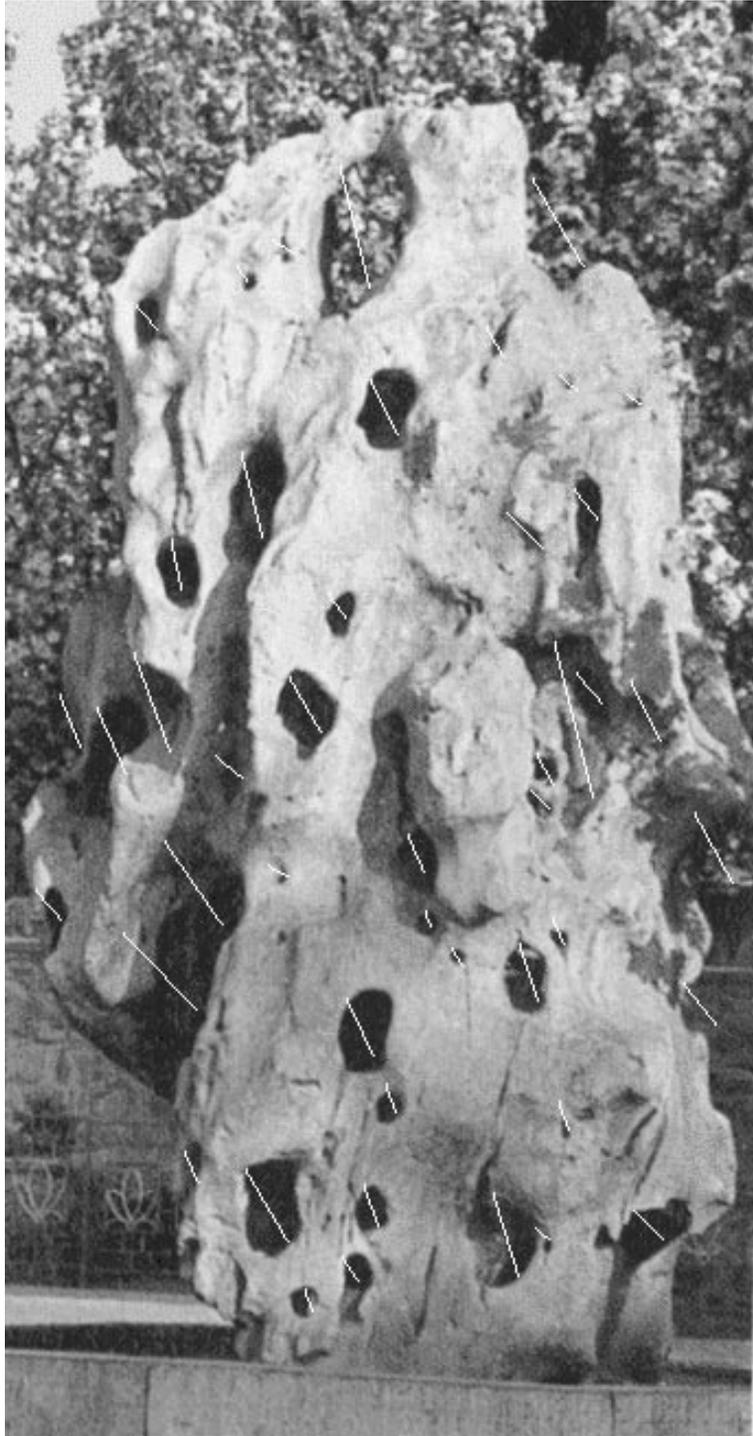


Figure 4.1: Sampled rock 1



Figure 4.2: Sampled rock 2



Figure 4.3: Sampled rock 3

function $F(X)$.

$$f(x) = dF/dx$$

$$F(X) = P(x_1 \leq X \leq x_2) = \int_{x_2}^{x_1} f(t)dt$$

A discrete density function is calculated for the sampled data by dividing the data into k bins and keeping track of frequency of occurrence for each bin. The expected values of X at the location of the bins are then calculated. These come from the density function of the random variable that the collected data is being compared to [alle90]. The chi-squared value is calculated by taking the sum of the scaled square of the difference of the observed elements O_i in bin i and the expected values E_i given by

$$E_i = N \int_{x_1}^{x_2} f(t)dt$$

where x_1 and x_2 are the bounds for bin i and N is the total number of elements.

$$\chi^2 = \sum_{i=1}^k \frac{(O_i - E_i)^2}{E_i}$$

χ^2 is then compared to the critical value $\chi_{m,\alpha}^2$, where α is the probability that a chi-squared variable with m degrees of freedom will be greater than $\chi_{m,\alpha}^2$. The degrees of freedom, m , is found from the number of independent parameters, p , used to generate the expected values E_i where

$$m = k - 1 - p.$$

If $\chi^2 < \chi_{m,\alpha}^2$ then we cannot reject the hypothesis that the two sets of data come from the same distribution.

4.5 Location of holes

The location of the center of holes can be found by taking the midpoints of all the lines collected. The distribution of these centers is tested to determine a distribution to be used for the location of the generated holes. This two-dimensional data can be extended to apply in three dimensions. The picture from which the data was collected is just a projection of a 3D rock onto a 2D surface. The two axes, X and Y, that were chosen are arbitrary depending on the camera position.

The centers of the holes are tested using the chi squared test to show whether they are positioned uniformly. The centers are placed in six bins of equal area as seen in Figure 4.4.

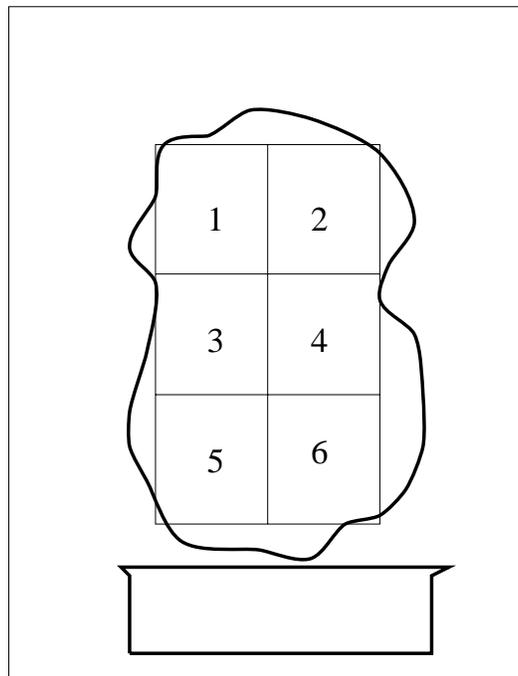


Figure 4.4: Bins for centers of holes

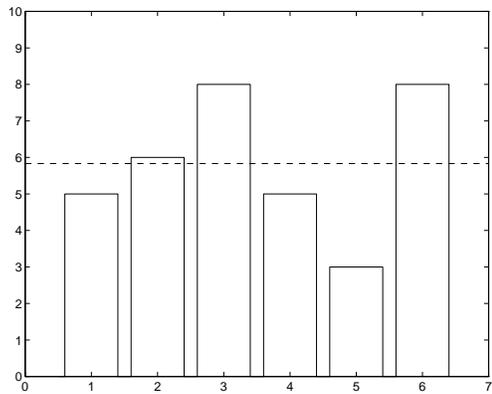
The values of the bins are compared to the average value of all the buckets. The chi squared values are 3.229, 5.0, and 1.323 respectively for the three sampled rocks. These values are all less than $\chi_{4,.05}^2 = 9.4877$ which means it is reasonable to assume they are uniformly distributed. The distribution of the buckets are shown in Figure 4.5. The distribution of the third axis Z in 3D space is assumed to be the same as that of X and Y. When creating the holes, a uniform distribution is used for generating the random locations. The uniform distribution is the standard distribution for the pseudo-random number generator, `drand48`, given with the C++ compiler.

4.6 Diameter

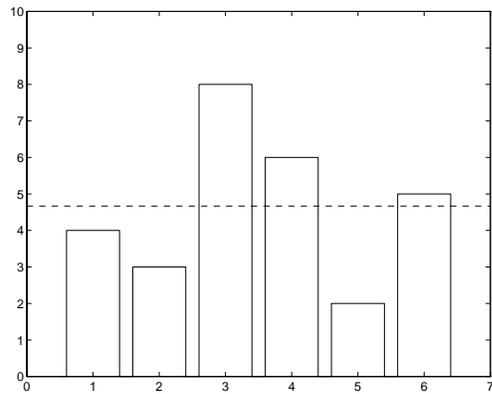
An approximation to the diameter of the holes can be found by taking the distance between the two end points of the line segment. The real holes and the generated holes are both three-dimensional but the data collected has only two dimensions. These two-dimensional holes are the projection of the intersection of the three-dimensional holes with the surface of the rock. A 2D illustration can be seen in figure 4.6.

There is a relationship between the distributions of the center and size of holes in two and three dimensions [serr82]. Given a distribution of disjoint balls whose centers appear in R^3 according to a stationary random process, let N_3 be the number of balls per unit volume, and $F_3(D)$ be the distribution of their diameters. Then

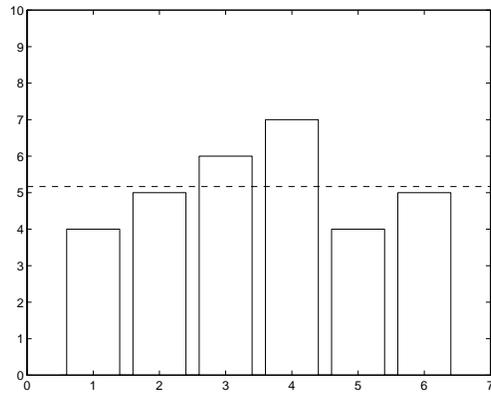
$$N_3 \times (1 - F_3(D))$$



(a) Rock 1 $\chi^2 = 3.229$



(b) Rock 2 $\chi^2 = 5.0$



(c) Rock 3 $\chi^2 = 1.323$

Figure 4.5: Locations of holes

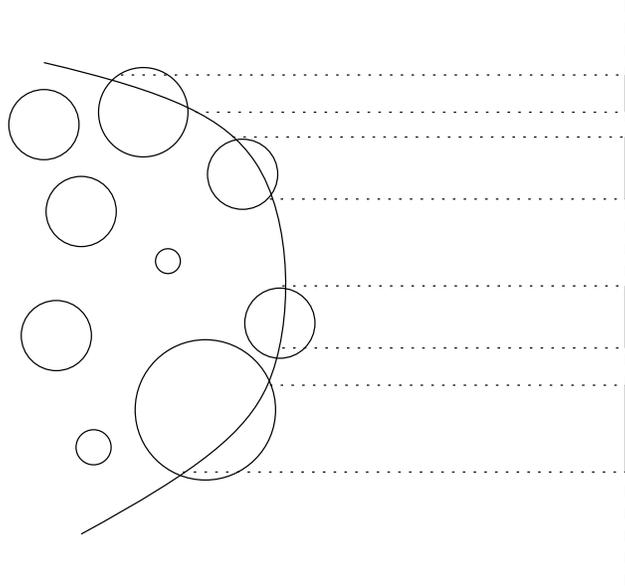


Figure 4.6: Holes projected onto a plane

is the number of centers of balls with diameters $\geq D$ per unit volume. Let N_2 be the number of circles per unit area on a plane which cuts through R^3 and $F_2(D)$ the distribution of the diameters of circles. The density function of the diameters is $f_2(d)$, which is equal to $F_2'(d)$.

Then we have the relationships:

$$N_3 = \frac{N_2}{\pi} \int_0^\infty \frac{F_2'(h)}{h} dh$$

$$N_3(1 - F_3(D)) = \frac{N_2}{\pi} \int_D^\infty \frac{F_2'(h)}{\sqrt{h^2 - D^2}} dh$$

These two equations can be combined to produce:

$$F_3(D) = 1 - \frac{\int_D^\infty \frac{F_2'(h)}{\sqrt{h^2 - D^2}} dh}{\int_0^\infty \frac{F_2'(h)}{h} dh}$$

The distribution function of diameters, $F_3(D)$, can thus be derived from $F_2(D)$.

When the circles from an intersecting plane are projected to another plane, the projected circles will become ellipses if the two planes are not parallel. The length of the diameter of a circle will be preserved as the major axis of the projected ellipse. Multiple planes can be used, all of which are projected onto a single plane. Ellipses projected from different planes would have different orientations, but the diameter of the original circle would still be the major axis. If the surface of the rocks in the photographs can be approximated as a collection of planes, then the statistics collected in 2D from the photographs will be valid. This is true if the curvature of the surface is low compared to the diameter of the holes intersected.

The discrete density function, $f_2(d)$, for the sampled diameters is calculated and used to produce its three dimensional counterpart. $f_2(d)$ is obtained by placing the sampled diameters into bins. The diameters have been scaled so the greatest diameter is 1. The integrals are approximated with summations.

$$\int_0^{\infty} \frac{F_2'(h)}{h} dh \simeq \sum_{\text{all bins}} \frac{1}{N} \times \frac{\text{num in bin} \times \Delta D}{D_{\text{center of bin}}}$$

$F_3(D)$ is calculated with the formulas given and differentiated to produce $f_3(D)$. The numbers calculated from the sample rock 1 can be seen in Table 4.1. The probability density functions given here have been multiplied by $N \times \delta D$. This is so that the numbers in the table correspond to the actual number of holes per bin.

The density function, $f_3(D)$ is compared to the uniform, normal, Poisson and exponential random distribution using the chi-squared test. The first bin or bins were not used in the chi-squared test because they were smaller than

Bin	1	2	3	4	5	6	7	8	9	10
D	.05	.15	.25	.35	.45	.55	.65	.75	.85	.95
$F_2(D)$	0	0.26	0.56	0.69	0.78	0.87	0.96	0.98	0.98	1
$F'_2(D) \times N \times \delta D$	0	12	14	6	4	4	4	1	0	1
$F_3(D)$	0	0.35	0.70	0.81	0.86	0.92	0.98	0.99	0.99	1
$F'_3(D) \times N \times \delta$	0	16.33	16.09	4.92	2.45	2.39	2.76	0.62	0	0.61

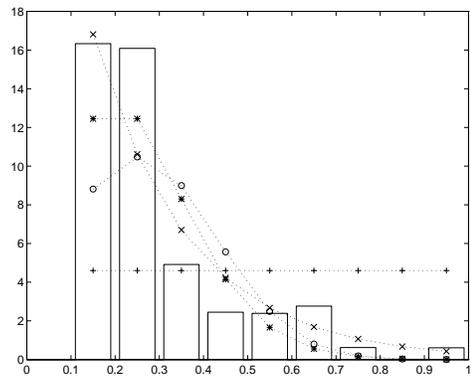
Table 4.1: Distribution and density data for rock 1

any of the sampled data. The chi-squared values for the three rocks can be seen in Table 4.2. The only chi-squared value which are below $\chi_{6,.05}^2 = 12.592$ for rock 1 and rock 3 and $\chi_{5,.05}^2 = 11.071$ for rock 2 is the exponential distribution. In Rock 2 the first two bins are empty. It is likely the distribution function used to generate the distribution of holes is exponential.

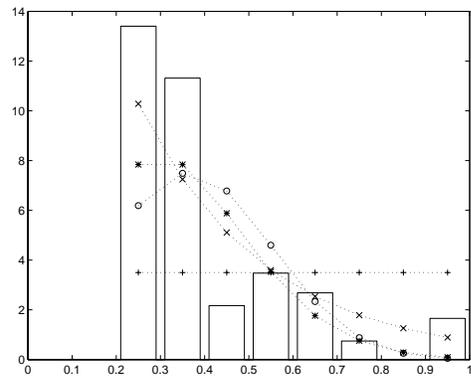
Distribution	Rock 1	Rock 2	Rock 3
Uniform	72.911	52.816	57.588
Normal	115.276	62.039	125.924
Poisson	55.989	34.346	258.785
exponential	5.714	7.462	10.872

Table 4.2: Chi squared values for the diameters

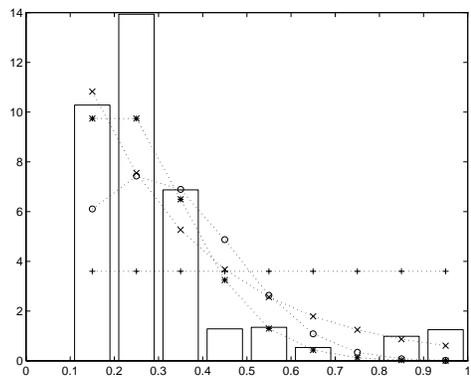
The distribution function of the collected data, $F_3(D)$ can be used to generate the diameter of the holes. The output from the standard uniform pseudo-random number generator is passed to the inverse of the new distribution resulting in random numbers of that distribution. The inverse function is calculated by intersecting horizontal lines with the distribution function, which is represented as a series of lines. The distribution functions $F_3(D)$ from the sampled data can be seen in Figure 4.8.



(a) Rock 1



(b) Rock 2

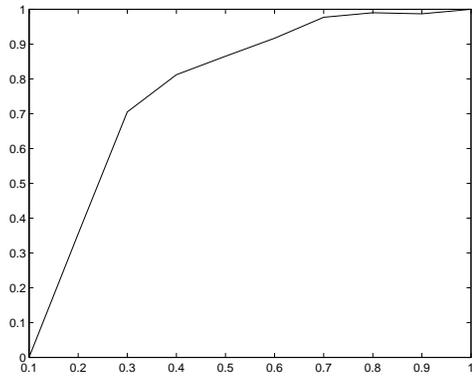


(c) Rock 3

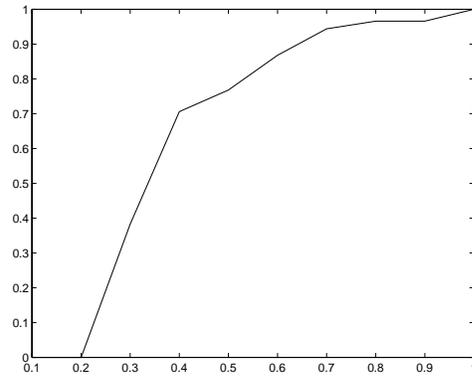
Data	Bar
Uniform	+
Normal	o
Poisson	*
exponential	x

(d) Key

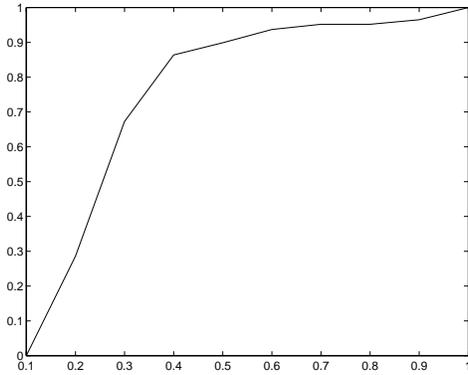
Figure 4.7: Diameters, $f_3(D)$



(a) Rock 1



(b) Rock 2



(c) Rock 3

Figure 4.8: Distribution functions, $F_3(D)$

Chapter 5

Results

5.1 Overview

Now we will describe the results of these methods. The algorithm described in Chapter 3 is able to place holes in arbitrary shaped objects, which generate a surface representation, for the final object. The use of stochastic processes, as described in chapter 4 allowed us to generate distribution of holes from photographs of real rocks.

5.2 Basic CSG operations

The operation of placing holes in an object is done through CSG operations while the object is stored in a voxel representation. The volumetric object is then converted into a surface representation. The implementation of this method can handle any closed polygonal object for the base shape and the

holes. A simple example of the CSG operation can be seen in Figure 5.1 where a sphere is subtracted from a cube. Figure 5.2 shows a more complex shape for the base shape and figure 5.3 shows a more complex shape for a hole. The jagged edges seen in Figure 5.2 are due to the Marching Cubes algorithm only taking into account local information. In some case it is ambiguous where the triangles should be placed. Cases 11 and 14, seen in Figure 3.6, are where these problems occur. The holes can also be randomly distributed throughout the space as seen in figure 5.4 where many holes are placed in a cube.

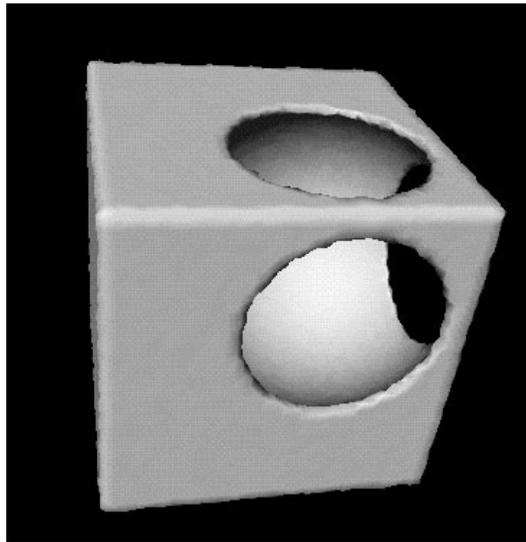


Figure 5.1: A sphere subtracted from a cube

5.3 More than just a bump map

One of the reasons a new technique for modeling holes was created is the fact that current texturing techniques by themselves are not adequate for handling the large holes of ornamental Chinese rocks. Bump maps and displacement



Figure 5.2: A sphere subtracted from a face

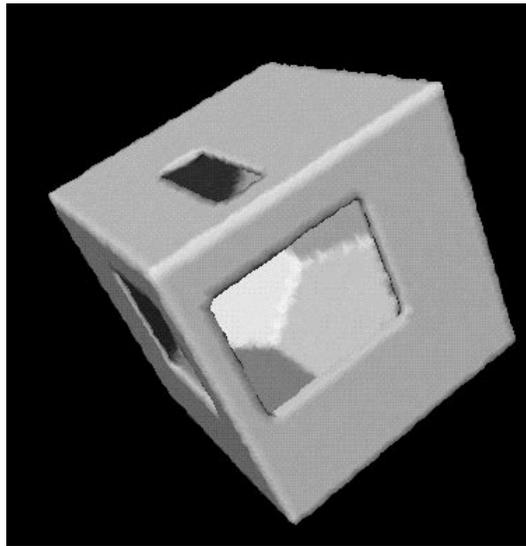


Figure 5.3: A dodecahedron subtracted from a cube

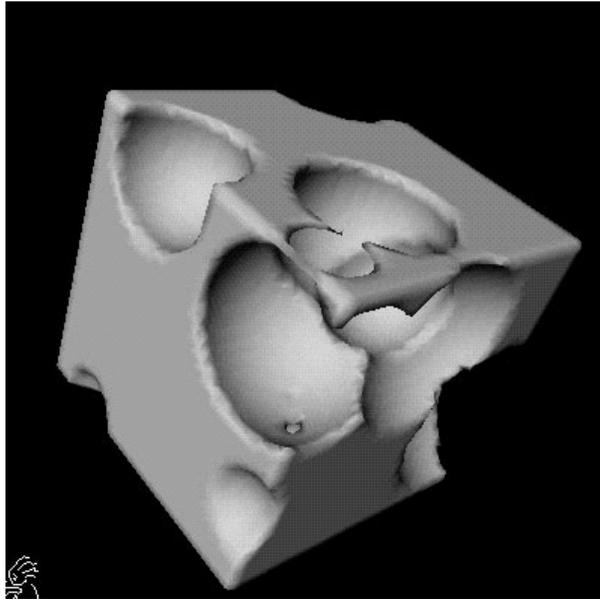
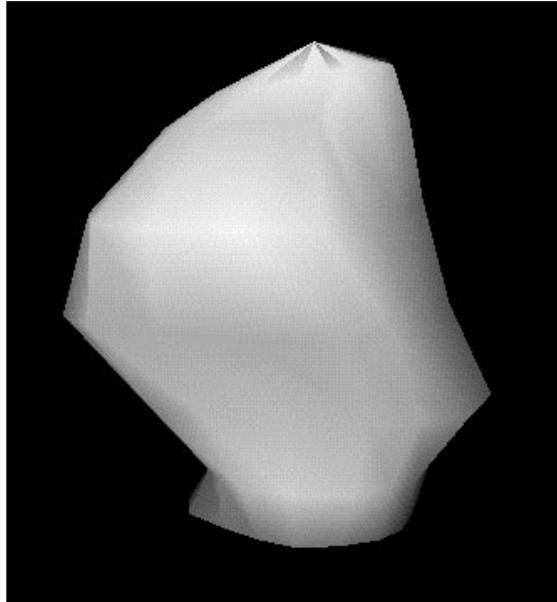


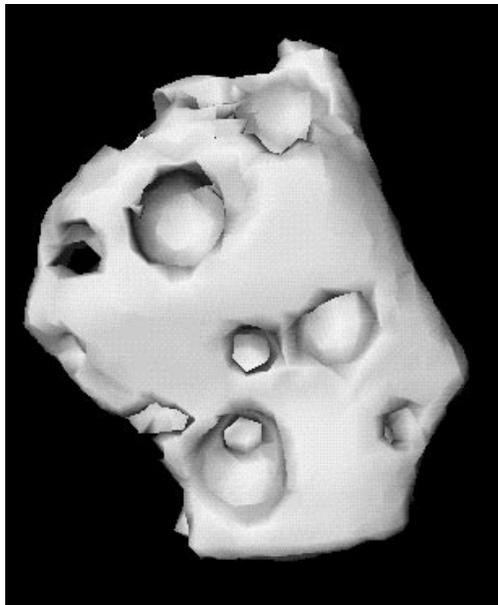
Figure 5.4: Many spheres subtract from a cube

maps can only affect the appearance near the surface. Figure 5.5(a) shows the base shape of a rock. Figure 5.5(b) shows a rock into which holes have been placed. There is a substantial difference in the appearance of these two rocks, but these pictures only show the geometric properties of the two objects. Texturing techniques have not been used yet.

Figure 5.6(a) shows the base shape with a complex texture on it. A similar texture is applied to the rock with holes, which is seen in Figure 5.6(b). The textured pictures were rendered in Alias. The texture applied to the base has a solid fractal color map, a solid bump map and a solid displacement map. The bump map is Alias solid marble texture with fractal intensity parameters. The displacement map is Alias solid rock texture. The texture applied to the rock with holes is the same without the displacement map. These textures



(a) Base shape



(b) Rock with holes

Figure 5.5: No texturing

were created by David Botta. The texturing does make a substantial difference. however the texturing alone cannot produce the desired results.

5.4 Distribution of holes

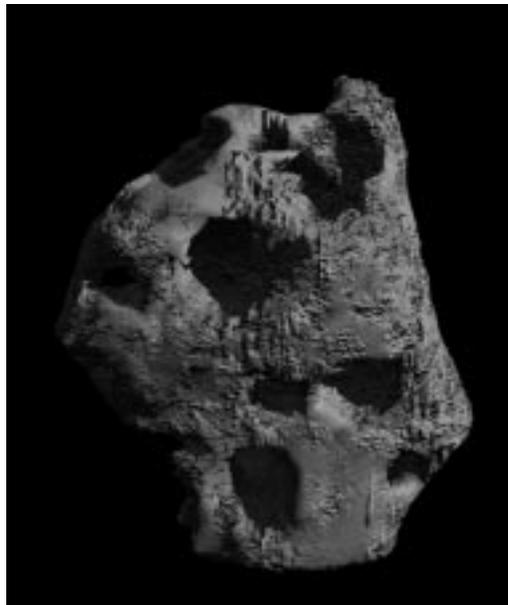
The use of random variables for placement of the holes was an important part of producing the realistic looking pictures of ornamental rocks. The use of data collected from pictures of real rocks was valuable for obtaining distribution functions to produce natural looking rocks. Figure 5.7(a) shows a rock with many holes where the placement of the holes was not random. The uniformity of the holes stand out and help make the image look unnatural. In figure 5.7(b) a uniform random variable is added for the location of the holes, but the size of the holes is still kept constant. This helps the appearance of the rock but uniformity in the size of the holes is still noticeable. In Figure 5.7(c) a uniform random variable, between 0 and the max hole size, is also used to choose diameter of the holes. This again enhances the appearance of the rock. The rock in Figure 5.7(d) uses the distribution function obtained from the sampled data from rock 1 for the diameter of the holes.

5.5 Level of detail

Being able to control the level of detail of an object is often a desired feature in modeling natural phenomena as well as other types of models. The modeling method described in this thesis has the capability of creating objects with

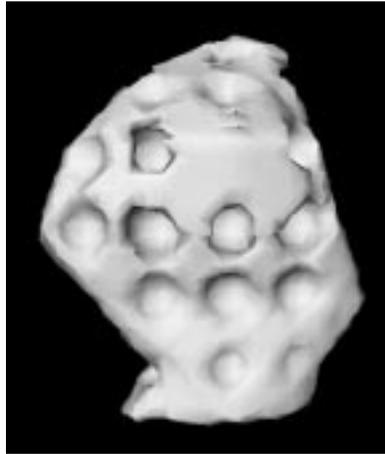


(a) Base shape

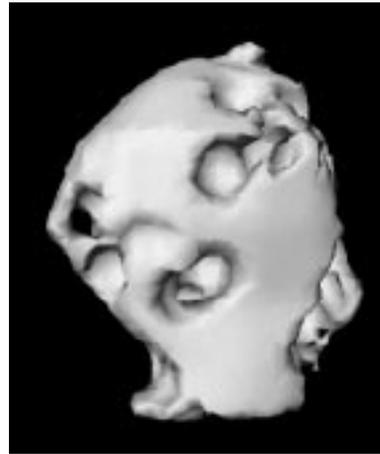


(b) Rock with holes

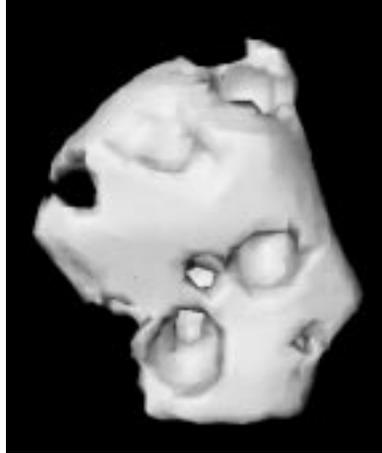
Figure 5.6: Texturing



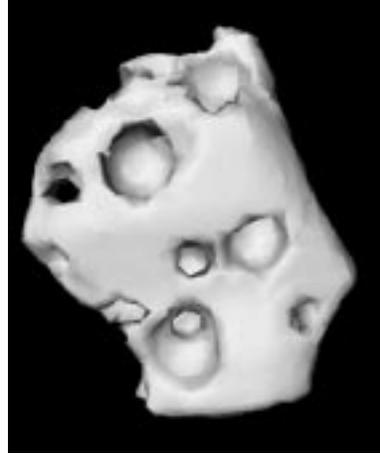
(a) No randomness in the holes



(b) uniform distribution for location, constant diameter



(c) uniform distribution for location and diameter



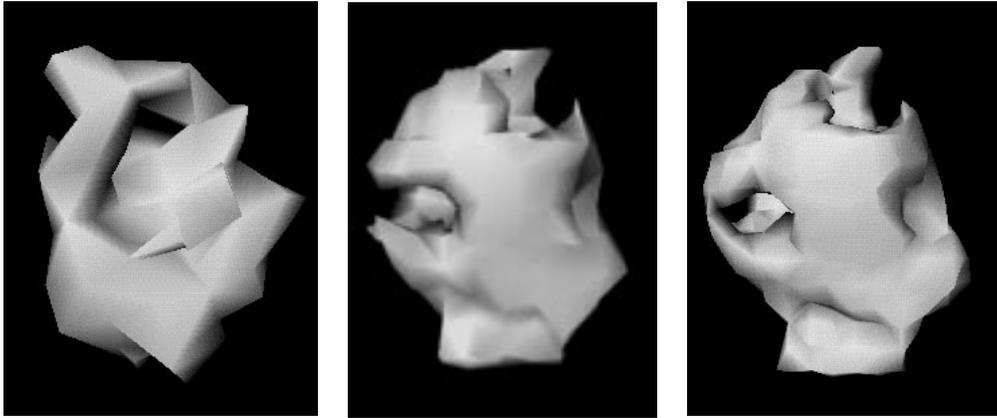
(d) diameter distribution from collected data

Figure 5.7: Distribution of holes

different levels of resolution. The number and size of the triangles in the final object is dependent on the number of voxels. An example of this can be seen in Table 5.1, Figure 5.8, Figure 5.9, and Figure 5.10. Figure 5.8 shows how the quality of a generated rock can vary depending on the resolution of the voxel set. The rock is not recognizable with only a voxel set of size 10 cubed. Aliasing effects from the coarse sampling are so pronounced that even the basic shape is lost. The basic shape of the rock can be seen starting with a voxel set of 15 cubed. As the size of the voxel set increases so does the quality of the image. For this size of picture the increase in quality starts to level off between 30 and 40 for the voxel set size. The pictures from figure 5.8(d) and figure 5.8(e) look almost the same even though the rock in figure 5.8(e) has almost twice as many triangles. Figure 5.9 shows different resolutions of the same rock in proportion to the screen size. This shows how various levels of detail could be used with the same object. The use of texturing makes a difference in determining the optimal resolution. In Figure 5.10, despite the fact that the picture size is larger than in Figure 5.8, there is little noticeable difference between the 20 voxel cubed and the 30 voxel cubed pictures. The texturing covers up the lack of high frequencies in the lower resolution rocks.

Figure 5.8	a	b	c	d	e
Number of Voxels cubed	10	15	20	30	40
Number of Triangles	261	777	1385	3315	5956

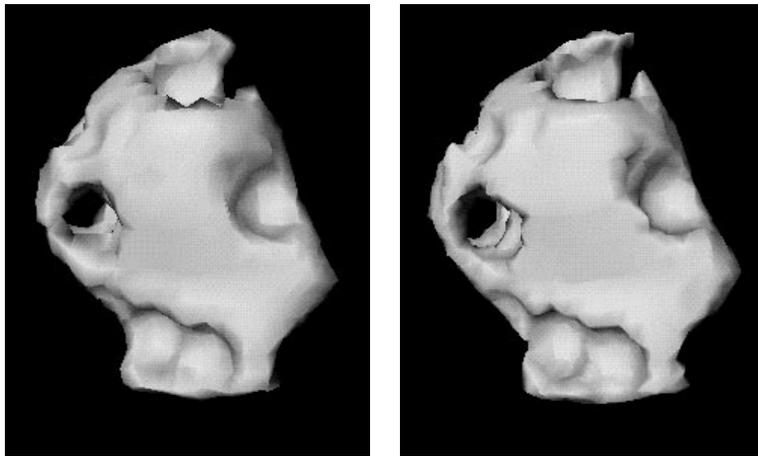
Table 5.1: Size and resolution of a modeled rock



(a) 10 voxels cubed

(b) 15 voxels cubed

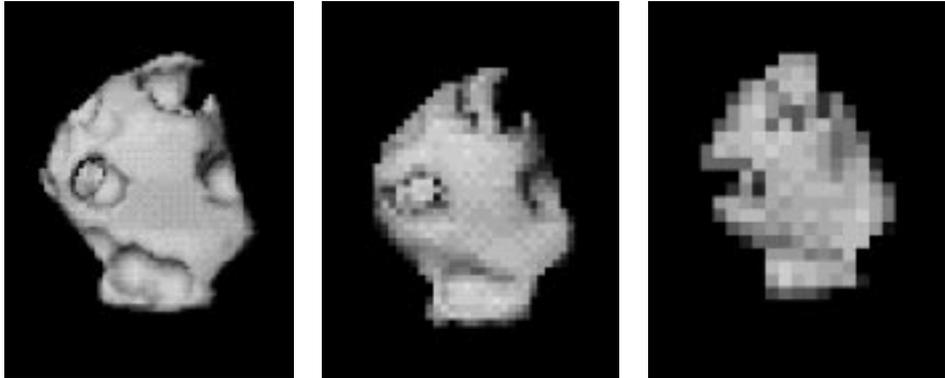
(c) 20 voxels cubed



(d) 30 voxels cubed

(e) 40 voxels cubed

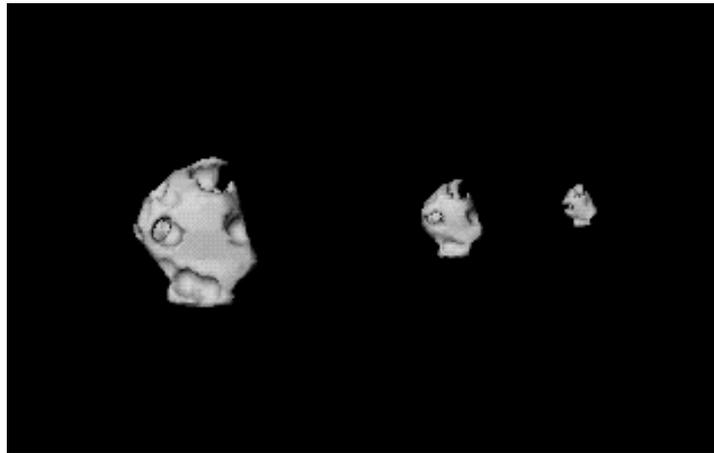
Figure 5.8: Different resolutions of same rock



(a) 40 voxels cubed
66 pixels high

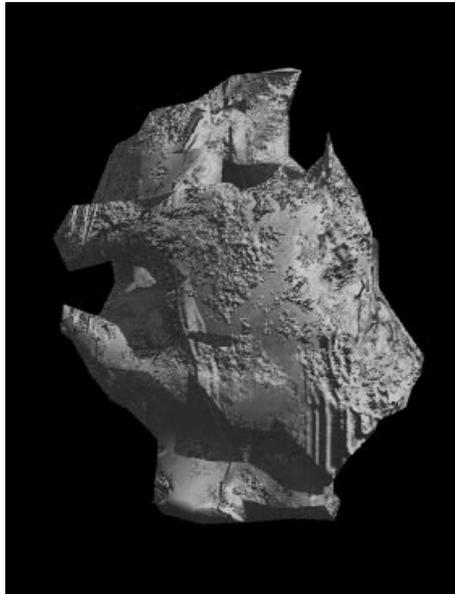
(b) 20 voxels cubed
34 pixels high

(c) 15 voxels cubed
19 pixels high

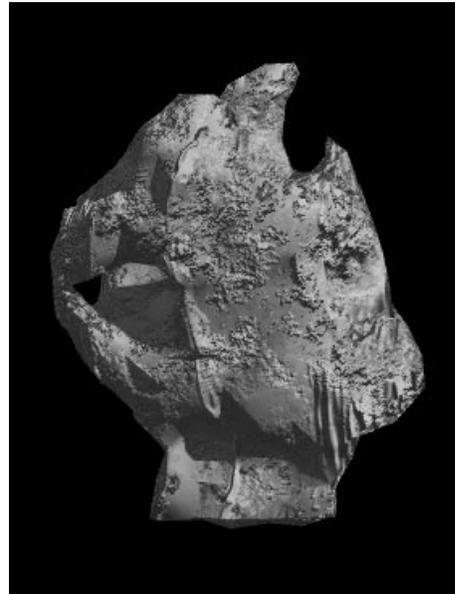


(d) All three

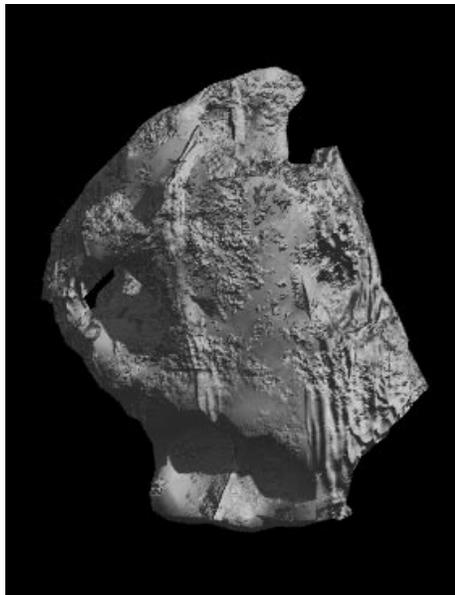
Figure 5.9: Different resolutions of same rock



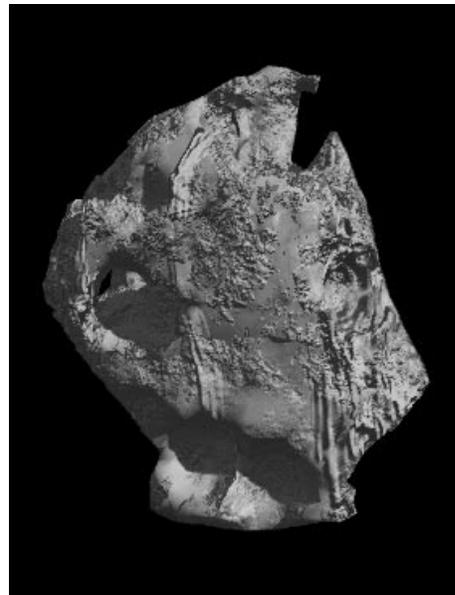
(a) 15 voxels cubed



(b) 20 voxels cubed



(c) 30 voxels cubed



(d) 40 voxels cubed

Figure 5.10: Different resolutions of same rock with texturing

5.6 Speed and Size

The amount of time it takes to produce a realistic image is an important factor in determining how useful a modeling technique is. The time for making an image of an ornamental rock can be broken into two parts, modeling and rendering. The rendering for the rocks is handled by a commercial rendering package, Alias. As far as the method described in this thesis is concerned, the time for rendering is dependent only on the number of triangles from which the rock is made. The amount of time the modeling stage takes is dependent on the size of the voxel set, the number of surfaces in the base shape, the number of holes, and the number of surfaces in each hole. The time complexity is

$$O(\text{Vox} \times (\text{Num Base Surfaces} + \text{Num Hole Surfaces} \times \text{NumHoles})).$$

When spheres are used as the holes, the number of surfaces in each hole is just one. Speed tests were done with different parameters for the variables listed above. They were run on a SGI Indigo2, with a 200Mhz, MIPS R4400. All the holes used were spheres. The results can be seen in Table 5.2 Table 5.3 and Table 5.4. The number of holes and the size of the base shape independently contribute to the processing time. A rock similar to the one in Figure 5.13 will take 3-5 minutes to produce.

5.7 Realistic looking rocks

The realistic appearance of computer generated images can be determined by comparing them with pictures of the real object. Figure 5.11 shows a

Number of Voxels	Number of Base Surfaces	Number of Hole Surfaces	Time in seconds
20^3	1	0	1
20^3	1	100	10
20^3	1	250	22
20^3	1	500	47
20^3	1	1000	93
20^3	1	2000	182

Table 5.2: Time depending on the number of holes

Number of Voxels	Number of Base Surfaces	Number of Hole Surfaces	Time in seconds
20^3	1	0	1
20^3	36	0	7
20^3	224	0	46
20^3	2588	0	597
20^3	1	250	22
20^3	36	250	27
20^3	224	250	59
20^3	2588	250	616

Table 5.3: Time depending on the number of surfaces in the base shape

Number of Voxels	Number of Base Surfaces	Number of Hole Surfaces	Time in seconds
20^3	1	0	1
30^3	1	0	3
40^3	1	0	21
20^3	1	250	22
30^3	1	250	85
40^3	1	250	216
20^3	224	0	46
30^3	224	0	162
40^3	224	0	362
20^3	224	250	59
30^3	224	250	205
40^3	224	250	490

Table 5.4: Time depending on the size of the voxel set

photograph of an ornamental garden stone with a computer generated one. Figures 5.13 and 5.12 show some stones using different base shapes. Since the base shape is arbitrary one can use “non-rock” shape. Figure 5.12 shows an example with a face as the base shape. The method described in this thesis is concerned with the larger scale holes and the general structure in the rocks. The finer detail is handled by standard texturing. This texturing has an effect on the appearance of the image but is not part of the method describes in this thesis. Generating realistic looking textures is a issue. There can and should be obtained directly from the real rock, ether from photograph and texture mapping or from a statistical analysis of the texture. This should be taken into consideration when comparing the images from Figure 5.11



(a) Real Rock



(b) Virtual Rock

Figure 5.11: Comparison of real rock and virtual rock

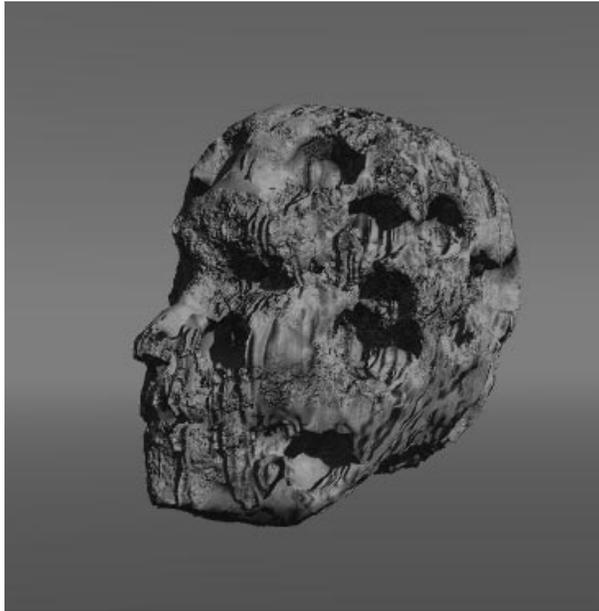


Figure 5.12: A face rock

5.8 Incorporating the Rocks into the Garden

Rocks created with the method described in this thesis are used in the Garden Project. Figure 5.14 shows a rock placed in a garden scene.



Figure 5.13: Another rock



Figure 5.14: A rock in the Yuan Ming Yuan

Chapter 6

Conclusion

This thesis described a technique used to model ornamental rocks from a Chinese garden. This technique is successful in being able to place holes in objects. Rocks have been created in this manner and imported into Alias for rendering. In Alias a texture is applied to improve the appearance of the color and finer details. This produces realistic looking ornamental rocks. The generation of one of these rocks can be done in a reasonable amount of time and the technique allows for control of the level of detail at which they are generated. These rocks are being used in scenes for the Yuan Ming Yuan garden project [wang97].

6.1 Future work

There are ways this work could be improved and expanded. One problem is that the resolution of the triangles is constant throughout the object. Some

areas of an object might need many small polygons to show the detail and other areas might only need a few large polygons. With the current method, in order to get a same level of detail in one area of an object, the high level of detail must be used throughout the object. Therefore in some case many more triangles are used than are necessary. It would be useful to address the problem of having different levels of detail on different areas of an object depending on the local curvature of the object or other parameters.

With the current method only one type of CSG operation at one level of depth can be performed. There is one base shape and from this shape objects can only be subtracted. The CSG operation could be extended. More complex rocks could then be modeled. There could be a large rock with holes with smaller rocks embedded into it. This sort of rock could be modeled by extending the current method with more CSG operations.

Developing realistic looking-rock textures will enhance the final image. Work could be done on extracting texturing data from pictures of real rocks. Information for the finer detail texturing can and obtained directly from the real rock, ether from photographs using texture mapping or from a statistical analysis of the texture.

The methods described in this thesis could be applied to other objects. Other types of rocks could be modeled using similar techniques as well as non-rock objects. Modeling of other objects with many holes in them such as a sponge or bread would be reasonable to attempt using this technique.

Bibliography

- [alle90] Arnold O. Allen. *Probability, Statistics, and Queing Theory: With Computer Science Applications*. Academic Press, Inc, second edition, 1990.
- [arvo91] James R. Arvo, editor. *Graphics Gems II*. Academic Press, San Diego, 1991.
- [bloo87] Jules Bloomenthal. “Polygonization of Implicit Surfaces”. Report CSL-87-2, Xerox PARC, May 1987.
- [buch91] John Buchanan. “The Filtering of 3d Textures”. *Proceedings of Graphics Interface '91*, pp. 53–60, June 1991.
- [ceri91] M. M. Cerimele, F. R. Guarguaglini, and L. Moltedo. “Visualizations for a numerical simulation of a flame diffusion model”. *Computers and Graphics*, Vol. 15, No. 2, pp. 231–235, 1991.
- [chib94] Niroshige Chiba, Ken Ohshida, Kazunobu Muraoka, Mamoru Miura, and Nobuji Saito. “A growth model having the abilities of growth-regulations for simulating visual nature of botanical trees”. *Computers and Graphics*, Vol. 18, No. 4, pp. 469–480, 1994.
- [eber90] David S. Ebert and Richard E. Parent. “Rendering and Animation of Gaseous Phenomena by Combining Fast Volume and Scanline A-buffer Techniques”. *Computer Graphics (SIGGRAPH '90 Proceedings)*, Vol. 24, pp. 357–366, August 1990.
- [fleis95] Kurt Fleischer, David Laidlaw, Bena Currin, and Alan Barr. “Cellular Texture Generation”. *SIGGRAPH 95 Conference Proceedings*, Annual Conference Series, pp. 239–248, August 1995.

- [fole90] James D. Foley, Andries van Dam, Steven K. Feiner, and John F. Hughes. *Computer Graphics, Principles and Practice, Second Edition*. Addison-Wesley, Reading, Massachusetts, 1990.
- [four80] Alain Fournier. *Stochastic Modeling in Computer Graphics*. Ph.D. thesis, U. of Texas at Dallas, Richardson, Texas, 1980.
- [four86] Alain Fournier and William T. Reeves. “A Simple Model of Ocean Waves”. *Computer Graphics (SIGGRAPH '86 Proceedings)*, Vol. 20, pp. 75–84, August 1986.
- [four89] Alain Fournier. “The modelling of natural phenomena”. *Proceedings of Graphics Interface '89*, pp. 191–202, June 1989.
- [fowl92] Deborah R. Fowler, Hans Meinhardt, and Przemyslaw Prusinkiewicz. “Modeling Seashells”. *Computer Graphics (SIGGRAPH '92 Proceedings)*, Vol. 26, pp. 379–388, July 1992.
- [gard84] Geoffrey Y. Gardner. “Simulation of Natural Scenes Using Textured Quadric Surfaces”. *Computer Graphics (SIGGRAPH '84 Proceedings)*, Vol. 18, pp. 11–20, July 1984.
- [glas90] Andrew S. Glassner, editor. *Graphics Gems I*. Academic Press, 1990.
- [heck94] Paul Heckbert, editor. *Graphics Gems IV*. Academic Press, Boston, 1994.
- [hung85] Chan Man Hung and Kwan Pui Ching, editors. *Life in the Forbidden City*. The Commercial Press, Ltd., Hong Kong Branch, September 1985.
- [jone96] M. W. Jones. “The production of Volume Data from Triangulated Meshes Using Voxelization”. *Computer Graphics Forum*, Dec 1996.
- [kaji84] James T. Kajiya and Brian P. Von Herzen. “Ray Tracing Volume Densities”. *Computer Graphics (SIGGRAPH '84 Proceedings)*, Vol. 18, pp. 165–174, July 1984.

- [kaji89] James T. Kajiya and Timothy L. Kay. “Rendering Fur with Three Dimensional Textures”. *Computer Graphics (SIGGRAPH '89 Proceedings)*, Vol. 23, pp. 271–280, July 1989.
- [laid86] David H. Laidlaw, W. Benjamin Trumbore, and John F. Hughes. “Constructive Solid Geometry for Polyhedral Objects”. *Computer Graphics (SIGGRAPH '86 Proceedings)*, Vol. 20, pp. 161–170, August 1986.
- [lewi89] John-Peter Lewis. “Algorithms for Solid Noise Synthesis”. *Computer Graphics (SIGGRAPH '89 Proceedings)*, Vol. 23, pp. 263–270, July 1989.
- [lore87] William E. Lorensen and Harvey E. Cline. “Marching Cubes: A High Resolution 3D Surface Construction Algorithm”. *Computer Graphics (SIGGRAPH '87 Proceedings)*, Vol. 21, pp. 163–169, July 1987.
- [max86] N. L. Max. “Light Diffusion through Clouds and Haze”. *Computer Vision, Graphics and Image Processing*, Vol. 33, No. 3, pp. 280–292, March 1986.
- [miya90] Kazunori Miyata. “A Method of Generating Stone Wall Patterns”. *Computer Graphics (SIGGRAPH '90 Proceedings)*, Vol. 24, pp. 387–394, August 1990.
- [muse93] The Place Museum, editor. *The Forbidden City*. Forbidden City Publishing House, 1993.
- [nish94] Tomoyuki Nishita and Eihachiro Nakamae. “Method of Displaying Optical Effects within Water using Accumulation Buffer”. *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series, pp. 373–381, July 1994.
- [nish96] Tomoyuki Nishita, Eihachiro Nakamae, and Yoshinori Dobashi. “Display of Clouds and Snow Taking Into Account Multiple Anisotropic Scattering and Sky Light”. *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, pp. 379–386, August 1996.

- [oppe86] Peter E. Oppenheimer. “Real Time Design and Animation of Fractal Plants and Trees”. *Computer Graphics (SIGGRAPH '86 Proceedings)*, Vol. 20, pp. 55–64, August 1986.
- [peac85] Darwyn R. Peachey. “Solid Texturing of Complex Surfaces”. *Computer Graphics (SIGGRAPH '85 Proceedings)*, Vol. 19, pp. 279–286, July 1985.
- [peac86] Darwyn R. Peachey. “Modeling Waves and Surf”. *Computer Graphics (SIGGRAPH '86 Proceedings)*, Vol. 20, pp. 65–74, August 1986.
- [pede96] Hans Køhling Pedersen. “A Framework for Interactive Texturing Operations on Curved Surfaces”. *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, pp. 295–302, August 1996.
- [perl85] Ken Perlin. “An Image Synthesizer”. *Computer Graphics (SIGGRAPH '85 Proceedings)*, Vol. 19, pp. 287–296, July 1985.
- [perl89] Ken Perlin and Eric M. Hoffert. “Hypertexture”. *Computer Graphics (SIGGRAPH '89 Proceedings)*, Vol. 23, pp. 253–262, July 1989.
- [pick89] Clifford A. Pickover. “A Short Recipe for Seashell Synthesis”. *IEEE Computer Graphics and Applications*, Vol. 9, No. 6, pp. 8–11, November 1989.
- [prus86] Przemyslaw Prusinkiewicz. “Graphical applications of L-systems”. *Proceedings of Graphics Interface '86*, pp. 247–253, May 1986.
- [prus88] Przemyslaw Prusinkiewicz, Aristid Lindenmayer, and James Hanan. “Developmental Models of Herbaceous Plants for Computer Imagery Purposes”. *Computer Graphics (SIGGRAPH '88 Proceedings)*, Vol. 22, pp. 141–150, August 1988.
- [prus93] Przemyslaw Prusinkiewicz, Mark S. Hammel, and Eric Mjolsness. “Animation of Plant Development”. *Computer Graphics (SIGGRAPH '93 Proceedings)*, Vol. 27, pp. 351–360, August 1993.
- [reed94] Todd Reed and Brian Wyvill. “Visual Simulation of Lightning”. *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series, pp. 359–364, July 1994.

- [reev83] W. T. Reeves. “Particle Systems – a Technique for Modeling a Class of Fuzzy Objects”. *ACM Trans. Graphics*, Vol. 2, pp. 91–108, April 1983.
- [reff88] Phillippe de Reffye, Claude Edelin, Jean Francon, Marc Jaeger, and Claude Puech. “Plant Models Faithful to Botanical Structure and Development”. *Computer Graphics (SIGGRAPH '88 Proceedings)*, Vol. 22, pp. 151–158, August 1988.
- [ricc73] A. Ricci. “A Constructive Geometry for Computer Graphics”. *Computer Journal*, Vol. 16, No. 2, pp. 157–160, May 1973.
- [rush87] Holly E. Rushmeier and Kenneth E. Torrance. “The Zonal Method for Calculating Light Intensities in the Presence of a Participating Medium”. *Computer Graphics (SIGGRAPH '87 Proceedings)*, Vol. 21, pp. 293–302, July 1987.
- [rush95] Holly Rushmeier, Anthony Hamins, and Mun Young Choi. “Volume Rendering of Pool Fire Data”. *IEEE Computer Graphics and Applications*, Vol. 15, No. 4, pp. 62–67, July 1995.
- [serr82] Jean Serra. *Image Analysis and Mathematical Morphology*. Academic Press, Inc., 1982.
- [stam93] Jos Stam and Eugene Fiume. “Turbulent Wind Fields for Gaseous Phenomena”. *Computer Graphics (SIGGRAPH '93 Proceedings)*, Vol. 27, pp. 369–376, August 1993.
- [wang97] LiFeng Wang, David Botta, Chris Ellefson, and Alain Fournier. “Modelling the Garden of Perfect Brightness”. *Computer Graphics Forum (Eurographics '97)*, September 1997.
- [watt90] Mark Watt. “Light-Water Interaction Using Backward Beam Tracing”. *Computer Graphics (SIGGRAPH '90 Proceedings)*, Vol. 24, pp. 377–385, August 1990.
- [webe95] Jason Weber and Joseph Penn. “Creation and Rendering of Realistic Trees”. *SIGGRAPH 95 Conference Proceedings*, Annual Conference Series, pp. 119–128, August 1995.

- [wern94] Josie Wernecke. *The Inventor Mentor*. Addison-Wesley Publishing Company, 1994.
- [worl96] Steven P. Worley. “A Cellular Texturing Basis Function”. *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, pp. 291–294, August 1996.
- [wyvi97] Brian Wyvill, Andrew Guy, and Eric Galin. “Extending the CSG Tree”. *Proceedings of the 1997 Western Computer Graphics Symposium*, April 1997.
- [yess79] C. I. Yessios. “Computer drafting of stones, wood, plant and ground materials”. *Computer Graphics (SIGGRAPH '79 Proceedings)*, Vol. 13, No. 3, pp. 190–198, August 1979.