RC23991 (W0606-132) June 28, 2006
Computer Science

IBM Research Report

Reverse Engineering Methods for Digital Restoration Applications

Ioana Boier-Martin
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Holly Rushmeier
Yale University
New Haven, CT

Reverse Engineering Methods for Digital Restoration Applications

Ioana Boier-Martin
IBM T. J. Watson Research Center
Hawthorne, New York, USA
ioana@us.ibm.com

Holly Rushmeier
Yale University
New Haven, Connecticut, USA
holly@acm.org

Abstract

In this paper we discuss the challenges of processing and converting 3D scanned data to representations suitable for interactive manipulation in the context of virtual restoration applications. We present a constrained parameterization approach that allows us to represent 3D scanned models as parametric surfaces defined over polyhedral domains. A combination of normal- and spatial-based clustering techniques is used to generate a partition of the model into regions suitable for parameterization. Constraints can be optionally imposed to enforce a strict correspondence between input and output features. We consider two types of virtual restoration methods: (a) a paint restoration method that takes advantage of the normal-based coarse partition to identify large regions of reduced metric distortion suitable for texture mapping and (b) a shape restoration approach which relies on a refined partition used to convert the input model to a multiresolution subdivision representation suitable for intuitive interactive manipulation during digital studies of historical artifacts.

1. Introduction

Reverse engineering is the process of taking an object apart for the purpose of analyzing its workings in detail, usually with the intention of constructing a new object of similar or extended functionality. In the case of 3D scanning, the objects are physical items for which digital replicas are created so that the original objects can be further studied, manipulated, and / or reproduced with the help of the computer. Today, 3D scanning has become a commodity service, with multiple applications, ranging from industrial design, to retail, to entertainment, to archeology and cultural heritage. Most often, the scanner software allows for the acquired data to be converted to an unstructured 3D mesh. This representation is typically difficult to manipulate beyond basic display and conversions to other representations are needed.

In this paper we focus on the demands of restoring digital objects for cultural heritage applications. However, the methods we present are relevant to many other areas.

This is a revised and extended version of our previous paper [12]. In that paper we specifically considered the problem of converting a triangle mesh to a Catmull-Clark multiresolution subdivision surface. Here we place that work in the context of cultural heritage restoration applications. This context motivates the problem of partitioning to facilitate texture map editing in addition to the partitioning needed for conversion to a subdivision surface. We reformulate our techniques to deal with the different partitions needed for intuitive texture map and shape editing.

1.1. Related Work

Our work is motivated by experiences in virtual restoration applications and we build on previous work in geometriy processing.

Scanning and virtual restoration. The pipeline for processing to form a texture mapped triangle mesh from sensed data is well established [7]. Commercial products (e.g., Polyworks®, RapidForm® are available to convert point clouds to meshes. These products also include some standard CAE/CAD analysis tools such as measurements on surfaces and comparisons of as-built to originally digitally designed parts. However, the needs of cultural heritage applications [21, 33, 27, 7] are different from traditional CAE/CAD in several different respects. First, the objects of interest are typically not regularly machined objects, but rather complex free-form shapes that were produced manually and have been worn or broken over time through natural processes. Second, while shape alone is sometimes of interest, the material appearance of the object as well as its shape are being studied. Third, several alteration iterations are needed to evaluate the validity of a change. Often times, judgments are made on the basis of appearance and matching images and textual descriptions of appearance, rather than on more readily quantified requirements such as making parts fit together or minimizing weight. Fourth, the population using digital models in cultural heritage is much different from the population using CAE/CAD tools. While cultural heritage specialists may have extensive technical training, their primary interest is in the history of the objects being studied and the people that used them, and not in the technology itself. The cost of hiring technical specialists and the difficulty of working through a mediator make it impractical to include pure technicians on most teams. Finally, the hardware and software resources available to cultural heritage projects are generally much more limited than those of engineering design projects.

Geometry processing. Considerable literature exists on building high-quality parameterizations over triangulated base domains (see [30] for a recent example and references). In contrast, very little work has been done on deriving quadrilateral base complexes for arbitrary meshes. The most recent work is that of Levy et al. [40]. Their periodic parameterizations are used to derive high-quality quadrilateral meshes. However, these meshes may be dense (the size of the quads is directly related to the size of the smallest tubular feature) and typically contain T-junctions so they are not directly usable as base meshes for subdivision surfaces. Eck et al. [19] describe an automatic method for fitting B-splines to meshes of arbitrary topology. A quad base domain is generated indirectly from a triangulated one by simplification followed by pairing of neighboring triangles. In general, approaches based on mesh simplification suffer from several shortcomings: it is not clear how to determine when to stop the simplification, geometric error typically drives the process with little or no control over the resulting topology and connectivity, the simplified mesh is a triangle mesh for which a quadrilateral decomposition has to be found, and constraints are difficult to enforce [29]. Generating quadrilateral meshes by pairing of triangles (see also $4-8$ subdivision schemes [44]) pose additional problems: complete pairings may not always exist and finding ones that minimize distortion is expensive. Arbitrary meshes are typically parameterized by cutting and flattening. An extreme example is the parameterization of an entire mesh over a square [22]. Maintaining consistency across seams is difficult, especially if the model is to be modified (e.g., edited, compressed). In [23] conformal parameterizations of complex surfaces are computed without cutting. However, the resulting parameterizations are highly non-uniform and controlling them requires manual topology modification. Recent methods have targeted restricted classes of models. Mappings to either a sphere [39] or a plane [26] were used to recover quadrilateral meshes for genus 0 models with and without boundary, respectively. For the latter class, an interesting quad-dominant anisotropic remeshing method was described in [2]. Previous methods also include the user-driven approaches of Krishnamurthy and Levoy [31] to fit tensor-product B-spline patches to irregular meshes and that of Guskov et al. [24] to build quad base domains as part of their hybrid mesh representation. Techniques for converting given models to quadrilateral meshes have also been proposed in the mesh generation community. Advancing front and packing are among the most common strategies [38, 5]). Typically, the resulting meshes have a large number of quads and are not suitable as parameterization domains.

Also related to our approach are mesh partitioning methods. The fuzzy clustering technique of [28] produces patches which are not homeomorphic to disks and cannot be directly used for parameterization. Other methods generate disk-like patches according to various criteria (e.g., [20, 41, 34, 16]), but no single method addresses the combination of patch shape, distortion, number of neighbors, alignment to features, and constraints.

Surazhsky et al. [43] use centroidal Voronoi tessellations to generate dense isotropic triangulations. The centroid updates are performed in 2D and require computing local parameterizations of model regions. This is a remeshing approach which does not produce a parameterization domain for the input model.

1.2. Goals and Contribution

The consideration of the needs of cultural heritage leads us to the following goals:

- Object representations should make minimal demands on computational resources during editing.
- Editing operations should be as direct as possible on the target representation.

The first goal requires that we have representations that do not require loading the full model to make edits. The second goal requires that we have representations that make sense to a viewer. In the case of the material or texture, we want textures that are partitioned along features that are meaningful to the user and are not distorted. This allows the user to update the texture map directly in a standard image editor (such as PhotoShop® or PaintShopPro®).

To achieve these goals we propose a segmentation approach that allows the user to focus on minimal representations of the object necessary to achieve a particular task. We present a novel unifying framework for manipulating both geometric and attribute data to allow the shape and attributes of the object to be edited directly and efficiently, without loading the full 3D model into memory. When shape edits are necessary, we rely on the ease-of-use of semi-regular parametric representations to perform the editing in an intuitive way. We reformulate our clustering-based techniques developed in [12] to allow us handle different types of data partitioning required by virtual restoration and we illustrate the application of these techniques with multiple examples in this domain.

2. Digital Model Creation

In this section we first describe some specific cultural heritage applications and their workflow. Next we review the main requirements of virtual restoration applications as considered in the present work.

2.1. Cultural Heritage Applications

Photography has long been an essential tool for documenting and communicating information. Nevertheless, photographs provide limited insight into the construction of objects, as they do not allow any interaction. Artists use photographs as starting points for rendering objects in their original states, but such depictions can not be tested for physical feasibility. The power of 3D digital objects is that both the shape and material of the objects can be altered in physically feasible ways, and then critically evaluated to examine the validity of the changes.

An example of changes applied to a 3D scanned object is shown in Fig 1. in [45]. Fig 1 shows a scanned model of a sculpture of the head of the pharaoh Akhenaton and two proposed restorations of the sculpture's nose. Fig 8 shows a small Greco-Roman queen statue and a proposed restoration of the surface colors. Other examples include a study of the breaking and repair of Michelangelo's Florence Pietaà [45], the restoration of the original shape and finish of the Nara Great Buddha [27] and the virtual restoration of the Parthenon friezes [1].

The past two decades have seen tremendous efforts in obtaining the sensed data and building the initial 3D models. However, the second part of the process, editing and evaluating the models, has received less attention. Ironically, it is the second part that poses greater challenges as it requires multiple repetitions.

2.2. The 3D Scanning Pipeline

A wide variety of technologies are available for measuring 3D shape and appearance [10]. Two major types of scanners are frequently used for shape measurement – triangulation and time-of-flight scanners. The principle of triangulation scanners is to view a spot of light from a known source position with a sensor in a known position. Knowing the baseline distance between emitter and sensor and the angles of emission and viewing with respect to this baseline allow the calculation of the location of the spot. The principle of time-of-flight scanners is to estimate distance to a spot along a known ray direction by measuring the time between the emission of a pulse of light and the sensing of its return after reflection from the target to be measured. Using either technology, the output of a 3D scanner is typically a range image – a 2D array of measured 3D points for one surface patch on the object.

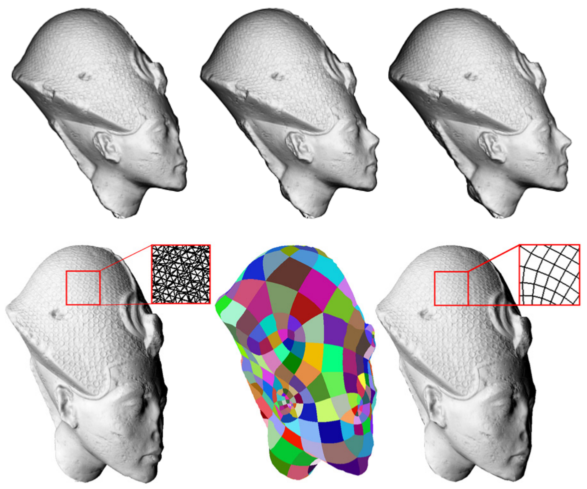


Figure 1. Top: Akhenaton statue with broken nose (left) and two virtual reconstructions (middle, right). Bottom: underlying parameterization computed using our method and used for interactive shape editing.

A pipeline of operations is necessary to convert collections of range images into a single object representation [7]. The pipeline begins by registering the range images into a global coordinate system. The points in overlapping range images are then adjusted along lines of sight to the scanner to account for measurement errors. The adjusted points are then joined into a single triangular mesh. Since scanners frequently oversample surfaces to capture small details, a simplification step usually follows the mesh generation step to reduce the number of vertices in the mesh.

Surface appearance properties, such as color and the magnitude of diffuse and specular reflectance, are generally captured using digital photography. The digital images are aligned to the geometric model either by calibration, or by finding correspondences between geometric features that can be computer from both the digital images and the geometric model. Once aligned, the digital images need to be processed to estimate the intrinsic surface properties [6, 32].

Once the digital images are processed to estimate surface properties, they need to be projected onto the geometric surface and combined into a single texture. To map a single image onto a 3D surface, the surface needs to be parameterized. In general surfaces need to be partitioned into pieces, and then projected or flattened onto a plane. Most commonly, early methods either used each triangle as a separate partition or relied on greedy approaches to produce groups of triangles forming nearly flat areas (e.g., [6]).

2.3. Virtual Restoration Requirements

The 3D scanning pipeline produces a triangle mesh model that is difficult to modify for virtual restoration. We can make restoration considerably easier for the user by partitioning the model into large patches for texture mapping and into quadrilaterals for conversion to a Catmull-Clark subdivision surface.

Virtual paint restoration. In the case of virtual paint restoration, there are two main categories of applications (see Fig 2): (a) cultural heritage studies which require color modifications across regions of the object (e.g., to test hypotheses regarding the original appearance of the objects or to simulate restoration procedures before they are performed) and (b) digital content creation applications which require touch up to remove scanning artifacts.

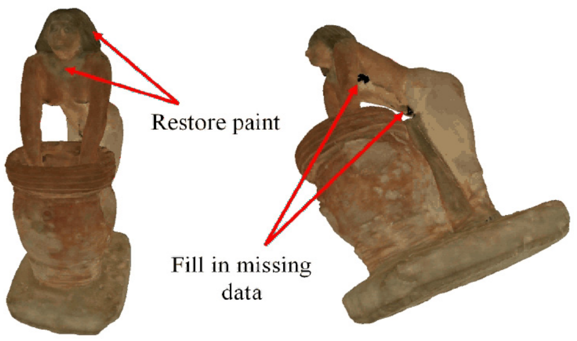


Figure 2. Virtual paint restoration is needed in different contexts: to remove artifacts introduced by scanning (right) or to study different hypotheses on how the original might have looked like (left).

At first it would seem that emulating physical painting on a 3D object would be the most natural and efficient method to perform paint restoration. Several commercial packages allow direct painting on digital objects. Painting on relatively simple digital objects is a straightforward task for trained digital artists. However, for the detailed models of free-form shapes obtained by scanning, direct painting is not practical. In many cases the full models can not even be loaded in memory for for interactive viewing, let alone for more elaborate interactions.

One alternative is to load a simplified version of the geometric object and paint the texture using this simplified version. Except for very limited simplification this technique causes distortions in the texture that make it difficult to see the paint adjustments that will appear correct on the full model (see Fig 4).

Another alternative is to use the *projection paint* mode used in products such as BodyPaint® and DeepPaint®. A simplified version of the model is positioned while a high resolution version of the model is rendered. The user paints on this high resolution image. The results are then reprojected back onto the model. This approach still requires the user to interact with a 3D system. It also resamples and reprojects texture that may not be of as high a resolution as the captured texture.

Our goal is to produce a texture map with large undistorted patches, and to insure that critical areas for editing are in undivided regions. Large patches make it easy to edit textures without introducing discontinuities. Distorted textures allow very large patches, but are very difficult to edit to apply specific details. For users who are not technical experts it is difficult to explain why a texture map shows a distorted view of the object, and how they should mentally transform the change in flat texture to the three dimensional object. The result is that when they edit a distorted texture they need to iteratively be reprojected on to the object in a 3D system to observe the effect.

In section 3.3 we describe a method that allows us to generate texture maps that can be edited within a standard 2D paint program, without the need for cumbersome 3D manipulation.

Virtual shape restoration. 3D shape restoration requires the ability to edit the geometry of the digital model to produce a desired shape. While it is conceivable to selectively edit the triangle mesh output by the scanning software directly [42], interactive methods for general purpose editing with constraints are yet to be found for such representations. In contrast, parametric surfaces have been extensively used for modeling and design. In particular, subdivision surfaces have become a fundamental representation for styling and have obtained validation through repeated use in entertainment as well as industrial applications [46]. Hence, we proceed to automatically derive a semi-regular representation that approximates the input mesh. The underlying base domain construction for this representation is described in section 3.3.

3. Partitioning

We now describe the overall clustering-based framework that produces the two different types of partitions discussed in the previous section.

3.1. Overview of Voronoi Partitioning Methods

We focus our attention on partitioning techniques that allow for easy-to-do virtual touch-ups. Our methods rely on concepts from clustering and quantization theory. In particular, we consider *Constrained Centroidal Voronoi Tessellations* (CCVT) [18] to create decompositions suitable for editing. Our key idea is to use CCVTs to segment a given 2-manifold into regions of reduced normal variation (i.e., almost flat) and then to use these resulting regions for the purpose of texturing the model, as well as for further decomposition for subsequent remeshing to support shape editing tasks. We briefly review the basic concepts related to CCVTs. Detailed properties can be found in [18]. Their applicability to segmentation tasks was studied in detail in [12].

Given a bounded domain $\Omega \subset R^n$ and a set of K sample points $\{c_i\}_{i=1}^K \in \Omega$, the *Voronoi partition* or *tessellation* $\mathcal{V}$ induced by $\{c_i\}_{i=1}^K$ on Ω is defined as:

$$\mathcal{V} = \{V_i = \{x \in \Omega : |x - c_i| < |x - c_j|, j = 1, \cdots, K, j \neq i\}, \\ i = 1, \cdots, K\}$$

The points $\{c_i\}_{i=1}^K$ are referred to as *generators* of the corresponding Voronoi regions V_i.

Definition 1. A *centroidal Voronoi tessellation* (CVT) is a Voronoi tessellation in which the centroids (i.e., centers of mass) of the regions serve as their generators.

CVTs are closely related to statistical clustering algorithms. For discrete data sets, CVTs can be identified with K-means clustering methods [17]. This concept can be adapted to produce CVTs over arbitrary surfaces, by restricting $\Omega = S \subset R^n$ to be a compact continuous surface and letting $\{c_i\}_{i=1}^K \in S$ be a set of sample points on it [18].

Definition 2. A *constrained centroidal Voronoi tessellation* (CCVT) is a Voronoi tessellation for which the *constrained mass centroid* c_i^* of each region V_i serves as its generator. The constrained mass centroid of a region $V_i \in S$ is defined as:

$$c_i^* = argmin_{c \in S} F_i(c), \text{ where } F_i(c) = \int_{V_i} \rho(x)|x - c|^2 dx$$

where $\rho(x)$ is a given density function over Ω.

3.2. Our Partitioning Framework

We consider input models represented as 2-manifold triangle meshes M_I of arbitrary topology, possibly with boundaries and possibly with an associated texture image. Feature curves along edges of M_I may be tagged as constraints. Such curves may be specified through user input or as a result of an automatic detection procedure.

We seek to build an atlas of patches covering the input model and suitable for texturing and subsequently for interactive shape editing. As our final representation we target a Catmull-Clark [13] multiresolution subdivision surface which allows for intuitive interactive shape editing operations to be performed [8, 9, 11]. We define the target representation through a control mesh hierarchy with L levels $M_{H_0}, \cdots, M_{H_{L-1}}$. Automatic conversion from an input triangle mesh (with feature constraints) to such a representation is a difficult problem. We propose a Voronoi-based clustering technique such that:

1. M_{H_0} is a coarse quadrilateral mesh with a predominant number of regular control points (i.e., valence 4 in the interior and valence 2 on the boundary; we define the *valence* of a control point as the number of faces adjacent to it).
2. $M_{H_{L-1}}$ is a fine mesh such that its control points (or alternatively, their projections onto the limit surface of subdivision) are located on the input mesh M_I.
3. Each mesh M_{H_i} is obtained from the coarser mesh $M_{H_{i-1}}$ by applying the Catmull-Clark [13] subdivision rules, for $i = 1, \cdots, L-1$. To accurately capture the input shape, the positions of control points on each level may be perturbed from the locations computed by subdivision using multiresolution detail vectors.
4. The texture of the input model can be transferred to the control meshes $M_{H_i}, i = 1, \cdots, L-1$ with reduced distortion.

Center-Based Segmentation. Our segmentation method makes use of CCVTs in two distinct stages. An initial decomposition is computed for the purpose of texture mapping. The requirements in this case are that the resulting regions be relatively flat height fields to ensure reduced distortion during mapping. Constraints must be considered to ensure natural partitioning according to features and to simplify the paint restoration processes. A second decomposition proceeds to further break down the clusters found in the first stage into regions suitable for remeshing. In this case, the requirements for the regions are much stricter, as the boundaries of the regions will be directly used to extract a base mesh for the multiresolution representation. In the first case, the CCVT is computed based on normal information. In the second case, a combination of positional and normal information is used to generate a CCVT with well-shaped regions for remeshing. For texturing and texture editing we look for partitions that satisfy the following requirement:

(R1) For each region there exists a direction $\overrightarrow{h}$ in space such that the corresponding geometry defines a height-field along $\overrightarrow{h}$ and can be approximated within some tolerance by a plane perpendicular to $\overrightarrow{h}$.

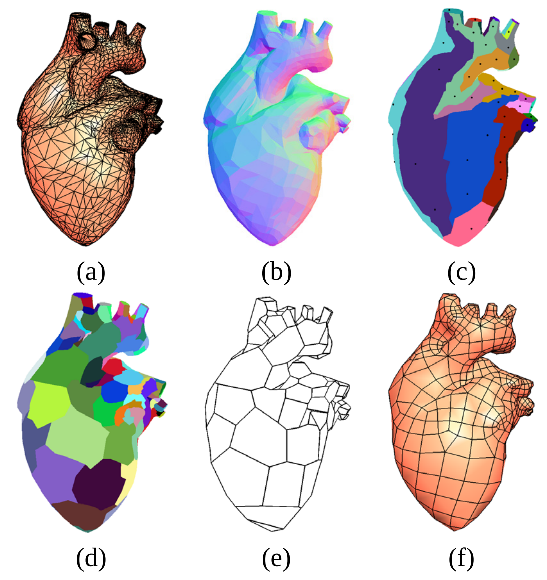


Figure 3. Segmentation steps: (a) Input. (b) Variation of normals across the model. (c) Normal-based clustering and initial placement of generators (black dots). (d) Final partition after spatial-based clustering (using the generators from (c)) and cluster refinement. (e) Coarse polyhedral approximation extracted from the partition in (d). (f) Coarse level of a Catmull-Clark representation after resampling and multiresolution analysis.

Subsequently, to parameterize the model for shape editing we further process the initial partition to also satisfy the following requirements:

(R2) Each region is homeomorphic to a disk.

(R3) The closed piecewise linear curve defining the boundary of each region can be approximated within some tolerance by a convex polygon.

(R1) is necessary because of the way in which textures are synthesized (see section 2.2). Also, a mapping-dependent distortion is likely to be introduced when the mesh geometry is parameterized onto the face. To reduce the amount of distortion and to avoid resampling problems, we enforce (R1). (R2) is needed if mesh faces are to correspond to regions and (R3) ensures that they are well-shaped.

Algorithm 1 provides a general-purpose method for segmentation according to generic properties associated with each triangular face of the input mesh (e.g., normal or location information) . The algorithm computes centroidal decompositions based on these properties using a MacQueen-type approach [35]:

Algorithm 1 (center-based clustering of mesh faces):

```
Given an input mesh $M$ with $F$ faces,
per-face property data items $\{d_f\}_{f=1}^{F}$,
and an initial set of $K$ generator items $\{c_i\}_{i=1}^{K}$:
Repeat
    For each face $f$ of $M$ do
        1. Find $c_{i^*}$ among $\{c_i\}_{i=1}^{K}$ closest to $d_f$
        2. $j$ = Label($f$), Label($f$) $= i^*$
        3. $c_{i^*} \leftarrow (|C_{i^*}|c_{i^*} + d_f)/(|C_{i^*}| + 1)$,
           $c_j \leftarrow (|C_j|c_j - d_f)/(|C_j| - 1)$,
until (convergence)
```

($|C|$ denotes the number of faces of cluster C; the label of each face is the index of the cluster to which it belongs)

Depending on the application (e.g., paint or shape restoration), we apply Algorithm 1 to obtain partitions suitable for the restoration operations.

3.3. Partitioning for Paint Restoration

We propose forming a texture that simplifies the work flow for restoring the surface texture and minimizes the amount of interaction the user needs to have in a 3D viewer. By appropriately partitioning the surface, we can form a texture map that can be updated using any standard 2D paint program.

An initial classification of input mesh faces into approximately flat regions / clusters suitable for texturing is obtained through center-based clustering of normals. Face normals are computed and Algorithm 1 is applied with $d_f = normal(f)$, for all input faces f. The initial generators are chosen to be a small fixed subset of the set of all possible unit normal vectors (those pointing from the origin to the vertices, mid-edges, and face centers of a cube centered at the origin). To reduce the influence of geometry discretization, the normals are smoothed prior to classification. The result is a partition $\mathcal{P}_\mathcal{N}$ of the input mesh into regions of reduced normal variation.

This process is illustrated in Fig 3b. The variation of normals over the heart model is shown using a linear mapping of normal vector components to RGB color. The resulting mesh decomposition is shown in Fig 3c.

In the absence of constraints, the resulting partition tends to conform to salient features of the model. However, the boundaries of the regions may not conform to semantic components of the model (e.g., follow a hairline). To enforce separation along such boundaries we allow the user to outline features of interest and we use clustering with constraints (see section 3.4) to segment the model.

Figs 8 and 9 illustrate the results of virtual paint restoration on two digital models corresponding to artifacts from the Egyptian National Museum collection (see section 4).

For models that only need texture restoration, no further partitioning is necessary.

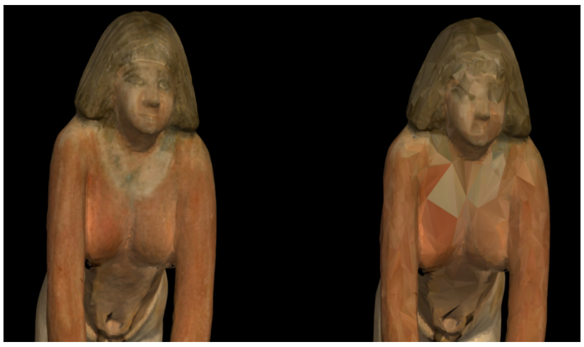


Figure 4. Drawback of 3D paint programs that require simplification: severe artifacts appear when the texture is re-mapped onto the simplified model.

3.4. Partitioning for Virtual Shape Restoration

For models that require shape restoration, we continue our Voronoi-based decomposition approach, starting from the partition $\mathcal{P}_\mathcal{N}$ previously obtained. For smooth, relatively simple shapes, $\mathcal{P}_\mathcal{N}$ has nice regions which are suitable for remeshing. However, this cannot be guaranteed. Since only a small number of regions are generated at this step, we can quickly check if all of them can be approximated by convex polygons. If this is not the case, we use the resulting decomposition for further refinement, as described next. In the interest of clarity, we defer the description of the shape check to section 4.

Cluster refinement. In contrast to the normal-based partitioning scheme just described, spatial CCVT-type decompositions of the mesh are guaranteed to produce almost circular regions centered around their generators. The main issue to be addressed is choosing the number of generators and their initial locations.

A common strategy to select initial generators is to place random samples over the mesh. Two things must be specified: the number of samples and their distribution. For our problem, it is difficult to estimate in advance how many generators to use: too many lead to base meshes with many faces, whereas too few fail to properly represent the input. Given a number of samples, their placement can be controlled using Monte Carlo methods or an error diffusion strategy [3], both of which can be computationally expensive for arbitrary models.

As an alternative, we use $\mathcal{P}_\mathcal{N}$ to estimate both the number of generators needed and their initial locations. Since regions of $\mathcal{P}_\mathcal{N}$ correspond to relatively flat model parts, the idea is to produce a refined decomposition in which clusters are centered on such flat spots. Ideally, we want to place samples along the medial axis of each region. In practice, we use the following heuristic method to avoid medial axis computation:

Algorithm 2 (region sampling):

```
Given a mesh region $R$ with $F_R$ faces:
1. Identify the set of boundary faces $B$
2. Uniformly distribute a set $S_R$ of $N$ random samples
      on $R \setminus B$
3. Select sample $s_1 \in S_R$ with the largest minimum
      distance to $B$
4. for $i = 2, \cdots, N$ do
      5. Select $s_i \in S_R$ farthest from $s_1, \cdots, s_{i-1}$
6. Retain $N_0$ of the $N$ selected samples
```

After generators have been placed on the mesh according to Algorithm 2, a new partition $\mathcal{P}_\mathcal{S}$ is computed with Algorithm 1, this time based on spatial information. We classify mesh faces according to their proximity (measured using Euclidean distance) to generators. We use $d_f = barycenter(f)$, for all input faces f.

Cluster cleanup. By construction, clusters obtained after the refinement stage are isotropic, almost circular in shape. If the generators are sufficiently dense, (R2) is guaranteed to be satisfied by all clusters [4]. However, since we use sparse generator sets, some clusters may not conform to (R2). The purpose of the cleanup phase is to enforce (R1), (R2), and (R3) on all clusters. For this, we treat the geometry of each region of $\mathcal{P}_\mathcal{S}$ individually. The height-field condition (R1) is checked first and if violated, the region is split using normal-based clustering. If any of the resulting subregions does not satisfy (R2) or (R3), the subregion is further split using spatial clustering. The normal-based split guarantees that the resulting regions are height fields. Subsequent spatial decompositions ensure that each height field is decomposed into regions with disk-topology and of approximately circular shape. Fig 3d shows the final decomposition of the heart model.

Coarse mesh extraction. Having found a partition of the input model that satisfies all of our requirements, the algorithm proceeds to generate a coarse mesh corresponding to it. Mesh vertices are placed at the points where three or more regions meet. Boundaries between regions define the edges of the coarse mesh.

Resampling and multiresolution analysis. Having found a coarse polygon mesh M_B that approximates the input model, we build a Catmull-Clark multiresolution representation by parameterizing the model over M_B. The faces of M_B correspond to regions that satisfy requirements (R1)-(R3). By virtue of the properties of CCVTs and Gersho's conjecture [18], the faces of M_B are predominantly hexagonal. Since all faces are convex polygons by construction, we can always find a partial quasi-conformal decomposition [36] as follows: split each face with $2k$ edges into $k - 1$ quads; split each face with $2k + 1$ edges into $k - 1$ quads and 1 triangle. We apply the decomposition recursively: first we find the quad with the lowest shape number (see section 4) that shares an edge with the face being quadrangulated, then we repeat the process for the remaining portion of the face. The decomposition is also perfect [36], as no Steiner points are introduced. This leads to a mesh consisting of predominantly quadrilateral faces (since the starting mesh was predominantly hexagonal) and a small number of triangular faces. One step of subdivision leads to a quadrilateral base mesh with a predominant number of regular vertices. Let M_{H_0} denote the quad mesh after one step of subdivision. M_{H_0} becomes a parameterization domain for the input model. To produce a hierarchy with subdivision connectivity, we subdivide M_{H_0} to the desired number of levels L and compute data on the finest level by resampling followed by multiresolution analysis similar to [47, 12].

Constrained parameterization. Our segmentation framework supports constraints in the form of closed curves along edges of the input model. Edges along feature curves are tagged and are used as clip boundaries during clustering. They also receive special handling during simplification and resampling. Fig 7 shows examples of segmentation with interior and boundary constraints.

4. Implementation and Results

Implementation details Our method has several parameters which can be set with default values. We briefly describe our choices. To check (R1), we compute the average normal of a region $\overrightarrow{h}$ and we test for the deviation of face normals from it. We also compute a plane perpendicular to $\overrightarrow{h}$ that passes through the center of the bounding box of the region. The region passes the test if the maximum distance from region vertices to this plane is within some tolerance (5% of the bounding box diagonal). In practice, we observed this latter test to be rarely needed, due to the way regions are created (normal-based clustering first).

We use a simple check to decide whether a region obtained by clustering satisfies (R3). We apply this check following normal-based clustering, to regions that are relatively flat, so it makes sense to measure the convexity of their boundaries. A coarse polygon is defined for each cluster with vertices at the confluence of three or more clusters. A polygon is considered convex if each of its interior angles is at most π. If a polygon is found to be convex, we then consider how well it approximates the boundary of the region. For this, we measure the maximum distance between points along the boundary curves and the corresponding coarse lines approximating them. If the distance is below a tolerance threshold, the approximation is acceptable. Since clusters are groups of faces, boundaries between them may appear jagged (a face equidistant from two generators is arbitrarily categorized in one of the corresponding clusters). We define the approximation tolerance as a constant times the average edge length in the cluster and we ignore such jaggedness in shape testing.

We use a fixed number of Laplacian smoothing iterations on the normal field prior to clustering to attenuate noise. The default number in our implementation is 10, however this value should be increased for very noisy data. For *random sampling* according to Algorithm 2, we followed the following strategy: uniformly distribute [37] a large number of samples over the region (we use $F_R/2$ samples) and of these select $N = F_R/4$ using a farthest-first approach; N_0 of the N samples are kept, where N_0 is the index of the sample for which the largest jump in the maxmin distance is observed [28].

Paint restoration. Fig 9 shows the result of segmenting an Egyptian artifact representing a woman making beer. The original statuette is 26.7 cm tall and was scanned at 1 mm resolution. The data was processed to remove redundant points in overlapping scans and to integrate the points into a single triangle mesh. The model has $112,478$ faces and was segmented into 30 regions (second from the left) in 3.6 minutes. User-imposed constraints (red curves; left) ensure that the face, chest, and hair of the model are properly separated to allow subsequent editing (i.e., without constraints the semantics of these parts is lost and they are arbitrarily split between regions, making it difficult to edit them). The middle images illustrate the corresponding texture map (note clearly identifiable head and chest textures). A comparison of the model before and after paint restoration is also shown (top right).

Fig 5 shows comparatively the texture maps obtained with a naive segmentation without constraints and with our technique. Features of interest such as face, hair, and necklace appear split between regions, making image-based editing difficult. In contrast, our segmentation produces a higher quality partition, with a much smaller number of regions and with region boundaries that respect user-imposed constraints, as well as salient geometric features.

Figure 8 shows a segmentation and detail painting of an Egyptian statue from the Greco-Roman period. Note that, in the absence of constraints, semantic features such as the face or the leg are not preserved.

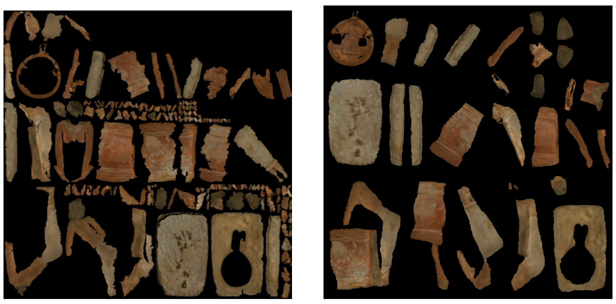


Figure 5. Left – naive segmentation of the model in Fig 8 without constraints results in many regions. Right – the result of our segmentation method.

Shape restoration. In Fig 1 we illustrate a high-resolution scan of a Akhenaton's head for which two different restorations of the nose have been created. We use a free-form variational editing tool (similar to [11]) to reconstruct the nose interactively. The user can smoothly deform the shape while preserving the Catmull-Clark connectivity. Many possibilities can be generated and evaluated quickly.

Additional parameterization examples are illustrated in Figs 7 and 9 (bottom right) for meshes with and without sharp features and boundaries.

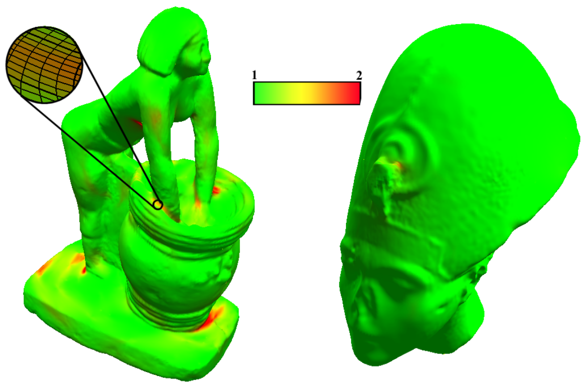


Figure 6. Color-coded visualization of the face shape numbers across two models (green corresponds to square shapes; gradation to red quantifies deviation from square).

Performance statistics We evaluate the quality of our output meshes using four different measures:

Hausdorff distance: provides a numerical estimate of the maximum distance between two meshes as the largest of two directed max-min distances from each mesh to the other. We use Metro [15] to report the results as a percentage of the mesh bounding box diagonal.

Volume ratio: quantifies the change in volume after remeshing (closed meshes only). It is defined as the ratio between the volume of the finest-level Catmull-Clark control mesh and that of the original.

Face shape distribution: characterizes the deviation of remeshed faces from an ideal shape (square in the isotropic case). Element shape quality is paramount in virtually all mesh-based computations, from finite element computations [14] to texture mapping and parameterization [41]. We plot a histogram of face shape measures for the faces of the base mesh. With the notations of Fig. **??**, we quantify the deviation of a 3D quadrilateral mesh face $(p_i, p_j, p_{j+1}, p_{j+2})$ from a square by considering the affine mapping of the triangle (p_i, p_j, p_{j+2}) to the right triangle with unit-length legs (O, q_1, q_2). As pointed out in [25, 41], the singular values of the Jacobian of this map characterize the local distortion between the right triangle and its 3D counterpart. We use the ratio of the singular values, i.e., the condition number of the Jacobian, as our shape measure. Simple calculations give $\mathcal{K}(J) = (|p_j - p_i|^2 + |p_{j+2} - p_i|^2)/(2A)$, where A is

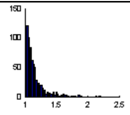
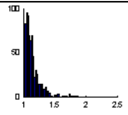
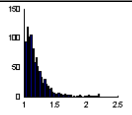
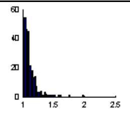
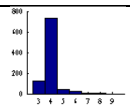
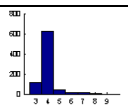
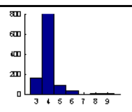
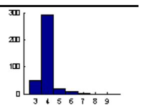
	heart	bunny	woman	pharaoh
Size:				
# F	7,412	71,040	112,478	315,462
# CF	173	155	219	71
Quality:				
HD	0.01	0.02	0.008	0.008
R_V	0.9994	0.9981	0.9991	0.9990
$\mathcal{H}_S$				
$\mathcal{H}_\nu$				

Table 1. Quality statistics: # F = number of input faces; # CF = number of coarse faces extracted with our method (before quadrangulation); HD = Hausdorff distance between input mesh and M_{H_3} (4th subdivision level); $R_V = V_{M_{H_3}}/V_I$ = ratio of output mesh volume (M_{H_3}) to the input mesh volume; $\mathcal{H}_S$ = histogram of face shape numbers for the quad base mesh M_{H_0}; $\mathcal{H}_\nu$ = histogram of vertex valences for M_{H_0}.

the area of (p_i, p_j, p_{j+2}). We define the *shape number* $\mathcal{S}$ of a quadrilateral face as the average of the four condition numbers at its vertices, normalized so that the shape number of a square equals 1.

Valence distribution: characterizes the regularity of the base mesh.

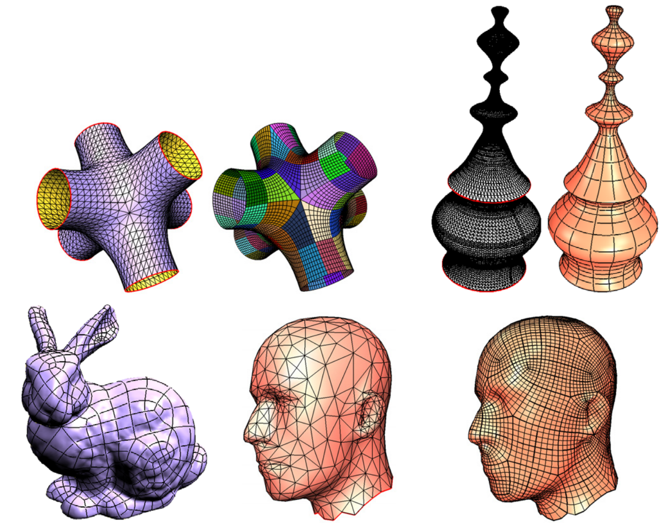


Figure 7. Quadrilateral parameterization examples (constraints shown in red).

Table 1 shows results for remeshed models with 4 levels of subdivision. The histograms confirm the quality of the base meshes, with a majority of well-shaped, almost square faces (shape numbers close to 1), a high percentage of regular vertices, and no highly irregular ones (valence 10 or higher). Face shape variation is also illustrated in Fig. 6.

Regarding time complexity, Algorithm 1 runs linearly through the input mesh faces and, for each face, through the set of generators. A simple test checking if any faces have changed clusters during an iteration is used as a *stopping criterion*. This is combined with a limit on the number of iterations to deal with cases when a face on the border between two clusters flips back and forth between them. We used a limit of 50 iterations for all our tests. The number of iterations to convergence for our models was less than 10. Algorithm 2 is also linear in the number of faces in each region. Multiresolution resampling is the bottleneck of the computation. To speed it up, we use a uniform grid approach as in [15] which gives reasonable results. In our tests, running times for the extraction of quadrilateral parameterization domains ranged from 6 to 40 seconds. Typical resampling times ran between a few seconds and 25 minutes for 4 levels of subdivision. The measurements were performed on a Pentium 4 3GHz PC with 2GB of RAM.

5. Conclusions and Future Work

In this paper we outlined the challenges of virtual restoration applications and we introduced novel methods for reverse engineering 3D objects for the purpose of interactive editing.

This work opens interesting possibilities for further exploration. Building anisotropic domains according to given direction fields is a natural next step. Another challenge is to use our constrained parameterization approach to generate maps between different models of similar shape with correspondence between features.

Acknowledgements

We thank Jingyi Jin and Richard Giantisco for their help. Portions of this work were performed as part of a collaboration with Dassault Systèmes. The Egyptian models are courtesy of the Egyptian Center for Documentation of Cultural and Natural Heritage and are available through the ”Eternal Egypt” website (www.eternalegypt.org).

References

[1] J. S. aand Ch. Tchou, N. Yun, P. Martinez, T. Hawkins, A. Jones, B. Emerson, and P. Debev. Digital reunification of the parthenon and its sculptures. In *4th International Symposium on Virtual Reality, Archaeology and Intelligent Cultural Heritage*, pages 41–50, 2003.

[2] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, and M. Desbrun. Anisotropic polygonal remeshing. *ACM TOG SIGGRAPH*, pages 485–493, 2003.

[3] P. Alliez, É. C. de Verdière, O. Devillers, and M. Isenburg. Isotropic surface remeshing. In *Proc. SMI03*, pages 49–58, 2003.

[4] N. Amenta, M. Bern, and M. Kamvysselis. A new Voronoi-based surface reconstruction algorithm. *Proc. SIGGRAPH 98*, pages 415–421, 1998.

[5] M. W. Bern and D. Eppstein. Quadrilateral meshing by circle packing. *Int. J. Comp. Geom. Appl.*, 10(4):347–360, 2000.

[6] F. Bernardini, I. M. Martin, and H. Rushmeier. High-quality texture reconstruction from multiple scans. *IEEE Transactions on Visualization and Computer Graphics*, 7(4):318–332, 2001.

[7] F. Bernardini and H. Rushmeier. The 3D model acquisition pipeline. *Computer Graphics Forum*, 21(2):149–172, 2002.

[8] H. Biermann, I. Martin, F. Bernardini, and D. Zorin. Cut-and-paste editing of multiresolution surfaces. In *Proc. SIGGRAPH 02*, pages 312–321, 2002.

[9] H. Biermann, I. Martin, D. Zorin, and F. Bernardini. Sharp features on multiresolution subdivision surfaces. In *Proceedings of Pacific Graphics 01*. IEEE, 2001.

[10] F. Blais. Review of 20 years of range sensor development. *Journal of Electronic Imaging*, 13(1):231–240, 2004.

[11] I. Boier-Martin, R. Ronfard, and F. Bernardini. Detail-preserving variational surface design with multiresolution constraints. In *Proc. SMI 04*, pages 119–128, 2004.

[12] I. Boier-Martin, H. Rushmeier, and J. Jin. Parameterization of triangle meshes over quadrilateral domains. In *Proc. SGP 04*, 2004.

[13] E. Catmull and J. Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. *CAD*, 10(6):350–355, 1978.

[14] P. G. Ciarlet. *Basic error estimates for elliptic problems. Handbook of Numerical Analysis*. North-Holland: Amsterdam, 1991.

[15] P. Cignoni, C. Rocchini, and R. Scopigno. Metro: Measuring error on simplified ssurfaces. *Computer Graphics Forum*, 17(2):167–174, 1998.

[16] D. Cohen-Steiner, P. Alliez, and M. Desbrun. Variational shape approximation. *ACM TOG SIGGRAPH*, pages 905–914, 2004.

[17] Q. Du, V. Faber, and M. Gunzburger. Centroidal Voronoi tessellations: Applications and algorithms. *SIAM Review*, 41:637–676, 1999.

[18] Q. Du, M. D. Gunzburger, and L. Ju. Constrained centroidal Voronoi tessellations for surfaces. *SIAM J. Sci. Comput.*, 24(5):1488–1506, 2003.

[19] M. Eck and H. Hoppe. Automatic reconstruction of B-spline surfaces of arbitrary topological type. In *Proc. SIGGRAPH 96*, pages 325–334, 1996.

[20] M. Garland, A. Willmott, and P. Heckbert. Hierarchical face clustering on polygonal surfaces. In *Proc. of ACM Symp. on Interactive 3D Graphics*, 2001.

[21] G. Godin, J. A. Beraldin, J. Taylor, L. Cournoyer, M. Rioux, S. El-Hakim, R. Baribeau, F. Blais, P. Boulanger, J. Domey, and M. Picard. Active optical 3d imaging for heritage applications. *IEEE Comput. Graph. Appl.*, 22(5):24–36, 2002.

[22] X. Gu, S. Gortler, and H. Hoppe. Geometry images. *ACM TOG SIGGRAPH*, 21(3):355–361, 2002.

[23] X. Gu and S.-T. Yau. Global conformal surface parameterization. In *Proc. SGP03*, pages 127–137, 2003.

[24] I. Guskov, A. Khodakovsky, P. Schröder, and W. Sweldens. Hybrid meshes: Multiresolution using regular and irregular refinement. In *Proc. Symp. Comp. Geom.*, pages 264–272, 2002.

[25] K. Hormann and G. Greiner. MIPS: An efficient global parametrization method. In *Curve and Surface Design*, pages 153–162. 2000.

[26] K. Hormann and G. Greiner. Quadrilateral remeshing. In *Proc. Vision, Modeling, and Viz 2000*, pages 153–162, 2000.

[27] K. Ikeuchi, A. Nakazawa, K. Hasegawa, and T. Ohishi. The Great Buddha project: Modeling cultural heritage for VR systems through observation. In *IEEE ISMAR03*, 2003.

[28] S. Katz and A. Tal. Hierarchical mesh decomposition using fuzzy clustering and cuts. *ACM TOG SIGGRAPH*, pages 954–961, 2003.

[29] Y. Kho and M. Garland. User-guided simplification. In *Proc. ACM I3D Symp*, pages 123 – 126, 2003.

[30] A. Khodakovsky, N. Litke, and P. Schröder. Globally smooth parameterizations with low distortion. *ACM TOG SIGGRAPH*, pages 350–357, 2003.

[31] V. Krishnamurthy and M. Levoy. Fitting smooth surfaces to dense polygon meshes. In *Proc. SIGGRAPH 96*, pages 313–324, 1996.

[32] H. P. A. Lensch, J. Kautz, M. Goesele, W. Heidrich, and H.-P. Seidel. Image-based reconstruction of spatial appearance and geometric detail. *ACM TOG*, 22(2):234–257, 2003.

[33] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk. The digital Michelangelo project: 3D scanning of large statues. In *SIGGRAPH '00*, pages 131–144, 2000.

[34] B. Lévy, S. Petitjean, N. Ray, and J. Maillot. Least squares conformal maps for automatic texture atlas generation. *ACM TOG SIGGRAPH*, pages 362–371, 2002.

[35] J. MacQueeen. Some methods for classification and analysis of multivariate observations. In *Proc. 5th Berkeley Symp. Math Statistics and Prob.*, pages 281–297, 1967.

[36] M. Müller-Hannemann and K. Weihe. Minimum strictly convex quadrangulations of convex polygons. In *Symp. Comp. Geom.*, pages 193–202, 1997.

[37] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin. Shape distributions. *ACM TOG*, 21(4):807–832, 2002.

[38] S. Owen, M. Staten, S. Canann, and S. Saigal. A survey of unstructured mesh generation technology. In *Proc. 7th Int. Meshing Roundtable*, pages 239–267, 1998.

[39] E. Praun and H. Hoppe. Spherical parameterization and remeshing. *ACM TOG SIGGRAPH*, pages 340–348, 2003.

[40] N. Ray, W. C. Li, B. Levy, A. Sheffer, and P. Alliez. Periodic global parameterization. Technical Report TR – submitted to ACM TOG, 2005.

[41] P. V. Sander, J. Snyder, S. J. Gortler, and H. Hoppe. Texture mapping progressive meshes. In *Proceedings of SIGGRAPH 01*, pages 409–416, 2001.

[42] O. Sorkine. Eurographics'05 State-of-the-art Report, 2005.

[43] V. Surazhsky, P. Alliez, and C. Gotsman. Isotropic remeshing of surfaces: a local parameterization approach. In *Proc. 12th Intl. Meshing Roundtable*, 2003.

[44] L. Velho. Quadrilateral meshing using 4-8 clustering. In *Proc. CILANCE'00*, pages 61–64, 2000.

[45] J. Wasserman. *Michelangelo's Florence Pieta'.* Princeton University Press, Princeton, NU, 2003.

[46] D. Zorin, P. Schröder, T. DeRose, L. Kobbelt, A. Levin, and W. Sweldens. SIGGRAPH'00 Course Notes, 2000.

[47] D. Zorin, P. Schröder, and W. Sweldens. Interactive multiresolution mesh editing. *Proceedings of SIGGRAPH 97*, pages 259–268, August 1997. ISBN 0-89791-896-7. Held in Los Angeles, California.

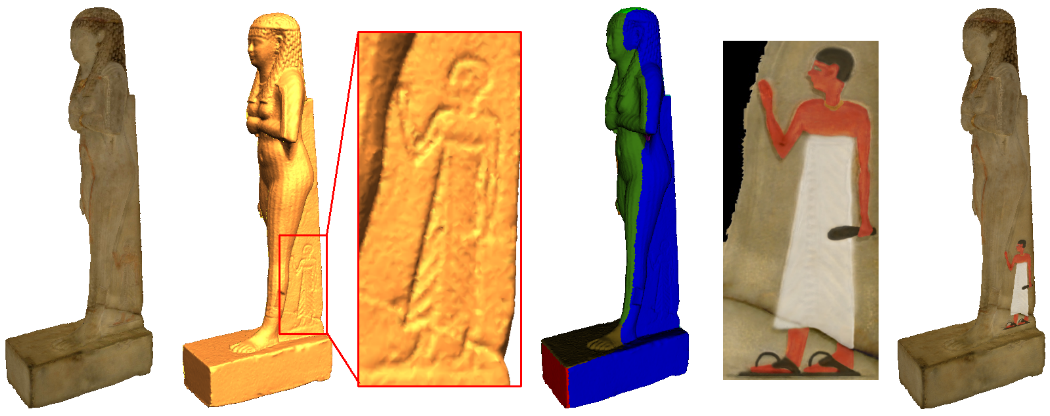


Figure 8. Left to right: input model with textures; geometry shaded without textures: human character detail on the queen's robe is enlarged for illustration purposes; segmentation using our method produces large height-field patches, well aligned with the main features of the model; previously painted 2D restoration of the character; the painted image is applied to the model by a simple cut-paste-and-blend operation in 2D texture space without the need for 3D intervention.

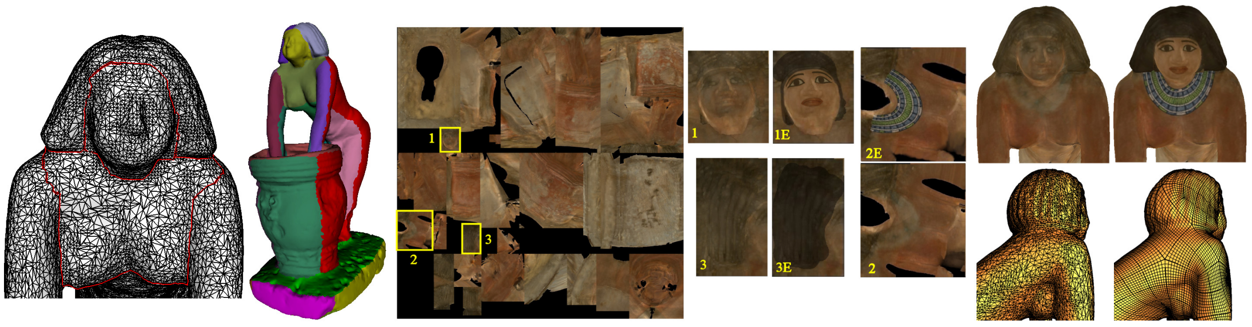


Figure 9. Left to right: constrained segmentation followed by image-based editing (edited regions are shown numbered before and after editing; editing results are marked "E"); before and after comparison for paint restoration; underlying mesh before and after quad parameterization (thick lines indicate patch boundaries).