A Compact Representation for Scanned 3D Objects

Bing Wang and Holly Rushmeier
Department of Computer Science
Yale University
{bing.wang, holly.rushmeier}@yale.edu

Abstract

We consider the representation of densely sampled scanned 3D objects. Scanning physical objects can be an efficient way of obtaining natural looking input for computer graphics image generation. Scanned meshes however require large amounts of storage. For applications such as computer graphics, preserving the look of objects rather than precise measurements is critical. We seek a representation that preserves characteristic features at both low and high spatial frequencies. Our proposed representation consists of a simplified base mesh and characteristic detailed patches. The detailed patches contain a spatially dense sampling of mean curvature on a small region on the object surface. A large initial set of patches covering the object is segmented into sets of patches using k-means clustering. Each cluster of patches is then compactly represented with Gaussian distributions. We present comparisons of a series of original scanned objects and the objects reconstructed from the compact representation.

1. Introduction

The widespread availability of 3D scanners makes it possible to get densely sampled models of physical objects. The scanned models, usually in the form of triangle meshes, are very precise and detailed representations of objects. Such representations are highly useful in applications such as the documentation of cultural heritage.

However, for other applications such as computer graphics, it is more important to preserve the characteristic features of objects than the precise measurements. From a particular scanned model, it is useful to generate a variety of novel models which look similar but different in order to populate a synthetic scene. Because the densely sampled triangle mesh doesn't directly encode the characteristic features of an object, it can not be used directly in authoring similar objects.

In this paper we present a compact representation of densely sampled model which consists of a much simplified base mesh and representative patches. This is based on the observation that local areas of objects often experience statistical similarity and structural correlation. For example, rocks are bumpy all around the surface and seashells have the characteristic spiral curves. We cluster into groups the densely sampled local patches using a k-means method and use Gaussian distributions to model each group of patches. This statistical model of local patches not only saves storage and network bandwidth, but also provides us with a method to generate similar models.

The main contribution of the paper includes: (1) a statistical model of local patches; (2) curvature-domain local shape descriptor; (3) a compact mesh representation based on statistical patch model; and (4) approaches of automatically authoring similar models.

The rest of the paper is organized as follows: after review the related literature in section 2, we introduce the statistical patch model in section 3 and the compact mesh representation in section 4. Results are shown in section 5 and future work is discussed in section 6.

2. Related work

The work presented here builds on several areas of geometric modeling – modeling using 3D scan data, multiresolution meshes, detecting and exploiting redundancy in models, and geometric texture transfer. We briefly review this previous work, and highlight how the problem and solution we consider differs from it.

Parts of existing models, including models produced by 3D scanning, can be used as input for new models [7]. However rather than using entire "chunks" of existing models, we hope to reuse characteristic shapes of objects, particularly natural objects such as rocks. Characteristic shapes are associated with materials [1].

In computer graphics, procedural modeling has dominated the creation of models of natural objects. For example, there have been efforts at finding mathematical descriptions of the basic shapes of natural objects such as shells [6]. To model natural irregularities in shape procedures such as Perlin noise [21] are added to geometries representing nat-

2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops
978-1-4244-4441-0/09/$25.00 ©2009 IEEE

ural shapes such as terrains defined as height fields [20]. Natural looking effects have relied on manual tuning of parameters, or using general image statistics [22]. Following one aspect of the procedural approach, we seek to define models in terms of an overall form that is modulated by high spatial frequency details.

Multiresolution representation for geometric models is a highly developed area for efficient shape editing and rendering, e.g. [9]. The representations consist of multiple meshes capturing geometric features at different frequency bands. Since eliminating redundancy is not a goal of the representation, precise detailed information is recorded at multiple levels. The mesh at a particular level can be obtained by adding the recorded details into the mesh at the adjacent lower level. In our work we concern with separating the basic form of an object and the shape details that are characteristic of the material forming the object, so we do not consider these multi-level representations but simply a two level representation.

Since we seek a compact representation, we look for redundancy in models. One approach to this is symmetry analysis of shapes which aims to find spatial invariant of geometry elements under 3D transfer, rotation and scaling, e.g. [16]. This kind of symmetry is widely presented in some objects such as buildings. Other objects like rocks don't necessarily have the "perfect" symmetry. The work in this paper doesn't rely on repetitions but rather on similarities in a statistical sense.

To find groups of similar, but not identical features, we consider clustering. Clustering has been applied long before for understanding images. Clustering textons into multiple groups is usually one of the basic steps for image segmentation and texture classification, e.g. [23]. For regular textures where textons are tiled in structure, researchers have proposed approaches to automatically identify the textons and extract their spatial organization, e.g. [14]. These approaches are similar to those of shape symmetry analysis, and don't extend to statistical textures directly. For the results of clustering, we will not try to find a specific regular structure.

Given a description of "typical" high spatial frequency features, a natural mechanism for applying these features to a basic form is texture transfer. Many methods have been developed recently for the transfer of geometric texture. These may be divided into two classes – methods that transfer a different material or different ornamental features to a base form, and methods that enhance the surface variations of the basic object. An example of a detailed additional material is adding a volumetric layer to a surface to represent fur [10]. Examples of ornamental features include wire meshes, basket weaves, or patterns of repeated small decals that are represented as meshes on a flat surface that are deformed and repeated to cover a shape [5, 24]. Additional material or ornamental features are generally modeled separately, rather than being extracted from an existing model.

The second category of geometric texture transfer, that enhances the surface variations of the base form rather than adding a new layer of material, is relevant to the problem we address. In this case, methods have been developed that attempt to extract detail from basic shape of one object, and then apply detail to a new shape. Bhat et al.[2] synthesize geometric textures using voxel definitions of the object and fine scale geometry. A vector field on the surface is used to guide the synthesis on the base surface. Lai et al.[12] transfer using geometry images rather than volumetric structures. In both cases the transferred geometry is found from an existing object by defining a smoothed and simplified base object. The geometric height field or geometric structure then depends on the specific simplification that is made. In our method we attempt to find a detailed surface description that is independent of arbitrarily specifying a surface depth to define the difference between small and large scale features.

In the general field of texture transfer it has been recognized that the texture should be correlated to large scale geometric features. Texture transfer that uses analysis of underlying geometry has been proposed in particular to capture the look of weathered objects [15, 17]. Accounting for pattern variations with large scale variations is important in weathered appearance since it depends on the actions of natural phenomena. Similarly, to capture the fine scale geometric variations of natural objects we will account for the underlying large scale features.

Many different types of surface maps have been proposed to model surface detail, including texture, bump, displacement and normal maps [19]. Maps are an effective way to represent fine scale features in a compact form. For example, de Toledo et al. consider using height fields as a geometry texture map over an entire object to store mesostructure [3]. However, synthesizing fine scale geometry as displaces or normals requires careful and time consuming attention to continuity to avoid obvious artificial discontinuities in surfaces.

A way to avoid the problems of arbitrarily defining a depth layer for details and the discontinuities in synthesis of displacement or normal maps, is to characterize details in the curvature domain. Eigensatz introduced the idea of modeling in the curvature domain [4]. Working with curvature captures the appearance of the object without first defining a base mesh for offset, and without compensating for a particular coordinate system. Working with a higher order quantity such as curvature and then synthesizing the surface avoids obvious surface discontinuities.

Based on this assessment of previous work, in the next sections we develop our approach for compactly represent-

ing surfaces. By working in the curvature domain we invert the usual process of simplifying a surface and then defining details by differencing the detailed and simplified meshes. In the next two sections we describe how we first analyze the curvature variations in small local areas (i.e. patches) on a mesh and define a small set of patch descriptions, and then define a compact mesh representation designed to make use of these patch descriptions.

3. A Statistical Model for Local Patches

Our analysis of an object's shape begins by considering the characteristics of small areas distributed over the object surface. We can consider characteristics without reference to a base mesh or to the global coordinate system by working in the curvature domain. Similar local areas are often observed at different surface locations of an object. We introduce a model in this section to express this similarity statistically using clustering and Gaussian distributions.

3.1. Shape descriptor for local patches

We consider a collection of vertices V, edges E and faces F that form a triangle mesh $M = \{V, E, F\}$, which approximates a smooth 2-manifold. Let $N_V = |V|$ denote the number of vertices, let p_i denote the i-th vertex and let x_i, n_i and H_i denote the position, normal and mean curvature of p_i, respectively.

Given a vertex p_i on M and a small radius r, there is a neighborhood of p_i which consists of all the points on M with a geodesic distance to p_i less than r. Such a neighborhood is called a local patch of p_i, denoted by P_i.

A shape descriptor is computed for each local patch. The shape descriptor should have the power to reconstruct the patch. Mean curvature is used because of its character of being intrinsic (independent of the global coordinate system) and because no reference base mesh is required (as is required to define a height field). Mean curvatures at vertices are computed using the mesh Laplacian with cotangent weights [18] as shown in equation 1, where A_i is one-third of the 1-ring neighborhood area of p_i, $N_1(i)$ is the set of 1-ring neighborhood vertices of p_i, and α_{ij} and β_{ij} are the opposite angles of edge (p_i, p_j), as shown in Figure 1. Mean curvatures at other surface points are computed by bilinearly interpolating vertex curvatures using barycentric coordinates. We uniformly sample the mean curvatures on the local patch and form a vector of mean curvatures as the shape descriptor for this patch.

$$2H_i n_i = \frac{1}{2A_i} \sum_{p_j \in N_1(i)} (\cot \alpha_{ij} + \cot \beta_{ij})(x_i - x_j) \quad (1)$$

Local parameterization is used to efficiently compute the local patch and the spatially regular sampling positions. For

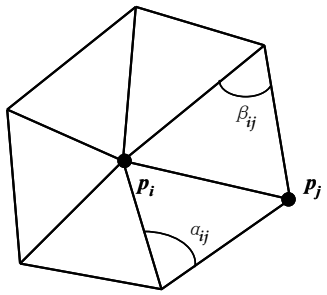


Figure 1. 1-ring neighborhood of a vertex

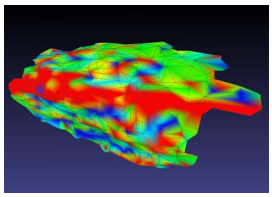
a particular vertex p_i, we first parameterize its neighborhood onto a 2D domain, with p_i mapped to the origin, using LSCM (Least Squares Conformal Maps) [13]. The parameterization produces, for each surface point within the neighborhood, a corresponding (u, v) parameterization coordinate. By approximating the geodesic distance on the surface with the Euclidean distance on the parameterization domain, the local patch P_i is then computed as the set of surface points whose parameterization coordinates have the property as $u^2 + v^2 < r^2$, that is, the set of points whose corresponding points on the parameterization domain locate within the circle centering at the origin and with a radius of r.

The sampling process for computing the shape descriptor is also done in the parameterization domain. Given a sampling distance d, all the sampling positions are found as the lattice points in the parameterization domain as shown in Figure 2(d). By composing the mean curvatures sampled at the corresponding surface points into a vector, we get the shape descriptor $S_i = (H_{ij})$ with $1 \leq j \leq N_s$, where H_{ij} is the mean curvature sampled at the j-th sampling point of the i-th patch and N_s is the number of sampling positions.

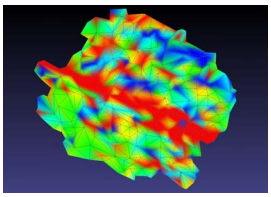
In practice, the parameterization domain is rotated before the computation of shape descriptor such that the most significant direction always coincides with the X-axis, as shown in Figure 2(c). The most significant direction is defined as the direction which has the maximum absolute value of average mean curvatures. An orientational histogram is used to compute this direction and only the mean curvatures at vertices are considered to accelerate the process.

3.2. K-means clustering and Gaussian distribution

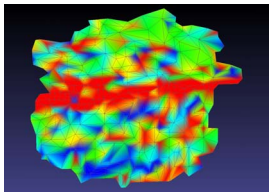
By sampling at different local areas along the object surface, we get multiple local patches and their shape descriptors. Let N_P denote the number of local patches. Those local patches may have distinct shapes. As shown in Figure 3, for example, patches along the edges of the object and those at the flatter area are quite different. The difference is measured with Euclidean distance in the high dimensional shape descriptor space. A k-means clustering method (Lloyd's al-



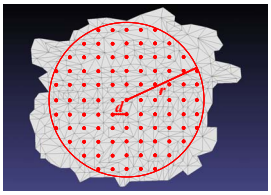
(a) Local patch in 3D. Mean curvatures are visualized with red for large and blue for small.



(b) Local patch in 2D

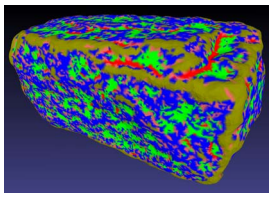


(c) Reoriented local patch in 2D

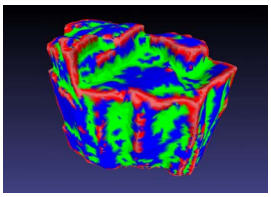


(d) Regular sampling

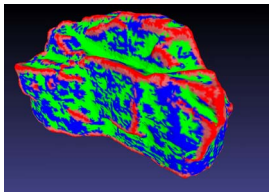
Figure 2. Compute shape descriptor for a local patch.



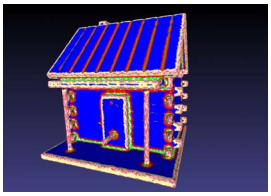
(a) Clustering result for stoneA.



(b) Clustering result for stoneC.



(c) Clustering result for stoneL.



(d) Clustering result for Wooden House.

Figure 3. Clustering results. Different color stands for local areas of different clusters.

gorithm, as described in [11]) is applied to group the N_P shape descriptors into K categories $G_i, 1 \leq i \leq K$, with $|G_i| = N_{G_i}$.

The shape descriptors in each group G_i form a cloud of N_{G_i} points in a N_s-dimensional space. We use a 1D Gaussian model to represent the distribution of the point cloud in each dimension, which produces a compact representation with the sacrifice of losing the correlation information between different dimensions. The Gaussian distribution for the j-th dimension of i-th group is shown in equation 2.

$$Gauss_{ij}(x) = \frac{1}{\sigma_{ij}\sqrt{2\pi}} \exp\left(\frac{-(x-\mu_{ij})^2}{2\sigma_{ij}^2}\right) \tag{2}$$

The mean μ_{ij} is computed as

$$\mu_{ij} = \frac{\sum_{k|S_k \in G_i} H_{kj}}{N_{G_i}}. \tag{3}$$

The standard deviation σ_{ij} is computed as

$$\sigma_{ij} = \sqrt{\frac{1}{N_{G_i}-1} \sum_{k|S_k \in G_i} (H_{kj} - \mu_{ij})^2}. \tag{4}$$

4. A compact mesh representation

The densely sampled triangle mesh can be compactly represented using a base mesh which captures the low frequency features and a group of Gaussian distributions which capture the high frequency features. Rather than using existing simplification methods (e.g. [8]) that see to maintain dimensional accuracy, we define a base mesh that is compatible with using Gaussian distributions of patch curvature described in section 3.

4.1. Encoding the base mesh

In order to record the low frequency and large scale features of the object, we first simplify the input dense mesh M to a base mesh M_b. The simplification is done in two steps, choosing the retained vertices and reconstructing the triangle mesh from the retained vertices.

We choose retained vertices from the original mesh M. The selection is constrained by two factors: (1) the vertices are uniformly sampled along the surface such that each has a neighborhood approximating the predefined patch size, that is, the distance between adjacent vertices should be about twice the patch radius r so there will be no large overlaps or gaps between patches; (2) the retained vertices should capture the geometric features as well as possible. Mean curvatures are used as guidance for vertex selection, and the vertex with larger absolute value of mean curvature will be selected with higher priority than others, since these vertices represent areas with distinctive features.

A triangle mesh is then built up from the retained vertices. We first flood-fill the surface of the mesh M from all of the retained vertices and get a partition of the mesh. Then we compute the dual graph of the partition to get a polygon mesh M' and M' is then converted to a triangle mesh M_b by further subdividing each non-triangle polygon into multiple triangles.

The resulting base mesh is formed by a much smaller set of vertices that all lie on the original mesh.

4.2. Encoding the details

The local shapes of the dense mesh is recorded using statistical patch models introduced in section 3. For each

retained vertex, we find the local patch on the dense mesh M with a predefined size r. We compute a shape descriptor for each neighborhood. We then perform patch analysis as shown in section 3 on those shape descriptors. Finally we get a Gaussian distribution parameters table of K rows and N_s columns. For each vertex in the base mesh M_b, we keep an index to the table rows to record the type of the local patch, and a scalar value to record the rotation angle for reorienting the local patch.

4.3. Decoding

We can reconstruct a dense mesh from the compact representation in four steps: (1) tessellate the base mesh into higher resolution; (2) for each original vertex on the base mesh, compute a local patch shape descriptor using the recorded statistical model and reorientation angle; (3) resample the shape descriptors so all vertices of the tessellated mesh have a mean curvature defined; (4) reconstruct the mesh from vertex mean curvatures.

The reconstruction is formulated as an energy minimization problem

$$\arg\min_X (w_H^2 \|L(X) - \mathbf{H}(X)\|^2 + w_p^2 \|X_b - X_b^0\|^2), \quad (5)$$

where X denotes the current vertex positions, $\mathbf{H}(X)$ denotes the vertex mean curvature normals, $L(X)$ denotes the mesh Laplacian with cotangent weights, X_b denotes the current positions of vertices of the base mesh and X_b^0 denotes the original positions of those vertices. w_H and w_p denote the weight of mean curvature constraints and the weight of positional constraints. That is, we need to solve in least squares sense the following system

$$\begin{pmatrix} w_H L \\ w_p I \end{pmatrix} X = \begin{pmatrix} w_H \mathbf{H}(X) \\ w_p X_b^0 \end{pmatrix}. \quad (6)$$

The system is nonlinear because both $L(X)$ and $\mathbf{H}(X)$ are nonlinearly dependent on X. We solve it by iteratively updating L and $\mathbf{H}$ using current X and then updating X by solving a linear system.

5. Results

We demonstrate in this section results of using statistical patch models to compactly represent dense triangle meshes and geometric detail transfer based on the compact representation. All the experiments were done on a 3.0 GHz Pentium IV PC with 3GB memory. Predefined values of patch size $r = 5.0mm$ and sampling distance $d = 0.5mm$ are used for all models. Table 1 shows the times of computing the compact representations and reconstructing meshes from them. Please note that we deliberately choose flat shading to render all the meshes to better show how the geometry of the meshes are represented.

5.1. Clustering

Clustering is used to find statistical similarities and characteristic features of objects. Figure 3 shows the clustering results of several objects, each having been clustered into 4-6 groups. Small number of clusters are used for better visualization. Ridges, valleys and flat areas are well separated into different groups. Because each local patch is reoriented to the most significant direction, similar patches with different 3D poses are still grouped together.

5.2. Compact representation

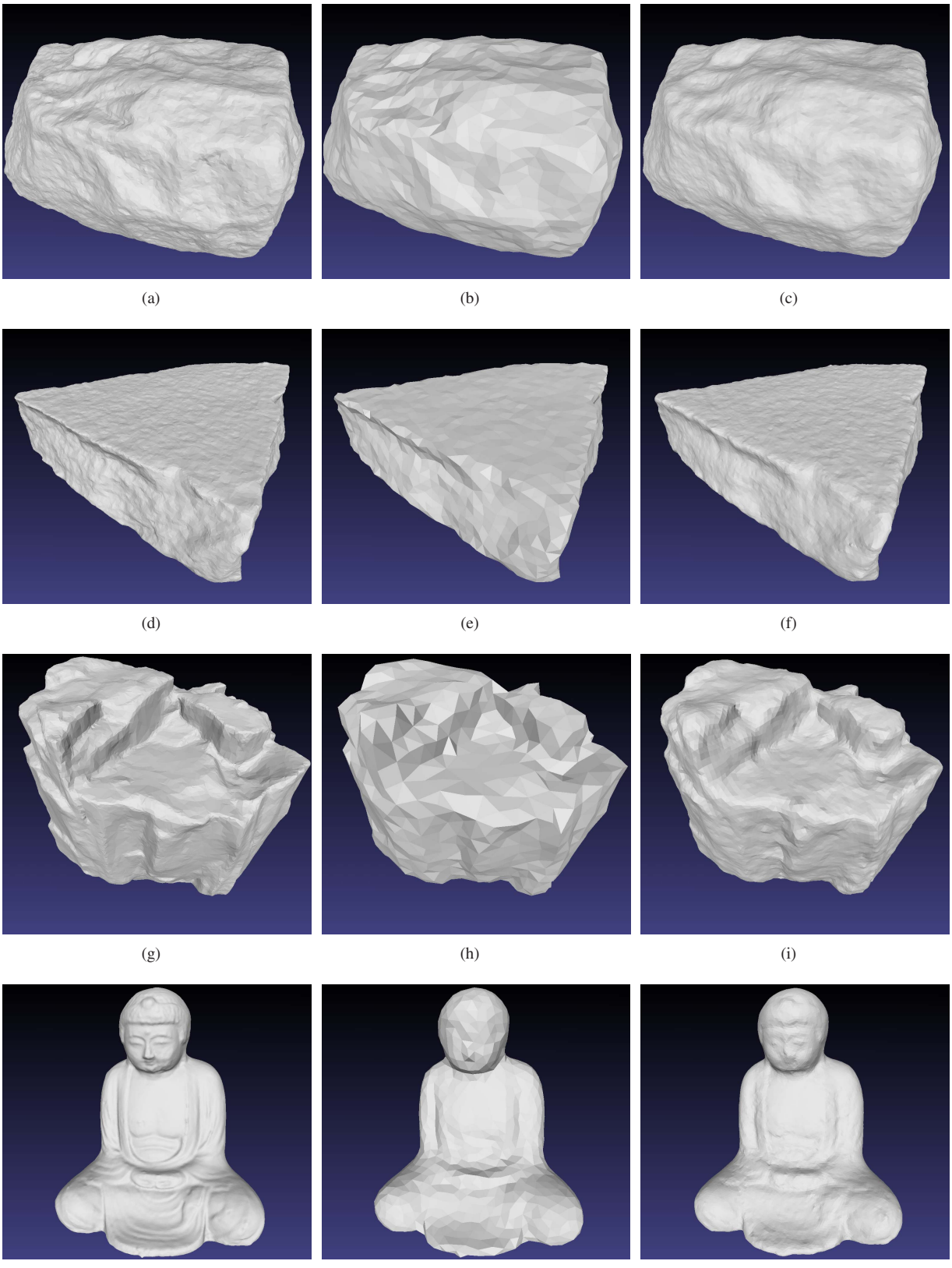
A dense triangle mesh is represented with a much simplified base mesh and representative local patch models. The size of the compact representation is therefore determined by two factors: (1) the size of the base mesh, which depends on the patch size r; (2) the size of the table of patch models, which depends on the number of clusters K and the sampling distance d. Since both r and d are predefined in the experiments, only K influences the final size of the representation. K for fractal surfaces such as bumpy rocks can be quite small for an acceptable reconstruction. In our experiments for various stones, we always set K to 4. For other objects like the Buddha shown in Figure 4(j) - 4(l), a larger K has to be used to account for its many features such as the nose, eyes and hairs, etc., which are different and unique.

5.3. Reconstruction

Figure 4 shows results of reconstructing dense meshes from compact representations. It can be seen that the compact representations add detail to the simplified meshes. However, compared to original meshes, the reconstructed ones are still missing high frequency information and thus look more smooth. We believe that is because the Gaussian distribution model is not adequate to accurately record detail geometry. We will exploit more advanced statistical models in future.

5.4. Detail transfer

The Gaussian distributions represent characteristic patches of objects, and can be directly used to transfer detail geometry from one object to another. In Figure 5, we show examples of transferring detail geometry of stoneA as shown in Figure 4(a) to several objects. The transfer is guided by the correlation between the based meshes. That is, for each local patch of the target base mesh, we compute a shape descriptor and then identify its cluster by finding its nearest neighbor in the space of the shape descriptors from the source mesh. In this way, we can get Gaussian distribution models from the source mesh for each local patch of the target.



(a) (b) (c)

(d) (e) (f)

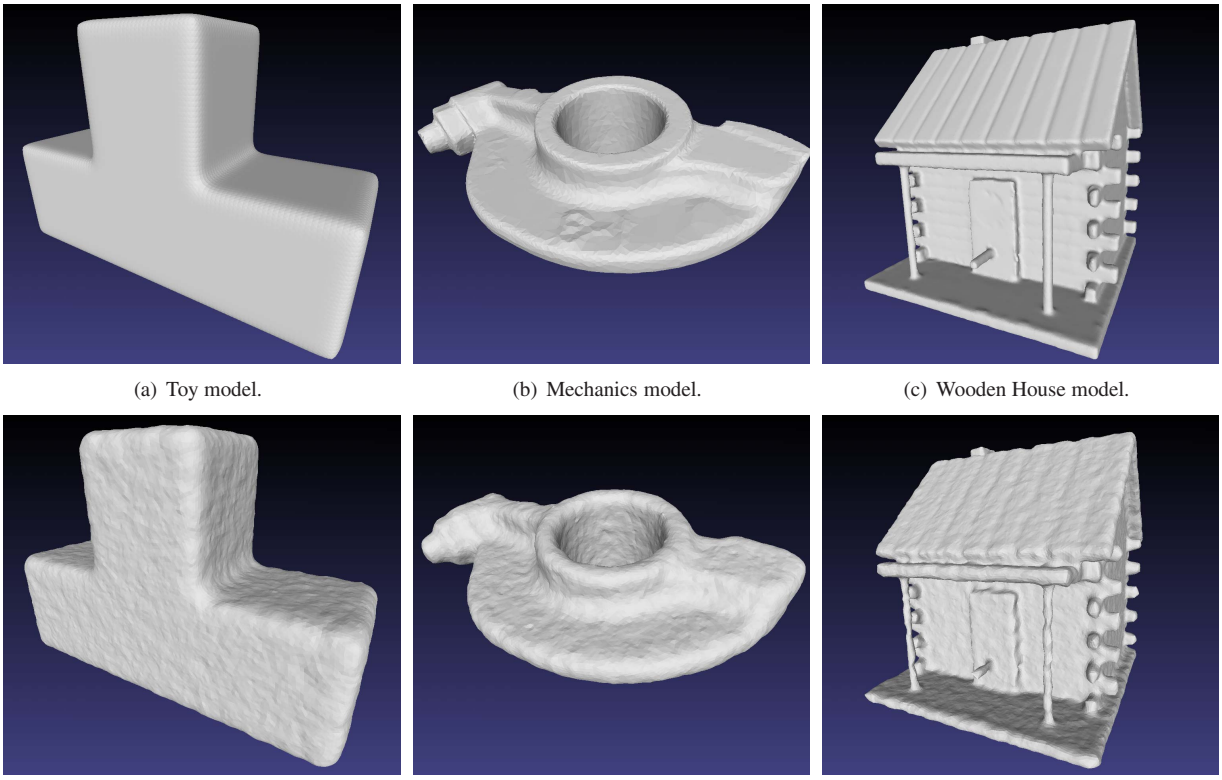
(g) (h) (i)

(j) (k) (l)

Figure 4. Reconstructions from compact representations. Left column: original meshes; center column: simplified meshes; right column: reconstructed meshes. First row: StoneA; second row: StoneB; third row: StoneC; last row: Buddha.

M	size of M	size of M_c	T_{simp}	T_{sd}	T_{stat}	$T_{compcurv}$	T_{recon}
StoneA	2.23	0.15	27	67	6	89	23
StoneB	1.90	0.12	32	40	1	62	15
StoneC	0.95	0.07	16	19	1	34	7
Buddha	3.81	0.59	108	30	15	49	10

Table 1. Timing results and sizes of representations from our experiments. Time is in seconds. Size is in MBytes. M_c stands for the compact representation which includes both the base mesh and the statistical patch models. T_{simp}, T_{sd} and T_{stat} stand for the time computing base mesh, shape descriptors and statistical patch models, respectively. $T_{compcurv}$ and T_{recon} stand for the time of recovering curvatures from statistical patch models and reconstructing mesh from curvatures.



(a) Toy model. (b) Mechanics model. (c) Wooden House model.

(d) Toy model with detail transferred. (e) Mechanics model with detail transferred. (f) Wooden House model with detail transferred.

Figure 5. Geometry detail transfer based on statistical patch models. First row: original meshes. Second row: meshes with transferred detail.

6. Future work

We have described a method for compactly representing 3D scanned objects that decomposes the object into a base form and a small set of detailed patches. This representation is useful for representing natural objects for graphics applications in which capturing characteristic features, rather than precise dimensions, is desirable. The proposed statistical model can be used to transfer geometric features and thus involve in authoring novel objects, which makes it different than traditional mesh compression methods and conventional detail-enriching rendering techniques based on surface maps.

The main value of this work is that it introduces a number of promising avenues for future research in the area of using 3D scanned data in authoring systems for objects in computer graphics. These directions include:

- Automatically determining the optimal patch size and number of clusters for the surface details. An obvious starting point for this is a brute force approach that tries multiple scales and selects the scale that best exploits redundancy.
- Developing a statistical patch description that accounts for correlations across dimensions.
- Finding the best method for determining the matching

of detailed patch to mesh vertex for newly authored shapes.

- Finding a method for statistically characterizing the base shape as well as the detailed features by analyzing a collection of similar scanned objects.

- Finding a method that includes variations of reflectance (i.e. spatial varying bidirectional reflectance distribution functions) that are correlated to geometric detail variations.

The success of research along these lines will facilitate the efficient modeling of complex natural scenes by eliminating manual parameter tuning and storing the results of modeling in a compact form.

Acknowledgements This material is based upon work supported by the National Science Foundation under Grant No. 0528204.

References

[1] E. Adelson. On seeing stuff: The perception of materials by humans and machines. In *Proceedings of the SPIE*, volume 4299, pages 1–12, 2001.

[2] P. Bhat, S. Ingram, and G. Turk. Geometric texture synthesis by example. In *SGP '04: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 41–44, New York, NY, USA, 2004. ACM.

[3] R. de Toledo, B. Wang, and B. Levy. Geometry textures. In *Computer Graphics and Image Processing, 2007. SIBGRAPI 2007. XX Brazilian Symposium on*, pages 79–86, 2007.

[4] M. Eigensatz, R. Sumner, and M. Pauly. Curvature-domain shape processing. In *Computer Graphics Forum*, volume 27, pages 241–250. Blackwell Publishing Ltd, 2008.

[5] G. Elber. Geometric texture modeling. *Computer Graphics and Applications, IEEE*, 25(4):66–76, July-Aug. 2005.

[6] D. R. Fowler, H. Meinhardt, and P. Prusinkiewicz. Modeling seashells. In *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, pages 379–387, New York, NY, USA, 1992. ACM.

[7] T. Funkhouser, M. Kazhdan, P. Shilane, P. Min, W. Kiefer, A. Tal, S. Rusinkiewicz, and D. Dobkin. Modeling by example. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, pages 652–663, New York, NY, USA, 2004. ACM.

[8] M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 209–216, New York, NY, USA, 1997. ACM.

[9] H. Hoppe. Progressive meshes. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 99–108, New York, NY, USA, 1996. ACM.

[10] J. T. Kajiya and T. L. Kay. Rendering fur with three dimensional textures. In *SIGGRAPH '89: Proceedings of the 16th annual conference on Computer graphics and interactive techniques*, pages 271–280, New York, NY, USA, 1989. ACM.

[11] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. Piatko, R. Silverman, and A. Y. Wu. The analysis of a simple k-means clustering algorithm. In *SCG '00: Proceedings of the sixteenth annual symposium on Computational geometry*, pages 100–109, New York, NY, USA, 2000. ACM.

[12] Y.-K. Lai, S.-M. Hu, D. X. Gu, and R. R. Martin. Geometric texture synthesis and transfer via geometry images. In *SPM '05: Proceedings of the 2005 ACM symposium on Solid and physical modeling*, pages 15–26, New York, NY, USA, 2005. ACM.

[13] B. Lévy, S. Petitjean, N. Ray, and J. Maillot. Least squares conformal maps for automatic texture atlas generation. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 362–371, New York, NY, USA, 2002. ACM.

[14] Y. Liu, W.-C. Lin, and J. Hays. Near-regular texture analysis and manipulation. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, pages 368–376, New York, NY, USA, 2004. ACM.

[15] J. Lu, A. Georghiades, A. Glaser, H. Wu, L. Wei, B. Guo, J. Dorsey, and H. Rushmeier. Context-aware textures. *ACM Transactions on Graphics-TOG*, 26(1), 2007.

[16] A. Martinet, C. Soler, N. Holzschuch, and F. Sillion. Accurate detection of symmetries in 3D shapes. *ACM Transactions on Graphics (TOG)*, 25(2):439–464, 2006.

[17] T. Mertens, J. Kautz, J. Chen, P. Bekaert, and F. Durand. Texture transfer using geometry correlation. *Rendering Techniques 2006*, page 273, 2006.

[18] M. Meyer, M. Desbrun, P. Schroder, and A. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. *Visualization and Mathematics*, 3:35–57, 2002.

[19] T. Möller, N. Hoffman, and E. Haines. *Real-time rendering*. AK Peters, Ltd., 2008.

[20] F. K. Musgrave, C. E. Kolb, and R. S. Mace. The synthesis and rendering of eroded fractal terrains. In *SIGGRAPH '89: Proceedings of the 16th annual conference on Computer graphics and interactive techniques*, pages 41–50, New York, NY, USA, 1989. ACM.

[21] K. Perlin. An image synthesizer. In *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 287–296, New York, NY, USA, 1985. ACM.

[22] E. Reinhard, P. Shirley, M. Ashikhmin, and T. Troscianko. Second order image statistics in computer graphics. In *APGV '04: Proceedings of the 1st Symposium on Applied perception in graphics and visualization*, pages 99–106, New York, NY, USA, 2004. ACM.

[23] M. Varma and A. Zisserman. A statistical approach to texture classification from single images. *International Journal of Computer Vision*, 62(1):61–81, 2005.

[24] K. Zhou, X. Huang, X. Wang, Y. Tong, M. Desbrun, B. Guo, and H.-Y. Shum. Mesh quilting for geometric texture synthesis. *ACM Trans. Graph.*, 25(3):690–697, 2006.