Fitting Subdivision Surfaces to Unorganized Point Data Using SDM

Kin-Shing D. Cheng[†] Wenping Wang[†] Hong Qin[‡] Kwan-Yee K. Wong[†] Huaiping Yang[†] Yang Liu[†]

[†] Department of Computer Science
The University of Hong Kong
Pokfulam Road, Hong Kong, China
{ksdcheng, wenping}@cs.hku.hk

[‡] Department of Computer Science
State University of New York at Stony Brook
Stony Brook, New York, U.S.A.
qin@cs.sunysb.edu

Abstract

We study the reconstruction of smooth surfaces from point clouds. We use a new squared distance error term in optimization to fit a subdivision surface to a set of unorganized points, which defines a closed target surface of arbitrary topology. The resulting method is based on the framework of squared distance minimization (SDM) proposed by Pottmann et al. Specifically, with an initial subdivision surface having a coarse control mesh as input, we adjust the control points by optimizing an objective function through iterative minimization of a quadratic approximant of the squared distance function of the target shape. Our experiments show that the new method (SDM) converges much faster than the commonly used optimization method using the point distance error function, which is known to have only linear convergence. This observation is further supported by our recent result that SDM can be derived from the Newton method with necessary modifications to make the Hessian positive definite and the fact that the Newton method has quadratic convergence.

1. Introduction

1.1. Problem Statement

We propose a method that fits a Loop's subdivision surface S [21] to a target surface Γ defined by a set of unorganized points. The availability of the connectivity information of the target points is not assumed. Although we use Loop's surface, the basic framework can be applied to other linear subdivision schemes as well. We iteratively update the subdivision surface S to make it converge to a target shape by optimizing an objective error function. Previous subdivision surface fitting methods use the distance between a point on the fitting surface and the corresponding foot point on the target surface for minimizing the objective function; we will call these methods *point distance minimization*, or PDM. We use a quadratic approximant of the squared distance (SD) error function proposed by Pottmann et al [32], and show that the SD error function leads to better fitting results than using the conventional point distance error function. Our method will be called SDM, standing for *squared distance minimization*.

Although the Gauss-Newton method is preferred for solving a general nonlinear least squares problem, PDM is still the predominant optimization method used for surface fitting in CAD and graphics. By proving that SDM can be derived directly from the Newton method, we show that SDM is a more refined optimization technique than the Gauss-Newton method, which omits the true Hessian of an objective function. The purpose of this paper is to demonstrate that SDM is indeed a superior optimization technique for fitting a subdivision surface than the commonly used PDM.

1.2. Related Work

The problem of computing a compact surface representation of a target shape given by a set of unorganized data points has many applications in computer graphics, CAD, and computer vision. A typical formulation of the problem is computing a piecewise smooth surface, which can be a B-spline surface (including a NURBS surface) or a subdivision surface, that approximates a given target shape within a pre-specified error tolerance. Compared with the traditional methods based on B-spline surfaces, the approach using subdivision surfaces has gained increasing attention due to its ability to deal with general object topology as well as arbitrary connectivity of the control mesh [44, 45]. Approaches of different categories were proposed over the past decade. These include local fitting approaches [9–11, 24–26, 28, 37], active surface approaches [4–6, 17, 19, 23, 27, 33, 39, 40, 42, 43], implicit surface approaches [1, 2, 29, 38] and other approaches [12, 13, 15, 16, 20].

Among the previous work, some are more relevant to our approach. In Hoppe et al's method [10, 11], an initial dense mesh is generated from a set of unorganized points and is then decimated to fit the target shape via optimization of an energy function. Finally, a smooth subdivision surface is obtained from the mesh again via optimization. In Ma et al's method [24–26], base surfaces are built for obtaining the parameter values of the data points. Then a least squares procedure is used to fit B-spline surfaces on general quadrilateral topology and Catmull-Clark surfaces on extraordinary corner patches. In [28], closest point search on Loop's surface is performed by combining Newton iteration and nonlinear minimization, followed by an optimization with respect to the L^2 metric. In these methods, when the geometric error between the fitting surface and the target shape needs to be measured or minimized, the error function is defined by the summation of the squared distance between a point on the fitting surface and its closest point on the target shape. Hence, these methods use PDM (point distance minimization). Such optimization schemes, also called the alternating methods [34], are typically used for solving separable nonlinear least squares problems, and are known to have only linear convergence. In this paper, we propose to replace the conventionally-used geometric error with the SD error function. The resulting method, called SDM (squared distance minimization), can be applied to any existing frameworks in which distance functions are used to define a goal function.

2. Preliminaries

2.1. SDM

Pottmann et al. [30–32] proposed a general paradigm of shape approximation based on the minimization of a novel quadratic approximant of the squared distance function. Let p denote the foot point on the target surface Γ of a sample point v_0 from a fitting surface, i.e. p is the closest point on Γ to v_0. Let $d = \|v_0 - p\|_2$. Let ρ_1 and ρ_2 be the principal curvature radii of the surface Γ at p. Let T_1 and T_2 be the unit vectors in the corresponding principal curvature directions. Let N be the unit normal vector, i.e. $N = T_1 \times T_2$. A quadratic appproximant of the squared distance function from a variable point v in the neighborhood of v_0 to Γ is given by

$$\begin{aligned} F_d^+(v) &= \frac{d}{d+|\rho_1|}[(v-p)T_1]^2 \\ &+ \frac{d}{d+|\rho_2|}[(v-p)T_2]^2 + [(v-p)N]^2 \end{aligned} \tag{1}$$

This formula is called the squared distance (SD) error function. Pottmann et al. applied SDM successfully to solve a series of geometric optimization problems, including the problem of fitting B-spline curves and surfaces to some smooth target shapes [31, 32]. Yang et al. [41] studied how to define initial shapes and adjust the number of control points when using the SDM for B-spline curve approximation. The ellipsoid in Figure 1 shows an iso-distance surface defined by the SD error function for surface fitting.

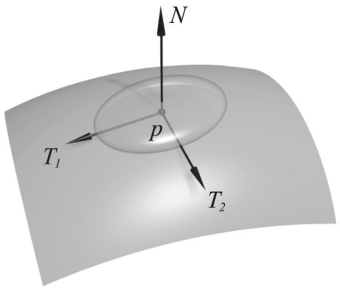

Figure 1. An iso-surface of the SD error function, with a local coordinate frame at the point p.

For comparison, the (squared) point distance (PD) error function is defined by

$$F^+(v) = \|v - p\|_2^2. \tag{2}$$

Accordingly, optimization schemes using the PD error function will be called PDM. Intuitively, unlike the PD error function, the SD error function takes the local geometry of the target surface Γ into account. In fact, SDM is a Newton method with proper modification of possibly indefinite Hessian; the proof of this fact is given in the Appendix. Hence, we expect better convergence behavior using SDM than using PDM, which is known to have only linear convergence.

2.2. Loop's Surface

Loop [21] proposed a subdivision scheme for triangulated control polyhedra. For one level of subdivision, a triangle is split into four triangles by adding on each edge a new vertex X given by

$$X = \frac{3}{8}(X_a + X_b) + \frac{1}{8}(X_c + X_d),$$

where X_a, X_b are the two vertices of the edge, and X_c, X_d are the other vertices of the two triangles that are incident to the edge X_aX_b. Then the original vertices are modified by the following rule:

$$\hat{X} = (1 - k\beta)X + \beta \sum_{i=1}^{k} X_i,$$

where k is the degree of the vertex X and X_i's are the neighboring points of X, $\beta = 3/16$ if $k = 3$, and

$$\beta = \frac{1}{k}\left[\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4}\cos^2\left(\frac{2\pi}{k}\right)\right)^2\right]$$

if $k > 3$.

3. Main Steps

Our SDM method has the following steps:

1. *Normalization of the target shape*:
   The target shape is normalized by scaling so that all data points fall in the cube $[0, 1]^3$. This is to make the values of the energy terms unit-less.
2. *Pre-computation of distance and curvature*:
   To quickly compute foot points for setting up error functions, the distance field of the target surface Γ and curvatures of Γ at all data points are computed in a preprocessing step.
3. *Initial mesh specification*:
   To obtain an initial control mesh, we build an octree for the target shape and then generate the initial mesh from the octree cells using the Marching Cubes method [22]. See Section 4.3.
4. *Sampling points on the fitting surface*:
   A set of sample points are generated on the limit surface for setting up the error functions and error evaluation. These sample points are generated using the approach devised by Stam [35, 36].
5. *Optimization*:
   Let $v_{k,0}$, $k = 1, 2, \ldots, N$, be sample points on the fitting subdivision surface. Let p_k be the foot point on Γ of the sample point $v_{k,0}$. Denote $d_k = \|v_{k,0} - p_k\|_2$. Let $\rho_{k,1}$ and $\rho_{k,2}$ be the principal curvature radii of Γ at p_k. Let v_k be a variable point in the neighborhood of $v_{k,0}$. We call v_k a variable sample point associated with v_{k0}. Then the SDM error function is defined as

$$F^+ = \frac{1}{N}\sum_{k=1}^{N} F^+_{d_k}(v_k) + \lambda F_s,$$

and

$$\begin{aligned} F^+_{d_k}(v_k) &= \frac{d_k}{d_k + |\rho_{k,1}|}[(v_k - p_k)T_{k1}]^2 \\ &+ \frac{d_k}{d_k + |\rho_{k,2}|}[(v_k - p_k)T_{k2}]^2 \\ &+ [(v_k - p_k)N_k]^2, \end{aligned}$$

where $F^+_{d_k}(\cdot)$ is the quadratic approximation for the squared distance function from a variable point, T_{k1} and T_{k2} are the unit vectors of the principal curvature directions, N_k is the unit normal vector of Γ at p_k, F_s is a smoothing term and λ is a constant.

Since the variable sample point v_k is a linear combination of the control points P_i, the function F^+ is a quadratic function of the P_i. Thus the updated control points P_i can be computed by solving a linear system of equations. Since each variable sample point v_k is only influenced by a small number of control points, the matrix for the resulting linear system of equations is sparse. We use a conjugate gradient (CG) method that exploits the sparsity of the coefficient matrix to solve the linear system of equations. The conjugate gradient solver is terminated if the relative error improvement is less than 10^{-7} or the number of iterations reaches 200.

6. *Fitting error evaluation*:
   After the control points have been updated, the maximum and average fitting errors are evaluated. The maximum approximation error E_m is defined by the maximum of the distances of all the sample points $v_{k,0}$ on the fitting surface S to the target shape Γ, i.e.

$$E_m = \max_k\{||p_k - v_{k,0}||_2\}.$$

   The average error E_a, i.e. l_2 error , is defined as

$$E_a = \left[\frac{1}{N}\sum_k ||p_k - v_{k,0}||_2^2\right]^{\frac{1}{2}}.$$

7 *Refinement and termination*: The fitting process is stopped if a pre-specified error threshold is satisfied. Otherwise, we repeat steps 4 to 6 until the result is satisfactory. Whenever the fitting error stops getting improved and stays above the error threshold, new control points are inserted in regions of large errors to provide greater degree of freedom for better fitting.

4. Implementation Issues

4.1. Curvature Pre-computation

The curvature information of the target shape can be pre-computed and stored for later use. There are existing methods, such as [8], for estimating curvatures from point clouds. We employ the following simple method for our purpose. For a given target point Q_i, its neighboring points $Q_{n(i,j)}$ are identified. In our examples, the neighborhood size is set to be 0.075, which is chosen based on data sampling density of our test examples. Let O_i denote the centroid of the neighboring points. Then, the principal curvature directions and the normal direction are computed as

the eigenvectors of the covariance matrix CV given by

$$CV = \sum_j (Q_{n(i,j)} - O_i)(Q_{n(i,j)} - O_i)^T.$$

After that, we fit a paraboloid $z = k_1x^2 + k_2y^2$ to the points $Q_{n(i,j)}$ in the local coordinate system formed with the principal curvature directions and the normal direction at Q_i. With the coefficients k_1 and k_2 determined, the principal curvatures are simply $2k_1$ and $2k_2$.

4.2. Distance Field

To obtain the distance d required by the SD error function, we compute the distance field of the target shape in preprocessing using the Fast Marching method [33] in a uniform grid with the grid spacing 0.02. During the optimization process, the distance for a sample point $v_{k,0}$ is computed by trilinear interpolation from the stored values in its neighboring grid points. Similar pre-computation techniques of the distance field have been used in [18, 41].

4.3. Initial Mesh Generation

To start with the surface reconstruction process, we adopt a simple scheme to generate an initial control mesh, since the construction of initial mesh is not the focus in this paper. We first construct an octree partition of the point cloud. Then, a mesh is obtained using the Marching Cubes algorithm [22]. The cell size of the octree is small enough so that the resulting mesh has the same topology as the target point cloud. To capture small features of a target shape, we apply the Marching Cubes algorithm with a sufficiently small cell size to obtain a dense initial mesh before simplifying the mesh adaptively to reduce the total number of triangles [7].

Another approach is to model small details by adding a displacement map over a smooth surface [14]. The visual output from this approach is impressive but it does not meet our goal of computing a complete surface representation for a point cloud.

4.4. Energy Term

We use an energy term F_s to increase the smoothness of the surface and discourage self-intersection. Following [23], the energy term is defined by

$$F_s \quad = \quad \frac{1}{n}\sum_{i=1}^{n} V(P_i)^T V(P_i),$$

where P_i, $i = 1, 2, \ldots, n$, are the control points and $V(\cdot)$ is a discretized version of Laplacian.

The coefficient λ for F_s needs to be determined carefully. If λ is too small, the term will have little influence and self-intersection may occur. On the other hand, if λ is too large, the fitting result may not be acceptable since the fitting surface will be too rigid to give small fitting errors. In our experiments, the initial value for λ is set to be 0.001. As the optimization proceeds, λ is reduced gradually at different rates for different target shapes.

4.5. Local Subdivision

When the result of the approximation is not as good as expected due to the lack of the degree of freedom provided by the current control points, new control points need to be inserted. Instead of applying the subdivision rule to all the triangles, we perform subdivisions only to the triangles that have large errors. This is referred to as *local subdivision*.

During the fitting process, the error between the subdivision surface and the target shape is measured so that regions with relatively large fitting errors are identified. The faces in these regions are then subdivided in a 1-to-4 manner (see Figure 2). To avoid undesirable T-vertices, the neighboring triangles are split, following the Red-Green Splitting scheme [3].

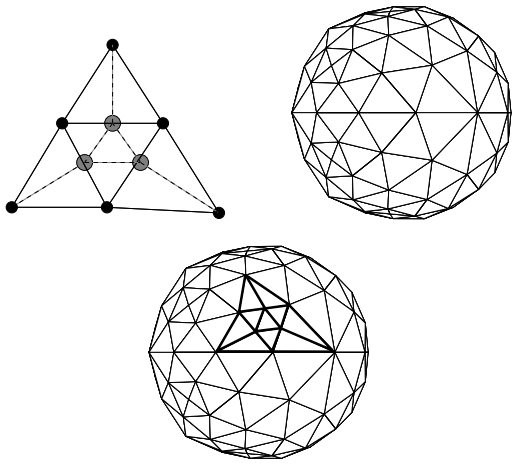

Figure 2. A triangle is split into four triangles. Neighboring triangles are also split.

After local subdivision, if the fitting error in most other regions are already acceptable, only the newly added points and their neighboring vertices are treated as variables and optimized. This saves computation time by avoiding solving a much larger linear system of equations.

4.6. Local Modification for the Connectivity of the Control Points

The connectivity of the control points can affect the fitting result. Since there are a large number of combinations of connectivity for the same set of control points and it is impractical to enumerate and test all the combinations, we conduct an edge flip test to edges in the regions that have large errors (see Figure 3). An edge flip is accepted if it results in a reduced error, following the strategy in [10].

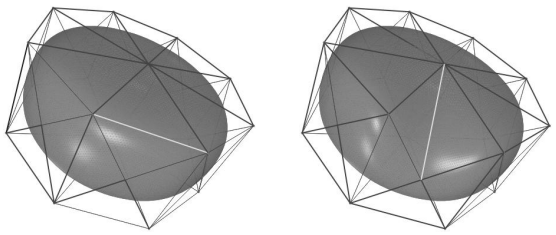

Figure 3. An edge flip of a control mesh and its effect on the limit surface.

5. Results

In this section, we present experimental results to compare the convergence behaviors of SDM and PDM, and give time and error statistics on some examples computed by SDM. All experiments were conducted on a PC with Intel Xeon 2.66 GHz CPU and 2.00 GB RAM. The models are scaled into the unit cube before surface fitting.

5.1. Comparisons of SDM and PDM

We use two data sets, the ellipsoid in Figure 5 and the ball joint in Figure 8, for comparing SDM and PDM. The code of PDM is adapted from that of SDM by replacing the SD error function (1) by the PD error function (2). Note that the distance field pre-computation is still needed for PDM, but the curvature pre-computation is not. The same initial control mesh is used by SDM and PDM for each of the two data sets. Figures 4 and 6 show the error curves of SDM and PDM for the two examples. Clearly, SDM takes far fewer iterations to attain an acceptable local minimum than PDM does. For the ellipsoid example, SDM converges quickly close to the minimum in less than 10 iterations while PDM converges to a similar value using about 150 iterations. Figure 5 shows the ellipsoid data, the initial mesh, and the optimized mesh computed by SDM; the optimized mesh computed by PDM is similar and so is not shown.

Regarding the computational efficiency, SDM needs much longer time than PDM on pre-computation, since curvature pre-computation required by SDM is unnecessary for PDM. Our experiments show that the time used by SDM on iterative optimization is about 30% to 50% more than PDM because SDM needs extra time to set up the more complex SDM error function.

5.2. More Examples

We now present three more examples to give the timing and error statistics of SDM. Figures 8, 7 and 9 show the data sets for a head, a ball joint, and an Armadrillo (*data source: http://www.cyberware.com*). The figures show the unorganized point data rendered using a point rendering technique, the initial meshes, the optimized control meshes, and the final subdivision surfaces with color error coding. Blue, green, yellow and red represent errors in the ranges $[0, 0.005)$, $[0.005, 0.01)$, $[0.01, 0.015)$ and $[0.015, \infty)$, respectively.

	head	ball joint	Armadrillo
# of data points	134345	137062	172974
curvature	303.33s	361.05s	550.27s
distance field	1.29s	0.71s	1.42s

Table 1. Time statistics for computing the curvature and the distance field. Time is measured in seconds.

	head	ball joint	Armadrillo
maximum error	0.0111	0.0090	0.0212
average error	0.0023	0.0024	0.0020
# of iterations	12	22	12
# of control points	4767	489	9602
total time	259.98s	30.19s	623.56s

Table 2. Time and error statistics for the examples. Time is measured in seconds. *# of control points* **refers to the number of vertices in the final optimized control mesh. The total time does not include the time on pre-computation.**

Table 1 gives the timing data for the preprocessing steps. Table 2 shows time and error statistics of SDM in computing the three examples. Table 3 shows the breakdown of the time used in the optimization step. We see in Table 3 that the

	setting up equations	solving equations	error evaluation
head	207.20s	23.48s	29.30s
ball joint	14.89s	5.75s	9.55s
Armadrillo	506.56s	51.75s	65.25s

Table 3. Time distribution for different tasks. Time is measured in seconds.

time for generating entries of the matrix of the linear equations is substantial when compared with other parts. Note that the number of data points affects only the time used for the preprocessing steps, but does not affect the time used in the optimization step, which is mainly determined by the number of control points.

6. Conclusion

We have presented the SDM method for fitting a subdivision surface to unorganized data points. Our experiments show that SDM converges much faster than PDM. Theoretically, we have proved that SDM is derived from the standard Newton method with necessary modifications to make the Hessian positive definite. For details, refer to the Appendix. Since the Newton method has quadratic convergence, this result explains the superior convergence behavior of SDM over PDM, which is known to have linear convergence.

A number of problems call for further research. First, a more effective method needs to be devised to determine the coefficient λ of the smoothness term. Second, features like edges and corners in data sets need to be detected in order to make the subdivision surfaces preserve the detected features. Third, we would like to apply the SDM approach to fitting surfaces to noisy point clouds, for which it is difficult to have accurate curvature estimation required by the method presented here. Finally, the SDM method presented here, like PDM, is still a local optimization technique, and therefore needs the provision of a good initial fitting surface. In this regard, there is a need to design an active subdivision scheme that allows the fitting surface to evolve from a simple initial shape to converge in a global manner to a given target surface, as done in other active contour approaches, such as the snake method [4, 17] and the level-set method [27, 33, 39, 42, 43].

References

[1] A. Adamson and M. Alexa. Approximating and Intersecting Surfaces from Points. In *Proceedings of the Eurographics Symposium on Geometry Processing*, pages 245–254, 2003.

[2] A. Adamson and M. Alexa. Approximating Bounded, Non-orientable Surfaces from Points. In *Shape Modeling International 2004*, pages 243–252, 2004.

[3] R. Bank, A. Sherman, and A. Weiser. Refinement Algorithm and Data Structures for Regular Local Mesh Refinement. In *Scientific Computing*, pages 3–17, 1983.

[4] A. Blake and M. Isard. *Active Contours*. Springer, 1998.

[5] Y. Duan and H. Qin. Extracting Boundary Surface of Arbitrary Topology from Volumetric Datasets. In *Volume Graphics Workshop 2001*, pages 149–158, 2001.

[6] Y. Duan and H. Qin. Intelligent Balloon: A Subdivision-based Deformable Model for Surface Reconstruction of Arbitrary Unknown Topology. In *Proceedings of the Sixth ACM Symposium on Solid Modeling and Applications*, pages 47–58, 2001.

[7] M. Garland and P. Heckbert. Surface Simplification using Quadric Error Metrics. In *Proc. SIGGRAPH '97*, pages 209–216, 1997.

[8] J. Goldfeather and V. Interrante. A Novel Cubic-Order Algorithm for Approximating Principal Direction Vectors. *ACM Transactions on Graphics*, 23(1):45–63, 2004.

[9] M. Halstead, M. Kass, and T. DeRose. Efficient, Fair Interpolation using Catmull-Clark Surfaces. In *Proc. SIGGRAPH '93*, pages 35–44, 1993.

[10] H. Hoppe. *Surface Reconstruction from Unorganized Points*. PhD thesis, the University of Washington, 1994.

[11] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface Reconstruction from Unorganized Points. In *Proc. SIGGRAPH '92*, pages 71–78, 1992.

[12] I. Ivrissimtzis, W.-K. Jeong, and H.-P. Seidel. Neural Meshes: Statistical Learning Methods in Surface Reconstruction. Technical Report MPI-1-2003-4-007, Max-Planck-Institut für Informatik, 2003.

[13] W.-K. Jeong, I. Ivrissimtzis, and H.-P. Seidel. Neural Meshes: Statistical Learning Based on Normals. In *Pacific Graphics 2003*, pages 404–408, 2003.

[14] W.-K. Jeong and C.-H. Kim. Direct Reconstruction of a Displaced Subdivision Surface from Unorganized Points. *Graphical Models*, 64(2):78–93, 2002.

[15] B. Jüttler and A. Felis. Least-Squares Fitting of Algebraic Spline Surfaces. *Advances in Computational Mathematics*, 17:135–152, 2002.

[16] T. Kanai. MeshToSS: Converting Subdivision Surfaces from Dense Meshes. In *Proceedings of the Vision Modeling and Visualization Conference 2001*, pages 325–332, 2001.

[17] M. Kass, A. Witkins, and D. Terzopoulos. Snakes – Active Contour Models. *International Journal of Computer Vision*, 1(4):321–331, 1987.

[18] S. Leopoldseder, H. Pottmann, and H. K. Zhao. The d^2 Tree: A Hierarchical Representation of the Squared Distance Field. Technical Report 101, Institute of Geometry, Vienna University of Technology, 2003.

[19] D. Liersch, A. Sovakar, and L. Kobbelt. Parameter Reduction and Automatic Generation of Active Shape Models. *Workshop Bildverarbeitung fur die Medizin*, 2003.

[20] N. Litke, A. Levin, and P. Schröder. Fitting Subdivision Surfaces. In *Proc. IEEE Visualization 2001*, pages 319–324, 2001.

[21] C. Loop. Smooth Subdivision Surfaces based on Triangles. Master's thesis, Department of Mathematics, University of Utah, 1987.

[22] W. Lorensen and H. Cline. Marching Cubes: A High Resolution 3-D Surface Construction Algorithm. *Computer Graphics*, 21:163–169, 1987.

[23] C. Lürig, L. Kobbelt, and T. Ertl. Hierarchical Solutions for the Deformable Surface Problem in Visualization. *Graphical Models*, 62:2–18, 2000.

[24] W. Ma and J. P. Kruth. Parameterization of Randomly Measured Points for Least Squares Fitting of B-spline Curves and Surfaces. *Computer-Aided Design*, 27(9):663–675, 1995.

[25] W. Ma and N. Zhao. Catmull-Clark Surface Fitting for Reverse Engineering Applications. In *Proc. GMP 2000*, pages 274–284, 2000.

[26] W. Ma and N. Zhao. Smooth Multiple B-spline Surface Fitting with Catmull-Clark Subdivision Surfaces for Extraordinary Corner Patches. *The Visual Computer*, 18:415–436, 2002.

[27] R. Malladi, J. A. Sethian, and B. C. Vemuri. Shape Modeling with Front Propagation: A Level Set Approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):158–175, 1995.

[28] M. Marinov and L. Kobbelt. Optimization Techniques for Approximation with Subdivision Surfaces. In *ACM Symposium on Solid Modeling and Applications*, pages 1–10, 2004.

[29] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, and H.-P. Seidel. Multi-level Partition of Unity Implicits. In *Proc. SIGGRAPH '03*, pages 463–470, 2003.

[30] H. Pottmann and M. Hofer. Geometry of the Squared Distance Function to Curves and Surfaces. *Visualization and Mathematics III, Springer*, pages 223–244, 2003.

[31] H. Pottmann and S. Leopoldseder. A Concept for Parametric Surface Fitting which avoids the Parametrization Problem. *Computer Aided Geometric Design*, 20:343–362, 2003.

[32] H. Pottmann, S. Leopoldseder, and M. Hofer. Approximation with Active B-spline Curves and Surfaces. In *Proc. Pacific Graphics 02*, pages 8–25, 2002.

[33] J. A. Sethian. *Level Set Methods and Fast Marching Methods Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. Cambridge University Press, 1999.

[34] T. Speer, M. Kuppe, and J. Hoschek. Global Reparametrization for Curve Approximation. *Computer Aided Geometric Design*, 15:869–877, 1998.

[35] J. Stam. Evaluation of Loop Subdivision Surfaces. In *SIGGRAPH'98 CDROM Proceedings*, 1998.

[36] J. Stam. Exact Evaluation of Catmull-Clark Subdivision Surfaces at Arbitrary Parameter Values. In *Proc. SIGGRAPH'98*, pages 395–404, 1998.

[37] H. Suzuki, S. Takeuchi, and T. Kanai. Subdivision Surface Fitting to a Range of Points. In *The Seventh Pacific Conference on Computer Graphics and Applications*, pages 158–167, 1999.

[38] I. Tobor, P. Reuter, and C. Schlick. Efficient Reconstruction of Large Scattered Geometric Datasets using the Partition of Unity and Radial Basis Functions. *Journal of WSCG 2004*, 12(3):467–474, 2004.

[39] R. Whitaker. A Level-Set Approach to 3D Reconstruction from Range Data. *The International Journal of Computer Vision*, 29(3):203–231, 1998.

[40] Z. Wood, M. Desbrun, P. Schröder, and D. Breen. Semi-Regular Mesh Extraction from Volumes. In *Proc. IEEE Visualization*, pages 275–282, 2000.

[41] H. P. Yang, W. Wang, and J. G. Sun. Control Point Adjustment for B-spline Curve Approximation. *Computer-Aided Design*, 36:639–652, 2004.

[42] H. K. Zhao, T. Chan, and S. Osher. A Variational Level Set Approach to Multiphase Motion. *Journal of Computational Physics*, 127:179–195, 1996.

[43] H. K. Zhao, S. Osher, B. Merriman, and M. Kang. Implicit and Non-parametric Shape Reconstruction from Unorganized Data using a Variational Level Set Method. *Computer Vision and Image Understanding*, 80:295–319, 2000.

[44] D. Zorin. *Subdivision and Multiresolution Surface Representations*. PhD thesis, Caltech, Pasadena, Calif, 1997.

[45] D. Zorin, P. Schröder, A. DeRose, J. Stam, L. Kobbelt, and J. Warren. Subdivision for Modeling and Simulation. In *SIGGRAPH '99 Course Notes*, 1999.

Appendix: SDM is a Newton method

We now show that SDM is a Newton method. A fitting subdivision surface $\mathcal{P}$, which is defined by the control points $P_i, i = 1, 2, \ldots, n$, and a target shape $\mathcal{T}$ which is twice differentiable are given. $P(u_k, v_k), k = 1, 2, \ldots, N$, which are linear combinations of the control points P_i, are sample points on $\mathcal{P}$ and $T(s_k, t_k)$ are their corresponding foot points on $\mathcal{T}$, where u_k, v_k and s_k, t_k are the surface parameters of the sample points and the foot points respectively.

Consider that the control points P_i of $\mathcal{P}$ are modified with a displacement vector $\mathcal{D} = (D_1, D_2, \ldots, D_n)^T$. Then the sample points on the modified surface are $P(\mathcal{D}; u_k, v_k)$ and their corresponding foot points on $\mathcal{T}$ are $T(s(k;\mathcal{D}), t(k;\mathcal{D}))$. Denote $s(k;\mathcal{D})$, $t(k;\mathcal{D})$ by s_k, t_k, $P(\mathcal{D}; u_k, v_k)$ by P and $T(s(k;\mathcal{D}), t(k;\mathcal{D}))$ by T.

In SDM, $\mathcal{D}$ needs to be computed such that the goal function

$$f = \frac{1}{2}\sum_{k=1}^{N}(P-T)^T(P-T) \tag{3}$$

is minimized, and subjected to the constraints:

$$\frac{\partial T^T}{\partial s_k}(P-T) = 0$$

and

$$\frac{\partial T^T}{\partial t_k}(P-T) = 0.$$

Note that the constraints are added since they are the necessary conditions for a minimum of f, which can easily be verified by differentiation.

Furthermore, for clarity, $P - T$ is denoted by E_k and $\frac{1}{2}E_k^T E_k$ is denoted by G_k afterwards.

Computing the gradient vector, we obtain

$$\begin{aligned}\bigtriangledown_{\mathcal{D}} G_k &= E_k^T \cdot \left(\frac{\partial P}{\partial D} - \frac{\partial T}{\partial s_k}\bigtriangledown_{\mathcal{D}} s_k - \frac{\partial T}{\partial t_k}\bigtriangledown_{\mathcal{D}} t_k\right) \\ &= E_k^T \frac{\partial P}{\partial D} \text{ (due to the constraints)}\end{aligned}$$

Note that $E_k^T \frac{\partial^2 P}{\partial D^2} = 0$ since P can be expressed as a linear combination of $\mathcal{D}$. We then compute the derivative of $\bigtriangledown_{\mathcal{D}} G_k$ which yields the Hessian

$$\bigtriangledown_{\mathcal{D}}^2 G_k = \left(\frac{\partial P^T}{\partial D} - \bigtriangledown_{\mathcal{D}}^T s_k \frac{\partial T^T}{\partial s_k} - \bigtriangledown_{\mathcal{D}}^T t_k \frac{\partial T^T}{\partial t_k}\right)\frac{\partial P}{\partial D} \quad (4)$$

From the constraints, we have the following two equations:

$$\begin{aligned}0 &= \bigtriangledown_{\mathcal{D}}\left(\frac{\partial T^T}{\partial s_k}(P-T)\right) \\ &= \frac{\partial^2 T^T}{\partial s_k^2}(P-T)\bigtriangledown_{\mathcal{D}} s_k + \frac{\partial^2 T^T}{\partial t_k \partial s_k}(P-T)\bigtriangledown_{\mathcal{D}} t_k + \\ &\quad \frac{\partial T^T}{\partial s_k}\left(\frac{\partial P}{\partial D} - \frac{\partial T}{\partial s_k}\bigtriangledown_{\mathcal{D}} s_k - \frac{\partial T}{\partial t_k}\bigtriangledown_{\mathcal{D}} t_k\right)\end{aligned}$$

and

$$\begin{aligned}0 &= \bigtriangledown_{\mathcal{D}}\left(\frac{\partial T^T}{\partial t_k}(P-T)\right) \\ &= \frac{\partial^2 T^T}{\partial t_k^2}(P-T)\bigtriangledown_{\mathcal{D}} t_k + \frac{\partial^2 T^T}{\partial s_k \partial t_k}(P-T)\bigtriangledown_{\mathcal{D}} s_k + \\ &\quad \frac{\partial T^T}{\partial t_k}\left(\frac{\partial P}{\partial D} - \frac{\partial T}{\partial t_k}\bigtriangledown_{\mathcal{D}} t_k - \frac{\partial T}{\partial s_k}\bigtriangledown_{\mathcal{D}} s_k\right)\end{aligned}$$

Without loss of generality, suppose (s_k, t_k) to be a local regular parameterization with s_k and t_k along the principal directions of the target surface at T such that $\frac{\partial T^T}{\partial s_k}\frac{\partial T}{\partial t_k} = \frac{\partial T^T}{\partial t_k}\frac{\partial T}{\partial s_k} = 0$. It follows that

$$\begin{aligned}0 &= \frac{\partial^2 T^T}{\partial s_k^2}(P-T)\bigtriangledown_{\mathcal{D}} s_k + \\ &\quad \frac{\partial T^T}{\partial s_k}\left(\frac{\partial P}{\partial D} - \frac{\partial T}{\partial s_k}\bigtriangledown_{\mathcal{D}} s_k - \frac{\partial T}{\partial t_k}\bigtriangledown_{\mathcal{D}} t_k\right) \\ &= \left(\frac{\partial^2 T^T}{\partial s_k^2}(P-T) - \frac{\partial T^T}{\partial s_k}\frac{\partial T}{\partial s_k}\right)\bigtriangledown_{\mathcal{D}} s_k + \frac{\partial T^T}{\partial s_k}\frac{\partial P}{\partial D}\end{aligned}$$

and

$$\begin{aligned}0 &= \frac{\partial^2 T^T}{\partial t_k^2}(P-T)\bigtriangledown_{\mathcal{D}} t_k + \\ &\quad \frac{\partial T^T}{\partial t_k}\left(\frac{\partial P}{\partial D} - \frac{\partial T}{\partial t_k}\bigtriangledown_{\mathcal{D}} t_k - \frac{\partial T}{\partial s_k}\bigtriangledown_{\mathcal{D}} s_k\right) \\ &= \left(\frac{\partial^2 T^T}{\partial t_k^2}(P-T) - \frac{\partial T^T}{\partial t_k}\frac{\partial T}{\partial t_k}\right)\bigtriangledown_{\mathcal{D}} t_k + \frac{\partial T^T}{\partial t_k}\frac{\partial P}{\partial D}\end{aligned}$$

From the above equations, we get

$$\begin{aligned}\bigtriangledown_{\mathcal{D}} s_k &= -\frac{\frac{\partial T^T}{\partial s_k}\frac{\partial P}{\partial D}}{\frac{\partial^2 T^T}{\partial s_k^2}(P-T) - \frac{\partial T^T}{\partial s_k}\frac{\partial T}{\partial s_k}} \\ \bigtriangledown_{\mathcal{D}} t_k &= -\frac{\frac{\partial T^T}{\partial t_k}\frac{\partial P}{\partial D}}{\frac{\partial^2 T^T}{\partial t_k^2}(P-T) - \frac{\partial T^T}{\partial t_k}\frac{\partial T}{\partial t_k}}\end{aligned}$$

Now, we substitute $\frac{\partial T}{\partial s_k} = T_1$, $\frac{\partial T}{\partial t_k} = T_2$, $\frac{\partial^2 T}{\partial s_k^2} = \kappa_1 N$, $\frac{\partial^2 T}{\partial t_k^2} = \kappa_2 N$, $\|P - T\|_2 = d$, where κ_1, κ_2 are the principal curvatures at T, T_1, T_2 are the unit tangent vectors along the principal directions at T and N is the unit normal vector at T. After the substitutions, we have:

$$\bigtriangledown_{\mathcal{D}} s_k = -\frac{T_1^T \frac{\partial P}{\partial D}}{d\kappa_1 - 1} \quad (5)$$

$$\bigtriangledown_{\mathcal{D}} t_k = -\frac{T_2^T \frac{\partial P}{\partial D}}{d\kappa_2 - 1} \quad (6)$$

Substitute (5) and (6) into (4),

$$\begin{aligned}\bigtriangledown_{\mathcal{D}}^2 G_k &= \frac{\partial P^T}{\partial D}\frac{\partial P}{\partial D} + \frac{\frac{\partial P^T}{\partial D}T_1 T_1^T \frac{\partial P}{\partial D}}{d\kappa_1 - 1} + \\ &\quad \frac{\frac{\partial P^T}{\partial D}T_2 T_2^T \frac{\partial P}{\partial D}}{d\kappa_2 - 1} \\ &= \frac{\partial P^T}{\partial D}(1 - T_1 T_1^T - T_2 T_2^T)\frac{\partial P}{\partial D} + \\ &\quad d\kappa_1 \frac{\frac{\partial P^T}{\partial D}T_1 T_1^T \frac{\partial P}{\partial D}}{d\kappa_1 - 1} + d\kappa_1 \frac{\frac{\partial P^T}{\partial D}T_2 T_2^T \frac{\partial P}{\partial D}}{d\kappa_2 - 1} \\ &= \frac{\partial P^T}{\partial D}NN^T\frac{\partial P}{\partial D} + \frac{d}{d-\rho_1}\frac{\partial P^T}{\partial D}T_1 T_1^T\frac{\partial P}{\partial D} + \\ &\quad \frac{d}{d-\rho_2}\frac{\partial P^T}{\partial D}T_2 T_2^T\frac{\partial P}{\partial D}\end{aligned}$$

where $\rho_1 = \frac{1}{\kappa_1}, \rho_2 = \frac{1}{\kappa_2}$.

We get the Hessian $\bigtriangledown^2 f = \sum_{k=1}^{m} \bigtriangledown^2 G_k$, and now we move on to prove that

$$\sum_{k=1}^{m}\left(\frac{d}{d-\kappa_1}(E_k^T T_1)^2 + \frac{d}{d-\kappa_2}(E_k^T T_2)^2 + (E_k^T N)^2\right)$$

is a quadratic approximant of (3):

$$f(P_c) + \bigtriangledown f(P_c)\mathcal{D} + \frac{1}{2}\mathcal{D}^T(\bigtriangledown^2 f(P_c))\mathcal{D}.$$

We only need to expand one term in (3). Denote $E_k^T E_k$ by f_k, and notice that $E_k^T T_1 = E_k^T T_2 = 0$, $\bigtriangledown_{\mathcal{D}} E_k = N\frac{\partial P}{\partial D}$, P_c are the sample points for the current iteration, T_c are the foot points of P_c and $P(P_c;\mathcal{D})$ are the sample points after the control points are updated with the displacement vector $\mathcal{D}$.

$$
\begin{aligned}
& f_k(P_c) + \bigtriangledown_{\mathcal{D}} f_k(P_c)\mathcal{D} + \frac{1}{2}\mathcal{D}^T(\bigtriangledown^2_{\mathcal{D}} f_k(P_c))\mathcal{D} \\
= \;& E_k^T(P_c)E_k(P_c) + 2E_k^T \bigtriangledown_{\mathcal{D}} E_k\mathcal{D} + \\
& \mathcal{D}^T\frac{\partial P^T}{\partial D}N^T N\frac{\partial P}{\partial D}\mathcal{D} + \\
& \frac{d}{d-\rho_1}D^T\frac{\partial P^T}{\partial D}T_1^T T_1\frac{\partial P}{\partial D}\mathcal{D} + \\
& \frac{d}{d-\rho_2}D^T\frac{\partial P^T}{\partial D}T_2^T T_2\frac{\partial P}{\partial D}\mathcal{D} \\
= \;& \left(E_k^T(P_c) + \mathcal{D}^T\frac{\partial P^T}{\partial D}N^T\right)\left(E_k(P_c) + N\frac{\partial P}{\partial D}\mathcal{D}\right) + \\
& \frac{d}{d-\rho_1}\left(P(P_c;\mathcal{D}) - P_c^T + P_c^T - T_c^T\right)T_1^T \\
& T_1\left(P^T(P_c;\mathcal{D}) - P_c + P_c - T_c\right) + \\
& \frac{d}{d-\rho_2}\left(P(P_c;\mathcal{D}) - P_c^T + P_c^T - T_c^T\right)T_2^T \\
& T_2\left(P^T(P_c;\mathcal{D}) - P_c + P_c - T_c\right) \\
= \;& (E_k^T(P_c;\mathcal{D})N^T N(E_k(P_c;\mathcal{D})) + \\
& \frac{d}{d-\rho_1}E_k^T(P_c;\mathcal{D})T_1^T T_1 E_k(P_c;\mathcal{D}) + \\
& \frac{d}{d-\rho_2}E_k^T(P_c;\mathcal{D})T_2^T T_2 E_k(P_c;\mathcal{D}) \\
= \;& \text{SDM formula}
\end{aligned}
$$

Hence, SDM is indeed a Newton method with constraints. In practice, we change $d-\rho_1$ and $d-\rho_2$ to $d+|\rho_1|$ and $d+|\rho_2|$ respectively for ensuring the quadratic form is positive semi-definite.

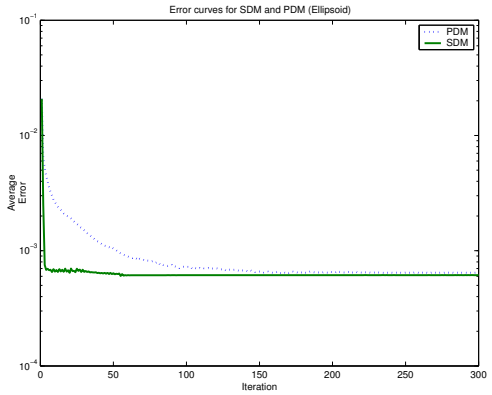

Figure 4. Error curves for SDM and PDM (Ellipsoid). No. of control points: 14.

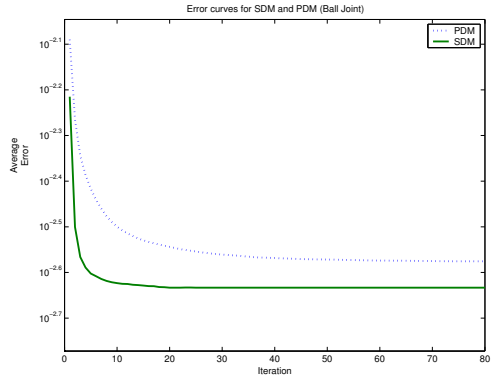

Figure 6. Error curves for SDM and PDM (Ball joint). No. of control points: 336.

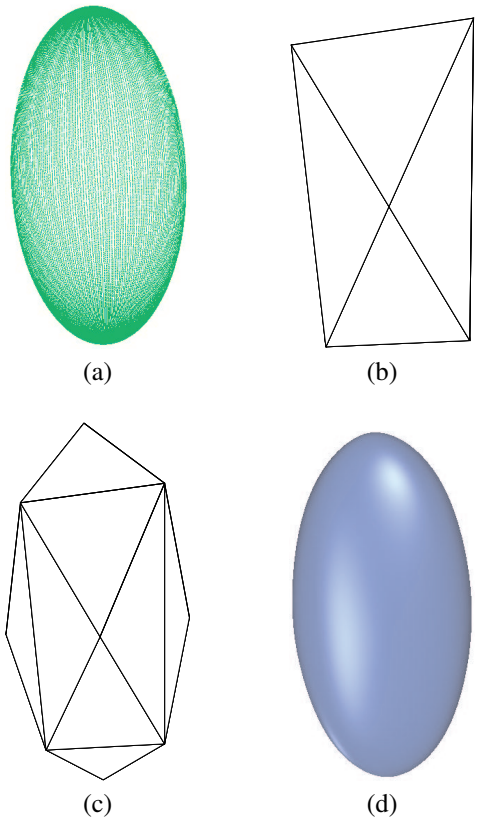

Figure 5. Ellipsoid: (a) Target point cloud. (b) Initial mesh. (c) Optimized mesh. (d) Optimized subdivision surface.

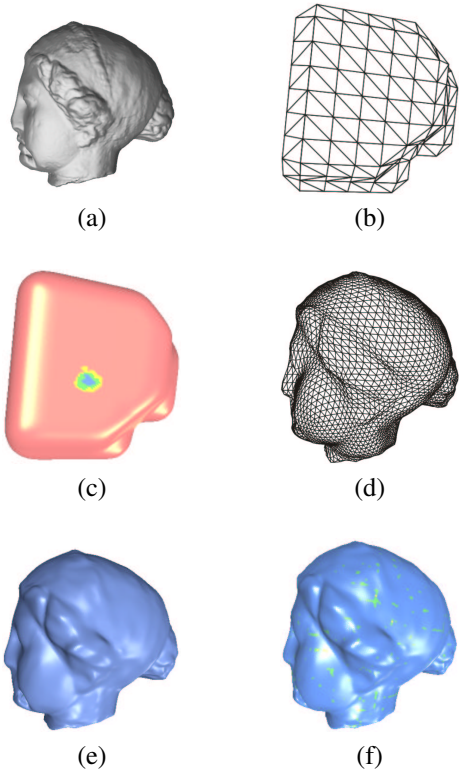

Figure 7. Head: (a) Unorganized point data. (b) Initial mesh. (c) Initial error. (d) Optimized mesh. (e) Subdivision surface. (f) Final error. No. of data points: 134345. No. of control points in the optimized mesh: 4767.

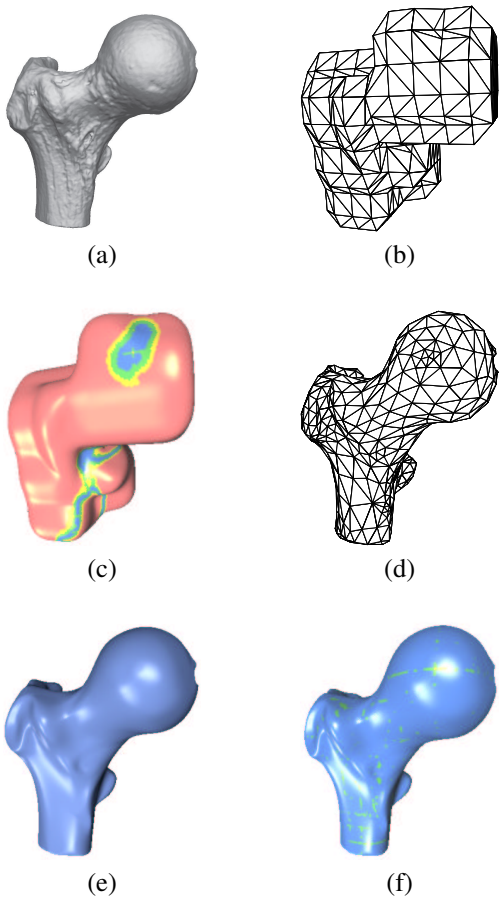

Figure 8. Ball joint: (a) Unorganized point data. (b) Initial mesh. (c) Initial error. (d) Optimized mesh. (e) Subdivision surface. (f) Final error. No. of data points: 137062. No. of control points in the optimized mesh: 489.

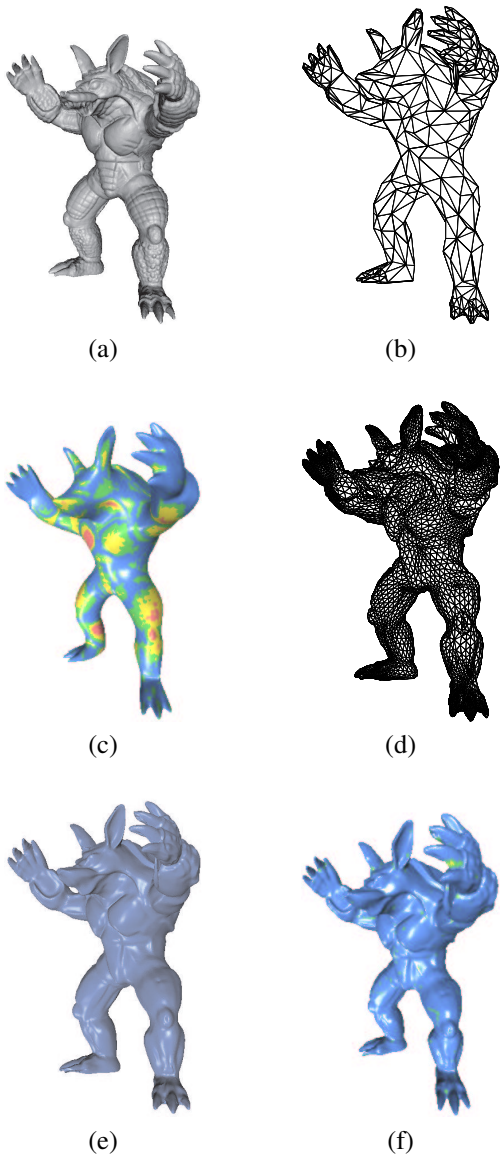

Figure 9. Armadrillo: (a) Unorganized point data. (b) Initial mesh. (c) Initial error. (d) Optimized mesh. (e) Subdivision surface. (f) Final error. No. of data points: 172974. No. of control points in the optimized mesh: 9602.