Advancing Front Method in Triangular Meshes Hole-Filling Application

Xiaochao Wang, Junjie Cao, Xiuping Liu
School of Mathematical Sciences
Dalian University of Technology
Dalian, China
wangxiaochao18@gmail.com

Baojun Li
School of Automotive Engineering
Faculty of Vehicle Engineering and Mechanics
Dalian University of Technology
Dalian, China

NOTE: In this text, the modified advancing front method (MAFM)is presented, which is the core part of the article. (X. Wang, J. Cao, X. Liu, B. Li, Advancing front method in triangluar meshes hole-filling application, Journal of Computer Aided Design and Computer Graphics 23 (2011) 1048–1054. In Chinese). Here, only the MAFM is introduced, the experiments and results can be found in above paper, which can be download form http://www.jjcao.net/publications.html.

Abstract

In order to restore the missing shape of the hole in triangular meshes, especially big ones locating at high curved region, a hole-filling algorithm based on advancing front method is proposed in this paper. After detecting the boundary of the hole, normals of the boundary vertices are well estimated. Combining with the Laplacian coordinate, the boundary vertices are classified into two types: concave and convex. Then, based on the normal, the concavity-convexity feature of each boundary vertex and a proper adjustment parameter, optimal vertices are carefully computed and new triangles are created to fill holes. Many experimental results show that our method has powerful ability to recover the missing shape with high quality triangular mesh for even big holes located at the high curved regions. Without post-processing, such as refinement and smoothing, the hole-filing meshes obtained by our method interpolates the shape and have consistent mesh distribution and smooth transition with the surrounding meshes.

1. Hole-filling

Modified advancing front method In this section, we adopt and modify the simple and robust AFM [1] to fill each hole by creating triangles on the boundary front. The main steps of the MAFM are as follows:

Step 1. Using the boundary vertices, say $\{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k\}$, of each hole to initial the front $\mathbf{F}$, i.e., $\mathbf{F} = \{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k\}$.

Step 2. Calculate the angle $\theta_i = \angle \mathbf{v}_{i-1}\mathbf{v}_i\mathbf{v}_{i+1}(\mathbf{v}_0 = \mathbf{v}_k, \mathbf{v}_{k+1} = \mathbf{v}_1)$ at each vertex $\mathbf{v}_i, 1 \le i \le k$, of the front and denote by θ_{min} the smallest one among all θ_i.

Step 3. Select a boundary vertex as the current front vertex by the **Min-Random rule** and create new triangles based on the rules shown in Fig.1. Produce new vertices, computed by the rule of **New vertex computing**, as the candidates of the front $\mathbf{F}$.

Step 4. For each candidate vertex, compute the distances between this vertex and every vertices of $\mathbf{F}$ and insert this vertex into $\mathbf{F}$ if the smallest distance is bigger than a given threshold.

Step 5. Update the front $\mathbf{F}$.

Step 6. Repeat Step 2 to Step 5 until the hole is filled.

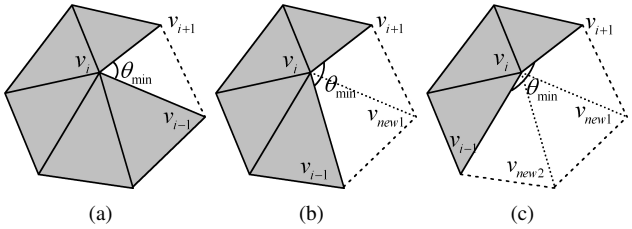

Figure 1: Rules of creating triangles. (a) $0 < \theta_{min} \le A$, one new triangle is created, but no new vertex is inserted; (b) $A < \theta_{min} \le B$, two new triangles are created and one new vertex is inserted; (c) $B < \theta_{min} \le \pi$, three new triangles are created and two new vertices are inserted. A and B are two threshold angles with default values $A = 85°$ and $B = 135°$.

1.1. The Min-Random rule

If $0 < \theta_{min} \le A$, the vertex corresponding to θ_{min} is selected to be current vertex, called minimum angle rule. If more than one vertices with the angle θ_{min}, anyone of them can be chosen as the current vertex. If $A < \theta_{min} \le \pi$, it is better to chose the current vertex randomly, called random rule. In this way, each boundary vertex has a chance to be selected as active vertex, which will result in better result.

1.2. Normal estimation

The normal of a vertex is usually computed by the weighted average of its 1-ring triangles' normals. In fact, it is only reasonable for an inner vertex. For a boundary vertex, it may leads to unexpected result, since there are no

completed 1-ring triangles, especially for the case of large 1-ring missing (see Fig.2(c)). It is clear that any boundary vertex of a hole should be an inner vertex of the input mesh if it is not damaged. Based on this observation, we construct complete 1-ring triangles to produce a more reasonable normal estimation.

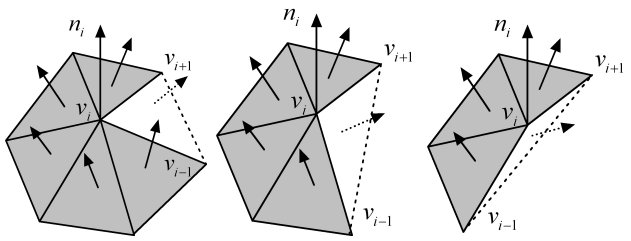

Figure 2: Normal estimation.

In Fig.2, $\mathbf{v}_i$ is the current boundary vertex, its descendent is $\mathbf{v}_{i+1}$, and its precursor is $\mathbf{v}_{i-1}$. In case (a), $0 < \theta_i \leq A$, the triangle $\triangle \mathbf{v}_{i+1}\mathbf{v}_i\mathbf{v}_{i-1}$ is added to $N_f(\mathbf{v}_i)$ and treated the same as other triangles of $N_f(\mathbf{v}_i)$ to calculate the normal of $\mathbf{v}_i$. In case (b), $A < \theta_i \leq B$, the $\triangle \mathbf{v}_{i+1}\mathbf{v}_i\mathbf{v}_{i-1}$ is also added to $N_f(\mathbf{v}_i)$ to calculate the normal of $\mathbf{v}_i$, but the weight of the normal of the the triangle $\triangle \mathbf{v}_{i+1}\mathbf{v}_i\mathbf{v}_{i-1}$ is scaled by A/θ_i. In case (c), $B < \theta_i \leq \pi$, since the missing of 1-ring triangles of $\mathbf{v}_i$ is too large, only two triangles respectively determined by $[\mathbf{v}_i, \mathbf{v}_{i+1}]$ and $[\mathbf{v}_{i-1}, \mathbf{v}_i]$ are used to calculate the normal of $\mathbf{v}_i$.

1.3. Convex/Concave analysis

Convexity/concavity is an important geometric property, which reflects the local shape of the models. The convexity/concavity of a vertex can be determined by its normal $\mathbf{n}_i$ and Laplacian coordinate $\boldsymbol{\delta}_i$ as follows (Fig.3(a))

$$Bool(\mathbf{v}_i) = \begin{cases} convex & \text{if} \quad \mathbf{n}_i \cdot \boldsymbol{\delta}_i \leq 0, \\ concave & \text{if} \quad \mathbf{n}_i \cdot \boldsymbol{\delta}_i > 0, \end{cases} \tag{1}$$

where ([2])

$$\boldsymbol{\delta}_i = \sum_{\mathbf{v}_j \in N(\mathbf{v}_i)} w_{ij}\mathbf{v}_j - \mathbf{v}_i, \tag{2}$$

with $w_{ij} = \omega_{ij} / \sum_{\mathbf{v}_j \in N(\mathbf{v}_i)} \omega_{ij}$ and ω_{ij} is the weight assigned to $\mathbf{v}_j$, with $\omega_{ij} = 1$ as the default value.

1.4. New vertex computing

Based on the analysis and preparations presented above, in this part, how to obtain the new vertex is introduced. First, we consider inserting one new vertex. Let the unit vector $\mathbf{v}_b$ be the bisector of $\angle \mathbf{v}_{i+1}\mathbf{v}_i\mathbf{v}_{i-1}$, Π be the tangent plane of $\mathbf{v}_i$, and π_i be the plane determined by $\mathbf{v}_i$, $\mathbf{n}_i$ and $\mathbf{v}_b$. All notations mentioned above are illustrated in Fig.3(b). The position of the new inserted vertex is computed as follows:

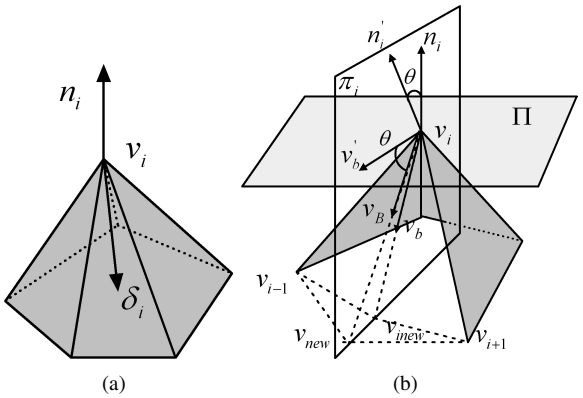

Figure 3: (a) The Laplacian coordinate; (b) New vertex computation.

Step 1. Set $\mathbf{v}_b^{''} = \mathbf{v}_b - (\mathbf{v}_b \cdot \mathbf{n}_i)\mathbf{n}_i$ and $\mathbf{v}_b^{'} = \mathbf{v}_b^{''}/\|\mathbf{v}_b^{''}\|$, where $\mathbf{v}_b \cdot \mathbf{n}_i$ is the inner product of $\mathbf{v}_b$ and $\mathbf{n}_i$. In fact, $\mathbf{v}_b^{'}$ is the vector obtained by rotating $\mathbf{v}_b$ around $\mathbf{v}_i$ on π_i to Π.

Step 2. Adjust $\mathbf{n}_i$ to $\mathbf{n}_i^{'}$, still locating on π_i, by setting $\mathbf{n}_i^{'} = \mathbf{n}_i + \alpha|\mathbf{n}_i \cdot \mathbf{v}_b|\mathbf{v}_b$, and $\mathbf{n}_i^{'} = \mathbf{n}_i^{'}/\|\mathbf{n}_i^{'}\|$. Here, α is a parameter to control the adjustment capability, ranging from 0 to 1 with default value 0.45.

Step 3. Denote by $\theta = \arccos(\mathbf{n}_i \cdot \mathbf{n}_i^{'})$ and $k = (\cos\theta - 1)/\mathbf{n}_i^{'} \cdot \mathbf{v}_b^{'}$. Then, the new inserting direction is set as $\mathbf{v}_B = (\mathbf{v}_b^{'} + k\mathbf{n}_i^{'})/\|\mathbf{v}_b^{'} + k\mathbf{n}_i^{'}\|$ if $\mathbf{v}_i$ is a convex vertex; $\mathbf{v}_B = (\mathbf{v}_b^{'} - k\mathbf{n}_i^{'})/\|\mathbf{v}_b^{'} - k\mathbf{n}_i^{'}\|$ if $\mathbf{v}_i$ is a concave vertex. In fact, $\mathbf{v}_B$ is obtained by rotating $\mathbf{v}_b^{'}$ on π_i by the angle θ away from or close to $\mathbf{n}_i^{'}$, respectively, depending on the convexity or concavity of $\mathbf{v}_i$.

Step 4. The new inserting vertex is computed by

$$\mathbf{v}_{new} = \mathbf{v}_i + \frac{\|\mathbf{v}^1 + \mathbf{v}^2\|}{2} \cdot \mathbf{v}_B,$$

where $\mathbf{v}^1 = \mathbf{v}_{i-1} - \mathbf{v}_i$ and $\mathbf{v}^2 = \mathbf{v}_{i+1} - \mathbf{v}_i$.

The case of inserting two new vertices is similar to inserting one new vertex. However, for the larger missing, in order to obtain pleasing results, there is a slightly difference in **Step 2**. Denote the trisection unit vectors of $\angle \mathbf{v}_{i+1}\mathbf{v}_i\mathbf{v}_{i-1}$ are $\mathbf{v}_{b1}$ and $\mathbf{v}_{b2}$. The new normal $\mathbf{n}_i^{'}$ is obtained by using $\mathbf{v}_b$ in $\mathbf{n}_i^{'} = \mathbf{n}_i + \alpha|\mathbf{n}_i \cdot \mathbf{v}_b|\mathbf{v}_b$, instead of $\mathbf{v}_{b1}$ and $\mathbf{v}_{b2}$ respectively. And use the same θ to compute the two new inserting directions $\mathbf{v}_{B1}$ and $\mathbf{v}_{B1}$ in **Step 3**. The final inserting vertices can be obtained by

$$\mathbf{v}_{new\,j} = \mathbf{v}_i + \frac{\|\mathbf{v}^1 + \mathbf{v}^2\|}{2} \cdot \mathbf{v}_{Bj},\ j \in \{1, 2\}.$$

References

[1] S. E. Georage, L.G, The advancing-front mesh generation revisited, J.Numer. Methods Eng 37 (7) (1994) 3605–3619.

[2] A. Nealen, T. Igarashi, O. Sorkine, M. Alexa, Laplacian mesh optimization, in: GRAPHITE '06, 2006, pp. 381–389.