Rendering of Unorganized Points with Octagonal Splats

Sun-Jeong Kim and Chang-Geun Song

Hallym University,
Division of Information Engineering and Telecommunications,
Chuncheon-si, Gangwon-do 200-702 Korea

Abstract. Point sets from range scanners are noisy and do not contain additional data such as normals, which makes it difficult to create a mesh for rendering. We use the moving least-squares (MLS) approximation to get normals and differential information, and approximate the surface near each point by an MLS surface. From this we obtain a splat of a new octagonal shape which approximates the local curvature at a point and allows fast and high-quality rendering. We show several results which demonstrate the practicality and benefits of our algorithm.

1 Introduction

Recent advances in scanning technology have led to a rapid increase in the availability and size of geometric data sets. Some consist of hundreds of millions of points. Point data are usually considered by generating and simplifying a triangular mesh. However some of the meshes obtained from scan data are too large to be displayed in real time, and the running-time and space requirements of mesh reduction and progressive display algorithms are impractical at this scale.

Point-based rendering schemes have evolved as an alternative to the use of intermediate triangle-based representations. Their major advantage is that point sets do not require any connectivity information, and no topological consistency conditions have to be satisfied. Moreover these schemes involve a hierarchical data structure which allows the efficient control of frame rates and visual quality.

Most point rendering algorithms handle points sampled from a large mesh, which means that the normals of points can be easily obtained from the input mesh. Grossman and Dally [4] converted geometric models into point-sampled data sets and then rendered them. They addressed the issues of sampling rate and gaps in the rendered images. The *Surfel* concept [8] follows from this work. This technique uses an octree-based structure and forward warping. A significant challenge for point rendering techniques is to reconstruct continuous surfaces from irregularly spaced point samples. In order to eliminate any gaps in the displayed objects, splat-based schemes are used. *QSplat* [9] replaces points by ellipses or rectangles in image space and uses a hierarchy of bounding spheres. Sphere radii and normal cones are saved to perform time-critical rendering. *Surface splatting* [11] is an object-space technique and focuses on texture filtering. Point rendering with Gaussian resampling kernels, also called *EWA splatting*, has provided the highest image quality so far, with antialiasing capabilities. This technique has been applied to interactive shape design [12]. To reduce the computational complexity of EWA splatting, Botsch et al. [3] proposed an approximation using look-up tables. The above point rendering schemes store a limited amount of information derived from the point primitives, such as normals, bounding spheres, and tangent plane disks. But Kalaiah and Varshney [5] store local differential geometric information with every point. Their approach renders a surface as a collection of local neighborhoods, consisting of a rectangle for each point, with per-pixel shading. Although the rendering quality is very high but their scheme cannot be applied to an irregular point set because they sample points uniformly in the parametric domain of a NURBS surface. Wu and Kobbelt [10] presented sub-sampling technique for dense point sets. They compute a minimal set of splats that covers a point set within a given error tolerance. To reduce the splat count, point simplification [7] is applied.

Levin [6] has recently introduced a new point-based surface representation called moving least-squares (MLS) surfaces. Based on this representation, Alexa et al. [1] have proposed new modeling and rendering techniques, and Amenta and Kil [2] have developed a new explicit definition of point-set surfaces and variants of the MLS surface.

In this paper, we propose a new algorithm to render completely unorganized points which have been obtained from a range scanning system. For each point, the normal vector is computed by the eigen-analysis of the covariance matrix and then with this normal an MLS surface is generated to approximate the points in a small neighborhood around it. After approximation, we construct a splat of an octagonal shape by extracting the positions and normals of vertices of octagon from an MLS surface. At last the points are rendered using these octagonal splats. Our contributions are the follows:

- We handle point sets straight from a range scanning system. Previous rendering algorithms make use of sampling order or other pre-existing information such as normals and differential information. But our technique allows the display of point sets that are genuinely unorganized.
- We render points with octagonal splats which allow for the curvatures at a point. The two orthogonal axes and normals of each octagon correspond to the estimated principal curvature directions and normals on an approximated smooth surface. And an octagon is faster to draw than a circle or ellipse.

2 Normal Estimation

Normal vectors are estimated by analyzing the local neighborhood of each point. Our computation is similar to [1]. Since there is no connectivity information in a point set, the local neighborhood of each point is constructed by k-nearest neighbors. Let r be a point and $\{r_0, \dots, r_{k-1}\}$ its k-nearest neighbors. The matrix of weighted covariances be defined by

$$C := \sum_{i=0}^{k-1} \theta_i (r_i - r)(r_i - r)^T \in \mathcal{R}^{3\times 3}, \tag{1}$$

with the weights $\theta_i = \theta(\|r_i - r\|)$. θ is a non-negative weight function, usually a Gaussian of the form:

$$\theta(d) = e^{-\frac{d^2}{h^2}},$$

where h is a fixed parameter reflected the anticipated spacing between neighboring points. Finding a global value of h is difficult for non-uniformly distributed point clouds. Similar to [7], for each point we choose different h value as the distance to the seventh closest neighbor point from it.

The eigenvector of C corresponding to the smallest eigenvalue gives an estimate for the normal direction. Since there is no consistent orientation over all points, we correct the normal directions by propagation along a minimal spanning tree. We also use the other two eigenvectors for computing a splat shape because they are approximately equal to the principal curvature directions $\boldsymbol{\kappa}_1$ and $\boldsymbol{\kappa}_2$.

3 MLS Approximation

We begin with the brief review of the definition of MLS surfaces [6]. An MLS surface is implicitly defined as points that project on to themselves under the MLS projection. The procedure to approximate this surface involves two steps: the approximation of a local reference plane and fitting a local bivariate polynomial to points projected on to the reference plane.

Let $\{r_i\}_{i\in I}$ be points on a surface S in $\mathcal{R}^3$ (possibly subject to noise). Given a point r, we first find a local reference plane $H = \{x | \langle \boldsymbol{n}, x\rangle - d = 0, x \in \mathcal{R}^3\}, \boldsymbol{n} \in \mathcal{R}^3, \|\boldsymbol{n}\| = 1$ in $\mathcal{R}^3$, and a point q on H (i.e. $\langle \boldsymbol{n}, q\rangle = d$ – see Figure 1) such that the following quantity is minimized:

$$\sum_{i\in I} (\langle \boldsymbol{n}, r_i\rangle - d)^2 \theta(\|r_i - q\|), \tag{2}$$

where $\langle\cdot,\cdot\rangle$ is the standard inner product in $\mathcal{R}^3$, $\|\cdot\|$ is the Euclidean distance in $\mathcal{R}^3$.

Now we compute a local bivariate polynomial approximation of S over the reference plane. Let $\{x_i\}_{i\in I}$ be the orthogonal projections of the points $\{r_i\}_{i\in I}$ on to H, represented in a specific orthonormal coordinate system on H, and let $f_i = \langle n, r_i\rangle - d, i \in I$ be the heights of the points $\{r_i\}_{i\in I}$ over H. We will choose the origin of the orthonormal coordinate system on H to be at q. We define a local approximation to S of degree m as a bivariate polynomial p that minimizes the weighted least-squares error:

$$\sum_{i\in I} (p(x_i) - f_i)^2 \theta(\|r_i - q\|). \tag{3}$$

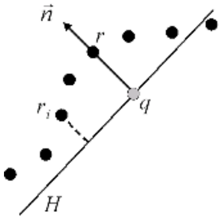


Fig. 1. The MLS projection. Given a point r, a local reference plane H and a point q on H are computed. An MLS surface is defined implicitly by the MLS projection as the set of points that project on to themselves.

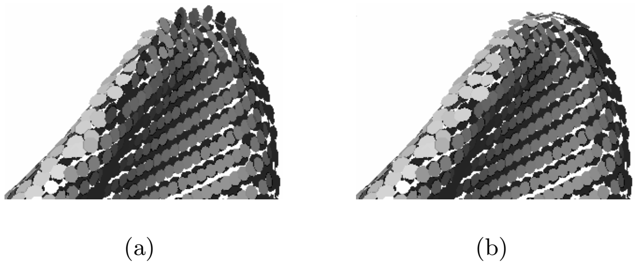


Fig. 2. Splat shapes with (a) the wrong-estimated normals on the top of a point-set by eigen-anlaysis and (b) the corrected normals by minimizing the MLS error

In this paper, we use a third-degree polynomial, because it produces a good fit, does not oscillate, and is quick to compute.

To obtain an MLS approximation surface, we should solve non-linear minimization problem. However, for simplifying this computation, we use an estimated normal as $\boldsymbol{n}$ and set $t = 0$ (i.e. $q = r$), since in our experiments an estimated normal is reliable and the position of a reference plane is not important after the normal direction is fixed. Even if an estimated normal is wrong, it can be corrected by choosing the one to minimize the MLS error among three eigenvectors (see Figure 2).

4 Octagonal Splat

After the MLS approximation, we construct an octagonal splat shape and its normal vectors for each point. The two orthogonal axes and normals of each octagonal splat correspond to the estimated principal curvature directions and normals on an approximated MLS surface. The octagonal shape is faster to render than the elliptical shape which many previous schemes constructed. For hole-free rendering, the Voronoi diagram is used. Because an octagonal splat covers the Voronoi cell which includes the point, there is no gap between splats.

On the reference plane H of a point r (Figure 3(a)), the Voronoi diagram of orthogonal projected points is computed (Figure 3(b)). A bounding square is

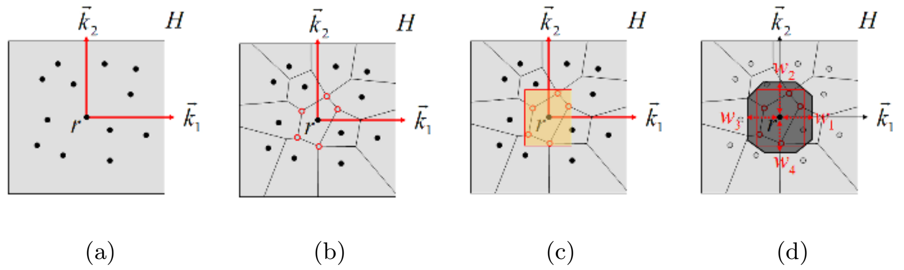


Fig. 3. Construction of an octagonal splat shape: (a) the orthogonal projections of neighbor vertices for each point r on to H, (b) the Voronoi diagram, (c) a bounding square which is constructed covering the Voronoi cell, (d) four parameters ω_1, ω_2, ω_3, and ω_4

constructed covering the Voronoi cell which includes the point r (Figure 3(c)), and then an octagon is constructed covering this square (Figure 3(d)). Now we decide the parameters ω_1, ω_2, ω_3, and ω_4 by measuring the distance from the point r to the octagon in the direction of principal axes $\boldsymbol{\kappa}_1$, $\boldsymbol{\kappa}_2$, $-\boldsymbol{\kappa}_1$, and $-\boldsymbol{\kappa}_2$ respectively. These parameters make an octagon whose width is $\omega_1 + \omega_3$ and whose height is $\omega_2 + \omega_4$. The center of an octagonal splat is the point $O(0,0)$ in the orthogonal coordinate system on H and its eight end-points are $x_i = \{(\omega_1, -s\omega_4), (\omega_1, s\omega_2), (s\omega_1, \omega_2), (-s\omega_3, \omega_2), (-\omega_3, s\omega_2), (-\omega_3, -s\omega_4), (-s\omega_3, -\omega_4), (s\omega_1, -\omega_4)\}$, where $s = \sqrt{2} - 1$. Therefore the vertices v_i of the octagon for each point r in $\mathcal{R}^3$ are computed as follows:

$$v_i = r + \alpha_i \boldsymbol{\kappa}_1 + \beta_i \boldsymbol{\kappa}_2 + \gamma_i \boldsymbol{n}, \tag{4}$$

where $x_i = (\alpha_i, \beta_i)$ and $\gamma_i = p(x_i) - p(O)$.

The octagon is not planar, but distorted to correspond to the estimated curvature at the point. In the last step, we compute the normals of vertices of the octagon. Because we have already an MLS polynomial function, in order to obtain the normals we just differentiate the polynomial partially and then transform them into the global coordinate system. The normals $\boldsymbol{n}_i$ of the vertices v_i in the local coordinate system is

$$\boldsymbol{n}_i = \Big(-\frac{p(x_i)}{\partial \boldsymbol{\kappa}_1}, \quad -\frac{p(x_i)}{\partial \boldsymbol{\kappa}_2}, \quad 1 \Big). \tag{5}$$

5 Implementation and Results

All our tests were run on a PC with a Pentium IV 2.4 GHz processor and 2GB of main memory. Figure 4 shows the pictures produced by our point rendering algorithm. These figures show how our octagonal splat shape avoids gaps in the rendered image.

We have compared our rendering results with those of QSplat [9] and Pointshop 3D [12] (Figure 5 and 6). QSplat uses flat splats such as circles, ellipses, or

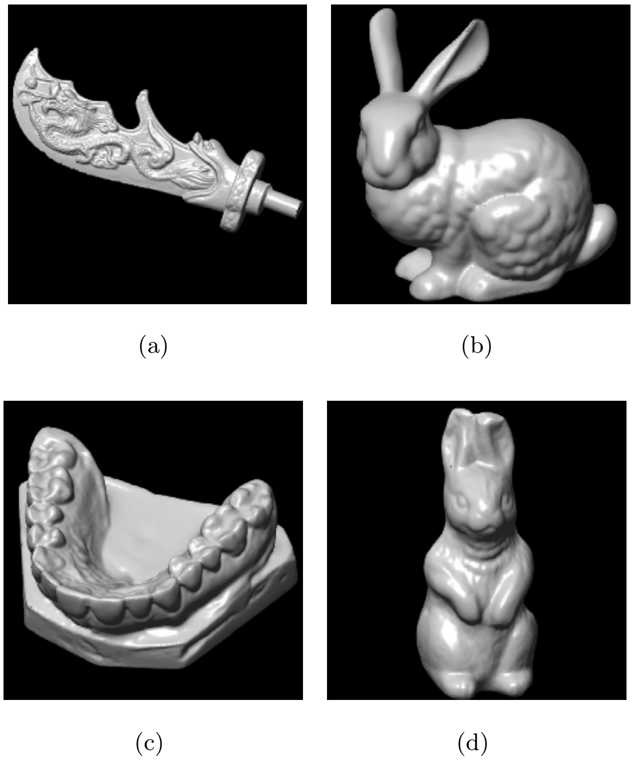


Fig. 4. Differential rendering of various point-sets: (a) the “Chinese Scimitar” (441,113 points) (b) the “Stanford Bunny” (34,834 points), (c) the “Teeth” (116,604 points), (d) the “Rabbit” (67,039 points)

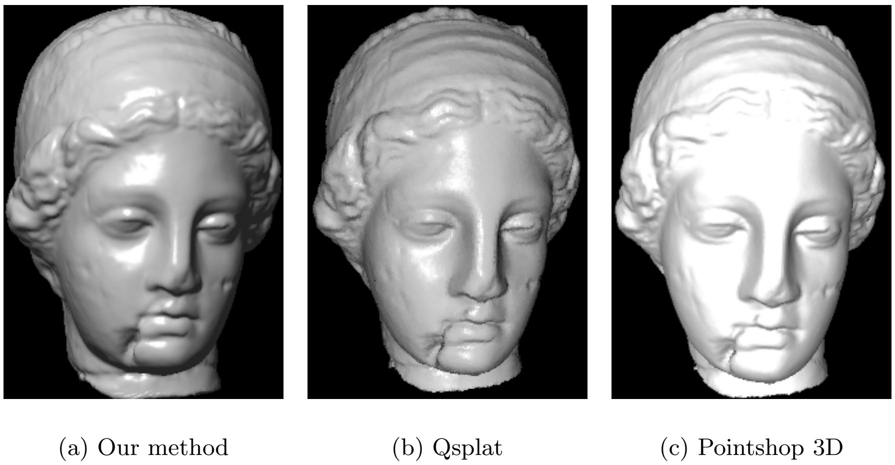
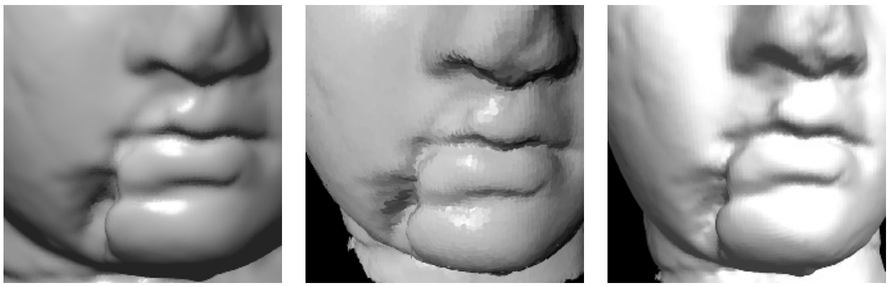


Fig. 5. Comparison of our technique with QSplat [9], and Pointshop 3D [12]: the “Igea” model (134,345 points)



(a) Our method (b) Qsplat (c) Pointshop 3D

Fig. 6. Close-up of Figure 5

rectangles, while ours is a curved octagon. Consequently, the visual quality of our results is higher than these of QSplat. Pointshop 3D samples the image on a regular grid based on Surfels. Therefore, the sampling technique may lead to aliasing. The close-ups in Figure 6 show that our rendering quality is higher than others.

6 Conclusion and Discussion

We have proposed a new scheme to render unorganized points. While previous schemes render sampled-point data, ours can handle scanned data which have no additional pre-existing information such as normals. To construct the splat shape and obtain the differential information, we made use of the MLS approximation. Also we built up octagonal splats for each point which reflected the curvature at each point.

During MLS approximation, it is very difficult to determine the nearest-neighbors distance in an irregular point-set. We used adaptive MLS surfaces that approximate the k nearest neighbors, but the best value of k remains undetermined. We believe that a preprocessing step to thin out over-dense data or fill holes would improve the robustness of our algorithm, and improve the quality of the resulting image. Many point rendering algorithms construct a hierarchical structure for real-time display. But the efficiency of this structure depends on the size of the dataset: it may actually slow down the display of small datasets. We need to pursue an efficient compromise.

Acknowledgments

This research was supported by Hallym University Research Fund, 2005 (HRF-2005 - 31) and the project for developing innovative local industries which was conducted by Regional Innovation Division in the Ministry of Commerce Industry and Energy of the Korean Government.

References

1. Alexa, M., Behr, J., Fleishman, S., Cohen-Or, D., Levin, D., and Silva, C. T. Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics 9*, 1 (2003), pp. 3–15.
2. Amenta, N. and Kil, Y. J. Defining point-set surfaces. *ACM Transactions on Graphics 23*, 3 (SIGGRAPH 2004), pp. 264–270.
3. Botsch, M., Wiratanaya, A., and Kobbelt, L. Efficient high quality rendering of point sampled geometry. *Proceedings of 13th Eurographics Workshop on Rendering* (2002), pp. 53–64.
4. Grossman, J. P. and Dally, W. J. Point sample rendering. In *Proceedings of the 9th Eurographics Workshop on Rendering* (1998), pp. 181–192.
5. Kalaiah, A. and Varshney, A. Differential point rendering. In *Proceedings of the 12th Eurographics Workshop on Rendering* (2001), pp. 139–150.
6. Levin, D. Mesh-independent surface interpolation. *Geometric Modeling for Scientific Visualization* (2004), Brunnett, G., Hamann, B., Muller, H., and Linsen, L. (eds.), Springer-Verlag, pp. 37–50.
7. Pauly, M., Gross, M., Kobbelt, L. Efficient simplification of point-sampled surfaces. In *Proceedings of IEEE Visualization 2002* (2002), pp. 163–170.
8. Pfister, H., Zwicker, M., Baar, J. V., and Gross, M. Surfels: surface elements as rendering primitives. In *Proceedings of SIGGRAPH 2000* (2000), pp. 335–342.
9. Rusinkiewicz, S. and Levoy, M. QSplat: a multiresolution point rendering system for large meshes. In *Proceedings of SIGGRAPH 2000* (2000), pp. 343-352.
10. Wu, J. and Kobbelt, L. Optimized sub-sampling of point sets for surface splatting. *Computer Graphics Forum 23*, 3 (Eurographics 2004), to appear.
11. Zwicker, M., Pfister, H., Baar, J. V., and Gross M. Surface splatting. In *Proceedings of SIGGRAPH 2001* (2001), pp. 371–378.
12. Zwicker, M., Pauly, M., Knoll, O., and Gross M. Pointshop 3D: An interactive system for point-based surface editing. *ACM Transactions on Graphics 21*, 3 (SIGGRAPH 2002), pp. 322–329.