컴퓨터 그래픽스 5장
2차원 그래픽스의 변환

2015년 2학기

학습 내용

- 2차원 그래픽스 변환
  - 기본 변환: 이동, 회전, 신축
  - 그 외, 반사, 밀림
  - 동차좌표계: 모든 변환에 똑같이 적용할 수 있도록 차수를 바꿈
  - 윈도우와 뷰포트: 무엇을 어디에 그리는가
  - 클리핑
    - 보이지 않는 부분의 객체를 그리지 않는다.

기본 기하 변환

- 기본 기하 변환
  - 이동 (Translation)
  - 회전 (Rotation)
  - 신축 (크기 변환, Scale)

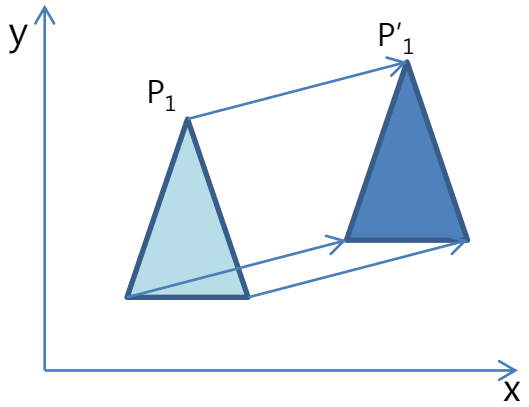
- Translation (이동)
  - 좌표계의 한 곳에서 다른 곳으로 직선 경로를 따라 객체의 위치를 바꾸는 것
  - 객체의 크기나 모양, 방향 등은 바뀌지 않는다.
  - P(x, y) -> P'(x', y')

    x' = $x+t_x$ y' = $y+t_y$

    (t_x, t_y): 이동 벡터 (translation vector)

    행렬을 사용하면

P' =



기본 기하 변환

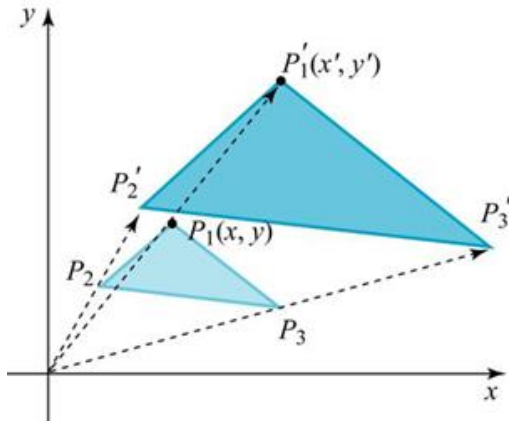
- Scaling (신축, 확대/축소)
  - 객체의 크기를 확대/축소 시킨다.
  - 객체의 크기뿐 아니라 기준점으로부터의 위치도 배율에 따라 변한다.
  - $x' = s_x \bullet x$ $\quad\quad$ $y' = s_y \bullet y$

    (s_x, s_y): 신축률 (scaling factor)

행렬을 사용하면

$P' =$

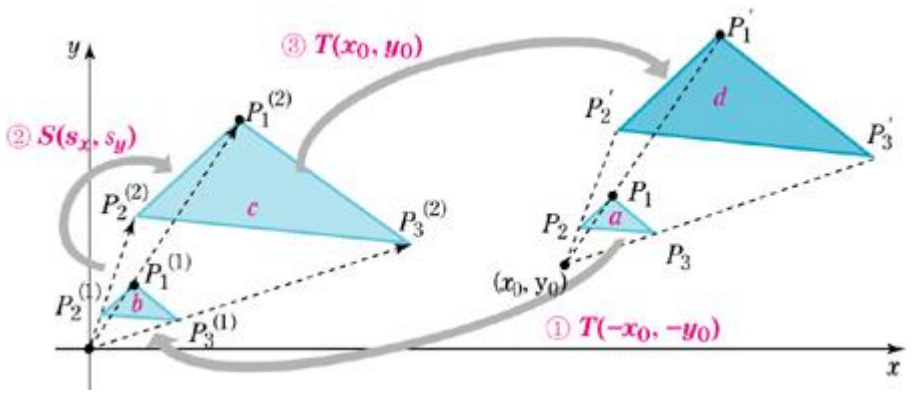
- $s > 1$:
- $s = 1$:
- $0 < s < 1$:
- $s < 0$:



기본 기하 변환

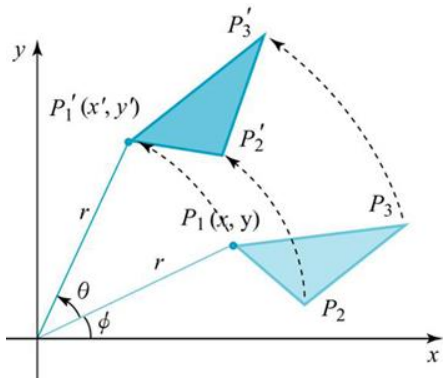
- 임의의 점 (x_0, y_0)에 대하여 신축률 (s_x, s_y)만큼 신축
  - 신축 기준점을 원점이 되도록 객체를 이동: $T(-x_0, -y_0)$
  - 원점에 대하여 신축: $S(s_x, s_y)$
  - 제자리로 이동: $T(x_0, y_0)$

  - x' =
  - y' =



기본 기하 변환

- Rotation (회전)
  - xy평면에서 원 경로를 따라 객체를 재배치
  - 객체의 모양 변화는 없이 객체가 놓여있는 방향이 변한다.
  - 회전각 : θ 회전점 (Pivot Point): (x_r, y_r)
    - $x' = r\cos(\Phi+\theta) = r\cos\Phi\cos\theta - r\sin\Phi\sin\theta = x\cos\theta - y\sin\theta$
    - $y' = r\sin(\Phi+\theta) = r\cos\Phi\sin\theta + r\sin\Phi\cos\theta = x\sin\theta + y\cos\theta$
  - 행렬을 사용하면
    P′ =

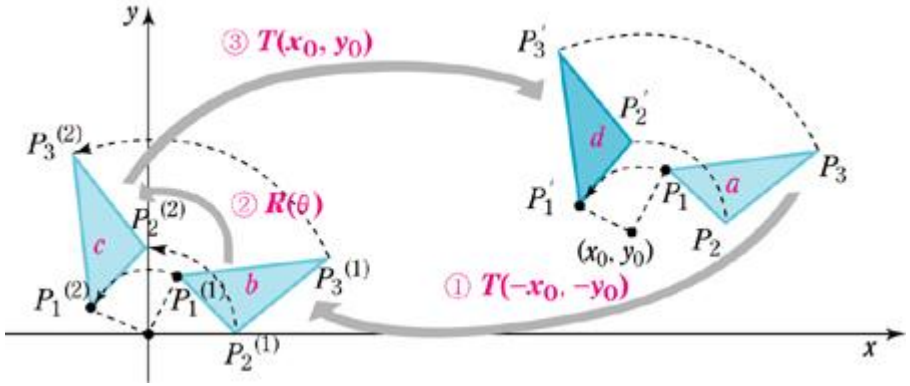


기본 기하 변환

- 임의의 점 (x_0, y_0)에 대하여 θ만큼 회전시키는 과정
  - 회전 중심점이 원점이 되도록 객체를 이동: T $(-x_0, -y_0)$
  - 원점을 중심으로 θ만큼 회전: R(θ)
  - 반대 방향으로 이동: T (x_0, y_0)

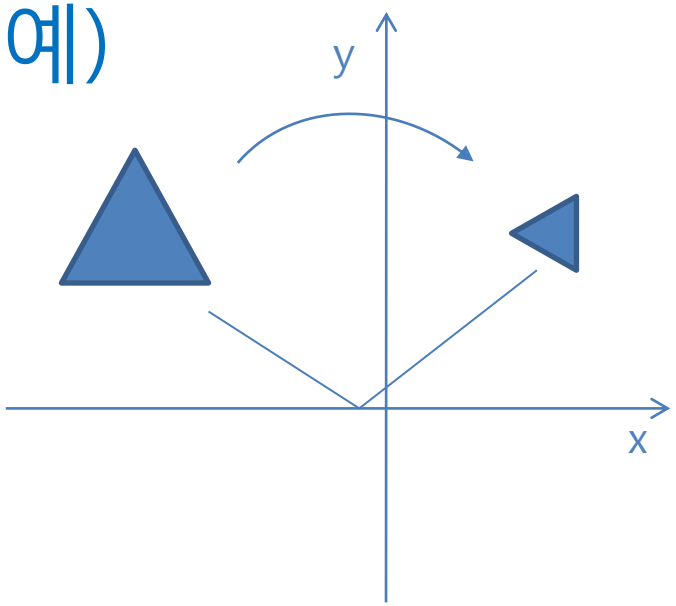
  - x' =

  - y' =

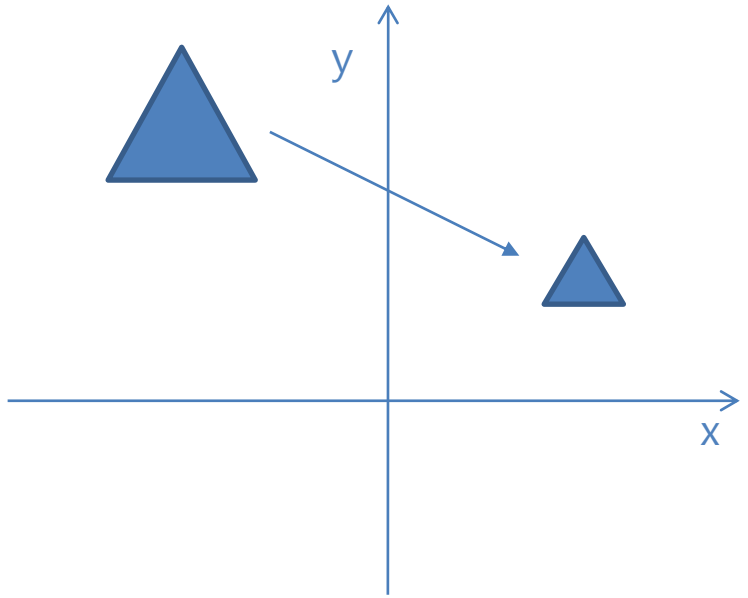


기본 기하 변환

- 예)



- 시계반대방향으로 90도, 2배 스케일
- 삼각형 좌표값 (-5, 5) (-10, 5) (-8, 10)
  → 변환 좌표값은?



- ½배 스케일, X축으로 10, Y축으로 -5 이동
- 삼각형 좌표값 (-5, 5) (-10, 5) (-8, 10)
  → 변환 좌표값은?

동차 좌표계 (Homogeneous Coordinate System)

- 동차 좌표계
  - 여러 단계의 변환행렬을 하나로 결합하여 표현하도록 하는 방법
  - 순차적인 기하변환을 처리할 때 각 단계별 좌표 값을 구하지 않고 바로 계산 하려면 행렬의 합(A)을 제거해야 함

    - $P_2 = M \cdot P_1 + A_1$
    - $P_3 = M_2P_2 + A_2 = M_2(M_1P_1 + A_1) + A_2$
      $= M_2M_1P_1 + M_2A_1 + A_2$

  - 동차 좌표계를 이용하여 기본 변환을 행렬 곱으로만 표현한다 → 변환을 간단히 처리, 계산량을 줄일 수 있다. 즉,

$$P_n = M_{n-1} \cdot P_{n-1} = M_{n-1} \cdot M_{n-2} \cdot P_{n-2} = \cdots$$
$$= M_{n-1} \cdot M_{n-2} \cdot \ldots \cdot M_1 \cdot P_1$$
$$= M \cdot P_1$$

동차 좌표계 (Homogeneous Coordinate System)

- 행렬식
  - 2차원의 점 P(x, y)를 동차 좌표계로 표현하면 → P(hx, hy, h) (h ≠0) → h는 임의의 값 → 한 점은 동차 좌표계에서 h의 값에 따라 여러 개의 좌표로 표현될 수 있다.
    - 동차 좌표계에서 한 점의 좌표 P(X, Y, h)로 주어지면 → 2차원 기하 평면에서 P(X/h, Y/h)로 대응된다.
    - $P(x_1, y_1, h_1)$, $P(x_2, y_2, h_2)$로 주어지면 $(x_1/h_1, y_1/h_1) = (x_2/h_2, y_2/h_2)$이면 2차원 기하 평면에서 동일한 점이 된다.
    - h = 1 이면 (x, y) → (x, y, 1)
  - 이동의 3차원 행렬식: T(tx, ty) =
  - 회전의 3차원 행렬식: R(θ) =
  - 신축의 3차원 행렬식: $S(s_x, s_y)$ =

합성 변환

- 연속된 기본 변환을 동차좌표계를 사용하여 하나의 행렬로 나타낼 수 있다. 하나의 변환 행렬로 표현한 합성 변환에서는 한번의 행렬 곱셈만 필요하다
  - 이동: 연속적으로 2번 이동하는 경우

$$P' = T(t_{x2}, t_{y2}) \bullet T(t_{x1}, t_{y1}) \bullet P$$
$$= T(t_{x2} + t_{x1}, t_{y2} + t_{y1}) \bullet P$$

  - 신축: 연속적으로 2번 신축 하는 경우

$$P' = S(s_{x2}, s_{y2}) \bullet S(s_{x1}, s_{y1}) \bullet P$$
$$= S(s_{x2} \bullet s_{x1}, s_{y2} \bullet s_{y1}) \bullet P$$

  - 회전 : 연속적으로 2번 회전하는 경우

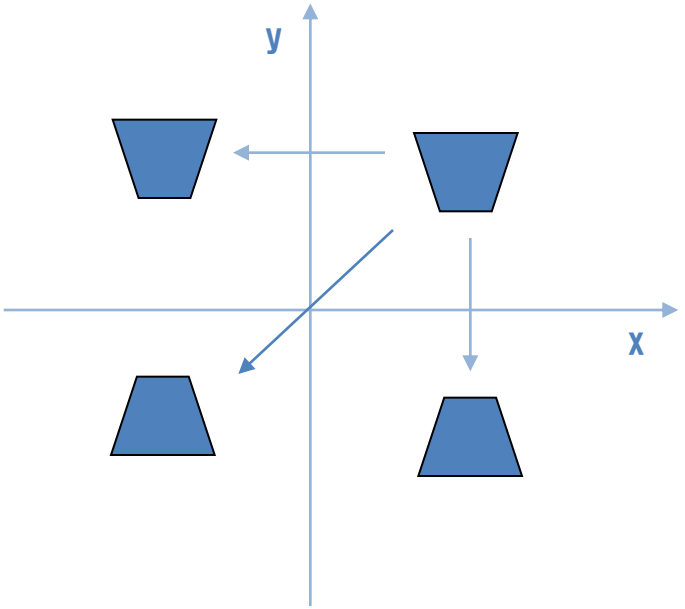
$$P' = R(\theta_2) \bullet R(\theta_1) \bullet P$$
$$= R(\theta_2 + \theta_1) \bullet P$$

합성 변환

- 행렬 곱셈의 성질
  - 모든 행렬은 곱셈에 대하여 결합법칙이 성립한다.
    - $(A \cdot B) \cdot C = A \cdot (B \cdot C)$
    - 여러가지 변환 행렬이 연속적으로 적용되었을 대, 행렬 곱셈을 처리하는 계산순서에 관계없이 변환 결과는 항상 동일하다.
  - 행렬은 곱셈에 대하여 일반적으로 교환법칙이 성립하지 않는다.
    - $A \cdot B \neq B \cdot A$
    - 변환 순서를 바꾸면 다른 결과가 된다.
  - 같은 종류의 기하변환을 나타내는 경우에는 행렬 곱셈에 대하여 교환법칙이 성립한다.

기타 기하 변환

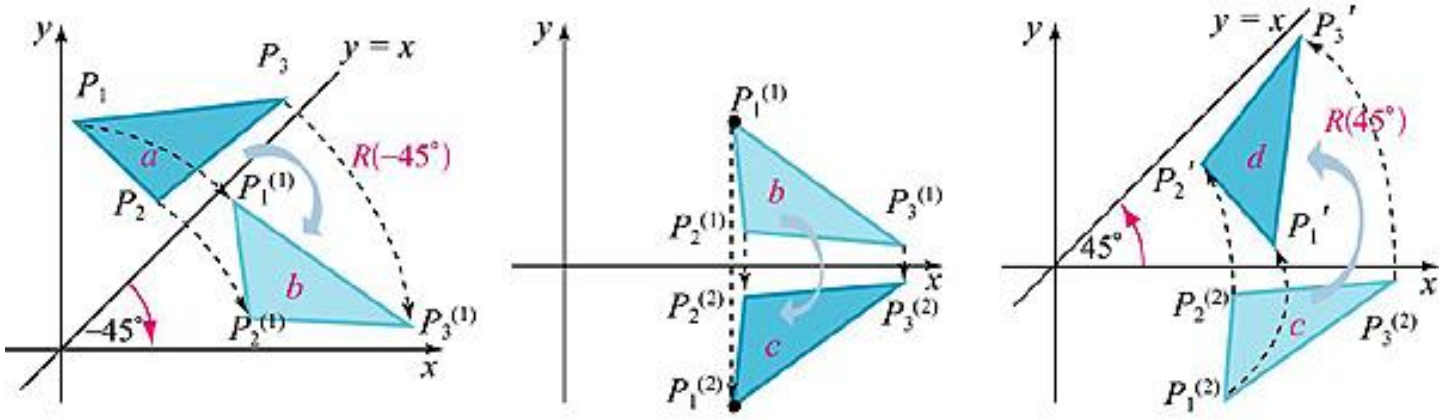
- Reflection (반사): 거울 영상
  - y=0 (x축)에 대하여 반사
  - x=0 (y축)에 대하여 반사
  - 원점 (0,0)에 대하여 반사



기타 기하 변환

- y=x에 대한 반사
  - 시계방향으로 45’ 회전
  - X축에 대하여 반사
  - 시계 반대 방향으로 45’ 회전

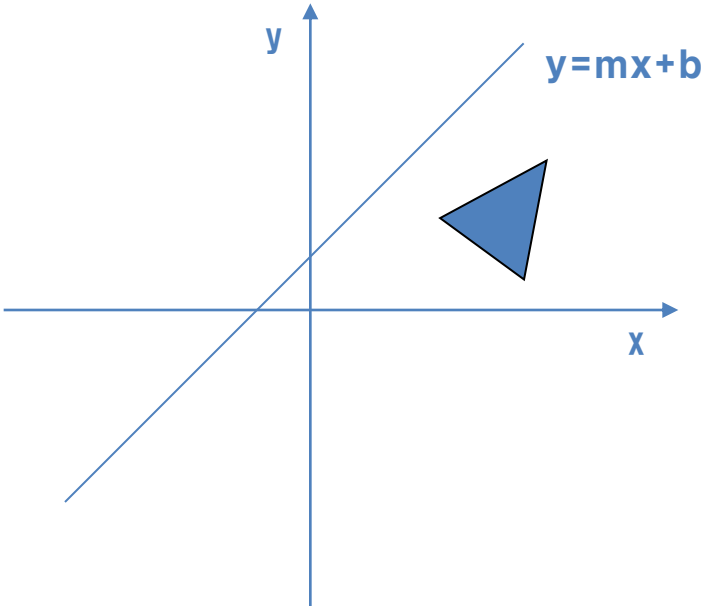
- y = -x에 대한 반사



(a) 다각형과 반사축을 회전 (b) 반사축을 기준으로 반사 (c) 반대방향으로 회전

기타 기하 변환

– y = mx + b에 대하여 반사

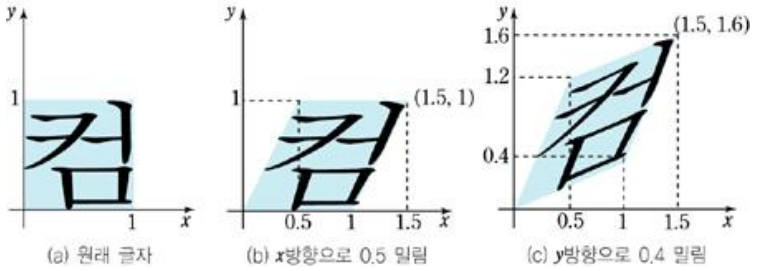


기타 기하 변환

- 밀림 (Shearing)
  - 2차원 평면상에서 객체의 한 부분을 고정시키고 다른 부분을 밀어서 생기는 변환
    - 고정된 지점에서 멀수록 밀리는 거리가 커진다. (고정된 지점과의 거리에 비례하여 밀리는 경우가 결정된다)
    - x축에 대한 밀림: y축에 대한 밀림:

      $x' = x + h_x \bullet y$ $x' = x$ (h_x :밀림 비율)

      $y' = y$ $y' = y + h_y \bullet x$ (h_y: 밀림 비율)

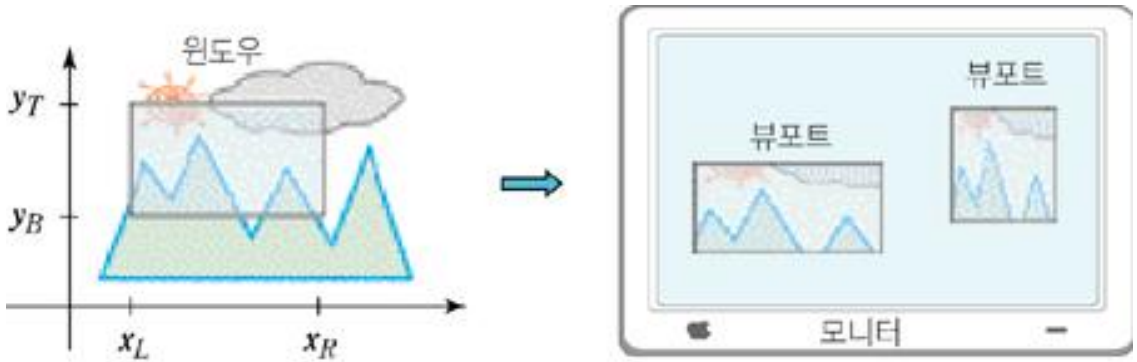


윈도우와 뷰포트

- Window와 viewport의 개념
  - Viewing pipeline

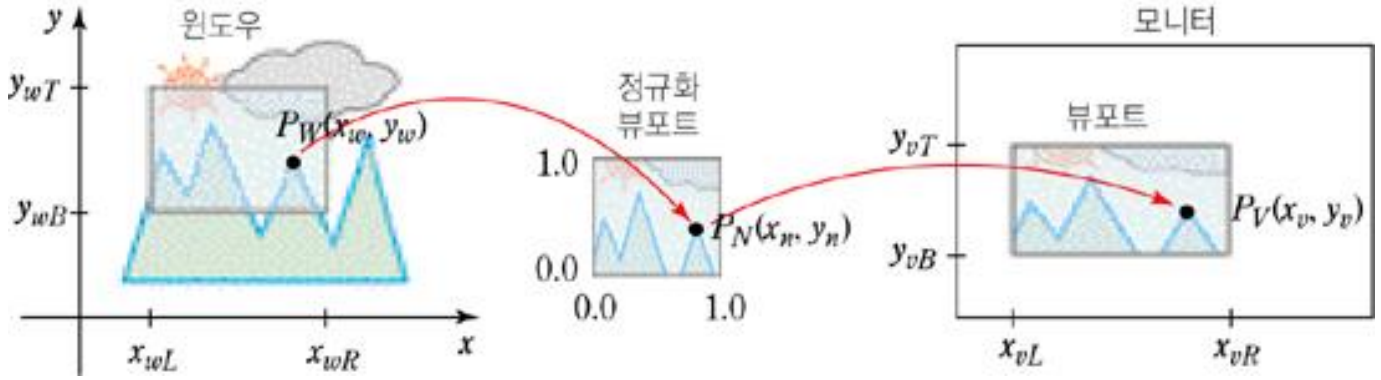
- 모델 좌표계: 개별 객체를 표현하기 위해 사용되는 좌표계
- 월드 좌표계: 각 모델 좌표계의 통합된 좌표계
- 뷰잉 좌표계: 출력장치에 출력 위히 및 크기 설정하여 뷰포트에 출력, 뷰포트 좌표계
- 정규 좌표계: 정규화된 좌표계
- 장치 좌표계: 출력하려는 장치 좌표계

윈도우와 뷰포트



- Window
  - 출력 장치에 표시하기 위해 선택된 세계 좌표 영역
- Viewport
  - 윈도우가 사상되는 출력 장치의 영역
- 윈도우-뷰포트 변환에 의한 효과
  - Zooming (zoom in/zoom out)
  - Panning
  - 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

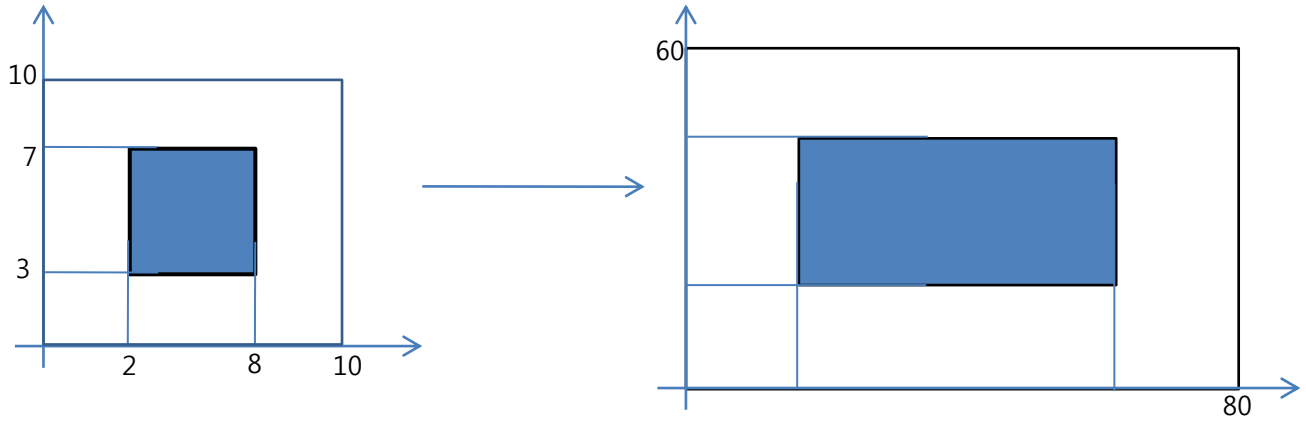
- $x_v = x_{vL} + (x_w - x_{wL})s_x$, $s_x = (x_{vR} - x_{vL})/(x_{wR} - x_{wL})$
- $y_v = y_{vB} + (y_w - y_{wB})s_y$, $s_y = (y_{vT} - y_{vB})/(y_{wT} - y_{wB})$

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 (0, 10, 0, 10) → 뷰포트 (0, 80, 0, 60)

    도형 좌표: (2, 3) (8, 7)로 이루어진 사각형일 때



클리핑 알고리즘

- Clipping의 개념
  - 윈도우-뷰포트 변환 시, 출력장치에 표시되어서는 안될 그림영역을 제거한 뒤, 나머지 그림영역을 출력화면에 나타내는 것
  - 월드 좌표 클리핑:
    - 윈도우를 설정할 때 윈도우 바깥 영역을 제거하여 윈도우 내부 영역만 뷰포트로 매핑시키는 방법
  - 뷰포트 클리핑:
    - 월드 좌표계를 표현된 그림 전부를 뷰포트로 매핑시킨 후 뷰포트 외부에 위치한 객체나 그림의 일부를 제거하는 방법
  - 두 클리핑이 모두 결과는 같다.
  - 월드 좌표계를 사용하면 계산 시간이 줄어든다.

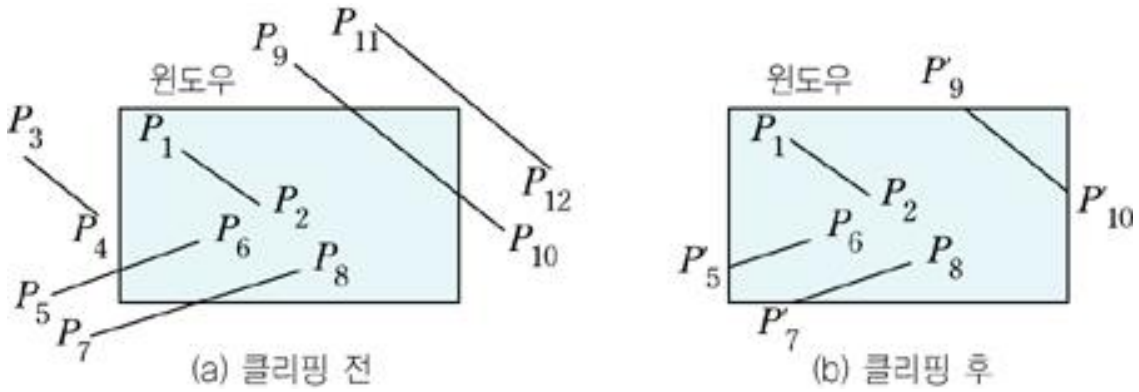
점 클리핑

- 점 클리핑
  - 클리핑 되는 객체가 점
  - 한 점 P(x, y)는

    - $x_L \le x \le x_R$, $y_B \le y \le y_T$

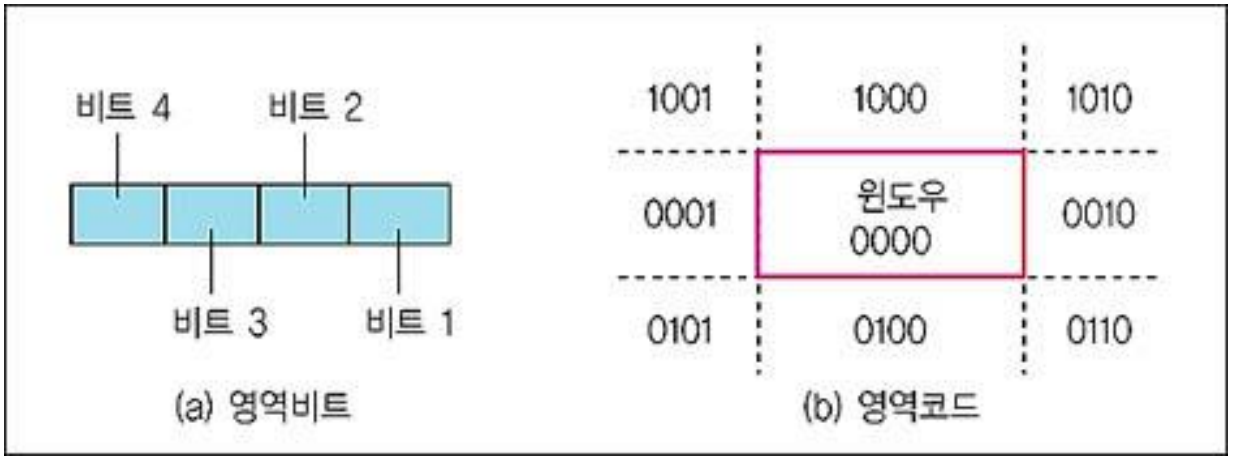
    이면 그려진다.

선 클리핑



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

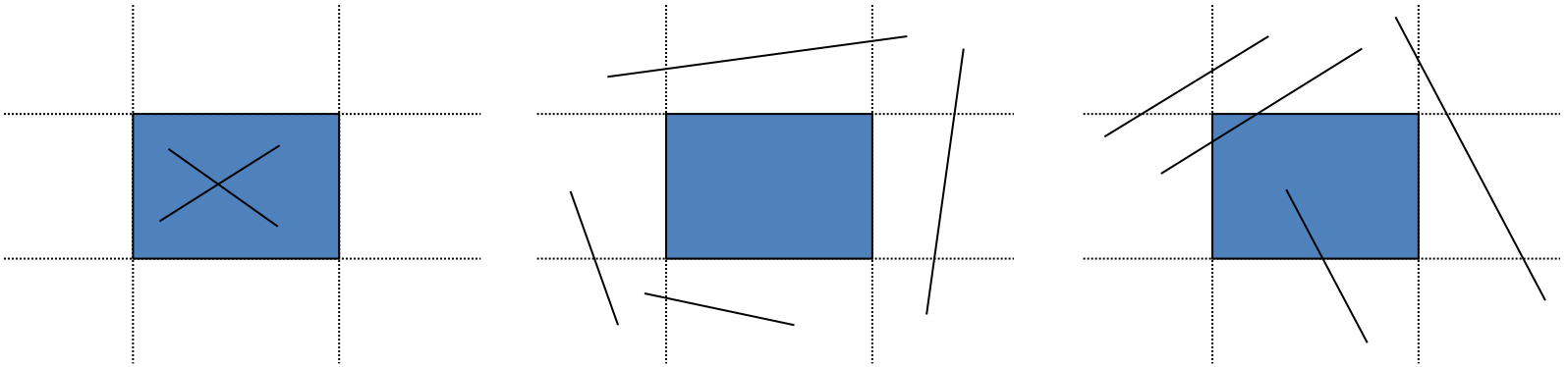
선 클리핑



- 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

선 클리핑

- 알고리즘 수행 과정
  - 양 끝점의 코드가 모두 0000이면 →
  - 양 끝점의 코드 중 한 쪽 코드는 0이고 다른 쪽 코드는 0이 아니면 →
  - 양 끝점 코드가 모두 0이 아니고, 양 끝점 코드간 AND 연산이 0이 아니면 →
  - 양 끝점 코드가 모두 0이 아니고, 양 끝점 코드간 AND 연산이 0이면 →



선 클리핑

- 주어진 선분에 대한 교차점 구하기
  - 수직 경계: $x = x_L$ 또는 $x = x_R$

    $y = y1 + m(x - x1)$
  - 수평 경계: $y = y_B$ 또는 $y = y_T$

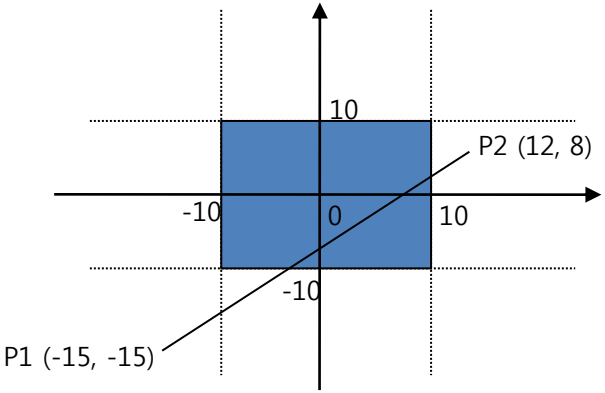
    $x = x1 + (y - y1)/m$

    $m = (y2 - y1) / (x2 - x1)$선분의 기울기,

    x1, y1: 선분의 끝 점

    x_L, x_R , y_B, y_T : 윈도우의 경계

- 예)



영역 클리핑

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

영역 클리핑

- 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