

컴퓨터 그래픽스

4. 2차원 그래픽스의 윈도우와 뷰포트

2025년 2학기

학습 내용

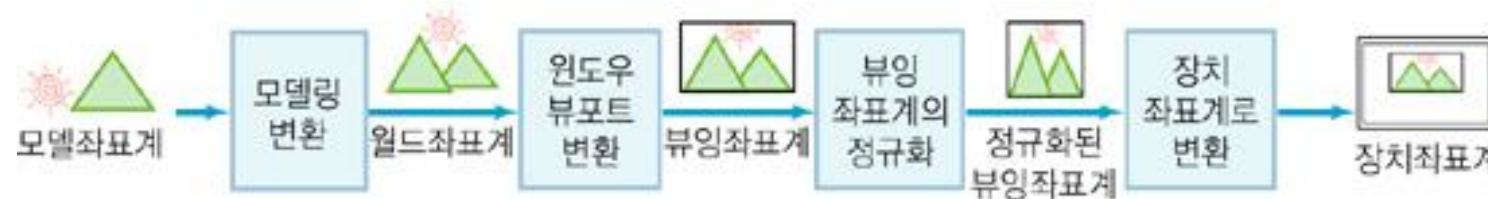
- 원도우와 뷰포트
 - 2차원 뷰잉 파이프라인: 원도우, 뷰포트
 - 클리핑 알고리즘

윈도우와 뷰포트

• 2차원 그래픽스의 뷰잉 파이프라인

- 뷰잉 파이프라인: 그림을 출력장치에 나타내야 할 때, 특정 부분만을 출력시키기도 하는데 이때 적절한 변환을 적용해야 하는데, 이 과정을 **뷰잉 파이프라인**이라고 한다.

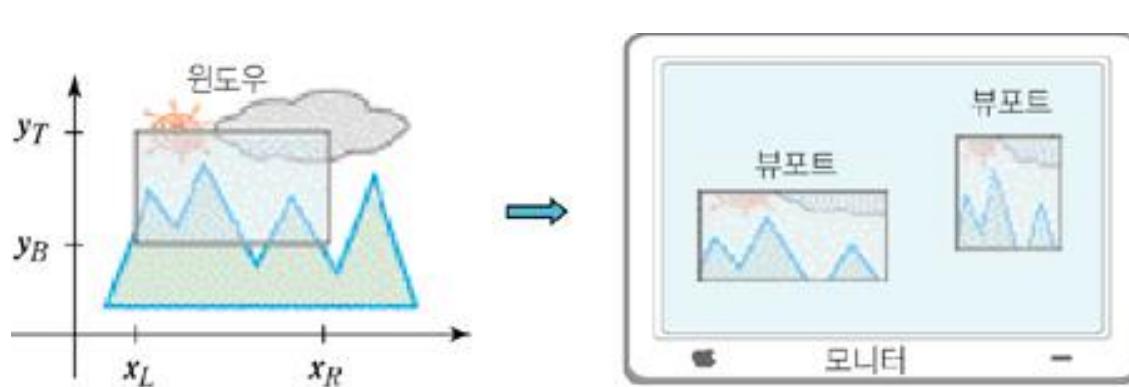
- 뷰잉 파이프라인 과정:



- 모델 좌표계: 개별 객체를 표현하기 위해 사용되는 좌표계
- 월드 좌표계: 각 모델 좌표계의 통합된 좌표계
- 뷰잉 좌표계: 출력장치에 출력 위치 및 크기 설정하여 뷰포트에 출력, 뷰포트 좌표계
- 정규 좌표계: 정규화된 좌표계
- 장치 좌표계: 출력하려는 장치 좌표계

윈도우와 뷰포트

- **Window**
 - 출력 장치에 표시하기 위해 선택된 세계 좌표 영역
- **Viewport**
 - 윈도우가 사상되는 출력 장치의 영역
- **윈도우-뷰포트 변환에 의한 효과**
 - Zooming 효과: zoom-in 또는 zoom-out 효과
 - Panning 효과: 카메라 각도를 돌려가면서 비디오 촬영하는 것과 같은 효과
 - Multi viewport 효과: 한번에 여러 개의 화면을 가질 수 있다.



윈도우와 뷰포트

- **윈도우-뷰포트 좌표 변환**

- (x_w, y_w) : 윈도우 내의 점

- (x_v, y_v) : 뷰포트 안의 점

- 행렬

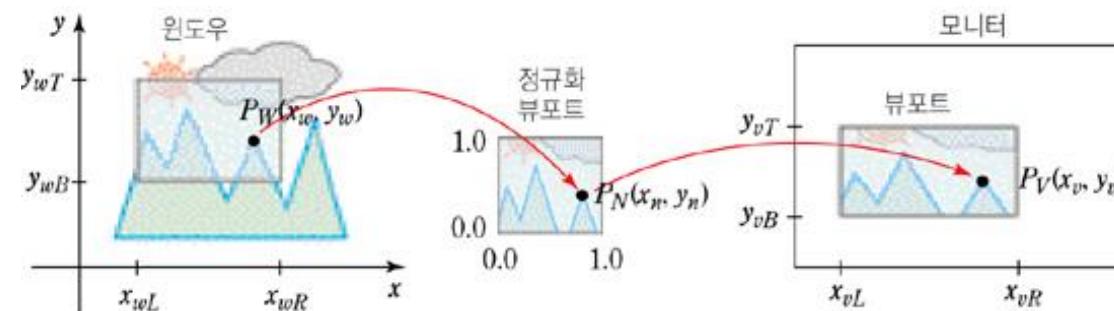
- $x_v = x_{vL} + (x_w - x_{wL})s_x,$

$$s_x = \frac{(x_{vR} - x_{vL})}{(x_{wR} - x_{wL})}$$

- $y_v = y_{vB} + (y_w - y_{wB})s_y,$

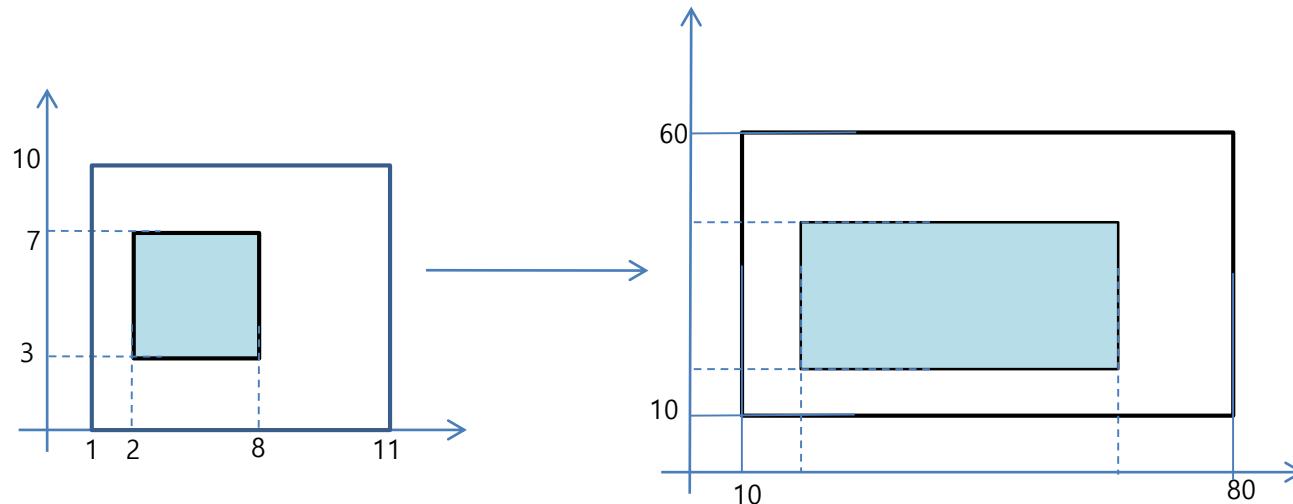
$$s_y = \frac{(y_{vT} - y_{vB})}{(y_{wT} - y_{wB})}$$

x_{wR}, x_{wL} : 윈도우의 x방향 최대값, 최소값
 y_{wT}, y_{wB} : 윈도우의 y방향 최대값, 최소값



윈도우와 뷰포트

- 예) 다음의 도형에 대하여 윈도우-뷰포트 변환이 주어졌을 때 변환 좌표 값
 - 윈도우 $(1, 0)(11, 10)$ \rightarrow 뷰포트 $(10, 10)(80, 60)$
도형 좌표: $(2, 3)(8, 7)$ 로 이루어진 사각형이 윈도우-뷰포트 변환 후 좌표값:



Clipping (클리핑)

- Clipping이란
 - 원도우-뷰포트 변환 시, 출력장치에 표시되어서는 안될 그림영역을 제거한 뒤, 나머지 그림영역을 출력화면에 나타내는 것
 - 월드 좌표 클리핑:
 - 원도우를 설정할 때 원도우 바깥 영역을 제거하여 원도우 내부 영역만 뷰포트로 맵핑 시키는 방법
 - 뷰포트 클리핑:
 - 월드 좌표계를 표현된 그림 전부를 뷰포트로 맵핑 시킨 후 뷰포트 외부에 위치한 객체나 그림의 일부를 제거하는 방법
 - 두 클리핑이 모두 결과는 같다.

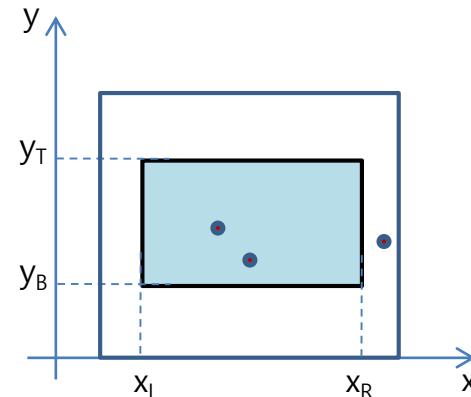
Clipping (클리핑): 점

- **점 클리핑**

- 클리핑 되는 객체가 점
- 한 점 $P(x, y)$ 는

- $x_L \leq x \leq x_R, \quad y_B \leq y \leq y_T$

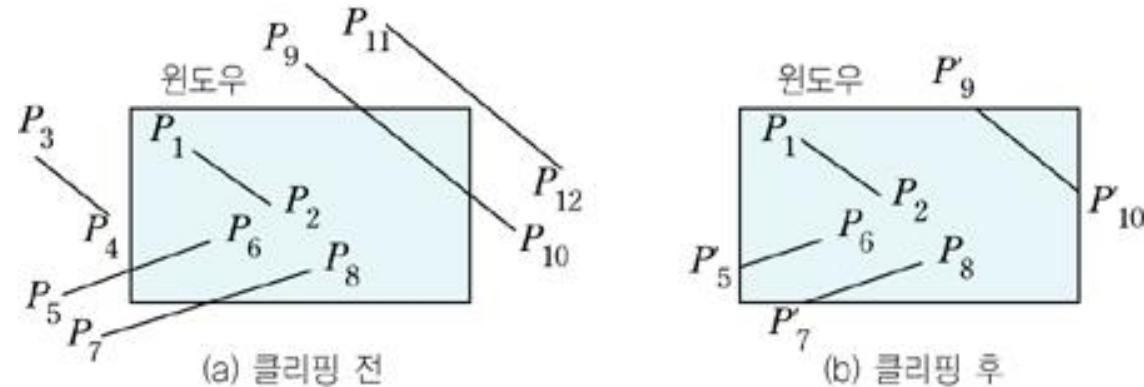
이면 그려진다.



Clipping (클리핑): 선

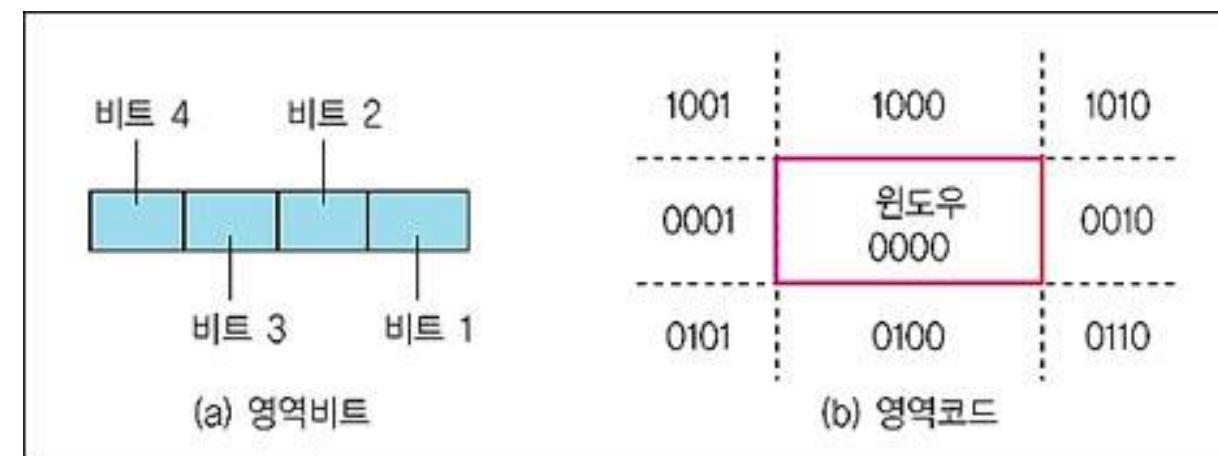
- **선 클리핑**

- 클리핑 되는 객체가 선분
- 선분에 대하여
 - 선분이 클리핑 영역의 내부 또는 외부에 완전히 포함되는가/포함되지 않는가
 - 부분적으로 속하는가
 - 속한다면 교차점은 어떻게 구하는가



Clipping (클리핑): 선 - Cohen-Sutherland 알고리즘

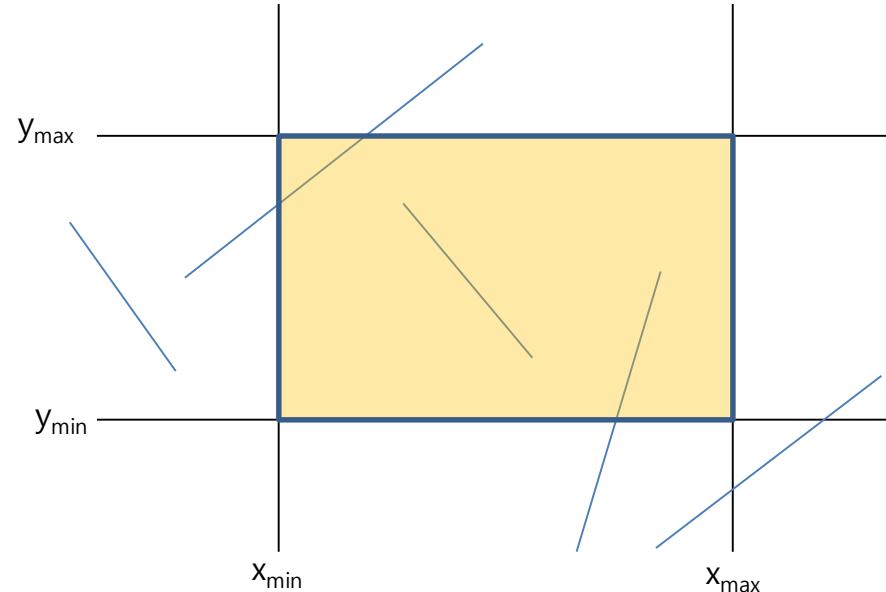
- Cohen-Sutherland 알고리즘
 - 윈도우를 중심으로 전체 그림 영역을 9개 영역으로 구분
 - 각 영역에 4비트를 사용하여 영역코드를 부여한다.
 - 비트 1: 윈도우의 왼쪽에 있으면 1
 - 비트 2: 윈도우의 오른쪽에 있으면 1
 - 비트 3: 윈도우의 아래쪽에 있으면 1
 - 비트 4: 윈도우의 위쪽에 있으면 1



Clipping (클리핑): 선 - Cohen-Sutherland 알고리즘

- 알고리즘 수행 과정

- ① 양 끝점의 코드가 모두 0000이면 →
- ② 양 끝점의 코드 중 한 쪽 코드는 0이고 다른 쪽 코드는 0이 아니면 →
- ③ 양 끝점 코드가 모두 0이 아니고, 양 끝점 코드간 AND 연산이 0이 아니면 →
- ④ 양 끝점 코드가 모두 0이 아니고, 양 끝점 코드간 AND 연산이 0이면 →



Clipping (클리핑): 선 - Cohen-Sutherland 알고리즘

- 주어진 선분에 대한 교차점 구하기

- 수직 경계: $x = x_L$ 또는 $x = x_R$

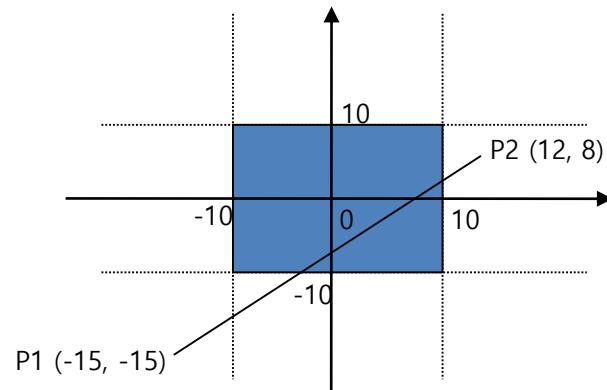
$$y = y_1 + m(x - x_1), \quad m = \frac{y_2 - y_1}{x_2 - x_1}$$

- 수평 경계: $y = y_B$ 또는 $y = y_T$

$$x = x_1 + \frac{(y - y_1)}{m}, \quad m = \frac{y_2 - y_1}{x_2 - x_1}$$

x_1, y_1 : 선분의 끝 점
 x_L, x_R, y_B, y_T : 윈도우의 경계

- 예)



Clipping (클리핑): 선 - Liang-barskey 알고리즘

- 매개변수 방정식을 이용하여 선분을 윈도우 경계에 대하여 자르는 알고리즘

- 선을 나타내는 매개변수 방정식은

- $$\begin{aligned} p(u) &= (1-u)p_1 + up_2 \\ &= p_1 + (p_2 - p_1)u \end{aligned} \quad 0 \leq u \leq 1$$

$$\text{즉, } x = x_1 + (x_2 - x_1)u$$

$$y = y_1 + (y_2 - y_1)u$$

- 끝점이 $P1 = (x_1, y_1)$ $P2 = (x_2, y_2)$ 인 선분일 때

- 매개변수 방정식 사용하여 임의의 점 $P(x, y)$ 을 표시

- $$x = x_1 + (x_2 - x_1)u \quad 0 \leq u \leq 1 \quad (x_2 - x_1 \rightarrow d_x)$$

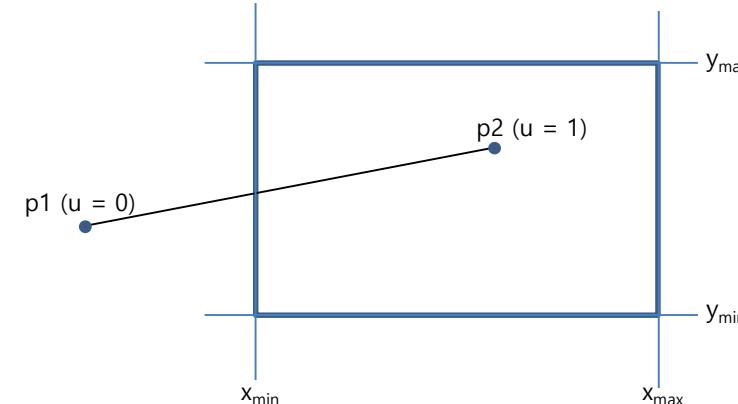
- $$y = y_1 + (y_2 - y_1)u \quad 0 \leq u \leq 1 \quad (y_2 - y_1 \rightarrow d_y)$$

- $$\bullet u = 0 \rightarrow x = x_1, y = y_1$$

- $$\bullet u = 1 \rightarrow x = x_2, y = y_2$$

- 선분 위에 있는 모든 점들은 아래의 조건을 만족

- $$x_{\min} \leq x \leq x_{\max}, \quad y_{\min} \leq y \leq y_{\max}$$



Clipping (클리핑): 선 - Liang-barskey 알고리즘

- 매개 변수 방정식으로 다시 작성하면

- $x_{\min} \leq x_1 + d_x u \leq x_{\max}$ --- ① ($d_x = x_2 - x_1$)
- $y_{\min} \leq y_1 + d_y u \leq y_{\max}$ --- ② ($d_y = y_2 - y_1$)

- ①은 다음과 같다

- » 왼쪽 가장자리에 대하여 $-d_x u < x_1 - x_{\min}$

- » 오른쪽 가장자리에 대하여 $d_x u < x_{\max} - x_1$

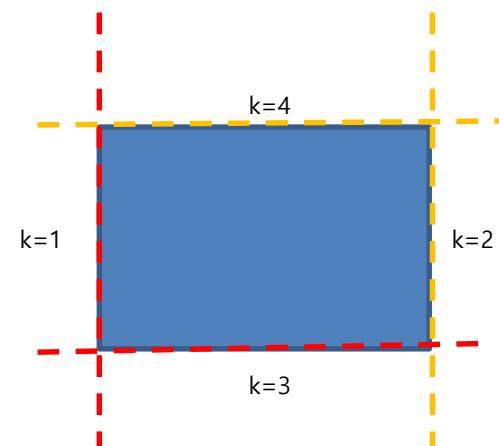
- ②도 같은 방식으로 바꿀 수 있다.

- » 아래쪽 가장자리에 대하여 $-d_y u < y_1 - y_{\min}$

- » 위쪽 가장자리에 대하여 $d_y u < y_{\max} - y_1$

- 위의 식은, $u p_k < q_k \quad k = 1, 2, 3, 4$

- $p_1 = -d_x \quad q_1 = x_1 - x_{\min}$ → left
- $p_2 = d_x \quad q_2 = x_{\max} - x_1$ → right
- $p_3 = -d_y \quad q_3 = y_1 - y_{\min}$ → bottom
- $p_4 = d_y \quad q_4 = y_{\max} - y_1$ → top



Clipping (클리핑): 선 - Liang-barskey 알고리즘

1) $p_k == 0$ 일 때,

- p_k 가 0 이면, k번째 가장자리와 평행

- $p_1 = -dx = -(x_2 - x_1) = 0 \rightarrow x_2 == x_1$

- $p_3 = -dy = -(y_2 - y_1) = 0 \rightarrow y_2 == y_1$

- $p_k = 0$ 이고, $q_k < 0$ 이면, 그 가장자리 영역 밖에 있다.

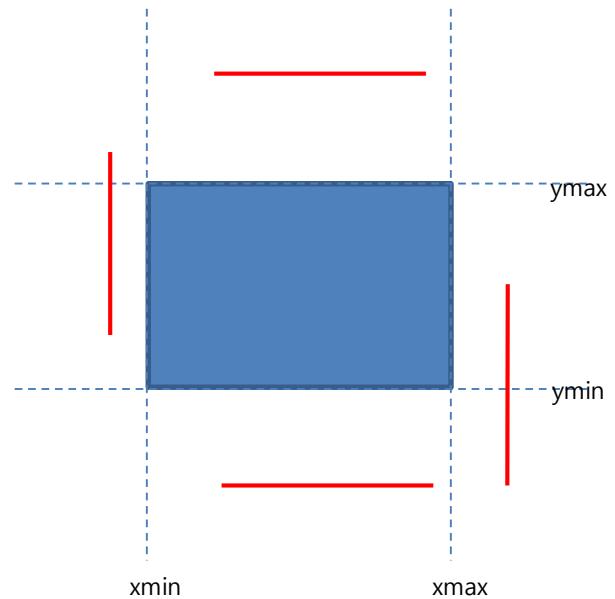
- a. $q_1 < 0 \rightarrow q_1 = (x_1 - x_{\min}) < 0 \rightarrow x_1 < x_{\min} \rightarrow$ 영역 밖에 있다.

- b. $q_2 < 0 \rightarrow q_2 = (x_{\max} - x_1) < 0 \rightarrow x_{\max} < x_1 \rightarrow$ 영역 밖에 있다.

- c. $q_3 < 0 \rightarrow q_3 = (y_1 - y_{\min}) < 0 \rightarrow y_1 < y_{\min} \rightarrow$ 영역 밖에 있다.

- d. $q_4 < 0 \rightarrow q_4 = (y_{\max} - y_1) < 0 \rightarrow y_{\max} < y_1 \rightarrow$ 영역 밖에 있다.

→ $p_k == 0$ 이고 $q_k < 0$ 이면, 영역 밖에 있으므로 안 그린다.



Clipping (클리핑): 선 - Liang-barskey 알고리즘

1) $p_k == 0$ 일 때,

- p_k 가 0 이면, k번째 가장자리와 평행

- $p_1 = -dx = -(x_2 - x_1) = 0 \rightarrow x_2 == x_1$

- $p_3 = -dy = -(y_2 - y_1) = 0 \rightarrow y_2 == y_1$

- $p_k = 0$ 이고, $q_k > 0$ 이면, 평행한 가장자리외의 가장자리와 만날 수 있다.

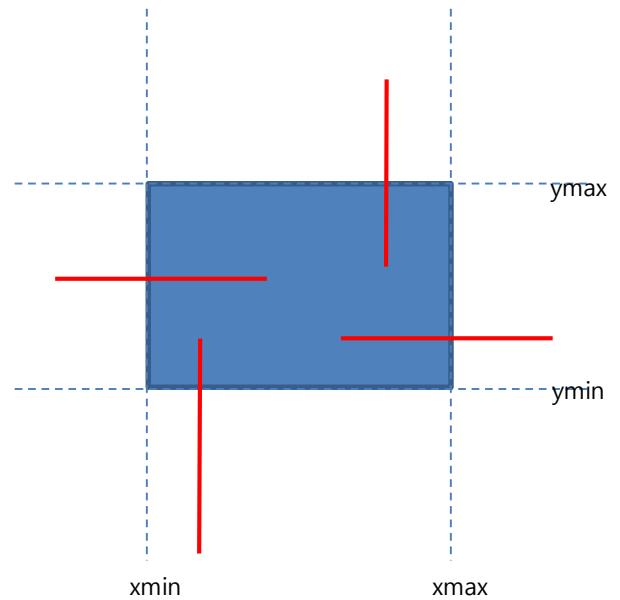
- a. $q_1 > 0 \rightarrow q_1 = (x_1 - x_{\min}) > 0 \rightarrow x_1 > x_{\min}$

- b. $q_2 > 0 \rightarrow q_2 = (x_{\max} - x_1) > 0 \rightarrow x_{\max} > x_1$

- c. $q_3 > 0 \rightarrow q_3 = (y_1 - y_{\min}) > 0 \rightarrow y_1 > y_{\min}$

- d. $q_4 > 0 \rightarrow q_4 = (y_{\max} - y_1) > 0 \rightarrow y_{\max} > y_1$

→ $p_k == 0$ 이고 $q_k < 0$ 이면, 평행한 가장자리외의 가장자리와 만날 수 있다.



Clipping (클리핑): 선 - Liang-barskey 알고리즘

2) $p_k \neq 0$ 일 때,

- $p_k \neq 0$ 이면, 선분이 경계선 중 하나와 평행하지 않다.

→ 그 선분의 무한한 연장선은 원도우의 네 개의 경계선과 어디에선가 교차한다.

- $p_k < 0$ 이면,

» $p_1 < 0 \rightarrow p_1 = -dx = -(x_2 - x_1) < 0 \rightarrow x_2 - x_1 > 0 \rightarrow x_2 > x_1$

- $p_1 < 0$ 이면 $p_2 = dx \rightarrow p_2 > 0$

» $p_3 < 0 \rightarrow p_3 = -dy = -(y_2 - y_1) < 0 \rightarrow y_2 - y_1 > 0 \rightarrow y_2 > y_1$

- $p_3 < 0$ 이면 $p_4 = dy \rightarrow p_4 > 0$

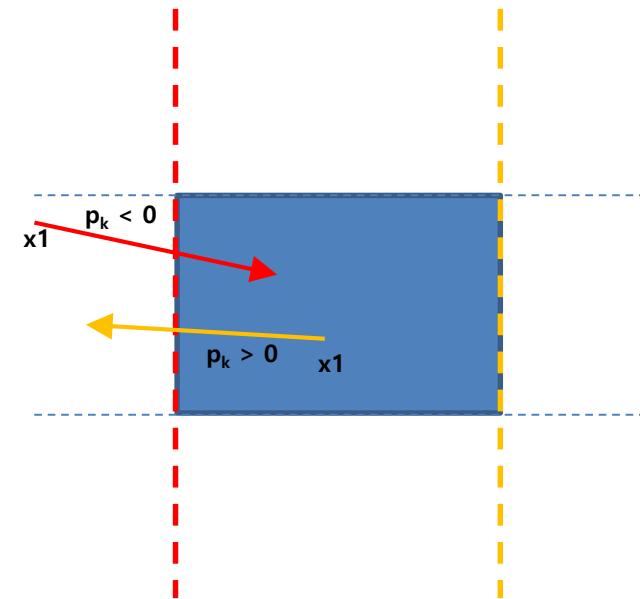
- $p_k > 0$ 이면,

» $p_1 > 0 \rightarrow p_1 = -dx = -(x_2 - x_1) > 0 \rightarrow x_2 - x_1 < 0 \rightarrow x_2 < x_1$

- $p_1 > 0$ 이면 $p_2 = dx \rightarrow p_2 < 0$

» $p_3 > 0 \rightarrow p_3 = -dy = -(y_2 - y_1) > 0 \rightarrow y_2 - y_1 < 0 \rightarrow y_2 < y_1$

- $p_3 > 0$ 이면 $p_4 = dy \rightarrow p_4 < 0$



Clipping (클리핑): 선 - Liang-barskey 알고리즘

2) $p_k \neq 0$ 일 때,

- 만약 $p_k < 0$ 이면, 직선은 밖→ 안으로 진행
- 만약 $p_k > 0$ 이면, 안 → 밖으로 진행
- 0이 아닌 p_k 에 대하여, 매개변수 u 의 값으로 가장자리와의 교차점을 찾을 수 있다. 즉,

- $u_k = q_k / p_k \quad (k = 1, 2, 3, 4)$

- $- k$ 에 대한 u 의 값에 대하여,

- $\gg p_k < 0$ 이면, $u_{start} = \max(u_k, 0)$

- $\gg p_k > 0$ 이면, $u_{end} = \min(u_k, 1)$

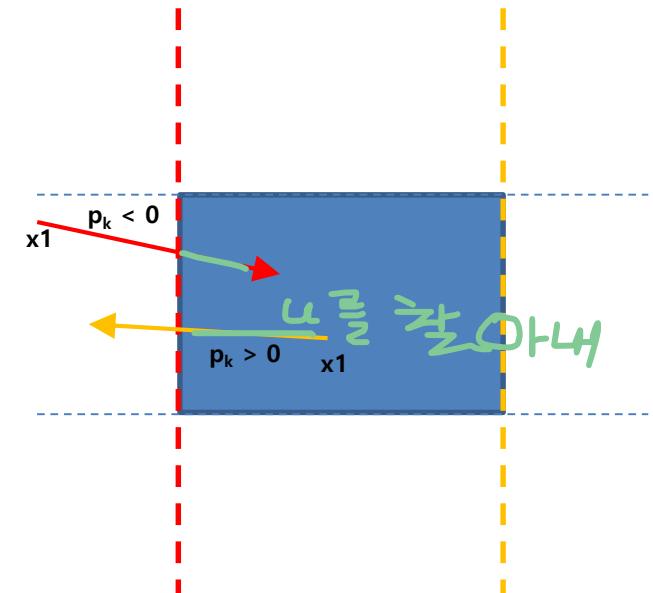
- 가장자리와의 교차점인 새로운 매개변수 u_{start} 와 u_{end} 에 대해서,

- $- \text{if } u_{start} > u_{end} \rightarrow \text{reject}$

- $- \text{if } u_{start} < u_{end} \rightarrow \text{새로운 좌표값}$

- $\gg \text{new_x1} = x_1 + u_{start} * dx, \quad \text{new_y1} = y_1 + u_{start} * dy$

- $\gg \text{new_x2} = x_1 + u_{end} * dx, \quad \text{new_y2} = y_1 + u_{end} * dy$



Clipping (클리핑): 선 - Liang-barskey 알고리즘

사용 예 미란식

- 예) 원도우 영역 $(5, 5)$ $(10, 10)$, $P_1 = (3, 8)$ $P_2 = (12, 2)$

$$x_{min} = 5 \quad y_{min} = 5 \quad x_{max} = 10$$

$$x_1 = 3 \quad y_1 = 8 \quad x_2 = 12 \quad y_2 = 2 \quad dx = 9 \quad dy = 6$$

$$P_k \quad q_{1c} \quad u_k$$

$$k=4 \quad P_4 = dy \quad q_4 =$$

$$P_1 = -dy = -9$$

$$u_1 = \frac{q_1}{P_1}$$

$$u_s = \max\left(\frac{r}{q_1}, -\frac{1}{3}\rho\right)$$

$$q_1 = x_2 - x_{min} = 3 - 5 = -2$$

$$u_2$$

$$u_e = \min($$

$$P_2 = 9 \quad q_2 =$$

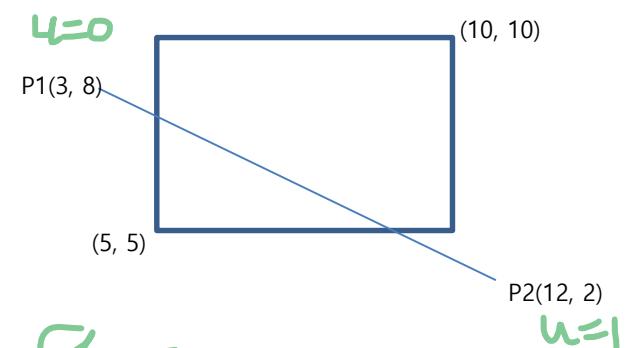
$$u_3$$

$$u_s < u_e$$

$$L = 3 \quad P_3 = 6 = dy$$

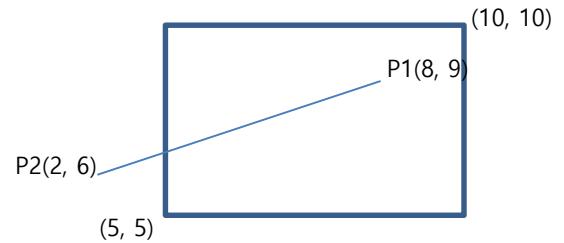
$$q_3 = y_1 - y_{min}$$

$$u_4$$



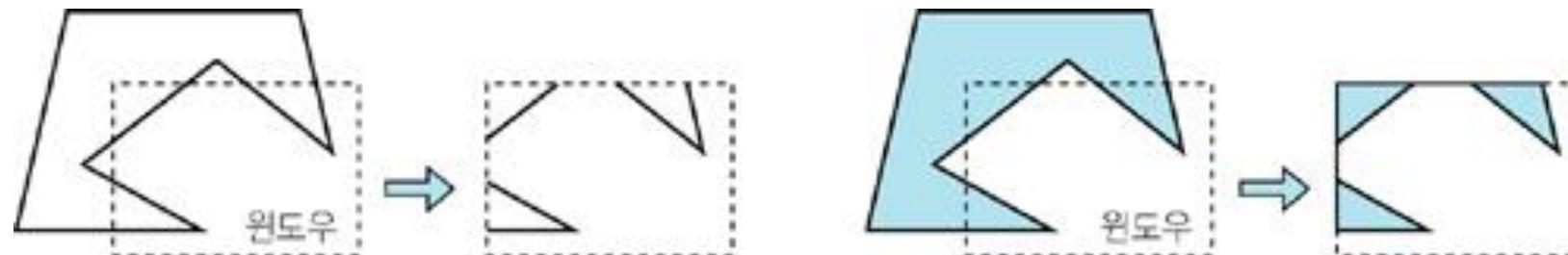
Clipping (클리핑): 선 - Liang-barskey 알고리즘

- 예) 윈도우 영역 $(5, 5)$ $(10, 10)$, $p1=(8, 9)$ $p2=(2, 6)$



Clipping (클리핑): 다각형 - Sutherland-Hodgeman 알고리즘

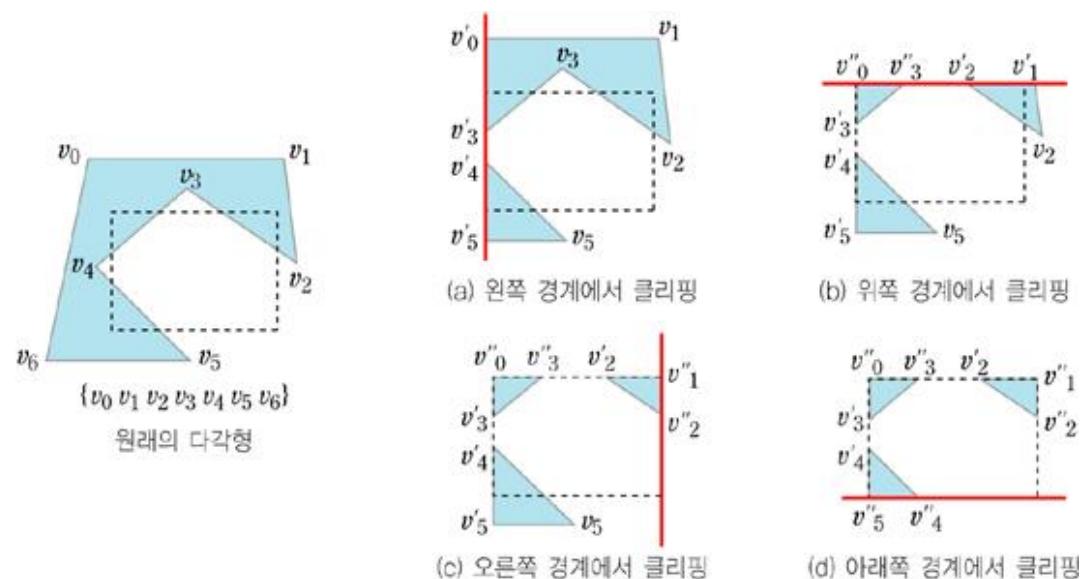
- 속이 빈 다각형(Hollow polygon) :
 - 선 클리핑 알고리즘 적용
- 속이 찬 다각형 :
 - 몇 개의 Closed filled polygon 생성



Clipping (클리핑): 다각형 - Sutherland-Hodgeman 알고리즘

• Sutherland-Hodgeman 알고리즘

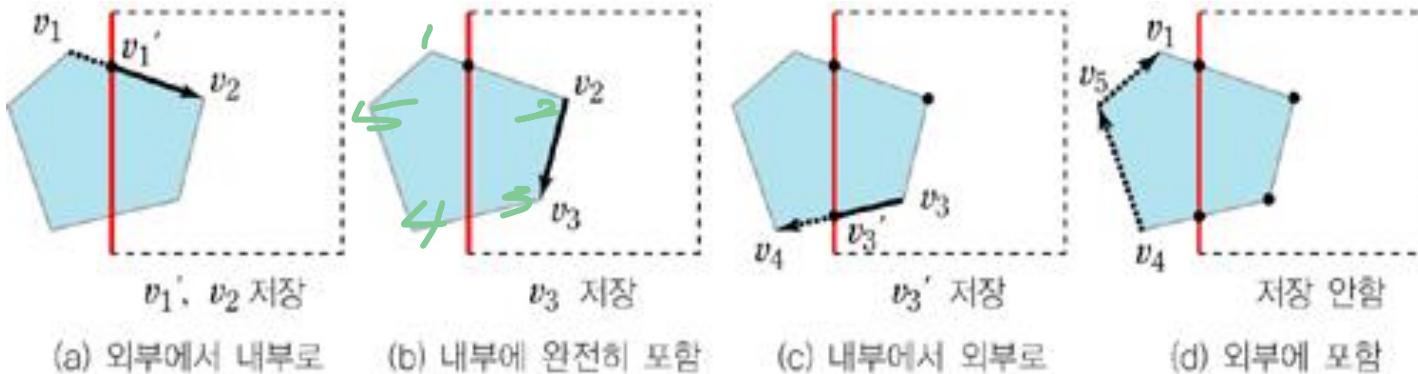
- 다각형의 모든 꼭지점이 윈도우의 내부 또는 외부에 완전히 포함되는지를 결정하여 다각형 전체를 제거하거나 선택하고 그 외의 경우에는 다음 알고리즘을 적용하여 다각형을 클리핑
- 한 경계변을 기준하여 이 변이 윈도 바깥쪽 영역에 속하는 다각형 부분은 클리핑 소거



Clipping (클리핑): 다각형

- Sutherland-Hodgeman 알고리즘

- 각 윈도우 경계 (상하좌우)에 대하여 다음 알고리즘을 적용
- 4가지 경우로 구분하여 다각형 꼭지점을 재구성



$\frac{P_1 \rightarrow P_2}{P_2 P_3} : 외 \rightarrow 내 \quad \left\{ P_1' \ P_2 \ P_3 \ P_3' \right\}$ 새 점 만들기

$\frac{P_2 P_3}{P_3 P_4} : 내$

$\frac{P_3 P_4}{P_4 P_5} : 내 \rightarrow 외$

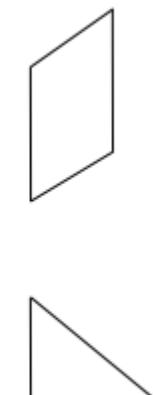
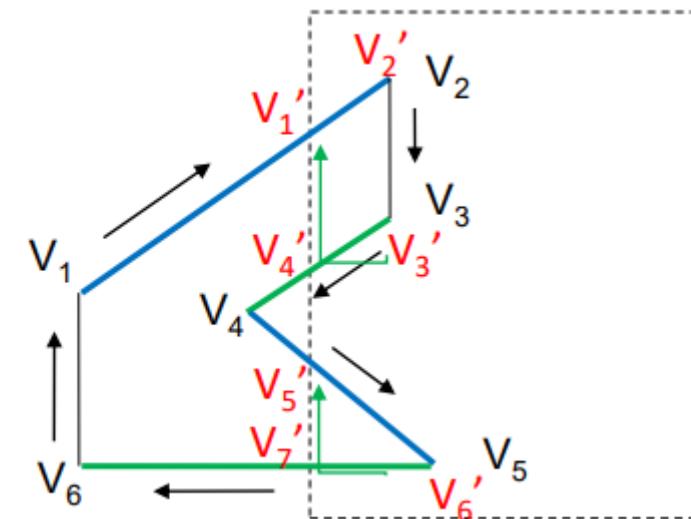
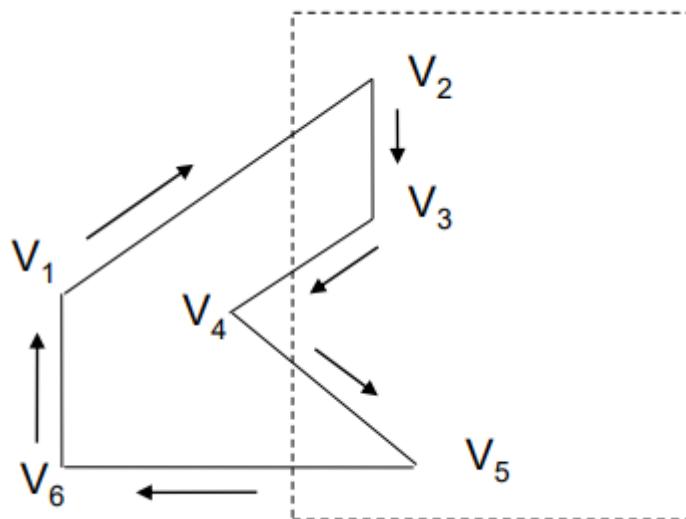
$\frac{P_4 P_5}{P_5 P_1} : 외$

$\frac{P_5 P_1}{P_1 P_2} : 외$

Clipping (클리핑): 다각형 – Weiler-Atherton 알고리즘

- **Weiler-Atherton 알고리즘**

- 오목 다각형 클리핑 시 발생할 수 있는 Sutherland-Hodgeman 알고리즘의 문제점을 해결
 - 시계방향으로 꼭지점을 따라간다
 - 외부 → 내부: 폴리곤 경계를 따라간다.
 - 내부 → 외부: 폴리곤 경계를 시계방향으로 따라간다.



이번 주에는

- 윈도우와 뷰포트
- 클리핑
 - 점 클리핑
 - 선 클리핑
 - 다각형 클리핑
- 다음 시간에는
 - 3차원 그래픽스의 기하변환