Geometry-aware Bases for Shape Approximation

Olga Sorkine, Daniel Cohen-Or, *Member, IEEE,* Dror Irony, and Sivan Toledo

Abstract— **We introduce a new class of shape approximation techniques for irregular triangular meshes. Our method approximates the geometry of the mesh using a linear combination of a small number of basis vectors. The basis vectors are functions of the mesh connectivity and of the mesh indices of a number of *anchor* vertices. There is a fundamental difference between the bases generated by our method and those generated by geometry-oblivious methods, such as Laplacian-based spectral methods. In the latter methods, the basis vectors are functions of the connectivity alone. The basis vectors of our method, in contrast, are *geometry-aware*, since they depend on both the connectivity and on a binary tagging of vertices that are "geometrically important" in the given mesh (e.g., extrema). We show that by defining the basis vectors to be the solutions of certain least-squares problems, the reconstruction problem reduces to solving a single sparse linear least-squares problem. We also show that this problem can be solved quickly using a state-of-the-art sparse-matrix factorization algorithm. We show how to select the anchor vertices to define a compact effective basis from which an approximated shape can be reconstructed. Furthermore, we develop an incremental update of the factorization of the least-squares system. This allows a progressive scheme where an initial approximation is incrementally refined by a stream of anchor points. We show that the incremental update and solving the factored system are fast enough to allow an on-line refinement of the mesh geometry.**

Index Terms— **shape approximation, basis, mesh Laplacian, linear least-squares**

I. INTRODUCTION

SHAPE approximation is an important problem in computer graphics and CAGD. Reducing the amount of data needed to represent a specific shape is often necessary for modeling, efficient storage and transmission of 3D models. Irregular triangle meshes are the predominant means of representing shapes, and in the last decade there has been a vast amount of work on mesh simplification techniques [1]. These techniques are closely related, and can be regarded as descendants of knot removal techniques developed for spline curves and surfaces [2]. Other approximation techniques, suited for semi-regular connectivity, are based on wavelet representations or subdivision surfaces [3]–[6].

Mesh simplification techniques aim to approximate a given shape with as few vertices or triangles as possible, while keeping the error of the approximation, in some given metric, lower than a prescribed tolerance. A different class of approximation techniques retains the original connectivity of the given mesh and approximates only its geometry [7]–[9]. Karni and Gotsman [7] introduce a spectral method where the mesh is approximated by reconstructing its geometry using a linear combination of a number of basis vectors. The basis is derived from the spectral decomposition of the Laplacian matrix associated with the mesh connectivity [10]. Chou and Meng [8] encode the geometry of the mesh using vector quantization of the displacement coordinates. Based on an analysis of the spectral basis of the Laplacian, Sorkine et al. [9] introduce a method where the quantization is applied to the geometry vector transformed by the Laplacian operator.

Laplacian-based methods are attractive for mesh processing, since they benefit from the powerful set of tools from linear algebra and signal processing. The eigenvectors of the mesh Laplacian matrix can be viewed as an extension of the Fourier transform basis functions for the irregular connectivity case, and the eigenvalues represent the frequencies [7], [11]. The spectral basis is readily defined on the given irregular mesh and does not require altering the input representation. In addition to geometry-compression applications [7], [12], spectral properties have been studied for the design of fairing filters and modeling tools [11], [13], mesh watermarking [14] and spherical parameterization [15].

However, together with their appealing properties, one must bear in mind that pure Laplacian-based methods are geometry-oblivious, since the basis vectors are functions of the connectivity alone. It is possible to use the geometric Laplace-Beltrami operator (see, e.g., [16]), however, its construction requires heavy use of the mesh geometry, which is not practical for compression applications. Our new *geometry-aware* methods derive the basis both from the mesh connectivity and limited geometrical information. The basis vectors in our methods are centered around selected "geometrically important" anchor vertices. This allows a terse capturing of impor-

All the authors are with the School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel.
E-mail: {sorkine|dcor|irony|stoledo}@tau.ac.il

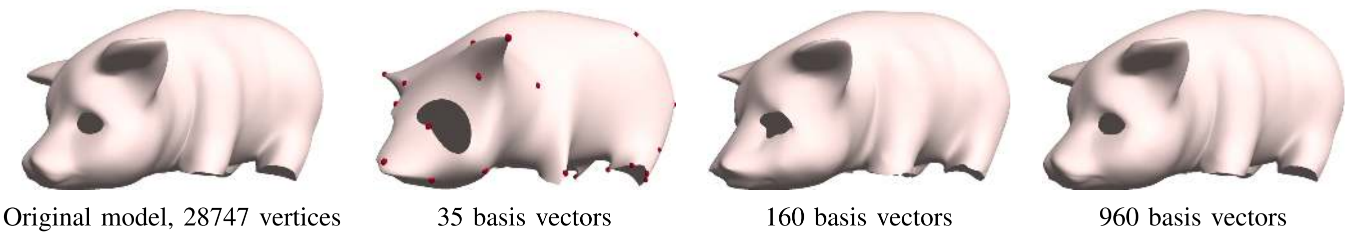


Fig. 1. Reconstruction of mesh geometry using geometry-aware bases. A geometry-aware basis function is centered around a certain *anchor* vertex of the mesh. The locations of the anchors used in reconstruction in the second left figure are marked by red spheres.

tant features of the surface and leads to compact and efficient representation of the mesh geometry. Figure 2 illustrates the reconstruction using such basis vectors on a 2D curve example. The bottom row shows the meshes reconstructed using geometry-aware bases. The locations of the anchors are marked by small dots. The reconstructed mesh passes close to the original locations of the anchor points, which enables good approximation of such features as the tips of the bird's wings and tail. For comparison, reconstruction of this mesh using an analogous number of spectral basis vectors misses out the features. This behavior is evident in large as well as in small scale.

It should be noted that explicit computation of the basis vectors is, generally speaking, too expensive for large meshes. Geometry representation using the Laplacian eigenbasis [7] requires finding a partial spectral decomposition of a large symmetric matrix. This computation is too expensive to be applied in practice to anything but small meshes.

The method that we present here avoids explicit computation of the underlying basis. Instead of directly representing the geometry by the coefficients of the linear combination of the basis vectors, we reduce the reconstruction problem to solving a sparse linear least-squares system, as explained in Section II. State-of-the-art least-squares solvers make the solution efficient and enable reconstruction of the mesh as a whole.

A. Overview

The proposed geometry-aware representation of a shape is a linear combination of k basis functions, which are vectors that assign a real value to each vertex of the mesh. The basis functions are an implicit function of the connectivity of the mesh and of the indices of k vertices that we call *anchors*. Each basis function is selected so that it fulfils the following conditions in the least-squares sense: it attains the value 1 at one of the anchors and 0 at the other anchors, and it is the smoothest among all the functions that satisfy these requirements (we also propose a slightly different definition for strictly interpolatory anchors, but the principle is the same). The smoothness of a function is defined in a discrete manner using the connectivity of the mesh. Specifically, we require that the position of a vertex deviates as little as possible from the average of its neighbors in the mesh. These definitions result in smooth basis functions that are easy to combine into an approximation that attains specific values at the anchors. Furthermore, a fast sparse least-squares solver with updating capability allows us to efficiently recover a representation of the shape from the coefficients of the linear combination.

A number of recent papers have shown that the connectivity of the mesh often encodes some useful information about the geometry of the shape that the mesh represents [17], [18]. Isenburg et al. [17] reconstruct a shape from the connectivity by a non-linear optimization of a uniform edge-length criterion. In [18] it was shown that augmenting the connectivity with a few well-placed anchors significantly increases the geometric value of the information encapsulated in the connectivity alone. The least-squares system that is used to reconstruct the so-called LS-mesh in [18] is essentially the same system that arises from our basis vectors. In this paper, we fully explore the application of geometry compression, both theoretically and experimentally. Progressive compression is made possible thanks to the proposed algorithm that quickly augments the existing representation with new anchors without fully solving the least-squares reconstruction system again. We rigorously analyze the underlying basis vectors, which provides a theoretical framework for studying this type of approximation approaches.

The effectiveness of adding anchors with geometric information was used earlier in [9] to reduce the low-frequency error caused by quantization of the differential coordinates of the mesh. There, a linear least-squares system was solved to reconstruct the mesh geometry from a quantized differential representation, and the work focused on the analysis of the visual impact of the quantization error. In our case, the mesh vertices do not hold any geometric information – it is entirely encapsulated in the basis functions.

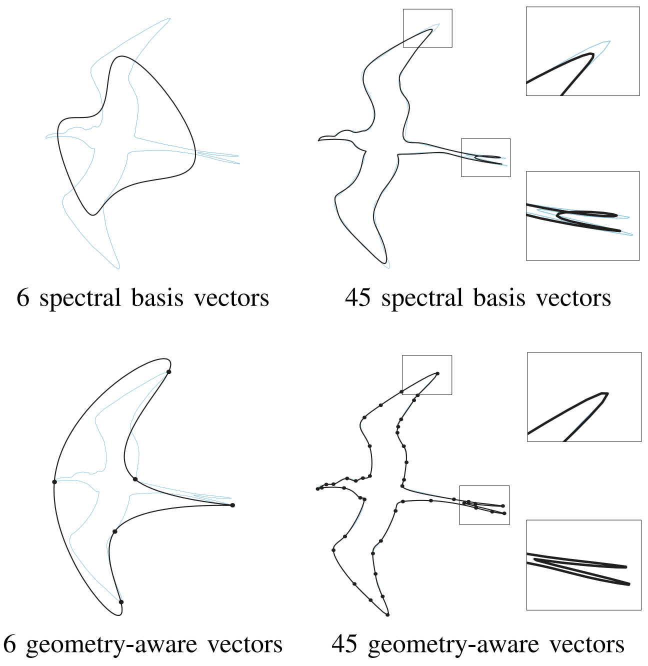


Fig. 2. Reconstruction of the swallow curve (simple closed path) using different bases. The top row shows reconstruction using the Laplacian eigenvectors, which are the discrete Fourier basis function in this case. The bottom row displays reconstruction with geometry-aware basis vectors. The reconstructed mesh is shown in black, while the original mesh is tinted in blue. The geometry-aware bases better approximate the features of the shape, on large as well as on small scales.

The main contributions of this paper include efficient algorithms for producing a geometric approximation of a shape and for recovering the approximate shape from the compact representation. Our algorithms are based on several advanced computational linear algebra tools: the ability to control the conditioning of the least-squares problems that we solve, the ability to solve them quickly, and the ability to quickly add anchors by updating a sparse factorization of an augmented Laplacian matrix. We also provide evidence that the new method compares favorably with spectral methods, both in terms of compression ratios for a given approximation error and in terms of running times. The paper explores the theory of augmenting the connectivity with geometric data, in search for a better understanding of shapes in general and approximation of irregular meshes in particular.

II. Geometry-aware bases

Most of the techniques for approximating and encoding mesh geometries represent the geometry as a linear combination of basis functions. In this section we present the specific basis functions that we use and explain why this basis is effective.

A *mesh function* is a real vector that assigns a value to each vertex in the mesh. A *basis function* is simply a mesh function, and a *basis* is a set of basis functions that spans $\mathbb{R}^n$, where n is the number of vertices in the mesh. The coordinates of the vertices, say the x coordinates, are a mesh function that expresses the location of the vertices in $\mathbb{R}^3$ as a linear combination of the functions of the *standard basis*, whose functions assign 1 to one vertex and 0 to all the others. The coordinates can also be expressed as a linear combination of other basis functions.

The bases that we use, like Laplacian-spectral bases, can be constructed by solving a series of minimization problems. This construction is perhaps not the most natural one for Laplacian-spectral bases, but it is the most natural for our bases. Let us describe this construction for the well-known Laplacian-spectral bases first. The combinatorial Laplacian of a mesh is the $n \times n$ symmetric positive semi-definite matrix $L = D - A$, where $A = (a_{ij})$ is the adjacency matrix ($a_{ij} = 1$ if vertices i and j are neighbors and $a_{ij} = 0$ otherwise) and D is the diagonal matrix whose ith entry on the diagonal equals the valency (degree) of vertex i.

Given the Laplacian L of the mesh, the first Laplacian-spectral basis function $\mathbf{u}_1$ is the function that minimizes[1] $\|L\mathbf{u}_1\|$ subject to $\|\mathbf{u}_1\| = 1$. The next basis function $\mathbf{u}_2$ is the one that minimizes $\|L\mathbf{u}_2\|$ subject to $\|\mathbf{u}_2\| = 1$ and to $\mathbf{u}_2 \perp \mathbf{u}_1$. In general, $\mathbf{u}_k$ minimizes $\|L\mathbf{u}_k\|$ subject to $\|\mathbf{u}_k\| = 1$ and to $\mathbf{u}_k \perp \operatorname{span}\{\mathbf{u}_1, \ldots, \mathbf{u}_{k-1}\}$. The functions $\mathbf{u}_k$ are the eigenvectors of L sorted by the eigenvalues. The minimization problems above favor smooth basis functions, because the transformation $\mathbf{x} \mapsto L\mathbf{x}$ assigns to each vertex i the difference between x_i and the average of its neighbors, multiplied by the number of neighbors. Therefore, $\mathbf{u}_1$ is the smoothest vector in $\mathbb{R}^n$, the constant vector, $\mathbf{u}_2$ is the smoothest mesh function orthogonal to $\mathbf{u}_1$, and so on. The first function $\mathbf{u}_1$ is always the same, while the shapes of the rest depend on the topology of the mesh.

A. Relaxed geometry-aware bases

Our basis also solves a series of minimization problems, but they are chosen in a geometry-aware manner. Given a set of k vertex indices $1 \le a_1, a_2, \ldots, a_k \le n$, the ith function $\mathbf{v}_i$ in our basis minimizes

$$\|L\mathbf{v}_i\|^2 + \left(\sum_{j \neq i} \omega^2 |(v_i)_{a_j} - 0|^2 + \omega^2 |(v_i)_{a_i} - 1|^2 \right).$$

[1]We use the following notation. Vectors are denoted by upright bold letters, e.g., $\mathbf{x}$, and their elements are denoted by italic letters, e.g., x_i. All the vectors in this paper are column-vectors and all the norms are 2-norms.

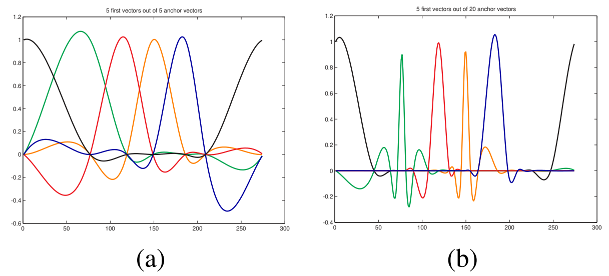


Fig. 3. Geometry-aware basis functions on a 1D domain. The mesh here is a simple closed path with 274 vertices. Plot (a) displays the five basis functions corresponding to a set of five anchors; (b) shows the first five basis functions out of a 20-anchor basis.

The interpretation of this minimization problem is the following. The basis function $\mathbf{v}_i$ minimizes the sum of two terms. The first term is the non-smoothness in $\mathbf{v}_i$, and the second is the deviation of $\mathbf{v}_i$ from given values in the k mesh locations $a_1, \ldots, a_k$, which we call *anchors*. These values are 1 at a_i and 0 at a_j, $j \neq i$. Therefore, $\mathbf{v}_i$ *tries simultaneously to be smooth everywhere, to be large at a_i, and to vanish on all the other a_j's.* The weight ω controls the impact of the anchors. Our algorithms never use basis functions other than the first k (the number of anchors), so there is no point in characterizing them. (Formally, all completions of this set of k functions to a basis of $\mathbb{R}^n$ are equivalent for our algorithms.)

Figure 3 shows five geometry-aware basis functions on a mesh consisting of a simple path. On this mesh, the first Laplacian-spectral basis functions are simply low-frequency sines and cosines. The geometry-aware functions are also fairly smooth, but most of their "energy" is concentrated near a single anchor. Basis functions for larger k are less smooth, because the anchors get closer to each other, forcing the functions to attain values near 0 and near 1 within short intervals. Intuitively, a few geometry-aware functions should allow us to approximate smooth mesh functions whose extrema are at or near the anchors more accurately than a few geometry-oblivious Laplacian-spectral functions.

We express approximations of mesh functions using a set of k anchors and the coefficients $\mathbf{c} = (c_1, \ldots, c_k)^T$ of the corresponding k geometry-aware functions $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k)$. Given this representation of the approximation, we reconstruct the approximation $\tilde{\mathbf{x}}$ in the standard basis by solving a single least-squares minimization problem,

$$\begin{aligned}\tilde{\mathbf{x}} &= \underset{\mathbf{x}}{\operatorname{argmin}} \left\{ \|L\mathbf{x}\|^2 + \sum_{i=1}^{k} \omega^2 |x_{a_i} - c_i|^2 \right\} = \\ &= \sum_{i=1}^{k} c_i \mathbf{v}_i = V\mathbf{c}\end{aligned}$$

The equality follows from the linearity of the minimum-norm solution to least-squares problems. The coefficient matrix $\tilde{L}$ of this least-squares problem has $n+k$ rows and n columns. The significance of this expression is that it shows that we can reconstruct $\tilde{\mathbf{x}}$ from V without any reference to the basis vectors V. Thus, assuming w.l.o.g. that $(a_1, a_2, \ldots, a_k) = (1, 2, \ldots, k)$, we reconstruct $\tilde{\mathbf{x}}$ by simply finding the vector that minimizes the norm of

$$\left[\begin{array}{c} L \\ \hline \omega\, I_{k\times k} \mid 0 \end{array}\right] \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} - \left[\begin{array}{c} | \\ 0 \\ | \\ \hline \omega\, c_1 \\ \vdots \\ \omega\, c_k \end{array}\right] . \tag{1}$$

Since L is typically very sparse, this least squares can be solved very quickly even when n is large. It should be noted that the reconstruction is not interpolatory at the given values on the anchors – it only approximates them in a least-squares sense.

There are at least three categories of constraints that we can apply to the anchors. The method that we presented above charges a quadratic penalty for deviations of $\tilde{\mathbf{x}}$ from $\mathbf{x}$ at the anchors. We can use different weights for these penalties and for the smoothness penalties. Another option is to use box constraints, which require that $\tilde{x}_{a_i}$ lies within a box centered around x_{a_i} [19]. The algorithmic issues in this approach are more complex than in the other approaches, so we have not pursued it. The third approach is interpolatory; it requires that $\tilde{x}_{a_i} = x_{a_i}$. This is the limiting case of the two other approaches. We explain this approach next.

B. An interpolatory scheme

We can create slightly different geometry-aware bases by forcing the basis functions to attain prescribed values at specific mesh locations. Given a set of k vertex indices $1 \leq a_1, \ldots, a_k \leq n$, the ith function $\mathbf{w}_i$ in the basis minimizes $\|L\mathbf{w}_i\|$ subject to $(w_i)_{a_i} = 1$ and $(w_i)_{a_j} = 0$ for $j \neq i$.

Given the indices of the k anchors and the coefficients $\mathbf{c}$ of the basis functions $(\mathbf{w}_1, \ldots, \mathbf{w}_k) = W$, the approximation $\hat{\mathbf{x}}$ can be reconstructed as follows. We use the equation $x_{a_i} = c_i$ to eliminate x_{a_i} from the system. This effectively deletes column a_i and row $n + a_i$ from the coefficient matrix $\tilde{L}$ (where $\omega = 1$) and changes the right-hand side. After all these equations are eliminated, the resulting coefficient matrix $\hat{L}$ has n rows and $n-k$ columns. To reconstruct the unknown values $x_j, j \notin \{a_i\}$, we solve the least-squares problem

$$\min_{\mathbf{x}} \|\hat{L}\mathbf{x} - (-\tilde{L}_{1:n,\{a_i\}}\mathbf{c})\| .$$

Combining the minimizer with the known values $x_{a_i} = c_i$ yields the approximation

$$\hat{\mathbf{x}} = \sum_{i=1}^{k} c_i \mathbf{w}_i = W\mathbf{c} \; .$$

The coefficient matrix $\hat{L}$ of this least-squares problem is smaller and sparser than $\tilde{L}$, so in general it will be even easier to solve the least-squares problem that reconstructs $\hat{\mathbf{x}}$.

The main disadvantage of this basis, compared to the relaxed basis $\{\mathbf{v}_i\}$, is that adding interpolatory anchors is computationally more expensive than adding least-squares anchors; we explain this issue below, in Section III. When a large weight ω is used for the anchors in the relaxed scheme, the solution effectively becomes very close to interpolatory. See Figure 4 that visualizes the influence of different ω's.

C. Approximating mesh functions

So far we have seen the basis functions and how to reconstruct an approximation given the indices of the anchors and the coefficients of the basis functions. We now turn to the question of how to generate the coefficients $\mathbf{c} = (c_1, \ldots, c_k)^T$ given a mesh function $\mathbf{x}$ and a set $a_1, \ldots, a_k$ of anchors.

Perhaps the best way to define $\mathbf{c}$ is by requiring that the approximation $\tilde{\mathbf{x}} = V\mathbf{c}$ or $\hat{\mathbf{x}} = W\mathbf{c}$ of a mesh function $\mathbf{x}$ be as close as possible, in the 2-norm, to $\mathbf{x}$. That is, to require that $\mathbf{c}$ minimizes $\|V\mathbf{c} - x\|$ or $\|W\mathbf{c} - x\|$ (depending on the basis used). Solving these systems is potentially expensive. A naive way to compute these optimal $\mathbf{c}$'s is to compute V or W explicitly, by solving the least-squares problems that define their columns, and then to solve the dense $n \times k$ least-squares problem. Note that to reconstruct $\tilde{\mathbf{x}} = V\mathbf{c}$ or $\hat{\mathbf{x}} = W\mathbf{c}$, we do not use an explicit representation of V or W.

For interpolatory geometry-aware bases, another natural way to choose $\mathbf{c}$ is by setting $c_i = x_{a_i}$. This ensures that $\hat{\mathbf{x}}$ coincides with $\mathbf{x}$ at the anchors. The 2-norm of the error $\hat{\mathbf{x}} - \mathbf{x}$ is likely to be higher than if we define $\mathbf{c}$ so as to minimize the error, but now the error is concentrated away from the anchors. It turns out that setting $c_i = x_{a_i}$ works well even for relaxed geometry-aware bases V. Employing large weights ($\omega \to \infty$) on the relaxed anchors effectively makes the relaxed scheme interpolatory, while maintaining the advantage of the updating capability (see Section III). In practice, we set $\omega = 10n$.

III. THE PROGRESSIVE SCHEME

One of the best aspects of relaxed geometry-aware bases is that we can quickly improve the approximation as soon as the location of additional anchors becomes known. This allows a client to display a rough approximation as soon as the location of a few anchors is received from a server or retrieved from storage.

When the locations of additional anchors become known to the client, it can produce a more accurate approximation by updating the system with the new information. The following system is solved:

$$\tilde{L}_{\text{new}}\mathbf{x} = (\mathbf{0}_{1\times n},\ \omega\, c_1, \ldots, \omega\, c_k, \omega\, c_{k+1}, \ldots, \omega\, c_{k+m})^T ,$$

where $\tilde{L}_{\text{new}}$ is the updated system matrix comprised of the previous $\tilde{L}$, and additional rows for the new anchors $a_{k+1}, \ldots, a_{k+m}$; $c_{k+1}, \ldots, c_{k+m}$ denote the new coefficients. The key to utilizing additional anchors is an efficient updating scheme to a sparse factorization of $\tilde{L}$. The system (1) can be solved using a sparse Cholesky factorization of the normal equations, $\tilde{L}^T\tilde{L} = R^T R$, where R is sparse and upper triangular. The factorization is done once, for an initial set of anchors. Suppose that we now add an anchor a_{k+1}. This adds a row to $\tilde{L}$, and adds ω^2 to the a_{k+1}th diagonal element of $\tilde{L}^T\tilde{L}$. To reconstruct the new approximation, we need a new Cholesky factorization of $\tilde{L}_{\text{new}}^T\tilde{L}_{\text{new}}$. Fortunately, we can update the previous factorization in time proportional to the number of nonzeros in R. Furthermore, the update does not modify the nonzero structure of R, only the numerical values of its entries.

We update R as follows. We essentially eliminate the single nonzero in the new row in $\tilde{L}$ using a series of Givens rotations that we perform on that row and on rows of R. The first rotation is performed on row a_{k+1} of R and annihilates the a_{k+1}th element in the new row. This, however, introduces nonzeros to several elements in the new row, elements with column indices greater than a_{k+1}. We then eliminate the nonzero element with the smallest column index in the new row, say index i, using a Givens rotation on row i of R. The Givens rotation never modifies the nonzero structure of rows in R, because the next row that we update is always the parent in the elimination tree of $\tilde{L}^T\tilde{L}$ of the previous row. Since we update R using a series of orthogonal transformations (the Givens rotations) and since the addition of a ω^2 to the diagonal of $\tilde{L}^T\tilde{L}$ only improves its conditioning, the updating process is always numerically stable.

Our incremental update method can be viewed as a special case of the general algorithm proposed by Davis and Hager [20]. Due to the specific structure of the change in $\tilde{L}$, the update in our case is particularly

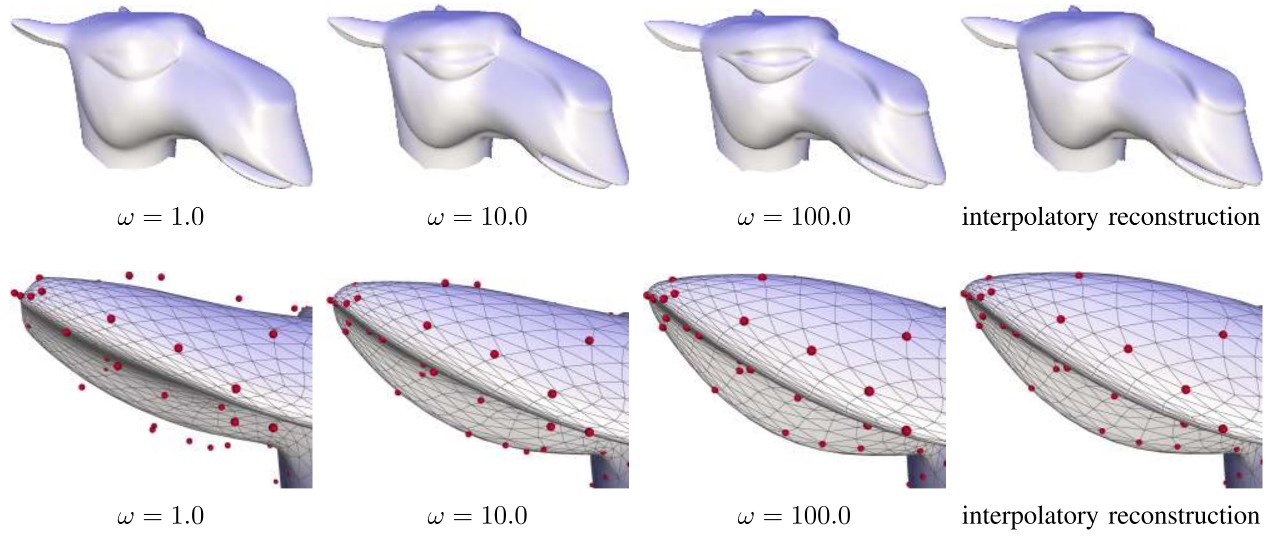


Fig. 4. The effect of different weights on the relaxed scheme. The first three columns display the reconstruction with the relaxed scheme using the same set of anchors, with different weights. The rightmost column shows the reconstruction using the interpolatory scheme. Close-up on the ear is shown in the bottom row; the red spheres denote the position of the anchor vertices in the original mesh. As the weight of the anchors grows, the reconstruction approaches to being interpolatory.

efficient. More specifically, Davis and Hager show how to update the Cholesky factor R when an arbitrary row is added or removed from $\tilde{L}$. Our algorithm solves a special case of this general update/downdate problem: the case of adding a row with a single nonzero. In the general case, the nonzero structure of R might change, and so does its elimination tree. These changes require a sophisticated algorithm to take care of sparsity. Furthermore, since R can fill as a result of an update, the cost of a series of updates can be hard to predict. In contrast, in our case the nonzero structure of R and the elimination tree do not change, so the path in the elimination tree from the vertex a_{k+1} to the root gives the sequence of elimination operations that must be performed. This special case is considerably simpler. Another difference between the algorithm of Davis and Hager and ours is that we use orthogonal Givens rotations to eliminate the new row in $\tilde{L}$, whereas they use nonorthogonal operations. As a consequence, our algorithm performs 4 floating-point operations per nonzero in R that is modified, and their algorithm performs only 2 per modified nonzero. Using fast Givens rotation in our algorithm would bring the two algorithm to the same cost per modified nonzero, but due to the insignificance of the update costs, we did not implement such an approach. In short, our algorithm is, essentially, a special case of [20]. But in our case, much of the machinery developed in [20] is not needed.

Since three-dimensional meshes typically have small vertex separators, and due to the special structure of our updates, we can provide a tighter bound on the cost of an update operation than was given by Davis and Hager. They show that the cost of an update operation is proportional to the number of nonzeros in R that are being modified. The same is true in our algorithm. However, in our case we can argue that under a reasonable assumption, the number of modified nonzeros in R is proportional to n, the size of the mesh; in most cases, n is much smaller than the number of nonzeros in R. Suppose that a mesh can be embedded on the surface of a body with bounded genus (that is, without many holes). Then the mesh has excluded minors, which implies that it has a $O(\sqrt{n})$ approximately-balanced vertex separator [21]. Once separated, the same holds for the parts. The separators form a tree, and the path in the elimination tree from row a_{k+1} to the root is also a path in this separator tree. The number of nonzeros in R that is modified is at most the sum of the squared sizes of the separators on this path, which is at most

$$(c\sqrt{n})^2 + \left(c\sqrt{(2/3)\,n}\right)^2 + \left(c\sqrt{(2/3)^2\,n}\right)^2 + \cdots <$$
$$< 1.8\,c^2\,n\ ,$$

for some constant c that depends on the genus. Note that for such meshes, the total number of nonzeros in R is $\Theta(n \log n)$, so the update only modifies a small fraction of them. In particular, the update is much cheaper than solving a single least-squares problem with the computed factor R.

In the graphics literature, updating linear systems of equations due to changes of boundary conditions was also performed by James and Pai [22]. However, they use the capacitance-matrix approach, where a change of rank s in the original system matrix requires $O(s^3)$ operations for update. This type of approach is not efficient when it comes to incremental updates, since the update has to be applied to the original factorization of the first $\tilde{L}$. Thus, the cost would be cubic in the total number of added anchors.

In general, updating a sparse factorization with arbitrary constraints can be both expensive and unstable. For example, updating the factorization of the interpolatory scheme is probably more difficult than updating R to accommodate additional relaxed anchors. However, it is easy to add relaxed anchors to a factorization of an interpolatory basis. What is important is what kind of constraint we add, not how the original factorization was produced. Therefore, we only employ relaxed anchors update, which is guaranteed to be stable.

IV. Selecting anchors

The norm of the approximation error $\|\mathbf{x} - \tilde{\mathbf{x}}\|$ is governed by two factors: the condition number of $\tilde{L}$ and the angle between $\mathbf{x}$ and $\text{span}\{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$. The first factor depends on the location of the anchors in the topology of the mesh, and is independent of the geometry of the shape. It is proven that $\tilde{L}$ is well-conditioned if, loosely speaking, no vertex is too far (in terms of mesh edges) from an anchor, i.e., if the anchors are well-distributed across the mesh graph. Theoretical bounds on the condition number of $\tilde{L}$, as well as a practical algorithm for choosing an initial set of anchors to condition $\tilde{L}$, can be found in [23]. The second factor depends on the interaction between the the geometry of the given shape and the basis functions $\mathbf{v}_1, \ldots, \mathbf{v}_k$.

We use an iterative greedy heuristic to reduce the angle between $\mathbf{x}$ and $\text{span}\{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$. Given a set of anchors $a_1, \ldots, a_{k-1}$, we compute an approximation $\tilde{\mathbf{x}}$ of the given shape and find the vertex on which $\tilde{\mathbf{x}}$ differs most from $\mathbf{x}$. That vertex becomes the next anchor, a_k. Since we try to approximate at least three mesh functions using the same anchors (the mesh function in three space dimensions), we actually select the vertex whose spacial 3D location in the approximated shape has the largest geometric distance to its location in the original mesh.

The above incremental selection scheme is well suited for progressive transmission of the mesh geometry: the server sends the anchors to the client in the same order in which they were chosen by the greedy algorithm.

V. Results

We have tested our shape approximation method on several 3D models. We report results only for the relaxed geometry-aware bases due to lack of efficient updating capability for the interpolatory scheme, as discussed in Section III. To reconstruct an approximation from an initial set of anchors, the client needs to compute the sparse factorization of $\tilde{L}$ (the connectivity is supposed to be already known) and to solve for the mesh functions $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$. When more anchors become known, the factorization is updated and we solve for $\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$ again. The running times of these key ingredients are summarized in Table I. The factorization is the most costly part, and is computed only once; the update and solve times are very small. We have used the direct solvers provided by TAUCS [24]. All our experiments were carried out on a 2.4 GHz Pentium 4 machine.

The compressed representation consists of the indices of the chosen anchors and the basis coefficients, which are the locations of the anchor vertices in the original model. The coefficients are uniformly quantized, and all the data is encoded using an arithmetic encoder. However, since the locations and the indices of the anchors are scattered across the mesh, entropy-encoding typically does not further reduce the size of the representation. Thus, roughly $k \log n$ bits are needed to represent the indices of the k anchors and $3kq$ bits for the coefficients, where q is the quantization level. Note that since our scheme favors smooth reconstructions, the approximated shapes do not suffer from "jaggies" effects that would be caused by quantization of *all* the x, y, z coordinates. We used q between 10 to 12 bits.

The results of approximations using varying numbers of basis vectors are shown in Figures 1 and 6. One can observe that the main features of the models, such as extruding parts, are captured in the very early stages of the progressive scheme (i.e. with a small number of basis vectors). We have compared our results with the method of Karni and Gotsman [7]. The spectral basis of the mesh Laplacian [7] is a natural candidate for comparison with our geometry-aware bases, since both compression methods preserve the mesh connectivity, unlike the compression schemes that require semi-regular remeshing [3]–[6]. We have carried out such a comparison; however, it is limited to small meshes only. As discussed above, the spectral method requires computing a partial eigendecomposition of the Laplacian, which is time- and space-consuming. We used MATLAB's `eigs` function to find the first several eigenvectors of some submeshes of the *Camel* model (see Figure 5). Computation of the first 1000 eigenvectors of a mesh with 3220 vertices took about 4 minutes on a 2.4 GHz machine with 2 GB or RAM. The computation used more than 1 GB of RAM, and indeed, on a similar machine with only 1 GB of RAM, the computation took about 20 minutes due to paging. Computing the

TABLE I

RUNNING TIMES IN SECONDS OF THE DIFFERENT COMPONENTS OF SOLVING THE LINEAR LEAST-SQUARES SYSTEMS. *Factor* STANDS FOR THE FACTORIZATION TIME OF THE NORMAL EQUATIONS MATRIX; *Solve* IS THE TIME OF SOLVING FOR A SINGLE MESH FUNCTION BY BACK-SUBSTITUTION; *Average update* IS THE AVERAGE TIME SPENT ON UPDATING THE FACTORIZATION BY ONE RELAXED ANCHOR (THE NUMBERS IN PARENTHESES DENOTE THE RANGE OF ANCHOR AMOUNTS OVER WHICH THE AVERAGE WAS COMPUTED). *Worst-case* STANDS FOR THE LONGEST UPDATE TIME OBSERVED OVER THE UPDATES OF THE PREVIOUS COLUMN.

Model	# vertices	Factor	Solve	Average update (range)		Worst-case
Camel hump	1334	0.031	0.002	0.00007	(1–1000)	0.0002
Camel mouth	3210	0.101	0.006	0.0002	(1–3000)	0.0003
Camel leg	3220	0.121	0.006	0.0002	(1–3000)	0.0004
Camel head	11381	0.503	0.029	0.0010	(1–10000)	0.0013
Pig	28747	1.558	0.065	0.0020	(1–28000)	0.0032
Camel	39074	2.096	0.073	0.0021	(1–39000)	0.0034
Feline	49864	2.750	0.110	0.0025	(1–49000)	0.0034
Max Planck	100086	7.713	0.240	0.0110	(1–100000)	0.0120
Igea	134345	11.826	0.444	0.0200	(1–130000)	0.0215

first 1000 eigenvectors of a mesh representing the entire head of the camel, with 11,381 vertices, took about 21 minutes on the 2 GB RAM machine. On larger meshes, the eigenvector computation simply failed due to lack of memory. For example, we were not able to compute more than about 5000 eigenvectors of the 11,381-vertex mesh, even on a machine with 2 GB RAM. We note that MATLAB's `eigs` function uses a state-of-the-art sparse eigensolver called ARPACK [25], which is implemented in Fortran. Thus, this performance is not due to MATLAB's interpreter and nor to a poor choice of algorithm; it is essentially the inherent cost of computing eigenvectors.

Figure 5 summarizes the comparison results for the tested small meshes in the form of rate-distortion curves. Typically, up to 10 - 20% of the n eigenbasis vectors are needed for visually lossless reconstruction. As suggested by Karni and Gotsman [7], we quantized the spectral coefficients to 14 bits. Stronger quantization leads to distortion of the reconstructed shape even when more than 50% of the full basis is used, since quantization in the transformed domain behaves differently than quantization in the standard basis. The spectral coefficients were compressed with an arithmetic encoder. The rate-distortion curves report three error metrics as a function of the file size of the compressed geometry: the max-norm error, the L^2 error measured by the Metro tool [26] and a simple RMS of distance between the mesh vertices. The graphs show that our method does a better job in terms of the max-norm metric, which is perhaps not surprising because the anchor selection scheme specifically aims at minimizing this norm. As for the L^2 and simple RMS metrics, the two algorithms perform practically the same. For the *Camel* hump mesh, which is fairly smooth and featureless, the geometry-oblivious spectral method performs only slightly better.

It should be mentioned that to alleviate the computation problem of the spectral basis, Karni and Gotsman [7] propose to partition the mesh into patches, each of small enough size to make its spectral decomposition feasible. In their subsequent work, Karni and Gotsman [12] use fixed bases, derived from 6-regular connectivity patches. However, partitioning the mesh is prone to visible discontinuity artifacts along the boundaries between the submeshes, similar to the blocking artifact in JPEG encoding. We emphasize that our method is computationally efficient while it achieves nearly equal performance in terms of compression ratios.

VI. CONCLUSIONS AND DISCUSSION

We have presented a method to approximate the geometry of a shape based on its connectivity and a number of anchor vertices. The "tagging" of the anchors, together with the connectivity, yield a geometry-aware basis that spans a subspace which is close to the given shape. The coefficients that approximate the shape in that subspace are readily given by the spatial location of the anchors. Reconstructing the approximated shape only requires the solution of a sparse least-squares problem. The technique is simple and easy to implement given the required linear algebra building blocks. The complexities of the geometry and the connectivity of the irregular mesh are completely hidden by the linear algebra objects, the matrices and the vectors. The efficiency of the technique stems from the existence of sophisticated linear algebra tools, such as sparse-matrix factorizations, updating techniques, and so on.

There are a number promising directions for future work. One is the relationship between the triangle count reduction and geometry encoding [27], [28]. The scheme that we presented is not fully progressive, in the sense that the mesh has always the full connectivity. It would be desirable to find a way to incorporate geometry-aware bases into progressive meshes [29], [30].

Another direction is to study the relation of our bases to non-uniform B-spline bases. We can view these bases along three axes: their orthogonality, their supports, and their applicability to irregular 3D meshes. In this design space, our method can be described as non-orthogonal, globally-supported bases for irregular meshes. The potential computational difficulties that are normally posed by non-orthogonality and global support are avoided here since we do not compute the basis vectors explicitly.

Our method can also be viewed as a family of regularization methods for a discrete ill-posed problem [31]. Any lossy geometry-encoding method tries to describe mesh functions with many degrees of freedom using relatively little data. Therefore, any such method is by definition ill-posed (many shapes have the same compressed representation). In our case, the data consists of the indices of the anchors and the values that the shape attains there. To reconstruct a unique shape from this data, one must add a side condition. The condition that we attach is a smoothness condition, that we impose in this irregular discrete case using the Laplacian matrix. To improve smoothness even further and/or to make the algorithmic challenges more manageable, we can relax the equality constraints at the anchors and replace them with either penalties or box constraints. This viewpoint would lead to exactly the same algorithms that we have developed in this paper.

The smoothness side condition that regularizes the reconstruction is clearly unsuitable for models that are not smooth. Our method is, however, suitable for models with localized sharp features, as long as many anchors are used in the vicinity of the sharp features. The reproduction of sharp features near anchors can be controlled by the weights of smoothness constraints versus the weights of location constraints.

We believe that this work contributes to a more profound understanding of shapes represented by irregular meshes. There is a broad spectrum of techniques to select a basis for effectively representing geometry, ranging from splines and parametric free-form surfaces to wavelet bases for image encoding. Recently, researchers began proposing using over-complete bases. This technique, known as basis pursuit [32], starts with a large and redundant set of basis vectors, and uses an optimization algorithm to try to find a combination of very few basis vectors that well approximate a given input vector (shape). This can sometimes lead to very sparse representations, but the costs of generating the basis vectors and finding a sparse representation are considerable. In that context, our method can be seen as a specific over-complete basis, and as a way to generate a sparse representation without resorting to an optimization or search algorithm.

Acknowledgment

We thank the editor and the reviewers for their comments and suggestions on this work. Models are courtesy of Cyberware, Stanford University and Max-Planck-Institut für Informatik. This work was supported in part by grants 572/00 and 8001/02 from the Israel Science Foundation (founded by the Israel Academy of Sciences and Humanities), by grant 2002261 from the US-Israeli Binational Science Foundation, by the Israeli Ministry of Science, by an IBM Faculty Partnership Award and by the German Israel Foundation (GIF).

References

[1] D. Luebke, B. Watson, J. D. Cohen, M. Reddy, and A. Varshney, *Level of Detail for 3D Graphics*. Elsevier Science Inc., 2002.

[2] T. Lyche, “Knot removal for spline curves and surfaces,” in *Approximation Theory VII*, E. W. Cheney, C. K. Chui, and L. L. Schumaker, Eds. Academic Press, Boston, 1993, pp. 207–227.

[3] M. Lounsbery, T. D. DeRose, and J. Warren, “Multiresolution analysis for surfaces of arbitrary topological type,” *ACM Transactions on Graphics*, vol. 16, no. 1, pp. 34–73, January 1997.

[4] A. Khodakovsky, P. Schröder, and W. Sweldens, “Progressive geometry compression,” in *Proceedings of ACM SSIGGRAPH 2000*, 2000, pp. 271–278.

[5] L. Kobbelt, “Discrete fairing and variational subdivision for freeform surface design,” *The Visual Computer*, vol. 16, no. 3-4, pp. 142–158, 2000.

[6] N. Litke, A. Levin, and P. Schröder, “Fitting subdivision surfaces,” in *IEEE Visualization 2001*, 2001, pp. 319–324.

[7] Z. Karni and C. Gotsman, “Spectral compression of mesh geometry,” in *Proceedings of ACM SIGGRAPH 2000*, July 2000, pp. 279–286.

[8] P. H. Chou and T. H. Meng, “Vertex data compression through vector quantization,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, no. 4, pp. 373–382, 2002.

[9] O. Sorkine, D. Cohen-Or, and S. Toledo, “High-pass quantization for mesh encoding,” in *Proceedings of ACM/Eurographics Symposium on Geometry Processing*, Aachen, Germany, 2003.

[10] M. Fiedler, “Algebraic connectivity of graphs,” *Czech. Math. Journal*, vol. 23, pp. 298–305, 1973.

[11] G. Taubin, “A signal processing approach to fair surface design,” in *Proceedings of SIGGRAPH 95*, 1995, pp. 351–358.

[12] Z. Karni and C. Gotsman, “3D mesh compression using fixed spectral bases,” in *Graphics Interface 2001*. Canadian Information Processing Society, 2001, pp. 1–8.

[13] H. Zhang and E. Fiume, “Butterworth filtering and implicit fairing of irregular meshes,” in *Proceedings of Pacific Graphics 2003*, 2003, pp. 502–506.

[14] R. Ohbuchi, A. Mukaiyama, and S. Takahashi, “A frequency-domain approach to watermarking 3d shapes,” *Computer Graphics Forum*, vol. 21, no. 3, pp. 373–382, 2002.

[15] C. Gotsman, X. Gu, and A. Sheffer, “Fundamentals of spherical parameterization for 3D meshes,” in *Proceedings of ACM SIGGRAPH 2003*, 2003, pp. 358–363.

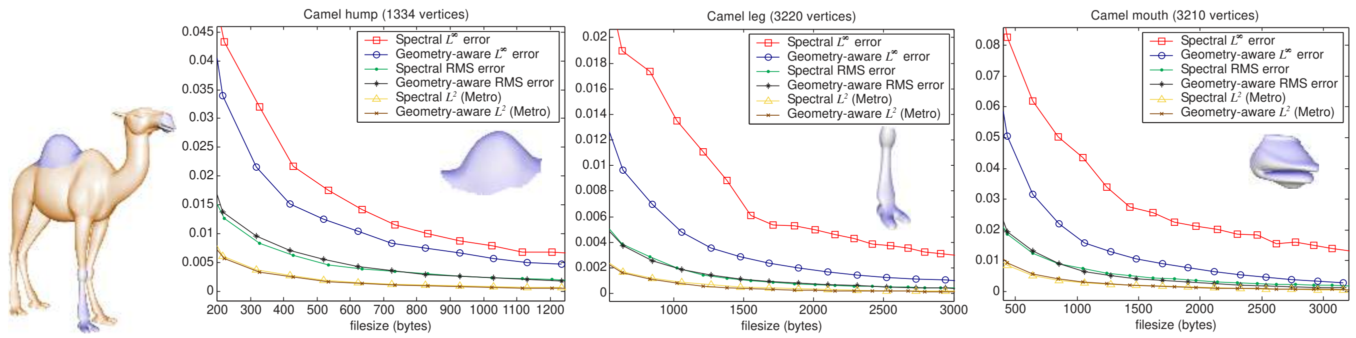
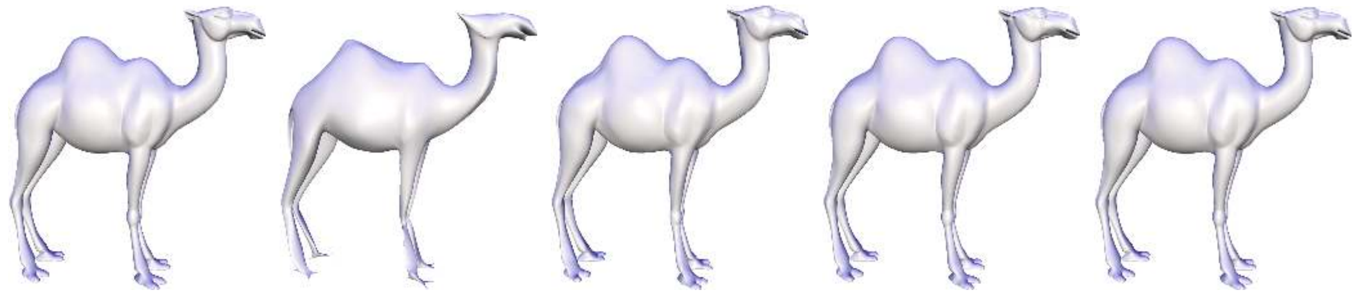


Fig. 5. Rate-distortion curves for small parts of the *Camel* model. The graphs display different error measures: L^∞ stands for $\max_i \|\mathbf{p}_i - \tilde{\mathbf{p}}_i\|$ where $\mathbf{p}_i = (x_i, y_i, z_i)$; RMS stands for the root-mean-square geometric distance between corresponding vertices in the original and approximated models; L^2 error was measured using the Metro tool. Our experiments show that the geometry-aware approximation method is very close to the spectral method in its performance. The L^∞ error of our method tends to be smaller, while the L^2 error is practically the same.



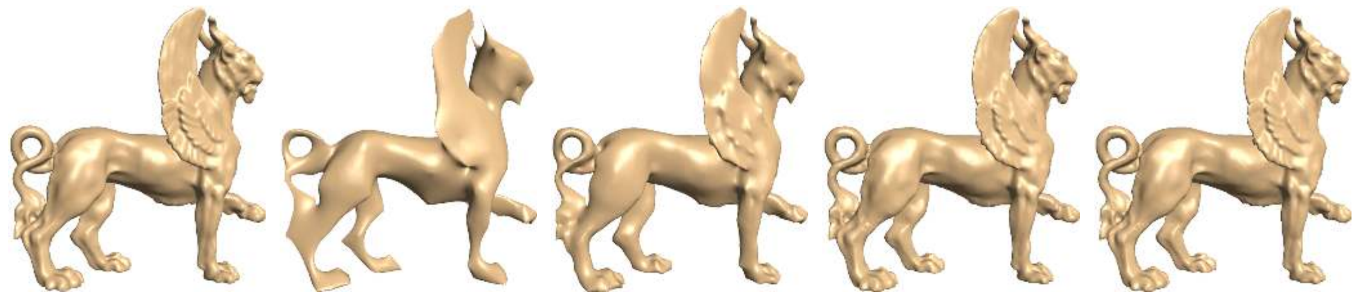
Original model, 39074 vertices

100 basis vectors, e=0.01 — 0.5KB (0.10 bits/vertex)

600 basis vectors, e=0.0022 — 3.3KB (0.69 bits/vertex)

1200 basis vectors, e=$9.8\cdot10^{-4}$ — 6.7KB (1.40 bits/vertex)

3600 basis vectors, e=$2.07\cdot10^{-4}$ — 19.8KB (4.15 bits/vertex)



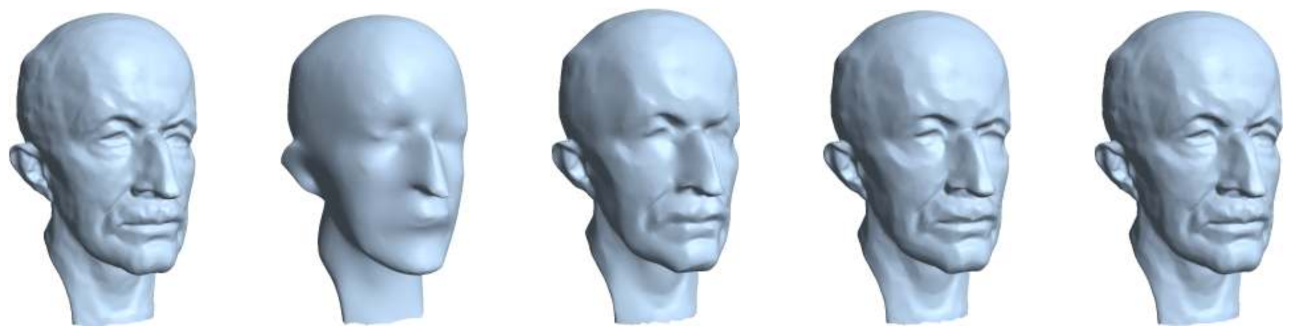
Original model, 49864 vertices

100 basis vectors, e=0.0098 — 0.6KB (0.09 bits/vertex)

500 basis vectors, e=0.0034 — 2.8KB (0.46 bits/vertex)

4000 basis vectors, e=0.0012 — 22.2KB (3.65 bits/vertex)

9000 basis vectors, e=$7.2\cdot10^{-4}$ — 50.1KB (8.23 bits/vertex)



Original model, 100086 vertices

100 basis vectors, e=0.0078 — 0.6KB (0.05 bits/vertex)

1000 basis vectors, e=0.0027 — 6.1KB (0.50 bits/vertex)

3000 basis vectors, e=0.0013 — 18.2KB (1.49 bits/vertex)

10000 basis vectors, e=$4.22\cdot10^{-4}$ — 60.5KB (4.95 bits/vertex)

Fig. 6. Reconstruction of several models using an increasing number of geometry-aware basis vectors. The sizes of the encoded geometry files are displayed below the models. The letter e denotes the L^2 error value. Refer to Table I for the timings.

[16] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr, "Implicit fairing of irregular meshes using diffusion and curvature flow," in *Proceedings of ACM SIGGRAPH 99*, Aug.8–13 1999, pp. 317–324.

[17] M. Isenburg, S. Gumhold, and C. Gotsman, "Connectivity shapes," in *Proceedings of IEEE Visualization 2001*, 2001, pp. 135–142.

[18] O. Sorkine and D. Cohen-Or, "Least-squares meshes," in *Proceedings of Shape Modeling International*. IEEE Computer Society Press, 2004, pp. 191–199.

[19] M. Adlers, "Sparse least squares problems with box constraints," Division of Numerical Analysis, Department of Mathematics, Linköpings Universitet, Linköping, Sweden, Linköping Studies in Science and Technology (Theses) 689, 1988.

[20] T. A. Davis and W. W. Hager, "Modifying a sparse cholesky factorization," *SIAM Journal on Matrix Analysis and Applications*, vol. 20, no. 3, pp. 606–627, 1999.

[21] N. Alon, P. Seymour, and R. Thomas, "A separator theorem for nonplanar graphs," *Journal of the American Mathematical Society*, vol. 3, pp. 801–808, 1990.

[22] D. L. James and D. K. Pai, "Artdefo: accurate real time deformable objects," in *Proceedings of ACM SIGGRAPH 99*, 1999, pp. 65–72.

[23] D. Chen, D. Cohen-Or, O. Sorkine, and S. Toledo, "Algebraic analysis of high-pass quantization," Tel Aviv University," Technical Report, May 2004.

[24] S. Toledo, TAUCS*: A Library of Sparse Linear Solvers, version 2.2*, Tel-Aviv University, Available online at http://www.tau.ac.il/~stoledo/taucs/, Sept. 2003.

[25] R. B. Lehoucq, D. C. Sorensen, and C. Yang, *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. Philadelphia: SIAM, 1998.

[26] P. Cignoni, C. Rocchini, and R. Scopigno, "Metro: Measuring error on simplified surfaces," *Computer Graphics Forum*, vol. 17, no. 2, pp. 167–174, 1998.

[27] D. King and J. Rossignac, "Optimal bit allocation in compressed 3D models," *Computational Geometry, Theory and Applications*, vol. 14, no. 1-3, pp. 91–118, 1999.

[28] P. Alliez and C. Gotsman, "Recent advances in compression of 3D meshes," in *Proceedings of the Symposium on Multiresolution in Geometric Modeling*, september 2003.

[29] H. Hoppe, "Progressive meshes," in *Proceedings of ACM SIGGRAPH 96*, August 1996, pp. 99–108.

[30] J. C. Xia and A. Varshney, "Dynamic view-dependent simplification for polygonal models," in *Proceedings of IEEE Visualization '96*, 1996, pp. 327–334.

[31] P. C. Hansen, *Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion*. Philadelphia: SIAM, 1997.

[32] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1999.

Olga Sorkine received the BSc degree in mathematics and computer science from Tel Aviv University in 2000. Currently, she is a PhD student at the School of Computer Science at Tel Aviv University. Her research interests are in computer graphics and include shape modeling, mesh processing and approximation.

Daniel Cohen-Or is an Associate Professor at the School of Computer Science at Tel Aviv University. He received a BSc in both Mathematics and Computer Science (1985), an MSc in Computer Science (1986) from Ben-Gurion University, and a PhD from the Department of Computer Science (1991) at State University of New York at Stony Brook. His current research interests include rendering, visibility, shape modeling and image synthesis.

Dror Irony is a PhD student in the School of Computer Science at Tel Aviv University. He received his BSc in mathematics and computer science in 1996 and his MSc in computer science in 2000, both from Tel Aviv University. Dror's Master thesis dealt with a new parallel communication-efficient dense linear solver and some related theoretic and practical results. His research today is focused in stable direct algorithms for sparse and banded matrices. From 1996 until 2000, Dror worked for Motorola Communication Israel.

Sivan Toledo is an associate professor of Computer Science at Tel Aviv University. He received his BSc and MSc from Tel Aviv University, both in 1991. He received his PhD from MIT in 1995, and worked as a postdoctoral associate at the IBM TJ Watson Research Center and at the Xerox Palo Alto Research Center before joining Tel Aviv University in 1998.