Unstructured Point Cloud Surface Denoising and Decimation Using Distance RBF K-Nearest Neighbor Kernel

Rixio Morales, Yunhong Wang, and Zhaoxiang Zhang

Beihang University, Department of Computer Science and Engineering,
XueYuan. 37, 10091 Beijing, China
rixiomorales@cse.buaa.edu.cn,
{yhwang,zxzhang}@buaa.edu.cn
http://scse.buaa.edu.cn

Abstract. In this work unstructured point clouds, resulting from 3D range acquisition are point wise-processed, using a proposed kd-tree nearest neighbor method, based in a generative data driven, local radial basis function's (RBF) support:$\phi(S, p_i(x_i, y_i, z_i))$, for the point set $S : \{p_i\}_{i\epsilon I}$, using surface statistic and a Gaussian convolution kernel, point sets are smoothed according to local surface features. As a minor contribution we also present a point cloud semi-rigid grid decimation method, based on a similar framework, using multi-core hardware, experiment results achieve comparable quality results with existing and more complex methods; time performance and results are presented for comparison.

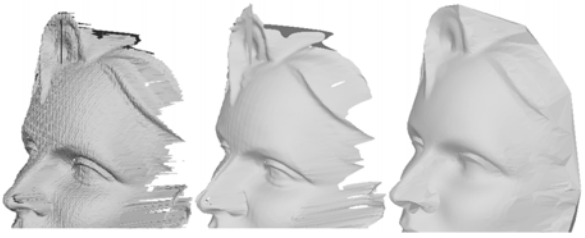
Keywords: Unstructured Point Cloud, Smoothing, Decimation, RBF, multicore.

1 Introduction

Raw unstructured 3D point clouds or point sets are a regular input for shape analysis,and suffer from lack of parametrization and data sparsity, several methods had been developed to tackle the noisy output of data acquisition sensors. Using the output models directly from the acquisition process, the sensor's noise characteristic, surface resolution, probability distribution function (pdf) and in-plane transformation parameters are not provided; instead a rough point number and the data itself, avoiding out-of-box processing without clearly analyze and preprocess the sampled surface. Complex methods should be applied to achieve denoising and rectification, like (U, V) orthogonal plane parametrization (for 2.5d range), spline approximation, constrained delaunay triangulation or tessellation, moving less square (MLS)[6] marching cubes tessellation, point ball pivoting surface generation, poisson reconstruction and others, dealing with noisy data and outliers that affect the final result and the original processing objective, requiring the implementation of different non-standard methods with user detailed interaction and in-deep knowledge of the input model.

To preprocess unstructured point clouds, we propose a simple method for surface smoothing and decimation, using nearest neighbor sampling, to locally process point

G. Qiu et al. (Eds.): PCM 2010, Part II, LNCS 6298, pp. 214–225, 2010.



a) b) c)

Fig. 1. Smoothing Results from the femme dataset a) Original b) Laplacian Smoothing c)Distance RBF K-nearest Neighbor Kernel, both using 4 iterations

sets (S). Based on the radial basis function main idea we defined $\phi(S, p_i(x_i, y_i, z_i))$, where $S : \{p_i\}_{i\epsilon I}$, as the real surface's sampled set, where $p_i \subset \Re^3$. RBF's response decreases or increases monotonically with distance from a central point, (p_i), distance, scale and radial function shape, are the modeling parameters, achieving a linear response if fixed. The method had been used for fitting samples in a curve for $\Re^2$ or a surface plane in $\Re^3$, function interpolation, Neural Network output among others. Motivations for this work is the fact that computer graphic hardware is optimized for on-chip triangle approximation, while triangle information (or facets) are usually orders of magnitude denser than the vertex data; in recent years more researchers are dealing with the new tools for mainstream implementation of standard shading language, mutlicore CPU and GPU, forming an interesting matter to work with point sets [14], whenever the nearest neighbors sampling resolution, captures the real surface features, point set methods are relative to polygon based counterparts.

Previous Works: Recovering the correct manifold from sampled surfaces, presents an active topic in the research community. Most of the literature in point sets smoothing deals with the the fitting of a tangent plane in a smooth surface, one of the most used techniques is related to the continuity theory of (smoothness) constraints in $(S \subset \Re^3)$, defining differentiability $\forall p\epsilon S$, and with continuous partial derivatives in all orders, by this definition the Laplace energy conservation function is used

$$\Delta f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2} = 0, \tag{1}$$

as a constraint to move (p) towards its Laplacian's direction. The seminal work of Laplacian smoothing or diffusion smoothing Taubin [1], generated different approaches, like weighted Laplacian, scale depended Laplacian, this approaches preserved the final triangulation connectivity and original point set, with some disadvantages, like: shape distortion, point drifting and volume reduction. Desbrun[3] used weighted surface normals, to deal with the point drifting. Extending this approach, *taubin smoothing* was introduced in [2], using two diffusion steps, one inwards and one outwards, to approximately preserve the mesh volume with highly effective results. Hildebrandt[4] introduced anisotropic smoothing, weighting the curvature and normals, reducing the edge diffusion while preserving low frequency details. Fleishman et al.[5] propose an

iterative method for bilateral filtering, based in image processing, as a non-linear filter, weighting the sampling points based on their inner similarity, using normals and vicinity points. Other methods use normal's or quadric estimation[21] like moving less square and parametric optimization [15]. The main advantage of our proposed method, its the simplicity of the implementation with comparative results.

2 Framework Overview

In our work we take the main idea of RBF and use it in two ways: the first is related to the surface sampling resolution approximation, during the initial parameter generation process detailed in (3.1), and the second as a spatial mapping support, better explained in(3.2); the contributions overview are shown in Figure 2. During the resolution pa-

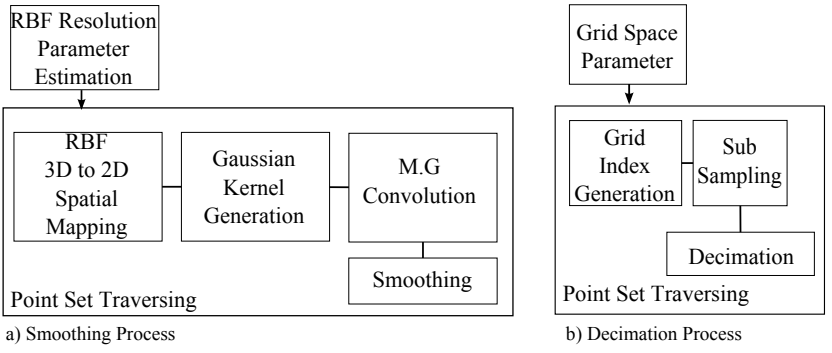


Fig. 2. Framework Overview, the *Point Set Traversing* is made in parallel

rameter estimation in Figure 2 a), an incremental radial sampling is used to find a self generative constrain, for the surface resolution, resulting in a defined point neighborhood selection function $N_i(p_i, r)^k = \{p_j : d^3(p_j, S) \leq r\}$,where d^3 is the Euclidean distance , $\|p_i - p_j\|_2$ from surface (*S*) to the point p_i. To find the best radius (*r*) for the available resolution (*k* number of samples), adapting the sampling to different density point clouds, without requiring the radial minimum distance for NN search.

Our method traverses point cloud subsets, by a parallel process, using actual multicore hardware. Per each sample, a set is generated with the neighbor support function $f_{N_i}(p_i)$, giving a point set approximation of the local surface near p_i, then, a matrix is mapped from the 3D point set to a 2D spatial $m \times m$ multidimensional symmetrical matrix, using distance based distribution,the point set components are mapped into each axis plane. The mapped point subset in the $M \times M \times 3$ matrix, is then smoothed using a convolution Gaussian kernel, $C = M \otimes G_i$, where

$$G_i(p_i, \sigma = k\prime/k) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}, \tag{2}$$

generating a local plane approximation. The final smoothing is done using a lower resolution mask, $m \times m = (3 \times 3, 5 \times 5)$. Hence smoothing the surface in separable

coordinate space, giving a new candidate point $p\prime_i = (f_{N_i}, p_i, k, \theta_{N_i k})$, where$\theta_{N_i}$ are the Gaussian K-Nearest Neighbor kernel parameters, generating a candidate C_{mxm}, matrix central point x_{ij} where $i = j, i = m/2$ from the convolution result we get $(S\prime)$.

Applying a similar framework, a semi-rigid grid decimation algorithm is presented, mapping $\Re^3 \rightarrow \Re^3$ from $x, y, z^d \rightarrow [u, v, w]^k$ where $k < d$; generating a sampling normalization for unstructured point sets, the parameter for this method is the desired grid size $T[u, v, w]$ and the *(k)* sample support points for the K-NN subsampling, the result is a new decimated point set $S\prime = (S, T, k)$; both schemes were implemented in multicore hardware, with thread-safe data-structure and processing, providing an alternative to serialized methods, while incrementing the implementation complexity, the resulting processing times are worthwhile as seen in Table:1. Results from the previously

Table 1. Processing time for registered bunny dataset (35.9K Points) from [18] in C++ Release mode 32 bits

Process	Time in ms
1.-Initial Loading and Parameters	28.2908
2.-Scale Conversion	2.28712
3.-Reload The Model	68.2858
4.- Smoothing Process and data creation	756.908

methods are suitable for further applications,like shape analysis, surface reconstruction,point set normalization, post-processing, pattern recognition, 3D modeling and other related discrete geometric applications as shown in Figure 3. Experiments for

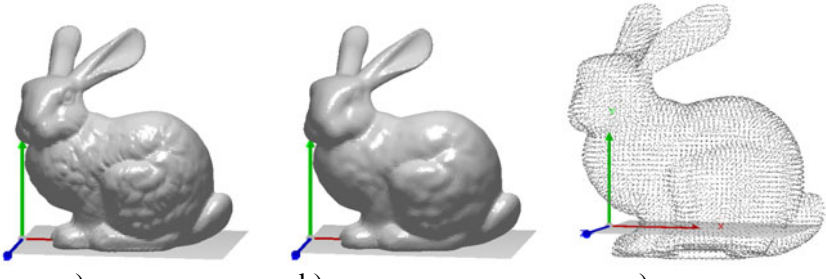


Fig. 3. Visualization results from dataset in Table.1 a) Original reconstructed b) Smoothing result c)Decimation result $(u, v, w) = [50, 50, 50]$

both contributions were performed using available standard 3D data-sets[18], reporting the visualization comparison among other methods.

3 Point Set Denoising

Inspired from the color image processing theory of spatial filtering, a gaussian convolution operation is proposed to attenuate noise and optionally remove outliers, initially

the local (K) Nearest Neighbors (K-NN), are mapped from a 3D space to a matrix $M_{m\times m\times 3}$ using a distance spatial operator, and convolved with a gaussian blur kernel in the (x, y, z) domain; below we will explain the details of the contribution.

3.1 Resolution and Initial Parameter Estimation

Nearest Neighbor Search: The design requirements for the neighbor search are crucial for the traverse performance, we use a binary search, implemented as a binary tree[10], other methods are also suitable for this step, like the GPU option in [9]. We define the term NNSearch$(x, k, r) : \{x_1, .., x_{k\prime}\}$, as the result of a function returning a maximum (k) points from point $x\epsilon\Re^3$; given a radial parameter (r), where $k\prime \leq k$, we use [11] to get a point's NN set $N_i^p(p_i) = \{x_i, \ldots, x_k\}, x_i \subset \Re^3$ of p_i. Vertex are loaded in $O(d \log d)$ time, providing a efficient search of $O(d^{1-1/3} + k)$ complexity where d is the point count, and k is the target output point count.

Generative RBF Parameter Generation: The point cloud statistical per axis information Mean and StDv (μ, σ) respectively, are recovered by direct calculation from $S\{p_i\}, i = 1, .., d$,where d is the total point set count. The radial search parameter (r) for the actual point cloud density is obtained by a K-NN Radial Basis search, using the the (S) calculated sampling mean point $\overline{p}$,

$$\overline{p}(S) = \frac{1}{n}\sum_{i=0}^{n} p_i \simeq NNSearch(p_{mean}, k, r_0), \tag{3}$$

to find its nearest neighbor density considering empty set results. The goal is to fetch an approximated radial sampling resolution, in the high density area, at $\overline{p}(S)$. An arithmetic spherical range search $r\prime_{i+1} = r_{i-1} + step$, is performed until the desired density (k) is achieved, hence generating the radial basis parameter $(r_{min} = r\prime)\epsilon\Re$ using the mean point estimated in (3), and initial radius r_0

$$r_{min} = rNNSearch(S, \overline{p}, k, r_0), . \tag{4}$$

After the parameters are generated, we traverse the point set (S), using (4) and (3) obtaining the local sampling surface in each point. In order to optimize the hardware performance, we scaled the data as remarked in Table.1, due to the affection of floating point precision operations in the nearest neighbor kd-tree distance calculations.

Remark 1. One of the restrictions for this method, is the selection of (k), in the case of non available support points $N_{k\prime p}, k\prime < k$, where $k\prime$ is the point count. This occurs in non continuous (pdf) spatial resolution data; we got the point average from the result set, and increases the result set until the desired (k); taking the difference between $c_k = k - k\prime$ as an important parameter for the further RBF gaussian kernel generation, explained in section 3.3. This method could be further improved by recursive search, from the nearest point set, with search radius $r\prime \leq r$, obtaining the related nearest points as in [12].

3.2 Radial Basis Function Spatial Mapping

To define the mapping parameter from the $\Re^3$ point set, to a $\Re^2$ spatial reference matrix, the function $F_{\Re^3 \to \Re^2}(p_i, p)$; should preserve the spatial relation between the central point p_i and the set N_i^p. For this we use the Euclidean distance, to generate a variable filter kernel matrix. The distance $d^3(x_i, x)$ from p_i among all the points in N_i^p calculated as $L_k : \{d_i, .., d_k\}$ where $d\epsilon\Re$. The new matrix $M_{\{}m \times m \times 3\}, k = m \times m$ is generated by ordering the L_k set in decreasing order, and placing the nearest points in a Clock Wise manner around p_i,generating the desired mapping relation in a multidimensional matrix $M_{i\times j\times k}$, a 3rd rank tensor, containing the spatial point relation in the final $k = 3$ dimensions. The axis components of the points N_i^p, ie. $M_{m,n,i} = T_i m \times n, i = 1, 2, 3$ where T_i are the spatially ordered vector components in each axis (x, y, z), Fig.4 a). In this step the choice of plane estimation was performed to arrange points in a disc manner, as the tangent plane estimation is not trivial, or computational efficient[13], during the complete traverse of the point set would present a computational burden.

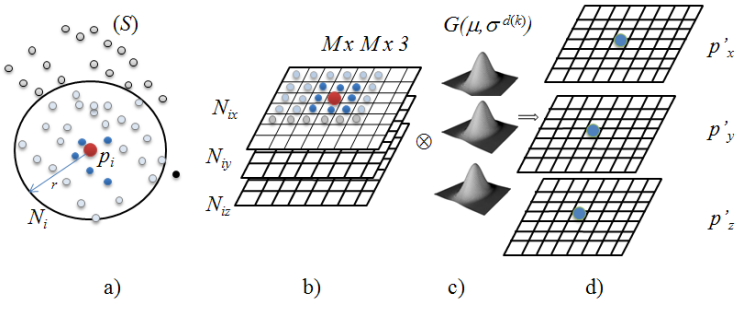


Fig. 4. Denoising a) Surface (S) Nearest Neighbor subset N_i, center point p_i,indicated r, contrasting nearest (*darker points*) shown in the 2D domain b) Distance mapping from N_i c)Gaussian Kernels $G(\mu, \sigma^{d(k)})$ for each axis N_{ij} d) New center point estimation in each component $p\prime_{x,y,z}$

3.3 Gaussian Kernel and Convolution

From the previous section mapping, the multidimensional matrix M is convolved using

$$m(x, y) * g(x, y) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} m(n, m) g(x - m, y - n). \tag{5}$$

with the kernel function G, $C = M \otimes G$ which depends on the (k), (r_{min}) and the local surface near the point p_i, using $N(\mu, \sigma^{d(k)})$,where $\mu = 0$ and $\sigma^{d(k)}$, are proportional to the ratio $R_k = \frac{k\prime}{k}$, relating the local point density, with the sampled point set. This ratio is employed to reduce the sampling error by weighting the distance relation during the convolution kernel. After gaussian coefficients are set, the discrete convolution operation is performed in the (k) planes. This method leaves space for optimization, ie. Arrange the second direction of σ, to provide a more elliptically fitted kernel, according to the real surface point, density and normals. In our method, arrange shape is near circular. Having the 3 planes spatial filtered matrices, each corresponding to a dimensional

axis, the final step is to fetch the center component of the convolution result, from each dimension, obtaining the new point $p\prime = \{c_x(x/2, y/2), c_y(x/2, y/2), c_z(x/2, y/2)\}$; Figure 4 presents the denoising method overview.

4 Point Set Decimation

The aim for this process is to simplify the point set $S \subset \Re^3$ preserving the shape's main characteristics, using a desired grid resolution input parameter $T = \{u, v, w\}, T \subset I^3$. The remaining input for this method is the point set vertex information; controlled by the generated grid resolution h_{step} at the desired sampling density k; several methods exist to achieve this decimation like Quadric Clustering,Monte Carlo Sub-Sampling among others.

4.1 Grid Index Generation

Initially the bin size h is obtained from each axis $X_{range} = max(X) - min(X)$

$$h_{step} = [x_{range}/u, y_{range}/v, z_{range}/w], \tag{6}$$

the step bin average distance $r_{bin} = \frac{\|h_{step}\|}{3}$ or (r_{min}) is calculated as the radial parameter in NNSearch(p_{min}, k, r_{bin}) radial support function. The $\Omega \subset \Re^3$ space is divided by an index grid generation, using the bin-size and the statistical initial point $p_{min} = min(x), min(y), min(z)$ as space initializer to avoid sampling points in a rigid manner, reducing the search space for hyper planes in the kd-tree search; by this means the grid is traversed iterating in $p_s = p_{min} + h_{step} \times idx)$, where $idx \subset I^3$ is a grid index driven multiplier in the 3D-Linear discrete space.

4.2 Subsampling Method

After the $L_k = NNSearch(p_s, k, r_{bin})$, set is obtained from each grid point, the candidate sample for the current space is the L set's sample mean

$$\overline{p_L} = \frac{1}{k\prime}. \sum_{i=0}^{k\prime} L_i. \tag{7}$$

Other interpolating methods like Shepards' IDW [17], were also employed but the noise induced by the irregular mesh sampling from the original point set, degenerated the final decimated grid. In the selection of the new point we expect to get a real smoothed point (approximation), according to the local surface characteristics, instead of a generative or interpolated new point; thus the simple average including the input point neighbor displayed satisfactory results, similar to vector fields, conserving the surface curvature proportional to the bin size and local surface details as shown in Figure 5, preserving holes in high density (S) as displayed in the experiment results section.

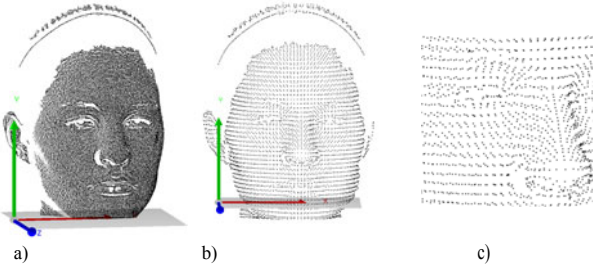


Fig. 5. Decimation Result Example a)Smoothed Point Cloud with 98.8K points. b) Resulting Grid with $u, v, w = 50$ with 6.2K points. c) Zoom in the *Eye Area* denoting the vector field like result.

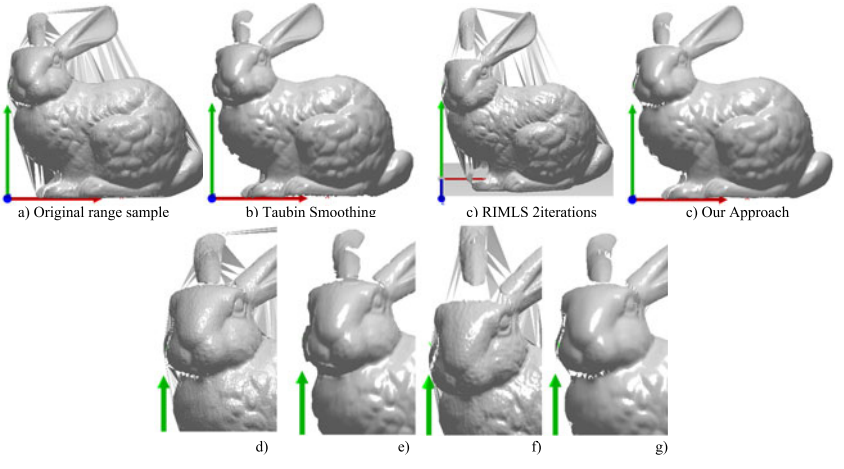


Fig. 6. Experiment 1, Smoothing method comparative results using scanner raw data a)Original data. b) Taubin Smoothing[19] Method. c) Our approach. Set d),e) and f) show the respective face details results.

5 Experimental Results and Analysis

The smoothing performance is shown in experiment 1) Figure.6, we employ real sensor data to show the filtering results, therefore using real noise, for this purpose, unregistered range images of 3d face databases and public models were processed. The experiment was performed using original scanner noisy sensor data from the Bunny dataset at 0°, with $40,276$ samples. For experiment 2) a registered point set is smoothed using different methods (Laplacian Smoothing[19], Taubin Smoothing[19], Two Step Smoothing[19], Laplacian Feature Preserving[19] and Robust Implicit Moving Less Square RIMLS[19]), results are shown in Figure.7. The parameters for our filter: $r0 = \epsilon$,$rstep = 0.1$, Mapping Tensor $M = 5 \times 5 \times 5$, and $k = 25$; by this way the only fixed

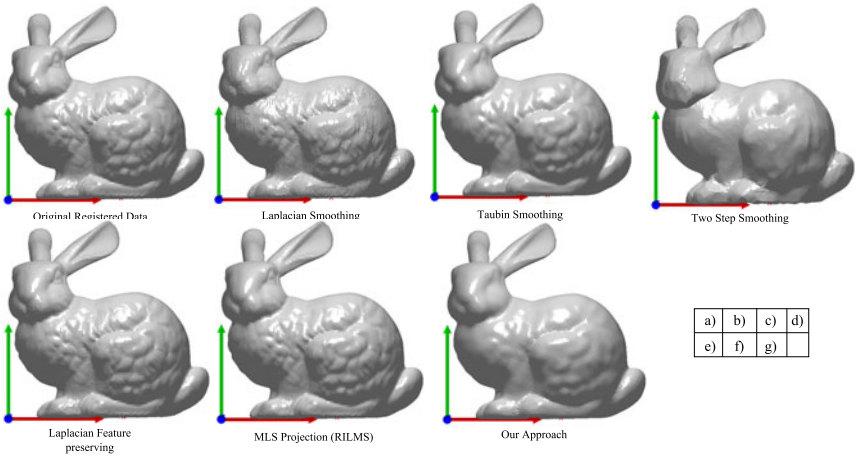


Fig. 7. Experiment 2. Comparative results using registered data from [18]. a) Original registered (pre-processed) point set and facets. b) Laplacian Smoothing from[19] , c) Taubin Smoothing Method []. d) Two Step Smoothing, e) Laplacian feature preserving, f) Robust Implicit Moving Less Square RILMS [19][15]. g) Our Approach.

parameter is the K samples to ensure enough minimal surface support, for the denoising filter, the outlier elimination (fairing) threshold $k\prime$, was set arbitrary to (3) due to possible triangle formation. For the implementation we make use of the OpenMp[16].

5.1 Experimental Results for Denoising

We achieve some interesting results using the tested datasets, specifically in high density point clouds, observing failures for particulary uneven sampled surfaces, like large objects, or artificially designed CAD models, with low resolution. Some failures are showed in Figure.5 f).

Normals smoothing, even is not pursued, is partially achieved due to the adjustment of the point cloud within the real surface, as can be seen in Figure 9. Our model results, presents significative difference on the normals reflectance, due to the gaussian smoothing result and denoising effectiveness of our approach, highlights are enhanced in the final rendering. From the results we could observe that the smoothing preserved most of high frequency features after the resulting triangulation, eliminating noise and outlier effectively as shown in experiment results Figure 7 set a) and b), Figure 1 display the reduction of sensor noise (Vertical Lines) with a balance of detail loss from our method compared with Laplacian Smoothing.

5.2 Experimental Results for Decimation

The Decimation results where partially explained in Figure 5; holes where conserved and main shape preserved, using middle to high density sampling point sets. The subsampling recovered the main shapes in most cases; decimation failure for artificial models with low point support are displayed in Figure 9 g).

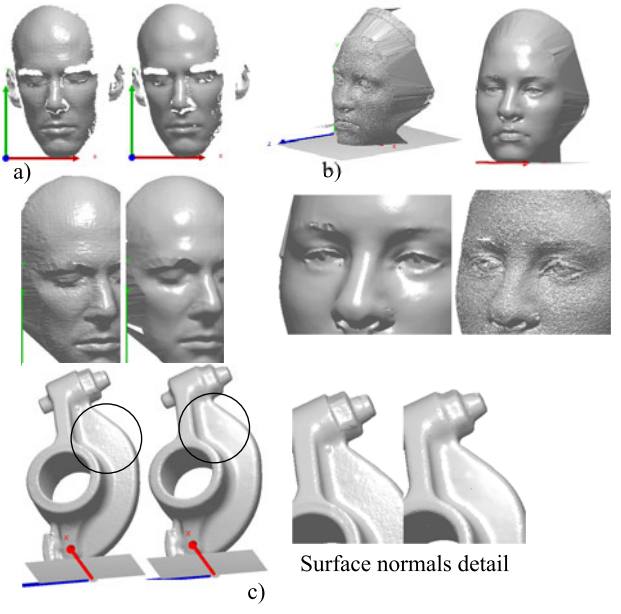


Fig. 8. Experiment 3. Smoothing method comparative results using real scanner raw data, set a)Smoothing using the Zander Set from [20]. Set b) Using a face model with high noise density. set c) Rocker Arm dataset. *Normal Reflection Details are indicated*.

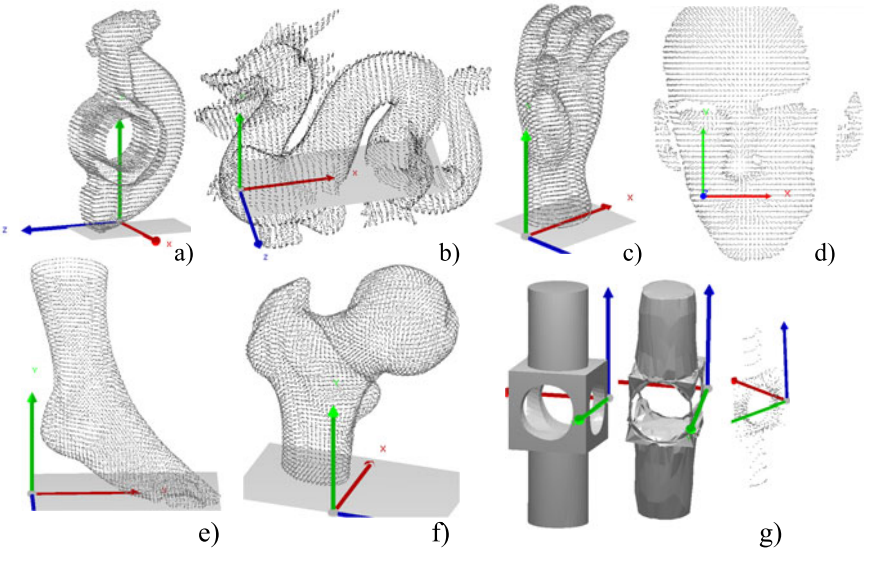


Fig. 9. Decimation experiment results using test datasets data a)Decimated Rocker Arm dataset. b) Decimated Dragon model. c) Results using Hand, d)Zander Set, e) decimated Foot and f) Ball Point set high density set. g)Mechanical Part showing the error from both techniques.

6 Conclusions

The experiments results denote the viability of the smoothing and decimation methods, for pre-processing unstructured point clouds; using generative methods and simple approximation computation, the results are inspiring to further study the details, implement the aforementioned modifications in remark 3.1, and use tangent plane approximation to modify the tensor M, taking the direction as well as the distance to derive the mapping. The approach limits will be explored to deal with a broader resolution of point sets using (oversampling); future research will use part of this contribution to normalize shapes for recognition.

Acknowledgement

This work is funded by the National Natural Science Foundation of China (No. 60873158), the National Basic Research Program of China (No. 2010CB327902), the Fundamental Research Funds for the Central Universities, and the Opening Funding of the State Key Laboratory of Virtual Reality Technology and Systems.

References

1. Taubin, G., Taubin, Y.G.: A Signal Processing Approach To Fair Surface Design. In: Proceedings of SIGGRAPH (1995)
2. Taubin, G., Taubin, Y.G.: Geometric Signal Processing on Polygonal Meshes a State of the Art Report. In: Eurographics (August 2000)
3. Desbrun, M., Meyer, M., Schreder, P., Barr, A.H.: Implicit Fairing of Irregular Meshes using Diffusion and Curvature Flow. In: Proceedings of SIGGRAPH 1999, pp. 317–324 (1999)
4. Hildebrandt, K., Polthier, K.: Anisotropic filtering of non-linear surface features. J. Computer Graphics Forum 23, 391–400 (2004)
5. Fleishman, S., Drori, I., Cohen-Or, D.: Bilateral mesh denoising. ACM Trans. Graph. 22(3), 950–953 (2003)
6. Levin, D.: The Approximation Power Of Moving Least-Squares. Mathematics of Computation 67, 1517–1531 (1998)
7. Hormann, K.: From Scattered Samples to Smooth Surfaces. In: Proc. of Geometric Modeling and Computer Graphics (2003)
8. Bradford Barber, C., Dobkin, D.P., Huhdanpaa, H.: The Quickhull algorithm for convex hulls. Acm Transactions On Mathematical Software 22(4), 469–483 (1996)
9. Qiu, D., May, S., Nüchter, A.: GPU-Accelerated Nearest Neighbor Search for 3D Registration. In: Fritz, M., Schiele, B., Piater, J.H. (eds.) ICVS 2009. LNCS, vol. 5815, pp. 194–203. Springer, Heidelberg (2009)
10. Friedman, J.H., Bentley, J.L., Finkel, R.A.: An Algorithm for Finding Best Matches in Logarithmic Expected Time. ACM Trans. Math. Softw. 3-3, 209–226 (1977)
11. Tsiombikas, J.: kdtree, A simple C library for working with KD-Trees:New BSD License, `http://code.google.com/p/kdtree/`
12. Mederos, B., Velho, L., De Figueiredo, L.H.: Robust smoothing of noisy point clouds. In: Proc. SIAM Conference on Geometric Design and Computing (2003)
13. Petitjean, S.: A survey of methods for recovering quadrics in triangle meshes. J. ACM Comput. Surv. 34, 211–262 (2002)

14. Kobbelt, L., Botsch, M.: A Survey of Point-Based Techniques in Computer Graphics. J. Computers Graphics 28, 801–814 (2004)
15. Oztireli, C., Guennebaud, G., Gross, M.: Feature Preserving Point Set Surfaces based on Non-Linear Kernel Regression. Computer Graphics Forum 28(2) (2009)
16. The OpenMP Board, The OpenMP API specification for parallel programming©1997-2008, http://openmp.org
17. Shepard, D.: A two-dimensional interpolation function for irregularly-spaced data. In: ACM 1968: Proceedings of the 1968 23rd ACM National Conference, pp. 517–524 (1968)
18. Standford University, Stanford University Computer Graphics Laboratory, The Stanford 3D Scanning Repository, http://graphics.stanford.edu/data/3Dscanrep/
19. 3D-CoForm project, MeshLab Software Version 1.2.3, GNU General Public License (2010), http://meshlab.sourceforge.net
20. California Tech Nathan Litke: The face compedium 3d model datasets: Copyright (2005), http://www.cs.caltech.edu/~njlitke/meshes/
21. Fan, H., Yu, Y., Peng, Q.: J. IEEE Transactions on Visualization and Computer Graphics 6(12), 312–324 (2010)