Construction of Manifolds via Compatible Sparse Representations

RUIMIN WANG, LIGANG LIU, ZHOUWANG YANG, KANG WANG, WEN SHAN,
JIANSONG DENG, and FALAI CHEN
School of Mathematical Sciences, University of Science and Technology of China

Manifold is an important technique to model geometric objects with arbitrary topology. In this article, we propose a novel approach for constructing manifolds from discrete meshes based on sparse optimization. The local geometry for each chart is sparsely represented by a set of redundant atom functions, which have the flexibility to represent various geometries with varying smoothness. A global optimization is then proposed to guarantee compatible sparse representations in the overlapping regions of different charts. Our method can construct manifolds of varying smoothness including sharp features (creases, darts, or cusps). As an application, we can easily construct a skinning manifold surface from a given curve network. Examples show that our approach has much flexibility to generate manifold surfaces with good quality.

CCS Concepts: • **Computing methodologies** → **Shape analysis**; • **Mathematics of computing** → *Mathematical optimization*

Additional Key Words and Phrases: Manifold, Sharp feature, Sparse representation, Curve network

ACM Reference Format:

Ruimin Wang, Ligang Liu, Zhouwang Yang, Kang Wang, Wen Shan, Jiansong Deng, and Falai Chen. 2016. Construction of manifolds via compatible sparse representations. ACM Trans. Graph. 35, 2, Article 14 (February 2016), 10 pages.
DOI: http://dx.doi.org/10.1145/2835488

1. INTRODUCTION

The construction of surfaces with arbitrary topology has attracted a great deal of attention in the geometric modeling community. Nonuniform rational B-splines (NURBS) are a standard surface representation in computer-aided design, which is based on the rigorous mathematics of spline theory and the paradigm of stitching patches together. It is cumbersome for NURBS to construct smooth geometric objects with arbitrary topology due to the difficulty of meeting geometric continuity constraints between adjacent surface patches. Subdivision surfaces, defined as the limits of recursive refinement algorithms, have advantages for modeling arbitrarily complicated geometry and topology. Yet they generally do not possess analytic representations, and special need is required to take care of extraordinary points whose geometric properties differ significantly from other regular regions.

Manifolds are another attractive representations for modeling geometric objects with arbitrary topology [Grimm and Hughes 1995; Ying and Zorin 2004; Gu et al. 2006]. Manifold is defined as a collection of surface patches with overlapping regions, and each patch corresponds to a chart by a one-to-one map. At the overlapping region of two charts, a transition function is needed to map one chart to another. The manifold representation can be considered as an extension of B-splines with arbitrary topology, which guarantees continuity of certain order by the definition of transition functions from chart to chart on the domain manifold. All transition functions are generally smooth analytical functions. The local geometry on each chart is a linear combination of some set of basis functions, such as polynomials [Ying and Zorin 2004], splines [Gu et al. 2006], and rational splines [Della Vecchia et al. 2008]. However, previous manifold-based approaches can hardly represent geometric objects with specified varying continuity across patches and usually handle the situation by taking special care and adopting a complicated design.

To model geometric objects with adaptive continuity and various geometric features such as creases, darts, and cusps, we propose a novel approach for manifold construction. The basic idea is as follows. We use a redundant set of atom functions (also called *dictionary*) with various features to represent local geometry. With given observation data on the geometry, we employ sparsity regularization instead of least-squares fitting to approximate the given data to avoid degeneracy due to the redundancy of atom functions. A global optimization is then proposed to guarantee a compatible sparse representation in the overlapping regions of different patches.

The fitting problem with the redundant dictionary is actually a sparse representation [Elad 2010] that accounts for most or all information of a geometry signal by a linear combination of a small number of elementary signals. The key reasons we employ the sparsity regularization in geometric processing are twofold. First, certain groups of atom functions usually cannot satisfy all properties; for example, some edges are sharp features and others are smooth, and thus a redundant dictionary repository is needed. Second, if a redundant set of atom functions is selected all at once, the overfitting phenomenon always occurs by least-squares approximation. Sparse representation can balance these two problems by choosing a different set of atom functions in different situations.

Sparse representation is a recently developed powerful tool for image and signal processing. Sparsity is extensively used in famous transforms, such as discrete Fourier transform (DFT),

Authors' addresses: R. Wang; email: wruimin@ustc.edu.cn; L. Liu (corresponding author); email: lgliu@ustc.edu.cn; Z. Yang; email: yangzw@ustc.edu.cn; K. Wang; email: wkcagd@mail.ustc.edu.cn; W. Shan; email: sw09001@mail.ustc.edu.cn; J. Deng; email: dengjs@ustc.edu.cn; F. Chen; email: chenfl@ustc.edu.cn.

DOI: http://dx.doi.org/10.1145/2835488

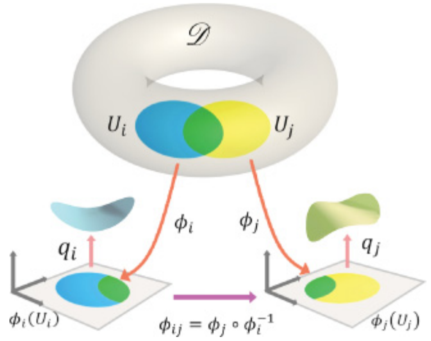


Fig. 1. The definition of a manifold.

wavelet transform, and singular value decomposition, to reveal elegant structures and achieve compact representations of signals of interest. However, the global compatibility of signals, i.e., the relationship between different signal patches, is ignored in traditional sparse representation due to the simplicity and regularity of images. But geometric objects usually have complicated overlapping regions between two charts, and they do not have regular parameterization as images do. Compatible sparse manifold representation becomes an ideal tool to solve such difficulties. Furthermore, it is a rather compact representation and has explicit formulation, and it can adaptively approximate geometries with various features.

The contribution of our work can be summarized as follows:

—We introduce a new approach to construct manifolds for geometric objects, which can adapt to various features of the objects.
—A global sparse optimization model is proposed to obtain a compatible sparse representation for the manifold surface.
—Our approach provides an easy way to construct a manifold surface from a 3D curve network.

2. RELATED WORK

Construction of manifold surfaces. Manifold is a technique to construct surfaces from domain manifolds. A manifold surface consists of a collection of charts with overlapping regions where transition functions are defined between them [Grimm and Zorin 2006] (Figure 1). Based on a manifold representation, it is simple to define auxiliary linear spaces of scalar or vector-valued fields on the surface, as the construction can be done in embedding space with any dimension. Grimm et.al. first introduced manifolds for surface modeling in computer graphics, where a manifold surface is built from a polygonal mesh using one chart per mesh element (vertices, edges, and faces) [Grimm and Hughes 1995]. Since then, several approaches have been proposed for constructing manifold surfaces over arbitrary mesh models over the past two decades [Navau and Garcia 2000; Ying and Zorin 2004; Gu et al. 2006; He et al. 2006; Wang et al. 2008; Della Vecchia et al. 2008; Della Vecchia and Jüttler 2009; Gallier et al. 2012]. A few of them are mentioned here. Navau and Garcia [2000] present a method for generating C^m manifold surfaces, which requires a preprocess of repeated Catmull-Clark subdivision steps to isolate the irregularities of the mesh. Ying and Zorin [2004] propose a manifold surface construction with C^∞ smoothness, whereas Tosun and Zorin [2011] propose a similar construction with C^k smoothness using spline functions. Based on spline theory, manifold splines and their variants, e.g., manifold T-splines and polycube splines, were introduced in Gu et al. [2006], He et al. [2006], and Wang et al. [2008], respectively. Grimm et al. [2009] adaptively fit the representation of meshes of varying resolutions and sampling quality but not the geometry. Constructions of smooth and piecewise rational free-form surfaces were given in Della Vecchia et al. [2008] and Della Vecchia and Juttler [2009], where a manifold surface is described as a collection of rational tensor-product and triangular surface patches. Li et al. [2006] automatically convert a mesh of arbitrary topology into a T-spline surface.

Differing from the existing methods on manifold construction, we represent the local function as a sparse representation of a set of redundant atom functions, called a *dictionary*. Furthermore, a compatible sparse representation is proposed for global compatibility. Various geometric features can be represented by atom functions based on sparse optimization, which is generally difficult to handle using existing approaches.

Modeling from curve networks. Constructing surfaces from 3D curve networks has long been studied in the geometric modeling and computer-aided design (CAD) community [Abbasinejad et al. 2011; Bessmeltsev et al. 2012; Zhuang et al. 2013]. Designers often start by specifying a network of 3D curves, which serves as a visual proxy of the desired model, and then construct a surface model interpolating the curve network. There are a variety of techniques for interpolating or approximating curve networks with smooth surfaces, such as Coons patches [Coons 1967], N-sided patches [Malraison 2000; Nasri et al. 2009], mesh surfaces [Schaefer et al. 2004; Nealen et al. 2007; Abbasinejad et al. 2011; Bessmeltsev et al. 2012], and subdivision surfaces [Schaefer et al. 2004; Nasri et al. 2009]. Pan et al. [2015] present a new method constructing curve network using flow-line curves. As it is highly ambiguous in defining surfaces on arbitrary 3D curve cycles, most of existing methods need to presegment the curve cycles into subcurve cycles and then construct surfaces over them.

As an application of sparse manifold representation, we can easily construct a manifold surface that approximates a given 3D curve network. Due to the sparsity of geometry, our approach can generate a reasonable surface over the whole curve network with global smoothness.

Sparse representation. Sparse representation has drawn a lot of attention over the past decade, as it can reveal certain structures of signals via the assumption of sparsity. It provides high-quality performance for image applications such as noise reduction, super-resolution, compression, and face recognition [Donoho 1995; Elad and Aharon 2006; Wright et al. 2009; Yang et al. 2010]. Elad [2010] comprehensively illustrates the whole framework of sparse representation and the underlying mathematical theory. Xu et al. [2015] introduce how sparsity is used in geometric processing. Recent work [Xiong et al. 2014] uses a dictionary learning framework to reconstruct gemoetry.

Geometry as a signal also contains redundant information and thus should be able to be sparsely represented. We represent the local function on a manifold chart as a sparse representation of a redundant dictionary of atom functions. The manifold construction benefits from this sparse representation. Unlike images, which have regular parameterization, geometric objects have complicated overlapping areas between two patches or charts. Thus, we develop an optimization algorithm for compatible sparse representation in manifold construction, which as far as we know is the first attempt to address this issue.

3. CONSTRUCTION

This section introduces the approach for constructing manifolds via an optimization framework of compatible sparse representations.

3.1 Manifold Surfaces

Manifold structure. Manifold modeling is a technology used to construct surfaces from a domain manifold [Grimm and Hughes 1995; Ying and Zorin 2004]. As shown in Figure 1, the domain manifold $\mathcal{D}$ is covered by a collection of charts $\{(U_i, \phi_i)\}_{i=1}^{n}$ with overlapping regions. Here $\phi_i : U_i \to \mathbb{R}^2$ is a one-to-one map (parameterization) from U_i to an open set $\phi_i(U_i)$ in a plane domain, which defines the local coordinate system for U_i. If two charts (U_i, ϕ_i) and (U_j, ϕ_j) overlap, the transition function $\phi_{ij} = \phi_j \circ \phi_i^{-1}$ describes the transformation from $\phi_i(U_i)$ to $\phi_j(U_j)$.

Local geometry. On each chart (U_i, ϕ_i), a local function $q_i(u, v)$ from $\phi_i(U_i)$ to Euclidean space $\mathbb{R}^d$ is constructed for defining the geometry locally, where (u, v) are the local coordinates of a point $\mathbf{x} \in U_i$.

Weight function. A set of weight functions $w_i : \phi_i(U_i) \to \mathbb{R}$ with partition of unity are defined on charts, i.e., $\sum_{j:\mathbf{x}\in U_j} w_j(\phi_j(\mathbf{x})) = 1$, where the summation subscript $\{j : \mathbf{x} \in U_j\}$ means that the summation is for all j belonging to the index set $\{j \mid \mathbf{x} \in U_j\}$.

Global geometry. The global geometry function on the domain manifold $\mathcal{D}$ is defined by $f = \sum_{j=1}^{n} w_j(\phi_j)q_j(\phi_j)$, which is practically evaluated on an individual chart (U_i, ϕ_i) as

$$f(\mathbf{x}) = \sum_{j:\mathbf{x}\in U_j} w_j(\phi_j(\mathbf{x}))q_j(\phi_j(\mathbf{x})), \quad \forall \mathbf{x} \in U_i. \tag{1}$$

According to manifold theory, the continuity of the global geometry function $f(\mathbf{x})$ depends on the continuity of transition functions $\{\phi_{ij}(u, v)\}$, weight functions $\{w_i(u, v)\}$, and local geometry functions $\{q_i(u, v)\}$.

3.2 Sparse Representation of Local Geometry

Using the given mesh as a domain mesh, suppose that we already have a collection of charts and their transition maps. Our immediate goal is to define the local geometry on each chart using a sparse representation.

Defining local geometry. Generally, the local function $q_j(u, v)$ on the chart (U_j, ϕ_j) is considered a linear combination of a set of atom functions $\mathcal{A} = \{a_l(u, v)\}_{l=1}^{L}$ defined on $\phi_j(U_j)$, i.e.,

$$q_j(u, v) = \sum_{l=1}^{L} c_{lj} a_l(u, v). \tag{2}$$

Denote $\mathbf{c}_j = (c_{1j}, \ldots, c_{Lj})^T$ as the coefficient vector of local function associated with chart (U_j, ϕ_j).

Fitting observation data. We view geometries from a perspective of signals, which are observed on the domain manifold. Assume that there are K observations on chart $\{(U_j, \phi_j)\}$ given by $\{((u_k, v_k), h_k)\}_{k=1}^{K}$. The observation signal h_k at the point (u_k, v_k) can be any scalar or vector values. For the vector values (like the coordinates of geometry positions) of observations, we process each component individually. Denote $\mathbf{h} = (h_1, \ldots, h_K)^T$ and $A = (a_{kl} = a_l(u_k, v_k))_{K\times L}$. The solution of least-squares fitting by minimizing $\|A\mathbf{c}_j - \mathbf{h}\|_2^2$ is given by $\hat{\mathbf{c}}_j = (A^T A)^{-1} A^T \mathbf{h}$ to predict the local geometry.

Problem of overfitting. Generally, polynomials are chosen as the set of atom functions. However, the choice of the maximal polynomial degree might result in artifacts in the least-squares fitting. Lower degrees make it difficult to match rich details of signal, whereas higher degrees lead to overfitting effects, like the small-scale surface artifacts shown in Figure 6 of Tosun and Zorin [2011]. Figure 2 shows another example.

Sparse representation. Inspired by the work on sparse coding in image processing [Elad 2010], we introduce a set of redundant atom functions $\mathcal{A} = \{a_l(u, v)\}_{l=1}^{L}$ (a dictionary) to represent the local geometry. As the atom functions are redundant, the local geometry can be represented by a sparse combination of these atom functions. The fitting problem with a redundant dictionary is actually a sparse representation that accounts for most or all information of a geometry signal by a linear combination of very few elementary signals. Thus, we come to a regression problem of sparse representation,

$$\min_{\mathbf{c}_j} \ \|A\mathbf{c}_j - \mathbf{h}\|_2^2 \ \text{ s.t. } \ \|\mathbf{c}_j\|_0 \leq \delta, \tag{3}$$

where δ is the sparsity parameter, which describes the number of atom functions, i.e., the support size, used to represent a local function. Minimization (3) with respect to $\mathbf{c}_j$ is a sparse coding operation that can be handled by the orthogonal matching pursuit (OMP) algorithm [Pati et al. 1993].

3.3 Compatible Sparse Representations

Problem of individually fitting. The sparse optimization (3) is performed to obtain the local function at each chart individually. However, the global function (1) that is defined by blending all local functions may not fit the observation data well (Figure 3(d)), as the local functions are not compatible with each other in the overlapping regions.

Fitting with global compatibility. To this end, we propose a new optimization framework for compatible sparse representations, which is able to guarantee the global compatibility of geometry functions in overlapping regions.

Suppose that the observation data of global geometry function f on the domain manifold are given as $\{(\mathbf{p}_k, h_k)\}_{k=1}^{K}$ and the local coordinates of these data on chart (U_j, ϕ_j) are $\{\phi_j(\mathbf{p}_k) = (u_k^j, v_k^j)\}$. Then we have

$$\begin{aligned} f(\mathbf{p}_k) &= \sum_{j:\mathbf{p}_k\in U_j} w_j(\phi_j(\mathbf{p}_k))q_j(\phi_j(\mathbf{p}_k)) \\ &= \sum_{j:\mathbf{p}_k\in U_j} w_j\big(u_k^j, v_k^j\big)q_j\big(u_k^j, v_k^j\big). \end{aligned} \tag{4}$$

Note that the function value of the global geometry at one point is influenced by several local functions that actually have relationships with others. Therefore, the ℓ_2-norm of fitting error on all points is computed as a whole and given by

$$E_{\text{fit}}(C) = \sum_{k=1}^{K} \left(\sum_{j:\mathbf{p}_k\in U_j} \left(w_j\big(u_k^j, v_k^j\big) \sum_{l=1}^{L} c_{lj} a_l\big(u_k^j, v_k^j\big) \right) - h_k \right)^2, \tag{5}$$

where $\mathbf{c}_j = (c_{1j}, \ldots, c_{Lj})^T$ is the coefficient vector of the local function on the chart (U_j, ϕ_j) and $C = (\mathbf{c}_1, \ldots, \mathbf{c}_n)$ is the matrix of decision variables.

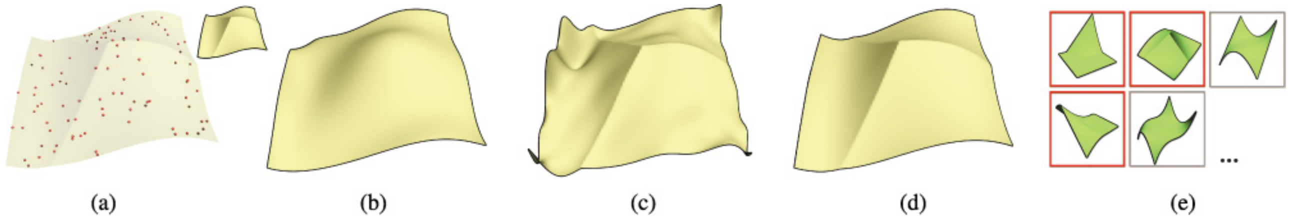


Fig. 2. Given a set of sampling points (100 points in total) from a C^0 surface (a), the obtained fitting surface via polynomials (b), redundant atom functions (c), and sparse representation (d). (e) The first five atom functions (C^0 shape functions in red boxes and polynomials in gray boxes) selected in the sparse representation. The sparsity is set at 25.

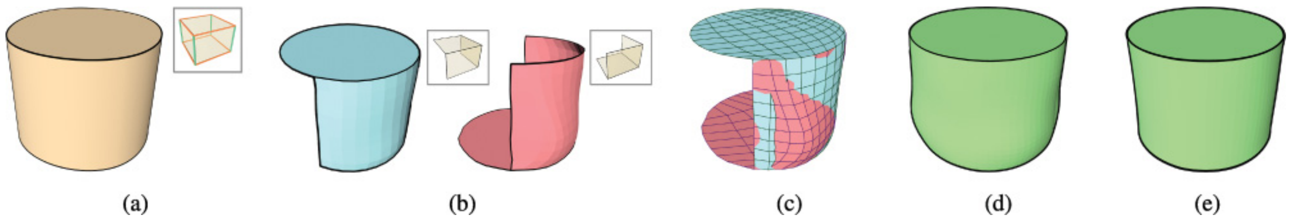


Fig. 3. Given subdivision surface with control mesh (a), the results of sparse representations on individual charts (b), the surface's appearance in overlapping regions (c), the result of a manifold (d) by simple blending, and the obtained surface (e) by compatible sparse representation.

To obtain the sparse coefficients of the local function on each chart, we formulate the compatible sparse representation of global geometry function as

$$\begin{aligned} &\min_{C} \; E_{\text{fit}}(C) \\ &\text{s.t. } \; \|\mathbf{c}_j\|_0 \le \delta, \; j = 1, 2, \ldots, n. \end{aligned} \tag{6}$$

It is worth pointing out that our compatible sparse representation (6) for global geometry differs from the existing methods of sparse coding by taking into account the global compatibility.

3.4 Optimization Solver

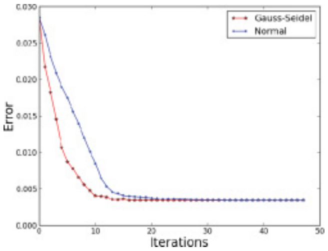
To solve the compatible representation system (6), we develop an iterative splitting algorithm that alternatingly updates each $\mathbf{c}_j$. At first, for each $j \in \{1, \ldots, n\}$, we separately obtain the solution of a local sparse optimization problem as an initialization, i.e.,

$$\begin{aligned} \mathbf{c}_j^{(0)} = \; &\arg\min_{\mathbf{c}_j} \sum_{\mathbf{p}_k \in U_j} \left(\sum_{l=1}^{L} c_{lj} a_l\left(u_k^j, v_k^j\right) - h_k \right)^2, \\ &\text{s.t. } \; \|\mathbf{c}_j\|_0 \le \delta. \end{aligned} \tag{7}$$

where each problem is solved by the algorithm [Pati et al. 1993]. In the t-th step, we iteratively update each $\mathbf{c}_j$ by fixing coordinates of other local functions $\{\mathbf{c}_i\}_{i \neq j}$ and solving the sparse optimization problem as follows:

$$\begin{aligned} \mathbf{c}_j^{(t+1)} = \; &\arg\min_{\mathbf{c}_j} \sum_{\mathbf{p}_k \in U_j} \left[w_j \sum_{l=1}^{L} c_{lj} a_l\left(u_k^j, v_k^j\right) \right. \\ &\left. + \sum_{i:\, \mathbf{p}_k \in U_{i \neq j}} w_i \sum_{l=1}^{L} c_{li}^{(t)} a_l\left(u_k^i, v_k^i\right) - h_k \right]^2 \\ &\text{s.t.} \|\mathbf{c}_j\|_0 \le \delta. \end{aligned} \tag{8}$$

Minimization with respect to $\mathbf{c}_j$ in (8), while assuming that the others $\{\mathbf{c}_i\}_{i \neq j}$ are fixed, can be done as a sparse coding operation that employs the OMP. The pseudocode of the compatible sparse representation is given in Algorithm 1. Note that our algorithm can also adopt the Gauss-Seidel iterative technique, in which $\mathbf{c}_i^{t+1}$ is used instead of $\mathbf{c}_i^t$ while updating $\mathbf{c}_j^{t+1}$ for $j > i$. The Gauss-Seidel technique can be used to accelerate the convergence as observed in our implementation. For instance, the right figure shows the steps of iteration for constructing the model of Figure 3 using and without using the Gauss-Seidel technique.



ALGORITHM 1: Optimization of Compatible Sparse Representation

```
Input: Charts $\{(U_j, \phi_j)\}$ and weight functions $\{w_j\}$, observations $\{(\mathbf{p}_k, h_k)\}$, the dictionary $\{a_l(u, v)\}_{l=1}^{L}$, and sparsity parameter $\delta$.
Output: Global geometry function $f$, which is sparsely represented by the coefficient vectors $\{\mathbf{c}_j\}_{j=1}^{n}$.
  Initialization Separately obtain the initial coefficient vector on each chart by solving (7).
  while [ $E_{\text{fit}}(C)$ is reduced ] {

      Update each $\mathbf{c}_j$ by minimizing (8), where the others $\{\mathbf{c}_i\}_{i \neq j}$ are fixed at their values from the last step.

  } endwhile
```

4. RESULTS AND APPLICATIONS

We demonstrate the flexibility of our proposed approach for constructing manifolds with varying features in this section.

4.1 Approximating Subdivision Surfaces

As did Ying and Zorin [2004], we adopt our approach to construct manifolds to approximate the subdivision surfaces. The manifolds constructed by Ying and Zorin [2004] are C^∞. Differently, we construct manifolds that can approximate subdivision surfaces with C^0 sharp features [DeRose et al. 1998].

Domain manifold. We use the same scheme of Ying and Zorin [2004] to define the charts per vertex, construct transition maps, and build weights with partition of unity (See Ying and Zorin [2004] for details). As the transition maps and weight functions are all C^∞, the smoothness of the manifold is determined by the smoothness of local functions defined on the charts.

Dictionary repository. To enable the capability of representing sharp features, we use a dictionary $\mathcal{A}$ that consists of two types of atom functions, i.e., $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2$. The first type of atom functions $\mathcal{A}_1$ includes all polynomials up to degree 14, i.e., a total of 120 polynomial functions, which are C^∞. The second type of atom functions $\mathcal{A}_2$ includes 55 C^0 shape functions that are usually used in FEM [Zienkiewicz and Taylor 1977] (See the Appendix for details.)

Observation data. We sample all control points from the refined mesh after two Catmull-Clark subdivision steps and compute the limit positions of these points as our observation data for the global geometry. The subdivision surfaces could have various sharp features, such as creases, darts, and cusps, if C^0 constraints are enforced on certain edges of control meshes [DeRose et al. 1998].

Manifold construction. The global geometry of manifold is obtained by fitting the observation data with the dictionary atom functions by solving (6).

4.2 Experimental Results

We first test our algorithm on some synthetic (clean and noisy) data and then apply our approach for approximating subdivision surfaces with various sharp features.

Robustness of sparse representation. To represent the local geometry, a solution is usually obtained by least-squares fitting the input data with a set of specified basis functions, e.g., using polynomials as in Ying and Zorin [2004]. Instead, our approach represents the local geometry via sparse optimization, i.e., sparsely choose suitable ones from a redundant set of atom functions. We test the least-squares method and our algorithm on clean and noisy synthetic data, as shown in Figure 4. The experimental results demonstrate the robustness of sparse representation, whereas the least-squares method leads to overfitting phenomenon in the presence of noise.

In addition, we approximate the known polynomial via our solver assuming that the sparsity is the same as the size of the polynomial. For instance, we extract samples from polynomial $0.6x^2 + 0.25y^2 + 0.09x^4y^2 + 0.12x^6y^7$ and solve the problem with sparsity 4. In this case, the OMP algorithm precisely recovers the polynomial. Actually, in our limited trial (about 100 times), the OMP algorithm accurately recovers all polynomials. However, the OMP algorithm can only guarantee the precise recovery when the bases in the dictionary are orthogonal to each other [Pati et al. 1993], whereas the bases we use do not fit such a property. Thus, we can only say that the practical performance of fitting polynomials is quite good.

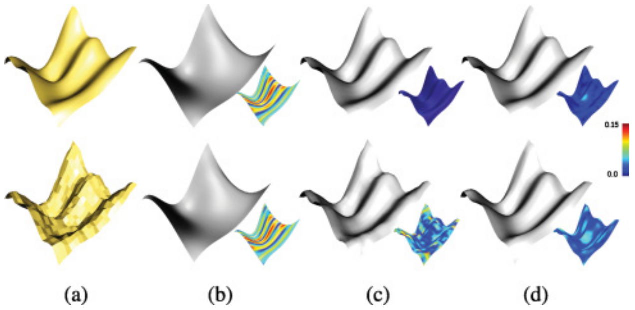


Fig. 4. The results of representing local geometry (a) using least-squares method and sparse representation with fitting error visualized by color map. Approximating results in the fashion of least-squares using polynomials with degree up to 5 (b) and 14 (c); approximating results using sparse representation (d). The input consist of clean data (upper row) and noisy data (lower row).

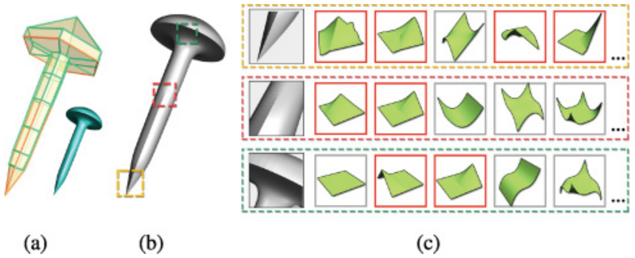


Fig. 5. Constructing manifold to approximate subdivision surface. (a) The input control mesh with enforced C^0 features shown in red. (b) The manifold generated by our approach. (c) The close-up views of three parts of sharp features including crease, dart, and cusp. The first five atom functions that are adaptively selected for representing the local features are shown.

Adaptivity to various features. Due to the redundancy of atom functions and the sparse optimization, our approach can generate manifolds that adapt to various features by choosing suitable atom functions to represent the corresponding features. As shown in Figure 5(a), we enforce C^0 constraints on some of the edges (shown in red) of the control mesh. Thus, various sharp features, including creases, darts, and cusps, can be created in the subdivision surface defined by the control mesh. Using the observation data on the subdivision surface, our approach produces a manifold surface as shown in Figure 5(b). The dashed rectangular regions in green, red, and yellow are zoomed and shown in Figure 5(c), respectively. It is seen that the surface part within the green region (bottom row in Figure 5(c)) represents a crease feature and that the surface parts within the red and yellow regions (middle row and upper row in Figure 5(c)) represent a dart feature and a cusp feature, respectively. The atom functions with the first five largest nonzero coefficients by solving (6) are shown on the right side of the corresponding surface parts. The C^0 atom functions are marked as red throughout the article.

Figure 6 shows another example of the manifold generated by our approach, which approximates various features, including crease, dart, and cusp. In Figure 7, various C^0 constraints are enforced in the control mesh (a) and the manifold surfaces generated by our approach can approximate the various features well. More examples of manifolds constructed by our approach are shown in Figure 8.

Sparsity parameter. The sparsity parameter δ in our algorithm controls the sparsity, i.e., the number of nonzero coefficients for the atom functions. Figure 9 shows an example illustrating how the constructed manifolds are affected by different values of the sparsity parameters. In general, the larger sparsity parameter tends to

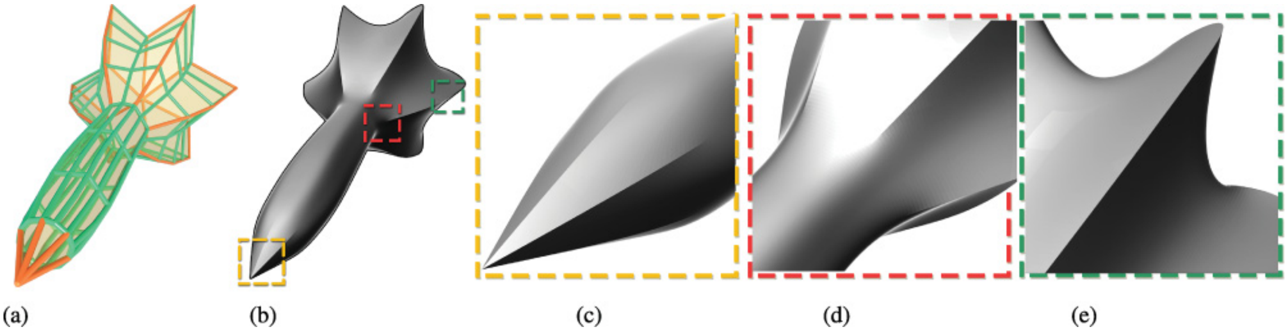


Fig. 6. From a mesh (a), our approach generates a manifold surface (b) approximating its Catmull-Clark subdivision surface. Sharp features are created in the subdivision surface by enforcing C^0 constraints on edges shown in red (a). The constructed manifold (b) approximates various features like crease, dart, and cusp well. The regions in red, green, and blue are zoomed and shown in (c), (d), and (e), respectively.

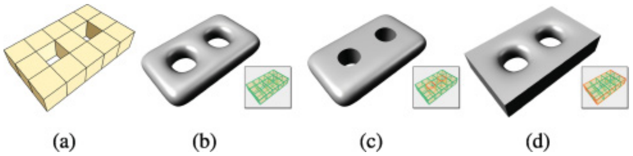


Fig. 7. (a) Control mesh. (b-d) Manifolds constructed by our approach to approximate the subdivision surface with various C^0 constraints on the edges (shown in red).

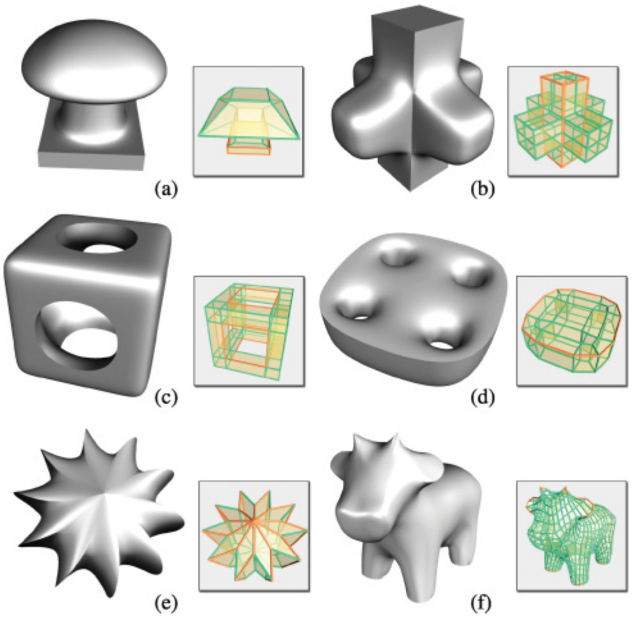


Fig. 8. Examples of manifolds constructed by our approach to approximate the subdivision surfaces with given control meshes in the lower right. C^0 constraints shown in red are enforced to created sharp features in the subdivision surfaces.

produce overfitting effects on the resulting manifolds (see Figure 9(e)), whereas a small one lacks the freedom to represent a certain shape (see Figure 9(c)). The sparsity parameter is empirically set to 25 in our implementation.

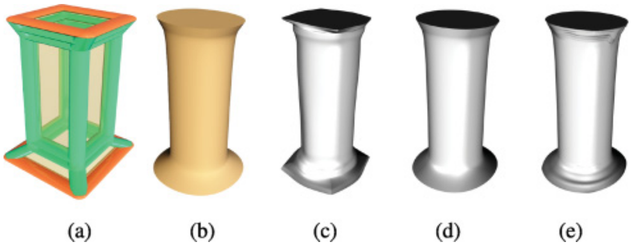


Fig. 9. Manifolds constructed with different sparsity parameters. (a) Input control mesh with C^0 constraints on the edges in red. (b) The limit subdivision surface with sharp features. (c-e) Manifolds constructed by our approach with $\delta = 5, 25, 55$, respectively.

Difference from Ying and Zorin [2004] and Tosun and Zorin [2011]. The differences among our approach, Ying and Zorin [2004], and Tosun and Zorin [2011] lie in the construction of local geometry, which are twofold. First, the method of Ying and Zorin [2004] uses polynomial functions, the method of Tosun and Zorin [2011] uses spline functions to represent local geometry, and our approach uses polynomial functions as well as C^0 shape functions as atom functions, having the capability to represent sharp features. Second, our approach adopts sparsity optimization to select a small number of atom functions to represent the local geometry. Thus, our approach has more flexibility to represent shapes with sharp features.

Figure 10 shows examples of constructing manifolds by using the methods of Ying and Zorin [2004] and Tosun and Zorin [2011] and ours. The parameters of Tosun and Zorin [2011] are set as $g = 5$ and the spline functions are C^3 continuous. In the upper row, no C^0 constraints are enforced in the edges. All methods approximate the smooth subdivision surface very well. In the lower row, C^0 constraints are enforced in some edges of the control mesh. Our method can capture sharp features in the subdivision surface, whereas the methods of Ying and Zorin [2004] and Tosun and Zorin [2011] produce artifact in the result.

Choice of atom functions. As mentioned in Section 4.1, the default set of atom functions in our implementation consists of polynomials up to degree 14 and C^0 shape functions. We have tested some different alternatives of atom functions and show in Figure 11 the results of approximating a subdivision surface by using only polynomials (see Figure 11(b)); only C^0 shape functions (see Figure 11(c)); aricher set of atoms consisting of polynomials,

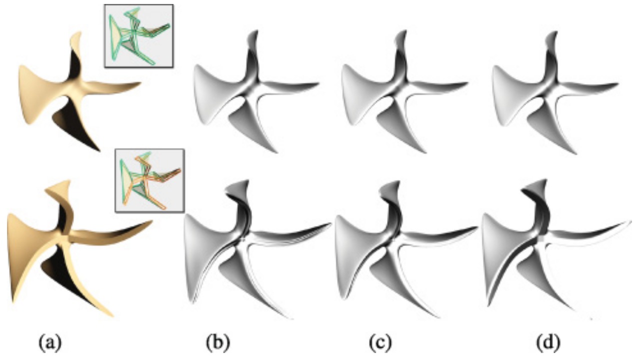


Fig. 10. Comparison with Ying and Zorin [2004] and Tosun and Zorin [2011]. (a) The limit subdivision surfaces of input control meshes (in the upper right) where edges in red are enforced with C^0 constraints. (b) Manifolds produced by the method of Ying and Zorin [2004]. (c) Manifolds produced by the method of Tosun and Zorin [2011]. (d) Manifolds generated by our approach. Upper row: Results from a smooth subdivision surface. Lower row: Results from a subdivision surface with sharp features.

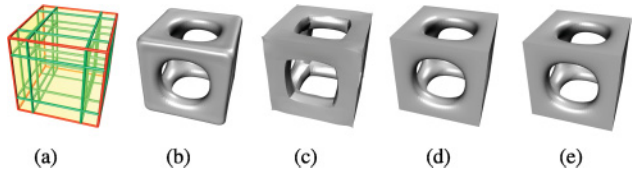


Fig. 11. Manifolds construction approximating subdivision surface with different atom functions. (a) Control mesh. (b) Manifolds generated with only polynomials. (c) Manifolds generated with only shape functions. (d) Manifolds generated with polynomials, spline functions, sine and cosine functions, and shape functions. (e) Manifolds generated with polynomials and shape functions.

splines [Tosun and Zorin 2011], trigonometric, and C^0 shape functions (see Figure 11(d)); and our default set of atom functions (see Figure 11(e)). The first two alternatives perform with poor approximation, and the third alternative can achieve good approximation as well as our chosen set of atom functions. Overall, our algorithm uses a default set of fewer atom functions and has adequate capability to approximate surfaces with various features.

Smoothness of manifold. The constructed manifold using our approach has a close-form representation as a linear combination of atom functions. Obviously, the manifold is at least C^0 continuous. C^0 can only happen on the curves where the basis functions are C^0. In other regions except the curves, the manifold is C^∞. Thus, the manifold constructed by our approach is C^∞ almost everywhere.

Advantages over subdivision surfaces. In geometric modeling, extra requirements are generally enforced on the smoothness of the surface. The manifold constructed by our approach can achieve higher smoothness at a chart if we discard the C^0 atom functions in the dictionary of this chart. Taking the mesh shown in Figure 12(a) as an example, the valence of the top vertex is 10. It is an extraordinary point and is thus only C^1 in the subdivision surface. In our approach, we can discard the C^0 shape functions from the dictionary of the chart of this vertex and the manifold is C^∞ at this point as shown in Figure 12(b). At the same time, the manifold can represent the sharp features if C^0 constraints are enforced at the edges in red as shown in Figure 12(a). This cannot be accomplished for subdivision surfaces, as they are only C^1 at extraordinary points.

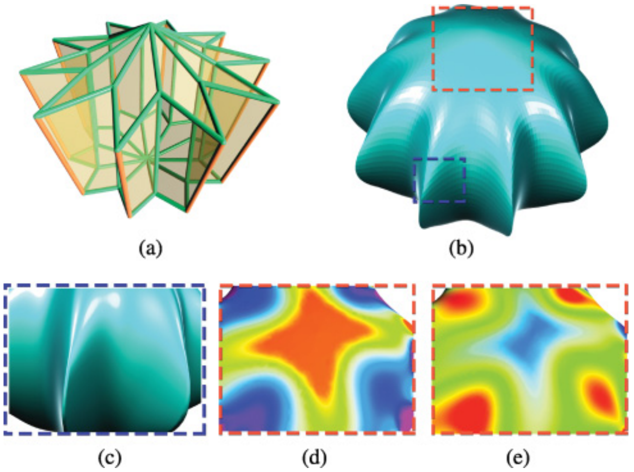


Fig. 12. (a) Control mesh with C^0 constraints shown in red. The top vertex of (a) has a valence of 10. We discard the C^0 atom functions from the dictionary at the chart around this vertex. The manifold surface constructed by our approach is shown in (b). The surface adapts to the sharp features in blue region and is C^∞ in the red region. (c) Close-up view of the blue region. (d,e) Gaussian curvature and mean curvature of the red region.

Table I. Statistics of Performance Including the Number of Charts, Average Number of Used C^0 Shape Functions for a Chart, and Time Cost of Models in Figure 8

Model: Figure 8	(a)	(b)	(c)	(d)	(e)	(f)
Charts (#)	24	74	64	56	42	802
C^0 Functions (#)	3.71	4.32	2.03	2.38	5.64	0.17
Time (s)	2.2	6.1	5.4	5.6	3.3	34.1

The color maps of Gaussian curvature and mean curvature of the local region around the vertex are shown in Figure 12(d) and (e), respectively, which shows that the manifold constructed by our approach remains fair on the top region.

Statistics. We implemented our algorithm on a dual-core 2.9GHz laptop with 8G memory. Generally, our algorithm takes dozens of seconds to construct a manifold with about 800 charts. Table I shows the statistics of the performance of our algorithm, including the number of charts, the average number of used C^0 shape functions for a chart, and time cost of models. Note that the sparsity is set at 25 in our implementation.

4.3 Modeling from Curve Network with Manifold

Due to the sparsity nature of geometry, the proposed manifold construction approach provides a nice scheme for constructing surfaces from a 3D curve network.

Curve network. The curve network consists of a set of curves that roughly reveal a 3D shape. The curves can be specified by users using sketch-based 3D painting tools [Bae et al. 2008, 2009] or extracted from some existing shapes.

Domain manifold. First, the cycles in the curve network are identified by the method of Zhuang et al. [2013]. The intersection points among cycles are regarded as vertices of the domain manifold. If there is a cycle curve connecting two vertices, we create a line

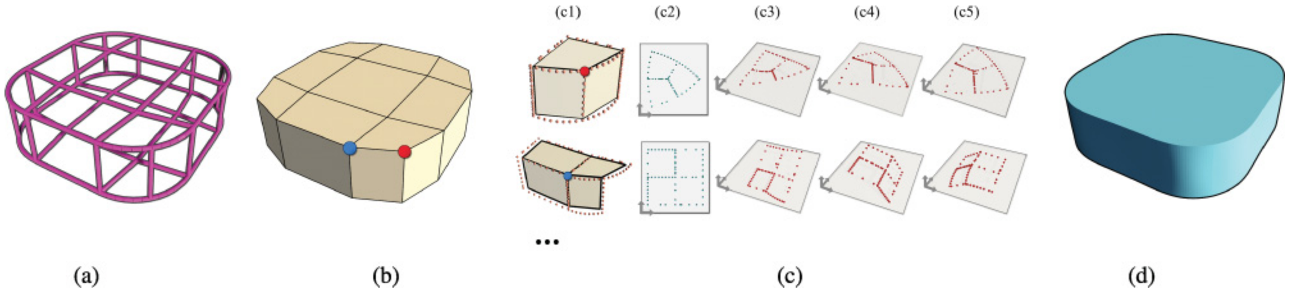


Fig. 13. The procedure for constructing a manifold surface from a given curve network (a). First we use the identified cycle information to build the domain manifold (b). Then we obtain observation data by local parameterization of curves on each chart (c). Finally, the manifold surface (d) is constructed using our approach. (c1) Sampling points on curves. (c2) Local coordinates of samples on individual charts. (c3-c5) The observation data of (x, y, z), respectively.

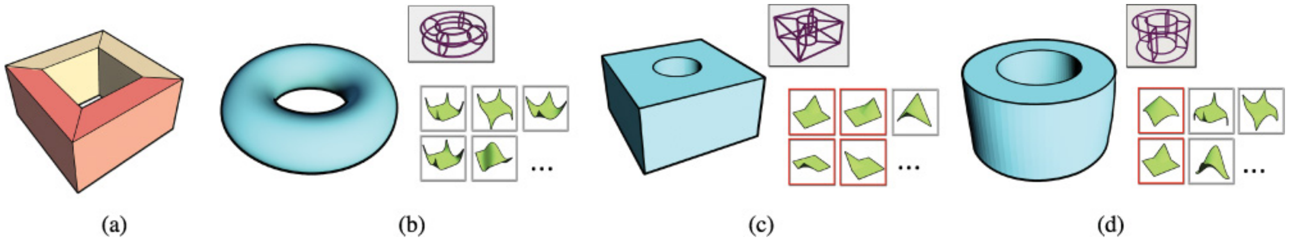


Fig. 14. Generating manifold from different geometries defined on the same domain manifold. (a) The domain mesh. (b-d) Three manifolds created from given curve networks (in the upper right) and the first five chosen atom functions (in the lower right). C^0 shape functions are shown with red frames.

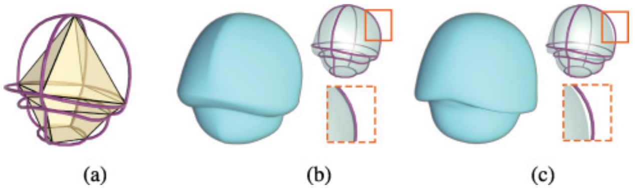


Fig. 15. The fitting of a mushroom like curve network. (a) Illustratration of the curve network and domain mainfold. The result of our method is shown as (b), and the result using only polynomials is shown as (c). Note that (b) fits better on the curves than (c) with C^0 functions.

between them and regard it as an edge of the domain manifold. Finally, the domain manifold is built over the generated polygonal mesh.

Parameterization of curves. Each curve corresponds to an edge in the domain manifold. For each curve, points are sampled from it and are parameterized in arc length along its corresponding edge on the domain manifold. The sampled points of all curves are then mapped to their local coordinates in $\phi_i(U_i)$ by the map ϕ_i. The spatial positions of the points are considered as the observation data of the geometry functions.

Examples. Ultimately, we can achieve a manifold surface that approximates the given curve network by our approach. Although the points on the curve network give very sparse observation data on the manifold, our method can still properly generate a manifold from this sparse information. Figure 13 shows an example to illustrate the whole process of creating a manifold from a given 3D curve network. Figure 14 shows how different manifold surfaces are constructed from three given curve networks, respectively, which are all defined on the same domain manifold. In this example, our approach adaptively chooses the atom functions to achieve varying smoothness along the curves.

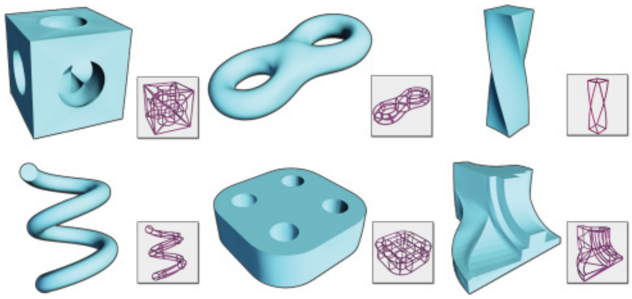


Fig. 16. Examples of manifold surfaces generated from 3D curve networks (in the lower right).

When our method is applied to fit a curve network with more complex geometry like a mushroom in Figure 15, our method smartly represent the geometry in a reasonable way (b). Since the network is fitted automatically and if one wants to get a smooth object, we can get the result using only polynomials (c). Both results conform to human intuition. In addition, (b) has a smaller error on existing curves.

More manifolds constructed from 3D curve networks are shown in Figure 16. The results demonstrate that our approach can produce intuitively reasonable surfaces from sparse 3D curve networks.

5. CONCLUSION AND FUTURE WORK

This article presents a novel approach for constructing manifolds over meshes. The local geometry is represented by a sparse representation of a dictionary consisting of redundant atom functions. A compatible sparse representation optimization is proposed to guarantee the global compatibility of the manifold. We

Table II. Statistics of All Results Showing the Number of Darts, Creases, and Edges, the Number of Charts, the Average Number of C^0 Functions Used, the Average Number of C^0 Functions Used in Smooth Charts, and the Computation Time

Model	Fig. 5	Fig. 6	Fig. 9	Fig. 10	Fig. 11(e)	Fig. 7(c)	Fig. 7(d)	Fig. 8(a)	Fig. 8(b)	Fig. 8(c)	Fig. 8(d)	Fig. 8(e)	Fig. 8(f)
Darts (#)	3	12	0	0	0	0	0	0	0	0	0	0	10
Cusps (#)	1	2	0	4	8	0	8	8	8	0	0	2	2
Creases (#)	4	12	2	4	12	4	12	13	12	4	1	10	2
Charts (#)	50	158	24	40	64	48	48	24	74	64	56	42	802
Avg. C^0 functions	4.2	3.35	2.08	1.25	1.97	2.08	2.31	3.71	4.32	2.03	2.38	5.64	0.17
Avg. C^0 on smooth charts	0.0	0.015	0.045	0.0	0.02	0.0	0.0	0.0	0.017	0.0	0.0	0.063	0.0
Time (s)	3.1	10.2	2.5	3.1	5.1	3.2	3.5	2.2	6.1	5.4	5.6	3.3	34.1

have demonstrated the flexibility and practicability of our approach on approximating subdivision surfaces for discrete meshes and modeling from 3D curve networks.

Real-world objects are usually of complex geometry and topology, but there remains sparsity to be discovered in their representations. Our work is the first attempt to generalize the Sparse-Land [Bruckstein et al. 2009] model onto a manifold.

Limitations. The dictionary allows redundancy, and thus there is much flexibility to choose various atom functions that can represent geometries with different smoothness, e.g., exponential, trigonometric, and splines. Although our current dictionary repository has proved its good performance in representing manifolds with various features, theoretically there is no guarantee for the dictionary to capture all geometric features in any target shapes. This leads to yet another approach for obtaining dictionaries that overcomes this limitation by adopting a learning point of view. The learning method on geometries will be a nice but challenging topic for us in future studies. Our framework of compatible sparse representation has shown its practicality and effectiveness in the construction of manifolds with varying smoothness. Nevertheless, our algorithm needs to solve a global optimization problem for maintaining the compatibility of a manifold. It currently cannot achieve real-time efficiency that is usually requested in local manipulation for the designers. One of the next directions is to speed up the algorithm with real time.

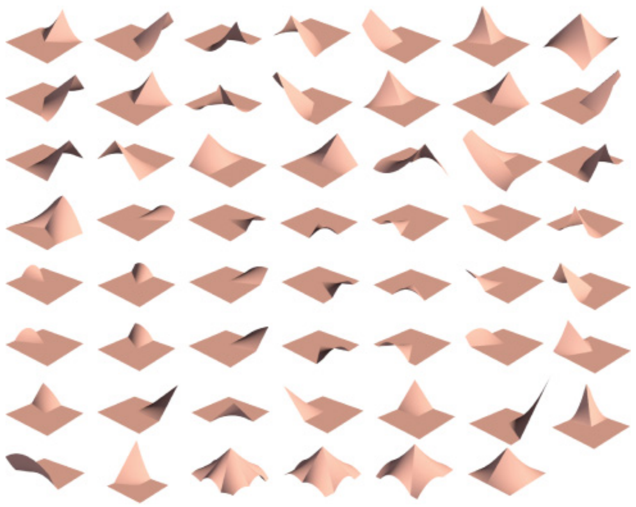
Future work. The Sparse-Land model generalized on the manifold is fascinating because of its universality and flexibility, which make many of the geometric processing tasks clear and simple, and the superior performance to which it leads in various applications. As with sparse coding in image processing, we can also apply our framework of compatible sparse representations to various tasks in geometric processing, e.g., reconstruction, inpainting, denoising, and compression. We believe that some of the extensions are feasible but not straightforward.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their comments and suggestions. The work is supported by the NSF of China (grants 11171322, 11371341, 11526212, 11571338, and 61222206), the One Hundred Talent Project of the Chinese Academy of Sciences, and the Fundamental Research Funds for the Central Universities (WK0010000051).

APPENDIX

The C^0 atom functions are constructed from shape functions in the finite element method (FEM) [Zienkiewicz and Taylor 1977]. Here we use four-node bilinear quadrilateral element shape functions. The general formulation for one node is that $N_1^e = \frac{1}{4}(1-\xi)(1-\eta)$. For the other three nodes, the procedure is the same, traversing the element cyclically. Considering that one chart may have different degree, we generally construct the 55 shape functions with degrees 3 through 7 (which is suitable for most domain manifolds). When the domain manifold has a vertex with degree more than 7, the dictionary repository would add the shape functions with that degree for this specified domain manifold. All functions are shown next.



An adequate statistics table is shown in Table II, containing the number of darts, creases, edges, and charts; the number of C^0 functions used on average; and the number of C^0 functions that show up on smooth charts. Actually, our method cannot guarantee that no C^0 functions are used on smooth charts, as OMP automatically decides what functions are chosen. Experientially,there are few C^0 functions used, and the coefficient is nearly zero.

REFERENCES

Fatemeh Abbasinejad, Pushkar Joshi, and Nina Amenta. 2011. Surface patches from unorganized space curves. *Computer Graphics Forum* 30, 5, 1379–1387.

Seok-Hyung Bae, Ravin Balakrishnan, and Karan Singh. 2008. ILoveSketch: As-natural-as-possible sketching system for creating 3D curve models. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology*. ACM, New York, NY, 151–160.

Seok-Hyung Bae, Ravin Balakrishnan, and Karan Singh. 2009. EverybodyLovesSketch: 3D sketching for a broader audience. In *Proceedings* *of the 22nd Annual ACM Symposium on User Interface Software and Technology*. ACM, New York, NY, 59–68.

Mikhail Bessmeltsev, Caoyu Wang, Alla Sheffer, and Karan Singh. 2012. Design-driven quadrangulation of closed 3D curves. *ACM Transactions on Graphics* 31, 6, 178.

Alfred M. Bruckstein, David L. Donoho, and Michael Elad. 2009. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM Review* 51, 1, 34–81.

Steven A. Coons. 1967. *Surfaces for Computer-Aided Design of Space Forms*. Technical Report. DTIC Document.

Giovanni Della Vecchia and Bert Jüttler. 2009. Piecewise rational manifold surfaces with sharp features. In *Mathematics of Surfaces XIII*. Springer, 90–105.

Giovanni Della Vecchia, Bert Jüttler, and Myung-Soo Kim. 2008. A construction of rational manifold surfaces of arbitrary topology and smoothness from triangular meshes. *Computer Aided Geometric Design* 25, 9, 801–815.

Tony DeRose, Michael Kass, and Tien Truong. 1998. Subdivision surfaces in character animation. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'98)*. ACM, New York, NY, 85–94.

David L. Donoho. 1995. De-noising by soft-thresholding. *IEEE Transactions on Information Theory* 41, 3, 613–627.

Michael Elad. 2010. *Sparse and Redundant Representations*. Springer.

Michael Elad and Michal Aharon. 2006. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing* 15, 12, 3736–3745.

Jean Gallier, Dianna Xu, and Marcelo Siqueira. 2012. Parametric pseudo-manifolds. *Differential Geometry and Its Applications* 30, 6, 702–736.

Cindy Grimm, Tao Ju, Ly Phan, and John Hughes. 2009. Adaptive smooth surface fitting with manifolds. *Visual Computer* 25, 5–7, 589–597.

Cindy Grimm and Denis Zorin. 2006. Surface modeling and parameterization with manifolds. In *ACM SIGGRAPH 2006 Courses*. ACM, New York, NY, 1–81.

Cindy M. Grimm and John F. Hughes. 1995. Modeling surfaces of arbitrary topology using manifolds. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'95)*. ACM, New York, NY, 359–368.

Xianfeng Gu, Ying He, and Hong Qin. 2006. Manifold splines. *Graphical Models* 68, 3, 237–254.

Ying He, Kexiang Wang, Hongyu Wang, Xianfeng Gu, and Hong Qin. 2006. Manifold t-spline. In *Geometric Modeling and Processing—GMP 2006*. Lecture Notes in Computer Science, Vol. 4077. Springer, 409–422.

Wan Chiu Li, Nicolas Ray, and Bruno Lévy. 2006. Automatic and interactive mesh to t-spline conversion. In *Proceedings of the 4th Eurographics Symposium on Geometry Processing (SGP'06)*.

P. Malraison. 2000. *N-sided Surfaces: A Survey*. Technical Report. Defense Technical Information Center.

A. Nasri, M. Sabin, and Z. Yasseen. 2009. Filling n-sided regions by quad meshes for subdivision surfaces. *Computer Graphics Forum* 28, 6, 1644–1658.

J. Cotrina Navau and N. Pla Garcia. 2000. Modeling surfaces from meshes of arbitrary topology. *Computer Aided Geometric Design* 17, 7, 643–671.

Andrew Nealen, Takeo Igarashi, Olga Sorkine, and Marc Alexa. 2007. FiberMesh: Designing freeform surfaces with 3D curves. *ACM Transactions on Graphics* 26, 3, 41.

Hao Pan, Yang Liu, Alla Sheffer, Nicholas Vining, Chang-Jian Li, and Wenping Wang. 2015. Flow aligned surfacing of curve networks. *ACM Transactions on Graphics* 34, 4, 127.

Yagyensh Chandra Pati, Ramin Rezaiifar, and P. S. Krishnaprasad. 1993. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Conference Record of the 1993 27th Asilomar Conference on Signals, Systems, and Computers*. IEEE, Los Alamitos, CA, 40–44.

Scott Schaefer, Joe Warren, and Denis Zorin. 2004. Lofting curve networks using subdivision surfaces. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*. ACM, New York, NY, 103–114.

Elif Tosun and Denis Zorin. 2011. Manifold-based surfaces with boundaries. *Computer Aided Geometric Design* 28, 1, 1–22.

Hongyu Wang, Ying He, Xin Li, Xianfeng Gu, and Hong Qin. 2008. Polycube splines. *Computer-Aided Design* 40, 6, 721–733.

John Wright, Allen Y. Yang, Arvind Ganesh, Shankar S. Sastry, and Yi Ma. 2009. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31, 2, 210–227.

Shiyao Xiong, Juyong Zhangy, Jianmin Zheng, Jianfei Cai, and Ligang Liu. 2014. Robust surface reconstruction via dictionary learning. *ACM Transactions on Graphics* 33, 6, 201.

Linlin Xu, Ruimin Wang, Juyong Zhang, Zhouwang Yang, Jiansong Deng, Falai Chen, and Ligang Liu. 2015. Survey on sparsity in geometric modeling and processing. *Graphical Models* 82, 160–180.

Jianchao Yang, John Wright, Thomas S. Huang, and Yi Ma. 2010. Image super-resolution via sparse representation. *IEEE Transactions on Image Processing* 19, 11, 2861–2873.

Lexing Ying and Denis Zorin. 2004. A simple manifold-based construction of surface of arbitrary smoothness. *ACM Transactions on Graphics* 23, 3, 271–275.

Yixin Zhuang, Ming Zou, Nathan Carr, and Tao Ju. 2013. A general and efficient method for finding cycles in 3D curve networks. *ACM Transactions on Graphics* 32, 6, 180.

Olgierd Cecil Zienkiewicz and Robert Leroy Taylor. 1977. *The Finite Element Method*. Vol. 3. McGraw-Hill London.

Received September 2014; revised October 2015; accepted October 2015