Algebraic Analysis of High-Pass Quantization

DORON CHEN, DANIEL COHEN-OR, OLGA SORKINE, and SIVAN TOLEDO
Tel-Aviv University

This article presents an algebraic analysis of a mesh-compression technique called *high-pass quantization* [Sorkine et al. 2003]. In high-pass quantization, a rectangular matrix based on the mesh topological Laplacian is applied to the vectors of the Cartesian coordinates of a polygonal mesh. The resulting vectors, called δ-coordinates, are then quantized. The applied matrix is a function of the topology of the mesh and the indices of a small set of mesh vertices (anchors) but not of the location of the vertices. An approximation of the geometry can be reconstructed from the quantized δ-coordinates and the spatial locations of the anchors. In this article, we show how to algebraically bound the reconstruction error that this method generates. We show that the small singular value of the transformation matrix can be used to bound both the quantization error and the rounding error which is due to the use of floating-point arithmetic. Furthermore, we prove a bound on this singular value. The bound is a function of the topology of the mesh and of the selected anchors. We also propose a new anchor-selection algorithm, inspired by this bound. We show experimentally that the method is effective and that the computed upper bound on the error is not too pessimistic.

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

General Terms: Algorithms

Additional Key Words and Phrases: Topological Laplacian operator, quantization, mesh geometry

1. INTRODUCTION

High-pass mesh quantization is a compression technique for three-dimensional polygonal meshes. This technique assumes that the connectivity of the mesh has already been encoded, and that it is, therefore, known to both the encoder and to the decoder. The goal of the technique is to compactly encode the coordinates of the vertices of the mesh. High-pass quantization, which was recently proposed by Sorkine et al. [2003], encodes the coordinates by applying a linear transformation based on the mesh Laplacian to the coordinates and quantizing the transformed coordinates. The decoder then applies another transformation to recover an approximation of the original coordinates from the quantized transformed data. The advantage of encoding the transformed coordinates lies in the fact that they can be aggressively quantized without introducing visually disturbing errors. As shown in Sorkine et al. [2003], the quantization error is mostly comprised of low-frequency bands, while the high-frequency components of the reconstructed surface are preserved. Since humans are usually more sensitive to changes in lighting

This work was supported in part by Grants from the Israel Science Foundation (founded by the Israel Academy of Sciences and Humanities), by the Israeli Ministry of Science, by an IBM Faculty Partnership Award, by a Grant from the US-Israel Binational Science Foundation, and by the German Israel Foundation (GIF).
Authors' address: School of Computer Science, Tel-Aviv University, Tel-Aviv 69978 Israel; email: {mycroft,dcor,sorkine,stoledo}@tau.ac.il.

(or normals) and the local high-frequency details of the surface, low-frequency errors are perceived as less visible.

Applying the mesh Laplacian to the coordinate prior to quantization is a bad idea. In high-pass quantization, we do not apply the Laplacian itself but rather a carefully constructed operator derived from it. The construction aims to control two aspects of the compression and decompression process. First, the Laplacian is singular, and it tends to be ill conditioned on large meshes. The singularity reflects the fact that Laplacian-transformed coordinates do not prescribe the absolute positioning of the mesh in space; this singularity is easy to handle. But the ill conditioning is more difficult to handle. If not addressed, the ill conditioning leads to a decompression operator with a large norm which greatly amplifies even small quantization errors. Our construction addresses the ill conditioning using so-called anchor points in the mesh. Anchor points are mesh points whose original coordinates are included in the encoded (transformed) mesh coordinates. In this article, we show how to estimate the condition of the Laplacian-derived operator from the connectivity of the mesh and the identity of the anchors. By adding anchors appropriately, we control the norm of the quantization error.

The shape of the error is the other aspect of the compression process that our construction aims to control. Laplacian coordinates, with or without anchors, can be thought of as smoothness constraints that the decompressor tries to satisfy. A small Laplacian coordinate at a mesh vertex implies that the mesh is smooth around that vertex, and a large Laplacian coordinate implies local roughness. Anchors add constraints on absolute positioning of the anchor vertices to the decompression process. The key to high-pass quantization is to use both smoothness and absolute positioning constraints at the anchors. This is what controls the shape of the quantization error.

Sorkine et al. [2003] presented the algebraic framework of high-pass quantization together with a partial argument that explained why it works well. More specifically, that argument showed how the eigenvalues of the linear transformations that the encoder and the decoder apply effect the quantization error. However, the analysis in Sorkine et al. [2003] is incomplete: (1) the analysis only applies to one class of matrices (so-called k-anchor invertible Laplacians) but not to the matrices that are actually used in the algorithm (k-anchor rectangular Laplacians); (2) the eigenvalues of the transformations are not analyzed; and (3) the effect of rounding errors on encoding and decoding is not analyzed. In this article we rectify the deficiencies of Sorkine et al. [2003]. In particular, we extend the analysis to show that the singular values of rectangular Laplacians can bound the encoding error, we present bounds on the eigenvalues and singular values of Laplacians, and we bound the effect of rounding errors on the method. The bounds that we derive for the singular/eigenvalues and for the encoding and rounding errors are given in terms of topological properties of the mesh, so the bounds are relatively easily to estimate. These topologically-derived bounds are also useful for selecting anchors, the extra vertices whose coordinates are used to decode the mesh.

We complement this analysis with a new anchor-selection algorithm and with experimental results. The new algorithm selects anchor points so as to minimize our theoretical error bound. The new experimental results further strengthen the claims in Sorkine et al. [2003] concerning the effectiveness of high-pass quantization, and they show how our theoretical bounds relate to the actual encoding errors. It should be noted that the bounds on the condition number of k-anchor rectangular Laplacians are useful for evaluating any methods based on such matrices such as mesh editing with differential coordinates [Lipman et al. 2004].

2. BACKGROUND: MESH COMPRESSION

Mesh compression involves two problems that are usually solved, at least conceptually, separately: the mesh *connectivity* encoding and the *geometry* encoding. While state-of-the-art connectivity encoding techniques are extremely effective [Touma and Gotsman 1998; Gumhold 2000; Alliez and Desbrun

2001; Khodakovsky et al. 2002], compressing the geometry remains a challenge. The encoded geometry is, on average, at least five times larger than the encoded connectivity even when the coordinates are prequantized to 10–12 bits. Finer quantization for higher precision increases the importance of effective geometry encoding even further.

Earlier works on geometry compression employed prediction-correction coding of quantized vertex coordinates. Linear predictors are usually used; the most common one is known as the parallelogram predictor [Touma and Gotsman 1998]. The displacements are compressed by some entropy encoder. Chou and Meng [2002] use vector quantization instead to gain speed.

Recent compression methods represent the mesh geometry using effective bases such as the spectral basis [Karni and Gotsman 2000] which generalizes the Fourier basis functions to irregular connectivity, or the wavelet basis [Khodakovsky et al. 2000]. The spectral encoding of Karni and Gotsman [2000] preserves the original connectivity of the mesh, and relies on the fact that it is known both to the encoder and the geometry decoder (this is also the case with the high-pass quantization method). The mesh compression framework of Khodakovsky et al. [2000] requires semiregular remeshing of the input mesh. While their method achieves excellent compression ratios, it is not connectivity-lossless which puts this work in a somewhat different category. In many cases, it is desirable to preserve the original connectivity of the mesh, especially when it carefully models certain features and is particularly adapted to the surface geometry. For a recent survey on mesh compression techniques, the reader is referred to Alliez and Gotsman [2005].

3. BACKGROUND: HIGH-PASS QUANTIZATION

This section reviews the high-pass mesh quantization method [Sorkine et al. 2003] that our article analyzes.

Sorkine et al. [2003] proposed a new approach to geometry quantization that works meshes with arbitrary connectivity. Instead of directly quantizing the Cartesian coordinates which may lead to errors that damage the high-frequency details of the surface, they proposed to first transform the coordinates to another space by applying the Laplacian operator associated with the mesh topology. The transformed coordinates are called δ-coordinates. The quantization is applied to the δ-coordinates, and the geometry of the mesh can be restored on the decoder side by solving a linear least-squares system defined by the extended Laplacian matrix which is described later in this section. They showed that introducing high-frequency errors by quantizing the δ-coordinates results in low-frequency errors in the reconstructed Cartesian coordinates and argued that low-frequency displacements in the surface geometry are less noticeable to the human eye than high-frequency displacements.

3.1 Quantization Errors Under Linear Transformations

Quantizing a vector x with continuous coefficients introduces an error q_x, where $x + q_x$ is the quantized vector. In this section, we show how to control the spectral behavior of the error using linear transformations. We assume that a simple fixed-point quantization is used so that the maximum quantization error $\max_i |q_i|$ is bounded by the expression $2^{-p}(\max_i x_i - \min_j x_j)$, using p-bit quantized coefficients.

Suppose that, instead of quantizing the input vector x, we first transform x into a vector Ax, using a nonsingular matrix A, and then quantize Ax. We denote the quantization error by q_{Ax} so that the new quantized vector is $Ax + q_{Ax}$. The elements of the quantized vector are now discrete as are those of $x + q_x$. We can recover an approximation of x from this representation by multiplying the quantized vector by A^{-1}:

$$A^{-1}(Ax + q_{Ax}) = x + A^{-1}q_{Ax}.$$

The error in this approximation is $A^{-1}q_{Ax}$, and we will shortly see that, under certain conditions, it behaves quite differently than q_x.

Assume that A has an orthonormal eigen-decomposition $AU = U\Lambda$, where U is unitary and Λ is diagonal. This assumption is satisfied when A is real and symmetric. Without loss of generality, we assume that $|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n|$, where $\lambda_i = \Lambda_{ii}$ are the eigenvalues of A. Since the processes we are concerned with are invariant to scaling A, we also assume that $|\lambda_1| = 1$. We express x as a linear combination of A's orthonormal eigenvectors, $x = c_1u_1 + c_2u_2 + \cdots + c_nu_n$, where u_i are the columns of U. We also have $Ax = c_1\lambda_1u_1 + c_2\lambda_2u_2 + \cdots + c_n\lambda_nu_n$. Similarly, since $A^{-1}U = U\Lambda^{-1}$, we can express the quantization error as $q_{Ax} = c'_1u_1 + c'_2u_2 + \cdots + c'_nu_n$, so

$$A^{-1}q_{Ax} = c'_1\lambda_1^{-1}u_1 + c'_2\lambda_2^{-1}u_2 + \cdots + c'_n\lambda_n^{-1}u_n.$$

The transformation A is useful for quantization when three conditions hold.

(1) For typical inputs x, the norm of Ax is much smaller than the norm of x,
(2) Quantization errors with large $c'_i\lambda_i^{-1}$ for large i (i.e., with strong representation for the last eigenvectors) are not disturbing,
(3) $|\lambda_n|$ is not too small.

The first point is important since it implies that $\max_i |(Ax)_i| \ll \max_i |x_i|$ which allows us to achieve a given quantization error with fewer bits. The best choice of norm for this purpose is, of course, the max norm, but since norms are essentially equivalent, the implication also holds if $\|Ax\|_2 \ll \|x\|_2$. Since $\|x\|_2^2 = \sum_i c_i^2$ and $\|Ax\|_2^2 = \sum_i c_i^2\lambda_i^2$, the above condition occurs if and only if the first c_i's are small compared to the last ones. In other words, the first point holds if A, viewed as a filter, filters out strong components of typical x's.

The importance of the second and third points stems from the fact that A^{-1} amplifies the components of q_{Ax} in the direction of the last eigenvectors. If A has tiny eigenvalues, the amplification by a factor λ_i^{-1} is significant for large i. Even if the small eigenvalues of A are not tiny, the error may be unacceptable. The quantization error $A^{-1}q_{Ax}$ always contains moderate components in the direction of eigenvectors that correspond to the small eigenvalues of A. When small error components in these directions distort the signal perceptively, the error will be unacceptable. Therefore, the last two points must hold for the quantization error to be acceptable.

It may seem that the norm of Ax is irrelevant to compression since one can shrink Ax by a simple scaling which is clearly useless for compression. The norm of Ax is relevant because we also demand that $|\lambda_1| = \|A\|_2 = 1$. The error is $A^{-1}q_{Ax}$ so $\|A^{-1}q_{Ax}\| \leq \|A^{-1}\|\|q_{Ax}\|$. Making Ax small by scaling A is useless because it will shring $\|q_{Ax}\|$ but will expand $\|A^{-1}\|$ by exactly the same factor. But making Ax small while maintaining $\|A\|_2 = 1$ is useful.

3.2 Laplacian Transformations

In the following, we discuss the Laplacian matrix of the mesh and its variants and show that these linear transformations work well as quantization transforms.

Let M be a given triangular mesh with n vertices. Each vertex $i \in M$ is conventionally represented using absolute Cartesian coordinates, denoted by $v_i = (x_i, y_i, z_i)$. We denote the *relative* or *δ-coordinates* of v_i as follows:

$$\delta_i = \left(\delta_i^{(x)}, \delta_i^{(y)}, \delta_i^{(z)}\right) = d_iv_i - \sum_{k=1}^{d} v_{i_k},$$

where d_i is the degree of vertex i, and i_k is i's k th neighbor. The transformation of the vector of absolute Cartesian coordinates to the vector of relative coordinates can be represented by the matrix $L = D - A$, where A is the mesh adjacency matrix, and D is the diagonal matrix $D_{ii} = d_i$.

The matrix L is called the *Laplacian* of the mesh [Fiedler 1973]. Laplacians of meshes have been extensively studied [Chung 1997] primarily because their algebraic properties are related to the combinatorial properties of the meshes they represent. The Laplacian is symmetric, singular, and positive semidefinite. The singularity stems from the fact that the system $Lx = \delta$ has an infinite number of solutions which differ from each other by a vector that is constant on each connected component of the mesh. Thus, we can actually recover x from δ if we know, in addition to δ, the Cartesian coordinate of one x_i in each connected component. We can formalize this method by dropping from L the rows and columns that correspond to one vertex in each connected component called the *anchor* of the component. The resulting matrix, which we call the *basic invertible Laplacian*, generates all the δ's that we need and is nonsingular. The next section explores other nonsingular variants of the Laplacian.

To explain why variants of the Laplacian are effective quantization transforms, we first have to introduce the notion of mesh frequencies (spectrum). The *frequency* of a real function x defined on the vertices of a mesh M is the number of zero crossings along edges,

$$f(x) = \sum_{(i,j)\in E(M)} \left\{ \begin{array}{ll} 1 & x_i x_j < 0 \\ 0 & \text{otherwise} \end{array} \right\},$$

where $E(M)$ is the set of edges of M so the summation is over adjacent vertices. It turns out that, for many classes of graphs including 3D meshes, eigenvectors of the Laplacian (and related matrices, such as our basic invertible Laplacian) corresponding to large eigenvalues are high-frequency mesh functions, and eigenvectors corresponding to small eigenvalues are low-frequency mesh functions. In other words, when $i \ll j$, $\lambda_i > \lambda_j$ and $f(u_i) \gg f(u_j)$. Furthermore, since 3D models are typically smooth, possibly with some relatively small high-frequency perturbation, the coordinate vectors x, y, and z often have a large low-frequency and a small high-frequency content. That is, the first c_i's are often very small relative to the last ones.

This behavior of the eigenvectors of Laplacians and of typical 3D models implies that the first property we need for effective quantization holds, namely, the 2-norm of Lx is typically much smaller than the norm of x, and therefore the dynamic range of Lx is smaller than that of x. Laplacians also satisfy the second requirement. As stated previously, eigenvectors associated with small eigenvalues are low-frequency functions that are typically very smooth. When we add such smooth low-frequency errors to a 3D model, large features of the model may slightly shift, scale, or rotate but the local features and curvature are maintained. Thus, errors consisting mainly of small-eigenvalue low-frequency eigenvectors are not visually disturbing.

However, simple Laplacian transformations do not satisfy our third requirement. The small eigenvalue of a basic invertible Laplacian is typically tiny; a good estimate for $|\lambda_n^{-1}|$ is the product of the maximum topological distance of a vertex from the anchor vertex and the number of vertices in the mesh (assuming there is one connected component; otherwise the maximum of this estimate over all components) [Guattery and Miller 2000; Boman and Hendrickson 2003]. For a typical n-vertex 3D mesh, the small eigenvalue is, therefore, likely to be $\Theta(n^{-1.5})$. This causes large low-frequency errors which are clearly visible in the example in Figure 1.

3.3 The *k*-anchor Laplacian

An effective way to increase the small eigenvalue of a Laplacian is to add more anchor points. This section analyzes the effect of two algorithm parameters on the magnitude and shape of the quantization error. One parameter is the number and location of the anchor points. The second parameter is the algorithm that transforms the relative (or δ) coordinates to the original coordinates.

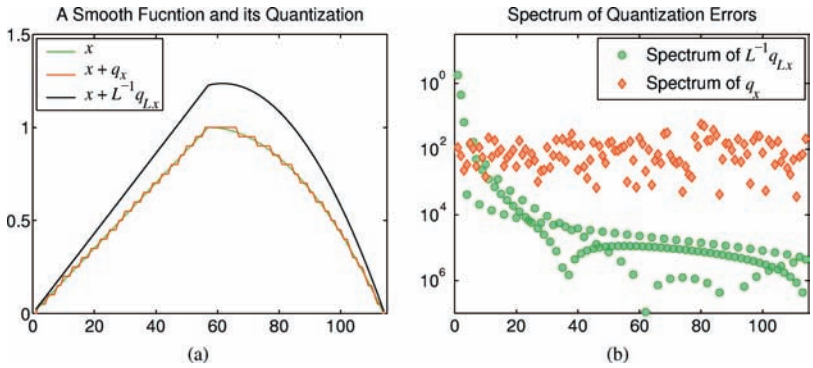

Fig. 1. An example of quantization errors in a one-dimensional mesh. The mesh here is a simple chain with 114 vertices (enumerated on the x-axis). (a) shows a smooth function x defined on the mesh, its direct quantization, and a Laplacian-transform quantization. The specific Laplacian that we use here is the 2-anchor invertible Laplacian, defined in Section 3.3, with anchors at vertices 1 and 114. The quantizations were performed with 20 discrete values uniformly distributed between the minimum and maximum absolute values of the vectors. The direct error vector is smaller in magnitude but has a strong high-frequency oscillatory nature, whereas the Laplacian-transformed error vector is smooth. (b) explains this observation by plotting, on a log scale the spectrum of the two errors. We can see that the direct quantization has moderate components in the direction of all eigenvectors of the Laplacian (i.e., all frequencies), whereas the Laplacian-transformed error has strong components in the direction of the smooth eigenvectors but very small components in the direction of high-frequency eigenvectors.

The relationship between the original coordinates x and the relative coordinates δ is given, up to a shift, by the linear system of equations $Lx = \delta$. When we add anchors, we essentially add constraints to this system of equations. Without loss of generality, we assume that the anchors are $x_1, \ldots, x_k$, the first k vertices of the mesh. For each anchor point x_{i_j}, $j = 1, \ldots, k$, we add the constraint $x_i = x_i$, where the left-hand side is taken to be an unknown and the right-hand side a known constant.

It may seem strange that we do not immediately substitute the known constant for the unknown, but the reason for this will become apparent later. The full system of constraints that defines the relationship between the absolute and relative coordinates is, therefore,

$$\left(\begin{array}{c|c} L \\ \hline I_{k\times k} & 0 \end{array}\right) x = \begin{pmatrix} Lx \\ x_{1:k} \end{pmatrix} = \begin{pmatrix} \delta \\ x_{1:k} \end{pmatrix}. \tag{1}$$

We denote this $(n+k)$-by-n matrix by $\tilde{L}$,

$$\tilde{L} = \left(\begin{array}{c|c} L \\ \hline I_{k\times k} & 0 \end{array}\right), \tag{2}$$

and call it the *k-anchor rectangular Laplacian*.

With k anchors, the quantized representation of the mesh consists of the quantized δ's and of the absolute coordinates of the anchors. Since we take k to be much smaller than n, there is no need to aggressively quantize the coordinates of the anchors, but they can be quantized as well. The quantized vector that represents the mesh is, therefore,

$$\begin{pmatrix} Lx + q_{Lx} \\ x_{1:k} + q_{x_{1:k}} \end{pmatrix} = \begin{pmatrix} Lx \\ x_{1:k} \end{pmatrix} + q_{\tilde{L}x} = \tilde{L}x + q_{\tilde{L}x}. \tag{3}$$

The matrix $\tilde{L}$ is rectangular and full rank. Suppose that we try to recover an approximation x' to x from $\tilde{L}x + q_{\tilde{L}x}$. Trying to compute the approximation x' by solving the constraint system $\tilde{L}x' = \tilde{L}x + q_{\tilde{L}x}$ for x' will fail since this system is overdetermined and, therefore, most likely inconsistent. An approximation x' can be computed in (at least) two ways. The simplest is to eliminate the last k rows from the system.

By adding row $n+j$ to row j, for $j = 1, \ldots, k$ and deleting row $n+j$, we obtain a square symmetric positive definite linear system of equations $\hat{L}x' = b$ which can be solved for x'. This transformation corresponds to multiplying both sides of the system $\tilde{L}x' = \tilde{L}x + q_{\tilde{L}x}$ by an n-by-$(n+k)$ matrix

$$J = \left(I_{n\times n} \left| \frac{I_{k\times k}}{0} \right. \right), \tag{4}$$

so

$$\hat{L} = J\tilde{L} \tag{5}$$

and $b = J(\tilde{L}x + q_{\tilde{L}x})$. We call $\hat{L}$ the *k-anchor invertible Laplacian*.[1]

The second method to obtain an approximation x' is to find the least-square solution x' to the full rectangular system $\tilde{L}x' \approx \tilde{L}x + q_{\tilde{L}x}$. It turns out that the norm of the quantization error is essentially the same in the two approximation methods, but the shape of the error is not. The shape of the error when using a least-squares solution to the rectangular system is smoother and more visually pleasing than the shape of the error resulting from the solution of the square invertible system.

4. ALGEBRAIC ANALYSIS OF *K*-ANCHOR LAPLACIANS

The norm and shape of the quantization errors in high-pass quantization depend on the spectrum and singular vectors of k-anchor Laplacians. This section presents a detailed analysis of the spectrum of these matrices. In particular, we prove bounds on their smallest and largest singular values. We are mostly interested in the spectrum of the k-anchor rectangular Laplacian since we can directly relate these to the magnitude of the quantization errors in high-pass quantization.

The section has two parts. The first part, consisting of Sections 4.1–4.3, bounds the norm of the error in high-pass quantization. The goal of this part of the section is to prove Lemma 4.9 and Theorem 4.11. Lemma 4.9 shows that the norm of the error is related to the smallest singular value of the rectangular k-anchor Laplacian. How small can this singular value be? Theorem 4.11 essentially shows that, if every vertex in the graph is reasonably close to an anchor, then this singular value cannot be small. We prove Theorem 4.11 by first proving a similar bound on the smallest eigenvalue of the invertible k-anchor Laplacian (in Theorem 4.3; this proof is complicated), and then showing how the small singular value of the rectangular Laplacian is related to the small eigenvalue of the invertible Laplacian.

The second part of the section consists of Section 4.4 which discusses the shape of the error.

4.1 The Eigenvalues of $\hat{L}$

In this section, we show how to bound from below the smallest eigenvalue of $\hat{L}$. Bounding the small eigenvalue from below ensures that the transformation $\hat{L}$ satisfies condition (3) in Section 3.1.

The largest eigenvalue $\lambda_{\max}(\tilde{L})$ is at most $2d_{\max} + 1$, where $d_{\max}$ is the maximal degree in the mesh [Chung 1997]. This bound is less important than the lower bound on the small eigenvalue since it only ensures that the norm of the transformed coordinates is never much larger than the norm of the absolute coordinates; in fact, we expect the transformed norm to be much smaller. We include the bound for completeness and also to show that, even when our quantization method is not very effective, it does not cause much harm.

We first show that bounding the spectrum of $\hat{L}$ proves a lower bound on the quantization error $x - x'$. The bound is similar to the analysis of the quantization error in Section 3, but it is not identical. The

[1]This definition of the k-anchor invertible Laplacian is different than the definition given in Sorkine et al. [2003]. The definition that we use here makes the analysis somewhat simpler. The difference is irrelevant to both the algorithms and the analysis since the k-anchor invertible Laplacian is not used in actual mesh encoding; it is only used as a technical tool in the analysis of the k-anchor rectangular Laplacian.

difference, which turns out to be quite minor, stems from the fact that we now quantize an $(n+k)$-vector, not an n-vector.

LEMMA 4.1. *The norm of the quantization error $x - x'$, resulting from solving*

$$\hat{L}x' = J\tilde{L}x' = J(\tilde{L}x + q_{\tilde{L}x}),$$

is bounded by $\|x - x'\|_2 \leq \sqrt{2}\lambda_{\min}^{-1}(\hat{L})\|q_{\tilde{L}x}\|_2$.

PROOF. We add the quantization error $q_{\tilde{L}x}$ to the right-hand side of Equation (1) and multiply both sides by J,

$$J\tilde{L}x' = J(\tilde{L}x + q_{\tilde{L}x}). \tag{6}$$

Because $J\tilde{L}x' = \hat{L}x'$, we can multiply both sides by $\hat{L}^{-1}$ to obtain

$$x' = \hat{L}^{-1}J(\tilde{L}x + q_{\tilde{L}x}) = \hat{L}^{-1}(\hat{L}x + Jq_{\tilde{L}x}) \; = \; x + \hat{L}^{-1}Jq_{\tilde{L}x},$$

so

$$\|x - x'\|_2 \leq \|\hat{L}^{-1}\|_2\|J\|_2\|q_{\tilde{L}x}\|_2.$$

We now bound the first two factors in the right-hand-side product. Because $\hat{L}$ is symmetric positive definite,

$$\|\hat{L}^{-1}\|_2 = \lambda_{\max}(\hat{L}^{-1}) = 1/\lambda_{\min}(\hat{L}) = \lambda_{\min}^{-1}(\hat{L}).$$

By the definition of the 1 and infinity norms,

$$\|J\|_2^2 \leq \|J\|_1\|J\|_\infty = 1 \cdot 2 = 2,$$

which completes the proof. □

This lemma shows that to preserve the bound on the norm of $x - x'$, the quantization error $q_{x_{1:k}}$ for the anchor points should be no larger than the quantization error q_{Lx} of the relative coordinates.

We now bound the smallest eigenvalue of $\hat{L}$. We express the lower bound in terms of a set of paths in the mesh. Given a set of anchor points, we assign each vertex a path to an anchor point. The bound uses the following three metrics of the set of paths.

Definition 4.2. The *dilation* ϑ of the set of paths is the length, in edges, of the longest path in the set. The *congestion* φ of the set is the maximal number of paths that use a single edge in the mesh. The *contention* ρ of the set is the maximal number of vertices whose paths lead to a single anchor point. The maximum is taken over all vertices for dilation, over all edges for congestion, and over all anchors for contention.

The smaller the dilation, congestion, and contention, the better the bound on the small eigenvalue of $\hat{L}$. Note that, for a single set of anchor points, we can assign many different sets of paths, some of which yield tighter bounds than others. In addition, even the best set of paths does not, in general, provide a completely tight bound. For more details, see Boman and Hendrickson [2003]. But the dependence of the bound on the dilation, congestion, and contention does provide us with guidelines as to how to select the anchor points. The next theorem is the main result of this Section.

THEOREM 4.3. *The smallest eigenvalue of $\hat{L}$ satisfies*

$$\lambda_{\min}(\hat{L}) \geq \frac{1}{\varphi \cdot \vartheta + \rho}.$$

We use the following strategy to prove this theorem. We will show how to factor $\hat{L}$ into $\hat{L} = VV^T$. The eigenvalues of $\hat{L}$ are the squares of the singular values of V so it suffices to bound the small singular value of V. The factor V will have a special structure in which each column corresponds to one edge of the mesh or to one anchor point. We will then use the given set of paths from vertices to anchor points to construct a matrix W such that $VW = I$, and show how the norm of W is related to the path structure. The equation $VW = I$ will allow us to relate the 2-norm of W which we can bound using the path set to the small singular value of V which we seek to bound.

The following definitions are used in the construction of the factor V.

Definition 4.4. The *edge-vector* $\langle ij \rangle$ in $\mathbb{R}^n$ is a vector with exactly two nonzeros, $\langle ij \rangle_{\min(i,j)} = 1$ and $\langle ij \rangle_{\max(i,j)} = -1$. The *vertex-vector* $\langle i \rangle$ in $\mathbb{R}^n$ is a vector with exactly one nonzero, $\langle i \rangle_i = 1$.

We associate an edge-vector $\langle ij \rangle$ with an edge connecting vertex i with vertex j. The following lemma demonstrates one of the connections between edges and their corresponding vectors.

LEMMA 4.5. *The edge-vectors of a simple path between vertices i and j span the edge-vector* $\langle ij \rangle$ *with coefficients* ± 1.

The following lemma describes a factorization of k-anchor Laplacian matrices:

LEMMA 4.6. *A k-anchor Laplacian matrix* $\hat{L}$ *can be factored into* $\hat{L} = VV^T$, *such that* $V = (\, V_1 \;\; V_2 \,)$, *where* V_2 *is a matrix of unscaled edge-vectors, each column corresponding to one nonzero off-diagonal in* $\hat{L}$, *and* V_1 *is a matrix of vertex-vectors, each column corresponding to an anchor point.*

PROOF. For each off-diagonal nonzero $\hat{l}_{ij} = -1$ (each edge of the mesh), V has a column containing the edge-vector $\langle ij \rangle$, and for each anchor j, V has a vertex-vector $\langle j \rangle$. The edge-vectors constitute V_2 and the vertex-vectors constitute V_1. It is easy to verify that $\hat{L} = VV^T$. For a more detailed proof, see Boman et al. [2004]. □

Given the above factorization, we bound the smallest singular value of V. Our course of action in bounding the smallest singular value of V is as follows: we shall find a matrix $W \in R^{m \times n}$ such that $VW = I_{n \times n}$. As the next lemma shows, the matrix G with the smallest 2-norm satisfying $VG = I_{n \times n}$ is the Moore-Penrose pseudo-inverse $G = V^+$ of V [Golub and Loan 1996, pages 257–258]. Therefore, any matrix W satisfying $VW = I_{n \times n}$ has the property $\|W\| \geq \|V^+\|$. We shall then find an upper bound C on $\|W\|$. Since $C \geq \|W\| \geq \|V^+\| = \frac{1}{\sigma_{\min}(V)}$, we will be able to conclude that $\sigma_{\min}(V) \geq \frac{1}{C}$. We first prove a technical lemma concerning the pseudo-inverse (this result is probably well-known, but we have not found it in the literature).

LEMMA 4.7. *Let V be a full-rank n-by-m real matrix, and let G be an m-by-n real matrix such that* $VG = I_{n \times n}$. *Then* $\|G\|_2 \geq \|V^+\|_2$.

PROOF. The singular values of V^+V are n ones and $m - n$ zeros, so its 2-norm is 1. We now show that, for any x with unit 2-norm, we have $\|V^+x\|_2 \leq \|Gx\|_2$. Let $c = \|Gx\|_2$, and let $y = Gx/c$, so $\|y\|_2 = 1$. We have $Gx = cy$, and multiplying V from the left on both sides, we get $x = Ix = VGx = cVy$. Multiplying now from the left by V^+, we get $V^+x = cV^+Vy$, so $\|V^+x\|_2 = \|cV^+Vy\|_2 \leq c\|V^+Vy\|_2 \leq c\|V^+V\|_2\|y\|_2 = c \cdot 1 \cdot 1 = c = \|Gx\|_2$. □

We are now ready to bound the singular values of V.

LEMMA 4.8. *Given a k-anchor Laplacian* $\hat{L}$ *with a factorization into edge-and vertex-vectors* $\hat{L} = VV^T$ *as in Lemma* 4.6, *and a set of paths*

$$\Pi = \{\pi_i = (i, i_1, i_2, \ldots, j) | i = 1, \ldots, n \text{ and } j \text{ is an anchor}\},$$

we have

$$\sigma_{\min}(V) \geq \frac{1}{\sqrt{\varphi(\Pi) \cdot \vartheta(\Pi) + \rho(\Pi)}}.$$

PROOF. Finding a matrix W satisfying $VW = I_{n\times n}$ is equivalent to finding a vector w_i for $i = 1, \ldots, n$ such that $Vw_i = e_i$, where $e_i = \langle i \rangle$ is the ith unit vector.

Let j_i be the anchor endpoint of π_i. It is easy to verify that

$$\langle ij_i \rangle = (-1)^{(i>j_i)} \sum_{(\ell_1,\ell_2)\in\pi_i} (-1)^{(\ell_1>\ell_2)} \langle \ell_1\ell_2 \rangle .$$

(We use the convention that a boolean predicate such as $(i > j)$ evaluates to 1 if it is true and to 0 otherwise.) By Lemma 4.6, all the edge-vectors in the summation are columns of V. To obtain w_i, it remains is to add or subtract $\langle j_i \rangle$, and perhaps to multiply by -1,

$$\langle i \rangle = (-1)^{(i>j_i)} \langle ij_i \rangle + \langle j_i \rangle .$$

The last two equations together specify w_i which contains only 1's, -1's, and 0's.

Now that we have found, column by column, a matrix W such that $VW = I_{n\times n}$, we partition the rows of W such that

$$VW = (V_1V_2)\begin{pmatrix} W_1 \\ W_2 \end{pmatrix}.$$

The rows of W_1 correspond to the columns of V_1, the vertex-vectors in V, and the rows of W_2 corresponds to the columns of V_2, the edge-vectors in V. We will bound the norm of W by bounding separately the norms of W_1 and of W_2. We first bound $\|W_1\|_2$:

$$\|W_1\|_2^2 \leq \|W_1\|_1 \, \|W_1\|_\infty = \left(\max_j \sum_i |[W_1]_{ij}|\right)\left(\max_i \sum_j |[W_1]_{ij}|\right) = 1 \cdot \rho(\Pi).$$

The 1-norm of W_1 is 1 since there is exactly 1 nonzero in each column of i, in position j_i, and its value is 1. The ∞-norm of W_1 is the contention of the path set since each row of W_1 corresponds to one anchor point, and it appears with value 1 in each path (column) that ends in it. Therefore, each row in W_1 contains at most $\rho(\Pi)$ 1's, and the other entries are all 0.

Bounding $\|W_2\|_2$ is similar. Each row in W_2 corresponds to one edge of the mesh and each column to a path in Π. Each edge is used in at most $\varphi(\Pi)$ paths, so $\|W_2\|_\infty = \varphi(\Pi)$. Each path contains at most $\vartheta(\Pi)$ edges, so $\|W_2\|_1 = \vartheta(\Pi)$. Therefore,

$$\begin{aligned}
\|W\|_2^2 &= \max_{\|x\|_2=1} \|Wx\|_2^2 \\
&= \max_{\|x\|_2=1} \left\| \begin{matrix} W_1x \\ W_2x \end{matrix} \right\|_2^2 \\
&= \max_{\|x\|_2=1} \left(\|W_1x\|_2^2 + \|W_2x\|_2^2\right) \\
&\leq \max_{\|x_1\|_2=1} \|W_1x_1\|_2^2 + \max_{\|x_2\|_2=1} \|W_2x_2\|_2^2 \\
&= \|W_1\|_2^2 + \|W_2\|_2^2 \\
&\leq \varphi(\Pi)\vartheta(\Pi) + \rho(\Pi).
\end{aligned}$$

The bound on $\sigma_{\min(V)}$ follows immediately from the bound on $\|W\|_2$ and from the discussion preceding the statement of the lemma. □

Now we can conclude that $\lambda_{\min}(\hat{L}) \geq \frac{1}{\varphi \cdot \vartheta + \rho}$. This follows from two facts: (1) in a symmetric positive definite matrix, the singular values are the same as the eigenvectors, therefore $\lambda_{\min}(\hat{L}) = \sigma_{\min}(\hat{L})$; (2) if $\hat{L} = VV^T$, then the singular values of $\hat{L}$ are the squares of the singular values of V (this follows directly from V's SVD decomposition).

We can now easily prove Theorem 4.3:

PROOF. $\lambda_{\min}(\hat{L}) = \sigma_{\min}(\hat{L}) = \sigma^2_{\min}(V) \geq \frac{1}{\varphi \cdot \vartheta + \rho}$. □

4.2 Bounding the Quantization Error Using the Singular Values of $\tilde{L}$

We now show that, if we define x' as the least-squares minimizer of $\|\tilde{L}x' - \tilde{L}x + q_{\tilde{L}x}\|_2$, the norm of the error $x - x'$ can be bounded using estimates on the singular values of $\tilde{L}$. The analysis that follows is equivalent to the analysis in Lemma 4.1, but for the case of $\tilde{L}$, the k-anchor rectangular Laplacian rather than for the case of $\hat{L}$, the square k-anchor invertible Laplacian.

LEMMA 4.9. *Let x' be the least-squares minimizer of $\|\tilde{L}x' - \tilde{L}x + q_{\tilde{L}x}\|_2$. The norm of the error $x - x'$ is bounded by*

$$\|x - x'\|_2 \leq \sigma^{-1}_{\min}(\tilde{L}) \|q_{\tilde{L}x}\|_2,$$

where $\sigma_{\min}(\tilde{L})$ denotes the nth and smallest singular value of $\tilde{L}$.

PROOF. We express x' in terms of the Moore-Penrose pseudo-inverse $\tilde{L}^+$ of $\tilde{L}$,

$$x' = \tilde{L}^+ \left(\tilde{L}x + q_{\tilde{L}x}\right) = x + \tilde{L}^+ q_{\tilde{L}x}.$$

Therefore,

$$\|x - x'\|_2 \leq \|\tilde{L}^+\|_2 \|q_{\tilde{L}x}\|_2 = \sigma^{-1}_{\min}(\tilde{L}) \|q_{\tilde{L}x}\|_2.$$ □

4.3 The Singular Values of $\tilde{L}$

The next step is to show that the singular values of $\tilde{L}$ cannot be much smaller than the smallest eigenvalue of $\hat{L}$. In fact, we show that they are at most a factor of $\sqrt{2}$ smaller.

The proof of Lemma 4.1 shows that the 2-norm of J is at most $\sqrt{2}$. It is easy to show that the norm is, in fact, exactly $\sqrt{2}$ and that all the singular values of J are either 1 or $\sqrt{2}$. The next lemma shows that the $\sqrt{2}$ bound on the norm of J ensures that $\sigma_{\min}(\tilde{L}) \geq \lambda_{\min}(\hat{L})/\sqrt{2}$.

LEMMA 4.10. *Let A, B, and C be matrices such that $AB = C$. Then*

$$\sigma_{\min}(B) \geq \frac{\sigma_{\min}(C)}{\sigma_{\max}(A)}.$$

PROOF. Suppose for contradiction that $\sigma_{\min}(B) = \varepsilon < \sigma_{\min}(C)/\sigma_{\max}(A)$. Then there exist vectors x and y such that $\|x\|_2 = \|y\|_2 = 1$, and $Bx = \varepsilon y$. (x and y are the right and left singular vectors corresponding to $\sigma_{\min}(B)$.) Therefore,

$$\|Cx\|_2 = \|ABx\|_2 = \|A\varepsilon y\|_2 = \varepsilon \|Ay\|_2 \leq \varepsilon \sigma_{\max}(A) \|y\|_2 = \varepsilon \sigma_{\max}(A) < \sigma_{\min}(C),$$

a contradiction. □

We can now prove the main theorem of this section.

THEOREM 4.11.

$$\sigma_{\min}(\tilde{L}) \geq \frac{\lambda_{\min}(\hat{L})}{\sqrt{2}}.$$

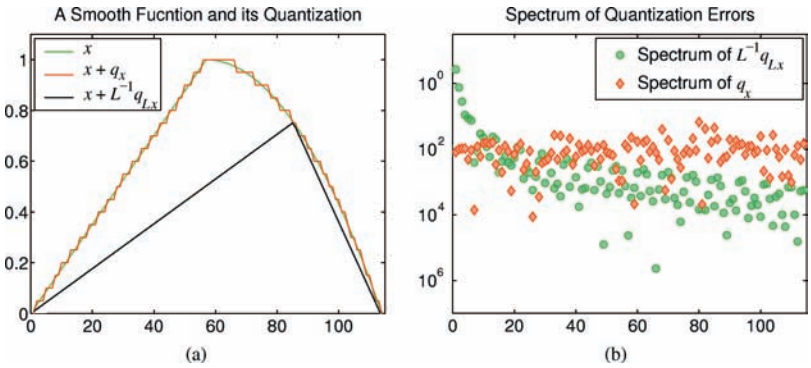

Fig. 2. The same mesh as in Figure 1, but with an additional anchor point at vertex 86. The Laplacian here is a 3-anchor invertible Laplacian with anchors at vertices 1, 86, and 114. The transformed quantization error is no longer smooth at the anchor point even though the vector x is smooth there.

PROOF. Since $J\tilde{L} = \hat{L}$, by the previous lemma

$$\sigma_{\min}(\tilde{L}) \geq \frac{\sigma_{\min}(\hat{L})}{\sigma_{\max}(J)} = \frac{\sigma_{\min}(\hat{L})}{\sqrt{2}} = \frac{\lambda_{\min}(\hat{L})}{\sqrt{2}}.$$ □

4.4 Singular Vectors and the Shape of the Error

Why do we propose to use a rectangular Laplacian rather than a square invertible one? The reason lies in the shape of the quantization error that each method generates. We have already seen that adding anchor points increases the smallest singular value of both the invertible and the rectangular Laplacians. Furthermore, in both cases, the 2-norm of the error $x - x'$ is bounded by $\sqrt{2}\lambda_{\min}^{-1}(\hat{L}) \left\| q_{\tilde{L}x} \right\|_2$, exactly the same bound. (The actual errors will differ and the norms will most likely differ since the bounds are not tight, but the bounds we proved are exactly the same.) We have found, however, that the shape of the error is visually better when we obtain the approximation x' from the rectangular Laplacian. The main difference between the two errors is that the rectangular approximation x' is usually smooth where x is smooth but the invertible approximation is not. The invertible approximation is almost always nonsmooth at the anchors where spikes seem to always appear.

The crucial observation is that the k-anchor invertible Laplacian essentially forces the error $x - x'$ to zero at the anchors and allows the error to grow as we get farther and farther away from the anchor points. When we obtain x' from solving a least-squares problem whose coefficient matrix is $\tilde{L}$, x' can differ from x everywhere, including at the anchor points. This allows x' to be smooth.

Formalizing this explanation is hard and is beyond the scope of this article. The error $x' - x$ consists, in both cases, mainly of the singular vectors of $\tilde{L}$ or $\hat{L}$ that correspond to the smallest singular values. If these singular vectors are smooth, the error $x - x'$ will be smooth, so x' will be smooth where x is smooth. Are these vectors smooth? The numerical example in Figure 2 indicates that the relevant singular/eigen vectors of $\hat{L}$ are not smooth. Our experiments also indicate that the singular vectors of $\tilde{L}$ that correspond to small singular values are smooth.

In this article, we do not attempt to prove these statements about the shape of the singular vectors. In general, the singular vectors of Laplacian and Laplacian-like matrices have not been researched as much as the singular values. It is generally believed that the vectors corresponding to small singular values are indeed smooth. This belief underlies important algorithms such as multigrid [Briggs et al. 2000] and spectral separators [Pothen et al. 1990]. Some additional progress towards an understanding of the relationships between the graph and the eigenvectors of its Laplacian were made recently by

Ben-Chen and Gotsman [2005]. On the other hand, there is also research that indicates that these vectors are not always well-behaved [Guattery and Miller 2000].

We leave the full mathematical analysis of the shape of the errors as an open problem in this article; the empirical evidence shown in Sorkine et al. [2003] supports our claim.

5. THE EFFECT OF ANCHOR POINTS ON NUMERICAL ACCURACY

So far, we have analyzed the norm of the error assuming that x' is the exact solution of $\hat{L}x' = J(\tilde{L}x + q_{\tilde{L}x})$ or the exact minimizer of $\|\tilde{L}x' - (\tilde{L}x + q_{\tilde{L}x})\|_2$. Since we cannot determine x' exactly using floating-point arithmetic, what we actually obtain is an approximation x'' to x'. The total error $x - x''$ depends on both $x - x'$ and $x' - x''$. In this section, we analyze the numerical error $x' - x''$ and show that it too depends primarily on the small singular values of the coefficient matrices $\hat{L}$ and $\tilde{L}$, and hence on the anchor points. The results in this section rely on standard error bounds from numerical linear algebra. For details on these error bounds, see, for example, Higham [2002] or Trefethen and Bau [2000]; the first reference is an encyclopedic monograph, the second a readable textbook.

We assume that the approximation x'' is obtained using a *backward stable* algorithm. For the invertible problem, this means that x'' is the exact solution of $(\hat{L} + \delta\hat{L})x'' = J(\tilde{L}x + q_{\tilde{L}x})$, where $\delta\hat{L}$ is a small perturbation such that $\|\delta\hat{L}\|/\|\hat{L}\| = O(\varepsilon_{\text{machine}})$, where $\varepsilon_{\text{machine}}$ is a small constant depending on the floating-point arithmetic, about 10^{-16} for double-precision IEEE-754 arithmetic, which is now used on virtually all computers. For the rectangular problem, backward stability means that x'' is the exact minimizer of $(\tilde{L} + \delta\tilde{L})x'' - (\tilde{L}x + q_{\tilde{L}x})$ for a similarly small perturbation.

Since $\hat{L}$ is a symmetric positive-definite matrix and since $\tilde{L}$ is full rank, most linear-equation solvers and most least-squares solvers are backward stable when applied to them. This includes sparse direct Cholesky factorization solvers for the square problem, sparse QR solvers for the rectangular least-squares problem, and most iterative algorithms for these problems.

When we obtain an approximation x'' using a backward stable algorithm, the relative norm of the so-called forward error $x' - x''$ is bounded by the *condition number* κ of the problem times $\varepsilon_{\text{machine}}$,

$$\frac{\|x'' - x'\|}{\|x'\|} = O(\kappa\varepsilon_{\text{machine}}). \tag{7}$$

For the invertible problem, the condition number is simply the condition number of the coefficient matrix,

$$\kappa_{\text{inv}} = \|\hat{L}\|\|\hat{L}^{-1}\|. \tag{8}$$

The norm in Equation (8) is the matrix norm induced by the vector norm in Equation (7). When we use the 2-norm in Equations (8) and (7), we have

$$\kappa_{\text{inv}} = \frac{\sigma_{\max}(\hat{L})}{\sigma_{\min}(\hat{L})}.$$

The quantity $\sigma_{\max}(\hat{L})/\sigma_{\min}(\hat{L})$ is called the *spectral condition number* of $\hat{L}$ and is denoted by $\kappa_2(\hat{L})$ or simply κ_2 when the matrix is clear from the context.

The condition number of least-squares problems is a little more complicated. We denote by θ the angle between the right-hand side $(\tilde{L}x + q)$ (here $q = q_{\tilde{L}x}$) and its projection into the column space of $\tilde{L}$. Since $\tilde{L}x$ is in this column space, the size of $\tan\theta$ is roughly proportional to $\|q\|_2/\|\tilde{L}x\|$, which is proportional to how aggressive the quantization is. Therefore, $\tan\theta$ will usually be small. We denote by η the quantity

$$\eta = \frac{\|\tilde{L}\|_2\|x\|_2}{\|\tilde{L}x\|_2}.$$

This quantity is bounded by $1 \leq \eta \leq \kappa_2(\tilde{L})$. In our case, unfortunately, η will not be large because $\tilde{L}x$ contains some values of x, namely the anchors, so its norm will not be much smaller than the norm of x. Given θ and η, we can express the condition number of solving least-squares problems,

$$\kappa_{\text{rect}} = \kappa_2(\tilde{L}) + \frac{\kappa_2(\tilde{L})^2 \tan\theta}{\eta}.$$

In our case, $\kappa_2(\tilde{L})$ and $\kappa_2(\hat{L})$ depend only on the small eigenvalue of $\hat{L}$, which we have already shown to be strongly influenced by the anchor points. Since $\sigma_{\max}(\hat{L}) = \lambda_{\max}(\hat{L}) \leq 2d_{\max} + 1$, where $d_{\max}$ is the maximal degree of a vertex in the mesh, and since $\sigma_{\max}(\tilde{L}) \leq \sqrt{2}\lambda_{\max}(\hat{L})$, in both cases, the largest singular value is bounded by a small constant so $\kappa(L) = O(\lambda_{\min}^{-1}(\hat{L}))$ for both L's.

THEOREM 5.1. *Let $\lambda = \lambda_{\min}(\hat{L})$, $\varepsilon = \varepsilon_{\text{machine}}$, and $q = q_{\tilde{L}x}$. The 2-norm of the error $x - x''$, when x'' is computed from the invertible Laplacian using a backward-stable algorithm, is bounded by*

$$\|x - x''\|_2 \leq O(\lambda^{-1}\|q\|_2 + \lambda^{-1}\varepsilon\|x\|_2 + \lambda^{-2}\varepsilon\|q\|_2).$$

PROOF. By Lemma 4.9, $\|x'\|_2 = \|x' + x - x\|_2 \leq \|x - x'\|_2 + \|x\|_2 \leq \lambda^{-1}\|q\|_2 + \|x\|_2$. The inequality and the discussion preceding the theorem yield

$$\begin{aligned}
\|x - x''\|_2 &= \|x - x' + x' - x''\|_2 \\
&\leq \|x - x'\| + \|x' - x''\|_2 \\
&\leq \lambda^{-1}\|q\|_2 + \|x' - x''\|_2 \quad \text{by Lemma 4.9} \\
&\leq \lambda^{-1}\|q\|_2 + O(\kappa_2(\hat{L})\varepsilon\|x'\|_2) \\
&= \lambda^{-1}\|q\|_2 + O(\lambda^{-1}\varepsilon(\lambda^{-1}\|q\|_2 + \|x\|_2)) \\
&= \lambda^{-1}\|q\|_2 + O(\lambda^{-1}\varepsilon\|x\|_2 + \lambda^{-2}\varepsilon\|q\|_2).
\end{aligned}$$

□

We now state the corresponding theorem for the least-squares case. The proof, which we omit, is identical except for the expression of the condition number.

THEOREM 5.2. *Let $\lambda = \lambda_{\min}(\hat{L})$, $\varepsilon = \varepsilon_{\text{machine}}$, and $q = q_{\tilde{L}x}$. The 2-norm of the error $x - x''$, when x'' is computed from the rectangular least-squares problem using a backward-stable algorithm, is bounded by*

$$\|x - x''\|_2 \leq O\left(\lambda^{-1}\|q\|_2 + \lambda^{-1}\varepsilon\|x\|_2 + \frac{\lambda^{-2}\tan\theta\varepsilon\|x\|_2}{\eta} + \lambda^{-2}\varepsilon\|q\|_2 + \frac{\lambda^{-3}\tan\theta\varepsilon\|q\|_2}{\eta}\right).$$

One way of solving the least-squares problem is by constructing and solving the so-called *normal equations*. This solution method relies on the fact that the least-squares minimizer x' is also the solution of the symmetric positive-definite linear system $\tilde{L}^T\tilde{L}x' = \tilde{L}^T(\tilde{L}x + q_{\tilde{L}x})$. Even when the normal equations are solved using a backward-stable algorithm, the whole algorithm is not backward stable with respect to the original least-squares problem. The computed solution satisfies only

$$\frac{\|x'' - x'\|_2}{\|x'\|_2} = O\big(\kappa_2(\hat{L})^2\varepsilon_{\text{machine}}\big).$$

Because the error bound is much larger in this case (and usually much larger in practice), this method is usually not recommended. However, since in our application, we can control and estimate $\kappa_2(\hat{L})$ by adding anchor points, we can ensure that even the normal-equations forward error is acceptable.

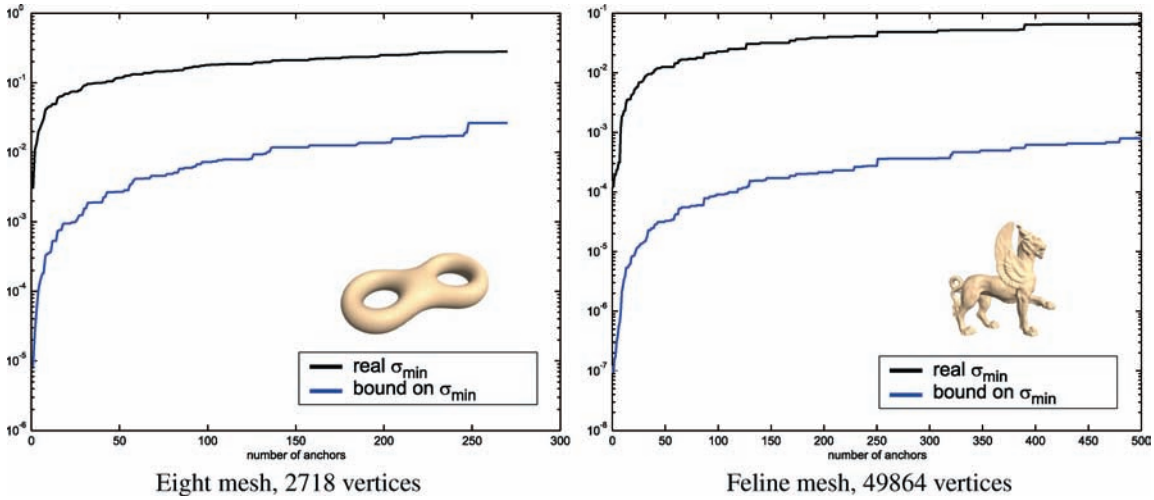

Fig. 3. Comparison of the congestion-dilation-contention bound on $\sigma_{\min}$ (see Theorem 4.11) with the actual value of $\sigma_{\min}$. The x-axis shows the number of anchors used.

6. ALGORITHMIC ISSUES AND RESULTS

Two algorithmic problems arise in the high-pass quantization method: the anchor-selection problem and the linear least-squares problem. This section explains how these issues can be addressed and shows some experimental results.

6.1 Evaluating the Bound on $\sigma_{\min}$

In order to exploit the theoretical results presented in the previous section, we need an algorithm to evaluate the lower bound on $\sigma_{\min}$, the smallest singular value of $\tilde{L}$. Given a set of anchor vertices $\{a_1, a_2, \ldots, a_k\}$, we are looking for some partition of all the mesh vertices into k subsets, such that we can define the values φ, ϑ, ρ (congestion, dilation, and contention) reasonably. Since finding a partition that strictly maximizes the bound in Theorem 4.11 does not seem feasible, we use the following heuristic. We simultaneously grow patches of vertices around the anchors by running k-source BFS. This algorithm produces a rather balanced partition that keeps the values of ϑ and ρ small. After the partition has been computed, the calculation of ϑ and ρ is straightforward. To compute φ, we use the parent pointers stored for each vertex during the BFS procedure. These pointers define the tree of paths from each vertex to the root (source anchor vertex). Clearly, the most loaded edges are the edges whose source vertex is the root. By counting the number of vertices in the subtrees hanging on those edges, we obtain their edge loads, and compute the maximum over all the k subsets.

We have compared the evaluation of the lower bound of $\sigma_{\min}$ with the real value of $\sigma_{\min}$ on moderately-sized meshes. The accurate value of $\sigma_{\min}$ was computed in MATLAB. Figure 3 shows two representative graphs summarizing this experiment. The horizontal axis in the graphs represents the number of anchor rows present in $\tilde{L}$. We incrementally added random anchor vertices and plotted the value of $\sigma_{\min}$ and the lower bound. As can be seen from these graphs, the bound differs from the real value by 1.5–2.5 orders of magnitude and behaves consistently with the real $\sigma_{\min}$. We can thus conclude that our theoretical bound is not too pessimistic and can be used in practical algorithms for choosing the anchors, as discussed in the following.

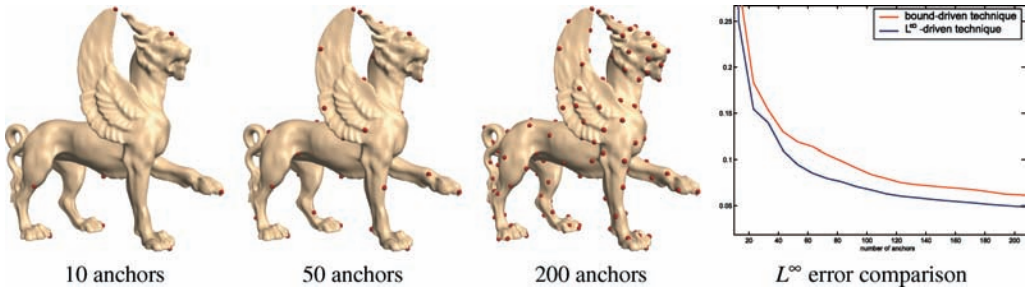

Fig. 4. Anchors on the Feline model chosen by the bound-driven algorithm. The first three images display stages of the incremental anchor-selection procedure. The even spacing of the anchors increases the lower bound on $\sigma_{\min}$. The graph compares the L^∞ reconstruction error for anchors chosen with the greedy scheme as in Sorkine et al. [2003] (in red) and the bound-driven scheme (in blue), when using the same level of δ-coordinates quantization (6 bits/coordinate).

6.2 Algorithms for Placing Anchor Points

Sorkine et al. [2003] use the following adaptive and greedy algorithm to select anchor points. They begin by placing one random anchor point and generating a 1-anchor rectangular Laplacian, denoted by $\tilde{L}_1$. They then use this matrix to transform the coordinates, quantize the δ-coordinates, compute an approximation x_1'', and compute the error $x - x_1''$. The second anchor is placed at the vertex with the largest error to yield $\tilde{L}_2$. These iterations continue until either a satisfactory error is attained, or until a given number k of anchors is placed.

The advantage of this scheme is that it directly attempts to minimize the reconstruction error rather than its bound. However, the first iterations of the greedy algorithm may compute ineffective anchors since, in the beginning, only a few anchors are used, and the matrix $\tilde{L}$ is thus ill-conditioned. Therefore, the first reconstructed vectors x_i'' will contain very high errors.

The congestion-dilation-contention bounds that we present in this article suggest another anchor-selection scheme, one that aims to maximize the lower bound on $\sigma_{\min}$. This scheme can be used to select enough effective anchors to ensure reasonable conditioning of $\tilde{L}$ and more anchors can then be added using the previous greedy algorithm.

As mentioned previously, it is hard to strictly maximize the bound on $\sigma_{\min}$. To choose anchors so as to make the bound expression larger, we again propose a heuristic method. It selects the anchors one by one, while minimizing the value of ϑ. The method operates as follows. We start with one randomly chosen anchor and compute its edge-distance from all the other vertices in the mesh by running BFS. The furthest vertex is chosen as the next anchor, and we proceed in the same manner. In the i-th iteration, we have a set of i anchor vertices; we run i-source BFS from these vertices to find the vertex that achieves the longest edge distance from an anchor (the value of ϑ). This vertex is assigned as the $(i + 1)$-th anchor. The procedure stops when we reach a large enough value of the bound or after a prescribed number of steps. It should be noted that actually there is no need to run the complete i-source BFS in every step. It is enough to run (partial) BFS from the last chosen anchor in order to update the distances. The front propagation of the BFS procedure stops whenever we meet a vertex whose old distance value is smaller than the distance that would be assigned by the current BFS.

Figure 4 shows some steps of the anchor-selection algorithm on the Feline model. The anchors are well-spaced which is favorable for the congestion-dilation-contention bound. The graph in Figure 4 plots the reconstruction of the max-norm as a function of the number of anchors. The red line denotes the values for anchors chosen with the bound-maximizing scheme, while the blue line represents the greedy scheme used in Sorkine et al. [2003]. As expected, the greedy scheme produces somewhat smaller errors since it operates directly to minimize the max-norm error. However, on larger meshes, the error-bound

minimization scheme gives an initial set of anchors to make $\tilde{L}$ well-conditioned and thus provides a good starting point for the greedy algorithm. Moreover, this scheme is much faster since it does not require reconstruction of the Cartesian coordinates at each iteration.

After the anchor placement strategy has been fixed, the shape of the mesh reconstructed with the high-pass quantization technique depends mainly on two factors: the level of δ-coordinates quantization and the amount of anchors. The Cartesian coordinates of the anchors should be mildly quantized to preserve accuracy. Adding anchors to the representation is cheap: if B is the number of bits per coordinate (typically, between 12–14), then a single anchor requires $3B + \log(n)$ bits ($\log(n)$ bits for the index of the anchor vertex). As suggested by the theoretical bounds in Theorems 4.3 and 4.11, we can keep the condition number of the system (and hence the L^2 error) constant by ensuring that the dilation, congestion, and contention are bounded by a constant. When the number of anchors is a constant fraction p of the number of mesh vertices n, the dilation, congestion, and contention are usually bounded by a constant or grow very slowly with n. As discussed in Sorkine et al. [2003], for visually acceptable value of L^2 error, p is rather small, up to 1%. This is due to the fact that the visual quality is more affected by the change of high-frequency details (e.g., surface normals or the surface local smoothness properties) rather than global low-frequency errors. Adding anchors to a fixed δ-quantization only helps to make the low-frequency error smaller, but hardly effects the high-frequency error. On the other hand, adding more bit planes to the δ-coordinates significantly reduces the high-frequency error as well as the low-frequency error (see the $\|q_{Lx}\|_2$ component of the L^2 error bound in Lemma 4.9). However, this is more expensive since adding a single bit per δ-coordinate requires the addition of n bits to the representation (prior to entropy-coding).

The visual tables in Figures 5 and 6 demonstrate the effect of adding anchors versus adding bits to the δ-coordinates. Each row in the tables displays reconstructed models with a varying number of anchors for a fixed δ-quantization level. As shown in the figures, the surface smoothness properties vary in different rows but not columns, while the surface general pose (affected by low-frequency error) decreases both in the rows (top to bottom as more bits are added to the δ-coordinates) and in the columns (left to right as more anchors are added). This is also supported numerically by the values of the S_q and M_q errors (see Sorkine et al. [2003]). In all the experiments, fixed quantization of 12 bits/coordinate was applied to the positions of the anchor vertices.

It is important to note that state-of-the-art geometry encoding methods, such as the wavelet compression [Khodakovsky et al. 2000], employ zerotree encoding with a clever bit-allocation scheme that adapts to the local surface shape. We believe that adaptive encoding will benefit our geometry encoding scheme as well; currently, we uniformly quantize the δ-coordinates of the entire mesh and encode them with a standard arithmetic encoder that does not fully exploit the specific nature of the data. This rather naive compression is obviously not optimal as supported by the statistics in Table I, where we compare the file sizes of the models compressed by our method with those of Khodakovsky et al. [2000]. However, in contrast to Guskov et al. [2000] and Khodakovsky et al. [2000] and others, our method does not require any remeshing. It would be appropriate to compare our method with the spectral compression of Karni and Gotsman [2000] since the latter method also preserves the original mesh connectivity but currently, it is infeasible to apply this method to meshes with more than a few thousand vertices because it requires computing the eigenvectors of the mesh Laplacian matrix on both the encoder and the decoder side.

6.3 Solving Least-Squares Problems

Decompressing a mesh function in the high-pass quantization method requires solving a linear least-squares problem. Sorkine et al. [2003] discussed this important algorithmic issue only briefly. To allow the reader a broader perspective on this issue, we survey here state-of-the-art least-square solvers.

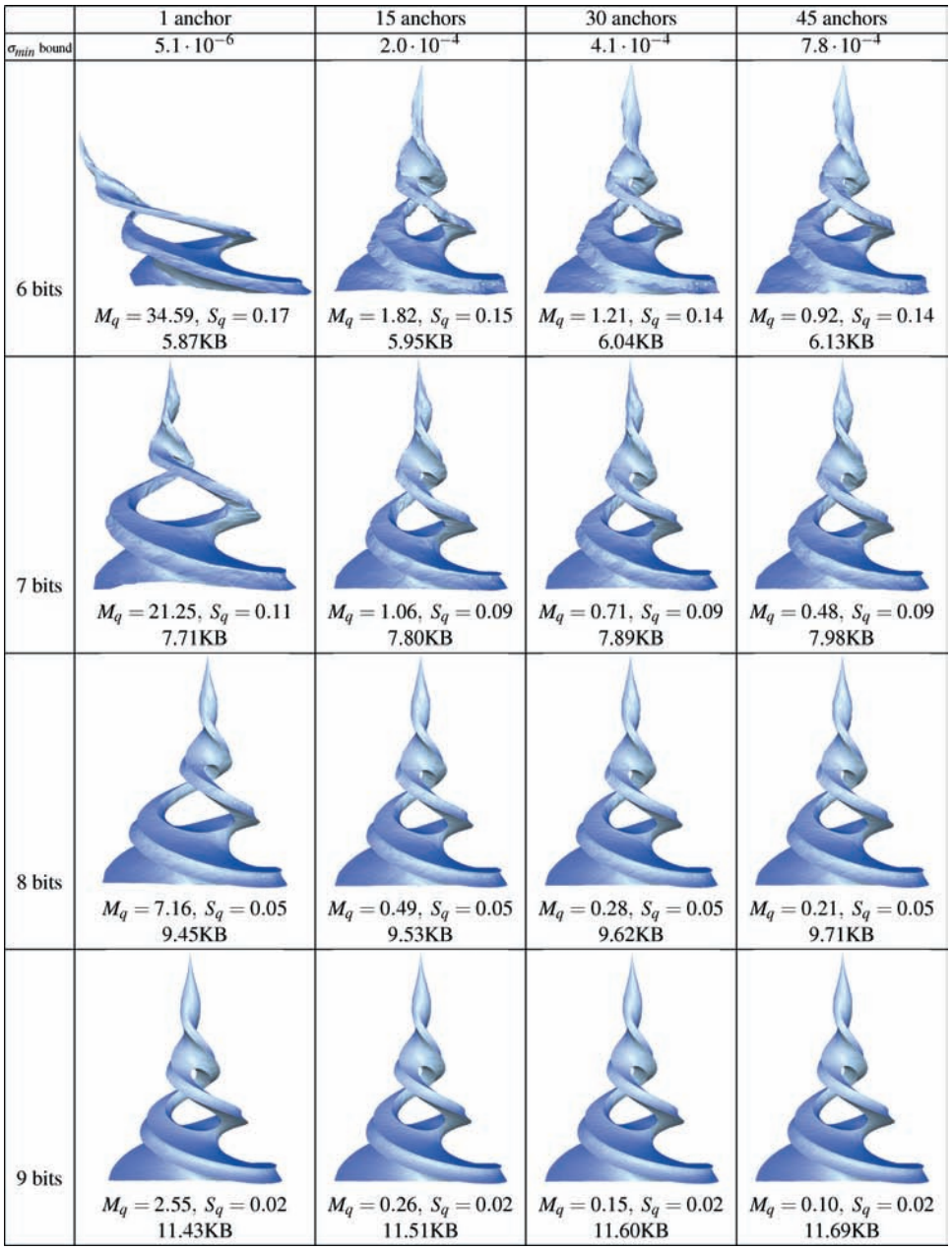

Fig. 5. Visual table of quantization results for the Twirl model (5201 vertices). The vertical axis corresponds to the number of bits per coordinate used in δ-quantization. The horizontal axis corresponds to the number of anchor points used. M_q and S_q denote the Euclidean RMS error and the smoothness error, respectively (see Sorkine et al. [2003]). The file sizes given below each reconstruction were obtained by arithmetic encoding of the quantized δ-coordinates and the anchors.

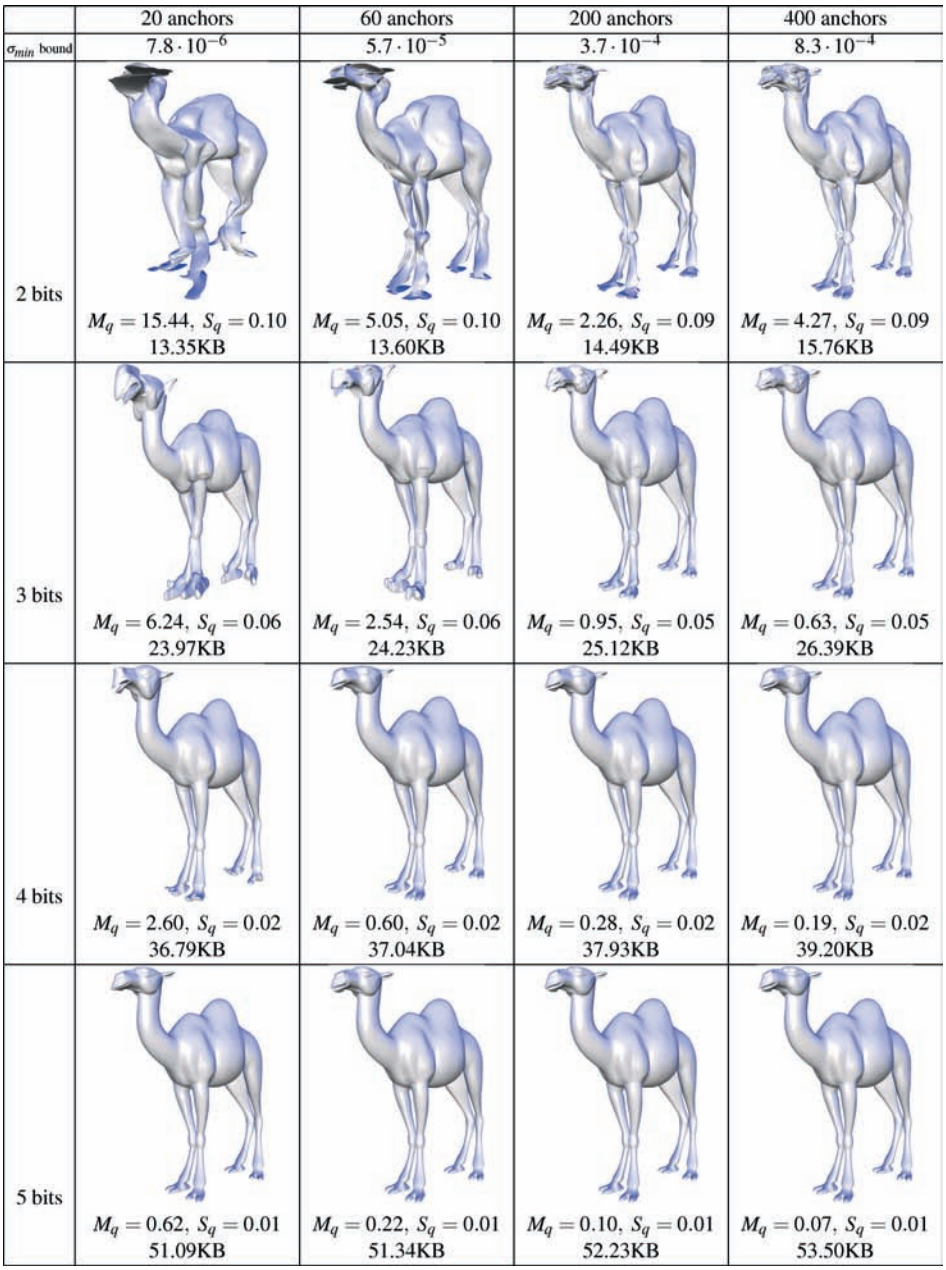

Fig. 6. Visual table of quantization results for the Camel model (39074 vertices). The vertical axis corresponds to the number of bits per coordinate used in δ-quantization. The horizontal axis corresponds to the number of anchor points used. M_q and S_q denote the Euclidean RMS error and the smoothness error, respectively (see Sorkine et al. [2003]). The file sizes given below each reconstruction were obtained by arithmetic encoding of the quantized δ-coordinates and the anchors.

Table I. Comparison Between our Geometry Encoding and the Wavelet Encoder of Khodakovsky et al. [2000] (The file sizes are displayed in percents-relative to the uncompressed mesh geometry. The models were compressed by both methods with approximately the same visual error on the order of 10^{-4} so that the compressed mesh is indistinguishable from the original.)

Model	Number of Vertices	Relative Wavelet Filesize (%)	Relative Highpass Filesize (%)
Rabbit	107,522	0.354	0.895
Bunny	118,206	0.429	0.662
Horse	112,642	0.321	0.457
Venus	198,658	0.336	0.543
Feline	258,046	0.389	0.790

We briefly mention some key algorithms, provide some sample performance data, and explain how the quantization and compression methods can be tailored to ensure fast decompression. For a more complete discussion of algorithms for sparse linear least-squares problems, see Björck's monograph [1996].

Sparse least-squares solvers fall into two categories, direct and iterative. Most direct solvers factor the coefficient matrix $\tilde{L}$ into a product of an orthonormal matrix Q and an upper triangular matrix R, $\tilde{L} = QR$. Once the factorization is computed, the minimizer $\hat{x}$ of $\|\tilde{L}x - b\|_2$ is found by solving the triangular linear system of equations $R\hat{x} = Q^T b$. This algorithm is backward stable. The matrix R is typically very sparse, although not as sparse as $\tilde{L}$; it is represented explicitly in such algorithms. In particular, since, in our case, the meshes are almost planar graphs and have small vertex separators, R is guaranteed to remain sparse [George and Ng 1988]. The matrix Q is not as sparse, but it has a sparse representation as a product of elementary orthogonal factors [George and Heath 1980; George and Ng 1986]. To reduce the work and storage required for the factorization, the columns of the input matrix $\tilde{L}$ are usually reordered prior to the factorization [George and Ng 1983; Heggernes and Matstoms 1996; Brainman and Toledo 2002; Davis et al. 2004].

Another class of direct solvers, which is normally considered numerically unstable, uses a triangular factorization of the coefficient matrix $\tilde{L}^T\tilde{L}$ of the so-called normal equations. Once triangular factor R is found (it is mathematically the same R as in the $\tilde{L} = QR$ factorization), the minimizer is found by solving two triangular linear systems of equations, $R^T(R\hat{x}) = \tilde{L}^T b$. This procedure is faster than the QR procedure but produces less accurate solutions because solving the normal equations is not backward stable. However, the accuracy of the solutions depends on the condition number of $\tilde{L}$ (ratio of extreme singular values), and, as we have shown in Section 5 the matrix $\tilde{L}$ is well-conditioned thanks to the anchors so, in this case, solving the normal-equations problem yields accurate solutions.

The running times and storage requirements of direct solvers can be further reduced by cutting the mesh into patches as proposed by Karni and Gotsman [2000] and solving on each patch separately. All the boundary vertices are then considered anchors to ensure that the solutions on different patches are consistent. We believe that this optimization would usually be unnecessary and that problems involving entire meshes can be solved efficiently, but we mention it as a way of handling extremely large cases. Note that to ensure that the patches are consistent, the k-anchor invertible Laplacian would need to be used here, not the k-anchor rectangular Laplacian.

In all direct methods, the factorization is computed once and used to solve for multiple mesh functions. Most of the time is spent in computing the factorization, and the cost of solving for a minimizer is negligible. Therefore, the cost of decompression using these methods is almost independent of the number of mesh functions (x, y, z, and perhaps other information such as color).

Direct methods are fast. Table II records the solution times for the models used in our experiments. The table shows the time to decompose the coefficient matrix of the normal equations into its triangular

Table II. Running Times of Solving the Linear Least-Squares Systems for the Different Models (Most time is spent on the factorization of the coefficient matrix which can be done during the transmission of the δ-coordinates. Solving for a single mesh function (x, y or z) takes only a negligible amount of time (see rightmost column). The experimental setup is described in the text.)

Model	Number of Vertices	Factorization (sec.)	Solving (sec.)
Eight	2,718	0.085	0.004
Twirl	5,201	0.098	0.006
Horse	19,851	0.900	0.032
Fandisk	20,111	1.091	0.040
Camel	39,074	2.096	0.073
Venus	50,002	3.402	0.112
Max Planck	100,086	7.713	0.240

factors and the subsequent solution time for one mesh function. For example, computing the triangular factorization of the horse, a model with 19,851 vertices, took 0.9 seconds on a 2.4GHz Pentium 4 computer, and solving for a single mesh function took 0.032 seconds once the factorization has been computed. The linear solver that we used for these experiments is TAUCS version 2.2 [Toledo 2003] which uses internally two additional libraries, ATLAS version 3.4.1 [Whaley et al. 2000] and METIS version 4.0 [Karypis and Kumar 1998]. TAUCS and METIS were compiled using the Intel C/C++ compiler version 7.1 for Linux, and ATLAS was compiled using GCC version 2.95.2. The options to the compilers included optimization options (`-O3`) and Pentium 4-specific instructions (`-xW` for the Intel compiler and inlined assembly language in ATLAS). For additional performance evaluations of TAUCS, see Rotkin and Toledo [2004] and Irony et al. [2004]. We did not have a code of similar performance for computing the sparse QR factorization, but we estimate that it should be about 4–6 times slower.

Even though direct methods are fast, their running times usually scale superlinearly with the size of the mesh. Iterative least-squares solvers, which do not factor the coefficient matrix, sometimes scale better than direct methods. Perhaps the most widely-used least-squares iterative solver is LSQR which is based on a Krylov bidiagonalization procedure [Paige and Saunders 1982a, 1982b]. Other popular solvers include CGLS, a conjugate-gradients algorithm for solving the normal equations [Elfving 1978; Björck and Elfving 1979], and CRAIG, an error-minimization bidiagonalization procedure [Craig 1955]; see also Paige and Saunders [1982b] and Saunders [1995].

The convergence of these methods depends on the distribution of the singular values of the coefficient matrix $\tilde{L}$ as well as on the initial approximation. In our case, $\tilde{L}$ is always well-conditioned so we can expect reasonably rapid convergence. Furthermore, the decoder knows the values of the mesh function at the anchor vertices. By interpolating these values at nonanchor vertices, the decoder can quickly produce a good initial approximation (note, however, that, even at the anchor points, the known values of the original mesh function need not coincide with the values of the least-squares minimizer).

The iterative methods mentioned can be accelerated by using a preconditioner, (informally, an approximate inverse of $\tilde{L}$). The relationship of our coefficient matrix $\tilde{L}$ to a graph Laplacian can probably be exploited when constructing a preconditioner since highly effective preconditioners have been discovered for Laplacians. The most important classes of such preconditioners are algebraic multigrid preconditioners [Brandt et al. 1984], incomplete Cholesky preconditioners [Meijerink and van der Vorst 1977; Gustafsson 1978], and more recently, support preconditioners [Vaidya 1991; Boman and Hendrickson 2003; Chen and Toledo 2003]. For further information about iterative solvers and preconditioning, see Barret et al. [1993], Axelsson [1994], Björck [1996], and Saad [1996].

7. CONCLUSIONS

In this article, we have shown that it is possible to rigorously bound the error in a lossy compression method for three-dimensional meshes. The article focuses on one particular compression method, that presented by Sorkine et al. [2003], but our analysis technique is probably applicable to a range of methods using similar matrices. In particular, our analysis also sheds light on the errors in a more recent compression method Sorkine et al. [2005] in which the encoder does not send the δ-coordinates to the decoder at all, only the anchor vertices. It is also useful for analyzing Laplacian-based mesh editing techniques [Lipman et al. 2004].

Our analysis bounds the total error generated by high-pass quantization, both the quantization component of the error and the rounding component. In other words, it accounts for the fact that the decoder uses floating-point arithmetic to reconstruct the mesh. On the other hand, our analysis does not cover the shape of the errors. Empirical results show that the error is smooth and, therefore, visually acceptable; these results are consistent with other applications of the small eigenvectors of Laplacians.

Our analysis yields an error bound that is easy to compute as we have shown in Section 6.1. This leads to two algorithmic benefits in addition to the insight on why high-pass quantization works. First, it can be used by an encoder to quickly encode a mesh to a prescribed error bound. That is, the encoder can quantize the coordinates and then add anchors until the computed error bound drops below a prescribed threshold. Since our error bound is not tight, the actual error will usually be smaller than that prescribed. The encoding might not be as economical as possible but it will be produced quickly and it will satisfy the prescribed error bound. Second, the computed bound can drive the anchor selection algorithm, as shown in Section 6.1.

We have used three algebraic techniques to prove our error bound. Two of them are quite novel. Our bound on the small eigenvalue of an invertible Laplacian is a relatively standard application of an area of combinatorial matrix theory called support theory, but the technique of separating of W into W_1 and W_2 is new. The main algebraic novelty in the article lies in the application of support theory to the analysis of the spectrum of rectangular matrices. The analysis of the rounding error is relatively straightforward.

Our research raises a number of interesting open problems for future research.

(1) Can one rigorously analyze the behavior of the eigenvectors of the Laplacian of 3D meshes? Our method works because, for a vector x of mesh coordinates, $\|Lx\|$ tends to be much smaller than $\|x\|$. This happens because most of the energy of x is concentrated in the subspace of R^n that is spanned by the eigenvectors of L that correspond to small eigenvalues. But does this always happen? The answer depends on the relationship between the eigenvectors of the Laplacian and typical mesh-coordinate vectors. Ben-Chen and Gotsman [2005] have taken the first step towards resolving this question. They have shown that, under certain probabilistic assumptions on the shape of 3D meshes, most of the energy of the mesh-coordinate vectors indeed lies in the sub-spaces spanned by the small eigenvectors. Another analysis, done by Guattery and Miller [2000] in a different context, may provide another perspective on the issue.

(2) Our bound on the small singular values of k-anchor Laplacian uses a maximal congestion-dilation-contention metric on an embedding of paths from all the vertices to the anchors. It is probably possible to derive other computable bounds that might sometimes be tighter such as a bound that depends on average dilation of this embedding.

(3) Can one solve the least-squares problems that arise in our method in time linear or almost linear in the size of the mesh? We have demonstrated reasonably small running times even for large meshes, but our solution method scales superlinearly. It would be useful to find solution methods with better scaling. Algebraic multigrid methods can almost certainly solve the invertible k-anchor Laplacian

equations in $O(n)$ work. We are not yet sure whether algebraic multigrid methods can also effectively solve the least-squares problem arising from the rectangular Laplacian. Another direction might be an iterative solver, such as LSQR or CGLS, coupled with an effective preconditioner. In particular, it would be interesting to know whether graph-theoretical pre-conditioners, such as support-tree [Gremban et al. 1995; Gremban 1996] and support-graph [Vaidya 1991; Bern et al. 2001; Spielman and Teng 2003; Boman et al. 2004] preconditioners can be adapted to this problem.

ACKNOWLEDGMENTS

We would like to thank Ram Cohen for the arithmetic encoder, and Andrei Khodakovsky for the wavelet compression code. The Twirl model is courtesy of Alexander Belyaev.

REFERENCES

ALLIEZ, P. AND DESBRUN, M. 2001. Valence-driven connectivity encoding for 3D meshes. *Comput. Graph. For. 20*, 3, 480–489.

ALLIEZ, P. AND GOTSMAN, C. 2005. Recent advances in compression of 3D meshes. In *Advances in Multiresolution for Geometric Modelling*, N. Dodgson, M. Floater, and M. Sabin, Eds. Springer-Verlag, 3–26.

AXELSSON, O. 1994. *Iterative Solution Methods*. Cambridge University Press, Cambridge, UK.

BARRET, R., BERRY, M., CHAN, T., DEMMEL, J., DONATO, J., DONGARRA, J., EIJKHOUT, V., POZO, R., ROMINE, C., AND VAN DER VORST, H. 1993. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM, Philadeplphia, PA.

BEN-CHEN, M. AND GOTSMAN, C. 2005. On the optimality of spectral compression of mesh data. *ACM Trans. Graph. 24*, 1, 60–80.

BERN, M., GILBERT, J. R., HENDRICKSON, B., NGUYEN, N., AND TOLEDO, S. 2001. Support-graph preconditioners. *SIAM J. Matrix Anal. Appl.* Submitted for publication.

BJÖRCK, Å. 1996. *Numerical Methods for Least Squares Problems*. SIAM, Philadelphia, PA.

BJÖRCK, Å. AND ELFVING, T. 1979. Accelerated projection methods for computing pseudoinverse solutions of systems of linear equations. *BIT 19*, 145–163.

BOMAN, E. G., CHEN, D., HENDRICKSON, B., AND TOLEDO, S. 2004. Maximum-weight-basis preconditioners. *Numer. Linear Algeb. Appl. 11*, 695–721.

BOMAN, E. G. AND HENDRICKSON, B. 2003. Support theory for preconditioning. *SIAM J. Matrix Anal. Appl. 25*, 3, 694–717.

BRAINMAN, I. AND TOLEDO, S. 2002. Nested-dissection orderings for sparse LU with partial pivoting. *SIAM J. Matrix Anal. Appl. 23*, 998–112.

BRANDT, A., MCCORMICK, S. F., AND RUGE, J. 1984. Algebraic multigrid (AMG) for sparse matrix equations. In *Sparsity and its Applications*, D. J. Evans, Ed. Cambridge University Press, Cambridge, UK. 257–284.

BRIGGS, W. L., HENSON, V. E., AND MCCORMICK, S. F. 2000. *A Multigrid Tutorial*, 2nd Ed. SIAM, Philadelphia, PA.

CHEN, D. AND TOLEDO, S. 2003. Vaidya's preconditioners: Implementation and experimental study. *Electr. Trans. Numer. Anal. 16*, 30–49.

CHOU, P. H. AND MENG, T. H. 2002. Vertex data compression through vector quantization. *IEEE Trans. Visual. Comput. Graph. 8*, 4, 373–382.

CHUNG, F. R. K. 1997. *Spectral Graph Theory*. American Methematical Society.

CRAIG, E. J. 1955. The n-step iteration procedure. *J. Math. Phys. 34*, 65–73.

DAVIS, T. A., GILBERT, J. R., LARIMORE, S. I., AND NG, E. G. 2004. A column approximate minimum degree ordering algorithm. *ACM Trans. Math. Softw. 30*, 3, 353–376.

ELFVING, T. 1978. On the conjugate gradient method for solving linear least squares problems. Tech. rep. LiTH-MAT-R-78-3, Linköping University, Sweden.

FIEDLER, M. 1973. Algebraic connectivity of graphs. *Czech. Math. J. 23*, 298–305.

GEORGE, J. A. AND HEATH, M. T. 1980. Solution of sparse linear least squares problems using Givens rotations. *Linear Algeb. Appl. 34*, 69–83.

GEORGE, J. A. AND NG, E. G. 1983. On row and column orderings for sparse least squares problems. *SIAM J. Numer. Anal. 20*, 326–344.

GEORGE, J. A. AND NG, E. G. 1986. Orthogonal reduction of sparse matrices to upper triangular form using Householder transformations. *SIAM J. Sci. Statist. Comput. 7*, 460–472.

GEORGE, J. A. AND NG, E. G. 1988. On the complexity of sparse QR and LU factorization of finite-element matrices. *SIAM J. Sci. Statist. Comput. 9*, 849–861.

GOLUB, G. H. AND LOAN, C. F. V. 1996. *Matrix Computations*, 3rd Ed. Johns Hopkins University Press.

GREMBAN, K., MILLER, G., AND ZAGHA, M. 1995. Performance evaluation of a parallel preconditioner. In *IEEE 9th International Parallel Processing Symposium*. Santa Barbara, CA. 65–69.

GREMBAN, K. D. 1996. Combinatorial preconditioners for sparse, symmetric, diagonally dominant linear systems. Ph.D. thesis, School of Computer Science, Carnegie Mellon University.

GUATTERY, S. AND MILLER, G. L. 2000. Graph embeddings and laplacian eigenvalues. *SIAM J. Matrix Anal. Appl. 21*, 3, 703–723.

GUMHOLD, S. 2000. New bounds on the encoding of planar triangulations. Technical rep. WSI–2000–1, (Jan.) Wilhelm-Schickard-Institut für Informatik, University of Tübingen, Germany.

GUSKOV, I., VIDIMČE, K., SWELDENS, W., AND SCHRÖDER, P. 2000. Normal meshes. In *Proceedings of ACM SIGGRAPH*. 95–102.

GUSTAFSSON, I. 1978. A class of first-order factorization methods. *BIT 18*, 142–156.

HEGGERNES, P. AND MATSTOMS, P. 1996. Finding good column orderings for sparse QR factorizations. Tech. rep. LiTH-MAT-1996-20, Department of Mathematics, Linköping University, Sweden.

HIGHAM, N. J. 2002. *Accuracy and Stability of Numerical Algorithms*, 2nd ed. SIAM, Philadelphia, PA.

IRONY, D., SHKLARSKI, G., AND TOLEDO, S. 2004. Parallel and fully recursive multifrontal supernodal sparse Cholesky. *Fut. Generat. Comput. Syst. 20*, 3, 425–440.

KARNI, Z. AND GOTSMAN, C. 2000. Spectral compression of mesh geometry. In *Proceedings of ACM SIGGRAPH*. 279–286.

KARYPIS, G. AND KUMAR, V. 1998. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput. 20*, 359–392.

KHODAKOVSKY, A., ALLIEZ, P., DESBRUN, M., AND SCHRÖDER, P. 2002. Near-optimal connectivity encoding of 2-manifold polygon meshes. *Graph. Models 64*, 3/4, 147–168.

KHODAKOVSKY, A., SCHRÖDER, P., AND SWELDENS, W. 2000. Progressive geometry compression. In *Proceedings of ACM SIGGRAPH*. 271–278.

LIPMAN, Y., SORKINE, O., COHEN-OR, D., LEVIN, D., RÖSSL, C., AND SEIDEL, H.-P. 2004. Differential coordinates for interactive mesh editing. In *Proceedings of Shape Modeling International*. 181–190.

MEIJERINK, J. A. AND VAN DER VORST H. A. 1977. An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix. *Mathe. Computat. 31*, 148–162.

PAIGE, C. C. AND SAUNDERS, M. A. 1982a. Algorithm 583 LSQR: Sparse linear equations and sparse least squares. *ACM Trans. Math. Softw. 8*, 195–209.

PAIGE, C. C. AND SAUNDERS, M. A. 1982b. LSQR. An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Softw. 8*, 43–71.

POTHEN, A., SIMON, H. D., AND LIOU, K.-P. 1990. Partitioning sparse matrices with eigenvectors of graphs. *SIAM J. Matrix Anal. 11*, 430–452.

ROTKIN, V. AND TOLEDO, S. 2004. The design and implementation of a new out-of-core sparse Cholesky factorization method. *ACM Trans. Math. Softw. 30*, 1, 19–46.

SAAD, Y. 1996. *Iterative Methods for Sparse Linear Systems*. PWS Publishing Company.

SAUNDERS, M. A. 1995. Solution of sparse rectangular systems using LSQR and CRAIG. *BIT 35*, 588–604.

SORKINE, O., COHEN-OR, D., IRONY, D., AND TOLEDO, S. 2005. Geometry-aware bases for shape approximation. *IEEE Trans. Visual. Comput. Graph. 11*, 2, 171–180.

SORKINE, O., COHEN-OR, D., AND TOLEDO, S. 2003. High-pass quantization for mesh encoding. In *Proceedings of the Eurographics / ACM SIGGRAPH Symposium on Geometry Processing*. 42–51.

SPIELMAN, D. AND TENG, S.-H. 2003. Solving sparse, symmetric, diagonally-dominant linear systems in time $o(m^{1.31})$. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*.

TOLEDO, S. 2003. TAUCS*: A Library of Sparse Linear Solvers, version 2.2*. Tel-Aviv University, Available at `http://www.tau.ac.il/~stoledo/taucs/`.

TOUMA, C. AND GOTSMAN, C. 1998. Triangle mesh compression. In *Graphics Interface*. 26–34.

TREFETHEN, L. N. AND BAU, III, D. 2000. *Numerical Linear Algebra*. SIAM, Philadelphia, PA.

VAIDYA, P. M. 1991. Solving linear equations with symmetric diagonally dominant matrices by constructing good preconditioners. Unpublished manuscript. Manuscript was presented at the IMA Workshop on Graph Theory and Sparse Matrix Computation, Oct., Minneapolis, MN.

WHALEY, R. C., PETITET, A., AND DONGARRA, J. J. 2000. Automated empirical optimization of software and the ATLAS project. Tech. rep., Computer Science Department, University Of Tennessee.

Received May 2004; revised May 2005; accepted July 2005