CONSTRUCTION OF THREE-DIMENSIONAL MODELS FOR STRUCTURAL OBJECTS FROM TOMOGRAPHIC IMAGES

Kevin Kuang-Hui Tseng and Ping Lu

School of Civil and Structural Engineering
Nanyang Technological University
Republic of Singapore
E-mail: ckhtseng@ntu.edu.sg

Abstract**:** In this paper, a new surface-based modeling algorithm is presented to construct 3D models of structural objects from 2D slices, which contain contours obtained from tomographic images. The main objective of this method is to produce the most precise possible model for branching bodies taking into consideration that the slices might be far apart in reality. Boolean operations will be used as the technique to solve the branching problem. The proposed framework is not pixel-based and a complete 3D solid model will be created. Examples show that the proposed approach is robust and superior to existing methods for this purpose.

Keywords**:** Solid Modeling, Geometric Modeling, B-rep, Boolean Operations, CAD, CAE

1. INTRODUCTION

Reconstructing the boundary of a solid object from a series of parallel planar cross-sections has attracted much attention during the past two decades. The original motivation comes from medical imaging applications where cross-sections of human organs are obtained by CT, ultrasound, or NMR. These cross-sections are the basis for interpolating the boundary of the organ. The interpolated object can then be displayed in graphics applications, or even manufactured by numerically controlled (NC) machining. This technique has been utilized in many applications in engineering, geology, and military, such as the nondestructive digitization of objects, reconstruction of the three-dimensional model of terrain from topographic elevation contours, etc.

However, until now, few people use it to solve problems in civil engineering where visualizing underground objects is a very important topic. For example, if a structural engineer is designing the foundation of a building in certain location, general geology of the site should be known. Traditional method to obtain the information is to check the site by drilling holes at only a few selected points. An easier and more efficient way would be to use the tomographic techniques. 3D models of the site can be formed from the recorded signals obtained from computer tomography. Thus, the soil layers and underground water level can be shown more precisely than the traditional engineering report. With this model, any faults, quarries, springs, swallow holes, mines or shaft, or other features, which will have a bearing on the foundation works, can be illustrated clearly.

While the construction of 3D models from 2D slices is often to create surfaces that can be displayed and interpreted visually, it also serves as a stage in the automatic analysis of the 3D shape. For example, models of underground buildings can be constructed from tomography images, either to assist in the comprehension of the structure of the object or to facilitate its automatic analysis, especially for the finite element analysis. When constructing models for further analysis, the performance of the algorithm is more important than the visual quality. So long as the shape is not distorted, speed of execution is an important issue.

The objective of this paper is to propose an economical approach for implementing such a scheme. The pixel-based methods are not suitable for our purpose because they tend to require a lot of processor time and memory, and give a surface that, while visually pleasing, exceeds the demands of the application and the quality of the original data.

There is a wealth of previous work in the area of 3D reconstruction from 2D slices. Most of the methods can be classified into two categories [7].

One is the volume-based method, such as ray-casting method, integration method, splatting methods, and V-buffer rendering method. The other is the surface-based method, which includes three fundamental approaches: contour-connecting method, opaque cube method [11] and marching cube method [5,6,14]. They typically fit surface primitives such as polygons, usually triangles, or patches to constant-value contour surfaces in volume dataset. Volume-based methods have been discarded here. They are popular in the graphic community, and produce very nice images. However, in modeling structural objects, the improvement in the visual quality does not justify the high coast in execution time and the computer storage space.

Contour-connecting algorithm typically addresses three different aspects of the construction problems, known as correspondence problem, tiling problem, and branching problem (Fig. 1). Most of the previous contour-connecting [8,9,12] approaches ignore the case of having a branching body, in which a single contour in one slice corresponds to two or more contours in the next; others explain how to deal with branching bodies, but their algorithm have several other limitations. Lin et al. [13] model branching regions by interpolating many intermediate contours. This method generates a smooth surface at the cost of a large number of triangles. Christansen et al. [4], Shantz [16], and Shinagawa et al. [17] suggested to form composite contours, adding fabricated vertices between the adjacent contours to model the saddle surface implied by the contours. Their scheme is not adequate when the contours have awkward sharps. There are situations where more than one intermediate node is required, and manual intervention is suggested. Meyers [15] presented a method of triangulating 'canyons' without automatic locating them. Because of the horizontal triangle problem associated with this approach, feeding the triangulation mesh into a surface-fitting program to regenerate the surface is necessary.

Another branching algorithm, proposed by Ekoule [6], created an interpolated contour when the branching is detected, which is triangulated with the source contour and all destination contours. The scheme also used convex hull, and is relatively costly and slow for similar results. Barequet [2] first matched and tiled similar portions between corresponding contours. Then, the clefts, which are the polygons formed by the untiled portions, are triangulated with (a variant of) a 3D minimum area triangulation technique. If the X-Y projections of the clefts are nested, bridges are added to break the nesting. This algorithm may produce horizontal triangles when a feature in one slice does not resemble the features in the other slice. Bajaj [1] improved the method suggested by Barequet [2] by triangulating the untiled region with its medial axis, which is projected to the middle height of the two adjacent slices. In practice, it is not necessarily the case. Boisonnat [3] constructed the Delaunay triangulation for each slice by projecting one triangulation onto the other, and obtaining a collection of the tetrahedral, aiming to maximize the sum of their volumes. Geiger [10] improved Boissonnat's method so that it can handle complicated branching and dissimilar contours. In this approach, all the points were triangulated even the ones at the same slice. This is considered one of the drawbacks of the Delaunay triangulation. Another drawback is the assumption that the branching points is located at the lower slice, which might not be true in reality.

Our approach, as most previous surface-based methods, addresses three fundamental problems separately. First, the correspondence is judged among contours with the same orientation. Then, the one-to-one body between each pair of corresponding contours is constructed. During the previous two steps, a flag of either '1' or '-1' will be set for each body to indicate whether the body is a real one or a hole. Finally, all the bodies flagged by '1' are united, and holes flagged by '-1' are subtracted. By using Boolean operations we solve the branching problem without adding bridges, triangulating untiled regions, or inserting interpolated slices. Many experimental results will be shown to demonstrate the efficiency, accuracy, and robustness of this proposed approach.

This paper is organized in the following fashion. A more precise definition of the research topic will be given in section2. Section 3 describes the modeling procedure in a detailed way. Section 4 presents some experimental results. Finally, the advantages and future work will be discussed in Section 5.

2. STATEMENT OF THE PROBLEM

This method can accept the input of either image slices or contour data. If the input data are image slices, they can be changed into contour data by using 2D marching cube algorithm [6]. In practice, we use 'Scion Image' to perform the job. Here we assume that the input is the contour data.

Input:

A series of planar slices, which are parallel to the XY-plane, and lie at height $Z = Z_1, Z_2 \dots Z_n$. Each slice consists of a list of closed and simple polygonal contours, which divide the slicing plane into a solid region and a void region. They represent the boundaries between "material" and "nonmaterial" areas. These contours do not intersect each other, but a contour may be enclosed in any number of other contours, which themselves may enclose other contours. The contours in a slice do not necessarily come from the same object, and an object may be represented by more than one contour in a slice. Each contour is given as a circular list of vertices, each specified by its (*x, y*) coordinates.

Output:

A polyhedral B-Rep solid model whose cross-sections coincide with the input slices.

Constructing a polyhedron between a pair of adjacent slices is independent of that of other pairs of slices. Therefore, a natural simplification of the modeling problem, also taken in most earlier works, is to consider only a single pair of successive parallel slices (S_1 and S_2) and to construct a model within the layer delimited by the planes of the slices, which interpolates between the given slices. The concatenation of these models will give a solution to the full problem.

Because only two adjacent slices are considered, contours can be classified into uContours (contours in the upper slice S_2) and bContours (contours in the lower slice S_1) without using the coordinate value of the z-axis.

3. MODELING PROCEDURE

3.1 Construction Criteria

The problem of shape reconstruction is underconstrained, which implies that there are many feasible solutions. Some reasonable correspondence and tiling rules are imposed to generate the most likely object which satisfies two basically requirements: first, the reconstructed surface on the slice should coincide with the original contours; second, the reconstructed surface and solid regions form closed surfaces of polyhedral. Before presenting the construction criteria, some useful definitions are described.

Definition 1. *Augmented contours*: New vertices are added in both contours at those points where the projection of that contour would cross another contour.

A single polygon divides the slice plane into two parts: interior (material) and exterior (void). But in realistic cases, one contour might be inside another contour. In order to reflect these complex cases, the concept of the orientation of one contour is introduced.

Definition 2. *The orientation of one contour*: Walking along the orientation of one contour, the solid region is on its left side. Therefore, the solid region is inside of a CCW (counterclockwise) contour and is outside of a CW (clockwise) contour.

Definition 3. *Corresponding contours*: If there is at least one vertex of C_1 located inside the projection of C_2, or at least one vertex of C_2 located inside C_1, and C_1, C_2 have the same contour orientation, then C_1, C_2 are said to be corresponding contours.

Definition 4. *Legal slice chord:* All slice chords satisfy criterion 1 to criterion 3 described later are called legal slice chords.

Definition 5. *OTV (optimal tiling vertex):* V_2 in one slice is the OTV of V_1 in another slice (V_2 = OTV (V_1)), if V_1V_2 is the shortest among all legal slice chords incident with V_1.

Criterion 1. Any overlapping vertices must be tiled.

Criterion 2. The slice chord T connecting vertices V_1 and V_2 can not intersect with any of the contours in both slices.

Criterion 3. Assume that C_1 (a CCW contour) is a corresponding contour of C_2 (a CCW contour too). If V_2' is outside of C_1, T' will not be in the side of V_2 (Fig. 2.a.) If V_2' is inside C_1, T' will be in the left side of V_2 (Fig. 2.b.) If V_2 is an overlapping vertex, T' will not be in the right side or left of V_2 and V_2' simultaneously (Fig. 2.c.)

3.2 Construction procedure

<u>*Step1: Data Representation and Storage*</u>

A C-style structure is defined to represent and store the information about one contour. Two information arrays are used to store the orientation and vertex coordinates of the contours in the lower slice and the upper slice, respectively. The information is read into these arrays from the input data files. Three relationship arrays are used to represent the relations among contours in the same slice or two adjacent slices, and they can have four different values, each represent a relation case.

<u>*Step2: Create Augmented Contours*</u>

Using the augmented contours instead of the original input contours allows the tiling of contour segments whose projections cross each other and grantees that Criterion 1 is satisfied.

If the projection of one contour segment crosses another contour segment at V_1', V_1' is added to both C_1 and C_2 as an overlapping vertex. If the intersection of one contour segment projection with another contour segment is a one-line segment, new vertices are inserted so that the overlapping part is a contour segment in both slices.

Step 3: Decide Correspondence Between Contours

We judge correspondence only between contours with the same orientation. Because of this, nested contours will be developed as bodies may include holes such as the tunnel structures.

From definition 3 it can be seen that deciding the correspondence between any two adjacent contours is to examine the number of vertices in one contour located inside the other contour. The algorithm presented in [18] is used to judge if the point is inside of the polygon or outside of it.

Step4, Construct Tiling Between Each Pair of Corresponding Contours.

If there is one contour in one slice corresponding to more than one contour in another slice, tile them respectively. Thus many one-to-one objects will be created. If the body is constructed between two CCW contours, it is flagged '1'; if between two CW contours, it is flagged '-1'.

First, the OTV is searched for each vertex of any pair of corresponding contours based on the distance. Starting from the closest vertex on the adjacent slice until one satisfies the definition of OTV is found. An OTV table is formed to store the OTV pairs. Next, forming tiling triangles between corresponding contours based on the OTV table. Identify the starting slice chord $Q_i P_j$ by searching the OTV table. This will then be used as the first edge when developing the tiling triangles.

Tiling triangles can be constructed in two passes. Check the submatrix $\begin{bmatrix} a_{i,j} & a_{i+1,j} \\ a_{i,j+1} & a_{i+1,j+1} \end{bmatrix}$, if $\sum_{m=i,i+1}^{n=j,j+1} a_{m,n} \geq 2$. Triangles can be developed in seven different cases. We deal with them in the first pass. If $\sum_{m=i,i+1}^{n=j,j+1} a_{m,n} = 1$, triangles will be formed in the second pass.

During the second pass, tiles are formed as many as possible. The following application of Boolean operations will decrease untiled points dramatically. Therefore, untiled vertices formed by this method are much less than that formed by other methods. Actually, in our experiments, even in the most extreme cases, there are very few untiled vertices. So, it should be acceptable to discard the untiled vertices.

Step 5: Applying Boolean Operations to Get the Intersection Curves

After above four steps, many one-to-one bodies are formed. Unite all the bodies flagged by '1' and subtract bodies flagged by '-1' to achieve the final body. Complicate branching bodies with holes can be constructed by the application of Boolean operations.

4. EXPERMINTAL RESULTS

The algorithm has been implemented in C++. Many experimental results on various complex examples are presented in this section.

In Fig. 3, a very typical branching case is used to illustrate the construction procedure. Fig. 4 is a one-to-three branching problem. From the results we can see the more branches a body contains the more precise model we can construct. Compare with previous methods, we can obtain very good models of branching bodies without adding bridges, inserting interpolate slices, or triangulate untiled regions.

Most previous algorithms ignore the nest problem, in which a contour may be enclosed other contours or enclose other contours. The building showed in Fig. 5 is an example of nest problem. Because the correspondence is judged between CW contours and CCW contours, respectively, our algorithm can deal with such cases perfectly.

Fig. 6 shows a very complicated branching body containing a hole. The corresponding contours are extremely distorted. Even in this case, by using Boolean operation and our correspondence judging rules, the branching and the hole are formed successfully.

5. CONCLUSIONS AND FUTUR WORK

We have described a framework for reconstructing models from sets of contours that is suitable for constructing models of structural objects, in which nest problem is very common, and sometimes constructing branching body is necessary. We have extended previous works by allowing for complex branching and nest to occur.

The key idea is to separate the complicated many-to-many problem into several one-to-one problems, which are already handled well by existing methods. Unlike most of the existing contour-connecting methods, Boolean operations are introduced as the fundamental tool. A branching body is achieved by finding the intersection curves among all constructed one-to-one bodies. The use of Boolean operations combined with our tiling scheme dramatically reduces the amount of untiled points and the approach is robust.

This proposed framework judges correspondence between CCW contours and CW contours respectively, and two different kinds of Boolean operations are applied based on the orientation of contours. Therefore, it is capable of dealing with the nested contour problems very well. Most of the previous methods had ignored the problem.

The results show that our technique reconstructs the boundary of various objects in an intuitively appealing manner. The resulting 3D models are more than adequate even in extreme cases of tiling between two seeming totally different slices.

Till now, our experiments are all simulated ones. In the future, we plan to test our algorithm with some real data collected from practical civil construction applications.

REFERENCES

[1] C.L Bajaj, E.J. Coyle, and K.N. Lin, "Arbitrary topology shape representation from planar cross sections", *Graphical Models & Image Processing*, pp. 524-543. 1996.

[2] G. Barequet and M. Sharir, "Piecewise-linear interpolation between polygonal slices", *10th Computational Geometry*, pp. 93-102. 1994.

[3] J. D. Boissonnat, "Shape reconstruction from planar cross sections", *Comput. Vision Graphics Image Process*, 44, pp. 1-29. 1988.

[4] H.N. Christiansen, and T.W. Sederberg, "Conversion of complex contour line definition into polygonal element mosaics", *Computer Graphics,* 12(3) pp. 187-192. 1978.

[5] H.E. Cline, W.E. Lorensen, S. Ludke, C. R. Crawford, B.C. Teeter. "Two Algorithms for three-dimensional reconstruction of tomograms", *Medical Phys*ics, 15(3) pp. 320-327. 1988.

[6] B. Ekoule, F.C. Peyrin, and C.L. Odet, "A triangulation algorithm from arbitrary shaped multiple planar contours", *ACM Trans. Graphics,* 10(2) pp. 182-199. 1991.

[7] T.T. Elvins, "A survey of algorithms for volume visualization", *Computer Graphics*, 26(3) pp. 194 –201. 1992.

[8] H. Fuchs, Z. M. Kedem, and S. P Uselton,. "Optimal Surface Reconstruction from Planar Contours", *Commun. ACM* 20, pp. 693-702. 1977.

[9] s. Ganapathy, and T.G. Dennehy, "A new general triangulation method for planar contours", *Computer Graphics,* 16(3), pp. 69-75. 1982.

[10] B. Geiger, "Three-Dimensional Modeling of Human Organs and Its Application to Diagnosis and Surgical Planning", *Technical Report 2105*, INRIA, France, 1993.

[11] G.T. Herman, and H.K., Liu, "Three-dimensional display of Human Organs from Computed Tomograms", *Computer Graphics and Image Processing,* Volume 9, Number 1, pp. 1-21. January 1979.

[12] E. Keppel, "Approximating complex surface by triangulation of contour lines", *IBM. J. Res*. pp. 2-11 Dev. 1975.

[13] W.C. Lin and S. Y. Chen, "A new surface interpolation technique of re constructing 3D objects from serial cross-sections", *Computer vision Graphics Image Process*, 48 pp. 124-143. 1989.

[14] W.E. Lorensen, H.E. Cline. "Marching Cubes: A High Resolution 3D Surface Construction Algorithm", *Computer Graphics*, 21(4) pp. 163-169. 1987.

[15] D. Meyers, S. Skinner, and K. Sloan, "Surfaces from contours", *ACM Trans. Graphics,* 11(3) pp. 228-258. Jul.1992.

[16] M. Shantz, "Surface definition for branching contour defined objects", *Computer Graphics,* 15(2) pp. 242-270. 1981.

[17] Y. Shinagawa and T. L. Kunii, "The homotopy model: A generalized model for smooth surface generation from cross sectional data", *Viaual Comput*er, 7 pp. 72-86. 1991.

[18] J.G. Sun. "Computer Graphics". *Tsing Hua University Press*, pp. 413-415. 1998.

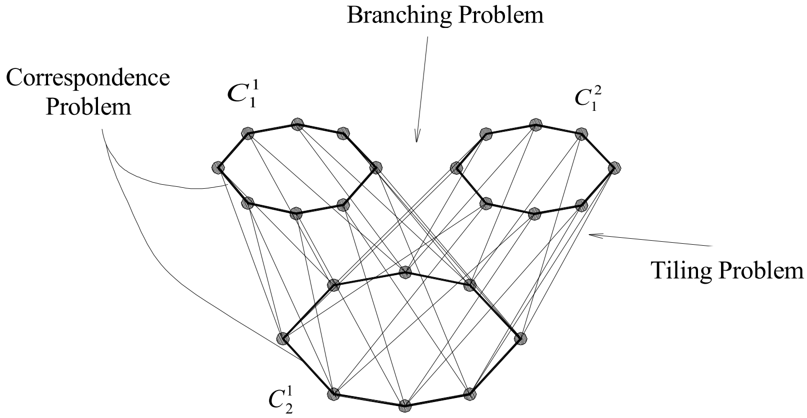
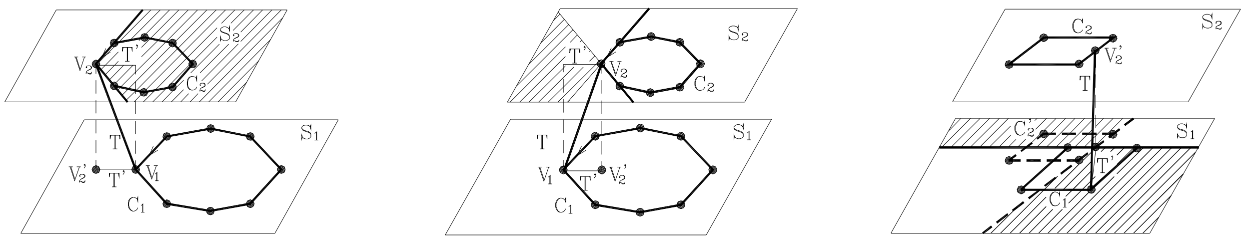
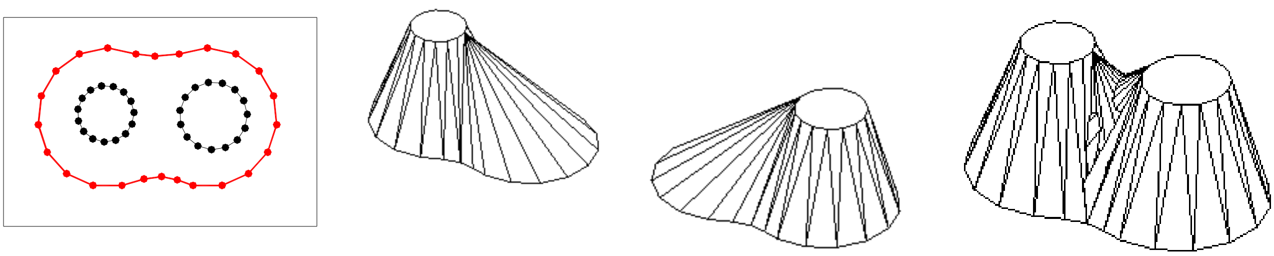


Fig. 1 Statement of correspondence problem, tiling problem, and branching problem.



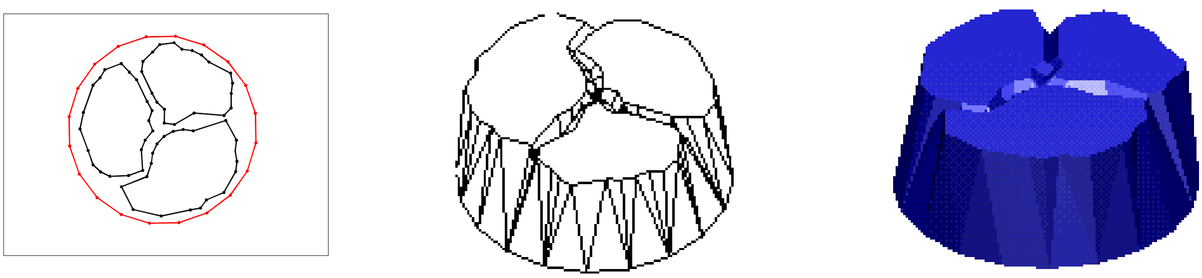
(a) (b) (c)

Fig. 2 Three cases of legal slice chord.



(a) (b) (c) (d)

Fig. 3 A one-to-two branching problem: (a) The top view of lower and upper slice; (b) and (c) Two one-to-one bodies constructed between two pair of corresponding contours respectively; (d) The result with hidden lines removed.



(a) (b) (c)

Fig. 4 A one-to-three branching problem: (a) The top view of lower and upper slice; (b) The result with hidden lines removed; (c) The shaded view of the result.

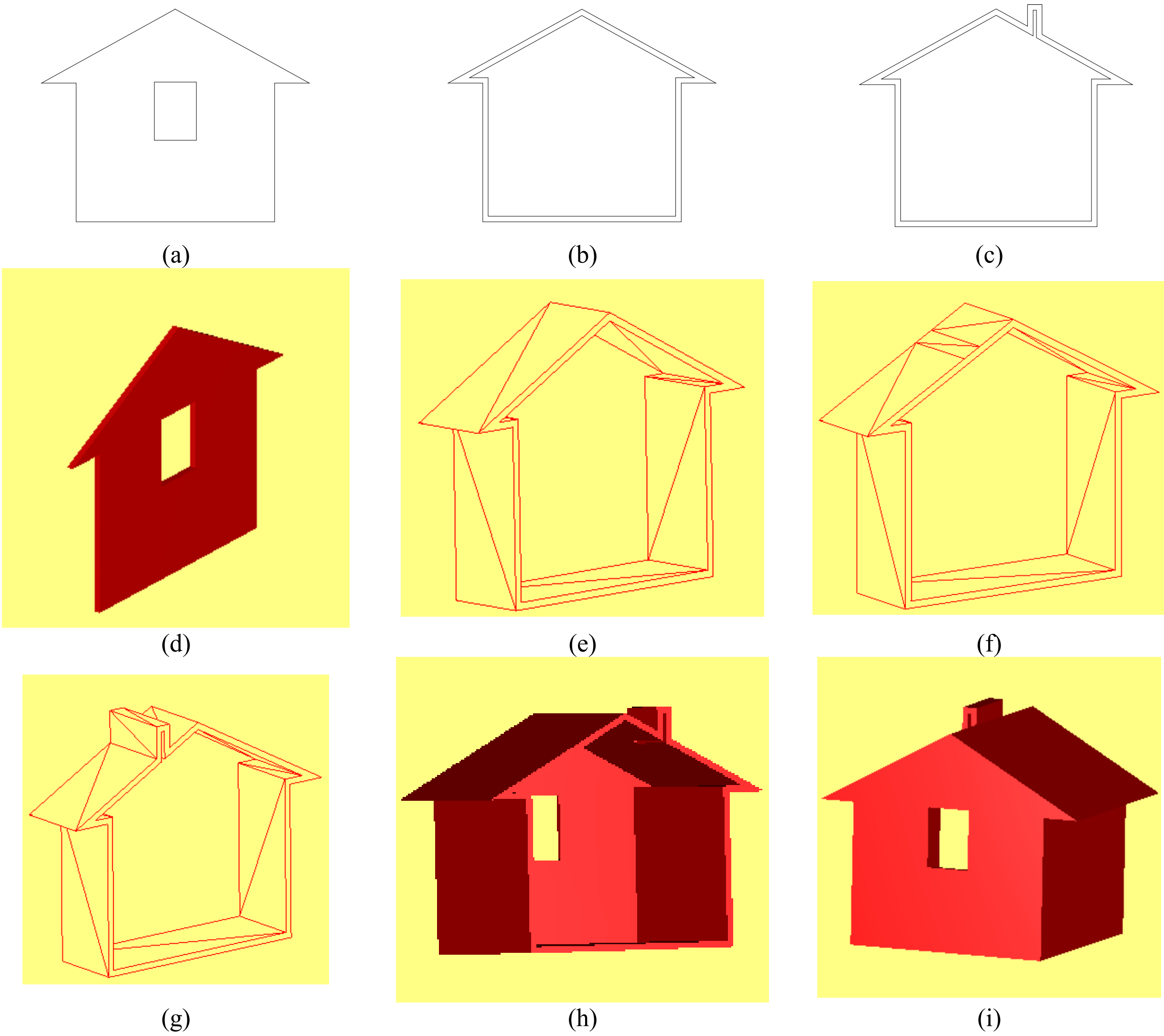


Fig. 5 *A simulate building:*: (a) (b) (c) Three cross sections of the building.; (d) shaded view of the wall constructed use slice (a) as both upper and lower slice;(e) (f) (g) models constructed between (e)-(e), (e)-(f) and (f)- (f) with hidden lines removed; (h) (I) The shaded view of the result seen from two different viewpoint.

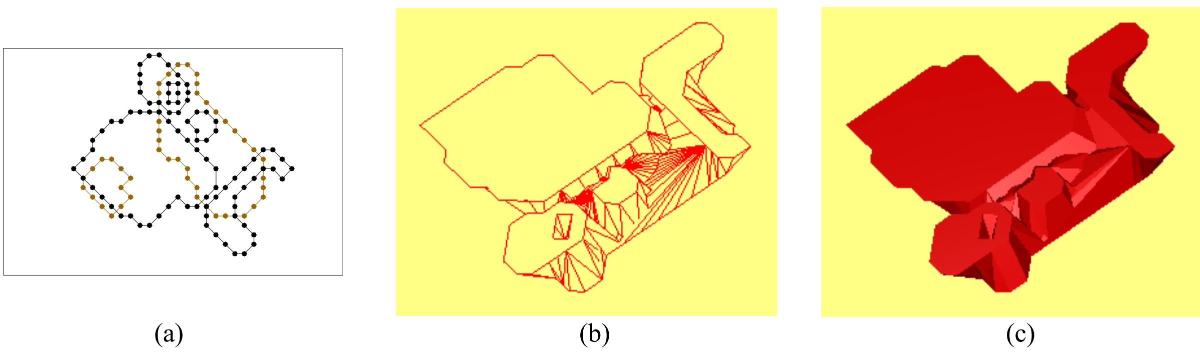


Fig. 6 *A complicated branching problem contains a hole*: (a) The top view of lower and upper slice; (b) The result with hidden lines removed; (c) The shaded view of the result.