EUROGRAPHICS 2017 / L. Barthe and B. Benes
(Guest Editors)

Volume 36 (2017), Number 2

Learning Detail Transfer based on Geometric Features

Sema Berkiten[1] Maciej Halber[1] Justin Solomon[2] Chongyang Ma[3] Hao Li[3,4,5] Szymon Rusinkiewicz[1]

[1]Princeton University [2]MIT [3]University of Southern California [4]USC Institute for Creative Technologies [5]Pinscreen

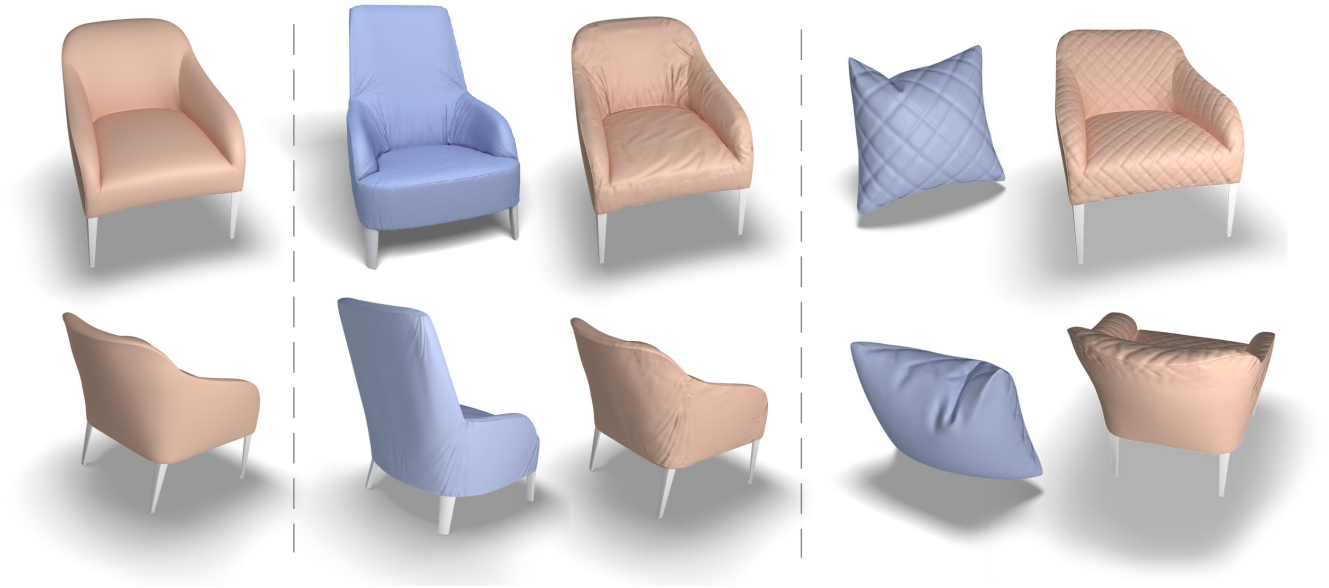


Figure 1: *Our algorithm learns combinations of geometric features that predict the spatial arrangement of details on surfaces.* ***Left:*** *Input (target) mesh without details.* ***Center/Right:*** *Details from each source mesh (blue) are synthesized on the target mesh (pink).*

Abstract

The visual richness of computer graphics applications is frequently limited by the difficulty of obtaining high-quality, detailed 3D models. This paper proposes a method for realistically transferring details (specifically, displacement maps) from existing high-quality 3D models to simple shapes that may be created with easy-to-learn modeling tools. Our key insight is to use metric learning to find a combination of geometric features that successfully predicts detail-map similarities on the source mesh; we use the learned feature combination to drive the detail transfer. The latter uses a variant of multi-resolution non-parametric texture synthesis, augmented by a high-frequency detail transfer step in texture space. We demonstrate that our technique can successfully transfer details among a variety of shapes including furniture and clothing.

1. Introduction

Increasingly customizable characters in video games, massive multi-user immersive virtual environments, and advances in 3D printing have all driven the demand for high-quality 3D models. It is time-consuming and expensive, however, to hand-model or scan 3D surfaces with realistic fine details. Furthermore, the number of high-quality 3D models freely available online or elsewhere is severely limited, forcing designers to generate models by hand or obtain them from repositories of low-resolution shapes. Even if a detailed shape is available in a desired *class*, edits that preserve fine-scale details are tedious at best. For this reason, most high-quality models continue to be generated by highly-trained 3D artists who sculpt all the details manually using modeling tools such as ZBrush [Pix15].

In this paper, we propose that the effort to create a single high-quality model may be amortized by transferring its details to a wide variety of easily-produced and widely-available low-resolution meshes. Specifically, we consider details that may be represented as displacement maps (so that their geometric extent is limited) and whose statistics are distinctive over different parts of the surface (so that they may be treated within the framework of texture analysis and synthesis). This encompasses surface features such as wrinkles, fabric patterns, wood grain, scratches, and cloth seams, which are critical to realism yet most typically missing in the low-

resolution meshes created by amateurs. However, while there is a rich literature on texture analysis, the chief difficulty lies in the fact that realistic surfaces contain many *different* kinds of details. We must therefore learn which parts of the source model to draw from when synthesizing details on every point of a target model.

The key novelty of our work is to use *metric learning* to find a combination of geometric features that best predicts which regions of the source mesh have similar details. The learned metric is then used to evaluate source-target similarity, which guides the texture synthesis. This builds upon the insight (also exploited in the work of Mertens et al. [MKC*06] and Lu et al. [LGG*07]) that the details on a surface are usually correlated with large-scale geometry.

It is important to note that our method does not attempt to find a one-to-one correspondence, or even a fuzzy correspondence [KLM*12], between the source and target meshes. Instead, we perform an analysis *entirely within the source mesh* to learn which geometric features are most important for predicting the similarity of local detail-texture statistics. One way of thinking about our system is to imagine that it solves an analogy problem (Figure 2): we learn predictive relationships M between a low-resolution source mesh S and its detail map D, and apply them to the target mesh S' to synthesize a detail map D'.

This approach lets us transfer details between models of significantly different size, complexity, and topology. For example, the center column in Figure 1 demonstrates that our algorithm can transfer details between similar shapes such as armchairs, while the right column illustrates that we can also successfully transfer the diamond pattern from a pillow to a chair. Moreover, notice that the diamond pattern occurs only on the front side of the pillow, and accordingly is transferred to only the front side of the chair. This is because our method learns that the presence of the diamond pattern is correlated with surface orientation. We demonstrate that the learned relationship between coarse geometry and texture detail is highly dependent on shape class and context, highlighting our method's contributions beyond existing surface correspondence and texture synthesis algorithms.

Contributions. The major new contributions of our work presented in this paper include:

- Learning the relationships between a wide range of geometric features and the statistics of a detail map expressed as a texture, via metric learning;
- Transferring surface details between different 3D shapes in a non-parametric texture synthesis framework;
- Combining coarse- and mid-scale synthesis over the surface with fine-scale synthesis in texture space, yielding an algorithm with the stability of the former and the quality and efficiency of the latter; and
- Releasing a database† of high-quality 3D models generated by artists, augmenting the currently-limited number of high-quality models available for research purpose.

† http://surfacedetails.cs.princeton.edu

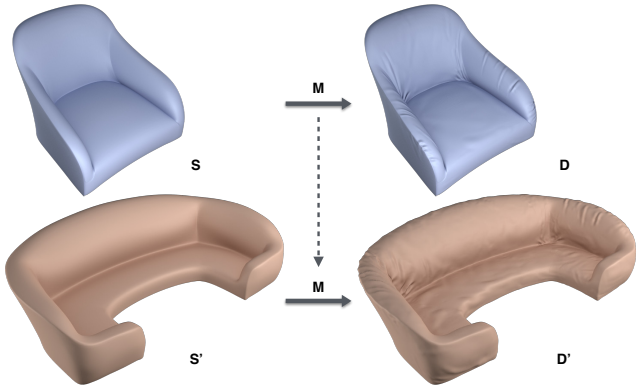


Figure 2: *Our algorithm learns a mapping M between a coarse source shape S and its detail map D, and uses the same mapping to synthesize a detail map D' for a target mesh S'.* ***Left:*** *the source and target models without details.* ***Right:*** *source and target meshes are rendered with their corresponding detail maps.*

2. Related Work

Texture Synthesis. Texture synthesis is a long-standing topic in the computer graphics literature; Wei et al. [WLKT09] present a comprehensive survey. The original study of texture synthesis was in the 2D image domain [HB95, EL99, WL00] and was later extended to synthesis of color patterns [Tur01, WL01] as well as synthesis of transparency and displacements over surfaces [YHBZ01]. More recent works in texture synthesis consider volumetric surface details [BIT04], transferring a geometric texture patch by using geometric images [LHGM05], mesh quilting [ZHW*06], using terrain as a height field [ZSTR07], feature-aligned texturing [XCOJ*09], material interpolation for appearance synthesis [DBP*15], and self-tuning optimization [KNL*15].

While most of the existing techniques are based on a Markov random field (MRF) assumption, i.e., that the texture pattern is both *local* and *stationary*, several methods have been proposed to handle non-stationary textures. Lu et al. [LGG*07] correlate the distributions of texture appearance to some simple context parameters, such as ambient occlusion and principal directions. Rosenberger et al. [RCOL09] automatically generate control maps for layered textures. More recently, Chen et al. [CFG*12] adapt the PatchMatch algorithm to 3D and transfers textures from one geometry to another with the help of some geometric features.

The previous work most relevant to our approach is that of Mertens et al. [MKC*06]. It uses canonical correlation analysis (CCA) to find correlations between RGB colors and geometric features, attempting to explain texture variations caused by processes such as dust accumulation and weathering. In contrast, our work employs a wider set of recently-developed geometric features along with the framework of metric learning, allowing meaningful correlations to be learned for coarser models and more complex geometry/texture relationships — see Figure 16 for a comparison. In addition, our method better handles structured textures such as wrinkles and sewing stitches, as compared to the highly-stochastic textures for which the work of Mertens et al. [MKC*06] is best suited.

Shape Correspondence. A key goal of our system is to establish which points on the source shape should be considered when synthesizing detail at each point on the target. The simplest version of this problem would be to compute a one-to-one correspondence between the source and target meshes, and a long line of research has been devoted to this problem [VKZHCO11]. Kim et al. [KLM*12] extend this line of work to consider *fuzzy* correspondences among a set of 3D models, with the goal of facilitating the exploration of shape collections. Solomon et al. [SPKS16] compute a fuzzy map from the source surface into the target by minimizing the distortion between their distance fields. However, most of existing techniques for shape correspondence require the shapes to have the same or similar global structure, which severely restricts their applicability to detail synthesis. Instead, our learned metric can be used to effectively compute a similarity score between *every* pair of points on the source and target shapes. This retains some notion of correspondence if the shapes are similar, while smoothly degrading if the shapes are dissimilar in overall structure.

Geometry Synthesis. One application we consider is the synthesis of wrinkle and crease patterns on cloth, furniture, and other shapes. Golovinskiy et al. [GMP*06] synthesize facial wrinkles assuming both input and training 3D face models are perfectly aligned. They segment these aligned faces into regions and calculate statistics on each of them to synthesize new wrinkles for each region separately. Wang et al. [WHRO10] synthesize wrinkles for cloth which closely fits the body, based on analysis of simulated examples. Rohmer et al. [RPC*10] generate dynamic wrinkles by using the stretch tensor of the coarse cloth animation as the guidance.

The broader problem of shape synthesis by example has received considerable attention in other forms. One line of research focuses on part-based synthesis of new models from shape repositories [KCKK12, XZCOC12, ALX*14]. Other methods leverage symmetry information to generate complex shapes from small examples [PMW*08, BWS10, HGM14]. Recently Ma et al. [MHS*14] use an analogy-driven framework to transfer style from an exemplar to a target model. In their framework, the source and the exemplar models need to have the same structure to build dense point-to-point correspondences. Furthermore, the input source and target shapes in their cases have to be of the same style to allow the computation of source-to-target analogy assembly. In our framework, instead of computing full correspondences, we transfer details from the source shape to the target shape via learning from geometric features, which works for more general input shapes.

3. Algorithm

In order to guide our texture synthesis algorithm, we need to know which areas on the source mesh should be considered for each point on the target. As hinted above, we dare not compute a full correspondence between source and target: that would leave our method fragile in the face of changes to gross shape, size, and topology. Instead, we argue that points on the target should draw from *geometrically similar* areas of the source. We can evaluate this similarity on the basis of a variety of geometric features, but this prompts another question: which features are most important? Our insight is that feature importance is not universal — in some situations the details might vary with curvature (such as the wrinkles on the armrest of the armchair at center in Figure 1), in others with normals (such as the diamond pattern on one side of the pillow at right in Figure 1), and in still other situations the variation might be well-predicted by one of the more recent feature descriptors such as the wave kernel signature (WKS) [ASC11]. So, we must learn feature importance for each *particular* source mesh and every *particular* detail texture.

This motivation gives rise to the pipeline in Figure 3. We begin with a source triangle mesh S and its detail map $D: S \to \mathbb{R}$, where values in D represent displacements from the base mesh S along surface normals — see Figure 4. For both S and the target mesh S', we use repeated remeshing to create lower-resolution versions.

Our key new step is to analyze which regions of D are self-similar and learn a metric M that predicts these similarities on the basis of geometric features G. The learned metric M is applied to target features G', allowing us to evaluate the similarity of points on the source and target. This, in turn, is used to guide a multi-resolution texture synthesis algorithm, which yields the target detail map D'. In a refinement step, we transfer the highest-frequency details from D directly to D', operating in texture space rather than on the mesh.

3.1. Input Preprocessing

Canonical Scale and Orientation. Our first preprocessing step is to scale each mesh consistently with the others, and orient it such that its semantically-meaningful "up" direction points along the positive y axis. While we perform this manually, we believe that automatic algorithms could perform nearly as well [FCODS08].

Mesh Hierarchy. We create a multi-scale mesh hierarchy for both the source and target meshes. We use an isometric remeshing algorithm proposed by Botsch and Kobbelt [BK04] to create multi-resolution mesh hierarchies for both the source and the target meshes. Specifically, we use the implementation provided by Möbius and Kobbelt [MK12] and create four-scale hierarchies in which the average edge length of each level is equal to half the average edge length in the previous level.

Tangent Frame. Our texture synthesis algorithm operates one patch at a time, and so we need to define the orientations of patches that will be used for search and synthesis (see Figure 5a-b). The first step is to define the orientation of a tangent frame at each vertex on the surface. We experimented with a variety of strategies to obtain tangent frames including using the gradient of an artist-provided uv map of texture coordinates, principal curvature directions, user-specified vector fields [Tur01], stripe pattern synthesis [KCPS15], and integrable PolyVector fields [DVPSH15]. However, all of these strategies in practice lead to inconsistent orientations between similar patches on the source and target meshes and led to poor results in our application. Exploring the problem of defining consistent tangent frames between meshes is an interesting avenue for future work, but for the purposes of this work we adopt a simple strategy that appears to work in most cases: we assume that the source and target models are consistently oriented and we project the constant vector $(1,0,0)$ onto the plane perpendicular to the surface normal n, using that as the first tangent vector u. The other vector v is defined as $n \times u$. While this results in some discontinuities, we have not found this to be a problem in practice.

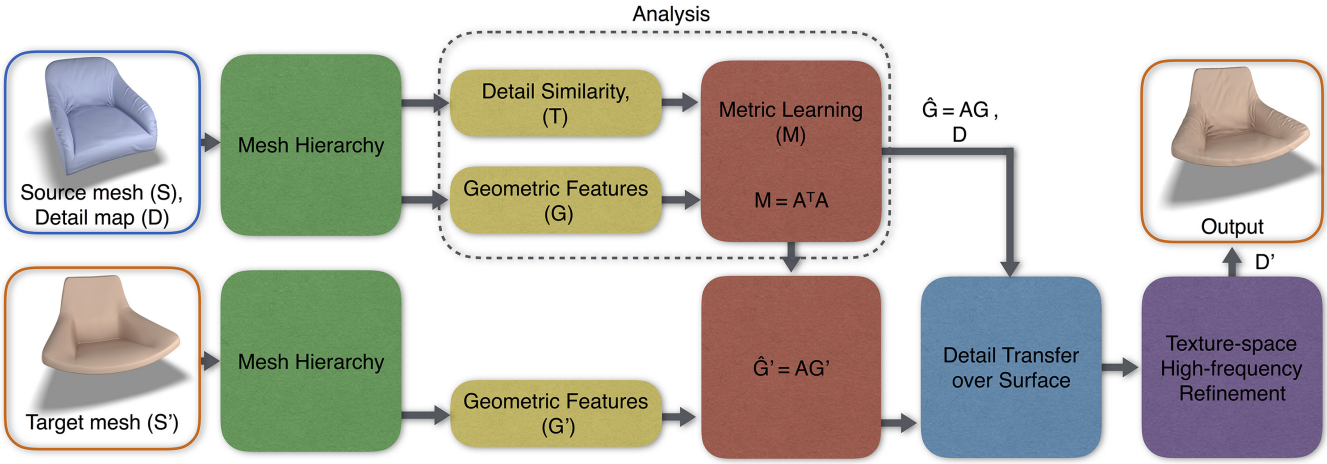
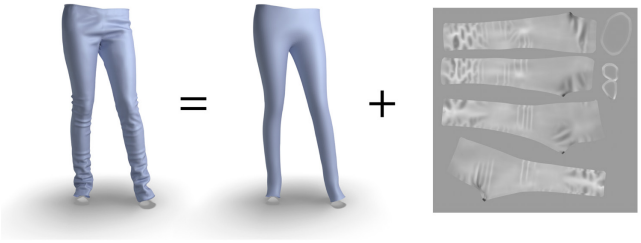


Figure 3: *Algorithm overview.*



Source **Low-poly Mesh** **Displacement map**

Figure 4: *A source mesh represented by a low-polygonal mesh and a displacement map — see supplemental material for more examples.*

3.2. Analysis

Our first step is to analyze how features in the detail map D correlate with the geometry of the source surface S. This proceeds in three steps: (1) determining which regions of D have similar textures; (2) computing a vector of geometric features G at each point on the surface; and (3) applying a metric learning algorithm to determine a linear transform of G that best matches the texture similarities. In the detail transfer stage (Section 3.3), this metric will be used to select patches on S whose details can be transferred to different points on S'.

3.2.1. Detail Features

In order to determine which regions of the surface have similar detail texture, we compute a "detail feature" for each vertex on the surface, characterizing the statistics of the texture in its neighborhood. We begin by mapping a 2D square patch on the tangent space around each vertex $\mathbf{v}$ onto the surface S via an approximate exponential map, as shown in Figure 5. We use a technique similar to that proposed by Melvaer and Reimers [MR12]. We choose this parameterization at each $\mathbf{v} \in S$ because in the smooth case it maps the tangent plane at $\mathbf{v}$ into S with minimal distortion about the center point [Hel78].

We then compute the texture statistics of each unfolded 2D patch.

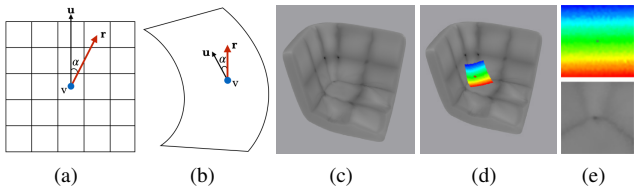


Figure 5: *Exponential mapping used to sample around vertex* $\mathbf{v}$. ***(a)*** *Tangent space of a surface* $\boldsymbol{S}$ *around vertex* $\mathbf{v}$ *with the tangent vector* u *and the relative location vector* r *of a pixel;* ***(b)*** *corresponding* u *and* r *vectors on the 3D surface* $\boldsymbol{S}$*;* ***(c)*** *example 3D model with detail map visualized as vertex colors;* ***(d)*** *exponential mapping around the corner of the seat;* ***(e)*** *corresponding 2D patches: a synthetic color mapping and actual sampled patch around the selected vertex.*

We use the low-dimensional feature vector proposed by Golovinskiy et al. [GMP*06], which consists of standard deviations of the histograms of steerable-pyramid filter bank outputs [HB95] at four orientations and four scales, along with a high-pass filter. Hence, the detail map around each vertex $\mathbf{v}_i$ is identified with a 17-dimensional feature t_i. Figure 6 shows the steerable pyramid outputs for the patch around the selected vertex marked in Figure 6b, along with the detail similarity between the selected point and all other points on the mesh. Histograms of the steerable pyramid are overlaid to demonstrate the varying standard deviation over different scales and orientations.

3.2.2. Geometric Features

We compute the following geometric features at each vertex of the source mesh:

- **Curvature (C):** The two principal curvatures, along with the mean curvature, Gaussian curvature, and L^2 norm of the two principal curvatures.
- **Height (H):** The y-coordinate of each vertex.
- **Normals (N):** The surface normal at each vertex. We smooth

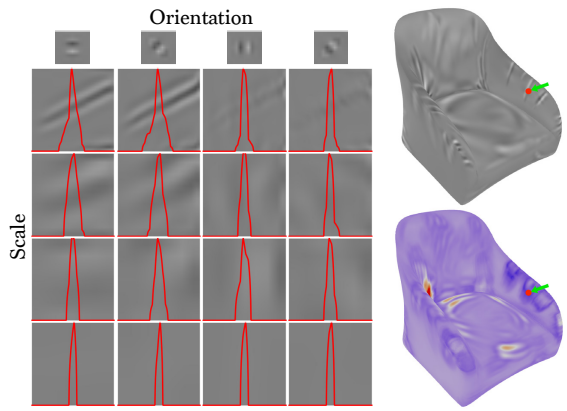


Figure 6: *Computation of detail features.* ***(a)*** *The steerable pyramid of a patch around the vertex highlighted in* ***b*** *for 4 scales (rows) and 4 orientations (columns), overlaid with their histograms;* ***(b)*** *a detail map with one point highlighted, together with its similarity (blue = similar, red = dissimilar) to all other vertices on the shape.*

them with a Gaussian kernel with standard deviation of four times the average edge length to get rid off noise in surface normals .

- **Shape diameter function (SDF):** A measurement of local object diameter, useful for separating thin and thick parts of the object [SSCO08].
- **Distance to segmentation boundary (Seg):** The geodesic distance from each vertex to the closest segmentation boundary obtained using SDF [SSCO08], normalized by the maximum distance within the each segment as described by Chen et al. [CSPF12].
- **Statistics on geodesic distances (Geo):** The mean, median, maximum, standard deviation, tenth percentile, and ninetieth percentile of geodesic distances from each vertex to all others [CSPF12].
- **Relative *xyz* coordinates (Box):** Each vertex's coordinates with respect to the center of the bounding box, normalized by the size of the corresponding dimension of the bounding box.
- **Heat kernel signature (HKS):** Measurements of the dissipation of heat from a point onto the rest of the shape over time [SOG09], sampled at 20 points in time.
- **Wave kernel signature (WKS):** The average probability to find a particle of a given energy at a given point [ASC11], sampled at 20 points on the frequency range.

Concatenating all of these features, we obtain a 60-dimensional geometric feature vector g_i at each vertex $\mathbf{v}_i$. Figure 7 visualizes each geometric feature for an armchair. A diverging color map [Mor09] is used, with blue mapped to the minimum value and red mapped to the maximum.

3.2.3. Metric Learning

Measuring distances or similarities between data is ubiquitous in machine learning, pattern recognition, and data mining, but it is generally difficult to tailor metrics for specific problems. This has led to the emergence of metric learning, which automatically adjusts a distance or similarity function using training data [Kul13, BHS15].

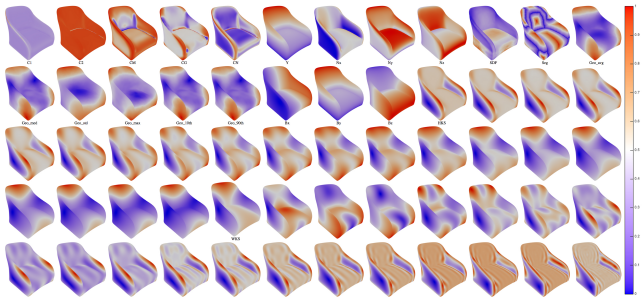


Figure 7: *Geometric features for an armchair. Each image visualizes one of the 60 geometric features, with minimum and maximum values of each feature mapped to blue and red, respectively.*

In this work, we use a supervised global distance metric learning method to find a transform to be applied to our geometric features, such that distances between transformed features are predictive of distances between statistics of surface details. Although the expressive power of a global metric is limited relative to localized alternatives, the resulting convex objective is easier to optimize and suffices for our application. Furthermore, computation of a single global metric is more resistant to over-fitting relative to nonlinear and local methods [BHS15].

We define G to be a geometric feature matrix, with entry G_{ij} corresponding to the i-th feature of the j-th vertex on the source mesh. Additionally, we define the detail similarity matrix T to have entries $T_{ij} = \langle t_i, t_j \rangle$, where t_i is the texture feature vector of the i-th vertex. Intuitively, assuming we subtract the mean texture features ahead of this step, we can think of T_{ij} as measuring the similarity of vertices i and j with respect to the texture features.

We wish to learn a metric on geometric features that mimics the metric on t_i. Formally, we can think of this problem as learning a distance function $d(\cdot,\cdot)$ between columns of G that imitates the distances encapsulated in T. For simplicity, we restrict ourselves to *Mahalanobis metrics* of the form

$$d(g_i, g_j) = (g_i - g_j)^\top M (g_i - g_j), \tag{1}$$

for some matrix $M \succeq 0$; the positive-definite constraint is needed for our metric to satisfy the triangle inequality. Since we create a mesh hierarchy on both target (on the low-resolution shape component) and source meshes with the same resolutions, meshes at the same levels will have similar geometric complexity (level-of-detail) and therefore are comparable by the Mahalanobis metric.

We propose learning M via the following optimization problem:

$$\begin{aligned} \min_M \quad & \|T - G^\top M G\|_{\text{Fro}}^2 + \lambda \|M\|_1 \\ \text{s.t.} \quad & M \succeq 0. \end{aligned} \tag{2}$$

We solve this convex problem using CVX [GB08]. The first objective term encourages the inner products of geometric features with respect to M to look similar to dot products of the texture features. The second objective term, modulated by a regularization parameter $\lambda \geq 0$, promotes *sparsity* of M [BHS15], in this case reflecting the intuition that most features are uncorrelated and hence should have entry $M_{ij} = 0$. The examples in this paper use $\lambda = 0.1$, which worked effectively for all of our experiments. This formulation looks

for a global metric as opposed to a localized one. In this setting of transferring the details for objects of the same class, the use of a global metric enables us to find a correlation between general, global shape and high frequency details. For example, the wrinkle patterns are expected to be different on a seat of a chair and its back, while we would expect these areas to be locally similar, especially when represented as simplified shape S.

We experimented with a variety of metric learning alternatives, most notably a formulation minimizing differences of L^2 distances instead of inner products. While still convex, the resulting optimization problem is considerably more difficult to solve because the quadratic terms become dense in the optimization variables. Our formulation yielded similar results in far less computation time.

Comparison to Context-Aware Textures. Lu et al. [LGG*07] propose a method for capturing time-varying textures and context parameters (a few geometric features) to control texture transfer. They first compute a diffusion map to represent the time-varying texture with a set of orthogonal functions on vertices. They then compute a correlation by taking the inner product of the diffusion map and context parameters (defined by ambient occlusion, signed mean curvature, principle curvature directions, normals in optimal and non-optimal surface directions). They only use these correlations to select the most important context parameter, since the numerical values of these correlations are not meaningful. To transfer time-varying textures, they provide a user interface in which the user can control the transfer by selecting a context parameter. We found that this approach is limited to mostly stochastic textures resulting from phenomena such as rusting, weathering, or chemical agents, for which the color variation can be simply explained with a single geometric feature such as ambient occlusion. We provide an empirical comparison later in the paper (Figure 16b).

Comparison to CCA. Mertens et al. [MKC*06] propose guiding geometric texture synthesis using canonical correlation analysis (CCA). Their work projects geometric features onto three directions that are predictive of RGB triplets, computed using a greedy sequence of eigenvector computations [HSST04]. We found that this low-dimensional projection and use of color rather than texture descriptors is not as good as metric learning at taking advantage of the rich relationships between shape and texture. Furthermore, our formulation (2) is globally optimal, allows for sparsity-based regularization, and is specifically tuned to the specific task of computing *distances* between features. We provide an empirical comparison to CCA later in the paper (Figure 16c).

3.3. Detail Transfer

Transferring details from one surface to another can be thought of as a texture synthesis problem. In the literature, there are various works on texture synthesis in both 2D and 3D. Although 2D domains have regular grid structures that enable fast processing, projecting back into 3D introduces distortion. On the other hand, achieving high resolution in 3D requires higher computational complexity than in 2D. To leverage the advantages of both methods, we propose a hybrid approach: first transferring details between 3D surfaces, then post-processing the synthesized detail maps, in texture space, to enhance fine details.

First, all the geometric features described in the previous section are computed for the full multi-resolution hierarchy of the target mesh S'. At this point, we have a new *Mahalanobis* distance function to compute geometric similarity between points on the source and target meshes, as follows:

$$d(g_i, g'_j) = (g_i - g'_j)^\top M (g_i - g'_j) \tag{3}$$

$$= \|Ag_i - Ag'_j\|_2^2, \tag{4}$$

where the positive semidefinite matrix M is factored as $M = A^\top A$ using Cholesky decomposition. We apply the transformation matrix A to all geometric feature vectors of both the source and the target meshes to simplify subsequent processing.

Algorithm 1 Detail transfer from source to target.

function TRANSFER-TEXTURE(H_{src}, H_{tgt}, n)
  H_{src}: mesh hierarchy for the source shape
  H_{tgt}: mesh hierarchy for the target shape
  n: patch size

  compute sweep order on H_{tgt}
  for each level in H_{tgt}, coarsest to finest:
    up-sample vertex colors
    SYNTHESIZE-LEVEL(H_{src}, H_{tgt}, n)
    reverse sweep order
    SYNTHESIZE-LEVEL(H_{src}, H_{tgt}, n)
    $n \leftarrow 2 * n + 1$

Algorithm 2 Detail transfer in a single level.

function SYNTHESIZE-LEVEL(H_{src}, H_{tgt}, n)
  sample patches around evenly-distributed source vertices
  for each target vertex, in sweep order:
    if displacement weight $w_{disp} > 0.5$ **then**
      skip the vertex
    sample a patch around the vertex
    match features to find candidate source patches
    match patches to find best candidate
    blend best patch into target mesh

3.3.1. Detail Transfer over Surfaces

We adapt a variant of non-parametric texture synthesis (which has been explored for both images [EL99] and 3D surfaces [Tur01]) to transfer details from one surface to another. We use a hierarchical method in which the synthesis progresses from the coarsest to finest level in a mesh hierarchy. We also propose using a sweep order based on the number of good candidates per vertex. In other words, vertices with fewer good candidate source patches should be synthesized first, because they have fewer degrees of freedom. The synthesis is guided by a two-step matching phase where both geometric features and already-synthesized details are considered. We also choose to synthesize a patch at a time, in preference to per-vertex synthesis, to both accelerate the process and preserve local coherency.

Our detail transfer algorithm is presented as Algorithms 1 and 2, above. The main blocks of the algorithm are as follows:

- **Sweep order:** The number of good correspondences per target vertex is computed by thresholding the (transformed) distances $d(g_i, g_j)$ from that vertex to all vertices on the source mesh. A few vertices with the fewest good correspondences are selected as initial *seed vertices*. Those seed vertices are then used to determine the sweep order, based on the geodesic distance from the seed vertices to others. However, to reduce order dependence, we reverse the sweep order at each iteration.
- **Sampling:** At each level in the hierarchy, the source mesh is sampled evenly and exponential surface patches around those samples are stored along with their embedded geometric features, as explained in Section 3.2.1.
- **Matching:** A two-step matching algorithm is used to find the best-fitting patch from the source mesh to the target mesh. First, a set of candidate patches are found from the source mesh based on geometric feature similarity. Second, among those candidates the one which is the most similar to the already-synthesized part of the detail map on the target mesh is selected. In both steps, a translational refinement is applied when searching for the most similar feature or displacement patch.
- **Blending:** The best candidate patch is copied onto the target mesh and blended seamlessly by using a weighted average controlled by two weights, w_{loc} and w_{val}, inversely proportional to: (i) the distance from the center of the patch, x_c, to the pixel location, x_i, and (ii) the difference between the new, d_{new}, and already synthesized, d_{curr}, displacement values, respectively. In particular, the per-vertex displacement weights w_{disp} are updated as follows:

$$w_{disp} \leftarrow w_{disp} + w_{loc}\, w_{val} \tag{5}$$

$$w_{loc} = f\left(\|x_c - x_i\| / (n/\sqrt{2})\right) \tag{6}$$

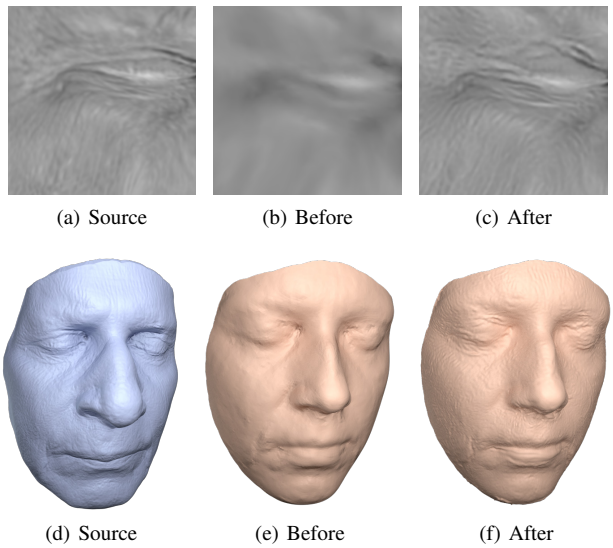
$$w_{val} = f\left(\|d_{curr} - d_{new}\| / d_{max}\right), \tag{7}$$

where n is the patch size, $f(x) = 2x^3 - 3x^2 + 1$ as in [Tur01], and $d_{max} = 1$, assuming the displacement values are in $[0,1]$.

3.3.2. Texture-space High-frequency Detail Transfer

Detail transfer over surfaces is limited by the finest mesh resolution in the mesh hierarchy. In order to transfer details with higher frequency than the mesh sampling, we propose a hybrid synthesis approach: detail transfer over surfaces (Section 3.3.1) followed by a refinement step in texture space (below).

The texture-space refinement begins by rasterizing the per-vertex displacements synthesized on the target mesh into a target texture map. We then consider small patches p_i from the source texture, centered at each vertex, along with their corresponding locations in the target texture. We warp each p_i to match the uv texture parameterization of the target, and further refine its translation to match the target texture as well as possible. Finally, we transfer only the high frequencies from p_i to the target, where the cutoff frequency is set according to the median edge length in uv texture space. The results before and after this refinement step are shown in Figure 8.



(a) Source (b) Before (c) After

(d) Source (e) Before (f) After

Figure 8: *High-frequency refinement of transferred texture.* ***(a)*** *Close-up image from a source detail map.* ***(b)*** *Before the refinement step, the transferred texture is blurrier than the original.* ***(c)*** *After texture-space refinement, high-frequency details are restored.* ***(d)*** *Source model with details,* ***(e), (f)*** *target model with details before and after the refinement step.*

4. Results

Wrinkles on Clothing. In Figure 9, we demonstrate that our framework can be used to add a realistic appearance of fabric to clothing models. We not only transfer details between shapes in the same category, such as pants, but also transfer details from a full-body scan to a single piece of clothing. In this case, the automatically-learned metric is able to restrict the textures that are selected to ones appropriate to the target. We believe that this framework has a wide variety of applications, perhaps extending to synthesizing details in areas with poor visibility on a 3D scanned model.

In all, we generated a significant number of high-quality 3D clothing models from 6 source models and 11 target models. See the supplementary materials for more results.

Style Transfer on Furniture. Figure 1 and 16e show some of the results of transferring details including wrinkles, seams, and fabric patterns among models of upholstered furniture. In each case, the source mesh is shown in blue, with subsequent pink models being the targets.

The results show that we are able to capture the semantically-meaningful distribution of details on the surface. For example, similar kinds of wrinkles and seams tend to appear in corresponding places on the surface, even in the presence of large differences in overall shape. Note also that our non-parametric texture synthesis algorithm is able to handle patterns ranging from structured to completely stochastic.

We also demonstrate in Figure 10 that our algorithm can be used on low-polygonal shape collections to convert them into high-quality models from little data (i.e. from a single armchair to four sofas with different sizes as shown in the figure) and produce similar-looking

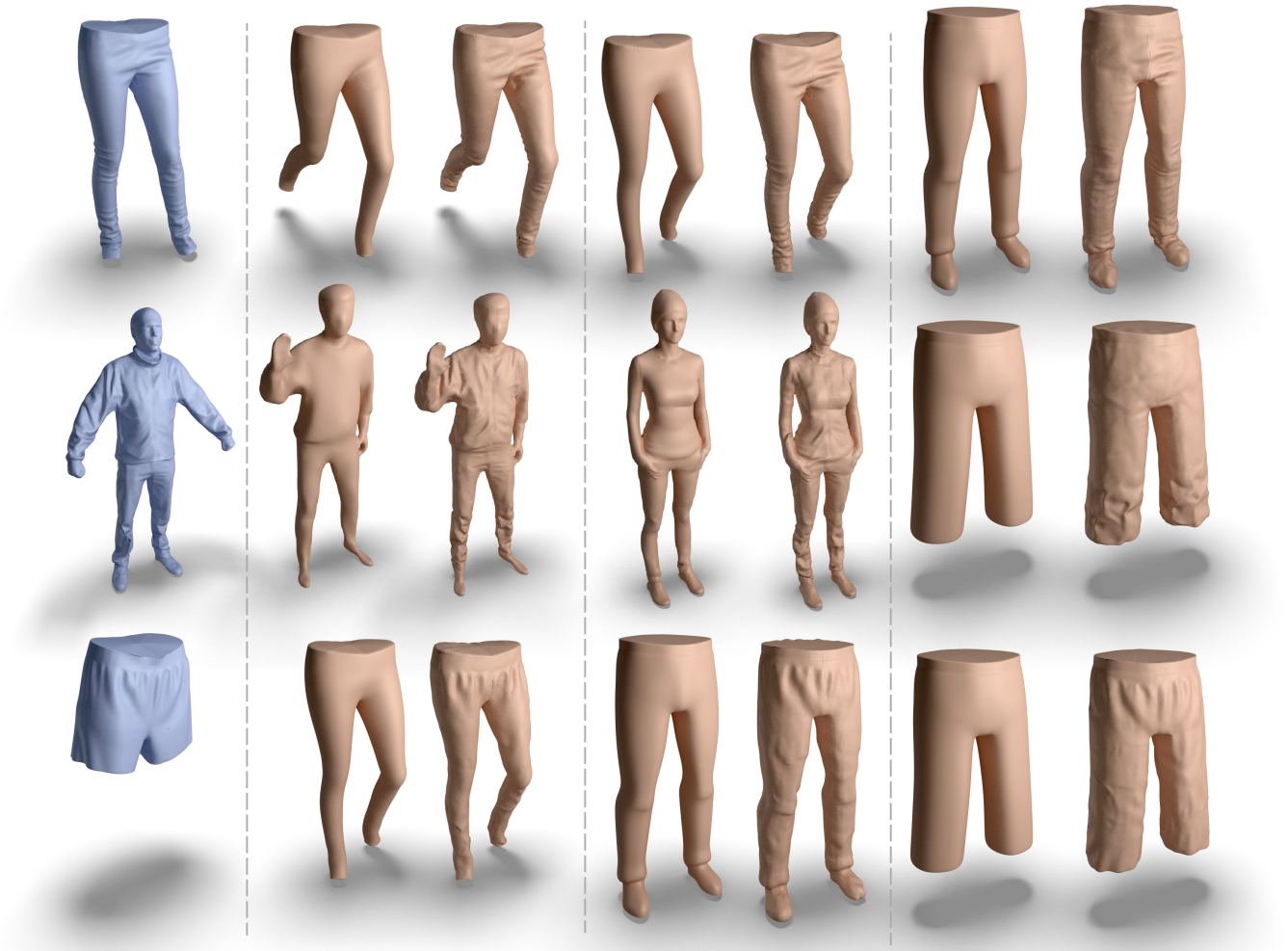


Figure 9: *Detail transfer for clothing models. For each source mesh (left column), we transfer details to three different target meshes.*

but not identical details which is important for a realistic appearance of details like wrinkles.

In the furniture category, 17 artist-designed source models were used to create a number of high-quality target models. More results are presented in the supplementary materials.

Other Transfer Results. We also demonstrate a few results on shapes from other classes. In Figure 11, we use the armadillo as a source mesh and transfer the details to a bunny and a pair of pants. Our algorithm transfers the square pattern from the armadillo's shell to the bunny's back, and the bumpy pattern on the armadillo's legs to the pants. Additionally, we apply our algorithm on few faces from the Merl face database [GMP*06], as shown in Figure 12 – see the supplemental material for further examples.

5. Experiments and Comparisons

Comparison to Other Algorithms. We compare our algorithm to several alternatives:

1. **No Metric Learning:** We disable the metric learning part of our algorithm, giving equal weight to each geometric feature. Specifically, we subtract the mean of each feature, and then divide by its standard deviation across the shape. This method can also be understood as a version of MeshMatch, proposed by Chen et al. [CFG*12], where the geometric features are equally weighted. As shown in Figure 16a, when the algorithm relies on all the geometric features equally and the source and target meshes have dissimilar local geometry, it fails to transfer meaningful details. For example, as shown in the middle column, it transfers wrinkles from the corner of the source armchair to the middle of the couch.
2. **Lu et al. [LGG*07]:** Although the method proposed by Lu et al. [LGG*07] focuses on time-varying textures, we adapt their idea of context-aware textures to compare to our method. Specifically, we compute the eigenvalue decomposition of the learned metric M and use the geometric feature (their "context parameter") corresponding to the highest eigenvalue. In other words, we transfer details using guidance from a single geometric feature to explain the correlation between geometry and details of the source mesh. In Figure 16b, the rightmost column shows that

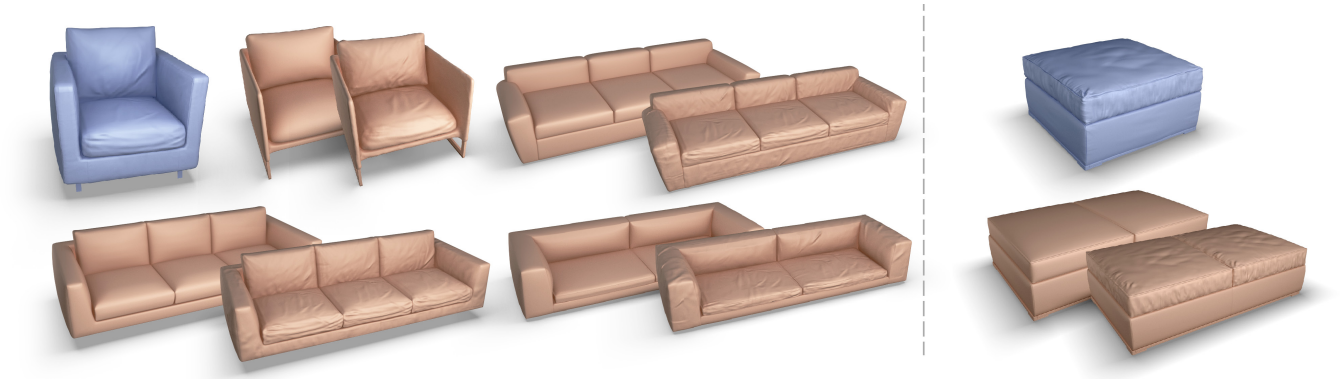


Figure 10: *Details from the source meshes shown in blue are transferred to multiple target models to demonstrate that our algorithm produces similar but not identical details.*

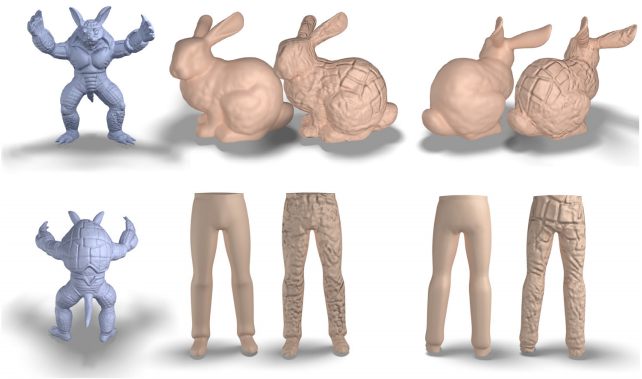


Figure 11: *Detail transfer from Armadillo to Bunny and pants.*

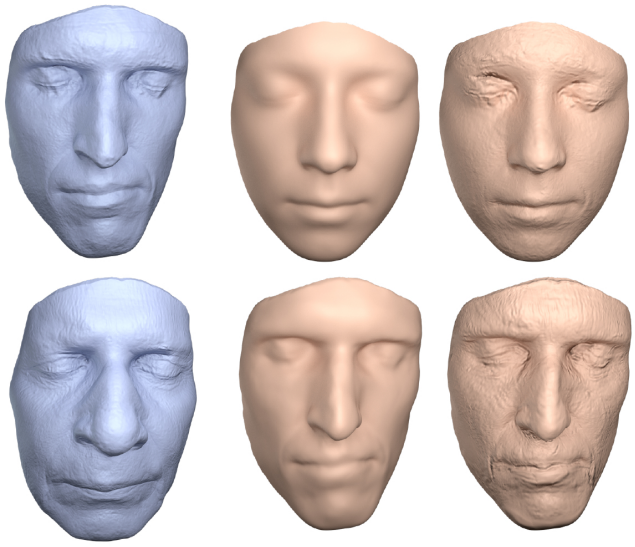


Figure 12: *Detail transfer on faces from Merl Database from the source models in blue to the target models in pink.*

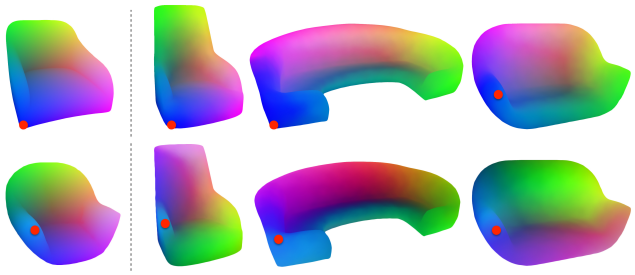


Figure 13: *Full correspondences produced by using Solomon et al. [SPKS16]. Red points are set manually as constraints. False coloring is used to visualize correspondences between the source and target meshes. The same correspondences are used in Figure 16d for comparison.*

this technique works if the details are highly correlated with a single geometric feature. In this example, the highest eigenvalue corresponds to mean curvature, which explains the distribution of the wrinkles on the surface very well. On the other hand, the first and second columns show that this method is not sophisticated enough to discover details whose relationship with the underlying geometry is not captured by a single descriptor.

3. **Canonical Correlation Analysis (CCA):** Previous work by Mertens et al. [MKC*06] uses CCA to find correlations between RGB and geometric features (a smaller set than is used in this paper). To provide fair comparison to metric learning, we use CCA to find the relationship between the detail-texture descriptors described in Section 3.2.1 and the geometric features listed in Section 3.2.2. We then use this correlation, instead of metric learning, to guide detail transfer. As shown in Figure 16c, CCA fails to capture a meaningful relationship between the details and the geometric features and results in poor guidance while our results in Figure 16e significantly differs from and outperforms CCA. For example, CCA cannot capture the locality of the details on the concavity between the couch's seat and backrest shown in the middle close-up of Figure 16c, while our method clearly does.
4. **Mesh correspondence:** One could also imagine using a mesh

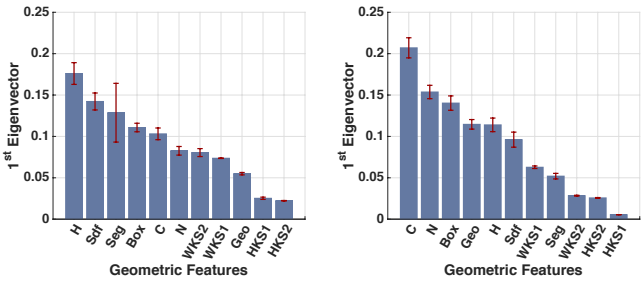
correspondence algorithm as guidance for texture transfer. Figure 16d illustrates the advantages of metric learning over this more restrictive pipeline. In this experiment, we use the method of Solomon et al. [SPKS16] to compute a probabilistic map from the source surface into the target while minimizing geodesic distortion (regularizer $\alpha = 0.005$); we mark a single point as a constraint. We then use the 10 highest-ranking matches for each target point to guide texture synthesis. Algorithms like [SPKS16] minimize distortion while seeking bijective maps. As illustrated in the results of this test, these properties can be problematic for texture synthesis. Textures may be better aligned with features captured by our metric, even if this induces stretch of the source onto the target. Furthermore, the search for a bijective mapping is subject to local minima. Figure 13 shows some instances in which our one-point constraint led to smooth but counterintuitive maps; these maps fail to capture critical semantic relationships thanks to the geometric variability of the models.

We also experimented with turning off the geometric feature matching altogether. However, with no constraint on which regions of the source mesh should be used at different locations on the target, the texture synthesis algorithm quickly "got stuck" by gravitating towards a single texture, most typically the one with lowest variation. This is because the matching part of the algorithm tries to minimize the difference between adjacent detail patches, and completely flat patches minimize this cost function most effectively.

Significance of Geometric Features. To demonstrate the significance of different geometric features used for the detail transfer, we learn the relationships between the surface details and the geometric features, as explained in Section 3.2.3, for models in two different categories: (i) clothing, with six models, and (ii) furniture, with 17 models. We plot the average of the first eigenvectors (i.e., corresponding to the largest eigenvalues) of matrices M for both categories in Figure 14. To make the graphs more compact, related geometric features are grouped into one; for instance, the "curvature" data point (C) represents the total significance of all the features derived from surface curvatures, as listed in Section 3.2.2. The only exceptions are the Heat Kernel Signatures (HKS) and Wave Kernel Signatures (WKS), which are partitioned into low-frequency (HKS1 and WKS1) and high-frequency (HKS2 and WKS2) components. The error bars on each data point represent variance of the corresponding class of features across models within a category.

We observe that the variation in the surface details for clothing is highly dependent on the height and SDF, while curvatures and surface normals are more important for furniture. On the other hand, the metric learning finds HKS features to be the least important for both categories. This can be explained by the presence of WKS features, since WKS is known to be more robust and distinctive compared to HKS.

Because meshes can vary in which geometric features are most strongly expressed, we also experimented with including the target mesh geometry into the metric learning — essentially, ensuring that the learned metric on the source mesh only includes features that are present and important on the target. To do this, we first find the principal components of the 60-dimensional geometric features of the target mesh, then project the source geometric features onto the



(a) Clothing (b) Furniture

Figure 14: *Significance of different geometric features, for* ***(a)*** *clothing and* ***(b)*** *furniture models. The y axis shows the average values of the first eigenvectors of learned matrices M, with the error bars in red representing variation of feature significance among the models in a shape category. Categories of geometric features are grouped along the x axis.*

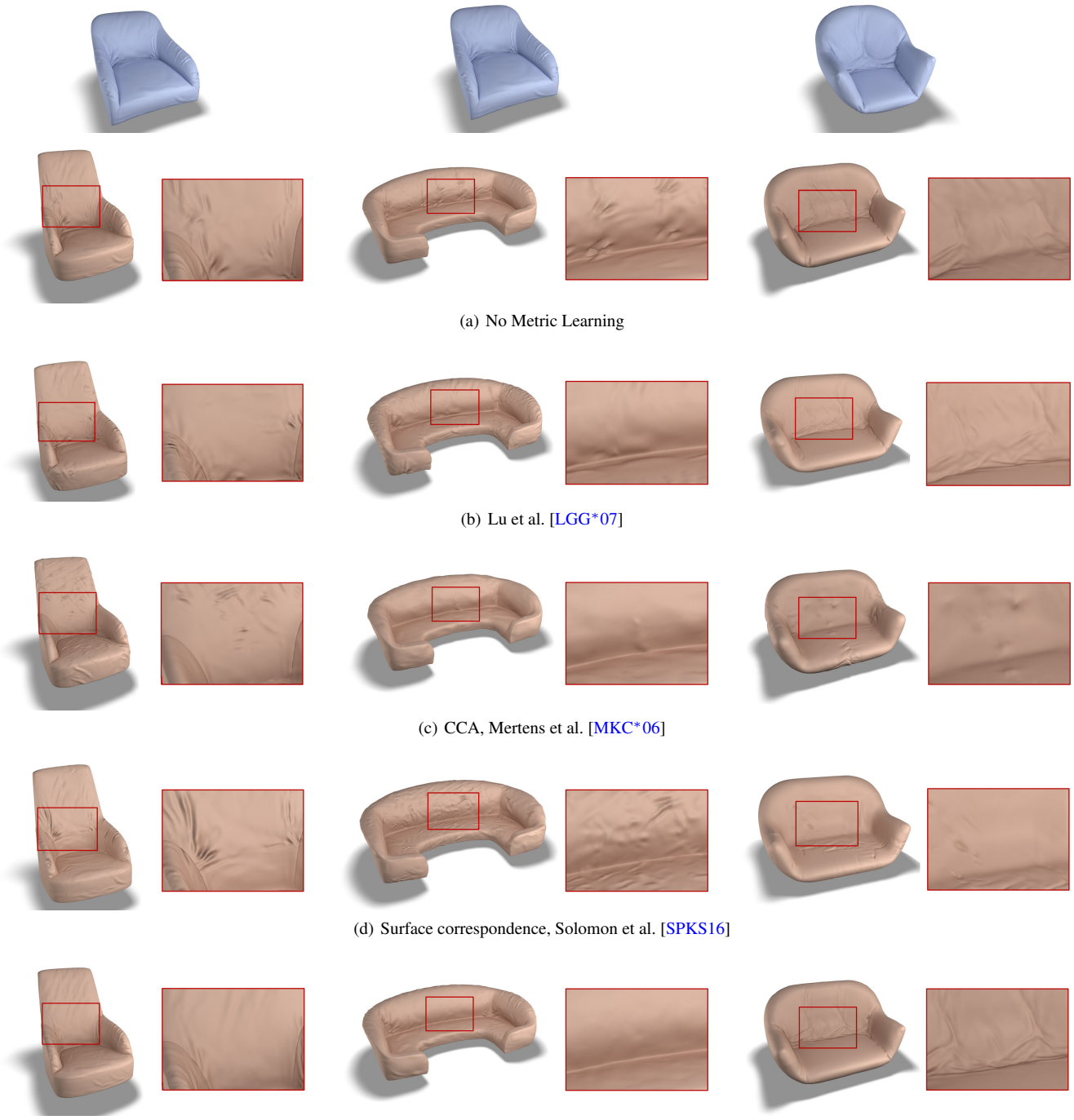
(a) (b) (c)

Figure 15: *Experiment evaluating the stability of detail transfer.* ***(a)*** *Source mesh;* ***(b)*** *target mesh, with details synthesized from the source mesh;* ***(c)*** *source mesh, with details transferred* ***back*** *from the mesh in (b).*

first k of those principal components; we experimented with using $k = \{5, 15, 30, 45, 60\}$. We use these projected geometric features, instead of the original ones, in the metric learning and the detail transfer. While we believe that there may be situations in which this yields a benefit, we did not observe significant improvements by adding this step. A more thorough exploration of using target feature distribution to guide the metric learning would be an interesting avenue for future work.

Stability of Detail Transfer. Figure 15 shows the result of an experiment intended to evaluate how well our learning step can match the distribution of textures across a model. Specifically, we first transfer details from the model in **a** to the one in **b**. Then, we use the synthesized model in **b** as the source mesh and transfer details from **b** back to the model in **a** (ignoring its original detail map), yielding the result in **c**. We observe that our framework is stable enough to produce a plausible result (although, as with any texture-synthesis result, we would not expect perfect agreement).

Performance and Statistics. The processing time of our unoptimized implementation of the algorithm varies depending on the size of the source and target meshes. We observe that smaller meshes such as the pants example (with the lowest resolution of $1.1k$ faces and the highest resolution of $70k$ faces in the mesh hierarchy) take a few minutes while larger meshes such as long couches (with the lowest resolution of $6k$ faces and the highest resolution of $400k$



(a) No Metric Learning

(b) Lu et al. [LGG*07]

(c) CCA, Mertens et al. [MKC*06]

(d) Surface correspondence, Solomon et al. [SPKS16]

(e) Our results

Figure 16: *Comparisons to several algorithms. Source meshes are shown at the top row in blue and the results are shown in pink.*

faces in the mesh hierarchy) may take around 30 minutes on an Intel Quad Core i7-4870HQ 2.5 GHz, 16 GB DDR3L, laptop computer.

6. Conclusions

This work presents an algorithm to transfer surface details from one shape to another in a non-parametric texture synthesis framework, guided by learned relationships between the surface details and the geometric features of the source shape. We showed that there is no one global set of geometric features to guide the surface detail transfer across different shape categories so a metric learning algorithm is needed to compute the appropriate weightings of different geometric features for each source shape and detail map.

Discussion. The success of our method depends on finding a meaningful correlation between geometric features and the distribution of details across a surface. This is most easily accomplished when the source and targets are in the same class, and indeed this was the case for which our algorithm was designed. When the source and target differ more drastically, we make the following observations:

- If the source has no regions closely matching the target, our algorithm will still find the *closest* match. As with all data-driven methods, a lack of training data can lead to suboptimal results.
- Because we use a rich feature set, including global features, even points with identical local geometry, such as the two armrests of an armchair, will have distinct features. This allows our algorithm to learn whether these regions should have similar or different textures.
- If two points have similar texture but different geometry, our algorithm will learn that similar geometric features are *not* predictive of texture (though, as mentioned above, "global" features may differ).
- If two objects come from completely different classes, our method may still produce intuitive results, such as in Figure 1, right, or Figure 11, bottom. In other situations, such as armadillo-to-armchair detail transfer, however, the result will not be meaningful.

Limitations and Future Work. One of the limitations of the detail transfer is the dependence on the tangent frame. Since (to our knowledge) there is no prior work that creates consistent tangent fields between two different 3D shapes, we rely on their initial alignment and the projection of a single global direction. Although we could search over different orientations during synthesis, this would result in higher computational complexity and inconsistently (and unnaturally) oriented details such as wrinkles. As future work, we would like to explore how to generate consistent tangent frames between shapes for texture synthesis over surfaces.

As with all MRF-based texture synthesis techniques, our algorithm performs less than ideally for highly-structured patterns. Specifically, neither the self-similarity of the detail map nor the geometric features that are used to transfer detail are precise enough to represent highly-structured repetitive patterns and exactly where they should appear on the surface. Exploring more sophisticated texture descriptors and geometric features tailored to structured patterns would be an interesting future direction.

In this work, we focus on transferring details from a single source mesh (to arbitrary targets), because of the limited number of high-quality 3D models available as input. Even though our framework can be used to extend the available collections of high-quality meshes, the resulting styles will still be limited to those present in the original source collection. An interesting avenue would be to explore interactive tools for novice users to create high-quality mesh detail maps quickly, possibly exploiting inputs such as photographs. Once there are more high-quality models available, we would like to explore transferring details from *collections* of shapes, instead of being limited to a single source mesh. Furthermore, while we believe that a deep learning approach to our problem is an interesting research direction, the need for large amounts of training data and memory has steered us away from such endeavor. Although it is possible to use pre-trained weights to fine tune a specific 2D problem with limited data, such architectures do not exist for 3D shapes yet.

While in this work we focus on small surface details, which are conveniently represented by displacement maps, recent work in 3D shape analysis has begun to focus on the understanding and transfer of larger-scale mesh "style" [MHS*14, LKS15, LHLF15, WSH*16]. We believe that our work may form part of a toolbox for analyzing the expression of style across different frequencies in 3D models, and it would be an interesting direction to explore unified solutions for transferring geometric elements of all scales between meshes.

Acknowledgements

We thank anonymous reviewers for their valuable comments and suggestions. We also thank Matt Furniss, Lingyu Wei, Tianye Li, and Artec Group for the models. This research is supported in part by NSF grants IIS-1421435 and IIS-1012147, NSF grant MSPRF-1502435, Adobe, Oculus & Facebook, Huawei, the Google Faculty Research Award, the Okawa Foundation Research Grant, the Office of Naval Research (ONR) / U.S. Navy grant N00014-15-1-2639, the Office of the Director of National Intelligence (ODNI) and Intelligence Advanced Research Projects Activity (IARPA) grant 2014-14071600010, and the U.S. Army Research Laboratory (ARL) grant W911NF-14-D- 0005. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of ODNI, IARPA, ARL, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purpose notwithstanding any copyright annotation thereon.

References

[ALX*14] ALHASHIM I., LI H., XU K., CAO J., MA R., ZHANG H.: Topology-varying 3D shape creation via structural blending. *ACM Trans. Graph. 33*, 4 (2014), 158:1–158:10. 3

[ASC11] AUBRY M., SCHLICKEWEI U., CREMERS D.: The wave kernel signature: A quantum mechanical approach to shape analysis. In *Proc. ICCV Workshops* (2011), pp. 1626–1633. 3, 5

[BHS15] BELLET A., HABRARD A., SEBBAN M.: *Metric Learning*. Morgan & Claypool, 2015. 5

[BIT04] BHAT P., INGRAM S., TURK G.: Geometric texture synthesis by example. In *Proc. SGP* (2004), pp. 41–44. 2

[BK04] BOTSCH M., KOBBELT L.: A remeshing approach to multiresolution modeling. In *Proc. SGP* (2004), pp. 185–192. 3

[BWS10] BOKELOH M., WAND M., SEIDEL H.-P.: A connection between partial symmetry and inverse procedural modeling. *ACM Trans. Graph. 29*, 4 (2010), 104. 3

[CFG*12] CHEN X., FUNKHOUSER T., GOLDMAN D. B., , SHECHTMAN E.: Non-parametric texture transfer using MeshMatch. *Adobe Technical Report 2012-2* (2012). 2, 8

[CSPF12] CHEN X., SAPAROV A., PANG B., FUNKHOUSER T.: Schelling points on 3D surface meshes. *ACM Transactions on Graphics (Proc. SIGGRAPH)* (2012). 5

[DBP*15] DIAMANTI O., BARNES C., PARIS S., SHECHTMAN E., SORKINE-HORNUNG O.: Synthesis of complex image appearance from limited exemplars. *ACM Trans. Graph. 34*, 2 (2015), 22:1–22:14. 2

[DVPSH15] DIAMANTI O., VAXMAN A., PANOZZO D., SORKINE-HORNUNG O.: Integrable polyvector fields. *ACM Trans. Graph. 34*, 4 (July 2015), 38:1–38:12. 3

[EL99] EFROS A. A., LEUNG T. K.: Texture synthesis by non-parametric sampling. In *Proc. ICCV* (Corfu, Greece, 1999), pp. 1033–1038. 2, 6

[FCODS08] FU H., COHEN-OR D., DROR G., SHEFFER A.: Upright orientation of man-made objects. *ACM Trans. Graph. 27*, 3 (2008), 42:1–42:7. 3

[GB08] GRANT M., BOYD S.: Graph implementations for nonsmooth convex programs. In *Recent Advances in Learning and Control*, Blondel V., Boyd S., Kimura H., (Eds.). 2008, pp. 95–110. 5

[GB14] GRANT M., BOYD S.: CVX: Matlab software for disciplined convex programming, version 2.1. http://cvxr.com/cvx, 2014.

[GMP*06] GOLOVINSKIY A., MATUSIK W., PFISTER H., RUSINKIEWICZ S., FUNKHOUSER T.: A statistical model for synthesis of detailed facial geometry. *ACM Trans. Graph. 25*, 3 (2006), 1025–1034. 3, 4, 8

[HB95] HEEGER D. J., BERGEN J. R.: Pyramid-based texture analysis/synthesis. In *Proc. ACM SIGGRAPH* (1995), pp. 229–238. 2, 4

[Hel78] HELGASON S.: *Differential geometry, Lie groups, and symmetric spaces*. Academic Press, 1978. 4

[HGM14] HUANG Q., GUIBAS L. J., MITRA N. J.: Near-regular structure discovery using linear programming. *ACM Trans. Graph. 33*, 3 (2014), 23. 3

[HSST04] HARDOON D. R., SZEDMAK S., SHAWE-TAYLOR J.: Canonical correlation analysis: An overview with application to learning methods. *Neural computation 16*, 12 (2004), 2639–2664. 6

[KCKK12] KALOGERAKIS E., CHAUDHURI S., KOLLER D., KOLTUN V.: A probabilistic model for component-based shape synthesis. *ACM Trans. Graph. 31*, 4 (2012), 55. 3

[KCPS15] KNÖPPEL F., CRANE K., PINKALL U., SCHRÖDER P.: Stripe patterns on surfaces. *ACM Trans. Graph. 34*, 4 (July 2015), 39:1–39:11. 3

[KLM*12] KIM V. G., LI W., MITRA N., DIVERDI S., FUNKHOUSER T.: Exploring collections of 3D models using fuzzy correspondences. *ACM Trans. Graph. 31*, 4 (2012), 54:1–54:11. 2, 3

[KNL*15] KASPAR A., NEUBERT B., LISCHINSKI D., PAULY M., KOPF J.: Self tuning texture optimization. *Computer Graphics Forum 34*, 2 (2015), 349–359. 2

[Kul13] KULIS B.: Metric learning: A survey. *Foundations and Trends in Machine Learning 5*, 4 (2013), 287–364. 5

[LGG*07] LU J., GEORGHIADES A. S., GLASER A., WU H., WEI L.-Y., GUO B., DORSEY J., RUSHMEIER H.: Context-aware textures. *ACM Trans. Graph. 26*, 1 (2007). 2, 6, 8, 11

[LHGM05] LAI Y.-K., HU S.-M., GU D. X., MARTIN R. R.: Geometric texture synthesis and transfer via geometry images. In *Proceedings of the 2005 ACM Symposium on Solid and Physical Modeling* (New York, NY, USA, 2005), SPM '05, ACM, pp. 15–26. 2

[LHLF15] LIU T., HERTZMANN A., LI W., FUNKHOUSER T.: Style compatibility for 3D furniture models. *ACM Trans. Graph. 34*, 4 (2015), 85:1–85:9. 12

[LKS15] LUN Z., KALOGERAKIS E., SHEFFER A.: Elements of style: Learning perceptual shape style similarity. *ACM Trans. Graph. 34*, 4 (2015), 84:1–84:14. 12

[MHS*14] MA C., HUANG H., SHEFFER A., KALOGERAKIS E., WANG R.: Analogy-driven 3D style transfer. *Comput. Graph. Forum 33*, 2 (2014), 175–184. 3, 12

[MK12] MÖBIUS J., KOBBELT L.: OpenFlipper: an open source geometry processing and rendering framework. In *Proc. Curves and Surfaces* (2012), pp. 488–500. 3

[MKC*06] MERTENS T., KAUTZ J., CHEN J., BEKAERT P., DURAND F.: Texture transfer using geometry correlation. In *Proc. EGSR* (2006), pp. 273–284. 2, 6, 9, 11

[Mor09] MORELAND K.: Diverging color maps for scientific visualization. In *Proc. ISVC* (2009), pp. 92–103. 5

[MR12] MELVÆR E. L., REIMERS M.: Geodesic polar coordinates on polygonal meshes. *Computer Graphics Forum 31*, 8 (2012), 2423–2435. 4

[Pix15] PIXOLOGIC I.: ZBrush. http://pixologic.com/zbrush, 2015. 1

[PMW*08] PAULY M., MITRA N. J., WALLNER J., POTTMANN H., GUIBAS L. J.: Discovering structural regularity in 3D geometry. *ACM Trans. Graph. 27*, 3 (2008), 43:1–43:11. 3

[RCOL09] ROSENBERGER A., COHEN-OR D., LISCHINSKI D.: Layered shape synthesis: Automatic generation of control maps for non-stationary textures. *ACM Trans. Graph. 28*, 5 (2009), 107:1–107:9. 2

[RPC*10] ROHMER D., POPA T., CANI M.-P., HAHMANN S., SHEFFER A.: Animation wrinkling: Augmenting coarse cloth simulations with realistic-looking wrinkles. *ACM Trans. Graph. 29*, 6 (2010), 157:1–157:8. 3

[SOG09] SUN J., OVSJANIKOV M., GUIBAS L.: A concise and provably informative multi-scale signature based on heat diffusion. In *Proc. SGP* (2009), pp. 1383–1392. 5

[SPKS16] SOLOMON J., PEYRÉ G., KIM V., SRA S.: Entropic metric alignment for correspondence problems. *ACM Transactions on Graphics (TOG) 35*, 4 (2016). 3, 9, 10, 11

[SSCO08] SHAPIRA L., SHAMIR A., COHEN-OR D.: Consistent mesh partitioning and skeletonisation using the shape diameter function. *Vis. Comput. 24*, 4 (2008), 249–259. 5

[Tur01] TURK G.: Texture synthesis on surfaces. In *Proc. ACM SIGGRAPH* (2001), pp. 347–354. 2, 3, 6, 7

[VKZHCO11] VAN KAICK O., ZHANG H., HAMARNEH G., COHEN-OR D.: A survey on shape correspondence. *Computer Graphics Forum 30*, 6 (2011), 1681–1707. 3

[WHRO10] WANG H., HECHT F., RAMAMOORTHI R., O'BRIEN J. F.: Example-based wrinkle synthesis for clothing animation. *ACM Trans. Graph. 29*, 4 (2010), 107:1–107:8. 3

[WL00] WEI L.-Y., LEVOY M.: Fast texture synthesis using tree-structured vector quantization. In *Proc. ACM SIGGRAPH* (2000), pp. 479–488. 2

[WL01] WEI L.-Y., LEVOY M.: Texture synthesis over arbitrary manifold surfaces. In *Proc. ACM SIGGRAPH* (2001), pp. 355–360. 2

[WLKT09] WEI L.-Y., LEFEBVRE S., KWATRA V., TURK G.: State of the art in example-based texture synthesis. In *Eurographics '09 State of the Art Reports (STARs)* (2009), Eurographics. 2

[WSH*16] WANG T. Y., SU H., HUANG Q., HUANG J., GUIBAS L., MITRA N. J.: Unsupervised texture transfer from images to model collections. *ACM Trans. Graph. 35*, 6 (2016), 177:1–177:13. 12

[XCOJ*09] XU K., COHEN-OR D., JU T., LIU L., ZHANG H., ZHOU S., XIONG Y.: Feature-aligned shape texturing. *ACM Trans. Graph. 28*, 5 (2009), 108:1–108:7. 2

[XZCOC12] XU K., ZHANG H., COHEN-OR D., CHEN B.: Fit and diverse: Set evolution for inspiring 3D shape galleries. *ACM Trans. Graph. 31*, 4 (2012), 57:1–57:10. 3

[YHBZ01] YING L., HERTZMANN A., BIERMANN H., ZORIN D.: Texture and Shape Synthesis on Surfaces. In *Eurographics Workshop on Rendering* (2001), Gortle S. J., Myszkowski K., (Eds.), The Eurographics Association. 2

[ZHW*06] ZHOU K., HUANG X., WANG X., TONG Y., DESBRUN M., GUO B., SHUM H.-Y.: Mesh quilting for geometric texture synthesis. *ACM Trans. Graph. 25*, 3 (2006), 690–697. 2

[ZSTR07] ZHOU H., SUN J., TURK G., REHG J. M.: Terrain synthesis from digital elevation models. *IEEE Transactions on Visualization and Computer Graphics 13*, 4 (2007), 834–848. 2