A Robust Reconstruction Algorithm of Displaced Butterfly Subdivision Surfaces from Unorganized Points

Byeong-Seon Jeong, Sun-Jeong Kim and Chang-Hun Kim

Dept. of Computer Science and Engineering, Korea University
1, 5-ka, Anam-dong, Sungbuk-ku, Seoul 136-701, Korea
{bsjeong, sunjeongkim, chkim}@korea.ac.kr

Fig. 1. Displaced Butterfly Subdivision Surfaces (DBSS): (a) Point cloud, (b) DBSS level 1, (c) DBSS level 2 and (d) DBSS level 3

Abstract. This paper presents a more robust reconstruction algorithm to solve the genus restriction of displaced subdivision surface (DSS) from unorganized points. DSS is a useful mesh representation to guarantee the memory efficiency by storing a vertex position as one scalar displacement value, which is measured from the original mesh to its parametric domain. However, reconstructing DSS from unorganized points has some defects such as the incorrect approximation of concave region and the limited application of genus-0. Based on volumetric approach, our new cell carving method can easily and quickly obtain the shape of point clouds and preserve its genus. In addition, using interpolatory subdivision scheme, our displaced butterfly subdivision surface is also effective multiresolution representation, because it samples exclusively new odd
vertices at each level, compared with previous works to resample all vertices of every level. We demonstrate that displaced butterfly subdivision surface is an effective multiresolution representation that overcome the topological restriction and preserve the detailed features nicely.

1 Introduction

Recently fast and accurate range scanners enable to model complex objects automatically. These scanners provide point data on the surface of an object for applications such as medicine, reverse engineering, and digital filmmaking. The computation of surfaces out of these point data is referred to as reconstruction. Surface reconstruction from the point data has been an actively researched area in computer graphics for several decades. And since subdivision mesh is very useful for the level of detail representation, editing and animation, various approaches have been proposed to remesh the irregular surfaces after the reconstruction. Especially [9] introduced the Displaced Subdivision Surfaces (DSS) as a new mesh representation to store one scalar displacement value, which is measured between the original mesh and its parametric domain, instead of three coordinates of a vertex position. It means the DSS dramatically reduces the amount of the required memory to the 1/3 compared to the existing mesh representations. However this technique has the limitation of starting from a detailed initial mesh. In [7] a direct reconstruction algorithm of the DSS from three dimensional point clouds is proposed. The shrink-wrapping algorithm of bounding box is used for the construction of an initial mesh. However it only works on the genus-0 models and fails to reconstruct the concave shape (see Fig. 2). Also, the DSS is not a truly multiresolution representation. It is just a single level representation of the subdivision mesh. To generate the multiresolution representation with the DSS, the displaced values of all vertices are resampled at every level. Since the representation of each level has the displaced values of all vertices, the whole size of the multiresolution representations is not effective. In this paper we propose a more robust algorithm which solves the problems mentioned above and also dramatically reduces the amount of required memory for the multiresolution representation. We use the cell carving approach based on the volume grid instead of the bounding box proposed in [7]. Firstly we create a volume grid of the uniform size specified by a user containing the input point cloud. From the outermost cells, we check whether a cell contains some points or not. If points are not found inside the cell, we remove it and check the next cell in the same manner. Through this procedure, we can easily and quickly obtain the set of cells which approximates the shape of the point clouds and preserves their genus. We convert this set of cells to the initial mesh using the triangulation followed by the shrink-wrapping. The resulting initial mesh guarantees the topology and the shape of the input model to be preserved. To achieve the memory effective multiresolution representation, we use the Modified Butterfly scheme in [11], one of the interpolatory subdivision schemes, when generating the parametric domain and sampling the displaced values. The interpolatory scheme is more efficient
than the approximating scheme in generating the multiresolution DSS, because
the displace values can be sampled only at the odd vertices per subdivision level.

![Teapot models](image)

**Fig. 2.** The initial mesh (Teapot model) reconstructed using the method in [7] and our
method: (a) *The initial mesh reconstructed using the method in [7]* and (b) *The one
reconstructed using our method.*

**Contributions**

- Our reconstruction algorithm is so robust that it is genus-free and feature
  preserving. Using the volumetric approach, i.e. Cell Carving, it can recon-
  struct the initial mesh approximating the point clouds of any topologies as
  well as concave regions. The high quality initial mesh allows the accuracy of
  the reconstruction to be increased.

- Our new displaced butterfly scheme makes DSS have smaller storage than
  previous works in the multiresolution representation. When generating a
  parametric domain, [7, 9] used approximating subdivision scheme, but it
  resulted in shrinkage effect and resampling all vertices of every subdivision
  level. However, Displaced Butterfly Subdivision Surface doesn’t shrink from
  the control mesh and samples only new odd vertices at each subdivision
  level.

2 Previous Work

The general process for the multiresolution mesh representation starts with re-
constructing the initial mesh from the point cloud, and then gets through the
steps of remeshing for the mesh to gain the subdivision connectivity, which makes
it possible to achieve the multiresolution representation. Now we will take a look
at the related previous work.
2.1 Reconstruction

Several criteria are used to evaluate reconstruction algorithms from the point cloud scanned from three dimensional objects. Those criteria include the quality of the reconstructed model, the speed of the reconstruction process, and the robustness of the algorithm. In [5, 6], the displacement function is defined by calculating the tangent plane of the points and used for finding the voxel information which is applied to the Marching cubes algorithm that reconstructs the mesh. The quality of the resulting mesh is improved later after the mesh optimization. The high complexity and low speed are the main drawbacks of this algorithm. In [2], to reconstruct the mesh from the point cloud, a three dimensional octree containing the point cloud is created and Delaunay triangulation is applied to each cell. The speed of Delaunay triangulation algorithm is sharply dropped when the number of points begins to increase. They applied the Delaunay triangulation algorithm to each cell separately to prevent the loss of speed and to deal with the large mesh data. But the Delaunay triangulation is basically too slow. In [1], firstly the approximating function of the point cloud is constructed using the Radial Basis Function (RBF). Then, Marching cubes algorithm is used to show the visual appearance of the model. This means that the reconstruction and the rendering processes are considered as separated parts. Though the quality of the resulting mesh is considerably high, this algorithm is also too slow to get the triangle mesh. In this paper, we want to reconstruct the coarse control mesh easily and quickly which is used to generate the parametric domain. So, we need a method to reconstruct fast and correctly the initial mesh without loss of any geometrical and topological features. For this, we used the cell carving and shrink-wrapping algorithms based on the volume grid. Intuitively the cell carving algorithm is the same that an artist carves the initial wood to make a sculpture.

2.2 Subdivision

A subdivision mesh consists of a control mesh and the subdivision mask or rule. A subdivision rule is applied to a control mesh at every subdivision level and makes the connectivity between meshes before and after subdivision. Since the subdivision mesh has such hierarchical connectivity, there are many applications like the level of detail representation, multiresolution editing, progressive transmission, etc. However we need remeshing techniques that convert irregular meshes into regular meshes which are subdivision meshes, because ordinary irregular meshes do not have any subdivision connectivity. In [8], the Shrink-wrapping approach is proposed to make an arbitrary mesh gain the subdivision connectivity. By using the bounding sphere surrounding the input arbitrary mesh, the simple control mesh is constructed. And it is subdivided to get the hierarchical connectivity. At last a subdivision mesh is shrink-wrapped over the arbitrary mesh to approximate its original shape. This algorithm restricts the input model to have only genus-0 topology. In [3], they partitioned the input model into surface patches which are homeomorphic to the disk, and each surface is parameterized.
onto a corresponding triangle of the simplified model. The simplified model is subdivided and lifted back onto the original surface according to the information of parameterization. They do not restrict the topology of an input model, but the partitioning preprocess is too costly high to get the subdivision mesh. In this paper, the shrink-wrapped mesh is genus-free and is not partitioned into the disk-like patches. We just simplify the initial mesh reconstructed from the point cloud and subdivide it to get the subdivision connectivity.

2.3 Displaced Subdivision Surfaces

The DSS was proposed in [9] on which the concept of this paper is based. The DSS uses only one scalar distance value instead of the three scalar coordinates to represent vertex position of the mesh. That enables us to reduce the memory requirement to 1/3 and to store, edit and transmit efficiently the large mesh data of high quality. The DSS reconstruction algorithm requires the domain surface constructed by subdividing the simplified control mesh, measures the distances between the vertices on the domain and the arbitrary input mesh, and assigns it to each vertex. However, if we want to generate the multiresolution representation of the DSS, we have to subdivide the control mesh from the beginning coarsest level to the target subdivision level to get one resolution of the mesh and to sample the displacement values of all the vertices. To get several resolution meshes, we must repeat the steps mentioned. This makes us lose the memory efficiency of the DSS and its reconstruction algorithm slow. To get more efficiency in generating the multiresolution DSS, we use the Modified Butterfly subdivision [11], which is one of the interpolatory schemes. First, the control mesh is subdivided just once using the Modified Butterfly scheme followed by sampling the distances between the odd vertices on the parametric domain and the point cloud, and the odd vertices moved by the sampled distances along the vertex normal direction. To generate the next resolution, in the same manner we subdivide the resulting mesh just one time followed by the distance sampling and the vertex moving processes. It is important that the sampling and moving processes are performed only on the newly introduced vertices at every subdivision level. That enables us to make the true multiresolution representation with far less memory requirement compared to the existing methods.

3 Robust Reconstruction Algorithm

Our algorithm consists of two steps. The first step is generating the control mesh by reconstructing the initial mesh from the point cloud followed by simplifying and projecting to the point cloud. The next is sampling the displacement values between the subdivided control mesh, i.e. the parametric domain and the original input point cloud.
Fig. 3. The volume grid and the six neighbors of each cell in the grid: (a) Volume grid and (b) Six neighbor cells

3.1 Control Mesh Generation

First of all, some definitions have to be clarified. Let the volume grid surrounding the point cloud be the set $G$ which consists of the whole cells of the grid. The size of the volume grid is specified by the user like $4 \times 4 \times 4$ (see Fig. 3(a)). According to the complexity of the shape of the point cloud, the size of the volume grid can be various. Each cell in the set $G$ has its six neighbors which are adjacent to its six faces (see Fig. 3(b)). By carving the volume grid, we can extract the shape of the point cloud which the remaining cells have. We call the carving procedure Cell Carving algorithm. It tests whether the cells have the points or not. The set $C$ consists of these candidate cells. At the beginning of the algorithm, the set $C$ contains the outermost cells of the volume grid. After the Cell Carving, we can get the cells approximating the surface of the point cloud. The surface set $S$ consists of these cells. And the other sets used in the Cell Carving algorithm are just temporary. Now the detail procedure of the Cell Carving algorithm is as follows (also see Fig. 4).

Step 1 Insert the outermost cells into the set $C$.
Step 2 Repeat the following substeps on each cell in the set $C$ until the set $C$ becomes empty.
   Step 2.1 If a cell contains some points, then insert it into the set $T$ and remove it from the set $C$.
   Step 2.2 If a cell does not contain any points, then perform the following substeps on its neighbor cells and remove it from the set $C$ and the set $G$.
   Step 2.2.1 If a neighbor cell contains some points, then insert it into set $T$, and remove it from set $C$ if it is in the set $C$. 

Step 2.2.2 If a neighbor cell does not contain any points and is not in the set \( C \) either, then insert it into the set \( N \).

Step 3 If there are some elements in the set \( N \), then move all the elements in the set \( N \) to the set \( C \) and return to the step 2.

Step 4 For each cell in the set \( T \), perform the following substeps.

Step 4.1 For each neighbor cell of this cell, perform the following substeps.

Step 4.1.1 If the neighbor cell contains some points and is not in the set \( T \) or \( S \), then insert it into the set \( W \).

Step 5 If there are some cells in the set \( W \), then move all the elements in the set \( T \) to the set \( S \), move all the elements in the set \( W \) to the set \( T \), and return to the step 4. Otherwise terminate the algorithm.

![Fig. 4. The Cell Carving algorithm: (a) the initial volume grid, (b) the test with the first candidate cell and its neighbors, (c) the first cell is removed because it is the empty cell, (d) the test with the second candidate cell and its neighbors, (e) the second cell is removed because it is empty too, (f) the test with the fifth candidate cell and its neighbors, (g) the fifth cell is not removed because it is the non-empty cell, (h) the test with the last candidate cell and its neighbors, and (i) the final result of the Cell Carving algorithm.](image)

After carving the volume grid, the cells in the set \( S \) represent the surface of the point cloud. Then we triangulate the quad faces on the cells in the set \( S \) which aren’t adjacent to any other cells to make the triangle mesh. We call those quad faces the air faces and call the air vertices the vertices of the cell such that are not adjacent to any other cells. There are four types of triangulation:

Type 1 If there are five air faces in a cell, then we collapse the four air vertices into one vertex. The collapsed vertex will be located at the averaged position
of the four vertices. As a result, there are five vertices, four from the non-air faces and one from the air faces (see Fig. 5(a)).

Type 2 If there are four air faces which are not arranged in the shape of the ribbon in a cell, then we collapse the two air vertices into one vertex. The collapsed vertex will be located at the averaged position of the two vertices. As a result, there are seven vertices, six from the non-air faces and one from the air faces (see Fig. 5(b)).

Type 3 If there are three air faces in a cell, which means that there is only one air vertex, then we triangulate three air faces (see Fig. 5(c)).

Type 4 Otherwise, we perform the general triangulation (see Fig. 5(d)).

![Diagram](image)

**Fig. 5.** The types of the triangulation: (a) Four air vertices, (b) Two air vertices, (c) One air vertex, and (d) General type.

This process guarantees the good aspect ratio of the triangles in the initial mesh. Consequently, we can minimize the distortion which could be raised at the displacement sampling stage. Now we apply the shrink-wrapping algorithm to the triangulated mesh over the point cloud. The shrink-wrapping process is the same as [7]. The projection and smoothing operators are repeatedly applied in shrinking the mesh over the point cloud. The projection operator moves each vertex in the direction of the closest point to it. The smoothing operator relaxes the vertices to be distributed equally in order to minimize the distortion produced by the projection. Now the shrink-wrapped mesh is approximating the shape of the point cloud and eventually it becomes the initial mesh. The next
step is the simplification of the initial mesh up to the user specified level to get the simpler one. If the initial mesh is sufficiently coarse, then the simplification is not necessary. In [7], to simplify the initial mesh they used a point based QEM method that is more time consuming than the original QEM in [4]. This is because the quality of the initial mesh was not good enough to guarantee them the adequate quality of the simplified one. However the accuracy of the initial mesh reconstructed by our algorithm is higher than the previous one and even enough to assure the quality of the simplified mesh so that we can use the original QEM in this paper. Finally, we need to fit the simplified mesh to the point cloud to generate the well-shaped parametric domain. To generate the parametric domain from the control mesh, Loop subdivision scheme was used in both [9] and [7]. Loop scheme in [10] which is one of approximating subdivision scheme causes the shrinkage effect, so the global energy minimization and local subdivision surface fitting have to be performed on the simplified mesh before subdividing it for the better approximation of the resulting parametric domain to the input model. For this, in [7], they use the local subdivision surface fitting algorithm which is faster than the global fitting in [9]. In [7], they computed the limit positions of the vertices of the control mesh which the subdivision scheme allows, and found the closest points to those positions. Then, they adjust the position of the vertices of the control mesh so that the limit position may be close to those points. But we need not to fit the surfaces like that. We only need to project the vertices of the control mesh to the point cloud because we use the interpolatory subdivision scheme not to have the shrinkage effect. After projecting the simplified mesh to the point cloud we have the control mesh (see Fig. 9).

3.2 Parametric Domain Generation and Displacement Sampling

The parametric domain is a subdivision mesh constructed from the control mesh by subdivision. In [7], they use Loop subdivision to generate the parametric domain. So they can only get the single resolution of the DSS because Loop scheme repositions all the vertices whenever it subdivides the control mesh. But, we use the interpolatory subdivision scheme which does not reposition the even vertices. Therefore we can construct the multiresolution mesh through a single process using the interpolatory subdivision scheme, i.e. Modified Butterfly scheme in [11]. Here are the details (also see Fig. 6):

Step 1 Subdivide the control mesh using the Modified Butterfly scheme.
Step 2 Measure the distances between the newly generated vertices on the subdivided mesh and the point cloud in the vertex normal direction. To measure the distances we use the triangle intersection test as [7] but the different method to gather the sampling points (see Fig. 7).
Step 3 Move each odd vertex in the vertex normal direction according to the measured distance respectively.
Step 4 If the subdivision level is fine enough, then terminate the procedure. Otherwise, subdivide the resulting mesh once again and return to the step 2.
To gather the sampling points, we must compute two distances. At first, we calculate the vertical distance of the point as Fig. 7(c) and the horizontal distance of the point as Fig. 7(d). Therefore the point sampling area is the cylinder whose height axis is along the vertex normal. If these distances of the point are smaller than the thresholds, then they becomes sampling points. Then we can sample the displacement between the candidate triangles which consist of three sampling points and the odd vertices using the triangle intersection method in [7]. After sampling the displacements, the odd vertices are moved up to the amount of the displacements along the normal of each vertex. Compared with our approach, the method in [7] sampled the displaced values of the whole vertices after subdividing the control mesh up to the user specified level (see Fig. 8). According to their method, to construct multi-level representation, resampling of all the vertices is necessary at every subdivision level. Therefore its multiresolution representation has the redundancy in the number of the displaced values. However our Displaced Butterfly Subdivision Surfaces (DBSS) can get more subdivision levels in a given memory space than their DSS, because it consists of the control mesh and the displaced values of the newly introduced vertices only at every subdivision level.
**Fig. 7.** The cylindrical distances for sampling points: (a) *Cylindrical point sampling*, (b) *Cylindrical thresholds*, (c) *Vertical distance*, and (d) *Horizontal distance*.
4 Results

We implemented our algorithm in Pentium IV 1.4 GHz CPU, 512MB memory and GeForce2 MX400 Graphic card. We can generate the control mesh fast and easily using the proposed Cell Carving algorithm. The bounding box algorithm used in [7] fails to reconstruct the non genus-0 models (see Fig. 2). It also fails to preserve the details of the complicated and highly curved models. With the proposed algorithm, we can reconstruct both the genus and the detail features of the model. For the multiresolution representation, the existing method requires to sample the displacement of all the vertices at every subdivision level. The proposed algorithm needs the displacement sampling only on the newly introduced vertices at each subdivision level. So, our DBSS can represent any levels of the multiresolution mesh (see Fig. 1, 9, and 10, and Table 1). We can easily infer that our approach to build the multiresolution representation is about more efficient than the previous one in the memory by 25%, because we store the scalar values of the odd vertices at each subdivision level (see Table 1).

5 Conclusion and Future Work

In this paper we proposed a robust algorithm that directly reconstructs a multiresolution Displaced Butterfly Subdivision Surfaces from three dimensional point clouds. Our Cell Carving algorithm is very simple but reconstructs robustly an initial mesh, because it preserves the genus and features of the input points. Since the displaced values are sampled at the odd vertices per subdivision level, our algorithm is effective in representing the multiresolution mesh, compared to the previous DSS, which must resample the displaced values of all
Fig. 9. The whole process of our algorithm (Torus model): (a) Point cloud, (b) Volume grid, (c) After cell carving, (d) Triangulated cells, (e) Initial mesh, (f) Control mesh, (g) DBSS level 1, (h) DBSS level 2, and (i) DBSS level 3
Table 1. The number of times of the displacement sampling in ours and the previous methods

<table>
<thead>
<tr>
<th>Schemes</th>
<th>Models</th>
<th>Level 1</th>
<th>Level 2</th>
<th>Level 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our</td>
<td>Torus</td>
<td>150</td>
<td>600</td>
<td>2,400</td>
</tr>
<tr>
<td>method</td>
<td>Rabbit (control mesh #v = 50)</td>
<td>315</td>
<td>1,260</td>
<td>5,040</td>
</tr>
<tr>
<td>Previous</td>
<td>Torus</td>
<td>200</td>
<td>800</td>
<td>3,200</td>
</tr>
<tr>
<td>method</td>
<td>Rabbit (control mesh #v = 107)</td>
<td>422</td>
<td>1,682</td>
<td>6,722</td>
</tr>
</tbody>
</table>

Fig. 10. The resulting Displaced Butterfly Subdivision Surfaces of the Mechanical part model: (a) Point cloud, (b) Control mesh, (c) DBSS level 1, (d) DBSS level 2, (e) DBSS level 3
the vertices of every subdivision level. Therefore our algorithm has the efficiency in term of time and space on the multiresolution representation. Our Displaced Butterfly Subdivision Surfaces can be widely used in computer graphics applications. However, since the CAD models with the sharp edges can hardly be represented using the DSS with the relatively small number of triangles, a new sampling scheme must be developed.

References

2. T. K. Dey, J. Giesen, and J. Hudson.: Delaunay Based Shape Reconstruction from Large Data. IEEE Symposium on Parallel and Large Data Visualization and Graphics (PVG 2001), pp.19-27, October 2001
10. C. Loop.: Smooth Subdivision Surfaces Based on Triangles. Masters thesis, Department of Mathematics, University of Utah, August 1997