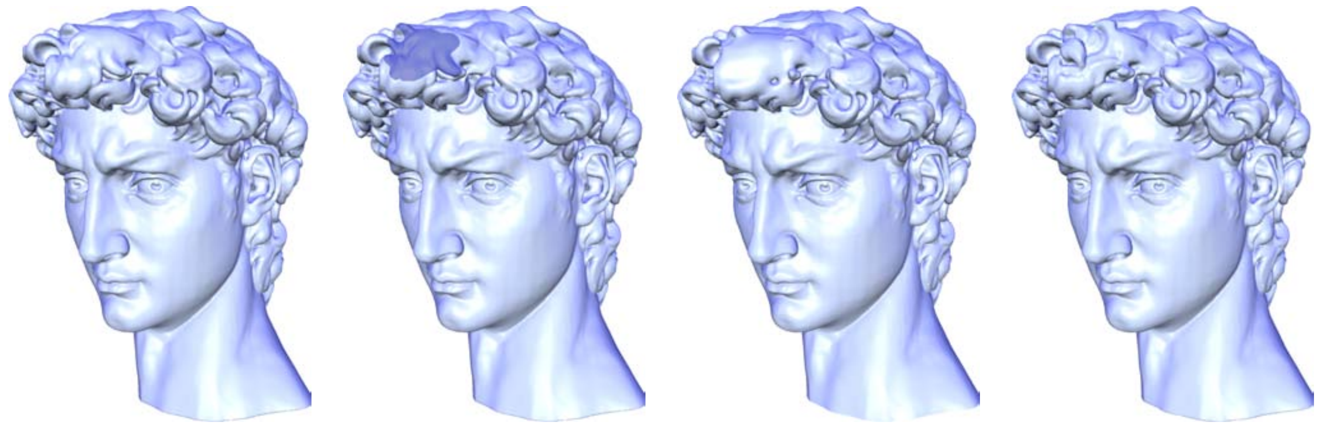
Context-based Surface Completion

Andrei Sharf
Tel Aviv University

Marc Alexa
Darmstadt University of Technology

Daniel Cohen-Or
Tel Aviv University



Original model

Marked area

Smooth Filling

Context-based Filling

Figure 1: Completing a hole in a point-based model. In the darker colored region we removed sample points to demonstrate the surface completion technique. In the middle right the region is filled with a smooth patch conforming with the densely sampled areas, and the result of our context-based surface completion is on the right.

Abstract

Sampling complex, real-world geometry with range scanning devices almost always yields imperfect surface samplings. These “holes” in the surface are commonly filled with a smooth patch that conforms with the boundary. We introduce a context-based method: the characteristics of the given surface are analyzed, and the hole is iteratively filled by copying patches from valid regions of the given surface. In particular, the method needs to determine best matching patches, and then, fit imported patches by aligning them with the surrounding surface. The completion process works top down, where details refine intermediate coarser approximations. To align an imported patch with the existing surface, we apply a rigid transformation followed by an iterative closest point procedure with non-rigid transformations. The surface is essentially treated as a point set, and local implicit approximations aid in measuring the similarity between two point set patches. We demonstrate the method at several point-sampled surfaces, where the holes either result from imperfect sampling during range scanning or manual removal.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—curve, surface, solid and object representations

1 Introduction

Most shapes are nowadays acquired with range scanning devices. The initial representation of the shape consists of several properly transformed depth images, resulting in an irregular point sampling of the surface. As some areas are invisible from some viewpoints, a fair number of scans are necessary to cover the whole surface, often requiring a tedious manual effort to find the right setup. Contact-free sensing, which typically facilitates triangulation for distance estimation, can access only those regions on the surface that are visible (and well reflecting) from a certain set of viewpoints. Physical probing requires enough open space around the point on the surface. Consequently, sampling complex, real-world geometry will almost always be imperfect in the sense that some regions of the visible physical surface are not covered with sample points, i.e. the surface sampling contains holes.

Additionally, surface editing operations can lead to large holes in the surface (i.e. when a part of an object is removed). These holes, as well, have to be filled in a manner that preserves/conforms with the natural properties of the shape (see e.g. Figure 1).

A common way to complete the surface is to fill these holes with a smooth surface patch that meets the boundary conditions of the hole [Curless and Levoy 1996; Davis et al. 2002; Ilic and Fua 2003; Liepa 2003; Verdera et al. 2003]. While this works well for holes that are small compared to the geometric variation in the surface, in general, more complex treatment is required: First, if fine geometric detail is present around the hole, it should not be missing in the completed region. Second, the topology of the missing piece could be more complex than a disk. Thus, we aim at completing missing parts of the surface by transferring appropriately sampled and fitting regions of the shape. We call the process of completing a surface based on the context of the given surface *context-based* surface completion. The figures in this document demonstrate that context-based surface completion progresses over smooth completion in many instances.

Our work is motivated by the success of recent methods for image completion by example [Criminisi et al. 2003; Drori et al. 2003; Jia and Tang 2003; Sun et al. 2003]. These methods build upon non-parametric texture synthesis methods [Efros and Leung 1999; Efros and Freeman 2001; Wei and Levoy 2000; Ying et al. 2001]. The basic idea is that textures, and to some extent missing parts in images, can be synthesized by tessellating regions with proper replications of pieces taken from some specified set of examples. In our work we employ the same framework: we complete missing parts in the surface by integrating patches taken from a given example set. Note that texture synthesis methods make use of the given spatial structure of the data (the regular grid of an image or the connectivity of a mesh). In contrast, the sampling of a two dimensional surface in three dimensions lacks such structure: the sampling is irregular and does not give rise to a parameter domain. This leads to several particular problems for completing a point sampled surface:

1. Defining a patch is difficult because of the missing parameter domain.
2. The boundary of a hole specifies neither the topology nor the geometry of the missing part.
3. The additional degrees of freedom in 3D transforms require aligning and orienting the patch with its surroundings.
4. There are no standard measurements for similarity between two (possibly partial) point sets.

Here we present a method for completing irregularly sampled surfaces. Instead of making a priori assumptions on the characteristics of the missing parts, the existing surface is analyzed, and by exploiting intra-shape similarities and neighborhood information, holes are filled incrementally. The main features of our approach are:

Multi-scale: The identification of the missing surface parts, the surface analysis, the similarity measures, and the filling, are applied in a top-down fashion, where finer details are inferred from coarser scales.

Signatures: We fit algebraic functions locally as a signature of the local surface. The signature is a low-dimensional descriptor used for measuring similarities between point sets.

Alignment: To fit the selected patch to its surrounding, a rigid transformation is applied. To further improve the fit, we apply an iterative closest point (ICP) technique together with a series of small elastic warps.

Another unique feature is that we directly process the point representation including the associated normals, which are computed from range images. However, few steps of the algorithm need a functional representation of parts of the surface. This functional representation should allow approximation of the surface on several scales as well as a parameter-free representation. We locally fit algebraic functions as proposed in [Ohtake et al. 2003]. However, here we compute implicit fits on all levels of the spatial hierarchy.

2 Related Work

Surfaces reconstructed from points acquired by scanners are typically incomplete and contain holes. Thus, surface reconstruction algorithms naturally need to address sampling issues (e.g., [Amenta et al. 1998; Bajaj et al. 1995]). Recently, with reconstruction methods becoming a standard way of geometry creation, several works have been dedicated to the problem of hole filling or repairing the reconstructed surface [Clarenz et al. 2004; Davis et al. 2002; Liepa 2003; Savchenko and Kojekine 2002; Verdera et al. 2003].

Davis et al. [2002] address the case where the holes are geometrically and topologically complex. In this case triangulation algorithms cannot be employed before applying some repairing. Their algorithm first constructs a volumetric signed distance function around the surface samples. Then an iterative Gaussian convolution propagates adjacent distance values to fill the holes.

Verdera et al. [2003] also use an implicit function to represent the surface. They model a PDE for the smooth interpolation of a given hole based on the normal vector field around the hole. In [Clarenz et al. 2004] a surface is repaired by an optimization process. It minimizes the integral of the squared mean curvature (the so-called Willmore energy) to yield a smooth surface.

Liepa [2003] introduces a geometric method for hole filling. Here it is assumed that the reconstructed incomplete surface is already triangulated. The method first detects the holes automatically by identifying the closed loops (i.e. boundary edges of the hole). Then the hole is triangulated [Barequet and Sharir 1995], and the triangulation is refined so that the triangle density agrees with the density of the triangles of the surroundings of the hole. Finally, a fairing step is performed to smooth the area of the filled hole.

All of the above methods assume generic smoothness priors on the missing surface and, thus, create a smooth patch that covers the hole. In contrast, our method learns the characteristics of the given surface and repairs the holes in a context-sensitive manner.

A notable exception is the work of Savchenko and Kojekine [2002], which is closer in spirit to our strategy. Their method warps a given shape model towards the missing region of the given surface using control points. This is followed by a fairing step along the boundary of the hole. Unlike our approach, their method is not automatic and requires some manual intervention. Also, the method is not context-based, does not involve learning, and a prior model must be given in advance.

As mentioned in the introduction, we were inspired by example-based image completion methods [Criminisi et al. 2003; Drori et al. 2003; Jia and Tang 2003], and by the advances in texture synthesis [Efros and Leung 1999; Efros and Freeman 2001; Wei and Levoy 2000; Ying et al. 2001]. Our surface synthesis method has much in common with the work of Hertzmann et al. [2002]. Both methods apply an object-space analysis to infer the characteristic of one part of an object onto some other part. Hertzmann et al. extend their notion of image analogy in [Hertzmann et al. 2001] to an object-space analogy. The behavior of a given curve is applied to other curves using statistical learning strategies. Curve analogies are similar to our approach in that finding the correspondence and proper transformation between arbitrary manifold shapes is challenging – though the restriction to curves somewhat simplifies the issues of topology and parameterization.

3 Data structure and terminology

Given a set of points $\{\mathbf{p}_i\}$ together with normals $\{\mathbf{n}_i\}$ sampled from a manifold surface $\mathcal{S}$, we wish to add points $\mathbf{q_i}$ to the set $\{\mathbf{p}_i\}$ so that the sampling of $\mathcal{S}$ is sufficient everywhere. The main idea is to add points that are rotated, translated, and possible warped copies of points from another region (this resembles the idea of fragment or patch-based techniques for images).

In contrast to techniques for images, surfaces lack a natural underlying spatial structure, which defines where and how to search and select adequate regions (i.e. patches). We explicitly construct such a structure by building a nested spatial hierarchy Ω over the set of points. The largest cell Ω_0 contains all points; a cell $\omega \in \Omega_l$ on level l is subdivided into cells on level $l+1$. In particular, we use boxes as cells and subdivide in a regular fashion at the mid-edges and the midpoint (i.e. $\{\Omega_l, l \in \{0,1,\ldots\}\}$ is an octree).

A cell ω containing a sufficient number of points $||\{\mathbf{p}_i \in \omega\}|| > m$ allows constructing a local surface approximation based on the

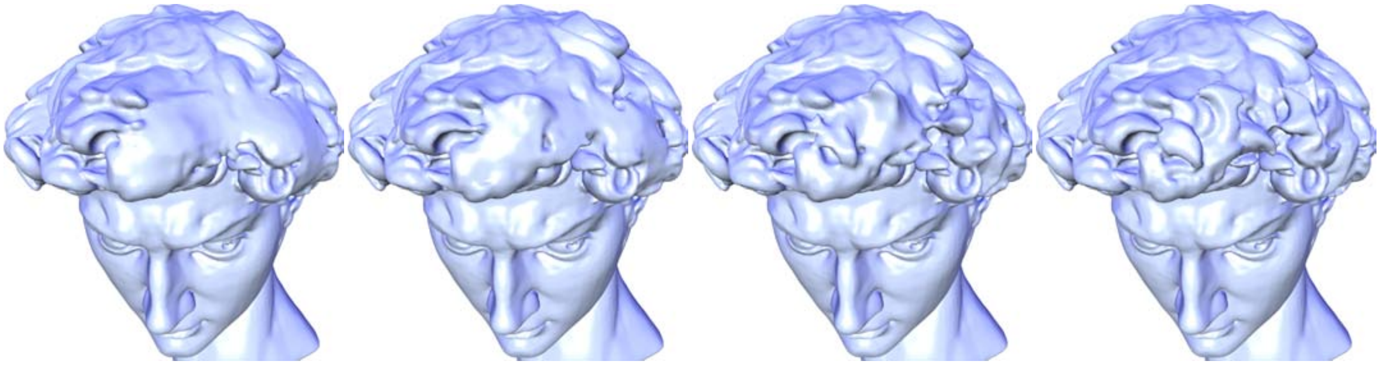


Figure 2: The reconstruction of a hole introduced in David's hair is applied top down, coarse to fine, where a large under-sampled area in the hair is first reconstructed at large coarse scales, and then refined through the levels of the hierarchy.

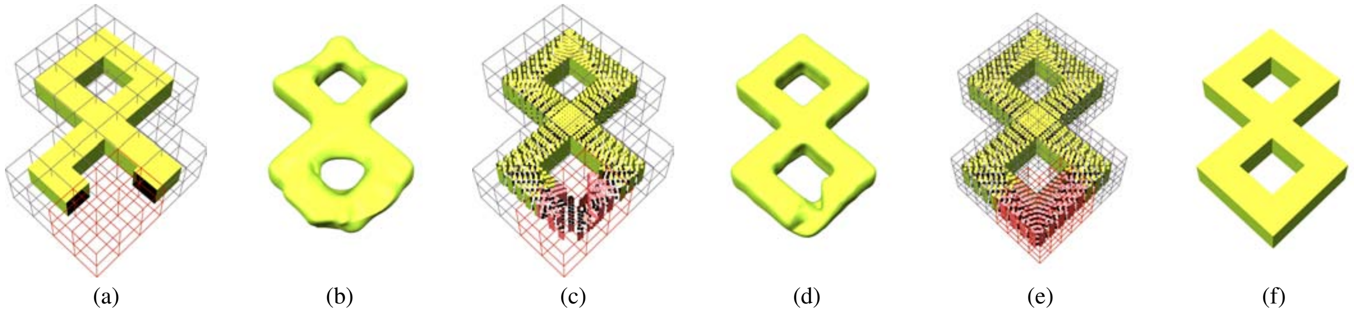


Figure 3: Completion process on block torus. Starting from the initial shape (a), based on some coarse approximation of the missing surface (b) we complete a large part of the model (c). Next, using the updated approximation of the missing surface (d), we complete the hole at a finer level of the hierarchy (e), leading to the final result in (f).

points. We choose to approximate the surface implicitly by fitting a trivariate low-degree polynomial $f_\omega \in \Pi_d : \omega \to \mathbb{R}$. The cell is subdivided if it contains enough points and the max norm error $\max_{\mathbf{p}_i \in \omega} |f(\mathbf{p}_i)|/||\nabla f(\mathbf{p}_i)||$ exceeds a given threshold ϵ. This idea is similar in spirit to the approach of Ohtake et al. [2003], however, without the need of using overlapping cells and blending in the region of overlap.

After this construction, we distinguish between *surface* cells and whether this classification is *valid* or *invalid*. In particular, cells are classified as follows:

Surface cells intersect the current surface approximation as derived from the current point set. A surface cell is:

- **Valid** if it contains at least m points, i.e. there are sufficiently many points for surface representation in the cell.
- **Invalid** if it contains less than m points, i.e. more points should be added to the cell for an appropriate representation of the surface.

Void cells contain no part of the current surface approximation. As every cell with at least m points by construction contains part of the surface approximation, every void cell contains less than m points.

Note that the distinction between void and invalid surface cells is dynamically updated during the process of surface completion. This is necessary since the finally completed surface is not known in advance, and refinements on fine levels in the hierarchy can lead to slight changes in the approximated surface so that the set of intersected cells by the implicit surface might change (see also Figure 5).

Using this terminology we present an overview of the algorithm in the next section and describe the details in subsequent sections.

4 Algorithm overview

The main idea of our approach can be stated as follows: For each invalid surface cell, *import* and *paste* the content of a valid surface cell that matches the surface approximation in and around the empty cell.

However, the process is potentially repeated at finer levels: Only if points have been imported at an appropriate level-of-detail, they are accepted as the final surface representation. Otherwise, the imported points are used merely as an intermediate representation to compute an updated local surface approximation. Then the cell is subdivided and the updated surface approximation is used to fill the cells in the subtree. Note that during the process of updating the surface approximation, the status of void and invalid surface cells can change (see also Figure 5).

The first step is the construction of an initial octree. Starting from the points in the largest cell, the octree is built based on the number of points and the approximation error in each cell. The local implicit approximations are built following the approach of Ohtake et al. [2003]. However, any reasonable way of constructing a local approximation of the shape from the points inside a cell can be used. An implicit representation f of the shape has the advantage that the distance estimation could be done quickly using Taubin's approximation [Taubin 1994], i.e. $d(\mathbf{x}, \{f = 0\}) \approx |f(\mathbf{x})|/||\nabla f(\mathbf{x})||$.

Note that a cell ω is subdivided only if the number of points inside the cell is larger than m and the approximation error $\max_{\mathbf{p}_i \in \omega} |f(\mathbf{p}_i)|/||\nabla f(\mathbf{p}_i)||$ is larger than ϵ. We use the depth of the octree as a measure of shape complexity (i.e. the amount of detail in that region of the shape).

Starting at the coarsest level, a shape signature (described in Section 6) is computed for each cell in the level, based on the shape approximation at the next coarser level in the octree (in pseudocode Fig. 4, Line 3).

Using the process explained in Section 5, a set of invalid surface cells is determined (Line 4). For each invalid surface cell we compute its adjacent valid surface cells at the same level in the hierarchy (note that we further subdivide octcells if necessary). This ensures that cells are surrounded by adjacent cells with a shape approximation at the same level in the octree. The set is ordered according to the number of valid adjacent cells.

The first cell in the ordered set of invalid surface cells is processed (Line 6). Based on the signature, a best matching valid surface cell is identified (Line 7). See Section 6 for details.

The contents of the best matching valid cell are pasted into the invalid cell (Line 8). This process requires finding an appropriate rigid transform. To further align the imported points with the surrounding shape we add a non-rigid transformation. Details of this step are described in Section 7. The imported points are then used to re-compute the local shape approximations in the cell (Line 12).

After filling the invalid cell with the points of the best matching valid cell, the fit is evaluated. The imported points are finally accepted and the status of the cell is changed to valid only if the adjacent valid cells have no descendants (i.e. the completion process has reached the level-of-detail of the surrounding cells). If the imported points lack level-of-detail as compared to the surrounding shape, they are discarded and only the updated local shape approximation remains for further use in the filling process (Line 13). Note that this exploits the correlation between cell size and local shape complexity.

This process is repeated until all cells are valid (or the maximum depth in the octree is reached). The top-down, coarse to fine reconstruction of the hole is demonstrated in Figures 2 and 3, where a large area of missing sample points is first reconstructed at large coarse scales, and then refined through the levels of the hierarchy.

```
COMPLETE-SURFACE()
 1  while $Stop \neq 1$ do
 2      $P' = P$
 3      $\forall\, \omega \in \Omega_l$ COMPUTE-SIGNATURE($\omega, S_l$)
 4      $\Omega'_l$ = FIND-INVALID-CELLS($\Omega_l, S_l$)
 5      while NotEmpty($\Omega'_l$) do
 6          Find $\omega'_j \in \Omega'_l$ s.t. $\omega'_j = argmax_j \|$ValidNeighbors$(\omega'_j)\|$
 7          Find $\omega_k \in \Omega_l \setminus \Omega'_l$ s.t. $\omega_k = argmin_k($Distance$(\omega'_j, \omega_k))$
 8          Fill $\omega'_j$ with points $p_i$ where $p_i \in \omega_k \bigcap P$
 9          $P' = P' \bigcup p_i$
10      endwhile
11      $l = l + 1$
12      Compute $S_l$ based on $P'$ using finer approximation
13      if ($l > max\_level$ or $S_l$ is fine enough around hole) then
14          $Stop = 1$
15  endwhile
```

Figure 4: Surface completion pseudocode.

5 Identifying invalid surface cells

The identification of a hole in a point-sampled surface, like other topological features, is typically an ill-defined problem. Here we use a simple heuristic that we found quite effective in practice. Nevertheless, since the method might not be sufficient in complex cases, the user can manually define invalid cells. The basic idea is to use the local shape approximation and intersect it with all cells. If a cell is intersected by the current shape approximation but contains less than m points, more points should be added to form an accurate representation of the surface in that cell.

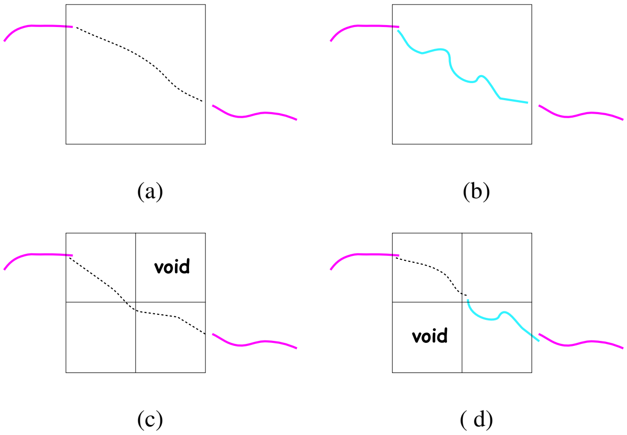


Figure 5: Illustrating the identification of a hole and the top-down incremental filling: (a) The input set of points in red defines an implicit surface (dashed curve) that intersects the large cell. (b) A set of points (in blue) is imported. (c) The cell is subdivided and for each sub-cell an implicit surface is defined based on the imported points. Note that the upper right subcell is considered void. (d) A set of points is imported and placed in the lower right sub-cell. This is then followed by a re-evaluation of the implicit surfaces at its siblings. Note that the lower left subcell is considered now void.

This approach limits the size of a hole: A hole in the surface sampling needs to be small enough so that it can be covered by a shape approximation at the coarser level of the octree. Conversely, because holes are only identified if they lead to cells with few points, holes need to be at least as large as the cells at the maximum depth of the octree.

Note that it is not enough to consider only cells with less than m points as the cell subdivision terminates when the surface approximation error is bounded. For example, a cell might well contain many points and a local shape approximation with small error, however, some part of this shape approximation would not be covered with points. One could argue that this “hole” is accurately covered by the local shape approximation and there is no need to fill it. We prefer to fill such holes so that the resulting point set represents a sufficiently dense sampling of the surface everywhere.

Thus, it is necessary to look ahead in the subtree of valid surface cells. Effectively, the candidate set for invalid surface cells consists of all cells containing less than m points in the full octree resulting from subdivision of the tree to maximum depth.

Given a cell $\omega \in \Omega_l$ containing less than m points, we need to decide whether the cell is invalid (i.e. intersects the current surface approximation) or void. A set Ξ_ω of cells carrying a local shape approximation is constructed. This set consists of:

1. The parent cell of ω. Note that this cell must contain a local shape approximation as it is either filled with original points or a shape approximation has been computed from an intermediate step of the process.

2. All adjacent cells on the level of ω carrying a local shape approximation.

The elements in $\hat{\omega} \in \Xi_\omega$ are inspected in turn. Let $f_{\hat{\omega}}$ be the local shape approximation of $\hat{\omega}$. If the zero set of $f_{\hat{\omega}}$ intersects ω we define it to be invalid (and it has to be filled); otherwise it is considered as valid and void.

To check efficiently for the intersection of $\{f_{\hat{\omega}} = 0\}$ and ω, the following procedure is used. It assumes that quadratic polynomials are used for a local implicit shape approximation

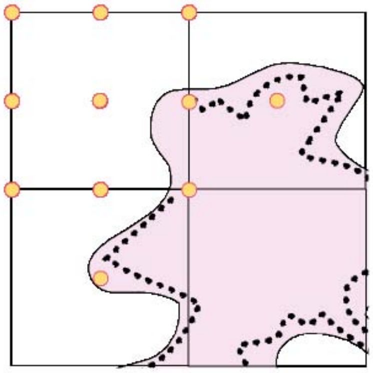


Figure 6: The signature of the invalid cell (upper left corner) contains the distances and directions computed on the colored set of points to the surface together with the tree depth at each valid adjacent cell.

(other local shape approximation techniques might require other approaches to check for intersection). Let the box be defined by $(x_1, y_1, z_1), (x_2, y_2, z_2)$.

1. The set $\{\mathbf{c}_i\}$ of points is initialized to the corners of ω, i.e. the set $\{(x_i, y_j, z_k), \quad i, j, k \in \{1, 2\}\}$.

2. The values $f_{\hat\omega}(\mathbf{c}_i)$ are computed. If there exists at least one positive and at least one negative value, ω intersects the shape approximation and the procedure is terminated (i.e. this is a trivial accept).

3. The derivatives $\nabla f_{\hat\omega}(\mathbf{c}_i)$ are computed. We call ω *x-monotone* if the partial derivatives w.r.t. the x-direction have the same sign. If ω is x, y, and z-monotone there is no intersection (i.e. this is a trivial reject).

4. Assume $f_{\hat\omega}(\mathbf{c}_i)$ are all positive. The minimum of $f_{\hat\omega}$ is bounded from below inside ω by
$$\min_{i,d\in\{x,y,z\}} c_{i_d} - \left|\frac{d_1 - d_2}{2}\right| f_d(\mathbf{c}_i)$$
If the bound is positive, the minimum inside ω is positive, so $f_{\hat\omega}$ is positive inside ω.

5. The box is subdivided regularly (i.e. eight sets of corner points $\{\mathbf{c}_i\}$ are generated) and the tests are repeated starting from step 2.

Obviously, this definition of invalid surface cells depends on the current set of local shape approximations. Once points are (temporarily) filled and the local shape approximations are updated, the void / invalid status has to be redefined.

6 Finding the best matching valid cell

Given a surface cell ω that has been identified as invalid, we want to find a valid cell ω' in a set Υ of cells with the same size as ω so that ω and ω' contain similar pieces of surface. This requires a shape distance measure for cells and the generation of a suitable set Υ, which are discussed in the following two sections. Then, finding the best match is performed by iterating over all cells in the candidate set and finding the cell whose distance to the invalid cell is minimal.

6.1 The signature of a cell

Rather than computing the similarity measure directly, a vector-valued signature is computed for each cell. The similarity of two cells is then given as the weighted Euclidean distance of the shape signatures.

The shape signature needs to be computed for valid as well as invalid surface cells. For practical reasons, the computation of the signature should be efficient. Also, we want to capture the shape at the appropriate scale (as given by the depth in the octree) and the amount of detail that is present at finer scales.

These considerations have led to the following main idea: The signature of ω comprises by two elements: First, the shape at the scale of the octree level is computed using the surface approximation in the cell at the next coarser level of the octree. This shape approximation is available for valid as well as invalid surface cells. Second, the amount of detail in ω is estimated as the depth of the subtrees in adjacent valid cells.

The local shape approximation is evaluated at a set of symmetrically placed points. The symmetry allows us to compare cells transformed according to the symmetry group. Let the shape of ω be defined by $(x_1, y_1, z_1), (x_2, y_2, z_2)$. We construct the following sets of points $\{\mathbf{c}_i\}$ (see also Figure 6):

1. A set of points $\{\mathbf{c}_i\}$ inside the cell; specifically, the center, the corners, the mid-edges, and the face-centers of ω, i.e. $((x_{b_1}+x_{b_2})/2, (y_{b_3}+y_{b_4})/2, (z_{b_5}+z_{b_6})/2), (b_1, \dots, b_6) \in \{1, 2\}^6, b_{2k-1} \le b_{2k}, k \in \{1, 2, 3\}$.

2. A set of points $\{\mathbf{a}_i\}$ in adjacent cells; specifically, the location of centers of cells adjacent to ω: $(x_1, y_1, z_1)/2 + (x_2, y_2, z_2)/2 + (b_1(x_1 - x_2), b_2(y_1 - y_2), b_3(z_1 - z_2)), (b_1, b_2, b_3) \in \{-1, 0, 1\}^3$.

We have found that the last set of points is particularly important as it takes into consideration the neighborhood of ω.

The amount of level-of-detail $\mathbf{l}_\omega$ in ω is a vector, which for each valid cell adjacent to ω, contains the depth of the subtree in this cell.

The signature of ω consists of five components:

1. The vector of signed distances in the points $\mathbf{c}_i$ evaluated using the shape approximation of the parent cell of ω:
$$\mathbf{s}^c_\omega = \left(\frac{|f_{\hat\omega}(\mathbf{c}_1)|}{||\nabla f_{\hat\omega}(\mathbf{c}_1)||}, \dots\right).$$

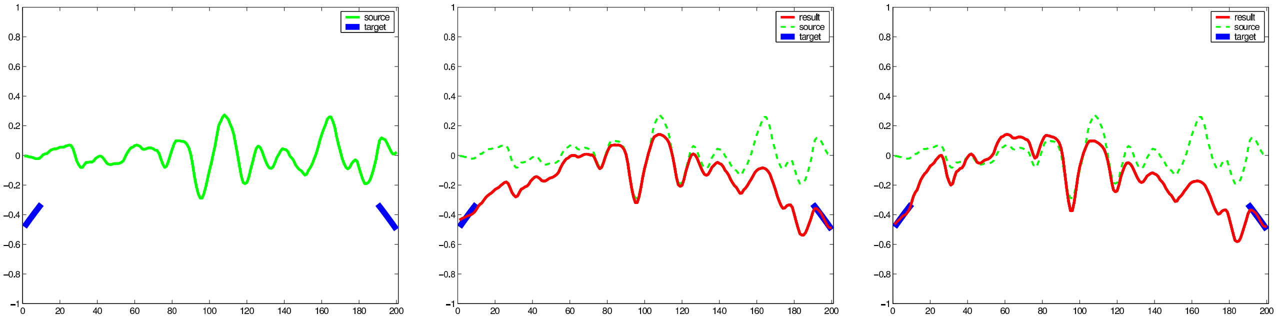
2. The vector $\mathbf{s}^a_\omega$ of signed distances in the points $\mathbf{a}_i$, defined as above, however, evaluated using the shape approximation of the respective adjacent cells.

3. The vector of directions from the points $\mathbf{c}_i$ to the closest point on the surface approximation in the parent cell, approximated here by $\mathbf{n}^c_\omega = (\nabla f_{\hat\omega}(\mathbf{c}_1), \dots)$.

4. The respective vector $\mathbf{n}^a_\omega$ of directions from the points $\mathbf{a}_i$ to the closest point of the surface approximation in the respective adjacent cell.

5. The vector of octree depth $\mathbf{l}_\omega$ (note that we use different types of implicit surface approximation as suggested by Ohtaka et al. [2003], however, we have found that taking this difference into consideration does not improve the signature).

Thus, the signature represents (mostly) a discrete sampling in ω of an approximation to the distance field of the shape approximation in $\hat\omega$. Because $f_{\hat\omega}$ is a low-degree polynomial approximation, the signature refers to a smooth version of the shape. Smoothing depends on the size of the cell, or, equivalently, the level in the hierarchy. This way, signatures are rather scale-independent. However, they reflect the amount of detail in the shape approximation through $\mathbf{l}_\omega$.



(a) (b) (c)

Figure 7: A simple 2D example of iterative non-rigid transformation implemented in MatLab for demonstration purposes. (a) The input setting; the green surface needs to be aligned with the two pieces in blue. (b) The warped surface in red after one iteration. (c) After a second iteration, the warped surface of (b) is warped again towards the blue pieces.

6.2 Weighted distances between signatures

The distance computation takes the different components of the signature into account. We distinguish between the distances, gradients, and octree levels, as well as between the points sampled in the cell or in adjacent cells. In particular, the distance $d(\omega, \omega')$ is computed as:

$$d(\omega, \omega') = w_i d_c(\omega, \omega') + (1 - w_i) d_a(\omega, \omega') + w_l d_l(\omega, \omega'), \quad (1)$$

where d_c is the distance inside the cell

$$d_c(\omega, \omega') = w_d \|\mathbf{s}^c_\omega - \mathbf{s}^c_{\omega'}\| + (1 - w_d)\|\mathbf{n}^c_\omega - \mathbf{n}^c_{\omega'}\|, \quad (2)$$

and d_a is the distance computed in adjacent cells

$$d_a(\omega, \omega') = w_d \|\mathbf{s}^a_\omega - \mathbf{s}^a_{\omega'}\| + (1 - w_d)\|\mathbf{n}^a_\omega - \mathbf{n}^a_{\omega'}\|. \quad (3)$$

The distance between octree levels d_l is essentially computed as the Euclidean distance, however, only for those vector elements that are defined in both, $\mathbf{l}_\omega$ and $\mathbf{l}'_\omega$.

The weight w_i represents a trade-off between shape samples inside the cells and samples in adjacent cells, while w_s represents the trade-off between distances and (approximated) gradients.

The weighting between the shape match inside the cell and in adjacent cells depends on the progress of the completion. As the completion process advances, the local surface approximation improves. At the same time, more detailed solutions require a better local fit. Thus, we increase w_i with the decreasing distance to the maximum depth in the hierarchy.

6.3 The candidate set Υ

The candidate set Υ for a cell $\omega \in \Omega_l$ consists of valid cells with the same size as ω. It contains all valid cells from the same level in the octree, i.e. $\Omega_l \subset \Upsilon$. However, a larger set of candidates leads to better results. We enlarge the set by also adding all elements of the symmetry group of the cell (i.e. rotations of the cell by π around the canonical axes and reflections).

Nevertheless, for complex shapes that might still not generate enough candidates, more candidates are generated by rotating the point set by angles of $\pi/2, \pi/3, \ldots$ and then constructing an octree over the rotated point set. This assures that the cells are aligned along the same axes.

For some applications where a series of similar objects are scanned, one can build a large training set with a rich set of examples. A class-based training set typically improves the performance of an unsupervised learning process (such as we employ here). However, in this work, we restricted the search to a single shape, and we leave the analysis of the potential benefits of using a large class of shapes for defining Υ for future work.

7 Transferring the sampled surface

Assume that we have determined a matching cell ω' for an invalid surface cell ω. The basic idea is to simply paste the points contained in ω' into ω (and then use that input for computing a new shape approximation if the fit has to be repeated on a finer level). Note that the shape distance measure is an average over several sampled values. Even the best fit could be non-conforming with large parts of the surrounding points. Thus, the inserted points are subjected to a rigid and an additional non-rigid transformation to conform as much as possible with a subset of the surrounding points.

7.1 Iterative rigid and non-rigid transformations

The points $\mathcal{F}_\omega$ to be inserted into ω are transformed to conform with the points in adjacent cells. The procedure is inspired by iterative closest point (ICP) procedures [Besl and McKay 1992; Chen and Medioni 1992; Zhang 1992] but adds a non-rigid term.

The set of neighbor points $\mathcal{N}_\omega$ is determined by collecting points in a ball around ω. The radius of the ball is incremented until at least $m/2$ neighbor points are collected. For each of the points $\mathbf{p}_i \in \mathcal{N}_\omega$ the closest point in $\mathbf{p}_{i(\mathcal{N}_\omega)} \in \mathcal{F}_\omega$ is computed. Based on this correspondence, a fitness-of-fit function can be defined as the sum of Euclidean distances between corresponding points:

$$d(\mathcal{N}_\omega, \mathcal{F}_\omega) = \sum_{\mathbf{p}_i \in \mathcal{N}_\omega} ||\mathbf{p}_i - \mathbf{p}_{i(\mathcal{N}_\omega)}||^2. \quad (4)$$

This function is first minimized subject to a rigid transform applied to the inserted points, i.e.

$$\min_{(\mathbf{R},\mathbf{t}) \in \mathrm{SE}(3)} d(\mathcal{N}_\omega, \mathbf{R}\mathcal{F}_\omega + \mathbf{t}). \quad (5)$$

The translation is derived by aligning the centroids of the two sets; the rotation is determined using the singular value decomposition (SVD). The closest point relationships are updated based on the rigid transformation $(\mathbf{R}, \mathbf{t})$ and the procedure is repeated until convergence (this is essentially the iterative closest point procedure (ICP)).

Next, an elastic warp function is determined to further minimize the fitness functional. This non-rigid transformation $\mathbf{T}$ can be expressed as a polynomial in matrix form, i.e.

$$\mathbf{T} = \{t_{ij}\}, \quad t_{ij} = r_{ij} + c_{ij} + f_{1j}x + f_{2j}y + f_{3j}z \quad (6)$$

where $\{r_{ij}\}$ are the coefficients of the rotation found in the first step, $\{c_{ij}\}$ are additional linear coefficients, and $\{f_{ij}\}$ are coefficients of a quadratic function (note that $f_{ij} = f_{ji}$). Minimizing

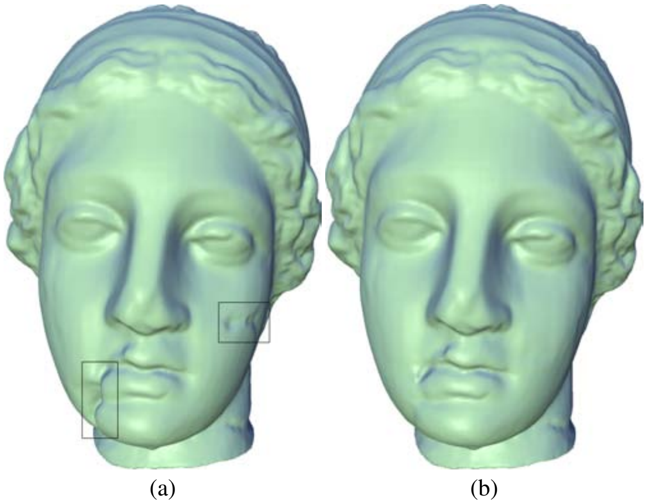


Figure 8: A smooth area around the cuts in Igea's face (a) is used to complete the cuts with smooth fitting parts (b).

$d(\mathcal{N}_\omega, \mathbf{T}\mathcal{F}_\omega)$ leads to a linear system in the unknown coefficients, which can be solved for the coeficients. We have found that this unconstrained solution sometimes leads to large quadratic coefficients $\{f_{ij}\}$ representing strongly deforming, visually disturbing non-rigid transformations. To avoid this, we add a regularization term: the squares of the quadratic coefficients are minimized. This leads to additional linear equations of the form $f_{ij} = 0$. The combined linear system resulting from both minimizations has to be solved in a least squares sense, which is done using the singular value decomposition (SVD).

After applying the so-defined non-rigid transformation T to the points in $\mathcal{F}_\omega$, the closest point relationship is re-established and the procedure is repeated. The process is illustrated in 2D in Figure 7.

8 Examples, limitations and performance

We have applied context-based surface completion to various point-sampled surfaces. The example in Figure 1 shows a non-trivial example where we removed a large set of samples from the hair of the statue. The missing part is completed with details taken from the existing parts of the hair. The completion process is fully automatic and is applied top down, where the first levels are coarse and smooth, and the finer details are added at a finer level of the hierarchy (see Figure 2).

Our training set is quite small as it is based only on the examples provided by the given object. In all our examples, we have enriched the training set by rotating the source objects by $\pi/4$ degrees in each of the X, Y and Z directions, and used mirroring[Drori et al. 2003; Kwatra et al. 2003]. This extended data set allows completing shapes with clear symmetric structure as in the "block-torus" model shown in Figure 3. The building blocks that complete the missing corner are taken from the opposite symmetric corner of the shape. Note that the completion cannot be computed in one trivial copy-and-paste step because the symmetry and appropriate scales are not given a priori and have to be inferred during the completion process.

Figure 10(a-d) demonstrates that the filling adapts well to the context of the surroundings. Note that in (d) the left side of the hole is more bumpy than the top of the Bunny's back. In the second row (e-h), we added Gaussian noise to the Bunny's surface to show that the reconstruction of the hole works on different scales.

One of the advantages of using implicit surface approximation is that it can easily deal with changes in topology. As explained, the identification of a hole is derived from the implicit surface approximations. As long as the initial approximation covers the insufficiently sampled regions, the subsequent levels refine and reconstruct the finer details. This is demonstrated in Figure 11, where a broken knot is repaired. In the top row, the broken knot has only one boundary, while in the bottom, the missing surface patch has two boundaries (i.e. is a topological cylinder rather than a disk).

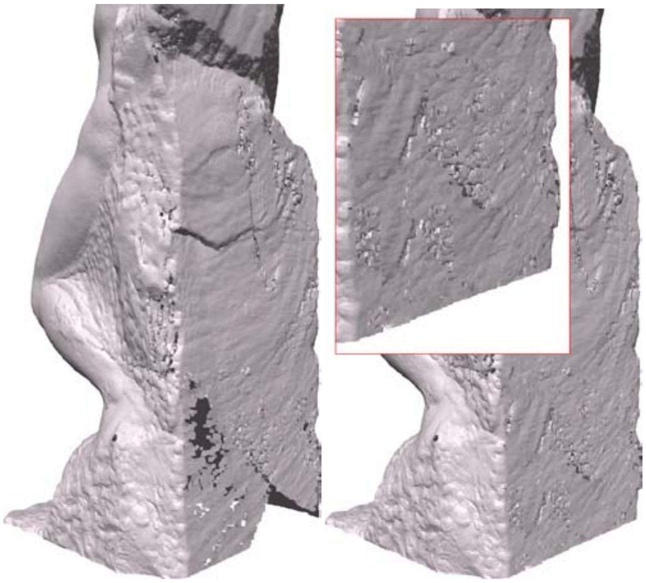


Figure 9: The back area of the sculpture ”Youth” by Michelangelo as reconstructed from the original scans (a). Context-based completion of the point-sampled surface(b).

Although our technique is fully automatic, user intervention is certainly possible. The user might want to specify what regions should be used for completion – effectively just limiting the example set. This extends 2D clone brushes used in image retouching applications (e.g. [Adobe 2002]) to surfaces in 3D.

Figures 12, 13 and 14 show raw point-sampled data acquired using only a few range scans so that some areas on the models are not adequately covered with sample points. We have additionally created large holes to demonstrate our method on surfaces with fine but non-stochastic geometric detail (in contrast to the relatively stochastic examples of the bunny and the knot, or the flat surface of the block torus). Note that the hole in the bone (see Figure 12) is completed with a surface that continues the ridge along the bone, whereas in a smooth completion, the ridge and the structural details on the bone are washed out. Similarly, the holes in the complex shape of the CAD model (Figure 13) are completed using the shape geometric features in context with the holes. The chair in Figure 14 is an interesting example: The large artificial holes introduced in the fabric are a challenge because of the overall smoothly varying surface with a clearly structured pattern. The problem lies in the fact that it is difficult to match high frequency geometry which is not sampled densely enough. The results show that our approach successfully aligns an imported patch with the surface. However, the slight misalignment in the detail structure is perceived as a jump in the lightness. Another example of the limitations of our method is shown in Figure 15, where an artificial hole introduced in Igea's eye area is not completed with the similar patch from the other eye area. As in 2D techniques, the completion process has no semantics. Thus, the completed surface is geometrically similar but loses the meaning.

The example shapes have fairly different numbers of point sam-

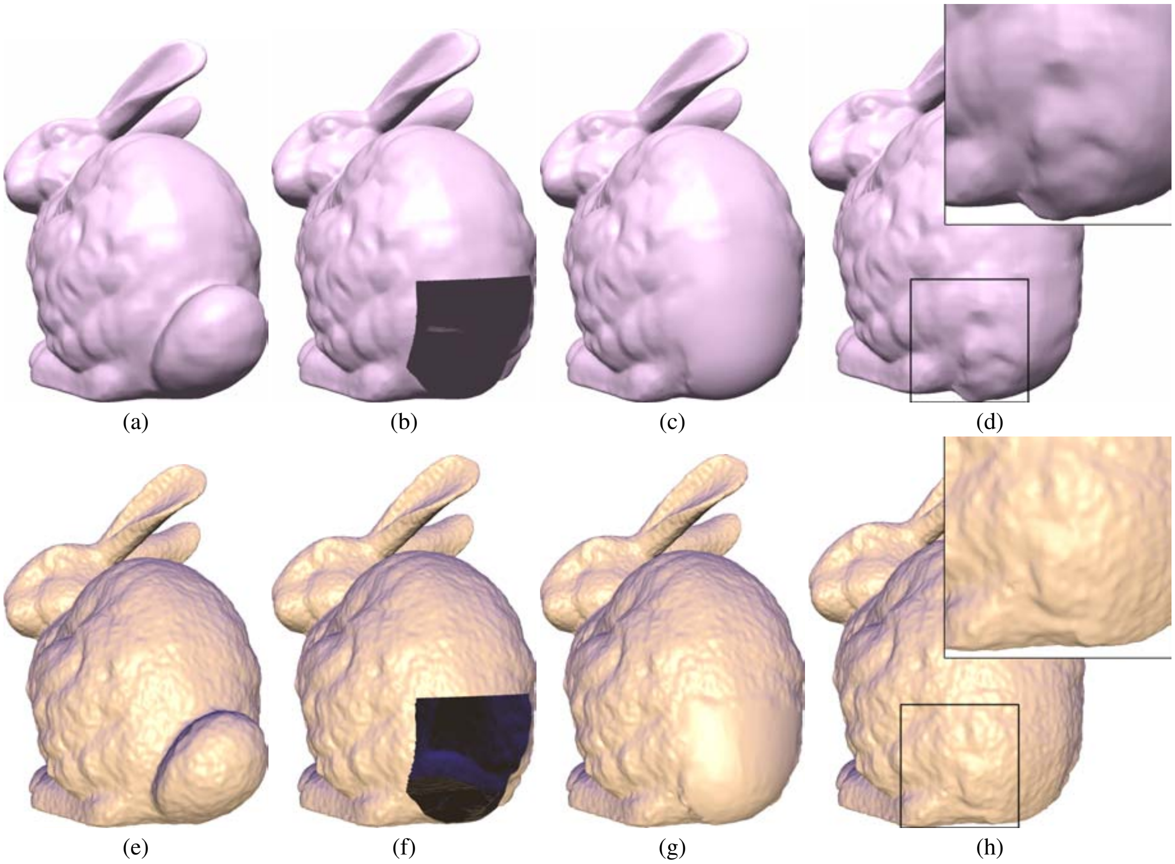


Figure 10: The tail of the Bunny is removed. Reconstructed by a smooth interpolation in (c) and (g), and by a context-based filling in (d) and (h). Note in (d) that the reconstructed area adapts to the context of its surroundings. In the second row, we added Gaussian noise to the Bunny surface to show that the reconstruction respects the context and performs equally well on different scales.

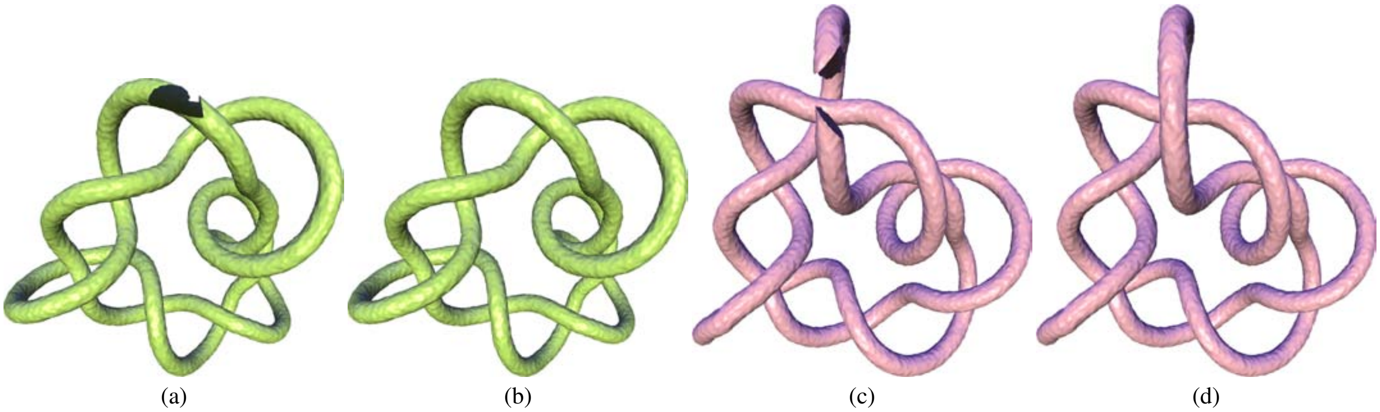


Figure 11: A broken knot is repaired. This example shows that the filling technique is not sensitive to the topology. As long the the initial algebraic surface approximation covers the missing part, the subsequent levels refine and reconstruct the finer details.

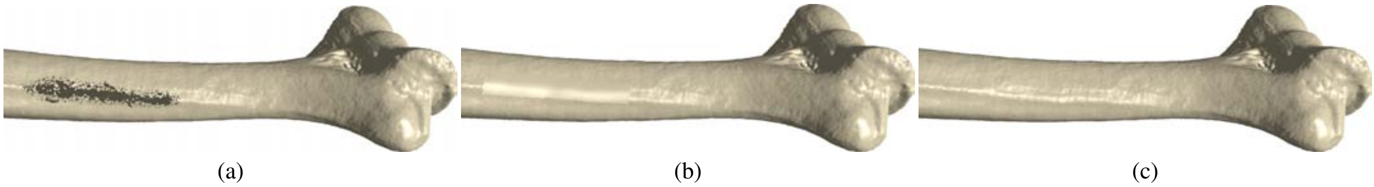


Figure 12: Completion of a missing region in a scan of a human bone (a). Smooth completion of the hole (b). Our method completes successfully the ridge in the surface from its context (c).

ples: We used the 1.7M points version of the 'Youth' model and the 0.5M points model of 'David', available from the Digital Michelangelo Project [Levoy et al. 2000]. The registered range scans of the CAD, chair and bone models contain 8M, 1.6M and 0.5M points respectively. The size of the other models is in the range of 10k to 150k points. We ran our experiments on an Intel Pentium 4 2.4Ghz with 1024Mb RAM on OS Win2000. The computation consists of three main parts: (i) preprocessing of the input point set, (ii) the computation of the approximated surface and (iii) the fit of an imported part. We have found that the computation times are surprisingly small: the preprocessing time directly depends on the size of the input set and takes up to five minutes for the largest models. The approximation of the surface requires up to 20 seconds per iteration, where each temporary fitting of points requires one iteration. Searching and fitting a patch based on the current surface approximation takes around 10 seconds. Most of the computation time is spent on the computation of an implicit surface to approximate the given point set. The search is the fastest step because we compute the signatures of all valid cells during preprocessing.

Our approach scales well with the number of points. However, we implemented an in-core system, so that in practice the number of points is limited. The main limitation of our algorithm is that the identification of the missing regions (or "holes") relies on a local surface approximation that provides a reasonable initial smooth surface approximation. If the points are too noisy, or poorly sampled, this process might fail. Another limitation is that the result of the surface completion procedure can only contain copies from the example set. If no appropriate examples exist, the match might be poor. A more subtle problem results from the irregular sampling. Our method is limited by the relation between the sampling density and the detail frequency. To capture fine structural details the cell must be small enough with respect to the detail size. Figure 14 shows a chair whose cushion consists of frequency details which are too high compared to the sampling density. Thus the synthesis of the surface cannot faithfully capture the cushion details.

9 Conclusions and future work

In this work we have introduced a context-based method for completing point-sampled surfaces in areas where the sampling is insufficient. In contrast to filling these areas with a smooth patch, our approach copies and adapts parts of the surface that respect the geometric characteristics around the filled region. The method is fully automatic and applicable in a wide range of scenarios. It is particularly useful to repair models resulting from range scanning as well as for all modelling operations that remove parts of the surface.

The spatial hierarchy has an obvious impact on the resulting surface. In the future, we would like to explore other spatial hierarchies and compare the results with the octree we have used. Another interesting avenue is to enlarge the search space of examples by building a shape database. However, a large set of examples would possibly require optimized search strategies for the identification of the best matching cell.

Our results show that the encouraging results of context-based methods can be extended to general surfaces without a regular parameterization. We would like to extend the method further to other classes of objects without an innate regular parameter domain, such as deformable or moving shapes, and light fields. For even better practical usage it seems important to couple the completion of the surface geometry with completion of surface textures using image completion methods.

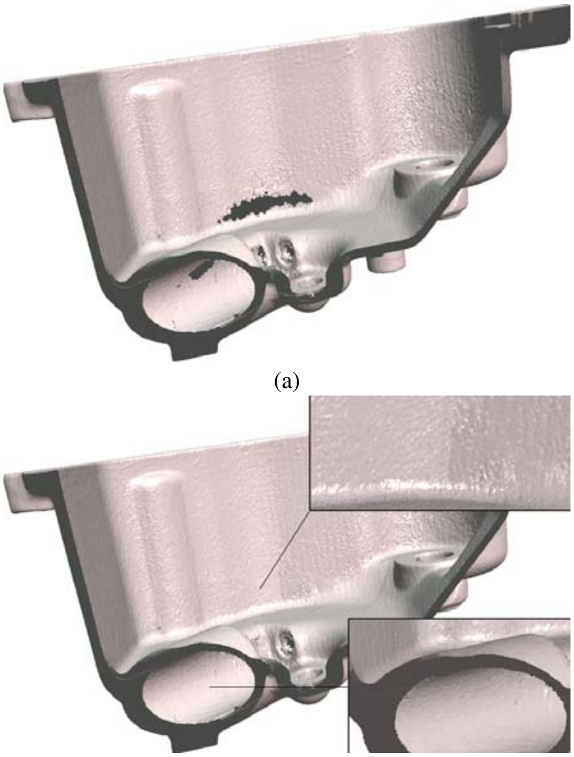


Figure 13: Due to the complexity of CAD models, scans typically result with holes. We slice the CAD model to show the interior and exterior holes (a) and their completion (b).

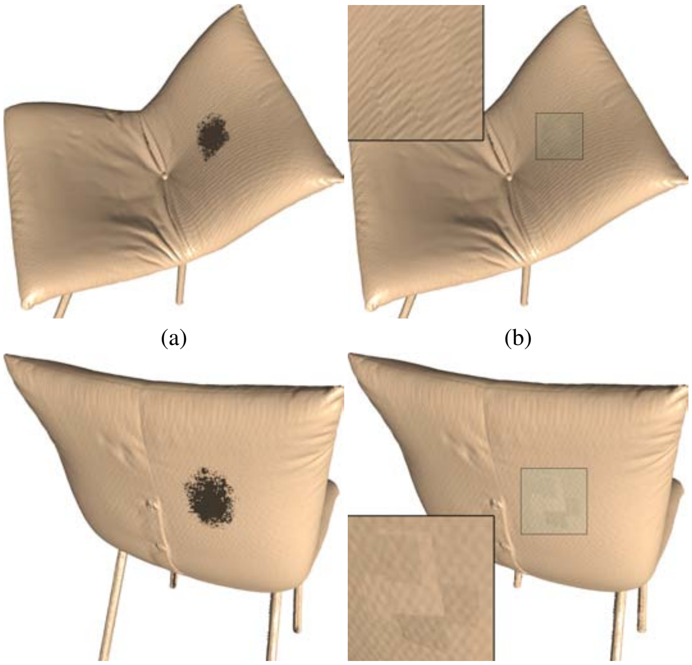
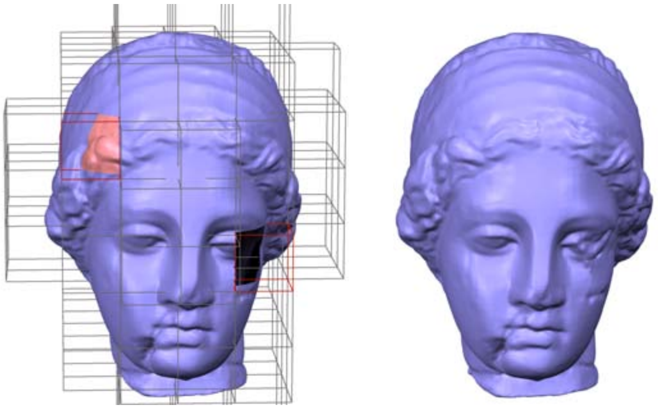


Figure 14: A chair model with artificially introduced holes. Our method was not able to align the completed patches with the small regular details of the chair's fabric.



(a) (b)

Figure 15: Igea's eye area is artificially removed (a). Our method completes the removed area using a matching patch from the hair (in pink) (b).

10 Acknowledgments

We would like to thank David Levin and Olga Sorkine for their fruitful discussions and support. This work was supported in part by the Israel Science Foundation founded by the Israel Academy of Sciences and Humanities and by the Israeli Ministry of Science. The models are courtesy of the Digital Michelangelo Project 3D Model Repository (Head of David and ”Youth” statue), the Stanford 3D Scanning Repository (bunny), Cyberware (Igea), Darmstadt University of Technology (bone, CAD model and chair) and the Imager Computer Graphics Laboratory of the University of British Columbia (knot).

References

Adobe, 2002. Adobe photoshop 7.0.

Amenta, N., Bern, M., and Kamvysselis, M. 1998. A new voronoi-based surface reconstruction algorithm. In *Proceedings of SIGGRAPH 98*, Computer Graphics Proceedings, Annual Conference Series, 415–422.

Bajaj, C. L., Bernardini, F., and Xu, G. 1995. Automatic reconstruction of surfaces and scalar fields from 3d scans. In *Proceedings of SIGGRAPH 95*, Computer Graphics Proceedings, Annual Conference Series, 109–118.

Barequet, G., and Sharir, M. 1995. Filling gaps in the boundary of a polyhedron. *Computer Aided Geometric Design 12*, 2, 207–229.

Besl, P., and McKay, N. 1992. A method for registration of 3d shapes. *IEEE Transaction on Pattern Analysis and Machine Intelligence 14*, 2, 239–256.

Chen, Y., and Medioni, G. 1992. Object modeling by registration of multiple range images. *Image and Vision Computing 10*, 3, 145–155.

Clarenz, U., Diewald, U., Dziuk, G., Rumpf, M., and Rusu, R. 2004. A finite element method for surface restoration with smooth boundary conditions. In *CAGD*. submitted, available from http://numerik.math.uni-duisburg.de/.

Criminisi, A., Pérez, P., and Toyama, K. 2003. Object removal by exemplar-based inpainting. In *2003 Conference on Computer Vision and Pattern Recognition (CVPR 2003)*, 721–728.

Curless, B., and Levoy, M. 1996. A volumetric method for building complex models from range images. In *Proceedings of SIGGRAPH 96*, Computer Graphics Proceedings, Annual Conference Series, 303–312.

Davis, J., Marschner, S. R., Garr, M., and Levoy, M. 2002. Filling holes in complex surfaces using volumetric diffusion. In *Proceedings of the 1st International Symposium on 3D Data Processing Visualization and Transmission (3DPVT-02)*, IEEE Computer Society, Los Alamitos, CA, G. M. Cortelazzo and C. Guerra, Eds., 428–438.

Drori, I., Cohen-Or, D., and Yeshurun, H. 2003. Fragment-based image completion. *ACM Transactions on Graphics 22*, 3 (July), 303–312.

Efros, A. A., and Freeman, W. T. 2001. Image quilting for texture synthesis and transfer. In *Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, 341–346.

Efros, A. A., and Leung, T. K. 1999. Texture synthesis by non-parametric sampling. In *IEEE International Conference on Computer Vision*, 1033–1038.

Hertzmann, A., Jacobs, C. E., Oliver, N., Curless, B., and Salesin, D. H. 2001. Image analogies. In *Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, 327–340.

Hertzmann, A., Oliver, N., Curless, B., and Seitz, S. M. 2002. Curve analogies. In *Rendering Techniques 2002: 13th Eurographics Workshop on Rendering*, 233–246.

Ilic, S., and Fua, P. 2003. Implicit meshes for modeling and reconstruction. In *2003 Conference on Computer Vision and Pattern Recognition (CVPR 2003)*, 483–492.

Jia, J., and Tang, C.-K. 2003. Image repairing: Robust image synthesis by adaptive nd tensor voting. In *2003 Conference on Computer Vision and Pattern Recognition (CVPR 2003)*, 643–650.

Kwatra, V., Schödl, A., Essa, I., Turk, G., and Bobick, A. 2003. Graphcut textures: Image and video synthesis using graph cuts. *ACM Transactions on Graphics 22*, 3 (July), 277–286.

Levoy, M., Pulli, K., Curless, B., Rusinkiewicz, S., Koller, D., Pereira, L., Ginzton, M., Anderson, S., Davis, J., Ginsberg, J., Shade, J., and Fulk, D. 2000. The digital michelangelo project: 3d scanning of large statues. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 131–144.

Liepa, P. 2003. Filling holes in meshes. In *Symposium on Geometry Processing*, 200–205.

Ohtake, Y., Belyaev, A., Alexa, M., Turk, G., and Seidel, H.-P. 2003. Multi-level partition of unity implicits. *ACM Transactions on Graphics 22*, 3 (July), 463–470.

Savchenko, V., and Kojekine, N. 2002. An approach to blend surfaces. In *CGI*, 139–150.

Sun, J., Zheng, N.-N., Tao, H., and Shum, H.-Y. 2003. Image hallucination with primal sketch priors. In *2003 Conference on Computer Vision and Pattern Recognition (CVPR 2003)*, 729–736.

Taubin, G. 1994. Distance approximation for rasterizing implicit curves. *ACM Transactions on Graphics 13*, 1 (Jan.), 3–42.

Verdera, J., Caselles, V., Bertalmio, M., and Sapiro, G. 2003. Inpainting surface holes. In *2003 International Conference on Image Processing 2003 ICIP*.

Wei, L.-Y., and Levoy, M. 2000. Fast texture synthesis using tree-structured vector quantization. In *Proceedings of ACM SIGGRAPH 2000*, Computer Graphics Proceedings, Annual Conference Series, 479–488.

Ying, L., Hertzmann, A., Biermann, H., and Zorin, D. 2001. Texture and shape synthesis on surfaces. In *Rendering Techniques 2001: 12th Eurographics Workshop on Rendering*, 301–312.

Zhang, Z. 1992. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision 13*, 2, 119–152.