Progressive Point Set Surfaces

Shachar Fleishman
and
Marc Alexa
and
Daniel Cohen-Or
and
Cláudio T. Silva

Progressive point set surfaces (PPSS) are a multilevel point-based surface representation. They combine the usability of multilevel scalar displacement maps (e.g. compression, filtering, geometric modeling) with the generality of point-based surface representations (i.e. no fixed homology group or continuity class). The multiscale nature of PPSS fosters the idea of *point-based modeling.* The basic building block for the construction of PPSS is a projection operator, which maps points in the proximity of the shape onto local polynomial surface approximations. The projection operator allows the computing of displacements from smoother to more detailed levels. Based on the properties of the projection operator we derive an algorithm to construct a base point set. Starting from this base point set, a refinement rule using the projection operator constructs a PPSS from any given manifold surface.

Categories and Subject Descriptors: I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling: Object hierarchies, Boundary representations; G.1.2 [**Mathematics of Computing**]: Approximation: Approximation of surfaces and contours

General Terms: Algorithms

Additional Key Words and Phrases: Moving least squares, Point-based modeling, Surface representation and reconstruction

1. INTRODUCTION

Point sets are emerging as a surface representation. The particular appeal of point sets is their generality: every shape can be represented by a set of points on its boundary, where the degree of accuracy typically depends only on the number of points. Point sets do not have a fixed continuity class or are limited to certain homology groups as in most other surface representations. Polygonal meshes, in particular, have a piecewise linear C^0 geometry, resulting in an unnatural appearance. To overcome the continuity problem, research has been devoted to image space smoothing techniques (e.g. Gouraud shading),

Author's address: Shachar Fleishman, Tel Aviv University; email: shacharf@cs.tau.ac.il;
Marc Alexa, TU Darmstadt; email: marc.Alexa@gris.informatik.tu-darmstadt.de;
Daniel Cohen-Or, Tel Aviv University; email: dcor@post.tau.ac.il;
Cláudio T. Silva, OGI School of Science and Engineering; email: csilva@cse.ogi.edu;

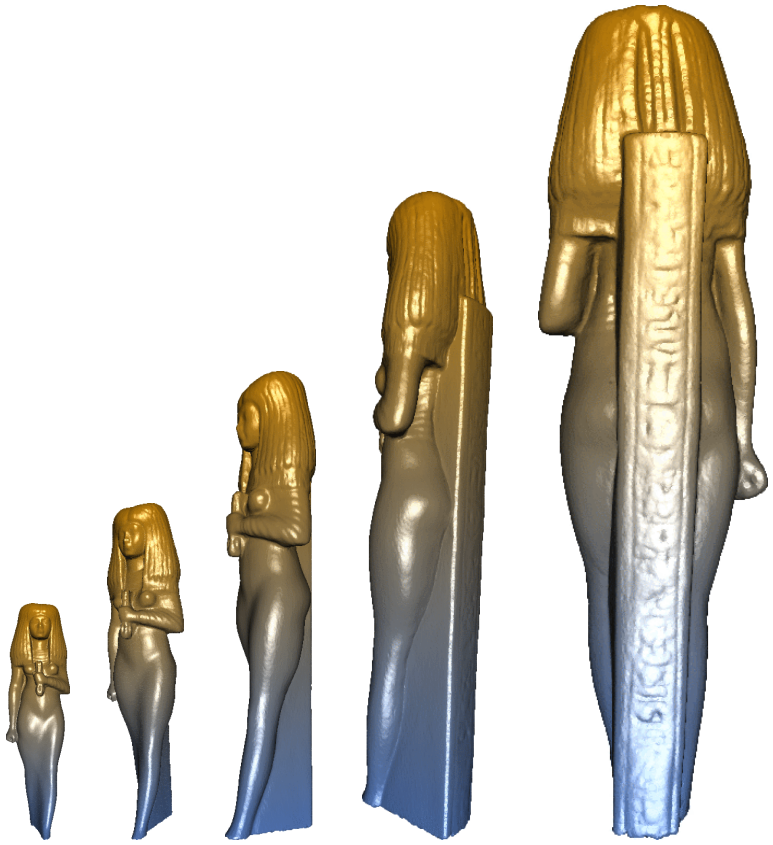


Fig. 1. The progressive series of the Isis model, on the left a model with 19K points progressively refined up to 189K points in the model on the right.

or procedures to smooth the model's geometry such as subdivision surfaces.

To define a manifold from the set of points, the inherent spatial interrelation among the points is exploited as implicit connectivity information. A mathematical definition or algorithm attaches a topology and a geometric shape to the set of points. This is non-trivial since it is unclear what spacing of points represents connected or disconnected pieces of the surface. Moreover, most surfaces are manifold, which limits the possibilities of using functions for global interpolation or approximation. Recently, Levin gave a definition of a manifold surface from a set of points [Levin 2000], which was used in [Alexa et al. 2001] to render shapes.

To achieve a certain geometric fidelity many points are needed to describe a shape. The necessary uniformity of the point's density might further increase the number of points. The relation between point density and accuracy calls for the definition of levels of detail and the notion of progressiveness. That is, the point set should have a base set that represents a coarse and a smooth version of the surface, which can be refined by a series of point insertions in the spirit of progressive meshes [Hoppe 1996], yet more analogous to MAPS [Lee et al. 1998; Guskov et al. 2000].

Progressive or multi-scale representations are useful not only to cut down on the amount of data but also for modeling and visualization purposes (See Figure 1). This is because of the connection between detail levels and spectral bands [Kobbelt et al. 1998; Kobbelt et al. 1998; Zorin et al. 1997]. Ideally, the geometric representations of the levels are relative to each other and independent of the position and orientation of the shape. This is achieved by decomposing the geometric components into a normal and a tangential direction, and encoding each level as a displacement of a coarser level [Khodakovsky et al. 2000; Guskov et al. 2000]. Furthermore, tangential components can be described implicitly by the refinement rule so that a single scalar per point is sufficient to encode the shape. Note that this also leads to an efficient geometry compression scheme.

This approach has been described and analyzed for mesh geometry using subdivision techniques by [Guskov et al. 2000; Lee et al. 2000]. Based on the method of moving least squares (MLS) presented in [Alexa et al. 2001; Levin 2000], in this paper we define a projection operator and a refinement rule. Together, they allow us to refine a given base point set towards a reference point set (input model). The projection operator defines a local tangential coordinate frame, which allows us to specify the position of inserted points, with a scalar representing the normal component. The tangential components are defined by the refinement rule. As such, the scheme is reminiscent of subdivision techniques for meshes.

Based on the properties of the refinement process we develop a simplification scheme for point sets to construct a base point set, which represents a smoother version of the original shape. The base point set is then refined by point insertion to add levels of detail. The surface could be refined adaptively creating point sets with densities varying with respect, say, to the viewing parameters or the local geometric behavior of the surface. In this paper we:

—introduce a point-based geometric modeling technique that is based on the MLS projection mechanism.

—present a progressive scheme for point set surfaces, where the levels of an object represent both coarse–to–fine and smooth–to–detailed hierarchy.

—use a local operator that allows accurate computation of local differential surface properties.

—apply an encoding scheme for compressing progressive point sets.

In Section 2, we proceed with a presentation of related work. We define the MLS surface and how to compute it in Section 3. In Section 4, we describe the progressive point set surfaces in detail. Section 5 describes how a progressive point set surface is encoded and discusses its efficiency. Results are shown in Section 6. We end with a discussion on the progressive point set surface and conclude in Sections 7 and 8.

2. RELATED WORK

Our work is related to the recent research efforts in developing point-based representations for shapes [Grossman and Dally 1998; Pauly and Gross 2001; Pfister et al. 2000; Rusinkiewicz and Levoy 2000]. Most of the mentioned techniques are targeted at fast processing and rendering of large point-sampled geometry. Our techniques are focused on advancing the “modeling” of primitives with points. In this respect our work fits into the field of *Digital Geometry Processing* (DGP) [Desbrun et al. 1999; Guskov et al. 1999;

Guskov et al. 2000; Lee et al. 1998; Taubin 1995], which aims at extending standard signal processing concepts to the manifold surface domain. Pauly and Gross [2001], have shown how to extend these techniques to point-based representations by collecting sets of points to patches, over which the points define an irregularly sampled function. Most approaches for meshes also construct a multiresolution representation by progressively refining a base domain and exploiting the connection of the refinement levels to spectral properties. Meshes are generally composed of two parts: the connectivity of the mesh; and the geometry (i.e., the position of the vertices). Few DGP techniques can be directly applied to such a representation (one example is the pioneering work presented in [Taubin 1995]). DGP algorithms require parameterization of the surface, which could be represented in mesh form as a subdivision surface [Lee et al. 1998; Kobbelt et al. 1999; Eck et al. 1995].

A recent work related to our surface representation is the work of Carr et al. [2001] which reconstructs 3D objects by fitting a global radial basis function (RBF) to point clouds. An RBF forms a solid model of an object, allowing analytic evaluation of surface normal, direct rendering and iso-surface extraction, similar to the properties of the surface representation we use.

Several different hierarchical representations have been proposed for geometric objects. Object simplification [Cignoni et al. 1994; Cohen et al. 1996; He et al. 1996; Hoppe et al. 1993; Zhou et al. 1997] is often used to generate a hierarchical representation, which could be used for many purposes, e.g. rendering [Duchaineau et al. 1997; El-Sana and Varshney 1999; Xia et al. 1997].

Linsen [2001] also describes a multiresolution representation of point-based objects. Similarly to our method, the detail points are inserted using a prediction operator. In our work we focus on a space efficient progressive representation of a point set. Another recent related work is the one by Pauly et al. [2002], which describes a number of point-based simplification methods. The method we present here for building the base point set is similar to their clustering method.

A leading technique for representing hierarchical meshes is *Progressive Meshes* [Hoppe 1996], a mesh representation of varying resolution where a series of edge-split operations progressively refines a base mesh up to the original resolution of the source. This representation motivates solutions to mesh simplification, progressive transmission and loading from a disk or from a remote server. A number of mesh compression and streaming techniques are based on this concept [Cohen-Or et al. 1999; Pajarola and Rossignac 2000; Taubin et al. 1998]. For a recent survey of mesh simplification and compression techniques see [Gotsman et al. 2001].

Subdivision surfaces [Catmull and Clark 1978; Warren and Weimer 2001] are defined by a topological refinement operator and a smoothing rule. Given a mesh of arbitrary topology, they refine the mesh towards a smooth limit surface. Point set surfaces are similar to subdivision surfaces, in that it is possible to add points to the defined smooth surface without additional information. However, to describe an arbitrary surface using subdivision techniques, inserted points need to be displaced. Normal Meshes [Guskov et al. 2000] as well as Displaced Subdivision Surfaces [Lee et al. 2000] demonstrate this idea of a multiresolution subdivision mesh where vertices are displaced by a single scalar value in the normal direction. These approaches are attractive since a single scalar value is easier to manipulate or store. The underlying concept could be understood as decomposing

the surface representation into a tangential and a normal component (see [Guskov et al. 2000]). Note that we use the same idea, however without coding the tangential component explicitly as the mesh connectivity but implicitly as point proximity.

3. MLS SURFACES

The MLS surface S_P of a set of points $P = \{\mathbf{p_i}\}, \mathbf{p_i} \in \mathbb{R}^3, i \in \{1, \ldots, N\}$ is defined implicitly by a projection operator. To project a point $\mathbf{r}$ onto S_P two steps are necessary: First, a local reference domain $H = (\mathbf{n}, d)$, where $\mathbf{d}$ is the origin of the plane and $\mathbf{n}$ is the normal to the plane is computed. Then, a local bivariate polynomial is fitted over H to the point set. More precisely, the local reference domain $H = \{\mathbf{x} | \langle \mathbf{n}, \mathbf{x} \rangle - \mathbf{d} = 0, \mathbf{x} \in \mathbb{R}^3\}, \mathbf{n} \in \mathbb{R}^3, \|\mathbf{n}\| = 1$ is determined by minimizing

$$\sum_{i=1}^{N} \left(\langle \mathbf{n}, \mathbf{p_i} - \mathbf{r} - t\mathbf{n} \rangle \right)^2 e^{-\|\mathbf{p_i} - \mathbf{r} - t\mathbf{n}\|^2 / h^2} \tag{1}$$

in all normal directions $\mathbf{n}$ and offsets t. Here $\mathbf{d} = \mathbf{r} + t\mathbf{n}$. A local coordinate system over H is defined by taking the standard basis for $\mathbb{R}^3$, e_1, e_2, e_3 and compute a rotation matrix M such that $n = M \cdot e_3$. The local coordinate system is defined by $(M \cdot e_1, M \cdot e_2)$. This rotation matrix is not uniquely defined. For our purposes, any solution will do as long as we use the same procedure to compute M in all of our computations.

Let $\mathbf{q}$ be the projection of $\mathbf{p_i}$ onto H, and f_i the height of $\mathbf{p_i}$ over H, i.e $f_i = \mathbf{n} \cdot (\mathbf{p_i} - \mathbf{q})$. The polynomial approximation g is computed by minimizing the weighted least squares error

$$\sum_{i=1}^{N} \left(g(x_i, y_i) - f_i \right)^2 e^{-\|\mathbf{p_i} - \mathbf{r} - t\mathbf{n}\|^2 / h^2}. \tag{2}$$

The projection of $\mathbf{r}$ is given by

$$MLS(\mathbf{r}) = \mathbf{r} + (t + g(0, 0))\mathbf{n}. \tag{3}$$

Formally, the surface S_P is the set of points ($\Omega \subset \Re$) that project onto themselves. In this definition, h is the anticipated spacing of the points. Thus, the point set together with an adequately chosen value for h defines the surface. As part of the projection procedure, we not only determine the position on the surface where a point projects to, but we also obtain high-order derivative information analytically, which can be used to accurately determine normal, curvature, etc. Detailed description on computing the reference domain and polynomial is presented in [Alexa et al. 2003] In the following, S_X will generally be the surface with respect to a point set X defined by the above procedure. We will call this the MLS-surface of X.

4. PROGRESSIVE POINT SET SURFACES

First, we give an overview of the concept of progressive point set surfaces and then explain the details in the following sections.

Given a reference (input) point set $R = \{\mathbf{r_i}\}$ defining a reference surface S_R. The point set R is reduced by removing points to form a base point set $P_0 \subset R$. This base point set defines a surface S_{P_0} which differs from S_R. Next, we will resample the surface adding more points so that the difference between our surface and S_R decreases.

The base point set P_0 is refined by inserting additional points yielding the set P_1. The refinement operator first inserts points independent of the reference set R, which means $P_1 \not\subset R$. Then, the inserted points are displaced so that the difference between the surfaces decreases, i.e. $d(S_{P_1}, S_R) < d(S_{P_0}, S_R)$.

This process is repeated to generate a sequence of point sets P_i with increasing size and decreasing difference from the reference surface. A progressive point set surfaces is defined as the MLS surface of $\mathcal{P} = P_0, P_1, \ldots$. Where each point set P_i is encoded by the (scalar) displacements of inserted points, yielding a compact representation of the progressive point set surface.

In the following we explain the necessary steps for this procedure in detail. We denote the reference domain plane of a point p with respect to a point set S as $H_S(p)$ and the polynomial is similarly denoted by $g_S(p)$.

4.1 The refinement operator

Let R be the reference point set as before and P be the point set to refine. The set P is refined by generating additional points $A = \{\mathbf{a_j}\}$, which are sampled in the local neighborhoods of the $\mathbf{p_i}$.

More specifically, let the local reference domain $H_p(\mathbf{p_i})$ of a point $\mathbf{p_i}$ (see Figure 2a) be determined as the local minimum of Eq. (1) with respect to the points in P. The reference plane $H_p(\mathbf{p_i})$ is sampled regularly at intervals ρ, yielding a set of (u, v) coordinates for the points $\mathbf{a'_j}$ on the plane. These points are placed in the neighborhood of the surface using the polynomial $\overline{g} = g_p(\mathbf{p_i})$ (defined by Eq. (2)), yielding the set of points $\mathbf{a_j} = (u, v, \overline{g}(u, v))$.

To find and encode the positions of the additional points, the plane $H_p(\mathbf{a_j})$ of $\mathbf{a_j}$ is computed, and then two local polynomial fits are computed on the basis of the given reference domain $H_p(\mathbf{a_j})$: The first polynomial g_p is with respect to the points in P and the second, g_r, is with respect to the points in R. The height of a point $\mathbf{a_j}$ is given as $g_r(0, 0)$, i.e. on the local polynomial fit to the points in the reference set (see Figure 2b).

Since the coordinate (u, v) is generated implicitly by specifying the regular sampling parameter ρ, only the height has to be encoded. Based on Eq. (3), the height of $\mathbf{a_j}$ can be expressed as the difference:

$$\Delta = g_r(0, 0) - g_p(0, 0). \tag{4}$$

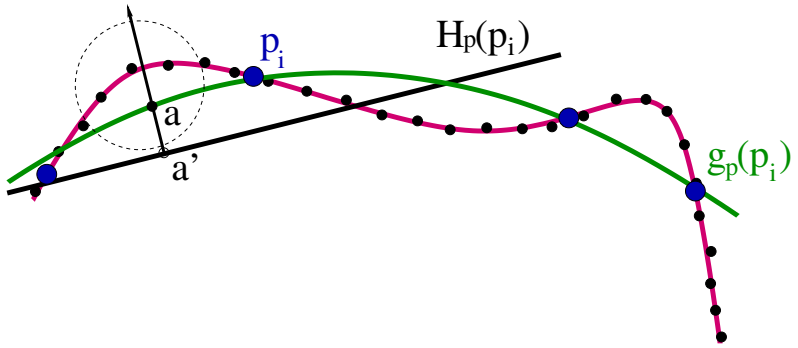
Since the distance between g_r and g_p is significantly smaller than the distance between g_r and the reference plane, the Δ can be efficiently encoded.

The regular sampling pattern has to be adapted to avoid oversampling and sampling of empty regions. We introduce two criteria for deciding whether to insert a point. A point $\mathbf{a_i}$ is inserted only if

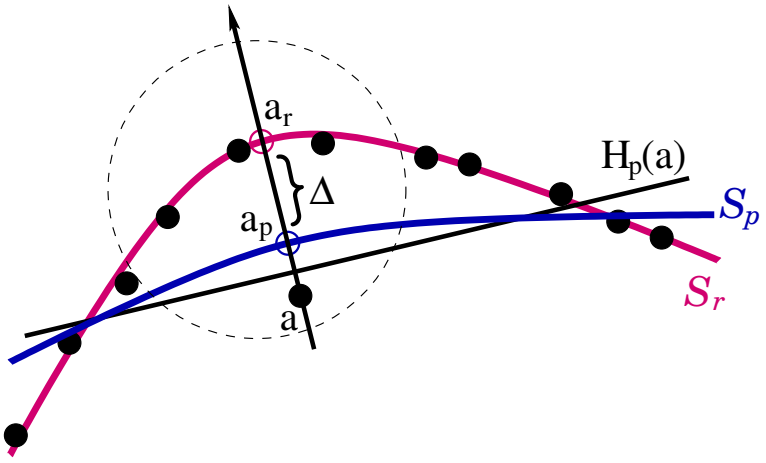
(1) none of the already inserted points $P \cup \{\mathbf{a_j}, j < i\}$ is closer than ρ and
(2) it is in the convex hull of points in P closer than h or, more formally, iff $\mathbf{a_i} \in \mathcal{CH}\{\mathbf{p_i} | \|\mathbf{p_i} - \mathbf{a_i}\| < h\}$.

The first criterion avoids oversampling, and the second aims at detecting boundaries of the manifold by defining the extent of the local neighborhood.

The sampling parameter ρ can be used to specify the refinement convergence. If ρ is halved in every refinement step the number of points approximately quadruples from one point set to the next. However, ρ can also be used to adapt the sampling density to the



(a)



(b)

Fig. 2. An illustration of the point set refinement process: (a) a new point a is generated in the neighborhood of the surface of the current level (the blue points). The reference plane $H_p(p_i)$ and polynomial $\overline{g} = g_p(p_i)$ of p_i are computed. By scanning the neighborhood of p_i, a new point a' on $H_p(p_i)$ is generated and projected on the polynomial $\overline{g}$, i.e. $a = \overline{g}(a')$. In (b), a is projected on MLS_p (the blue curve) by computing its reference plane $H_p(a)$ and polynomial $g_p(a)$. Next, a is projected again on S_R (defined by the black points) using the same reference plane, but with the appropriate polynomial $g_r(a)$. Finally, the detail value $\Delta = a_r - a_p$ is computed.

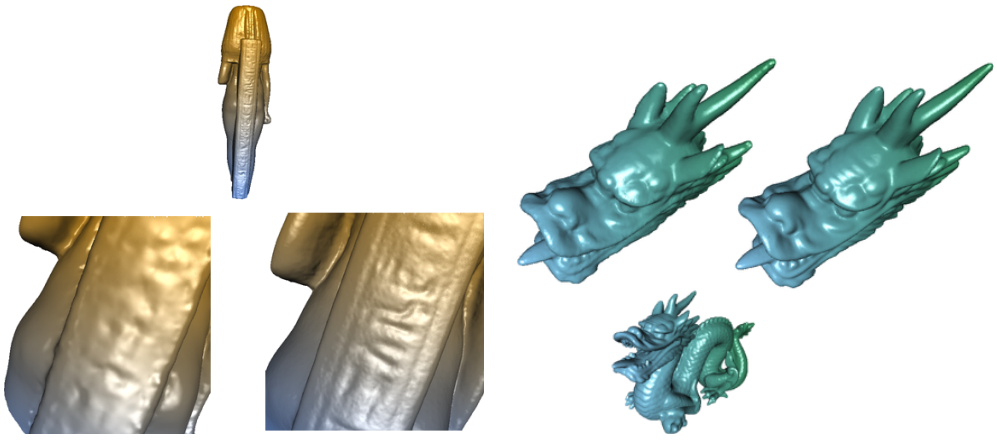


Fig. 3. Close-ups of smooth renderings for different levels in the hierarchy of a PPSS. The base level of a PPSS defines a smooth surface, which is visualized here by upsampling the smooth surface by adding points without displacing them. Note that higher levels add the missing details to the PPSS representation.

application needs, e.g. visible regions of the surface or local curvature if piecewise linear approximations are needed.

As the point density increases from one refinement level to the next, also h should be adapted. We adapt h to the change in ρ, for example, if ρ is halved in every step, so is h. We assume, however, that a suitable h is given for the base point set. This is discussed in the following section.

4.2 Constructing the base point set

The refinement operator uses local reference domains on the basis of the reduced point set to compute polynomial fits to the reference point set. This requires the local reference domain of a point $\mathbf{p_i}$ with respect to the points in P and to the points in R to be about the same. We use this requirement as a criterion for reducing the reference point set to the base point set.

Given a neighborhood size h and a maximum deviation ϵ, let Q_i be the point set which results from removing the points in a h-neighborhood around $\mathbf{r_i}$, i.e.

$$Q_i = \{\mathbf{r_k} | \|\mathbf{r_k} - \mathbf{r_i}\| > h\}. \tag{5}$$

A point $\mathbf{r_i}$ can be used in the base point set if its original reference domain $H_R(\mathbf{r_i})$ is close to the reference domain $H_{Q_i}(\mathbf{r_i})$ with respect to the reduced point set Q_i. More specifically, the distance between $H_R(\mathbf{r_i})$ and $H_{Q_i}(\mathbf{r_i})$ is measured as the scalar product between their normal components (see Section 3).

In practice, the points in R are visited in random order. If $\mathbf{r_i}$ can be included in P_0, all points in the h-neighborhood around $\mathbf{r_i}$ are discarded for inclusion in P_0. The process terminates after all points were tested.

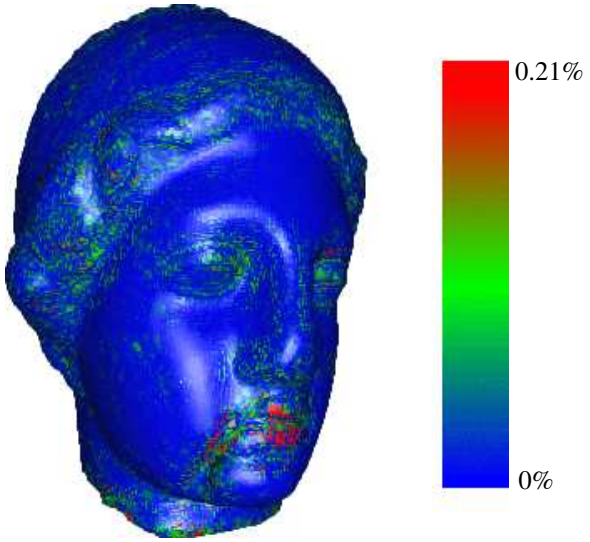


Fig. 4. A color-coding of the magnitude of the displacement for the Venus model. Note that smooth regions have small displacements, while regions containing fine detail need larger displacements The magnitude of the displacements essentially corresponds to the energy in the respective frequency band.

5. THE PPSS ENCODING

Figure 4 illustrates the magnitudes of the displacements of the progressive set. Observing that the vast majority of the displacements are of small magnitudes gives raise to a space efficient encoding scheme. To create an encoding of P_{i+1} given P_i, we perform the routine described in Section 4.1. New points are generated and projected both on S_{P_i} and on S_R. The difference between the two projections is the *displacement* denoted by Δ. Since the distance between the two surfaces is small, the displacements are merely the details of the surface and can be encoded in a small number of bits.

To decode P_{i+1}, a reverse procedure is applied. New points are generated as described in the encoding procedure. For each point $\mathbf{r}$, the reference plane and polynomial are computed using Eqns. (1),(2). Then the point is projected using a modified Eq. (3) as follows:

$$MLS(\mathbf{r}) = \mathbf{r} + (t + g(0,0) + \Delta)\mathbf{n}. \tag{6}$$

For efficient storage, we quantize the displacement values to a user-specified accuracy. An error bound defines the maximal tolerated error with respect to the diagonal of the object's bounding box. The range of the displacements and the error bound defines the number of bits required to properly represent the displacements. Recall that the decoding procedure is highly dependent on performing the exact same procedure that the encoder performed. Quantizing the values and reconstructing them creates minor differences that may lead to somewhat different sets of points that are added to each level. If this occurs the encoder and decoder may no longer be synchronized. Therefore, in the encoding process the displacements are quantized to guarantee that the decoder generates the exact set of points that is encoded.

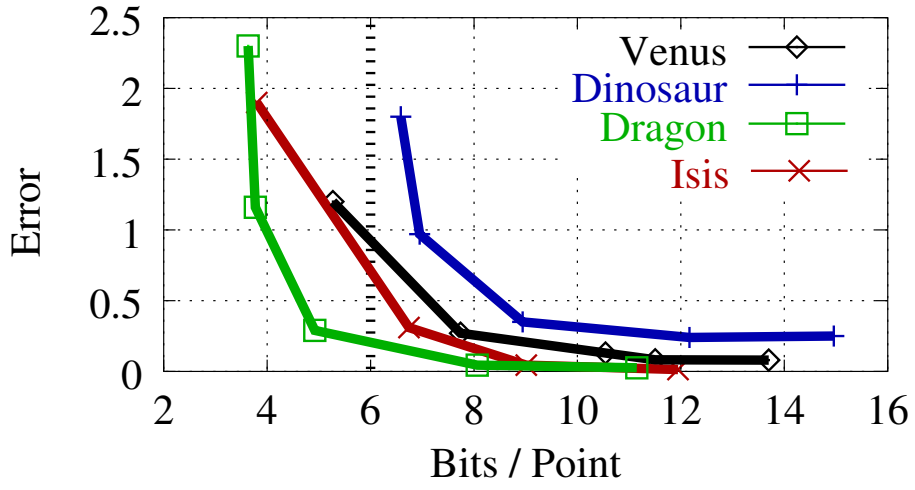


Fig. 5. Rate distortion curve: shows a comparison of size vs. accuracy achievable with our compression method. The error is measured on a scale of 10^{-4} of the bounding box of the model.

Name	Points	Points in P_0	bits / displacement (error 10^{-4}) 0.001	0.01	0.1	0.5
Dragon	437K	16K	9.7 (0.026)	6.6 (0.044)	3.4 (0.29)	2.4 (1.6)
Isis	187K	9K	9.9 (0.01)	6.9 (0.04)	4.7 (0.31)	1.8 (1.9)
Venus	134K	7K	9.2 (0.08)	8.3 (0.13)	5.5 (0.27)	3.0 (1.2)
Dino	56K	4K	10.9 (0.25)	8.1 (0.24)	4.9 (0.35)	2.9 (0.97)

Table I. Achieved bit-rates for given error bounds. The user can specify an error bound on the displacement values. Depending on the error bound, a quantization scheme is chosen, which influences the number of bits necessary to encode the displacements. The small number of quantization levels typically results in a systematically smaller error as compared to the error bound (shown in parentheses).

6. RESULTS

We have implemented the progressive point set representation as described in the previous sections and applied it to several models. Table I summarizes the results by showing the average number of bits per displacement required with respect to error tolerance. The error is expressed with respect to the diagonal of the bounding box of the given model. Note that the error is merely subject to the quantization applied to the displacements. For the models in Table I, five (for the dino model) to seven (for the dragon model) levels were generated. Our experiments as shown in Table II and Figures 5 and 7 suggest that an average of five bits per point yields a pleasing visual quality. The base point set P_0 is compressed by triangulating the points using the BPA [Bernardini et al. 1999] and applying a mesh compression tool [Gotsman et al. 2001]. The time to compress the models in Table I range

Inserted points	**Bits / Δ**	**Total points**	**Total size**	**Average b/p**	**error (10^{-4})**
476370	2.3	493142	139K	3.66	2.7
472330	2.4	489102	144K	3.77	1.6
440907	3.5	457679	192K	4.92	0.29
439652	6.6	456424	364K	8.06	0.044
439840	9.7	456612	534K	11.1	0.026

Table II. A comparison of the size vs. accuracy for the Dragon model. Each line shows the size of the model as a function of the number of bits used for quantization of the displacement values. The base point set contains 16772 points compressed to 37.4 bits / point.

from several minutes to 120 minutes on Pentium™ III 1Ghz, our code was not optimized and computes the MLS reference plane using non-linear optimization.

Table II shows the compression achieved by varying the number of bits for the displacement values. We measured the accuracy of the reconstructed model in the spirit of Metro [Ciampalini et al. 1997], i.e. by sampling distances in normal direction. To measure the distance between an MLS surface S_1 defined by a set of points P_1 and the reference MLS surface S defined by P, we sample arbitrary points in the neighborhood of S_1, and use the MLS projection procedure to project each point on S_1 and on S. The average difference between the two projections is the error.

We compared the PPSS–based compression technique with techniques for multiresolution mesh compression. In particular, we have used Khodakovsky's mesh compression technique [Khodakovsky et al. 2000] to generate meshes with increasing accuracy with respect to a reference mesh. The vertices of the reference mesh were used to build a PPSS. Since we do not have connectivity information the BPA was used to generate meshes from the point sets. The resulting meshes have been compared to the reference using Metro. Figure 6 shows a visualization of the results. Note that the PPSS does not fit the piecewise linear geometry of the reference mesh (as the mesh compression technique) but the MLS surface defined by the vertices. This adds some bias to the resulting error for the PPSS.

Figure 7 shows a series of progressively refined point sets, where the shaded images are rendered by an upsampling procedure [Alexa et al. 2001], which requires no displacements. The rendering performs a local refinement of the surface around each point of the model. The images in the second column of Figure 8 are rendered with the above upsampling method and the images in the third column are rendered using the OpenGL™ glPointSize function. Since the MLS surface is continuous and smooth, the quality of the upsampled renderings is higher than a splat rendering.

The MLS surface is smooth and as such does not reconstruct sharp features (since it is not able to model discontinuities in its derivatives). While reconstructing point samples of CAD models with sharp features (see Figure 9, those sharp features are smoothed out, while the rest of the object is reconstructed faithfully. Our method deals with boundaries by computing the convex hull of the neighborhood of a point (as previously described in Section 4.1). For sharp edges in boundaries like the rectangular hole in Figure 9c and d, our method converges to round corners with radius of the size of the input feature size, that is, the spacing between points.

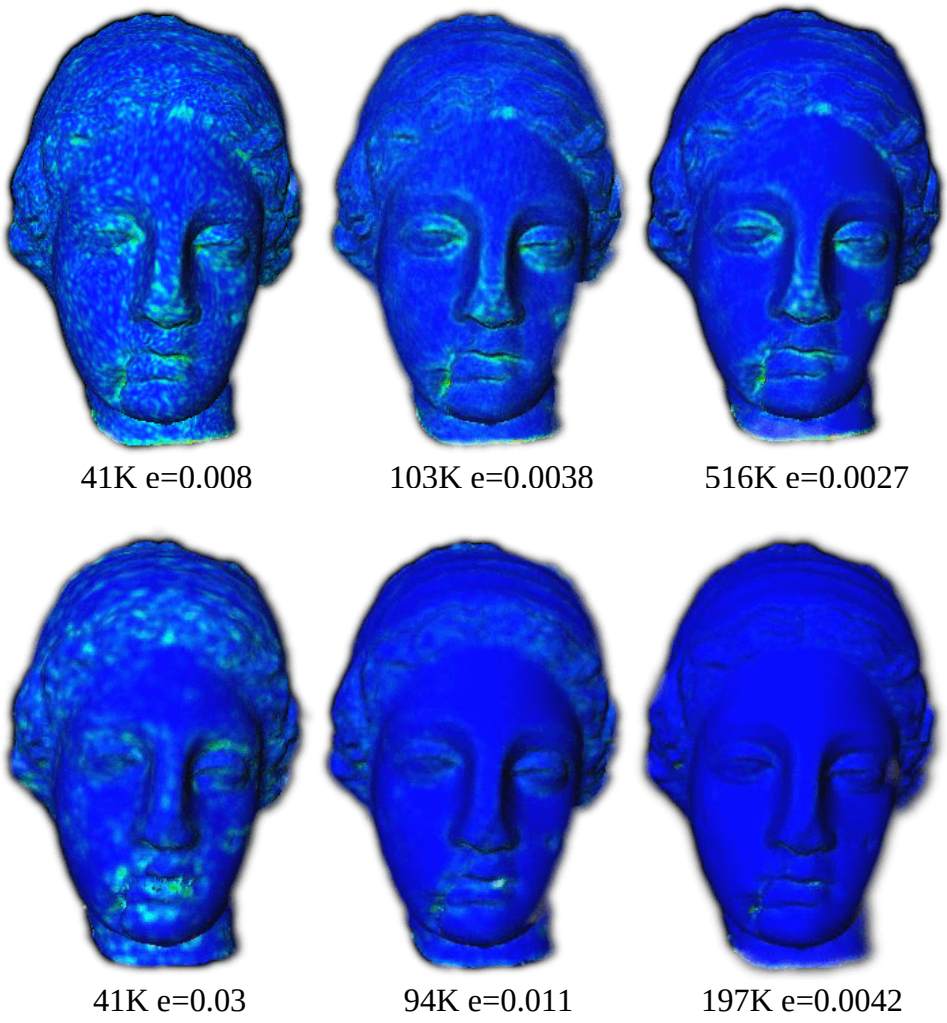


Fig. 6. Comparison of meshes using Metro. The top row displays several steps during the refinement process of Khodakovsky's algorithm. The numbers below the figures show their size and mean error with respect of the bounding box of the object, as reported by the I.E.I-CNR Metro tool. The bottom row displays three meshes reconstructed from a PPSS and compared to the input mesh. Note that the PPSS does not fit the reference mesh but rather the smooth MLS-surface over the vertices. The point sets are triangulated using BPA to be able to apply Metro. Color ranges from blue to green with respect to the error. Note that the Metro tool normalizes the color map to the maximal error of the model being colored.

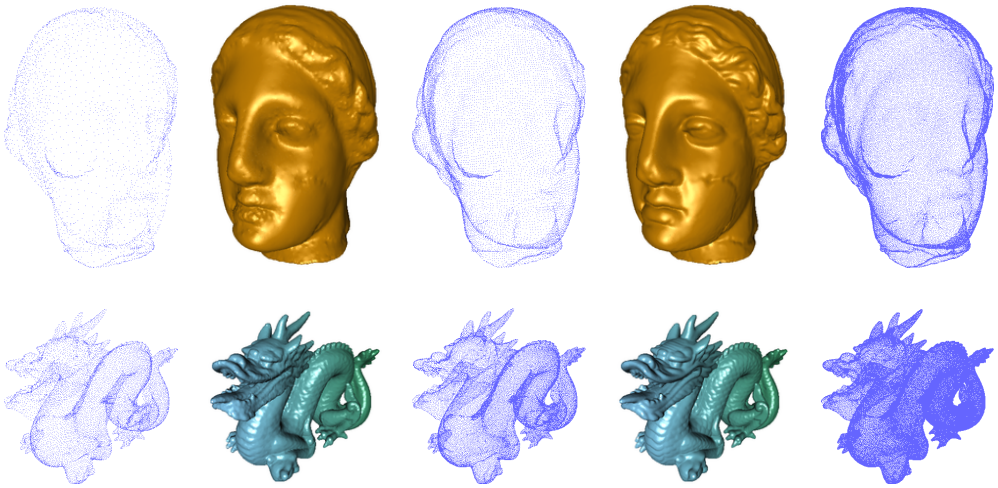


Fig. 7. A progressive series of the Venus and Dragon models.

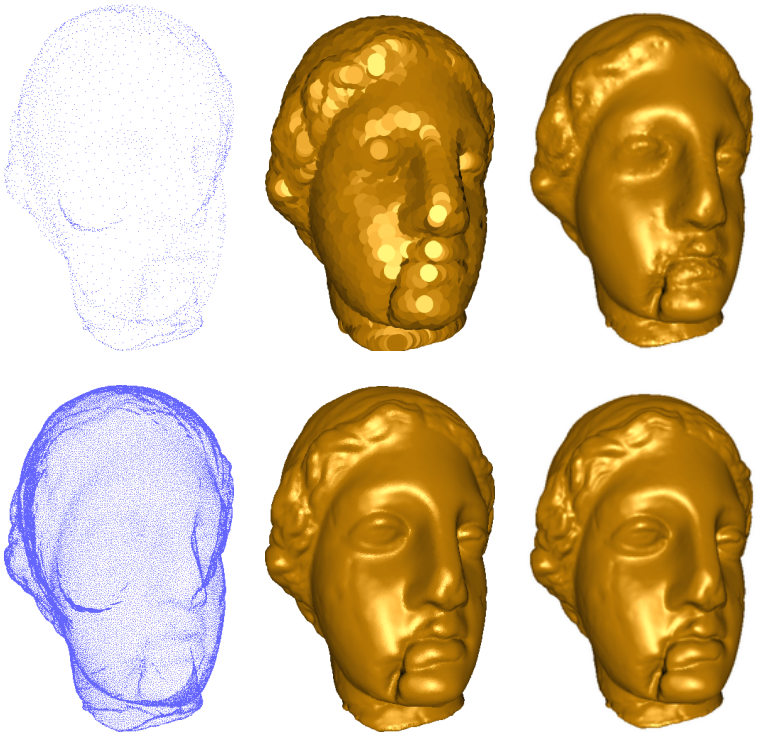
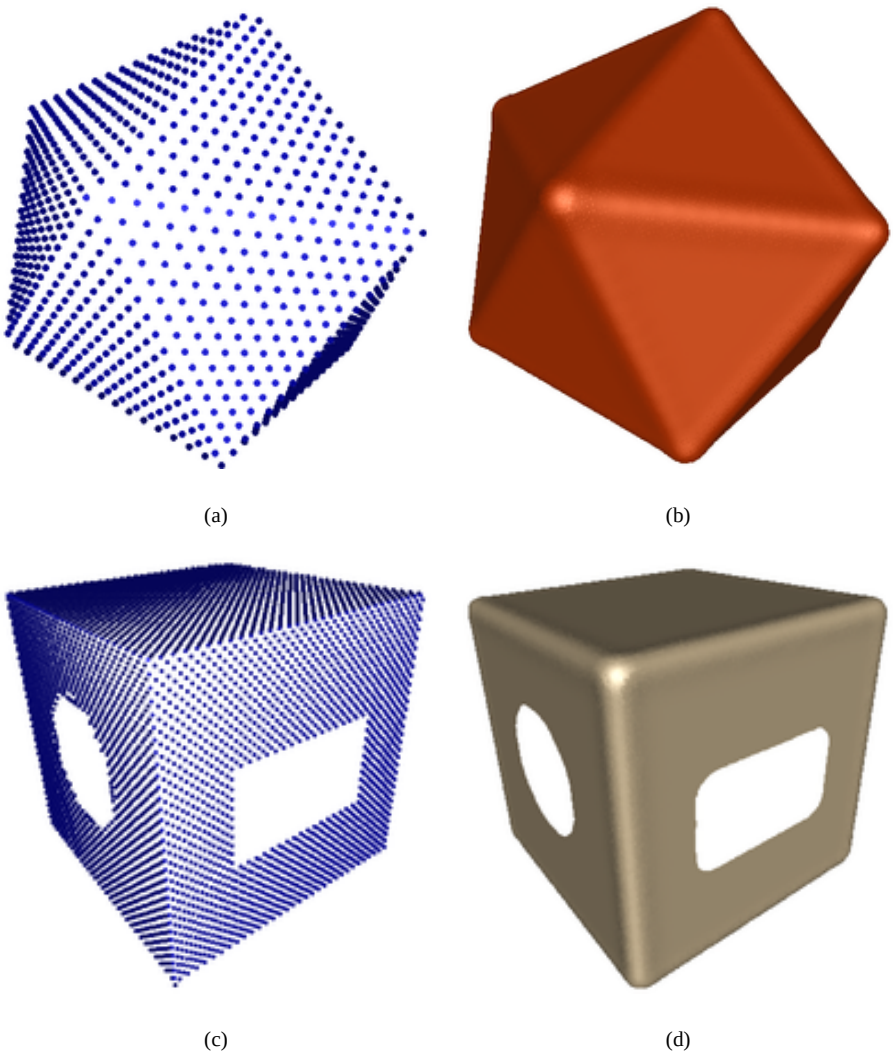


Fig. 8. Two intermediate representations of the Venus model in the hierarchy. On the left we show the set of points. In the middle, the set of points are rendered by splatting using OpenGL™. The images on the right are rendered using an MLS upsampling procedure, requiring no additional data.



(a) (b)

(c) (d)

Fig. 9. Reconstruction of sharp edges and boundaries: On the left (a) and (c), we show the input models. (b) and (d) are the reconstructions of the models using our algorithm.

7. DISCUSSION

As in other multilevel shape representations, the levels essentially correspond to different bands in the spectrum of the shape. The Gaussian weight function leads to a Gaussian filtering of the shape (compare Eqns. (1) and (2)). The spatial radius h of the filter is inversely related to the Gaussian in the frequency domain. Thus, the base point set represents the shape with the most relative energy in the low frequency bands, while the refinement levels add higher frequency bands (see the shape hierarchy in Figure 1, and the details in Figure 3). The projection operator allows us to compute the scalar displacement necessary to lift a point from one level to the next. The displacements are with respect to local frames (as in [Kobbelt et al. 2000]). The magnitude of this displacement is, thus, a measure of the energy in the respective frequency band. This is illustrated with color codes in Figure 4.

The MLS surface definition is based on differential geometry, namely that the surface can be locally approximated by a function. If this assumption is not met, we fail to define the plane (Eq.1) and cannot reconstruct the surface. This ill condition happens when the surface is not sampled densely enough in areas of high curvature or in areas with low SNR, i.e., when the noise is larger than the expected feature size. These conditions can be identified, see [Alexa et al. 2003] for more details.

Because point-sampled objects contain no explicit topological information, it is necessary to make assumptions about the underlying sampling density in order to properly reconstruct this connectivity information. In general, this is a tough problem, and substantial practical and theoretical work has been performed in the area [Amenta and Bern 1999]. Provably good techniques, e.g. [Amenta et al. 1998], have been shown to be effective, but quite slow for actual use, mostly due to the need to compute 3D Delaunay triangulation of the complete point sets. For modeling point sets, simpler (and computationally cheaper) techniques have been used. Zwicker et al. [2002] and Pauly et al. [2002] use k-nearest neighbor queries to reconstruct a neighborhood. This works well for objects with fairly isotropic and uniform sampling density; furthermore the k-nearest neighbor query may unintuitively fill "real" holes in the model. Linsen [2001] uses the *angle criterion* for choosing the neighbors of a points, assuming the object is of genus zero, or otherwise define some threshold on the point spacing. The approach we currently use for defining the neighborhood of a point is to query for nearest neighbors in a user defined radius, assuming that the distribution of points is fairly uniform. This assumption is reasonable for scanned objects. As shown in [Alexa et al. 2003], this method can deal with some variation in the input point density, but will fail if the density varies significantly.

Recently, Kalaiah and Varshney [Kalaiah and Varshney 2001] introduced an effective method for rendering point primitives that requires the computation of the principal curvatures and a local coordinate frame for each point . This approach is natural for the MLS surface representation since it requires a local coordinate frame and the principal curvature for each. During the MLS projection procedure a local coordinate frame is computed, and the principal curvatures can be estimated analytically from Eq. (2).

8. CONCLUSIONS

Our technique has several unique features. First of all, it works directly on points, avoiding the need to triangulate the source point set, and the costly remeshing of the surface into subdivision connectivity. This is especially important for large and detailed datasets. Another important property of our technique is its locality. This leads to a numerically

stable computation, linear time and space complexity and small memory footprint, which may lead to the development of out of core algorithms. Furthermore, progressive point set representations lead to a compression scheme, where the range of the error decreases with each level in the hierarchy. As shown above, at each level it is possible to upsample the surface from the sparse representation.

The lack of connectivity in the representation could also be the source of shortcomings. As shown in [Amenta et al. 1998], there are limits on surface samplings which can be proven to define a given surface. That is, our approach might need a relatively dense base set to resolve possible ambiguities in the modeling of certain complex surfaces. Moreover, it is considerably easier to handle discontinuities by triangulated models.

A considerable amount of research has been devoted to developing and optimizing the "mesh-base world". Many advanced methods require a local parameterization and local differential properties. The MLS projection procedure inherently has these qualities, for example, texture synthesis techniques may be applied on surfaces using surface marching similar to [Turk 1992; 2001]. We believe that the importance of point set representation and of MLS surfaces in particular is likely to play an increasing role in 3D geometric modeling.

ACKNOWLEDGMENTS

Part of this work was done while Shachar Fleishman and Cláudio Silva were at AT&T Labs-Research.

The datasets appear courtesy of Cyberware and the Stanford 3D scanning repository.

This paper would not be possible without the many people who made their software available: Fausto Bernardini for his implementation of the Ball-Pivoting Algorithm; Igor Guskov, Andrei Khodakovsky, Peter Schroeder, and Wim Sweldens for their remeshing and compression code; Craig Gotsman and Virtue Ltd for "Optimizer"; and The Visual Computing Group of CNR-Pisa for Metro. We would like to thank Wagner Correa for help throughout this project.

This work was supported in part by the Israel Science Foundation founded by the Israel Academy of Sciences and Humanities, and by the Israeli Ministry of Science, and by a grant from the German Israel Foundation (GIF).

REFERENCES

ALEXA, M., BEHR, J., COHEN-OR, D., FLEISHMAN, S., LEVIN, D., AND SILVA, C. T. 2001. Point set surfaces. In *IEEE Visualization 2001*. 21–28. ISBN 0-7803-7200-x.

ALEXA, M., BEHR, J., COHEN-OR, D., FLEISHMAN, S., LEVIN, D., AND SILVA, C. T. 2003. Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics 9*, 1 (January), 3–15.

AMENTA, N. AND BERN, M. 1999. Surface reconstruction by Voronoi filtering. *Discrete Comput. Geom. 22*, 4, 481–504.

AMENTA, N., BERN, M., AND KAMVYSSELIS, M. 1998. A new voronoi-based surface reconstruction algorithm. *Proceedings of SIGGRAPH 98*, 415–422. ISBN 0-89791-999-8. Held in Orlando, Florida.

BERNARDINI, F., MITTLEMAN, J., RUSHMEIER, H., SILVA, C., AND TAUBIN, G. 1999. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics 5*, 4 (October - December), 349–359. ISSN 1077-2626.

CARR, J. C., BEATSON, R. K., CHERRIE, J. B., MITCHELL, T. J., FRIGHT, W. R., MCCALLUM, B. C., AND EVANS, T. R. 2001. Reconstruction and representation of 3d objects with radial basis functions. *Proceedings of SIGGRAPH 2001*, 67–76. ISBN 1-58113-292-1.

CATMULL, E. AND CLARK, J. 1978. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design 10*, 350–355.

CIAMPALINI, A., CIGNONI, P., MONTANI, C., AND SCOPIGNO, R. 1997. Multiresolution decimation based on global error. *The Visual Computer 13*, 5, 228–246. ISSN 0178-2789.

CIGNONI, P., FLORIANI, L. D., MONTONI, C., PUPPO, E., AND SCOPIGNO, R. 1994. Multiresolution modeling and visualization of volume data based on simplicial complexes. *1994 Symposium on Volume Visualization*, 19–26. ISBN 0-89791-741-3.

COHEN, J., VARSHNEY, A., MANOCHA, D., TURK, G., WEBER, H., AGARWAL, P., FREDERICK P. BROOKS, J., AND WRIGHT, W. 1996. Simplification envelopes. *Proceedings of SIGGRAPH 96*, 119–128. ISBN 0-201-94800-1. Held in New Orleans, Louisiana.

COHEN-OR, D., LEVIN, D., AND REMEZ, O. 1999. Progressive compression of arbitrary triangular meshes. *IEEE Visualization '99*, 67–72. ISBN 0-7803-5897-X. Held in San Francisco, California.

DESBRUN, M., MEYER, M., SCHRÖDER, P., AND BARR, A. H. 1999. Implicit fairing of irregular meshes using diffusion and curvature flow. *Proceedings of SIGGRAPH 99*, 317–324. ISBN 0-20148-560-5. Held in Los Angeles, California.

DUCHAINEAU, M. A., WOLINSKY, M., SIGETI, D. E., MILLER, M. C., ALDRICH, C., AND MINEEV-WEINSTEIN, M. B. 1997. Roaming terrain: Real-time optimally adapting meshes. *IEEE Visualization '97*, 81–88. ISBN 0-58113-011-2.

ECK, M., DEROSE, T. D., DUCHAMP, T., HOPPE, H., LOUNSBERY, M., AND STUETZLE, W. 1995. Multiresolution analysis of arbitrary meshes. *Proceedings of SIGGRAPH 95*, 173–182. ISBN 0-201-84776-0. Held in Los Angeles, California.

EL-SANA, J. AND VARSHNEY, A. 1999. Generalized view-dependent simplification. *Computer Graphics Forum 18*, 3 (September), 83–94. ISSN 1067-7055.

GOTSMAN, C., GUMHOLD, S., AND KOBBELT, L. 2001. Simplification and compression of 3d meshes. In *European Summer School on Principles of Multiresolution in Geometric Modelling (PRIMUS), Munich*.

GROSSMAN, J. P. AND DALLY, W. J. 1998. Point sample rendering. *Eurographics Rendering Workshop 1998*, 181–192. ISBN 3-211-83213-0. Held in Vienna, Austria.

GUSKOV, I., SWELDENS, W., AND SCHRÖDER, P. 1999. Multiresolution signal processing for meshes. *Proceedings of SIGGRAPH 99*, 325–334. ISBN 0-20148-560-5. Held in Los Angeles, California.

GUSKOV, I., VIDIMCE, K., SWELDENS, W., AND SCHRÖDER, P. 2000. Normal meshes. *Proceedings of SIGGRAPH 2000*, 95–102. ISBN 1-58113-208-5.

HE, T., HONG, L., VARSHNEY, A., AND WANG, S. W. 1996. Controlled topology simplification. *IEEE Transactions on Visualization and Computer Graphics 2*, 2 (June). ISSN 1077-2626.

HOPPE, H. 1996. Progressive meshes. *Proceedings of SIGGRAPH 96*, 99–108. ISBN 0-201-94800-1. Held in New Orleans, Louisiana.

HOPPE, H., DEROSE, T., DUCHAMP, T., MCDONALD, J., AND STUETZLE, W. 1993. Mesh optimization. *Proceedings of SIGGRAPH 93*, 19–26. ISBN 0-201-58889-7. Held in Anaheim, California.

KALAIAH, A. AND VARSHNEY, A. 2001. Differential point rendering. In *Rendering Techniques*, S. J. Gortler and K. Myszkowski, Eds. Springer-Verlag.

KHODAKOVSKY, A., SCHRÖDER, P., AND SWELDENS, W. 2000. Progressive geometry compression. *Proceedings of SIGGRAPH 2000*, 271–278. ISBN 1-58113-208-5.

KOBBELT, L., CAMPAGNA, S., AND SEIDEL, H.-P. 1998. A general framework for mesh decimation. *Graphics Interface '98*, 43–50. ISBN 0-9695338-6-1.

KOBBELT, L., CAMPAGNA, S., VORSATZ, J., AND SEIDEL, H.-P. 1998. Interactive multi-resolution modeling on arbitrary meshes. *Proceedings of SIGGRAPH 98*, 105–114. ISBN 0-89791-999-8. Held in Orlando, Florida.

KOBBELT, L. P., BAREUTHER, T., AND SEIDEL, H.-P. 2000. Multiresolution shape deformations for meshes with dynamic vertex connectivity. *Computer Graphics Forum 19*, 3 (August), 249–260. ISSN 1067-7055.

KOBBELT, L. P., VORSATZ, J., LABSIK, U., AND SEIDEL, H.-P. 1999. A shrink wrapping approach to remeshing polygonal surfaces. *Computer Graphics Forum 18*, 3 (September), 119–130. ISSN 1067-7055.

LEE, A., MORETON, H., AND HOPPE, H. 2000. Displaced subdivision surfaces. *Proceedings of SIGGRAPH 2000*, 85–94. ISBN 1-58113-208-5.

LEE, A., SWELDENS, W., SCHRÖDER, P., COWSAR, L., AND DOBKIN, D. 1998. Maps: Multiresolution adaptive parameterization of surfaces. *Proceedings of SIGGRAPH 98*, 95–104. ISBN 0-89791-999-8. Held in Orlando, Florida.

LEVIN, D. 2000. Mesh-independent surface interpolation. Tech. rep., Tel-Aviv University. http://www.math.tau.ac.il/~levin.

LINSEN, L. 2001. Point cloud representation. Tech. Rep. 3, Fakultaet fuer Informatik, Universitaet Karlsruhe.

PAJAROLA, R. AND ROSSIGNAC, J. 2000. Compressed progressive meshes. *IEEE Transactions on Visualization and Computer Graphics 6*, 1 (January - March), 79–93. ISSN 1077-2626.

PAULY, M. AND GROSS, M. 2001. Spectral processing of point-sampled geometry. *Proceedings of SIGGRAPH 2001*, 379–386. ISBN 1-58113-292-1.

PAULY, M., GROSS, M., AND KOBBELT, L. 2002. Efficient simplification of point-sampled surfaces. *to appear IEEE Visualization*.

PFISTER, H., ZWICKER, M., VAN BAAR, J., AND GROSS, M. 2000. Surfels: Surface elements as rendering primitives. *Proceedings of SIGGRAPH 2000*, 335–342. ISBN 1-58113-208-5.

RUSINKIEWICZ, S. AND LEVOY, M. 2000. Qsplat: A multiresolution point rendering system for large meshes. *Proceedings of SIGGRAPH 2000*, 343–352. ISBN 1-58113-208-5.

TAUBIN, G. 1995. A signal processing approach to fair surface design. *Proceedings of SIGGRAPH 95*, 351–358. ISBN 0-201-84776-0. Held in Los Angeles, California.

TAUBIN, G., GUEZIEC, A., HORN, W., AND LAZARUS, F. 1998. Progressive forest split compression. *Proceedings of SIGGRAPH 98*, 123–132. ISBN 0-89791-999-8. Held in Orlando, Florida.

TURK, G. 1992. Re-tiling polygonal surfaces. In *Computer Graphics (Proceedings of SIGGRAPH 92)*. Vol. 26. Chicago, Illinois, 55–64. ISBN 0-201-51585-7.

TURK, G. 2001. Texture synthesis on surfaces. *Proceedings of SIGGRAPH 2001*, 347–354. ISBN 1-58113-292-1.

WARREN, J. AND WEIMER, H. 2001. *Subdivision Methods for Geometric Design: A Constructive Approach*. Morgan Kaufmann.

XIA, J. C., EL-SANA, J., AND VARSHNEY, A. 1997. Adaptive real-time level-of-detail-based rendering for polygonal models. *IEEE Transactions on Visualization and Computer Graphics 3*, 2 (April - June). ISSN 1077-2626.

ZHOU, Y., CHEN, B., AND KAUFMAN, A. E. 1997. Multiresolution tetrahedral framework for visualizing regular volume data. *IEEE Visualization '97*, 135–142. ISBN 0-58113-011-2.

ZORIN, D., SCHRÖDER, P., AND SWELDENS, W. 1997. Interactive multiresolution mesh editing. *Proceedings of SIGGRAPH 97*, 259–268. ISBN 0-89791-896-7. Held in Los Angeles, California.

ZWICKER, M., PAULY, M., KNOLL, O., AND GROSS, M. 2002. Pointshop 3d: An interactive system for point-based surface editing. *ACM Transactions on Graphics 21*, 3 (July), 322–329. ISSN 0730-0301 (Proceedings of ACM SIGGRAPH 2002).