SCIENCE CHINA
Information Sciences

• **RESEARCH PAPER** •

May 2012 Vol. 55 No. 5: 1123–1137
doi: 10.1007/s11432-011-4433-2

Simultaneous structure and geometry detail completion based on interactive user sketches

YANG Shen[1]*, QI Yue[1] & QIN Hong[2]

[1]*State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing* 100191*, China;*
[2]*Department of Computer Science, State University of New York at Stony Brook, Stony Brook, New York* 11794-4400*, USA*

Received May 25, 2010; accepted April 6, 2011

Abstract We articulate a novel approach to geometric model completion via interactive sketches in this paper. First, the initial incomplete model (with holes) is decomposed into a base model and a high-frequency component, which represents global rough shape and geometric details, respectively. We then repair the base model via smooth hole-filling, and compute the geometry detail image using high frequency information. One novel element of our approach is that we allow users to interactively sketch a few structural curves that span across hole regions, with a goal to repair both local geometric details and global structure. With the help of local parameterization, we convert detailed geometry into gradient-domain images which can propagate along user-specified sketches. By integrating recovered gradient-domain images and base shape, we can generate a complete model that faithfully recovers both global structure and local details. The salient contribution of this paper is the unified approach for user interaction, global structure, and geometry details towards high-fidelity model completion. We demonstrate our new approach using a number of examples that exhibit salient global structure as well as local geometry details.

Keywords model repair, geometric modeling, interactive sketch, 3D graphics

Citation Yang S, Qi Y, Qin H. Simultaneous structure and geometry detail completion based on interactive user sketches. Sci China Inf Sci, 2012, 55: 1123–1137, doi: 10.1007/s11432-011-4433-2

1 Introduction

3D models are ubiquitous in virtual reality, computer games, culture heritage, digital medicine, etc. Model repair remains a challenging problem for interactive 3D graphics due to imperfectness of data acquisition devices. Besides hardware limitation and unavoidable noise during data collection processes, acquired models may have holes due to other factors such as illumination, material, and self-occlusion. Furthermore, filling holes enabled by recovering operation during mesh editing is far from trivial. This paper aims to tackle the challenging problem of model completion. The novelty of our algorithm is founded upon the integrated method that simultaneously addresses the problem of missing global structure and local geometric details, and we achieve the goal of model completion through user interaction.

*Corresponding author (email: sheen.young@gmail.com)

1.1 Related work and motivation

A typical method of model repair is to produce appropriate patches to fill the holes with certain geometric information and satisfy the boundary condition. The commonly-used computational pipeline is as follows: 1) identify the hole boundary; 2) fill holes with appropriate patches; and 3) optimize the patches (e.g., Liepa's method in [1]). In [1], the advantage is that it not only satisfies the boundary condition but also approximates the density and smoothness of adjacent meshes. However, the repaired patches are intrinsically smooth without reconstructing the geometric details of surrounding regions. A different approach is based on volumetric embedding, yet producing similar smooth results just like the above method. Such volumetric embedding methods are equivalent to a surface reconstruction process being applied to the hole region. The general idea is to convert the input model into an intermediate volumetric representation from which we reconstruct the surface mesh by certain types of iso-surfacing. Popular examples include [2–4], etc. The strength of such methods is that it can handle any type of mesh data and guarantee the 2-manifold requirement after repair. Nevertheless, it can only deal with 3D closed models, and no mechanism is devised for transplanting geometric details from surrounding regions to hole region.

When the surrounding regions contain rich geometric details, the aforementioned algorithms are unable to make full use of the geometric information from the known region. In principle, the hole region after repair (done by these methods) will be too smooth to have satisfactory visual effects and meaningful quantitative measurements. One way to tackle this problem is context-sensitive model repair and completion. One popular idea is critically inspired by the sample-driven image restoration method, which essentially transplants the mesh patches in the known region to the hole region according to the geometric details of known region. Such approach aims to generate some patches according to the information from the known region, and then fills them into the hole region through model transfer. The patches not only satisfy the boundary condition, but also retain the geometric detail characteristics of the surrounding regions (e.g., [5–8]). As a result, models after repair offer a better sense of reality, especially by users' visual judgement. Parallelling to the above development, another repair method is based on template. The aforementioned methods will break down when the model is missing larger geometric area or having topological defect. To conduct model repair, a complete model, with the same topology or similar geometry shape, must be specified by the user or automatically retrieved from the database as a template (e.g., [9,10]). The key of the template-driven method is how to correspond between defect models and their templates. In general, shape registration and matching is extremely challenging for non-similar models with holes.

Although the above methods are suitable and successful for certain scenarios and can obtain good results with well-behaved datasets, they typically fail when we apply them to hole-filling where significant global structure information is also missing. For instance, to complete the hole in Figure 1(a), our experiment by using [1] is less satisfactory (Figure 1(b)). This is because there is no mechanism to transplant structure information from known region to hole region. Similar problems also arise in image restoration, where it is difficult to detect and recover the missing global structure information during image completion. To solve this problem, Sun et al. [11] first proposed an image restoration method through user interaction to propagate important structure information from known to unknown regions. The general idea is to add a number of user-specified curves cross-cutting the known and unknown region, and these curves will guide the general direction for information transport and recovery.

1.2 Our approach and contribution

Strongly inspired by image restoration research, we propose a user-centered interactive method to transplant global structure information during the hole-filling process, while reconstructing local geometric details (see Figure 2 for the entire algorithmic pipeline).

To streamline the entire algorithm, we assume that the underlying model is a 2-manifold explicitly represented by a triangular mesh and every vertex has consistent normal. For any 2-manifold mesh model, we can traverse its data structure to locate the hole boundary. By analyzing its topology, we are capable

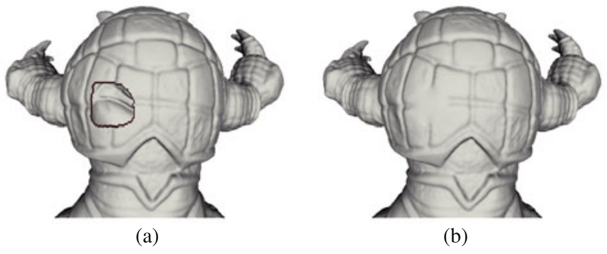

Figure 1 The hole missing the salient structure can not be well repaired using conventional methods. (a) The hole of Armadillo before repair; (b) the hole of Armadillo after repair using Liepa's method.

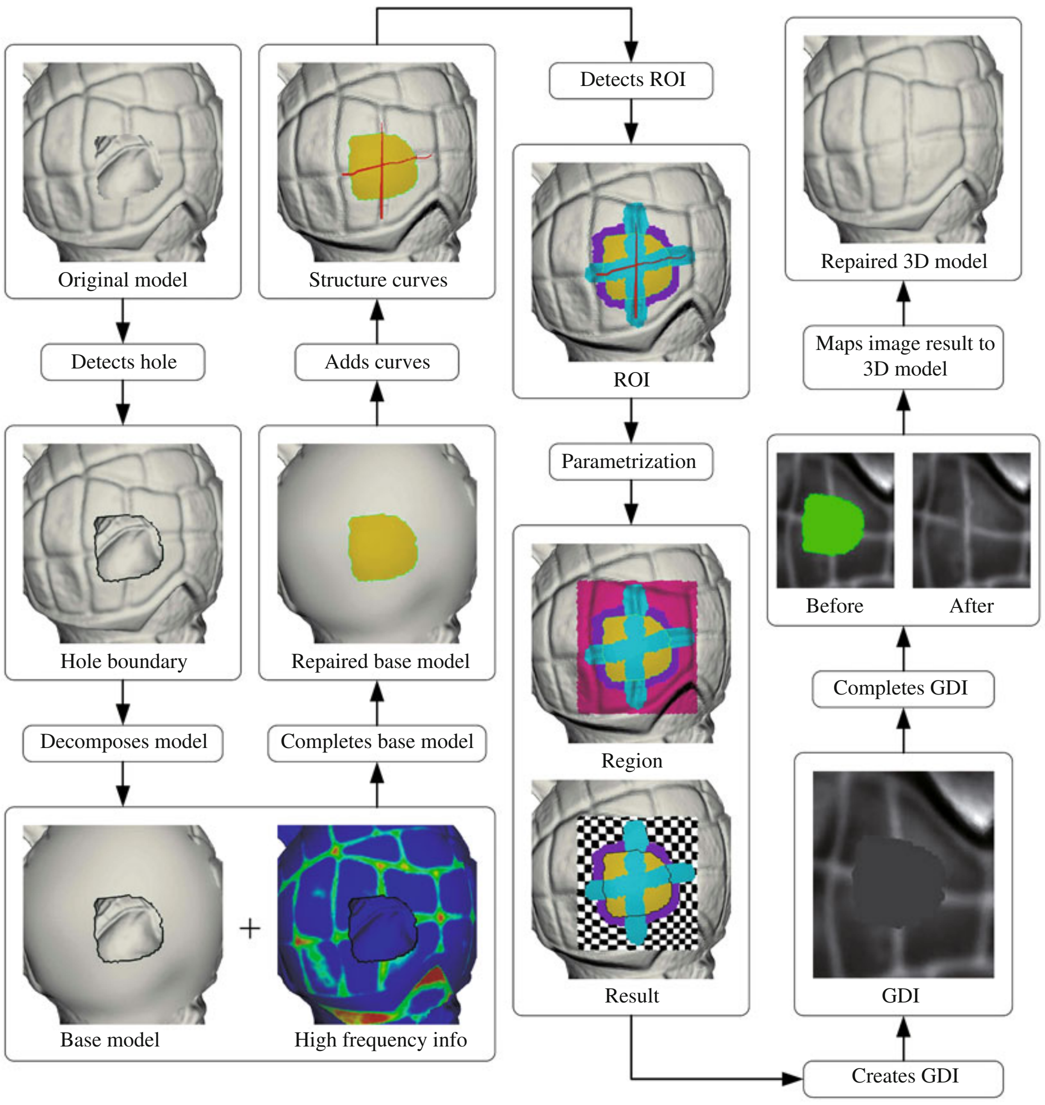

Figure 2 The entire model repair pipeline. The coloring schemes are hole colored in green, filled hole in yellow, user sketches in red, ROI in cyan, boundary region in purple, selected region in pink, local parameterization in black and white square. High frequency picture is also color-coded.

of identifying and enumerating all the hole regions.

Our primary goal is to correctly repair the missing global structure of the hole region. Since the global structure is equivalent to the geometric details at the semantic scale, we must analyze the geometric characteristics of the known region based on its scales. We resort to the popular multi-scale method by decomposing the model into a base shape and a high-frequency component that can explicitly record the geometric details characteristics of each vertex. The base model should have the same topology as the original model. The high-frequency part stores the offset of each vertex of the original model to the corresponding vertex in the base model. After model decomposition, we first repair the holes in the base model with smooth patches, and we then transplant both the geometric details characteristics and the global structure to the smoothed region. In particular, we devise a filtering method to create the base model, which guarantees that each vertex on the original model is only moving to the corresponding vertex along the normal direction.

In order to properly restore the missing global structure of the hole region, we allow users to interactively sketch curves that can span across the known and unknown regions for direction guidance and information transport. This interface design can be directly applied to the complete base model because of its smoothness nature. One key technical issue is to properly identify the region of interest (ROI) that encloses the user-specified curve network, which implicitly corresponds to the semantic-scale global structure. Subsequently, the vertices in the hole region and surrounding regions will be locally parameterized and mapped onto a 2D domain. The parametric domain will be re-sampled to generate a geometry detail image (GDI) at the vertex scale. Because of the model decomposition in the earlier stage, the 3D hole completion problem is transformed into a 2D image restoration problem. We articulate an improved method of image restoration that can restore the geometry detail image under the guidance of the structure curves. After we integrate the curve network and geometric details into the base model, the initial model is correctly repaired towards shape completion.

The salient contribution of this paper is to develop a general-purpose framework to restore the missing geometric details and global structure for model repair, and we achieve these two goals simultaneously. Our framework provides a easy-to-use graphical user interface (GUI) for users to specify the curves which imply the missing global structure intuitively, and then automatically repair the model based on the semantics of surrounding regions. Also, this method can be applied to perform context-based mesh repair and extended to complete the holes with islands conveniently. Our approach has been proved to be intuitive, effective, and accurate in practice through extensive experiments.

1.3 Definition and symbols

To streamline the subsequent technical discussion, we now document mathematical notations and their definitions. The mesh model to be repaired is M, and the geometry of M consists of the position p_i^M of each vertex in $\mathbb{R}^3$, namely $P_M = \{p_1^M, p_2^M, \ldots, p_V^M\}, p_i^M \in \mathbb{R}^3$, where V is the number of vertices. M can be decomposed into base model B and vertex geometric details Δ, which is denoted by $M \to B \otimes \Delta$. The vertex in B is $P_B = \{p_1^B, p_2^B, \ldots, p_V^B\}$, $p_i^B \in \mathbb{R}^3$, and the unit normal vector in B is $N = \{n_1, n_2, \ldots, n_V\}, n_i \in \mathbb{R}^3$. $\Delta = \{\delta_1, \delta_2, \ldots, \delta_V\}$, $\delta_i \in \mathbb{R}$. Then, $\forall \delta_i \in \Delta, \delta_i = (p_i^B - p_i^M) \cdot n_i$ is the offset that v_i moves from M to B along the normal direction. We have $p_i^M = p_i^B - \delta_i \cdot n_i$, namely M can be calculated according to B and Δ.

Although there may be many holes in M, we explain our repair method on one hole Ω_M in M that also corresponds to the hole Ω_B in B. The mesh patch (used to fill in Ω_M) is denoted by M_Ω, the model after repair is denoted by M^0, and $M^0 = M \oplus M_\Omega$. Accordingly, the mesh patch (used to fill in Ω_B) is denoted by B_Ω, the model after repair is denoted by B^0, and $B^0 = B \oplus B_\Omega$. Suppose the set of geometric details in B_Ω is Δ_Ω, and the whole geometric details after repair denoted by Δ^0 and $\Delta^0 = \Delta \oplus \Delta_\Omega$. If suitable Δ_Ω can be located, we can calculate M_Ω according to B_Ω and Δ_Ω, which means $B_\Omega \otimes \Delta_\Omega \to M_\Omega$.

The reminder of this paper will detail our step-by-step procedure for model decomposition, identifying the region of interest (ROI), construction of geometry detail image (GDI), interactive user sketches, and GDI transfer into the hole region.

2 Diffusion-driven base model computation

It is a common strategy to compute base model through mesh fairing. The mesh fairing approach originates from the de-noising method in image processing. For example, Taubin [12] proposed a mesh processing method based on Laplacian flow, which applied the image filter technique to 3D mesh fairing. Laplacian operator, which has both normal and tangent components, is an isotropic filter operator. Therefore, vertex drifting cannot be avoided when it is adopted for mesh fairing. Fleishman [13] and Jones [14] extended the bilateral filtering [15] from image processing to 3D mesh fairing, but these methods did not enforce vertices to move along the normal direction, which also leads to vertex drifting.

In our system, we have to ensure vertex movement along normal direction between original model M and base model B. However, the normal directions of vertices in B are unknown before mesh fairing. Thus we have to first estimate normal direction for each vertex in B, and then compute B by smoothing M.

Let $G_\sigma(x)$ denote Gaussian filter: $G_\sigma(x) = e^{-x^2/2\sigma^2}$. Due to the large differences between vertex normals in the original model and the base model, in order to enforce the proper moving direction of each vertex during iterative diffusion, we shall estimate the unit normal vector set in B. Consider the surface's sample variation, we use Voronoi area A_q of vertex q as weight. We also introduce dot product of two normals as normal difference weight $D_q = n_q \cdot n_p$. To acquire proper estimation of base model's normals, we use the simple robust Gaussian filter method. Similar to the unilateral filter theory, suppose the set of the neighboring vertices of p is $N(p)$, we have

$$\hat{n}_p = \frac{\sum_{q\in N(p)} n_q A_q D_q G_\sigma(\|q-p\|)}{\sum_{q\in N(p)} A_q D_q G_\sigma(\|q-p\|)}. \tag{1}$$

We improve mesh fairing algorithms based on a new first-order predictor, and smooth the mesh with the operator. The vertex displacement of M relative to its corresponding vertex in B is recorded, which gives rise to the decomposition of M into B and Δ. Different from Jones' method, the predictor Π_q of the vertex $q \in N(p)$ which belongs to the neighbor of vertex p is defined as the plane determined by point q and normal of point p, and prediction $\Pi_q(p)$ of p for q is the projection of point p on plane S_q (Figure 3).

For p, its new spatial location is estimated by the weighted average of each predicted vertex in the neighboring region, and $\forall q \in N(p)$, spatial distance is defined as $\|q-p\|$. Therefore, the estimation of vertex p after diffusion becomes

$$\hat{p} = \frac{\sum_{q\in N(p)} \Pi_q(p) A_q D_q G_\sigma(\|q-p\|)}{\sum_{q\in N(p)} A_q D_q G_\sigma(\|q-p\|)}. \tag{2}$$

Our improved algorithm can make the vertex move along the normal direction of the corresponding vertex of base model approximatively. It is not necessary to consider all vertices in the entire model, to accelerate the computation speed, only the vicinity of the hole can influence the effect of mesh completion.

3 User sketches and ROI selection

After the original model is decomposed into base model B and geometric details, we use popular methods such as [1] to fill the hole on the base model, producing the surface B^0. Then users can interactively sketch some curves indicating significant global structure. These curves should span across the known region and the hole region. The L curves added by users are $C = \{c_1, c_2, \ldots, c_L\}$, and two curves $c_i, c_j \in C, i \neq j$ in C are allowed to intersect over B^0.

The ROI is defined as the region that may include apparent structural information covered by all the structural curves $c_i \in C$. ROI can be divided into the ROI of known region and the ROI of hole region. The central idea is to construct ROI of the hole region according to the ROI of known region. As the curve $c_i \in C$ is discretized as a set of discrete points on B^0 which do not contain any real structural information, we must analyze the shape features of known regions covered by these curves and properly define the ROI so that the structural information can propagate along the curves. Similar to [11], once the missing global structural information is transplanted via front propagation along the ROI, the entire hole region is divided into a set of subregions by the boundaries of ROI that represents the global structure.

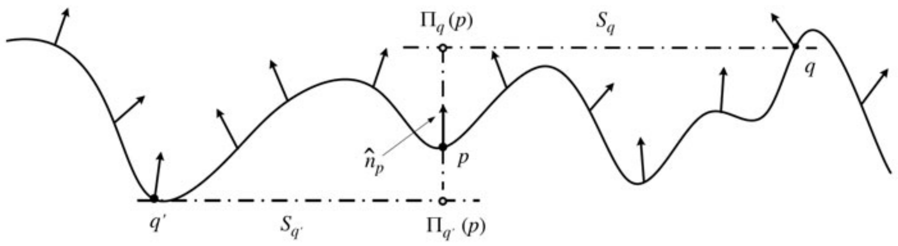

Figure 3 Illustration of the first-order predictor.

These subregions will not contain any important global structural information, so available model repair methods can be employed towards model completion.

The process of detecting ROI is equivalent to model segmentation subject to constraints. According to the curves added by users, the known regions passed by curves should be divided into structural regions and non-structural regions. In principle, the non-structural regions do not participate in the structural propagation process. We propose an automatic ROI segmentation for known regions based on region growing. For the structural curves $C = \{c_1, c_2, \dots, c_L\}$ input by the user, the corresponding ROI should be properly identified, labeled as $R = \{r_1, r_2, \dots, r_L\}$. Without loss of generality, we detail how to find a ROI covered by one structural curve below.

The key for model segmentation is to use differential properties of each vertex such as curvature, as well as the geometric feature description of its neighboring domain, to serve as segmentation criteria. In our system, we first find an initial vertex set V_0 containing the maximum structural features and compute its statistical characteristics. We then gradually enlarge its neighborhood. The statistical characteristics S_V of the vertex set V is defined as $S_V = \sum_{v \in V} \delta_v / \|V\|$ and the difference between the statistical characteristics S_{V_1} and S_{V_2} is $\mathit{diff}(S_{V_1}, S_{V_2}) = (S_{V_1} - S_{V_2})/S_{V_2}$. Suppose V_i is the vertex set with the structural features after iterating i times, if the difference $\mathit{diff}(S_{V_i}, S_{V_0})$ of statistical characteristics between S_{V_i} and S_{V_0} is more than the user-defined threshold, we will stop growing the neighborhood and consider that the ROI is complete. Before detecting ROI, we have already decomposed the original model into the base model B and the geometric details Δ. In our experiments, we set the difference threshold of statistical characteristics to be $0.2 - 0.3$. In addition, we allow the user to adjust the threshold according to the ROI detection result or specify the width of the ROI in known region if the structure information is ambiguous or complex.

Assume that the ROI from known region has the width k (where k means the k-ring neighborhood), we must use the width information to control the region growing of ROI in the hole region. The simplest strategy is to conduct the envelope operation by uniformly growing the structure curve in the hole region in order to have a "fat" curve that consistently defines the entire ROI from the structural point of view. Note that, the ROI r_i corresponds to the structural curve c_i, and ROIs may partially overlap (Figure 2).

4 GDI and model repair algorithm

4.1 Geometry detail image generation

Geometry detail image (GDI) generation is the parameterization and resampling process. First, mesh vertices are mapped onto 2D points by means of parameterization. Then, a gray image is generated by resampling geometry details of the model. The GDI is computed by linearly interpolating the information associated with each triangle. The GDI is an equivalent representation in 2D that encodes the high frequency displacement of the model.

Parameterization methods are critical to GDI creation. Our parameterization method should try to minimize the distortion. In particular, we employ the mean value coordinate parameterization, which can avoid negative energy coefficients and generate quasi-conformal mapping towards the goal of minimizing distortion. In the parametric domain, in order to minimize the area distortion caused by the parameterization process, we allow users to directly specify a near-rectangle region (via projection) which can cover the hole region and its ROI, then the boundary of the selected region is mapped onto the rectangle's four sides in 2D by chord-length parameterization. We compute the weights of internal vertices and solve a large, sparse least squares linear system.

After parameterization, each vertex will remember its 2D coordinates, and the gray-valued height function encodes the high frequency information by linear mapping. We resample the 2D parametric domain using regular grids to compute GDI. We must carefully address the GDI's resolution problem. If the resolution is too low, the characteristics of high curvature in 3D will be lost because of down-sampling. If the resolution is too high, it will cause much higher computational cost in the later stages. In 2D parametric domain, let width and height of the rectangle region be width and height, respectively. The average length of all edges from the model in parametric domain is computed by size, and step= $\alpha\times$ size is the sampling step. The GDI resolution is $m \times n$, where $m = \lfloor\text{width/step}\rfloor$, $n = \lfloor\text{height/step}\rfloor$ and α is a user-specified constant. The gray values of the pixels corresponding to the hole region should be undefined in GDI. α is set to 0.3 in our experiments and it can be adjusted according to the requirements of sampling quality and computation speed.

During the GDI generation process, user-specified structure curves C and their corresponding ROI set R are also mapped onto GDI pixels, which can explicitly remember the structure curves and their ROI in the image domain. After GDI is successfully propagated, the discrete representation can be mapped back to 3D for model completion in the hole region.

4.2 Overview of GDI propagation and completion

The research of image inpainting has gained much popularity (e.g., sample-based shape reconstruction method in [16], gradient-domain method in [17]). Yet, these methods cannot repair the missing global structure. Although Sun et al. [11] proposed a method based on structure propagation to repair the missing global structure, their method uses the concept of Belief Propagation (BP) proposed by [18] in order to obtain a global optimization solution, which is not computationally effective. Due to the significant differences between GDI and regular 2D images, existing image repair methods do not function well for GDI, especially when it comes to tackle the challenging problem of structure propagation. We improve the existing image repair methods (based on [11, 16, 17]) and design an efficient GDI repair algorithm.

We assume that GDI to be repaired is I, the image after repair is I^0, unknown region in I is Ω_I, boundary of Ω_I is $\partial\Omega_I$, and known region in I is I/Ω_I. Each pixel in I contains coordinate (p_x, p_y), gray value $gray(p)$, confidence information $confidence(p)$, gradient information $G(p)$ which includes gradient information $G_x(p)$ along x and gradient information $G_y(p)$ along y, structure curve $curve(p)$, $ROI(p)$ and distance field $distance(p)$. The curves are formulated as $C_I = \{c_1^I, c_2^I, \ldots, c_L^I\}$, where $c_i^I \in C_I$ is a set of pixels of I. The corresponding ROI of C_I in I is $R_I = \{r_1^I, r_2^I, \ldots, r_L^I\}$, where $r_i^I \in R_I$ is also a set of pixels in I. Let pixel patch Ψ be a set of all pixels in a square region which centers around $center(\Psi)$, and the source pixel patch and target pixel patch involved during the process of repair are denoted by Ψ_s and Ψ_t, respectively. First, the global structure of Ω_I is propagated and completed based on C_I and R_I. We do not naively fill the region in which C_I span across in Ω_I with pixels' gray value from the known region I/Ω_I. Instead, we transplant pixels' gradient information in I/Ω_I to Ω_I. The entire computational pipeline is detailed in Algorithm 1.

4.3 Initialization and pre-computation

We initialize all the characteristics for all pixels in GDI at the beginning. The confidence information of pixel p is set 0, if $p \in \Omega_I$ or 1, if $p \in I/\Omega_I$. The gradient information of pixel p within the known regions is initialized as $G_x(p(x,y)) = gray(p(x+1,y)) - gray(p(x,y))$ and $G_y(p(x,y)) = gray(p(x,y+1)) - gray(p(x,y))$. The curves $curve(p)$ are initialized as

$$curve(p) = \begin{cases} \emptyset, & \forall c^I \in C_I, p \notin c^I, \\ S_{\text{curve}} = \{n_1, n_2, \ldots, n_m | n_i \in \mathbb{N}\}, & \forall n \in S_{\text{curve}}, \exists c_n^I \in C_I \text{ s.t. } p \in c_n^I. \end{cases} \quad (3)$$

Algorithm 1: The GDI repair algorithm

```
$\forall p \in I$, initialize the quantities of $confidence(p)$, $curve(p)$, $ROI(p)$, etc.;
$\forall p \in I/\Omega_I$, compute $G(p)$;
$\forall p \in I$, compute $distance(p)$;
$T \leftarrow \emptyset$, $Todo \leftarrow \emptyset$;
//compute the priorities of all pixel patches in $\partial\Omega_I$, and add them into a set of target pixel patches $T$;
for all $\Psi, center(\Psi) \in \partial\Omega_I$ do;
  Compute $priority(\Psi)$;
  $T \leftarrow T + \{\Psi\}$;
// add the pixel patches in a ROI to a set which is to be repaired $Todo$;
for all $\Psi \in T$ do;
  if $ROI(center(\Psi)) \neq \emptyset$ then;
    $Todo \leftarrow Todo + \{\Psi\}$;
    $T \leftarrow T - \{\Psi\}$;
// repair the gradient information of pixel patches in ROI region;
while $Todo \neq \emptyset$ do;
  Find the pixel patch with the highest priority in $Todo$ as $\Psi_t$;
  $p \leftarrow center(\Psi_t)$;
  if $p$ only belongs to one ROI then;
    Find $\Psi_s$ which is most similar to $\Psi_t$ from ROI containing $p$;
    Copy the gradient of $\Psi_s$ to the unknown region of $\Psi_t$;
  else;
    if $p$ belongs to $n$ ROI; Find $\Psi_s^i$ which is most similar to $\Psi_t$ from the $i (i = 1, \ldots, n)$ ROI where $p$ exists;
    Based on $\Psi_s^i, i = 1, \ldots, n$, generate a pixel patch $\Psi_s$;
    Copy the gradient domain of $\Psi_s$ to the unknown region of $\Psi_t$;
  Update the confidence information of unknown pixels in $\Psi_t$;
  Update $Todo$;
// add the remaining pixel patches to the set $Todo$ which is to be repaired;
$Todo \leftarrow T$;
// repair the gradient domain information of left pixel patches
while $Todo \neq \emptyset$ do
  Find the patch $\Psi_t$ with the highest priority in $Todo$;
  Find $\Psi_s$ which is most similar to $\Psi_t$ from the region except ROI;
  Copy the gradient domain of $\Psi_s$ to the unknown region of $\Psi_t$;
  Update the confidence information of unknown pixels in $\Psi_t$;
  Update $Todo$;
Reconstruct $I^0$ based on the gradient domain information.
```

The ROI $ROI(p)$ is initialized as

$$ROI(p) = \begin{cases} \emptyset, & \forall r_i^I \in R_I, p \notin r_i^I, \\ S_{ROI} = \{n_1, n_2, \ldots, n_m | n_i \in \mathbb{N}\}, & \forall n \in S_{ROI}, \exists r_n^I \in R_I \text{ s.t. } p \in r_n^I. \end{cases} \quad (4)$$

Then, we define the distance field of pixel p, $distance(p)$, as

$$distance(p) = \begin{cases} \emptyset, & \forall r_i^I \in R_I, p \notin r_i^I, \\ D = \{(i,d) | i \in S_{curve}, d \in \mathbb{N}\}, & \forall (i,d) \in D, \exists r_i^I \in R_I \text{ s.t. } p \in r_i^I, d = L^1(p, c_i^I), \end{cases} \quad (5)$$

where L^1 is used to denote the first-order Minkowski distance. It is commonly known that the distance field of pixel p can be computed through any breadth-first-search algorithm.

4.4 Priority detection and updating

As illustrated in Algorithm 1, we adopt the sample-pixel-patch-based approach. We repair the pixel patches in unknown ROI by transplanting information from known ROI. During this process, the order of pixel repair is important as it will affect the final result. In particular, we adopt the greedy algorithm, and the target pixel patches having the highest priority will be repaired first. The priority of the target pixel patches is defined as $priority(\Psi_t) = \sum_{p\in\Psi_t} \|G(p)\| confidence(p)$, where $\|G(p)\| = \sqrt{G_x^2(p) + G_y^2(p)}$. The priority of Ψ_t is determined by two factors: the gradient of all pixels in Ψ_t and the proportion of known region in Ψ_t. The gradient $G(q)$ of all unknown pixels $q \in \Psi_t \cap \Omega_I$ in Ψ_t is updated by the gradient in source pixel patches, and the details can be found in section 4.6. The confidence $confidence(q)$ of q is also updated as $confidence(q) = \sum_{p\in\Psi_t\cap I/\Omega_I} confidence(p)/\|\Psi_t\|$, where $\|\Psi_t\|$ is the number of pixels in Ψ_t. Meanwhile, after Ψ_t is repaired, $\partial\Omega_I$ (the edges of Ω_I) will also change, so the new target pixel patches will be generated, along with the updated priority of these pixel patches.

4.5 Similarity determination for pixel patches

According to the priority of pixel patches, we can choose the pixel patch Ψ_t, which has the highest priority among all target pixel patches T. Then we need to seek the pixel patch Ψ_s which is most similar to Ψ_t from the candidate set of source pixel patches S. Similar to the determination criterion in [17], we compare the difference of two pixel patches' value and their gradient to measure their similarity. The distance between pixel patch Ψ_s and Ψ_t is defined as $d(\Psi_s, \Psi_t) = d_{gray}(\Psi_s, \Psi_t) + d_{grad}(\Psi_s, \Psi_t)$, where $d_{gray}(\Psi_s, \Psi_t) = \sum_{p_s\in\Psi_s, p_t\in\Psi_t} \|gray(p_s) - gray(p_t)\|$ and $d_{grad}(\Psi_s, \Psi_t) = \sum_{p_s\in\Psi_s, p_t\in\Psi_t} \|G(p_s) - G(p_t)\|$.

Pixel p_s and p_t are the corresponding pixels in pixel patches Ψ_s and Ψ_t, respectively. For gray-valued distance d_{gray}, only pixels with known values ($p_t \in \Psi_t \cap I/\Omega_I$) are computed. For gradient distance d_{grad}, only pixels with known gradient are computed. In addition, $\|G(p_s) - G(p_t)\|$, the gradient distance between two pixels is defined as: $\sqrt{(G_x(p_s) - G_x(p_t))^2 + (G_y(p_s) - G_y(p_t))^2}$. According to the distance between the pixel patches $d(\Psi_s, \Psi_t)$, we can seek the pixel patch Ψ_s which is most similar to Ψ_t that satisfies: $\Psi_s = \arg\min_{\Psi_i\in S} d(\Psi_i, \Psi_t)$ as the candidate.

4.6 Gradient field propagation and inpainting

As shown in Algorithm 1, we recover the gradient of unknown pixels in Ψ_t based on the pixel patch Ψ_s which is most similar to Ψ_t. As for the pixel patch Ψ_t that belongs to unknown ROI, we simply need to seek the source pixels from Ψ_s that match with the unknown pixels of Ψ_t, and then transplant their gradient directly, as shown in Figure 4. When dealing with intersecting ROI, we believe that the geometric detail characteristics of these regions are related to all the known ROIs. Since we do not use the BP algorithm or its variants and we completely avoid the global optimization process in the interest of interactive performance. We simply construct the proper gradient value of pixels in unknown ROI by the blending method, as shown in Figure 4.

If Ψ_t belongs to several ROIs, i.e., Ψ_t stays in an intersecting area of several ROIs, we define the width of every ROI to be the distance of the largest first-order Minkowski in the ROI. Given that Ψ_t belongs to $r_1^I, \ldots, r_n^I$, the width of corresponding ROI is $w_1^I, \ldots, w_n^I$. According to the similarity measure in two pixel patches, the most similar pixel patches to Ψ_t found from ROIs are $\Psi_s^1, \ldots, \Psi_s^n$, pixel $p_t \in \Psi_t$, distances to all structure curves are $d_t^1, \ldots, d_t^n$ and p_t, most similar pixels are $p_s^1, \ldots, p_s^n$, and the distances from $p_s^1, \ldots, p_s^n$ to corresponding structure curves are $d_s^1, \ldots, d_s^n$. We reconstruct the gradient of unknown pixels in Ψ_t by

$$G(p_t) = \frac{\sum_{i=1}^n w_1^i w_2^i G(p_s^i)}{\sum_{i=1}^n w_1^i w_2^i}, \tag{6}$$

where $w_1^i = e^{-\|d_s^i - d_t^i\|/w_i}$ and $w_2^i = e^{-d(\Psi_s^i, \Psi_t)}$.

Figure 5 shows the gradient repair progression along x and y of GDI for Armadillo model, respectively. Our algorithm first repairs the gradient information of pixels belonging to ROI inside the hole Ω_I of GDI, and then handles the gradient of pixels outside ROI yet still in Ω_I.

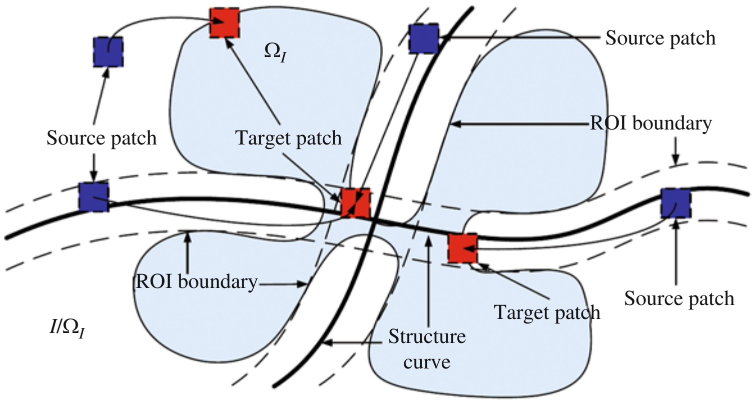

Figure 4 The repairing scheme for intersecting ROI.

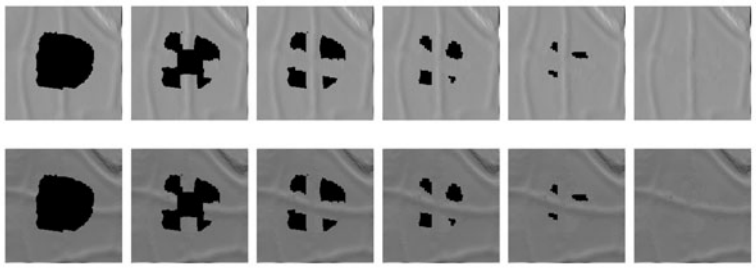

Figure 5 The gradient propagation process along x (above row) and y (below row) for GDI.

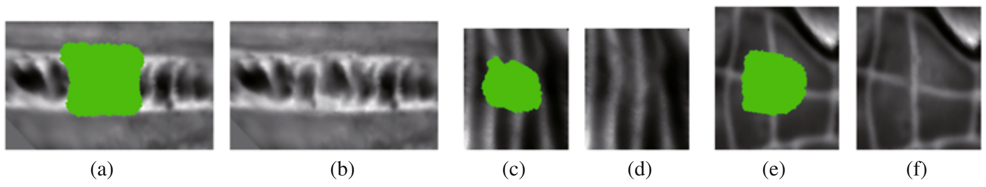

Figure 6 Igea, Dinosaur and Armadillo models' GDI before and after its reconstruction. Hole is colored in green. (a) Igea model's GDI with the hole; (b) Igea model's GDI after reconstruction; (c) Dinosaur model's GDI with the hole; (d) Dinosaur model's GDI after reconstruction; (e) Armadillo model's GDI with the hole; (f) Armadillo model's GDI after reconstruction.

4.7 Gradient-driven image reconstruction

We reconstruct the image by integrating gradient G^0. Essentially, we compute I^0 whose gradient is best approximated by G^0. This is done via minimization of $\| \bigtriangledown I^0 - G^0 \|$. By introducing the Laplacian operator $\triangle$ and divergence operator div, the problem can be conveniently converted to compute the Poisson equation in [19]: $\triangle I^0 = \mathrm{div}(G^0)$.

Because the gray-valued function of $\partial\Omega_I$ in GDI I is known, we use it as Dirichlet boundary condition and compute the above Poisson equation to restore the image I^0. Figure 6 is the images of Igea, Dinosaur, and Armadillo models' GDI before and after their reconstruction.

5 Filling holes with islands

We now discuss how to extend our method to complete the hole with isolated islands. As we mentioned above, the given model is converted into a multi-resolution representation first. Then, the base model is completed through filling smooth patch and the detailed information is encoded in GDI and completed in image domain. Finally, the reconstruction detail is transferred back to the model.

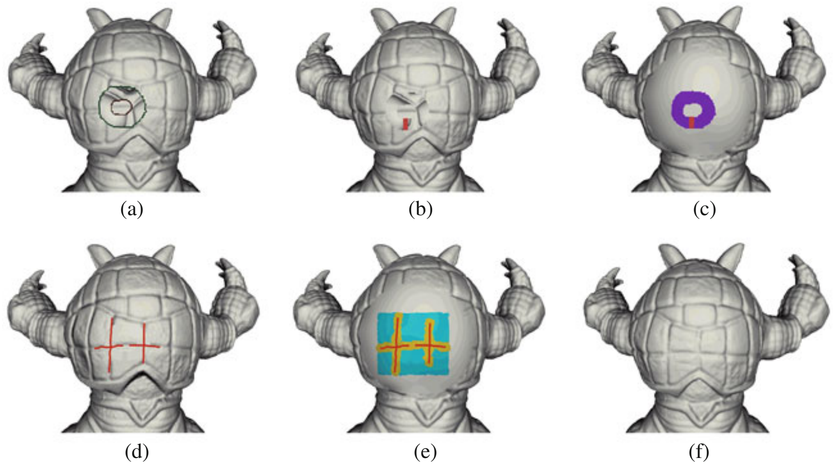
Figure 7 The result of the hole with the island completion in Armadillo. (a) The hole with the island.; (b) the rectangle region added by the user; (c) the result of the base model completion; (d) the structure curves specified by the user; (e) the ROI region (yellow) and the parameterization region (blue); (f) the result of the whole mesh repair.

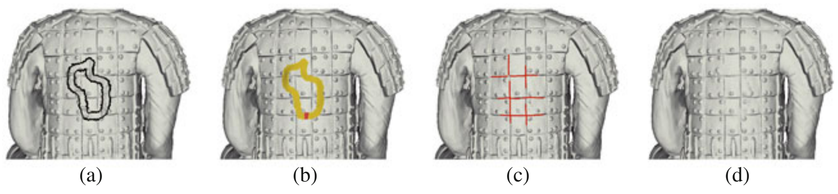
Figure 8 The result of the hole with the island completion on Terra Cotta. (a) The hole with the island; (b) the result of the base model completion; (c) the structure curves specified by the user; (d) the result of the whole mesh repair; (e) the result of the hole with the island completion on Terra Cotta.

Reviewing all steps of our method, the main challenge is that the boundary-based method for hole-filling [1] cannot generate a unique solution for base model consisting of several disconnected components. To address this issue, we present a method to deal with the hole with isolated islands during base model creation: 1) The user specifies some rectangle regions linking the known region with the islands (Figure 7(b)); 2) we triangulate the rectangle regions [1] added by the user and refine the filled mesh [20]; 3) we treat the mesh within the added rectangle as the known region, and then triangulate the remaining region inside the original hole [1] and refine the filled mesh [20]; and 4) we use the second-order weighted umbrella operator to smooth all the filled triangular mesh with the fixed boundaries of the hole and the islands [21](Figure 7(c)).

Once the base model is repaired, a few structure curves are specified by users (Figure 7(d)), and the procedures mentioned in the above sections are employed to complete the rest of repair steps. Figure 7(f) shows the result of Armadillo model repair and Figure 8 shows the key procedures for completing the hole with an island on Terra Cotta.

6 Experimental results

Although the method presented above aims to complete the hole missing global structure information, it can also be generalized to perform a context-based mesh repair without user's input easily. Figure 9 shows the procedure of completing a hole on Bunny without user's interaction. Our goal is to fill the hole on Bunny (Figure 9(a)) with the context geometric details. After mesh decomposition and base model completion (Figure 9(b)), the user may not need sketch any curve over the repaired base model since this hole does not miss any salient global structure information. We simply map the hole region and the surrounding regions onto a 2D domain and generate its GDI (Figure 9(c)), and then restore the missing

Table 1 The detailed performance statistics of our method

	Armadillo	Dinosaur	Igea	Terra Cotta
Vertices	172335	55974	133160	257749
Diffused vertices	9699	9257	11431	12184
Diffusion time (s)	9.7	7.8	8.2	10.4
Base model hole-filling time (s)	1.3	0.2	3.1	0.5
Parameterization time (s)	0.2	0.1	0.4	0.2
GDI size (pixels)	167 × 179	101 × 120	298 × 157	152 × 161
Initialization time (s)	0.1	0.03	0.2	0.1
Gradient map completion (s)	35.2	6.6	41.5	15.4
Image reconstruction from gradient (s)	0.4	0.1	1.0	0.3
Mesh patch generation (s)	0.002	0.001	0.003	0.002

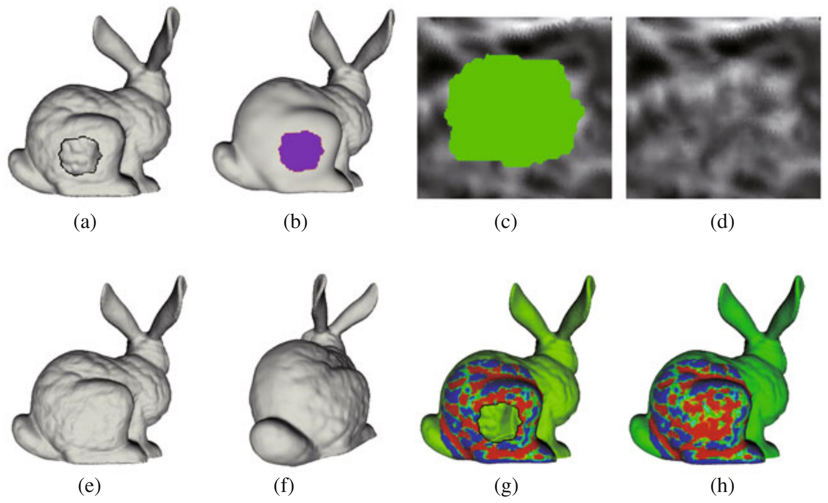

Figure 9 Context-based mesh completion. (a) The hole in Bunny; (b) the result of base model completion; (c) the GDI before repair; (d) the GDI after repair; (e) the result of Bunny model repair; (f) the result of Bunny model repair (side view); (g) the geometry detail before completion; (h) the geometry detail after completion.

high-frequency information in 2D domain (Figure 9(d)). Finally, the reconstruction detail information is transferred back to the model (Figure 9(e) and Figure 9(f)). Figure 9(g) and Figure 9(h) show the geometry detail of Bunny model before and after completion.

Figures 10–13 show the detailed comparison for Armadillo model, Igea model, Dinosaur model, and Terra Cotta model by using Liepa's, Ju's, Kazhdan's, and our current methods, respectively. As clearly demonstrated in these figures, when comparing with the previous model repair algorithms for hole-filling, our method is especially suitable for recovering the global structure during the hole-filling process with much better results. Table 1 details the performance statistics of our method when being applied to different models.

7 Discussion and conclusions

Potentially, there might be two possible time-consuming processes in our hole-filling algorithm: model decomposition and GDI computing. This is because that, we need to pre-estimate the vertex normal on the base model that corresponds to the original model in order to displace the vertices only along the normal direction of the base model, and we also need to devise the improved filtering algorithm for extracting the base model and the geometric characteristics of vertices. All of these extra efforts naturally give rise to more computational cost during model smoothing. Yet, comparing with the voxel-based repair methods ([2–4]), ours is a true local repair method that only focuses on the hole region and its nearby

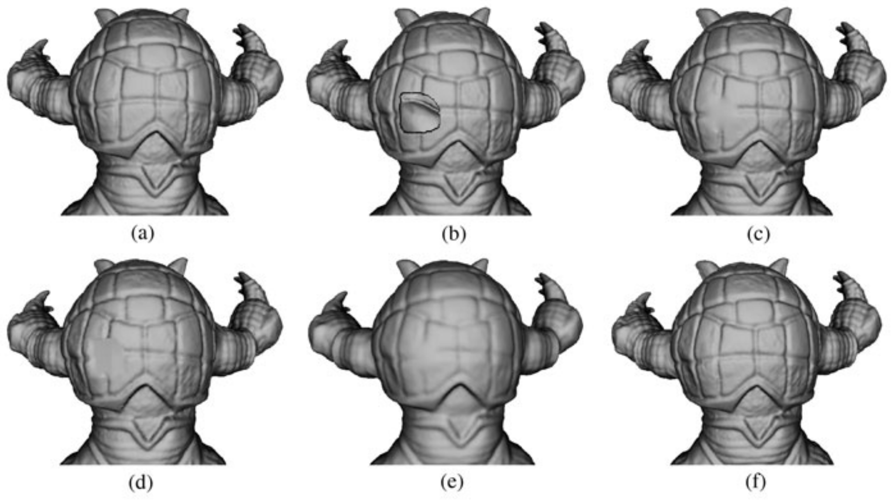

Figure 10 Different results and their comparison for Armadillo repair. (a) Armadillo model; (b) the hole in Armadillo; (c) the result using Liepa's method; (d) the result using Ju's method; (e) the result using Kazhdan's method; (f) the result using our method.

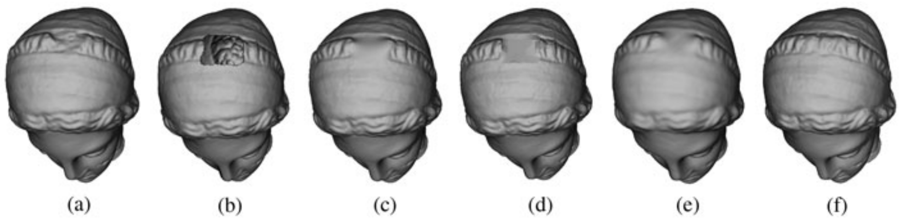

Figure 11 Different results and their comparison for Igea repair. (a) Igea model; (b) the hole in Igea; (c) the result using Liepa's method; (d) the result using Ju's method; (e) the result using Kazhdan's method; (f) the result using our method.

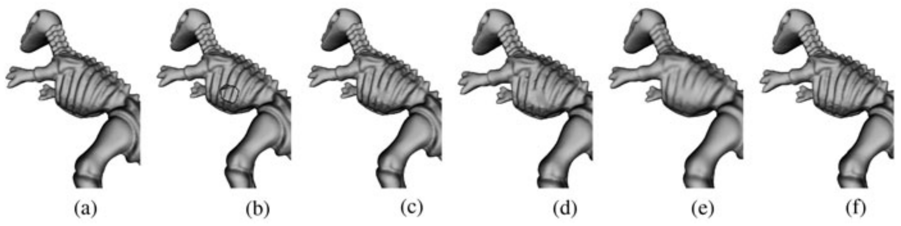

Figure 12 Different results and their comparison for Dinosaur repair. (a) Dinosaur model; (b) the hole in Dinosaur; (c) the result using Liepa's method; (d) the result using Ju's method; (e) the result using Kazhdan's method; (f) the result using our method.

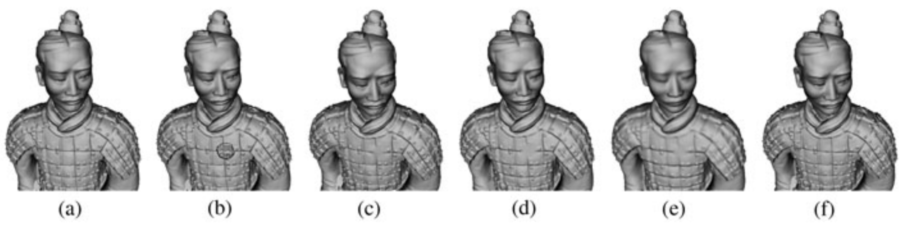

Figure 13 Different results and their comparison for Terra Cotta repair. (a) Terra Cotta model; (b) the hole in Terra Cotta; (c) the result using Liepa's method; (d) the result using Ju's method; (e) the result using Kazhdan's method; (f) the result using our method.

region. As a result, we can expect the improved performance for both model decomposition and GDI computation.

Another issue of concern is that our GDI reconstruction is sample-based. We must seek the most similar source pixel block from the known region for information transfer. In general, traversing the entire model is rather time-consuming. Nonetheless, we have already designed an interactive mechanism that effectively constrains the ROI to be in the vicinity of user-specified curves, which can well match with the global structure. As a result, we only need to conduct local search within or without ROI for transplanting global structure or local geometric details. Therefore, it enhances the accuracy of model completion.

Our method converts the mesh repair in 3D to the image repair in 2D planar domain, which means that the hole to be filled and the region surrounding the hole should be parameterized into a 2D planar domain. In addition, the boundary-based method such as [1] is used to complete the base model, but this procedure cannot repair the sharp feature in the model properly, so it is hard to complete the hole missing the distinctive edges and corners through our method, because sharp features (including edges and corners) do not exist in the vicinity of hole regions and only structure information is designed to propagate along user sketches.

In summary, we have articulated an interactive, intuitive, and efficient model completion method. By conveying the user's knowledge to the model completion procedure, our new algorithm simultaneously reconstructs the global structure and the local geometric details during the hole-filling process. This model completion method is capable of propagating the global structure along the user-defined curve network. From the experimental results, we have observed that, with the help of interactive user sketches, excellent results can be achieved while overcoming typical drawbacks of conventional model completion methods. We plan to explore further improvements and conduct thorough evaluations/comparisons in our future work.

Acknowledgements

This work was in part supported by National Natural Science Foundation of China (Grant Nos. 60533070, 60773153), Key Grant Project of Ministry of Education of China (Grant No. 308004), National High Technology Research and Development Program of China (Grant No. 2009AA012102), Beijing Natural Science Foundation (Grant No. 4102037). Qin Hong's research in this work was supported in part by National Science Foundation (USA) (Grant Nos. IIS-0949467, IIS-0710819, IIS-0830183).

References

1 Liepa P. Filling holes in meshes. In: Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, Aire-la-Ville, Switzerland, 2003. 200–205

2 Davis J, Marschner S R, Garr M, et al. Filling holes in complex surfaces using volumetric diffusion. In: Proceedings of the 1st International Symposium on 3D Data Processing Visualization and Transmission, Padova, Italy, 2002. 428–861

3 Ju T. Robust repair of polygonal models. ACM Trans Graph, 2004, 23: 888–895

4 Kazhdan M, Bolitho M, Hoppe H. Poisson surface reconstruction. In: Proceedings of the 4th Eurographics Symposium on Geometry Processing, Aire-la-Ville, Switzerland, 2006. 61–70

5 Sharf A, Alexa M, Cohen-Or D. Context-based surface completion. ACM Trans Graph, 2004, 23: 878–887

6 Nguyen M X, Yuan X R, Chen B Q. Geometry completion and detail generation by texture synthesis. Vis Comput, 2005, 21: 669–678

7 Park S, Guo X H, Shin H Y, et al. Surface completion for shape and appearance. Vis Comput, 2006, 22: 168–180

8 Xiao C X, Zheng W T, Miao Y W, et al. A unified method for appearance and geometry completion of point set surfaces. Vis Comput, 2007, 23: 433–443

9 Kraevoy V, Sheffer A. Template-based mesh completion. In: Proceedings of the 3rd Eurographics Symposium on Geometry Processing, Aire-la-Ville, Switzerland, 2005. 13

10 Pauly M, Mitra N J, Giesen J, et al. Example-based 3D scan completion. In: Proceedings of the 3rd Eurographics Symposium on Geometry Processing, Aire-la-Ville, Switzerland, 2005. 23

11 Sun J, Yuan L, Jia J Y, et al. Image completion with structure propagation. ACM Trans Graph, 2005, 24: 861–868

12 Taubin G. A signal processing approach to fair surface design. In: Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques, New York, NY, USA, 1995. 351–358

13 Fleishman S, Drori I, Cohen-Or D. Bilateral mesh denoising. ACM Trans Graph, 2003, 22: 950–953

14 Jones T R, Durand F, Desbrun M. Non-iterative, feature-preserving mesh smoothing. In: Proceedings of SIGGRAPH'03, New York, NY, USA, 2003. 943–949

15 Tomasi C, Manduchi R. Bilateral filtering for gray and color images. In: Proceedings of the 6th International Conference on Computer Vision, Washington DC, USA, 1998. 839

16 Criminisi A, Perez P, Toyama K. Object removal by exemplar-based inpainting. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Madison, USA, 2003. 721–728

17 Shen J B, Jin X G, Zhou C, et al. Technical section: Gradient based image completion by solving the poisson equation. Comput Graph, 2007, 31: 119–126

18 Pearl J. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. San Francisco: Morgan Kaufmann Publishers Inc., 1988

19 Perez P, Gangnet M, Blake A. Poisson image editing. ACM Trans Graph, 2003, 22: 313–318

20 Pfeifle R, Seidel H P. Triangular B-splines for blending and filling of polygonal holes. In: Proceedings of the Canadian Information Processing Society Conference on Graphics Interface 96, Toronto, Canada, 1996. 186–193

21 Kobbelt L, Campagna S, Vorsatz J, et al. Interactive multi-resolution modeling on arbitrary meshes. In: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, New York, NY, USA, 1998. 105–114