The goal of 3D surface simplification is to reduce the storage cost of 3D models. A 3D animation model typically consists of several 3D models. Therefore, to ensure that animation models are realistic, numerous triangles are often required. However, animation models that have a high storage cost have a substantial computational cost. Hence, surface simplification methods are adopted to reduce the number of triangles and computational cost of 3D models. Quadric error metrics (QEM) has recently been identified as one of the most effective methods for simplifying static models. To simplify animation models by using QEM, Mohr and Gleicher summed the QEM of all frames. However, homogeneous coordinate problems cannot be considered completely by using QEM. To resolve this problem, this paper proposes a robust homogeneous coordinate transformation that improves the animation simplification method proposed by Mohr and Gleicher. In this study, the root mean square errors of the proposed method were compared with those of the method proposed by Mohr and Gleicher, and the experimental results indicated that the proposed approach can preserve more contour features than Mohr’s method can at the same simplification ratio.

2. Related Works

2.1. Quadric Error Metrics. Because the QEM method proposed by Garland and Heckbert [21] has recently been...
identified as one of the most effective simplification methods, numerous experts have studied it, extending its applications in level of detail (LOD) [23], radiosity rendering [23], and animation model simplification [22]. QEM involves contracting vertex \( V_1 \) and vertex \( V_2 \) into a new vertex, \( V \), the position of which is obtained according to the minimal distance between a vertex and the conjoining triangles, as shown in Figure 2. This minimal distance represents the error of simplification. This method enables the outer shape of a 3D model to be retained after the model is simplified and reduces the errors caused by surface simplification. The new estimated position of \( v \) strongly influences changes in the surrounding triangles.

QEM involves measuring the error by using a quadric metric as follows:

\[
\text{Error}(v) = v^T \cdot (\sum K_p) \cdot v,
\]

where \( v \) is an estimated point and \( K_p \) represents a quadric metric defined by plane \( p: ax + by + cz + d = 0 \). When \( a^2 + b^2 + c^2 = 1 \), the distance from any vertex \( v \) to the plane \( p \) is represented as \( (p^Tv)^2 = v^T(p^Tp)v = v^TK_pv \). In other words, \( K_p \) can be represented as follows, and the initial matrix \( Q \) can be obtained by conglomerating the planes for the triangles meeting at \( v \):

\[
Q = \sum_{p \in \text{planes}(v)} K_p,
\]

\[
K_p = p \cdot p^T = \begin{bmatrix}
    a^2 & ab & ac & ad \\
    ab & b^2 & bc & bd \\
    ac & bc & c^2 & cd \\
    ad & bd & cd & d^2
\end{bmatrix},
\]

where \( \text{planes}(v) \) indicates the triangles meeting at the vertex \( v \). When the vertex pair \((v_1, v_2)\) is simplified into a new vertex \( v \), the matrix \( Q \) of vertex \( v \) is converted into \((Q_1 + Q_2)\), where \( Q_1 \) and \( Q_2 \) are the quadric matrices of vertices \( v_1 \) and \( v_2 \), respectively. In other words, the cost of contraction is simply computed as \( v^T(Q_1 + Q_2)v \). This method rapidly provides a low-error solution for surface simplification.

2.2 Deformation Sensitive Decimation. Although QEM features advantages such as a high computation speed and a low average error, it can be used to simplify only static models. That is, QEM can obtain a good simplified result for one of all frame examples in an animation but cannot work well for each frame example at the same time. For example, in the animation model containing five frame examples shown in Figure 3(a), the first frame example is simplified first, and the fifth frame example is simplified using the collapse sequence based on the first frame example, as shown in Figure 3(b). The first frame example can be simplified adequately by using this approach, but the fifth frame example cannot.

To apply QEM in simplifying all frame examples in an animation model, Mohr and Gleicher [22] proposed DSD method. This method is used to generate the contraction cost by considering all frame examples in an animation model. The method involves tracking the error quadric and summing the matrices per vertex for each frame example. The error measurement is modified as follows:

\[
\sum_{i=1}^{k} \bar{v}_i^T (Q_{i,v_1} + Q_{i,v_2}) \bar{v}_i,
\]

where \( k \) is the number of frame examples, \( \bar{v}_i \) represents the new vertex formed by collapsing vertices \( v_1 \) and \( v_2 \) in the \( i \)th frame example, and \( Q_{i,v_1} \) and \( Q_{i,v_2} \) represent the error quadrics of vertices \( v_1 \) and \( v_2 \), respectively.

Experimental results indicated that when DSD is used to simplify the animation model in Figure 3(c), the simplification of the fifth frame example is superior to that generated using the QEM of first frame example.
2.3. Progressive Multiresolution Meshes. To apply QEM in high degree of nonrigid deformation, Kircher and Garland proposed a new multiresolution method to deform surfaces [24]. This method comprises two major steps. First, a multilevel mesh is created for the first frame model in the deformation sequence. Each level is numbered in an increasing order from fine to coarse, starting with $M_0$. This method uses QEM to cluster all vertices in $M_0$, generate a simplified model $M_1$, and complete the initial hierarchy $H_0$ gradually.

In the second step, a reclustering algorithm is employed to improve the time-varying hierarchies for each subsequent frame model. To update the time-varying hierarchies, a swap approach is used to recluster vertices; in other words, the $(i+1)$th hierarchy $H_{i+1}$ is generated from the $i$th hierarchy $H_i$, and all swap operations varying from $H_i$ to $H_{i+1}$ are recorded, as shown in Figure 4.

According to the experimental results [24], the simplification errors generated by progressive multiresolution meshes were between the DSD and Direct QSlim. Because this method updated hierarchies from the first frame hierarchy, the models similar to the first frame model (frames 1, 13, 25, 37, and 49) were simplified well. By contrast, the models that greatly differed from the first frame model, such as frames 6, 18, 30, and 42, could not be simplified effectively using progressive multiresolution meshes. Local maximal simplification errors were observed in frames 6, 18, 30, and 42, and these errors were nearly equal to those of the DSD.

Many studies have recently referenced DSD in proposing numerous related works, including progressive multiresolution meshes [24], articulated meshes [25, 26], animation key-frame extraction [27], and analytic error metrics [28]. However, one shortfall of the DSD method, the homogeneous coordinate problem, must be resolved.

To manage homogeneous coordinates, DSD entails replacing the final row of the summed quadric matrix $Q$ with $\begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}$ and determining the $Q^{-1}$ to obtain the optimal
vertex position. However, $Q^{-1}$ does not always exist. Therefore, we propose a RHCT to improve the DSD method.

### 3. Robust Homogeneous Coordinate Transformation

An animation model generally comprises several frame examples, and each frame example has a distinct pose. The surface information, such as information on the position and normal vector, of a vertex differs among frame examples. In brief, the local coordinate system of a vertex varies among the frame examples of an animation model. Before an animation model is simplified, a vertex present in different frame examples should be transformed to one local coordinate space. This transformation is conducted by applying the following theorems.

**Theorem 1** (Euler’s rotation [29]). In 3D space, the transform for rotating a point around the origin of a plane is expressed as follows:

$$R_\beta = \begin{bmatrix} \cos \beta & -\sin \beta & 0 \\ \sin \beta & \cos \beta & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$  \hspace{1cm} (4)

Any rotation may be described using three angles. If the rotations are written using rotation matrices $D, C,$ and $B,$ then a general rotation $A$ can be written as $A = B \times C \times D.$

**Theorem 2** (equivalence of an orthogonal matrix to a rotation matrix [31]). A rotation matrix is a matrix that, when multiplied by a vector, rotates that vector in its $n$-dimensional domain. A rotation matrix is equivalent to an orthogonal matrix, and an orthogonal matrix is defined as a matrix of which the transpose is equal to its determinant, which is equal to 1:

$$Q^{-1} = Q^T, \hspace{1cm} |Q| = 1.$$  \hspace{1cm} (5)

According to Theorem 1, the matrices for rotation around the $x$-axis by $\alpha,$ around the $y$-axis by $\beta,$ and around the $z$-axis by $\gamma$ are expressed as:

$$R_{\alpha,x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & -\sin \alpha \\ 0 & \sin \alpha & \cos \alpha \end{bmatrix},$$

$$R_{\beta,y} = \begin{bmatrix} \cos \beta & 0 & \sin \beta \\ 0 & 1 & 0 \\ -\sin \beta & 0 & \cos \beta \end{bmatrix},$$

$$R_{\gamma,z} = \begin{bmatrix} \cos \gamma & -\sin \gamma & 0 \\ \sin \gamma & \cos \gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$  \hspace{1cm} (6)

**Theorem 3** (rotating a point around an arbitrary axis [30]). Assume the axis of rotation is given by the unit vector: $\vec{n} = ai + bj + ck.$ $P(x_p, y_p, z_p)$ is the point that is rotated by angle $\theta$ to $P'(x'_p, y'_p, z'_p),$ as shown in Figure 5. The rotation transform is:

$$x' = \frac{x}{K(1+c)} + \frac{a}{K(1+c)}K \cos \theta$$

$$y' = \frac{y}{K(1+c)} + \frac{b}{K(1+c)}K \cos \theta$$

$$z' = \frac{z}{K(1+c)} + \frac{c}{K(1+c)}K \cos \theta.$$  \hspace{1cm} (7)

where $K = 1 - \cos \theta.$

To transform the vertex in different frame examples to one local coordinate space, the local coordinate spaces in
the first frame example are used as major coordinate spaces. Each local coordinate laid on each frame example must be transformed to that on the first frame example. Suppose that the normal vector and tangent plane of a vertex \( V_{i,j} \) in frame \( j \) are noted at \( n_{i,j} = [n_{x,i,j}, n_{y,i,j}, n_{z,i,j}]^T \) and \( t_{i,j} \), respectively, and \( M_{i,j} \) represents the mesh of connecting vertex \( V_{i,j} \). To transform the mesh \( M_{i,j} \) to the coordinate space of \( M_{i,1} \), the normal vector \( n_{i,j} \) and tangent plane \( t_{i,j} \) at vertex \( V_{i,j} \) must be transformed to \( n_{i,1} \) and \( t_{i,1} \) at vertex \( V_{i,1} \). According to Theorem 3, the vertex of local coordinate space of \( M_{i,1} \) by rotating the normal vector \( n_{i,j} \):

\[
T_{i,j} = \begin{bmatrix}
\tilde{n}_{x,i,j}^2K + \cos \theta_{i,j} & \tilde{n}_{x,i,j}\tilde{n}_{y,i,j}K - \tilde{n}_{z,i,j} \sin \theta_{i,j} & \tilde{n}_{x,i,j}\tilde{n}_{z,i,j}K + \tilde{n}_{y,i,j} \sin \theta_{i,j} & 0 \\
\tilde{n}_{x,i,j}\tilde{n}_{y,i,j}K + \tilde{n}_{z,i,j} \sin \theta_{i,j} & \tilde{n}_{y,i,j}^2K + \cos \theta_{i,j} & \tilde{n}_{y,i,j}\tilde{n}_{z,i,j}K - \tilde{n}_{x,i,j} \sin \theta_{i,j} & 0 \\
\tilde{n}_{x,i,j}\tilde{n}_{z,i,j}K - \tilde{n}_{y,i,j} \sin \theta_{i,j} & \tilde{n}_{y,i,j}\tilde{n}_{z,i,j}K + \tilde{n}_{x,i,j} \sin \theta_{i,j} & \tilde{n}_{z,i,j}^2K + \cos \theta_{i,j} & 0 \\
0 & 0 & 0 & 1
\end{bmatrix},
\]

\[
T_{i,j}^{-1} = T_{i,1}^{-1} = \begin{bmatrix}
\tilde{n}_{x,i,j}^2K + \cos \theta_{i,j} & \tilde{n}_{x,i,j}\tilde{n}_{y,i,j}K + \tilde{n}_{z,i,j} \sin \theta_{i,j} & \tilde{n}_{x,i,j}\tilde{n}_{z,i,j}K - \tilde{n}_{y,i,j} \sin \theta_{i,j} & 0 \\
\tilde{n}_{x,i,j}\tilde{n}_{y,i,j}K - \tilde{n}_{z,i,j} \sin \theta_{i,j} & \tilde{n}_{y,i,j}^2K + \cos \theta_{i,j} & \tilde{n}_{y,i,j}\tilde{n}_{z,i,j}K + \tilde{n}_{x,i,j} \sin \theta_{i,j} & 0 \\
\tilde{n}_{x,i,j}\tilde{n}_{z,i,j}K + \tilde{n}_{y,i,j} \sin \theta_{i,j} & \tilde{n}_{y,i,j}\tilde{n}_{z,i,j}K - \tilde{n}_{x,i,j} \sin \theta_{i,j} & \tilde{n}_{z,i,j}^2K + \cos \theta_{i,j} & 0 \\
0 & 0 & 0 & 1
\end{bmatrix},
\]

where \( k \) is the number of frame examples, \( \bar{v}_i \) is the new vertex formed by collapsing vertices \( v_1 \) and \( v_2 \) in the \( i \)th frame example, \( Q_{v_1,v_i} \) and \( Q_{v_2,v_i} \) represent the error quadrics of vertices \( v_1 \) and \( v_2 \), respectively, \( T_{v_i,v_j} \) is the matrix that transforms all faces incident to \( v_1 \) from the \( i \)th frame example to the first frame example, and \( T_{v_i,v_j} \) is the matrix that transforms all faces incident to \( v_2 \) from the \( i \)th frame example to the first frame example.

4. The Proposed Algorithm

The proposed algorithm refers to and amends DSD methods and includes eight major steps described as follows.

**Step 1** (calculate the QEM of each vertex \( v_{i,j} \)). Its QEM is \( Q(v_{i,j}) = Q_{v_1,v_i} + \Sigma K_p \), where \( p \) denotes the triangle plane that contains the point \( v_{i,j} \), and \( K_p \) represents \( 4 \times 4 \) metric of the plane \( p \), as shown in (2).

**Step 2** (calculate the transformation matrix and modify the QEM). The transformation matrix \( T_{i,j} \) shown in (8) must be
Figure 6: Simplification results of the horse animation model (frame examples 1, 21, and 41).
calculated, and then the new QEM is represented as follows. Additionally, \( Q'(v_{i,j}) \) will be \( Q(v_{i,j}) \) when \( j \) equals 1:

\[
Q'(v_{i,j}) = \sum_{p \in \text{planes}(v_{i,j})} T_{i,j} \times K_p = T_{i,j} \times \left( \sum_{p \in \text{planes}(v_{i,j})} K_p \right) = T_{i,j} \times Q(v_{i,j}).
\] (11)

**Step 3.** Sum all QEMs of vertex \( v_i \) in different frame example:

\[
Q'(v_i) = \sum_{j=1}^{k} Q'(v_{i,j}).
\] (12)

**Step 4** (choose each \((v_m, v_n)\) pair to calculate the minimum error produced due to simplification). The \((v_m, v_n)\) pair must satisfy the following conditions:

(a) \((v_m, v_n)\) is an edge or

(b) \((v_m, v_n)\) is not an edge and \(\|v_m - v_n\| < t\), where \(t\) denotes a threshold value specified by the user.

**Step 5.** Choose the lowest error vertex-pair \((v_m, v_n)\) as an object of simplification.

**Step 6.** Contract the vertex-pair \((v_m, v_n)\) into \(v\) and calculate its QEM \( Q(v) \), where \( Q(v) = Q(v_m) + Q(v_n) \).

**Step 7.** Update all the information of the vertices adjacent to \(v_m\) and \(v_n\).

**Step 8.** Repeat the previous steps until the assigned number of triangles is reached after simplification.

Through Steps 1–3 calculation, the summed QEM with RHCT is obtained, and it is used to contract vertex pairs. The experimental results presented here demonstrate that the RHCT helps to truly reduce the error due to simplification.

### 5. Experimental Results

This study used the Intel Core i7-2670QM 2.2 GHZ CPU, which features 4 GB of memory, as the main experimental environment, and Visual C++ was used as a programming development tool. Animation models of a horse, a camel, and an elephant were the subjects of the experiment. Each animated sequence comprised 48 frame examples. In the experiment, the root mean square (RMS) errors generated when using DSD to simplify the three animation models were compared with those generated by the proposed method. The experimental results obtained when the horse model was tested are shown in Figure 6, where Figure 6(a) is the original model which consisted of 8431 vertices and 16842 triangles. Figures 6(b) and 6(c) show the results obtained when the horse model was simplified into 5000 triangles by using DSD and the proposed method, respectively.

To estimate the distortion errors of simplified animation models, this study used Metro [32] to calculate the RMS error for each simplified frame example in an animation model and then to determine the mean of these RMS errors. Table 1 and Figure 7 show the mean distortion errors that occurred when the horse animation was simplified into 10000 triangles, 5000 triangles, 2000 triangles, and 1000 triangles by using the DSD and RHCT methods. The RMS errors of the two methods differed substantially when the models were simplified. The proposed method produced \(0.828 \times 10^{-3}\) RMS errors when it was used to simplify the horse model into 5000 triangles and, compared with the model simplified using DSD and the original model comprising 16842 triangles, reduced RMS errors by 53.9% to within an error range of \(1.796 \times 10^{-3}\). When the model was simplified into 2000 and 1000 triangles, the proposed method achieved improvements of 63.7% and 72%, respectively, compared with DSD.

According to the experimental results, when the horse animation model was simplified into 5000 triangles by using DSD, all simplified frame examples remained similar to the original frame examples. However, the shape features deteriorated gradually when the horse examples were simplified into 2000 triangles by using DSD. RHCT was used to determine a suitable vertex position for each simplified vertex pair and retain more shape features than DSD, does as shown in Figure 8.

Each original frame example in the camel model contained 43813 triangles and 21887 vertices. Figure 9 shows the original examples and the simplification results generated using DSD and RHCT. Table 2 and Figure 10 show comparisons of the errors produced by DSD and the proposed
Figure 8: The shape of horse hoof is destroyed gradually when simplifying by the DSD.

Figure 9: Simplification results of the camel animation model (frame examples 3, 23, and 43).
Figure 10: RMS error comparison for simplifying camel animation model.

(a) Simplifying by the DSD (2000 triangles)  
(b) Simplifying by the RHCT (2000 triangles)

Figure 11: The thigh shape of camel is destroyed after simplifying by the DSD.

Figure 12: The simplification results of the elephant animation model.

Table 2: RMS error comparison for simplifying camel animation model (unit: $10^{-3}$).

<table>
<thead>
<tr>
<th>Triangles</th>
<th>The DSD</th>
<th>The RHCT</th>
<th>Improvement rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>10000</td>
<td>1.852</td>
<td>0.494</td>
<td>73.3%</td>
</tr>
<tr>
<td>5000</td>
<td>6.243</td>
<td>0.845</td>
<td>86.5%</td>
</tr>
<tr>
<td>2000</td>
<td>9.599</td>
<td>1.834</td>
<td>80.9%</td>
</tr>
<tr>
<td>1000</td>
<td>15.715</td>
<td>3.294</td>
<td>79.0%</td>
</tr>
</tbody>
</table>

Table 3: RMS error comparison for simplifying elephant animation model (unit: $10^{-3}$).

<table>
<thead>
<tr>
<th>Triangles</th>
<th>The DSD</th>
<th>The RHCT</th>
<th>Improvement rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>10000</td>
<td>2.468</td>
<td>0.751</td>
<td>69.6%</td>
</tr>
<tr>
<td>5000</td>
<td>4.779</td>
<td>1.390</td>
<td>70.9%</td>
</tr>
<tr>
<td>2000</td>
<td>10.909</td>
<td>2.920</td>
<td>73.2%</td>
</tr>
<tr>
<td>1000</td>
<td>17.370</td>
<td>4.957</td>
<td>71.5%</td>
</tr>
</tbody>
</table>

In addition, when the camel animation model was simplified into 2000 triangles by using DSD, the outline of the camel leg deteriorated severely. By contrast, the simplified examples produced using the RHCT method retained their major shapes, as shown in Figure 11.

In the experiment, besides taking the horse and camel models for verification, the elephant animation model is taken for experimental testing. The original elephant model contained 84637 triangles and 42321 vertices. Figure 12 shows the simplification results. Table 3 and Figure 13 show a comparison of the error produced using DSD and the proposed method; when the elephant examples were simplified to 10000 triangles by using DSD, the mean RMS error was 0.002468. When the proposed method was used, the error was 0.000751, which is 69.6% lower than that of DSD. Moreover, when the model was simplified to 2000 triangles, the proposed method reduced the error by 73.2% compared with DSD.

Regarding the shape preservation of the simplified elephant example, the shape of the trunk deteriorated severely and even became hollow when the model was simplified to 5000 triangles by using the DSD method, as shown in Figure 14(a). By contrast, RHCT identified a suitable position for the new vertex after simplifying a vertex pair. Therefore, the outline of trunk was kept as intact as possible when it was simplified using the proposed method, as shown in Figure 14(b).

In addition, the progressive multiresolution meshes [24] mentioned in Section 2 also provided wonderful simplification results. This method reduced the local minimal simplification errors of the models similar to the first frame model. However, the local maximal simplification errors, which were
nearly equal to those generated by the DSD, could not be reduced. By contrast, RHCT reduced the considerable local maximal simplification errors, as shown in Figure 15.

Moreover, to compare the appearance changes in the simplified models, the experiment employed the tensor-based perceptual distance measure (TPDM) [33] and mesh structural distortion measure 2 (MSDM2) [34] to estimate perceived quality. Both measures were executed in MEPP software [35]. The obtained perceived quality values were between 0 and 1, tending toward 1 when the visualization of the measured objects differed and equaled 0 when they were identical. This experiment simplified the horse sequence...
to 5000 triangles and used TPDM and MSDM2 to measure the perception distortion, as shown in Tables 4 and 5 and Figure 16.

According to the results in Table 4, the average perception distortion caused by the simplification of the DSD was 0.14588, and the distortion was 0.12097 when the RHCT was employed, a value that was 17.07% smaller than that obtained using the DSD. In MSDM2, the DSD and RHCT produced 0.39796 and 0.33999 average perception distortions, respectively, when they were employed to simplify the horse model into 5000 triangles; in other words, the proposed method achieved improvements of 11.87% to 16.68% compared with the DSD.

6. Conclusions

The QEM method has recently been proven to be one of the most effective simplification methods. DSD uses summed QEM to simplify and reduce the storage cost of animation models. It then uses revised quadric matrices to resolve
Table 4: Perceived quality comparison for simplified horse animation models using TPDM.

<table>
<thead>
<tr>
<th>Frame</th>
<th>The DSD</th>
<th>The RHCT</th>
<th>Improvement rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.14417</td>
<td>0.11536</td>
<td>19.98%</td>
</tr>
<tr>
<td>2</td>
<td>0.14574</td>
<td>0.1169</td>
<td>19.79%</td>
</tr>
<tr>
<td>3</td>
<td>0.14756</td>
<td>0.11965</td>
<td>18.91%</td>
</tr>
<tr>
<td>4</td>
<td>0.14641</td>
<td>0.12315</td>
<td>15.89%</td>
</tr>
<tr>
<td>5</td>
<td>0.14726</td>
<td>0.12353</td>
<td>16.11%</td>
</tr>
<tr>
<td>6</td>
<td>0.14799</td>
<td>0.1246</td>
<td>15.81%</td>
</tr>
<tr>
<td>7</td>
<td>0.14695</td>
<td>0.12536</td>
<td>14.69%</td>
</tr>
<tr>
<td>8</td>
<td>0.14601</td>
<td>0.1232</td>
<td>15.62%</td>
</tr>
<tr>
<td>9</td>
<td>0.14547</td>
<td>0.1233</td>
<td>15.24%</td>
</tr>
<tr>
<td>10</td>
<td>0.14529</td>
<td>0.11978</td>
<td>17.56%</td>
</tr>
<tr>
<td>11</td>
<td>0.1431</td>
<td>0.11931</td>
<td>16.62%</td>
</tr>
<tr>
<td>12</td>
<td>0.14455</td>
<td>0.11749</td>
<td>18.72%</td>
</tr>
<tr>
<td>Average</td>
<td>0.14588</td>
<td>0.12097</td>
<td>17.07%</td>
</tr>
</tbody>
</table>

Table 5: Perceived quality comparison for simplified horse animation models using MSDM2.

<table>
<thead>
<tr>
<th>Frame</th>
<th>The DSD</th>
<th>The RHCT</th>
<th>Improvement rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.38888</td>
<td>0.32441</td>
<td>16.58%</td>
</tr>
<tr>
<td>2</td>
<td>0.39903</td>
<td>0.3395</td>
<td>14.92%</td>
</tr>
<tr>
<td>3</td>
<td>0.40074</td>
<td>0.34292</td>
<td>14.43%</td>
</tr>
<tr>
<td>4</td>
<td>0.40216</td>
<td>0.34614</td>
<td>13.93%</td>
</tr>
<tr>
<td>5</td>
<td>0.40539</td>
<td>0.35548</td>
<td>12.31%</td>
</tr>
<tr>
<td>6</td>
<td>0.40217</td>
<td>0.35443</td>
<td>11.87%</td>
</tr>
<tr>
<td>7</td>
<td>0.39941</td>
<td>0.34918</td>
<td>12.58%</td>
</tr>
<tr>
<td>8</td>
<td>0.40186</td>
<td>0.34418</td>
<td>14.35%</td>
</tr>
<tr>
<td>9</td>
<td>0.39619</td>
<td>0.33311</td>
<td>15.92%</td>
</tr>
<tr>
<td>10</td>
<td>0.39283</td>
<td>0.3273</td>
<td>16.68%</td>
</tr>
<tr>
<td>11</td>
<td>0.3927</td>
<td>0.32993</td>
<td>15.98%</td>
</tr>
<tr>
<td>12</td>
<td>0.39418</td>
<td>0.33327</td>
<td>15.45%</td>
</tr>
<tr>
<td>Average</td>
<td>0.39796</td>
<td>0.33999</td>
<td>14.57%</td>
</tr>
</tbody>
</table>

the problem of homogeneous coordinates. However, the revised quadric matrices cannot be used to estimate a suitable position for the contraction vertex formed by a vertex pair, leading to severe deterioration of the outlines of simplified animation models. To solve this problem, this paper proposes using RHCT to determine the appropriate position of the contraction vertex. Experimental results indicated that the proposed method can reduce the errors caused by simplification and enables the shape features of simplified animation to be retained.

Conflict of Interests

The author declares that there is no conflict of interests regarding the publication of this paper.
Acknowledgments

The author thanks the anonymous reviewers for helping to improve this paper. The author also thanks Professors Chassery, Wang, and Torkhani for providing the TPDM codes. Additionally, this research was supported by the National Science Council, Taiwan, under Grants NSC98-2221-E-159-019 and NSC100-2622-E-159-004-CC3.

References


Submit your manuscripts at http://www.hindawi.com