Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/080447

Remeshing Visual Hull Approximation by Displaced Butterfly Subdivision Surfaces

Jung Lee[1], *Chang-Hun Kim*[1] *and Sun-Jeong Kim*[2,*]

[1] Dept. of Information and Communication, Korea University, Seoul, Korea
[2] Dept. of Ubiquitous Computing, Hallym University, ChoonCheon, Korea

Received: 16 Aug. 2013, Revised: 18 Nov. 2013, Accepted: 19 Nov. 2013
Published online: 1 Jul. 2014

Abstract: This paper proposes a new procedure for generating a displaced butterfly subdivision surface from multiple images. First, point geometry of the target object is recovered by combining LDC(Layered Depth Cube) surfel sampling scheme with the concept of visual hull based on the input images. Then the subdivision surface is generated approximating to the recovered point cloud. We use a variant displaced subdivision scheme, where scalar displacement, in the direction of a local normal, is computed via the MLS(Moving Least Squares) approximation. The resulting subdivision surface is a mesh with subdivision connectivity providing a high-quality and efficient approximation to the given images. And it is able not only to represent a natural level of detail structure of the surface, but it is also to be memory-efficient by taking advantage of smoothness properties. Experimental results show the quality of our algorithm.

Keywords: Displaced butterfly subdivision surfaces, visual hull, MLS approximation

1 Introduction

Recently, complex 3D models are easily acquired from 3D scanning devices. Despite the rapid advances of 3D scanning devices, there are so many objects those geometry cannot be acquired because they are too big or disappeared in the past. In this paper, we use the concept of the visual hull to allow us to generate the target surface geometry from images. A lot of image-based modeling approaches have tried to capture the geometry of such objects. Most of them reconstruct point cloud data.

Volumetric carving algorithms [14,22] recover the voxels on the target shape from multiple images. A target shape with voxel geometry is created by iteratively eliminating invalid voxels from the initial bounding volume. Eliminating a voxel is based on its color-consistency, which enables the resulting digital solid to follow the original surface clearly. Plane sweeping technique is introduced to free the carving process from vision constraints such as occlusion.

After the point geometry recovery process, we need some tessellation methods to construct its surface geometry for its rendering, animation, and another uses. The computation of surfaces parametric surfaces or polygonal meshes - from the point data is referred to as reconstruction.

The surface reconstruction from a point cloud has been a major interest in computer graphics and geometric modeling for several decades and there are so many researches which give dense irregular polygonal surfaces as output. One of the algorithms in the first generation is the marching cubes algorithm [19] which is commonly used for extracting iso-surfaces from volume data. Although it has some defects, it has been applied to many other reconstruction algorithms. Hoppe et al. are pioneers who proposed an algorithm to reconstruct directly a subdivision surface from unorganized points [9]. Since they used an energy optimization procedure, the quality of their results is very high but it takes too long to compute. For incremental updating, a volumetric approach [6] was proposed. It calculates weighted function and signed distance function incrementally, and then assigns them to voxels. Besides, there are many approaches based on Voronoi diagrams [2], radial basis function [6], meshless parameterization [5], and so on. Some recent works are concerned with the reconstruction of a surface from under-sampled or missing data using global optimization techniques [10,24].

* Corresponding author e-mail: sunkim@hallym.ac.kr

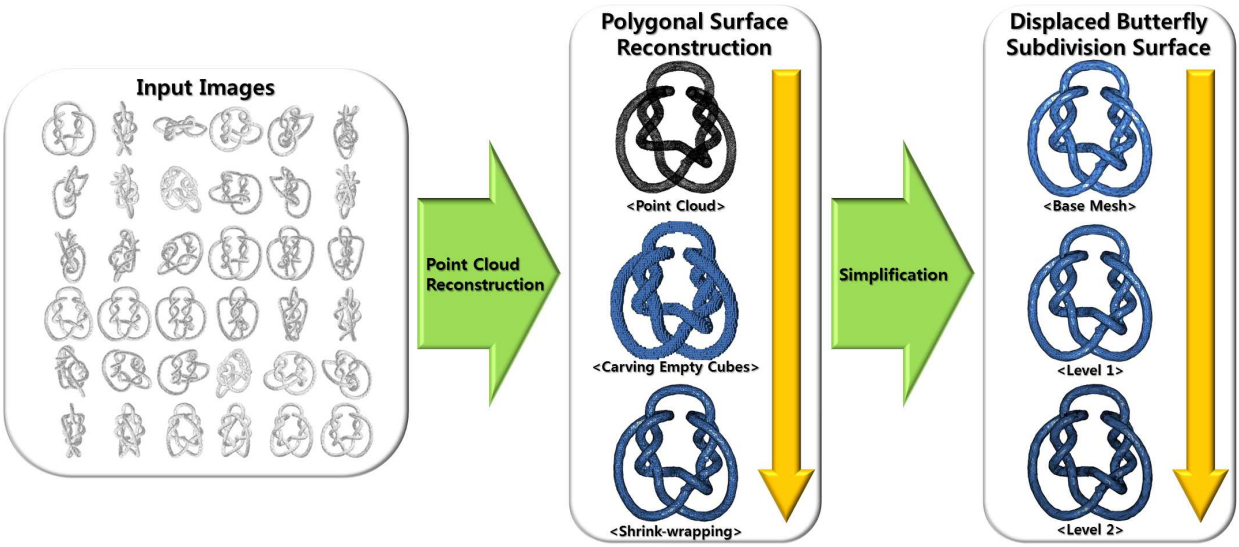


Fig. 1: A brief overview of our method

In order to create continuous surfaces from irregularly spaced point samples, splat-based approaches have been also suggested. QSplat [21] replaces points by ellipses or rectangles in image space. A hierarchical bounding sphere structure facilitates time-critical rendering. Surface splatting [20] applies EWA(Elliptical Weighted Average) texture filtering in object space and has achieved the highest image quality so far, including anti-aliasing capabilities.

Grossman and Dally [8] sampled point data sets from geometric models and then rendered them. They addressed the issues of sampling rate and gaps in the point-rendered images. Alexa et al. have proposed a new method [1] to compute and render point-set surfaces using a MLS(Moving Least Squares) approximation. A new explicit definition of point-set surfaces [3] has also been introduced to represent various point primitives such as splats and surfels.

Since subdivision is a very useful representation for level of detail description for editing and animation, many approaches have been proposed to re-mesh an irregular triangulation obtained in the reconstruction, in order to convert an arbitrary triangulation into a semi-regular one, namely, having subdivision connectivity. The approach of Eck et al. [7] partitions a given mesh by growing Voronoi tiles in order to construct a base mesh, parameterizes each tile using harmonic maps, and then subdivides the base mesh simultaneously, re-sampling the positions of odd vertices using the parameterization. In a shrink-wrapping approach for re-meshing [13], a semi-regular mesh wraps the original genus zero surface and shrinks. When shrinking, two operations are performed: attracting forces pull vertices in the direction of the given surface and relaxing forces push vertices apart in order to minimize the local distortion energy of the mesh.

The displaced subdivision surface(DSS) [16] is the new surface representation used to describe a detailed surface model as a scalar-valued displacement over a smooth domain surface. It offers a number of benefits, including geometry compression, editing, animation, scalability, and adaptive rendering. In particular, the encoding of the fine details as a scalar function makes the representation extremely compact. While the above approaches convert the irregular representation into a semi-regular one after the reconstruction process, Jeong and Kim [12] tried to reconstruct a DSS directly from a point cloud. Their approach fails to reconstruct concave objects and works only for genus zero models.

We reconstruct a subdivision surface from unorganized points on the visual hull surface of the target object. The visual hull surface is approximated based on the images of the target object. The points on the visual hull surface are sampled by applying LDC surfel sampling scheme on the input images. Then the points are used to construct subdivision surface. The subdivision surface is well-known to be a natural level-of-detail and memory efficient representation of the polygonal surface. Especially displaced butterfly subdivision surfaces (DBSS) have more efficiency of memory than general DSS that samples the displacement from the approximate domain surface subdivided by a Loop scheme. Because we use the interpolatory subdivision method like a modified butterfly scheme, we dont need to sample the displacement of all vertices but just those of odd vertices for each level. Figure 1 shows our algorithm overview. The file size of a polygonal mesh, the level two of DBSS

is about 1,129KB but we can generate it at any time using a base mesh(71KB) and its displacement(460KB).

One of our challenges is to sample the displacement between a subdivision surface and point. If point clouds have some holes, they should be filled and the displacement should be sampled. We can solve these problems by using an MLS surface which is good at approximating a point set. While DSS samples the displacement between a domain surface and an original polygonal surface, DBSS does it between a butterfly subdivision surface and an MLS approximating surface of point clouds (see Figure 2). Particularly we will improve the weight function of an MLS approximation to fill holes and deal with high-curvature regions.

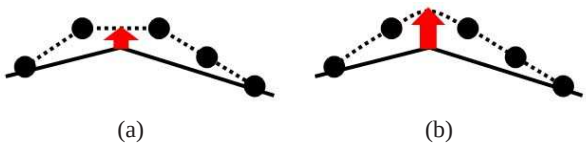


Fig. 2: Sampling scalar displacement along the direction of normal vector : (a) to a polygonal surface, and (b) to an MLS surface.

In this paper, we propose general procedure to approximate (noisy) point cloud data of models of any genus by a subdivision surface. To handle scattered points with noise, we use the recent method of moving least-squares(MLS) for surfaces [17]. The final reconstructed approximation is a displaced subdivision surface, convenient for many possible applications. The scalar displacement offsets along vertex normal onto a smooth subdivision surface to represent a detailed surface. It offers a compact representation and it has many other advantages for tasks such as editing, animation, etc.

The contribution of this work is in choosing the right procedure that takes us safely from the object-captured images to the final displaced subdivision surface. We can handle models of any genus, and achieve a high quality approximation when possible, and an efficient level of detail recovery. Note that we applied MLS approximation to the given point cloud in both the triangulation and the subdivision steps. Let us summarize the main procedures of the algorithm and their important features.

–**The point recovery step** samples points on the visual hull surface of the target object.
–**The culling step** handles the genus problem.
–**The shrink-wrap step** performs MLS approximation to the data to make triangles and improve them.
–**Adaptive coarsening of the triangles** keeps tolerance and genus to construct the base mesh.
–**Subdividing using local least-squares polynomial approximation** applies ideas from MLS for surfaces to have higher approximation quality.
–**Improved weight function in the MLS steps** handles outliers and features of high curvature.

2 Point geometry sampling based on visual hull

We will now explain in detail how to sample points from multiple images. We create points assumed to be on the visual hull surfaces by combining LDC surfel sampling scheme (see Figure 3).

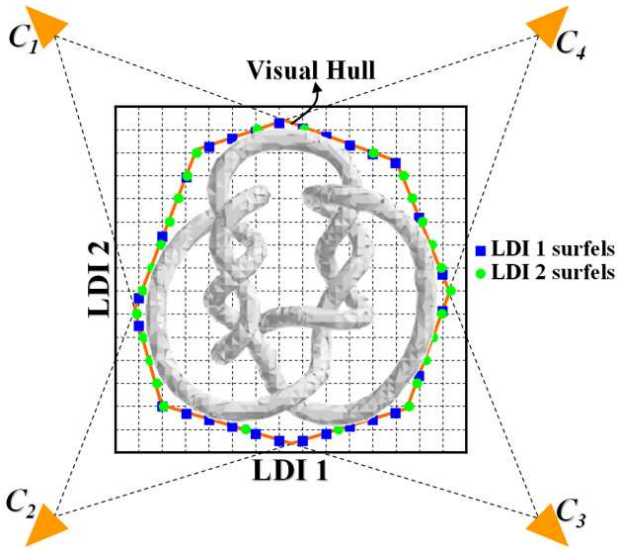


Fig. 3: An illustration of surfel sampling from a visual hull

To create points from multiple images, several processing steps are performed as seen in Figure 4. First, we create an LDC structure which consists of 6 LDI(Layered Depth Image) planes with user-specified sampling resolution m. Then we sweep each LDI plane in the positive and negative directions of x, y, and z axes. No ray casting is executed which was used in the previous surfel sampling approach [20]. As each LDI plane moves, some cameras are activated to test the validity of each LDI pixel. All the LDI pixels are tested to find whether they show consistent colors in the images corresponding to the activated cameras. The surface point is created at the position of color-consistent pixel. This process is repeated until no more surface point is reconstructed from the sweeping of 6 LDI planes. After all the space has been swept, the sampling process is completed.

2.1 Camera activation

Before testing the color-consistency of each LDI pixel, we must determine which camera can clearly see the LDI

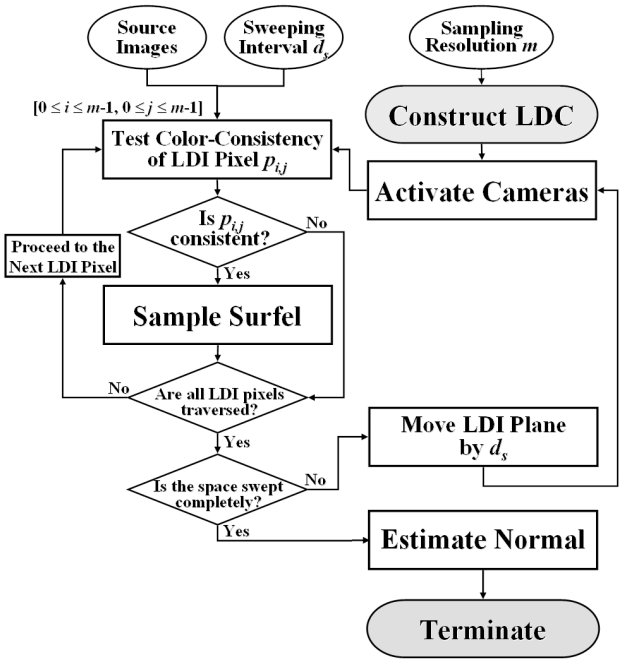


Fig. 4: Point sampling overview

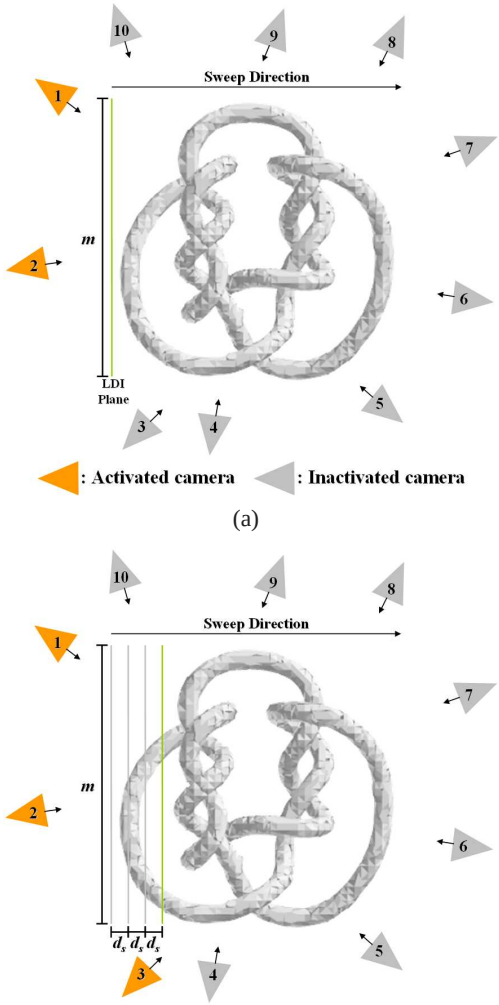


Fig. 5: Camera activation overview

pixels on the LDI plane of current sweep position. By combining LDC sampling with the plane sweeping method [14], we maintain an ordinal visibility that dramatically solves the self-occlusion problem occurred in imaging process.

We move the sweep planes, called LDI planes in this paper, in the positive and negative directions of x, y, and z axes. In the previous method, the sweep planes were incremented along each coordinate direction by the voxel grid size. Instead of that way, we move the LDI planes with user-specified sampling interval d_s (see Figures 5 and 6). By controlling d_s appropriately, the lack of precision introduced by the previous space discretization method [14,22] is largely avoided.

As LDI plane moves, each input camera is activated when it is behind the LDI plane and the angle between its viewing direction and sweep direction is within a threshold θ. To reduce the influence of blurring in the imaging process, we set the threshold θ to 60^o and this value works well. In Figure 5(a), only Cameras 1 and 2 are initially activated. As LDI the moves in the sweep direction, Camera 3 is activated as the LDI plane passes it (see Figure 5(b)). Although Camera 10 is also passed by LDI plane, it cannot be activated because its viewing direction is far from the sweep direction.

2.2 Color-consistency constraint

After the camera activation, we sample 3D surface points at the LDI pixels that are assumed to be on the surfaces of visual hull. As widely known, visual hull represents the maximal shape and appearance computed from multiple images. Seitz et al. introduces color-consistency constraint to compute visual hull accurately and robustly [22]. They noticed that the same point on the surface should be the consistent color in any images that capture it. That is to say, if a certain point has the consistent color in any visible images, it is probably on the valid surface of the target object. In this paper, all of LDI pixels are tested whether they show consistent colors in the images of the activated cameras. At the color-consistent LDI pixel positions, surface points are sampled.

Figure 6 shows a 2D example of the color-consistency test. In Figure 6(a), LDI pixel 1 is determined not to be on the surface because it has different colors in the images taken by Cameras 1 and 2. LDI pixel 2 shows consistent colors in all cameras that can see it, so a surface point is created at that position and its color is assigned to be red.

LDI pixel 3 is determined to be in empty space without color-consistency test because its projection is in the background area of Camera 2. From the known property of visual hull and color-consistency constraint, if a certain area is projected onto the background of any image, it is absolutely outside the target object. After moving the LDI plane through the user-specified distance d_s, the same process is repeated on all LDI pixels. Notice that Camera 3 is not included in testing the color-consistency of LDI pixel 4, because this LDI pixel projects on the occupied area of that cameras image. This means that LDI pixel 4 is not visible from Camera 3 according to the property of ordinal visibility. By testing LDI pixel 4 with Cameras 1 and 2, another surface point is created at that position. For the same reason, camera 2 is not considered in testing LDI pixel 5.

3 Displaced butterfly subdivision surface construction

One of the difficulties in reconstruction from a point cloud is the topology identification. To solve this, first of all we build a representative mesh by carving empty cubes from outside in the bounding volume grid of appropriate size, and shrink-wrap the outer surface of non-empty cubes onto the point cloud. And then we simplify the representative mesh to construct a base triangular mesh. If coarse blocks are used for generating a base mesh without simplification step, there will be a possibility of missing the genus of point data. The cost function of simplification estimates the error between an original set of points and a triangular mesh for a base mesh to preserve the appearance of point clouds. At last we subdivide the base mesh and at the same time sample the displacement value for each odd vertex in the direction of its normal based on MLS approximation.

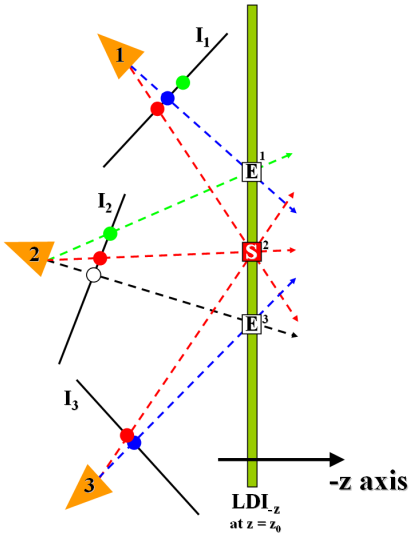
3.1 Notation

$\mathscr{P}$ is the set of all the points is the point cloud. A triangulation $\mathscr{T}$ consists of a set of vertices $V(\mathscr{T})$, a set of edges $E(\mathscr{T})$, and a set of triangles $T(\mathscr{T})$. For each $e \in E(\mathscr{T})$ and $v \in V(\mathscr{T})$, we denote by $v \in e$ the relation that v is a boundary vertex of the edge e. For each $t \in T(\mathscr{T})$ and $v \in V(\mathscr{T})$, we denote by $v \in t$ the relation that v is a vertex of the triangle t.

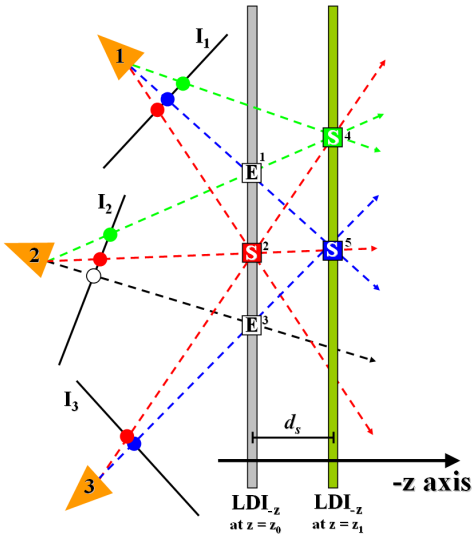
For each $e \in E(\mathscr{T})$, Q_e is the quadrilateral formed by the two triangles sharing e. For $v \in V(\mathscr{T})$ we denote by C_v the cell of v in $\mathscr{T}$, namely $C_v = \{t \in T(\mathscr{T}) : v \in t\}$. For $e \in E(\mathscr{T})$ we denote by C_e the cell of e in $\mathscr{T}$, namely $C_e = \{C_v : v \in e\}$ (Figure 7(a)).

For each $v \in V(\mathscr{T})$, the vector $\mathbf{n}_v = n(v, \mathscr{T})$ is the average of the normals to all $\{t \in T(\mathscr{T}) : v \in t\}$. To each $v \in V(\mathscr{T})$, we attached a subset $\mathscr{P}_v$ of points from $\mathscr{P}$ which are close to v.

In our algorithm the surface from which the point cloud is sampled is locally approximated by a quadratic



(a)



(b)

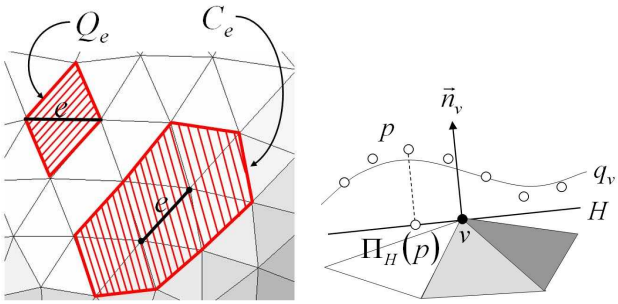
E: Empty pixel S: Sampled pixel

Fig. 6: 2D illustration of color-consistency test with the LDI plane sweeping in the -z direction: (a) when LDI_{-z} at z = z_0, (b) when LDI_{-z} at z = z_1.

MLS polynomial (Figure 7(b)). The quadratic polynomial at a vertex v relative to a reference plane H_v through v with normal $\mathbf{n}_v$ and to a set $\mathscr{P}_v$ of k-nearest neighboring points of v, (k is the user parameter between 10 and 20.) is the quadratic polynomial q_v minimizing

$$\sum_{p \in \mathscr{P}_v} \|q_v(\Pi_{H_v}(p)) - f_v\|^2 e^{-\frac{\|p-v\|^2}{h_v^2}} \qquad (1)$$

among all quadratic polynomials. Here $h_v = \sqrt{\frac{\pi R^2}{|\mathscr{P}_v|}}$, where R is the radius of the smallest bounding sphere including



(a) (b)

Fig. 7: The quadrilateral Q_e and the cell C_e of an edge e and the MLS projection.

the points of $\mathscr{P}_v$, Π_{H_v} is the orthogonal projection to the plane H_v, and $f_v = (p - \Pi_{H_v}(p)) \cdot \mathbf{n}_v$.

3.2 Construction of a representative mesh

To construct a representative mesh from a point cloud, first we split the bounding box which includes the point cloud into cubes. The size of the cubes should depend on the sampled surface. By carving the empty cubes from outside, we can extract the shape of the surface to be approximated. At the beginning of this procedure, we remove the outmost empty cubes. We repeat this steps until there are no more the empty outmost cubes. Then we triangulate the boundary faces of the remaining block of cubes, by adding a diagonal to each to face, and obtain the triangulation $\mathscr{T}_0$.

To get an initial triangulation of the sampled surface we shrink-wrap $\mathscr{T}_0$ to the point cloud. We apply the shrink-wrapping approach [13] rather than the marching cubes algorithm [19], because a reconstructed surface by marching cubes algorithm requires a post-processing like hole-filling and removal of small triangles [15]. The shrink-wrapping approach has two basic operations: projection followed by regularization. The projection operator moves a vertex to the corresponding the MLS surface and the regularization operator moves a vertex in direction tangent to the surface to separate vertices apart. To improve the regularity of the resulting triangulation, with the projected vrtices and the same connectivity, denoted by $\mathscr{T}_1$, we apply to $\mathscr{T}_1$ the regularization operator. These operators are given by

$$Proj(v) = \Pi_{H_v}(v) + q_v(\Pi_{H_v}(v))\mathbf{n}_v, \quad v \in V(\mathscr{T}_0) \tag{2}$$

$$Reg(v) = t_v + q_v(t_v)\mathbf{n}_v, \quad v \in V(\mathscr{T}_1) \tag{3}$$

where $t_v = \frac{1}{|C_v|}\Pi_{H_v}(\sum_{u \in C_v} u)$. The resulting triangulation $\mathscr{T}_2$ is a good representative mesh of the geometry and topology of the surface.

The neighborhood $\mathscr{P}_v \in \{\mathscr{P}_u : u \in V(\mathscr{T}_i)\}, i = 1,2,3$ consists of the closest k points from the point cloud to the vertex v with k a user parameter. In our simulation k was 64 or 128. The simulations with $k = 32$ were unsatisfactory.

3.3 Construction of a base mesh

We construct a triangular base mesh by simplifying the representative mesh, using edge collapse with a cost function which estimates the error incurred by the edge collapse. The simplification algorithm consists of two parts – computation of cost function of $e \in E(\mathscr{T})$ and collapsing edges.

Computation of the cost function

For each edge, we compute its cost function by virtually collapsing it, and estimating the error between a local approximation to the surface at the point of collapse and the cell of that point in the collapsed mesh. This is done by following steps:

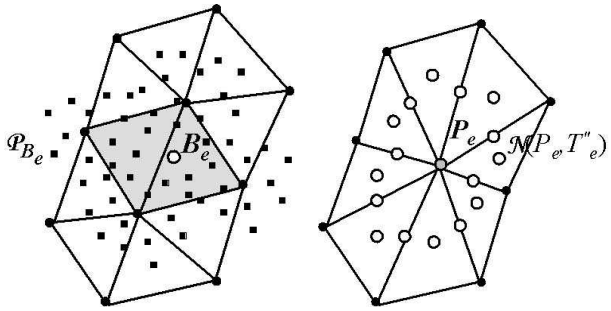
- **STEP 1** – We compute the butterfly point B_e corresponding to e relative to $\mathscr{T}$.
- **STEP 2** – We virtually collapse the edge e to the point B_e and then denote by $\mathscr{T}_e'$ the resulting triangulation in the cell C_e.
- **STEP 3** – We attach virtually to B_e the points from $\{\mathscr{P}_v : v \in Q_e\}$ which are closest to B_e than to the four vertices of Q_e. We denote this set of points by $\mathscr{P}_{B_e}$.
- **STEP 4** – We determine a reference plane H_e for B_e, as the plane touching B_e with normal $\mathbf{n}_e = n(B_e, \mathscr{T}_e')$. We denote by $\Pi_e(p)$ the orthogonal projection of a point p on H_e, represented by a local coordinate system of H_e such that $\Pi_e(B_e) = (0,0)$.
- **STEP 5** – We compute the quadratic MLS relative to B_e and H_e, based on the points in $\mathscr{P}_{B_e}$ and denote it by q_e.
- **STEP 6** – We compute the point $P_e = (0,0,q_e(0,0))$ (with respect to the coordinate system relative to H) and virtually collapse the edge e to P_e. The resulting virtual triangulation is denoted by $\mathscr{T}_e''$.
- **STEP 7** – The cost function of the edge e in $E(\mathscr{T})$ is the value,

$$F(e,\mathscr{T}) = \max\{\|q_e(\Pi_e(p)) - p\|^2 : p \in \mathscr{N}(P_e, \mathscr{T}_e'')\} \tag{4}$$

with $\mathscr{N}(P_e, \mathscr{T}_e'') = \{\text{midpoint of } \varepsilon : \varepsilon \in E(\mathscr{T}_e'')\} \cup \{\text{barycenter of } t : t \in T(\mathscr{T}_e'')\}$.

Simplification step

We choose $e^* \in E(\mathscr{T})$ such that $F(e^*,\mathscr{T}) = \min_{e \in E(\mathscr{T})} F(e,\mathscr{T})$. We collapse e^* to the point $p_{e^*} = (0,0,q_{e^*}(0,0))$. Then we update the triangulation by replacing the triangulation in $C_{e^*}(\mathscr{T})$ by $\mathscr{T}_{e^*}''$ and the neighborhoods of the vertices after the



(a) before (b) after

Fig. 8: Virtual edge collapse; the set of points $\mathscr{P}_{B_e}$ are approximated by an MLS surface. The cost function is defined by the maximum height of the points of the set $\mathscr{N}(P_e, \mathscr{T}''_e)$ from the reference plane H_e.

collapse. We attach each point in $\mathscr{P}_v$, $v \in e^*$ to the closest vertex to it among $v \in V(\mathscr{T}''_{e^*})$. Finally we update the cost function relative to the newly formed triangulation. This update is only local, since the collapse of e^* only affects the cost function of the edges of triangles with edge in $\mathscr{T}''_{e^*}$.

3.4 Displaced butterfly subdivision surface

After subdividing the polygonal mesh with the modified butterfly scheme [26], we sample the displacements by local approximation based on quadratic MLS. For each edge $e \in E(\mathscr{T})$, we compute the butterfly point B_e. We denote by $\mathscr{T}'$ the refined triangulation after subdivision. We define $\mathscr{P}_{B_e} = \{P_v : v \in Q_e\}$. We determine a reference plane H_e for B_e, as the plane touching B_e with normal $\mathbf{n}_e = n(B_e, \mathscr{T}')$. We compute the quadratic MLS polynomial q_e relative to B_e and H_e, based on $\mathscr{P}_{B_e}$. The displacement value d_e is

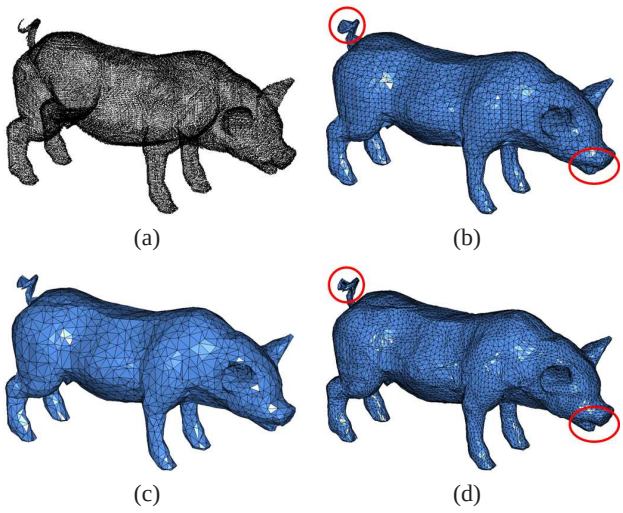
$$d_e = \|(0,0,q_e(0,0)) - B_e\|. \quad (5)$$

After sampling the displacements d_e of all edges, we update the neighboring points from the point cloud. We attach to the displaced point $B_e + \mathbf{n}_e d_e$ the neighborhood $\mathscr{P}_{B_e}$. For each $v \in Q_e$ we remove from $\mathscr{P}_v$ the points that are closer to $q_e(B_e)$ than to v.

4 Results and discussion

We used 36 input images with a resolution of 512^2 for the point cloud recovery of each model, which were captured at evenly distributed camera positions. We captured the input images using evenly distributed virtual cameras to evaluate the robustness and performance of our method.

As shown in the red circled areas in 9(b) and 9(c), some poorly shaped features are corrected after the simplification and subdivision steps are performed. This is because they are restructured into the inner area of the point cloud during the simplification step, while their displacement is sampled outward.



(a) (b) (c) (d)

Fig. 9: Pig model result. (a) Recovered point cloud(105,733 points) (b) Shrink-wrapped polygonal model(5,557 vertices and 11,110 faces) (c) Simplified base mesh(2,002 vertices and 4,000 faces) (d) Final DBSS(8,002 vertices and 16,000 faces)

Figures 9 and 10 show the reconstruction results when using our algorithm. From a point cloud (a) we created a polygonal mesh structure (b) by shrink-wrapping and we constructed a base mesh (c) by simplifying a shrink-wrapped mesh using a cost function based on the modified butterfly subdivision scheme and MLS approximation. Finally, we produced a displaced butterfly subdivision surface (d) using the modified butterfly subdivision method and by sampling the scalar-valued offset between a subdivision surface and MLS surface. The processing time required to reconstruct the DBSS from point data was as little as 10 seconds. We can perform parallel processing for rapid reconstruction even if a given point set is massive, because our method uses local approximation.

Our method is automatic, except for the description of the cube blocks used for carving empty cubes. The reconstruction result will have holes if the size specified is smaller than any holes. It is possible that the displacement values will be increased or incorrect if the size is increased, because the edges of the representative mesh are distant from the point clouds. However, we can sometimes skip the simplification step if users select cubes of a suitable size. Thus, the representative mesh that is shrink-wrapped from coarse cube blocks becomes

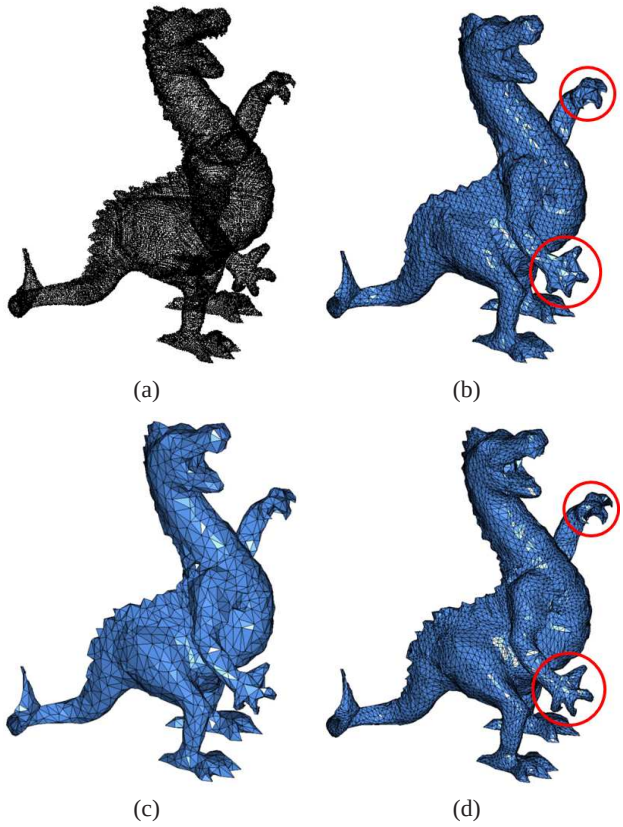


Fig. 10: Dragon model result. (a) Recovered point cloud(126,912 points) (b) Shrink-wrapped polygonal model(7,409 vertices and 14,814 faces) (c) Simplified base mesh(2,002 vertices and 4,000 faces) (d) Final DBSS(8,002 vertices and 16,000 faces)

a direct base mesh for subdivision with simple object shapes such as a sphere or a cylinder.

5 Conclusions and future work

In this paper, we proposed a novel method for creating a DBSS that approximates points sampled from the visual hull based on MLS. Our LDI plane sweeping simplifies the visibility computation to maintain ordinal visibility during the surface point sampling step.

We use the color-consistency constraint of carving theory to obtain points closer to the true surfaces of a target object. Our DBSS exploits the property of MLS approximation during the generation of a base mesh while sampling displacement values from a subdivision surface. To enhance the memory efficiency, we use a butterfly subdivision scheme rather than an approximation scheme to sample the offsets of odd vertices only.

This reduces the data volume because it only requires a scalar value to express each vertex, in addition to the storage costs of the coarse base mesh. Our experimental results show that this method works well with various complex objects.

In future work, we hope to develop adaptive subdivision and sampling capacities. Feature-sensitive sampling of sharp edges is also required.

Acknowledgement

This work was supported by a Korea university grant and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2012-0005713).

This work was supported by the Industrial Strategic technology development program(No.10041784, Techniques for reproducing the background and lighting environments of filming locations and scene compositing in a virtual studio) funded by the Ministry of Knowledge Economy(MKE, Korea).

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (No. 2011-0015072).

References

[1] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. Silva, Computing and rendering point set surfaces, IEEE Transactions on Visualization and Computer Graphics, **9**, 3-15 (2003).

[2] N. Amenta, M. Bern, and M. Kamvysselis, A new voronoi-based surface reconstruction algorithm, Proc. ACM SIGGRAPH, **98**, 415–421 (1998).

[3] N. Amenta and Y.J. Kil, Defining point-set surfaces, ACM Transactions on Graphics(Proc. ACM SIGGRAPH 2004), **23**, 264-270 (2004).

[4] A. Bermano, A. Vaxman, and C. Gotsman, Online reconstruction of 3d objects from arbitrary cross-sections, ACM Transactions on Graphics(Proc. ACM SIGGRAPH Asia 2011), **30**, 113:1-113:11 (2011).

[5] J. Carr, R. Beatson, J. Cherrie, T. Mitchell, W. Fright, B. McCallum, and T. Evans, Reconstruction and representation of 3d objects with radial basis functions, In Proc. ACM SIGGRAPH, **2001**, 67–76 (2001).

[6] B. Curless and M. Levoy, A volumetric method for building complex models from range images, Proc. ACM SIGGRAPH, **96**, 303-312 (1996).

[7] M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery, and W. Stuetzle, Multiresolution analysis of arbitrary meshes, Proc. ACM SIGGRAPH, **95**, 173-182 (1995).

[8] J. P. Grossman and W.J. Dally, Point sample rendering, Proc. of the 9th Eurographics Workshop on Rendering, (1998).

[9] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle, Piecewise smooth surface reconstruction, Proc. ACM SIGGRAPH, **94**, 295-302 (1994).

[10] A. Hornung and L. Kobbelt, Robust reconstruction of watertight 3d models from non-uniform sampled point clouds without normal information. In Eurographics Symposium on Geometry Processing (SGP 2006), 41-50 (2006).
[11] H. Huang, D. Li, H. Zhang, U. Ascher, and D. Cohenor, Consolidation of unorganized point clouds for surface reconstruction, ACM Transactions on Graphics(Proc. ACM SIGGRAPH Asia 2009), **28**, 176:1-176:7 (2009).
[12] W.-K. Jeong and C.-H. Kim, Direct reconstruction of displaced subdivision surface from unorganized points, Graphical Models, **64**, 78-93 (2002).
[13] L. Kobbelt, J. Vorsatz, U. Labsik, and H.-P. Seidel, A shrink wrapping approach to remeshing polygonal surfaces, Computer Graphics Forum(Proc. Eurographics 99), **18**, 119-130 (1999).
[14] K. N. Kutulakos and S. M. Seitz, A theory of shape by space carving, International Journal of Computer Vision, **38**, 199-218 (2000).
[15] U. Labsik, K. Hormann, M. Meister, and G. Greiner, Hierarchical iso-surface extraction, Journal of Computing and Information Science in Engineering, **2**, 323-329 (2002).
[16] A. Lee, H. Moreton, and H. Hoppe, Displaced subdivision surfaces, Proc. ACM SIGGRAPH, **2000**, 85-94 (2000).
[17] D. Levin, Mesh-independent surface interpolation, Geometric Modeling for Scientific Visualization, 37-49 (2003).
[18] C. Loop, S. Schaefer, T. Ni, and I. C. No, Approximating subdivision surfaces with gregory patches for hardware tessellation, ACM Transactions on Graphics(Proc. ACM SIGGRAPH Asia 2009), **28**, 151:1-151:9 (2009).
[19] W. Lorensen and H. Cline, Marching cubes: A high resolution 3d surface reconstruction algorithm, Computer Graphics(Proc. ACM SIGGRAPH 87), **21**, 163-169 (1987).
[20] H. Pfister, M. Zwicker, J. V. Baar, and M. Gross, Surfels: Surface elements as rendering primitives, Proc. ACM SIGGRAPH, **2000**, 335-342 (2000).
[21] S. Rusinkiewicz and M. Levoy, Qsplat: A multiresolution point rendering system for large meshes. Proc. ACM SIGGRAPH, **2000**, 343-352 (2000).
[22] S. M. Seitz and C. R. Dyer, Photorealistic scene reconstruction by voxel coloring, International Journal of Computer Vision, **35**, 151-173 (1999).
[23] S. Shalom, A. Shamir, H. Zhang, and D. Cohen-Or, Cone carving for surface reconstruction, ACM Transactions on Graphics(Proc. ACM SIGGRAPH Asia 2010), **29**, 150:1-150:10 (2010).
[24] A. Sharf, T. Lewiner, G. Shklarski, S. Toledo, and D. Cohen-Or, Interactive topology-aware surface reconstruction, ACM Transactions on Graphics(Proc. ACM SIGGRAPH 2007), **26**, (2007).
[25] A. Tagliasacchi, M. Olson, H. Zhang, G. Hamarneh, and D. Cohen-Or, Vase: Volume-aware surface evolution for surface reconstruction from incomplete point clouds, Computer Graphics Forum(Special Issue of Symposium on Geometry Processing 2011), **30**, 1563-1571 (2011).
[26] D. Zorin, P. Schröder, and W. Sweldens, Interpolating subdivision for meshes with arbitrary topology, In Proc. ACM SIGGRAPH, **96**, 189-192 (1996).

Jung Lee is a research professor in the department of information & communication in Korea University. His research interests include Computer Graphics, Image Processing, and Image-based Modeling/Rendering.

Chang-Hun Kim is a Professor in the Department of Brain and Cognitive Engineering at Korea University. His research interests include Fluid Simulation and Geometric Modeling. He is also a member of IEEE Computer Society and ACM.

Sun-Jeong Kim is an associate professor in the department of ubiquitous computing in Hallym University. Her research interests include Geometric Modeling, Scientific Visualization, Virtual Reality and Augmented Reality.