MATHEMATICAL ENGINEERING
TECHNICAL REPORTS

Improvement of the Method for Making Quad Meshes through Temperature Contours

Minori OKABE, Shinji IMAHORI
and Kokichi SUGIHARA

METR 2009–05 March 2009

DEPARTMENT OF MATHEMATICAL INFORMATICS
GRADUATE SCHOOL OF INFORMATION SCIENCE AND TECHNOLOGY
THE UNIVERSITY OF TOKYO
BUNKYO-KU, TOKYO 113-8656, JAPAN

WWW page: http://www.keisu.t.u-tokyo.ac.jp/research/techrep/index.html

Improvement of the Method for Making Quad Meshes through Temperature Contours

Minori Okabe* Shinji Imahori* Kokichi Sugihara*

March 12, 2009

Abstract

There are several methods to reconstruct quad meshes from triangular meshes. In this article, we improve the method using temperature contours. We discuss how to avoid boundary conditions being ill posed and assure injectivity of the parameterization by controlling the weights on edges. The proposed method will be one step toward a fully automatic system, because the result of automatic partition of the surface into patches can be used.

1 Introduction

Triangular meshes are often used for representing surfaces of objects in computer graphics and simulation. Whereas quad meshes have the advantage of triangular meshes in many applications such as texture mapping in computer graphics, FEM in simulation and NURBS in modeling.

Constructing quad meshes directly from point clouds is difficult because we do not have efficient tools such as Voronoi diagrams in case of creating triangular meshes. Thus most of methods reconstruct triangular meshes into quad meshes.

One major trend is following some vector fields on the mesh. Principal directions are often chosen as the vector fields. One approach with vector fields is tracing them by numeric integration [1]. Another approach is defining functions on the mesh whose isolines follow the vector fields [7, 6]. This approach needs solving non-linear optimization instead of numeric integration. Another trend is defining functions on the given triangular mesh by parameterization [3, 8], which only needs linear computation and gives pure quad meshes.

We propose a new method to generate quad meshes, which is based on and improves the method by tong et al. [8]. In this method, quad meshes

*Graduate School of Information Science and Technology, The University of Tokyo, {minori_okabe, imahori, sugihara} @mist.i.u-tokyo.ac.jp

are obtained through temperature contouring. It is simple and efficient in calculation, but not enough from a practical viewpoint. We discuss the feasibility of an optimization problem appearing in the algorithm and assure injectivity of the parameterization by controlling the weights on edges.

2 Outline of the proposed method

We describe the outline of our algorithm, which is similar to that of tong's method [8].

To achieve quad meshes, we give each vertex of the given triangular mesh u and v values, and draw integer contour lines from them. Assigning two values is equivalent to planar parameterization. Because the given mesh is not homeomorphic to a disk in general, we have to divide it into some patches homeomorphic to disks. However, if patches are mapped into the plane independently, the continuity of contours is not assured, which is a requirement for quad meshes. For this purpose, some conditions of "jump" and "rotation" on patch boundaries are necessary. In case that the function value of a continuous contour line changes across a patch boundary, it should jump by a certain integer value. Moreover, in case that a contour line changes its function and direction, that is, it switches from u to v, u to $-u$ and so on, the rotating angle of the u-v coordinates should be equal to an integral multiple of $\frac{\pi}{2}$.

Now, the outline of our method is as follows:

1. Divide the mesh into patches.
2. Decide the connectivity conditions between patches — jump values and rotating angles.
3. Solve a linear system to obtain u and v values for each vertex and draw contours.

Hereafter, we call the mesh derived from the division of the original mesh into patches *meta-mesh*, and vertices, edges and facets of the meta-mesh *meta-vertices*, *meta-edges* and *meta-facets* respectively. A vertex meeting three or more patches corresponds to a meta-vertex. To make pure quad meshes, whose facets have just four vertices, u and v values of meta-vertices should be integer in each adjacent patches. We note that meta-vertices are the candidates of singular vertices of the resulting quad meshes, where singular vertices mean vertices adjacent to other than four facets.

3 Details of each procedure

3.1 Division into patches

The first procedure is to divide the original triangular mesh into patches homeomorphic to disks. As this division may affect the resulting quad

meshes, it should reflect some geometric features of the mesh. Here we use VSA method [2], in which each divided region is designed to approximate the plane through alternate update of the approximate planes and the patch division.

In VSA method, the patches tend to meet where the curvature is relatively large. This is a desirable property for quad meshes. Moreover, we can roughly control the number of singular vertices by adjusting the number of patches.

It is to be noted that some inputs from VSA method are rejected since some patches are not homeomorphic to disks. It is because VSA method guarantee each patch to be connected, but not to be homeomorphic to a disk. To resolve this problem, patches should be cut into subpatches homeomorphic to disks.

3.2 Solving a linear system

We explain the last procedure before the second one for understanding. Similar to the ordinary planar parameterization, u and v values of a vertex is set to the weighted average of u and v values of the neighboring vertices:

$$\sum_{j \in n_i} w_{ij} \begin{pmatrix} u_i - u_j \\ v_i - v_j \end{pmatrix} = 0,$$

where n_i is the set of the neighboring vertices of vertex i. Here, we use the mean value coordinates [5] as weights w_{ij} (see Section 3.4).

But for vertices on boundaries between patches, this formula is not sufficient because we admit jumps on u and v values and rotation of the u-v coordinates. Let vertex i be on the patch named '$-$', and patch '$+$' be one of adjacent patches of patch '$-$'. n_i^+ (resp., n_i^-) means the partial set of n_i on patch '$+$' (resp., '$-$').

For example, if u values differ by p_1 from one patch to another and v by p_2, the condition is as follows:

$$\sum_{j \in n_i^-} w_{ij} \begin{pmatrix} u_i - u_j \\ v_i - v_j \end{pmatrix} + \sum_{j \in n_i^+} w_{ij} \begin{pmatrix} u_i - (u_j + p_1) \\ v_i - (v_j + p_2) \end{pmatrix} = 0.$$

In case of rotation, when the contours of u switched into contours of v and v to $-u$ and the jump values are r_1 and r_2, the condition can be written down as follows:

$$\sum_{j \in n_i^-} w_{ij} \begin{pmatrix} u_i - u_j \\ v_i - v_j \end{pmatrix} + \sum_{j \in n_i^+} w_{ij} \begin{pmatrix} u_i + (v_j - r_1) \\ v_i - (u_j + r_2) \end{pmatrix} = 0.$$

Conditions for other rotations (i.e., 180° and 270°) can be written down similarly.

u and v values of meta-vertices are given in advance by the second procedure. This is corresponding to boundary conditions of planar parameterization. Injectivity of this mapping is referred later in Section 3.4.

3.3 Deciding parameters

We decide the jump values and rotating angles for meta-edges and u and v values of meta-vertices. Once the region on the u-v plane to which each patch is mapped is assigned, the connectivity of patches is automatically derived.

When two meta-vertices are adjacent on the meta-mesh, they should be connected by a contour line. That is, each meta-vertex is on a lattice point of the u-v plane and each meta-facet is mapped into a polygon with right angles on the u-v plane. By this condition, determining the shape results in simple calculations. In addition to the method in [8], we restrict the shape to a rectangle in order to avoid self intersection and assure injectivity. Thus what should be done is to decide the angles of meta-vertices on each patch and the lengths of meta-edges.

The angles for each patch is set to minimize the difference of the length of facing edges. Here, we use the shortest path length on the original mesh instead of the actual path length of the boundaries in order to make quad meshes equilateral.

Next we determine the length of each meta-edge. For each facet, the sums of the lengths of facing meta-edges must be equal. This can be represented as a linear condition $A\boldsymbol{x} = 0$, where $\boldsymbol{x}$ is the variants vector corresponding to the lengths of meta-edges. We consider minimizing the difference from the shortest path length of corresponding two vertices on original mesh represented by $\boldsymbol{c}$:

$$\begin{aligned} &\min \|\boldsymbol{x} - \boldsymbol{c}\|, \\ &\text{s.t. } A\boldsymbol{x} = 0, \\ &\quad \boldsymbol{x} \in \mathbb{Z}_+^n, \end{aligned}$$

where n is the number of meta-edges and $\mathbb{Z}_+$ is the set of positive intergers. By using 1-norm, this problem can be represented as a linear programming problem with integer conditions:

$$\begin{aligned} &\min t_1 + t_2 + \cdots + t_n, \\ &\text{s.t. } A\boldsymbol{x} = 0, \\ &\quad -t_i \le c_i - x_i \le t_i \ (i \in \{1, \ldots, n\}), \\ &\quad \boldsymbol{x} \in \mathbb{Z}_+^n, \\ &\quad t_i \ge 0 \ (i \in \{1, \ldots, n\}). \end{aligned} \tag{$*$}$$

The feasibility of this problem is discussed later in Section 5.

If the shape of each patch is obtained through the angles of meta-vertices and the lengths of meta-edges, u and v values of meta-vertices are specified by fixing arbitrarily u and v values for one meta-vertex and the direction of one meta-edge. After doing this for each patch, jump values and rotating angles between patches are automatically obtained.

3.4 Guarantee for injectivity

Since injectivity is not guaranteed, triangles may flip, which results in no pure quad meshes. In addition, the triangles around singular vertices have a tendency to flip and this hurts the quality of the resulting quad mesh.

In ordinary planar parameterization, a theorem in [5] is known to guarantee the injectivity. From this theorem, a mapping is guaranteed to be injective if (1) a triangular mesh homeomorphic to a disk is 3-connected, (2) the weights on edges have positive values, and (3) the boundary is mapped into a convex polygon. To apply this theorem to each patch, we examine the three conditions — 3-connectedness, positivity and convexity.

3-connectedness: For a mesh homeomorphic to a disk, the graph derived from it is 3-connected if there exists no edge which is not on any patch boundary but whose adjacent vertices are both on patch boundaries. When such an edge exists, we insert a vertex on the edge and divide the two adjacent triangles accordingly.

Moreover we do the following procedure in advance. We check whether edges whose two adjacent vertices are on the same patch boundary exist or not. If there exist, we convert the boundary to use these edges. This modification does not change the topology of the meta-mesh. It has some smoothing effect on the resulting quad mesh because the boundaries of patches may be the edges of the resulting quad mesh.

Positivity: The mean value coordinates [4] are chosen as weights because of its positivity.

Convexity: When boundaries are mapped into convex polygons, each patch is mapped into a rectangle because convex polygons with right angles are surely rectangles. Weights of edges on patch boundaries are set to extremely large values. Then the vertices on patch boundaries are placed on the line between two meta-vertices. In result, the vertices on the boundaries are positioned into a convex shape.

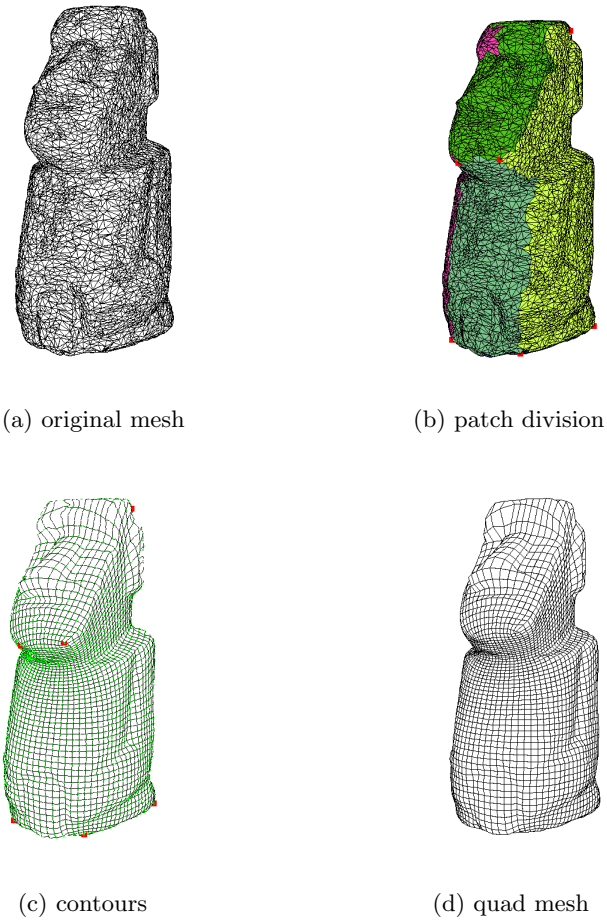
Under these conditions injectivity of the mapping is guaranteed. But this hurts the quality of the resulting quad mesh because the seams become visible. To avoid this, we decrease the weights of the edges on the patch boundaries toward the original weights unless there exist flipped triangles. If triangles are flipped by the decreasing to the original weight, we use a binary search to find smaller weights under the condition that any triangle

is not flipped.

4 Experimental results

In this section we show some experimental results. We first show results along the overall process in Figure 1. The original triangular mesh has about 7000 vertices (Figure 1(a)). It is divided into 7 patches as shown in Figure 1(b). Figure 1(c) shows the contour lines of u and v values and Figure 1(d) shows the resulting quad mesh.

Red points in Figure 1 mean meta-vertices. The whole calculation needs a few minutes on an ordinary personal computer.



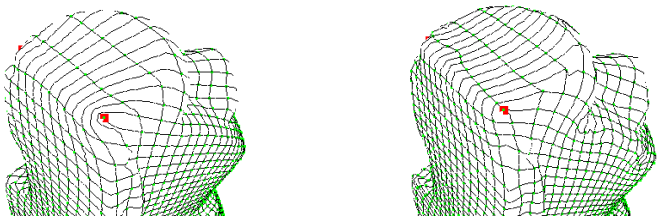
(a) original mesh (b) patch division

(c) contours (d) quad mesh

Figure 1: Process from a triangular mesh to a quad mesh

Next we show the effect of controlling the weights of edges on patch boundaries. In Figure 2(a), the mean value coordinates are used as weights.

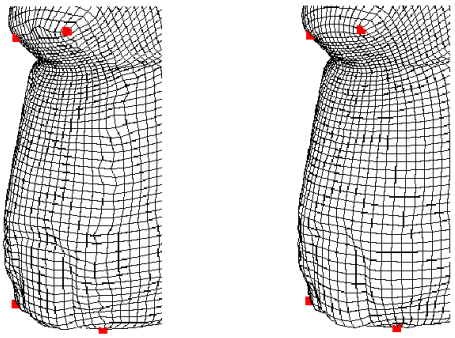
The meta-vertex in the middle has only one contour because some of triangles around this are flipped. On the other hand, the weights of the edges on the patch boundaries are added by a sufficiently large value in Figure 2(b). In this case no triangle is flipped and all the meta-vertices have the appropriate numbers of contours.



(a) (b)

Figure 2: The comparison of contours between original weights and increased weights

Finally, we show how increased weights damage the quality of the resulting quad meshes and it is refined by adjusting weights. The weights of the edges on the patch boundaries are added by a sufficiently large value in Figure 3(a). On the other hand, these weights are cut down as long as there exists no flipping triangle in Figure 3(b). It is found that smoothness increases as a result of cutting the weights down.



(a) (b)

Figure 3: The comparison of quad meshes between increased weights and adjusted weights

5 Discussion on the feasibility

In this section we discuss on the feasibility of the optimization problem(∗) appearing when deciding the length of meta-edges (Section 3.3). This problem may not have feasible solutions. See Figure 4 as an example.

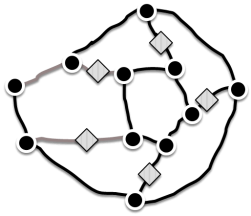


Figure 4: An example of edges with 0 length — if gray boundaries are set to be facing on the u-v plane, the length of the upper left boundary is inevitably equal to 0.

One solution is to divide patches until each patch has four meta-vertices because there is at least one feasible solution (i.e., all meta-edges have the same length). If a patch has even meta-vertices, it is possible by inserting meta-edges. In contrast, in case of a patch with odd meta-vertices, it is impossible without inserting meta-vertices. In this case, by inserting meta-vertices on meta-edges derived from a path of patches from a patch with odd meta-vertices to other patch with odd meta-vertices, patches with odd meta-vertices can be converted to have even meta-vertices without changing the parity of other patches (see Figure 5). It is possible because the number of patches with odd meta-vertices is surely even.

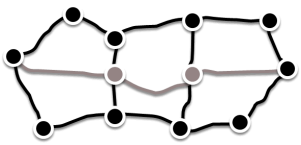


Figure 5: An example of inserting meta-vertices — each patch at both ends has an odd number of meta-vertices. By inserting two gray meta-vertices, all the patches have even meta-vertices and are divided so as to have four meta-vertices.

6 Conclusion

In this article, we improved the method for constructing quad meshes using temperature contours from a practical viewpoint. We discussed on the feasibility of the optimization problem and proposed an approach of splitting meta-facets. Moreover we assured injectivity of the mapping by adjusting

the weights of edges on patch boundaries. Increased weights assure injectivity but also damage the quality of the resulting quad meshes. In order to adjust weights, we used a bisection algorithm which requires iterative solving of linear systems. Working out more efficient methods for weight adjustment is an issue in the future.

Acknowledgments

We would like to thank CGAL library. The model in the figures is courtesy of AIM@SHAPE. This work is supported by the Grant-in-Aid for Scientific Research(b) (No. 20360044) and for Exploratory Research (No. 19650003) of the Japanese Society for Promotion of Science.

References

[1] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, and M. Desbrun. Anisotropic polygonal remeshing. *ACM Trans. Graph.*, 22(3):485–493, 2003.

[2] D. Cohen-Steiner, P. Alliez, and M. Desbrun. Variational shape approximation. *ACM Trans. Graph.*, 23(3):905–914, 2004.

[3] S. Dong, P.-T. Bremer, M. Garland, V. Pascucci, and J. C. Hart. Spectral surface quadrangulation. *ACM Trans. Graph.*, 25(3):1057–1066, 2006.

[4] M. S. Floater. Mean value coordinate. *Comput. Aided Geom. Des.*, 20(1):19–27, 2003.

[5] M. S. Floater. One-to-one piecewise linear mappings over triangulations. *Math. Comput.*, 72(242):685–696, 2003.

[6] F. Kälberer, M. Nieser, and K. Polthier. Quadcover - surface parameterization using branched coverings. *Computer Graphics Forum*, 26(3):375–384, Sept. 2007.

[7] N. Ray, W. C. Li, B. Lévy, A. Sheffer, and P. Alliez. Periodic global parameterization. *ACM Trans. Graph.*, 25(4):1460–1485, 2006.

[8] Y. Tong, P. Alliez, D. Cohen-Steiner, and M. Desbrun. Designing quadrangulations with discrete harmonic forms. In *SGP '06: Proceedings of the fourth Eurographics symposium on Geometry processing*, pages 201–210, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.