Point normal orientation and surface reconstruction by incorporating isovalue constraints to Poisson equation

Dong Xiao[a,c], Zuoqiang Shi[b,d,*], Siyu Li[a,c], Bailin Deng[e], Bin Wang[a,c,**]

[a] *School of Software, Tsinghua University, Beijing, China*
[b] *Yau Mathematical Sciences Center, Tsinghua University, Beijing, China*
[c] *Beijing National Research Center for Information Science and Technology, Beijing, China*
[d] *Yanqi Lake Beijing Institute of Mathematical Sciences and Applications, Beijing, China*
[e] *School of Computer Science and Informatics, Cardiff University, Wales, UK*

Abstract

Oriented normals are common pre-requisites for many geometric algorithms based on point clouds, such as Poisson surface reconstruction. However, it is not trivial to obtain a consistent orientation. In this work, we bridge orientation and reconstruction in the implicit space and propose a novel approach to orient point cloud normals by incorporating isovalue constraints to the Poisson equation. In implicit surface reconstruction, the reconstructed shape is represented as an isosurface of an implicit function defined in the ambient space. Therefore, when such a surface is reconstructed from a set of sample points, the implicit function values at the points should be close to the isovalue corresponding to the surface. Based on this observation and the Poisson equation, we propose an optimization formulation that combines isovalue constraints with local consistency requirements for normals. We optimize normals and implicit functions simultaneously and solve for a globally consistent orientation. Thanks to the sparsity of the linear system, our method can work on an average laptop with reasonable computational time. Experiments show that our method can achieve high performance in non-uniform and noisy data and manage varying sampling densities, artifacts, multiple connected components, and nested surfaces. The source code is available at `https://github.com/Submanifold/IsoConstraints`.

Keywords: Point normal orientation, Surface reconstruction, Unoriented point clouds, Poisson equation

1. Introduction

Point sets with consistently oriented normals are pre-requisites for many applications in computer graphics, such as modeling, rendering, and reconstruction (Schertler et al., 2017; Metzer et al., 2021). Although it is easy to obtain unoriented normals via local plane or surface fitting (Pearson, 1901; Cazals and Pouget, 2005), globally consistent orientation—where all normals point to the same side of the surface—cannot be completely determined locally. This makes normal orientation a challenging task in geometric modeling.

Many traditional orientation approaches focus on local consistency, i.e., adjacent points should have similar orientations (Hoppe et al., 1992; Xie et al., 2003; König and Gumhold, 2009). This is typically achieved by propagating the normal orientation along a minimum spanning tree. However, the propagation process can be easily affected by noise or sharp features. Moreover, local errors may be propagated to large areas due to the lack of global supervision. Wang et al. (2012) and Schertler et al. (2017) formulate the orientation process as an energy optimization problem. Although the minimization is globally optimal, the constraints are mainly dominated by local consistency. When the orientation of a region is inconsistent with

*Corresponding author at: Yau Mathematical Sciences Center, Tsinghua University, Beijing, China
**Corresponding author at: School of Software, Tsinghua University, Beijing, China
Email addresses: `xiaod18@mails.tsinghua.edu.cn` (Dong Xiao), `zqshi@tsinghua.edu.cn` (Zuoqiang Shi), `lisiyu21@mails.tsinghua.edu.cn` (Siyu Li), `DengB3@cardiff.ac.uk` (Bailin Deng), `wangbins@tsinghua.edu.cn` (Bin Wang)

the surrounding regions, the energy function only increases near the boundary of this region. Therefore, local errors may still cause large-scale inconsistent orientation. Given that local consistency is usually not sufficient to guarantee robustness, some approaches consider other types of conditions. Katz et al. (2007) check the visibility of points and achieve view-dependent orientation and reconstruction. Metzer et al. (2021) propose a dipole propagation scheme and orient point patches in a global electric dipole field. Robust global orientation remains a non-trival problem with room for improvements.

The problem of implicit unoriented reconstruction, where an implicit surface is reconstructed from a point cloud without normals, is also closely related to normal orientation. It is pointed out in Mullen et al. (2010) that the two tasks are of nearly the same difficulties. Some remarkable unoriented reconstruction approaches have appeared in recent years such as PGR (Lin et al., 2022) and iPSR (Hou et al., 2022). In this work, we mainly focus on bridging orientation and reconstruction in the implicit space.

In implicit surface reconstruction, the surface is expressed as the isosurface of an implicit function in the ambient space. For consistency between the extracted surface and the point cloud shape, the function values at the sample points should be close to the isovalue corresponding to the surface. We call this property the *isovalue constraints*. In this work, we incorporate isovalue constraints to the Poisson equation and propose a new optimization approach that solves for normals and the implicit function in one linear equation.

The Poisson equation gives the relationship between the implicit function and the vector field generated by the point normals (Kazhdan et al., 2006; Kazhdan and Hoppe, 2013). Therefore, we aim to search for a globally consistent normal orientation so that the sample points satisfy the isovalue constraints. Given a set of sample point positions, we optimize their normals and the implicit function values simultaneously with an appropriate boundary condition. We minimize a target function that enforces the isovalue constraints, the Poisson equation, as well as the local consistency of normals. To numerically solve the optimization, we discretize the Poisson equation and transform the minimization into a sparse linear least squares problem. To the best of our knowledge, this is the first work that optimizes both the implicit function and normals in a single target function on the Poisson reconstruction framework.

We conduct experiments on a variety of datasets with both synthetic and real scanned point clouds. The results show that our method can achieve competitive performance on a common laptop CPU, and produce robust global orientation on datasets with different sampling densities, noise levels, thin structures, and sharp features. It is also suitable for surfaces with nested structures and multiple connected components. Moreover, we demonstrate that our approach can be effectively extended to large-scale data with millions of points by globally subsampling representative point sets and orienting the dense point cloud normals based on the implicit field generated by the reference set.

2. Related works

Although unoriented normals of point clouds can be directly estimated by local fitting using PCA (Pearson, 1901) or Jets (Cazals and Pouget, 2005), it is difficult to determine whether the normal is pointing inward or outward of the surface via local properties only (Metzer et al., 2021). Hence, deriving a consistent orientation of point cloud normals is a challenging problem. In this section, we review several existing techniques on normal orientation and implicit unoriented reconstruction. The techniques are classified into two categories: (1) orientation based on local consistency, and (2) orientation dominated by other factors. When classifying a method into the second category, we do not mean completely ignoring local properties, but mean applying additional supervision to ensure normal consistency. Unoriented reconstruction approaches are generally classified into the second category.

2.1. Orientation based on local consistency

The main idea of local consistency-based approaches is to propagate the known orientations to neighboring points. The seminal work in Hoppe et al. (1992) constructs a minimum spanning tree (MST) based on the similarity between adjacent normals and propagates the orientations along the tree. However, noise and sharp features may cause incorrect propagation paths. In addition, local errors may be propagated to larger areas and cause severe degradation of the performance. Although several improved flip criterion methods

have been proposed (Xie et al., 2003; König and Gumhold, 2009), their robustness is still unsatisfactory for varying inputs. Some methods formulate orientation as a global optimization problem instead. Wang et al. (2012) minimize a combination of the Dirichlet energy and the coupled-orthogonality deviation to ensure that the normals are consistent with the adjacent points and perpendicular to the surface. Schertler et al. (2017) formulate normal propagation as a graph-based energy minimization problem and solve it by quadratic pseudo-Boolean optimization (QPBO) (Boros et al., 1991). Jakob et al. (2019) propose a parallel greedy solver on the GPU for graph-based minimization. The objective functions of these methods are still dominated by local consistency items. When inconsistent orientation occurs in a region, the energy function only increases near the region boundary. Thus, local errors may still cause large-scale inconsistent orientation due to the lack of global supervision. Moreover, these approaches may fail when orienting point clouds with multiple connected components or nested structures.

2.2. Implicit unoriented reconstruction and orientation based on other factors

Since local consistency is usually not sufficient to obtain robust orientation for varying inputs, some works apply other strategies to determine orientation. Mello et al. (2003) construct a signed distance function defined on a simplicial decomposition of a bounding box, where the sign of the distance value indicates the in/out information with respect to the input point set. Xie et al. (2004) cluster the point set into singly-oriented groups and determine the global orientation by the active contours. Katz et al. (2007) utilize a visibility-based heuristic to recognize outer surfaces and generate view-dependent orientation and reconstruction. Alliez et al. (2007) build a tensor field from the Voronoi diagram and solve for an implicit function whose gradient aligns with the principal axes of the field. Mullen et al. (2010) obtain a signed distance function from an unsigned distance approximation by ray shooting and sign propagation.

In recent years, researchers have made considerable progress in implicit unoriented reconstruction. VIPSS (Huang et al., 2019) sets up gradient norm constraints of the signed distance function and solves it by Duchon's energy. This method achieves good performance on sparse samples and 3D sketches. However, the cubic time and space complexity restrict its application on larger-scale problems. PGR (Lin et al., 2022) regards normals and surface elements in the Gauss formula as unknowns and optimizes the parametric function space. However, it involves a dense linear system that needs to be solved on the GPU to achieve reasonable efficiency, which makes it impractical for large-scale problems. By contrast, our method only needs to solve a sparse linear system and can work on a common laptop CPU with high orientation accuracy and scalability. iPSR (Hou et al., 2022) runs Poisson reconstruction in an iterative manner and updates normals using the generated surface of the last iteration. Although it exhibits outstanding performance and generates high-fidelity results, it currently lacks a theoretical guarantee for convergence. Different from iPSR, our method establishes an optimization problem in the implicit function space, which can be solved using a single sparse linear system. Our method also performs well on thin structures with noise.

In addition to traditional approaches, deep neural networks have been applied to gather information of different scales for orientation and reconstruction (Guerrero et al., 2018; Park et al., 2019; Erler et al., 2020; Xiao et al., 2022; Wang et al., 2022). Among them, Wang et al. (2022) utilize local patches for unoriented normal estimation and apply the global features for consistent orientation. However, these methods usually require substantial training data. Some works learn the implicit function directly from the input point cloud and eliminate the need for training data (Atzmon and Lipman, 2020; Gropp et al., 2020; Ma et al., 2021; Wang et al., 2021; Ben-Shabat et al., 2022). However, they need to operate an individual network for each shape. Peng et al. (2021) propose a differentiable Poisson solver for both optimization-based and learning-based unoriented reconstruction. Different from other approaches, the point positions are also optimized in their work. Metzer et al. (2021) apply deep networks for local patch orientation and propose a dipole propagation strategy across the coherent patches to achieve global consistency. However, the patch partition strategy may affect the robustness of this method, especially on thin structures.

3. Method

In this section, we present our optimization formulation that bridges orientation and reconstruction in the implicit function space. We treat normals as variables and optimize them such that the isosurface of the

implicit function is consistent with the sample points, i.e., the function values at the sample points are close to the isovalue, which forms the isovalue constraints. Our method incorporates the isovalue constraints to the discretized Poisson equation, which forms the target function together with the local consistency energy. The resulting formulation optimizes the implicit function and the point normals simultaneously, which can be solved using a sparse linear least squares system.

3.1. Discretizing the Poisson equation

To formulate our optimization for both the implicit function and the normals, we need to first establish a relationship between them. In this work, we model their correlation using a discretized Poisson equation similar to the one used in Poisson surface reconstruction (PSR) (Kazhdan et al., 2006; Kazhdan and Hoppe, 2013). In the following, we first provide a brief review of PSR as preliminaries.

Given the oriented sample points $\mathcal{S} = \{(p_i, n_i)_{i=1}^N\}$ where $p_i \in \mathbb{R}^3$ is the point position and $n_i \in \mathbb{R}^3$ is the normal, PSR first builds an octree $\mathcal{O}$ from the point cloud. Then a vector field $\vec{V}$ is computed from the normals as

$$\vec{V}(q) = \sum_{s_i=(p_i,n_i)\in\mathcal{S}} \tilde{F}_{s_i}(p_i, q) \cdot n_i \tag{1}$$

with

$$\tilde{F}_{s_i}(p_i, q) = \frac{1}{W(p_i)} \sum_{o\in\mathcal{N}_D(s_i)} \alpha_{o,s_i} F_o(q), \tag{2}$$

where $\mathcal{N}_D(s_i)$ denotes the set of neighbor nodes for s_i in the octree of the same depth D. α_{o,s_i} is the coefficient for trilinear interpolation. $F_o(q)$ is an approximated Gaussian smooth function, whose value decreases when o and q are far apart. $W(p_i)$ is related to the local sampling density. For more details, please refer to Section 4 of Kazhdan et al. (2006). In general, the vector field of a point $q \in \mathbb{R}^3$ is dominated by the normals of its nearby sample points. In the inset figure below, we illustrate the vector field generated by a set of samples. Here the red points represent the sample points, whereas the blue arrows indicate the generated vector field.

We can express Equation 1 in a matrix form by concatenating the normals into a vector $n = (n_1^T, n_2^T, \ldots, n_N^T)^T \in \mathbb{R}^{3N}$, such that there exists a $3 \times 3N$ matrix $\mathscr{F}(q)$ where

$$\vec{V}(q) = \mathscr{F}(q)n. \tag{3}$$

We can ignore the small entries of $\mathscr{F}(q)$ due to the locality of the Gaussian function, so that $\mathscr{F}(q)$ becomes a sparse matrix. Equation 3 indicates that the vector field can be expressed as a linear function of the point normals.

The indicator function $\chi(q)$ is a typical implicit function with values of 0 outside and 1 inside the surface. By aligning the gradient of the indicator function $\chi(q)$ with the vector field $\vec{V}$, the indicator function and the vector field should satisfy the Poisson equation

$$\Delta\chi = \nabla \cdot \vec{V}. \tag{4}$$

In PSR, the indicator function is represented as a linear combination of a set of compactly supported B-spline basis functions, where each octnode $o \in \mathcal{O}$ is assigned with a basis at the node center, i.e.,

$$\chi(q) = \sum_{o\in\mathcal{O}} x_o \mathscr{B}_o(q), \tag{5}$$

where $\{x_o\}$ are the interpolation coefficients to be solved, and $\mathscr{B}_o(q)$ is a three-dimensional B-spline basis function of degree two centered at o. In addition, the Poisson equation 4 is also projected onto a series of basis functions. This results in a sparse linear system

$$Ax = b, \tag{6}$$

where x concatenates the coefficients $\{x_o\}$ for the basis functions, and the matrix A and the vector b has components

$$A_{ij} =< \nabla\mathscr{B}_i, \nabla\mathscr{B}_j >_{[0,1]^3}, \quad b_i =< \nabla\mathscr{B}_i, \vec{V} >_{[0,1]^3} . \tag{7}$$

For details of the projection process, please refer to Section 3 of Kazhdan and Hoppe (2013).

Different from PSR which solves the Equation 6 to obtain the coefficients x for the indicator function, we treat both x and the normals n as unknowns and use Equation 6 to derive their relation. To this end, we note that the vector field $\vec{V}$ is represented as a linear function of the normals in Equation 3. Therefore,

$$b_i =< \nabla\mathscr{B}_i, \vec{V} >= \int_{[0,1]^3} (\nabla\mathscr{B}_i^T(q)\vec{V}(q))\ dq = [\int_{[0,1]^3} (\nabla\mathscr{B}_i^T(q)\mathscr{F}(q))\ dq]\ n. \tag{8}$$

Using this condition, we can write the vector b as a linear function of n:

$$b = Bn, \tag{9}$$

where B is a sparse matrix due to the locality of $\mathscr{B}_i$ and $\mathscr{F}$. It follows that the coefficients x and the normals n should satisfy the relation

$$Ax = Bn. \tag{10}$$

This condition will be used to formulate the objective function of our optimization problem.

3.2. Formulation of the objective function

In this section, we formulate the objective function of our optimization for the indicator function coefficients x and the point normals n by enforcing a set of constraints that they should satisfy.

3.2.1. Poisson equation constraints

In the previous subsection, we use the Poisson equation to derive the relation 10 between the the indicator function and the normals. As we treat both the coefficients x and the normals n as variable, the condition 10 needs to be enforced by our optimization. To this end, we introduce the following term into the objective function to penalize the violation of the condition:

$$E_{Poi}(x, n) = ||Ax - Bn||^2. \tag{11}$$

3.2.2. Isovalue constraints

In implicit reconstruction, the reconstructed surface corresponds to an isosurface of the indicator function. Given a point cloud that is sampled from the surface, the indicator function values at the sample points should be close to the isovalue of the surface. In this way, the extracted isosurface is consistent with the point shape. Therefore, we enforce such isovalue constraints in our optimization. Our goal is to make the indicator function equal to 0 outside the surface and 1 inside the surface. If the normals are well oriented, then the indicator function value changes rapidly from 0 to 1 near the surface. Thus, we choose the isovalue to be the midpoint $\frac{1}{2}$, and use it as the indicator function value at the sample points. Then the isovalue constraints can be written as

$$\chi(p_i) = \sum_{o\in\mathcal{O}} x_o\mathscr{B}_o(p_i) = \frac{1}{2}, \quad i = 1, 2, ..., N. \tag{12}$$

This can be written in a matrix form

$$Ux = \frac{1}{2}\vec{1}, \tag{13}$$

where the matrix $U \in \mathbb{R}^{N\times|\mathcal{O}|}$ has coefficients $u_{ij} = \mathscr{B}_j(p_i)$ (i is index of the sample point and j is index of the sorted octnode $o \in \mathcal{O}$), and $\vec{1}$ is a vector whose components are all 1s. Note that the B-spline basis

functions are compactly supported, U is a sparse matrix. Then we enforce the isovalue constraints via the following term in the objective function:

$$E_{iso}(x) = ||Ux - \frac{1}{2}\vec{1}||^2. \tag{14}$$

Remark. In screened Poisson surface reconstruction (SPSR) (Kazhdan and Hoppe, 2013), point constraints are introduced in their formulation to fit the isovalue. The major difference between the isovalue constraints in method and the point constraints in SPSR is that SPSR requires point normals as input, while our method treats normals as variables. Therefore, the isovalue constraints of our approach can be spread to normals during optimization with the bridging of Equation 11.

The isovalue constraints and the aforementioned Poisson equation constraints in Section 3.2.1 are basic terms of our formulation. They allow us to search for a globally consistent normal orientation such that the indicator function values of Poisson reconstruction are equal to the isovalue on the sample points.

3.2.3. Local consistency constraints

Our formulation also enforces the consistency between normals at adjacent points. Inspired by Wang et al. (2012), we adopt a Dirichlet energy term $E_D(n)$ to penalize the difference between adjacent normals, and a coupled-orthogonality deviation term $E_{COD}(n)$ to enforce the orthogonality condition between the normal vectors and the underlying surface. First, we construct a graph $\mathcal{E}$ for the sample points using their k-nearset neighbors (KNN), where two points i and j are connected with an edge when i is in the KNN of j or j is in the KNN of i (we set $k = 10$ in our experiments). Using this graph, we define the terms $E_D(n)$ and $E_{COD}(n)$ as follows:

$$E_D(n) = \frac{1}{2}\sum_i \sum_{j\in\mathcal{N}(i)} w_{ij}||n_i - n_j||^2, \tag{15}$$

$$\begin{aligned} E_{COD}(n) &= \sum_i \sum_{j\in\mathcal{N}(i)} w_{ij}[(\frac{p_j - p_i}{||p_j - p_i||})^T(n_i + n_j)]^2 \\ &= \sum_i \sum_{j\in\mathcal{N}(i)} w_{ij}[(n_i + n_j)^T \frac{(p_j - p_i)(p_j - p_i)^T}{||p_j - p_i||^2}(n_i + n_j)], \end{aligned} \tag{16}$$

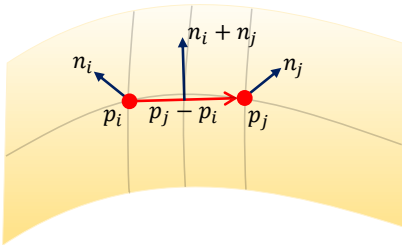
where

$$w_{ij} = \exp\left(-\frac{||p_i - p_j||}{\rho^2}\right), \quad \rho = \max_{(p_i,p_j)\in\mathcal{E}} \frac{||p_i - p_j||}{2}. \tag{17}$$

The rationale for the term $E_{COD}(n)$ can be explained as follows. Suppose p_i and p_j are two nearby points on a quadratic surface patch and their geodesic midpoint is the center of the surface patch (see the inset figure below). Then their normal vectors n_i and n_j should satisfy (Rusinkiewicz, 2019)

$$(\frac{p_j - p_i}{||p_j - p_i||})^T(n_i + n_j) = 0. \tag{18}$$

Since a smooth surface can be locally approximated around each point using a quadratic height-field surface over the tangent plane at the point, two nearby points on the surface and their normals should satisfy condition 18 approximately. Thus we use the term $E_{COD}(n)$ to enforce this condition.



We introduce the local consistency energy $E_{loc}(n)$ as the sum of $E_D(n)$ and $E_{COD}(n)$. Similar to Lin et al. (2022), we apply the L2 regularizer to ensure the stability of optimization and avoid oscillation especially for noisy inputs. Therefore,

$$E_{loc}(n) = E_D(n) + E_{COD}(n) + n^T n. \tag{19}$$

From Equation 15 and 16, we notice that $E_D(n)$ and $E_{COD}(n)$ are quadratic functions of $n = (n_1^T, n_2^T, \ldots, n_N^T)^T \in \mathbb{R}^{3N}$. Thus, there exists a matrix M such that

$$E_{loc}(n) = n^T M n. \tag{20}$$

M is sparse due to the locality of nearest neighbors.

Local consistency constraints mainly act as a regularizer and do not dominate the optimization. Decent reconstructions can also be achieved without the local consistency energy. However, enforcing local consistency is beneficial for our method especially for noisy inputs. We will show that they improves the reconstruction quality both qualitatively and quantitatively in the experiments.

3.2.4. Boundary condition

Combining Equation 11, 14 and 20, the overall objective function $E(x, n)$ can be written as

$$E(x, n) = E_{iso}(x) + \alpha E_{Poi}(x, n) + \beta E_{loc}(n), \tag{21}$$

where α and β are weights specified by the user. During experiments, the parameter α keeps unchanged. β differs for noisy and noise-free inputs and is relatively small, mainly to maintain stability and avoid oscillation. The parameters do not need to be adjusted frequently.

We want the indicator function to be 0 outside the surface, $\frac{1}{2}$ on the surface, and 1 inside the surface. Therefore, the overall objective function $E(x, n)$ in Equation 21 must be minimized subject to a Dirichlet boundary condition, that is, the indicator value on the bounding box should be 0. Otherwise, a trivial solution $\chi(x) = \frac{1}{2}, \forall x \in \mathbb{R}^3$ and $n = \vec{0}$ exists. We set the bounding box to be slightly larger than the original object. Let $\mathcal{B}$ be the octnodes of the border, we force the coefficients of the basis functions at $\mathcal{B}$ to be 0, that is

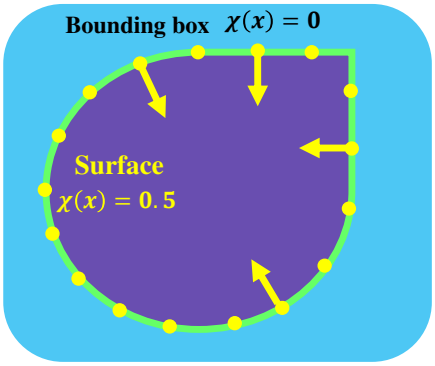
$$\begin{aligned} &min_{x,n}\ E(x, n) \\ &\text{s.t. } x_i = 0, i \in \mathcal{B}. \end{aligned} \tag{22}$$

According to the Poisson equation, the variation of the indicator function is mainly caused by the vector field formed by the point normals. As shown in the inset figure below, if the indicator function equals to 0 at the bounding box and 0.5 at the surface (represented by the green outline), then we can predict the normal directions to be vertical to the circle and pointing inward according to the variation of the indicator values. The indicator function transitions from 0 to 0.5, similar to "surrounding" the surface from the bounding box. Therefore, our method is good at identifying the inward (outward) normals of the outermost surfaces, which is helpful for noisy thin structures.

3.3. Solving

From Equation 21, the objective function can be expressed as follows:

$$E(x, n) = ||Ux - \frac{1}{2}\vec{1}||^2 + \alpha||Ax - Bn||^2 + \beta(n^T M n). \tag{23}$$



The boundary condition $x_i = 0, i \in \mathcal{B}$ can be enforced by crossing out the boundary components of x and the corresponding columns of matrices A and U in the equation. This optimization problem can then be transformed into an unconstrained least squares problem by $\nabla_x E(x, n) = 0$ and $\nabla_n E(x, n) = 0$. It is equivalent to solving the linear system

$$\tilde{A}\tilde{x} = \tilde{b}, \tag{24}$$

where

$$\tilde{A} = \begin{pmatrix} U^T U + \alpha A^T A & -\alpha A^T B \\ -\alpha B^T A & \alpha B^T B + \beta M \end{pmatrix}, \quad \tilde{x} = \begin{pmatrix} x \\ n \end{pmatrix}, \quad \tilde{b} = \begin{pmatrix} U^T \frac{1}{2}\vec{1} \\ 0 \end{pmatrix}. \tag{25}$$

We solve this linear system by the conjugate gradient method (CG) (Hestenes and Stiefel, 1952). U, A, B and M are all sparse matrices. It is worth noting that we do not explicitly calculate the matrix $\tilde{A}$ during the solving process because U^TU, A^TA and B^TB are not necessarily sparse. However, each step of the conjugate gradient algorithm can be written as the product of a matrix and a vector, or the inner product of two vectors. For instance, $U^TUx = (U^T(Ux)), x^TU^TUx = (Ux, Ux)$. Therefore, the sparsity of the matrices can be maintained during the solving process. For more details, please refer to Appendix A.

Let $|\mathcal{O}|$ be the number of octnodes, the size of matrix A is $|\mathcal{O}| \times |\mathcal{O}|$ and dominates the main time and space resources during calculation since $|\mathcal{O}|$ is always larger than the point number N. Fortunately, the matrix is sparse with nonzero element $O(|\mathcal{O}|log(|\mathcal{O}|))$. The reason is that the construction of A is similar to that of PSR. Thus, the number of its nonzero elements is equivalent to the complexity of executing the Algorithm 1 of Kazhdan and Hoppe (2013). The logarithm occurs due to the octree depth. For B and U, each sample point is only related to the nearby nodes and corresponding basis functions at each depth. Therefore, they will not exhibit higher complexity. When conducting multiplication of a sparse matrix and a vector, the time complexity mainly depends on the number of nonzero elements of the matrix. The time complexity of the optimization process is $O(m|\mathcal{O}|log(|\mathcal{O}|))$, where m is the number of conjugate iterations.

3.4. Orientation and reconstruction

Let $(\hat{x}, \hat{n})$ be the solution of Equation 22, due to memory and time limitation, we set the max tree depth to be 7 during optimization and perform 300 CG iterations. This may be not enough for high-fidelity reconstruction, but is sufficient for deciding the orientation of normals in most cases. Therefore, we do not apply the optimized indicator coefficients $\hat{x}$ for reconstruction. Instead, we apply the optimized normals $\hat{n}$ for orientation and input the oriented normals to screened Poisson surface reconstruction (SPSR) (Kazhdan and Hoppe, 2013) with max depth 10. SPSR uses a conforming cascadic solver depth by depth without solving the large linear system including the octnodes of all depths.

To conduct normal orientation, we treat the optimized normals $\hat{n}$ as references. For each point, we estimate its unoriented normal (no inside/outside information) $\tilde{n}_i$ by Jets (Cazals and Pouget, 2005). We also have an optimized normal $\hat{n}_i$ from Equation 22. We set the final normal prediction of this point to $\tilde{n}_i$ if $\tilde{n}_i \cdot \hat{n}_i \geq 0$ and $-\tilde{n}_i$ otherwise. In this manner, we can obtain the inward normals—the normals of the outermost surface are pointing inward. For most cases, using oriented Jets normals (i.e., Jets normals oriented by our method) results in slightly higher accuracy than directly using the optimized $\hat{n}$ because Jets method mainly focuses on local properties and usually generates accurate unoriented normals. However, in cases when the local surface fitting is inaccurate such as the extremely sparse 3D sketch inputs (the first two examples of Figure 8), we apply the optimized $\hat{n}$ (after normalization) as the normal prediction.

Our approach can support per-point optimization (i.e., feeding all the point normals for optimization) for more than 100K points in common devices. For large-scale point clouds (entailing millions of points), we can subsample a representative point set from the dense point cloud and orient the whole point cloud using the implicit field generated by the representative set. When the representative set is well oriented by our method, we run SPSR to generate the implicit function of the set. The gradient field of the implicit function can be calculated by $\nabla\chi(q) = \sum_{o\in\mathcal{O}} x_o \nabla\mathscr{B}_o(q)$. For each point $\tilde{p}_j$ in the dense point cloud with unoriented Jets normal $\tilde{n}_j$, we calculate $\nabla\chi(\tilde{p}_j) \cdot \tilde{n}_j$ to determine the normal orientation (flip the normal if the inner product is lower than 0). This implicit orientation method is simple but effective, as the optimization of Equation 22 is dominated by isovalue constraints and does not rely on density sampling when the basic shape is included. In most cases, such a scale is sufficient to contain the entire basic shape and decide the normal orientation of the dense point cloud. Then, a high-fidelity reconstruction can be achieved by the oriented dense point cloud. In addition, excessive many variables are not beneficial to carry out optimization. The use of representative sets can relax the complexity of our optimization. The representative set approach is mainly inspired by Metzer et al. (2021). The difference is that Metzer et al. (2021) utilize the dipole implicit field, whereas our approach applies the Poisson implicit field.

4. Experiments

We conduct various qualitative and quantitative experiments on orientation accuracy and reconstruction quality to demonstrate the efficacy of our approach. The experiments are performed on different datasets with varying noise, sampling densities, and artifacts.

4.1. Comparisons

We compare our method with representative and state-of-the-art orientation and unoriented reconstruction approaches. For the dataset, we use the benchmark data complied by a recent survey paper (Huang et al., 2022) that collects a large number of shapes from different repositories such as 3DNet (Wohlkinger et al., 2012), ABC (Koch et al., 2019), Thingi10K (Zhou and Jacobson, 2016), and 3D Scans (Oliver, L. et al.). The shapes are classified as simple, ordinary, or complex according to the complexity of the shape topology and the difficulties for reconstruction.

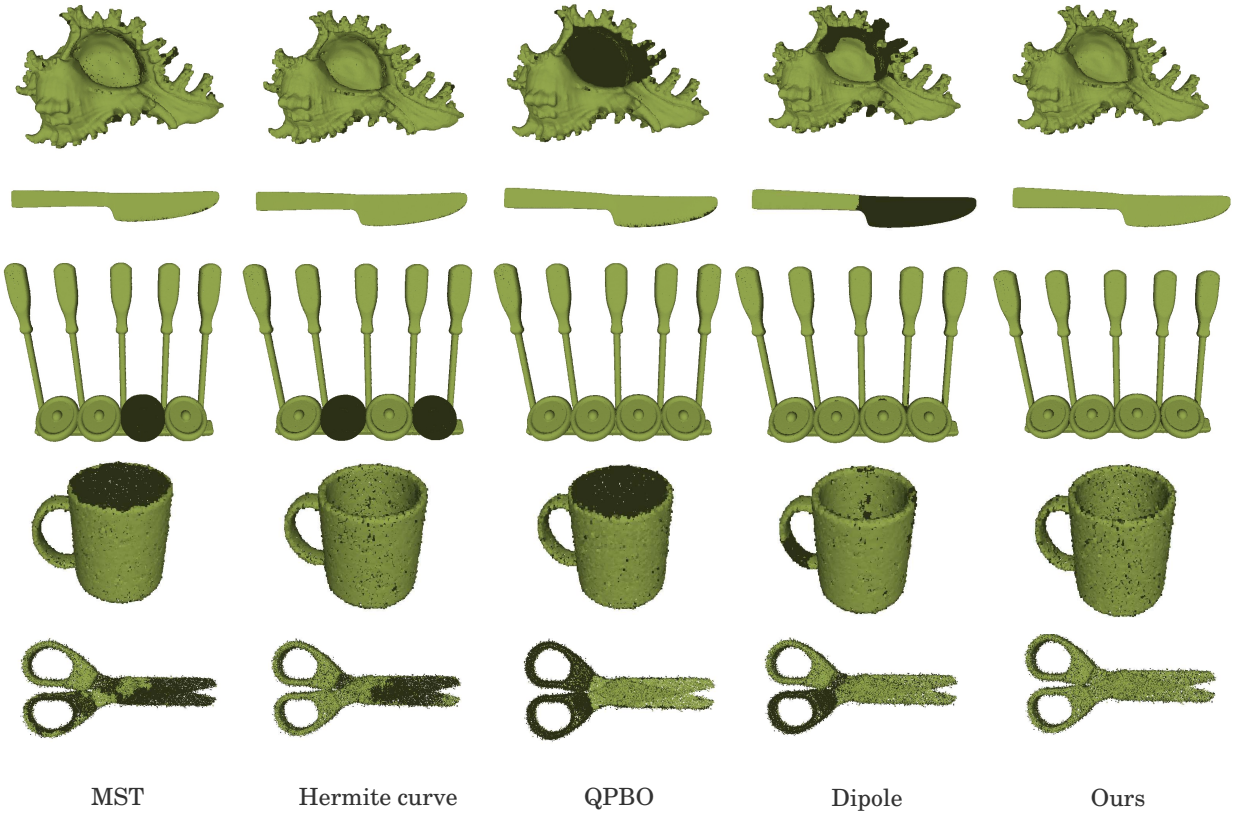


Figure 1: Qualitative comparisons with MST (Hoppe et al., 1992), Hermite curve (König and Gumhold, 2009), QPBO (Schertler et al., 2017) and Dipole (Metzer et al., 2021) in terms of orientation accuracy. Our method achieves high performance in point clouds with complex topology, thin structures, and noise.

We carry out our experiments on two different sampling styles: 1. non-uniform but nearly no noise and 2. varying noise. For non-uniform sampling, we choose the complex shapes (total 162 shapes) and directly apply the point clouds provided by the benchmark (Huang et al., 2022) with 160K points per shape. They are scanned by the Blensor simulator (Gschwandtner et al., 2011). In our approach, we use a representative set of $\frac{1}{3}$ original points (about 53K) for optimization and orient the whole point cloud according to the reference set. We set the parameters to $\alpha = 10^4$ and $\beta = 10^{-4}$ for no-noise inputs.

For var-noise sampling, we use the ordinary shapes (total 486 shapes) and directly sample 40K points from the surface by trimesh (Haggerty, D. et al., 2019). To obtain varying noise, we add randomized Gaussian

Table 1: Quantitative comparisons with MST (Hoppe et al., 1992), Hermite curve (König and Gumhold, 2009), QPBO (Schertler et al., 2017) and Dipole (Metzer et al., 2021) in terms of orientation accuracy.

Method	Complex non-uniform		Ordinary var-noise	
	Avg.	Acc.> 97%	Avg.	Acc.> 90%
MST	97.1%	87.2%	86.4%	66.6%
Hermite curve	98.6%	92.3%	90.2%	76.6%
QPBO	98.4%	89.1%	87.2%	67.9%
Dipole	95.8%	77.8%	90.1%	73.0%
Ours	**99.3%**	**96.3%**	**93.7%**	**85.2%**

noise with varying noise ratio $[0.5, 1.0]$ and varying noise standard derivation $[0.002, 0.01]$ for each shape. The noise ratio represents the proportion of noisy points among all points and is different for each shape. We conduct per-point optimization (no representative set is required) for var-noise sampling and set parameters to $\alpha = 10^4$ and $\beta = 10^{-2}$. The parameter settings are maintained throughout the experiment.

During qualitative comparison, we also select several data from Berger et al. (2013), some 3D sketches from Huang et al. (2019) and some real scanned point clouds from the aforementioned benchmark (Huang et al., 2022).

First, we compare our results with four orientation methods MST (Hoppe et al., 1992), Hermite curve (König and Gumhold, 2009), QPBO (Schertler et al., 2017) and Dipole (Metzer et al., 2021) in terms of orientation accuracy, which indicates the proportion of correctly oriented normals (where the dot product with the ground truth normal is greater than zero) among all sample points. The unoriented normals are all predicted by Jets (Cazals and Pouget, 2005). For non-uniform sampling, the ground truth normal of a point is taken as the surface normal of the ground truth mesh where the point is located. For noisy sampling, the ground truth normal of a point is taken as the normal when it is sampled. Table 1 shows the quantitative comparisons. In addition to average accuracy, we also count the proportion of shapes whose accuracy is greater than a certain threshold for each method (97% for complex non-uniform dataset and 90% for ordinary var-noise dataset). For instance, our approach achieves orientation accuracy higher than 97% in 96.3% (156/162) of the shapes in the complex non-uniform dataset. The results indicate that our method achieves high performance. Local consistency-based methods (MST, Hermite curve and QPBO) may fail in orienting point clouds with multiple connected components in one process. Therefore, these shapes are removed when calculating the accuracy of them.

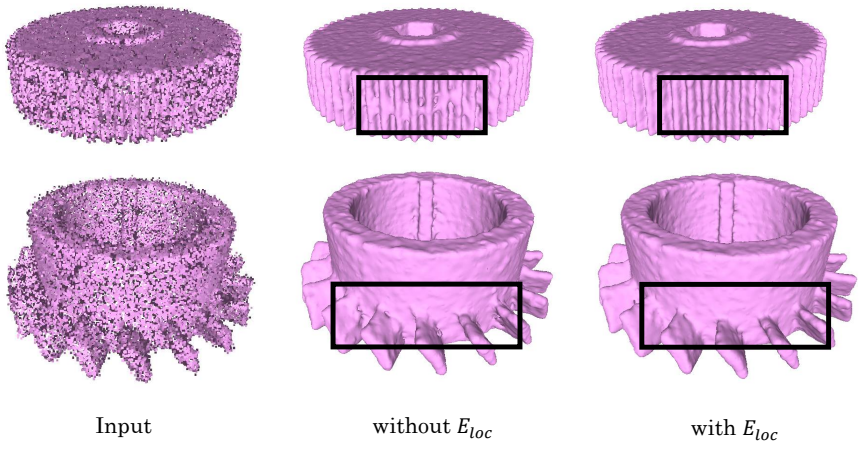


Figure 2: The reconstruction results with and without the local consistency energy E_{loc} of two noisy inputs. We use black rectangles to emphasize the differences. It can be seen that E_{loc} brings about better orientation consistency and improves the reconstruction quality near the zigzag region of the input shapes.

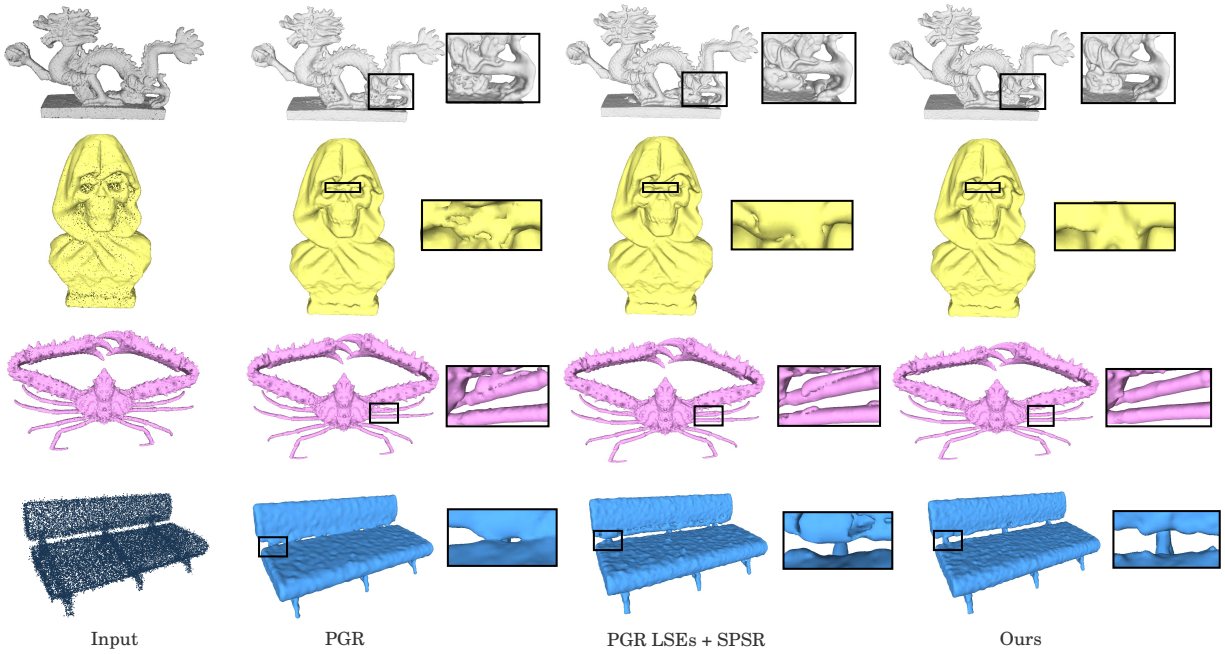


Figure 3: Qualitative comparisons with PGR (Lin et al., 2022). PGR may over-smoothen the surface in noisy inputs. Our method performs better orientation accuracy and generates more details.

The qualitative comparisons are shown in Figure 1. The oriented point clouds are visualized by MeshLab (Cignoni et al., 2008). The first three examples are noise-free, and the last two contain thin structures with noise. Our method shows good orientation accuracy. MST, Hermite curve and QPBO are local consistency-based approaches. Therefore, local errors may cause large-scale inconsistent orientation. Moreover, their performances degrade when the noise is large. Dipole separates the point clouds into patches, reducing its robustness when the wrong partition occurs, especially when thin structures exist. It is also sensitive to parameters and requires careful adjustment for complex shapes. Our approach exhibits high robustness for complex topology, thin structures, and noise.

If the L2 regularizer in Equation 19 is removed, then the orientation accuracy of our method on the ordinary var-noise dataset is reduced by 0.2%. Removing the entire E_{loc} reduces the accuracy by 0.6%. This scenario indicates that our approach is dominated by isovalue constraints and achieves decent results without the local consistency energy. Meanwhile, the local consistency energy improves the average orientation accuracy of our method. We also visually demonstrate its effectiveness in Figure 2, where we show the reconstruction results with and without the local consistency energy E_{loc} of two noisy inputs. We use black rectangles to emphasize the differences. It can be seen that E_{loc} brings about better orientation consistency and improves the reconstruction quality near the zigzag region of the input shapes.

In addition to orientation approaches, we also compare the reconstruction performance of our method with the most recent unoriented reconstruction approaches PGR (Lin et al., 2022) and iPSR (Hou et al., 2022). In iPSR (and SPSR), point weight is an important parameter that controls the degree of fitting points. We set the point weight to 10.0 for noise-free inputs and 1.0 for noisy inputs. In PGR, parameters k_w and α control its ability to manage noise. We set $k_w = 7, \alpha = 2.0$ for noise-free inputs and $k_w = 64, \alpha = 5.0$ for noisy inputs.

PGR optimizes the product of normals and surface elements (also named as *linearized surface elements* (LSEs) in Lin et al. (2022)) in the Gauss formula. The linear system of PGR is dense. Therefore, it is always executed on GPU and limits to about 50K points in 2080Ti with 11GB memory. By contrast, our approach solves a sparse linear system and can work entirely on CPU. We can carry out per-point optimization of 100K points in common devices. Moreover, our method exhibits better average orientation accuracy in complex

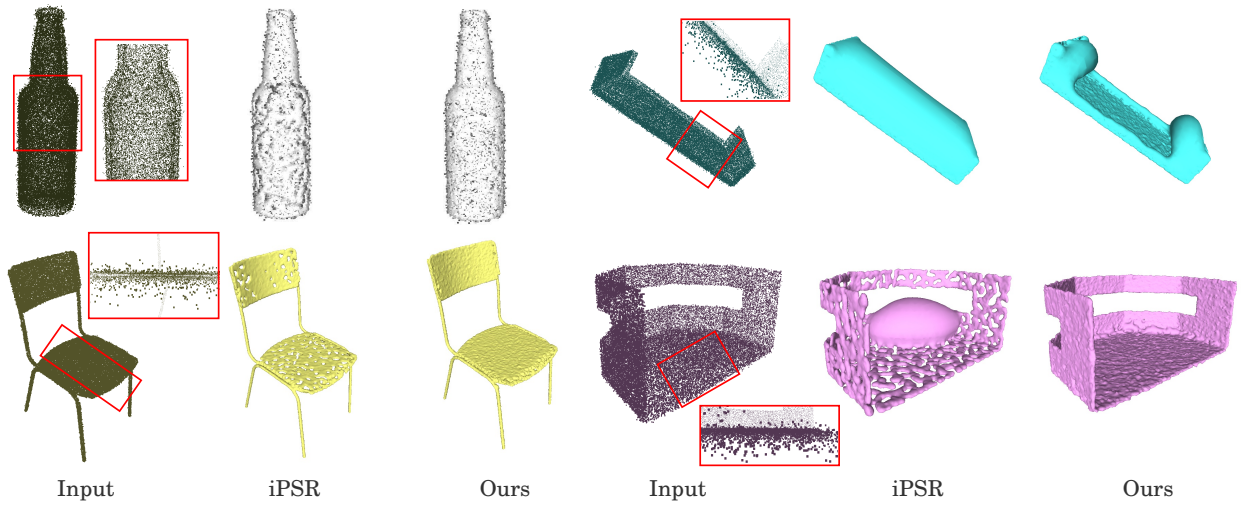


Figure 4: Qualitative comparisons with iPSR (Hou et al., 2022). Our method is good at identifying the normal orientation of the outermost surface and performs better in thin structures with noise.

Table 2: Running time of our approach on a usual laptop CPU. The time includes all IOs and reconstructions.

Model	Node count	Point number	Representative set	Running time
kitten (3rd Figure 8)	$67,753$	$5,210$	-	$23s$
bar_chair36 (5th Figure 9)	$155,953$	$40K$	-	$85s$
think10k66952 (2nd Figure 9)	$167,761$	$160K$	$53K$	$119s$
mailbox (4th Figure 9)	$209,929$	$0.7M$	$100K$	$239s$
buddha (6th Figure 8)	$267,897$	$5M$	$100K$	$456s$

non-uniform and ordinary var-noise datasets. When normalized, the LSEs in PGR can also represent normals. The average orientation accuracy of PGR is 96.4% in no-noise data and 91.9% in var-noise data, and that of our approach is 99.3% and 93.7%, respectively. In PGR, the surface elements are in the optimized variable. Moreover, several approximations are required to solve the singularity of the Gauss formula. Hence, the difference between LSEs and normals is evident. In Figure 3, we show the reconstruction of PGR and the results of taking the normalized LSEs for orientation and reconstructing by SPSR. The results show that our method generates good details, for instance, the crab legs. PGR also over-smoothens the surface for noisy inputs.

iPSR iteratively processes SPSR and computes the normals from the surface of the preceding iteration. Although iPSR is an outstanding work and always performs high-fidelity reconstructions in complex topologies, its convergence currently has no theoretical proof due to the lack of explicit formulation of isosurfacing. Our method optimizes the normals in the implicit space and represents the solution by a linear equation. In Section 3.2.4, we demonstrate that our method is good at identifying the outermost surface and the corresponding inward (outward) normals. Therefore, we can achieve good performance in thin structures with noise. This is shown in Figure 4 and the 5th example of Figure 9. In Figure 4, the wall of the wine bottle and the base of the chair is thin with big noise. Therefore, judging whether the normal of a point should be inward the surface or outward is difficult. The results show that our approach deals with this situation better.

4.2. Managing various situations

In this section, we examine the ability of our approach in handling various kinds of inputs, including artifacts not performed in Section 4.1 such as missing parts, sharp edges, and nested surfaces. We test our

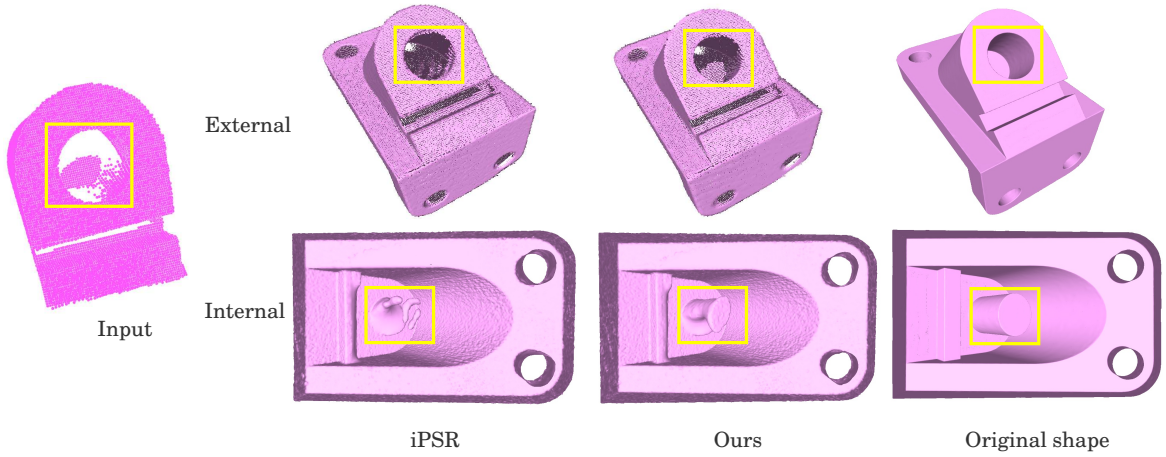


Figure 5: Results on the sampling with missing parts in the sunken cylinder of an anchor from both external and internal view. Our method achieves better orientation and reconstruction performance.

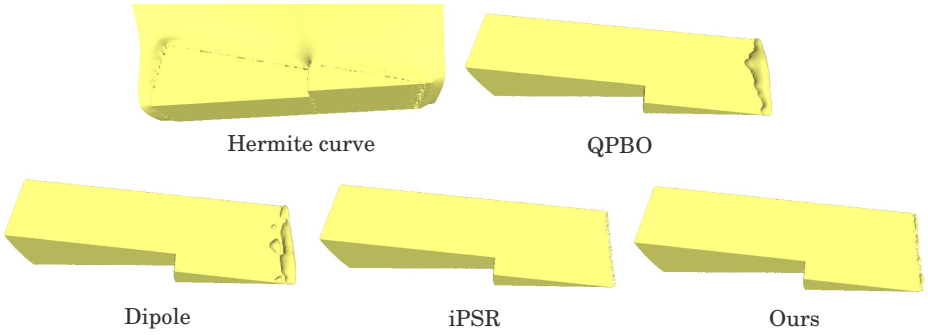


Figure 6: Results on sharp edges. All methods encounter some challenges to orient the sharp acute angle. Hermite curve even generates wrong orientation in the whole top side surface. Our approach achieves a relative decent effect in this situation.

approach in different sampling densities from hundreds to millions. Moreover, we utilize real scanned point clouds to validate the effectiveness of our method.

Figure 5 shows an *anchor* model with missing parts in the point cloud. The original shape of the anchor is shown in the rightmost column. The first row is the external view, and the second row is the internal view (looking from the internal of the shape). However, the sunken cylinder of the anchor is not scanned completely, causing a missing region in the input. The left two columns are orientation and reconstruction performances of iPSR and ours with the scanned data. It can be seen from the external view that our method generates correct normal orientation for the sunken cylinder although only a few points are scanned in this region. Therefore, we achieve better reconstruction performance which is obvious from the internal view.

Figure 6 shows a comparison of sharp edges among different approaches. We can see that all methods encounter some challenges to orient the sharp acute angle. Hermite curve even generates wrong orientation in the whole top side surface. Our approach achieves a relative decent effect in this situation. In Figure 7, we exhibit an example of nested surface with three layers at $r = 0.5, r = 0.75$, and $r = 1.0$. We show the orientation result and the implicit space generated by the oriented point cloud. The green color represents the isovalue. Our method generates correct normal orientation: the normals are alternative in and out among adjacent layers. Managing nested surfaces is also one of the advantages of our method over visibility-based and propagation-based approaches.

Figure 8 shows our reconstructions of different sampling densities from 3D sketch of 999 points to dense sampling with 5 million points. Our method exhibits high performance for these examples. In Figure 9, we

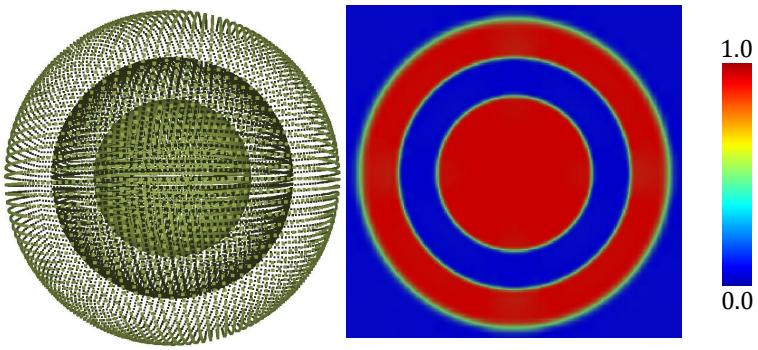


Figure 7: Our reconstruction on the nested surface with three layers on $r = 0.5, r = 0.75$, and $r = 1.0$. We show the orientation results and the implicit space generated by the oriented point cloud. The green color represents the isovalue.

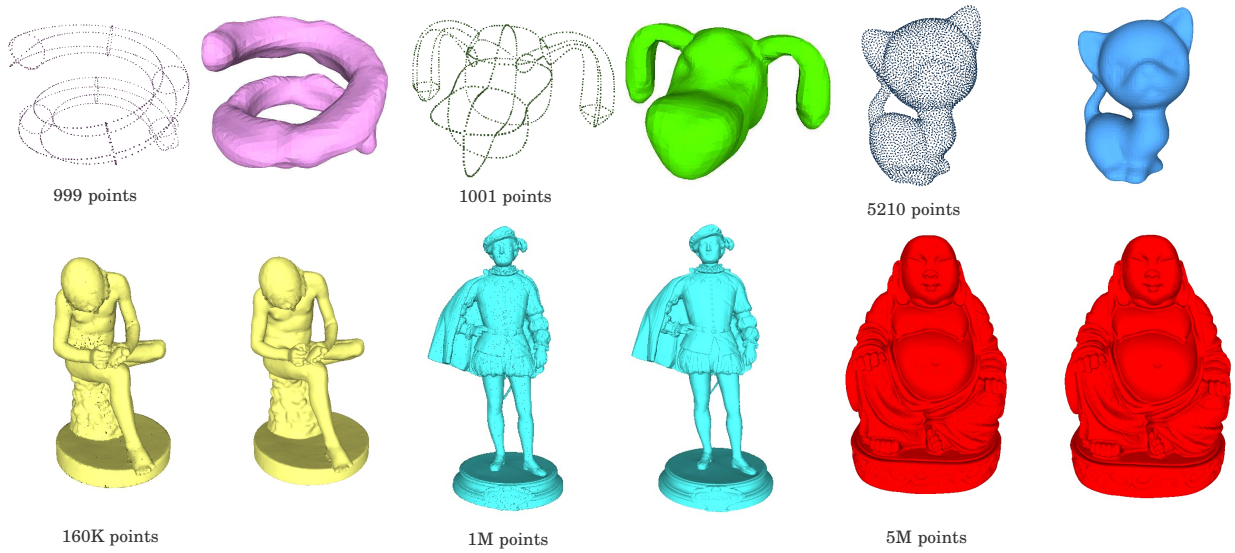


Figure 8: Our reconstructions on point clouds of different sampling densities from 3D sketch of 999 points to dense sampling with 5 million points.

show qualitative comparisons on a variety of inputs involving extremely sparse samplings, complex topology, multiple connected components, outliers, and noise. It can be seen that our method effectively manages these situations and achieves well-rounded performance among all methods. In Figure 10, we show our reconstructions of several real scanned point clouds provided by Huang et al. (2022). Our method produces consistently oriented normals and therefore generates detailed reconstruction when the scanned point cloud is accurate. Even when the scanned point cloud is noisy with missing regions, our method can still achieve a decent performance.

4.3. Running time

Table 2 exhibits the running time of our approach in a laptop with AMD Ryzen 5 5600H CPU @ 3.3GHz and total of 24GB memory in two memory slots. The running time includes all processes: IOs, orientations, reconstructions, representative point samplings, etc. We show the tree node count (during optimization), point number, representative set point number, and running time of several examples. If the representative set point number is "-" in the table, then we have conducted per-point optimization (no representative set is used) in this shape. For the *buddha* model with 5 million points, we sample 100K points for the representative

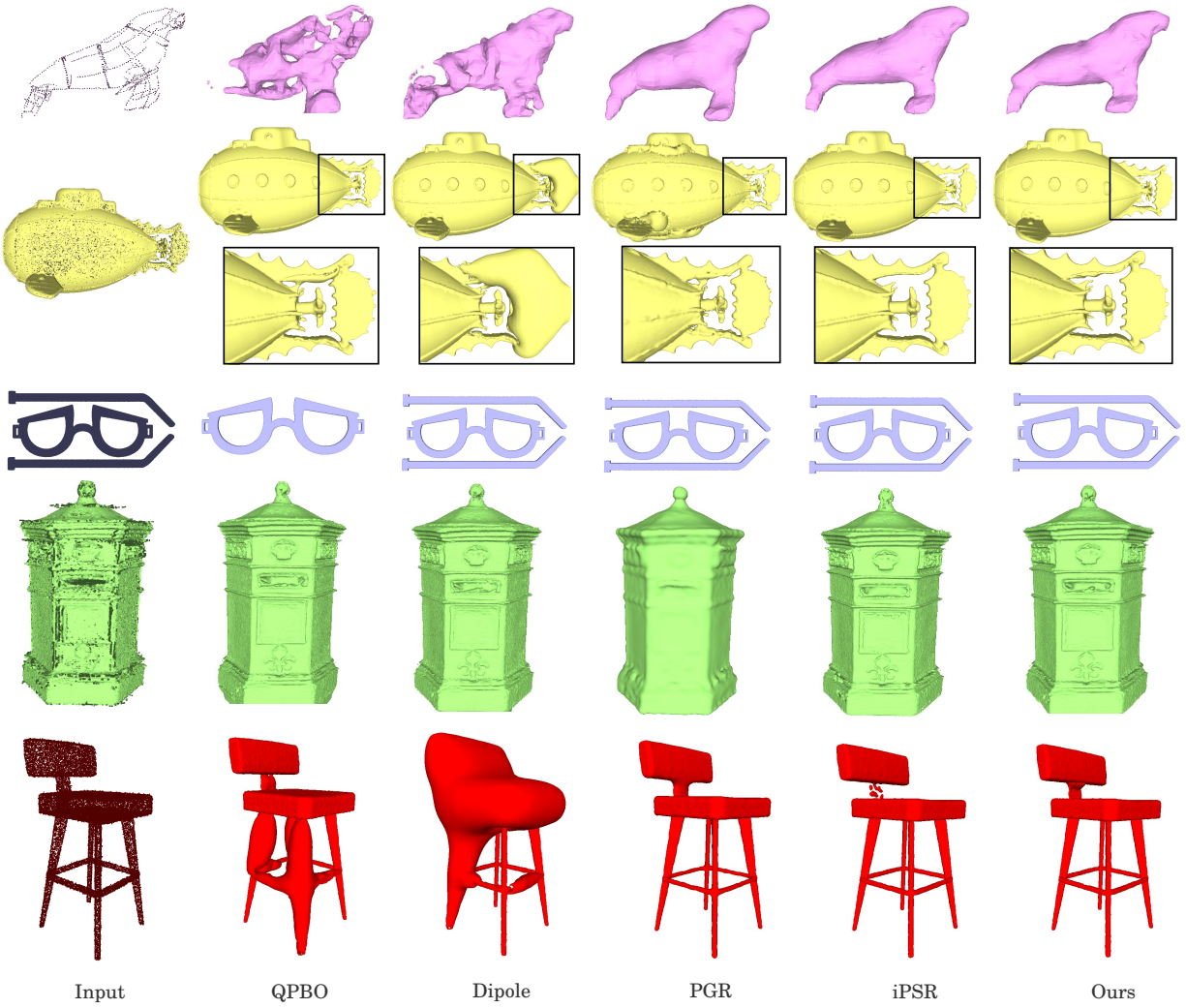


Figure 9: Qualitative comparisons on a variety of inputs involving extremely sparse samplings, complex topology, multiple connected components, outliers, and noise. Our method effectively manages these situations and achieves well-rounded performance among all methods.

set. The time for orienting representative points is 214s, and the time for other processes (all IOs, orienting the dense point cloud by the representative set) is 242s, for a total of 456s. For other examples, most time is spent on optimization.

We take *bar_chair36* with 40K points for comparison among different methods. The running time includes orientation and reconstruction. We do not apply representative set in this example and records 85s. Local consistency-based methods MST, Hermite curve and QPBO are fast in CPU and only take 4s, 12s and 4s respectively. iPSR spends 49s in 17 iterations. Dipole applies a deep neural network to keep local orientation consistency. Therefore, it relies on GPU and takes 211s on a server with Intel(R) Xeon(R) Silver 4210 CPU @ 2.20GHz and RTX 2080Ti GPU. PGR solves a dense linear system and relies heavily on computing resources with time complexity quadratic to the point numbers. We change its kdtree implementation from "KDTree" to "cKDTree" in SciPy (Virtanen, P. et al., 2019) to achieve fast width calculation. It takes about 87s on the RTX 2080Ti GPU. If executed in CPU, then it takes several hours. Applying a representative set reduces the running time of our method. For example, our approach takes 239s for *mailbox*, and iPSR takes 854s in 30 iterations.

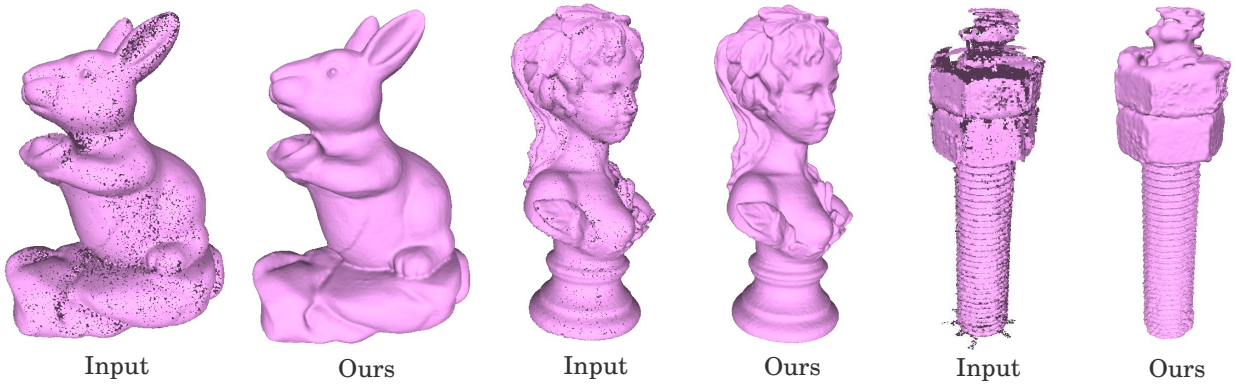


Figure 10: Our reconstructions on real scanned point clouds.

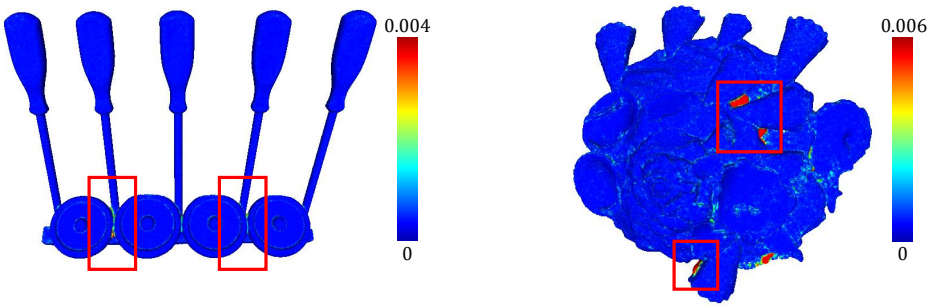


Figure 11: Our method may produce inaccurate orientation in non-outermost surfaces that are very close to each other. This is a limitation of our method.

5. Conclusion and future works

In this work, we propose an orientation and reconstruction approach that incorporates isovalue constraints to the Poisson equation. We bridge orientation and reconstruction in the implicit space and express the orientation problem into a sparse linear system. Our method shows competitive performance for varying noise and artifacts, and works on an average laptop CPU. We believe that communicating orientation and reconstruction in the implicit function space is interesting and follow-up works can be carried out.

Our method still has some limitations. Although our approach does not rely on GPU, the solving process of the linear system still requires a large amount of memory (about 10.9GB for *mailbox* and 12.6GB for *buddha*). It mainly depends on the tree node count. However, most laptops have more than one memory slots to support the usage of 16GB to even 32GB. In the future, we can try the depth-by-depth solving similar to SPSR instead of including all nodes in a large linear system. Moreover, our method may produce inaccurate orientation in non-outermost surfaces that are very close to each other. Figure 11 shows the error map of our reconstructions relative to the ground truth meshes. The high error places demonstrate this limitation. The possible reason is that the indicator function changes rapidly in this area. However, the octree depth during optimization is not sufficiently high (recall that the implicit space is spanned by B-spline basis functions centered at octnodes). Conducting depth-by-depth solving is still a potential solution to support large depth during optimization.

Acknowledgements

We would like to thank the anonymous reviewers for their valuable feedback. This research was supported by the National Key R&D Program of China (2020YFB1708900) and the National Natural Science Foundation

of China (12071244).

Appendix A. Details of the conjugate gradient optimization

This appendix presents the details of the conjugate gradient optimization of our method. It also explains why the sparsity of the matrices can be maintained during the solving process. In Section 3.3, the optimization problem is transformed into a solution of a linear system

$$\tilde{A}\tilde{x} = \tilde{b}, \tag{A.1}$$

where

$$\tilde{A} = \begin{pmatrix} U^TU + \alpha A^TA & -\alpha A^TB \\ -\alpha B^TA & \alpha B^TB + \beta M \end{pmatrix}, \qquad \tilde{x} = \begin{pmatrix} x \\ n \end{pmatrix}, \qquad \tilde{b} = \begin{pmatrix} U^T\frac{1}{2}\vec{1} \\ 0 \end{pmatrix}. \tag{A.2}$$

Let $\tilde{x}^{(k)} = (x^{(k)}, n^{(k)})$ be the results of the k-th iteration and is initialized as $x^{(0)} = 1e^{-3} \times \vec{1}, n^{(0)} = \vec{0}$. The conjugate graident algorithm can be written as follows: let $\tilde{r}^{(0)} = \tilde{b} - \tilde{A}\tilde{x}^{(0)}, \tilde{p}^{(0)} = \tilde{r}^{(0)}$, where

$$\tilde{A}\tilde{x}^{(0)} = \begin{pmatrix} U^TUx^{(0)} + \alpha A^TAx^{(0)} - \alpha A^TBn^{(0)} \\ -\alpha B^TAx^{(0)} + \alpha B^TBn^{(0)} + \beta Mn^{(0)} \end{pmatrix}. \tag{A.3}$$

Denote

$$\tilde{p}^{(k)} = \begin{pmatrix} p_x^{(k)} \\ p_n^{(k)} \end{pmatrix}, \qquad \tilde{r}^{(k)} = \begin{pmatrix} r_x^{(k)} \\ r_n^{(k)} \end{pmatrix}, \tag{A.4}$$

then, for each iteration,

$$\alpha_k = \frac{(\tilde{r}^{(k)}, \tilde{r}^{(k)})}{(\tilde{p}^{(k)}, \tilde{A}\tilde{p}^{(k)})} = \frac{(r_x^{(k)}, r_x^{(k)}) + (r_n^{(k)}, r_n^{(k)})}{(Up_x^{(k)}, Up_x^{(k)}) + \alpha[(Ap_x^{(k)}, Ap_x^{(k)}) + (Bp_n^{(k)}, Bp_n^{(k)}) - 2(Ap_x^{(k)}, Bp_n^{(k)})] + \beta(p_n^{(k)}, Mp_n^{(k)})}, \tag{A.5}$$

$$\tilde{x}^{(k+1)} = \tilde{x}^{(k)} + \alpha_k\tilde{p}^{(k)}, \tag{A.6}$$

$$\tilde{r}^{(k+1)} = \tilde{r}^{(k)} - \alpha_k\tilde{A}\tilde{p}^{(k)}, \tag{A.7}$$

$$\beta_k = \frac{(\tilde{r}^{(k+1)}, \tilde{r}^{(k+1)})}{(\tilde{r}^{(k)}, \tilde{r}^{(k)})} = \frac{(r_x^{(k+1)}, r_x^{(k+1)}) + (r_n^{(k+1)}, r_n^{(k+1)})}{(r_x^{(k)}, r_x^{(k)}) + (r_n^{(k)}, r_n^{(k)})}, \tag{A.8}$$

$$\tilde{p}^{(k+1)} = \tilde{r}^{(k+1)} + \beta_k\tilde{p}^{(k)}, \tag{A.9}$$

where

$$\tilde{A}\tilde{p}^{(k)} = \begin{pmatrix} U^TUp_x^{(k)} + \alpha A^TAp_x^{(k)} - \alpha A^TBp_n^{(k)} \\ -\alpha B^TAp_x^{(k)} + \alpha B^TBp_n^{(k)} + \beta Mp_n^{(k)} \end{pmatrix}. \tag{A.10}$$

Note that $U^TUx = (U^T(Ux))$ and $x^TU^TUx = (Ux, Ux)$. Therefore, each step of the conjugate gradient algorithm can be written as the product of a matrix and a vector, or the inner product of two vectors. The sparsity of the matrices can then be maintained during the solving process. The complexity and the number of iterations are described in Section 3.3.

References

Alliez, P., Cohen-Steiner, D., Tong, Y., Desbrun, M., 2007. Voronoi-based variational reconstruction of unoriented point sets, in: Proceedings of the 5th Eurographics Symposium on Geometry Processing, Eurographics Association. pp. 39–48.

Atzmon, M., Lipman, Y., 2020. SAL: sign agnostic learning of shapes from raw data, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, Computer Vision Foundation / IEEE. pp. 2562–2571.

Ben-Shabat, Y., Koneputugodage, C.H., Gould, S., 2022. DiGS: divergence guided shape implicit neural representation for unoriented point clouds, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, Computer Vision Foundation / IEEE. pp. 19301–19310.

Berger, M., Levine, J.A., Nonato, L.G., Taubin, G., Silva, C.T., 2013. A benchmark for surface reconstruction. ACM Trans. Graph. 32, 20:1–20:17.
Boros, E., Hammer, P.L., Sun, X., 1991. Network flows and minimization of quadratic pseudo-Boolean functions. Tech. rep., Technical Report RRR 17-1991, RUTCOR Research Report .
Cazals, F., Pouget, M., 2005. Estimating differential quantities using polynomial fitting of osculating jets. Comput. Aided Geom. Des. 22, 121–146.
Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., Ranzuglia, G., 2008. MeshLab: an open-source mesh processing tool, in: Eurographics Italian Chapter Conference, Eurographics. pp. 129–136.
Erler, P., Guerrero, P., Ohrhallinger, S., Mitra, N.J., Wimmer, M., 2020. Points2Surf: learning implicit surfaces from point clouds, in: 16th European Conference on Computer Vision, Springer. pp. 108–124.
Gropp, A., Yariv, L., Haim, N., Atzmon, M., Lipman, Y., 2020. Implicit geometric regularization for learning shapes, in: Proceedings of the 37th International Conference on Machine Learning, PMLR. pp. 3789–3799.
Gschwandtner, M., Kwitt, R., Uhl, A., Pree, W., 2011. Blensor: blender sensor simulation toolbox, in: Advances in Visual Computing - 7th International Symposium, Springer. pp. 199–208.
Guerrero, P., Kleiman, Y., Ovsjanikov, M., Mitra, N.J., 2018. PCPNet: learning local shape properties from raw point clouds. Comput. Graph. Forum 37, 75–85.
Haggerty, D. et al., 2019. trimesh 3.9. URL: https://trimsh.org/.
Hestenes, M.R., Stiefel, E.L., 1952. Methods of conjugate gradients for solving linear systems. Journal of Research of the National Bureau of Standards (United States) 49.
Hoppe, H., DeRose, T., Duchamp, T., McDonald, J.A., Stuetzle, W., 1992. Surface reconstruction from unorganized points, in: Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques, ACM. pp. 71–78.
Hou, F., Wang, C., Wang, W., Qin, H., Qian, C., He, Y., 2022. Iterative Poisson surface reconstruction (iPSR) for unoriented points. ACM Trans. Graph. 41, 128:1–128:13.
Huang, Z., Carr, N., Ju, T., 2019. Variational implicit point set surfaces. ACM Trans. Graph. 38, 124:1–124:13.
Huang, Z., Wen, Y., Wang, Z., Ren, J., Jia, K., 2022. Surface reconstruction from point clouds: a survey and a benchmark. CoRR abs/2205.02413. arXiv:2205.02413.
Jakob, J., Buchenau, C., Guthe, M., 2019. Parallel globally consistent normal orientation of raw unorganized point clouds. Comput. Graph. Forum 38, 163–173.
Katz, S., Tal, A., Basri, R., 2007. Direct visibility of point sets. ACM Trans. Graph. 26, 24.
Kazhdan, M.M., Bolitho, M., Hoppe, H., 2006. Poisson surface reconstruction, in: Proceedings of the 4th Eurographics Symposium on Geometry Processing, Eurographics Association. pp. 61–70.
Kazhdan, M.M., Hoppe, H., 2013. Screened Poisson surface reconstruction. ACM Trans. Graph. 32, 29:1–29:13.
Koch, S., Matveev, A., Jiang, Z., Williams, F., Artemov, A., Burnaev, E., Alexa, M., Zorin, D., Panozzo, D., 2019. ABC: a big CAD model dataset for geometric deep learning, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, Computer Vision Foundation / IEEE. pp. 9601–9611.
König, S., Gumhold, S., 2009. Consistent propagation of normal orientations in point clouds, in: 14th International Workshop on Vision, Modeling, and Visualization, DNB. pp. 83–92.
Lin, S., Xiao, D., Shi, Z., Wang, B., 2022. Surface reconstruction from point clouds without normals by parametrizing the Gauss formula. ACM Trans. Graph. 42, 14:1–14:19.
Ma, B., Han, Z., Liu, Y., Zwicker, M., 2021. Neural-Pull: learning signed distance function from point clouds by learning to pull space onto surface, in: Proceedings of the 38th International Conference on Machine Learning, PMLR. pp. 7246–7257.
Mello, V., Velho, L., Taubin, G., 2003. Estimating the in/out function of a surface represented by points, in: Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications, ACM. pp. 108–114.
Metzer, G., Hanocka, R., Zorin, D., Giryes, R., Panozzo, D., Cohen-Or, D., 2021. Orienting point clouds with dipole propagation. ACM Trans. Graph. 40, 165:1–165:14.
Mullen, P., de Goes, F., Desbrun, M., Cohen-Steiner, D., Alliez, P., 2010. Signing the unsigned: robust surface reconstruction from raw pointsets. Comput. Graph. Forum 29, 1733–1741.
Oliver, L. et al., . Three D Scans: Free 3D scan archive. URL: https://threedscans.com.
Park, J.J., Florence, P., Straub, J., Newcombe, R.A., Lovegrove, S., 2019. DeepSDF: learning continuous signed distance functions for shape representation, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, Computer Vision Foundation / IEEE. pp. 165–174.
Pearson, K., 1901. On lines and planes of closest fit to systems of points in space. Philosophical Magazine 2, 559–572.
Peng, S., Jiang, C., Liao, Y., Niemeyer, M., Pollefeys, M., Geiger, A., 2021. Shape as points: a differentiable Poisson solver, in: Annual Conference on Neural Information Processing Systems, pp. 13032–13044.
Rusinkiewicz, S., 2019. A symmetric objective function for ICP. ACM Trans. Graph. 38, 85:1–85:7.
Schertler, N., Savchynskyy, B., Gumhold, S., 2017. Towards globally optimal normal orientations for large point clouds. Comput. Graph. Forum 36, 197–208.
Virtanen, P. et al., 2019. SciPy 1.0-fundamental algorithms for scientific computing in Python. CoRR abs/1907.10121. arXiv:1907.10121.
Wang, J., Yang, Z., Chen, F., 2012. A variational model for normal computation of point clouds. Vis. Comput. 28, 163–174.
Wang, S., Liu, X., Liu, J., Li, S., Cao, J., 2022. Deep patch-based global normal orientation. Comput. Aided Des. 150, 103281.
Wang, Z., Wang, P., Dong, Q., Gao, J., Chen, S., Xin, S., Tu, C., 2021. Neural-IMLS: learning implicit moving least-squares for surface reconstruction from unoriented point clouds. CoRR abs/2109.04398. arXiv:2109.04398.
Wohlkinger, W., Aldoma, A., Rusu, R.B., Vincze, M., 2012. 3DNet: large-scale object class recognition from CAD models, in: IEEE International Conference on Robotics and Automation, IEEE. pp. 5384–5391.

Xiao, D., Lin, S., Shi, Z., Wang, B., 2022. Learning modified indicator functions for surface reconstruction. Comput. Graph. 102, 309–319.

Xie, H., McDonnell, K.T., Qin, H., 2004. Surface reconstruction of noisy and defective data sets, in: Proceedings of the 15th IEEE Visualization Conference, IEEE Computer Society. pp. 259–266.

Xie, H., Wang, J., Hua, J., Qin, H., Kaufman, A.E., 2003. Piecewise C^1 continuous surface reconstruction of noisy point clouds via local implicit quadric regression, in: Proceedings of the 14th IEEE Visualization Conference, IEEE Computer Society. pp. 91–98.

Zhou, Q., Jacobson, A., 2016. Thingi10K: a dataset of 10,000 3D-printing models. CoRR abs/1605.04797. arXiv:1605.04797.