Volume Graphics (2005)
E. Gröller, I. Fujishiro (Editors)

Representation of Objects with Sharp Details in Truncated Distance Fields

Pavol Novotný[1] and Miloš Šrámek[2]

[1] Faculty of Mathematics, Physics and Informatics, Comenius University in Bratislava, Slovakia
[2] Commission for Scientific Visualization, Austrian Academy of Sciences, Vienna, Austria

Abstract

We present a new approach for voxelization of implicit solids which contain sharp details. If such objects are processed by common techniques, voxelization artifacts may appear, resulting, among others, in jaggy edges in rendered images. To cope with this problem we proposed a technique called Sharp Details Correction. *The main idea is to modify objects during the process of voxelization according to the representability criterion. This means that sharp edges end vertices are rounded to a curvature, which depends on the grid resolution. Thus, we obtain artifact-free voxelized solids which produce alias-free images.*

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Curve, Surface, Solid, and Object Representations

Keywords: sharp details, implicit solids, voxelization, truncated distance fields, sampling theory, artifacts, jaggy edges, representability criterion, fast marching

1. Introduction

Volume Graphics (VG) is a subarea of Computer Graphics (CG), in which modeled objects are represented solely as sampled three-dimensional (3D) scalar or vector fields in the form of 3D grids of volumetric primitives—voxels. As opposed to the traditional CG, where objects are represented analytically, this alternative approach often leads to significant simplification of the rendering pipeline, since the numerous classes of object definitions are replaced by a single one, the voxel [KCY93]. While in CG objects are rendered directly from their analytic description, in VG an additional step, voxelization, is necessary, which converts the analytical object description to the volumetric one. Once the object is converted, an object independent volumetric renderer is used for its rendering. Since the volumetric description is similar to the one obtained by tomographic scanners, this approach can take advantage of the wide plethora of hardware and software volume visualization techniques [PHK*99,Lev88].

Voxelization is in principle a sampling process, and therefore all consequences of sampling theory regarding discretization as well as quantization apply. The early binary voxelization techniques [Kau87] discretized objects only in two levels—background and foreground—which resulted in severe reconstruction artifacts [YCK92]. Later in order to circumvent this problem filtered techniques were proposed [WK94], where an object occupancy function (a function returning 1 for a point inside the object and 0 otherwise) was low-pass filtered prior to sampling. Although filtering improves the possibility to reconstruct surfaces of the original objects significantly and enables even simulation of surface refraction in static images, it is still not precise enough to produce artifact-free animated image sequences [SK99]. This goal was achieved only by today's state-of-the-art *distance field (DF)* techniques, where the distance-to-surface function is sampled instead of the occupancy function [SK99, BMW98, Gib98, FPRJ00, BSC00, PF01, BC02]. The reason why the DF representation results in much preciser reconstruction of object surfaces resides in the fact that DF is a linear field and as such can be reconstructed *precisely* by linear filters (the trilinear filter for reconstruction of the field and central differences for that of its gradient) [MMMY97]. Of course, the linearity of the DF is to a certain level violated for all objects but halfspaces, which results in reconstruction errors [Gib98]. However, it was shown that this error is very low except in the vicinity of

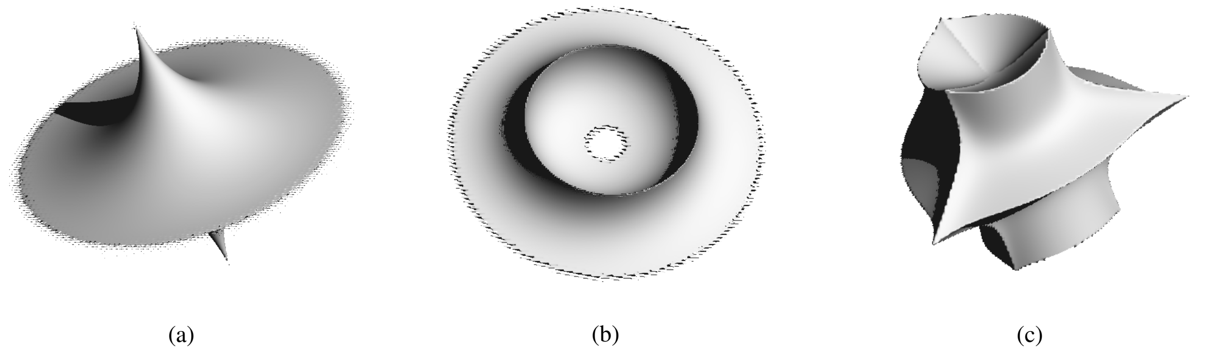

Figure 1: *Objects voxelized using the trivial voxelization technique [SK99, SK00]. (a) superellipsoid, (b) supertoroid, (c) supershape.*

small or sharp details with a characteristic dimension comparable to the size of the reconstruction filter kernel [SK99].

The DF object representation techniques can be classified according to different criteria. In a *full* DF representation, distances are assigned to all voxels of the volumetric grid [BMW98, FPRJ00], as opposed to *clamped* or *truncated distance fields (TDFs)*, where distances are stored only in a thin layer near the object surface (*transitional area*) [SK99, BC02]. In the latter case, a value is stored in distant voxels, just indicating if the voxel is outside of the object or inside. The advantage of the former approach is that it enables the implementation of certain techniques (e.g., offset surfaces) easily, while, the second one is computationally less demanding and it enables the storage of several objects in the same volumetric grid. The missing distance information in the off-surface parts of the scene is itself not a shortcoming, since techniques exist, which allow to compute the full field if necessary [SJ01].

From a different point of view, the DF techniques can be classified according to the grid resolution, which can be either *fixed* or *adaptive*. The adaptive approaches strive to solve the aforementioned problem of representation of small surface details by a locally increased sampling [FPRJ00]. Unfortunately, this increases both the temporal and algorithmic complexity. In order to circumvent this problem, techniques halfway between the fixed and adaptive sampling approach were also proposed, which take advantage of the truncated DF representation. The *two level* technique of Bærentzen [BC02] labels voxels as exterior, interior and transitional, where each transitional voxel represents a second level high resolution subgrid. A modified run-length encoding was used by Novotný [Nov03], taking advantage of the fact that voxelized object data features long runs of outside and inside voxels, interrupted by short sequences of transitional voxels.

The question of object representability by means of its sampled distance field has been theoretically addressed by Bærentzen *et al* [BSC00, BC02], who define a morphologic criterion for suitability of an object for voxelization at a given resolution. According to it, a surface of an object can be successfully reconstructed from the corresponding DF, if the object is both S_r-open and S_r-closed in respect to a spherical structuring element with radius r. In other words, one can roll a sphere with radius r on both inner and outer side of the whole object's surface, in such a way that each point of the surface can be touched by the sphere from both the sides. It is obvious, that suitable solids do not have any sharp details. In other objects, the sharp details have to be smoothed out before voxelization. The radius r of the sphere is defined by the size of the reconstruction filter. If only the DF is stored, r is defined by the size of the gradient reconstruction kernel and $r = \sqrt{6}$. If the gradient of the field is stored along with the distances, r is determined by the size of the trilinear reconstruction kernel, $r = \sqrt{3}$.

In our work we are interested in voxelization of objects which are defined by an implicit function $f(x) = 0$. If we have no special requirements on the function f, there is no guarantee that these implicit solids do fulfill the aforementioned representability criterion—they can contain sharp details. For example, solids from the class of superellipsoids, supertoroids and supershapes with certain parameters are of this kind. If we try to voxelize them in a standard way [SK99, SK00], they result in disturbing artifacts—jaggy edges—around the critical areas (Figure 1).

In this paper we propose a *Sharp Details Correction (SDC)* method to address this problem. Our goal is to voxelize these solids using TDFs in a way to get correct data, i.e., the reconstruction error should by suitably bounded. We know from the sampling theory that in the discrete space we are able to capture data details just up to a certain level. In other words, the highest feasible frequency is given by the sampling density. As edges are details with unbounded frequency, we are not able to represent them in discrete data. As we have mentioned, if we try to voxelize the objects with edges anyway, we encounter the problem of jaggy edges. A more correct approach is to modify data during the voxelization process to make edges suitably rounded—their curvature should correspond to the maximal permissible frequency, which is given by the above mentioned representability criterion [BSC00, BC02].

We solve this problem by dividing the voxelization into two stages. In the first one we compute values of voxels in the whole volume in a standard way [SK99, SK00] and mark *critical* voxels (will be explained later). These voxels form a *critical area*. In the second stage the critical voxels acquire new values according to the representability criterion by extrapolating density and gradient values from their non-critical neighbours using a modified version of the Fast Marching Method (FMM) [Bær01] and by applying ideas of our earlier paper [NDS04], where CSG operations with voxelized solids were discussed.

The paper is divided into 6 sections. In the second one we describe our volume representation and in the third one we give an overview of related techniques. The algorithm is explained in detail in the fourth section. In the last two sections we present results and propose directions of the future research.

2. Truncated Distance Fields Representation (TDFs)

Representation of volumetric models by TDFs is illustrated in Figure 2. There are three categories of voxels (*transitional, outside, inside*) which form transitional, outside and inside areas. Voxels are transitional if they are located in the surface vicinity—their distance from the surface is less than the radius r. They store information that is required for the surface reconstruction: density (corresponds to the distance from the surface) and the direction of the density gradient (indicates the direction of the surface normal). Density depends linearly on the distance and changes from 0 to 1 between outside and inside areas. Outside and inside voxels store just constant values (0, 1). Boundaries between outside, transitional and inside area are called *outside* and *inside surfaces*. The true surface passes in the middle of the transitional area and can be reconstructed by interpolation and thresholding. The radius of the transitional area r is set to $\sqrt{3}$ as explained in the first section. To reduce memory requirements we use a modified run-length compression described in [Nov03].

3. Related Work

3.1. Voxelization of Implicit Solids

Trivial voxelization of implicit solids, given by the implicit equation $f(x,y,z) = 0$, is according to [SK99, SK00] based on linear approximation of f in the surface vicinity. The distance $D(X)$ of a point $X = (x,y,z)$ to the surface can be esti-

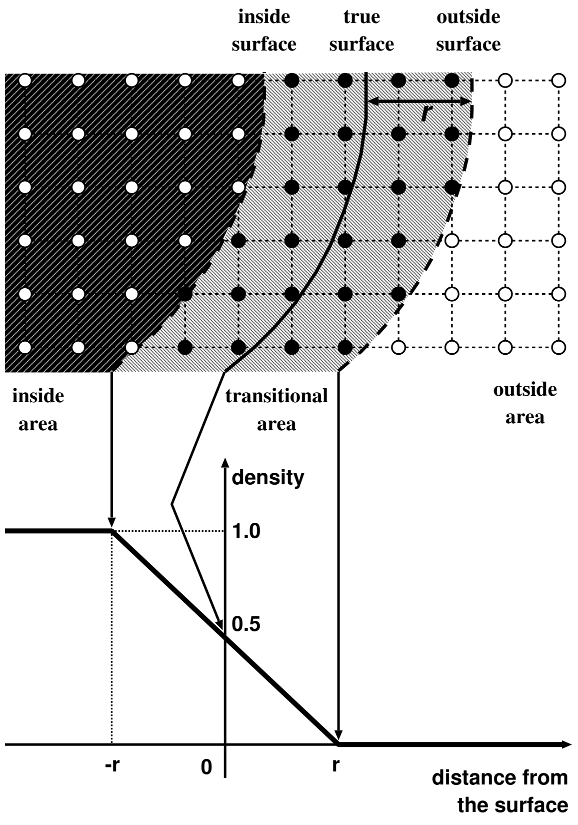

Figure 2: *Truncated distance fields representation*

Density changes linearly in the transitional area and is constant 0/1 in the outside/inside area. Direction of the density gradient is stored just in transitional voxels.

mated here using formula

$$D(X) = \frac{f(X)}{\| \bigtriangledown f(X)\|}\ , \qquad (1)$$

where $\| \bigtriangledown f(X)\|$ is the gradient magnitude. In the TDFs representation, density $d(X)$ stored in the volume depends on the distance $D(X)$ as follows:

$$d(X) = \begin{cases} 1 & \text{for } D(X) < -r \\ 0 & \text{for } D(X) > r \\ \frac{1}{2}\left(1 - \frac{D(X)}{r}\right) & \text{elsewhere} \end{cases}\ , \qquad (2)$$

where r is radius of the transitional area. The linear approximation (1) is acceptable provided that f is C^1 continuous, which is not the case in edge areas of many objects.

3.2. CSG Operations with Voxelized Solids

The result of a CSG operation with two solids may contain edges in general and so we encounter the same problem as in the case of implicit solids with sharp details—objects are not representable. In [BC02] a technique was proposed for modification of a voxelized solid by a CSG operation with an analytically tool defined. This method takes care of the object representability, so that the sharp details of the CSG result are suitably rounded.

A similar problem of CSG operations with two voxelized solids stored in TDFs was discussed in [NDS04]. The proposed technique assumes that both operands are representable. Therefore, the linear approximation can be used for local reconstruction of the surface. For example, to estimate the resulting value v_R (density, gradient) in the case of intersection one has to perform the following steps:

- Read information v_A, v_B from input voxels. In the vicinity of edges it is necessary to modify this information taking values of neighbouring voxels.
- Reconstruct inside surfaces of input volumes A, B approximated by planes α, β. For each plane its distance from voxel V and its normal vector (given by density and the direction of its gradient) are known. Denote A^*, B^* halfspaces bounded by α, β and oriented according to surfaces of original solids A, B.
- Find the nearest point X of the intersection $A^* \cap B^*$. Calculate density and its gradient in V on the basis of the length and the direction of vector $\overrightarrow{VX}$.

Other CSG operations (union, subtraction) are performed in a similar way.

Further, two postprocessing methods were proposed for correction of the voxelization artifacts. In the revoxelization technique [SDB01] high curvature areas of a voxelized volume were first detected by a Laplacian filter. Then, new density values were computed, taking the representability criterion into account, by searching for a nearest point on the inside surface for each of their voxels. Later, Museth *et al* [MBWa02] proposed a technique to smooth the high curvature areas by a level-set based blending operator.

4. The algorithm

As it was mentioned in the first section, our algorithm works in two stages. In the first one we compute density and its gradient in the whole volume and identify critical voxels by local analysis of the gradient of the implicit function. In the second stage we compute new densities and normals (i.e., normalized gradient) in the critical voxels by extrapolating and propagating information from their non-critical neighbours by the modified FMM technique and by taking the voxelization criterion into account.

4.1. Voxelization and Detection of the Critical Area

During the first stage of the voxelization process density and gradient are evaluated in each voxel using (1) and (2). Thus, voxels are classified in inside, outside and transitional. In the transitional class critical voxels are further detected by the *normal consistency test*.

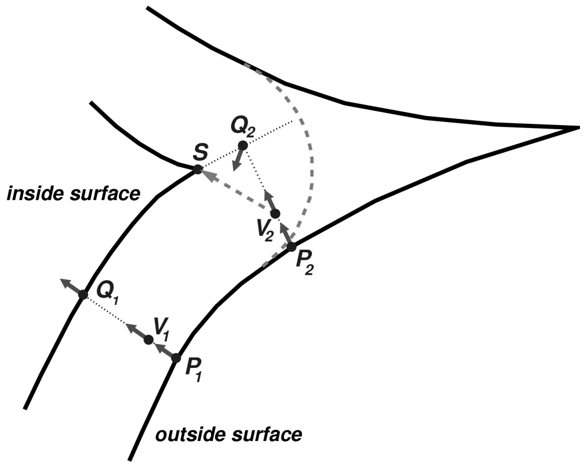

Figure 3: *The normal consistency test*

There should be approximately the same normals in points P, Q as in the voxel V (the first case). If it is not the case, the voxel is located near an edge or vertex and its value must be recalculated (the second case), so we mark it as critical. Density and normal in V_2 should be set according to the vector $\overrightarrow{V_2S}$ where S is the nearest point of the inside surface. So, the outside surface should change in the way depicted by the dashed line.

Consider a smooth surface and a line crossing it in a perpendicular direction. In such a case, gradient direction along the line, at least in the surface vicinity, should be constant or vary only a little. This is the main idea of the normal consistency test.

We compute density in given voxel V using (1) and (2). If V is transitional we check the normal consistency (Figure 3). First, we find points P, Q on the basis of density and gradient in V as feet of a perpendicular from V to outside and inside surfaces. Then we compute normals in these points. In the ordinary case they are the same as in V (or their values are very close to each other—we set the tolerance experimentally to 15 degrees). But, if V lies in the edge vicinity, these directions are very different and we mark V as critical.

4.1.1. Adjustment of the Critical Area

The critical area can contain very thin parts where obviously no transitional voxels can lie (Figure 9c, colour plate). The reason is that the thickness of the transitional area is $2\sqrt{3}$. So, to represent a detail on the corrected solid we need area of thickness more than 6 (at least double thickness of transitional area). Parts of the critical area with smaller thickness can therefore be removed. We do it using a modified version of dilation and erosion (opening) well known from morphology.

Our version of erosion removes from the critical area all voxels V having the following properties:

- At least one 6-neighbour of V is outside or inside.
- No 6-neighbour of V is transitional.

Dilation checks each voxel V initially marked as critical and adds it back to the critical area if at least one neighbour of V is critical. To adjust the critical area we perform the erosion three times followed by three repeats of the dilation. The result of this process is illustrated in Figure 9d (colour plate).

At the end of the test we have four classes of voxels in the grid:

outside: They lie outside of the object.
inside: They lie inside of the object.
transitional: They lie in the transitional area and their densities and normals are correct and known.
critical: We do not know yet where they lie. Their values (densities and normals) have to be recalculated according to the representability criterion.

4.2. Completion of Information in the Critical Area

In the second stage of voxelization we calculate correct values in all critical voxels. To simplify the description of our method consider critical voxels in an edge area first, where two object sides (faces) meet. Our goal is to extrapolate densities and normals from non-critical voxels of these two faces A and B to the critical voxels in the edge area. Therefore, at the end of this step, each critical voxel stores two densities d_A and d_B and two normals $\overrightarrow{n_A}$, $\overrightarrow{n_B}$. We consider these values as information about two voxelized solids and perform the CSG operation as described in Section 3.2 according to [NDS04].

4.2.1. Extrapolation of Information in the Critical Area

To extrapolate information from the transitional area to the critical one we use a modified version of the well-known FMM technique [Set99]. For this purpose we construct an auxiliary data structure that enables to store several *entries* in one critical voxel. The entry can be of two types—*frozen* or *narrow band (NB)*—and consists of these components:

- Distance q from the initial voxel set. It is required for the FMM front evolution algorithm.
- Density d and normal $\overrightarrow{n}$. These values are propagated from the non-critical face voxels.

As the FMM runs we have to compute new values d, $\overrightarrow{n}$ along with the evaluation of q by averaging values from neighbouring frozen entries F_i of voxels V_i:

$$d = \frac{\sum_{i=1}^{k} d_i^*}{k} \tag{3}$$

$$\overrightarrow{n} = \frac{\sum_{i=1}^{k} \overrightarrow{n_i}}{\|\sum_{i=1}^{k} \overrightarrow{n_i}\|}, \tag{4}$$

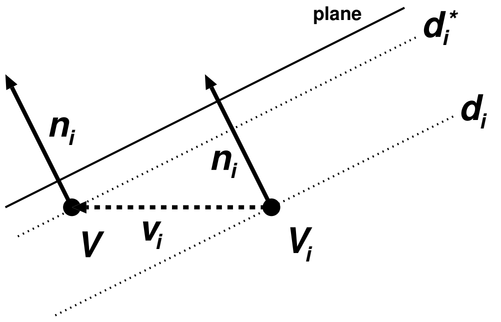

Figure 4: *Calculation of density in a neighbouring voxel*

We use density d_i and vectors v_i, n_i in the voxel V_i to estimate density d_i^* in the voxel V.

where k is the number of frozen neighbours, $\overrightarrow{n_i}$ is the normal in F_i and d_i^* is the density of F_i modified to correspond with the shift from the neighbouring voxel to the current one (Figure 4):

$$d_i^* = \frac{\overrightarrow{n_i} \cdot \overrightarrow{v_i}}{2r} + d_i \,. \tag{5}$$

In the last formula d_i is the density of F_i, r is the radius of the transitional area and $\overrightarrow{v_i}$ is the transitional vector $(V - V_i)$ (voxel is here considered as a point in 3D space). The formula comes from the fact that d_i and $\overrightarrow{n_i}$, situated in V_i, define a plane that we want to define using d_i^* and $\overrightarrow{n_i}$, situated in V. Note that the density d in the transitional area depends on the distance D linearly according to the formula (2).

The main difference of our version of the FMM technique with respect to the original resides in the fact that propagation of a front extrapolating values from, say, face A, does not stop as soon as it hits the front expanding from B in an opposite direction. Rather, their propagation continues in their original directions until either all critical voxels are processed or until the extrapolated density indicates that the voxel is safely outside of the transitional region (Figure 10, colour plate). In order to not to mix the overlapping fronts together, in evaluation of (3) and (4) just *consistent* frozen voxel entries are used (we call two entries consistent, if the angle between their normals is smaller than 35 degrees—a value estimated experimentally).

In propagation of the fronts we do not keep track of the faces, where the fronts were initialized. Rather, we decide about the number of fronts in a voxel and about the direction of their propagation solely by the consistency criterion. Therefore, the algorithm is not limited to propagation of two fronts along a single edge (this was just a simplified example for explanation). It can correctly handle several fronts merging in polyhedral vertices and even in a cone-shaped vertex (Figure 5a).

4.2.2. Final Distance Evaluation

When the evolution of expanding front has been finished we analyze information stored in each critical voxel and compute the final voxel densities according to the representability criterion. Each frozen entry (density and normal) in a voxel defines a plane, which locally approximates inside surface (Section 2) of a halfspace object. Therefore, in the case of two entries, we proceed in the same way as in the case of CSG intersection of solids ([NDS04], Section 3.2): we find a nearest point on the intersection line of both planes and store in the voxel a corresponding distance and direction vector.

If there are more than two entries in the voxel, we calculate the final value of the voxel sequentially. We take the first and the second entry from the list and create a new entry using the intersection operation. This new entry is then combined with the third entry from the list—using the intersection operation again. We continue this computation until the list of results is empty.

5. Results

The SDC method has been tested on both simple implicitly defined objects as cube and regular octahedron and complex objects as superellipsoids, supertoroids and supershapes with various parameters (Figure 5). The algorithm can be used for voxelization of a wide range of solids which contain sharp details. As the result we obtain objects which have these details suitably rounded to be representable in discrete grid. The actual rounding of edges depends on the grid resolution (Figure 6) in a way to correspond with the representability criterion [BSC00, BC02]. Thus, we avoid jaggy edges and other disturbing artifacts in rendered images typical for the trivial voxelization techniques (Section 3.1).

Of course, we need some extra processing time in comparison with the trivial approach. The dependence of factor $\tau = t_{SDC}/t_T$ on the grid resolution is depicted in Figure 7, where t_{SDC} is the time of SDC method and t_T is the time of the trivial one. We can see that the increase of processing time in high resolution data is about 30% and it is not more than 65% for all objects in all resolutions we have tested. Naturally, the processing time depends on the size of the critical area which is determined by the sharpness of edges and their length.

6. Conclusions and Future Work

We presented a new SDC technique for voxelization of implicit solids which contain sharp details. If such objects are voxelized using trivial routines, the well-known phenomenon of jaggy edges, caused by undersampling, occurs. As edges are details with unbounded frequency, no resolution is high enough for their representation. If we want to store such data in a discrete grid, the only correct way is to

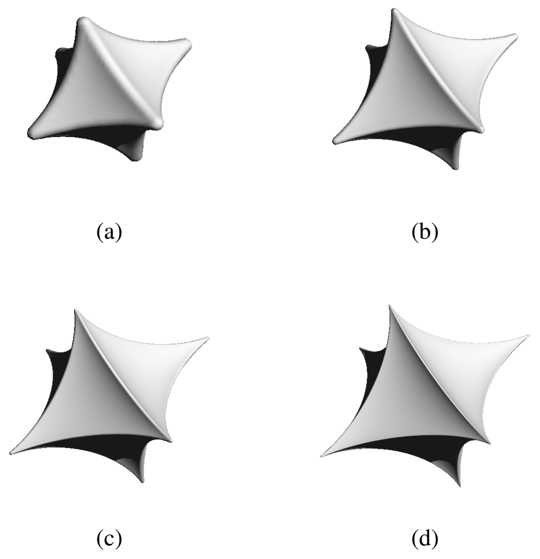

Figure 6: *Edges are rounded as to fit the given resolution. Dimensions of the grid: (a)* 64^3 *(b)* 128^3 *(c)* 256^3 *(d)* 512^3.

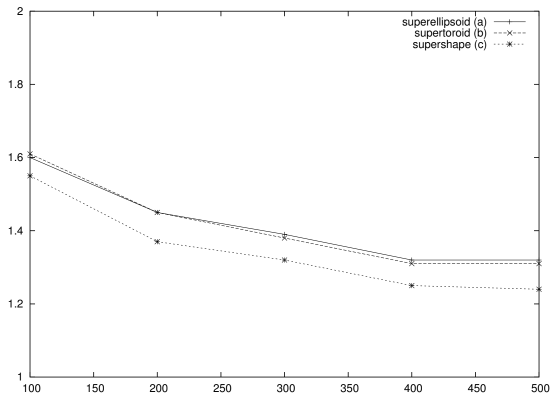

Figure 7: *The time complexity of the algorithm*

There is the resolution n of volume (grid $n \times n \times n$) on the x-axis and the ratio t_{SDC}/t_T on the y-axis, where t_{SDC} is the time of SDC method and t_T is the time of the trivial one.

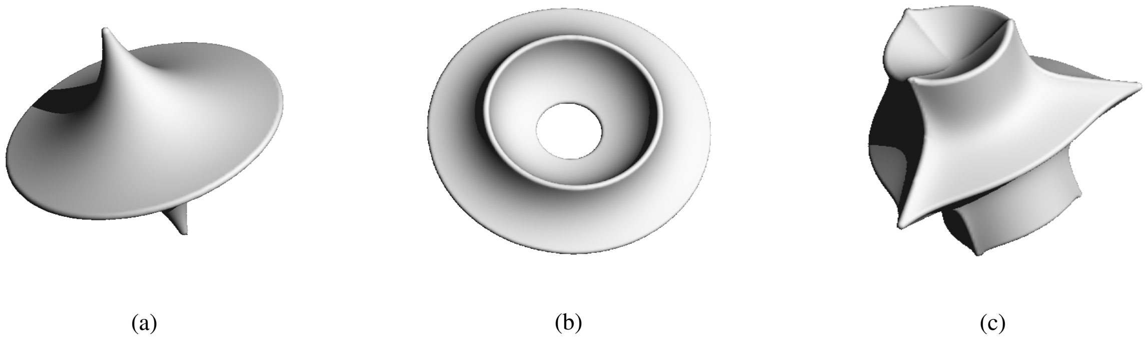

Figure 5: *Solids voxelized using SDC method*

Edges and vertices were rounded during the voxelization process to get representable objects. Compare results with Figure 1.

modify it to be suitable for the given resolution. We have solved this problem by rounding edges and vertices according to the representability criterion (Figure 6). As we can see from comparison of Figures 1 and 5, no artifacts are present in the images.

If we voxelize the objects with very sharp details, artifacts can still sometimes occur in the vertex area. We assume that it is caused by the sequential approach in the computation of the CSG intersection when more than two frozen entries are in the given voxel. Our goal is to solve this problem by computing the CSG intersection of all objects at once.

The SDC technique can at the moment be applied only to objects with convex edges (tag *c* in Figure 8). It comes from the fact that we apply the CSG intersection in all critical voxels. It is possible to modify the method for objects which contain just non-convex corners (tag *n* in Figure 8)—we only need to apply CSG union in each critical voxel instead of the intersection. But, if we have a solid with both edges and corners (Figure 8) the method fails, since we do not know, which operation (intersection or union) to perform. The situation is complicated especially in the areas around vertices where it is necessary to find the proper combination of intersection and union with several solids. This problem calls for further extension of the technique by a local curvature analysis. We will attempt to solve this in the future.

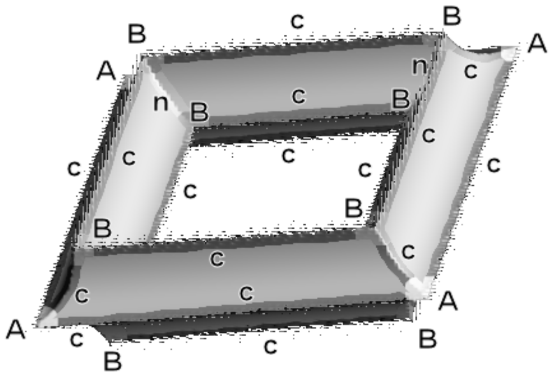

Figure 8: *Object with non-convex sharp details*

This object contains both convex (c) and non-convex (n) edges. In vertices where edges of both types meet together (B), a complex analysis of given situation is necessary.

References

[Bær01] BÆRENTZEN J. A.: *On the implementation of fast marching methods for 3D lattices*. Tech. rep., Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, 2001.

[BC02] BÆRENTZEN J. A., CHISTENSEN N. J.: A technique for volumentric CSG based on morphology. In *Volume Graphics'01* (Stony Brook, NY, June 2002), Mueller K., (Ed.), pp. 71–79.

[BMW98] BREEN D., MAUCH S., WHITAKER R.: 3D scan conversion of CSG models into distance volume. In *IEEE Symposium on Volume Visualization* (1998), pp. 7–14.

[BSC00] BÆRENTZEN J. A., SRAMEK M., CHRISTENSEN N. J.: A morphological approach to voxelization of solids. In *The 8-th International Conference in Central Europe on Computer Graphics, Visualization and Digital Interactive Media 2000* (Pilsen, Czech republic, 2000), pp. 44–51.

[FPRJ00] FRISKEN S. F., PERRY R. N., ROCKWOOD A. P., JONES T. R.: Adaptively sampled distance fields: A general representation of shape for computer graphics. In *Siggraph 2000, Computer Graphics Proceedings* (2000), Akeley K., (Ed.), Annual Conference Series, ACM Press / ACM SIGGRAPH / Addison Wesley Longman, pp. 249–254.

[Gib98] GIBSON S.: Using distance maps for accurate surface reconstruction in sampled volumes. In *IEEE Symposium on Volume Visualization* (1998), pp. 23–30.

[Kau87] KAUFMAN A.: Efficient algorithms for 3D scan-conversion of parametric curves, surfaces, and volumes. *Computer Graphics (SIGGRAPH '87 Proceedings) 21*, 4 (July 1987), 171–179.

[KCY93] KAUFMAN A., COHEN D., YAGEL R.: Volume graphics. *IEEE Computer 26*, 7 (July 1993), 51–64.

[Lev88] LEVOY M.: Display of surfaces from volume data. *IEEE Computer Graphics and Applications 8*, 3 (May 1988), 29–37.

[MBWa02] MUSETH K., BREEN D. E., WHITAKER R. T., ARR A. H. B.: Level set surface editing operators. *ACM Transactions on Graphics (TOG) 21*, 3 (2002), 330–338.

[MMMY97] MÖLLER T., MACHIRAJU R., MUELLER K., YAGEL R.: Evaluation and design of filters using a Taylor series expansion. *IEEE Transactions on Visualization and Computer Graphics 3*, 2 (1997), 184–199.

[NDS04] NOVOTNY P., DIMITROV L. I., SRAMEK M.: Csg operations with voxelized solids. In *Proceedings of the Computer Graphics International 2004* (Heraklion, Crete, Greece, 2004), Werner B., (Ed.), IEEE Computer Society Press, pp. 370–373.

[Nov03] NOVOTNY P.: *Representation of geometric solids by gradient and distance fields*. Master's thesis, Comenius University Faculty of Mathematics, Physics and Informatics, Bratislava, Slovakia, May 2003.

[PF01] PERRY R. N., FRISKEN S. F.: Kizamu: A system for sculpting digital images. In *Siggraph 2001, Computer Graphics Proceedings* (2001), Fiume E., (Ed.), Annual Conference Series, ACM Press / ACM SIGGRAPH / Addison Wesley Longman, pp. 47–56.

[PHK*99] PFISTER H., HARDENBERGH J., KNITTEL J., LAUER H., SEILER L.: The volumepro real-time ray-casting system. In *Siggraph 1999, Computer Graphics Proceedings,* (Los Angeles, 1999), Rockwood A., (Ed.), Addison Wesley Longman, pp. 251–260.

[SDB01] SRAMEK M., DIMITROV L. I., BÆRENTZEN J. A.: Correction of voxelization artifacts by revoxelization. In *Volume Graphics'01, Proceedings of the Joint IEEE TVCG and Eurographics Workshop* (Stony Brook, NY, June 21–22, 2001), Mueller K., Kaufman A., (Eds.), pp. 265–275.

[Set99] SETHIAN J. A.: *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 1999.

[SJ01] SATHERLEY R., JONES M. W.: Hybrid distance field computation for volumetric objects. In *Proceedings International Workshop on Volume Graphics 2001* (NY, USA, 2001), Kaufman A., Lorensen B.,, Mueller K., (Eds.), SUNY at Stony Brook, pp. 121–133.

[SK99] SRAMEK M., KAUFMAN A.: Alias-free voxelization of geometric objects. *IEEE Transactions on Visualization and Computer Graphics 5*, 3 (1999), 251–266.

[SK00] SRAMEK M., KAUFMAN A.: `vxt`: a c++ class library for object voxelization. In *Volume Graphics*, Chen M., Kaufman A. E.,, Yagel R., (Eds.). Springer Verlag, London, 2000, pp. 119–134.

[WK94] WANG S., KAUFMAN A.: Volume-sampled 3D Modelling. *IEEE Computer Graphics and Applications 14*, 5 (Sept. 1994), 26–32.

[YCK92] YAGEL R., COHEN D., KAUFMAN A.: Discrete ray tracing. *IEEE Computer Graphics and Applications 12*, 5 (September 1992), 19–28.