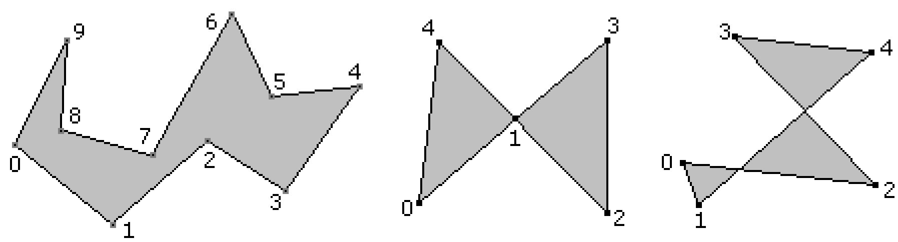
Triangulation by Ear Clipping

David Eberly
Magic Software, Inc.
http://www.magic-software.com

Created: November 18, 2002
Modified: September 4, 2003 (Removed incorrect statement about adjacent ear remaining an ear.)

1 Introduction

A classic problem in computer graphics is to decompose a simple polygon into a collection of triangles whose vertices are only those of the simple polygon. By definition, a simple polygon is an ordered sequence of n points, $\vec{V}_0$ through $\vec{V}_{n-1}$. Consecutive vertices are connected by an edge $\langle \vec{V}_i, \vec{V}_{i+1} \rangle$, $0 \le i \le n-2$, and an edge $\langle \vec{V}_{n-1}, \vec{V}_0 \rangle$ connects the first and last points. Each vertex shares exactly two edges. The only place where edges are allowed to intersect are at the vertices. A typical simple polygon is shown in Figure 1



simple nonsimple nonsimple

Figure 1. The left polygon is simple. The middle polygon is not simple since vertex 1 is shared by more than two edges. The right polygon is not simple since the edge connecting vertices 1 and 4 is intersected by other edges at points that are not vertices.

If a polygon is simple, as you traverse the edges the interior bounded region is always to one side. I assume that the polygon is counterclockwise ordered so that as you traverse the edges, the interior is to your left. The vertex indices in Figure 1 for the simple polygon correspond to a counterclockwise order.

The decomposition of a simple polygon into triangles is called a *triangulation* of the polygon. A fact from computational geometry is that any triangulation of a simple polygon of n vertices always has $n-2$ triangles. Various algorithms have been developed for triangulation, each characterized by its asymptotic order as n grows without bound. The simplest algorithm, called *ear clipping*, is the algorithm described in this document. The order is $O(n^2)$. Algorithms with better asymptotic order exist, but are more difficult to implement. Horizontal decomposition into trapezoids followed by identification of monotone polygons that are themselves triangulated is an $O(n \log n)$ algorithm [1, 3]. An improvement using an incremental randomized algorithm produces an $O(n \log^* n)$ where $\log^* n$ is the iterated logarithm function [5]. This function is effectively a constant for very large n that you would see in practice, so for all practical purposes the randomized method is linear time. An $O(n)$ algorithm exists in theory [2], but is quite complicated. It appears that no implementation is publicly available.

2 Ear Clipping

An *ear of a polygon* is a triangle formed by three consecutive vertices $\vec{V}_{i_0}$, $\vec{V}_{i_1}$, and $\vec{V}_{i_2}$ for which no other vertices of the polygon are inside the triangle. In the computational geometry jargon, the line segment between $\vec{V}_{i_0}$ and $\vec{V}_{i_2}$ is a *diagonal* of the polygon. The vertex $\vec{V}_{i_1}$ is called the *ear tip*. A triangle consists of a single ear, although you can place the ear tip at any of the three vertices. A polygon of four or more sides always has at least two nonoverlapping ears [4]. This suggests a recursive approach to the triangulation. If you can locate an ear in a polygon with $n \geq 4$ vertices and remove it, you have a polygon of $n-1$ vertices and can repeat the process. A straightforward implementation of this will lead to an $O(n^3)$ algorithm.

With some careful attention to details, the ear clipping can be done in $O(n^2)$ time. The first step is to store the polygon as a doubly–linked list so that you can quickly remove ear tips. Construction of this list is an $O(n)$ process. The second step is to iterate over the vertices and find the ears. For each vertex $\vec{V}_i$ and corresponding triangle $\langle \vec{V}_{i-1}, \vec{V}_i, \vec{V}_{i+1} \rangle$ (indexing is modulo n, so $\vec{V}_n = \vec{V}_0$ and $\vec{V}_{-1} = \vec{V}_{n-1}$), test all other vertices to see if any are inside the triangle. If none are inside, you have an ear. If at least one is inside, you do not have an ear. The actual implementation I provide tries to make this somewhat more efficient. It is sufficient to consider only reflex vertices in the triangle containment test. A *reflex vertex* is one for which the interior angle formed by the two edges sharing it is larger than 180 degrees. A *convex vertex* is one for which the interior angle is smaller than 180 degrees. The data structure for the polygon maintains four doubly–linked lists simultaneously, using an array for storage rather than dynamically allocating and deallocating memory in a standard list data structure. The vertices of the polygon are stored in a cyclical list, the convex vertices are stored in a linear list, the reflex vertices are stored in a linear list, and the ear tips are stored in a cyclical list.

Once the initial lists for reflex vertices and ears are constructed, the ears are removed one at a time. If $\vec{V}_i$ is an ear that is removed, then the edge configuration at the adjacent vertices $\vec{V}_{i-1}$ and $\vec{V}_{i+1}$ can change. If an adjacent vertex is convex, a quick sketch will convince you that it remains convex. If an adjacent vertex is an ear, it does not necessarily remain an ear after $\vec{V}_i$ is removed. If the adjacent vertex is reflex, it is possible that it becomes convex and, possibly, an ear. Thus, after the removal of $\vec{V}_i$, if an adjacent vertex is convex you must test if it is an ear by iterating over over the reflex vertices and testing for containment in the triangle of that vertex. There are $O(n)$ ears. Each update of an adjacent vertex involves an earness test, a process that is $O(n)$ per update. Thus, the total removal process is $O(n^2)$.

The following example, using the simple polygon of Figure 1, shows at a high level how the algorithm is structured. The initial list of convex vertices is $C = \{0, 1, 3, 4, 6, 9\}$, the initial list of reflex vertices is $R = \{2, 5, 7, 8\}$, and the initial list of ears is $E = \{3, 4, 6, 9\}$. The ear at vertex 3 is removed, so the first triangle in the triangulation is $T_0 = \langle 2, 3, 4 \rangle$. Figure 2 shows the reduced polygon with the new edge drawn in blue.

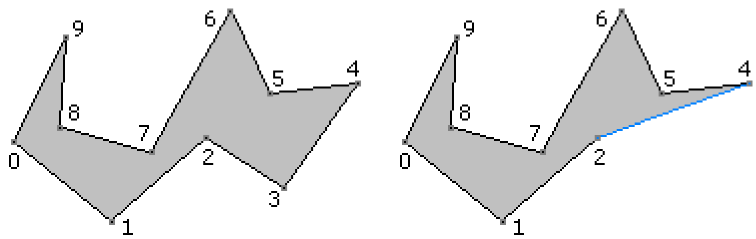


Figure 2. The right polygon shows the ear $\langle 2, 3, 4\rangle$ removed from the left polygon.

The adjacent vertex 2 was reflex and still is. The adjacent vertex 4 was an ear and remains so. The reflex list R remains unchanged, but the edge list is now $E = \{4, 6, 9\}$ (removed vertex 3).

The ear at vertex 4 is removed. The next triangle in the triangulation is $T_1 = \langle 2, 4, 5\rangle$. Figure 3 shows the reduced polygon with the new edge drawn in blue.

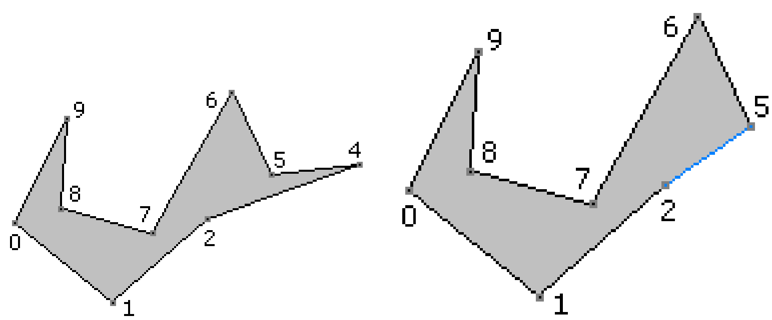


Figure 3. The right polygon shows the ear $\langle 2, 3, 4\rangle$ removed from the left polygon.

The adjacent vertex 2 was reflex and still is. The adjacent vertex 5 was reflex, but is now convex. It is tested and found to be an ear. The vertex is removed from the reflex list, $R = \{2, 7, 8\}$. It is also added to the ear list, $E = \{5, 6, 9\}$ (added vertex 5, removed vertex 4).

The ear at vertex 5 is removed. The next triangle in the triangulation is $T_2 = \langle 2, 5, 6\rangle$. Figure 4 shows the reduced polygon with the new edge drawn in blue.

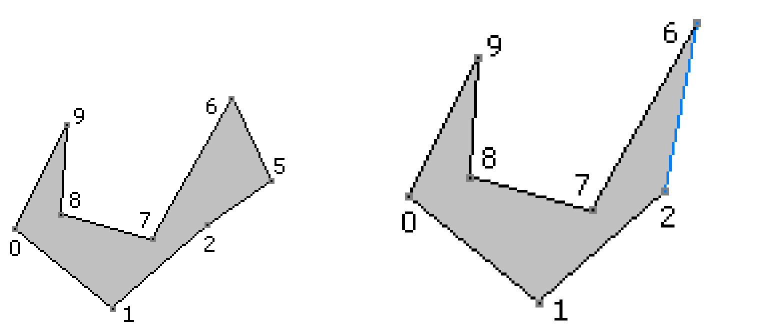


Figure 4. The right polygon shows the ear $\langle 2, 5, 6\rangle$ removed from the left polygon.

The adjacent vertex 2 was reflex, but is now convex. Although difficult to tell from the figure, the vertex 7 is inside the triangle $\langle 1, 2, 6\rangle$, so vertex 2 is not an ear. The adjacent vertex 6 was an ear and remains so. The new reflex list is $R = \{7, 8\}$ (removed vertex 2) and the new ear list is $E = \{6, 9\}$ (removed vertex 5).

The ear at vertex 6 is removed. The next triangle in the triangulation is $T_3 = \langle 2, 6, 7\rangle$. Figure 5 shows the reduced polygon with the new edge drawn in blue.

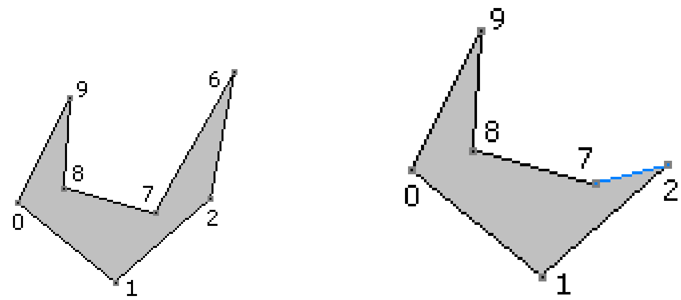


Figure 5. The right polygon shows the ear $\langle 2, 6, 7\rangle$ removed from the left polygon.

The adjacent vertex 2 was convex and remains so. It was not an ear, but now it becomes one. The adjacent vertex 7 was reflex and remains so. The reflect list R does not change, but the new ear list is $E = \{9, 2\}$ (added vertex 2, removed vertex 6). The ear list is written this way because the new ear is added first before the old ear is removed. Before removing the old ear, it is still considered to be first in the list (well, the list is cyclical). The remove operation sets the first item to be that next value to the old ear rather than the previous value.

The ear at vertex 9 is removed. The next triangle in the triangulation is $T_4 = \langle 8, 9, 0\rangle$. Figure 6 shows the reduced polygon with the new edge drawn in blue.

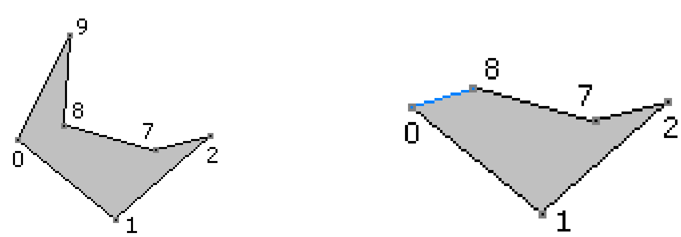


Figure 6. The right polygon shows the ear $\langle 8, 9, 0\rangle$ removed from the left polygon.

The adjacent vertex 8 was reflex, but is now convex and in fact an ear. The adjacent vertex 0 was convex and remains so. It was not an ear but has become one. The new reflex list is $R = \{7\}$ and the new ear list is $E = \{0, 2, 8\}$ (added 8, added 0, removed 9, the order shown is what the program produces).

The ear at vertex 0 is removed. The next triangle in the triangulation is $T_5 = \langle 8, 0, 1\rangle$. Figure 7 shows the reduced polygon with the new edge drawn in blue.

Figure 7. The right polygon shows the ear $\langle 8, 0, 1 \rangle$ removed from the left polygon.

Both adjacent vertices 8 and 1 were convex and remain so. Vertex 8 remains an ear and vertex 1 remains not an ear. The reflex list remains unchanged. The new ear list is $E = \{2, 8\}$ (removed vertex 0).

Finally, the ear at vertex 2 is removed. The next triangle in the triangulation is $T_6 = \langle 1, 2, 7 \rangle$. Figure 8 shows the reduced polygon with the new edge drawn in blue.

Figure 8. The right polygon shows the ear $\langle 1, 2, 7 \rangle$ removed from the left polygon.

At this time there is no need to update the reflect or ear lists since we detect that only three vertices remain. The last triangle in the triangulation is $T_7 = \langle 7, 8, 1 \rangle$. The full triangulation is show in Figure 9.

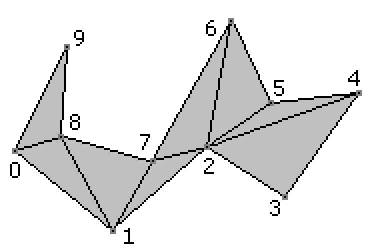


Figure 9. The full triangulation of the original polygon.

References

[1] B. Chazelle and J. Incerpi, *Triangulation and shape complexity*, ACM Trans. on Graphics, vol. 3, pp. 135–152, 1984.

[2] B. Chazelle, *Triangulating a simple polygon in linear time*, Discrete Comput. Geom., vol. 6, pp. 485–524, 1991.

[3] A. Fournier and D.Y. Montuno, *Triangulating simple polygons and equivalent problems*, ACM Trans. on Graphics, vol. 3, pp. 153–174, 1984.

[4] G.H. Meisters, *Polygons have ears*, Amer. Math. Monthly, vol. 82, pp. 648–651, 1975.

[5] R. Seidel, *A simple and fast incremental randomized algorithm for computing trapezoidal decompositions and for triangulating polygons*, Computational Geometry: Theory and Applications, vol. 1, no. 1, pp. 51–64, 1991.