THREE-DIMENSIONAL RECONSTRUCTION OF THE FRACTAL SETS

Z Mihajlovic B.Sc.
Faculty of Electrical Engineering, Department of Electronics,
Zagreb, Croatia

S. Mihajlovic, B.Sc.
HPT, Telecommunication Center of Croatia,
Zagreb, Croatia

Abstract

In this paper several methods for reconstruction of 3D images are presented. Some of these methods use the geometric primitives, and other render the object directly. Methods that visualize data without fitting geometric primitives are named volume rendering methods. They made superior results, but they are time consuming and have great memory requirements. Methods that fit geometric primitives first are the cuberille method and the marching cube method.

Julia set is used as data for different volume visualization methods. These methods do not depend on scaling factor, so any sampled data or vector fields of three spatial dimensions can be visualized.

1 INTRODUCTION[1]

Three dimensional volume visualization is the emerging and rapidly growing field of computer graphics and imaging. It is concerned with representation, manipulation, and understanding of the volume data [4]. Many objects and natural phenomena in our surroundings, and results of complex computational simulations in many fields, are 3D volume of data. The problem is to display clearly as much information as possible.

The need for volume visualization appears in many scientific fields, for many purposes such as medical imaging, visualization of molecular structures, flight simulations trough terrain, simulations of the fluid flow, simulations of thunderstorm, and many other. To understand, discover or communicate phenomena, scientists want to compute the phenomena, interactively explore it, and visualize hidden or inner details. For some purposes it is important to render the object realistically as the surfaces, for some it is important to emphasize existence of high values or 'hot spots', and for some purposes spatial correlation in the data is not important, but for perception and understanding it is important to render cloud-like objects.

In this paper on the same object, different methods are applied. Two of these methods make binary classification, fit geometric primitives, and then using traditional way of rendering geometric primitives, render the object. The third method renders the object directly according to data, using ray tracing [5,6]. The drawback of this third method is in

[1]This work has been carried out within the project 2-06-278 "Distributed Real -Time Computer Systems" founded by the Ministry for Science, Tehnology and Informatic of the Republic of Croatia.

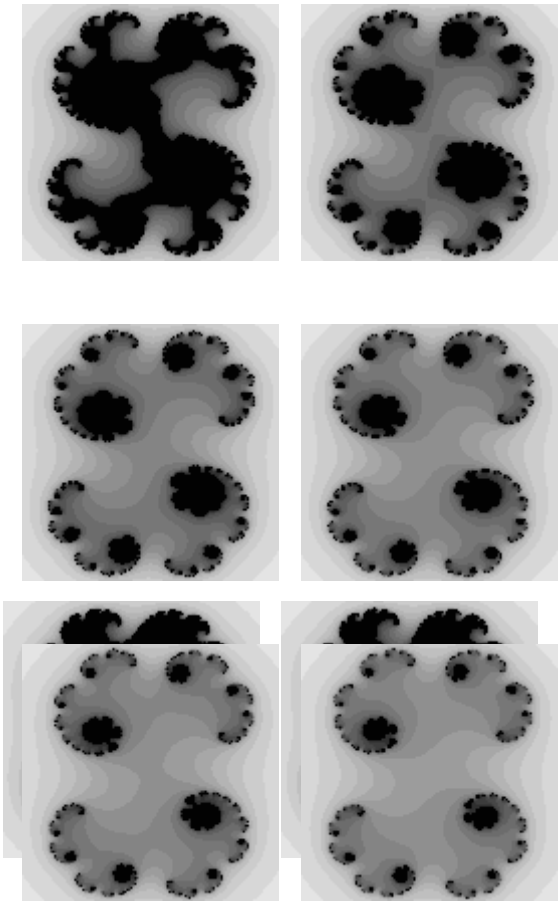

Figure 1. Eight slices from sequence of 64 slices of fractal Julia set.

time and space requirements, and alliasing effect that occurs when it is applied.

The problem of alliasing is concerned in this paper. When high resolution of data is applied, the resulting image is very time and space consuming, and when low resolution of data is used, aliasing effect in image space appears. When low resolution is applied, three linear interpolations in sample points on ray that is traced are applied. The resulting image, despite the low resolution of data is much better then when just one value of data is calculated per sample point.

2 DATA FOR VOLUME VISUALIZATION

The data for volume visualization is typically a set of scalar or vector values defined on a grid in three space. The grid is, in most cases, rectilinear and the data is assigned to the point values representing constant values for the volume elements. So, unit volume element or voxel, has a numeric value (or values) associated with it, which represents some measurable properties of a small cube of real objects, mathematics model of some object, or some other simulated values.

A typical example, in medical imaging, for data acquisition is computed tomography (CT) scanner, that estimate the radiological density (x-ray attenuation) for each voxel within a three-dimensional region, and magnetic resonance (MR) scanner, that measures the density of protons in tissue [2]. Being noninvasive, these methods become increasingly attractive in medical and some other imaging.

The data used in this article is mathematically generated. This data presents the convergence of some nonlinear iterative function in the complex plane [9, 10]. Julia sets are generated from iteration of function $z_{n+1}=z_n^2+c$, where c is constant complex value. In Figure 1 sequence of the Julia fractal sets is presented. The range in complex plane for this sequence is for z_{re} from -1.0 to 1.0, and for z_{im} from -1.15 to 1.15. Eight pictures from sequence of 64 slices are presented. This sequence is generated linearly changing the value of z_0 from (0.31,0.04) to (0.4,0.04). The resolution of each picture in sequence is 128 x 128 , and the stopping criteria is 20 iterations or $|z| < 10.000$.

3 RENDERING MODELS FOR 3D IMAGES

There are two common technics for rendering 3D images, one is more traditional and it uses the geometrical model, and the other is termed as volume rendering, where both the viewing and the shading stage are involved.

Methods that fit geometric primitives, may use information from slices to find contours and dispaly them, or to find connection between neighbour slices, and render the object using geometric primitives based on interconnection of contours. The other way is to find the border of object using volumetric data insted of concerning individual slices. Cuberille and marching cube method use the volumetric data to render the border of object. All of these methods make binear classification in data to define border. Volume rendering method use the inner informathion, without binary threeshold classifications, so better visualization may be achived.

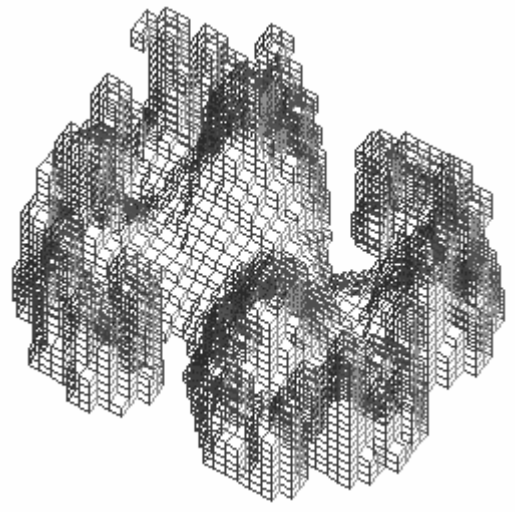

Figure 2. Wire frame for cuberille model.

3.1 Connection of contours

One of the first attempts to form geometric model is based on displaying of contours from slices or on displaying of connections of contours with some surfaces. From individual slices two-dimensional contours can be manually traced or automatically extracted. Points from extracted contours can be connected in adjacent slices to form triangle mesh or higher order surface patches. The problem is in branching structures when the number and the shape of contours in several sections are not the same. In that case the interactive work of users is needed to define the shape of branching structures, which may slow down the rendering.

3.2 Cuberille model

In cuberille approach each solid voxel is treated as a small cube [1]. Depending on threshold value, the border between two materials is set. In this approach binary classification is made indicating where particular material is present. The faces of the small cubes that indicate the border of the final object are extracted and presented. In the *Figure 2* mesh of cubes that is generated from slices of sequence in *Figure 1,* but in lower resolution, is shown.

A major problem in the cuberille model is surface shading. In the cuberille environment, a boundary surface is used as an approximation to an object surface. In rendering the boundary surface, one desires to make it appear as similar as possible to the unknown object surface. The appearance of the surface depends on the angle between the normal on the surface and the direction of the light.

A major problem with constant shading is that only six orientations of surface are possible, and only three are visible from a single point of view. So, the boundary surface will have sharp edges at points where

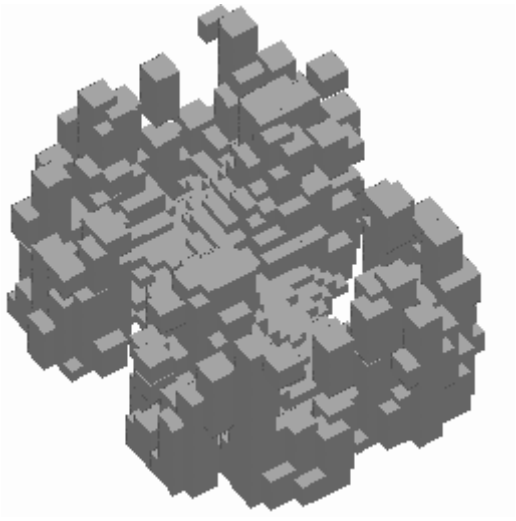

Figure 3. Constant shading with z-buffer.

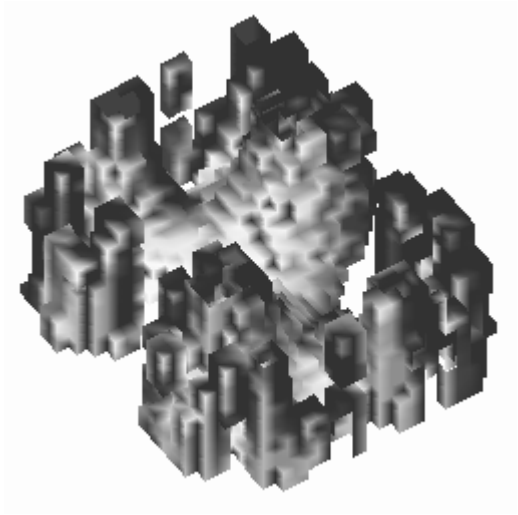

Figure 4. Gouraud shading on cuberille model.

corresponding object surface is quite smooth. Figure 3 presents the constant shading on the same mash as in Figure 2.

To overcome the problem of artificial edges, some other methods to estimate the normal, and other methods for shading may be used. The idea is to estimate the object normal at the peaks of cube based on the volumetric data. That normal then may be used in Gouraud shading of cubes [8]. The resulting image is presented in *Figure 4*. The components of local gradient at the point x, y, z are N_X, N_Y, N_Z given by equations (1), (2) and (3). D(i, j, k) is value of the data at the point i, j, k, and Δx, Δy, Δz, are lengths of each cube.

$$N_x(i,j,k) = \frac{D(i+1,j,k) - D(i-1,j,k)}{\Delta x} \quad (1)$$

$$N_y(i,j,k) = \frac{D(i,j+1,k) - D(i,j-1,k)}{\Delta y} \quad (2)$$

$$N_z(i,j,k) = \frac{D(i,j,k+1) - D(i,j,k-1)}{\Delta z} \quad (3)$$

The inherent problem in this method is that it inherently introduces artifacts. The marching cube method creates a polygonal representation of constant density surfaces, using interpolation in each cube instead of representing small cubes, and thereby avoid the problem of artificial edges.

3.3 Marching cubes

The marching cubes algorithm [7] determines how the surface intersects each cube and then marches to the next cube. These algorithms also apply binary classification, using threshold value to determine for each cube which cube's vertices exceed the value of threshold. These vertices are inside the surface. As there are eight vertices, and each of them may be in or out the surface there are 256 ways a surface can intersect the cube. For each of the 256 cases the topology of the triangulated surface is determined in look-up table. As the surfaces are created by connecting lineary interpolated points on the edges, for each cube, the normals of surfaces are not discretized like in the cuberille model. In *Figure 5* there is an example of triangulated mesh on the same data as in Figures 2, 3, and 4.

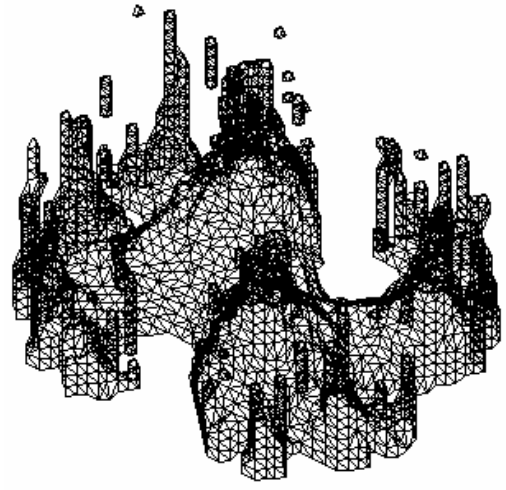

Figure 5. Triangular mesh for marching cube method.

In this approach shading is also important stage. If the constant shading is used, the result is shown in *Figure 6*, and the result is better when estimation of normals as local gradient is applied, using the expressions (1), (2), (3). The problem in classic calculation of normals, where all polygon normals incidents with one

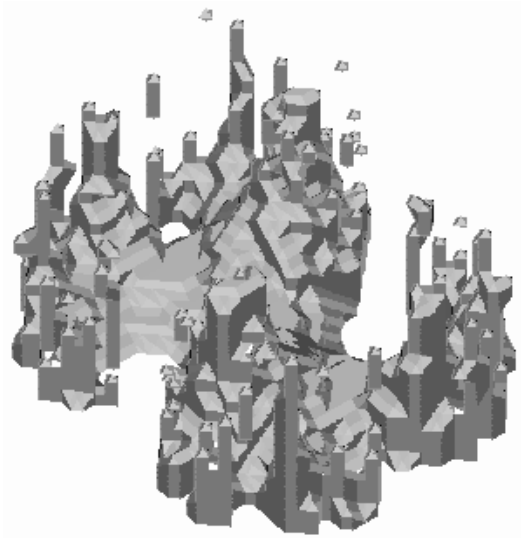

Figure 6. Constant shading for the model on figure 5.

pick are used to estimate the normal in that pick, is that the number of polygons is very large. In that case all polygons of three neighbor slices must be stored, and that requires a large amount of memory. When local gradient is used as estimation for normals, the calculated polygons may be rendered directly, so storage is needed only for four input slices of data.

The local gradient is calculated in cube's vertices, and then bilinear interpolated to achieve normals in polygon vertices which lie on the cube edges. The estimated normals are then used to calculate the intensity for each vertice, so Gouraud shading may be applied. The result is shown in *Figure 7*.

The problem in this method is that for some cases missing of polygons, or ill defined polygons may appear, so additional testing of neighbors is required [13]. As this method uses

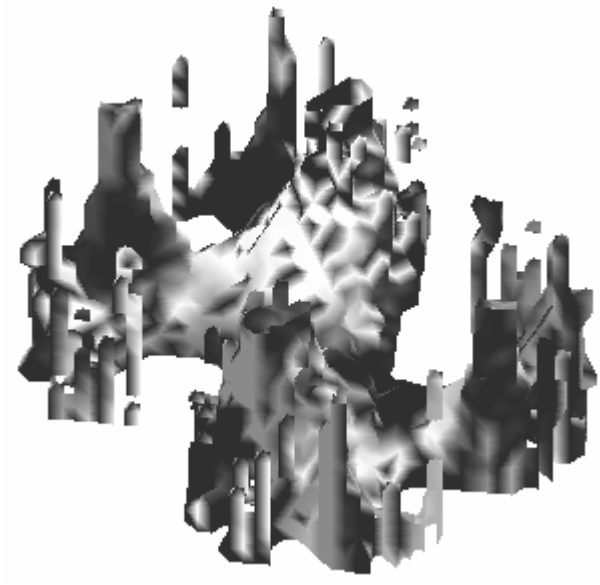

Figure 7. Gouraud shadeng on the marching cube model with z-buffer.

the polygonal representation, it is not appropriate for representation of some objects as clouds, dust, fog or smoke. The problem is also in binary classification, so in the resulting picture only sharp edges may appear.

For some purposes the rendering of transparent surfaces is important, like rendering electron density clouds in molecular structures, or rendering soft tissues in medical applications, while bone structure is opaque. That approach needs inside information, and models that use binary classification are not appropriate.

3.4 Volume rendering

To avoid the problem of binary classification, researchers have begun exploring ray tracing methods [5, 6, 12], wherein the intermediate geometric representation is omitted. Images are formed by tracing all data samples and projecting contributions of visible samples onto the picture plane. As all data samples are included, the improvement is that volume rendering creates a mechanism for displaying weak or fuzzy surfaces. It also allows us to separate shading and classification operations. This separation implies that surface shading does not depend on the success or

failure of classification. In this approach, to

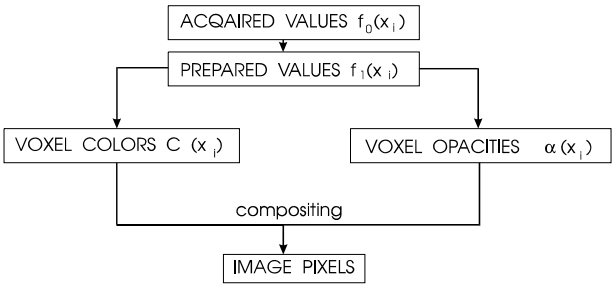

Figure 8. Separation of shading and classification operations.

each data sample the opacity $\alpha(x_i)$ is associated, so binary classification is avoided. This approach is summarized in *Figure 8*.

Components of color C $c_r(x_i)$, $c_g(x_i)$, $c_b(x_i)$, for each data sample (voxel) are calculated independently using local gradient in the voxel peaks and the value of data. Rays are then cast into these two arrays. For each ray, components of color and opacities are composted to get the color in image space.

This approach is achieved by simplifications made on transport theory model where extracting the essential content of a 3D data field by "virtual" particles passing the field is made [3, 11]. The concept of "virtual" particles generalize the models for tracing light rays in complex environments in computer graphic, where the interaction of the light with the object is governed by optical laws, and another simplification in this transport theory concept is appropriate in volume rendering model.

The problem is that the transport theory model proposes line integral during the path on the ray, and simplification uses the sample points to accumulate the contributions of colors and opacities on the ray. The expressions for calculating contributions of opacity $\alpha(x_i)$ and color c is given by (4) and (5)

$$\alpha_{out} = \alpha_{in} + \alpha(1 - \alpha_{in}) \quad (4)$$

$$C_{out} = C_{in} + C(1 - \alpha_{in}) \quad (5)$$

were α and C ere contributions of instant position, α_{in}, C_{in} are input values, and α_{out}, C_{out} are output values. The final C is obtained from expression $C = C_{out} / \alpha_{out}$.

If the sample points omit the important voxel, black points may appear in the final image. The sample points are on K evenly spaced locations, along the ray. If only the

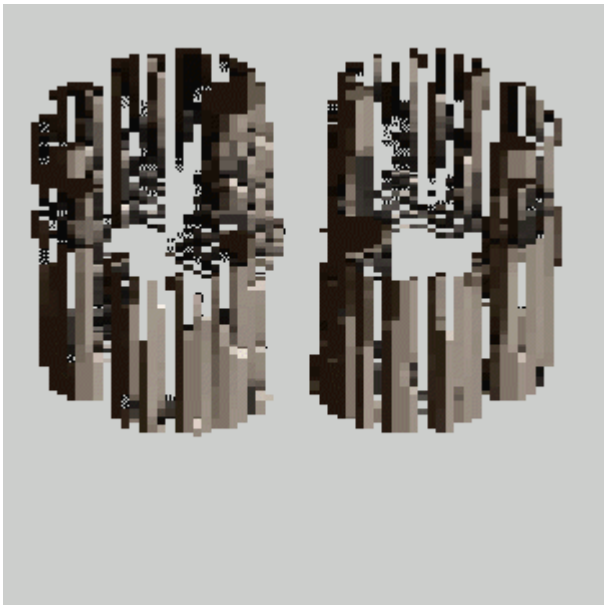

Figure 9. Volume rendering without interpolation in the sample points.

closest location of eight voxels is used, in the resulting image artificial edges of voxels may appear. The *Figure 9* presents this problem.

When higher resolution in voxel space is used artificial edges disappear only when resolution in voxel space is higher than resolution in pixel space. To avoid artificial edges, there-linear interpolation in the sample points, among eight vertices of cube in which sample points occur, must be calculated. This step requires additional calculations which may slow down the rendering.

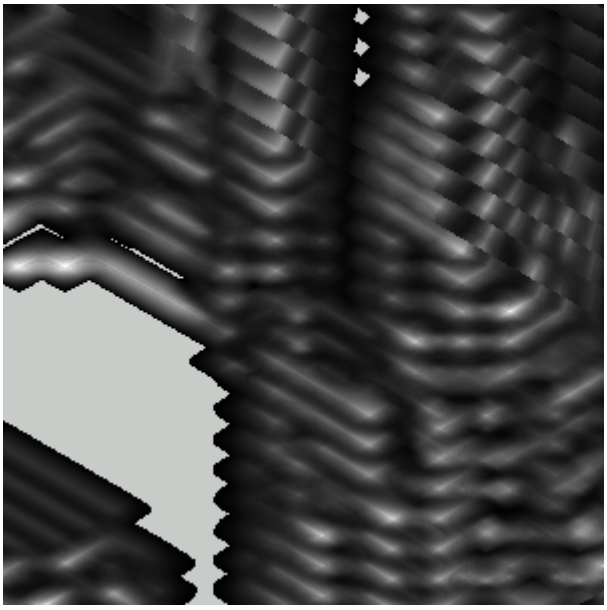

Figure 10. In the final image strips may appear if the distance between sample points is too large.

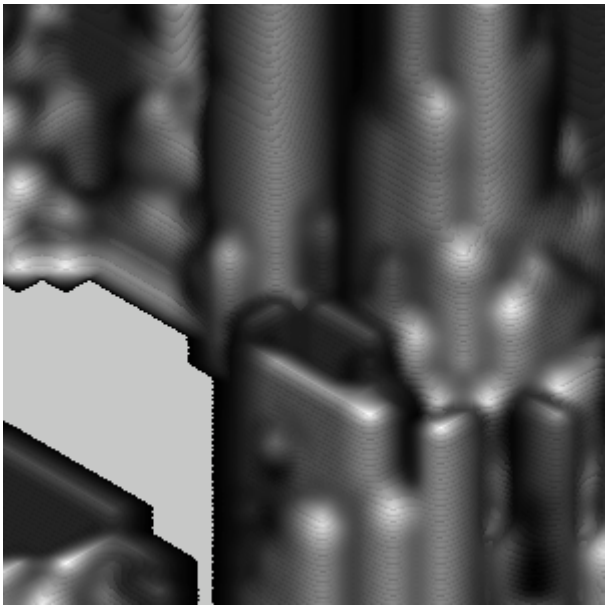

Figure 11. Three times shorter distances between sample points then in figure 10 make better result.

Another problem is if the distance between sample points is too large. In that case the stripes in the final image may appear. Namely, the evenly spaced sample locations, and rectilinear voxel space interfere, so, depending on the view position, the strips in the final image may appear. The *Figure 10* presents the strips in the final image if the distance between two sample points is too large and if the distance is 3 times shorter result is as in *Figure 11*. In these pictures view position is zoom in the center of the picture, so details can be considered.

4 CONCLUSION

The volume visualization is new and rapidly growing field in computer graphic. There are several different approaches to that problem. Methods that make binary classification may be used in medical imaging but there are not appropriate for representations of fuzzy objects. The volume rendering method makes superior results, but it is time and space consuming. The required time rising if the final image should be better, when interpolations and thickening of the sample points is used.

REFERENCES

[1] L.S.Chen, G.T.Herman, R.A.Reynolds, J.K.Udapa, Surface Shading in the Cuberille Environment, *IEEE Computer Graphics and Applications*, Vol. 5, No. 12, (December 1985), 33-43.

[2] A.K.Jain, Fundamentals of Digital Image Processing, Prentice-Hall International 1989, A Division of Simon & Schuster, NJ, USA.

[3] J.T.Kajiya, B.P.Von Herzen, Ray Tracing Volume Densities, *Computer Graphics*, Vol. 18, No. 3, (July 1984), 165-174.

[4] A.Kaufman, 3D Volume Visualization, *Eurographics '90*, Tutorial Note 12, 1990.

[5] M.Levoy, Display of Surfaces from Volume Data, *IEEE Computer Graphics and Applications*, Vol. 8, No. 3, (May 1988), 29-37.

[6] M. Levoy, Efficient Ray Tracing of Volume Data, *acm Transactions on Graphics*, Vol. 9, No. 3, (July 1990) 245-261.

[7] W.E.Lorensen, H.E.Cline, Marching Cubes: A High Resolution 3D Surface Construction Algorithm, *Computer Graphics*, Vol. 21, No. 4, (July 1987), 163-169.

[8] S.Mihajlovi}, @.Mihajlovi}, Z.Urh, The Algorithm for fast Rendering and Shading of Polygons, Zbornik radova: XXXV jugoslavenske konferencije ETANA, Ohrid, !991, 345-354.

[9] H.O.Pitgen, P.H.Richter, The Beauty of Fractals, Springer-Verlag, Berlin, Heideberg 1986.

[10] H.O.Pitgen, D.Saupe, The Science of Fractal Images, Springer-Verlag, New Yourk, Heideberg 1988.

[11] P. Sabella, A Rendering Algorithm for Visualizing 3D Scalar Fields, *Computer Graphics*, Vol. 22, No. 4, (August 1988) 51-58.

[12] H.K.Tuy, L.T.Tuy, Direct 2D Dispaly of 3D Objects, *IEEE Computer Graphics and Applications*, Vol. 4, No. 10, (October 1984), 29-33.

[13] J.Wilhelms, A. Van Gelder, Topological Considerations in Isosurface Generation, *Computer Graphics*, Vol. 24, No. 5, (November 1990), 79-86.