Eurographics Symposium on Geometry Processing (2005)
M. Desbrun, H. Pottmann (Editors)

Mesh Editing with Curvature Flow Laplacian

Oscar Kin-Chung Au[1†] Chiew-Lan Tai[1] Hongbo Fu[1] Ligang Liu[2]

[1]Hong Kong University of Science & Technology [2]Zhejiang University

1. Introduction

Differential coordinates are essentially vectors encoded in the global coordinate system. Since the local features on a mesh are deformed and rotated during editing, the differential coordinates must somehow be transformed to match the desired new orientations, otherwise distortion like shearing and stretching will occur. This transformation problem is basically a chicken-and-egg problem: the reconstruction of the deformed surface requires properly oriented differential coordinates, while the reorientation of these coordinates depend on the unknown deformed mesh. We present an iterative Laplacian-based editing framework to solve this transformation problem. The only user input required are the positions of the handles, not their local frames. Thus our system supports simple point handle editing. Our iterative updating process finds the best orientations of local features, including the orientations at the point handles.

2. Laplacian Editing

Let $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n)$ be the mesh vertex positions, and i^* be the index set of vertices adjacent to $\mathbf{v}_i$. The *Laplacian Coordinate* (LC) of a vertex $\mathbf{v}_i$ is $\mathbf{l}_i = \sum_{j \in i^*} w_{ij}(\mathbf{v}_j - \mathbf{v}_i)$, where w_{ij} is the weight of the edge (i, j) corresponding to vertex $\mathbf{v}_i$. In matrix form, it is $\mathbf{l} = \mathbf{LV}$, where $\mathbf{L}$ is an $n \times n$ matrix with elements derived from w_{ij}. We refer to these elements as the *Laplacian coefficients*. The basic idea of Laplacian editing is to find the positions $\mathbf{V}'$ of the deformed mesh by minimization, $\arg\min_{\mathbf{V}'} \left\|\mathbf{LV}' - \mathbf{l}\right\|^2$, constrained by the positions of some selected vertices as the handles of the model [S*04, L*04]. This is equivalent to solving a sparse linear system $\mathbf{AV}' = \mathbf{b}$ in least squares sense. Thus $\mathbf{V}'$ can be solved from the normal equations $\mathbf{A}^T\mathbf{AV}' = \mathbf{A}^T\mathbf{b}$.

Previous related methods cannot produce good results when the handles involve large angle rotation or are translated distantly from their original locations. Lipman et al. [L*04] used an intermediate reconstructed surface to guess the new orientations of the LCs. Sorkine et al. [S*04] employed implicitly defined transformations onto the LCs. However, it is untenable for large angle rotation and anisotropic scaling. Yu et al. [Y*04] solved the transformation problem by propagating the transformations of handles to all vertices. Lipman et al. [L*05] encode the vertex differences in local frames and minimize the least squares error of the changes in the local frames. Since both approaches need the transformations (or local frames) of the handles as input, if the handles only undergo translation, there is no transformation change to be propagated or minimized (see Figure 1); thus these approaches cannot avoid shearing and stretching distortion caused by handle translation.

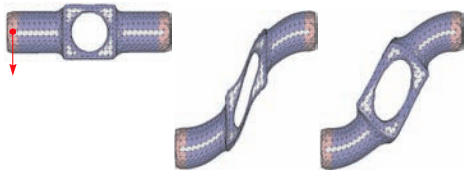

Figure 1: *Result of previous methods if a handle is only translated, not rotated (middle). Our method deforms the local features naturally based on only the position (not transformation) of the handle (right).*

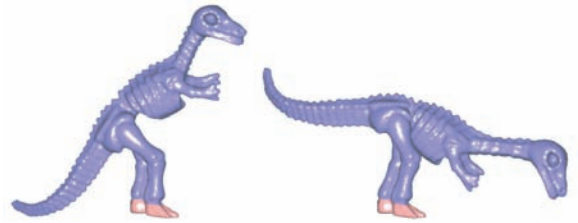

Figure 2: *(left) Input model with handles at the feet, nose tip, and tail end. (right) Editing by moving the point handles at the nose tip and tail.*

3. Curvature Flow Laplacian Editing

We observe that, to reduce distortion, the edited mesh should retain (1) parameterization information (shapes of triangles); (2) geometry information (sizes of local features). To separate the two types of information, we adopt the curvature flow Laplace operator: the edge (i, j) has weight $w_{ij} = \cot\alpha_{ij} + \cot\beta_{ij}$, where α_{ij} and β_{ij} are the two angles opposite the edge. Now the LC is an approximation of the integrated mean curvature normal at $\mathbf{v}_i$: $\mathbf{l}_i = \sum_{j \in i^*} w_{ij}(\mathbf{v}_j - \mathbf{v}_i) = 4\,Area_i\kappa_i\mathbf{n}_i$, where $Area_i$ is the one ring triangles area of vertex $\mathbf{v}_i$, and κ_i and $\mathbf{n}_i$ are the mean curvature and unit normal vector at $\mathbf{v}_i$, respectively.

We consider the local parameterization information as

captured by the Laplacian coefficients, and the geometry information as encoded by the magnitudes of the LCs. As the LCs are in the directions of the vertex normals, we regard the normals as not containing any local information (computable on the fly). Thus, our approach tries to keep the magnitudes of LCs and the Laplacian coefficients similar before and after editing. Since both sets of information generally depend on the vertex positions nonlinearly, we propose an iterative updating method that improves the vertex positions $\mathbf{v}_i$ and the LC $\mathbf{l}_i$ in each iteration, minimizing parameterization and geometry distortions progressively.

Algorithm. Let $\mathbf{v}_i^t$ and $\mathbf{l}_i^t$ be the vertex positions and the LCs at time t, respectively, and let $\mathbf{v}_i^0 = \mathbf{v}_i$ and $\mathbf{l}_i^0 = \mathbf{l}_i$.

Step 1. Update the vertex positions
We use the current $\mathbf{l}_i^t$ to compute the vertex positions $\mathbf{v}_i^{t+1}$; that is, we fix the LCs and solve the normal equations with the current handle positions as constraints.

Step 2. Update the Laplacian coordinates
We update the LCs to match the current deformed surface; that is, we fix the vertex positions $\mathbf{v}_i^{t+1}$ and compute the new LCs $\mathbf{l}_i^{t+1}$. We use the mean curvature normals computed from the current vertex positions $\mathbf{v}_i^{t+1}$ as the values of $\mathbf{l}_i^{t+1}$, but scale them to have the magnitudes of the original $\mathbf{l}_i^0$, in order to keep the original feature sizes.

During editing, the 1-ring structure of a vertex $\mathbf{v}_i$ may change between being convex and concave, making the computed curvature flow normal of $\mathbf{v}_i$ flip undesirably inward or outward. Naturally, we require the LC vector at each vertex to consistently point inward (outward) if the original LC points inward (outward). We compare the orientations of the original LC $\mathbf{l}_i^0$ and the current LC $\mathbf{l}_i^{t+1}$, and reflect $\mathbf{l}_i^{t+1}$ about the tangent plane if necessary.

We iterate the two updating procedures until the vertex positions converge. When that happens, $\mathbf{L}^t$ and $\mathbf{L}^0$ tend to be similar, thus retaining the original parameterization information. Since the magnitudes of the LCs are maintained, the local feature sizes are also retained. Figure 2 shows a deformation example with well oriented local features.

Rescaling LCs. Stretching or squashing distortion may occur when the distances between handles (thus dihedral angles) are changed drastically. Merely reorienting the LCs, while retaining their magnitudes, cannot give small parameterization distortion. Rescaling the LCs (modifying the feature sizes) to maintain the dihedral angles can produce more natural results with less parameterization distortion. Since the LCs are linear combinations of vertex positions, both

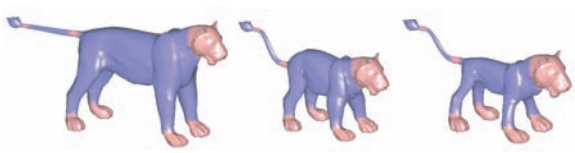

Figure 3: *A baby lion (right) cloned from her mother (left). Deformed model with (right) and without (middle) rescaling the Laplacian coordinates.*

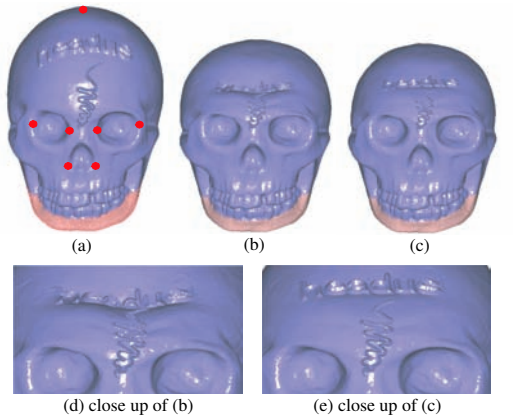

Figure 4: *Editing the skull (a) by moving the topmost handle. Deformed mesh without (b) and with (c) rescaling the LCs.*

have the same scaling factors under isotropic scaling. We provide the option to rescale each LC to have its magnitude equals the average edge length at the vertex. Figure 3 shows an example where the global feature is too big if the LCs are not rescaled. Figure 4 demonstrates that rescaling the LCs can eliminate undesired distortion, which is dependent on the geometry complexity and thus is difficult for user to design a scaling field for the LCs [Y*04].

Other applications. Since the magnitudes of the LCs approximate the integrated mean curvatures, our framework is useful for modifying mesh geometry via updating the curvature field, for example, assigning a constant curvature to obtain a spherical mapping or smoothing the curvature field to achieve non-shrinking smoothing (Figure 5).

Figure 5: *Updating the geometry via modifying curvature field. (left) Input model, (middle) non-shrinking smoothing, (right) spherical mapping.*

References

[L*04] LIPMAN Y., ET AL.: Differential coordinates for interactive mesh editing. In *Proceedings of Shape Modeling International* (2004), pp. 181–190. 1

[L*05] LIPMAN Y., ET AL.: Linear rotation-invariant coordinates for meshes. *ACM Trans. Graph. 24* (2005). 1

[S*04] SORKINE O., ET AL.: Laplacian surface editing. In *SGP* (2004), pp. 179–188. 1

[Y*04] YU Y., ET AL.: Mesh editing with Poisson-based gradient field manipulation. *ACM Trans. Graph. 23*, 3 (2004), 644–651. 1, 2