Shen CH, Zhang GX, Lai YK *et al.* Harmonic field based volume model construction from triangle soup. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 25(3): 562–571 May 2010

Harmonic Field Based Volume Model Construction from Triangle Soup

Chao-Hui Shen[1] (沈超慧), Guo-Xin Zhang[1] (张国鑫), *Student Member*, *CCF*, Yu-Kun Lai[2] (来煜坤)
Shi-Min Hu[1] (胡事民), *Senior Member*, *CCF*, and Ralph R. Martin[2]

[1] *Tsinghua National Laboratory for Information Science and Technology*
Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
[2] *School of Computer Science, Cardiff University, Cardiff, U.K.*

E-mail: {chench08, zgx03}@mails.tsinghua.edu.cn; Yukun.Lai@cs.cardiff.ac.uk; shimin@tsinghua.edu.cn; Ralph.Martin@cs.cardiff.ac.uk

Received January 12, 2010; revised February 24, 2010.

Abstract Surface triangle meshes and volume data are two commonly used representations of digital geometry. Converting from triangle meshes to volume data is challenging, since triangle meshes often contain defects such as small holes, internal structures, or self-intersections. In the extreme case, we may be simply presented with a set of arbitrarily connected triangles, a "triangle soup". This paper presents a novel method to generate volume data represented as an octree from a general 3D triangle soup. Our motivation is the Faraday cage from electrostatics. We consider the input triangles as forming an approximately closed Faraday cage, and set its potential to zero. We then introduce a second conductor surrounding it, and give it a higher constant potential. Due to the electrostatic shielding effect, the resulting electric field approximately lies in that part of space outside the shape implicitly determined by the triangle soup. Unlike previous approaches, our method is insensitive to small holes and internal structures, and is observed to generate volumes with low topological complexity. While our approach is somewhat limited in accuracy by the requirement of filling holes, it is still useful, for example, as a preprocessing step for applications such as mesh repair and skeleton extraction.

Keywords volume model, triangle soup, harmonic field, representation conversion, mesh repair

1 Introduction

In computer graphics, geometric models can be represented in various ways, such as subdivision surfaces, spline surfaces, triangle meshes, volumetric representation etc. Triangle meshes are the most widely used surface representation, and are typically produced by 3D range scans or designed using computer-aided design software. Volumetric representation is another commonly used representation, especially in important applications such as medical imaging (e.g., CT and MRI data), scientific visualization, virtual reality and simulation. Converting between these two representations is an essential step in various geometry processing processes. The volumetric representation also has the advantage that it always corresponds to some valid configuration in space, unlike a triangle mesh, which may represent a self-intersecting surface. Indeed, various methods of producing triangle meshes lead to invalid models, containing defects such as small holes, unwanted internal structures, or self-intersections. In an extreme case, connectivity information may be lacking, and we may just be presented with a "triangle soup" (see Fig.1(a)). Well-known algorithms[1-2] have been proposed to robustly convert volumetric representations to triangle meshes. However, the inverse problem, i.e., converting mesh models to volumetric representation, is challenging due to the potential presence of defects. We present here a novel method for doing

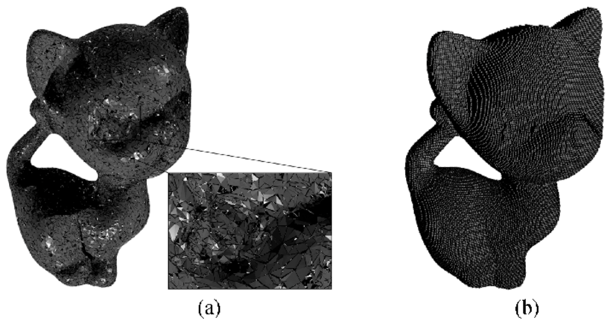


Fig.1. Kitten model. (a) "Triangle soup". (b) Volume data produced by our method.

Regular Paper
This work was supported by the National Basic Research 973 Program of China under Grant No. 2006CB303106, the National Natural Science Foundation of China under Grant Nos. 90718035 and U0735001.

this, which can be a beneficial preprocessing step for various geometry processing applications.

The process of converting triangle meshes to a volumetric representation is referred to as *scan-conversion* or *voxelization*. Clearly, the problem of producing an octree is closely related to that. The key issue is to determine which voxels in 3D space lie inside or outside the model. However, for imperfect triangle meshes, defects like missing triangles (holes), self-intersecting triangles, and internal disconnected triangles prevent this from being a straightforward decision. In the extreme case, we may simply have a triangle soup, and the lack of connectivity information makes it yet harder to decide what is inside and what is outside. Presented with such an input, many current volume construction methods such as [3-5] fail to produce satisfactory results. Such defective meshes may be restored by mesh repair methods[6] and a well-known method which can cope with triangle soups is given in [7], but the output may be of undesirably high genus if there are internal structures inside the mesh.

This paper presents a conversion method which takes a triangle soup as input, and outputs a volumetric representation in the form of an octree. Our method is based on the properties of harmonic fields. The motivation for our method is the Faraday cage, a construction used in electrostatics to shield a region from voltages. If a closed conducting shell is placed in an electric field, it provides a shielding effect: there is no field inside the shell. Even if there are small gaps, the interior is still well shielded. Thus, to generate a volumetric model, we consider the input triangles as forming a more-or-less closed Faraday cage, with potential zero. Outside and surrounding it, we then construct a second conductor having a higher constant potential. By considering the electric field magnitude at each point of space, which should be zero inside the cage, we have a method for determining whether each point is inside or outside the input triangle model.

Because of its robustness in the presence of internal structures and small holes, our method has various potential applications as a beneficial preprocessing step in geometry processing. For example, mesh repair methods like [7] are sensitive to internal structures, which may output results of unnecessarily high genus due to topological redundancy. The results can be substantially improved if we selectively pass to the mesh repair pipeline only those triangles from the input triangle mesh close to the boundary of the volume generated by our method. Another use of our approach is in skeleton extraction. Skeletons are important shape descriptors for object representation and recognition, but they cannot be robustly extracted from mesh models with defects such as internal structures. In this case, we can convert the imperfect mesh to volumetric representation first, and then adopt existing robust and efficient methods of extracting skeletons from volume models like [8].

Thus, the main contribution of our work is a novel approach to generating a volumetric representation from a triangle soup, which may serve as a beneficial preprocessing step for various geometry processing applications. Although other volume construction methods like [4-5, 9] have been proposed, our method differs from them in having the following two main advantages: (i) our method is insensitive to small holes and internal structures, and (ii) our method produces a volume with low topological complexity.

The rest of the paper is organized as follows. In Section 2, we briefly review the related work. In Section 3, we overview our method; details are in Section 4. Experimental results are given in Section 5 and concluding remarks are given in Section 6.

2 Related Work

The key problem during volume model construction is to determine whether a given voxel in 3D space lies inside or outside the model. We thus briefly review and classify existing techniques in terms of how they make this decision.

To classify the interior and exterior of a polygonal model, approaches like [10-11] are used to assess the visibility of the surface from a surrounding sphere of cameras. However, simple viewpoint-visibility tests may fail for imperfect meshes, since some interior points may be visible through small holes in the surface, as for the Kitten model shown in Fig.1(a).

In [3, 12], triangle meshes are voxelized using those grid cells that intersect the model surface, which therefore only generates a thin-shelled volumetric representation. There is thus no explicit representation of inside and outside, only the boundary, which is unsuitable for many applications. To fill the interior of the model, a boundary filling algorithm is used in [4] after voxelizing triangles in a similar way. Similarly, in [13], an octree is constructed and the inside/outside property of cells is determined using a robust seed algorithm. However, these two methods fail if the input contains defects in the form of gaps in the input triangle mesh.

In [5], a signed volume is generated using a signed distance field, computed as the minimum signed distance from each cell to the input triangles. The authors also use an octree structure to adaptively sample the field. Fast computation of this field using graphics hardware is presented in [14]. However, these methods assume that the input triangle mesh is consistently oriented, which is not true for all input meshes, and is not assumed by our method.

In [15], an implicit representation of the object's surface is constructed from oriented points using radial basis functions. This allows holes to be smoothly filled by extrapolation. Alternatively, surface reconstruction from oriented points can be formulated as a Poisson problem[16], which yields a well-conditioned sparse linear system similar to ours. Yet another approach is given in [17], where a moving least-squares method is used to interpolate gaps between the input polygons. However, these methods again require consistent orientation of the input data, and furthermore, any unwanted internal triangles are retained.

Nooruddin and Turk[9] give two methods of voxelization based on different principles, parity-counting and ray-stabbing. The former method simply generalizes to 3D the well-known method of determining whether a point is interior to a polygon in 2D. The ray-stabbing method involves casting rays from each grid cell and voting based on intersections on the ray with the model. Neither method is robust as small gaps in the surface may cause a large portion of the cells at a distant location to be misclassified. They may also be confused by extra unwanted internal triangles.

A robust method which can guarantee to produce a closed model from a triangle soup is given in [7]. However, as noted by the author, if the input model contains internal structures, the output may be of high genus, which is undesirable in most applications. Our method does not suffer from this problem, and is observed to generate volumes with low topological complexity.

3 Overview

Our volume construction method is motivated by the *Faraday cage* from physics[18], and uses the approach illustrated in Fig.2. Suppose our input object is an approximately closed electrical conductor, whose potential is set to zero. A second conductor is placed outside it and surrounding it, and given a higher potential, generating an electric field. Due to the shielding effect of the inner conductor, the electric field magnitude should be approximately zero inside it. By comparing the electric field magnitude to a small threshold, we get a method for determining whether a given point lies inside or outside the input model. Obviously, there maybe places outside the model that could possibly have zero gradient magnitude. For highly non-convex objects, if the external conductor has a rather different shape to the inner conductor, the electric field may also be quite small in some places outside. To ensure that the method works well for such objects, we first compute a coarse object contour using an enclosing cube as the external conductor. Then we dilate the coarse object contour (i.e., offset it outwards) to get a new external conductor with a similar shape to the original object, and fairly close to it. We then use the method again with this new outer conductor to get a more accurate location for the object boundary.

Thus, our volume construction method is composed of three steps, which are detailed in Section 4:

• *Discretization.* During discretization, we embed the input triangle model in a tight (but not contacting) bounding cube and an initial octree representation of the input triangle model is computed by marking voxels that intersect with the triangles as *boundary voxels*. The depth of the octree is specified by the user or a default value is calculated from the triangle size, as explained later.

• *Coarse Contour Generation.* To get a contour for the outer conductor that is a similar shape and close to the input model, a harmonic field is coarsely constructed inside the bounding cube with potential value zero on boundary voxels (those containing triangles) and a constant higher potential value on the boundary of the cube. We make use of the octree structure to perform fast computation. We extract those voxels whose

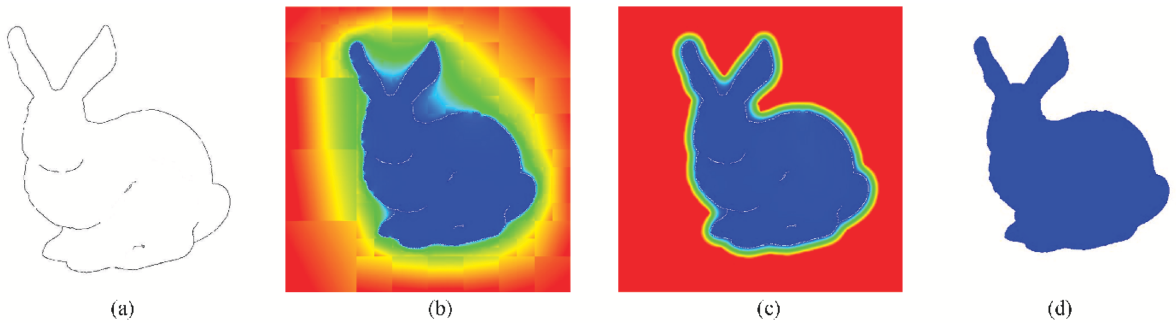


Fig.2. Steps in our method (illustrated in 2D for simplicity). (a) Input model. (b) Discretization and coarse field computation on a non-uniform grid. (c) Offset contour and more accurate field generation. (d) Extracted voxels for which field gradient is below a threshold.

field value is below a threshold as an initial estimation of the volume to be voxelized. We next apply a morphological dilation operation to this volume, to get a new coarse contour which has a similar shape to the object. Using this in place of the cube as the outer conductor allows us to get an almost uniform field between inner and outer conductors, even for highly non-convex objects, allowing us to straightforwardly use thresholding later to determine a more accurate boundary for the object.

• *Volume Extraction.* We now construct a more accurate harmonic field inside the new coarse contour. The potential value of the boundary voxels (containing triangles) is again set to zero, and a higher potential is applied to the new coarse contour. We use a uniform grid between the conductors to compute the harmonic field in this case, which gives more accurate results. Voxels whose field gradient is below a given threshold are extracted as the final volumetric representation of the input triangle soup.

4 Harmonic Field Model for Volume Construction

We now describe our harmonic field based volume construction method. We first define and give properties of harmonic fields. We then show how such a field can be used to construct a volume model from the triangle soup. We also elaborate on the details of our method in this section.

4.1 Definition and Properties

A harmonic field on a domain $\Omega \subset \mathbb{R}^3$ is a scalar field that satisfies Laplace's equation:

$$\nabla^2 V = \frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} + \frac{\partial^2 V}{\partial z^2} = 0. \tag{1}$$

If a scalar field is harmonic, locally the value of V at a point $\boldsymbol{r}(x, y, z)$ is the average value of V over a spherical surface S of radius R centered at it, which yields the following mean value formula:

$$V(\boldsymbol{r}) = \frac{1}{4\pi R^2} \oint_S V \, \mathrm{d}a. \tag{2}$$

In physics, an electric potential field is a typical example of such a field. Using electrostatics concepts to motivate the construction of volumetric harmonic fields also appeared in [19].

To solve Laplace's equation, two kinds of boundary conditions are typically used to determine the field. For Dirichlet boundary conditions, the values of V are given on certain boundaries, while for Neumann boundary conditions, the gradient of V is specified on the boundaries.

A harmonic field has several useful properties. From the mean value formula in (2), it turns out that V can have no local maxima or minima inside the boundary. Extremal values of V are sure to occur at the boundaries, which is called the *min-max* property[18].

For the purpose of volume construction, we set up the following harmonic field model. We first consider the continuous case, and then the solution to the model in the discrete case in Subsection 4.2. For a given triangle soup, let the set of triangles be $\{T_1, \ldots, T_n\}$. We place them within some chosen coarse contour. A harmonic field is then constructed inside the coarse contour using the following Dirichlet boundary conditions:

$$V(p) = \begin{cases} 0, & p \in T_i, \\ C, & p \in \partial B, \end{cases} \tag{3}$$

where $V(p)$ is the field value at boundary point p, and ∂B denotes the coarse contour surrounding the model. C is any positive potential value which we set to 1 for simplicity. Values at points inside the coarse contour are then determined by Laplace's equation.

Ideally, if the input model was a closed manifold mesh, the solution to Laplace's equation would yield $V = 0$ inside the mesh and the gradient ∇V would also be zero inside the mesh. Furthermore, in the region between the model and the coarse contour, the min-max property of harmonic fields tells us that there are no local minima or maxima of the field within this region.

Now, during the second stage of the process, when the gap D between the model and the coarse contour is small, and approximately uniform (because the coarse contour has been found by dilating the coarse model), the gradient magnitude in the region between the model and the coarse contour is almost constant, i.e., $\|\nabla V\| \approx C/D$. Therefore, we can use the gradient magnitude as a criterion to decide whether a given point is inside or outside the model by comparing it to a threshold.

In practice, the input is a triangle soup, which may contain gaps, holes and self-intersecting triangles. In such cases, the gradient magnitude of the field is not exactly zero inside the shape. However, such mesh defects have little impact on the harmonic field inside the model[18]. Therefore, inside/outside classification at a given point can still be carried out by examining the gradient magnitude there.

In an ideal case, the threshold could be set to 0. But as the gradient magnitude is not exactly zero inside the input model in practice, we set it to $\alpha C/D$. This more flexible criterion coincides with the physical model (where electric field is almost shielded) and practically allows us to fill small holes and internal structures, producing a volume with low topological complexity in most cases. We discuss the choice of α when presenting our experimental results in Section 5.

We next give details of applying the above idea to volume construction from triangle soup.

4.2 Volume Construction

4.2.1 Discretization

The first step of our method is discretization. We first embed the input model into a bounding cube of size $S = 1.1M$, where M is the model size. The discretization step constructs an octree representation within this cube corresponding to the input triangle soup. The octree is built incrementally as triangles are read in. Leaf node cells are those containing triangles, and are referred to as *boundary voxels*. The depth of the octree L may be selected by the user. Larger L results in lower distance error between the original mesh and the volumetric representation, but smaller L allows a more rapid computation. To capture fine details, the depth of the octree can be set by default so that the leaf cell size is equal to the average triangle size T, giving a default value of octree depth $L = \log_2(S/T)$. If we choose an octree level deeper than this, a triangle may be split across several voxels. Ideally, for perfect meshes, α could be chosen to be almost zero, and the distance error between the input mesh and the output volume model should improve as L increases. In practice, to fill gaps for imperfect meshes, α must be chosen at some finite non-zero values, which places an upper limit on the accuracy of the volume model. Further increases in L will not improve the distance between the volume model and the input mesh. Nevertheless, this limit to accuracy does not prevent the approach from becoming a useful preprocessing step, and e.g., performing internal structure removal for mesh repair. A 2D example of this process is shown in Fig.3; this figure should be compared to Fig.2.

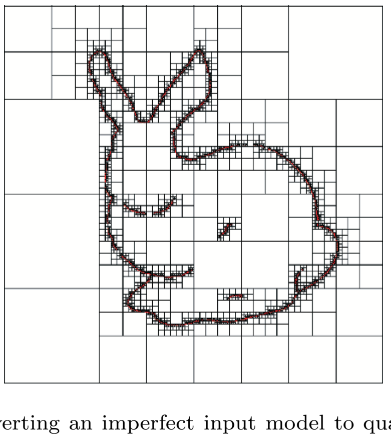


Fig.3. Converting an imperfect input model to quadtree representation. The boundary voxels are marked red.

4.2.2 Coarse Contour Generation

The second step of our method is to generate a coarse contour that is approximately a constant distance from the input model surface. There could be some regions outside the model that have almost zero gradient magnitude, such as the saddle of the harmonic field. Directly thresholding the gradient magnitude may lead to redundant components in result. However, it can be alleviated if we can estimate useful components at first. That is one of the reasons why we construct a coarse contour in this step. For this purpose, we first construct a harmonic field inside the bounding cube, using the octree grid to perform the computation. This is not at all accurate, but is sufficient to generate a suitable outer conductor. In detail, the potential value is set to 0 on leaf (boundary) voxels, and to the value 1 on the surface of the bounding cube. Voxels whose potential value is below a threshold (set to 0.01 in our experiments) are used as a coarse approximation to the volume of the original model. Note that, in this case, we use the potential instead of the field gradient to determine the approximate volume, as it has continuous variation and is sufficiently accurate for this purpose. Note that this step closes gaps in the original voxelized boundary, and discards any internal structure. We now apply a morphological dilation operation to this approximate volume to provide a new coarse contour which is at an almost constant distance from it. In the rest of this section, we further consider how to compute the coarse contour, then elaborate on the dilation operation.

Numerical solutions to Laplace's equation can be found using the finite difference method[20]. We use a seven point finite difference on a uniform grid to approximate the second derivative:

$$\begin{aligned}\nabla^2 V(x,y,z) \approx\; & V(x+h,y,z)+V(x-h,y,z)+\\ & V(x,y+h,z)+V(x,y-h,z)+\\ & V(x,y,z+h)+V(x,y,z-h)-\\ & 6V(x,y,z), \qquad (4)\end{aligned}$$

where h is the voxel size.

A finite difference method on a uniform grid in 3D is computationally expensive both in time and memory. However, as we only need a contour that approximates the input model, it is not necessary to accurately compute the field. Therefore, we exploit the octree based decomposition of the space found in Subsection 4.2.1 to efficiently compute the initial contour, as done in [21]. Let the octree cells be $\{c_1, \ldots, c_n\}$. For each cell c_i, let its neighbors be $N(c_i) = \{c_{ij}^n$,

$j = 1, 2, \ldots\}$. Each leaf cell is associated with a potential value $V(c_i)$ in the field. A discrete version of the mean value formula (2) can now be written as follows, so that the potential at each non-boundary cell satisfies Laplace's equation:

$$V(c_i)\sum_{j=1}^{|N(c_i)|}\omega_{ij} - \sum_{j=1}^{|N(c_i)|}\omega_{ij}V(c_{ij}^{n}) = 0, \qquad (5)$$

where ω_{ij} is the contact area between two neighboring cells c_i and c_j. Dirichlet boundary conditions are used: the boundary voxels (containing triangles) are given a potential of 0, while voxels at the boundary of the cube are connected to a virtual cell with potential 1.

We can rewrite these equations in matrix form as $\boldsymbol{Ax} = \boldsymbol{b}$, where $\boldsymbol{x} = \{V(c_1), \ldots, V(c_n)\}^{\mathrm{T}}$. $\boldsymbol{A}$ and $\boldsymbol{b}$ can be deduced from (5) and the boundary conditions. As shown in [22], the matrix $\boldsymbol{A}$ is positive semi-definite and the solution to this linear system is uniquely determined. Note that the total number of cells in the octree is $O(N)$, where N is the number of input triangles.

After the coarse field has been computed, the cells whose potential value lies below a threshold (set to 0.01 here) are extracted as the initial approximation of the volume model.

We next use a morphological dilation operation (see [9]) to expand the approximate volume outwards to obtain the new coarse contour. Following [9], we first compute a distance map from the approximate volume, and mark any voxel with distance less than the dilation distance D as being inside the new coarse contour. We set $D = 0.05S$ in our experiments, where S is the size of the bounding cube. As we start from a non-uniform grid, we must subdivide octree cells where necessary.

4.2.3 *Volume Extraction*

Once the new coarse contour has been computed, we are ready to more accurately determine a voxelization of the input model. We construct another harmonic field with potential value 0 on the boundary voxels (those containing triangles) and apply potential of 1 to the new dilated coarse contour surrounding the triangle soup.

Again we must solve the Laplace's equation. We now need a more accurate solution, and a uniform grid must be used inside the coarse contour. However, subdividing all octree cells inside the coarse contour is unnecessary. To reduce the time taken, we make use of the fact that some cells with large size and small potential value in the octree lie inside the model. We exclude cells that lie inside the coarse contour, with level smaller than a number K and potential value smaller than the threshold used in Subsection 4.2.2. Larger K leads to more rapid computation but may not be accurate enough, while smaller K is more accurate but more time-consuming. In practice, we find $K = L - 1$ (L is the octree level described in Subsection 4.2.1) to generally be a good compromise. The number of cells whose potential is to be determined depends on the size of the gap between the model and the coarse contour, which is still $O(N)$: the gap size does not depend on the number of triangles. Again, using (4), we set up and solve a linear system, using a uniform grid.

Finally, the voxels whose gradient magnitude is below a given threshold are extracted as interior voxels and included in the volumetric representation of the input triangle soup (see Fig.2(d)). The choice of the parameter α, which determines the threshold, is discussed in the next Section.

5 Experimental Results

We now present various experimental results produced by our harmonic field based volume construction method, and discuss them.

5.1 Implementation Details

Our volume construction algorithm was implemented in C++ on a Windows platform using an Intel Core 2 Duo 3GHz computer with 4GB RAM. The main steps of our method involve solving two linear system. In our implementation, we solve the linear system using an iterative overrelaxation method. For a given octree level, the number of cells in the cube is $O(N)$, where N is the number of input triangles. Therefore, the time complexity of computing the initial contour and volume extraction is $O(kN)$, where k is the number of iterations. We applied our method to various models; the times taken for a specified octree level are shown in Table 2.

5.2 Effect of Parameters

We first tested the effect of varying the parameter α. To compare the input triangle soup to the constructed volume, we place a large number of sample points (50 000 were used in our experiments) on the input triangle model, uniformly with respect to area, and compute the distance from the center of each voxel on the boundary of the volume to the closest sampled point from the triangles. The largest such distance is used as a measure of *distance error*, of how the volume model deviates from the input model. Different values of α and corresponding error measurements of some models are listed in Table 1. In this table, the only models free from defects are the Horse and Kitten models, and are thus the only models where this distance is truly meaningful. The errors reported are relative to the average

Table 1. Variation with α of Model Error, Measured Relative to Average Input Triangle Size

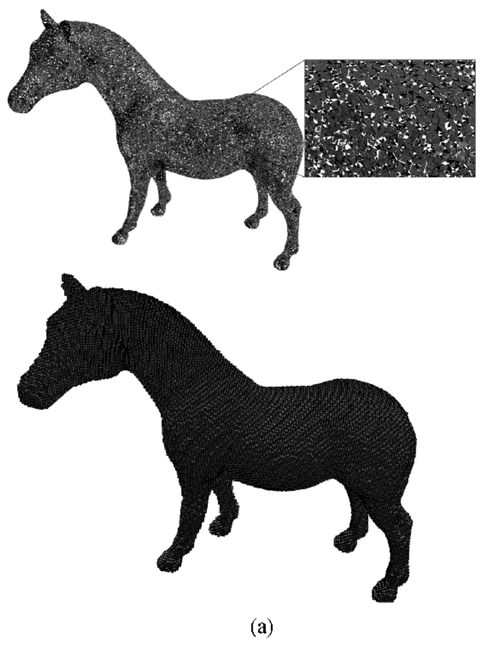
α	Sofa	Tank	Horse	Kitten	Jar
0.01	0.465	0.698	0.958	1.117	0.718
0.05	0.465	0.698	0.958	1.117	3.986
0.10	0.465	0.698	0.958	1.117	7.688
1.00	6.635	3.011	3.450	3.176	10.751
10.00	10.362	8.345	7.309	11.198	10.165

triangle size of each input model. Note that, when α is small, errors change little, which verifies our expectation that there is a shielding effect produced by the triangle model. When α exceeds 1.0 (a threshold almost equal to the gradient magnitude of the field inside the gap), larger errors are observed, as some voxels outside the model may now be included in it. As α gets even larger, errors increase, as further voxels outside the model are included. Since the errors change little when α is small, we empirically determine that we may set $\alpha = 0.05$ for most of the models in this paper. For the Jar model (Fig.8), which is highly non-convex, a smaller α is preferred; we discuss this further in Subsection 5.4. Note that the distance error is affected by both the octree level L and the parameter α. It is difficult to make a universal choice of α, as in principle, the smaller the α is, the more accurately we can approach the triangle mesh, but we must take into account the need to fill holes, and their size, and the nature of concavities of the model. There is also a tradeoff between efficiency and accuracy with respect to the octree level L, and we suggest setting L to the default value described in Subsection 4.2.1. This is sufficient in practice when our method is used as a preprocessing step for tasks such as internal structure removal for mesh repair, or skeleton extraction from volumetric models[8], where robustness is more important than accuracy.

5.3 Robustness and Comparison

We next show the results of applying our method to some typical but imperfect models like Kitten (Fig.1(a)), Horse (Fig.4(a)) and Fertility (Fig.5). To test the robustness of our method, we synthetically distorted the models by randomly perturbing positions of vertices and removing some faces. Observe that the constructed volumes exhibit a good appearance despite these defects. The genus of each model is listed in Table 2.

Fig.6 shows some models created using computer-aided design software. Note that these models contain self-intersections and internal structures, which is not unusual when triangle models are output by commercial CAD software. Fig.4(b) shows a Dragon model with certain internal structures that are difficult to detect. The method proposed in [7] is capable of handling these models but the generated results are of unnecessarily high genus due to topological redundancy in the form of handles, cavities, and disconnected components, as indicated in Table 2. Such complex topology can be simplified by methods such as those in [23], but it is more efficient to avoid producing them if possible when converting to volumetric representation. Our method is insensitive to such internal structures and is observed to generate volumes with low topological complexity (again, see Table 2). For imperfect meshes, there is no


(a)

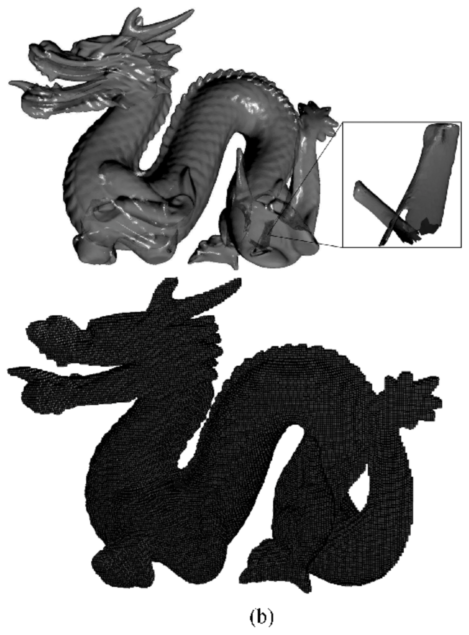

(b)

Fig.4. Results produced. (a) Defective Horse model. (b) Dragon model with internal structures.

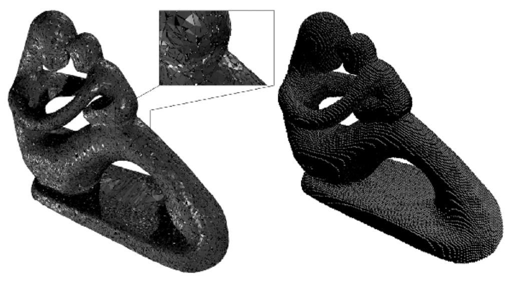


Fig.5. A defective Fertility model and generated volume model.

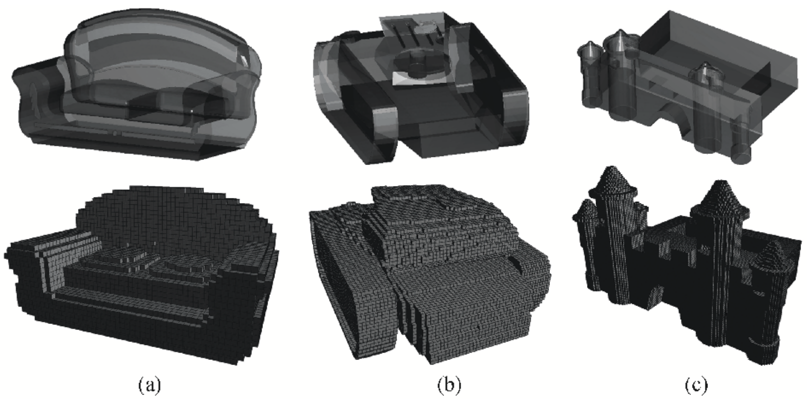


Fig.6. Results of applying our method to various CAD models.

Table 2. Performance for Various Models with Specified Octree Level

Model	Figure	Input Triangles (k)	Octree Level	Build Octree (s)	Generate Coarse Contour (s)	Extract Volume (s)	Output Genus, Our Method	Output Genus[7]
Sofa	6(a)	8	6	0.32	0.13	0.24	1	41
Tank	6(b)	4	7	0.46	0.78	0.86	4	54
Jar	8	19	7	1.08	2.06	3.01	0	0
Castle	6(c)	9	8	0.98	3.84	3.96	0	4
Kitten	1	30	8	1.80	10.85	11.64	1	1
Horse	4(a)	67	8	2.54	12.15	13.15	0	2
Dragon	4(b)	200	9	10.18	33.74	43.81	2	5
Fertility	5	480	9	19.63	45.23	55.41	4	4

ground truth of the correct genus, but the results of our method are certainly lower, and preferable in general. This makes our method suitable for severing as a preprocessing step for mesh repair methods, in which we feed only those triangles that are sufficiently close to the boundary of the volume to the subsequent mesh repair pipeline. Fig.7 shows an example. Before we apply the mesh repair method of [7], we selectively pass the triangles whose closest distances to the boundary of the volume are no larger than the distance error described in Subsection 5.2. Note that directly applying the method in [7] causes unnecessarily topological redundancy (also see Table 2). By using our method as a preprocessing step, we get a repaired model with lower topological complexity.

5.4 Limitations and Discussion

As shown by the previous experiments, our method is good at preserving global shape and is insensitive to small holes and internal structures of models. Therefore, in many cases, it produces a volume with low topological complexity. But it also has limitations. Due to the properties of harmonic fields, if the input model contains some parts that are very close to each other (as in the Spiral model in Fig.9) or deep concavities (as in the Jar model in Fig.8), the gradient magnitude is small in such regions. This can be seen clearly in our

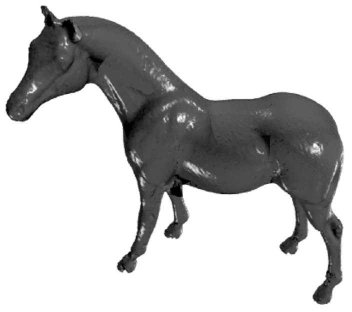


Fig.7. The result of repairing the Horse model using our method as a preprocessing step of [7].

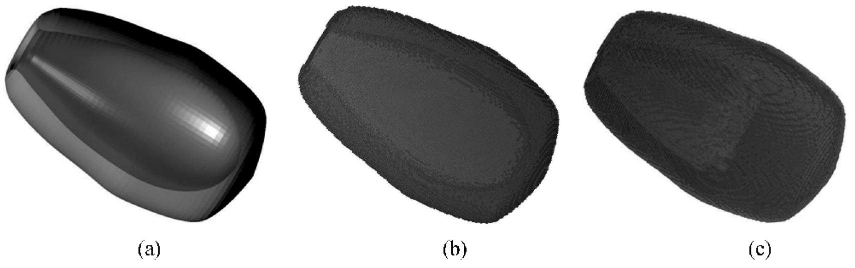


Fig.8. Jar model with a deep concavity (a), and generated volume models with (b) $\alpha = 0.01$ and (c) $\alpha = 0.10$.

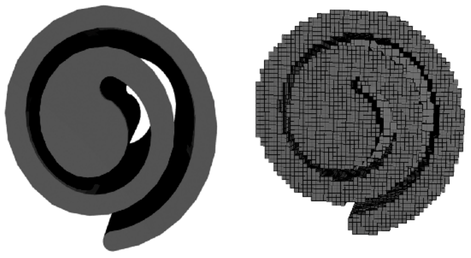


Fig.9. A mesh with some parts very close to each other, and the volume model generated by our method.

2D illustration in Fig.2(b). Note that the field varies slowly in concave regions, e.g., near the base of the two ears of the Bunny model: the fixed zero value boundary condition causes the potential to be almost constant in such concave regions.

As a result, it may be difficult to choose parameters to avoid such spaces being filled by our method. For example, in the Jar model in Fig.8, the gradient magnitude is small inside the Jar, with the result that if α is not chosen small enough, some of the interior space is filled by our method, resulting in the large errors shown in Table 1. Volume models generated with two different choices of α are shown in Fig.8. Notice that larger α causes the volume model to expand a little, particularly in the interior space. The application of our method to a further somewhat extreme case, the Spiral model, is shown in Fig.9. The field is almost constant inside the gap. In this case, choosing $\alpha = 0.10$ leads to our method filling the gap. (Of course, even if we can find a suitable α, L, the octree depth, must also be chosen large enough so that the voxel size is suitable compared to the size of the gap.) If α is set to a larger value, we can fill larger holes. However, the result may deviate from the original model too much if α is too large (see Table 1). Therefore, there is tradeoff between filling holes and accuracy. We intend to investigate the relation between α and the size of largest holes that can be filled in the future.

Actually, as these two examples are perfect meshes, we can choose α to be very small to alleviate this phenomenon. However, for real models, if we choose α too small, small gaps in the mesh elsewhere may not be satisfactorily filled. In fact, such a problem is almost impossible to overcome. If gaps in the mesh are of a similar size to real features, it may not be possible to distinguish an intentional feature from missing data. Choosing α large enough to avoid this issue will cause the generated volume to expand a little compared to the original triangle soup. However, this has little impact when our method is used as a preprocessing step. For mesh repair, we can discard internal triangles by keeping those triangles that are sufficiently close to the boundary of the volume model, and then use a method like [7], which is complementary in being excellent at handling missing triangles, but has poor behavior in the case of internal structures. For skeleton extraction from volume models, methods like [8] only require an approximate input to capture the nature of the shape skeleton.

6 Conclusion and Future Work

In this paper, we present a novel method to construct volume model from triangle soups, which has applications as a beneficial preprocessing step for various geometry processing tasks like mesh repair and skeleton extraction. In our method, we consider the input triangles as forming a more-or-less closed Faraday cage, with potential zero, and a second conductor, outside it and surrounding it, is given a higher potential. We may then determine whether a given point in space is inside or outside the input model by considering the resulting electric field. Our method is insensitive to small holes and internal structures and is observed to generate volumes with low topological complexity. In the future, we intend to further develop the method. One possible solution to the issue of accuracy may be to adaptively vary α across the mesh, assuming we can distinguish what is a concavity from an unwanted hole. Apart from that, we think the gradient magnitude of the harmonic field is related to the curvature and we intend to investigate its usage in shape analysis.

References

[1] Lorensen W E, Cline H E. Marching cubes: A high resolution 3D surface construction algorithm. In *Proc. SIGGRAPH 1987*, Anaheim, USA, July 27-31, 1987, pp.163-169.

[2] Ju T, Losasso F, Schaefer S, Warren J. Dual contouring of hermite data. In *Proc. SIGGRAPH 2002*, San Antonio, USA, July 21-36, 2002, pp.339-346.

[3] Huang J, Yagel R, Filippov V, Kurzion Y. An accurate method for voxelizing polygon meshes. In *Proc. 1998 IEEE Symposium on Volume Visualization*, Research Triangle Park, USA, Oct. 24, 1998, pp.119-126.

[4] Oomes S, Snoeren P, Dijkstra T. 3D shape representation: Transforming polygons into voxels. In *Proc. SCALE-SPACE 1997*, Utrecht, The Netherlands, July 2-4, 1997, pp.349-352.

[5] 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 *Proc. SIGGRAPH 2000*, New Orleans, USA, July 23-28, 2000, pp.249-254.

[6] Ju T. Fixing geometric errors on polygonal models: A survey. *J. Comput. Sci. Technol.*, 2009, 24(1): 19-29.

[7] Ju T. Robust repair of polygonal models. *ACM Trans. Graph.*, 2004, 23(3): 888-895.

[8] Ju T, Baker M L, Chiu W. Computing a family of skeletons of volumetric models for shape description. *Computer-Aided Design*, 2007, 39(5): 352-360.

[9] Nooruddin F S, Turk G. Simplification and repair of polygonal models using volumetric techniques. *IEEE Transactions on*

Visualization and Computer Graphics, 2003, 9(2): 191-205.

[10] Nooruddin F S, Turk G. Interior/exterior classification of polygonal models. In *Proc. VIS 2000*, Salt Lake City, USA, Oct. 8-13, 2000, pp.415-422.

[11] Zhang E, Turk G. Visibility-guided simplification. In *Proc. VIS 2002*, Boston, USA, Oct. 27-Nov. 1, 2002, pp.267-274.

[12] Dachille F, Kaufman A E. Incremental triangle voxelization. In *Proc. Graphics Interface 2000*, Montreal, Canada, May 15-17, 2000, pp.205-212.

[13] Andújar C, Brunet P, Ayala D. Topology-reducing surface simplification using a discrete solid representation. *ACM Trans. Graph.*, 2002, 21(2): 88-105.

[14] Sigg C, Peikert R, Gross M. Signed distance transform using graphics hardware. In *Proc. VIS 2003*, Seattle, USA, October 19-24, 2003, pp.83-90.

[15] Carr J C, Beatson R K, Cherrie J B, Mitchell T J, Fright W R, McCallum B C, Evans T R. Reconstruction and representation of 3D objects with radial basis functions. In *Proc. SIGGRAPH 2001*, Los Angeles, USA, Aug. 12-17, 2001, pp.67-76.

[16] Kazhdan M, Bolitho M, Hoppe H. Poisson surface reconstruction. In *Proc. SGP 2006*, Cagliari, Italy, June 26-28, 2006, pp.61-70.

[17] Shen C, O'Brien J F, Shewchuk J R. Interpolating and approximating implicit surfaces from polygon soup. In *Proc. SIGGRAPH 2004*, Los Angeles, USA, Aug. 8-12, 2004, pp.896-904.

[18] Griffiths D J, Inglefield C. Introduction to Electrodynamics. Prentice Hall, 1999.

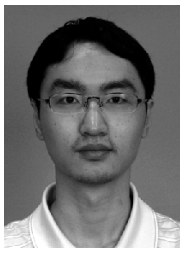
[19] Li X, Guo X, Wang H, He Y, Gu X, Qin H. Harmonic volumetric mapping for solid modeling applications. In *Proc. SPM 2007*, Beijing, China, June 4-6, 2007, pp.109-120.

[20] Kreyszig E. Advanced Engineering Mathematics. John Wiley, 2005.

[21] Rosell J, Iniguez P. A hierarchical and dynamic method to compute harmonic functions for constrained motion planning. In *Proc. IROS 2002*, Lausanne, Switzerland, Sept. 30-Oct. 4, 2002, pp.2335-2340.

[22] Grady L. Random walks for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2006, 28(11): 1768-1783.

[23] Zhou Q Y, Ju T, Hu S M. Topology repair of solid models using skeletons. *IEEE Transactions on Visualization and Computer Graphics*, 2007, 13(4): 675-685.



Chao-Hui Shen received the Bachelor's degree in computer science from Tsinghua University, Beijing, in 2008. He is currently a Ph.D. candidate in the Department of Computer Science and Technology, Tsinghua University. His research interests include computer graphics, geometric modeling, and processing.

Guo-Xin Zhang received the Bachelor's degree in computer science from Tsinghua University, Beijing, in 2007, and is currently a Ph.D. candidate in the Department of Computer Science and Technology, Tsinghua University. His research interests include computer graphics, geometric modeling, and image processing. He is a student member of CCF.

Yu-Kun Lai received his Bachelor's and Ph.D. degrees in computer science from Tsinghua University in 2003 and 2008, respectively. He is currently a lecturer of visual computing in the School of Computer Science and Informatics, Cardiff University, Wales, UK. His research interests include computer graphics, geometry processing, computer-aided geometric design and computer vision.

Shi-Min Hu is currently a chair professor in the Department of Computer Science and Technology, Tsinghua University, Beijing. He received the Ph.D. degree from Zhejiang University in 1996. His research interests include digital geometry processing, video processing, rendering, computer animation, and computer-aided geometric design. He is on the editorial boards of Computer Aided Design (Elsevier) and Journal of Computer Science and Technology (Springer). He is a senior member of CCF.

Ralph R. Martin has been working in geometric computing since 1979, obtaining his Ph.D. degree in 1983 from Cambridge University. Since then he has been at Cardiff University, as professor after 2000. He is also a guest professor at Tsinghua University and Shandong University in China, and the deputy director of Scientific Programmes of the Welsh Institute of Visual Computing. His publications include about 200 papers and 10 books covering such topics as solid modelling, surface modelling, reverse engineering, intelligent sketch input, mesh processing, video processing, computer graphics, vision based geometric inspection, and geometric reasoning. He is a fellow of the Institute of Mathematics and Its Applications, and a member of the British Computer Society. He is on the editorial boards of Computer Aided Design, Computer Aided Geometric Design, the International Journal of Shape Modelling, CAD and Applications, and the International Journal of CADCAM.