

# Point-Based Multiscale Surface Representation

MARK PAULY  
Stanford University  
LEIF P. KOBBELT  
RWTH Aachen  
and  
MARKUS GROSS  
ETH Zurich

---

In this article we present a new multiscale surface representation based on point samples. Given an unstructured point cloud as input, our method first computes a series of point-based surface approximations at successively higher levels of smoothness, that is, coarser scales of detail, using geometric low-pass filtering. These point clouds are then encoded relative to each other by expressing each level as a scalar displacement of its predecessor. Low-pass filtering and encoding are combined in an efficient multilevel projection operator using local weighted least squares fitting.

Our representation is motivated by the need for higher-level editing semantics which allow surface modifications at different scales. The user would be able to edit the surface at different approximation levels to perform coarse-scale edits on the whole model as well as very localized modifications on the surface detail. Additionally, the multiscale representation provides a separation in geometric scale which can be understood as a spectral decomposition of the surface geometry. Based on this observation, advanced geometric filtering methods can be implemented that mimic the effects of Fourier filters to achieve effects such as smoothing, enhancement, or band-pass filtering.

Categories and Subject Descriptors: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—*Boundary representation, curve, surface, solid, and object representations*

General Terms: Algorithms

Additional Key Words and Phrases: Surface representations, shape modeling, spectral filtering, morphing, scale space, geometric modeling

---

## 1. INTRODUCTION

Shape modeling applications range from CAD for car and airplane design, e-commerce, and medical applications to character design for games and movies, and more. This variety in applications fields is reflected in a great diversity of shape modeling approaches and shape representations. The latter are often tailored to the specific needs of the application in mind. Due to hardware restrictions, most early surface representations were optimized for low memory usage, using higher order approximation

---

This work has been supported by the joint Berlin/Zurich graduate program Combinatorics, Geometry, and Computation, financed by the ETH Zurich and the German Science Foundation (DFG).

Author's address: M. Pauly, Institute for Computational Science, ETH Zurich, IFW C 26.2, Haldeneggsteig 4/Weinbergstrasse, CH-8092 Zurich, Switzerland; email: pauly@inf.ethz.ch.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or direct commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 1515 Broadway, New York, NY 10036 USA, fax: +1 (212) 869-0481, or [permissions@acm.org](mailto:permissions@acm.org).  
© 2006 ACM 0730-0301/06/0400-0177 \$5.00

or interpolation methods. Examples are tensor-product spline surfaces that were introduced in the 60s and are still widely used in commercial CAD systems. When memory became cheap and new specialized graphics chips emerged, polygon-based surfaces, such as triangle meshes and subdivision surfaces, became dominant. Since the global consistency constraints of these surfaces are less restrictive, they are more suited for handling simple as well as complex geometry, in particular, when the shape and/or topology undergoes frequent changes during editing or simulation. Recently, point-based modeling has become popular, taking the trend of reducing the global structure of a surface representation a step further at the cost of using more and simpler modeling primitives. Initially proposed as a rendering primitive [Levoy and Whitted 1985], point samples have also been used for shape and appearance modeling [Pauly et al. 2003; Kalaiah and Varshney 2003; Zwicker et al. 2002]. Point-based representations offer a number of advantages which are mainly the result of their structural simplicity and lack of a global connectivity graph or parameterization. Thus insertion, deletion, or repositioning of point samples is trivial which supports efficient, dynamic adaptation of the local sampling rate. This makes point-based representations particularly suitable for dynamic settings where the geometry of the model and/or the accuracy of the approximation changes frequently. Examples include large geometric deformations in shape modeling [Pauly et al. 2003] or simulation [Carlson et al. 2002; Feldman et al. 2003; Müller et al. 2004; Pauly et al. 2005], or level-of-detail representations [Pauly et al. 2002a].

This work extends previous point-based representations from a single-scale description of the model surface towards a true multiscale shape representation. The concept of scale is fundamental to human perception. Hence scale-space methods have become increasingly popular and widespread in data analysis and modeling (see Lindeberg [1994] or Weickert [1998] for an overview). So far, scale-space techniques have mainly been applied to functional data such as sound, images, or flow fields. This work goes beyond these earlier efforts in that it applies the concept of scale-space to irregularly sampled, discrete manifold surfaces. The main motivation is to enhance the description of a geometric model by explicitly modeling surface detail at different geometric scales. Hence, our multiscale surface representation provides a set of surface approximations at different levels of smoothness. This allows the user to manipulate a model in a flexible and intuitive way without having to pay close attention to whether or not the surface detail is properly affected. Thus the user can freely choose on which level of scale his editing operations should be applied, either coarse global edits on the general shape of the object or modifications on the fine surface detail.

A key issue in the definition of a multiscale surface representation is the interconnection between successive levels in the hierarchy. To obtain intuitive editing semantics, it is essential that changes in one level are propagated naturally to the next higher levels [Kobbelt et al. 1998, 1999]. This can be achieved by encoding each level of the discrete multiscale representation as a normal displacement of its immediate smoother approximation. Thus the difference between two successive levels can be expressed as a set of scalar detail coefficients that approximates for each point the distance between the two levels in normal direction [Kobbelt et al. 1998; Guskov et al. 1999; Zorin et al. 1997; Botsch and Kobbelt 2004]. These detail coefficients can be considered as discrete frequency bands that allow spectral filtering methods to be applied to point-sampled models.

It is important to note the distinction between a multiscale and a multiresolution surface representation. The former describes a surface at different levels of smoothness without any reference to a particular sampling distribution. The latter, on the other hand, refers to a set of surface approximations with varying sampling resolution, thus describing a surface at different levels of coarseness. Multiresolution surface representations have been used successfully in the context of efficient rendering [Rossignac and Borrel 1993], surface compression and progressive transmission [Khodakovsky et al. 2000], surface analysis [Eck et al. 1995; Hubeli and Gross 2001], and morphing [Lee et al. 1999]. The main motivation is to represent a surface at different levels of resolution to handle large

complex models in a scalable way. Recent surface editing systems combine multiscale and multiresolution representations, using, for example, multiresolution subdivision surfaces [Zorin et al. 1997]. However, Kobbelt et al. [1998] observed that rigid multiresolution representations like subdivision hierarchies can lead to less flexibility when defining editing metaphors. In this article we focus on surface editing and filtering so we free ourselves from the restrictions of multiresolution and only build a geometric hierarchy, that is, a multiscale representation. This entails that the full number of degrees of freedom, that is, the number of discrete samples in the original point cloud, is available at all levels.

The construction of a discrete multiscale representation requires two main building blocks:

- a *smoothing operator*, that is, a geometric low-pass filter that generates successively smoother approximation levels of a given input surface, and
- a *decomposition operator*, that is, a method to encode each level relative to the next smoother level to ensure intuitive detail preservation.

Since our method is based on a least squares projection operator, we briefly review this method in Section 2. We will then give a short introduction to scale-space in Section 3 and show how this concept can be generalized to manifold surfaces. In Section 4, we define our discrete multiscale representation by introducing the decomposition and reconstruction operators that are used to switch between subsequent levels in the hierarchy. Section 5 presents different surface smoothing operators based on a discretization of the surface diffusion equation and a least squares Gaussian filter. The implementation of the decomposition operator is described in Section 6 where two alternatives, ray shooting and projection, are discussed. Section 7 shows how smoothing and decomposition operators can be combined into a single operator based on the weighted least squares projection. In Section 8, we show results and applications, focusing on shape modeling using free-form deformation and discrete spectral filtering. We conclude in Section 9 with a detailed discussion and some ideas for future work.

## 2. WEIGHTED LEAST SQUARES APPROXIMATION

Meshless methods have become increasingly popular for defining a smooth surface from a set of point samples [Levin 2003; Alexa et al. 2001; Fleishman et al. 2003; Adamson and Alexa 2003, 2004; Amenta and Kil 2004]. Our method is based on the implicit surface definition proposed in Adamson and Alexa [2003] and Pauly [2003]. Given an unstructured point cloud  $P = \{\mathbf{p}_i = (x_i, y_i, z_i) | 1 \leq i \leq n\}$  as input, a smooth manifold surface  $S_P$  is defined as the zero set of a function

$$f(\mathbf{x}) = \mathbf{n}(\mathbf{x})^T(\mathbf{x} - \mathbf{a}(\mathbf{x})), \quad (1)$$

where  $\mathbf{a}(\mathbf{x})$  is the weighted average of sample points

$$\mathbf{a}(\mathbf{x}) = \frac{\sum_i \mathbf{p}_i \phi_{\mathbf{x}}(\|\mathbf{p}_i - \mathbf{x}\|)}{\sum_i \phi_{\mathbf{x}}(\|\mathbf{p}_i - \mathbf{x}\|)}. \quad (2)$$

The vector  $\mathbf{n}(\mathbf{x})$  is an approximation of the surface normal computed as the direction of smallest weighted covariance by minimizing

$$\sum_{i=1}^n (\mathbf{n}(\mathbf{x})^T(\mathbf{p}_i - \mathbf{a}(\mathbf{x})) \cdot (\mathbf{x}))^2 \phi_{\mathbf{x}}(\|\mathbf{p}_i - \mathbf{x}\|). \quad (3)$$

We use a Gaussian kernel function  $\phi_{\mathbf{x}}(r) = e^{-r^2/h_{\mathbf{x}}^2}$ , where  $h_{\mathbf{x}}$  is an adaptive scale parameter that determines the feature size of the resulting surface in the vicinity of  $\mathbf{x}$  [Pauly et al. 2002a].

Given this implicit definition of the surface  $S_P$ , a projection operator  $\Psi_P(\mathbf{x})$  can be implemented that takes a point  $\mathbf{x}$  near the surface and projects it onto the surface. The operator  $\Psi_P(\mathbf{x})$  is usually

implemented as an iterative procedure that recursively projects a given point  $\mathbf{x}$  onto the least squares plane defined by  $\mathbf{a}(\mathbf{x})$  and  $\mathbf{n}(\mathbf{x})$ . More details can be found in Adamson and Alexa [2004] and Pauly [2003]. Here we just summarize some characteristics of this implicit surface definition that are important to our method.

- The weighted least squares approximation is *meshless*, that is, it does not require any connectivity between point samples or other global information such as a global parameterization.
- Since the Gaussian kernel function diminishes quickly with distance, the region of influence is effectively *local*. This means that only a local neighborhood of point samples needs to be considered in the approximation which leads to an efficient evaluation of the projection operation.
- The resulting surface is *smooth*. The projection operator essentially implements a low-pass filter where the degree of smoothness depends on the scale parameter.
- The approximation is *adaptive*. By incorporating a local sampling density estimate into the kernel function, the region of influence can be adapted to the local distribution of samples in the point cloud, see Pauly et al. [2002a] for details.

Note that this definition of an implicit surface from a cloud of point samples is closely related to the moving least squares method introduced by Levin [2003] (see also Amenta and Kil [2004]).

### 3. SCALE-SPACE

Scale-space methods have been used extensively in image and volume data analysis for the last two decades. The idea is to model a signal at different approximation levels, or scales, to better analyze the inherent structures of the signal. Given a  $d$ -dimensional signal  $f : \mathbb{R}^d \rightarrow \mathbb{R}$ , its linear scale-space representation  $L : \mathbb{R}^d \times \mathbb{R} \rightarrow \mathbb{R}$  is defined as the convolution

$$L(\cdot, t) = f(\cdot) \otimes g(\cdot, t), \quad (4)$$

where  $t$  is the scale parameter and

$$g(\mathbf{x}, t) = \frac{1}{(2\pi t)^{d/2}} e^{-\mathbf{x}^T \mathbf{x} / (2t)} \quad (5)$$

is a Gaussian kernel whose standard deviation  $\sigma_t$  is related to the scale parameter through  $\sigma_t = \sqrt{t}$ . The scale-space representation of Equation (4) is often used in combination with Fourier techniques which allow an efficient evaluation of the convolution operation. The generating equation of a linear scale-space representation  $L$  is the linear diffusion equation [Koenderink 1984] so  $L$  can also be obtained as the solution to a diffusion process

$$\frac{\partial L}{\partial t} - \lambda \cdot \Delta L = 0, \quad (6)$$

where  $\lambda$  is the diffusion constant and  $\Delta$  the Laplacian operator. The initial conditions of Equation (6) are given as  $L(\mathbf{x}, 0) = f(\mathbf{x})$ , suitable boundary conditions depend on the specific application. A detailed discussion of scale-space techniques can be found in Lindeberg [1994] or Weickert [1998].

#### 3.1 Scale-Space for 2-Manifold Surfaces

In this section, we describe how the concept of scale-space can be extended from functions to general surfaces embedded in 3-space. Since the diffusion equation does not require a global parameterization, it can easily be generalized from the functional to the manifold setting. Assume a continuous surface

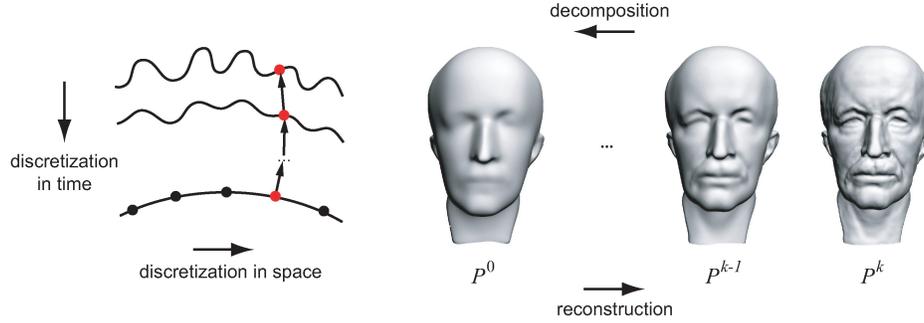


Fig. 1. Discrete multiscale representation. Left: 2D illustration, right: 3D surface model.

$S$  is given. The surface diffusion equation defines a continuous set of evolving surfaces  $S(t)$  subject to

$$\frac{\partial \mathbf{x}}{\partial t} - \lambda \Delta \mathbf{x} = 0, \quad (7)$$

where  $\mathbf{x} \in S(t)$  is a point on the surface,  $t \in [0, \infty) \subset \mathbb{R}$  denotes the time or scale parameter, and  $\Delta \mathbf{x} = \kappa \cdot \mathbf{n}$  is the Laplace-Beltrami operator applied to the coordinate functions with  $\kappa$  the mean curvature and  $\mathbf{n}$  the surface normal at  $\mathbf{x}$  (see do Carmo [1976]). The initial condition to Equation (7) is given as  $S(0) = S$ . Equation (7) defines an evolving surface where each point on the surface moves in the direction defined by the surface normal with a speed given by the mean curvature. This method, also known as mean curvature flow [Dziuk 1991], has been studied in the context of evolving interfaces [Sethian and Osher 1988] and surface fairing [Desbrun et al. 1999; Clarenz et al. 2000]. Note that this extension of scale-space to 2-manifold surfaces does not have the properties of axiomatic scalespace theory such as noncreation of maxima. Mean curvature flow can actually lead to an increase in curvature and even create singularities, for example, when applied to a dumbbell shape (see also Section 5.3).

#### 4. DISCRETE MULTISCALE SURFACE REPRESENTATION

Given the definition of scale-space for manifold surfaces, we can now define our discrete multiscale surface representation. The goal is to represent a given surface  $S$  by a discrete set of point clouds  $\mathbf{P} = \{P^0, \dots, P^k\}$  where each of these point clouds defines an approximation to  $S$  at a different scale level with  $P^0$  as the smoothest approximation. Thus  $\mathbf{P}$  can be understood as a discretization of Equation (7) both in space and in time (or scale), see Figure 1. To define adequate semantics for this discrete multiscale surface representation, subsequent approximation levels need to be encoded relative to each other in a meaningful way. This is done by means of a decomposition operator that encodes a detailed surface as a normal displacement of its smooth approximation. As will be demonstrated, normal displacements ensure intuitive editing semantics and provide a compact representation of surface detail. While the decomposition operator defines the transition from detailed to smooth approximation, the reconstruction operator describes the inverse operation, that is, it adds detail to a smooth surface.

Formally, a discrete multiscale representation for point-sampled surfaces can be defined as follows. Let  $P = \{\mathbf{p}_1, \dots, \mathbf{p}_n\}$  be a point cloud representing a surface  $S$  and  $S(t)$  a continuous multiscale representation of  $S$  as defined by Equation (7). A discrete, point-based multiscale representation of  $S$  is a sequence of point clouds  $\mathbf{P} = \{P^0, \dots, P^k\}$ , such that

—for all  $l \in \{0, \dots, k-1\}$ , there exists a  $t_l \in [0, \infty)$  such that the surface represented by  $P^l$  approximates  $S^l = S(t_l)$  with  $t_l > t_{l+1}$ , where  $t_k = 0$  and  $P^k = P$ ,

- $|P^l| = |P| = n$  for all  $l \in 0, \dots, k$ ,
- for all  $\mathbf{p}_i^l \in P^l$  and all  $l \in \{1, \dots, k\}$ , there exists a  $\mathbf{p}_i^{l-1} \in P^{l-1}$  such that

$$\mathbf{p}_i^l = \mathbf{p}_i^{l-1} + d_i^{l-1} \cdot \mathbf{n}_i^{l-1}, \quad (8)$$

where  $\mathbf{n}_i^{l-1}$  is the surface normal at  $\mathbf{p}_i^{l-1}$  and  $d_i^{l-1}$  a scalar-valued detail coefficient.

Thus each sample  $\mathbf{p}_i \in P$  is represented by a point  $\mathbf{p}_i^0 \in P^0$  plus a sequence of normal displacement offsets  $d_i^0, \dots, d_i^{k-1}$ . To reconstruct the position of  $\mathbf{p}_i^l$  at a certain level,  $l$  the point  $\mathbf{p}_i^0$  is recursively displaced in normal direction, that is,

$$\mathbf{p}_i^l = \mathbf{p}_i^0 + d_i^0 \cdot \mathbf{n}_i^0 + \dots + d_i^{l-1} \cdot \mathbf{n}_i^{l-1}. \quad (9)$$

Let  $\mathbf{D} = \{D^0, \dots, D^{k-1}\}$ , where  $D^l = \{d_1^l, \dots, d_n^l\}$  is the set of detail coefficients at level  $l$ . The reconstruction operator  $+ : (\mathbf{P}, \mathbf{D}) \rightarrow \mathbf{P}$  can be defined by applying Equation (8) for each point of the argument point cloud, such that

$$P^l = P^{l-1} + D^{l-1}. \quad (10)$$

Then the reconstruction of a point cloud  $P^l$  can be written as

$$P^l = P^0 + \sum_{m=0}^{l-1} D^m, \quad (11)$$

which directly corresponds to Equation (9). The inverse of the reconstruction operator is the decomposition operator that determines the detail coefficients of the normal displacement offset between two point clouds. The definition of a discrete multiscale representation implies that each sample point  $\mathbf{p}_i \in P$  is active on all levels. The polygon  $\mathbf{p}_i^0, \dots, \mathbf{p}_i^k$  defines the normal trajectory of  $\mathbf{p}_i$  since each polygon edge  $\mathbf{p}_i^l \mathbf{p}_i^{l+1}$  is aligned to the normal vector  $\mathbf{n}_i^l$ . At each level  $l$ , the point set  $P^l$  defines the model surface  $S^l$  at the corresponding scale with level 0 the smoothest approximation. Note that no subsampling operator is applied to the lower levels since unambiguous reconstruction of the normal trajectory requires that each intermediate point  $\mathbf{p}_i^l$  is present.

To build a discrete multiscale representation from a given point-sampled surface, we thus require a smoothing operator to create successively smoother approximations of the input surface and a decomposition operator to encode subsequent levels in the hierarchy.

## 5. SMOOTHING OPERATOR

In Section 3, scale-space representations have been defined using either a convolution with a Gaussian or as the solution of a diffusion process. These equivalent definitions suggest two different approaches for deriving smoothing operators for discrete surfaces. Traditionally, a discretization of the surface diffusion equation has been used which leads to an iterative scheme for surface fairing (Section 5.1). Alternatively, one can exploit the low-pass filtering effect of the weighted least squares projection which resembles the implementation of a Gaussian convolution filter (Section 5.2).

### 5.1 Surface Smoothing Based on Diffusion

For discrete surfaces, a variety of surface fairing methods have been introduced in recent years based on a discretization of Equation (7) [Dziuk 1988; Taubin 1995; Desbrun et al. 1999; Pinkall and Polthier 1993]. Special attention has to be given to a suitable discretization of the Laplacian as neither global parameterization nor a regular sampling pattern are provided in the surface setting. Taubin [1995] pioneered these surface fairing methods and presented various approximate low-pass filters for triangle

meshes using different discrete approximations of the Laplacian. His approach aims at generalizing signal processing techniques to manifold surfaces. For this purpose, he defines discrete geometric frequencies (see also Gross and Hubeli [2000] and Karni and Gotsman [2000]) as the eigenvectors of the Laplacian matrix which describes the discretization of the Laplacian as a weighted sum of adjacent vertices. This discretization leads to the common iterative explicit Euler integration formula for approximate Gaussian mesh smoothing

$$\mathbf{v}' = \mathbf{v} + \lambda \Delta \mathbf{v}, \quad (12)$$

where  $\mathbf{v}'$  is the new vertex position, and  $\Delta \mathbf{v}$  is some discrete approximation of the Laplacian at vertex  $\mathbf{v}$ . Taubin demonstrates that this iterative scheme attenuates high frequencies while preserving low frequencies and thus implements approximate low-pass filter behavior.

This concept can be extended to point clouds by replacing the one-ring neighborhood relation with a spatial neighborhood relation, for example, the  $k$ -nearest neighbors (see Linsen and Prautzsch [2002] or Pauly et al. [2002a] for a discussion of different point neighborhood relations). The degree of smoothness can be controlled by the parameter  $\lambda$  (see Equation (12)), the number  $m$  of iterations, and the size  $k$  of the local neighborhoods of the sample points. To improve the efficiency of the computations, local neighborhoods should be computed at the beginning of the smoothing operation and cached during the iteration. This also increases the stability of the smoothing filter since it prevents clustering effects due to the tangential drift of sample points (see also Linsen [2001]).

A well known characteristic of iterative Laplacian smoothing is the slow convergence for low geometric frequencies. While high frequencies are attenuated quickly, a substantial number of iterations is required to smooth out lower frequency detail. To alleviate this problem, Kobbelt et al. [1998] introduced multilevel fairing, using a complete v-cycle of alternating smoothing/subsampling, and up-sampling/smoothing steps. This scheme has been generalized to point clouds in Pauly et al. [2002b] where additionally a method for local volume preservation has been presented.

## 5.2 Surface Smoothing Based on Least Squares Filtering

An alternative approach for surface smoothing makes use of the low-pass filter characteristics of the least squares projection operator (see Section 2). A smooth approximation of a given point set  $P$  can be obtained by simply projecting all points of  $P$  onto the implicit surface defined by  $P$ . By adjusting the kernel width  $h$  in  $\phi_{\mathbf{x}}$  (see Equation (3)), different degrees of smoothness can be obtained. In fact,  $h$  can be understood as a continuous scale parameter in the sense of the scale-space representation described previously. This means that the least squares projection replaces the fairing operator for creating the scale-space approximation of a given surface  $S$ .

One difficulty with using a large kernel width in the projection is that large subsets of the point cloud have to be considered in the least squares optimization which quickly leads to excessive computation times. A substantial improvement in performance can be obtained by integrating a decimation operator. Since large kernels imply that each individual point contributes less to the optimization, clusters of points can be replaced by a single point without significant loss of accuracy (see also Alexa et al. [2001]). Thus by first decimating the point cloud, the projection operator can be evaluated much more efficiently. For decimation we use the hierarchical clustering method described in Pauly et al. [2002a] that adaptively separates a given point cloud into clusters, each of which is replaced by its centroid. This averaging step already implements a low pass-filter, thus the method is not prone to resampling artifacts such as surface aliasing. Starting with the original point cloud  $P = P^k$ , clustering builds a hierarchy  $P \rightarrow Q^{k-1} \rightarrow \dots \rightarrow Q^0$  of point clouds with successively lower sampling density. Note that, even though a decimation is applied, the projected point clouds  $P^l = \Psi_{Q^l}(P)$ ,  $l = 0, \dots, k-1$  still have the same number of sample points as  $P$ . The

decimation only affects the intermediate point clouds  $Q^l$  that define the base surface of the projection operator.

A crucial question remains: What is the appropriate decimation rate for a certain kernel width? The qualitative argument given indicates that the larger the kernel width, that is, the smoother the resulting surface should be, the can the decimation rate can be without losing too much accuracy. However, a precise quantitative relation is difficult to formulate. Thus a different approach is chosen that controls the degree of smoothness using the sampling resolution. The kernel width is coupled to the local sampling density, for example, as a constant times the local sample spacing [Pauly et al. 2002a]. Thus instead of specifying a discrete set of kernel widths, the discrete scale-space approximation is defined by a set of sampling resolutions  $n^k, \dots, n^0$  such that  $n^k = n = |P^k|$ ,  $|Q^l| = n^l$ , and  $n^l > n^{l-1}$  for  $1 \leq l \leq k$ . A suitable choice for the  $n^l$  is a geometric series, that is,  $n^{l-1} = \lfloor \gamma^l \cdot n \rfloor$ , where  $\gamma \in (0, 1)$  is the decimation factor. This approach defines a logarithmic number of discrete scale-space levels in the number of input samples  $n$  and corresponds to pyramid algorithms used in multiresolution methods [Zorin et al. 1997].

### 5.3 Discussion

Smoothing operators based on diffusion or the least squares projection operation are only approximate low-pass filters. Since no global, distortion-free parameterization exists for manifolds in general, applying these filters can lead to deformations such as volume shrinkage. Another issue is that the projection tends to become unstable for very large filter widths or very low sampling densities due to successive decimation. For example, consider an approximation of a sphere where the kernel width has been chosen so large that all points on the sphere have about the same weight. It is not clear what the correct least squares plane for such a configuration should be. For practical purposes, however, these problems are less of an issue since the kernel function for the smoothest level in the decomposition is typically not in this region of instability. Note that the use of decimation for the projection essentially follows the same motivation as the multilevel fairing methods of Kobbelt et al. [1998] and Pauly et al. [2002b] as it avoids excessive computation times for the attenuation of low frequencies.

## 6. DECOMPOSITION OPERATOR

Suppose a discrete scale-space approximation  $\mathbf{Q} = \{Q^0, \dots, Q^k\}$  of a surface  $S$  has been computed using geometric low-pass filtering as described in Section 5. To encode  $Q^l$  with respect to  $Q^{l-1}$ , the detail coefficients  $D^{l-1}$  need to be computed. In general, however, it is not possible to find for each  $\mathbf{q}_i^l \in Q^l$  a corresponding sample  $\mathbf{q}_i^{l-1} \in Q^{l-1}$  such that the displacement is in normal direction, that is,

$$(\mathbf{q}_i^{l-1} - \mathbf{q}_i^l) \cdot \mathbf{n}_i^{l-1} = 0. \quad (13)$$

Thus to obtain a discrete multiscale representation  $\mathbf{P} = \{P^0, \dots, P^k\}$  for the surface  $S$ , the surfaces represented by the  $Q^l$  need to be resampled to fulfill the normal displacement criterion. We will describe two alternatives to compute the sequence  $\mathbf{D}$  of sets of detail coefficients and the sequence  $\mathbf{P}$  from the discrete scale-space approximation  $\mathbf{Q}$ . Suppose two subsequent levels  $S^{l-1}$  and  $S^l$  are given, represented by two point clouds  $Q^{l-1}$  and  $Q^l$ , respectively. The goal is to find two point clouds  $P^{l-1}$  and  $P^l$ , approximating the surfaces  $S^{l-1}$  and  $S^l$ , and a set of detail coefficients  $D^{l-1}$  such that  $P^l = P^{l-1} + D^{l-1}$ .

### 6.1 Bottom-Up Encoding by Ray-Shooting

One approach is to shoot a ray from each  $\mathbf{q}_i^{l-1} \in Q^{l-1}$  in normal direction and find the intersection point  $\mathbf{r}_i^l \in S^l$  with the surface  $S^l$ . Then the corresponding detail coefficient is given as

$$d_i^{j-1} = \|\mathbf{r}_i^l - \mathbf{q}_i^{l-1}\|. \quad (14)$$



Fig. 2. Multiscale encoding: The point cloud in the middle is obtained as a normal displacement of the point cloud on the left. The right image shows the corresponding detail coefficients, where blue indicates maximum negative displacement and red maximum positive displacement for outward pointing normals.

Furthermore,

$$P^{l-1} = Q^{l-1} \quad (15)$$

and

$$P^l = \{\mathbf{r}_i^l | i = 1, \dots, n\}. \quad (16)$$

This means that the point cloud  $Q^{l-1}$  representing the smooth surface  $S^{l-1}$  is left unchanged, while the point cloud  $Q^l$  representing the detailed surface  $S^l$  is re-sampled to fulfill the normal displacement condition. This method of ray-shooting has been applied successfully in previous mesh-based approaches, for example, in Guskov et al. [2000] to build a normal mesh hierarchy. An algorithm for intersecting a ray with the implicit surface defined in Section 2 has been introduced in Adamson and Alexa [2003]. In this method, a point on the ray is iteratively projected onto the surface until it converges to a point both on the ray and the surface.

## 6.2 Top-Down Encoding by Projection

The second alternative would be to start with a point  $\mathbf{q}_i^l \in Q^l$  and project it onto the surface  $S^{l-1}$ . Thus a point  $\mathbf{r}_i^{l-1} = \Psi_{Q^{l-1}}(\mathbf{q}_i^l) \in S^{l-1}$  is obtained, where  $\Psi_{Q^{l-1}}$  is the least squares projection operator with respect to the point cloud  $Q^{l-1}$ . This yields

$$d_i^{j-1} = \|\mathbf{q}_i^l - \mathbf{r}_i^{l-1}\|. \quad (17)$$

Furthermore,

$$P^{l-1} = \{\mathbf{r}_i^{l-1} | i = 1, \dots, n\} \quad (18)$$

and

$$P^l = Q^l. \quad (19)$$

Here the smooth surface is resampled, while the detailed point cloud remains unaltered.

Figure 2 illustrates the encoding of two subsequent levels. Note that the projection operator as defined in Section 2 is not strictly orthogonal, that is, the direction of projection does not coincide exactly with the surface normal. As described in Adamson and Alexa [2004], a strictly orthogonal projection can be obtained by restricting the projection direction to the gradient of the implicit function  $f$ . This method is computationally more involved, however. In practice, the simple projection procedure leads to satisfactory results since the deviation from the true normal is negligible when projecting onto the

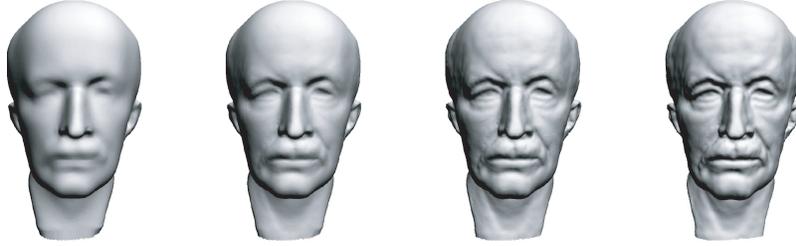


Fig. 3. Linear blend between two subsequent levels of a multiscale representation.

low-pass filtered surface  $S^{l-1}$ . Compared with ray shooting, the top-down is in general more stable since the surface  $S^{l-1}$  is a smooth approximation of the surface  $S^l$ . Thus resampling  $S^{l-1}$  will lead to fewer sampling artifacts such as aliasing, compared to resampling of  $S^l$ .

### 6.3 Continuous Representation

Note that, even though the sequence of point clouds  $\mathbf{P} = \{P^0, \dots, P^k\}$  defines a discrete sample along the scale axis, a continuous scale-space approximation can be obtained by interpolation. For example, a linear blend between two successive levels  $P^{l-1}$  and  $P^l$  can be defined as

$$P^l(\alpha) = P^{l-1} + \alpha \cdot D^{l-1}, \quad (20)$$

where  $\alpha \in [0, 1]$  is the blending parameter,  $P^l(0) = P^{l-1}$ , and  $P^l(1) = P^l$ . This corresponds to a linear blend between each individual sample described as

$$\mathbf{p}_i^l(\alpha) = \mathbf{p}_i^{l-1} + \alpha \cdot d_i^{l-1} \cdot \mathbf{n}_i^{l-1}. \quad (21)$$

Figure 3 illustrates a linear blend between two subsequent levels.

## 7. INTEGRATED SMOOTHING AND DECOMPOSITION OPERATOR

Recall that the construction of a multiscale representation requires an alternating series of two basic operations: A low-pass filter (smoothing operator) to obtain a smoother approximation of a given surface and an encoding step (decomposition operator) to establish the connection between two successive levels. We have shown how the latter can be implemented using the least squares projection method and, similarly, in Section 5.2, it has been demonstrated how the same projection can be used for surface smoothing. Thus we can combine the low-pass filter and encoding steps into a single operator.

The complete construction of the multiscale representation then proceeds as follows (see Figure 4). We start with the original point cloud  $P^k = P$  and apply the decimation operator to obtain  $Q^{k-1}$ . Using an appropriate Gaussian kernel, we then project all points of  $P^k$  onto the implicit surface defined by  $Q^{k-1}$ . This yields a low-pass filtered version of  $P^k$ , the point cloud  $P^{k-1}$ . We then obtain the detail coefficients  $D^{k-1}$  by computing the point-wise Euclidean distance as in Equation (17). Now we can iterate this procedure of decimation and projection until the desired level of smoothness is achieved.

The cost for the construction of the multiscale representation is dominated by the least squares projections. In comparison, the decimation typically requires less than 5% of the total computation time. Since a single projection is applied for low-pass filtering and decomposition, the total number of projections is  $k \cdot n$ , where  $k$  is the number of discrete scale levels and  $n = |P|$  is the number of point samples. Note that the successive decimation of the point cloud does not reduce the number of required projections, since each point cloud  $P^l$  has the same number of sample points. Instead, the efficiency of the projection is greatly increased for higher smoothness levels since fewer samples have to be considered in the least squares optimization.

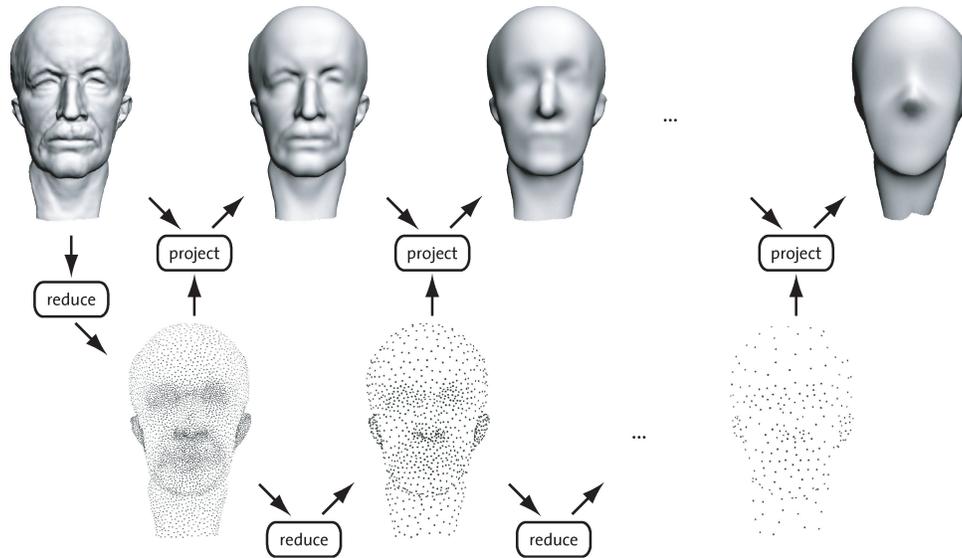


Fig. 4. Building a discrete multiscale representation.

Still, since the full projection is rather costly, the computational effort for the construction of a multiscale representation is quite significant. However, once this representation is built, reconstruction of individual levels is very efficient since it only requires the points to be displaced in the normal direction according to the detail coefficients.

As previously mentioned above, the multiscale point cloud representation is a discrete sample of the continuous representation both in space and in scale. In this context, the low-pass filter determines the sampling in the scale dimension by controlling the smoothness of the approximation. The decomposition algorithm, on the other hand, needs to find the base points on the smoother level to define the detail coefficients and thus determines the sampling in the spatial dimension.

## 8. RESULTS AND APPLICATIONS

This section presents various applications for the multiscale representation. The focus is on surface editing and filtering, other applications will be addressed in the future work section.

### 8.1 Multiscale Free-Form Deformation

As discussed in the introduction, one of the main motivations for defining a multiscale surface representation, is the separation of geometric detail from the overall shape of an object. This supports high-level editing semantics for shape modeling where the user does not have to focus on correct handling of surface detail over all scales. We illustrate the advantage of our representation for the example of free-form deformation [Sederberg and Parry 1986], a widely-used shape modeling paradigm in current 3D modeling systems. By defining a space warping function  $F : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ , a given surface is deformed by displacing each surface point according to this deformation function. To specify  $F$ , we apply the method presented in Pauly et al. [2003] where a simple painting metaphor is used to compute the warping function.

Figure 5 illustrates the difference between single-scale and multiscale deformation on a synthetic example of a plane with sixteen little handles (a). We define a deformation that lifts a circular region in the center of the plane in normal direction, using a Gaussian blending function. The result of this

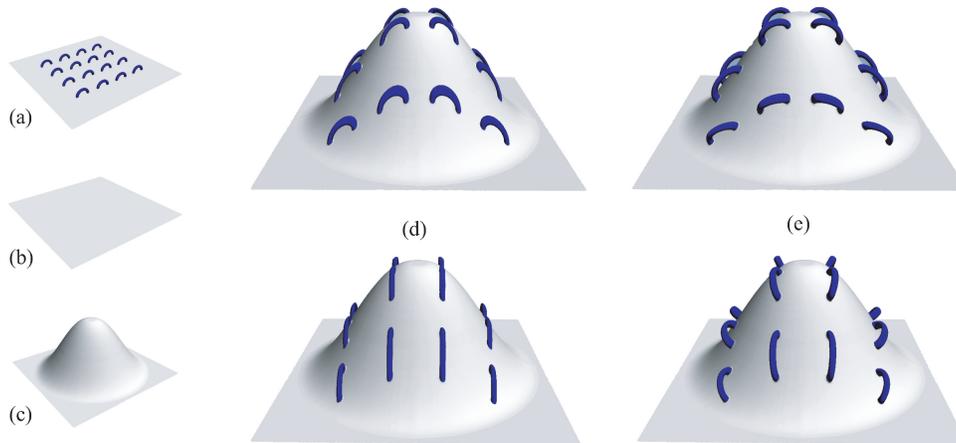


Fig. 5. Multiscale vs. single-scale modeling. (a) original surface, (b) smooth base domain, (c) deformed base domain, (d) single-scale deformation, (e) multiscale deformation.

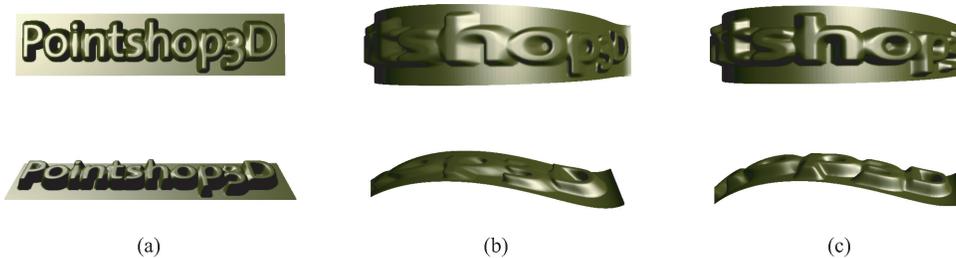


Fig. 6. Multiscale vs. single-scale modeling. (a) original surface, (b) single-scale deformation, (c) multiscale deformation.

deformation is shown in (c). Note that, since all points are displaced in the direction parallel to the plane normal, the blue handles are no longer orthogonal to the base surface. In a multiscale setting, we first encode the surface of (a) onto the plane shown in (b) using the projection method discussed in Section 6.2. This plane is then deformed under the same deformation function as for (c), and the surface detail is reconstructed using normal displacements. As a result, the surface detail, that is, the blue handles, are still orthogonal to the base surface. Note also that the difference in genus between two levels in the multiscale decomposition ((a) and (b)) does not pose problems as long as the smooth approximation still allows unambiguous reconstruction of normals for reconstruction.

A second example is shown in Figure 6. The surface of (a) has been smoothly deformed in a single-scale fashion in (b) and using the multiscale representation in (c). Observe that, as a consequence of the deformation, the embossed text in (b) has been sheared and is no longer orthogonal to the surface base plane. A more intuitive result is obtained in (c) where the embossing is still in normal direction which yields a more natural deformation compared to the single-scale operation.

## 8.2 Multiscale Surface Filtering

As a second application, we consider surface filtering. Recall that a multiscale representation  $\mathbf{P} = \{P^0, \dots, P^k\}$  is built by successively applying low-pass filters with wider filter kernels. Since the detail coefficients  $D^l$  encode the difference between two subsequent levels, they can be understood as the result of a band-pass filter applied to the original point cloud  $P^k$ . Thus the detail coefficients can be considered

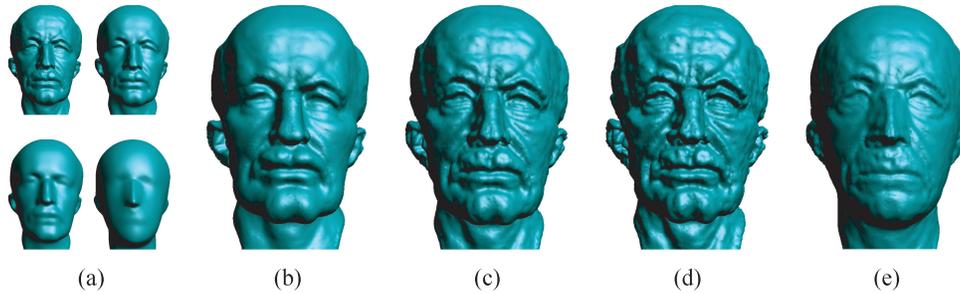


Fig. 7. Spectral filtering on the Max Planck model. The left images show the multiscale decomposition, the filtered surfaces are shown on the right. Scaling coefficients from left to right: (2, 2, 1), (1, 1, 2), (1, 1, 3), (0, 0, 1).

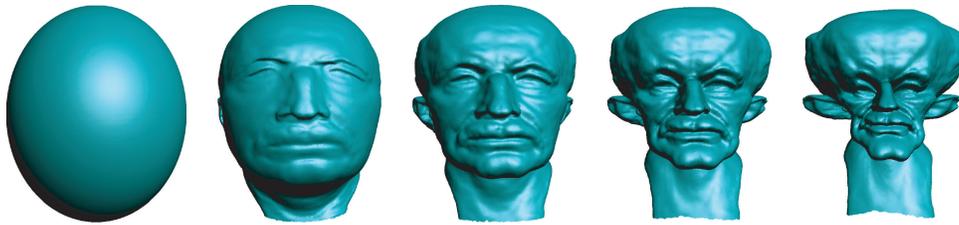


Fig. 8. Encoding the Max Planck model onto a sphere. From left to right, detail coefficients have been scaled to 0.0, 0.5, 1.0, 1.5, and 2.0.

as the spatial representation of a spectral band. However, since the low-pass filters are not ideal, the spectral bands are overlapping and cannot be used to determine the coefficients of a spectral transform as in, for example, the Fourier transform. Nevertheless, the decomposition still allows spectral filters to be implemented by scaling the detail coefficients accordingly.

Figure 7 shows various filters that modify the sample positions of a given input point cloud. In (a), a 4-level multiscale decomposition  $\mathbf{P} = \{P^0, P^1, P^2, P^3\}$  is shown. The filtered models are obtained by scaling the detail coefficients, that is, as  $P^0 + a_0D^0 + a_1D^1 + a_2D^2$ , where  $(a_0, a_1, a_2)$  define the scaling factors. In (b), an enhancement of the lower frequencies has been obtained using the scaling factors (2, 2, 1). Similarly, enhancement of higher frequencies is shown in (c) and (d) using (1, 1, 2) and (1, 1, 3) as scaling factors, respectively. Finally, in (e), spectral bands have been removed using (0, 0, 1), leading to a band-stop filter effect. These filtering effects would be very difficult to achieve without the multiscale decomposition.

The multiscale surface representation defines a decomposition of a point-sampled surface into discrete frequency bands. Note that, even though many analogies to Fourier analysis exist, and many concepts of Fourier theory serve as a motivation for the techniques presented, this representation should not be understood as an extension of the Fourier transform to discrete surfaces. At most, the multiscale approach allows the implementation of various filtering methods that mimic the characteristics of Fourier-based spectral filters.

As indicated, the multilevel decomposition operator introduced in Section 4 does not require any correspondence of the sampling patterns between two subsequent levels as long as the two surfaces are geometrically close. This can be exploited for advanced filtering applications that use entirely different models to create a decomposition. Figure 8 shows such an example where the Max Planck model has been encoded onto an ellipsoid. Scaling the detail coefficients leads to different filtering effects. For scaling factors between zero and one, a blend between the ellipsoid and the Max Planck model can

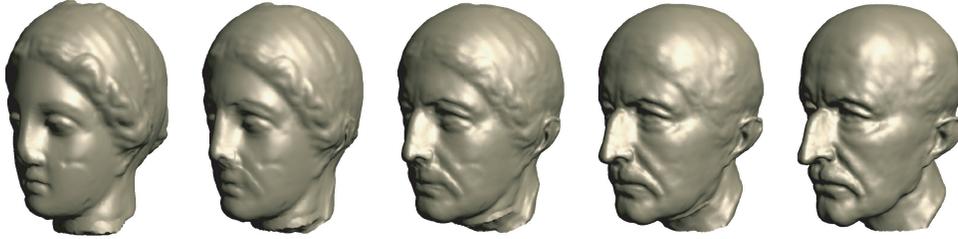


Fig. 9. Morphing the Igea and Max Planck models. The scaling factors  $(\alpha, \beta)$  of Equation (22) vary from left to right as  $(1.0, 0.0)$ ,  $(0.75, 0.25)$ ,  $(0.5, 0.5)$ ,  $(0.25, 0.75)$ ,  $(0.0, 1.0)$ .

be obtained. Scaling factors larger than one lead to enhancement effects similar to Figure 7. In this example, where the base domain is not a low-pass filtered version of the detailed surface, our system currently relies on the user to pick a suitable coarse scale model (the ellipsoid in the Figure 8). If this model differs significantly from the encoded model or contains high curvature regions, self-intersections are likely to occur.

### 8.3 Morphing

The multiscale filtering method can also be adapted to implement simple morphing operations (see Alexa et al. [2000] or Lee et al. [1999] for more general morphing methods). Assume that two surfaces  $S_1$  and  $S_2$  represented by two point clouds  $P_1$  and  $P_2$  are given. The goal is to find a point cloud  $P_M$  that describes a linear blend between the two. For this purpose a common base surface  $S_B$  defined by  $P_B$  has to be found such that  $S_B$  is geometrically close to both  $S_1$  and  $S_2$ . In a first step,  $P_1$  and  $P_2$  are projected onto  $S_B$ , yielding two point clouds  $P'_1$  and  $P'_2$ . To implement a morphing operation between the two models, a one-to-one correspondence of the sample points needs to be defined. This can be achieved by interpolating the detail coefficients of  $P'_2$  onto the sampling distribution of  $P'_1$  or vice versa. A linear blend of the two surfaces can then be obtained using

$$\mathbf{p}_i(\alpha, \beta) = \mathbf{p}_i + (\alpha \cdot d_i^1 + \beta \cdot d_i^2) \cdot \mathbf{n}_i, \quad (22)$$

where  $\alpha, \beta \in [0, 1]$  with  $\alpha + \beta = 1$ ,  $\mathbf{p}_i \in P'_1$ ,  $\mathbf{n}_i$  is the surface normal at  $\mathbf{p}_i$ ,  $d_i^1$  is the detail coefficient of model  $P'_1$ , and  $d_i^2$  is the interpolated detail coefficient of model  $P'_2$ . For interpolation, a simple scheme based on  $k$ -nearest neighbors can be used, see also Pauly et al. [2003].

Note that this method is only guaranteed to work correctly if both input models are already geometrically close and at least one of the two surfaces can be defined by a functional mapping from the common base domain. Also the method cannot handle rotations which limits the scope of morphing operations that can be performed.

Figure 9 shows an example of a morph between the Igea and Max Planck models where both surfaces have first been scaled to a common bounding box. Then the models have been aligned manually to create a rough correspondence of surface feature points. The base domain has been chosen as the maximum ellipsoid that fits into the common bounding box. As Figure 10(a) illustrates, the scaling factors are not restricted to a convex combination but can be chosen arbitrarily to enable more flexible blending operations.

In Figure 10(b), we show an application where the user can directly paint the scaling factors of the detail coefficients onto the surface. To blend between edited and unmodified surface, we use a smooth fall-off function to compute a convex combination between the filtered and unfiltered positions, similar to an alpha mask.

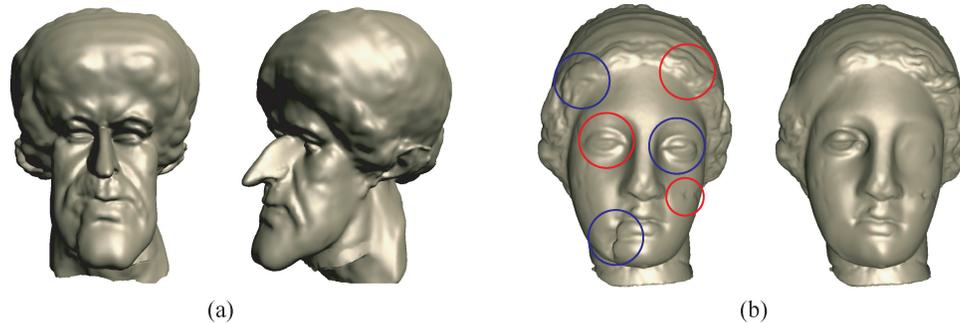


Fig. 10. (a) blend of the Igea and Max Planck models. The scaling factors  $(\alpha, \beta)$  of Equation (22) have been set to  $(1.0, 1.0)$ . (b) Interactive painting of scaling factors on the Igea model. In areas marked by blue circles, the scaling factors have been reduced, leading to a smoothing effect. Areas encircled in red have been enhanced using scaling factors larger than one.

## 9. CONCLUSIONS AND FUTURE WORK

We have presented a new multiscale surface representation for discrete surfaces. This shape description represents a given model at different levels of smoothness, explicitly encodes the geometric detail between these levels, and uses normal displacements for intuitive and consistent reconstruction. Based on weighted least squares approximation, we combine smoothing and decomposition operators into a single projection which allows efficient construction of the hierarchy. The main benefit of our approach is the decoupling of shape and detail at various scales. We show the usefulness of this description for shape modeling using free-form deformation and spectral filtering applications.

We found that point primitives are a natural representation for defining a discrete multiscale surface model. Using the meshless projection method, connectivity information is not required at any stage of the decomposition or reconstruction process. Freeing ourselves from the consistency constraints of a global connectivity graph even brings a number of advantages. Since the base points of the encoding, that is, the points on which a sample of the detailed surface is projected, are not attached to a fixed triangle mesh, we do not require point in triangle location. Self-intersections during encoding which would lead to flipped triangles are much less of an issue.

A challenging question is the combination of multiscale representations with boolean operations for shape modeling. The problem is that when two objects are combined using a union, difference, or intersection operation, the topology of the resulting surface can be very different depending on the level of the multiscale representation on which the operation has been performed. The only solution we see at this stage is to restrict boolean operations to those cases where consistent topology across all levels is guaranteed.

One problem that does not occur in the functional setting is that the modifications of the detail coefficients can lead to self-intersections which would corrupt the model surface. Our current implementation does not include a mechanism for detecting self-intersection. However, Botsch and Kobbelt [2003] presented such a method which might be adapted to our setting.

The concept of scale space has been used extensively for feature detection on functional data. Our representation could be used in the same context (see also Pauly et al. [2003]); it would be interesting, for example, to track features across different scales to measure persistence [Edelsbrunner et al. 2002]. This might also be useful to determine the discrete levels of the decomposition, that is, the points on the scale axis where an explicit encoding is performed. By separating the scale axis into intervals of strong feature persistence, we could obtain a natural separation of the geometric detail across scale. This

means that, whenever a large number of features vanishes or disappears, a new level in the hierarchy would be inserted.

Since the least squares filter with radially symmetric Gaussian kernel is isotropic, features are not preserved, and the directional aspect of features is ignored. It would be interesting to extend our scheme using anisotropic smoothing operators, for example, the bilateral filters presented in Fleishman et al. [2003] and Jones et al. [2003].

## REFERENCES

- ADAMSON, A. AND ALEXA, M. 2003. Approximating and intersecting surfaces from points. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*. Eurographics Association, 230–239.
- ADAMSON, A. AND ALEXA, M. 2004. On normals and projection operators for surfaces defined by point sets. In *Proceedings of the Eurographics Symposium on Point-Based Graphics*. Eurographics Association, 150–155.
- ALEXA, M., BEHR, J., COHEN-OR, D., FLEISHMAN, S., LEVIN, D., AND SILVA, C. T. 2001. Point set surfaces. In *Proceedings of the Conference on Visualization*. IEEE Computer Society Press, 21–28.
- ALEXA, M., COHEN-OR, D., AND LEVIN, D. 2000. As-rigid-as-possible shape interpolation. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, 157–164.
- AMENTA, N. AND KIL, Y. J. 2004. Defining point-set surfaces. *ACM Trans. Graph.* 23, 3, 264–270.
- BOTSCH, M. AND KOBBELT, L. 2003. Multiresolution surface representation based on displacement volumes. *Comput. Graph. Forum* 22, 483–491.
- BOTSCH, M. AND KOBBELT, L. 2004. An intuitive framework for real-time freeform modeling. *ACM Trans. Graph.* 23, 3, 630–634.
- CARLSON, M., MUCHA, P. J., VAN HORN, III, R. B., AND TURK, G. 2002. Melting and flowing. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. ACM Press, 167–174.
- CLARENZ, U., DIEWALD, U., AND RUMPF, M. 2000. Anisotropic geometric diffusion in surface processing. In *Proceedings of the Conference on Visualization*. IEEE Computer Society Press, 397–405.
- DESBRUN, M., MEYER, M., SCHRÖDER, P., AND BARR, A. H. 1999. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, 317–324.
- DO CARMO, M. P. 1976. *Differential Geometry of Curves and Surfaces*. Prentice Hall.
- DZIUK, G. 1988. Finite elements for the beltrami operator on arbitrary surfaces. Partial differential equations and calculus of variations. S. Hildebrandt, R. Leis, Eds. *Lecture Notes in Mathematics* vol. 1357, 142–155.
- DZIUK, G. 1991. An algorithm for evolutionary surfaces. *Numerische Mathematik* 58, 6, 603–611.
- ECK, M., DE ROSE, T., DUCHAMP, T., HOPPE, H., LOUNSBURY, M., AND STUETZLE, W. 1995. Multiresolution analysis of arbitrary meshes. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, 173–182.
- EDELSBRUNNER, H., LETSCHER, D., AND ZOMORODIAN, A. 2002. Topological persistence and simplification. In *Proceedings of the 41st IEEE Symposium on the Foundations of Computer Science*. 454–463.
- FELDMAN, B. E., O'BRIEN, J. F., AND ARIKAN, O. 2003. Animating suspended particle explosions. *ACM Trans. Graph.* 22, 3, 708–715.
- FLEISHMAN, S., COHEN-OR, D., ALEXA, M., AND SILVA, C. T. 2003. Progressive point set surfaces. *ACM Trans. Graph.* 22, 4, 997–1011.
- FLEISHMAN, S., DRORI, I., AND COHEN-OR, D. 2003. Bilateral mesh denoising. *ACM Trans. Graph.* 22, 3, 950–953.
- GROSS, M. AND HUBELI, A. 2000. Eigenmeshes. Tech. rep., Department of Computer Science, ETH Zurich.
- GUSKOV, I., SWELDENS, W., AND SCHRÖDER, P. 1999. Multiresolution signal processing for meshes. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, 325–334.
- GUSKOV, I., VIDIMCE, K., SWELDENS, W., AND SCHRÖDER, P. 2000. Normal meshes. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, 95–102.
- HUBELI, A. AND GROSS, M. 2001. Multiresolution feature extraction from unstructured meshes. In *Proceedings of the Conference on Visualization*. IEEE Computer Society Press.
- JONES, T. R., DURAND, F., AND DESBRUN, M. 2003. Non-iterative, feature-preserving mesh smoothing. *ACM Trans. Graph.* 22, 3, 943–949.
- KALAIHAH, A. AND VARSHNEY, A. 2003. Statistical point geometry. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*. Eurographics Association, 107–115.

- KARNI, Z. AND GOTSMAN, C. 2000. Spectral compression of mesh geometry. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, 279–286.
- KHODAKOVSKY, A., SCHRÖDER, P., AND SWELDENS, W. 2000. Progressive geometry compression. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, 271–278.
- KOBBELT, L., CAMPAGNA, S., VORSATZ, J., AND SEIDEL, H.-P. 1998. Interactive multi-resolution modeling on arbitrary meshes. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, 105–114.
- KOBBELT, L., VORSATZ, J., AND SEIDEL, H.-P. 1999. Multiresolution hierarchies on unstructured triangle meshes. *Computat. Geometry J. Theory Appl.*, 5–24.
- KOENDERINK, J. J. 1984. The structure of images. *Biological Cybernetics* 50, 363–370.
- LEE, A. W. F., DOBKIN, D., SWELDENS, W., AND SCHRÖDER, P. 1999. Multiresolution mesh morphing. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, 343–350.
- LEVIN, D. 2003. Mesh-independent surface interpolation. In *Geometric Modeling for Scientific Visualization*. Springer-Verlag, 37–50.
- LEVOY, M. AND WHITTED, T. 1985. The use of points as display primitives. Tech. rep., Department of Computer Science, University of North Carolina at Chapel Hill.
- LINDBERG, T. 1994. *Scale-Space Theory In Computer Vision*. Kluwer Academic Publishers.
- LINSEN, L. 2001. Point cloud representation. Tech. rep., Faculty of Computer Science, University of Karlsruhe.
- LINSEN, L. AND PRAUTZSCH, H. 2002. Fan clouds—an alternative to meshes. In *Proceedings of Dagstuhl Seminar on Theoretical Foundations of Computer Vision—Geometry, Morphology, and Computational Imaging*.
- MÜLLER, M., KEISER, R., NEALEN, A., PAULY, M., GROSS, M., AND ALEXA, M. 2004. Point based animation of elastic, plastic and melting objects. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '04)*. ACM Press, 141–151.
- PAULY, M. 2003. Point primitives for interactive modeling and processing of 3d geometry. Ph.D. thesis, Computer Science Department, ETH Zurich.
- PAULY, M., GROSS, M., AND KOBBELT, L. P. 2002a. Efficient simplification of point-sampled surfaces. In *Proceedings of the Conference on Visualization*. 163–170.
- PAULY, M., KEISER, R., ADAMS, B., DUTRE, P., GROSS, M., AND GUIBAS, L. J. 2005. Meshless animation of fracturing solids. *ACM Trans. Graph.* 24, 3, 957–964.
- PAULY, M., KEISER, R., AND GROSS, M. 2003. Multi-scale feature extraction on point-sampled surfaces. In *Comput. Graph. Forum.* 22, 281–289.
- PAULY, M., KEISER, R., KOBBELT, L. P., AND GROSS, M. 2003. Shape modeling with point-sampled geometry. In *ACM Trans. Graph.* 22, 641–650.
- PAULY, M., KOBBELT, L., AND GROSS, M. 2002b. Multiresolution modeling of point-sampled geometry. Tech. rep., Department of Computer Science, ETH Zurich.
- PINKALL, U. AND POLTHIER, K. 1993. Computing discrete minimal surfaces and their conjugates. *Experim. Math.* 2, 1, 15–36.
- ROSSIGNAC, J. AND BORREL, P. 1993. Multi-resolution 3d approximations for rendering. *Model. Comput. Graph.*, 455–465.
- SEDERBERG, T. W. AND PARRY, S. R. 1986. Free-form deformation of solid geometric models. In *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, 151–160.
- SETHIAN, J. A. AND OSHER, S. 1988. Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations. *J. Computat. Physics* 79, 12–49.
- TAUBIN, G. 1995. A signal processing approach to fair surface design. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, 351–358.
- WEICKERT, J. 1998. *Anisotropic Diffusion in Image Processing*. Teubner-Verlag.
- ZORIN, D., SCHRÖDER, P., AND SWELDENS, W. 1997. Interactive multiresolution mesh editing. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, 259–268.
- ZWICKER, M., PAULY, M., KNOLL, O., AND GROSS, M. 2002. Pointshop 3d: an interactive system for point-based surface editing. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press, 322–329.

Received October 2003; revised June 2004; accepted September 2005