Chapter 4
Graphics Output Primitives

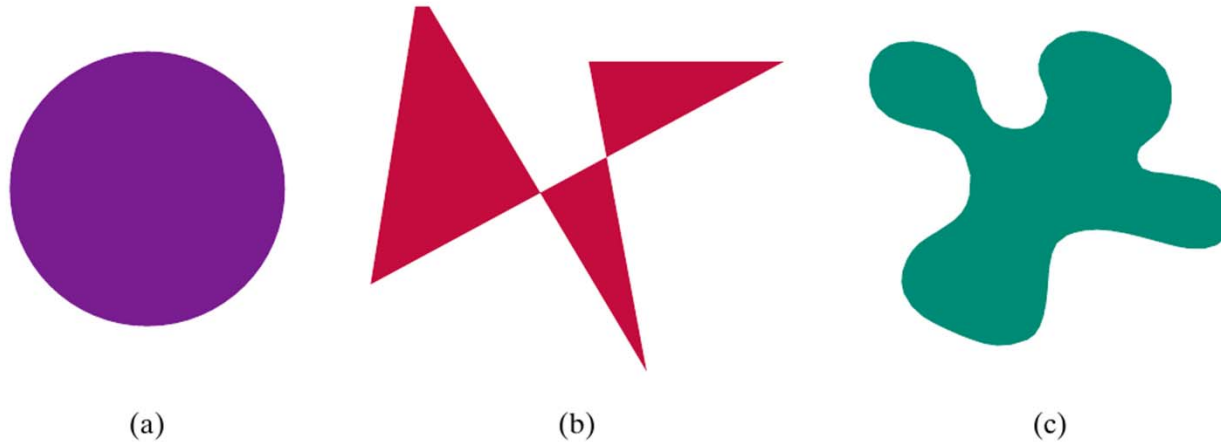Myung-Soo Kim

Seoul National University
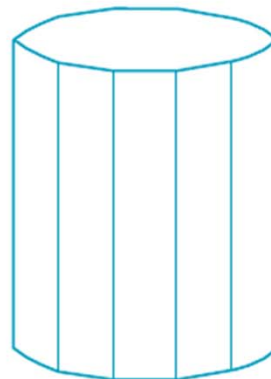
http://cse.snu.ac.kr/mskim

http://3map.snu.ac.kr

Fill–Area Primitives

(a)　　　　(b)　　　　(c)

FIGURE 3–40 Solid-color fill areas specified with various boundaries. (a) A circular fill region. (b) A fill area bounded by a closed polyline. (c) A filled area specified with an irregular curved boundary.
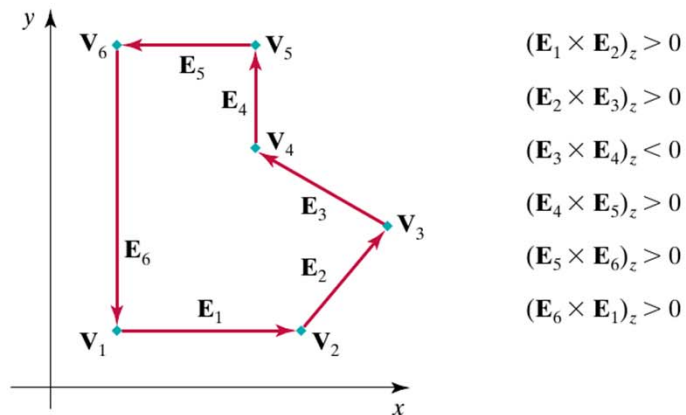

FIGURE 3–41 Wire-frame representation for a cylinder, showing only the front (visible) faces of the polygon mesh used to approximate the surfaces.

Polygon Fill-Areas
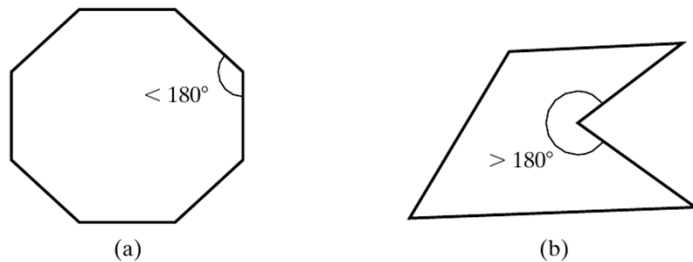
FIGURE 3-42   A convex polygon (a), and a concave polygon (b).

$$(E_1 \times E_2)_z > 0$$
$$(E_2 \times E_3)_z > 0$$
$$(E_3 \times E_4)_z < 0$$
$$(E_4 \times E_5)_z > 0$$
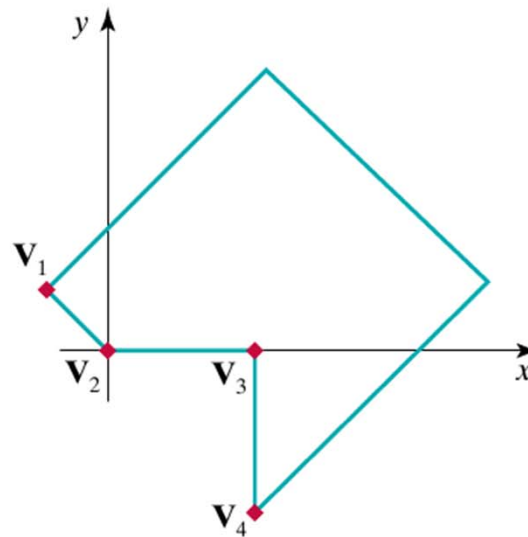$$(E_5 \times E_6)_z > 0$$
$$(E_6 \times E_1)_z > 0$$

FIGURE 3-43   Identifying a concave polygon by calculating cross products of successive pairs of edge vectors.

삼각형이나 convex polygon들로 분할하는 최적화된 알고리즘들은 Computational Geometry 분야에서 아주 많이 연구되었다. 여기서는 non-convex한 vertex를 간단하게 찾는 알고리즘을 소개한다.

Splitting Concave Polygons
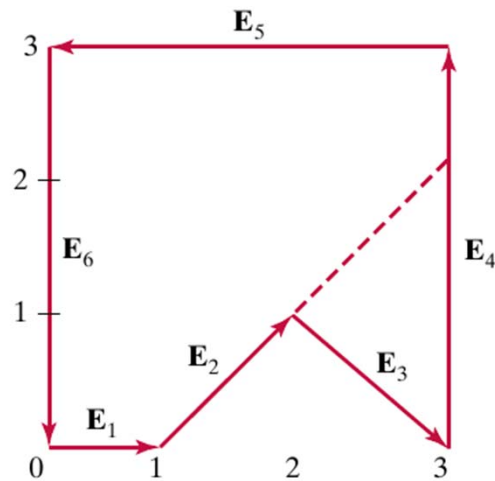
FIGURE 3-44 Splitting a concave polygon using the vector method.

FIGURE 3-45 Splitting a concave polygon using the rotational method. After moving V_2 to the coordinate origin and rotating V_3 onto the x axis, we find that V_4 is below the x axis. So we split the polygon along the line of $\overline{V_2 V_3}$, which is the x axis.

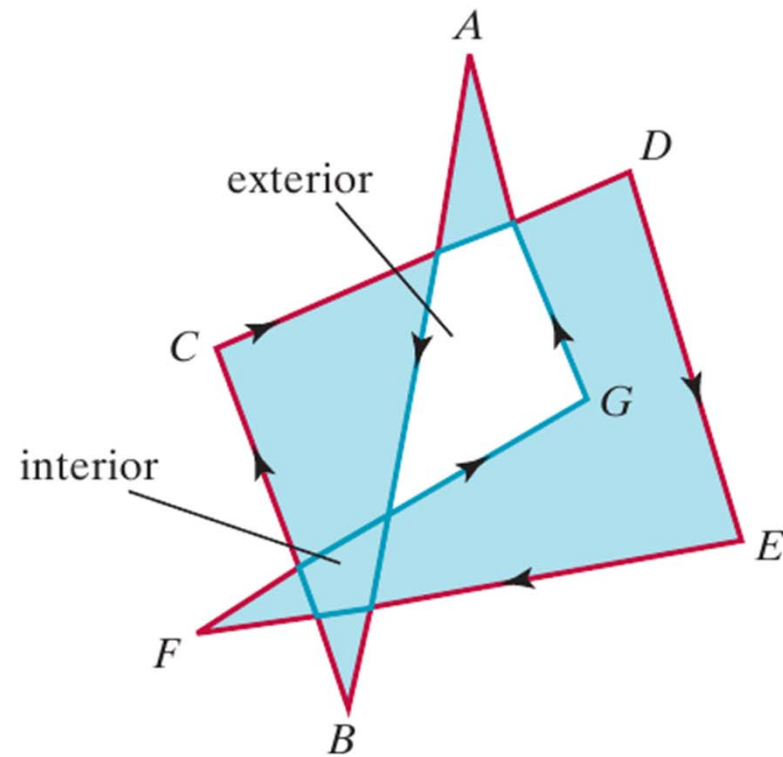오른쪽의 알고리즘은 전체의 다각형을 변환하는 과정이 비효율적이다.

E2 를 지나는 직선을 구하고 다음에 연속적으로 나오는 vertex들을 직선의 식에 넣어보면서 직선방정식의 부호가 바뀌는 부분에서 두 직선의 교점을 구하면 E4가 분할되는 교점의 좌표를 쉽게 구할 수 있다.
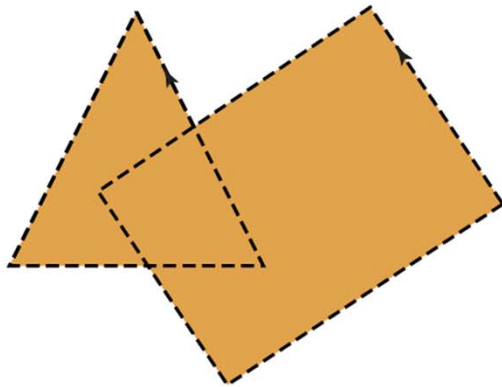
Inside-Outside Tests
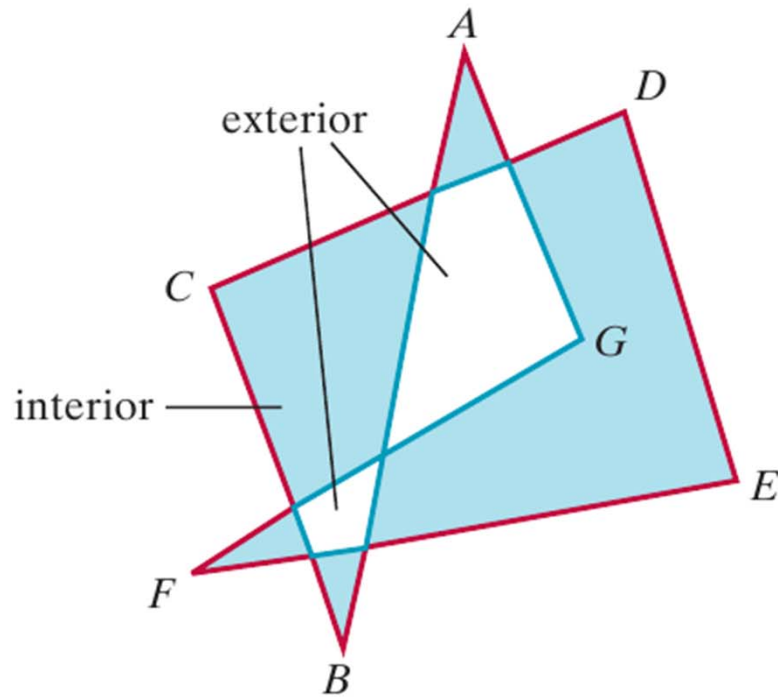
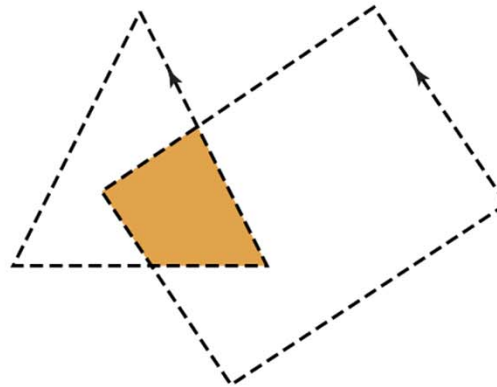
Odd-Even Rule
(a)

Nonzero Winding-Number Rule
(b)

그림(b)의 진행방향을 반대로 하여 시계반대방향으로 진행하는 것이 주로
사용된다. 다음 페이지의 예에서는 시계반대방향으로 진행한다.

Boolean Ops using Winding #s

FIGURE 3–47 A fill area defined as a region that has a positive value for the winding number. This fill area is the union of two regions, each with a counterclockwise border direction.

FIGURE 3–48 A fill area defined as a region with a winding number greater than 1. This fill area is the intersection of two regions, each with a counterclockwise border direction.
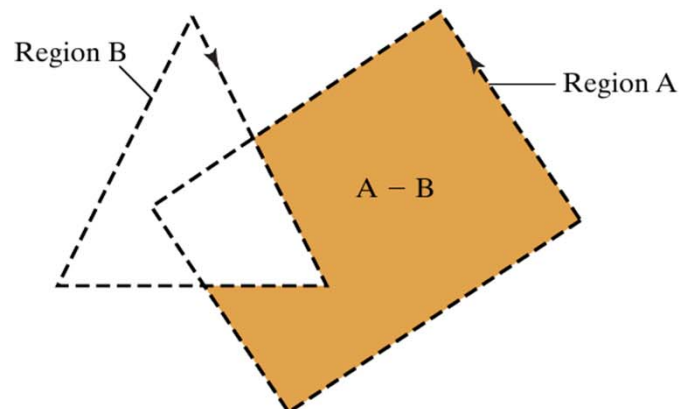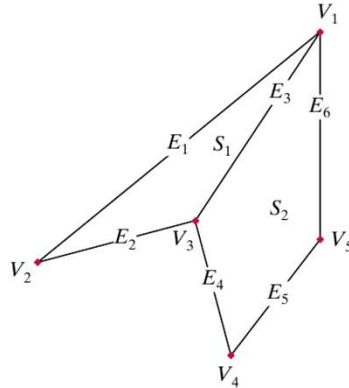
Region B

Region A

A − B

FIGURE 3–49 A fill area defined as a region with a positive value for the winding number. This fill area is the difference, A − B, of two regions, where region A has a positive border direction (counterclockwise) and region B has a negative border direction (clockwise).

Polygon Tables

$$
\begin{array}{ll}
E_1: & V_1, V_2, S_1 \\
E_2: & V_2, V_3, S_1 \\
E_3: & V_3, V_1, S_1, S_2 \\
E_4: & V_3, V_4, S_2 \\
E_5: & V_4, V_5, S_2 \\
E_6: & V_5, V_1, S_2
\end{array}
$$

FIGURE 3–51 Edge table expanded to include pointers into the surface-facet table.

Listing the geometric data in three tables, as in Fig. 3-50, provides a convenient reference to the individual components (vertices, edges, and surface facets) for each object. Also, the object can be displayed efficiently by using data from the edge table to identify polygon boundaries. An alternative arrangement is to use just two tables: a vertex table and a surface-facet table. But this scheme is less convenient, and some edges could get drawn twice in a wire-frame display. Another possibility is to use only a surface-facet table, but this duplicates coordinate information, since explicit coordinate values are listed for each vertex in each polygon facet. Also the relationship between edges and facets would have to be reconstructed from the vertex listings in the surface-facet table.

We can add extra information to the data tables of Fig. 3-50 for faster information extraction. For instance, we could expand the edge table to include forward pointers into the surface-facet table so that a common edge between polygons could be identified more rapidly (Fig. 3-51). This is particularly useful for rendering procedures that must vary surface shading smoothly across the edges from one polygon to the next. Similarly, the vertex table could be expanded to reference corresponding edges, for faster information retrieval.

VERTEX TABLE		EDGE TABLE		SURFACE-FACET TABLE	
V_1:	x_1, y_1, z_1	E_1:	V_1, V_2	S_1:	E_1, E_2, E_3
V_2:	x_2, y_2, z_2	E_2:	V_2, V_3	S_2:	E_3, E_4, E_5, E_6
V_3:	x_3, y_3, z_3	E_3:	V_3, V_1		
V_4:	x_4, y_4, z_4	E_4:	V_3, V_4		
V_5:	x_5, y_5, z_5	E_5:	V_4, V_5		
		E_6:	V_5, V_1		

삼각형의 개수가 많아지면, vertex와 edge의 개수도 증가하고, 따라서 vertex와 edge들의 index를 표시하는 정보의 양이 크게 증가한다. 이는 심각한 문제이다.

Plane Equations

$$A x + B y + C z + D = 0 \tag{3-59}$$

$$(A/D)x_k + (B/D)y_k + (C/D)z_k = -1, \qquad k = 1, 2, 3 \tag{3-60}$$

Cramer's rule, as

$$A = \begin{vmatrix} 1 & y_1 & z_1 \\ 1 & y_2 & z_2 \\ 1 & y_3 & z_3 \end{vmatrix} \qquad B = \begin{vmatrix} x_1 & 1 & z_1 \\ x_2 & 1 & z_2 \\ x_3 & 1 & z_3 \end{vmatrix}$$

$$C = \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix} \qquad D = -\begin{vmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{vmatrix} \tag{3-61}$$

Expanding the determinants, we can write the calculations for the plane coefficients in the form

$$\begin{aligned} A &= y_1(z_2 - z_3) + y_2(z_3 - z_1) + y_3(z_1 - z_2) \\ B &= z_1(x_2 - x_3) + z_2(x_3 - x_1) + z_3(x_1 - x_2) \\ C &= x_1(y_2 - y_3) + x_2(y_3 - y_1) + x_3(y_1 - y_2) \\ D &= -x_1(y_2 z_3 - y_3 z_2) - x_2(y_3 z_1 - y_1 z_3) - x_3(y_1 z_2 - y_2 z_1) \end{aligned} \tag{3-62}$$

Front and Back Polygon Faces

N=(A, B, C)
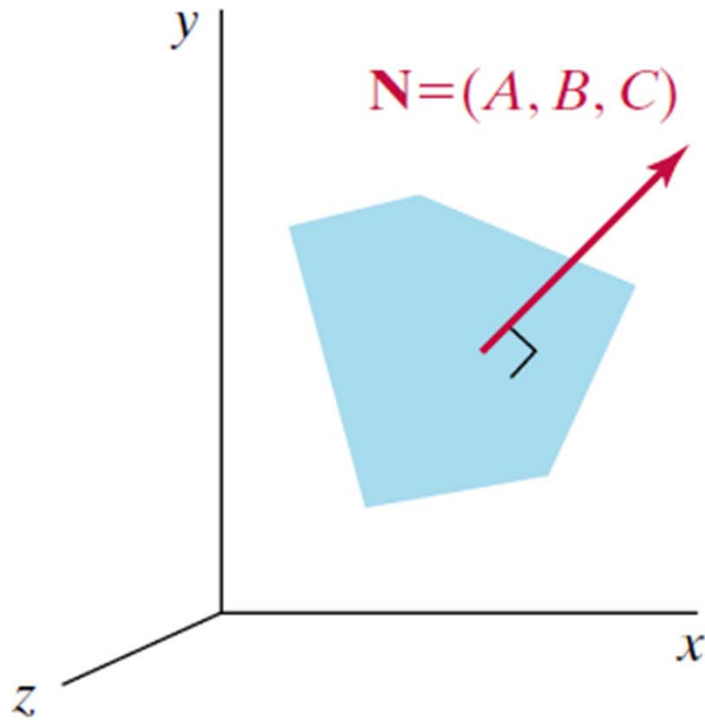
FIGURE 3–53 The normal vector **N** for a plane described with the equation $Ax + By + Cz + D = 0$ is perpendicular to the plane and has Cartesian components (A, B, C).

if $Ax + By + Cz + D < 0$, behind the plane

if $Ax + By + Cz + D > 0$, in front of the plane