A Study of Geometric Shape of Polygons for Additive Manufacturing

Tsutomu Kinoshita[1)] Katsuhisa Yamanaka[2)] Kouichi Konno[2)] Yoshimasa Tokuyama[3)]

[1)] Faculty of Engineering, Tohoku Gakuin University.

[2)] Faculty of Science and Engineering, Iwate University.

[3)] Faculty of Engineering, Tokyo Polytechnic University.

***Abstract*— In recent years, 3D data becomes very popular and highly-qualified 3D printers are available with low costs. This enables a lot of people to use laminate modeling devices. In this present condition, however, few techniques have been established to evaluate the quality of the objects output by laminate modeling devices. This paper thus studies how 3D data in such devices affects the quality of printed objects and examines a technique to optimize input data. As the first step to achieve this purpose, we focus on triangular polygon shapes as 3D input data and examine the effect to the quality of printed objects. This paper examines how triangulation pattern change affects 3D model shapes, without changing the location of vertices. In this study, 3D measured point clouds are used. All triangulation patterns are defined by using the same vertices. When a pattern is generated, triangular faces are aligned so that they do not overlap. The polygons generated from each pattern are evaluated. The surface areas of triangles in each pattern and the length of edges are compared and the total surface areas of whole shapes are examined.**

Keywords— Polygon mesh, Additive manufacturing, Objective evaluation

I. Introduction

3D scanners are popular in recent years. Such devices are used to measure shapes three dimensionally and the obtained point clouds are used to build polygon models, which are applied to various purposes. In archaeology, techniques for three-dimensional operations are also widely studied. Relics with high academic values are measured by 3D scanners and the 3D data are used as digital archives. The replicas of such relics are output by 3D printers for exhibition in museums, which is one of the applications of digital archives.

The density of the measured point clouds is too high as input data for 3D printers to generate replicas. The density is required to be thinned to generate polygons of STL or other formats. Generally, the techniques to generate polygons from point clouds are classified into three types: (1) the sequence of measured point clouds is used as it is, (2) Delaunay triangulation is used, and (3) surface approximation by implicit function is used[1].

This paper aims to examine how triangulation is affected by fixing the number of points in input point clouds and their coordinates and changing only the triangulation patterns of the triangles (connection of points).

II. Proposed Method

A. Outline of the Process

To examine our proposal for this paper, 3D point clouds generated by 3D measurement devices are used.

The outline is as follows: 3D points are projected onto a 2D plane to represent the points in a 2D space; in the 2D space, vertices are connected with straight lines to generate triangulation patterns; and triangles are generated in the 3D space and the generated shapes are evaluated.

The examination is performed in the following procedure:

(1) A boundary curve (closed curve) is defined to enclose all the input 3D vertices. The boundary curve is calculated from the sequences of the measured point clouds.

(2) For the input 3D vertices, the minimum square plane is defined. The 3D vertices are projected onto the defined plane and their coordinates are converted to those on the 2D plane.

(3) The boundary curve defined in step (1) is also defined in the 2D coordinates.

(4) By using the method described in the next section, B. Triangulating Polygons, the triangulation patterns are generated in the 2D space. The triangles are to be generated only inside the boundary curve. Two or more patterns might be defined for triangles.

(5) According to the patterns defined in step (4), triangles are generated in the 3D space. The generated shapes (polygons) are evaluated for each triangulation pattern.

B. Triangulating Polygons

Now, we explain how to prepare a data set of triangulation patterns. We are given a point cloud in a 2D plane and a boundary curve which is a sequence of line-segments between two points and includes all the points inside. See Fig.1(a) for an example. Then, we wish to generate a data set of triangulation patterns of the inside of the boundary curve.

Our method is based on an enumeration algorithm of triangulation patterns by Katoh and Tanigawa [2]. The algorithm enumerates all the triangulation patterns of the convex hull of a given point set. Using the algorithm as a subroutine, we generate a data set of triangulation patterns.

978-1-5386-2615-3/18/$31.00

We define some notations. Let S be a point cloud in a 2D plane. An *edge* (i,j) is the line segment connecting the two points i and j in S. For three points p, q, and r in S, a triple {p, q, r} is a *triangle* if they are not collinear and the convex hull of p, q, and r contains no point in S. A *boundary curve* of S is a closed sequence of line-segments between two points in S such that every point in S is included in its inside region or its contour. Note that the convex hull of S is one of boundary curves of S. An edge in S is *external* if it is contained in the boundary curve of S, and *internal* otherwise. Let P be a boundary curve of S. A *triangulation pattern* of S and P is a set E of edges in S such that (1) E contains all the external edges and (2) E divides the inside region of P of S into triangles. For instance, Fig.1(d) shows a triangulation pattern of a point cloud and boundary curve in Fig.1(a). For the case that a boundary curve is the convex hull of S, we define more restricted triangulation patterns, called edge-constrained triangulation pattern. Suppose that P is the convex hull of S. Let F be a set of edges of S such that no two edges in F intersect. An *edge-constrained triangulation pattern* of S, P, and F is a triangulation pattern of S and P containing all the edges in F. We call the edge sets here *constrained edge sets*.

Katoh and Tanigawa [2] proposed an algorithm that enumerates all edge-constrained triangulations for a given point cloud, the convex hull of the cloud, and a constrained edge set. On the other hand, in this paper, we are required to generate a data set of triangulation patterns of a given point cloud and a boundary curve. Therefore, we cannot apply their algorithm directly to generate patterns.

Now, the outline of our method is as follows. Let S be a point cloud in a 2D plane, and let P be a boundary curve of S. First, we design a method for enumerating all the triangulation patterns of S and P using Katoh and Tanigawa's algorithm as a subroutine. The method enumerates all the triangulations patterns of S and P. However, the size of the obtained data is extremely huge so it is difficult to use the enumeration data as it is. Therefore, using the enumeration method, we also design a method for generating only a subset of enumeration data set with some parameters. By adjusting the parameters, we can have a data set with a suitable number of patterns.

We first design an algorithm that enumerates all the triangulation patterns of S and P. For the purpose, we use the edge-constrained enumeration so that only the triangulation patterns of S and P are enumerated, as described below. Then, we remove the constrained edges from the enumerated triangulations. The obtained triangulation patterns are the ones of S and P. The outline described above is shown in Fig. 1.

Now, we explain how to construct a constrained edge set. Intuitively, a constrained edge set consists of edges outside of P and inside of the convex hull of S. Formal definitions are as follows. Let R be the region inside the convex hull of S and outside P. A pair (i, j) of points in S is *fixed* if (1) (i, j) is an edge of P, (2) (i, j) is an edge on the contour of the convex hull of S, or (3) (i, j) is properly included in R. Let F_f be a set of fixed edges, say a *fixed edge set*, of S and P such that no two edges intersect and R is triangulated. For example, Fig.1(b) shows such set of fixed edges of the point cloud and the boundary curve in Fig.1(a).

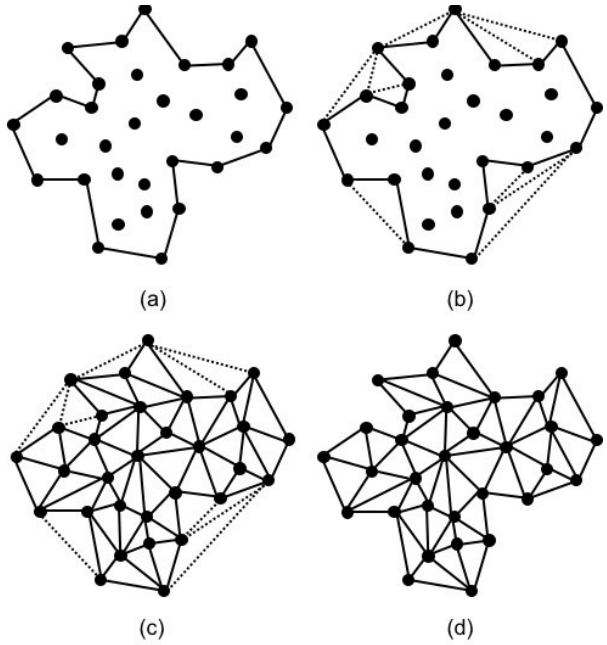


Fig. 1. (a) An example of an input: a point cloud S and a boundary curve P. The edges on P are represented by solid edges. (b) The set of fixed edges of S and P. The dashed edges are the edges in the region inside the convex hull of S and outside P. (c) We apply Katoh and Tanigawa's algorithm to the triangulation pattern and constrained edge set in (b), then triangulation patterns are enumerated. This figure shows only one triangulation pattern among the enumerated patterns. (d) We removed the dashed edges from each triangulation, then, all the triangulation patterns of S and P are obtained. Only one example of a triangulation pattern is also shown here.

We apply Katoh and Tanigawa's algorithm to S, the convex hull of S, and F_f, then we removed the fixed edges in F_f except for the edges on P. The obtained triangulation patterns are the ones of S and P. See Figs.1(c) and (d) for an example.

As described above, we can enumerate all the triangulation patterns of S and P. However, the enumerated data set of triangulation patterns is extremely huge. Therefore, we next design an algorithm that generates only a subset of the enumeration data set.

Katoh and Tanigawa's algorithm enumerates all the triangulation patterns by traversing a rooted tree structure called a triangulation tree [2]. The *triangulation tree* of a point cloud S and a constrained edge set F is a rooted tree such that (1) its nodes correspond to triangulation patterns of S, the convex hull of S, and F, and (2) its edges correspond to parent-child relations, each of which is defined using the diagonal flip of an edge. See [2] for further detail.

To obtain a subset of triangulation patterns, we traverse a subtree of the triangulation tree, not the whole of the tree. The details are as follows. Suppose that we are given a triangulation pattern T of S and P with a fixed edge set. (A triangulation pattern can be constructed by any method. For example, we can use an algorithm that constructs Delaunay triangulations.) Next, we repeat to finding parents (or children) from T in the triangulation tree. Let T' be the obtained triangulation pattern, and let t be the number of finding parents. Let us assume that if $t > 0$, t means the number of finding parents, otherwise t means

the number of finding children (when we find children, we choose randomly choose a child from candidates of children in each step). In addition, we flip the edges of T' one by one. The flipped edges are chosen randomly. Let T'' be the obtained triangulation patterns, and let r be the number of the flipped edges in T'. Then, we traverse the subtree rooted at T'', then only the triangulation patterns in the subtree are obtained. Note that the algorithm above can adjust the number of output triangulations patterns using t and r as parameters.

The algorithm above output triangulation patterns similar to T (if we set a small value to r), since for adjacent two triangulation patterns in a triangulation tree, one pattern is obtained from the other by only one diagonal flip. Hence, we have to choose the initial triangulation pattern T such that T satisfies desired conditions if needed. In our experiment, it is required that triangulation patterns contain only short edges, and so the following condition is checked: The length of every edge in a triangulation pattern is equal to or smaller than a parameter d. If the condition is true, then a triangulation pattern is inserted into a data set, otherwise, a pattern is discarded. We apply our method to the point cloud and the boundary curve shown in Section III. In our experiment, we set t = 30, d = 25(The width and height of the bounding box of our point cloud, shown in the next section, are 63.9308 and 59.8413, respectively), and r = 10.

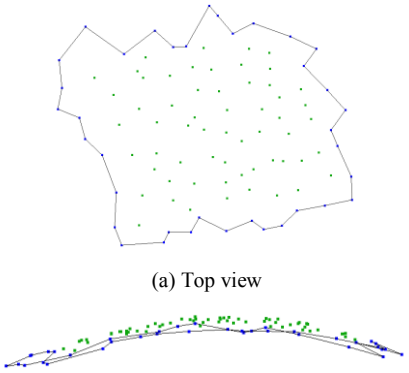
III. Experimental Results

Our algorithm was applied to the clay pot data shown in Fig. 2. The data in Fig. 2 is a 3D point cloud generated by a 3D measurement device from a clay pot exhibited in Learning Center of Ruins in Morioka City. Fig. 2 (a) shows the polygon model generated from sequences of the measured point clouds and Fig. 2 (b) is the point cloud model whose points are thinned to 1/100 of the original. Our algorithm is applied to the data in Fig. 2 (b).A bounding box was defined on the section enclosed with the red square in Fig. 2 (b) and the experiment was performed to the point clouds in the bounding box.

(a) Original data (b) 1/100 point cloud

Fig. 2. Point cloud data

Fig. 3 shows the points in the bounding box and the boundary curve. Fig. 3 (a) is the top view of the point cloud and Fig. 3 (b) is the side one. This point cloud was input to generate the minimum square plane and the points are mapped in a 2D space by using the coordinates of the projected points, which is shown in Fig. 4. The red line in Fig. 4 represents the *x* axis and the green one represents the *y* axis. Fig. 5 shows an example of triangulation applied to the 2D points of Fig. 4 by using the method described in section B. Triangulating Polygons. Fig. 5 (a) shows an example of triangulation with the minimum dispersion of triangle surface area in a 3D space and (b) shows the maximum one, among the examined triangulations. When we pay attention to the section enclosed with the red square in (b), the triangulation pattern differs from the one in (a). Such triangulation patterns are obtained as many as possible.



(a) Top view

(b) Side view

Fig. 3. 3D point clouds in a bounding box

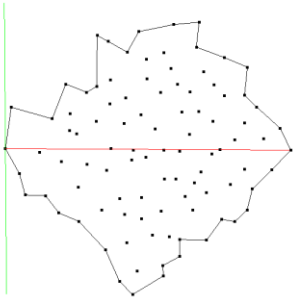
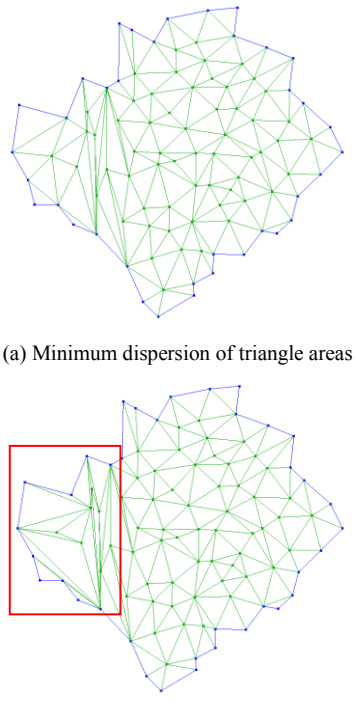


Fig. 4. Points in a 2D space

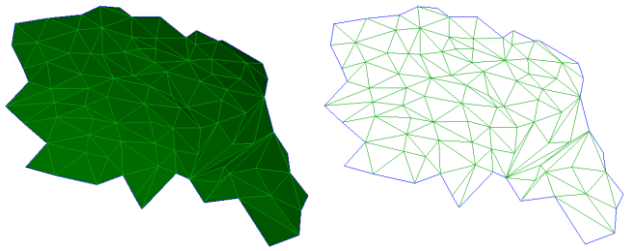


(a) Minimum dispersion of triangle areas

(b) Maximum dispersion of triangle areas

Fig. 5. Triangulations in a 2D space

Fig. 6 shows triangles generated in a 3D space from the 2D triangulation pattern of Fig. 5(a). Fig. 6 (a) shows the polygon faces and (b) shows the polygon edges. Our numerical experiment is performed to approximately 28,000 triangulation patterns.



(a) Polygon faces (b) Polygon edges

Fig. 6. Polygons in a 3D space

Table I is the result of calculation applied to approximately 28,000 triangulation patterns, showing the maximum, minimum, average, total and dispersion of triangle surface areas, showing the maximum and minimum values and the polygons generated from the sequence of the measured point clouds. Among the triangles in the polygon, the minimum triangles were compared to find an approximately 2.9-fold difference, and the maximum ones were compared to find an approximately 2.0-fold difference. Another big difference is also found for the dispersions. Variations of the surface areas differ depending on the triangulation patterns. Table II is the result of calculation in the same manner for triangle edges and shows the numbers. Depending on the patterns, the side lengths, maximum and minimum values have approximately 1.2-fold difference.

The result clarified that different alignment of triangles in the same point cloud leads to a large difference of shapes. If triangles are aligned differently for various purposes, we can obtain expected data with high quality. On the other hand, the calculation cost is quite high for 2D triangulation patterns. Because of this high cost, our algorithm was not applied to all the patterns obtained from the point cloud and approximately 28,000 patterns were used for the experiment. This number of patterns, which is a part of all patterns, resulted in not so large difference between the maximum and minimum surfaces areas and lengths of sides.

For our experiment, XVL Kernel and XVL assistant of Lattice Technology, Co., Ltd. are used.

TABLE I. AERA EVALUATION

	Triangle Area				
	Avg	***Min***	***Max***	***Total***	***Dispersion***
Minumum	15.86	1.26	44.67	2425.92	49.69
Maximum	15.95	3.50	91.65	2439.72	104.1
Polygons generated from the sequence of the measured point clouds	15.87	0.52	44.67	2427.71	52.77

TABLE II. EDGE EVALUATION

	Triangle Edge			
	Avg	***Min***	***Max***	***Dispersion***
Minumum	6.79	1.92	21.59	8.58
Maximum	7.29	2.14	26.21	15.47
Polygons generated from the sequence of the measured point clouds	6.67	1.92	13.93	5.015

IV. CONCLUSION

In this paper, we presented that different alignment of triangles in the same point cloud leads to a difference of result shapes, while all of triangulation patterns have not yet been verified because of operation speed and some other issues. We will expand our algorithm for better operation speed and availability to more complex shapes and surface models.

REFERENCES

[1] Y. Nagai, “Geometry Interface for 3D Scanning,” Journal of the Japan Society for Precision Engineering Vol.79, No.6, 2013.

[2] N. Katoh and S. Tanigawa, “Enumerating edge-constrained triangulations and edge-constrained non-crossing geometric spanning trees,” Discrete Applied Mathematics, Vol.157, No.17, pp.3569-3585, 2009.