
Hybrid Approach For Repair of Geometry With Complex Topology

Amitesh Kumar¹ and Alan M. Shih²

¹ University of Alabama at Birmingham, AL, USA amitesh@uab.edu

² University of Alabama at Birmingham, AL, USA ashih@uab.edu

Summary. A discrete geometry can have artifacts such as holes, intersections, non-manifold edges, mesh fragment among other defects depending upon its origin. These kinds of defects sometime cause the geometry to be unsuitable for any further use in computational simulation in absence of a satisfactory geometry repair technique. There are two main approaches to geometry repair, surface based and volume based. Surface based approaches, in general, provide better quality results when they work but require that the input model already satisfies certain quality requirements to be able to guarantee a valid output. Many of these requirements cannot even be met or checked automatically. Volume based approaches, in general, can guarantee watertightness but they usually significantly change the underlying model in this process and are computationally more expensive.

A hybrid approach for mesh repair, combining surface based approach and a two step volume based approach is being presented in this paper. The two steps in the volume based approach are heat diffusion solution as the first step and Poisson surface reconstruction from oriented points in 3D space as the second step. This approach presents a reliable method for the repair of those defective discrete surface geometries which otherwise could not be completely repaired using existing surface-based techniques due to geometric and topological complexities presented as holes, isles, intersections and small overlaps.

Keywords: geometry repair, volume based approach, volumetric diffusion.

1 Introduction

A preferred approach for geometry creation is to do so with mesh generation tools directly. This approach in theory can reduce the conversion errors which are introduced during geometry translation between various Computer-Aided Design (CAD) packages and the mesh generation tools. However, most of the mesh generation tools do not have sophisticated solid modeling capabilities as the CAD systems do. As a result, geometries for most of the sophisticated real-world applications are first produced on CAD systems and then imported

into the mesh generation tools. Geometry can also come from other sources in discrete forms. For example, scanned data using range scanning devices such as a laser scanner, can produce fairly accurate geometric model defined by point clouds, which can be turned into discrete elements. In the image-based, patient-specific biomedical applications, geometry can be produced by reconstructing from the segmented contours on each image slice. Regardless of the source of the geometry data, or the form in which they are represented (parametrically or discretely), the geometries obtained can have many defects due to the data conversion errors or ambiguities in the process. One of the major defects encountered in many geometries is the lack of water-tightness. A surface mesh, or geometry, is considered to be not watertight or non-manifold in the following two cases:

1. It has edges which are shared by only one polygon, i.e. the edges lie on the boundary. The occurrences of a set of connected boundary edges create a hole in the surface.
2. It has edges which are shared by more than two polygons. This kind of non-manifold mostly occurs in CAD applications due to improper stitching of surfaces patched together to generate a desired geometric model.

The most common type of mesh defects or artifacts encountered are holes or isles, singular vertex, handle, gaps, overlaps, inconsistent orientation, complex edges and intersections [1]. A number of research papers have tried different approaches in an attempt to address this issue using various automated and intelligent methods. Those approaches broadly fall in two main categories: volume-based repair methods and surface-based repair methods.

This research work tries to address surface defects of the type of topologically simple as well as complex holes in a discrete geometry, by using diffusion equation as a system of governing equation to obtain a convergent solution in the volumetric domain. That would help us in generating a set of well placed, regularly sampled and correctly oriented points in the areas of discontinuity, which are later used in surface reconstruction using Poisson's Surface Reconstruction technique [29], [31]. In this work, a robust and fully automatic hybrid methods is being presented which utilizes the strengths of both surface based and volume based techniques and can handle dirty geometry with holes, isles and intersections.

2 Previous Works

2.1 Surface Based Methods

Surface based repair methods ([2]–[12]) operate directly on the input data and hence need to explicitly identify and resolve artifacts on the surface. As an example, narrow gaps could be removed by snapping boundary elements (vertices and edges) onto each other or by stitching triangle strips in between

the gap. Holes could also be closed by a triangulation that minimizes a certain error term. Intersections could be located and resolved by explicitly splitting edges and triangles although robustly resolving intersections is usually a very expensive process in terms of computational cost due to numerical accuracies. Surface based repair methods only minimally perturb the input model and are able to preserve the model structure in areas that are away from artifacts. In particular, structure that is encoded in the connectivity of the input (e.g. curvature lines) or material properties that are associated with triangles or vertices are usually well preserved in the areas away from the artifact. Furthermore, these algorithms introduce only a limited number of additional triangles [1].

These class of methods, however, usually require that the input model already satisfies certain quality requirements to be able to guarantee a valid output. Many of these requirements cannot be guaranteed or even be checked automatically hence these algorithms are rarely fully automatic and require manual post-processing. Other artifacts, like gaps between two closed connected components of the input model that are geometrically close to each other, are quite difficult to identify. As a result a guaranteed repair using only Surface based technique is not always possible.

Turk and Levoy [2] proposed a mesh zipping algorithm tailored to fuse range images using surface-based approach. Barequet and Kumar [3] and Barequet and Sharir [4] describe the use of an interactive system that closes small gaps, generated by CAD programs while joining the surfaces by stitching and triangulation within the hole. Borodin *et al.* [6] propose a progressive gap closing algorithm that works by vertex edge contraction accompanied with insertion of vertices on the boundary edges and progressively contracting the edge. Leipa [7] describes a method for filling holes by a weight-based hole triangulation, mesh refinement based on the Delaunay criterion and mesh fairing based on energy minimization. Jun [8] describes an algorithm based on stitching a planar projection of a complex hole in 2D space and projecting the stitched patch back in 3D space. Branch *et al.* [9] suggest a method for filling holes in triangular meshes using a local radial basis function. Pernot *et al.* [10] describe a method to fill holes by first cleaning up the geometry by removing unwanted triangles and then filling the holes with a patch of the disk topology by inserting a point in the middle and connecting all the nodes on the hole boundary with it.

The usability of most of these algorithms is constrained by their assumptions related with shapes, sizes, or sources of the holes and other defects on the mesh surface.

2.2 Volume Based Methods

The key to all volume based methods lies in converting a surface model into an intermediate volumetric representation. Volume based techniques are then

used to classify and orient all voxels of the volumetric representation representing whether the particular voxel lies inside, outside or on the surface of the geometry. This classification determines and defines the topology and geometry of the reconstructed model which is extracted out of the intermediate volume representation. Due to their very nature, volumetric representations do not allow for artifacts like intersections, holes, gaps, overlaps or inconsistent normal orientation [1]. Volumetric algorithms are typically fully automatic and produce watertight models and depending on the type of volume and they can often be implemented in a very robust manner. Surface mesh from the intermediate volumetric representation is usually extracted using a surface extraction technique like Marching Cubes [13].

Volume based approaches ([14] – [20]) to mesh repair also produce some undesirable effects. The conversion to and from a volume leads to a resampling of the model. It often introduces aliasing artifacts, loss of model features and destroys any structure that might have been present in the connectivity of the input model [1]. The number of triangles in the output of a volumetric algorithm is usually much higher than that of the input model and thus has to be decimated in a post-processing step. The quality of the output triangles often degrades in this process as well. As a result, quality of the repaired mesh needs to be improved as a post-processing step to the volume based geometry repair techniques. Finally, volumetric representations are quite memory intensive so it is hard to run them at high resolutions.

Curless and Levoy [14] proposed a hole filling method based on volumetric diffusion optimized for range scanning devices. Murali and Funkhouser [15] use Spatial subdivision using BSP-Tree and determination of solid regions using region adjacency relationship to construct a set of polygons from the solid regions. Davis *et al.* [16] presented a method of in which the voxels of the volumetric representation of a surface mesh is classified as inside or outside with the help of distance map generated using line of sight information. Nooruddin and Turk [17] suggested the use of parity count and ray stabbing to repair the intermediate volumetric representation. Ju [18] presented a method for generating signs of voxels using octree. Bischoff *et al.* [19] proposed an improved volumetric technique to repair arbitrary polygonal soup using adaptive octree. Podolak and Rusinkiewicz [20] described a method using atomic volumes based on hierarchical non-intersecting graph representation in 3D space.

3 Our Work

Our work is based on a hybrid approach to mesh Repair which uses both Surface and volume based techniques.

The surface based technique which is being used in our hybrid approach is closely related to our previous work [11] and [12] with some improvements. The main features of it are:

- Support for non-manifold mesh.

- Explicit identification of holes and their sorting based on their sizes in terms of number of edges.
- Octree based search for locating points, edges and triangles.
- Sizing and refinement controlled by point insertion at centroid and edge swapping based on Delaunay criteria.
- Hole patching using localized NURBS based surface definition.
- Hole patching only supported for discrete geometries with simple topologies. The presence of isles in the hole region is neither detected nor supported in the surface repair mode.

Our surface based technique, in general, would produce good results when it would work. However as is the case with all other surface based techniques, they are not robust and require that the input model already satisfies certain quality requirements to be able to guarantee a valid output. Many of these requirements cannot even be met or checked automatically for all the input geometries. As a result when the surface based techniques fail we use the output of the surface based technique as an input to our volume based repair algorithm.

The work most closely related with our volume based technique appears in Davis *et al.* [16]. However our method is different in many ways. The main features of our volume based technique are as following:

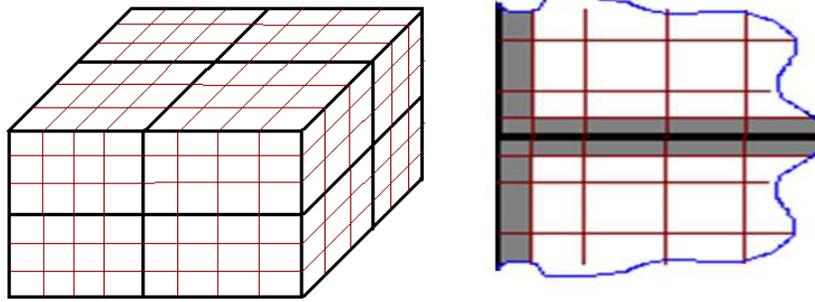
- No assumption regarding source of model and hence no additional requirements such as line of sight information.
- Identification of independent and non-intersecting solution columns in 3D space.
- Convergent Diffusions equation based solution in each of the solution columns.
- Extraction of surface patch in the hole region away from the original input surface.
- Extraction of point and their normals from input geometries and surface fragments, in case of dirty geometries, and the extracted patch.
- Poisson surface reconstruction to get a watertight output model using the extracted oriented point set.

The above mentioned steps would be discussed in the next few sections. We would be using three discrete geometries namely a modified Stanford Bunny [35] with a few extra holes, Laurent Hand [36] and Chinese Lion [37] to demonstrate the results of our work.

3.1 Voxelization of Discrete Geometry

The first step in volume based repair method is to convert the surface mesh into a volumetric mesh. In this research, for simplicity, we have used a Cartesian grid to represent the data. The voxelization is only performed in non-intersection regions near the surface defects. We call each of these non-intersecting region as a “*solution column*”. Each of the solution column is

represented as a uniform Cartesian grid. A Cartesian grid can be represented as a block of 3D tiles as shown in figure 1(a). The voxel size is a function of the average edge lengths of the triangles in the column so the points and triangles generated with this method have the same sample density as those in its neighborhood in the surface mesh. The tiles are padded with an extra layer of ghost cells along the boundary as shown in figure 1(b). The memory allocation is completely dynamic and the information exchange between the contiguous tiles is completely hidden from the user with an abstraction layer. This kind of memory allocation for representation of the Cartesian mesh has two major benefits:



(a) A Cartesian Grid composed of multiple blocks in 3D (b) Ghost cells at the interface of two tiles shown in gray color in a 2D Cartesian grid

Fig. 1. Description of the cartesian solution space

1. Depending on the cache size and the size of each tile, this configuration may speed up the diffusion solver performance due to cache effect despite the cost associated with information exchange between ghost cells at the end of every iteration.
2. The solver requires an exact temporary copy of the tile in the intermediate step to compute and transfer data. If we have a small tile size then only a small intermediate amount of memory would be needed. This would prevent duplication of larger tile, that would instead need to fit the whole model.

The information exchange between ghost cells is completely hidden from the user. Hence the user does not experience any differences while using the APIs compared to the situation when whole block is composed of a single tile. The discrete geometry is composed of individual triangles. Intersection of each triangle with the *voxels* of the Cartesian grid lying within its bounding box is checked using AABB Triangle-Box intersection algorithm [32], [33]. All the

voxels found intersecting with triangles are masked as “*model voxels*” with a static value of ‘0’. All the voxels adjacent to the models with ‘0’ value along the positive normal of the triangle plane are masked as “*heated*” and given a static positive value of “+1” while all the voxels in the opposite direction of the positive normal are masked as “*cold*” and given a negative value of “-1”. This process creates a voxelized representation of the discrete geometry having a neutral value (“0”) sandwiched between hot and cold side. The rest of the voxels are dynamic and can change values during the solution. In our implementation, we assume that the input discrete geometry along with all present input surface fragments, have consistent orientation with respect to each other, for the purpose of defining the boundary conditions. The input geometry may not have normal information present. As a result normal generation is performed at every vertex of the input geometry in order to sign those unsigned voxels of the cartesian grid column which are touching the embedded geometry. In the hybrid approach being presented in this work, parts of the original input geometry along with all the surface fragments as well as the output surface patches from surface based approach are embedded in the voxelized non-intersecting cartesian grid columns prior to generating a diffusion equation based solution in those columns. Figure 2(a) shows non-intersecting solution columns on Stanford Bunny.

3.2 Diffusion Equation

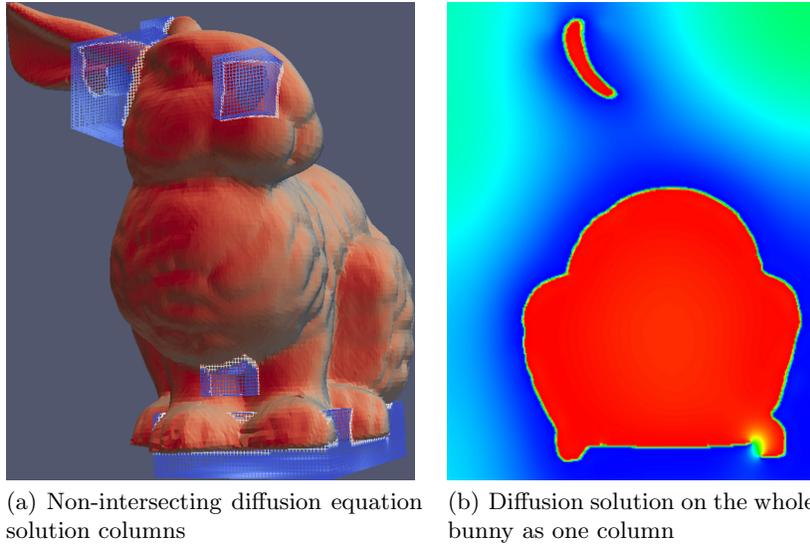


Fig. 2. Diffusion Solutions for the Stanford Bunny

Diffusion is a time-dependent process, constituted by random motion of given entities and causing the statistical distribution of these entities to spread in space. The concept of diffusion is tied to notion of mass transfer, driven by a concentration gradient. The diffusion equations can be obtained easily from Continuity equation when combined with the Fick's first law, which assumes that the flux of the diffusing material in any part of the system is proportional to the local density gradient [34]. Diffusion equation is a partial differential equation continuous in both space and time which describes density fluctuations in a material undergoing diffusion. Diffusion equation can be simplified and written as,

$$\frac{\partial \phi}{\partial t} = \alpha \Delta^2 \phi + S \quad (1)$$

Where α is a constant and S is a source term. In our formulation we assume that there are no source terms, Hence the equation becomes,

$$\frac{\partial \phi}{\partial t} = \alpha \Delta^2 \phi \quad (2)$$

Using *forward difference* scheme in time and *central difference* scheme in space based on Taylor's series expansion, the diffusion equation for a Cartesian grid simplifies to,

$$\phi_{i,j,k}^{n+1} = \phi_{i,j,k}^n + \alpha \Delta t \left[\begin{aligned} & \left(\frac{\phi_{i+1,j,k}^n - \phi_{i,j,k}^n}{\Delta x^2} - \frac{\phi_{i,j,k}^n - \phi_{i-1,j,k}^n}{\Delta x^2} \right) \\ & + \left(\frac{\phi_{i,j+1,k}^n - \phi_{i,j,k}^n}{\Delta y^2} - \frac{\phi_{i,j,k}^n - \phi_{i,j-1,k}^n}{\Delta y^2} \right) \\ & + \left(\frac{\phi_{i,j,k+1}^n - \phi_{i,j,k}^n}{\Delta z^2} - \frac{\phi_{i,j,k}^n - \phi_{i,j,k-1}^n}{\Delta z^2} \right) \end{aligned} \right] \quad (3)$$

where $\Delta x, \Delta y$ and Δz are the grid spacings of a Cartesian grid in x, y and z directions respectively. Equation (3) provides the explicit numerical solution which is central difference in space and forward difference in the time domain for the diffusion equation given in equation (2).

After performing stability analysis on our numerical scheme using Von Neumanns analysis and Courant–Friedrichs–Lewy (CFL) condition for the stability of the numerical scheme in three dimensions (3D), we further come with the inequality defining the values for the constants given in equation (3).

$$0 \leq \alpha \Delta t < \frac{\Delta x_{min}^2}{6} \quad (4)$$

Equation (4) also implies that the scheme is numerically stable for a value of $\alpha \Delta t$ satisfying the above given condition.

We define the Cartesian grid with the analogy of a curved thin plate in 3D space whose one side is heated while the other side is cold. Diffusion equations presented earlier are used to find a steady state solution. For illustration purpose, figure 2(b) shows the rendering of one slice of Stanford bunny after

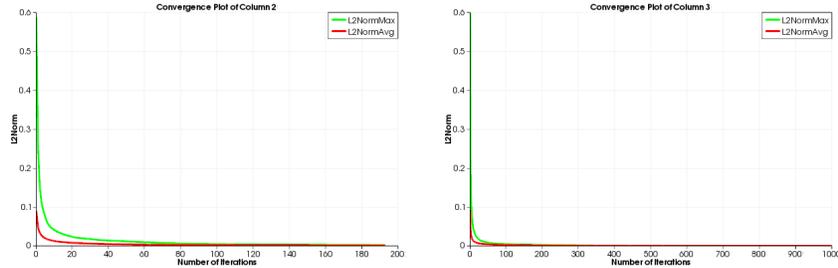
120 iterations when the whole model is used as one solution column. However in practice, the whole model is rarely, if ever, embedded as whole in one single solution column as represented in 2(a).

$$l_2norm_{\max}^{n+1} = \max \left(\sqrt{(\phi_{i,j,k}^{n+1} - \phi_{i,j,k}^n)^2} \right) \quad (5)$$

$$l_2norm_{\text{avg}}^{n+1} = \frac{1}{i \times j \times k} \sum \sqrt{(\phi_{i,j,k}^{n+1} - \phi_{i,j,k}^n)^2} \quad (6)$$

$$\%l_2norm_{\text{change}}^{n+1} = \frac{\text{abs}(l_2norm_{\text{avg}}^{n+1} - l_2norm_{\text{avg}}^n)}{l_2norm_{\text{avg}}^{n+1}} \times 100.0 \quad (7)$$

Equations (5), (6) and (7) define three measures of change for each successive iteration for our solution. These are $l_2norm_{\max}^{n+1}$, $l_2norm_{\text{avg}}^{n+1}$ and $\%l_2norm_{\text{change}}^{n+1}$ respectively. In our case, we use $\%l_2norm_{\text{change}}^{n+1}$ as our measure of change and use an arbitrary number (0.5%) or 1000 iterations, whichever happens earlier, to determine whether or not convergence has been reached for a particular solution column. Figures 3(a) and 3(b) show convergence plots of columns 2 and 3 for the Stanford Bunny. In both the cases the plots are asymptotic and seem to converge within a few iterations. However looking at the log plots shown in corresponding figures 4(a) and 4(b), it becomes clear that actual convergence is reached much later.



(a) Convergence plots of l_2norm_{avg} and l_2norm_{\max} of solution Column 2 (b) Convergence plots of l_2norm_{avg} and l_2norm_{\max} of solution Column 3

Fig. 3. Convergence plots of two solution columns of Stanford Bunny

Explicit schemes are known to be notoriously slow for convergence. The convergence becomes slower as we refine the grid to a finer resolution. To circumvent this problem, we provide the capability to be able to use a primitive algebraic interpolation approach using coarse grid to initialize the flow field for the fine grid. This allows the overall solution field to be initialized using the preliminary solution from the coarser grid, which can reduce the time needed to convergence. First the solver is run on a coarse grid for a number

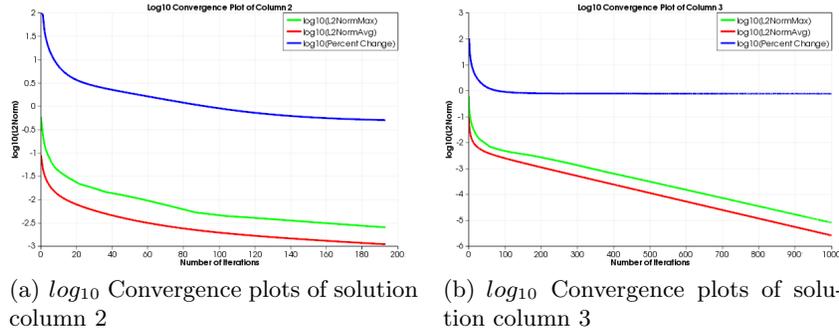


Fig. 4. \log_{10} Convergence plots of two solution columns of Stanford Bunny

of iterations and then its solution is interpolated onto the finer Cartesian grid as initial values for faster convergence. It should however be noted that, we are not using algebraic interpolation in the examples being presented in this work.

3.3 Extraction of Consistently Oriented Surfaces and PointSet

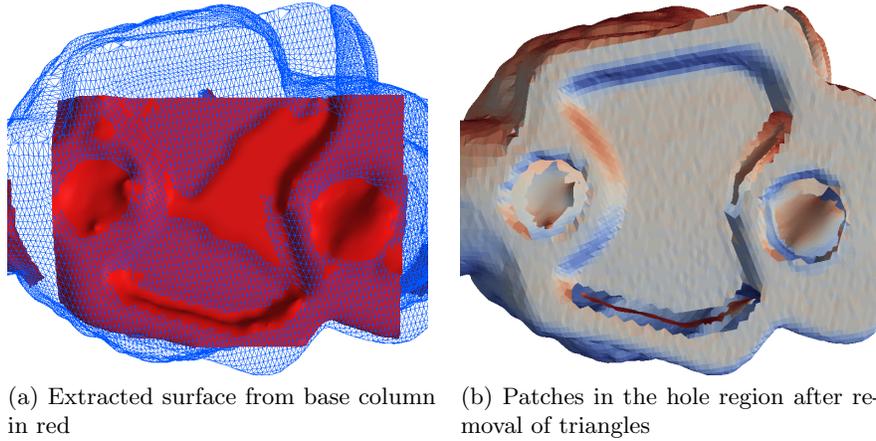


Fig. 5. Extracted surface and patches after diffusion equation solution in the base column

Zero-set of the numerical solution for the diffusion equation as described in the previous section provides a closed surface. This zero-set surface is otherwise only open at the places where it intersects with the extremities of the Cartesian grid. Extraction of the surfaces using a suitable contouring method

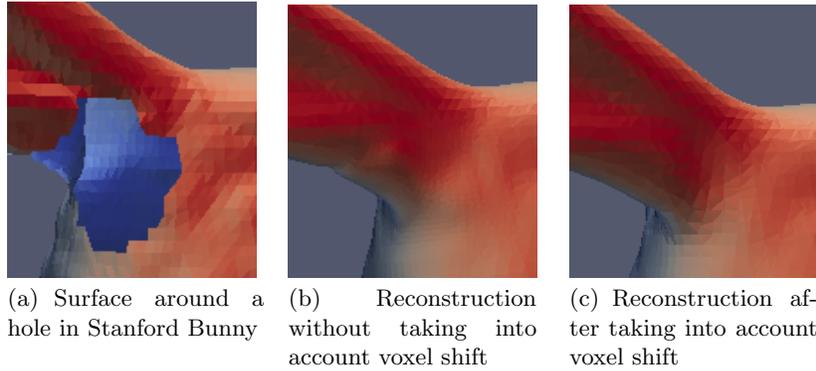


Fig. 6. Shift in the position of reconstructed surface due to voxelization

should provide us with the desired zero-set surface. In our implementation we have used Marching Cubes algorithm [13] due to ease of use and availability. Once the surfaces are extracted, All those triangles are removed which intersect with those voxels which embed input surface mesh. This leaves us with only those triangles which lie within the hole regions in the original surface mesh as a patch. Parts of input geometry is embedded in a number of solution columns in voxels, while surface extraction contouring is done by evaluating values on the voxel corners. As a result, it was observed that conversion of surface mesh to a volume mesh and subsequent contouring causes a noticeable shift of about half a voxel in the position of the extracted surfaces in comparison with the original input surface. If left uncorrected, the combination of original input surface and extracted surface patch would produce a noticeable bump in the output reconstructed surface as shown in figures 6(a), 6(b) and 6(c). The shift is corrected by shifting the points on the extracted surface by half voxel in the direction away from the positive normal. Furthermore, normals are calculated and generated at every points in the extracted patches as well as input surface and mesh fragments to create a pointset. The pointset loosely represents a surface mesh with well sampled points everywhere including hole regions and is used as an input for surface reconstructions.

4 Surface Reconstruction and Results

Surface reconstruction and surface fitting from point samples is a well studied problem in computer graphics and has applications in a number of areas including reverse engineering. Reconstruction itself is a very challenging area due to uneven sampling of points, noisy data, scan misregistration among other problems [29]. There are a number of schemes for surface reconstruction based on implicit form among other techniques. These implicit surface fitting methods are either global or local in nature. Global fitting methods commonly

define the implicit function as the sum of radial basis functions (RBFs) centered at the points. Local fitting methods consider subsets of nearby points at a time. These methods are well studied and have been compared in a number of papers including but not limited to [23, 24, 25, 26, 27, 28, 29, 30] among others.

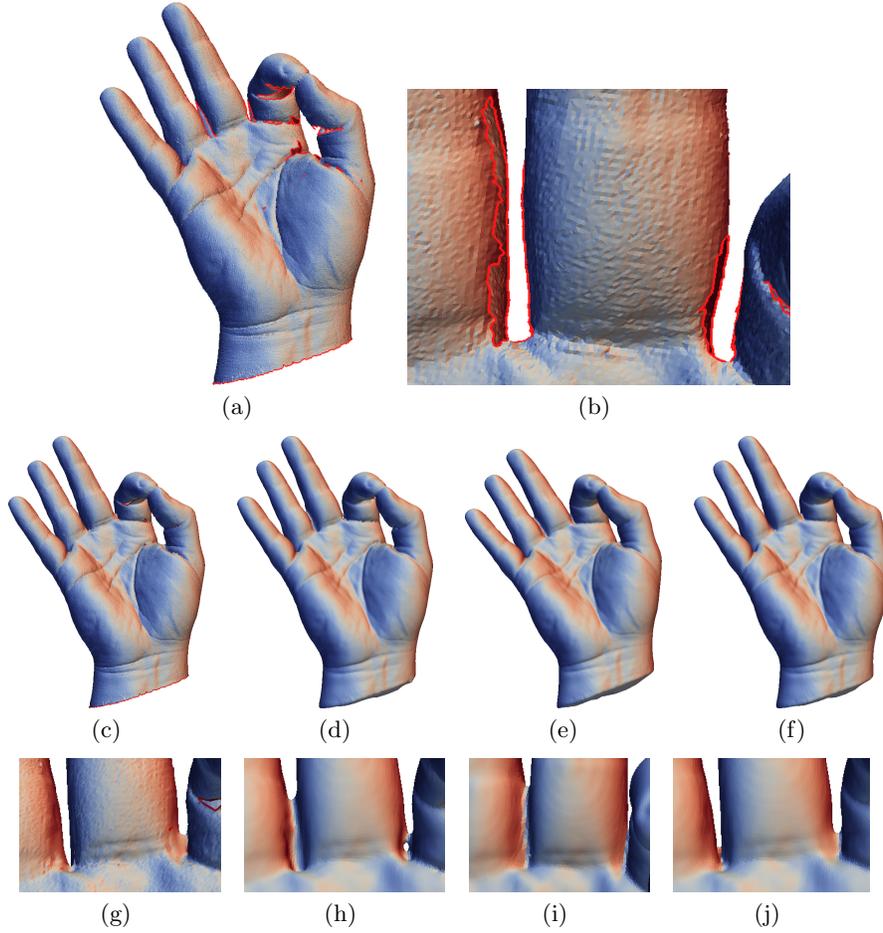


Fig. 7. Mesh repair on Laurent's Hand [36] in different configurations: (a) Original model of Laurent's hand, (b) closeup of model around fingers, (c) repair based on surface approach, (d) repair based on volume approach, (e) repair based on Poisson's reconstruction, (f) repair based on hybrid approach, (g) closeup after repair based on surface approach, (h) closeup after repair based on volume approach, (i) closeup after repair based on Poisson's reconstruction and (j) closeup after repair based on hybrid approach

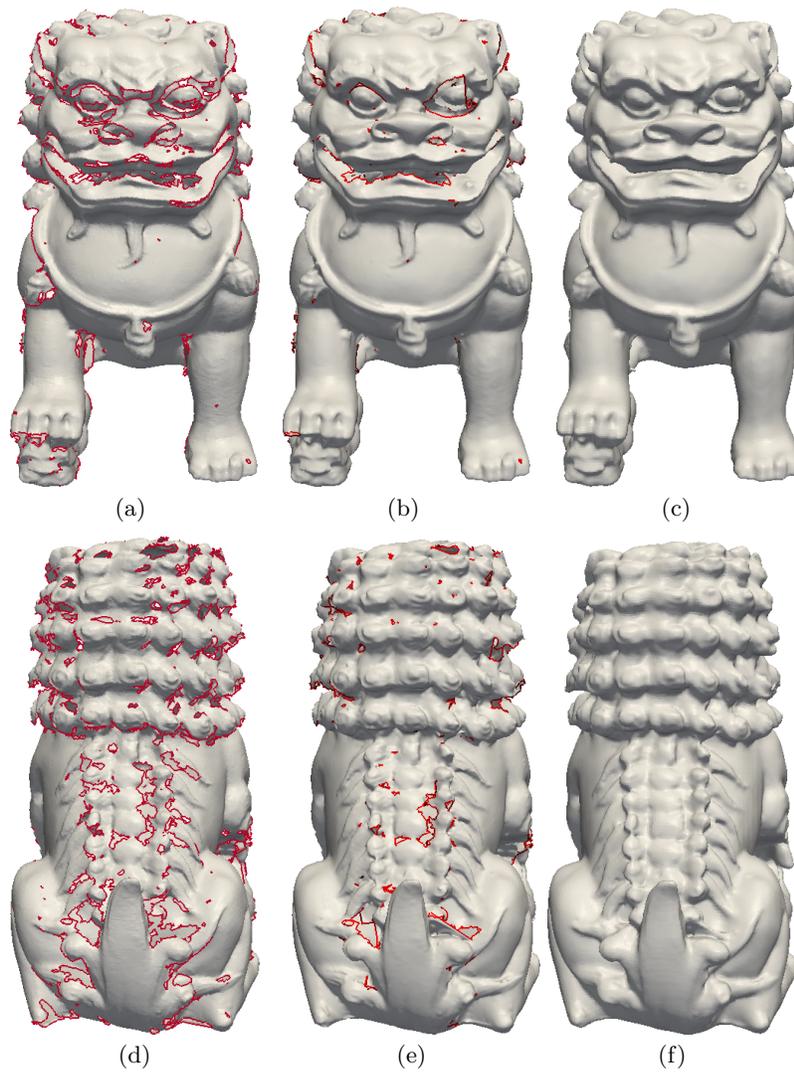


Fig. 8. Chinese Lion (a), (d): Front and back of original model , (b), (e): Front and back of model after mesh repair based on surface approach and (c), (f): Front and back of model after mesh repair based on hybrid approach

In this paper, we do not try to create any new method for surface reconstruction, but we would be using an existing well established method for it. We chose Poisson Surface Reconstruction as described in [29] as its open source implementation is widely available [31]. Although Poisson surface reconstruction provides a watertight surface, however its accuracy is dependent on the sampling of input points. We, in this method, try to provide a well sampled pointset with correctly oriented normals as an input to the Poisson reconstruction so that we could get an output surface which is not only smooth, well behaved but also is feature sensitive due to our hybrid approach. We use [31] implementation of Poisson reconstruction at octree refinement level 10 in our examples. Figures 7(a) - 7(j) show a visual comparison of results obtained on Laurent Hand model. Figure 7(a) shows the original model with discontinuities. Figure 7(b) shows a closeup image around the middle two fingers. Figure 7(c) shows the mesh repair using only surface based approach. We can clearly see in figure 7(g) that although the repair approximates the surface well but it fails to repair all the discontinuities. Figures 7(d), 7(e) and 7(f) shows the result after volume based repair followed by Poisson Surface reconstruction, only Poisson reconstruction and hybrid approach which uses both surface and volume based techniques, respectively. Looking at the closeup images it can be noticed that fingers in volume based approach 7(h) and only Poisson based approach 7(i) are fused together which is clearly an undesirable outcome. Figure 7(j) shows the closeup image from the hybrid approach where middle two fingers are not fused together and hence is a better result.

Figures 8(d), 8(e) and 8(f) show the front side of Chinese Lion [37] as original model, model after repair based on surface based technique and repair based on hybrid approach, respectively. Similarly, figures 8(a), 8(b) and 8(c) show the backside of Chinese Lion [37] as original model, model after repair based on surface based technique and repair based on hybrid approach, respectively. Lines in red show discontinuities present in the model.

The input models being used and presented in this section are of high resolution and have high level of details and geometric complexities. The input Chinese Lion model had a number of isles or small surface fragments along with a large number of holes. The results presented in this section demonstrate that merely using either surface or volume based techniques may not be sufficient as only one of those in isolation may not be able to repair the model completely or they may generate a result which may not be reliable or satisfactory. Combining surface and volume approach based techniques as a hybrid approach helps us in preserving the details as well as helps us in providing better results in that process. This was especially demonstrated using Laurent Hand model in figures 7(a) - 7(j).

5 Conclusion

Surface based methods explicitly try to identify surface artifacts prior to operating on them and require that input geometry meets some mesh quality conditions. This is sometimes not feasible causing the mesh repair techniques to fail. However the surface based methods have some compelling advantages over the volume based methods, which emphasize that we should try to employ surface based techniques wherever possible to get better quality results compared to when we are only using volume based methods for geometry repair.

Volume based approaches like the ones described in this research can be used to repair the models with artifacts that surface based models otherwise cannot robustly handle. They however also pose some potential problems. The conversion to and from a volume leads to resampling of the model. It often introduces aliasing artifacts, loss of model features and destroys any structure that might have been present in the connectivity of the input model. The focus of this work is to address the need to develop a method to obtain a watertight geometry from a geometric model that could have the presence of holes of complex topology. Despite all their shortcomings, volumetric algorithms can solve some problems in geometry repair robustly which can not be handled by surface based approaches alone. As a result we have presented a hybrid method for repairing a discrete geometry by combining surface and volume based methods, which in many cases provides superior results when compared with either Surface or volume based techniques in isolation. We have presented a number of repaired models in support of the results presented in this work.

6 Acknowledgement

We would like to express our gratitude towards Stanford 3D Scanning Repository [35] and AIM@SHAPE Shape Repository for putting test models in public domain, some of which were used in this paper to present and validate our results. We would also like to express our gratitude towards Laurent Saboret *et al.* [36], [37] of INRIA for making “Laurent Hand” and “Chinese Lion” models publicly available.

References

1. Botsch M, Pauly M, Kobbelt L, Alliez P, Lévy B, Bischoff S and Rössl C (2007) Geometric modeling based on polygonal meshes. ACM SIGGRAPH 2007 Courses - International Conference on Computer Graphics and Interactive Techniques
2. Turk G. and Levoy M (1994) Zippered polygon meshes from range images. In Proceedings of ACM SIGGRAPH '94, 311–318

3. Barequet G, Sharir M (1995) Filling gaps in the boundary of a polyhedron. *Computer Aided Geometric Design*, 12:207–229
4. Barequet G, and Kumar S (1997) Repairing CAD Models. *Proceedings of IEEE Visualization 1997*, 363-370
5. Guézic A, Taubin G, Lazarus F, Horn B (2001) Cutting and Stitching: Converting Computer sets of Polygons to Manifold Surfaces. *IEEE Transactions on Visualization and Graphics*, 7(2):136–151
6. Borodin P, Novotni M, Klein R (2002) Progressive Gap Closing for mesh repairing. In J. Vince and R. Earnshaw, editors, *Advances in Modelling, Animation and Rendering*, Springer Verlag, 201–213
7. Liepa P (2003) Filling Holes in Meshes. *Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry processing*, Eurographics Association, 200–205
8. Jun Y (2005) A Piecewise Hole Filling Algorithm in Reverse Engineering. *Computer-Aided Design*, 37:263–270
9. Branch J, Prieto F, Boulanger P (2006) A Hole-Filling Algorithm for Triangular Meshes using Local Radial Basis Function. *Proceedings of the 15th International Meshing Roundtable*, Springer, 411–431
10. Pernot JP, Moraru G, Vernon P (2006) Filling holes in meshes using a mechanical model to simulate the curvature variation minimization. *Computer and Graphics*, 30(6):892–902.
11. Kumar A, Shih MA, Ito Y, Ross HD, Soni KB (2007) A Hole-Filling Algorithm Using Non-Uniform Rational B-Splines. *Proceedings of 16th International Meshing Roundtable*, Springer Berlin Heidelberg, ISBN: 978-3-540-75102-1, 169–182
12. Kumar A, Ito Y, Yu T, Ross HD, Shih MA (2011) A novel hole patching algorithm for discrete geometry using non-uniform rational B-spline. *International Journal for Numerical Methods in Engineering*, Published Online <http://onlinelibrary.wiley.com/doi/10.1002/nme.3157/abstract>
13. Lorensen EW, Cline EH (1987) Marching Cubes: A high resolution 3D surface construction algorithm. *ACM Computer Graphics*, 21(4)
14. Curless B, Levoy M (1996) A volumetric method for building complex models from range images. *Computer Graphics*, 30:303–312
15. Murali MT and Funkhouser AT (1997) Consistent solid and boundary representations from arbitrary polygonal data. In *Proceedings of Symposium on Interactive 3D Graphics*, 155–162
16. Davis J, Marschner SR, Garr M, Levoy M (2002) Filling holes in complex surfaces using volumetric diffusion. In *Proceedings of First International Symposium on 3D Data Processing, Visualization, Transmission*, 428–861
17. Nooruddin FS, Turk G (2003) Simplification and repair of polygonal models using volumetric techniques. *IEEE Transactions on Visualization and Computer Graphics*, 9(2):191–205
18. Ju T (2004) Robust repair of polygonal models. In *Proceedings of ACM SIGGRAPH '04*. *ACM Transactions on Graphics*, 23:888–895
19. Bischoff S, Pavic D, and Kobbelt L (2005) Automatic restoration of polygon models. *Transactions on Graphics*, 24(4):1332–1352
20. Podolak J, Rusinkiewicz S (2005) Atomic volumes for mesh completion. In *Symposium on Geometry Processing*

21. Hoppe H, DeRose T, DuChamp T, HalStead M, Jin H, McDonald J, Schweitzer J, Stuetzle W (1994) Piecewise Smooth Surface Reconstruction. In Proceedings of ACM SIGGRAPH '94, New York, NY, 295–302
22. Amenta N, and Bern M (1999) Surface Reconstruction by Voronoi Filtering. *Discrete & Computational Geometry*, 22(4):481–504
23. Amenta N, Hoi SC, Kolluri R (2001) The power crust, unions of balls, and the medial axis transform. *Computational Geometry: Theory and Applications*, 19:127–153
24. Carr CJ, Beatson KR, Cherrie BJ, Mitchell1 JT, Evans RT, Fright RW, McCallum CB (2001) Reconstruction and representation of 3D objects with Radial Basis functions, In Proceedings of ACM SIGGRAPH '01, 67–76
25. Dey KT, and Goswami S (2003) Tight Cocone : A Water-tight Surface Reconstructor. *Journal of Computing and Information Science in Engineering*, 3(4):302–307.
26. Dey KT, and Goswami S (2004) Provable surface reconstruction from noisy samples. *Computational Geometry*, 35(1-2):124–141
27. Shen C, O'Brien FJ, Shewchuk RJ (2004) Interpolating and approximating implicit surfaces from polygon soup. In Proceedings of ACM SIGGRAPH '04, 896–904
28. Casciola G, Lazzaro D, Montefusco BL, Morigi S (2005) Fast surface reconstruction and hole filling using positive definite radial basis functions. *Journal of Numerical Algorithms*, 39:289–305
29. Kazhdan M, Bolitho M, Hoppe H (2006) Poisson Surface Reconstruction. In Proceedings of the fourth Eurographics symposium on Geometry processing (SGP '06), Switzerland, 61–70
30. Mullen P, Goes DF, Desbrun M, Cohen-Steiner D, Alliez P (2010) Signing the Unsigned: Robust Surface Reconstruction from Raw Pointsets. *Eurographics Symposium on Geometry Processing 2010*, 29(5):1733-1741
31. Doria D, and Gelas A (2010) Poisson Surface Reconstruction for VTK. *The VTK Journal*. <http://www.insight-journal.org/download/viewpdf/718/2/download>
32. Akenine-Möller T (2005) Fast 3D triangle-box overlap testing. In *ACM SIGGRAPH 2005 Courses*, Los Angeles, California. J. Fujii, Ed. SIGGRAPH '05. ACM, New York
33. Akenine-Möller T (2010) AABB-triangle overlap test code. Source code is located at <http://jgt.akpeters.com/papers/AkenineMoller01/tribox.html>
34. (2011) See Diffusion Equation on Wikipedia. http://en.wikipedia.org/wiki/Diffusion_equation
35. (2011) Stanford Bunny at The Stanford 3D Scanning Repository. <http://graphics.stanford.edu/data/3Dscanrep/>
36. Saboret L, Attene M, Alliez P (2011) Laurent Hand at AIM@SHAPE Shape Repository. <http://shapes.aim-at-shape.net/viewgroup.php?id=785>
37. Saboret L, Attene M, Alliez P (2011) Chinese Lion at AIM@SHAPE Shape Repository. <http://shapes.aim-at-shape.net/viewgroup.php?id=783>