Vis Comput (2017) 33:385–398
DOI 10.1007/s00371-016-1208-1

ORIGINAL ARTICLE

Shape-controllable geometry completion for point cloud models

Long Yang[1,2] · Qingan Yan[1] · Chunxia Xiao[1]

Published online: 6 February 2016

Abstract Geometry completion is an important operation for generating a complete model. In this paper, we present a novel geometry completion algorithm for point cloud models, which is capable of filling holes on either smooth models or surfaces with sharp features. Our method is built on the physical diffusion pattern. We first decompose each pass hole-boundary contraction into two steps, namely normal propagation and position sampling. Then the normal dissimilarity constraint is incorporated into these two steps to fill holes with sharp features. Our algorithm implements these two steps alternately and terminates until generating no new hole boundary. Experimental results demonstrate its feasibility and validity of recovering the potential geometry shapes.

Keywords Point cloud model · Geometry completion · Sharp features · Normal propagation · Position sampling

1 Introduction

Benefiting from its simple representation, point cloud model has been widely used in the last two decades [5,13,21]. Although capture devices have been improved substantially, the scanned data still contain deficient holes in certain situations. Moreover, we often confront abraded surfaces and damaged models with different deficiencies. All these holes need to be completed appropriately.

Many techniques have been proposed to deal with this ill-posed problem. The existing methods, such as [3,9,15,23,24, 28,33], are designed to fill holes for polygonal mesh models. Most of these methods define geometry completion operators by utilizing connection topology and generate robust hole-filling results. For more details about mesh completion, please refer to the surveys of Ju [18], Campen et al. [6] and Attene et al. [2].

Filling holes directly on point cloud models currently turns out to be an essential requirement for many practical applications. Early work [8] presents an overall pipeline of geometry completion for point cloud models. Although many techniques [7,19,20,25,29,30,32] work on point cloud models, most of them only generate smooth hole-filling results.

In contrast to smooth hole filling, sometimes it makes special sense to complete a hole by preserving sharp features (i.e., edge/corner/apex) or using the least materials; see Fig. 1b, e. Since general smooth hole-filling methods cannot complete the protruding features plausibly, it is still a difficult task to recover the potential sharp features on a deficient point cloud surface.

Our work is inspired by the observation that geometry completion should reasonably provide potential shape options for deficient regions. Therefore, in this paper, we propose a novel hole-filling approach for point cloud models.

Our algorithm simulates the energy diffusion process and progressively contracts a hole boundary until the hole is closed. Unlike the existing boundary propagating method [10], it could control the propagating process effectively. The main contributions of this paper are as follows:

– Presenting a unified geometry completion algorithm that recovers both smooth and feature-preserved holes for

Electronic supplementary material The online version of this article (doi:10.1007/s00371-016-1208-1) contains supplementary material, which is available to authorized users.

✉ Chunxia Xiao
cxxiao@whu.edu.cn

[1] School of Computer, Wuhan University, Wuhan, China

[2] College of Information Engineering, Northwest A&F University, Yangling, China

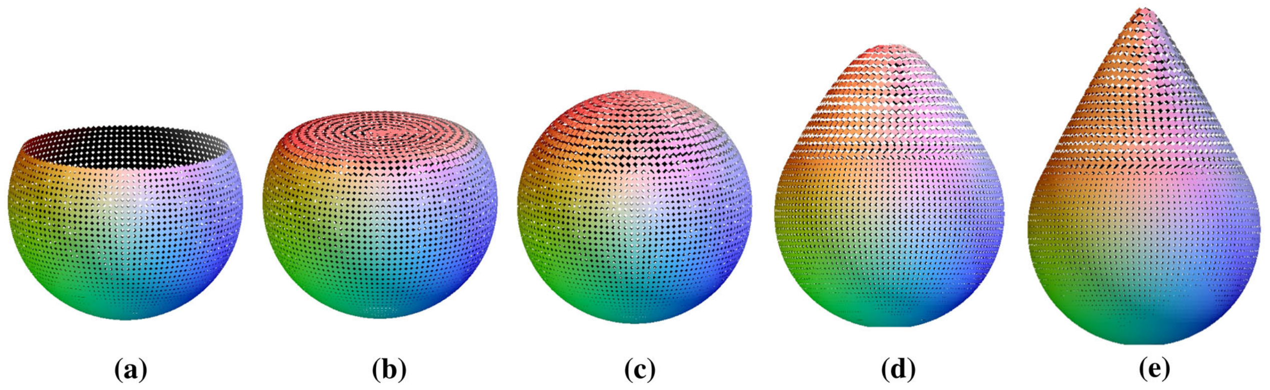


Fig. 1 Our shape-controllable geometry completion algorithm recovers a large deficient sphere (**a**) with different neighborhood radiuses r. **b**, **d** and **e** use a constant size of r, while (**c**) employs a set of decreasing values of r. All these results use the identical elastic force parameter $\sigma_r = 0.22$ *BBL* (the bounding box diagonal length of the input model) and are generated without the normal dissimilarity constraint. **a** A deficient sphere, **b** $r = 0.4$ *BBL*, **c** $r = 0.25$ *BBL*, $r = r/1.1$, **d** $r = 0.11$ *BBL*, **e** $r = 0.05$ *BBL*

point cloud models. Sharp features are reproduced by controlling the hole-boundary contracting process.
– Developing a new position sampling operator based on elastic force to generate the filling points. It avoids local and global reconstructions so that points on non-hole regions keep unchanged.

Before elaborating our algorithm, we briefly review the related techniques of geometry completion and feature reconstruction for point cloud models in the next section.

2 Related work

Many surface reconstruction methods, whether global or local fitting of the scattered point cloud data, could fill holes to a certain extent. Carr et al. [7] employ radial basis function (*RBF*) to construct a global signed distance field (*SDF*) for the original point set. It could fill the deficient holes and generate a smooth surface. The *MPU* method [25] uses piecewise quadratic fittings to construct the feature preserved *SDF*. Kazhdan et al. [20] present a *Poisson reconstruction* method which uses piecewise constant indicator gradients to construct a potential surface for the input point cloud. A new enhanced version is *screened Poisson reconstruction* [19]. It could reconstruct the feature preserved surface faithfully even if the input model contains small deficiencies.

The *Volfill* method [10] simulates the heat diffusion process to propagate *SDF* from known parts to the adjacent hole region. Once the diffusion was completed, a hole is filled. This method could fill holes with complex shapes due to its local shape propagation in a progressive way. However, simply propagated *SDF* cannot describe the sharp turn of a surface exactly. It also needs the support of local space subdivision. Weyrich et al. [30] employ moving least square (*MLS*) projection to replace the distance field estimation in [10]. Since lacking effective constraints for the diffusion process, it still cannot faithfully recover the holes with large deficiency or holes with extreme feature, especially for the sharp (edge/corner) regions.

Another type of hole-filling technique, example-based geometry completion, recovers deficient regions by finding similar patches from either the input model [14,27,29,31] or other models belonging to the same category [22]. Example-based hole-filling method suits the hole whose boundary region has a similar shape on the known model. Hence, it does not effectively handle a hole whose shape cannot be learned from non-hole regions.

To produce sharp features, Fleishman et al. [12] segment a point model into piecewise smooth regions based on robust statistics. Öztireli et al. [26] present an *RIMLS* method. It combines the robust local kernel regression with the implicit *MLS* to describe sharp features. Recently, Huang et al. presented an edge-aware resampling (*EAR*) method [16] for point cloud models. It generates sharp features by progressively resampling a surface to approach its feature edges. Although these methods could generate appealing sharp features, they need sufficient samples near or on the feature regions. Therefore, they cannot fill holes with large data deficiency.

To recover the missed features, state-of-the-art techniques [15,24,32] resort to interactive method. Harary et al. [15] and Ngo and Lee [24] are designed for meshes and adopt the strategy which first recovers the feature curve under user interactions, then fills the divided smooth sub-holes. *Morfit* [32] can recover some complex surfaces by interactively manipulating the curve skeleton and profile curve of the input point cloud model. Through feature editing, it can reconstruct sharp edges. *Morfit* requires an initial skeleton and applies to the generalized cylinder objects whose topology can be described by the curve skeleton. Differing from these techniques, our method recovers sharp features in an automatic hole-filling process.

3 Formulation of our geometry completion algorithm

Our geometry completion algorithm takes advantage of the local propagation properties [10]. To repair sharp features, it goes a step further to decompose the boundary contracting process into two primary steps: normal propagation and position sampling. The former controls orientations of filled points as well as the shape of recovered surface, while the latter practically generates a new boundary for one-pass hole-boundary contraction. These two steps are implemented alternately during the hole-filling process so that they could benefit from each other.

This decomposition gives shape-controlling chance to our algorithm. We incorporate the normal dissimilarity constraint into both steps of normal propagation and position sampling for recovering sharp features in hole regions.

3.1 Algorithm overview

Given the oriented point cloud, our geometry completion algorithm first implements a preprocessing to denoise the input point cloud surface. After determining a hole boundary, it contracts the hole region by propagating its boundary iteratively until no new boundary is generated. The main procedure of our method can be concisely interpreted as Algorithm 1.

Algorithm 1 Shape controllable geometry completion for point cloud models.

Input: a point cloud model with normals;
Output: the completed point cloud model.

```
1.  input a model $\mathsf{P}$ (has a hole $\Omega'$) with normals $\mathsf{N}$;
2.  preprocessing: $N = f_1(\mathsf{N})$ and $P = f_2(\mathsf{P})$ (§3.2);
3.  determine the hole-boundary $\partial\Omega'$ (§3.3);
4.  Do   /* iteratively hole-boundary contraction. */
5.    propagate the normals of points near $\partial\Omega'$ (§3.4);
6.    If generated new boundary $\partial\Omega$ (§4.1)
7.      generate $B$ by sampling new boundary $\partial\Omega$ (§4.2);
8.      $\partial\Omega' \longleftarrow B$;
9.    EndIf
10. Until no new boundary $\partial\Omega$ is generated
11. End
```

3.2 Preprocessing

Our algorithm takes as input a set of points $\mathsf{P} \subset R^3$ and their normals $\mathsf{N} \subset R^3$. To find a faithful hole boundary, it first implements a two-stage filtering preprocessing for the input point cloud model. Specifically, we enforce bilateral filtering [11] on both orientations and positions of the input points, respectively. For a certain point p_i with its normal n_i ($i = 1, \ldots, m_p$ and m_p is the number of input points), the filtered normal and position can be expressed as $n_i = f_1(\mathsf{n}_i)$ and $p_i = f_2(\mathsf{p}_i)$ correspondingly. We denote the filtered normals and point cloud as N and P, respectively.

3.3 Determining the hole boundary

Although there are some hole-boundary detecting operators [1,4,8] for point cloud model, they are not robust enough for the hole near a sharp region. In this paper, for a hole Ω' on the filtered point set $P \subset R^3$, we present a *divide and conquer* approach to determine its boundary $\partial\Omega'$ effectively.

It first selects a small number of feature points f_i' ($i = 1, \ldots, m_f$ and m_f is the number of the selected feature points) sequentially along the boundary of a specified hole so that each boundary segment between f_i' and f_{i+1}' approximates a local linearity. We insert these feature points into a boundary sequence B'.

For a specified segment $f_i' f_{i+1}'$, the point b_m' on the hole boundary corresponding to the middle point M of $f_i' f_{i+1}'$ is determined by choosing the nearest neighbor from input point cloud to M. If the chosen b_m' is a new boundary point (has not entered B'), we insert it into B' between f_i' and f_{i+1}'. With the chosen b_m', our algorithm *recursively* implements the same operation on segments $f_i' b_m'$ and $b_m' f_{i+1}'$, respectively. This process terminates when no new boundary point corresponding to the middle point of each new segment is found.

We iteratively implement the above recursive operations for all segments. Finally, the constructed point sequence $B' \subset P$ composes the discrete representation of the initial hole boundary $\partial\Omega'$ (see an example in the accompanying video).

3.4 Normal propagation

To compute the contracted boundary $\partial\Omega$, our algorithm needs to construct a normal field for those sampling points on the new boundary in advance.

We assign a new point sequence B as the discrete representation of $\partial\Omega$. The normal n_i of a candidate filling point b_i is calculated by the weighted sum of normals from its local neighbors $N_r(b_i)$:

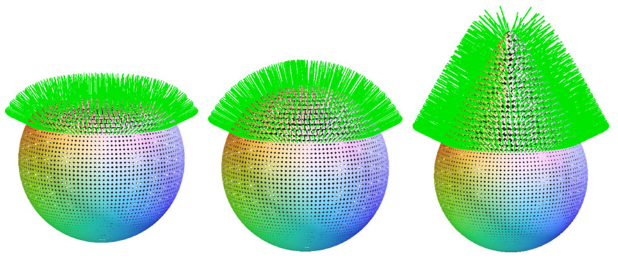


Fig. 2 Different hole-filled results, from *left* to *right* corresponding to Fig. 1b, c, e respectively, are guided by distinct processes of normal propagation

$$n_i = \frac{1}{K(b_i)} \sum_{p \in N_r(b_i)} n_p * g_1(\|p - b_i\|)\, g_2\left(\|n_p - n_{i'}\|\right), \tag{1}$$

$$K(b_i) = \sum_{p \in N_r(b_i)} g_1(\|p - b_i\|)\, g_2\left(\|n_p - n_{i'}\|\right), \tag{2}$$

where r is the neighborhood radius of b_i, g_1 is the Gaussian distance weight between different points with standard deviation σ_d and g_2 is the Gaussian normal dissimilarity weight with standard deviation σ_n.

Equation (1) resembles bilateral normal filter [17,26] formally. It mainly differs in the purpose that we intend to infer the unknown normal for a candidate position rather than filter a known normal. In fact, we cannot offer a reference normal n_i to compute the difference for g_2 between a neighboring normal n_p and the normal of candidate b_i. Instead, we take the normal $n_{i'}$, corresponding to a point $b_i^{'}$ (on the former hole boundary $B^{'}$) whose position is most close to the candidate position b_i, as the reference normal in Eq. (1).

Note that we use the normal dissimilarity weight g_2 to constrain our normal propagation process and further to control the hole-filling shape. If a candidate b_i locates near a sharp feature, the neighboring normals on the other side will hold large values of normal dissimilarity and contribute less to n_i, while the neighboring normals on the same side contribute more. Therefore, the recovered hole boundary could preserve sharp features. Moreover, σ_n is an adjustable parameter in our normal propagation process. A large value leads to smooth orientation, while a small value results in normal propagation with orientation preservation. This constrained normal propagation combined with progressively boundary contraction contributes to the shape controllable capability of our algorithm (see Fig. 2).

3.5 Position sampling

Guided by the propagated normal, position sampling for one-pass boundary contraction should concern two objectives. First, the new generated boundary must match its surrounding surface. Second, the new filled points should hold a reasonable distribution. The latter requires a sequential and practical contraction of the hole boundary in one-pass iteration and guarantees overall decrease of the hole region.

We define the discrepancy value of a filled point b_i, denoted as $E_1(b_i)$, to measure the matching degree with its neighboring surface. The smaller the value of E_1, the better is the matching degree b_i gets. Meanwhile, our algorithm introduces *elastic force* to control the distribution of new filled points. A candidate point b_i is deemed to be a good sampling only if it locates in the equilibrium position and receives the minimum force, denoted as $E_2(b_i)$, from its neighbors on the former and the current boundaries. Combining these two objectives of E_1 and E_2 (both will be defined specifically in Sect. 4), we formulate our position sampling of one-pass boundary contraction as a minimizing problem of objective function (3),

$$E(B) = \arg\min_{B} \sum_{b_i \in B} \{E_1(b_i) + E_2(b_i)\}, \tag{3}$$

where B, formed by the latest one-pass filled points, represents the discrete new hole boundary. The solution of Eq. (3) should minimize the total elastic forces of the filled points and the discrepancy between newly generated hole boundary B and the existing surface.

4 Generating a new hole boundary

Although we have built an objective function for the hole-boundary contracting process, an optimal new boundary curve might not exist for Eq. (3). It is because there are countless sampling patterns and the trivial solution makes the objective function minimum. Moreover, determining a new boundary in the hole region is also an underdetermined problem, since we do not have sufficient conditions to constrain our sampling operation. Hence, the intention to solve Eq. (3) precisely is unadvisable. Instead, we resort to an approximate strategy to address this sampling problem.

We propose a new indirect sampling operator, also including two sequential operations both based on *elastic force*, to construct a new hole boundary B approximately. It first computes a control curve C for the new hole boundary B according to those samples on the former pass boundary $B^{'}$. Then under the constraint of the control curve C, one-pass position sampling on a $2D$ manifold is reduced to a linear $1D$ sampling along C. The introduced control curve restricts new sampling points in a limited band and makes sampling problem well posed. Thus, the new sampled boundary B offers a sound approximate solution for objective function (3).

The control curve C derived from the former pass boundary $B^{'}$ should respect the local shape of the existing surface. We optimize the position of each control point relying on both its local neighbors and the new normal field (defined in Sect. 3.4). Thereafter, we sample along control curve C and implement position optimization for each sampled point as well. From these sampled points, our algorithm constructs the next pass boundary control curve if it does not reach convergence.

4.1 Constructing hole-boundary control curve

4.1.1 Definition of elastic force

Our algorithm leverages Gaussian function to simulate elastic force and control the distance of a sampling point from its neighbors. Given a new candidate control point $c_i^{'}$, as shown in Fig. 3, it represents the equilibrium position $O_{b_i^{'}}$ with respect to a former pass boundary point $b_i^{'}$ along its propagating direction. $O_{b_i^{'}}$ receives elastic forces from its neighboring points on the former pass hole boundary. We define the elastic force from a neighbor q as

$$r_q\left(O_{b_i^{'}}\right) = 1.0 - \exp\left(\left(\|O_{b_i^{'}} - q\| - \|O_q - q\|\right)\Big/\sigma_r^2\right), \tag{4}$$

where O_q is the equilibrium position of neighbor q along the direction of vector $\overrightarrow{qO_{b_i^{'}}}$. Actually, once the neighbor q is given, the elastic force received by $O_{b_i^{'}}$ from q can be determined by the distance from O_q to $O_{b_i^{'}}$ along the direction of *repulsive force*. Note that, for a system of elastic force, we assign the repulsive direction as the positive direction and the attractive one as the negative direction. Therefore, the definition of $r_q(O_{b_i^{'}})$ can be simplified as Eq. (5):

$$r_q\left(O_{b_i^{'}}\right) = 1.0 - \exp\left(\left|O_{b_i^{'}} - O_q\right|_{\overrightarrow{qO_{b_i^{'}}}}\Big/\sigma_r^2\right), \tag{5}$$

where $|A|_{\overrightarrow{l}}$ denotes the signed distance of vector A along the direction $\overrightarrow{l}$.

σ_r is another adjustable parameter. Its value can refer to the parameter σ_d [in Eq. (1)]. In our algorithm, σ_r determines the sampling density for a hole region. Specifically, it controls the equilibrium position for a given point along the specified direction. For example, in Fig. 3, the equilibrium position $O_{b_i^{'}}$ can be computed by solving the following equation (see "Appendix"),

$$\exp\left(\left|b_i^{'} - O_{b_i^{'}}\right|_{\overrightarrow{b_i^{'}O_{b_i^{'}}}}\Big/\sigma_r^2\right) = 10^{-4}. \tag{6}$$

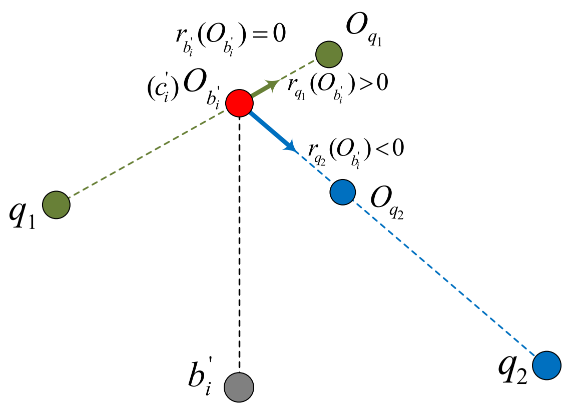


Fig. 3 Elastic forces of a candidate position $O_{b_i^{'}}$ received from its two different neighbors q_1 and q_2. $O_{b_i^{'}}$ is the equilibrium position of $b_i^{'}$ along its contracting direction

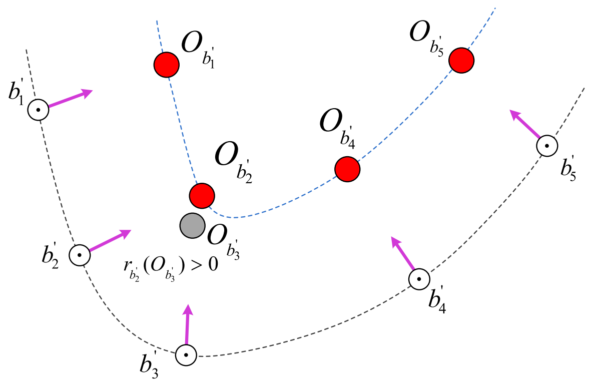


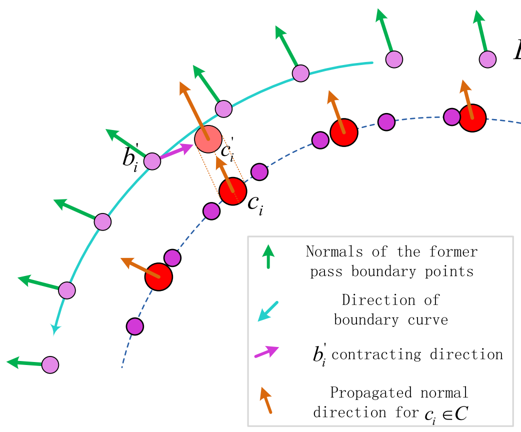
Fig. 4 A 2D illustration of the equilibrium positions $O_{b_1^{'}}$, $O_{b_2^{'}}$, $O_{b_3^{'}}$, $O_{b_4^{'}}$ and $O_{b_5^{'}}$ for different boundary points $b_1^{'}$, $b_2^{'}$, $b_3^{'}$, $b_4^{'}$ and $b_5^{'}$ along their contracting directions, respectively. The control points (*red circles*) are those equilibrium positions which receive no repulsive forces from any other adjacent boundary points

Figure 4 shows an example and illustrates the equilibrium positions derived from different boundary points along their contracting directions according to Eq. (6).

4.1.2 Boundary control curve

In our algorithm, the new boundary control curve is constructed from those equilibrium positions corresponding to the former pass hole-boundary points.

For a specified boundary point $b_i^{'}$, our algorithm computes a vector, which is the cross product from the normal of $b_i^{'}$ to the orientation of the boundary curve. We take this vector as the local contracting direction of the boundary curve $B^{'}$ (shown in Fig. 5). We search the location of $c_i^{'}$ depending on Eq. (6) from the former pass boundary point $b_i^{'} \in B^{'}$ along its contracting direction. The calculated candidate of control point $c_i^{'}$ does not necessarily match well the boundary shape.



B'
B
b_i'
c_i'
c_i
Normals of the former pass boundary points
Direction of boundary curve
b_i' contracting direction
Propagated normal direction for $c_i \in C$

Fig. 5 Generating the control points and then sampling along the new control curve. *Red circles* denote control points, while *small purple circles* are sampling points

We use neighboring points on the existing model to optimize its position according to the discrepancy definition in Eq. (7).

The optimized control point c_i will be discarded if it receives any *repulsive forces* from other boundary points. Finally, by joining all the reserved control points consecutively, we obtain the new boundary control curve. Examples are shown by red circles in Figs. 4 and 5.

4.2 Sampling along the boundary control curve

We initialize an empty filling sequence B and push the first control point c_0 into it as the first sampling point b_0. Then, we iteratively compute the next new sampling point b_{i+1} (i.e., the equilibrium position of b_i along the control curve C) starting from b_1 until each control point has been traversed. Our algorithm also implements position optimization for each b_i to match the shape of the local surface. The normals of new sampled points are computed following Eq. (1).

Once our method gets a new boundary sampling point set B, as the sequence of small purple circles shown in Fig. 5, it finishes one-pass hole-boundary contraction. By executing the main loop between the fourth line and the tenth line in Algorithm 1, the hole-filling procedure will converge if it generates no new control points after all points on the former boundary $B^{'}$ have been traversed.

In practice, if the ratio between the number of generated control points and the number of boundary points in $B^{'}$ is below a certain threshold, it means that the boundary no longer contracts noticeably. In our experiments, we terminate our algorithm when the ratio is lower than 30 %.

4.3 Position optimization

To match the local surface shape, a position candidate (either a control candidate or a sampling candidate, for the sake of clarity we just explain the sampling candidate) has to be optimized according to its local neighbors. For a new candidate b_i, we define its discrepancy $E_1(b_i)$ as the sum of weighted offsets with respect to its neighboring points along the normal direction of b_i. Specifically, the discrepancy of b_i is measured by the total local offsets:

$$\text{offsets}(b_i) = \frac{1}{K(b_i)} \sum_{p \in N_r(b_i)} g_1 * g_2 * \left| p - b_i \right|_{\overrightarrow{n_{b_i}}}, \qquad (7)$$

where n_{b_i} denotes the normal of candidate b_i, g_1 and g_2 are the distance weight and the normal dissimilarity weight, respectively, and $K(b_i)$ is defined as the same in Eq. (2). Therefore, the optimized position can be obtained by updating candidate b_i as:

$$b_i = b_i + n_{b_i} * \text{offsets}(b_i). \qquad (8)$$

Note that the normal n_{b_i} probably does not match b_i after implementing this position optimization. The recomputed normal n_{b_i} will also cause the mismatch that the updated b_i is not the best matching position with respect to the new normal any more. Theoretically, position optimization is an iterative process and will be converged finally when the position and the normal stop updating. In practice, the convergence will be reached quickly. We implement two iterations of normal updating and position optimization without triggering noticeable artifacts in our experiments.

Equations (7) and (8) indicate that local normals and the normal dissimilarity constraint benefit position sampling, especially sampling near a sharp region. In turn, the refined position sampling combined with the normal dissimilarity constraint improves normal propagation faithfully in the feature region, as explained in Eq. (1). Consequently, normal dissimilarity constraint and the mutual enhancement between normal propagation and position sampling enable our method to fill holes with sharp features.

4.4 Feature constraint

To complete a surface containing sharp features, our position sampling (stated in Sect. 4.2) may cause local overshooting samples. An example is shown in the right of Fig. 7. To eliminate this phenomenon, we introduce a sampling constraint for the sharp feature's completion.

Specifically, during the boundary contracting process, a few control points close to a sharp region may overshoot the local surface as the topmost and rightmost control points shown in Fig. 6. Our method holds these control points for keeping the chance of sampling near the sharp region (as sample b_i). However, it might give rise to the overshooting samples as well (candidate $b_{i+1}^{'}$ and $b_{i+1}^{''}$). These overshooting samples will trigger the divergence of our algorithm. We

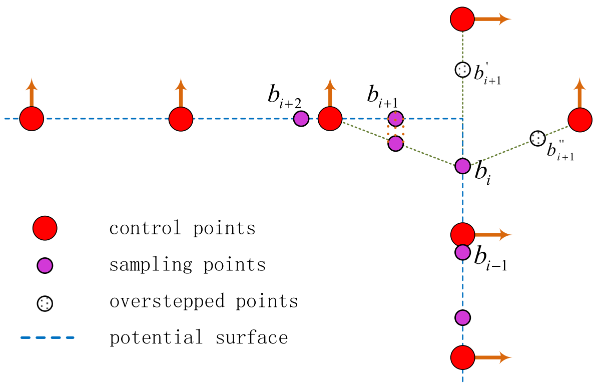


Fig. 6 Combination constraint for recovering sharp features. The sampling candidates overshooting the local potential surface (as b'_{i+1} and b''_{i+1}) will have at least one positive offset value corresponding to a neighboring control point and will be rejected by our algorithm

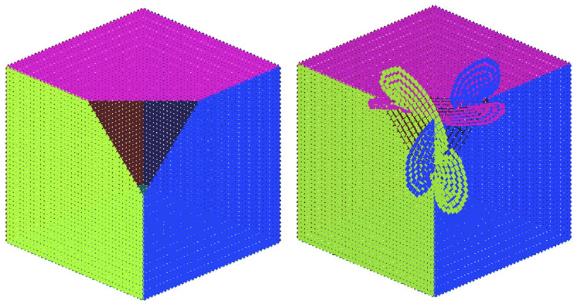


Fig. 7 Overshooting samples occur (*right*) when we fill the deficient corner of a cube (*left*) without the combination constraint

utilize a *combination condition* to check these overshooting samples (Fig. 7).

For the control curve locating in a convex region, all sampled points should lie either on the boundary or in the inner part of the polygon if no overshooting occurs. This "combination constraint" means that the offset, starting from the tangent plane of a neighboring control point to the sampling candidate, should always be a negative value or zero with respect to the normal direction of this control point.

In Fig. 6, the offsets of the candidate sample b'_{i+1} contain a positive value. Actually, it overshoots the horizontal potential surface and should be discarded. Another sampling candidate b''_{i+1}, which contains positive offset and overshoots the vertical surface, is also rejected. In contrast, for a concave boundary such offsets of a candidate should always be non-negative values. Thus, once a negative offset appears, the sampling candidate must have overshot the local concave surface and will be discarded by our algorithm.

By employing the combination constraint, our algorithm eliminates the overshooting samples during the position sampling process. The fifth column in Fig. 8 exhibits the sharp results of geometry completion under the combination constraint.

5 Results and discussion

We test our geometry completion algorithm on both synthetic and scanned surfaces to explore its capability.

The different completed results of a deficient sphere in Fig. 1 show the shape control capability of our algorithm. We use the neighborhood radius r of normal propagation to control the hole-boundary contraction. More neighboring points will be involved so that the orientation of the hole boundary converges quickly if a large r is assigned. Only a few neighbors will participate in the normal propagation if we set a small r, and the orientation of hole boundary will strictly respect the local shape of the existing model. By taking different r, our algorithm generates 4 distinct shapes, as shown in Fig. 1b–e. Figure 1e demonstrates a recovered cone shape with a small r (0.05) multiplied by a default value, the bounding box diagonal length of the input model. We denote this value as "*BBL*".

We compare our algorithm with three representative techniques of *Volfill* [10], *MPU* [25] and *screened Poisson reconstruction* [19]. The results produced by the three existing techniques on six deficient point cloud models are shown in Figs. 8, 9 and 10. Note that these three techniques reconstructed the input models and generated mesh results. Our method fills a hole by contracting its boundary, so that non-hole regions remain unchanged. For comparison, we display these results in a point cloud pattern.

Screened Poisson reconstruction method fills all holes robustly, but generates smooth results. *Volfill* algorithm generates feature preserved results for cube, pyramid, fandisk and dihedral models to a certain extent. But it still fails to recover sharp features. The *MPU* algorithm generates a sharp apex for the pyramid and recovers the sharp edge for the dihedral. But it fills the non-hole region and expands the boundaries of these two open models (see the bottom of pyramid from a side view in Fig. 8 and the left part of the dihedral in Fig. 9). In contrast, due to rigorous constraint of normal dissimilarity (see the small σ_n in Table 1), our method could strictly control the boundary propagation to recover the sharp features for these models. The results are shown in Figs. 8 and 9.

Besides completing the sharp features, our method could recover *round surfaces* by loosening the normal dissimilarity constraint. The completed results on cube, pyramid and fandisk models are shown in the last column of Fig. 8. For the "heart" model with a large hole, compared with *Volfill* [10], our algorithm can effectively control the boundary contraction by slightly decreasing the normal dissimilarity parameter σ_n. It recovered a desirable surface with continuous curvature change; see comparison of the third and the fifth results in the last row of Fig. 8.

For the real scanned models (Figs. 11, 12, 13) and the noise-contaminated models (Figs. 9, 10), our method imple-

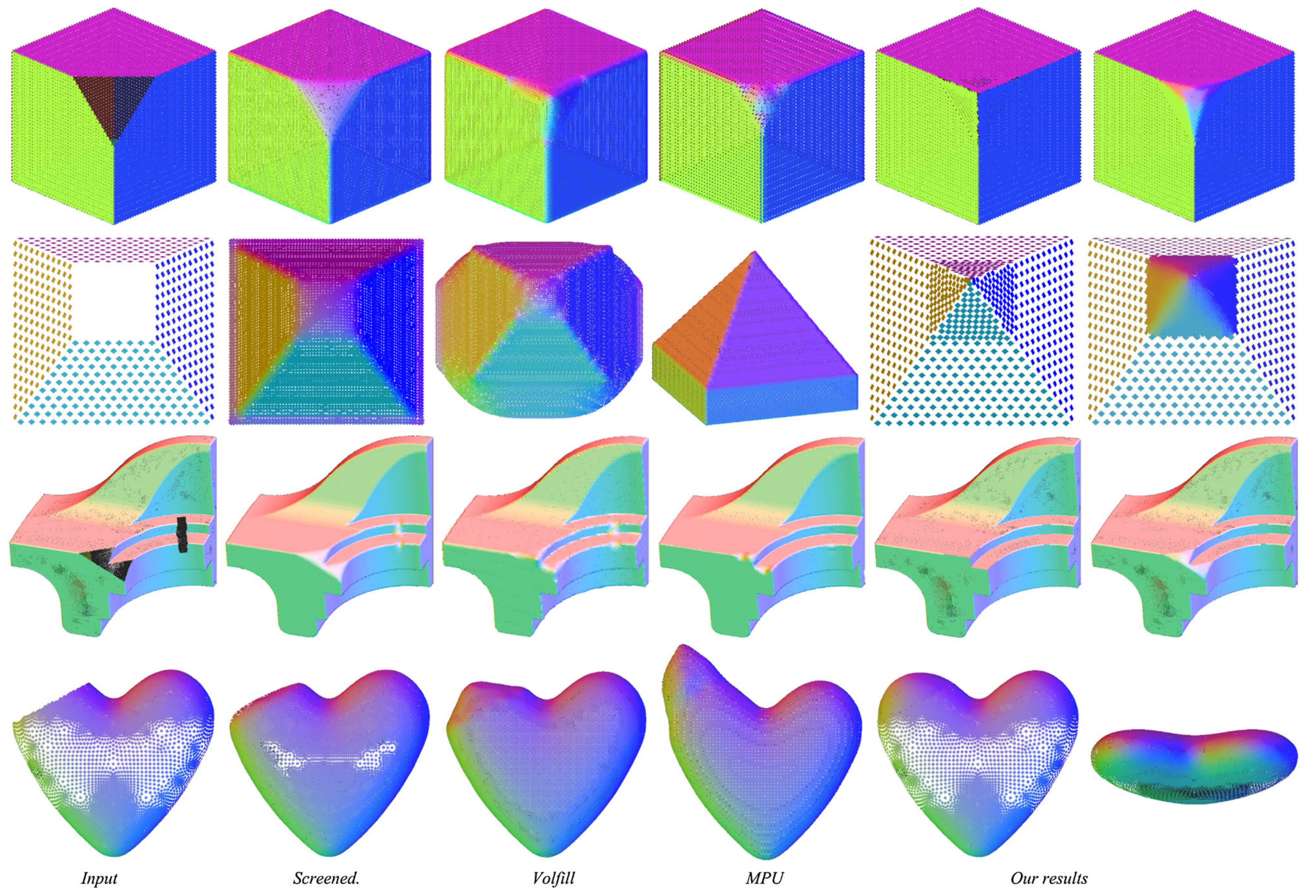


Fig. 8 The *first three rows* are the geometry completion results of a deficient cube, an incomplete pyramid and a destroyed fandisk model. The results from the *second* to the *fifth columns* correspond to *screened Poisson reconstruction*, *Volfill*, *MPU* and our methods, respectively. The *last column* is the round surface generated by our approach. The *fourth row* is a deficient "heart" shape surface. It is completed by *screened Poisson reconstruction*, *Volfill*, *MPU* and our algorithm successively. The *rightmost* is the *top view* of our result

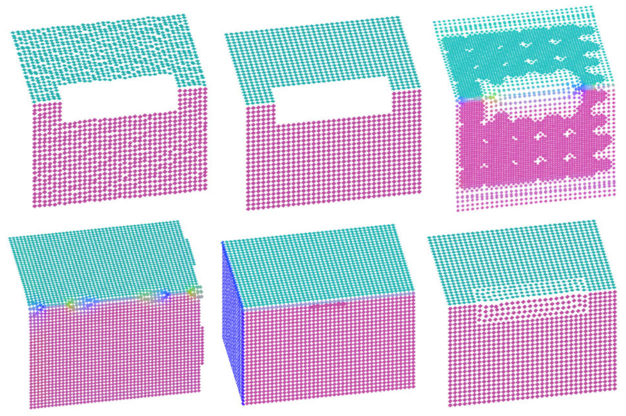


Fig. 9 Preprocessing and completing a dihedral model. Figures from *top-left* to *bottom-right* are the input deficient model, denoised dihedral and hole-filled results generated by *screened Poisson reconstruction*, *Volfill*, *MPU* and our method, respectively

ments an anisotropic filtering preprocessing (Sect. 3.2) to get a relatively neat model. We detect a hole boundary on the denoised model and then implement the geometry completion.

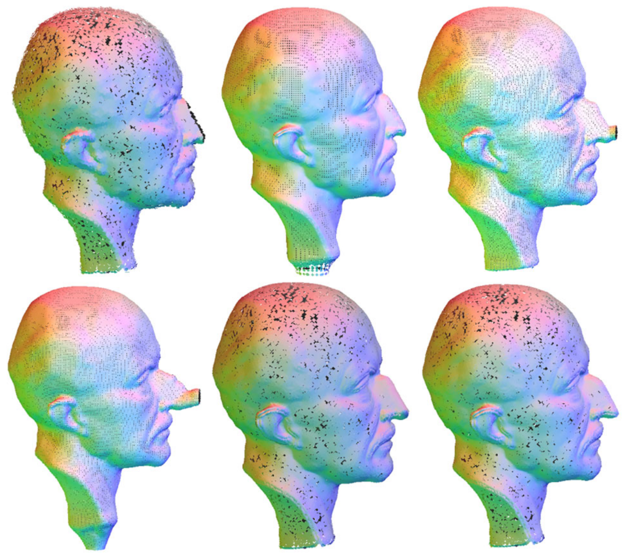


Fig. 10 Completing a noised Planck model. Figures from *top-left* to *bottom-middle* are the input noised model, hole-filled results produced by *screened Poisson reconstruction*, *Volfill*, *MPU* and our algorithm, respectively. The last figure is the ground truth model

Table 1 Our experimental settings of the core parameters and the statistic data for most hole-filling cases

Model	Figures	r (BBL)	σ_n (BBL)	σ_r (BBL)	Orig. point (num.)	Hole-bound. point (num.)	Iterative times	Filled point (num.)
Sphere	1c	0.25; $r = r/1.1$	Sup	0.22	4320	120	14	738
Sphere	1e	0.05	Sup	0.22	4320	120	22	1004
Cube	8(1–5)	0.05	0.1	0.25	9192	51	12	448
Pyramid	8(2–5)	0.05	0.1	0.25	572	32	11	268
Dihedral	9(2–3)	0.05	0.1	0.25	1520	66	4	160
Planck nose	10	0.04	0.6	0.14	25,052	31	78	508
			0.9	0.13		26	8	
Sculpture	11	0.013	Sup	0.12	105,727	148	21	2077
Printer	12	0.015	1.5	0.13	64,647	68	15	998
Hand (11 holes)	13	–	–	–	203,723	–	–	10,387
Sweeping surface (circle)	15	0.03	Sup	0.16	8313	51	15	769

“Sup” means loosening the normal dissimilarity constraint. “Figure 8(1–5)” denotes the 5th figure of the 1st row in Fig. 8

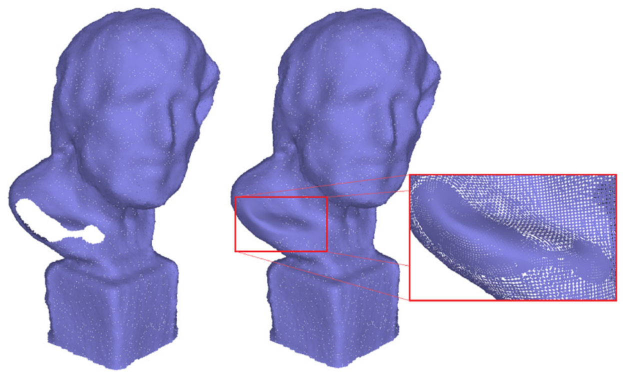


Fig. 11 A destroyed sculpture (*left*) is completed by our method. The details of the recovered region is shown in a close-up view (*right*)

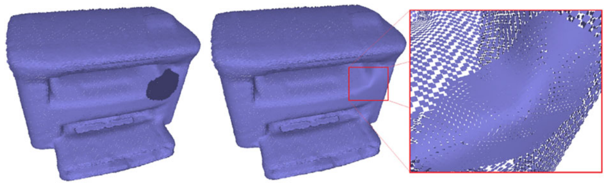


Fig. 12 Our method handles a scanned printer (*left*) and generates the hole-filled result. The close-up view is also given (*right*)

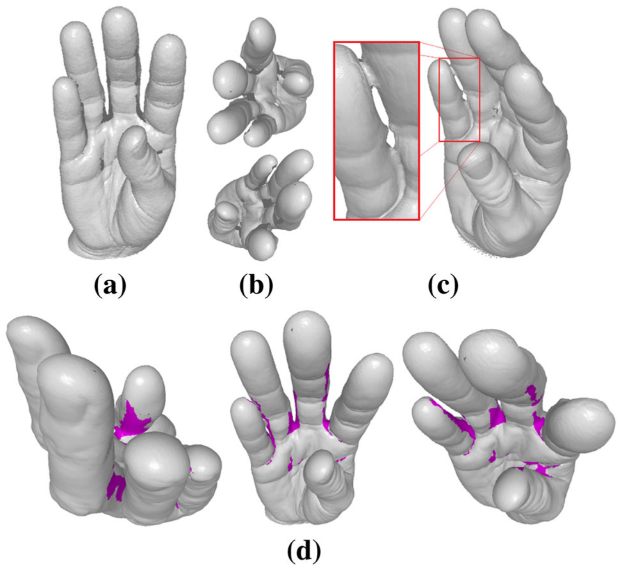


Fig. 13 A scanned hand (**a**) is first denoised (**b**) and then completed by the *screened Poisson reconstruction* method (**c**). **d** The repaired results of our method from different viewpoints

Figure 10 is the Planck model with the destroyed nose. Since we want to generate the straight nose bridge and the round nose tip, we separate this hole boundary into two parts. The straight nose bridge is first generated with a relatively small σ_n, as listed in Table 1. Finally we fill the round nose tip with a little bit bigger σ_n, as shown in Fig. 10.

A scanned sculpture model, in Fig. 11, contains a large deficiency which is composed of two connected holes. During the boundary contracting process, our method marks the encountered boundary parts as the non-updatable boundary control points and skips position sampling in these regions (see the accompanying video). The trajectories of boundary propagation demonstrate that the combination of control curve and elastic force fulfills our position sampling appropriately. Figure 12 shows a scanned printer model with coarse input normals. Our algorithm is not sensitive to the accuracy of initial normals. The deficient corner (containing both concave and convex features) is recovered by our method.

A scanned hand with many holes is displayed in Fig. 13. Too close a distance between adjacent fingers leads to mutual

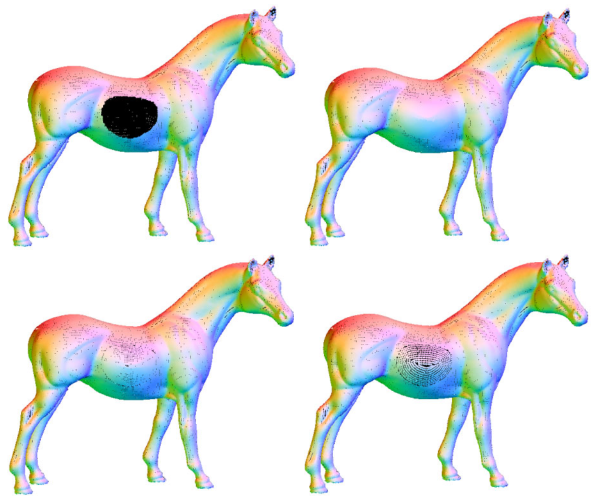


Fig. 14 Completing a horse model. Figures from *top-left* to *bottom-right* are input model, recovered results with different elastic force parameters σ_r corresponding to 0.14 *BBL*, 0.15 *BBL* and 0.16 *BBL*, respectively

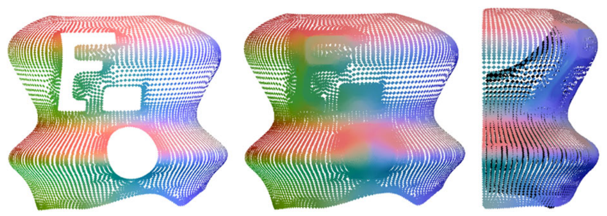


Fig. 15 Recovering a sweeping surface with three different holes by using our method. The *left* is the input deficient sweeping surface. The *middle* and the *right* are our completed results from different viewpoints

interference when *screened Poisson reconstruction* is implemented. Our method avoids this influence by taking the constraint of normal dissimilarity. We fill the complex hole region using piecewise boundary contraction. Figure 13 shows our repaired hand from different viewpoints (for more details, see the accompanying video).

Our method could natively treat the smooth holes. We loosen the normal dissimilarity constraint to complete a deficient horse and a sweeping surface with open boundary. The results are shown in Figs. 14 and 15, respectively.

There are three parameters that need to be assigned for our algorithm, including the neighborhood radius r, the parameter of normal dissimilarity σ_n and the elastic force parameter σ_r. A relatively small σ_n, corresponding to a strict normal dissimilarity constraint, is needed for filling holes around sharp regions. σ_r has direct proportion relationship with the repulsive force. Large repulsive force means less sampling points, while small repulsive force produces more sampling points (see Fig. 14). The values of these parameters for most examples are given in Table 1.

Table 2 Evaluation of the results generated by different methods on five models

	Cube	Pyramid	Dihedral	Planck	Fandisk
Scr.					
Max	0.2048	0.1266	0.0532	0.0221	0.0276
Ave	0.0070	0.0070	0.0048	0.0016	0.0008
Vol.					
Max	0.0729	0.0975	0.0982	0.0248	0.0153
Ave	0.0063	0.0050	0.0051	0.0017	0.0008
MPU					
Max	0.0619	0.0828	0.0495	0.0830	0.0141
Ave	0.0047	0.0059	0.0022	0.0051	0.0008
Our					
Max	0.0046	0.0114	0.0034	0.0099	0.0049
Ave	0.0031	0.0027	0.0019	0.0012	0.0001

Since the highlight of our method is repairing the geometry feature, we quantitatively evaluate our results in terms of recovering sharp features. Five models (cube, pyramid, dihedral, Planck and fandisk) are chosen. We normalize each model into a unit cube and calculate the errors for all points on each recovered model. The closest distance from a point on a repaired model to the ground truth surface is taken as the error measurement. Three synthesized complete surfaces (cube, pyramid and dihedral with sharp features) and two original models (unbroken Planck and fandisk) are taken as the ground truth.

Table 2 reports the maximum and average errors for all results generated by different methods. Note that the errors that occurred on the filled non-hole regions, including the bottom of the pyramid (*Volfill* and *MPU*), the left part of the dihedral (*MPU*) and the bottom of the Planck model (*screened Poisson*, *Volfill* and *MPU*), were excluded. In Table 2, our algorithm has the least values on both maximum and average errors for each model.

Figure 16 shows the colored errors for five models completed by four methods. Some obvious errors along sharp edges were found on results produced by *screened Poisson* and *Volfill* methods. The existing methods failed to complete sharp corners on several cases. Our method recovered faithful sharp features for these deficient edges and corners. The recovered trajectories on both pyramid and dihedral models show that our algorithm performs each step with a low repairing error.

Limitations The limitations of our method mainly exist in three aspects. First, it needs to select a few feature points on a hole boundary for generating the initial boundary. Thus, it is a semi-automatic approach. Second, just depending on the constrained local propagation, our method will fail if two-thirds of a sphere has been cut. For this kind of highly

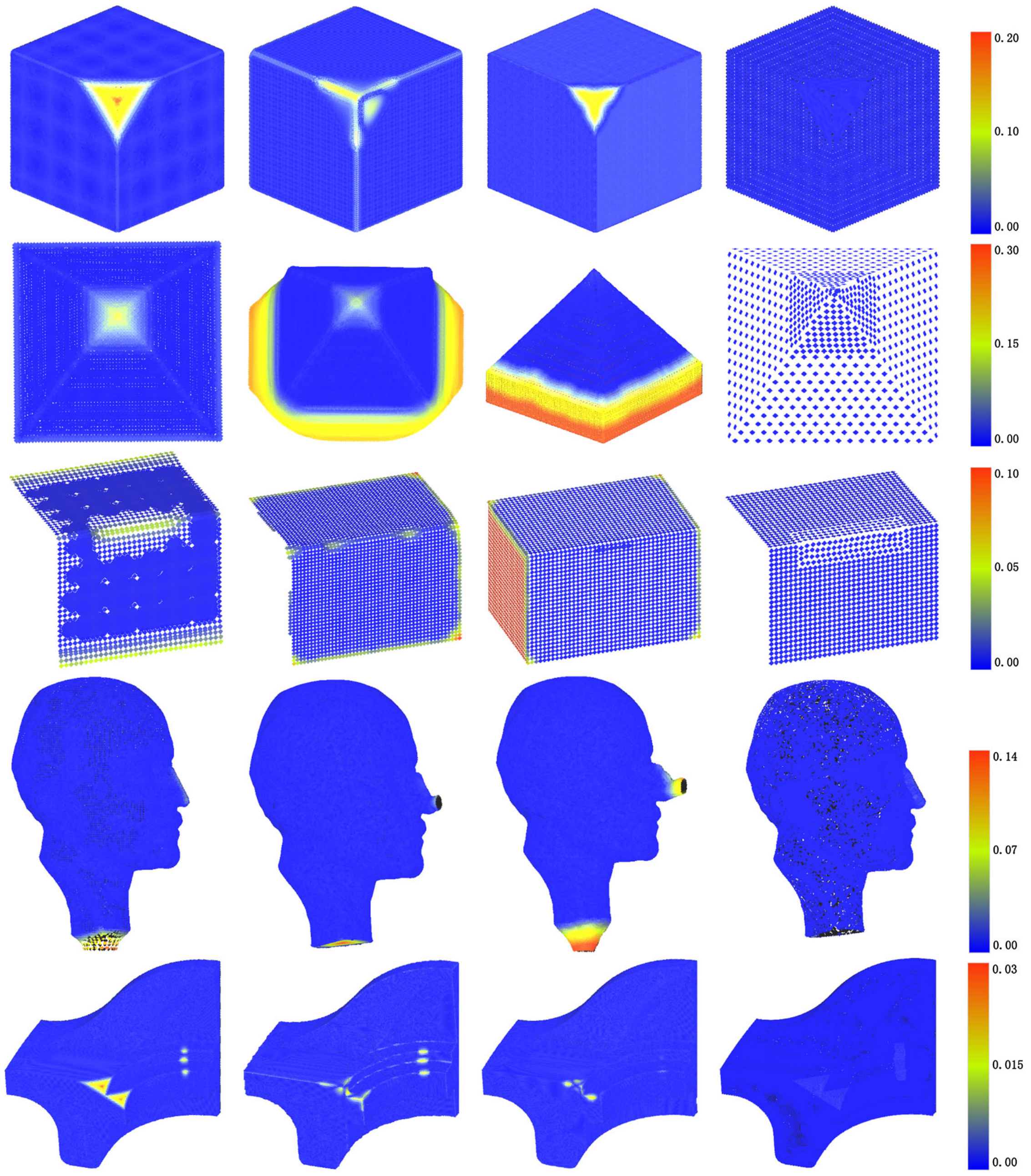


Fig. 16 Error plots of the quantitative evaluation. From *top* to *bottom*, cube, pyramid, dihedral, Planck and fandisk. From *left* to *right*, *screened Poisson*, *Volfill*, *MPU* and our method

ill-posed case which has more than half shape missed, more priori normal variations should be integrated in our normal propagation to generate the desired result. The last one is that our method currently focuses on shape recovery and does not treat the lost geometry details of a hole if it contains high-frequency features.

6 Conclusions

We devised a shape-controllable geometry completion algorithm for point cloud models. It provides potential shape options for those hole regions that probably contain sharp features. Our method inherits the merits of the local propagation

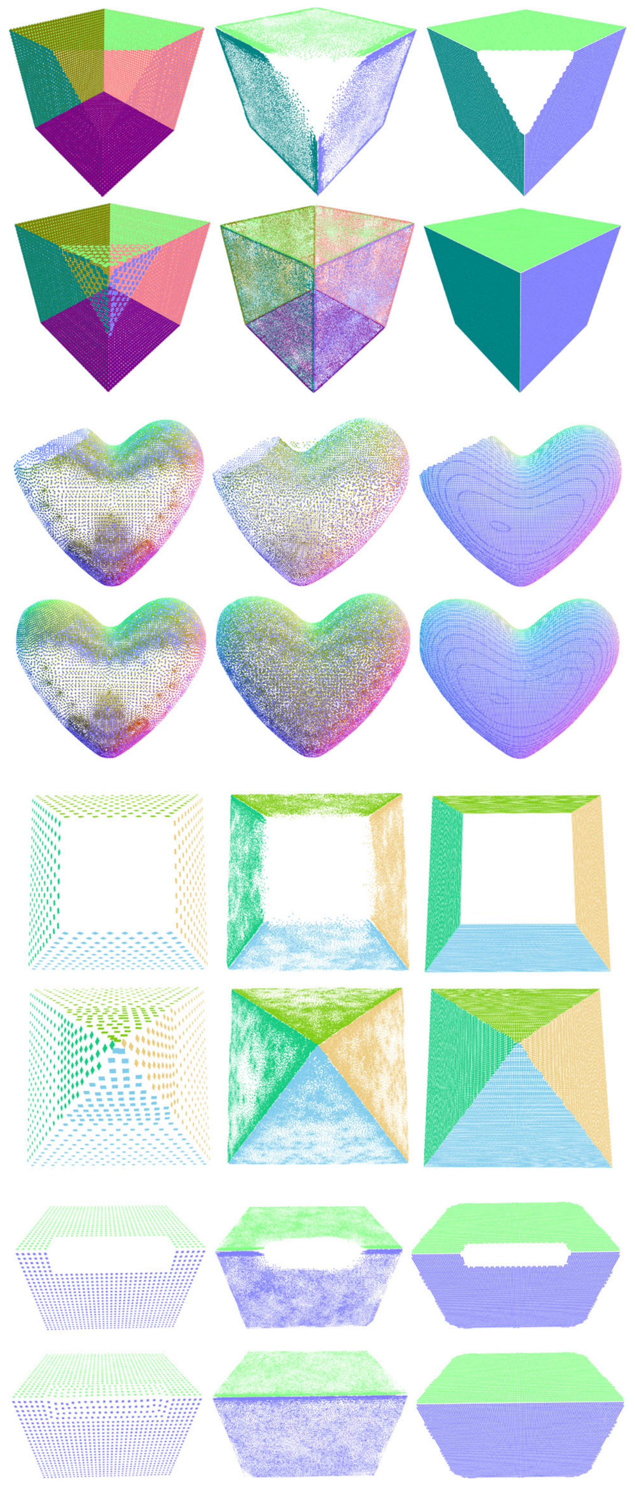


Fig. 17 Our approach benefits surface reconstruction. We implement two state-of-the-art algorithms (*EAR* [16] and *RIMLS* [26]) for sharp feature reconstruction on four deficient point cloud models. For each group, the *middle* and the *right figures* of the *first row* show the results of *EAR* and *RIMLS* methods directly working on the original point model, respectively. The *left figure* of the *second row* is our hole-filled result. The *middle* and the *right figures* of the *second row* are the corresponding results of *EAR* and *RIMLS* methods based on our result

pattern. It augments the capability of recovering sharp features by incorporating normal dissimilarity constraint into the decomposed normal propagation and position sampling operations. By defining the elastic force and introducing the boundary control curve, our method has appropriately addressed the sampling problem for point cloud hole filling. Those filled points shown in our experiments exhibit a reasonable distribution on the hole regions.

The completed point cloud model will practically benefit $3D$ surface reconstruction and many follow-up applications. With our hole-filled results, two sharp feature-preserved reconstruction methods of *EAR* [16] and *RIMLS* [26] generated intriguing results; see the reconstructed surfaces in Fig. 17.

In the future, we would like to develop an automatic hole-boundary *recognition* technique to enhance our geometry completion approach. Another natural thought of the following work is to explore the detail recovering method for those deficient point cloud surfaces containing high-frequency geometry features.

Acknowledgements The authors would like to thank the anonymous reviewers for their valuable comments and insightful suggestions. This work was partly supported by the National Basic Research Program of China (No. 2012CB725303), the NSFC (No. 61472288), NCET (NCET-13-0441), the Fundamental Research Funds for the Central Universities (2042015kf0181), and the State Key Lab of Software Engineering (SKLSE-2015-A-05).

Appendix: Computing the equilibrium position

To compute the equilibrium position $O_{b_i'}$ for a former pass hole-boundary point b_i', as shown in Fig. 3, we introduce a point q_0 whose position just exactly locates in $O_{b_i'}$, as depicted in Fig. 18. The equilibrium position of q_0 on the direction of vector $\overrightarrow{O_{b_i'} b_i'}$ must have the same position with point b_i', that is to say O_{q_0} coincides with the position of b_i'. Taking the elastic force received by O_{q_0} from b_i' into account, its value should be the positive maximum (equals 1, corresponding to the maximum repulsive force) according to the definition of the elastic force in Eq. (5). The overlap positions can be seen as the extremely close distance between O_{q_0} and b_i'. Without loss of generality, we assign this repulsive force along the vector $\overrightarrow{b_i' O_{b_i'}}$. Therefore, we have $r_{b_i'}(O_{q_0}) = 1$, specifically $1.0 - \exp\left(\left|O_{q_0} - O_{b_i'}\right|_{\overrightarrow{b_i' O_{b_i'}}} / \sigma_r^2\right) = 1$. By substituting O_{q_0} with b_i', we have $\exp\left(\left|b_i' - O_{b_i'}\right|_{\overrightarrow{b_i' O_{b_i'}}} / \sigma_r^2\right) = 0$.

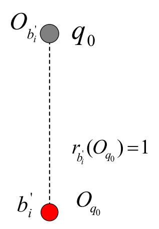


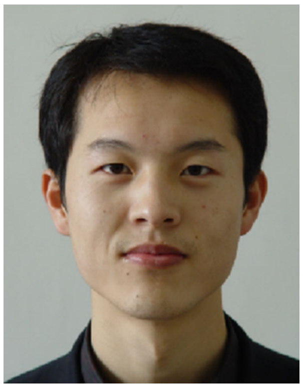
Fig. 18 Computing the equilibrium position $O_{b_i'}$ for a point b_i'

Our purpose is to compute the equilibrium position $O_{b_i'}$ for point b_i'. So, we need to take the logarithm for the above equation. However, the right side of this equation equals zero and cannot enforce a logarithm operation immediately. For the sake of numerical computing, we use a small constant 10^{-4} to approximate instead of zero and make our computation feasible. Finally, we can use the following equation to compute $O_{b_i'}$ for b_i' if the parameter σ_r is assigned,

$$\exp\left(|b_i' - O_{b_i'}|_{\overrightarrow{b_i' O_{b_i'}}} / \sigma_r^2 \right) = 10^{-4}.$$

References

1. Adamson, A., Alexa, M.: Approximating bounded, nonorientable surfaces from points. In: Shape Modeling Applications, 2004. Proceedings, IEEE, pp. 243–252 (2004)
2. Attene, M., Campen, M., Kobbelt, L.: Polygon mesh repairing: an application perspective. ACM Comput. Surv. (CSUR) **45**(2), 15 (2013)
3. Bac, A., Tran, N.V., Daniel, M.: A multistep approach to restoration of locally undersampled meshes. In: Advances in Geometric Modeling and Processing, pp. 272–289. Springer, Berlin (2008)
4. Bendels, G.H., Schnabel, R., Klein, R.: Detecting holes in point set surfaces. J WSCG. **14**, 89–98. ISSN 1213–6972 (2006)
5. Berger, M., Tagliasacchi, A., Seversky, L.M., Alliez, P., Levine, J.A., Sharf, A., Silva, C.T.: State of the art in surface reconstruction from point clouds. In: Eurographics 2014—State of the Art Reports, The Eurographics Association, pp. 161–185 (2014)
6. Campen, M., Attene, M., Kobbelt, L.: A practical guide to polygon mesh repairing. In: Proceedings of the 2012 Eurographics, p. t4. Cagliari, Italy (2012)
7. Carr, J.C., Beatson, R.K., Cherrie, J.B., Mitchell, T.J., Fright, W.R., McCallum, B.C., Evans, T.R.: Reconstruction and representation of 3D objects with radial basis functions. In: Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, pp. 67–76. ACM (2001)
8. Chalmovianskỳ, P., Jüttler, B.: Filling holes in point clouds. In: Mathematics of Surfaces, pp. 196–212. Springer, Berlin (2003)
9. Chen, C.Y., Cheng, K.Y.: A sharpness-dependent filter for recovering sharp features in repaired 3D mesh models. IEEE Trans. Vis. Comput. Graph. **14**(1), 200–212 (2008)
10. Davis, J., Marschner, S.R., Garr, M., Levoy, M.: Filling holes in complex surfaces using volumetric diffusion. In: 3D Data Processing Visualization and Transmission, 2002. Proceedings. First International Symposium on, IEEE, pp. 428–441 (2002)
11. Fleishman, S., Drori, I., Cohen-Or, D.: Bilateral mesh denoising. In: ACM Transactions on Graphics (TOG), vol. 22, pp. 950–953. ACM (2003)
12. Fleishman, S., Cohen-Or, D., Silva, C.T.: Robust moving least-squares fitting with sharp features. In: ACM Transactions on Graphics (TOG), vol. 24, pp. 544–552. ACM (2005)
13. Gross, M., Pfister, H.: Point-Based Graphics. Morgan Kaufmann, Los Altos, CA (2011)
14. Harary, G., Tal, A., Grinspun, E.: Context-based coherent surface completion. ACM Trans. Graph. (TOG) **33**(1), 5 (2014)
15. Harary, G., Tal, A., Grinspun, E.: Feature-preserving surface completion using four points. In: Computer Graphics Forum, vol. 33, pp. 45–54 (2014)
16. Huang, H., Wu, S., Gong, M., Cohen-Or, D., Ascher, U., Zhang, H.R.: Edge-aware point set resampling. ACM Trans. Graph. (TOG) **32**(1), 9 (2013)
17. Jones, T.R., Durand, F., Desbrun, M.: Non-iterative, feature-preserving mesh smoothing. In: ACM Transactions on Graphics (TOG), vol. 22, pp 943–949. ACM (2003)
18. Ju, T.: Fixing geometric errors on polygonal models: a survey. J. Comput. Sci. Technol. **24**(1), 19–29 (2009)
19. Kazhdan, M., Hoppe, H.: Screened poisson surface reconstruction. ACM Trans. Graph. (TOG) **32**(3), 29 (2013)
20. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Proceedings of the Fourth Eurographics Symposium on Geometry Processing (2006)
21. Kobbelt, L., Botsch, M.: A survey of point-based techniques in computer graphics. Comput. Graph. **28**(6), 801–814 (2004)
22. Kraevoy, V., Sheffer, A.: Template-based mesh completion. In: Symposium on Geometry Processing, Citeseer, pp. 13–22 (2005)
23. Lévy, B.: Dual domain extrapolation. ACM Trans. Graph. (TOG) **22**(3), 364–369 (2003)
24. Ngo, H.T.M., Lee, W.S.: Feature-first hole filling strategy for 3D meshes. In: VISIGRAPP, pp. 53–68 (2012)
25. Ohtake, Y., Belyaev, A., Alexa, M., Turk, G., Seidel, H.P.: Multi-level partition of unity implicits. In: ACM SIGGRAPH 2005 Courses, p. 173. ACM (2005)
26. Öztireli, A.C., Guennebaud, G., Gross, M.: Feature preserving point set surfaces based on non-linear kernel regression. In: Computer Graphics Forum, vol. 28, pp. 493–501 (2009)
27. Park, S., Guo, X., Shin, H., Qin, H.: Shape and appearance repair for incomplete point surfaces. In: Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on, IEEE, vol. 2, pp. 1260–1267 (2005)
28. Pernot, J.P., Moraru, G., Véron, P.: Filling holes in meshes using a mechanical model to simulate the curvature variation minimization. Comput. Graph. **30**(6), 892–902 (2006)
29. Sharf, A., Alexa, M., Cohen-Or, D.: Context-based surface completion. In: ACM Transactions on Graphics (TOG), vol. 23, pp. 878–887. ACM (2004)
30. Weyrich, T., Pauly, M., Keiser, R., Heinzle, S., Scandella, S., Gross, M.: Post-processing of scanned 3D surface data. In: Proceedings of the First Eurographics conference on Point-Based Graphics, Eurographics Association, pp. 85–94 (2004)
31. Xiao, C., Zheng, W., Miao, Y., Zhao, Y., Peng, Q.: A unified method for appearance and geometry completion of point set surfaces. Vis. Comput. **23**(6), 433–443 (2007)
32. Yin, K., Huang, H., Zhang, H., Gong, M., Cohen-Or, D., Chen, B.: Morfit: Interactive surface reconstruction from incomplete point clouds with curve-driven topology and geometry control. ACM Trans. Graph. **33**(6), 202:1–202:12 (2014)
33. Zhao, W., Gao, S., Lin, H.: A robust hole-filling algorithm for triangular mesh. Vis. Comput. **23**(12), 987–997 (2007)



Long Yang received his B.Sc. degree and M.Sc. degree from Chang'an University and Northwest A&F University, in 2004 and 2010, respectively. Currently, he is working toward his Ph.D. degree at the School of Computer, Wuhan University. His research interests include computer graphics, digital geometry processing and point-based graphics.

Qingan Yan is a Ph.D. candidate at the School of Computer, Wuhan University. His research interests include computer vision, computational photography and image collection-based 3D reconstruction.

Chunxia Xiao received his B.Sc. and M.Sc. degrees from the Mathematics Department of Hunan Normal University in 1999 and 2002, respectively, and Ph.D. degree from the State Key Lab of CAD & CG of Zhejiang University in 2006. Currently, he is a professor at the School of Computer, Wuhan University, China. From October 2006 to April 2007, he worked as a postdoc at the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, and during February 2012 to February 2013, he visited University of California-Davis for 1 year. His research areas include computer graphics, computer vision and image and video processing.