Journal of Mathematical Modelling and Algorithms **1:** 283–300, 2002.

Optimal Hierarchies for Quadrilateral Surfaces

KUO-LIANG CHUNG[1], WEN-MING YAN[2] and JUNG-GEN WU[3]
[1]*Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, No. 43, Section 4, Keelung Road, Taipei, Taiwan 10672, R.O.C. e-mail: klchung@cs.ntust.edu.tw*
[2]*Department of Computer Science and Information Engineering, National Taiwan University, No. 1, Section 4, Roosevelt Road, Taipei, Taiwan 10764, R.O.C. e-mail: ganboon@csie.ntu.edu.tw*
[3]*Department of Information and Computer Education, National Taiwan Normal University, No. 162, Section 1, Hoping E. Road, Taipei, Taiwan 10610, R.O.C. e-mail: jgwu@ice.ntnu.edu.tw*

(Received: 21 June 2002)

Abstract. Multiresolution representation of quadrilateral surface approximation (MRQSA) is a useful representation for progressive graphics transmission in networks. Based on two requirements: (1) minimum mean square error and (2) fixed reduction ratio between levels, this paper first transforms the MRQSA problem into the problem of solving a sequence of near-Toeplitz tridiagonal linear systems. Employing the matrix perturbation technique, the MRQSA problem can be solved using about $24mn$ floating-point operations, i.e. linear time, if we are given a polygonal surface with $(2m-1) \times (2n-1)$ points. A numerical stability analysis is also given. To the best of our knowledge, this is the first time that such a linear algebra approach has been used for solving the MRQSA problem. Some experimental results are carried out to demonstrate the applicability of the proposed method.

Mathematics Subject Classification (2000): 65F05.

Key words: multiresolution representation, near-Toeplitz tridiagonal systems, quadrilateral surface, stability analysis.

1. Introduction

In computer graphics, computer-aided design, finite-element method, and solid modeling, we often model a polygonal surface by quadrangulations [8, 11–15, 17], each quadrilateral surface with four points. In fact, a triangular mesh can be converted into a quadrilateral mesh under a variety of constraints [16]. Given a quadrilateral surface, say P_s', in three-dimensional (3D) space, the quadrilateral surface approximation problem is to find a quadrilateral surface, say P_s, to approximate P_s' under an error tolerance. Basically, P_s can be viewed as a coarse representation of P_s'. In network communication, a 3D quadrilateral surface can have different levels/resolutions of representations, i.e. multiresolution. The lower-level representation has higher resolution, but is more complex. Each level of representation should not be totally independent of the others. This multiresolution representation is a typical paradigm for progressive transmission in the network.

In progressive transmission, a coarse rendering of the quadrilateral surface is sent first to give the receiver an early impression of content, i.e. a high-level representation, then subsequent transmission provides details of progressively finer resolution, i.e. a low-level representation. At the receiving end, the initial quadrilateral surface appears very coarse and comes steadily into 'focus'. Progressive transmission allows the receiver to terminate the transmission of a quadrilateral surface as soon as its shape is recognized, or as soon as further details of the quadrilateral surface cease to be of interest. Progressive transmission of a complex quadrilateral surface over a low speed channel, such as Internet applications, is superior and has many advantages over transmission in a single step.

Suppose we are given a quadrilateral surface with $(2m - 1) \times (2n - 1)$ points [8, 11]. Based on two requirements: (1) minimum mean-square error and (2) fixed reduction ratio between levels, this paper first transforms the MRQSA problem into the problem of solving n near-Toeplitz tridiagonal linear systems (NTTLS's), where the coefficient matrix in each NTTLS is of order $m \times m$. Employing the matrix perturbation technique, the MRQSA problem can be solved using about $24mn$ floating-point operations, i.e. linear time. To the best of our knowledge, this is the first time that such a numerical linear algebra-based approach is presented for solving the MRQSA problem. Furthermore, a numerical stability analysis is also given. To the best of our knowledge, this is the first time that such a linear algebra approach has been presented for solving the MRQSA problem. Some experimental results are carried out to demonstrate the applicability of the proposed method.

The rest of this paper is organized as follows. Section 2 presents how to transform the MRQSA problem into NTTLS's. Section 3 presents the proposed linear-time algorithm for solving NTTLS's. The numerical stability analysis is given in the same section. Two experimentations are carried out in Section 4 and some concluding remarks are addressed in Section 5.

2. Transforming the MRQSA Problem into NTTLS's

For clarity, this section first shows how to transform the multiresolution representation of the polygonal curve approximation (MRPCA) problem into the problem for solving one NTTLS. Then our nontrivial derivation is given to extend this transformation to map the MRQSA problem into a problem for solving multiple NTTLS's.

In a previous work [4], Chan and Chin first transformed the MRPCA problem into the problem of solving a NTTLS. Then, they presented a filter's convolution scheme to solve that NTTLS, but in [4] it lacks the time complexity analysis and numerical stability analysis in their approximate solution due to the filter's approach. Instead of using the filter's convolution scheme, a matrix perturbation technique is used here to solve the transformed NTTLS. The detailed number of floating-point operations required in our method and the related stability analysis

for solving the MRPCA problem and the MRQSA problem will be given in the next section.

For completeness, we revisit the transformation to map the polygonal curve approximation problem into a NTTLS. Suppose we are given a polygonal curve P'_c with $(2m-1)$ points which are represented by the set $\{P'_k = (x_k, y_k, z_k)$ for $k = 1, 2, \ldots, 2m-1\}$. Without loss of generality, we first want to obtain a coarser optimal approximation representation of P'_c, say P_c, under the minimum mean-square error and such that the reduction ratio between P'_c and P_c is $1/2$. This two-level polygonal curve approximation problem is called the *polygonal curve approximation problem*. Later, we will extend it to the MRPCA problem.

Following the similar approach used in [4], as shown in Figure 1, the coarse approximation P_c is denoted by the set $\{P_k = (x_k, y_k, z_k), k = 1, 2, \ldots, m\}$, which will be determined later. For $k = 1, 2, \ldots, m$, let Q_k be the midpoint of the line segment $P_k P_{k+1}$. As shown in Figure 1, the three points P_k, Q_k, and P_{k+1} in the polygonal curve P_c corresponding to the three points P'_{2k-1}, P'_{2k}, and P'_{2k+1} in the polygonal curve P'_c, respectively, will be used to measure the mean square error.

Based on the minimum mean-square error criterion, equivalently, the polygonal curve approximation problem is to determine $\{P_k$ for $k = 1, 2, \ldots, m\}$ such that the sum of the following square of distances is minimized:

$$\begin{aligned} f &= \sum_{k=1}^{m} d^2(P_k, P'_{2k-1}) + \sum_{k=1}^{m-1} d^2(Q_k, P'_{2k})^2 \\ &= f_1 + f_2 + f_3, \end{aligned}$$

where

$$f_1 = \sum_{k=1}^{m} (x_k - x'_{2k-1})^2 + \sum_{k=1}^{m-1} \left(\frac{x_k + x_{k+1}}{2} - x'_{2k} \right)^2,$$

$$f_2 = \sum_{k=1}^{m} (y_k - y'_{2k-1})^2 + \sum_{k=1}^{m-1} \left(\frac{y_k + y_{k+1}}{2} - y'_{2k} \right)^2,$$

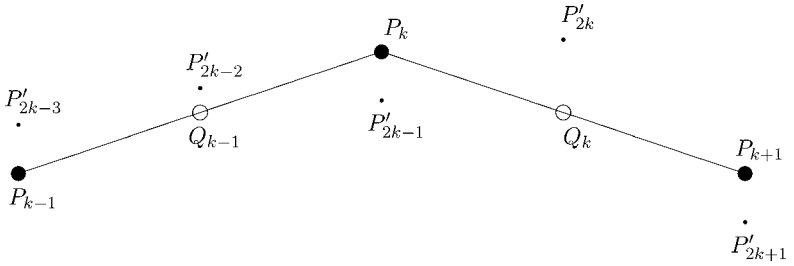

Figure 1. The relation between P'_c and P_c.

and

$$f_3 = \sum_{k=1}^{m}(z_k - z'_{2k-1})^2 + \sum_{k=1}^{m-1}\left(\frac{z_k + z_{k+1}}{2} - z'_{2k}\right)^2.$$

To minimize f, we must minimize f_1, f_2, and f_3 separately. We only derive the formula for minimizing f_1 since the related derivation is also applicable to derive the formulas for minimizing f_2 and f_3, respectively.

Let $d_{i,j}$ be 0 when $i = j$ and 1 when $i \neq j$. Differentiating f_1 with respect to x_k, it yields

$$\frac{\mathrm{d}f_1}{\mathrm{d}x_k} = 2(x_k - x'_{2k-1}) + d_{k,1}\left(\frac{x_{k-1} + x_k}{2} - x'_{2k-2}\right) + d_{k,n}\left(\frac{x_k + x_{k+1}}{2} - x'_{2k}\right),$$

for $k = 1, 2, 3, \ldots, m$. In order to obtain the set $\{x_k \text{ for } 1 \leqslant k \leqslant m\}$ for minimizing f_1, let $\mathrm{d}f_1/\mathrm{d}x_k = 0$. Then we have

$$4(x_k - x'_{2k-1}) + d_{k,1}(x_{k-1} + x_k - 2x'_{2k-2}) + d_{k,m}(x_k + x_{k+1} - 2x'_{2k}) = 0.$$

Hence, we have

$$d_{k,1}x_{k-1} + (4 + d_{k,1} + d_{k,m})x_k + d_{k,m}x_{k+1} = b_k,$$

where

$$b_k = 2(d_{k,1}x'_{2k-2} + 2x'_{2k-1} + d_{k,m}x'_{2k}).$$

Let

$$A_p = \begin{pmatrix} 5 & 1 & & & \\ 1 & 6 & 1 & & \\ & \cdot & \cdot & \cdot & \\ & & 1 & 6 & 1 \\ & & & 1 & 5 \end{pmatrix}_{p\times p},$$

we thus have

$$A_m \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}. \tag{1}$$

Up to now, we have transformed the polygonal curve approximation problem into the problem for solving one NTTLS with order $m \times m$. We now return to our main research issue, the MRQSA problem.

Following notations and definitions similar to the ones used in [5, 6], suppose we are given a polygonal surface P'_s with $(2m - 1) \times (2n - 1)$ points which are represented by the set

$$\{P'_{k,l} = (x_{k,l}, y_{k,l}, z_{k,l}) \text{ for } k = 1, 2, \ldots, 2m - 1 \text{ and } l = 1, 2, \ldots, 2n - 1\}.$$

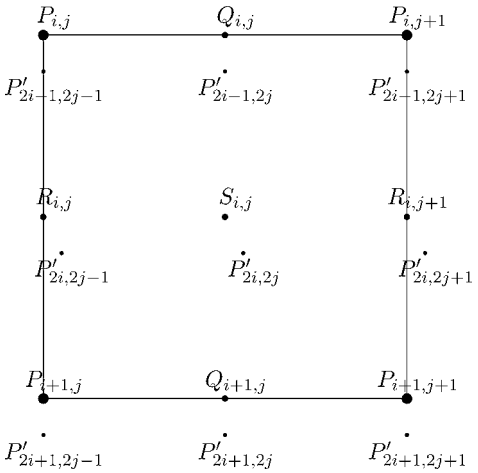

Figure 2. The relation between P'_s and P_s.

Without loss of generality, we first want to obtain a coarser optimal approximation representation of P'_s, say P_s, under the minimum mean square error and such that the reduction ratio between P'_s and P_s is 1/4. This two-level polygonal surface approximation problem is called *the PSA problem*. Later, we will extend it to solve the multiresolution representation PSA problem, i.e. the MRQSA problem.

In what follows, we want to extend the above transformation to map the so-called PSA problem into the problem for solving n NTTLS's, each one with order $m \times m$. The related derivation is somehow nontrivial.

As shown in Figure 2, the coarse approximation P_s is denoted by the set

$$\{P_{k,l} = (x_{k,l}, y_{k,l}, z_{k,l}), k = 1, 2, \ldots, m \text{ and } l = 1, 2, \ldots, n\},$$

which will be determined later. For $k = 1, 2, \ldots, m$, $l = 1, 2, \ldots, n$, we let $Q_{k,l}$ be the midpoint of the line segment $P_{k,l}P_{k,l+1}$ and let $R_{k,l}$ be the midpoint of the line segment $P_{k,l}P_{k+1,l}$. We let $S_{k,l}$ be the midpoint of the line segment $R_{k,l}R_{k,l+1}$. In fact, the point $S_{k,l}$ is the midpoint of the line segment $Q_{k,l}Q_{k+1,l}$. The point $S_{k,l}$ also can be considered as the midpoint of the quadrilateral polygon with four corners $P_{k,l}$, $P_{k,l+1}$, $P_{k+1,l}$, and $P_{k+1,l+1}$.

The four points $P_{i,j}$, $Q_{i,j}$, $R_{i,j}$, and $S_{i,j}$ in the polygonal surface P_s corresponding to the four points $P'_{2i-1,2j-1}$, $P'_{2i-1,2j}$, $P'_{2i,2j-1}$, and $P'_{2i,2j}$ in the polygonal surface P'_s, respectively, will be used to measure the mean square error.

Based on the minimum mean square error criterion, equivalently the PSA problem is to determine

$$\{P_{k,l} \text{ for } k = 1, 2, \ldots, m, l = 1, 2, \ldots, n\}$$

such that the sum of the following square of distances is minimized:

$$
\begin{aligned}
f &= \sum_{i=1}^{m}\sum_{j=1}^{n} d^2(P_{i,j}, P'_{2i-1,2j-1}) + \sum_{i=1}^{m}\sum_{j=1}^{n-1} d^2(Q_{i,j}, P'_{2i-1,2j})^2 + \\
&\quad + \sum_{i=1}^{m-1}\sum_{j=1}^{n} d^2(R_{i,j}, P'_{2i,2j-1}) + \sum_{i=1}^{m-1}\sum_{j=1}^{n-1} d^2(S_{i,j}, P'_{2i,2j})^2 \\
&= f_1 + f_2 + f_3,
\end{aligned}
$$

where

$$
\begin{aligned}
f_1 &= \sum_{i=1}^{m}\sum_{j=1}^{n}(x_{i,j} - x'_{2i-1,2j-1})^2 + \sum_{i=1}^{m}\sum_{j=1}^{n-1}\left(\frac{x_{i,j} + x_{i,j+1}}{2} - x'_{2i-1,2j}\right)^2 + \\
&\quad + \sum_{i=1}^{m-1}\sum_{j=1}^{n}\left(\frac{x_{i,j} + x_{i+1,j}}{2} - x'_{2i,2j-1}\right)^2 + \\
&\quad + \sum_{i=1}^{m-1}\sum_{j=1}^{n-1}\left(\frac{x_{i,j} + x_{i,j+1} + x_{i+1,j} + x_{i+1,j+1}}{4} - x'_{2i,2j}\right)^2, \\
f_2 &= \sum_{i=1}^{m}\sum_{j=1}^{n}(y_{i,j} - y'_{2i-1,2j-1})^2 + \sum_{i=1}^{m}\sum_{j=1}^{n-1}\left(\frac{y_{i,j} + y_{i,j+1}}{2} - y'_{2i-1,2j}\right)^2 + \\
&\quad + \sum_{i=1}^{m-1}\sum_{j=1}^{n}\left(\frac{y_{i,j} + y_{i+1,j}}{2} - y'_{2i,2j-1}\right)^2 + \\
&\quad + \sum_{i=1}^{m-1}\sum_{j=1}^{n-1}\left(\frac{y_{i,j} + y_{i,j+1} + y_{i+1,j} + y_{i+1,j+1}}{4} - y'_{2i,2j}\right)^2,
\end{aligned}
$$

and

$$
\begin{aligned}
f_3 &= \sum_{i=1}^{m}\sum_{j=1}^{n}(z_{i,j} - z'_{2i-1,2j-1})^2 + \sum_{i=1}^{m}\sum_{j=1}^{n-1}\left(\frac{z_{i,j} + z_{i,j+1}}{2} - z'_{2i-1,2j}\right)^2 + \\
&\quad + \sum_{i=1}^{m-1}\sum_{j=1}^{n}\left(\frac{z_{i,j} + z_{i+1,j}}{2} - z'_{2i,2j-1}\right)^2 + \\
&\quad + \sum_{i=1}^{m-1}\sum_{j=1}^{n-1}\left(\frac{z_{i,j} + z_{i,j+1} + z_{i+1,j} + z_{i+1,j+1}}{4} - x'_{2i,2j}\right)^2.
\end{aligned}
$$

To minimize f, we must minimize f_1, f_2, and f_3 separately. As described previously, we only derive the formula for minimizing f_1 since the related derivation is also applicable to derive the formulas for minimizing f_2 and f_3, respectively. The polygonal surface P_s consists of many quadrangulations, each quadrangulation with four line segments.

Differentiating f_1 with respect to $x_{k,l}$, yields

$$\begin{aligned}
\frac{\mathrm{d}f_1}{\mathrm{d}x_{k,l}} &= 2(x_{k,l} - x'_{2k-1,2l-1}) + d_{l,n}\left(\frac{x_{k,l} + x_{k,l+1}}{2} - x'_{2k-1,2l}\right) + \\
&\quad + d_{l,1}\left(\frac{x_{k,l-1} + x_{k,l}}{2} - x'_{2k-1,2l-2}\right) + \\
&\quad + d_{k,m}\left(\frac{x_{k,l} + x_{k+1,l}}{2} - x'_{2k,2l-1}\right) + d_{k,1}\left(\frac{x_{k-1,l} + x_{k,l}}{2} - x'_{2k-2,2l-1}\right) + \\
&\quad + \frac{1}{2}d_{k,m}d_{l,n}\left(\frac{x_{k,l} + x_{k,l+1} + x_{k+1,l} + x_{k+1,l+1}}{4} - x'_{2k,2l}\right) + \\
&\quad + \frac{1}{2}d_{k,m}d_{l,1}\left(\frac{x_{k,l-1} + x_{k,l} + x_{k+1,l-1} + x_{k+1,l}}{4} - x'_{2k,2l-2}\right) + \\
&\quad + \frac{1}{2}d_{k,1}d_{l,n}\left(\frac{x_{k-1,l} + x_{k-1,l+1} + x_{k,l} + x_{k,l+1}}{4} - x'_{2k-2,2l}\right) + \\
&\quad + \frac{1}{2}d_{k,1}d_{l,1}\left(\frac{x_{k-1,l-1} + x_{k-1,l} + x_{k,l-1} + x_{k,l}}{4} - x'_{2k-2,2l-2}\right) \\
&= \frac{1}{8}[16 + 4(d_{k,1} + d_{k,m} + d_{l,1} + d_{l,n}) + (d_{k,1} + d_{k,m})(d_{l,1} + d_{l,n})]x_{k,l} + \\
&\quad + \frac{1}{8}(4 + d_{l,n} + d_{l,1})d_{k,m}x_{k+1,l} + \frac{1}{8}(4 + d_{l,n} + d_{l,1})d_{k,1}x_{k-1,l} + \\
&\quad + \frac{1}{8}(4 + d_{k,m} + d_{k,1})d_{l,n}x_{k,l+1} + \frac{1}{8}(4 + d_{k,m} + d_{k,1})d_{l,1}x_{k,l-1} + \\
&\quad + \frac{1}{8}(d_{k,1}d_{l,1}x_{k-1,l-1} + d_{k,1}d_{l,n}x_{k-1,l+1} + \\
&\quad + d_{k,m}d_{l,1}x_{k+1,l-1} + d_{k,m}d_{l,n}x_{k+1,l+1}) - \\
&\quad - 2x'_{2k-1,2l-1} - (d_{l,n}x'_{2k-1,2l} + d_{l,1}x'_{2k-1,2l-2} + \\
&\quad + d_{k,m}x'_{2k,2l-1} + d_{k,1}x'_{2k-2,2l-1}) - \\
&\quad - \frac{1}{2}(d_{k,m}d_{l,n}x'_{2k,2l} + d_{k,m}d_{l,1}x'_{2k,2l-2} + \\
&\quad + d_{k,1}d_{l,n}x'_{2k-2,2l} + d_{k,1}d_{l,1}x'_{2k-2,2l-2}),
\end{aligned}$$

for $k = 1, 2, 3, \ldots, m$ and $l = 1, 2, 3, \ldots, n$.

In order to obtain the set $\{x_{k,l}$ for $1 \leqslant k \leqslant m$ and $1 \leqslant l \leqslant n\}$ for minimizing f_1, let $\mathrm{d}f_1/\mathrm{d}x_{k,l} = 0$. Then we have

$$\begin{aligned}
&(4 + d_{k,1} + d_{k,m})(4 + d_{l,1} + d_{l,n})x_{k,l} + \\
&\quad + (4 + d_{l,n} + d_{l,1})d_{k,m}x_{k+1,l} + (4 + d_{l,n} + d_{l,1})d_{k,1}x_{k-1,l} + \\
&\quad + (4 + d_{k,m} + d_{k,1})d_{l,n}x_{k,l+1} + (4 + d_{k,m} + d_{k,1})d_{l,1}x_{k,l-1} + \\
&\quad + (d_{k,1}d_{l,1}x_{k-1,l-1} + d_{k,1}d_{l,n}x_{k-1,l+1} + d_{k,m}d_{l,1}x_{k+1,l-1} + d_{k,m}d_{l,n}x_{k+1,l+1}) \\
&\quad = b_{k,l}, \qquad (2)
\end{aligned}$$

where

$$\begin{aligned} b_{k,l} &= 16x'_{2k-1,2l-1} + 8(d_{l,n}x'_{2k-1,2l} + d_{l,1}x'_{2k-1,2l-2} + \\ &\quad + d_{k,m}x'_{2k,2l-1} + d_{k,1}x'_{2k-2,2l-1}) + \\ &\quad + 4(d_{k,m}d_{l,n}x'_{2k,2l} + d_{k,m}d_{l,1}x'_{2k,2l-2} + d_{k,1}d_{l,n}x'_{2k-2,2l} + d_{k,1}d_{l,1}x'_{2k-2,2l-2}). \end{aligned}$$

Before solving Equation (2), we must calculate $b_{k,j}$ for $k = 1, 2, \ldots, m$ and $l = 1, 2, \ldots, n$. In what follows, all the time complexity requirement is counted in terms of the number of floating-point operations required.

LEMMA 1. *It takes about* $10mn$ *floating-point operations for preparing all* b_{kl}*'s.*

Proof. From

$$\begin{aligned} b_{k,l} &= 16x'_{2k-1,2l-1} + 8(d_{l,n}x'_{2k-1,2l} + d_{l,1}x'_{2k-1,2l-2} + \\ &\quad + d_{k,m}x'_{2k,2l-1} + d_{k,1}x'_{2k-2,2l-1}) + \\ &\quad + 4(d_{k,m}d_{l,n}x'_{2k,2l} + d_{k,m}d_{l,1}x'_{2k,2l-2} + d_{k,1}d_{l,n}x'_{2k-2,2l} + d_{k,1}d_{l,1}x'_{2k-2,2l-2}) \\ &= 4d_{k,1}(d_{l,1}x'_{2k-2,2l-2} + 2x'_{2k-2,2l-1} + d_{l,n}x'_{2k-2,2l}) + \\ &\quad + 8(d_{l,1}x'_{2k-1,2l-2} + 2x'_{2k-1,2l-1} + d_{l,n}x'_{2k-1,2l}) + \\ &\quad + 4d_{k,m}(d_{l,1}x'_{2k,2l-2} + 2x'_{2k,2l-1} + d_{l,n}x'_{2k,2l}), \end{aligned}$$

we first calculate

$$a_{k,l} = d_{l,1}x'_{k,2l-2} + 2x'_{k,2l-1} + d_{l,n}x'_{k,2l},$$

for $k = 1, 2, \ldots, 2m-1$ and $l = 1, 2, \ldots, n$ and it needs about $6mn$ floating-point operations. Further, we compute

$$b_{k,l} = 4(d_{k,1}a_{2k-2,l} + 2a_{2k-1,l} + d_{k,m}a_{2k,l}),$$

for $k = 1, 2, \ldots, m$ and $l = 1, 2, \ldots, n$ and it needs about $4mn$ floating-point operations. In total, it takes about $10mn$ floating-point operations to prepare all the b_{kl}'s for $k = 1, 2, \ldots, m$ and $l = 1, 2, \ldots, n$. We complete the proof. □

From the analysis in Lemma 1, it takes about $10mn$ floating-point operations to calculate $b_{k,l}$ for $k = 1, 2, \ldots, m$ and $l = 1, 2, \ldots, n$.

From the left-hand side of Equation (2), we have

$$\begin{aligned} &(4 + d_{k,1} + d_{k,m})(4 + d_{l,1} + d_{l,n})x_{k,l} + \\ &\quad + (4 + d_{l,n} + d_{l,1})d_{k,m}x_{k+1,l} + (4 + d_{l,n} + d_{l,1})d_{k,1}x_{k-1,l} + \\ &\quad + (4 + d_{k,m} + d_{k,1})d_{l,n}x_{k,l+1} + (4 + d_{k,m} + d_{k,1})d_{l,1}x_{k,l-1} + \\ &\quad + (d_{k,1}d_{l,1}x_{k-1,l-1} + d_{k,1}d_{l,n}x_{k-1,l+1} + d_{k,m}d_{l,1}x_{k+1,l-1} + d_{k,m}d_{l,n}x_{k+1,l+1}) \\ &= d_{k,1}[d_{l,1}x_{k-1,l-1} + (4 + d_{l,n} + d_{l,1})x_{k-1,l} + d_{l,n}x_{k-1,l+1}] \times \\ &\quad \times (4 + d_{k,1} + d_{k,m})[d_{l,1}x_{k,l-1} + (4 + d_{l,n} + d_{l,1})x_{k,l} + d_{l,n}x_{k,l+1}] + \\ &\quad + d_{k,m}[d_{l,1}x_{k+1,l-1} + (4 + d_{l,n} + d_{l,1})x_{k+1,l} + d_{l,n}x_{k+1,l+1}]. \end{aligned} \tag{3}$$

Let

$$t_{k,l} = d_{l,1}x_{k,l-1} + (4 + d_{l,n} + d_{l,1})x_{k,l} + d_{l,n}x_{k,l+1} \tag{4}$$

for $k = 1, 2, \ldots, m$ and $l = 1, 2, \ldots, n$, then by Equations (2) and (3), we have

$$d_{k,1}t_{k-1,l} + (4 + d_{k,1} + d_{k,m})t_{k,l} + d_{k,m}t_{k+1,l} = b_{k,l},$$

for $k = 1, 2, \ldots, m$ and $l = 1, 2, \ldots, n$.

Let

$$A_p = \begin{pmatrix} 5 & 1 & & & & \\ 1 & 6 & 1 & & & \\ & \cdot & \cdot & \cdot & & \\ & & & 1 & 6 & 1 \\ & & & & 1 & 5 \end{pmatrix}_{p\times p}$$

and from Equation (4), we have

$$A_m \begin{pmatrix} t_{1,l} \\ t_{2,l} \\ \vdots \\ t_{m,l} \end{pmatrix} = \begin{pmatrix} b_{1,l} \\ b_{2,l} \\ \vdots \\ b_{m,l} \end{pmatrix} \tag{5}$$

for $l = 1, 2, \ldots, n$.

In fact, Equation (5) consists of n NTTLS's to be solved, where the coefficient matrix in each system is of order $m \times m$. From Lemma 1, it is known that $10mn$ floating-point operations are required to prepare all the $b_{k,l}$'s.

3. The Proposed Algorithm and Stability Analysis

In this section, first a stable linear-time algorithm is presented to solve the PSA problem. Employing the matrix perturbation concept used in [7], the PSA problem can be solved using about $19mn$ floating-point operations. Further, we discuss how the MRQSA problem can be solved using about $24mn$ floating-point operations. Finally, a detailed stability analysis is given.

LEMMA 2. *It takes* $4p$ *floating-point operations for solving one NTTLS.*

Proof. For exposition, let $\mathbf{x} = t_{*,l}$, $\mathbf{b} = b_{*,l}$, $p = m$,

$$L_p = \begin{pmatrix} 1 & & & & \\ \frac{1}{a} & 1 & & & \\ & \cdot & \cdot & & \\ & & \frac{1}{a} & 1 & \\ & & & \frac{1}{a} & 1 \end{pmatrix}_{p\times p}, \qquad U_p = \begin{pmatrix} a & 1 & & & \\ & a & 1 & & \\ & & \cdot & \cdot & \\ & & & a & 1 \\ & & & & a \end{pmatrix}_{p\times p}$$

and $A'_p = L_p U_p$, then we have

$$A'_p = \begin{pmatrix} a & 1 & & & \\ 1 & b & 1 & & \\ & \cdot & \cdot & \cdot & \\ & & 1 & b & 1 \\ & & & 1 & b \end{pmatrix}_{p\times p},$$

where $b = a + 1/a$. Comparing the two matrices A_p and A'_p, let $b = 6$. We thus have $a + 1/a = 6$, so $a = 3 \pm 2\sqrt{2}$. We select $a = 3 + 2\sqrt{2}$ to make the two matrices L_p and U_p diagonally dominant and we have

$$A_p - A'_p = \begin{pmatrix} 2 - 2\sqrt{2} & & & \\ & & & \\ & & & \\ & & & -1 \end{pmatrix}_{p\times p}.$$

To solve $A_p\mathbf{x} = \mathbf{b}$, we first solve $A'_p\mathbf{z} = \mathbf{b}$, since

$$\begin{aligned} A_p\mathbf{z} &= A'_p\mathbf{z} + (2 - 2\sqrt{2})z_1\mathbf{e}_1 - z_p\mathbf{e}_p \\ &= \mathbf{b} + (2 - 2\sqrt{2})z_1\mathbf{e}_1 - z_p\mathbf{e}_p \\ &= A_p\mathbf{x} + (2 - 2\sqrt{2})z_1\mathbf{e}_1 - z_p\mathbf{e}_p, \end{aligned}$$

where

$$\mathbf{e}_1 = \underbrace{(1, 0, \ldots, 0)^t}_{p} \quad \text{and} \quad \mathbf{e}_p = \underbrace{(0, \ldots, 0, 1)^t}_{p}.$$

We thus have

$$A_p(\mathbf{z} - \mathbf{x}) = (2 - 2\sqrt{2})z_1\mathbf{e}_1 - z_p\mathbf{e}_p.$$

From

$$A_p(\underbrace{c, c^2, c^3, \ldots, c^k}_{k}, \underbrace{0, \ldots, 0}_{p-k})^t =$$
$$(\underbrace{5c + c^2, c(1 + 6c + c^2), c^2(1 + 6c + c^2), \ldots, c^{k-2}(1 + 6c + c^2), c^{k-1}(1 + 6c)}_{k},$$
$$\underbrace{c^k, 0, \ldots, 0}_{p-k})^t,$$

setting $1 + 6c + c^2 = 0$, we have $c = -3 \pm 2\sqrt{2}$. We select $c = -3 + 2\sqrt{2} = -0.172$. As a result, we have

$$A_p(\underbrace{c, c^2, c^3, \ldots, c^k}_{k}, \underbrace{0, \ldots, 0}_{p-k})^t$$
$$= (\underbrace{-1-c, 0, 0, \ldots, -c^{k+1}}_{k}, \underbrace{c^k, 0, \ldots, 0}_{p-k})^t.$$

Let

$$\mathbf{u} = (\underbrace{c, c^2, \ldots, c^k}_{k}, \underbrace{0, \ldots, 0}_{p-k})^t,$$

then we have

$$A_p\mathbf{u} = (2-2\sqrt{2})\mathbf{e}_1 - c^{k+1}\mathbf{e}_k + c^k\mathbf{e}_{k+1}.$$

Similarly, let

$$\mathbf{v} = (\underbrace{0, \ldots, 0}_{p-k}, \underbrace{c^k, c^{k-1}, \ldots, c^2, c}_{k})^t,$$

then we have

$$A_p\mathbf{v} = (2-2\sqrt{2})\mathbf{e}_p - c^{k+1}\mathbf{e}_{p-k+1} + c^k\mathbf{e}_{p-k}.$$

Therefore, it yields

$$A_p(\mathbf{z} - \mathbf{x} - z_1\mathbf{u} - \frac{1}{2-2\sqrt{2}}z_p\mathbf{v})$$
$$= z_1[c^{k+1}\mathbf{e}_k - c^k\mathbf{e}_{k+1}] + \frac{1}{2-2\sqrt{2}}z_p[c^{k+1}\mathbf{e}_{p-k+1} - c^k\mathbf{e}_{p-k}].$$

Let $\mathbf{y} = \mathbf{z} - z_1\mathbf{u} + \frac{1}{2}(1+\sqrt{2})z_p\mathbf{v}$, we thus have

$$A_p(\mathbf{y} - \mathbf{x}) = z_1[c^{k+1}\mathbf{e}_k - c^k\mathbf{e}_{k+1}] - \tfrac{1}{2}(1+\sqrt{2})z_p[c^{k+1}\mathbf{e}_{p-k+1} - c^k\mathbf{e}_{p-k}].$$

Since $c = -3+2\sqrt{2} = -0.172$, the right-hand side of the above equality is neglectable when k is large enough. For example, $k = 10$. Hence, we can solve $A_p\mathbf{x} = \mathbf{b}$ approximately by first solving $A'_p\mathbf{z} = \mathbf{b}$ and then obtaining $\mathbf{y} = \mathbf{z} - z_1\mathbf{u} + \frac{1}{2}(1+\sqrt{2})z_p\mathbf{v}$. The vector $\mathbf{y}$ is an approximate solution for $A_p\mathbf{x} = \mathbf{b}$.

To solve $A'_p\mathbf{z} = \mathbf{b}$, we first solve $L\mathbf{w} = \mathbf{b}$ and it takes $2p$ floating-point operations; we then solve $U\mathbf{z} = \mathbf{w}$ which takes $2p$ floating-point operations. Finally, the evaluation of $\mathbf{y} = \mathbf{z} - z_1\mathbf{u} + \frac{1}{2}(1+\sqrt{2})z_p\mathbf{v}$ needs constant floating-point operations for fixed k, e.g., $k = 10$.

The algorithm is:

******Solve $L\mathbf{w} = \mathbf{b}$******
$\alpha \leftarrow 1/a$
$w_1 \leftarrow b_1$
for $i \leftarrow 2$ to p
    $w_i \leftarrow b_i - \alpha * w_{i-1}$
end for
*****Solve $U\mathbf{z} = \mathbf{w}$, then store $\mathbf{z}$ to $\mathbf{y}$******
$y_p \leftarrow \alpha * w_p$
for $i \leftarrow p - 1$ downto 1
    $y_i \leftarrow \alpha * (b_i - y_{i+1})$
end for
*****Calculate $\mathbf{y} \leftarrow \mathbf{z} - z_1\mathbf{u} + \frac{1}{2}(1 + \sqrt{2})z_p\mathbf{v}$******
$c \leftarrow -3 + 2\sqrt{2}$
$t_1 \leftarrow -y_1$
$t_2 \leftarrow \frac{1}{2}(1 + \sqrt{2})$
for $i \leftarrow 1$ to k
*****k is a small integer, e.g., $k = 10$******
    $t_1 \leftarrow t_1 * c$
    $t_2 \leftarrow t_2 * c$
    $y_i \leftarrow y_i + t_1$
    $y_{p+1-i} \leftarrow y_{p+1-i} + t_2$
end for.

Consequently, solving $A_p\mathbf{x} = \mathbf{b}$ takes about $4p$ floating-point operations. Since

$$A_p(\mathbf{y} - \mathbf{x}) = z_1[c^{k+1}\mathbf{e}_k - c^k\mathbf{e}_{k+1}] - \tfrac{1}{2}(1 + \sqrt{2})z_p[c^{k+1}\mathbf{e}_{p-k+1} - c^k\mathbf{e}_{p-k}],$$

we have

$$\begin{aligned}
&\|A_p(\mathbf{y} - \mathbf{x})\|_\infty \\
&\quad = \|z_1[c^{k+1}\mathbf{e}_k - c^k\mathbf{e}_{k+1}] - \tfrac{1}{2}(1 + \sqrt{2})z_p[c^{k+1}\mathbf{e}_{p-k+1} - c^k\mathbf{e}_{p-k}]\|_\infty \\
&\quad \leqslant |z_1c^k| + \tfrac{1}{2}(1 + \sqrt{2})|z_pc^k| \\
&\quad \leqslant \frac{3 + \sqrt{2}}{2}|c^k|\|\mathbf{z}\|_\infty.
\end{aligned}$$

In order to the measure the error between the approximate solution $\mathbf{y}$ and the exact solution $\mathbf{x}$, we need to prove that for any vector $\mathbf{w}$, we have

$$\|\mathbf{w}\|_\infty \leqslant \tfrac{1}{4}\|A'_p\mathbf{w}\|_\infty \quad \text{and} \quad \|\mathbf{w}\|_\infty \leqslant \tfrac{1}{4}\|A_p\mathbf{w}\|_\infty.$$

The following will provide this proof:

For exposition, we first let $A'_p\mathbf{w} = \mathbf{d}$. From

$$A'_p = \begin{pmatrix} 3+2\sqrt{2} & 1 & & & & \\ 1 & 6 & 1 & & & \\ & \cdot & \cdot & \cdot & & \\ & & & 1 & 6 & 1 \\ & & & & 1 & 6 \end{pmatrix}_{p\times p},$$

we have

$$\begin{aligned} (3+2\sqrt{2})w_1 + w_2 &= d_1, \\ w_{i-1} + 6w_i + w_{i+1} &= d_i, \quad \text{for } 2 \leqslant i \leqslant p-1, \\ w_{p-1} + 6w_p &= d_p. \end{aligned}$$

Suppose $\|\mathbf{w}\|_\infty = |w_1|$. Due to $(3+2\sqrt{2})w_1 = d_1 - w_2$, we thus have

$$(3+2\sqrt{2})\|\mathbf{w}\|_\infty = |(3+2\sqrt{2})w_1| \leqslant |d_1| + |w_2| \leqslant \|\mathbf{d}\|_\infty + \|\mathbf{w}\|_\infty,$$

that is, $(2+2\sqrt{2})\|\mathbf{w}\|_\infty \leqslant \|\mathbf{d}\|_\infty$. Suppose $\|\mathbf{w}\|_\infty = |w_i|$ for some i, $2 \leqslant i \leqslant p-1$. Because of $6w_i = d_i - w_{i-1} - w_{i+1}$, we have

$$6\|\mathbf{w}\|_\infty = |6w_i| \leqslant |d_i| + |w_{i-1}| + |w_{i+1}| \leqslant \|\mathbf{d}\|_\infty + 2\|\mathbf{w}\|_\infty,$$

that is, $4\|\mathbf{w}\|_\infty \leqslant \|\mathbf{d}\|_\infty$. Suppose $\|\mathbf{w}\|_\infty = |w_p|$. Because of $6w_p = d_p - w_{p-1}$, we have

$$6\|\mathbf{w}\|_\infty = |6w_p| \leqslant |d_p| + |w_{p-1}| \leqslant \|\mathbf{d}\|_\infty + \|\mathbf{w}\|_\infty,$$

that is, $5\|\mathbf{w}\|_\infty \leqslant \|\mathbf{d}\|_\infty$. Combining the analysis in the three cases, we have

$$\|\mathbf{w}\|_\infty \leqslant \max\left(\frac{1}{2+2\sqrt{2}}, \frac{1}{4}, \frac{1}{5}\right)\|\mathbf{d}\|_\infty = \frac{1}{4}\|\mathbf{d}\|_\infty.$$

That is, we have $\|\mathbf{w}\|_\infty \leqslant \frac{1}{4}\|A'_p\mathbf{w}\|_\infty$. By the same arguments, it is easy to show that $\|\mathbf{w}\|_\infty \leqslant \frac{1}{4}\|A_p\mathbf{w}\|_\infty$. We complete the proof. □

By the above analysis, we have

$$\|\mathbf{y}-\mathbf{x}\|_\infty \leqslant \tfrac{1}{4}\|A_p(\mathbf{y}-\mathbf{x})\|_\infty \quad \text{and} \quad \|\mathbf{z}\|_\infty \leqslant \tfrac{1}{4}\|A'_p\mathbf{z}\|_\infty = \tfrac{1}{4}\|\mathbf{b}\|_\infty.$$

Further, we have

$$\begin{aligned} \|\mathbf{y}-\mathbf{x}\|_\infty &\leqslant \tfrac{1}{4}\|A_p(\mathbf{y}-\mathbf{x})\|_\infty \\ &\leqslant \frac{3+\sqrt{2}}{8}|c^k|\|\mathbf{z}\|_\infty \\ &\leqslant \frac{3+\sqrt{2}}{32}|c^k|\|\mathbf{b}\|_\infty, \end{aligned}$$

so the relative residual is bounded by

$$\frac{\|\mathbf{y}-\mathbf{x}\|_\infty}{\|\mathbf{b}\|_\infty} \leqslant \frac{3+\sqrt{2}}{32}|c^k|.$$

Setting $\sqrt{2} = 1.4142135$ and $k = 10$, the relative residual is bounded by 2.4×10^{-9}. From a practical viewpoint, the proposed method for solving one NTTLS is quite stable. We complete the proof. □

From Lemma 2, each NTTLS in Equation (5) can be solved using $4m$ floating-point operations with a quite good relative residual. In fact, the proposed results (see Lemmas 1 and 2) also provide a linear-time algorithm with good stability for solving the polygonal curve approximation problem.

Since there are n NTTLS's to be solved in Equation (5), we overall need $4mn$ floating-point operations for solving Equation (5). Combining the time required in preparing all the $b_{k,l}$'s (see Lemma 1), we have the following result:

LEMMA 3. *Equation* (5) *can be solved using* $14mn$ *floating-point operations.*

From Equation (3), we finally want to solve

$$A_n \begin{pmatrix} x_{k,1} \\ x_{k,2} \\ \vdots \\ x_{k,n} \end{pmatrix} = \begin{pmatrix} t_{k,1} \\ t_{k,2} \\ \vdots \\ t_{k,n} \end{pmatrix}, \tag{6}$$

for $k = 1, 2, \ldots, m$. Note that the vector at the right-hand side of Equation (6) has been obtained in Lemma 3.

It is observed that Equation (6) consists of m NTTLS's to be solved and the coefficient matrix in each system is of order $n \times n$. Each NTTLS in Equation (6) can be solved using $4n$ floating-point operations. Since there are m equations to be solved in Equation (6), we have the following result:

LEMMA 4. *Equation* (6) *can be solved using* $4mn$ *floating-point operations.*

Combining the results of Lemma 3 and Lemma 4, we have the result.

THEOREM 1. *Following the method mentioned above, the PSA problem can be solved using* $18mn$ *floating-point operations.*

As described in Section 2, the reduction ratio between any two consecutive levels in the MRQSA representation is $1/4$. From Theorem 1, it takes about $18mn \times \frac{4}{3}$ $(= 18mn(1 + \frac{1}{4} + (\frac{1}{4})^2 + \cdots))$ floating-point operations to solve the MRQSA problem. We thus have the main result.

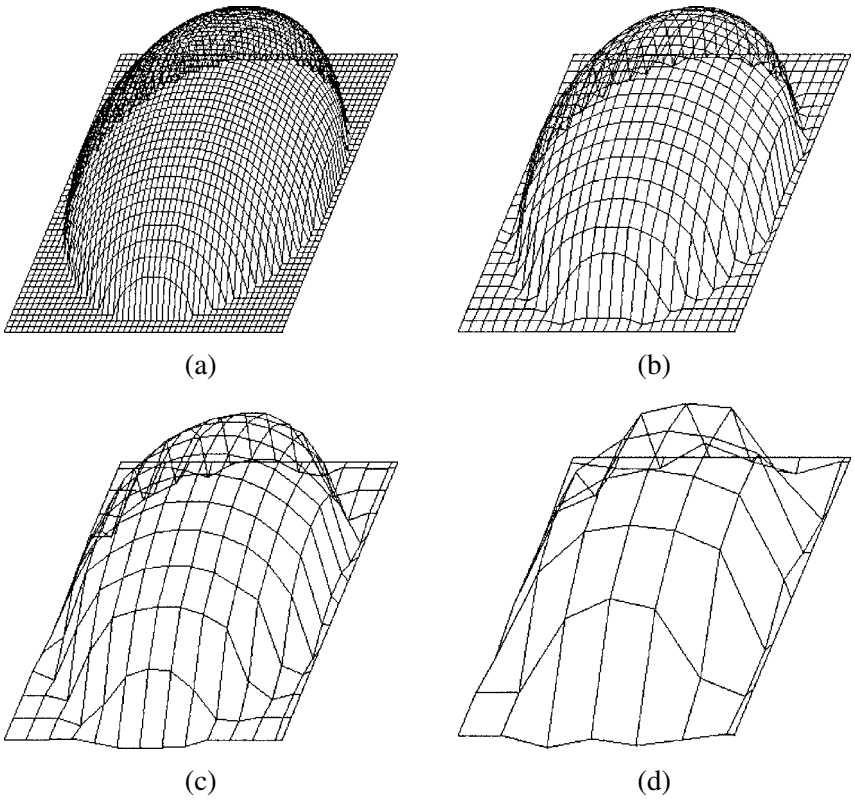

Figure 3. The first simulation with four levels.

THEOREM 2. *Using the proposed algorithm, the MRQSA problem can be solved in about* $24mn$ *floating-point operations, i.e.* $O(mn)$ *time.*

To the best of our knowledge, this is the first time that such a linear algorithm with thorough time complexity analysis and stability analysis has been used for solving the MRQSA problem.

If the given polygonal surface is circular, e.g., a torus, using the similar derivation mentioned in this paper, the sequence of the corresponding tridiagonal systems is circular. For saving space of context, the detailed derivation is omitted.

4. Experimental Results

The algorithm that we have presented allows the representation of a mesh consisting of a large number of quadrangulations at multiple levels of detail, satisfying the two criteria: (1) minimum mean square error and (2) fixed reduction ratio, say $1/4$, between levels, and requiring linear time.

We have applied the proposed algorithm to two meshes. Figures 3(a) and 4(a) show the first mesh and the second mesh, respectively. Figures 3(b)–(d) and

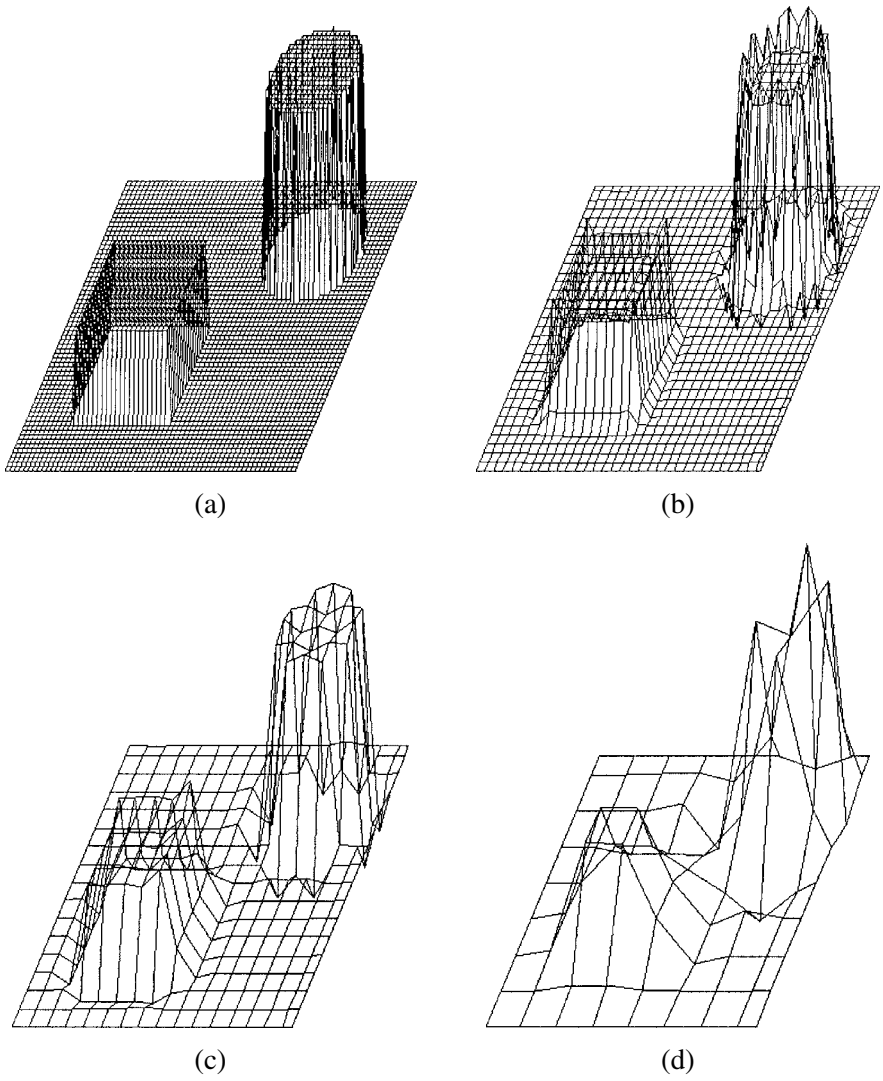

Figure 4. The second simulation with four levels.

4(b)–(d) illustrate the meshes at various levels. The experimental results demonstrate encouraging results.

5. Conclusions

The MRQSA problem is practical in network progressive transmission. Given a polygonal surface with $(2m-1)\times(2n-1)$ points, the main contributions are threefold: (1) providing a new derivation to map the MRQSA problem into a sequence of NTTLS's, (2) presenting a linear-time algorithm needing $24mn$ floating-point

operations, and (3) giving a detailed stability analysis for the proposed algorithm. The first contribution also extends the result of Chan and Chin [4] from the domain of polygonal curves to the domain of polygonal surfaces. Two experiments have been carried out to demonstrate the applicability of the proposed results.

If the given polygonal surface is circular, e.g., a torus, using the similar derivation mentioned in this paper, the sequence of the corresponding tridiagonal systems is circular. To save space, the detailed derivation is omitted.

Another favored mesh used in many applications is the triangular mesh [3]. Constructing hierarchies for triangle meshes is also very important. In [9, 10], some efficient triangle-collapse operations are presented for build the level-of-detail representation of triangle meshes. From the transformation from the MRQSA problem with reduction ratio $1/4$ to the solutions of $m+n$ NTTLS's mentioned in this paper, it is an interesting research issue to apply the results of this paper to designing a numerical linear algebra-based algorithm for the multiresolution representation of triangle meshes.

Acknowledgements

The authors appreciate the anonymous referees and the Editor-in-Chief Prof., M. Rayward-Smith, for their valuable comments that lead to the improved version of this paper. K.-L.C. was supported by NSC89-2213-E011-061; W.-M.Y. was supported by NSC87-2119-M002-006; and J.-G.W. was supported by NSC89-2614-H-003-001-F020.

References

1. Bartels, R. H., Beatty, J. C. and Barsky, B. A.: *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling,* Morgan Kaufmann, San Mateo, CA, 1987.
2. Berg, M. and Dobrindt, K. T. G.: On levels of detail in terrains, Technical Report, Utrecht University, 1995.
3. Bern, M. and Eppstein, D.: Mesh generation and optimal triangulation, In: F. K. Hwang and D. Z. Du (eds), *Computing in Euclidean Geometry*, World Scientific, Singapore, 1992.
4. Chan, K. W. and Chin, F.: Optimal multiresolution polygonal approximation, In: *The Third Annual International Conference COCOON'97*, 1997, pp. 32–41.
5. Cheng, F. and Goshtasby, A.: A parallel B-spline surface fitting algorithm, *ACM Trans. Graphics* **8**(1) (1989), 41–50.
6. Cheng, F., Wasilkowski, G. W., Wang, J., Zhang, C. and Wang, W.: Parallel B-spline surface interpolation on a mesh-connected processor array, *J. Parallel Distributed Comput.* **24**(2) (1995), 224–229.
7. Chung, K. L. and Yan, W. M.: Parallel B-spline surface fitting on mesh-connected computers, *J. Parallel Distributed Comput.* **35** (1996), 205–210.
8. Foley, J. D., van Dam, A., Feiner, S. K. and Hughes, J. F.: *Computer Graphics: Principles and Practice,* 2nd edn, Chapter 11: Representing Curves and Surfaces, Addison-Wesley, Reading, Mass., 1996.
9. Gieng, T. S., Hamann, B., Joy, K. I., Schussman, G. L. and Trotts, I. J.: Constructing hierarchies for triangle meshes, *IEEE Trans. Visualization Comput. Graphics* **4**(2) (1998), 145–161.

10. Gross, M. H., Staadt, O. G. and Gatti, R.: Efficient triangular surface approximations using wavelets and quadtree data structures, *IEEE Trans. Visualization Comput. Graphics* **2**(2) (1996), 130–143.
11. Hearn, D. and Baker, M. P.: *Computer Graphics*, 2nd edn, Chapter 10: Three-dimensional Object Representations, Prentice-Hall, Englewood Cliffs, 1994.
12. Heighway, E.: A mesh generator for automatically subdividing irregular polygons into quadrilaterals, *IEEE Trans. Magnetics* **19**(6) (1983), 2535–2538.
13. Ho-Le, K.: Finite element mesh generation methods: A review and classification, *Computer Aided Design* **20** (1988), 27–38.
14. Joe, B.: Quadrilateral mesh generation in polygonal regions, *Computer Aided Design* **27**(3) (1995), 209–222.
15. Lubiw, A.: Decomposing polygonal regions into convex quadrilaterals, *Proc. 1st ACM Sympos. Computational Geometry*, 1985, pp. 97–106.
16. Ramaswami, S., Ramos, P. and Toussaint, G.: Converting triangulations to quadrangulations, *Comput. Geom.* **9** (1998), 257–276.
17. Watt, A. and Policarpo, F.: *The Computer Image*, Chapter 2: Representation and Modeling of Three-dimensional Objects, Addison-Wesley, 1999.