Machine Vision and Applications (2014) 25:1747–1759
DOI 10.1007/s00138-013-0582-1

SPECIAL ISSUE PAPER

Detail-generating geometry completion for point-sampled geometry

Ren-fang Wang · Yun-peng Liu · De-chao Sun · Hui-xia Xu · Ji-fang Li

Received: 11 February 2013 / Revised: 19 October 2013 / Accepted: 4 November 2013 / Published online: 15 December 2013

Abstract In this paper, we present a novel method for detail-generating geometry completion over point-sampled geometry. The main idea consists of converting the context-based geometry completion into the detail-based texture completion on the surface. According to the influence region of boundary points surrounding a hole, a smooth patch covering the hole is first constructed using radial base functions. By applying region-growing clustering to the patch, the patching units for further completion with geometry details is then produced, and using the trilateral filtering operator formulated by us, the geometry-detail texture of each sample point on the input geometry is determined. The geometry details on the smooth completed patch are finally generated by optimizing a constrained global texture energy function on the point-sampled surfaces. Experimental results demonstrate that the method can achieve efficient completed patches that not only conform with their boundaries, but also contain the plausible 3D surface details.

Keywords Geometry completion · Texture synthesis · Trilateral filtering operator · Geometry detail

1 Introduction

With the rapid popularity of web 2.0 content, enormous multimedia files are emerging every day [1]. So users can easily become overwhelmed when faced with vast quantity of information returned by web search engines. In order to provide users with a pleasing overview, therefore, extensive attentions have been drawn to how to process multimedia data quickly, extract their characteristics accurately and visualize them intuitively [2–4]. The contextual information is an important one employed usually in the fields of image processing, computer vision and computer graphics so on. In this paper, we use the contextual texture to complete the missing point-sampled geometry with details.

Point-sampled geometry (PSG) without topological connectivity is normally generated by sampling the boundary surface of physical 3D objects with 3D-scanning devices [5]. Due to occlusions, low reflectance, and measurement error in the scanning, the acquired PSG is normally imperfect, i.e., it contains holes. In addition, surface editing operations can result in large holes in the surface. These holes have to be filled in a manner that not only conforms to the global shape of entire surface but also generates plausible geometric details, that is, the patched geometry should not only be smoothly connected at the boundaries, but also contain geometry details similar to those on the existing surfaces. Such completion remains a challenge due to the irregularity of sampling and hole boundaries.

In recent years, a variety of geometry completion methods for PSG have been introduced, and they can be roughly categorized into the following three groups. (1) Implicit surface-based completion: using radial basis functions (RBFs) [6–11] or moving least squares (MLS) [12,13], the implicit surfaces at the hole region are reconstructed and sampled so as to achieve geometry completion. However, patches generated by these methods are normally smooth and do not convey geometric details. (2) Similarity-based completion: according to geometry similarities in 3D [14–18], the missing regions are completed by matching and copying 3D structures in the existing geometry or solving a PDE equation [15]. Although details are achieved by these methods, their complexities are relatively higher and they have to deal with the complex boundary conditions. (3) Example-based

R. Wang (✉) · Y. Liu · D. Sun · H. Xu · J. Li
College of Computer Science and Information Technology, Zhejiang Wanli University, Ningbo 315100, China
e-mail: renfangwang@gmail.com

completion: the completion is fulfilled with reference to a database of priori complete 3D objects from which similar surfaces are selected and blended to perform completion on the missing regions [19–21]. While achieving highly plausible reconstructions, example-based completion very much depends on the existence of suitable surface parts fitting into the missing area. This is clearly limited to the availability of such suitable a priori knowledge.

Considering image completion [22–25], its aim is to fill holes in an image by extending information available in the remaining image. Accordingly, geometry completion is similar in spirit to the image counterpart. Inspired by Kwatra et al. [26] who formulated the problem of image texture synthesis as minimization of an energy function, we introduce a novel method for detail-generating geometry completion over PSG, based on global optimization of texture quality. The basic idea in our method is to consider geometry completion as detail-texture completion on PSG, which is performed via minimizing a constrained global energy using the existing detail textures on the surfaces as the input texture sample. In detail, our major contributes are as follows:

- Based on variance and average curvature, we present a method for determining the influence region of boundary points, which can calculate this region automatically according to the complexity of hole boundary. In terms of influence region, we can utilize RBFs to create a smooth patch matching with the existing geometry seamlessly.
- We design a trilateral filtering operator and use it to compute the geometry-detail texture of each point on the existing geometry, which is regarded as the input texture sample. Moreover, we construct the regular texture grid of sample point by eigen analysis of the covariance matrix of its local neighborhood, which can act as a counterpart to the square neighborhood of pixel.
- We compute the geometry details on the smooth completed patch by optimizing a constrained global texture energy function. To this end, we employ region-growing clustering to segment the smooth completed surface over hole into *patching units*. Using texture synthesis technique based on optimizing an energy function, the local geometry-detail textures on patching unit are synthesized. Consequently, we achieve detail-generating geometry completion by converting textures back to geometry details.

The rest of the paper is organized as follows. In Sect. 2, we briefly review some related work. We give an overview of our method in Sect. 3, and describe smooth surface completion and geometry-detail completion in detail in Sects. 4 and 5, respectively. Experimental results are presented in Sect. 6. We conclude the paper in Sect. 7.

2 Related work

Many traditional methods focus mainly on surface-based completion with a smooth created patch filling the missing region and satisfying the boundary conditions. Carr et al. [6] used globally supported RBF to fit data points by solving a large dense linear system. Ohtake et al. [7] and Xiao [27] proposed a hierarchical approach for 3D scattered data interpolation with compactly supported RBF or algebraic sphere. Based on MLS, Wang and Oliveira [13] and Ahn et al. [12] presented an algorithm for filling holes on surfaces reconstructed from point clouds. Although these methods can effectively fill missing regions with smooth surface patches, unnatural shapes may be generated when the missing parts are large and surrounding shape is complex since they cannot generate a complex surface.

In order to reconstruct geometry detail, the completion methods based on context have recently been developed. Savchenko and Kojekine [28] warped a given shape model towards the missing region using control points followed by performing a fairing step along the boundary of the hole. Sharf et al. [14] extended the texture synthesis technique from 2D image to PSG so as to complete the missing geometry. As in the 2D case of texture synthesis, the characteristics of the given surface are analyzed, and the hole is iteratively filled by copying patches from the data regions. In this paper, the proposed method reconstructs the geometric details on the smooth patch by completing texture in the geometry-detail texture space. Consequently, our method can avoid some complex operations such as the similarity measurement between point sets and rigid transformation of the 3D points set as compared to Sharf et al.'s [14].

In order to generate complex shapes in missing regions, on the other hand, the methods based on examples have been proposed. Pauly [19] presented a method using a database of 3D shapes to provide geometric priors for regions of missing data. The method retrieves suitable context models from the database, warps the retrieved models to conform with the input data, and consistently blends the warped models to obtain the final consolidated 3D shape. Clearly, the method is limited to the availability of such suitable a priori knowledge. Similarly, the methods have also been developed based on employing example shapes in other parts of the model [20,29]. They calculate the similarities of shape between missing and data regions, and generate complex shape by copying the similar patch to the missing region. The algorithm proposed in the literature [20] requires that there exists a similar patch in the model whose size is almost the same as a missing region. The algorithm proposed in the literature [29] copies similar local patches successively to the missing region from the outer to the inner parts. While they can achieve complex surfaces, a discontinuous surface is easily produced on the seam in the completed model since the

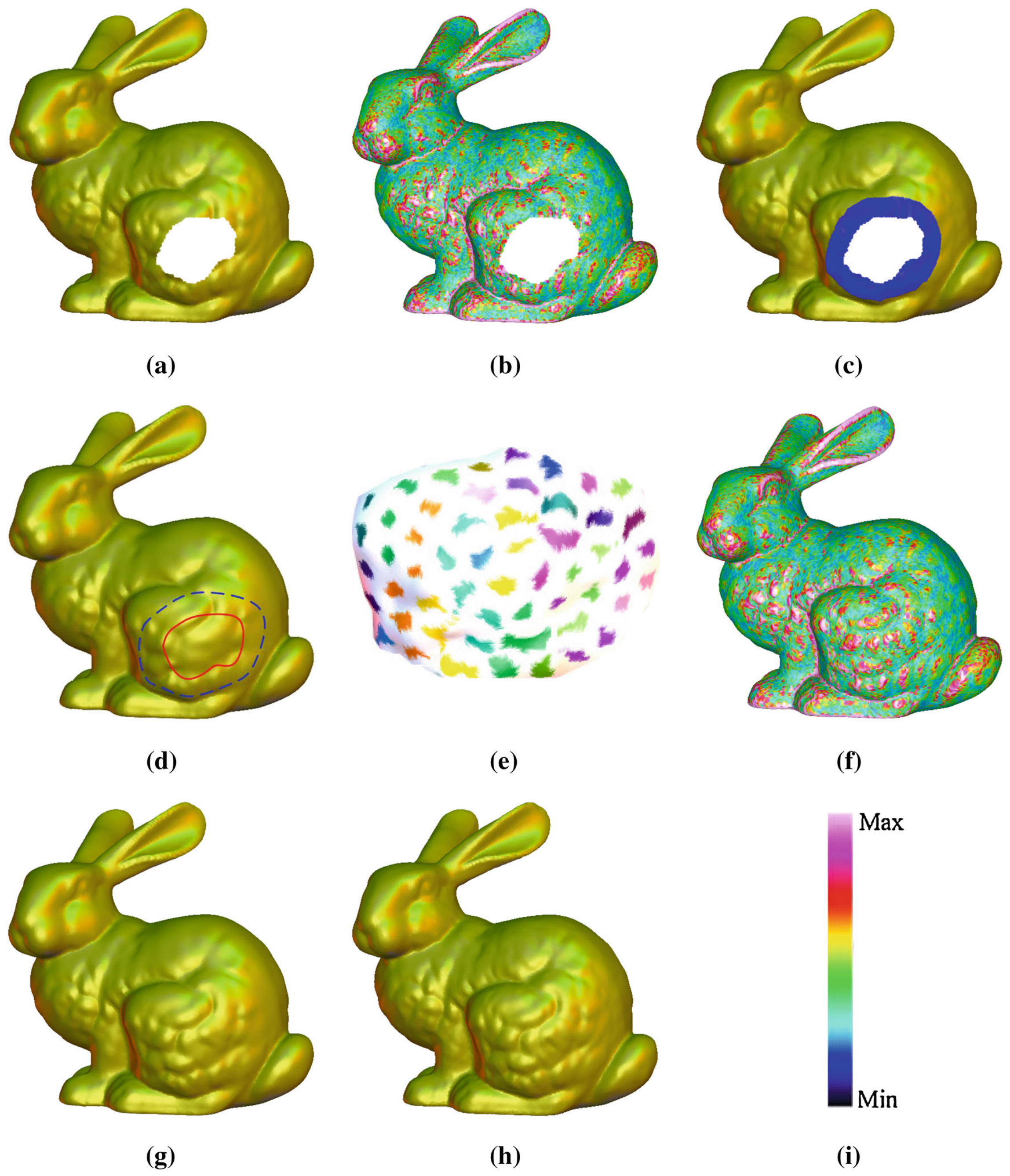


Fig. 1 Overview of our geometry completion pipeline. **a** The defective PSG; **b** the geometry-detail textures of **a**; **c** the influence region (*in blue*) of boundary points; **d** the smooth completed patch (the result of [3]); **e** the patching units; **f** the combination of **b** and the generated geometry details on **e**; **g** the completed PSG; **h** the actual PSG for comparison; **i** the palette for visualization of geometry detail and error (color figure online)

surface is fixed once it is copied. using RBFs on the influence region of hole boundary, in this paper, our algorithm guarantees that the reconstructed patches smoothly blend into the data region. By optimizing a constrained global texture energy function, moreover, the algorithm generates the geometry details on the smooth completed patch and can thus complete the missing regions successfully without a discontinuous surface.

3 Overview

Figure 1 gives an overview of geometry completion pipeline and our surface completion method consists mainly of the following four steps:

1. Smooth surface completion. After detecting the boundary points at hole, we first define their influence region (e.g.,

Fig. 1c) based on variance and average curvature. Using RBFs on the influence region, the hole is then filled with a smooth surface patch (e.g., Fig. 1d).

2. Generation of the patching units. By exploiting the region-growing clustering method, the smooth completed surface over hole is segmented into uniform patches, which are referred to as *patching units* in this paper, and each unit overlaps with its neighboring ones (e.g., Fig. 1e).
3. Construction of geometry-detail texture grid. We adopt the trilateral filtering operator to determine the geometry-detail texture of each sample point on the input geometry (e.g., Fig. 1b). Based on covariance analysis, further, the regular texture grid of sample point on a geometry is constructed, which is as a counterpart to the square neighborhood of a pixel in 2D image space.
4. Synthesis of geometry-detail texture. According to patching units and texture grids, the local geometry-detail textures of hole region are synthesized (e.g., Fig. 1f) using texture synthesis technique based on optimizing a constrained global texture energy function. After converting textures back to geometry details, we finally achieve detail-generating geometry completion over PSG (e.g., Fig. 1g).

4 Smooth surface completion

Before we can start reconstructing the missing surface in hole region, we first have to detect the hole boundary. As PSG is unstructured surface representation with no adjacency or connectivity information, the boundary detection is a nontrivial task. In this paper, we use the approach presented in [30], that robustly identifies loops of boundary points. The basic workflow of this hole detection scheme is as follows: for each point, a boundary probability is computed, which reflects the probability that the point is located on or near hole in the surface sampling. By utilizing the boundary coherence, the boundary points are then identified and the closed loops circumscribing hole are constructed in a shortest cost path manner. For more detailed introduction for detecting boundary points and extracting boundary loops, please refer to [30].

Once the boundary loop of hole has been extracted, the hole is filled with a smooth surface patch. We need to first define the influence region of boundary loop, i.e., the vicinity points of hole, which are employed to approximate the hole using RBFs. Clearly, one could simply choose neighborhood of fixed size of the boundary points as the influence region. In order to ensure that the reconstructed patches blend with the data regions $P_D = \{p_1, \ldots, p_n\}$ in a PSG in a smooth way, we introduce a method based on variance and average curvature, which can calculate the influence region automatically according to the complexity of hole boundary. The method consists of the following three steps.

Step 1. By covariance analysis, a reference plane is constructed, which is the best-fit plane to the boundary point set B of hole. The covariance matrix [31] for B is defined as

$$C = \begin{bmatrix} \boldsymbol{p}_{B_1} - \bar{\boldsymbol{p}}_B \\ \boldsymbol{p}_{B_2} - \bar{\boldsymbol{p}}_B \\ \vdots \\ \boldsymbol{p}_{B_m} - \bar{\boldsymbol{p}}_B \end{bmatrix}^{\mathrm{T}} \cdot \begin{bmatrix} \boldsymbol{p}_{B_1} - \bar{\boldsymbol{p}}_B \\ \boldsymbol{p}_{B_2} - \bar{\boldsymbol{p}}_B \\ \vdots \\ \boldsymbol{p}_{B_m} - \bar{\boldsymbol{p}}_B \end{bmatrix},$$

where $\boldsymbol{p}_{B_i} \in B$ $(i = 1, \ldots, m)$ is a boundary point and $\bar{\boldsymbol{p}}_B = \sum \boldsymbol{p}_{B_i}/m$ is the centroid of B. Since $\boldsymbol{C}$ is symmetric and positive semi-definite, all eigenvalues λ_i $(i = 0, 1, 2)$ are of real value and the unit eigenvectors $\boldsymbol{v}_i$ $(i = 0, 1, 2)$ form an orthogonal basis. Assuming that $\lambda_0 \leq \lambda_1 \leq \lambda_2$, the two eigenvectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ span the reference plane and $\boldsymbol{v}_0$ represents the plane normal, i.e., the reference plane $(\boldsymbol{x} - \bar{\boldsymbol{p}}_B)\boldsymbol{v}_0 = 0$ through $\bar{\boldsymbol{p}}_B$ minimizes the sum of the squared distance to these boundary points.

Step 2. The distance variance E from each boundary point to the reference plane is given by $E = \frac{1}{m}\sum_{i=1}^{m}(d_i - u)^2$, where d_i is the distance of boundary point $\boldsymbol{p}_{B_i}$ from the reference plane and u is the mean of distance (i.e., $u = \frac{1}{m}\sum_{i=1}^{m} d_i$). The average curvature of boundary curve at knot points is calculated as $\bar{k} = \frac{1}{l}\sum_{i=1}^{l} k_i$, where l is the knot number of boundary curve and k_i is the curvature of curve at a knot point.

Step 3. The boundary curve is first projected onto the reference plane, as illustrated by the green dashed curve in Fig. 2. We can then determine the convex hull of the projected curve on this plane, as illustrated by the yellow curve in Fig. 2, and enlarge the convex hull to generate its offset curve with an

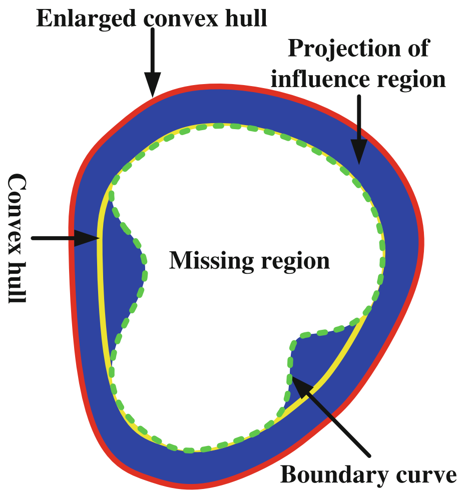


Fig. 2 Determining the projection of influence region (*colored in blue*). The *green dashed curve* denotes the boundary curve and the *yellow curve* is its convex hull. The *red curve* is the offset curve of this convex hull (color figure online)

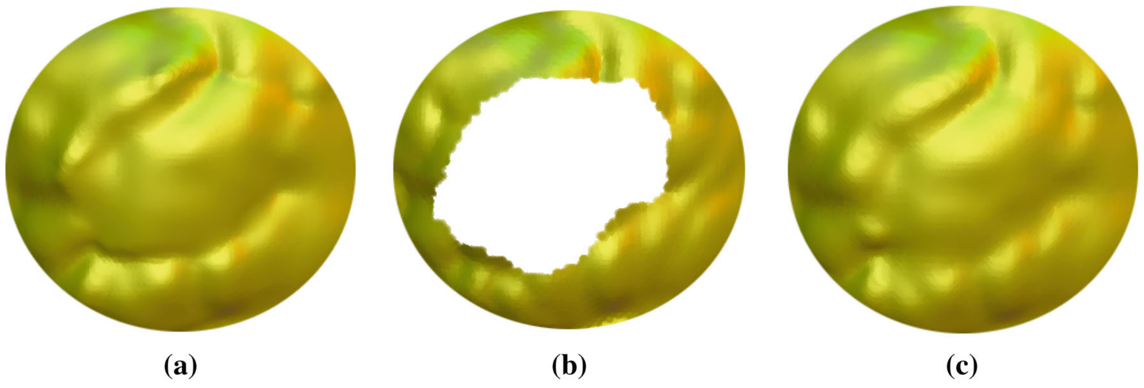


Fig. 3 Comparison of smooth completion. **a** Interpolation of the influence region determined by k-nearest neighbors of boundary points; **b** hole; **c** interpolation of the influence region determined by our method

offset distance, as illustrated by the red curve in Fig. 2. The region between the boundary curve and the enlarged convex hull is determined as the projection of influence region, as demonstrated by the blue region in Fig. 2. Accordingly, we can define the influence region as the part corresponding to this projection.

Because of the complexity of boundary curve, there may be self-intersection in its projected curve. Considering this situation, we use the convex hull of the projected curve to represent it in Step 3. When enlarging this convex hull, the offset distance R is derived automatically as $R = \alpha E + \beta \overline{k}$ $(\alpha + \beta = 1)$ according to the distance variance and average curvature.

The influence region identified by the above method is utilized to interpolate the missing portion of surface using RBFs presented in [7]. In [7], a coarse-to-fine hierarchy of scattered data is constructed using spatial down sampling. The coarsest level is first interpolated and a point set of the hierarchy is interpolated as an offsetting of the interpolating function computed at the previous level. According to [7], the number of subdivision levels is taken as $\lceil -\log_2(\sigma_0/(2\sigma_1))\rceil$, where σ_0 is the support size for the single-level interpolation and σ_1 is set to $0.75L$ (L is the length of a diagonal of the bounding parallelogram). For more detailed introduction, please refer to [7]. So the hole could be filled with the smooth surface patch and new points for filling the hole can be obtained by re-sampling the patch using the marching cubes algorithm [32]. Therefore, smooth surface completion is accomplished by adding these re-sampled points, each of which is referred to as a *smooth completed point*, to the original PSG. Figure 1c shows the influence region (in blue) of hole in bunny model, used as input for RBF interpolation. Figure 1d shows the smooth completed patch filling the hole.

In Fig. 3, we illustrate comparison of smooth completion via interpolating the influence region determined respectively by k-nearest neighbors of boundary points and our method. It can be noticed that the result generated by our method is preferable over the former. Our method by enlarging the convex hull of boundary curve leads to a comparatively regular region such that the reconstructed patch can blend into the data region more smoothly. Notice that, in this paper each model is rendered using a point-based rendering technique presented by us in [33].

5 Geometry-detail completion

We discuss this stage of our algorithm in four subparts. First, we briefly review an approach for texture synthesis based on global optimization proposed in [26], and specifically describe the generation of patching units, construction of geometry-detail texture grid and synthesis of geometry-detail texture, respectively.

5.1 Globally optimized texture synthesis

Kwatra et al. [26] presented an approach for 2D texture synthesis based on global optimization of texture quality with respect to a similarity metric, which is based on Markov random field (MRF) similarity criterion. They optimized the entire texture with an Expectation Maximization (EM)-like algorithm by defining a global texture energy to be $E_t(\mathbf{x}; \{\mathbf{z}_p\}) = \sum_{p \in X^\dagger} ||\mathbf{x}_p - \mathbf{z}_p||^2$, where $\mathbf{x}$ is the vectorized version of the target texture X that we want to synthesize, i.e., it is formed by concatenating the intensity values of all pixels in X. For a pre-specified neighborhood width w, N_p represents the neighborhood in X centered around pixel p and $\mathbf{x}_p$ denotes the sub-vector of $\mathbf{x}$ that corresponds to the pixels in N_p. Moreover, Z denotes the input sample to be used as reference and $\mathbf{z}_p$ is the vectorized pixel neighborhood in Z whose appearance is most similar to $\mathbf{x}_p$ under the Euclidean norm. For computing the energy, only the neighborhoods centered around pixels in a set $X^\dagger \subset X$ are considered. Consequently, a subset of neighborhoods is picked that sufficiently overlap with each other and the energy is defined only over this subset [26].

Table 1 Symbol and notation

R_c	Radius of bounding sphere of a cluster
R_C	Magnified radius
$\|E\|$	Average *edge length* of PSG
c_i	Geometry-detail texture
$N_k()$	k-Nearest neighbors
$N_{R_C}()$	Neighborhood (radius is equal to R_C)
C_{Ti}	Patching unit
$\boldsymbol{G}_{Ti}$	Texture grid of patching unit
$\boldsymbol{G}_{Si}$	Input texture grid
c_t^{ij}	Texture value of grid point in $\boldsymbol{G}_{Ti}$
c_s^{ij}	Texture value of grid point in $\boldsymbol{G}_{Si}$.
c_{tx}^{ij}	First partial derivative in the x direction in $\boldsymbol{G}_{Ti}$
c_{sx}^{ij}	First partial derivative in the x direction in $\boldsymbol{G}_{Si}$
c_{ty}^{ij}	First partial derivative in the y direction in $\boldsymbol{G}_{Ti}$
c_{sy}^{ij}	First partial derivative in the y direction in $\boldsymbol{G}_{Si}$
d	Texture similarity

Further, the controllable texture synthesis is performed by augmenting the texture energy function with an additional term representing the control criteria. The energy representing this criterion is expressed as the sum of squared distances between the synthesized and specified pixel values, i.e., $E_c(\mathbf{x}; \mathbf{x}^c) = \sum_{k \in \Phi} (\mathbf{x}(k) - \mathbf{x}^c(k))^2$, where Φ is the set of constrained pixels and $\mathbf{x}^c$ is a vector containing the specified color values. So the total optimized energy is defined as

$$E(\mathbf{x}) = E_t(\mathbf{x}; \{\mathbf{z}_p\}) + \lambda E_c(\mathbf{x}; \mathbf{x}^c) \qquad (1)$$

where λ is a relative weighting coefficient.

In this paper, we extend this controllable texture synthesis to PSG for geometry detail completion. For the reason that image pixels are usually aligned on a regular and equispaced grid, which is in general not true for a PSG, the main issue of extending it consists of how to build the correspondence between irregularly 3D sampling points and regular 2D texture samples. Before we explain our method in detail, let us introduce some symbols and notations in Table 1 that will be used in the following subsection.

5.2 Generation of the patching units

In order to attain the patching units, we utilize the region-growing clustering method [31] to divide the point set including all the input points and smooth completed points. Their clustering process is summarized as follows: starting from a random seed point p_1, a cluster C_1 is built by successively adding nearest neighbors. This incremental region-growing is terminated when the size of the cluster reaches a maximum bound. The next cluster C_2 is then constructed by starting the incremental growth with a new seed $\boldsymbol{p}_2$ chosen from the neighbors of C_1 and excluding all points of C_1 [31]. This procedure is repeated until each point in the point set is distributed into a some cluster.

We adapt this method by taking the terminated condition of incremental region-growing as $R_c > s|E|$, where R_c is the radius of bounding sphere of a cluster C_i, whose center is at the centroid of C_i; $|E| = \sum_{i=1}^{n} r_{i\ \min}/n$ is the average *edge length* of P_D, $r_{i\ \min}$ is the distance between $\boldsymbol{p}_i$ and its nearest point; s is a relative coefficient adjusting the radius size, that is set to $s \in [1.7, 4.0]$ in our implementation. In order to maintain the coherence of the completed model, as such, we magnify the radius R_c to ensure that the neighboring clusters overlap sufficiently with each other. We take the magnified radius R_C as $R_C = R_c + \delta|E| = (s + \delta)|E|$, where δ is a relative coefficient controlling the overlapping region between the clusters. Here, the augmented cluster including at least a smooth completed point is referred to as a *patching unit* (as shown in Fig. 1e), represented by the notation C_{Ti}.

In Fig. 4, we show the surface clustering of Igea model using our region-growing clustering method. The clusters without overlapping region are illustrated in Fig. 4a, the clusters with smaller overlapping area in Fig. 4b, and clusters with larger overlapping area in Fig. 4c. In Figs. 1e, 4b–c, 6d and 8d, the white color indicates the overlapping area of neighboring clusters.

5.3 Construction of geometry-detail texture grid

As our idea of detail-generating geometry completion is to adapt the controllable texture synthesis to the 3D surfaces, we need to produce the textures of sample point in P_D, which can represent the geometry details of input PSG. Laplacian coordinates can describe the local details of the surface, which has been widely used in digital geometry processing of triangle meshes such as editing [34] and smoothing [35] so on. Notice that the Laplacian coordinates of vertex are defined as the difference between the Cartesian coordinates of vertex and the center of mass of its 1-ring neighbors in the mesh. The choice of its weights decides the properties of Laplacian coordinates, and three popular choices are the uniform weights, the cotangent weights and the mean-value weights [34]. Since PSG contains no topological connectivity, k-nearest neighbors are substituted for 1-ring neighbors in the mesh and weight can only be taken as the uniform one or its variation, which do not consider the normal and curvature variations between sample point and its neighbor. However, the normal and curvature variations reflect the intrinsic surface feature.

So we design the following trilateral filtering operator, which is similar to our previous scheme [36], to determine the geometry-detail texture c_i of each point p_i in P_D

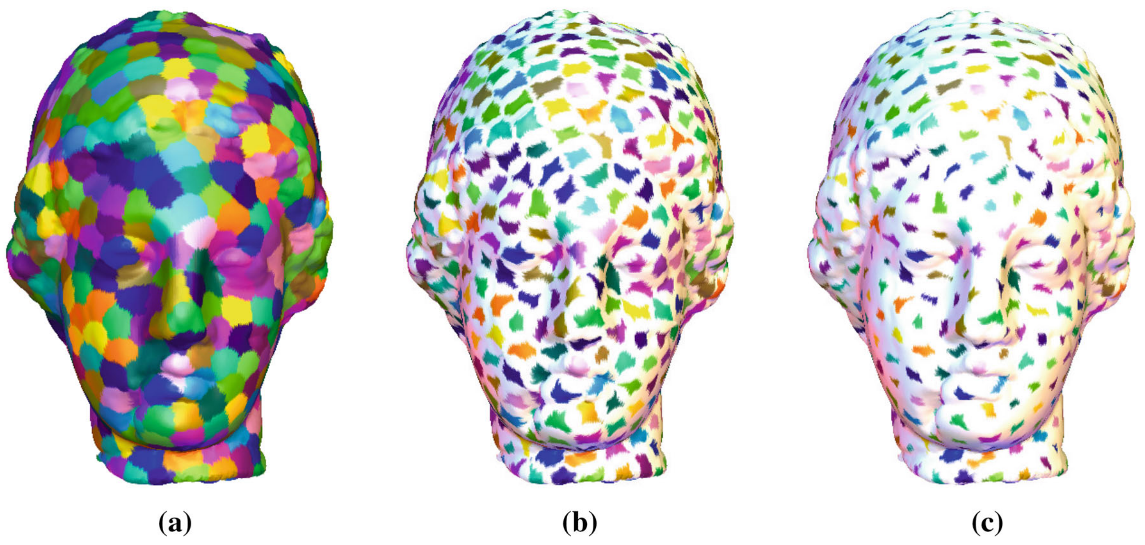


Fig. 4 Clustering of Igea model. **a** Clusters without overlapping region; **b** clusters with smaller overlapping area; **c** clusters with larger overlapping area

$$\begin{cases} c_i = \frac{\sum_{q_{ij}\in N_k(p_i)} w_{ij}\langle \boldsymbol{n}_i, \boldsymbol{q}_{ij}-\boldsymbol{p}_i\rangle}{\sum_{q_{ij}\in N_k(p_i)} w_{ij}} \\ w_{ij} = w_c(||\boldsymbol{q}_{ij}-\boldsymbol{p}_i||)w_s(||\langle \boldsymbol{n}_i, _{ij}-\boldsymbol{p}_i\rangle||)w_h(||k_{H_{ij}}-k_{H_i}||) \end{cases} ,\quad (2)$$

where $\boldsymbol{n}_i$ is the surface normal at the point $\boldsymbol{p}_i$, $N_k(\boldsymbol{p}_i)$ is the neighborhood of p_i, k_{H_i} is the mean curvature and $w(x)$ is a Gaussian kernel: $w_c(x) = \exp(-x^2/2\sigma_c^2)$, $w_s(x) = \exp(-x^2/2\sigma_s^2)$ and $w_h(x) = \exp(-\exp(-||k_{H_{ij}} - k_{H_i}||^2/2))$. The term $w_h(x)$ denotes that the influence on the weight w_{ij} increases with an increase in the curvature difference $(k_{H_{ij}} - k_{H_i})$ so that the neighbor with a bigger curvature difference has a bigger weight. Accordingly, the geometry-detail texture calculated by this trilateral filtering operator reflects the local geometry feature efficiently. In practice, we take the parameter σ_c as $\sigma_c = r/2$, where r is the radius of the enclosing sphere of $N_k(\boldsymbol{p}_i)$, and σ_s as the standard deviation of the projections of the vector $(\boldsymbol{q}_{ij} - \boldsymbol{p}_i)$ onto $\boldsymbol{n}_i$. Because the operator considers not only the point positions and normals but also the curvatures; the geometry-detail texture computed by it can reflect the local geometry detail at each point more effectively. Figure 1b, f demonstrates the detail-texture value visualization for the defective and completed model, respectively. Figure 1i illustrates the palette for visualization of detail texture.

After determining the geometry-detail texture of each point, we can construct the regular texture grid $\boldsymbol{G}_i$ of sample point $\boldsymbol{p}_i$ on PSG as follows, which is as a counterpart to the square neighborhood of pixel in 2D image space. Based on covariance analysis again (described in detail in Sect. 4), the eigenvalues and unit eigenvectors of covariance matrix of $N_{R_C}(\boldsymbol{p}_i)$ are first determined. The local coordinate frame $[\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}]$ with origin $\boldsymbol{p}_i$ is then established: let $\boldsymbol{Z}$ be $\boldsymbol{v}_0$ if $\boldsymbol{v}_0 \cdot \boldsymbol{n}_i > 0$, be $-\boldsymbol{v}_0$ otherwise; let $\boldsymbol{Y}$ be $\boldsymbol{v}_1$ and $\boldsymbol{X} = \boldsymbol{Y} \times \boldsymbol{Z}$.

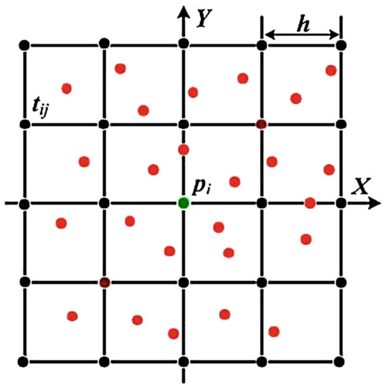


Fig. 5 Construction of texture grid

Further, we can calculate each projection $\boldsymbol{q}_j$ of the vector $(\boldsymbol{q}_{ij} - \boldsymbol{p}_i)$ onto the plane $\boldsymbol{X}\boldsymbol{p}_i\boldsymbol{Y}$, where $\boldsymbol{q}_{ij}$ with the texture value c_j is one of $\boldsymbol{p}_i$'s neighbors (i.e., $\boldsymbol{q}_{ij} \in N_{R_C}(\boldsymbol{p}_i)$). Consequently, a regular texture grid $\boldsymbol{G}_i$ of $M \times M$ centered at $\boldsymbol{p}_i$ can be built through the following process: (1) compute the radius R_G of bounding circle of the 2D point set $\{\boldsymbol{q}_j\}$ in the plane $\boldsymbol{X}\boldsymbol{p}_i\boldsymbol{Y}$, whose center is at $\boldsymbol{p}_i$, and the $\boldsymbol{G}_i$'s interval is defined as $h = 2R_G/(M-1)$. (2) Determine the texture value of each grid point $\boldsymbol{t}_{ij}$ as the Gaussian-weighted average of texture values of its k-nearest neighbors $N_k(\boldsymbol{t}_{ij})$

$$c_{ij} = \sum_{\boldsymbol{q}_j \in N_k(\boldsymbol{t}_{ij})} (w_t(\boldsymbol{q}_j) \cdot c_j) \Big/ \sum_{\boldsymbol{q}_j \in N_k(\boldsymbol{t}_{ij})} w_t(\boldsymbol{q}_j) \quad (3)$$

where w $(\boldsymbol{x})$ is a Gaussian kernel, we take $w_t(\boldsymbol{q}_j) = \exp(-||\boldsymbol{q}_j - \boldsymbol{t}_{ij}||^2/(2\sigma_t^2))$ and $\sigma_t = R_G/2$. In practice, k is set to 9 and M is set to 5 or 7. Figure 5 illustrates the constructed grid with the projected points highlighted in red.

5.4 Synthesis of geometry-detail texture

We apply the constrained texture synthesis algorithm (described in Sect. 5.1) to PSG for geometry detail completion. As the controllable texture synthesis is developed in the 2D image setting, we have built the correspondence between point-sampled surfaces and 2D texture samples so as to optimize Eq. (1) in PSG setting. The geometry-detail set $\{c_i\}$ in Sect. 5.3 is regarded as the input texture, i.e., Z in Eq. (1); the patching unit set $\{C_{Ti}\}$ in Sect. 5.2 becomes the target texture that we want to synthesis, i.e., X in Eq. (1); in Sect. 5.3, the texture grid $\boldsymbol{G}_i$ that contains none of all the projection points of smooth completed point corresponds to the neighborhood of the pixel in the input sample Z, i.e., $\mathbf{z}_p$; the set of points in the influence region computed in Sect. 4 corresponds to the set of constrained pixels and the corresponding vector $\mathbf{x}^c$ is formed by concatenating the texture values of these points. Similar to that presented in Sect. 5.3, the initial texture grid $\boldsymbol{G}_{Ti}$ of a patching unit can also be established, which corresponds to $\mathbf{x}_p$ in Eq. (1). The only difference between them lies at the origin, that is we locate the origin of $\boldsymbol{G}_{Ti}$ at the point closest to the centroid of C_{Ti}.

In order to facilitate neighborhood matching, the similarity metric between two texture neighborhoods is expressed as the following two formulations

$$d = \sum_{i,j=1}^{M} (c_t^{ij} - c_s^{ij})^2 \tag{4}$$

$$d = \sum_{i,j=1}^{M} (|c_t^{ij} - c_s^{ij}| + |c_{tx}^{ij} - c_{sx}^{ij}| + |c_{ty}^{ij} - c_{sy}^{ij}|) \tag{5}$$

where c_t^{ij} represents the texture value of grid point in a target texture grid $\boldsymbol{G}_{Ti}$, and c_s^{ij} is the corresponding one in an input texture grid $\boldsymbol{G}_{Si}$. c_{tx}^{ij} and c_{sx}^{ij} mean the first partial derivatives in the x direction in the target and input grid respectively; c_{ty}^{ij} and c_{sy}^{ij} is the one in the y direction. The similarity metric in (4) is simply expressed as the sum of squared distances between the synthesized and given texture values, and however, the one in (5) takes into account not only texture value of grid but also texture anisotropism. By means of the metric in (4), we first find the k-nearest neighbors $N_k(\boldsymbol{G}_{Ti})$ of $\boldsymbol{G}_{Ti}$ in the set $\{\boldsymbol{G}_{Si}\}$ of input texture grids based on the locality-sensitive hashing technique [37], which can efficiently perform approximate nearest neighbor search in high dimensions. We then choose the metric in (5) to determine the best matching neighborhood in $N_k(\boldsymbol{G}_{Ti})$ with $\boldsymbol{G}_{Ti}$. According to the above two steps, in fact, we select the matching neighborhood in a coarse-to-fine manner so that the final one best matches $\boldsymbol{G}_{Ti}$ in terms of both texture value and anisotropism.

After establishing the correspondence between 3D sampling points and regular 2D texture samples, we employ the EM-like algorithm like [26] to optimize the constrained texture energy (i.e., Eq. 1) over X. Starting with the hole boundary, in initialization step, we successively find the best matching neighborhood $\boldsymbol{G}_{Si}$ in $\{\boldsymbol{G}_{Si}\}$ with the texture grid $\boldsymbol{G}_{Ti}$ of patching unit containing the constrained points, as per the above matching procedure. For the patching unit composed only of smooth completed points, we will assign a random neighborhood from $\{\boldsymbol{G}_{Si}\}$ to its grid. In the M-step, we need to minimize Eq. (1) with respect to the set $\{\boldsymbol{G}_{Si}\}$ of input neighborhoods; keeping $\{\boldsymbol{G}_{Ti}\}$ fixed at the value estimated in the E-step and for each $\boldsymbol{G}_{Ti}$, its nearest neighbor from $\{\boldsymbol{G}_{Si}\}$ is found out under the constrained conditions by applying our matching procedure. In the E-step, we need to minimize Eq. (1) with respect to $\{\boldsymbol{G}_{Ti}\}$, which is solved by the following procedure. By setting the derivative of Eq. (1) with respect to $\mathbf{x}$ to zero, a linear system of equations is yielded, that can be solved for $\mathbf{x}$ subject to the constrained vector $\mathbf{x}^c$. If the two patching units C_{Ti} and C_{Tj} overlap with each other, then $\boldsymbol{G}_{Ti}$ and $\boldsymbol{G}_{Tj}$ will also contain common *pixels*. Each of the common pixels may take different texture values from $\boldsymbol{G}_{Si}$ and $\boldsymbol{G}_{Sj}$. The minimization procedure assigns each common pixel an texture value that is equal to the average of the corresponding values in $\boldsymbol{G}_{Si}$ and $\boldsymbol{G}_{Sj}$.

By repeating the two steps iteratively until convergence, we can achieve the final synthesized texture, i.e., we have found the best-match input neighborhood $\boldsymbol{G}_{Si}$ for each patching grid $\boldsymbol{G}_{Ti}$ and the texture value c_i at point $\boldsymbol{p}_i$ enclosed in $\boldsymbol{G}_{Ti}$ can then be obtained using bilinear interpolation. After converting the texture values back to the geometry details and adding them to the corresponding points (i.e.,$\boldsymbol{p}_i' = \boldsymbol{p}_i + c_i \boldsymbol{n}_i$), consequently, we finally achieve the detail-generating geometry completion over PSG.

6 Experimental results

The proposed algorithm was implemented on a Microsoft Windows XP PC with Core i3 2.93GHz CPU and 4GB of memory. In all cases, the input to the algorithm consisted of a PSG. We demonstrate our method for successfully completing holes with geometry details in Figs. 1, 6 and 8. The defective Bunny model with 138,456 sample points, Igea model with 132,893 sample points and Rockarm model with 280627 sample points are illustrated respectively in Figs. 1a, 6a and 8a; Figs. 1d, 6c and 8c demonstrate the smooth completion respectively and Figs. 1g, 6f and 8f demonstrate the detail-generating completion respectively. Notice that by refining these initial models with our method, the missing regions are completed with geometry-detail surfaces smoothly blending into the initial surfaces.

In Fig. 7, we compare our geometry completion with the completion using the trilateral filtering operator and Laplacian operator, respectively. It can be noticed that our result

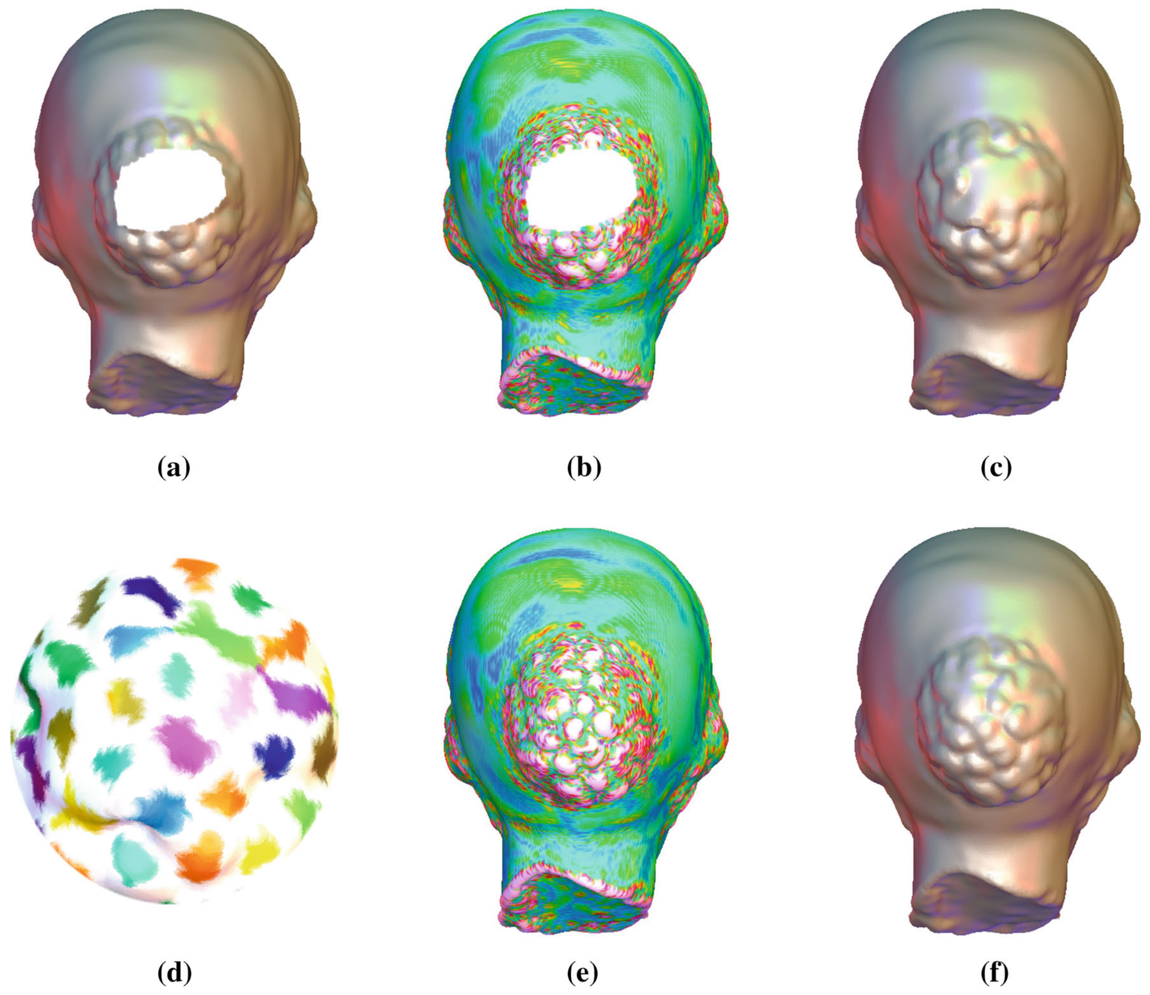


Fig. 6 Geometry completion of Igea model. **a** The defective model; **b** the geometry-detail textures of **a**; **c** the smooth completed patch (the result of [3]); **d** the patching units; **e** the generated geometry details on **d**; **f** the completed model

(Fig. 7d) visually appears slightly more convincing than the latter's (Fig. 7c). In order to evaluate the quality of the completed surfaces efficiently, we measure its error compared to the actual surfaces and errors are visualized using the palette in Fig. 1i. It is worth mentioning that all the input PSGs are scaled into a unit bounding box in advance and both the completed surfaces are re-sampled in the same manner. Table 2 shows the related data statistics for completing the defective PSGs. From the Max. and Avg. errors, we can notice that our method can generate a higher-quality completion result than the latter's. Clearly, error visualization (Fig. 7e–f) visually conveys this information more efficiently.

One sees the influence region of hole as shown in Fig. 1c used as input for the RBF interpolation so as to fulfill the smooth surface completion, as shown respectively in Figs. 1d, 6c and 8c. Since we adopt RBFs in [7], on the other hand, the smooth surface completion can be regarded as the filling result obtained with the method of [7]. For comparison, we demonstrate the completed model by our method in Fig. 1g and the actual model in Fig. 1h, and notice that such a completion is quite plausible. Meanwhile, we also show the close-up of the smooth completed patch (i.e., the result of [7]) respectively in Figs. 7b and 8g, the one of the detail-generating completed patch respectively in Figs. 7d and 8g, and the one of the actual model respectively in Figs. 7a and 8i. It can again be noticed that the hole geometry with detail surfaces generated by our method appears more convincing than the smooth filling [7]. This is mainly because our method can generate the plausible details on the smooth completed patch by optimizing a constrained global texture energy function. The geometry-detail texture determined by our trilateral filtering operator, on the other hand, can efficiently express the local geometry detail at each point.

Since there are relatively less sample points in the influence region, the cost of the smooth surface completion by the RBF interpolation is much less and the computational

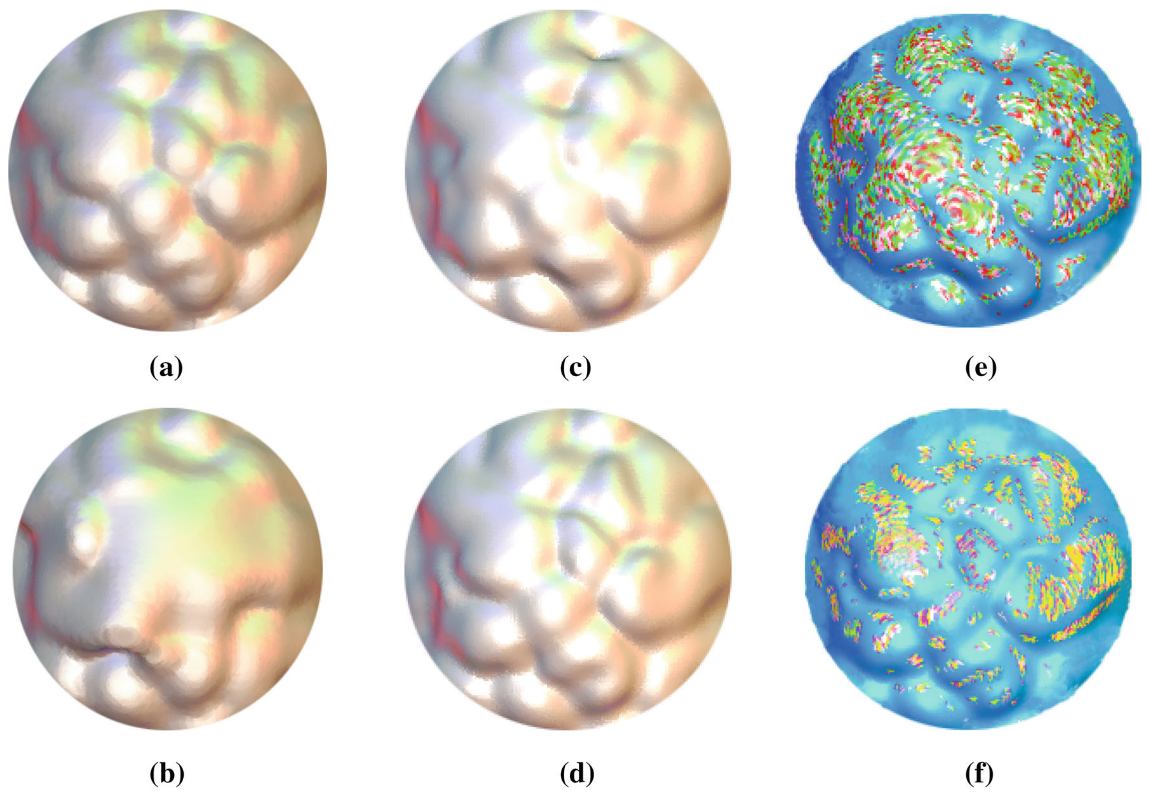


Fig. 7 Comparison of geometry completion. **a** The actual model; **b** the smooth completion (the result of [3]); **c** the result using Laplacian operator; **d** the result using trilateral filtering operator; **e** error visualization of **c**; **f** error visualization of **d**

Table 2 Statistics for completing the defective PSGs

PSG	T_S (s)	T_D (s)	Iters.	Max. error ($\times 10^{-4}$)		Avg. error ($\times 10^{-5}$)	
				TR	LP	TR	LP
Bunny	30	22	29	43	67	28.37	37.69
Igea	27	18	24	58	81	32.40	43.16
Rockarm	43	37	15	35	49	19.21	25.34

Here T_S stands for the execution time of smooth surface completion. T_D is the execution time of an iteration
Iters. the number of iterations. *TR* the trilateral filtering operator and *LP* Laplacian operator

complexity of our approach is dominated by the geometry-detail completion, which is performed by the globally optimized texture synthesis based on the nearest neighbor search. To accelerate this search, we utilize the locality-sensitive hashing technique to perform nearest neighbor search in high dimensions. In Fig. 1, the execution time of smooth surface completion is 30 s and in the stage of geometry-detail completion, the execution time for an iteration takes about 22 s and 29 iterations are needed. The former time is 27 s, in Fig. 6; the latter time is about 18 s and 24 iterations are needed. In Fig. 8, the former time is 43 s; the latter time is about 37 s and 15 iterations are needed.

7 Conclusion

In this paper, we have presented a novel approach for detail-generating geometry completion over PSG using the textual texture. The approach is mainly composed of two stages: smooth surface completion and geometry detail completion. Based on the influence region of hole boundary, in the first stage, we employ RBFs to create a smooth completed patch blending smoothly into the data region. Based on the globally optimized texture synthesis, in the second stage, the geometry-detail textures on the smooth patch are completed and the geometry detail completion is achieved by converting textures back to geometry details. Since the method for globally optimized texture synthesis relies heavily on the image structure regularity, the main difficulty of extending it to PSG is how to construct the regular texture grid as a counterpart to the square neighborhood of pixel. We first compute the geometry-detail texture of each point using the trilateral filtering operator. By exploiting the region-growing clustering method, the patching units overlapping with its neighboring ones are then obtained. Based on covariance analysis, the regular texture grid of sample point is finally constructed. Our experimental results demonstrate that this proposed algorithm can generate the surface shape in the missing region that not only blends smoothly into the data region, but also contains the plausible 3D surface details.

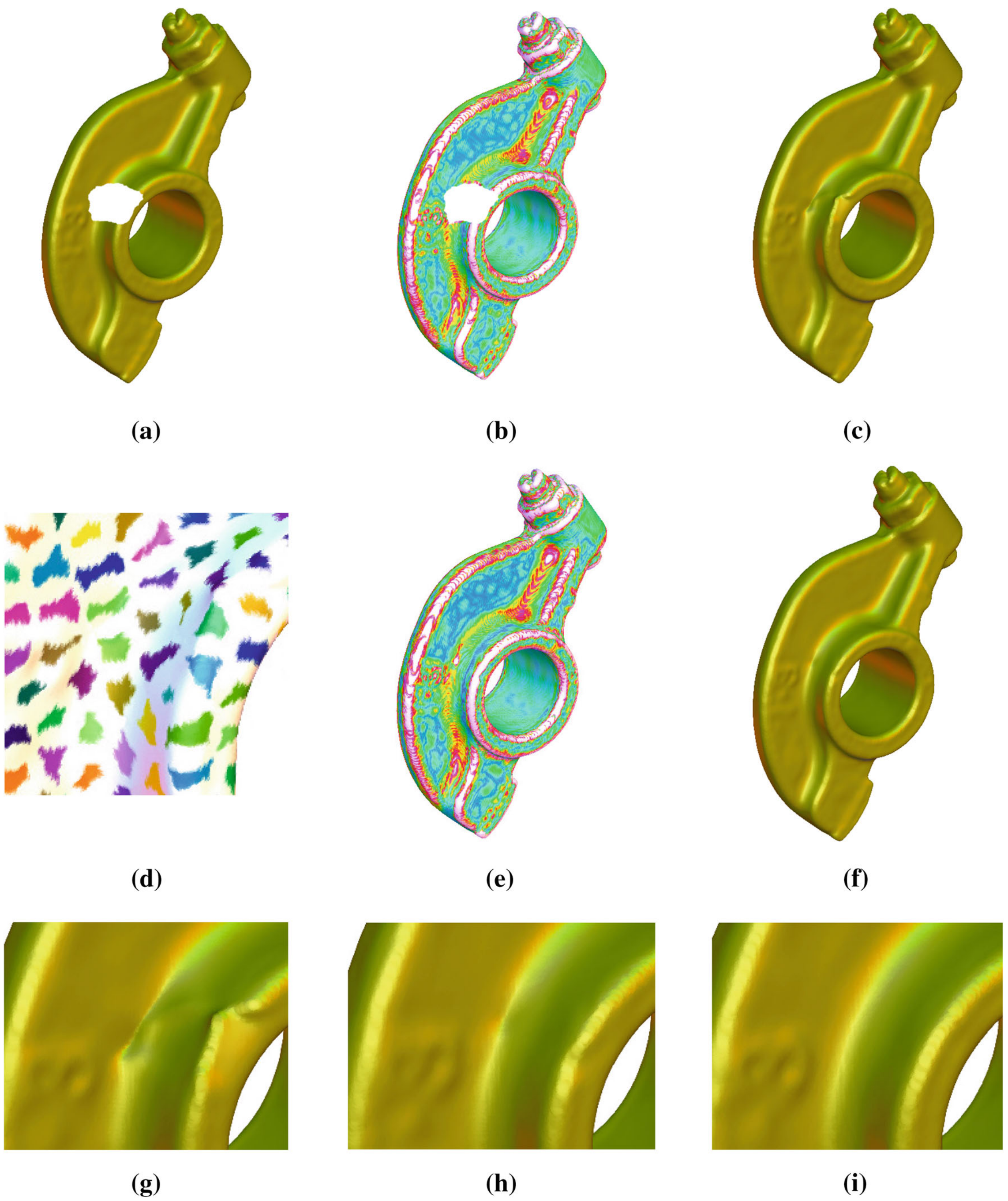


Fig. 8 Geometry completion of Rockarm model. **a** The defective model; **b** the geometry-detail textures of **a**; **c** the smooth completed patch (the result of [3]); **d** the patching units; **e** the generated geometry details on **d**; **f** the completed model; **g** a close-up of **c**; **h** a close-up of **f**; **i** a close-up of the actual model

Acknowledgments We would like to thank the anonymous reviewers for their valuable comments and insightful suggestions. The work is supported in part by the National Natural Science Foundation of China (Grant Nos. 61073074 and 61303144), the Natural Science Foundation of Zhejiang Province (Grant No. Y1090137), the Project of Science and Technology Plan for Zhejiang Province (Grant No. 2012C21004), Ningbo Natural Science Foundation (Grant Nos. 2013A610111 and 2013A610096) and the Project in Science and Technique of Ningbo Municipal (Grant No. 2012B82003). Models are courtesy of Stanford University and EU Aim@Shape project.

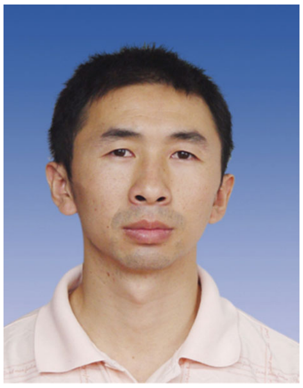
References

1. Hong, R.C., Tang, J.H., Tan, H.K., Ngo, C.W., Yan, S.C., Chua, T.S.: Beyond search: event-driven summarization for web videos. ACM Trans. Multimed. Comput. Commun. Appl. **7**(4), 35 (2011)
2. Wang, M., Hong, R.-C., Li, G.-D., Zha, Z.-J., Yan, S.-C., Chua, T.-S.: Event driven web video summarization by tag localization and key-shot identification. IEEE Trans. Multimed. **14**(4), 975–985 (2012)

3. Hong, R.-C., Wang, M., Li, G.-D., Nie, L.-Q., Zha, Z.-J., Chua, T.-S.: Multimedia question answering. IEEE MultiMed. **19**(4), 72–78 (2012)
4. Hong, R.-C., Zha, Z.-J., Gao, Y., Chua, T.-S., Wu, X.-D.: Multimedia encyclopedia construction by mining web knowledge. Signal Process. **93**(8), 2361–2368 (2013)
5. Wang, R.-F., Wang, Q., Xue, B.-B., Yang, Q., Li, J.-F.: Simplification of point-sampled geometry with feature preservation. IET Image Process. **5**(4), 299–305 (2011)
6. 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 SIGGRAPH'2001, pp. 67–76. Los Angeles (2001)
7. Ohtake, Y., Belyaev, A., Seidel, H.P.: A multi-scale approach to 3D scattered data interpolation with compactly supported basis functions. In: Proceedings of Shape Modeling International, pp. 153–161. Seoul (2003)
8. Casciola, G., Lazzaro, D., Montefusco, L.B., Morigi, S.: Fast surface reconstruction and hole filling using positive definite radial basis functions. Numer. Algorithms **39**(1–3), 289–305 (2005)
9. Chen, F.Z., Chen, Z.Y., Ding, Z., Ye, X.Z., Zhang, S.Y.: Filling holes in point cloud with radial basis function. J. Comput. Aided Design Comput. Graph. **18**(9), 1414–1419 (2006). (in Chinese with English abstract)
10. Süßmuth, J., Meyer, Q., Greiner, G.: Surface reconstruction based on hierarchical floating radial basis functions. Comput. Graph. Forum. **29**(6), 1854–1864 (2010)
11. Du, X.-W., Che, X.-J.: A hierarchical approach to 3D scattered data interpolation with radial basis functions. In: proceedings of Computer-Aided Design and Computer Graphics (CAD/Graphics), pp. 262–267. Jinan (2011)
12. Ahn, S.J., Yoo, J., Lee, B.G., Lee, J.J.: 3D surface reconstruction from scattered data using moving least square method. In: Proceedings of ICIAP, pp. 719–726. Cagliari (2005)
13. Wang, J., Oliveira, M.M.: Filling holes on locally smooth surfaces reconstructed from point clouds. Image Vis. Comput. **25**(1), 103–113 (2007)
14. Sharf, A., Alexa, M., Cohen-Or, D.: Context-based surface completion. ACM Trans. Graph. **23**(3), 878–887 (2004)
15. Park, S., Guo, X., Shin, H., Qin, H.: Shape and appearance repair for incomplete point surfaces. In: Proceedings of the International Conference on Computer Vision, pp. 1260–1267. Beijing (2005)
16. Kawai, N., Sato, T., Yokoya, N.: Surface completion by minimizing energy based on similarity of shape. In: Proceedings of ICIP'2008, pp. 1532–1535. San Diego (2008)
17. Kawai, N., Sato, T., Yokoya, N.: Efficient surface completion using principal curvature and its evaluation. In: Proceedings of ICIP'2009, pp. 521–524. Cairo (2009)
18. Kawai, N., Zakhor, A., Sato, T., Yokoya, N.: Surface completion of shape and texture based on energy minimization. In: Proceedings of ICIP'2011, pp. 897–900. Brussels (2011)
19. Pauly, M., Mitra, N.J., Giesen, J., Guibas, L., Gross, M.: Example-Based 3D Scan Completion. In: Proceedings of the Third Eurographics Symposium on Geometry Processing, pp. 23–32. Vienna (2005)
20. Park, S., Guo, X., Shin, H., Qin, H.: Surface completion for shape and appearance. Vis. Comput. **22**(3), 168–180 (2006)
21. Pauly, M., Mitra, N.J., Wallner, J., Pottmann, H., Guibas, L.: Discovering structural regularity in 3D geometry. ACM Trans. Graph. **27**(3), 1–11 (2008)
22. Sun, J., Yuan, L., Jia, J., Shum, H.-Y.: Image completion with structure propagation. ACM Trans. Graph. **24**(3), 861–868 (2005)
23. Komodakis, N., Tziritas, G.: Image completion using efficient belief propagation via priority scheduling and dynamic pruning. IEEE Trans. Image Process. **16**(11), 2649–2661 (2007)
24. Barnes, C., Shechtman, E., Finkelstein, A., Goldman, D.: PatchMatch: a randomized correspondence algorithm for structural image editing. ACM Trans. Graph. **28**(3), 1–11 (2009)
25. Yang, Y., Zhu, Y., Peng, Q.: Image completion using structural priority belief propagation. In: Proceedings of the 17th ACM International Conference on Multimedia, pp. 717–720. Beijing (2009)
26. Kwatra, V., Essa, I., Bobick, A., Kwatra, N.: Texture optimization for example-based synthesis. ACM Trans. Graph. **24**(3), 795–802 (2005)
27. Xiao, C.-X.: Multi-level partition of unity algebraic point set surfaces. J. Comput. Sci. Technol. **26**(2), 229–238 (2011)
28. Savchenko, V., Kojekine, N.: An approach to blend surfaces. In: Proceedings of Computer Graphics International, pp. 139–150. Bradford (2002)
29. Dong, J., Ma, S., Li, L., Yu, Z.: Hole filling on three-dimensional surface texture. In: Proceedings of ICME'2007, pp. 1299–1302. Beijing (2007)
30. Bendels, G.H., Schnabel, R., Klein, R.: Detecting holes in point set surfaces. J. WSCG **14**, 89–96 (2006)
31. Pauly, M., Gross, M., Kobbelt, L.P.: Efficient simplification of point-sampled surfaces. In: Proceedings of IEEE Visualization, pp. 163–170. Boston (2002)
32. Lorensen, W., Cline, H.: Marching cubes: a high resolution 3D surface construction algorithm. In: Proceedings of the SIGGRAPH'1987, pp. 163–169. Anaheim (1987)
33. Wang, R.-F., Li, J.-F., Yang, Q., Zhang, S.-Y.: Fast High-Quality Rendering of Point-Sampled Geometry. J. Comput. Aided Design Comput. Graph. **22**(2), 191–197 (2010). (in Chinese with English abstract)
34. Wang, H., Chen, H., Su, Z., Cao, J., Liu, F., Shi, X.: Versatile surface detail editing via Laplacian coordinates. Vis. Comput. **27**(5), 401–411 (2011)
35. Wei, M., Shen, W., Qin, J., Wu, J., Wong, T.T., Heng, P.A.: Feature-preserving optimization for noisy mesh using joint bilateral filter and constrained Laplacian smoothing. Opt. Lasers Eng. **51**(11), 1223–1234 (2013)
36. Wang, R.-F., Chen, W.-Z., Zhang, S.-Y., Zhang, Y., Ye, X.-Z.: Similarity-based denoising of point-sampled surfaces. J. Zhejiang Univ Sci. A **9**(6), 807–815 (2008)
37. Georgescu, B., Shimshoni, I., Meer, P.: Mean shift based clustering in high dimensions: a texture classification example. In: Proceedings of the ICCV'2003, pp. 456–463. Nice (2003)

Ren-fang Wang received the PhD from the School of Computer Science and Technology, Zhejiang University, in 2008, and is currently a full professor at the School of Computer Science and Information, Zhejiang Wanli University, China. His research interests include digital geometry processing and digital image processing.

Yun-peng Liu received the PhD from the School of Computer Science and Technology, Zhejiang University, in 2013, and is currently an associate professor at the School of Computer Science and Information, Zhejiang Wanli University, China. His research interests include video analysis, video coding, pattern recognition, and machine learning.



De-chao Sun is a PhD student in Signal and Information Processing at Ningbo University, and is also a senior engineer at the School of Computer Science and Information, Zhejiang Wanli University, China. His research interests include digital geometry processing and embedded system.

Hui-xia Xu received her PhD from Department of Mathematics, Zhejiang University, in 2008, and is currently an associate professor at Institute of Mathematics, Zhejiang Wanli University, China. Her research interests include computer-aided geometric design and digital geometry processing.

Ji-fang Li received the M.S. from the School of Computer Science and Technology, Huazhong University of Science and Technology, in 2008, and is currently a full professor at the School of Computer Science and Information, Zhejiang Wanli University, China. Her research interests include digital image processing and computer graphics.