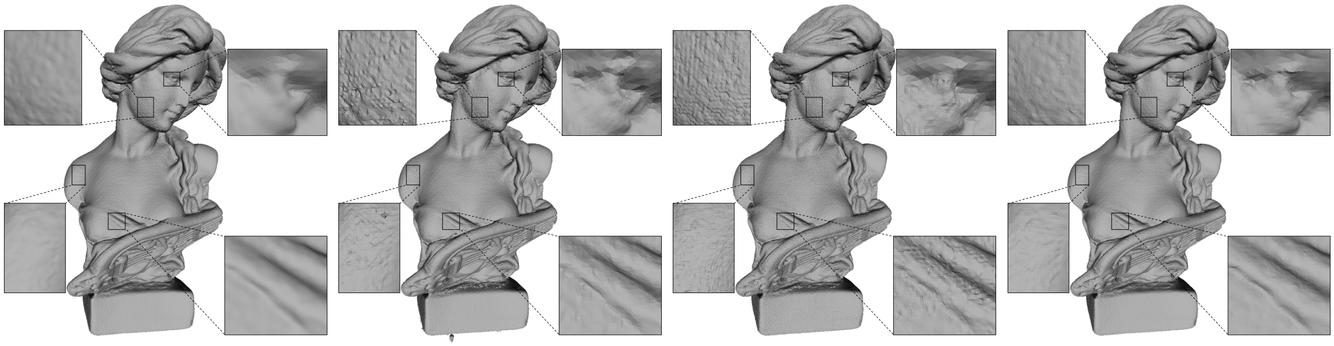
Gauss Surface Reconstruction



(a) *PR* ($|v| = 1.4$) ($t_1 = 98.14$) **(b)** *SSD* ($|v| = 1.2$) ($t_1 = 29.82$) **(c)** *SPR* ($|v| = 1.3$) ($t_1 = 56.58; t_{10} = 11.48$) **(d)** *GR* ($|v| = 1.4$) ($t_1 = 67.29; t_{10} = 7.92; t_g = 2.78$)

Figure 1: *Reconstructions of the Lady model by Poisson Reconstruction (PR) [Kazhdan et al. 2006], Smoothed Signed Distance Reconstruction (SSD) [Calakli and Taubin 2011], Screened Poisson Reconstruction (SPR) [Kazhdan and Hoppe 2013], and our Gauss Reconstruction (GR). The Lady model is a real-world scanned data with 0.5 millions samples. $|v|$ denotes the number of vertices in millions of the reconstructed mesh, and t_1, t_{10} and t_g denote the running time in seconds of the reconstructions with single thread, 10 threads and GPU respectively.*

Abstract

In this paper, we introduce a surface reconstruction method that can perform gracefully with non-uniformly-distributed, noisy, and even sparse data. We reconstruct the surface by estimating an implicit function and then obtain a triangle mesh by extracting an iso-surface from it. Our implicit function takes advantage of both the indicator function and the signed distance function. It is dominated by the indicator function at the regions away from the surface and approximates (up to scaling) the signed distance function near the surface. On one hand, it is well defined over the entire space so that the extracted iso-surface must lie near the underlying true surface and is free of spurious sheets. On the other hand, thanks to the nice properties of the signed distance function, a smooth iso-surface can be extracted using the approach of marching cubes with simple linear interpolations. More importantly, our implicit function can be estimated directly from an explicit integral formula without solving any linear system. This direct approach leads to a simple, accurate and robust reconstruction method, which can be paralleled with little overhead. We call our reconstruction method Gauss surface reconstruction. We apply our method to both synthetic and real-world scanned data and demonstrate the accuracy, robustness and efficiency of our method. The performance of Gauss surface reconstruction is also compared with that of several state-of-the-art methods.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling;

Keywords: Surface Reconstruction, Point Cloud, Gauss Lemma

1 Introduction

Surface reconstruction is a classic problem that has been studied for more than three decades. Many elegant methods are available, and in this paper, we focus on implicit methods. Indicator function is a popular choice of implicit function [Kazhdan et al. 2006; Kazhdan and Hoppe 2013]. However, the indicator function is discontinuous on the surface. In order to obtain a good approximation of surface normals, one has to apply a smoothing filter to the indicator function such as that in Poisson reconstruction [Kazhdan et al. 2006; Kazhdan and Hoppe 2013]; otherwise, the recovered surface may not be smooth, as shown in Figure 2. Meanwhile, the methods with smooth filtering often find it difficult to control approximation error and thus tend to overfit or over-smooth the data. Another widely used function in surface reconstruction is the signed distance function [Hoppe et al. 1992; Curless and Levoy 1996]. Unlike the indicator function, a signed distance function is smooth near the surface, which makes it easier to extract a smooth watertight surface from its zero level-set compared with other methods. However, the signed distance function is difficult to compute in the area away from the surface; moreover, the resulting methods are often sensitive to noise and may generate spurious surface sheets. One natural idea is to construct a hybrid function, in which the indicator function dominates regions away from the surface, and the signed distance function controls the near-surface part.

[Calakli and Taubin 2011] attempted to estimate such hybrid function by minimizing some energy function. In this paper, we propose an explicit integral formula for constructing such function based on the well-known Gauss Lemma in the potential theory(e.g., [Wendland 2009]). Gauss Lemma gives an integral formula for the indicator function. Here, we further modify the Gauss Lemma to give signed distance function near the surface while keeping the indicator function intact away from the surface. Our implicit function can be directly estimated from this integral formula, without solving a linear system as in Poisson reconstruction, or minimizing an energy function as in [Calakli and Taubin 2011].

In our integral formula, the integrand is near singular at the sample points and global over the whole computational domain, which introduce some difficulties to evaluate the integral accurately and efficiently. To overcome the singularity of the integrand, we introduce a method called *disk integration* to compute the integral near the singularity. Meanwhile, the globalness of the integral formula makes the algorithm quite slow. To address this issue, we use the famous fast multipole method(FMM) [Greengard and Rokhlin 1987] to accelerate the computation. By integrating all these pieces together, explicit integral formula, disk integration and FMM, we obtain an accurate, efficient and robust surface reconstruction method. We call this method Gauss surface reconstruction, as the integral formula comes from the Gauss Lemma.

Our proposed Gauss reconstruction algorithm inherits many nice properties such as robustness against noise and missing data, free of spurious surface sheets away from the input samples and easy to recover iso-surface from indicator function- and signed distance- based reconstruction methods. Thus, our method performs gracefully even with poor quality data, including non-uniformly-distributed, noisy, and even sparse data. Furthermore, our direct approach makes the reconstruction algorithm simple and accurate. More importantly, our Gauss reconstruction has a natural parallel implementation and an overhead that is almost negligible.

Figure 1 shows the comparison of our Gauss reconstruction with several state-of-the-art methods on the real-world scanned Lady model. All reconstructions are computed using an octree with a maximum depth of 10. From Figure 1, we can see that our Gauss reconstruction generates a high quality reconstruction of the Lady model: it preserves the details well while avoiding overfitting the input samples. The parallel implementation of our Gauss reconstruction almost achieves a perfect linear speed up, and with the GPU implementation, our method greatly outperforms other methods in speed.

2 Related Works

Surface reconstruction Surface reconstruction from point cloud has attracted great attention in the past thirty years, both in theory and in practice. Many related algorithms have been proposed. We give a brief review to those relevant to our work. There are two main categories: combinatorial algorithms and implicit algorithms.

Combinatorial methods take (part of) input sample points as vertices and reconstruct output meshes by determining the connectivity of input samples. Many of these methods are based on the Voronoi diagram or its dual Delaunay triangulation, including Crust [Amenta et al. 1998], Power Crust [Amenta et al. 2001], Cocone [Amenta et al. 2002], Robust Cocone [Dey and Goswami 2004], Wrap [Edelsbrunner 2003] and flow complex [Giesen and John 2008]. These methods have shown good theoretical results, in practice, however, they are usually sensitive to noise and may produce jagged surfaces. In [Kolluri et al. 2004], a spectral based approach is proposed to smooth the surface. In [Xiong et al. 2014], a learning approach is proposed to treat geometry and connectivity reconstruction as one joint optimization to improve reconstruction quality.

Implicit methods attempt to estimate implicit functions from input samples and extract iso-surfaces to generate triangle meshes. Poisson reconstruction and its variant [Kazhdan et al. 2006; Kazhdan and Hoppe 2013] are most relevant to our work, which estimate indicator functions of unknown models. In [Muraki 1991; Walder et al. 2005], Radial Basis Functions (RBFs) are used as bases for defining implicit functions, where coefficients of bases are determined by fitting input data. Since RBFs are global, FMM is employed to improve the efficiency [Carr et al. 2001]. The signed distance function is a natural choice as an implicit function for surface reconstruction, where implicit function can be estimated either locally as distances to tangent planes of nearby samples [Hoppe et al. 1992; Curless and Levoy 1996] or globally by minimizing the fitting error [Calakli and Taubin 2011]. In [Amenta and Kil 2004; Dey and Sun 2005; Levin 1998], moving least squares (MLS) is used to define implicit surfaces, which are extremal sets of certain energy. MLS is associated with a nice projection operator that can be used for surface smoothing. Unlike our method, the implicity function in MLS is often only meaningful near the surface and thus the reconstruction of MLS may generate spurious surface sheets away from the surface. Finally, [Fuhrmann and Goesele 2014] defined the implicit function as the sum of compactly supported basis functions. By leveraging the extra scale information input, it performs well on large, redundant and potentially noisy datasets. The surfaces reconstructed by implicit methods often do not interpolate input samples; therefore, they are smoother than those reconstructed by combinatorial methods.

As stated in [Berger et al. 2014], surface normal plays an important role in surface reconstruction, however, challenging to obtain when certain information not present. Therefore, surface reconstruction based on unoriented point cloud is also drawing much attention. [Chen et al. 2013] computed the higher-order local approximations of non-oriented input gradients based on a MLS formulation. In [Alliez et al. 2007], a Voronoi-PCA estimation is performed, which results in a tensor field encoding normal information, and then computes the implicit function to recover the surface. To process the unoriented data, we just use the “compute normals for point sets function” in meshlab with default parameters to estimate the normal as a preprocess. Experiments show that our method is quite robust to point normal. The accurate estimation of normal is not a mandatory requirement.

Solution to the Laplace equation The solution to the Laplace equation varies from one method to another. For example, [Kazhdan et al. 2006; Kazhdan and Hoppe 2013] turned it into a linear system defined on the B-spline basis and solved the sparse linear system to get a solution with explicitly defined Dirichlet/Neumann boundary constraints. The boundary element method (BEM), as applied in our method, is another mathematically beautiful tool for evaluating the solution to the Laplace equation, which is also widely used in different areas (e.g., mesh segmentation [Jacobson et al. 2013]). One advantage of the BEM based solution is that no boundary conditions should be explicitly imposed. Unlike the direct usage of BEM kernel in [Jacobson et al. 2013], we make some modifications to the original kernel for easy interpolation in the current work. In addition, the BEM-based solution is easy to accelerate by using an accurate hierarchical estimation introduced by [Jacobson et al. 2013] or an approximated FMM method. In our Gauss reconstruction, we apply the FMM method for faster performance. Meanwhile, the accuracy near the surface, which is very important to our method, is guaranteed by an innovative scheme called *disk integration*.

Iso-surface extraction For the iso-surface extraction, marching cubes [Lorensen and Cline 1987] and its adaptation to octree [Wilhelms and Van Gelder 1992] are the most popular methods. Many efficient variants or extensions have been proposed. Primal MC methods, such as that proposed by [Kazhdan et al. 2007] extract a watertight mesh by means of edge trees where the positions of the iso-value-crossings are defined. [Schaefer and Warren 2004] extracted the iso-surface with sharp features by aligning dual grid vertices with implicit function features. Delaunay refinement-based methods [Boissonnat and Oudot 2005] produce good quality triangle meshes, although they are less efficient and are difficult to parallelize.

3 Gauss Reconstruction

Our problem can be stated as follows: the input data S is a set of oriented points $S = \{s_1, s_2, ..., s_n\}$, each consisting of a position $s.p$ and an outward normal $s.\vec{N}$, sampling the boundary $\partial\Sigma$ of an

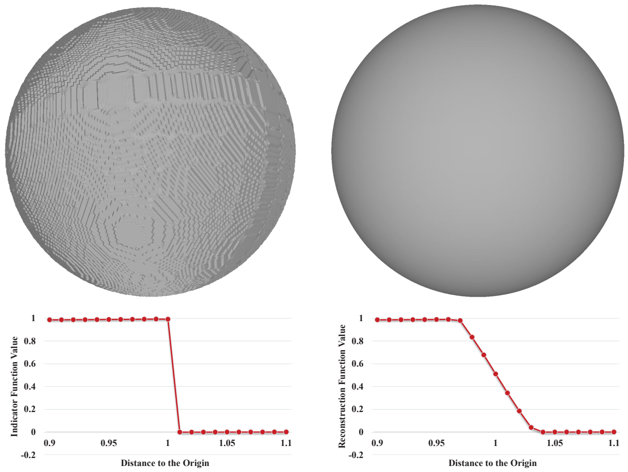


Figure 2: *Left column: The reconstruction from the indicator function. The top shows the resulting mesh and the bottom shows the indicator function around the north pole restricted to the diameter passing the north pole. Right column: The reconstruction from the Gauss reconstruction function. The top shows the resulting mesh and the bottom shows the Gauss reconstruction function around the north pole restricted to the diameter passing the north pole.*

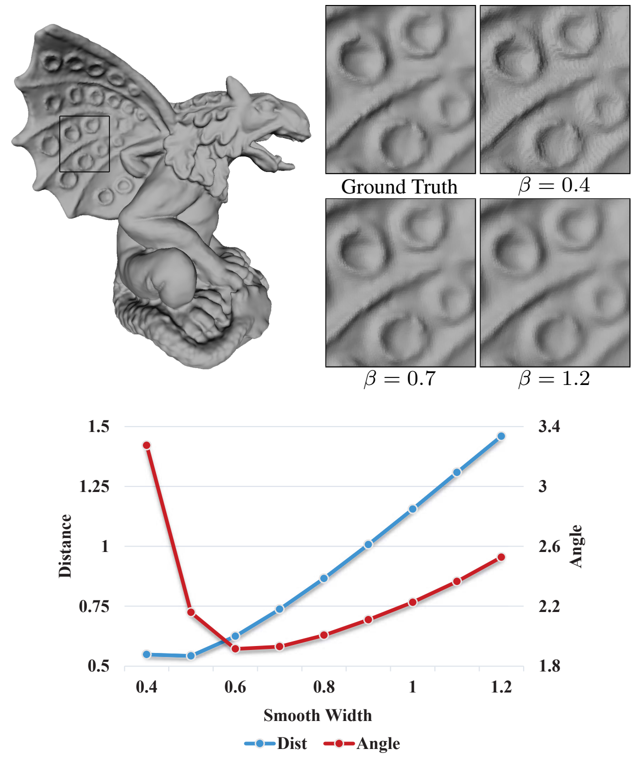


Figure 3: *Choice of width coefficient. The first row shows visual effects; the second row shows the average position error (Dist) and the average angle error using the reconstruction benchmark [Berger et al. 2013].*

unknown region $\Sigma \in \mathbb{R}^3$, i.e., $s.p$ lies on or near the surface, and $s.\vec{N}$ approximates the surface normal near the position $s.p$. Our goal is to reconstruct a triangle mesh approximating the boundary $\partial\Sigma$.

3.1 Reconstruction function

Our method reconstructs the surface by estimating an implicit reconstruction function $\tilde{\chi}$ combining a near-surface *signed distance function* and an off-surface *indicator function*, which makes it easy to extract the level-set while enjoying robustness of indicator function-based methods. In this section, we will introduce an explicit integral formula to estimate the implicit reconstruction function. First, we have that the indicator function χ of the region Σ has an explicit integral formula, which is given in the well-known Gauss Lemma in the potential theory [Wendland 2009].

Lemma 3.1 (Gauss Lemma)**.** *Let Σ be an open region in $\mathbb{R}^3$. Consider the following double layer potential: for any $x \in \mathbb{R}^3$*

$$\chi(x) = \int_{\partial\Sigma} \frac{\partial G}{\partial \mathbf{n}_y}(x,y)\mathrm{d}\tau(y), \tag{1}$$

where $\mathbf{n}_y$ is the outward normal of $\partial\Sigma$ at y, $\mathrm{d}\tau(y)$ is the surface area form of $\partial\Sigma$ at y, and G is the fundamental solution of the Laplace equation in $\mathbb{R}^3$, which can be written explicitly as:

$$G(x,y) = -\frac{1}{4\pi\|x-y\|}. \tag{2}$$

Then, $\chi(x)$ is the indicator function of Σ, i.e.

$$\chi(x) = \begin{cases} 0 & x \in \mathbb{R}^3 \setminus \bar{\Sigma} \\ 1/2 & x \in \partial\bar{\Sigma} \\ 1 & x \in \Sigma \end{cases} \tag{3}$$

Note that

$$\frac{\partial G}{\partial \mathbf{n}_y}(x,y) = -\frac{1}{4\pi}\frac{(x-y)\cdot\mathbf{n}_y}{\|x-y\|^3},$$

which we call the kernel function, and denoted by $K(x,y)$.

The integral formula (1) has many good properties. Nevertheless, given the indicator function χ, the resultant triangle mesh by iso-surfacing χ, denoted by M, lies in a small tubular neighborhood of the surface $\partial\Sigma$, namely the Hausdorff distance between M and $\partial\Sigma$ is small. However, given that the function χ is discontinuous at $\partial\Sigma$, the normal of a triangle in M may not approximate the normals of $\partial\Sigma$ at the points close to the triangle, see Figure 2. Furthermore, the kernel function $K(x,y)$ becomes singular when x is approaching y. To accurately evaluate the indicator function χ at the points close to the surface $\partial\Sigma$, one needs a very dense sampling of the surface, which becomes practically implausible.

To address these two issues, our strategy is to modify the indicator kernel function K. For a point $x \in \mathbb{R}^3$, we associate a width $x.w$ and modify the kernel function $K(x,y)$ for any $y \in \partial\Sigma$ as:

$$\tilde{K}(x,y) = \begin{cases} K(x,y), & \|x-y\| \geq x.w, \\ 0, & \|x-y\| < x.w. \end{cases} \tag{4}$$

The reconstruction function can be stated as:

$$\tilde{\chi}(x) = \int_{\partial\Sigma} \tilde{K}(x,y)\mathrm{d}\tau(y). \tag{5}$$

Note that $\tilde{K}(x,y)$ remains the same as $K(x,y)$ for any $y \in \partial\Sigma$ with $\|x-y\| \geq x.w$, and hence $\tilde{\chi}(x) = \chi(x)$ for any x with $d(x,\partial\Sigma) \geq x.w$.

At a point x with $d(x,\partial\Sigma) < x.w$,

$$\tilde{\chi}(x) = \int_{\partial\Sigma} K(x,y)\mathrm{d}\tau(y) - \int_{B_x(x.w)\cap\partial\Sigma} K(x,y)\mathrm{d}\tau(y).,$$

where $B_x(r)$ is the ball in $\mathbb{R}^3$ centered at x and of radius r. Notice that $x.w$ is always a small number, which means $B_x(x.w) \cap \partial\Sigma$ is a small piece of Σ. Under the assumption that the surface Σ is smooth, $B_x(x.w) \cap \partial\Sigma$ can be well approximated by a disk. Therefore, this approximation implies that

$$\begin{aligned}\tilde{\chi}(x) &\approx \chi(x) + \frac{1}{4\pi}\int_0^{2\pi}\int_0^{\sqrt{x.w^2-d(x)^2}} \frac{d(x)}{(d(x)^2+r^2)^{\frac{3}{2}}} r\mathrm{d}r\mathrm{d}\theta \\ &= \frac{1}{2} + \frac{d(x)}{2(x.w)} \end{aligned} \tag{6}$$

where $d(x)$ is the signed distance from x to $\partial\Sigma$. As Figure 2 shows, $\tilde{\chi}(x)$ is strictly a signed distance function near the surface, which is very desirable for extracting iso-surface [Calakli and Taubin 2011].

The integral formula (5) is direct and very simple. Note that the estimation of the implicit function $\tilde{\chi}$ at different points x is completely independent from each other, which leads to a natural parallel algorithm.

Next, we move to the specification of width $x.w$. Note that we only need to specify the width for the grid vertices $\mathcal{V}$. For a grid vertex $v \in \mathcal{V}$, let $v.B$ be the set of the leaf nodes in $\mathcal{O}$ having v as one of their vertices. Set $v.w$ to be β times the side length of the smallest cube in $v.B$, where β is a constant, which we call width coefficient. Then, we define the neighboring vertices $v.\mathcal{V}$ of v in the octree so that a grid vertex u is in $v.\mathcal{V}$ if u and v are connected by an edge of a cube in $v.B$. Here, it is highly possible that $v.w$ and $u.w$ differ significantly even when u and v are neighbors, and the resultant function $\tilde{\chi}$ may become rough. To address this issue, we further smooth $v.w$ by averaging the widths over the neighbors, namely set

$$v.w = \frac{\sum_{u\in v.\mathcal{V}} u.w}{|v.\mathcal{V}|},$$

and repeat this averaging step for k times. In the paper, we set $k = 20$.

The width coefficient provides a way to control the trade-off between the position accuracy and the smoothness of the reconstruction. See Figure 3. The larger the β is, the smoother the reconstructed surface, but with a less accurate position. Of course, if β is chosen too big, both position accuracy and angle accuracy decrease. A typical value of β is practically set to be 0.7.

3.2 Disk integration

Note that although the kernel function $\tilde{K}$ is not singular, there may still exist some near-singularity problems owing to the specification of the small width coefficient. To address this issue, we propose an approach called *disk integration*. Recall that the input data S samples the surface $\partial\Sigma$. Imagine that each sample point $s \in S$ represents a neighboring region on $\partial\Sigma$, denoted as $s.V$, such that the set $\{s.V\}_{s\in S}$ decomposes the surface $\partial\Sigma$. One can think of $s.V$ as the Voronoi region of s on $\partial\Sigma$. Then $\tilde{\chi}(x) = \sum_{s\in S} C(x,s)$ where

$$C(x,s) = \int_{s.V} \tilde{K}(x,y)\mathrm{d}\tau(y). \tag{7}$$

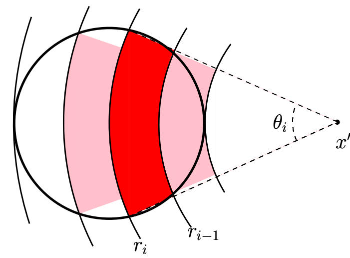


Figure 4: *Illustration of integral domain (shaded region) of disk integration.*

Note that $s.V$ is unknown and we use a disk perpendicular to $s.\vec{N}$ to approximate $s.V$. The radius of this disk is estimated as the average distance to the k-nearest samples in S. In this paper, we fix $k = 10$ for all samples. We denote this disk $s.D$, its radius $s.r$, and take the area of $s.D$ as the surface area $s.A$.

We approximate $C(x,s)$ using $\int_{s.D}\tilde{K}(x,y)\mathrm{d}y$. Note that even over the simple domain $s.D$, the above integration cannot be calculated explicitly. Our strategy is to approximate $s.D$ using k layers of partial annuli (see the shaded regions in Figure 4), and over each layer the integration of the kernel function $\tilde{K}(x,y)$ can be calculated analytically. Let x' be the projection of x on the plane containing $s.D$. Denote $C(r)$ as the circle centered at x' of the radius r, and $A(r,R)$ as the annulus centered at x' of the inner radius r and the outer radius R. Let $r_0 = \min_{y\in s.D}\|x'-y\|$ and $r_k = \max_{y\in s.D}\|x'-y\|$, and $r_i = r_0 + \frac{i(r_k-r_0)}{k}$, for $0 \le i \le k$. Here, r_0 is 0 if x' is in the disk. Let θ_i be the central angle of the arc $C(r_i)\cap s.D$, and F_i be the fan spanned by the same arc. The partial annulus at the ith layer is $F_i \cap A(r_{i-1}, r_i)$. Set $d = \|x - x'\|$. Then $C(x,s)$ is approximated by $DI(x,s) = \sum_{1\le i\le k} c_i$ where

$$\begin{aligned} c_i &= \int_{F_i\cap A(r_{i-1},r_i)} \tilde{K}(x,y)\mathrm{d}y \\ &= -\frac{1}{4\pi}\int_0^{\theta_i}\int_{r_{i-1}}^{r_i} \frac{d}{(d^2+r^2)^{3/2}} r\mathrm{d}r\mathrm{d}\theta \\ &= \frac{\theta_i d}{4\pi}\left(\frac{1}{\sqrt{d^2+r_{i-1}^2}} - \frac{1}{\sqrt{d^2+r_i^2}}\right) \end{aligned}$$

In the paper, we fix the number of layers $k = 20$.

Furthermore, notice that if the point x is far away from the sample s so that the integral function $\tilde{K}(x,y)$ over $s.D$ becomes well-approximated by the constant $\tilde{K}(x,s)$, then $C(x,s)$ can simply be evaluated by $DC(x,s) = \tilde{K}(x,s)s.A$. Set $R(x,s) = \frac{\|x-s\|+s.r}{\|x-s\|-s.r}$. One can verify that the larger $R(x,s)$ is, the closer the function $\tilde{K}(x,y)$ over $s.D$ is to the constant $\tilde{K}(x,s)$. In this paper, when $R(x,s) > 2$, we approximate $C(x,s)$ using $DC(x,s)$.

Using the disk integration, we can achieve high accuracy in computing the integral. Figure 2 (left column) shows the indicator function of the unit sphere restricted to points passing the center, which were estimated using the above approach from 1000 random samples. The Hausdorff distance between the reconstructed triangle mesh and the original sphere is less than 5×10^{-3}. Another advantage of disk integration is that it naturally handles the missing data. The holes resulting from the missing data are covered by disks, and

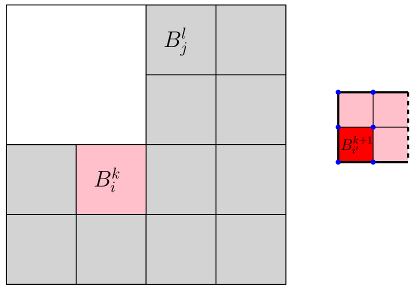


Figure 5: *The cubes $\mathcal{O}$ at depth k may not cover the entire domain due to the adaptivity of $\mathcal{O}$. The red cube $B_{i'}^{k+1}$ is a subcube of the pink cube B_i^k. The blue dots in B_i^k form set of grid vertices $\mathcal{V}_i^k$ in B_i^k.*

the integral formula integrates all the disks together to give a water-tight surface automatically.

The disk integration is a specific strategy to address the singularity problem in our integral formula (5). However, this strategy does not apply to other reconstruction methods that do not have integral formula.

3.3 Fast Multipole Method

Given that the the estimation of $\tilde{\chi}(x)$ using the integral formula (1) is global, one has to integrate the kernel function $\tilde{K}(x, y)$ over the entire surface $\partial\Sigma$ to obtain a correct estimation of $\tilde{\chi}(x)$. In particular, one can not perform thresholding based on the value of $\tilde{K}(x, y)$ and skip integrating the region where $\tilde{K}(x, y)$ is small. To see this, imagine Σ is a ball of radius r, and x is the center of the ball. For $y \in \partial\Sigma$, $\tilde{K}(x, y)$ can be made arbitrarily small by choosing the radius r that is large enough. However, $\tilde{\chi}(x)$ remains as the constant 1, independent of r. Therefore, to estimate $\tilde{\chi}$ at m different locations, a native implementation requires at least $O(mn)$ operations. Recall that n is the number of samples in S. Fortunately, the kernel function $\tilde{K}(x, y)$ over two distant regions can be well-approximated by a constant function. This enables us to speed up the estimation of χ by using the well-known *fast multipole method (FMM)*, which improves the complexity to $O(m + n \log n)$.

In this subsection, we describe an implementation of FMM for the estimation of the Gauss reconstruction function $\tilde{\chi}$. An octree is employed as the multi-resolution data structure in FMM, and the same octree is also used for isosurfacing $\tilde{\chi}$.

Given a set of samples S and a maximum tree depth D, the octree is the minimal octree so that each sample falls into a leaf node of depth D. For a non-uniform sampling, we follow [Kazhdan et al. 2006] and reduce the depth for the samples in the sparse regions. We denote $\mathcal{O}$ as the resultant octree, and $\mathcal{V}$ as the set of grid vertices of the octree $\mathcal{O}$. Our goal is to evaluate the Gauss reconstruction function at $\mathcal{V}$. Now consider the cubes $\{B_i^k\}_i$ of $\mathcal{O}$ at depth k, see Figure 5. A cube B_i^k may be half open, i.e., does not contain the faces with the maximum x, or y, or z coordinate, unless they are on the boundary. See the pink cube in Figure 5. Let $\mathcal{V}_i^k = \mathcal{V} \cap B_i^k$ (See the blue dots in B_i^k in Figure 5), and $S_i^k = S \cap B_i^k$. For a set X, we denote $|X|$ the cardinality of X. Let $\bar{v}_i^k$ be the representative grid of B_i^k defined by

$$\bar{v}_i^k = \frac{\sum_{v \in \mathcal{V}_i^k} v}{|\mathcal{V}_i^k|},$$

and $\bar{s}_i^k$ be the representative sample of B_i^k defined by

$$\bar{s}_i^k.p = \frac{\sum_{s \in S_i^k} s.A \cdot s.p}{\sum_{s \in S_i^k} s.A},$$

$$\bar{s}_i^k.\vec{N} = \frac{\sum_{s \in S_i^k} s.A \cdot s.\vec{N}}{\sum_{s \in S_i^k} s.A}, \text{ and}$$

$$\bar{s}_i^k.A = \sum_{s \in S_i^k} s.A.$$

The disk $\bar{s}_i^k.D$ is centered at $\bar{s}_i^k$, perpendicular to $\bar{s}_i^k.\vec{N}$, and of the area $\bar{s}_i^k.A$. Let a_k be the side length of the cubes at depth k. The basic idea of our implementation of FMM is as follows. We start with the cube at depth 1. In general, consider two cubes B_i^k and B_j^l at depth l and depth k, respectively. Note that B_i^k and B_j^l may be the same cube. If $\|\bar{s}_i^k - \bar{v}_j^l\| \geq ca_k$, then for any grid vertex $v \in \mathcal{V}_j^l$, we approximate $\sum_{s \in S_i^k} C(v, s)$ using $C(\bar{v}_j^l, \bar{s}_i^k)$. Otherwise, we repeat the above procedure for any pairs of subcubes, one in B_i^k and the other in B_j^l until both are leaf nodes. Only when both are leaf nodes do we estimate $C(v, s)$ for an individual sample $s \in S_i^k$ and an individual grid vertex $v \in \mathcal{V}_j^l$. Moreover, when we invoke the estimation of $C(\bar{v}, \bar{s})$ for a representative grid vertex $\bar{v}$ and a representative sample $\bar{s}$, we assume that $\bar{v}$ and $\bar{s}$ are far away from each other and compute $DI(\bar{v}, \bar{s})$ or $DC(\bar{v}, \bar{s})$ using the kernel function $\tilde{K}$. Therefore, there is no need to associate a width to a representative grid vertex $\bar{v}$. In the paper, we fix the constant $c = \sqrt{2}$. Pseudocode 1 shows our FMM implementation.

```
function FMM(B_i^k, B_j^l, f : V → ℝ)
    if ||s̄_i^k − v̄_j^l|| ≥ c a_k then
        evaluate e ≈ C(v̄_j^l, s̄_i^k)
        f(v) = f(v) + e for any v ∈ V_j^l.
    else
        if both B_i^k and B_j^l are leaves then
            for all s ∈ S_i^k and v ∈ V_j^l do
                evaluate e ≈ C(v, s)
                f(v) = f(v) + e;
            end for
        else if Neither B_i^k nor B_j^l is a leaf then
            for all B_{i'}^{k+1} ⊂ B_i^k and B_{j'}^{l+1} ⊂ B_j^l do
                FMM(B_{i'}^{k+1}, B_{j'}^{l+1}, f)
            end for
        else if B_i^k is a leaf and B_j^l is not a leaf then
            for all B_{j'}^{l+1} ⊂ B_j^l do
                FMM(B_i^k, B_{j'}^{l+1}, f)
            end for
        else
            for all B_{i'}^{k+1} ⊂ B_i^k do
                FMM(B_{i'}^{k+1}, B_j^l, f)
            end for
        end if
    end if
end function
```

Pseudocode 1: *FMM.*

3.4 Iso-surface extraction

In this subsection, we introduce a way to calculate the iso-value and perform the interpolation.

To make the samples locate uniformly inside and outside the surface $\partial\Sigma$, we simply set iso-value γ to the median of $\tilde{\chi}$ values at sample positions.

Concerning the way to interpolate, we assume two adjacent vertices $v_i \in \mathcal{V}$ and $v_j \in \mathcal{V}$ that come across the iso-surface, with $(\tilde{\chi}(v_i) - \gamma) \cdot (\tilde{\chi}(v_j) - \gamma) < 0$. From Equation (6), we observe that given the signed distance $d(v)$, the function value $\tilde{\chi}(v) - \frac{1}{2}$ is inversely proportional to its associated width $v.w$, where $\frac{1}{2}$ is the ideal iso-value. Then, the location of crossing point $\tilde{v}_{ij} \in \partial\Sigma$ can be stated as the linear interpolation of $(\tilde{\chi}(v) - \gamma) \cdot v.w$ given by:

$$\frac{\tilde{v}_{ij} - v_i}{v_j - v_i} = \frac{0 - (\tilde{\chi}(v_i) - \gamma) \cdot v_i.w}{(\tilde{\chi}(v_j) - \gamma) \cdot v_j.w - (\tilde{\chi}(v_i) - \gamma) \cdot v_i.x}. \quad (8)$$

Finally, marching cubes [Lorensen and Cline 1987] is applied to extract the iso-surface.

In the end, we summarize our Gauss reconstruction in Pseudocode 2.

function GaussRecon(S: samples, D: maximum depth, β: width coefficient)
  Estimate $s.r$ for each sample $s \in S$
  Given D, construct an adaptive octree $\mathcal{O}$
  Compute representative samples $\bar{s}$ for all cubes in $\mathcal{O}$.
  Compute representative grid vertices $\bar{v}$ for all cubes in $\mathcal{O}$.
  Given β, estimate $v.w$ for each grid vertex $v \in \mathcal{V}$
  Initialize $f : \mathcal{V} \to \mathbb{R}$ to be zero.
  Call **FMM**(B_1^1, B_1^1, f).
  Set the iso-value as the median of f.
  Extract the iso-surface M using marching cubes over $\mathcal{O}$.
  Output M.
end function

Pseudocode 2: *GaussRecon*

3.5 Parallel and GPU implementation

For the grid vertices v, the estimation of the Gauss reconstruction function $\chi(v)$ is independent from each other, which leads a straightforward parallel implementation. In particular, we open new threads to execute the calls of $FMM(B_i^k, B_j^l, f)$ with $k, l \leq c$. The parameter c is chosen so that we have just enough threads so that the load on each core is balanced and the overhead of multi-threads is minimized simultaneously. In the paper, we set $c = 5$ for CPU parallel and $c = 10$ for GPU implementation.

4 Results

In this section, we evaluate our Gauss reconstruction (GR) in terms of accuracy, robustness, and efficiency, and compare its performance to those of the state-of-the-art methods, including Poisson reconstruction [Kazhdan et al. 2006] (PR) and its variant screened Poisson reconstruction [Kazhdan and Hoppe 2013] (SPR), smooth signed distance reconstruction [Calakli and Taubin 2011] (SSD) and the dictionary learning reconstruction [Xiong et al. 2014]. Note that we perform the comparison using the most recent implementation of these methods available online. In particular, using the most recent implementation, the performance of SSD has greatly improved compared with those reported in [Kazhdan and Hoppe 2013]. We follow [Kazhdan and Hoppe 2013], and set the weights for the different terms of the energy functional in SSD as 1 for value, 1 for gradient, 0.25 for Hessian, after which we set the data fitting weight $\alpha = 4$ in SPR. Unless we explicitly state that we use other values, we by default set the maximum depth $D = 10$ for octree construction in all methods and the width coefficient $\beta = 0.7$ in our Gauss reconstruction. All the experiments are performed on a Windows 7 workstation with an Intel Xeon E5-2690V3 CPU @2.6GHz and Nvidia GeForce GTX TITAN X GPU.

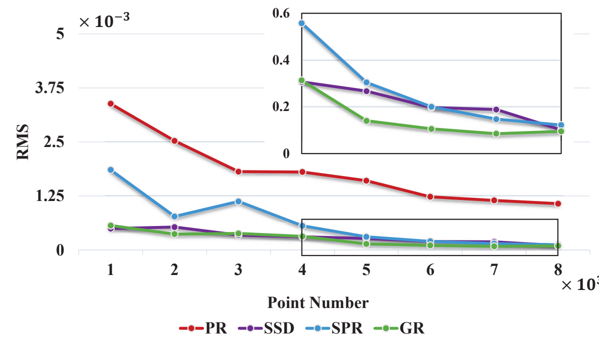


Figure 6: *The average error RMS of the reconstructions by different methods. The sub-figure on top-right is the zoom-in on the marked box.*

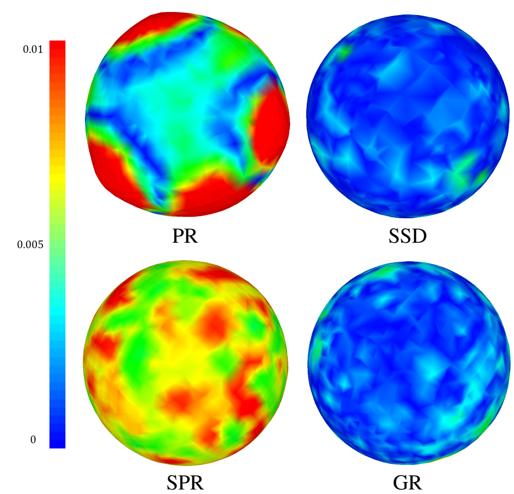
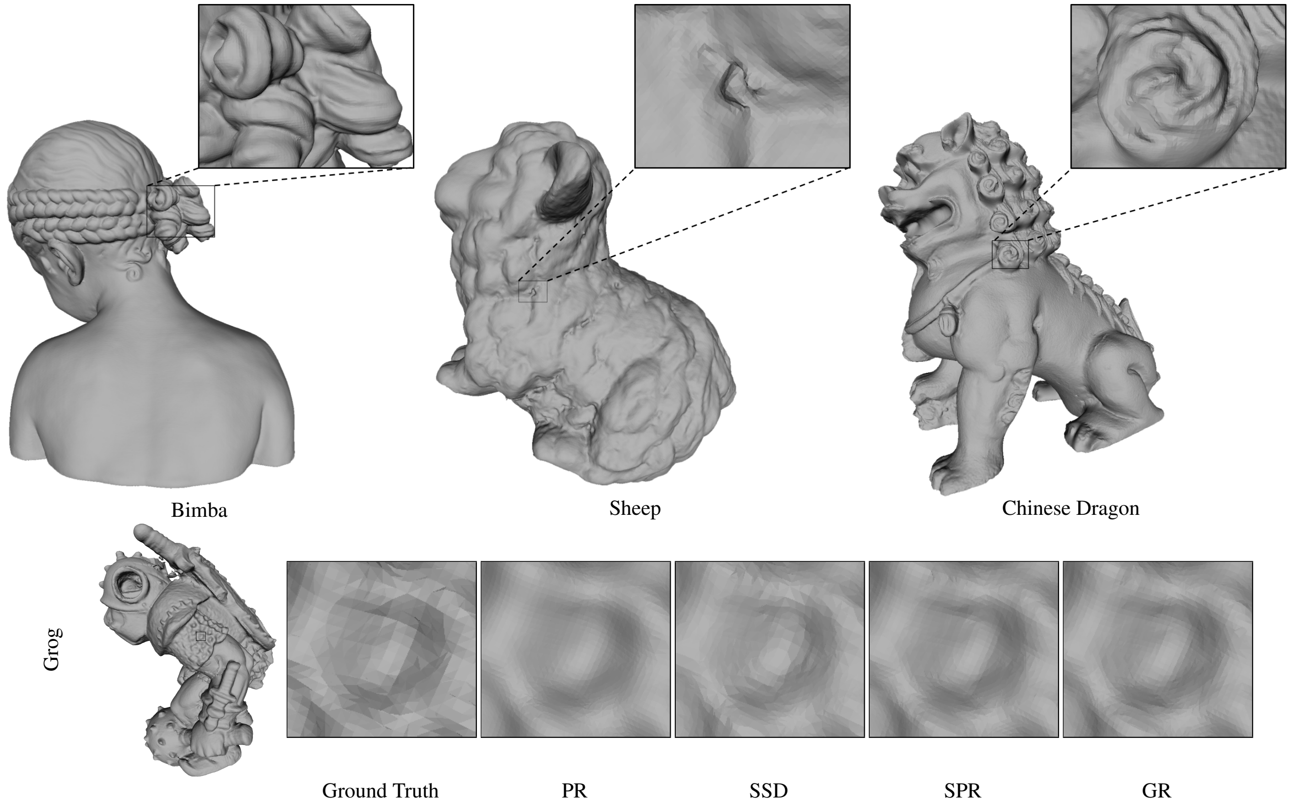


Figure 7: *The reconstructed unit sphere from* 1000 *random samples. The color illustrates the RMS (relative to the bounding box diaganol) error distribution: small error in blue and big error in red.*

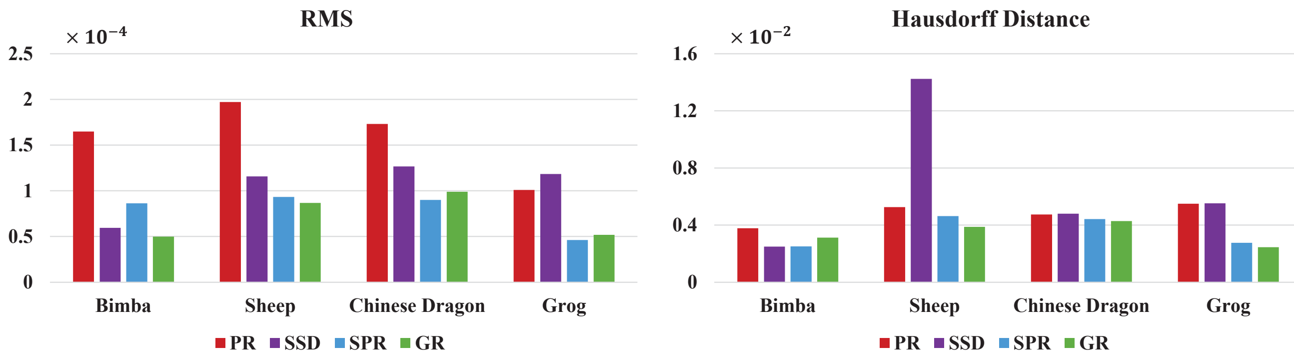
4.1 Accuracy

First, we consider the reconstruction of unit sphere from samples, in which the accurate ground truth is known. We generate 1000 to 8000 samples according to a Gaussian mixture of eight Gaussian in $\mathbb{R}^3$ and then radially project them into unit sphere. We use the average error RMS to measure the quality of the reconstructed surface.

Figure 6 shows the error statistics of the reconstructions by different methods. As can be seen, our GR performs the best and PR has the largest error. For 1000 samples, we color the RMS error (relative to the bounding box diagonal) for each vertex to visualize the error distribution. See Figure 7. In this case, the sphere obtained by PR is visually not round.



(a) *Reconstructed meshes.*

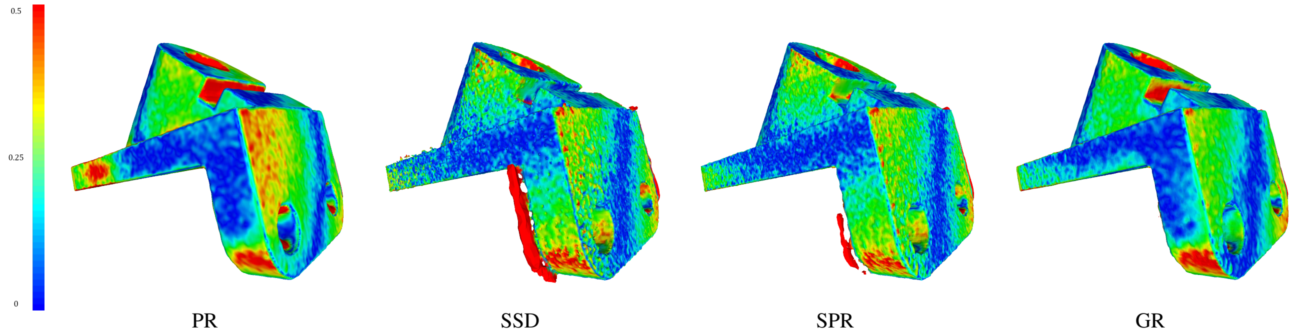


(b) *The RMS approximation error and the Hausdorff approximation error for the reconstructions of four point sets: Bimba, Sheep, Chinese dragon and Grog.*

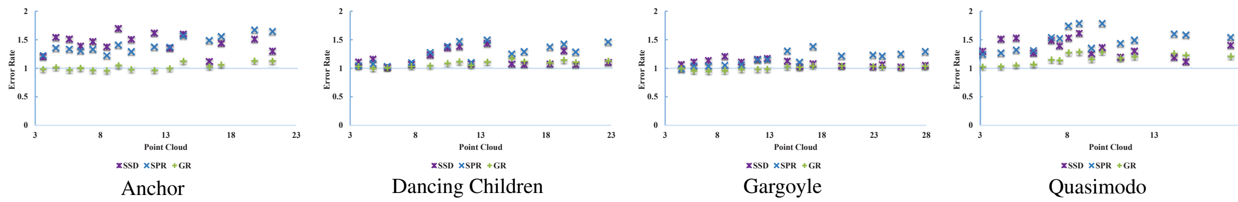
Figure 8: *The accuracy illustration. The running time: Bimba($|v| = 0.50$, PR: 62.20s, SSD: 35.04s, SPR: 73.15s, GR: 40.01s), Sheep($|v| = 0.16$, PR: 31.66s, SSD: 22.38s, SPR: 24.99s, GR: 16.18s), Chinese dragon($|v| = 0.66$, PR: 109.43s, SSD: 44.28s, SPR: 96.02s, GR: 74.25s), Grog($|v| = 0.88$, PR: 178.68s, SSD: 59.44s, SPR: 133.68s, GR: 112.71s). The number of samples is in millions.*

Next, we consider the general models. To estimate the numerical accuracy of the reconstruction results, we follow the same strategy as that used by [Berger et al. 2013]. First, we sample points from a known mesh, or simply take its vertices, and then reconstruct surfaces with this point set. Next, we use the Metro tool [Cignoni et al. 1998] to compute the Hausdorff distance (measuring the worse error) and the mean distance (measuring the average error) between the reconstructed mesh and the known mesh. Figure 8 shows the result. In general, SPR and GR have a comparable performance on this set of models and both outperform PR and SSD.

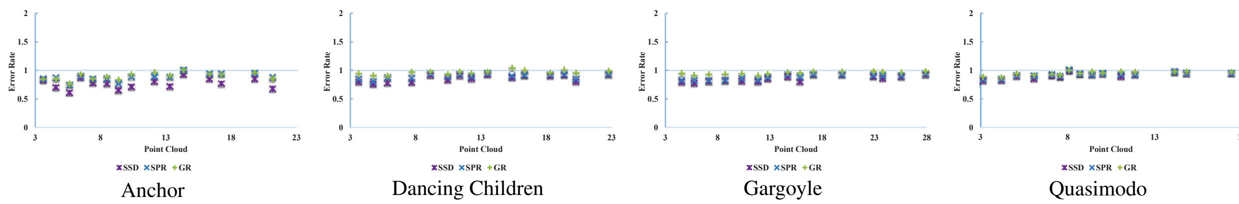
We also apply the reconstruction methods to the data from the reconstruction benchmark [Berger et al. 2013]. Due to the limited space, we only report the results on four data sets, namely: Anchor, Dancing Children, Gargoyle and Quasimodo. Following [Kazhdan and Hoppe 2013], we set the maximum depth $D = 9$ in this experiment. The error shown in Figure 9 is relative to that of PR. From Figure 9a, we can see that PR and GR generate visually similar results while SPR and SSD produces extra spurious sheets near the surface. However, the accuracy of GR is much better than that of PR. Figures 9b and 9c show the average angle error and the average position error, respectively. For this set of examples, overall, PR performs the best in terms of angle accuracy but the worst in position accuracy. In comparison, SSD performs the best in terms of position accuracy. However, from Figure 9a, SSD may overfit the data. Our GR seems to achieve a better balance between position accuracy and angle accuracy.



(a) *Visualization of position errors for reconstruction of the Anchor model. Errors are visualized using a blue-green-red colormap, with blue corresponding to smaller errors and red to larger ones.*



(b) *Average angle error (relative to that of PR).*



(c) *Average position error (relative to that of PR).*

Figure 9: *Results from the reconstruction benchmark.*

4.2 Robustness

In this subsection, we test our GR over the noisy data, including both synthetic Gaussian noise and real-world scanned data with noise and missing data, after which we compare the performance of different reconstructions. In the end, we apply an incomplete data set on GR to test its ability of filling holes.

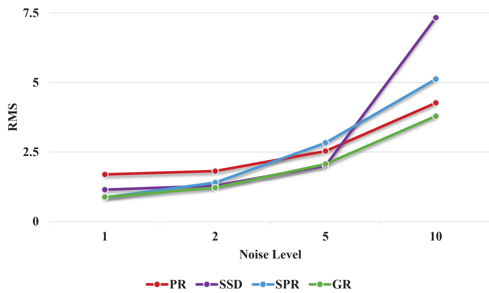
Synthetic noise In this example, we add to the Armadillo model the different levels of noise by perturbing both positions and normals of the samples according to Gaussian distribution of different variances.

Figure 10b shows the reconstructed surfaces using the proposed GR from the samples perturbed by a Gaussian with variance equal to 0.005 times the diagonal of the bounding box. Figure 10c shows the details of reconstructions at different noisy levels by zooming in the region marked in Figure 10b. As can be seen, SPR and SSD apparently overfit the data and are, therefore, sensitive to noise and reconstruct bumpy surfaces. In comparison, PR always produces smooth reconstructions, but its accuracy is the lowest. See Figure 10a. The surfaces reconstructed by GR are also smooth, and able to preserve more details at the same time; therefore, they are more accurate.

Real-world Scanned Data We apply the reconstruction methods to the sampling obtained by scanning several real-world models using Konica-Minolta Vivid 9i Laser Scanner. The obtained samplings contain both noise and missing data, and is highly non-uniform and unoriented. We use the "compute normals for point sets function" in meshlab with default parameters to estimate the normal as a preprocess. See the first column in Figure 11. In these examples, we set the width coefficient $\beta = 1.4$ in our GR. Visually, the reconstructions generated by PR and GR are comparable, and have better quality than those obtained by SSD and SPR, which again obviously overfit the data.

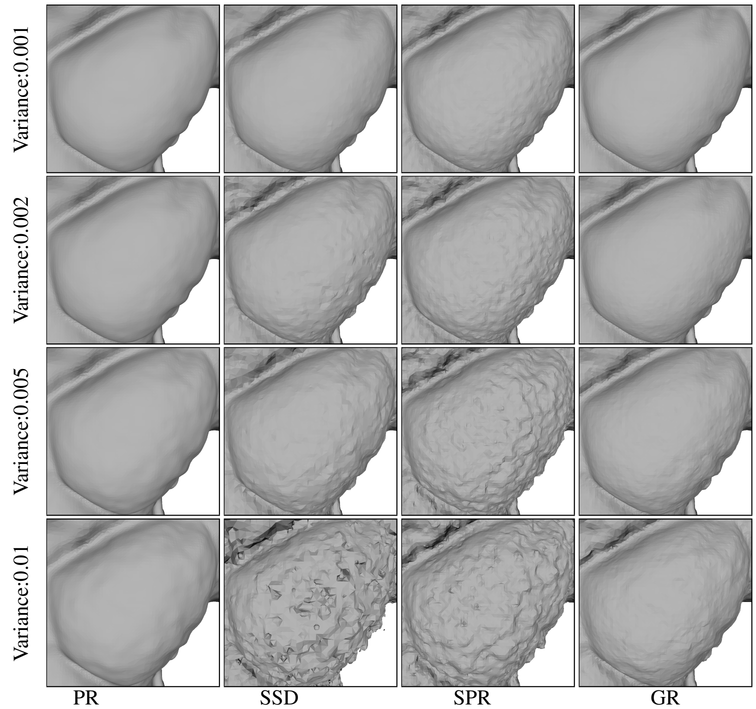
In addition, we apply our method to the well-known Merlion model with noise and missing data in comparison to the state-of-the-art explicit method [Xiong et al. 2014]. As Figure 12 shows, both methods reconstructed smooth and accurate models. The left column shows the reconstruction result of GR where octree depth is set to 10. The right column shows the detailed logo of [Xiong et al. 2014] and GR where octree depth is set to 9 and 10. With comparable output vertices, it's obvious that our method at depth 9 can achieve similar accuracy as [Xiong et al. 2014]. In addition, one limitation of [Xiong et al. 2014] is that the number of output vertices cannot exceed the input number, thus the output mesh cannot be as detailed as possible. However, in GR, the reconstructed result can be much more delicate. What's more, as stated in [Xiong et al. 2014], the running time of [Xiong et al. 2014] is a bit slower than that of SPR. And the speed of SPR is comparable to our method with single thread (See Section 4.3 for detail). It can be inferred that our method is faster than [Xiong et al. 2014].

Incomplete datasets In this subsection, we test our method on the owl model with large parts of missing data and great noise. From Figure 13, we can see that our method performs quite well in processing the missing part. Moreover, our method is quite resilient to noise and recovers the feature in detail as well.



(a) *RMS errors*

(b) *Reconstructed Armadillo by GR, Variance: 0.005*



(c) *Zoom-in on the left chest*

Figure 10: *Reconstructed surface of Armadillo from the samples perturbed by Gaussian noise of different variance. The variance is relative to the diameter of the bounding box.*

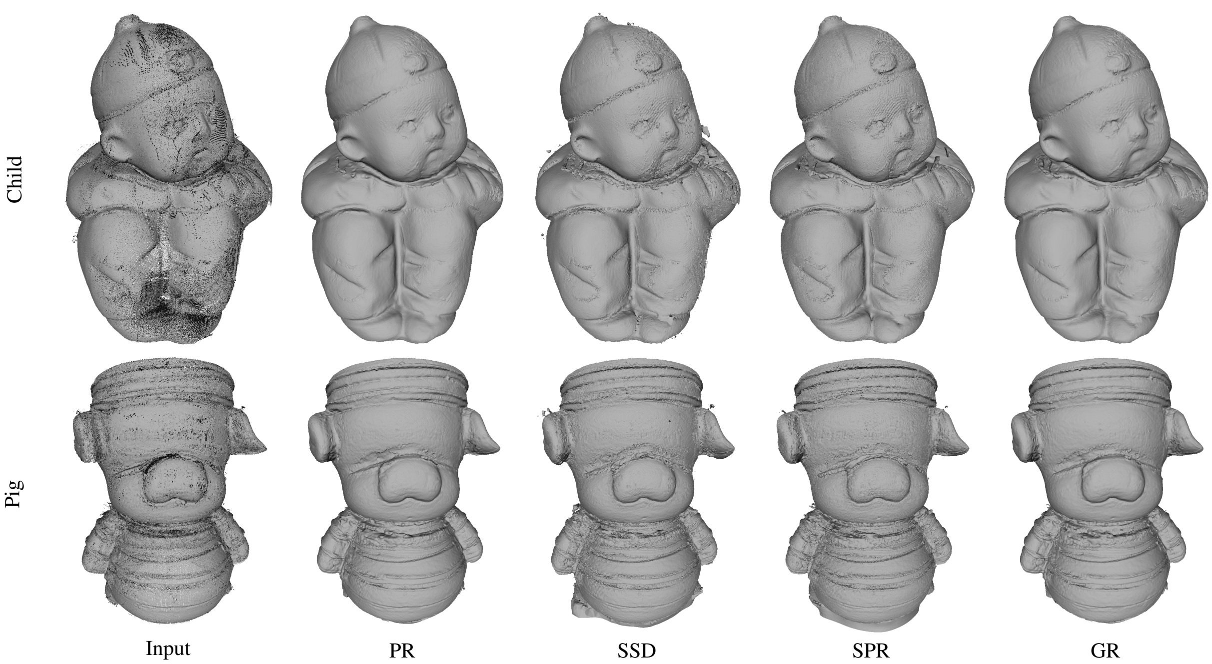


Figure 11: *The reconstructions of real-world scanned data.*

4.3 Efficiency

In this subsection, we show the efficiency of our Gauss reconstruction, particularly its parallel implementation. The running time shown in Table 1 excludes the time for data input/output.

As Table 1 shows, PR (version 3.0) is the slowest method among the four reconstructions. In the single thread implementation, SSD (version 3.0) is the fastest mainly because of the employment of hash octree, and our GR is comparable to that of SPR (version 8.0). Note that the current implementation of PR, SPR and GR does not

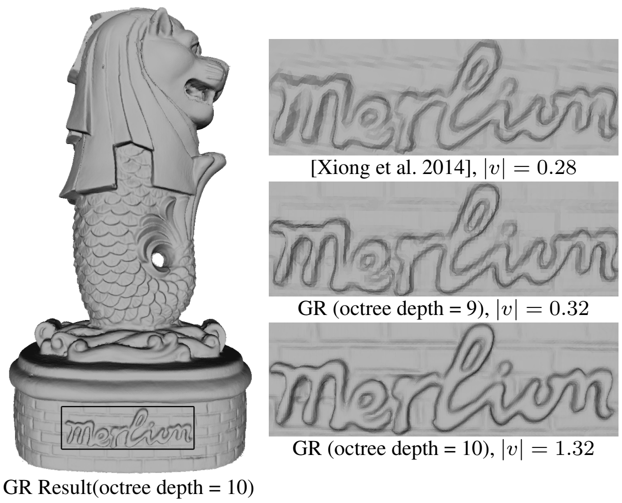


Figure 12: *The reconstructed model from merlion by [Xiong et al. 2014] and GR of different octree depth. $|v|$ denotes the number of vertices in millions of the reconstructed mesh*

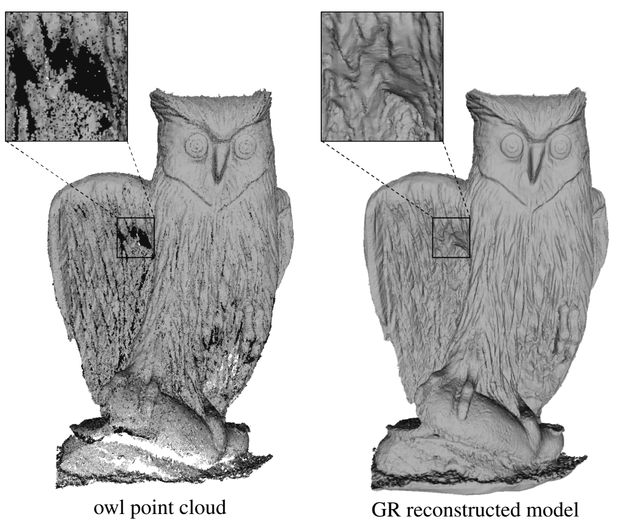


Figure 13: *The reconstructed model from owl point cloud with large missing data.*

Model	Cores	Time in Seconds			
		PR	SSD	SPR	GR
Grog	CPU 1 core	178.68	59.44	133.68	112.71
	CPU 10 cores	–	–	27.48	13.04
	GPU	–	–	–	3.052
Bimba	CPU 1 core	62.19	35.04	73.15	40.01
	CPU 10 cores	–	–	15.46	4.68
	GPU	–	–	–	1.77
Pig	CPU 1 core	169.64	58.16	116.69	111.81
	CPU 10 cores	–	–	20.93	12.79
	GPU	–	–	–	4.59
Child	CPU 1 core	135.51	50.44	105.24	82.54
	CPU 10 cores	–	–	18.67	9.44
	GPU	–	–	–	3.47

Table 1: *Running time on different models. The output mesh vertices in million: Grog: PR(3.26), SSD(2.55), SPR(3.56), GR(3.28); Bimba: PR(1.11), SSD(1.40), SPR(1.97), GR(1.11); Pig: PR(2.10), SSD(1.96), SPR(2.64), GR(2.12); Child: PR(1.88), SSD(1.87), SPR(2.33), GR(1.88).*

use hash octree.

For the multi-threads implementation, we can see from Table 1 that the parallel implementation of our GR has almost negligible overhead and achieves a nearly perfect linear speedup. In addition, the GPU implementation performs even better, almost 25 times quicker than the speed with single thread. In Table 1, we also show the running time of the parallel implementation of SPR, which is available to the public. As can be seen, GR is about twice as fast as SPR.

5 Conclusions

We have presented a surface reconstruction method called Gauss surface reconstruction. Our method is based on the implicit function that combines the near-surface signed distance function and off-surface indicator function. Thus, Gauss surface reconstruction enjoys the following benefits of both methods: resiliency against noise and missing data, free of spurious sheets, and easy recovery to the zero-level-set of the surface. Moreover, our reconstruction function is estimated directly based on Gauss Lemma without solving any linear system. This direct approach, aided by disk integration and FMM, makes our Gauss surface reconstruction simple, accurate, and easy to achieve parallel implementation. Therefore, the proposed method is very efficient.

In the future, we will try to further speed up the algorithm by using hash octree or achieve better implementation of FMM algorithm on GPU. In addition, we plan to study the theoretical property of Gauss reconstruction to particularly analyze both position approximation error and normal approximation error.

References

ALLIEZ, P., COHEN-STEINER, D., TONG, Y., AND DESBRUN, M. 2007. Voronoi-based variational reconstruction of unoriented point sets. In *Symposium on Geometry processing*, vol. 7, 39–48.

AMENTA, N., AND KIL, Y. J. 2004. Defining point-set surfaces. *ACM Trans. Graph. 23*, 3 (Aug.), 264–270.

AMENTA, N., BERN, M., AND KAMVYSSELIS, M. 1998. A new voronoi-based surface reconstruction algorithm. In *Proceedings of SIGGRAPH '98*, ACM, 415–421.

AMENTA, N., CHOI, S., AND KOLLURI, R. K. 2001. The power crust, unions of balls, and the medial axis transform. *Computational Geometry 19*, 2, 127–153.

AMENTA, N., CHOI, S., DEY, T. K., AND LEEKHA, N. 2002. A simple algorithm for homeomorphic surface reconstruction. *Internat. J. Comput. Geom. & Applications 12*, 125–141.

BERGER, M., LEVINE, J. A., NONATO, L. G., TAUBIN, G., AND SILVA, C. T. 2013. A benchmark for surface reconstruction. *ACM Trans. Graph. 32*, 2 (Apr.), 20:1–20:17.

BERGER, M., TAGLIASACCHI, A., SEVERSKY, L., ALLIEZ, P., LEVINE, J., SHARF, A., AND SILVA, C. 2014. State of the art in surface reconstruction from point clouds. In *EUROGRAPHICS star reports*, vol. 1, 161–185.

BOISSONNAT, J.-D., AND OUDOT, S. 2005. Provably good sampling and meshing of surfaces. *Graph. Models 67*, 5 (Sept.), 405–451.

CALAKLI, F., AND TAUBIN, G. 2011. Ssd: Smooth signed distance surface reconstruction. *Computer Graphics Forum 30*, 7, 1993–2002.

CARR, J. C., BEATSON, R. K., CHERRIE, J. B., MITCHELL, T. J., FRIGHT, W. R., MCCALLUM, B. C., AND EVANS, T. R. 2001. Reconstruction and representation of 3d objects with radial basis functions. In *Proceedings of SIGGRAPH '01*, ACM, 67–76.

CHEN, J., GUENNEBAUD, G., BARLA, P., AND GRANIER, X. 2013. Non-oriented mls gradient fields. In *Computer Graphics Forum*, vol. 32, Wiley Online Library, 98–109.

CIGNONI, P., ROCCHINI, C., AND SCOPIGNO, R. 1998. Metro: measuring error on simplified surfaces. *Computer Graphics Forum 17*, 2, 167–174.

CURLESS, B., AND LEVOY, M. 1996. A volumetric method for building complex models from range images. In *Proceedings of SIGGRAPH '96*, ACM, 303–312.

DEY, T. K., AND GOSWAMI, S. 2004. Provable surface reconstruction from noisy samples. In *Proceedings of SCG '04*, ACM, 330–339.

DEY, T. K., AND SUN, J. 2005. An adaptive mls surface for reconstruction with guarantees. In *Proceedings of SGP '05*, Eurographics Association, 43–52.

EDELSBRUNNER, H. 2003. Surface reconstruction by wrapping finite sets in space. In *Discrete and Computational Geometry*, vol. 25 of *Algorithms and Combinatorics*. Springer Berlin Heidelberg, 379–404.

FUHRMANN, S., AND GOESELE, M. 2014. Floating scale surface reconstruction. *ACM Trans. Graph. 33*, 4 (July), 46:1–46:11.

GIESEN, J., AND JOHN, M. 2008. The flow complex: A data structure for geometric modeling. *Computational Geometry 39*, 3, 178–190.

GREENGARD, L., AND ROKHLIN, V. 1987. A fast algorithm for particle simulations. *Journal of computational physics 73*, 2, 325–348.

HOPPE, H., DEROSE, T., DUCHAMP, T., MCDONALD, J., AND STUETZLE, W. 1992. Surface reconstruction from unorganized points. In *Proceedings of SIGGRAPH '92*, ACM, 71–78.

JACOBSON, A., KAVAN, L., AND SORKINE-HORNUNG, O. 2013. Robust inside-outside segmentation using generalized winding numbers. *ACM Trans. Graph. 32*, 4 (July), 33:1–33:12.

KAZHDAN, M., AND HOPPE, H. 2013. Screened poisson surface reconstruction. *ACM Trans. Graph. 32*, 3 (July), 29:1–29:13.

KAZHDAN, M., BOLITHO, M., AND HOPPE, H. 2006. Poisson surface reconstruction. In *Proceedings of SGP '06*, Eurographics Association, 61–70.

KAZHDAN, M., KLEIN, A., DALAL, K., AND HOPPE, H. 2007. Unconstrained isosurface extraction on arbitrary octrees. In *Symposium on Geometry Processing*, vol. 7.

KOLLURI, R., SHEWCHUK, J. R., AND O'BRIEN, J. F. 2004. Spectral surface reconstruction from noisy point clouds. In *Proceedings of SGP '04*, ACM, 11–21.

LEVIN, D. 1998. The approximation power of moving least-squares. *Math. Computation 67*, 1517–1531.

LORENSEN, W. E., AND CLINE, H. E. 1987. Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of SIGGRAPH '87*, ACM, 163–169.

MURAKI, S. 1991. Volumetric shape description of range data using "blobby model". In *Proceedings of SIGGRAPH '91*, ACM, 227–235.

SCHAEFER, S., AND WARREN, J. 2004. Dual marching cubes: Primal contouring of dual grids. In *Computer Graphics and Applications, 2004. PG 2004. Proceedings. 12th Pacific Conference on*, IEEE, 70–76.

WALDER, C., CHAPELLE, O., AND SCHÖLKOPF, B. 2005. Implicit surface modelling as an eigenvalue problem. In *Proceedings of ICML '05*, ACM, 936–939.

WENDLAND, W. 2009. On the double layer potential. *Analysis, Partial Differential Equations and Applications: The Vladimir Mazya Anniversary Volume*, 319–334.

WILHELMS, J., AND VAN GELDER, A. 1992. Octrees for faster isosurface generation. *ACM Trans. Graph. 11*, 3 (July), 201–227.

XIONG, S., ZHANG, J., ZHENG, J., CAI, J., AND LIU, L. 2014. Robust surface reconstruction via dictionary learning. *ACM Trans. Graph. 33*, 6 (Nov.), 201:1–201:12.