Laplacian Meshes Deformation Based on the Offset of Sketching

Sha Chenming
School of Software, Harbin University of Science and Technology, Harbin, China
Email: shachm@163.com

Zhang Xiaojing
School of Software, Harbin University of Science and Technology, Harbin, China
Email: xjz81@163.com

Yue Yajie
School of Software, Harbin University of Science and Technology, Harbin, China
Email: jielu1981@yahoo.com.cn

***Abstract*—Surface representation and processing is one of the key topics in computer graphics and geometric modeling, since it greatly affects the range of possible applications. In this paper, we propose a new Laplacian meshes deformation based-on the offset of sketching to solve the drawback that Laplacian coordinates are not invariant under rotation. First, we correct Laplacian coordinates rotation by the offset of sketching, and then generate the deformed models by solving linear system in least squares sense with Gauss-Seidel algorithm. In contrast with the traditional Laplacian deformation, our method can achieve natural deformed models by only using the boundary of ROI as constraints, which makes the manipulation more simplicity.**

***Index Terms*—Laplacian coordinates, mesh deformation, sketching**

I. Introduction

Meshes have become a widespread and popular representation of models in computer graphics. A fundamental modeling operation is mesh deformation which changes local or whole geometric details of the mesh as user wants.

Mesh deformation is useful in a variety of applications in computer modeling and animation. Many techniques have been developed to provide natural looking deformations. Manipulating and modifying a surface while preserving the geometric details is important for various surface editing operations, including free-form deformations, cut and paste, fusion, morphing, and others. Note that the absolute position of the vertices in a mesh is not important for these operations; what is needed is a local description of the shape which is not dependent on the particular placement of the shape in the Euclidean space. Laplacian coordinates proposed by Alexa[1] is one of the commonly used methods in descriptions of mesh details. The Laplacian of a graph is the difference of a vertex position from the centroid of its neighbors. Laplacian coordinates are a linear function of the global mesh geometry, which allows efficient converting between absolute and intrinsic representations by solving a sparse linear system.

Laplacian can preserve the intrinsic geometry of the original meshes surface during the deformation. And Laplacian coordinates are invariant under translation (of absolute geometry), but they are not invariant to scaling and rotation, which poses the main practical problem. Laplacian is changed under affine transformation, so it should be corrected according to the affine of vertices during the deformation.

Aimed at the drawbacks of Laplacian coordinates deformation, we pull sketch technology in Laplician coordinate deformation. Our method first recomputes the Laplacian coordinates of vertices in ROI (region of interest) by computing the rotation based on the sketch, and then reconstructs the mesh 3D coordinates to get the deformed models using the corrected Laplacian coordinates. This method's main steps are as follows:

1) Recognizing the region of interest. First select the start and end region of interest, then use the breadth-first search method, the starting point to start the search, until it reaches the end stop.

2) Building the reference curve. We begin with a free-form sketch of the reference curve in the image plane. Having regularized the reference curve in the image plane, we must project it into the 3-D world space of the model.

3) Determine the target curve. In the reference plane, the length of the reference line and reference line control point location, given the target curve.

4) Find the rotation. Within the region of interest each vertex, mapped to the reference according to the minimum distance line, the projection corresponding to the target curve at the point of rotation is the vertex in the reference plane rotation.

5) Generated the deformed model. Using the Gauss-Seidel algorithm to solve the linear equations in the least

squares sense. Then, reconstruct the deformed mesh models.

In contrast with the traditional Laplacian deformation, our method can achieve natural deformed models by only using the boundary of ROI as constraints, which makes the manipulation more simplicity.

II. Related Work

Editing three-dimensional shapes has been an important research area in geometric modeling and computer graphics. It is a challenging problem since a good editing tool should be intuitive and easy to use, and at the same time flexible and powerful. We are interested in editing an existing surface, probably acquired with scanning devices. If the surface is smooth, deformations should remain smooth. If the surface contains geometric details, these details should be preserved. Shapes that contain geometric details, like those acquired from real-world objects, require special editing tools to preserve the details. The editing operation should naturally change the shape and simultaneously respect the structural detail. The standard approach to detail-preserving modeling operations uses a multi-resolution representation of the mesh. It enables large-scale editing on a coarse level and naturally propagates modifications to the finer levels. The different levels can be considered as geometric frequencies or resolution of detail, where the coarsest level refers to a smooth surface. Roughly speaking, the editing modifies a coarse level, and the modified version of the next finer level is computed by “adding” the displacements. This is iterated over the hierarchy until the finest level of the detailed surface is reconstructed.

In this section we briefly overview mesh editing techniques for geometric modeling as they have evolved in the recent years.

Freeform deformation (FFD) is used in commercial software such as 3D Studio and Maya. A general treatment can be found in [2]. While energy minimization and FFD methods work well for smooth surfaces, multi-resolution editing is better suited for detailed geometry such as that acquired from scanning devices. A model is first decomposed into a smooth base shape and a set of geometric details, represented as displacements in a local coordinate frame. After modifying the base shape with some freeform deformation, the details can be re-inserted. The problem with these methods is that the displacement vectors are manipulated independently at each vertex. Artifacts can appear in highly deformed regions because details are not coupled and preserved uniformly over the whole surface.

The simplest form of differential coordinates is the Laplacian coordinates. The powerful properties of Laplacian coordinates for mesh representation are not new and have been exploited in various ways.

Alexa [1] shows that Laplacian coordinates can be effective for morphing and briefly discusses their potential for free-form modeling. He proposes to use differential coordinates to perform local morphing and deformation of the mesh, suggesting differential coordinates as a local mesh description, which would be more suitable to constrain under a global deformation of the mesh. This work also mentions the difficulty in using affine-invariant coordinates for mesh representation: the vertex neighborhood cannot always define a local frame (due to linear dependency), and thus the problem is numerically unstable.

Yu et al. [3] introduce an editing technique, formulated by manipulation of the gradients of the coordinate functions (x, y, z) defined on the mesh. The surface is reconstructed by solving the least-squares system resulting from discretizing the Poisson equation $\Delta f = g$ with Dirichlet boundary conditions. Lipman et al. [4] reconstruct the surface from discrete Laplacians of the mesh functions and spatial boundary conditions by solving a very similar least-squares system. Both works point out the main problem of this approach: the need to rotate the local frames that define the gradients, or the Laplacians, to preserve the orientation of the local details. They propose to remedy this problem by explicit assignment of the local rotations. Lipman et al. estimate the local rotations of the frames on the underlying smooth surface, and Yu et al. propagate the rotation of the editing handle, defined by the user, to all the vertices of the region of interest.

Sorkine et al. [5] suppose that each vertex only contains rotation part. They first compute the initial solution on the position of deformed mesh’s vertices, and then estimate rotation transformation of vertices during deformation which was used to correct Laplacian coordinate sequence of original mesh. In the end, they can get an optimal solution. This method can correct the effect taking by local rotation transformation, but the estimation is a heuristic search processing, and it has some drawbacks such as complex implementation, slow speed, especially repeat iterative in details abundant domains. For this purpose, Sorkine[6] proposed a new technology in 2004. They provide a technique that makes Laplacian coordinates invariant to rotation and isotropic scaling. Using this technique, they develop useful surface editing operations, which preserve the intrinsic geometry of the surface as much as possible given the constraints of the modeling operations. But it can only be used in small angle rotation and uniform scaling.

Zhou[7] extends the ideas mentioned above to the volumetric domain to solve the problem of large deformations. In this paper, they present a novel deformation technique that achieves convincing results for large deformations. It is based on the volumetric graph Laplacian, which represents volumetric details as the difference between each point in a 3D volume and the average of its neighboring points in a graph. They built two kinds of volumetric graphs: an inside graph fills the interior volume of the mesh and prevents large volume changes, while an outside graph prevents local self-intersection. The method improves the effect of mesh deformation, but the ability of correction in these methods is limited.

In 1996, Zeleznik[8] first proposed a technology called SKETCH, and then developed by Kho and Garland[9]. By using sketch technology, users sketch a reference curve in the image plane both to determine a region of interest and to serve as a means of controlling an individual deformation. By sketching a second curve indicating the desired deformation of the reference curve, users can easily achieve the deformation of the entire region of interest specified by the reference curve. By constructing a mapping of the region of interest onto the reference curve, it also provides a simple method for controlling additional parameters such as local twisting or scaling. Curves or skeletons in this type of methods can provide a natural means of capturing the structure of surfaces. Thus, deforming surfaces by editing those entities provides a good means of achieving large scale deformations. Sketch-based interfaces have emerged as one of the more popular approaches to building user-friendly deformation tools.

III. Laplacian Meshes Deformation

The algorithm of mesh deformation based-on Laplacian coordinates makes use of Laplacian instead of absolute Cartesian to express local geometric details.

Further, the representation of a vertex should be linear in the absolute coordinates. This is because both the transformation from absolute to relative coordinates, and vice versa, should be numerically stable to compute. A linear mapping leads to solution of linear systems for each of the transforms, which is, compared to other possibilities, the simplest possible solution. The relative representation aims at making the shape of the mesh invariant to translation or, ideally, invariant under affine transforms. If a vertex were represented in the affine space of its neighbors, invariance under affine transforms would trivially follow. The extension to triangulations in R^3 is difficult because vertices of the neighborhood are not necessarily affinely independent in R^3. We introduce a translation-invariant scheme and briefly elaborate on the affinely independent scheme in the followings

Let the mesh M be described by a pair (K, V), where K describes the connectivity and $V=\{v_1,v_2,\ldots,v_n\}$ describes the geometric positions of the vertices in R^3. We use the following terminology: the neighborhood ring of a vertex i is the set of adjacent vertices $N_i=\{j|(i,j)\in K\}$ and the degree d_i of this vertex is the number of elements in N_i. We assume that the mesh is connected.

Instead of using absolute coordinates V, we would like to describe the mesh geometry using a set of differentials $\Delta = \{\delta_i\}$. Specifically, coordinate i will be represented by the difference between v_i and the averages of its neighbors:

$$\boldsymbol{\delta}_i = \mathbf{v}_i - \frac{1}{d_i}\sum_{j\in N(i)} \mathbf{v}_j \qquad (1)$$

The transformation between V and Δ can be described in matrix algebra. Let A be the mesh adjacency matrix and D=diag($d_1,\ldots,d_n$) be the degree matrix. Then $\Delta=LV$, where $L=I-D^{-1}A$ is the transformation matrix. The matrix L is commonly considered as the Laplacian operator of the mesh with connectivity A. Laplacian coordinates are invariant under translation, but sensitive to linear transforms. L has rank n−1, which means V can be recovered from Δ by fixing one vertex and solving a linear system.

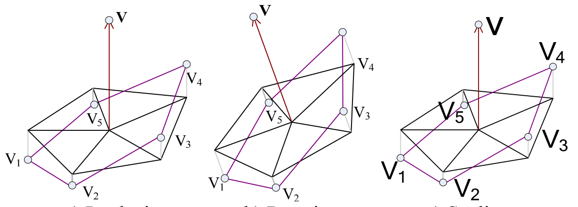
The approach to performing modeling operations using Laplacian coordinates Δ is to fix the absolute position of several vertices, i.e.,

$$\mathbf{v}'_i = \mathbf{u}_i, i=m,\ldots,n, \qquad (2)$$

and solve for the remaining vertices $\{ v'_i \}$, $i \in \{1, \ldots, m-1\}$ by fitting the Laplacian coordinates of the geometry $\boldsymbol{V}'$ to the given Laplacians Δ. It has been observed that the solution behaves better if the constraints $\{u_i\}$ are satisfied in a least squares sense rather than exactly. This results in the following error functional:

$$\min_{\boldsymbol{V}'}\left(\left\|\boldsymbol{L}\boldsymbol{V}'-\Delta\right\|^2 + \sum_{i=m}^{n}\left\|\mathbf{v}'_i - \mathbf{u}_i\right\|^2\right) \qquad (3)$$

Which has to be minimized to find a suitable set of coordinates $\boldsymbol{V}'$. Solving this quadratic minimization problem results in a sparse linear system of equations. The first item of the formula represents the requirement that minimize the loss of the original mesh Laplacian coordinates, and the second one represents the requirement that minimize the error of feature vertices. A compromise between the above items yields a natural mesh deformation effect.



a) Laplacian b) Rotation c) Scaling

Figure 1. Laplacian Transformations

The Laplacian of deformed mesh is approaching the one of original mesh, so it can preserve the geometric details of original mesh in some extent during the deformation. The rationale of fitting given Laplacian coordinates is that details of the shape are preserved, as the relative location of vertices is encoded in Δ. As mentioned, however, these coordinates are sensitive to linear transformations (see figure 1). Thus, the detail structure of the shape can be translated, but not rotated or scaled. If the constraints u_i imply a linear transform, the details are not transformed accordingly.

Because Laplacian coordinates change with the affine, the Δ of the original mesh should not be considered invariant, it should be considered changing with the mesh deformation and corrected according to affine transformation of vertices happened.

Assuming that each vertex contains only rotation part under affine transformation, first compute the initial solution of the vertex position on deformed mesh using above formula, then estimate the rotation transformation Ri of each vertex from the initial solution during the deformation, the original mesh to correct the coordinates of the Laplacian sequence to make up for the Laplacian coordinates transform from the impact of rotation, to be the Laplacian coordinates after deformation sequence of estimates, and then by solving the linear find the small least problem is further optimization of the second solution.

Although the method can transform the local rotation of the impact of certain mesh deformation correction, but the estimation on rotation transformation of the vertices is a heuristic search processing, and the estimation on vertex normal vector is high precision, complex, speed slow, especially in the region with rich geometric details to be iterative repeat. However, in addition to the vertex affine transformation occurs outside the rotating components should also consider scaling, shearing and translation elements, for which the literature [4] proposed to vertices v_i in the deformation process of local transformation is approximately a linear transformation T_i:

$$\mathbf{T}_i = \begin{bmatrix} s & -h_3 & h_2 & t_x \\ h_3 & s & -h_1 & t_y \\ -h_2 & h_1 & s & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

In the small angle rotation, the matrix T_i can be considered as an affine transformation of the actual good approximation. And correcting Laplacian coordinate δ_i with T_i, as T_i can be expressed as a linear transformation of the original mesh vertices v_i and deformation of the mesh to be seeking a linear function of the vertex, which can be carried by the following equation optimal solution:

$$\min_{V'\mathbf{T}_i}\left(\sum_{i=1}^{N}\left\|\mathrm{L}(v_i') - \mathbf{T}_i\delta_i\right\|^2 + \sum_{i=m}^{n}\left\|v_i' - u_i\right\|^2\right) \tag{4}$$

$$= \min_{V'}\left(\left\|\mathrm{L}V' - \mathbf{T}(V,V')\boldsymbol{\Delta}\right\|^2 + \sum_{i=m}^{n}\left\|v_i' - u_i\right\|^2\right)$$

The correction algorithm is only when the vertices of the local transformation contains only a small angle rotation and uniform scaling, can be enough to approximate linear transformation T_i, while the deformation process of rotation and a larger portion of non-uniform scaling transformation, the correction capability is still limited.

In order to reduce the mesh distortion occurs during deformation, in [5] proposed a method to avoid using voxel mesh size of the original amount of significant change in the method of Laplacian coordinate mesh to improve the effect of deformation. In this method, the original mesh model while adding a number of internal and external samples, additional sampling the original mesh vertices and the vertex connectivity in a three-dimensional map, and re-calculate the three-dimensional map coordinates of the vertices of the Laplacian. Accordingly, the mesh deformation is constrained, improving the mesh deformation effect.

In this paper, the drawbacks of the Laplacian method, we propose a new mesh deformation method based on the offset sketch using sketching lines offset to correct the Laplacian coordinates after deformation, to deformations purposes.

IV. Laplacian Meshes Deformation Based On The Offset Of Sketching

Manipulating and modifying a surface while preserving the geometric details is important for various surface editing operations.

By the formula (3) shows the mesh after deformation of the Laplacian approximation of the original mesh coordinates of the Laplacian sequence. But because the coordinates of Laplacian transform and change with the rotation, so the equation $\Delta = LV$ in the Laplacian representation of the original mesh coordinates Δ should not be considered invariant, should be considered with the mesh deformation and changes and be based on the vertex of the affine transformation happened to be amended.

This article assumes that the occurrence of each vertex contains only rotating components of affine transformation is proposed for the rotation transformation of the improved method based on the Laplacian sketching offset mesh deformation method by calculating the corresponding point of the partial sketching shift of the vertex on the mesh to get the amount of rotation R, then use the Gauss - Seidel algorithm to optimize the solution of the following equation:

$$\min_{\boldsymbol{V'}}\left(\left\|\boldsymbol{LV'} - \mathbf{R}(\boldsymbol{V},\boldsymbol{V'})\boldsymbol{\Delta}\right\|^2 + \sum_{i=m}^{n}\left\|\boldsymbol{v}_i' - \boldsymbol{u}_i\right\|^2\right) \tag{5}$$

This method's main steps are as follows:

1) Recognizing the Region of Interest. The region of interest is that part of the mesh to which the deformation will actually be applied. Determine the regional deformation that occurred in the region of interest. The underlying assumption of our system is that the region of interest is the part of the surface "covered" by the user's sketch curve. First select the start and end region of interest, then use the breadth-first search method, the starting point to start the search, until it reaches the end stop. See Figure 2. Green Point is the starting point, blue point that end, the red for the region of interest.

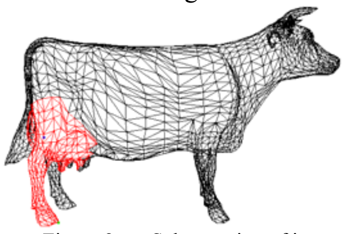


Figure 2. Select region of interest

2) Building the reference curve. We begin with a free-form sketch of the reference curve in the image plane. We represent the raw sketch curve as a collection of line segments taken directly from mouse events produced by the user's stroke. This raw curve is likely to be fairly noisy, especially when drawn with a mouse rather than a tablet device. Therefore, before proceeding, we smooth and regularize the raw sketch.

Having regularized the reference curve in the image plane, we must project it into the 3-D world space of the model. We first compute the point of intersection of a ray from the view point through the first point on the sketch curve. This hit point, along with the normal of the viewing plane, defines a plane in world space parallel to the image plane. We project the sketch curve onto this plane to compute the 3-D reference curve. (Blue line in Figure 3 below)

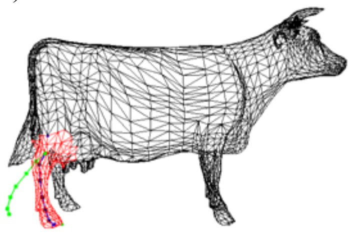


Figure 3. Illustrates the reference line

3) Determine the target curve. In the reference plane, the length of the reference line and reference line control point location, given the target curve (green line shown in Figure 3). In our sketch-based deformation, the end result is obviously controlled by the shape of the target curve. This allows the user to draw a fairly simple base target curve and then interactively adjust its shape to achieve a specific intended deformation.

4) Find the rotation. Within the region of interest each vertex, mapped to the reference according to the minimum distance line, the projection corresponding to the target curve at the point of rotation is the vertex in the reference plane rotation. Finally, gaining the view transform from rotation R.

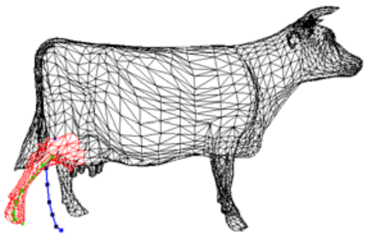


Figure 4. Based on sketching deformation

5) Generated the deformed model. Using the Gauss-Seidel algorithm to solve the linear equations in the least squares sense. Then, reconstruc (see equation (5)) the deformed mesh models (see Figure 4).

According to step 5 we can see from the formula, this approach focuses on Step 4, solve the rotation R. The following details the process of solving R.

The primary deformation that we support is accomplished by sketching. The user simply draws a new target curve, which we interpret as a deformation of the original reference curve. Our system then deforms the entire region of interest in an analogous way. This provides the user with intuitive and flexible control over the object's shape. Figure 1-4 show a simple example of this style of deformation.

A. Rotation Transformation

First, calculate the target point on the curve relative to the reference curve for each rotation angle $\theta(i)$ (Figure 5).

Secondly, the region of interest at every point, if all of its adjacent points within the region of interest, the point is internal point, if there is not a region of interest adjacent points; the point is defined as the boundary points. Each region of interest is mapped to the internal point of the first reference plane, and then finding the reference curve from the point nearest fold line, the fold line corresponding to the target curve is the point of rotation of the direction of the reference plane rotation.

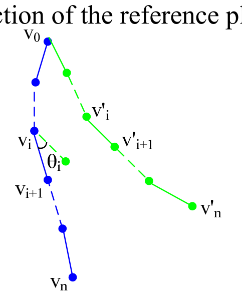


Figure 5. Blue reference curve, green target curve

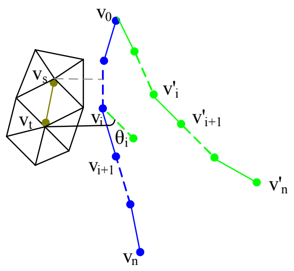


Figure 6. Smoothing

Then, smooth the amount of rotation on vertices in the region of interest. Shown in Figure 6, the mesh vertices v_s, v_t to adjacent points, the shortest distance from v_s to the reference point falling on the curve segment $<v_{i-1}, v_i>$, so the amount of rotation of the vertex v_s is $\theta(i-1)$, and the shortest distance from v_t to the reference point falling on the curve segment $<v_i, v_{i+1}>$, so the amount of rotation of the vertex v_t is $\theta(i)$. This situation makes the deformation have a big distortion at the point v_s, v_t. In this paper, we solve this problem by the following smoothing operations. For every point within the region of interest, taking the rotation point of rotation of adjacent vertices of the weighted values (see Equation 6).

$$R_i = \frac{1}{d_i} \sum_{j \in N(i)} R_j \qquad (6)$$

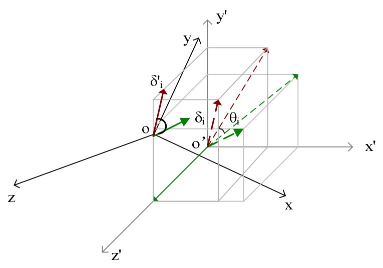


Figure 7. Coordinates transformation

Finally, find Laplacian coordinate $\boldsymbol{\delta}_i(x_i, y_i, z_i)$ of inside vertices within the region of interst, through the view transformation can get the coordinates $\boldsymbol{\delta}'_i(x'_i, y'_i, z'_i)$ of the current view, shown in Figure 7. Expressed (x'_i, y'_i) as polar coordinates $(\rho_i \cos\varphi_i, \rho_i \sin\varphi_i)$, to get the new coordinates after rotation $\theta(i)$ is:

$(\rho_i\cos(\varphi_i - \theta_i), \rho_i\sin(\varphi_i - \theta_i))$, at this time $\boldsymbol{\delta}'_i$ is represented as $(\rho_i\cos(\varphi_i - \theta_i), \rho_i\sin(\varphi_i - \theta_i), z'_i)$, through the inverse transform point $\mathbf{v}_i$ rotation revised by adding a new Laplace coordinates $\mathbf{R}_i\boldsymbol{\delta}_i$.

B. Mesh Reconstruction

In the following, we will be working with the above defined differential surface representation. The final step of any such manipulation must be surface reconstruction, or, in other words, we need to recover the Cartesian coordinates of *M*'s vertices.

In order to uniquely restore the global coordinates, we need to solve a full-rank linear system. Assuming *M* is connected, we need to specify the Cartesian coordinates of one vertex to resolve the translational degree of freedom. Substituting the coordinates of vertex *i* is equivalent to dropping the *i*th row and column from *L*, which makes the matrix invertible. However, as we shall see shortly, usually we place more than one constraint on spatial locations of M's vertices.

This article will point the boundary region of interest as a constraint; use the following Gauss - Seidel algorithm for solving minimum mean-square problem.

Known n-linear equations Ax = y:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = y_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = y_2 \\ \cdots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = y_n \end{cases} \quad (7)$$

Structural equations Ax = y Gauss - Seidel iterative algorithm as follows: first set $a_{ii} \neq 0$, i = 1,2, ..., n; the formula 7 in each equation $a_{ii}x_i$ stay in equation on the left, the rest of the equation are moved to the right; equation on both sides divided by a_{ii}, get the following equations with the solution:

$$\begin{cases} x_1 = -\dfrac{a_{12}}{a_{11}}x_2 - \ldots - \dfrac{a_{1n}}{a_{11}}x_n + \dfrac{y_1}{a_{11}} \\ x_2 = -\dfrac{a_{21}}{a_{22}}x_1 - \ldots - \dfrac{a_{2n}}{a_{22}}x_n + \dfrac{y_2}{a_{22}} \\ \cdots \\ x_n = -\dfrac{a_{n1}}{a_{nn}}x_1 - \ldots - \dfrac{a_{nn-1}}{a_{nn}}x_{n-1} + \dfrac{y_n}{a_{nn}} \end{cases} \quad (8)$$

Remember $b_{ij} =- a_{ij} / a_{ii}$, $g_i = y_i / a_{ii}$, more than the diagonal of the equations take the k-th iteration x_i values, the following take the first diagonal k +1 iteration values, structure the Gaussian - Seidel iteration form as:

$$\begin{cases} x_1^{k+1} = b_{12}x_2^k + b_{13}x_3^k + \ldots + b_{1n}x_n^k + g_1 \\ x_2^{k+1} = b_{21}x_1^{k+1} + b_{23}x_3^k + \ldots + b_{2n}x_n^k + g_2 \\ \cdots \\ x_n^{k+1} = b_{n1}x_1^{k+1} + b_{n2}x_2^{k+1} + \ldots + b_{n,n-1}x_{n-1}^{k+1} + g_n \end{cases}$$

(9)

Formula 9 shows that the Gauss - Seidel iteration as long as the formation of the matrix $B = (b_{ij}) = (-a_{ij}/a_{ii})$, in program make vectors $X_2 = (x_1^{k+1}, x_2^{k+1}, \ldots, x_n^{k+1})^T$, $X_1 = (x_1^k, x_2^k, \ldots, x_n^k)^T$. Its core part is to calculate iterative $X_2 = B * X_1 + g$, just in time to put x_i^{k+1} at the position of x_i^k. The core part of Gauss - Seidel iterative algorithm is as follows:

```
WHILE $||X_1-X_2||_\infty>10^{-6}$
        For (u:=1; u<=n; u++)
            $X_1$[u]= $X_2$[u];
        For (i:=1; i<=n; i++)
        {
            s:=g[i];
            for (j:=1; j<=n&&j!=I; j++)
                {
                    s:=s+b[i][j]* $X_2$[j];
                }
            $X_2$[i]:=s;
    }
```

The convergence of iterative algorithms is the assumption that the premise of using WHILE loop control structure. More generally, the algorithm can be changed in the WHILE loop observed by controlling the number of cycles and the calculation error of the loop ends.

In summary, the transition from the global Cartesian representation to differential representation is performed by a linear operator with local support: the mesh Laplacian. And vice versa: in order to recover the Cartesian coordinates from the differential representation, one needs to solve a sparse linear least-squares problem. This provides a powerful framework for surface manipulation: we will apply different modifications to the differential coordinates (such as quantization for compression purposes) and/or pose additional modeling constraints, and then reconstruct the surface by solving the least-squares problem.

V. Results

In this section, we consider several examples of using our system to edit unstructured polygon meshes. All results were generated by interactive editing on a standard consumer-level Windows PC. We render all sample models with flat shading in order to better highlight the structure of the surface mesh.

In order to verify the validity of this method, we proposed in this paper based on the Laplacian sketch offset mesh deformation method in the experiment. For 3D mesh models [10], first identified the region of interest, as shown in red below, and then give the reference curve and target curve, calculate the area of interest within the vertices of rotation, and finally use the Gauss - Seidel algorithm for reconstruction mesh.

Figures 8-11 demonstrate the results of our technique. In figure 8, we first select the right front leg of the Dinosaur as region of interest, then in the region of interest draw the reference curve, the reference curve project onto nearest surface of the three-dimensional mesh, then the target curve is given by the user, correct Laplacian coordinates rotation by the offset of sketching, and then generate the deformed models by solving linear system in least squares sense with Gauss-Seidel algorithm. Similarly, we have the rest of the Dinosaur application of our technology; at last, it can yield the deformed models that user needs. For example, we are here to simulate a walking Dinosaur.

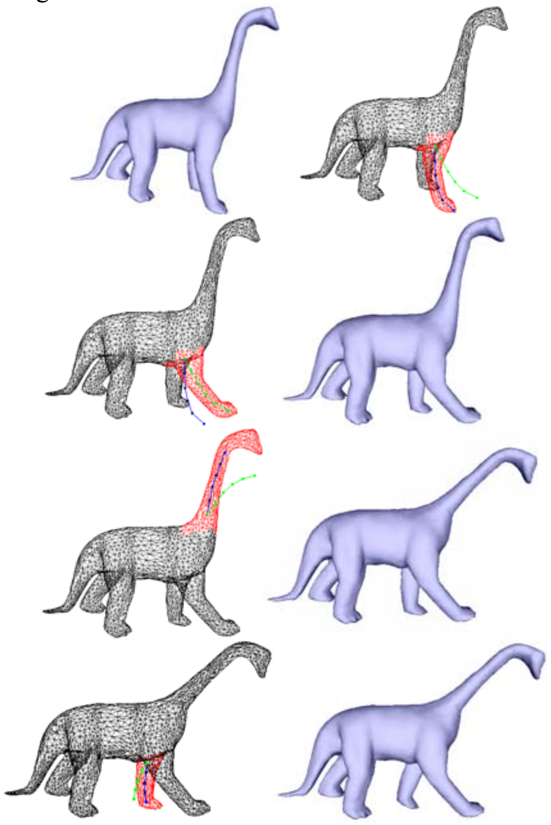


Figure 8. Dinosaur Deformation

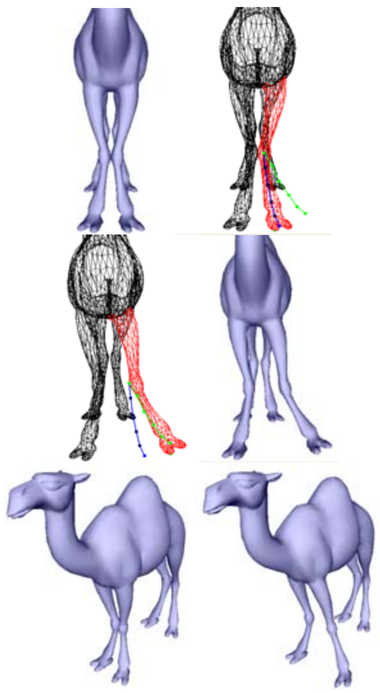


Figure 9. Camel Deformation

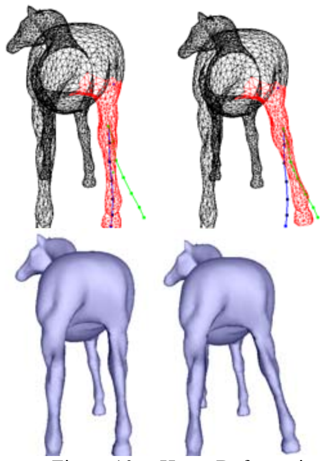


Figure 10. Horse Deformation

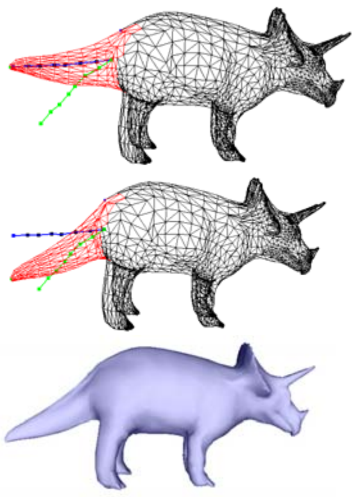


Figure 11. Triceratops Deformation

VI. CONCLUSIONS

Described in detail in this paper based on the coordinates of the deformation method of Laplacian, Laplacian method is the Laplacian coordinates of each point defined as the point coordinates and its adjacent point difference between the weighted vector, the method deformation problems Solving linear least into the problem, calculation and operation are very convenient, but because the Laplace coordinates with the affine and change, especially when the strain contains a greater degree of rotation, scaling ingredients, there will be some distortion.

For lack of the Laplace method, this paper presents a new grid based on the offset contour deformation method using contour lines offset to correct the Laplacian coordinates to achieve the purpose of deformation. First, determine the reference curve and target curve, calculated curve relative to the reference target curve rotation, rotation with the correct region of interest within the coordinates of each point in Laplace, by Gauss - Seidel iterative algorithm for solving linear least the second problem, get the reconstructed mesh. Application of this method can be a natural deformation effect, to avoid the Laplace transform coordinates in the rotation next problem.

REFERENCES

[1] Alexa.M, Differential coordinates for local mesh morphing and deformation, *The visual Computer*. 2003. 19(2). 105-114.

[2] Milliron.T, Jensen.R, Barzel.r, et al, A frame-work for geometric warps and deformations. *ACM Trans. Graphics 21, 1, 2002*. 20–51.

[3] Y.Yu, K.Zhou, D.Xu, et al, Mesh editing with Poisson-based gradient field manipulation, *ACM Transactions on Graphics (TOG), 2004, 23(3)*: 644-651.

[4] Lipman.Y, Sorkine.O, Levin.D, et al, Differential coordinates for interactive mesh editing, *In proceedings of shape modeling international, IEEE computer society press*, 2004. 181-190.

[5] Sorkine.O, Cohen-or.D, Toledo.S, High-pass quantization for mesh encoding, *In proceedings of the Eurographics/ ACMSIGGRAPH symposium on geometry processing*. 2003. 42-51.

[6] Sorkine.O, Cohen-or.D, Lipman.Y, et al, Laplacian surface editing, *In proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, 2004. 175-184.

[7] Zhou.K, Huang. P.J, Snyder. P.J, et al, Large mesh deformation using the volumetric graph laplacian, *In proceedings of ACM SIGGRAPH 2005*, 496-503.

[8] R.C.Zeleznik, K.P.Herndon, J.F.Hughes, SKETCH: an interface for sketching 3D scenes, *In proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 1996. 163-170.

[9] Y.Kho, M.Garland, Sketching mesh deformation, *In proceedings of the 2005 symposium on Interactive 3D graphics and games*, 2005. 147-154.

[10] http://shapes.aim-at-shape.net/viewmodels.php

Sha Chenming was born in Heilongjiang Province, China, in 1981 .She received the B.S. degree in Computer Science and Technology from Beijing Normal University, China in 2004, and M.S. degree in Computer application technology from Beijing University of Technology, China in 2007. Now she is working at Harbin University of Science and Technology, Harbin, China as a Lecturer.

Zhang Xiaojing received the B.S. degree in Communication Engineering from Northeast Dianli University in 2002. And she received the M.S. degree in Software Engineering from Yunnan University in 2005. Her primary research area focused on Embeded System. Currently she is a Lecturer of the School of Software, Harbin University of Science and Technology, Harbin, China.

Yue Yajie received the B.S. degree in automation from Harbin University of Science and Technology in 2004. And she received the M.S. degree in Control Theory and Control Engineering from Harbin University of Science and Technology in 2007. She has been working in Harbin University of Science and Technology from 2007. Currently her primary research area focused on IC design and Computer Architecture.