Shape and Appearance Repair for Incomplete Point Surfaces

Seyoun Park [*] Xiaohu Guo [†] Hayong Shin [*] Hong Qin [†]

KAIST, Daejeon, South Korea [*]
parksy@vmslab.kaist.ac.kr
hyshin@kaist.ac.kr

SUNY at Stony Brook, NY, USA [†]
xguo@cs.sunysb.edu
qin@cs.sunysb.edu

Abstract

This paper presents a new surface content completion framework that can restore both shape and appearance from scanned, incomplete point set inputs. First, the geometric holes can be robustly identified from noisy and defective data sets without the need of any normal or orientation information, using the method of active deformable models. The geometry and texture information of the holes can then be determined either automatically from the models' context, or semi-automatically with minimal users' intervention. The central idea for this repair process is to establish a quantitative similarity measurement among local surface patches based on their local parameterizations and curvature computation. The geometry and texture information of each hole can be completed by warping the candidate region and gluing it to the hole. The displacement for the alignment process is computed by solving a Poisson equation in 2D. Our experiments show that the unified framework, founded upon the techniques of deformable models, local parameterization, and PDE modeling, can provide a robust and elegant solution for content completion of defective, complex point surfaces.

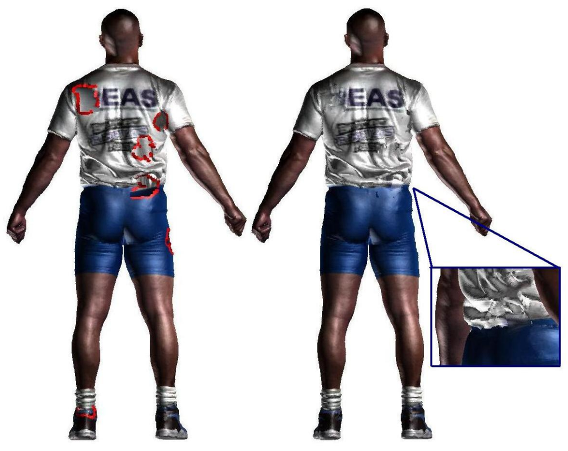

Figure 1. Shape and appearance repair for the Male model: (a) the holes in the original point set can be detected automatically; the points near the hole boundaries are rendered in red color; (b) both the shape and appearance of the hole regions can be repaired automatically based on available context information.

1. Introduction

With the ever-increasing popularity of data acquisition devices, the task of surface content completion is becoming a critical step in the entire reverse engineering pipeline for computer vision. Considering different digitizing techniques that are currently available, many optical devices often produce defective data samples that are subject to local absence of data. This incompleteness of scanned data inputs is mainly due to occlusions, low reflectance, constraints in the scanner placement, etc. In addition, in certain digital information restoration applications (especially in archeology), the scanned objects (e.g., antiques, or art pieces) may be incomplete due to some local missing parts that have been eroded away or ruined over a long historic time span. In either case, both shape and appearance information needs to be repaired in order to facilitate the downstream processing of the digital content.

In this paper, we propose a new surface completion framework that can repair both shape and appearance of defective point sets. Our system takes as input a set of point samples (possibly noisy) that is assumed to be a closed manifold without any normal or orientation information. The point cloud is first pre-processed by a deformable-model-based region partition approach to determine the orientations of the point samples, and infer the in-and-out relationship, all in the same stage [21]. Towards this goal, we use an octree discretization of the original point cloud data in order to build its distance field. The holes in the original point set can be robustly detected using the method of active deformable models by seeking all the saddle points of the unsigned distance field, which will uniquely identify all the missing parts of the scanned data inputs. The automatic hole detection algorithm is more robust than the previous methods that were based on analyzing the point distribution in each octree cell. Then, we extend the key idea of context-based surface completion [16] (when the scanned model contains enough context information) to conduct model repair for both geometry and appearance. Rather than analyzing the shape similarities based on the signed distance fields through volumetric embedding in [16], we propose to perform local parameterizations directly over the surface patches in each octree cell up to a certain layer under the condition that each octree cell contains at most one surface patch that is homeomorphic to a topological disc. After the local parameterizations are carried out and the patches are brought into correspondence, we are able to analyze both shape and appearance similarities based on the curvature and color information of the parameterized patches. It may be noted that it is extremely challenging to perform quantitative comparison of the color information without local parameterizations. Finally, we solve a partial differential equation (PDE) over a 2D domain (i.e., Poisson equation on 2D) to acquire its "warped" shape and the corresponding appearance and apply them to the hole region. In comparison with other volumetric PDE-based approaches [16], our planar PDE method (as a result of local parameterizations) is much more efficient and robust. We conduct the above operations for all the identified holes and finish the model repair process.

2. Related Work

Surface completion of the scanned point sets needs to be naturally integrated into a surface reconstruction algorithm. The original point samples can be interpolated using alpha shapes[7, 2], crusts [1], or balls[3]. [8] used the signed distance field to reconstruct the surface from point clouds. [4] used globally supported radial basis functions (RBFs) to fit data points by solving a large dense linear system. [13] proposed Multi-level Partition of Unity Implicits for constructing a piecewise implicit surface from large-scale point clouds.

Another class of surface reconstruction methods for point clouds is called the active contour method (or Deformable Models) [11, 20, 23, 21], which has been exploited extensively in Computer Vision. Among their many advantages, deformable models are very robust in dealing with noisy data sets. In this paper, we also utilize the active contour method for robustly locating holes.

Surface completion can be also conducted by solving certain partial differential equations. [6] addressed the problem of hole filling via isotropic volumetric diffusion, which can handle geometrically and topologically complicated holes. [19] used implicit surfaces to fill the holes via geometric partial differential equations derived from image inpainting algorithms. [5] repaired the surfaces as an optimization process by minimizing the integral of the square mean curvature.

While all of the above methods create a smooth patch to cover the hole region, the context-based approaches repair the holes according to its context information. [17] warps a given shape model towards the missing region of the given surface using control points, followed by a fairing step along the boundary of the hole. [16] can automatically choose a local patch that is geometrically similar to the hole region. Our approach in this paper is conceptually similar to [16] for automatic context-based hole filling. Nonetheless, different from the volumetric embedding approach taken in [16], our method is solely based on local parameterizations, in which case not only geometry, but also appearance information can be repaired automatically based on context information by only solving a planar PDE system over a 2D domain.

3. Algorithm Overview

The entire processing pipeline of our surface completion framework is outlined as follows:

1. Preprocessing and hole detection: given a set of noisy, defective point samples as input, a deformable-model-based method is utilized to determine the orientation of each point sample and remove noises; in the meantime, the holes in the original point cloud can be automatically detected (Section 4);

2. Local "digital signature" generation: we only perform local parameterizations for each patch bounded by the octree cells (Section 5.1); based on the local parameterizations, a curvature-centered "digital signature" is devised and computed for each local patch (Section 5.2 and 5.3), which can be subsequently employed for both geometry and appearance comparison in a quantitative way;

3. Filling holes: for each hole detected from Step 1 above, we search for the most similar local patch to the hole region by comparing their "digital signatures"; then the best candidate is selected and this local patch needs to be aligned with the hole region using the iterative closest point (ICP) method (Section 6.1), and finally, its geometry and color information can be warped to fit the hole region by solving Poisson equations (Section 6.2).

Figure 2 shows the general pipeline of our surface completion system. To achieve efficiency, three kinds of data

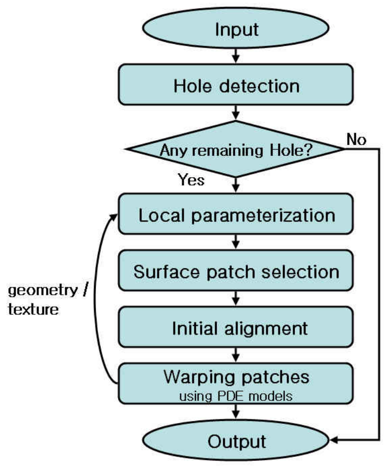

Figure 2. The general framework and data pipeline of our surface completion system.

structures are constructed as a preprocess. First, we embed an input point set in an octree structure. Also, the hierarchical volumetric grid structure is used to construct the distance field in section 4. Finally, we utilize a kd-tree for the fast retrieval of neighboring points in our system.

4. Finding Holes

We utilize the active contour method of [21] as a preprocessing stage for the original point set surface. Given a set of noisy, defective point samples without any normal or orientation information, the active contour method of [21] can be utilized to facilitate determining the orientation of each point sample, removing outliers and noises. We will show in this paper that the active contour method can also be utilized to determine automatic hole detection. In this section, we will briefly introduce the active contour method and explain how to utilize it to find the holes. For more detail on its technical settings and its applications to orientation determination and outliers removal, please refer to [21].

If the input point set, Γ, is assumed to be a closed manifold, we can naturally divide the volumetric space into the inside and outside regions. The active contour method is a commonly used approach to determine the inside or outside of a surface. To avoid leakage through holes, most deformable models resort to the minimization of certain strain energies. In [21], they take a different approach by launching two active contours growing at both sides of the hole. The active contours travel at the same speed and keep the same distance to the surface, and they will finally collide at the center of the hole. For any spatial region visually bounded by a set of surface samples, there exists at least a space point inside the surface which has a local maximal unsigned distance field. Therefore, it is sufficient to launch an active contour seed at each local maximum point besides the one in the outer bounding space. Figure 3 shows the idea of the active contour method. We launch an active contour at each local maximum of the unsigned distance field as well as in the outer bounding space (Figure 3 (c)). The active contours grow and shrink toward the surface and try to keep the same distance to the surface, and at the end the whole surface is sandwiched between these active contours (Figure 3 (d)). Please note that the hole region is sandwiched by an inner and outer contour, and the center of the hole is a saddle point of the unsigned distance field (Figure 3 (e)).

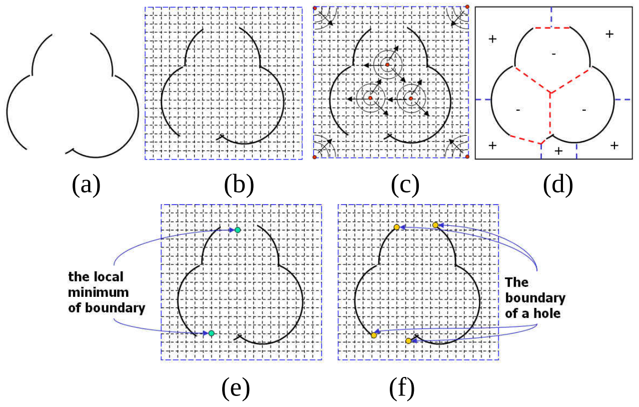

Figure 3. Active contour method for automatic hole detection.

In our implementation, the continuous unsigned distance field is discretized onto a volumetric grid. Each grid point is associated with a distance value, which is defined as its distance to the nearest nonempty cell. Each local maximum grid point initiates an active contour. All points on the active contours are sorted with a heap by their distance to the surface, and the furthest point grows first. This can guarantee that all parts of the active contours grow with nearly the same distance to the surface. When the active contours collide, we can check for the distance value associated with each grid point. If the distance value is greater than a user-specified threshold, this grid point can be considered the center of a hole. The performance of this volumetric approach can be improved by introducing a hierarchical version of this algorithm (see [21] for more detail), which allows us to obtain a $O(n^2 \log n)$ speed, when n is the diameter of the volumetric grid. When a hole Υ is detected, we can identify its boundary $\partial\Upsilon$ by first collecting all the surface points Γ residing in the neighborhood of the boundary grid points. We can check over each point $\mathbf{p} \in \Gamma$, and project the neighboring points of $\mathbf{p}$ onto its tangent plane. If the angle of the open fan area (the region where no neighboring point resides) is larger than certain threshold angle, we consider $\mathbf{p}$ as belonging to the boundary of the hole $\partial\Upsilon$.

5. Patch Comparison based on Local Parameterization

After the holes of the original point surface have been identified, they should be filled in a way that is minimally distinguishable from their surrounding regions. If the original scanned model has enough context information, it should be automatically filled in the way that conforms with the natural properties of the model. This can be achieved by automatically translating, rotating, and possibly warping copies of points from another region which is most similar (both in shape and appearance) to the hole region.

Without global parameterizations, surfaces lack a natural intrinsic spatial structure, which can facilitate searching and selecting similar surface regions for the holes. To tackle this underlying difficulty, we have to resort to a volumetric embedding structure (in this paper we utilize the hierarchical octree structure) to define where and how to search for and select adequate patches.

5.1. Local Parameterization

In order to compare both the shape and appearance of the local surface patches $\Gamma_i \subset \Gamma$, we choose to perform local parameterizations for the surface patches in each octree cell up to a certain layer λ. λ is determined from the highest layer of the existing holes, and it should also satisfy the condition that each octree cell beneath the layer contains at most one surface patch that is homeomorphic to a topological disc; otherwise we do not parameterize the "high-genus" surface patch ($genus \geq 1$) and leave it alone. If the surface patch contains no hole, several existing local parameterization methods can be utilized to parameterize each local patch. In this paper, we utilize the minimum distortion parameterization method in [24].

Let $X : [0,1] \times [0,1] \rightarrow \Gamma_i$ be the mapping from a parameter domain D_i to a surface patch Γ_i. Then the parameterization problem becomes the minimization problem with the following cost function:

$$C(X) = \sum_{j \in M} \{X(\mathbf{s}_j) - \mathbf{p}_j\}^2 + \epsilon \int_{D_i} \gamma(\mathbf{s}) d\mathbf{s}, \qquad (1)$$

where $\gamma(\mathbf{s}) = \int_\theta (\frac{\partial^2}{\partial r^2} X_\mathbf{s}(\theta, r))^2 d\theta$, and $\mathbf{s} = (u, v) \in D_i$, $\mathbf{p} \in \Gamma_i$. $X_\mathbf{s}(\theta, r)$ denotes local polar reparameterization defined as $X_\mathbf{s}(\theta, r) = X(\mathbf{s} + r[\begin{array}{c}\cos(\theta)\\ \sin(\theta)\end{array}])$. The first term in (1) represents squared error of the parameterization given by constraints and the second term estimates the distortion by integrating squared curvature $\gamma(\mathbf{x})$ in parameter domain D. Let the inverse mapping of X be $U : \Gamma_i \rightarrow [0,1] \times [0,1]$. To minimize Equation (1) and obtain U, Zwicker et al. [24] gave a discretization method under the point cloud setting. Due to the space limitations, we do not specify their implementation details here.

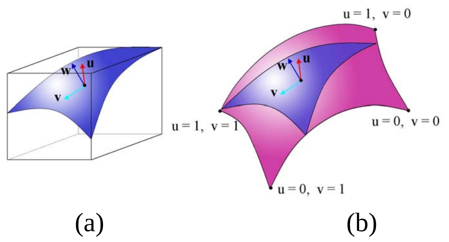

Figure 4. (a) A local surface patch inside the octree cell; (b) automatically selecting local "4-sided" surface patches.

Since the octree cells are aligned with the world coordinate system (instead of object-space), the surface patches inside the octree cells can be of arbitrary shape. For example in Figure 4 (a), the local patch inside the octree cell is "3-sided". This would make our local surface parameterization undesirable if it maps an "n-sided" surface patch onto the $[0,1] \times [0,1]$ parameter domain. This problem can be remedied by selecting local "4-sided" patches in an object-space fashion. We use principal component analysis (PCA) to obtain the principal directions $\mathbf{u}$, $\mathbf{v}$, $\mathbf{w}$ of the local patch, where the origin $\mathbf{o}$ of these local axes resides at its center of mass. Given an user specified patch length L, we can check if a point $\mathbf{p}$ belongs to the "4-sided" local patch by testing if $\overrightarrow{\mathbf{op}} \cdot \mathbf{u} \leq L$ and $\overrightarrow{\mathbf{op}} \cdot \mathbf{v} \leq L$, where $\overrightarrow{\mathbf{op}} = \mathbf{p} - \mathbf{o}$. The checking process can start from $\mathbf{o}$. If it belongs to the "4-sided" local patch, we can continue to check its neighbors in a recursive way. The four farthest corner points can be set as positional constraints in the first term of Equation (1) (see Figure 4 (b)).

If the surface patch contains holes, the distortion part in (1) is hard to estimate around the hole boundary $\partial\Upsilon$. We parameterize Γ_i by fixing parameter values of hole boundary using the reference plane fitted by the points in Γ_i. Let n_h be the number of holes in Γ and the k-th hole Υ_i^k be contained in the surface patch Γ_i. We use principal component analysis (PCA) to obtain the principal directions $\mathbf{u}$, $\mathbf{v}$, $\mathbf{w}$ of Γ_i, where the local reference plane is spanned in $\mathbf{u}$ and $\mathbf{v}$ directions. Having the reference plane as a parameter domain D_i, we can calculate parameter values of points $\mathbf{p} \in \partial\Upsilon_i^k$ by direct projection (see Figure 5). After getting the parameter values for the boundary points in Υ_i^k, we can set them as additional constraints (first term) in Equation (1), and get the parameter values for the other points in Γ_i.

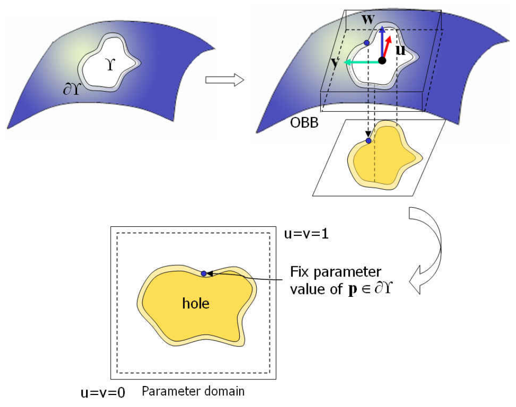

Figure 5. Local parameterizations of patches with a hole.

5.2. Curvature Estimation

After the local parameterization, we can analyze both geometry and appearance similarities based on the curvature and color information of the parameterized patches. Color information at each point for comparing appearance is typically given from the scanned data. The curvature values for comparing geometry need to be estimated at each point. There are several existing methods [14, 18] to compute surface curvature from a triangular mesh. In this paper, we take a similar *Normal Voting* approach of [14] to approximate the surface curvature at each point in the point cloud.

In our discretized point cloud setting, we consider the local neighborhood $\{\mathbf{q}_1, ..., \mathbf{q}_m\}$ around $\mathbf{p}$ to estimate the principal curvature of the surface at point $\mathbf{p}$. For each neighboring point $\mathbf{q}_i$, we estimate the normal curvature at point $\mathbf{p}$ as:

$$\kappa_i = \frac{\Delta\vartheta_i}{\Delta s}, \tag{2}$$

where ϑ_i is the turning angle, and s is arc length. We project the normal of $\mathbf{q}_i$ (denoted as $\mathbf{N}_{\mathbf{q}_i}$) onto the plane Π_i that contains $\mathbf{N}_\mathbf{p}$, $\mathbf{p}$, and $\mathbf{q}_i$, to obtain the projected vector $\tilde{\mathbf{N}}_{\mathbf{q}_i}$. $\Delta\vartheta_i$ is the change in turning angle, and it can be computed as:

$$\cos(\Delta\vartheta_i) = \frac{\mathbf{N}_\mathbf{p} \cdot \tilde{\mathbf{N}}_{\mathbf{q}_i}}{||\tilde{\mathbf{N}}_{\mathbf{q}_i}||}. \tag{3}$$

The change in arc length Δs can be estimated as the geodesic distance from $\mathbf{q}_i$ to $\mathbf{p}$.

After computing the directional normal curvature from each neighboring point $\mathbf{q}_i$, we can get the symmetric matrix:

$$\mathbf{M}_\mathbf{p} = \frac{1}{2\pi}\sum \omega_i \kappa_i \mathbf{T}_i \mathbf{T}_i^t, \tag{4}$$

where the weight ω_i must satisfy the constraint $\sum \omega_i = 2\pi$. $\mathbf{M}_\mathbf{p}$ has eigenvectors that are equivalent to the principal directions $\{\mathbf{T}_1, \mathbf{T}_2\}$, and its eigenvalues $\{m_\mathbf{p}^1, m_\mathbf{p}^2\}$ are related to the principal curvatures $\{\kappa_\mathbf{p}^1, \kappa_\mathbf{p}^2\}$ as:

$$\kappa_\mathbf{p}^1 = 3m_\mathbf{p}^1 - m_\mathbf{p}^2, \quad \kappa_\mathbf{p}^2 = 3m_\mathbf{p}^2 - m_\mathbf{p}^1. \tag{5}$$

5.3. Finding Similar Patches

To find similar patches, we use curvature information for geometry comparison and (R, G, B) color values for appearance. Since the local parameterizations can be performed in arbitrary orientation, we can not directly compare two patches based on the information at the same parameter values. Instead, we select several signatures which represents both the geometry and color information of a given patch. For geometry, we select 6 signatures $(f_1, ..., f_6)$ related to the curvatures, considering the fact that the curvatures are the intrinsic geometric properties of a surface patch. They are defined as:

$$\begin{aligned} f_1 &= \frac{\sum_j \kappa_{1j}}{n_i}, \quad f_2 = \max_j\{\kappa_{1j}\}, \quad f_3 = \min_j\{\kappa_{1j}\}, \\ f_4 &= \frac{\sum_j \kappa_{2j}}{n_i}, \quad f_5 = \max_j\{\kappa_{2j}\}, \quad f_6 = \min_j\{\kappa_{2j}\}, \end{aligned} \tag{6}$$

where κ_{1j} is the maximum curvature of $\mathbf{p}_j \in \Gamma_i$, κ_{2j} is the minimum curvature of $\mathbf{p}_j$, and n_i is the number of points in Γ_i. For texture comparison, we use 9 signature values, $g_1, ..., g_9$, which represent the average, the maximum, and the minimum color value of R, G, B, respectively.

Given a hole patch $\Upsilon_k, (k \in \{0, ..., n_h\})$, we need to compare it with other surface patches $\Gamma_i, (i = 0, ..., n_\lambda)$ using signatures. We define the similarity function $\mathrm{S}(f_j^{\Upsilon_k}, f_j^{\Gamma_i})$ (simply, S_j) as:

$$\mathrm{S}_j = (1 - \frac{d_j}{d_{\max,j}})^r, \tag{7}$$

where $f_j^{\Upsilon_k}$ is the j-th signature value of the patch Υ_k, $d_j = |f_j^{\Upsilon_k} - f_j^{\Gamma_i}|$, $d_{\max,j} = \max_j\{d_j\}, j = 0, ..., n_i$. r represents the sensitivity of S along d_j, and we simply set $r = 2$.

Then, the similarity between two surface patches is defined as a normalized value of the weighted sum, which is obtained by comparing each signature:

$$\mathrm{similarity}(\Upsilon_k, \Gamma_i) = \frac{\sum_j w_j \mathrm{S}_j}{\sum_j w_j}. \tag{8}$$

After calculating the similarity value of each surface patch, we can find the most similar patch Γ_{i^*}, which has the maximum similarity value as a candidate to fill Υ. However, the automatic approach may not always guarantee to give the best results from a human's perspective. So, our algorithm can provide several candidates that have the maximum similarity values, and the users can select one of them as Γ_{i^*}.

Also, the most similar patches for geometry and texture fillings are not necessarily the same, so we can find different Γ_{i^*} for geometry and texture separately.

6. Filling Holes using PDE Models

6.1. Initial Alignment

Before filling a hole with the selected copy region, the initial alignment of these two regions should be performed with respect to a rigid body transformation. Because we have no point in the hole, the transformation from $\partial\Gamma_{i^*}$ to $\partial\Upsilon$ should be used to align the copy region to the hole. Here $\partial\Upsilon$ is the banded region around the hole, and $\partial\Gamma_{i^*}$ is the set of points on Γ_{i^*} having the same parameter values as $\partial\Upsilon$. In our current work, we use iterative closest point (ICP) method which gives a quite reasonable result for the registration between two banded regions. Let $\mathbf{T}$ be the (4×4) matrix which represents rigid body transformation. Let $\mathbf{q}_j \in \partial\Gamma_{i^*}$ be the nearest point of $\mathbf{p}_j \in \partial\Upsilon$, and the distance between two point sets is defined as:

$$d(\partial\Upsilon, \partial\Gamma_{i^*}) = \sum_j ||\mathbf{p}_j - \mathbf{q}_j||^2. \tag{9}$$

The traditional ICP method is to find the matrix $\mathbf{T}$ which satisfies:

$$\min_{\mathbf{T}} d(\partial\Upsilon, \mathbf{T}\partial\Gamma_{i^*}). \tag{10}$$

Since the geometric alignment does not guarantee the alignment between the parameter domains, reparameterization is required after aligning geometric shapes. This can be achieved by adding constraints in Equation (1) with the parameter values of $\mathbf{p'}_j \in \partial\Upsilon$, which are the nearest points of $\mathbf{q}_j \in \partial\Gamma_{i^*}$.

A similar procedure can be applied to the texture alignment by modifying the distance between two points $\mathbf{p}_1 = (x_1, y_1, z_1, r_1, g_1, b_1)$ and $\mathbf{p}_2 = (x_2, y_2, z_2, r_2, g_2, b_2)$ to include additional color information. We use the following distance measure similar to the *Color ICP* method proposed in [9]:

$$d_6(\mathbf{p}_1, \mathbf{p}_2) = \left[\begin{array}{c} (x_1 - x_2)^2 + w_1(r_1 - r_2)^2 + \\ (y_1 - y_2)^2 + w_2(g_1 - g_2)^2 + \\ (z_1 - z_2)^2 + w_3(b_1 - b_2)^2 \end{array} \right]^{\frac{1}{2}}. \tag{11}$$

In our experiments we choose $w_1 = w_2 = w3 = 1/3$.

6.2. Poisson Model for Hole Filling

Partial differential equation (PDE) techniques have been widely used for many visual computing applications. The PDE methods model graphical objects as solutions of certain elliptic PDEs with boundary constraints inside the parametric domain. The PDE model uses only boundary conditions to recover all of the whole interior information and offers high-order continuity as well as energy minimization properties. Since the local surface patches are all parameterized, both the geometry and color information of the surface patch that has highest similarity can be blended with the hole patch by solving planar (2D) PDEs.

In our implementation, we choose the Poisson equation with Dirichlet boundary conditions[15, 22], which is a second order PDE and can be solved more efficiently:

$$\nabla^2 f = \operatorname{div} \mathbf{h} \quad \text{over } \Omega, \quad \text{with } f|_{\partial\Omega} = f^*|_{\partial\Omega}, \tag{12}$$

where $\mathbf{h}$ is the guidance vector field, $\nabla^2 = (\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2})$, and div $\mathbf{h}$ is the divergence of $\mathbf{h}$. It can be verified that the Poisson equation (12) is the solution of the minimization problem:

$$\min_f \int_\Omega ||\nabla f - \mathbf{h}||^2, \quad \text{with } f|_{\partial\Omega} = f^*|_{\partial\Omega}. \tag{13}$$

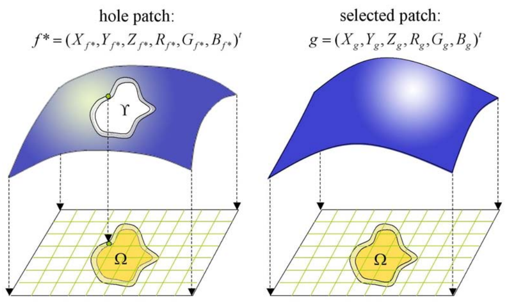
hole patch:
$f^* = (X_{f^*}, Y_{f^*}, Z_{f^*}, R_{f^*}, G_{f^*}, B_{f^*})^t$
selected patch:
$g = (X_g, Y_g, Z_g, R_g, G_g, B_g)^t$
Υ
Ω
Ω

Figure 6. The selected "most" similar patch is warped to the hole patch by solving a Poisson equation, where the guidance function g is simply the geometry and texture information of the selected patch.

If the guidance field $\mathbf{h}$ is the gradient of some guidance function g (i.e. $\mathbf{h} = \nabla g$), we can define a correction function $\tilde{f} = f - g$. In this way the Poisson equation becomes the Laplacian equation with boundary conditions:

$$\nabla^2 \tilde{f} = 0 \text{ over } \Omega, \quad \text{with } \tilde{f}|_{\partial\Omega} = (f^* - g)|_{\partial\Omega}. \tag{14}$$

In our hole filling problem, the guidance function g can be simply the geometry and color information of the selected surface patch (see Figure 6). The parametric domain of both the selected surface patch and the hole patch are discretized into planar grid. The selected surface patch can be blended with the hole region by solving Equation (14) for the XYZ coordinates and RGB color channels respectively (i.e. $f = (X_f, Y_f, Z_f, R_f, G_f, B_f)^t$) using the preconditioned conjugate gradient method.

7. Experimental Results

Our system is implemented on a Microsoft Windows XP PC with Xeon 2.80GHz CPU, 2.00GB of RAM. We tested our system on several incomplete point surfaces which are acquired by manually removing samples from the initial point surfaces, and recorded the statistics of our system performance in Table 1. The Chameleon model in Figure 7 has one hole in the body which was artificially removed. The shape and appearance of the missing part is completed automatically with details taken from the existing parts of the body. Figure 8 shows an example of a scanned Buddha model with one large hole, which is detected and filled automatically. The sharp feature of rocker arm model in Figure 10 can be also repaired gracefully based on its context information. The Iphigenie model and the Male model in Figure 9 and Figure 1 are both non-trivial examples with several complicated missing parts, which are all detected and repaired automatically conforming to their context information. The number of surface patches used for Section 5.3 is dependent on the level of the octree. In the case of Iphigenie model, 47 patches are used averagely for comparison.

Table 1. The time performance of our shape and appearance completion algorithm (in seconds).

model	points	holes	finding	sig.	filling
Buddha	13,942	1	172	6	114
Chameleon	99,835	1	193	17	108
Iphigenie	144,222	4	134	39	431
Male	145,177	6	153	71	653
Rocker arm	39,501	1	103	12	87

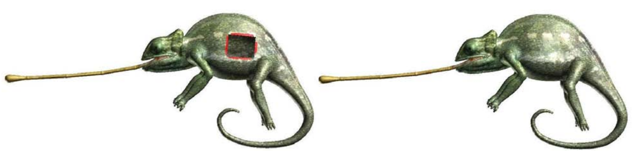

Figure 7. Repairing the Chameleon model; the red points are near the boundary of the hole.

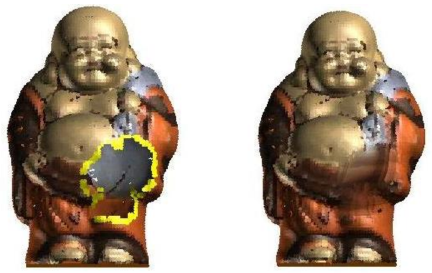

Figure 8. Repairing the Buddha model with one large hole; the yellow points are near the boundary of the hole.

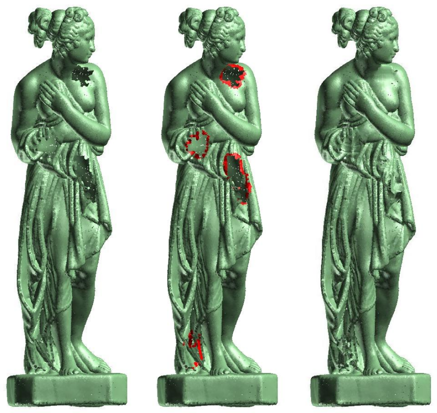

Figure 9. Repairing multiple holes of the Iphigenie model.

8. Conclusion

In this paper we have developed a novel surface content completion framework that can repair both shape and appearance of incomplete point set inputs. The entire model repair pipeline consists of hole detection, patch comparison based on local parameterization, computation of digital signature for patches, identification of the most resembling patch for each hole, and hole filling via the cut-and-paste operation. We utilize the active contour method to facilitate robust hole detection from the noisy and defective data sets. Our surface content completion enables automatic context-based shape and appearance filling simultaneously without any user intervention. We use local parameterizations to align patches in order to extract their curvature-driven digital signature and to solve a PDE on 2D domain for warping the patch to cover the hole region. Our surface completion framework and its constituents are of particular value to computer vision applications such as model reconstruction and shape matching.

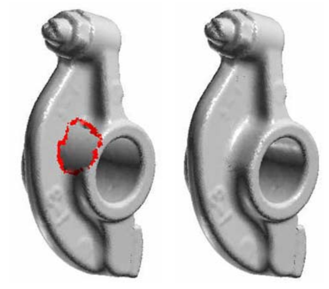

Figure 10. Repairing the sharp feature of the Rocker arm model.

Acknowledgements

This research work was conducted in Stony Brook University when Ms. Seyoun Park was an exchange Ph.D candidate at Computer Science Department and Center for Visual Computing. This work was partially supported by the KRF grant M07-2003-000-20301-0 to S. Park, and by the NSF grant ACI-0328930, the ITR grant IIS-0326388, Alfred P. Sloan Fellowship to H. Qin.

References

[1] N. Amenta, M. Bern, M. Kamvysselis, “A new voronoi-based surface reconstruction algorithm,” *Proc. SIGGRAPH’98*, pp. 415-421, 1998.

[2] C. L. Bajaj, F. Bernardini, G. Xu, “utomatic reconstruction of surfaces and scalar fields from 3d scans.” *Proc. SIGGRAPH’95* pp. 109-118, 1995.

[3] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, G. Taubin, “The ball-pivoting algorithm for surface reconstruction.” *IEEE TVCG*, Vol.4, pp. 349-359, 1999.

[4] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, T. R. Evans, “Reconstruction and representation of 3d objects with radial basis functions,” *Proc. SIGGRAPH’01*, pp. 67-76, 2001.

[5] U. Clarenz, U. Diewald, G. Dziuk, M. Rumpf, R. Rusu, “A finite element method for surface restoration with smooth boundary conditions,” *Computer Aided Geometric Design*, Vol 5, pp. 427-445, 2004.

[6] J. Davis, S. R. Marschner, M. Garr, M. Levoy, “Filling holes in complex surfaces using volumetric diffusion,” *Proc. 1st International Symposium on 3D Data Processing, Visualization, and Transmission*, pp. 428-438, 2002.

[7] H. Edelsbrunner, E. P. Mucke, “Three-dimensional alpha shapes,” *ACM TOG* Vol. 13, 1, pp. 43-72, Jan. 1994.

[8] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle, “Surface reconstruction from unorganized points,” *Proc. SIGGRAPH’92*, pp. 71–78, 1992.

[9] A. E. Johnson, S. B. Kang, “Registration and Integration of Textured 3-D Data,” *Proceedings of the International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, pp. 234-241 , 1997.

[10] T. Ju, ”Robust Repair of Polygonal Models”, *ACM TOG*, Vol.23, 3, pp. 888-895, 2004

[11] M. Kass A. W., D. Terzopoulos, “Snakes: Active contour models,” *International Journal of Computer Vision* Vol. 1, 4, pp. 321–331, 1987.

[12] T. Masuda, “Filling the Signed Distance Field by Fitting Local Quadrics”, *Proc. 3DPVT*, pp. 1003-1010, 2004

[13] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, H. P. Seidel, “Multi-level partition of unity implicits,” *ACM TOG* Vol. 22, 3, pp. 463-470, July 2003.

[14] D. L. Page, A. Koschan, Y. Sun, J. Pail, and M. A. Abidi, “Robust Crease Detection and Curvature Estimation of Piecewise Smooth Surfaces from Triangle Mesh Approximations Using Normal Voting,” *Proc. CVPR*, Vol. 1, pp. 162-167, 2001

[15] P. Perez, M. Gangnet, A. Blake, “Poisson image editing,” *ACM TOG* Vol. 22, 3, pp. 313-318, July 2003.

[16] A. Sharf, M. Alexa, D. Cohen-Or, “Context-based surface completion,” *ACM TOG*, Vol. 23, pp. 878-887, Aug. 2004.

[17] V. Savchenko, N. Kojekine, “An approach to blend surfaces,” *Proc. Computer Graphics International*, 2002.

[18] G. Taubin, “Estimating the tensor of curvature of a surface from a polyhedral approximation,” *Proc. International Conference on Computer Vision*, pp. 902-907, 1995.

[19] J. Verdera, V. Caselles, M. Bertalmo, G. Sapiro, “Inpainting surface holes,” *Proc. IEEE International Conference on Image Processing*, 2003.

[20] R. T. Whitaker, “A level-set approach to 3d reconstruction from range data,” *International Journal of Computer Vision 29*, Vol. 3, pp. 203-231, Sept. 1998.

[21] H. Xie, K. McDonnell, H. Qin, “Surface reconstruction of noisy and defective data sets,” *Proc. Visualization ’04*, pp. 259-266, 2004.

[22] Y. Yu, K. Zhou, D. Xu, X. Shi, H. Bao, B. Guo, H. Y.Shum, “Mesh editing with poisson-based gradient field manipulation,” *ACM TOG* Vol. 23, pp. 644-651, Aug. 2004.

[23] H. K. Zhao, S. Osher, R. Fedkiw, “Fast surface reconstruction using the level set method,” *Proc. the IEEE Workshop on Variational and Level Set Methods*, p. 194, 2001.

[24] M. Zwicker, M. Pauly, O. Knoll, M. Gross, “Pointshop3d: An interactive system for point-based surface editing,” *Proc. SIGGRAPH’02*, pp. 322-329, 2002.