Chapter 4

Experiment results

In this chapter, we will show some experiment results. Since the cutting method and the filling method are purely topological processes, their effects are not visible. We used a visualization technique to illustrate the results. In addition, we also used different colors to distinguish among various types of edges and vertices. Furthermore, different colors were used to separate surfaces that belonged to different connected components. In the experiments, we painted the singular vertices, singular edges, the isolated singular vertices and the non-boundary edges with the orange color, the yellow color, the pink color, and the light-blue color, respectively. As for the different connected components, we randomized a large number of combinations of the colors.

4.1 The Experiments

We partitioned the procedure of our experiment into two steps, including the cutting step and the filling step. In the cutting step, we had two basic definitions on connected components. The first is that if two faces share an edge that is not a singular edge, the two faces belong to the same connected component. The second is that we add an additional parameter called non-boundary edge. If two faces share an edge that is neither a singular edge nor a non-boundary edge, the two faces are in the same connected component.

In the hole filling stage, we used two strategies in the triangulation process. In the conventional method, a triangulation algorithm uses dynamic programming technique to identify a minimum weight patch to fill a hole. In this work, we extended the algorithm to find both minimum and maximum weight patches. Then, we chose one of the patches, either the minimum patch or the maximum patch, to satisfy our need because we did not handle the geometrical intersection problem.

As for the weighting function of the hole filling method, we used two control parameters, area and angle, to derive it. In most cases, the area parameter is good enough to handle the one hole problem. We need the angle parameter only when the hole is a crenellation one.

4.2 Data Selection

We know that there might be some connected components produced after executing our cutting method. Some of the existing algorithm do not care if the mesh is broken, but some algorithm do expect the broken part which need to be handled as less as possible. For example, if we only need one connected component in a 3D visual salience data retrieval process, then we have to eliminate some redundant connected components of a mesh until a mesh finally contain only one connected component.

We use two criteria to select a connected component out of several. They are the area and the number of faces. We set a threshold for both of them, and if the value of a connected component is not larger than the threshold, the block of surface is cataloged into deleted list. If we want to decrease the number of small-area connected components, we just set the threshold 1% in the area and 1% in the number of faces. Then any connected components under the threshold will be eliminated. If we need the largest meaningful one of the mesh model, we set the threshold to be 50% or more. The result will be one connected component with the largest area and largest number of faces; otherwise, all the connected component of the mesh model will be deleted.

4.3 Experiment Results

The first set of experiments we conducted was on a mesh model of a dinosaur as shown in Fig. 4.1. The dinosaur model had 6041 vertices and 12040 faces. Our algorithm identified 4 singular edges and 6 singular vertices in this mesh model. Fig. 4.1(a) shows the original input. We used a dotted square to show the singular edges and singular vertices located between the rear legs and the body of the dinosaur in Fig. 4.1(b). Let us view more detail inside the dotted square. As indicated in Fig. 4.1(c), $\{$A, B, C$\}$ is a face; $\{$A, B$\}$ and $\{$B, C$\}$ are singular edges; A, B, and C are singular vertices. $\{$A, C$\}$ is a boundary edge. After executing the cutting algorithm, the triangular face is separated from the mesh model, becoming another connected component as shown in Fig. 4.1(d). Moreover, the number of vertices was increased to 6049, and the number of faces did not change. In fact, the triangular face is not important because the percentage of its area and face number were not significant. In Fig. 4.1(e), we deleted the triangular face $\{A, B, C\}$ because it did not pass the area (1%) and the face number (1%) threshold. After executing the cutting algorithm, the next step was to perform hole-filling algorithm. It is hard to see the difference from the mesh model. However, after the filling process, two triangular faces were actually added into the mesh model. In addition to the holes generated when performing the cutting algorithm, there were two extra holes located near the eyes of the dinosaur model. Fig. 4.1(g) and 4.1(h) illustrate the dinosaur model before and after the filling process was executed, respectively. In the filling process, we only use the area of a face as the weight.

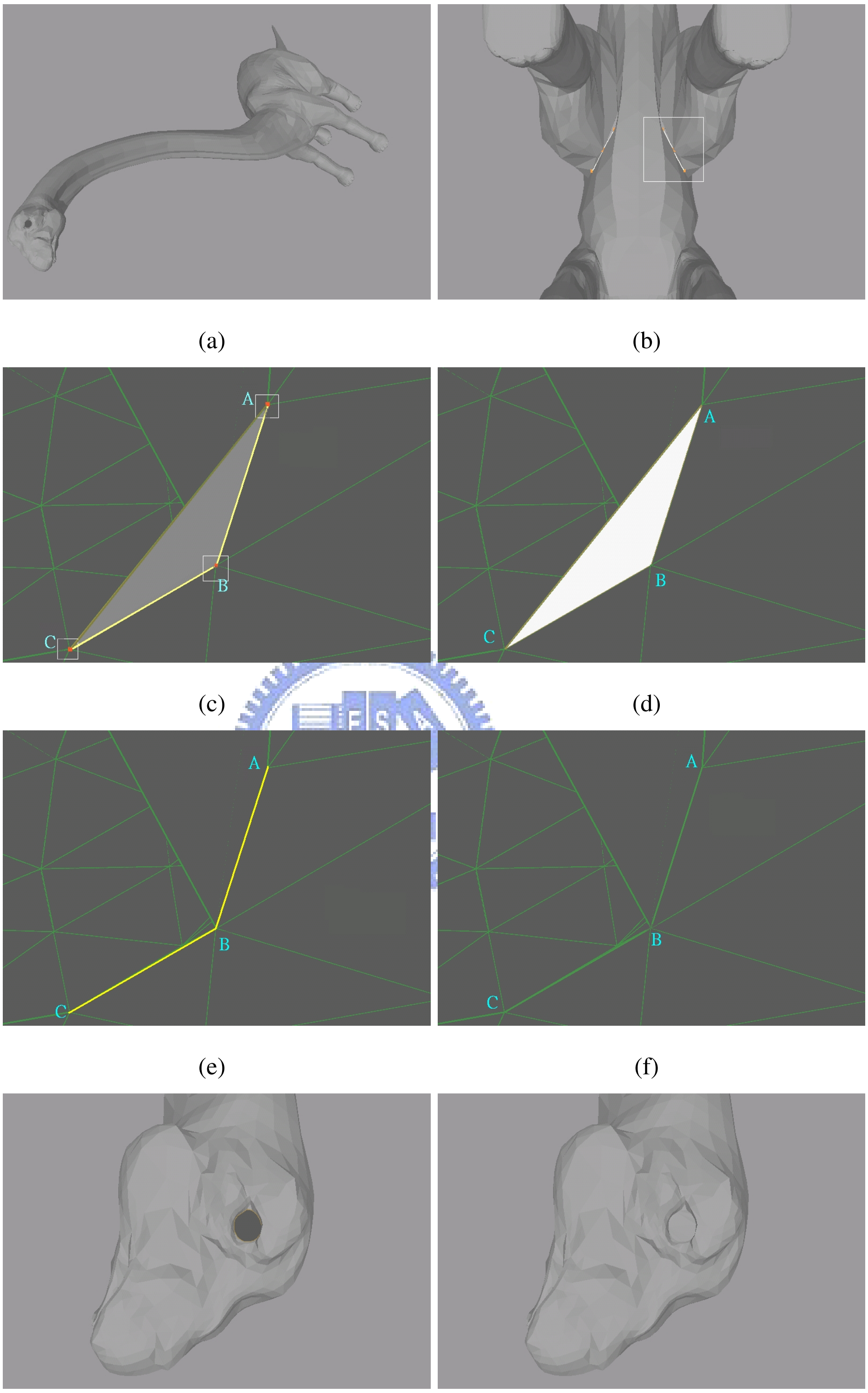
The second set of experiments that we conducted was on a crenellation model. As shown in Fig. 4.2(a), we can see there are two connected components on the mesh model, one is white triangular face and the other is the primary part of the model. In the cutting process, we applied two algorithms that had different definitions on connected component to check the result. With the first definition, the original definition of connected component, we were able to derive the result shown in Fig. 4.2(b). With the result, we remove the white part from the mesh since its area and face number were both below 1%. As to the results obtained by applying the second definition, two faces are in the same connected component if the edge they share is neither a singular edge nor a non-boundary edge, we got a perfect hole in which all the boundary faces have zero or only one boundary edge. Now we have two sets of cutting results and the next step is hole-filling. We first applied the hole-filling algorithm on the cutting result shown in Fig. 4.2(b), and the result is shown in Fig. 4.2(d). The objective of a filling process is to find a minimum-area patch. After the filling process, we found that the resultant mesh contains many topological singularities. Fig. 4.2(e) shows another set of results also derived from Fig. 4.2(b). The main difference with the previous set of experiments was its filling strategy. The filling strategy adopted here was to find minimum-area and minimum-angle patch. Obviously, the results obtained in this set of experiments were better than 4.2(d) and there was no topological problem anymore. Fig. 4.2(f) illustrates the results obtained by applying filling algorithm on Fig. 4.2(c). The strategy adopted was the minimum-area approach. We can make a comparison between the filling results of Fig. 4.2(d) and Fig. 4.2(f). They were both obtained by applying the same filling strategy, but the results shown in Fig. 4.2(f) apparently didn't contain any topological singularities. That was because we have already removed the non-boundary edges, so the triangulation algorithm never found a vertex pair and then produced new topological problems. Fig. 4.2(g) and Fig. 4.2(h) were results obtained by applying different filling strategies. Although the results obtained using this input model were not as good as the one shown in Fig. 4.2(e), we still say our goal is achieved because the mesh model does not have topological singularities and the hole is filled after the cutting and filling process.

Another example was a bunny mesh model that had many holes inside. In this experiment, we used the original definition and our own definition for the cutting process, and in the filling process we applied the minimum-area and the minimum-angle triangulation algorithm. Fig. 3(c) and Fig. 3(e) show the results after executing the cutting algorithm. Let's see the white squares in Fig. 3(c). When we applied the original definition, the non-boundary edges still existed in the mesh after executing the cutting process. Then after applying the filling process, we can find a triangular face bounded by a white square that contains one singular edge and two singular vertices (in Fin. 3(d)). Fig. 3(f) is the better result chosen from the two sets of experiments after applying all processes. We also draw the solid-outline to show the triangulation result. It is possible to conduct some refinement operations such as smoothing or subdivision to make the results better. However, we did not perform the above mentioned operations in our fixing method. The thing that we most concerned was to remove topological singularities and to fill holes when a triangular mesh came in.

Basically, the cutting process is high-level operation, and the issues needed to be addressed were clearly defined. The effect of cutting is that it will make the mesh split. Therefore, if there are many broken parts existing in a resultant mesh is not a good result. Under the above situations, some stitching processes could be employed to operate on the broken mesh to solve this problem. However, the stitching issue was not the major concern in our process. Thus, we simply used a selecting process to remove the small parts which belonged to a mesh.

Although our method is useful in removing topological singularities and in filling holes, there are still some special mesh models that cannot be fixed. We can easily identify two example mesh models that cannot be repaired by our method because the faces of their mesh models are almost disconnected components. For example, Fig. 4.4 shows two mesh models that cannot be solved by our method. The mesh model shown in Fig. 4.4(a) does not have topological problems, but has many holes inside. In this case, almost every face has a hole. The castle mesh has 26540 vertices and 13114 triangles. According to our filling method, we first identified the boundary edge and group them into sets of holes. The second step of our approach was triangulation algorithm so as to find a minimum-weight patch for every hole. Finally, we derived a resultant mesh with 26450 vertices and 26228 triangles. It is apparent that we have doubled the number of faces after hole-filling process. In fact, the selected castle model was a special model that our method could not deal with. And the problem of Fig. 4.4(b) was exactly with Fig. 4.4(a).

Self-intersection was another issue because it might cause problems when the hole-filling algorithm was applied. Fig. 4.5 shows an angel model that is a typical case which might cause the self-intersection problem. The self-intersection effect appeared in the case of Fig. 4.5(d). Although we have generated a mesh without topological singularities and holes, we would consider the result not good. The failure cases shown in Fig. 4.4 and Fig. 4.5 were both caused by the same reason, that is, the definition of a hole. Since the original definition of a hole did not take the geometry factor into account, it is possible that the intersection problem emerged.



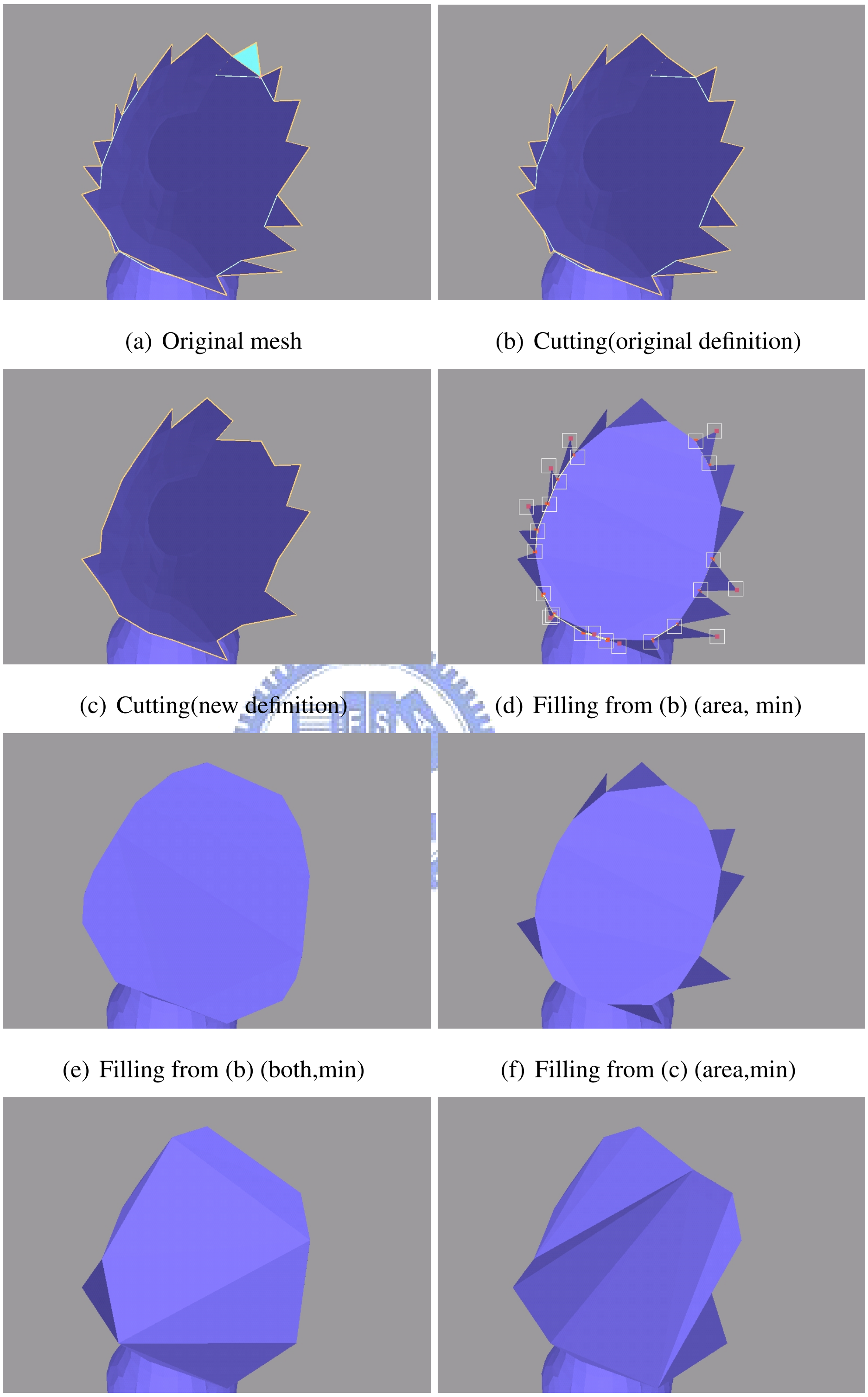
(a) (b)

(c) (d)

(e) (f)

(g) (h)

Figure 4.1: Dinosaur model



(a) Original mesh

(b) Cutting(original definition)

(c) Cutting(new definition)

(d) Filling from (b) (area, min)

(e) Filling from (b) (both,min)

(f) Filling from (c) (area,min)

(g) Filling from (c) (both,min)

(h) Filling from (c) (both,max)

Figure 4.2: Dinosaur model with crenellation hole