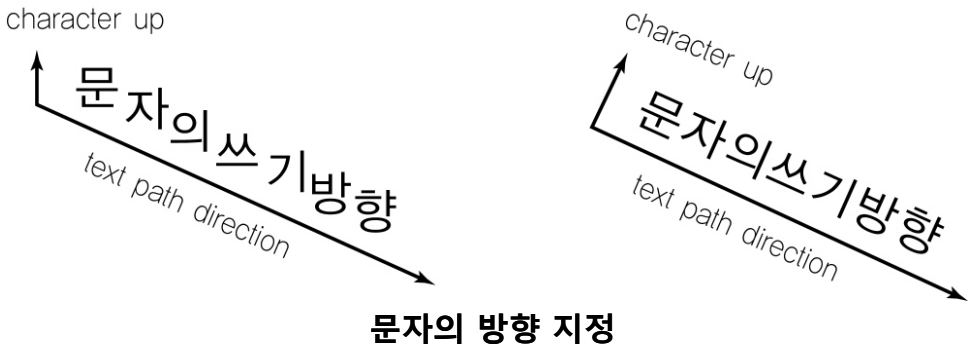
2차원 그래픽스의 기본(2)

박영진

문자와 텍스트의 속성

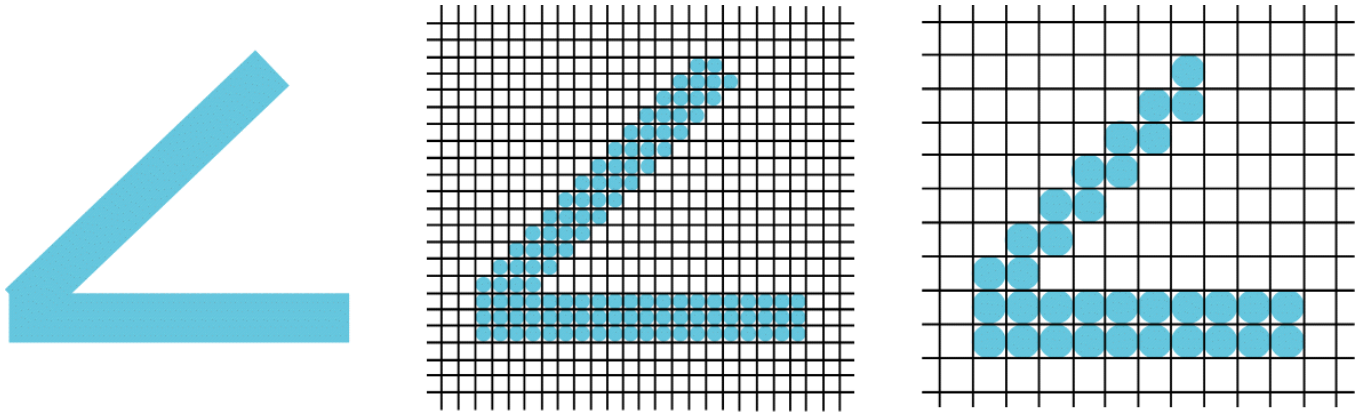
- 문자(Character)의 주요 속성
  - 폰트, 색상, 크기 (폭과 높이), 스타일 (굵은체, 이탤릭, 밑줄 등), ...
- 텍스트(Text 또는 String)의 속성
  - 세우기 방향(Orientation) : Character Up 벡터로 표현
  - 쓰기 방향(Direction) : Text Path Direction으로 표현
  - 정렬(Alignment), 행 간격, 문자 간격 등



문자의 방향 지정

래스터 출력의 문제점 : 앨리어싱

- 앨리어싱(Aliasing)
  - 래스터 출력장치에서 디지털화 과정의 샘플링 오차로 인한 왜곡현상
  - 사선의 굵기가 수평선의 굵기와 다르게 보이는 현상
  - 사선이나 곡선부분에 나타나는 계단현상

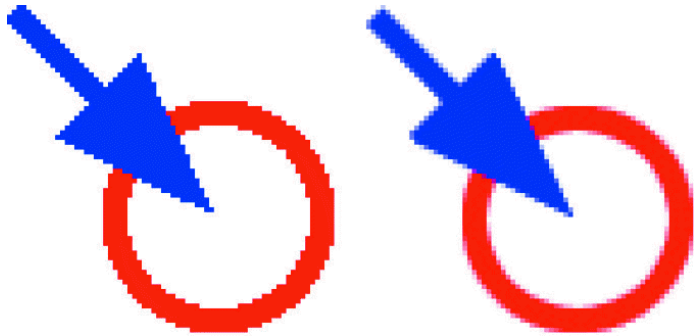


(a) 원래 그림 (b) 고해상도 출력 (c) 저해상도 출력

래스터 출력의 문제점

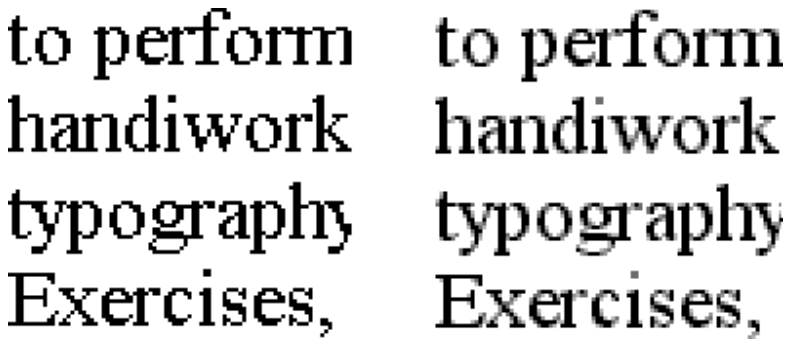
안티앨리어싱

- 안티앨리어싱(Antialiasing)
  - 컬러 또는 회색조(Gray) 출력 장치에서 경계가 부드럽게 보이도록 하는 기법
  - 물체의 경계 픽셀에서 물체와 배경의 색상을 혼합해서 그린다.
  - 선 그리기, 다각형 채우기, 문자 생성 등에 적용이 가능



앤티앨리어싱 전　　앤티앨리어싱 후

도형의 앤티앨리어싱

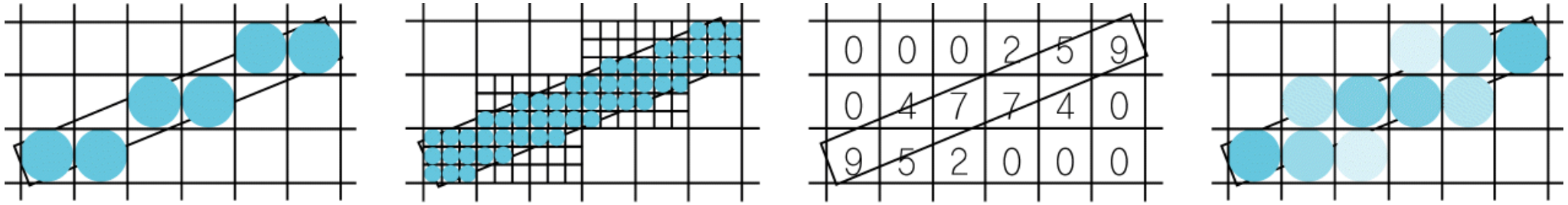


앤티앨리어싱 전　　앤티앨리어싱 후

문자의 앤티앨리어싱

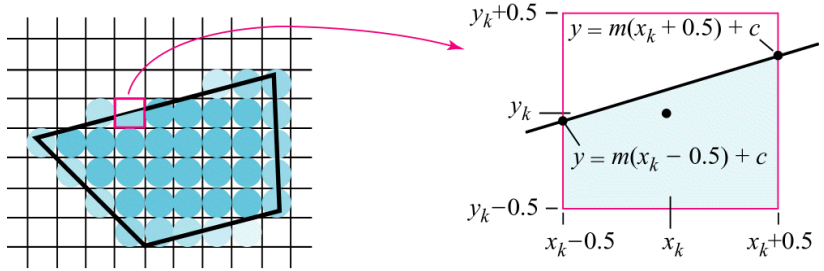
안티앨리어싱 기법

- 수퍼샘플링(Super Sampling) 기법
  - 하나의 픽셀 영역을 여러 개로 분할하는 수퍼샘플링 과정을 적용, 원래의 해상도로 환원할 때 픽셀의 명암 값을 계산
  - 픽셀의 영역에 포함되는 고해상도 픽셀의 개수에 비례하여 명암 값을 계산



(a) 원래 해상도 (b) 수퍼샘플링 (c) 각 픽셀의 명암값 (d) 출력 결과

- 영역 샘플링(Area Sampling) 기법
  - Pitteway와 Watkinson이 제안: 얼마만큼 다각형의 내부영역에 포함되는지 포함된 면적비율을 계산



(a) 다각형 경계의 앤티앨리어싱 (b) 픽셀에서의 면적계산

2차원 그래픽스의 변환

박영진

기본적인 기하 변환

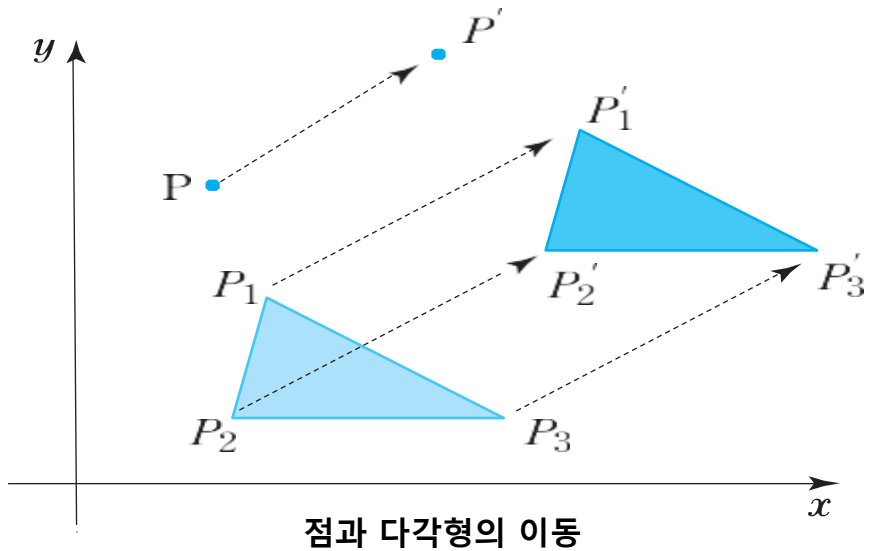
- 이동(Translation)
- 신축(확대/축소, Scaling)
- 회전(Rotation)

이동(Translation)

- 한점 $P(x, y)$가 $T(t_x, t_y)$만큼 이동

$$x' = x + t_x, \quad y' = y + t_y$$

$$P' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} t_x \\ t_y \end{bmatrix} + \begin{bmatrix} x \\ y \end{bmatrix} = P + T(t_x, t_y)$$



점과 다각형의 이동

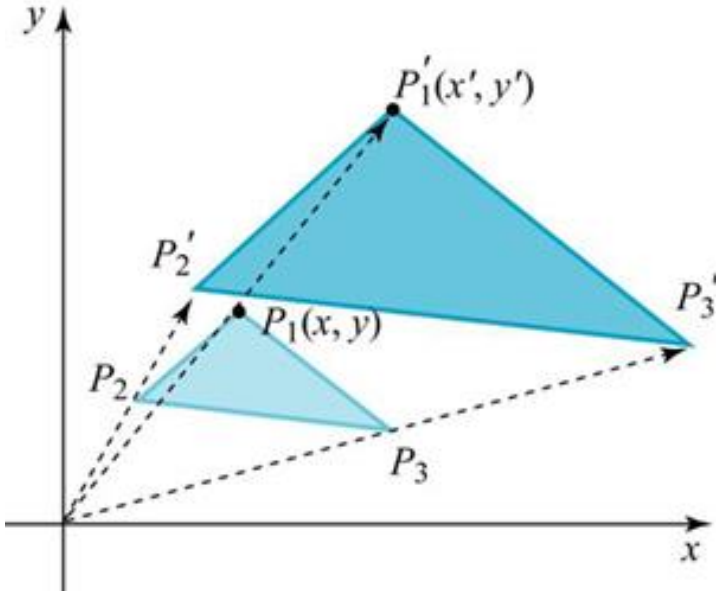
- 객체의 이동: 다각형의 경우 각 꼭지점을, 원의 경우 중심점을 이동하여 다시 그린다.

신축(확대/축소, Scaling)

- 한 점 $P(x\,, y)$가 원점을 기준으로 $S(s_x, s_y)$ 만큼 신축

$$x' = s_x \cdot x, \qquad y' = s_y \cdot y$$

$$P' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = S(s_x, s_y) \cdot P$$

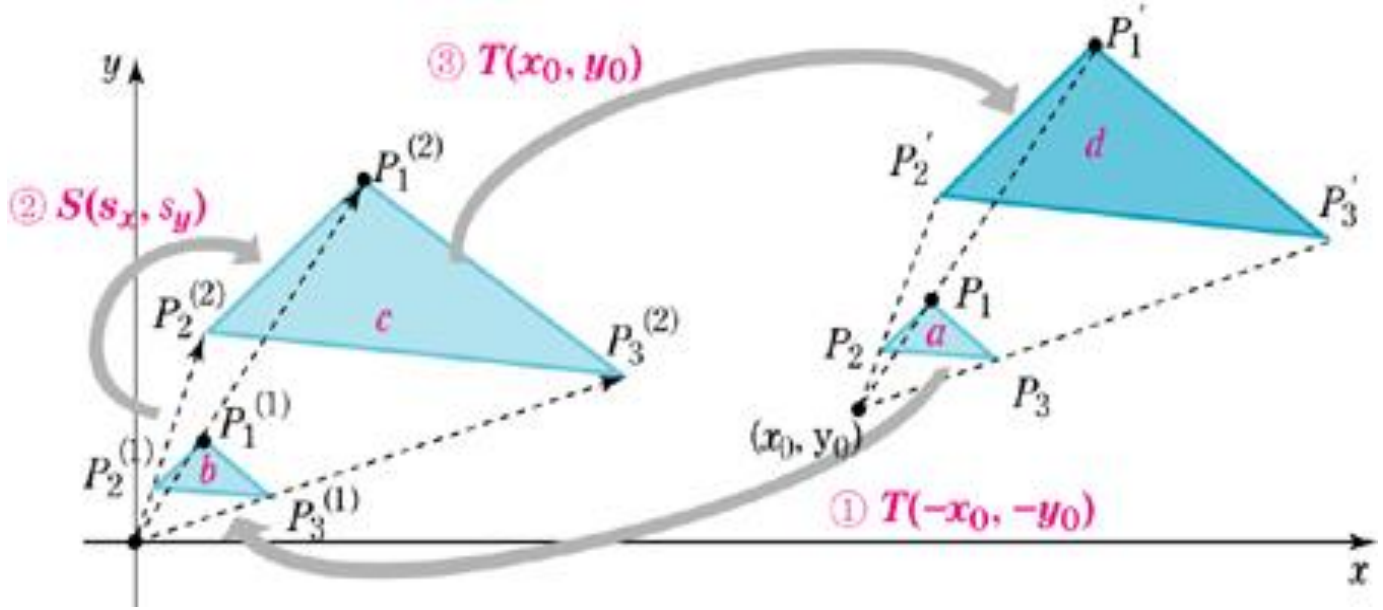


원점을 기준으로 하는 다각형의 신축

신축(확대/축소, Scaling)

- 임의의 점에 대한 신축
  (1) 기준점을 원점으로 이동: $\boldsymbol{T}(-x_0, -y_0)$
  (2) 원점에 대하여 신축 : $\boldsymbol{S}(s_x, s_y)$
  (3) 원래 위치로 이동 : $\boldsymbol{T}(x_0, y_0)$

$$\boldsymbol{P'} = \boldsymbol{T}(x_0, y_0) \cdot \boldsymbol{S}(s_x, s_y) \cdot \boldsymbol{T}(-x_0, -y_0) \cdot \boldsymbol{P}$$



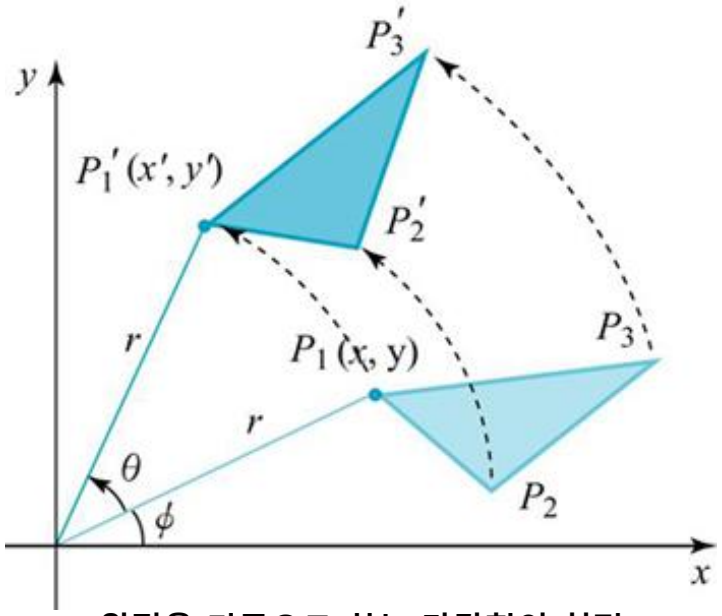
임의의 점을 기준으로 하는 다각형의 신축

회전(Rotation)

- 한점 P(x, y)를 원점을 중심으로 θ 만큼 회전

$$x' = r \cdot \cos(\phi + \theta) = x \cdot \cos\theta - y \cdot \sin\theta$$
$$y' = r \cdot \sin(\phi + \theta) = x \cdot \sin\theta + y \cdot \cos\theta$$

$$P' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = R(\theta) \cdot P$$

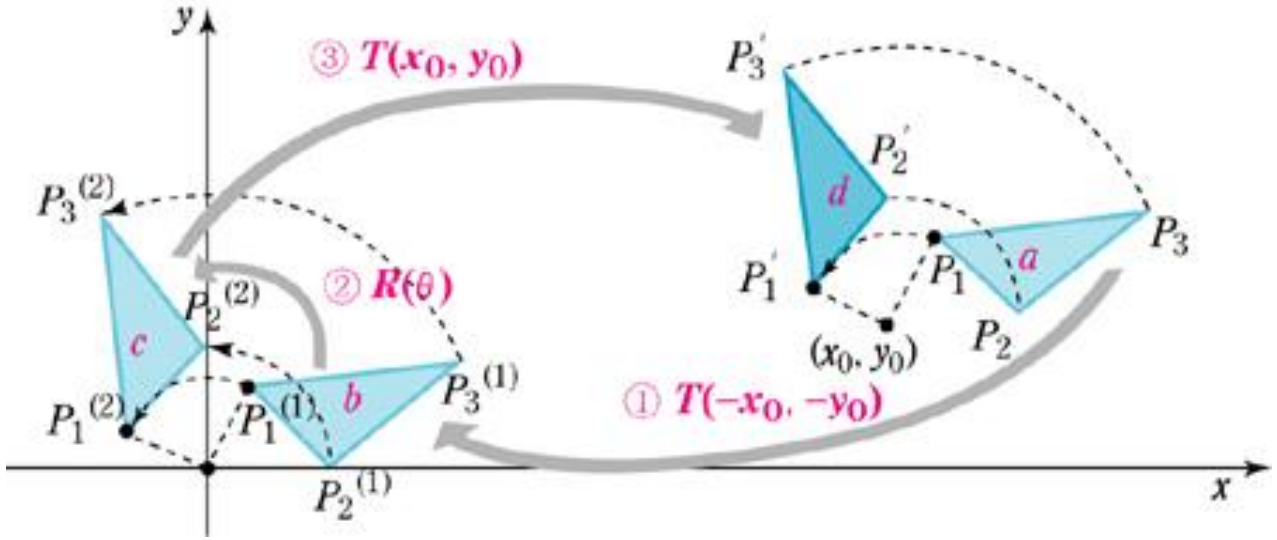


원점을 기준으로 하는 다각형의 회전

회전(Rotation)

- 임의의 점에 대한 회전
  (1) 기준점을 원점으로 이동: $\boldsymbol{T}(-x_0, -y_0)$
  (2) 원점을 중심으로 회전 : $\boldsymbol{R}(\theta)$
  (3) 원래 위치로 이동 : $\boldsymbol{T}(x_0, y_0)$

$$\boldsymbol{P}' = \boldsymbol{T}(x_0, y_0) \cdot \boldsymbol{R}(\theta) \cdot \boldsymbol{T}(-x_0, -y_0) \cdot \boldsymbol{P}$$



임의의 점을 기준으로 하는 다각형의 회전

동차좌표계의 필요성

- 기본변환은 P' = M · P + A 의 행렬 형태로 표현 가능
- 순차적인 기하변환을 처리할 때 각 단계별 좌표 값을 구하지 않고 바로 계산하려면 행렬의 합(A)을 제거해야 함

→ 동차좌표계를 이용하면 행렬의 곱으로만 표현 가능

$$P' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = T(t_x, t_y) \cdot P$$

동차 좌표계에서 행렬로 표현한 이동 변환

$$P' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = S(s_x, s_y) \cdot P$$

동차 좌표계에서 행렬로 표현한 신축 변환

$$P' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = R(\theta) \cdot P$$

동차 좌표계에서 행렬로 표현한 회전 변환

합성 변환

- 연속적인 이동 (t_{x1}, t_{y1})와 (t_{x2}, t_{y2})을 한 경우

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_{x_2} \\ 0 & 1 & t_{x_1} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & t_{x_1} \\ 0 & 1 & t_{y_1} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_{x_1}+t_{x_2} \\ 0 & 1 & t_{y_1}+t_{y_2} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- 임의의 점에 대한 신축 → 이동 + 신축 + (역)이동

$$P' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1\,0\,x_0 \\ 0\,1\,y_0 \\ 0\,0\,1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1\,0\,-x_0 \\ 0\,1\,-y_0 \\ 0\,0\,1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & x_0(1-s_x) \\ 0 & s_y & y_0(1-s_y) \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

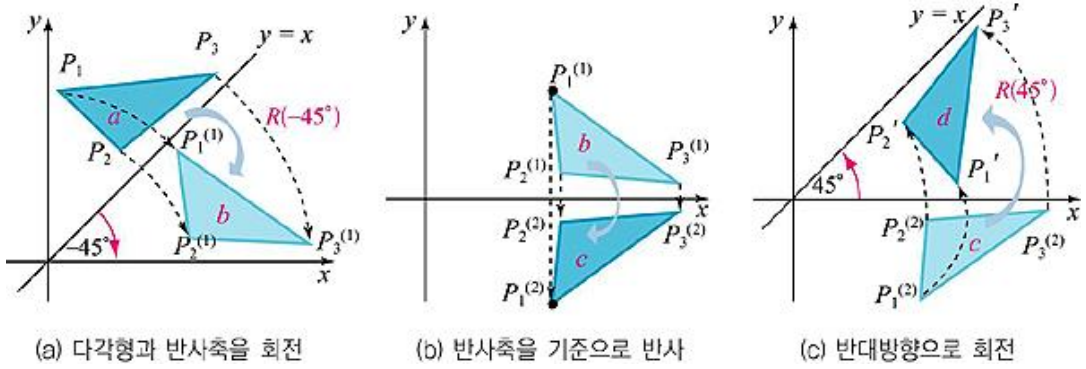
- 임의의 점에 대한 회전 → 이동 + 회전 + (역)이동

$$P' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1\,0\,x_f \\ 0\,1\,y_f \\ 0\,0\,1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1\,0\,-x_0 \\ 0\,1\,-y_0 \\ 0\,0\,1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

반사(Reflection)

- x축, y축, 원점에 대한 반사 : y좌표값 부호 변경, x좌표값 부호 변경, 모두 변경
- $y = x$ 에 대한 반사 :
  (1) 45° 시계방향 회전
  (2) x축에 대한 반사
  (3) 45° 반시계방향 회전

$$P' = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = RF \cdot P$$



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

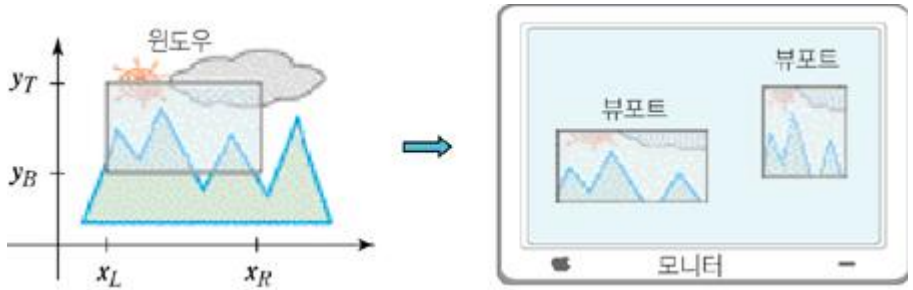
직선 $y = x$ 에 대한 반사

윈도우와 뷰포트(Viewport)

- 뷰잉 파이프라인(윈도우와 뷰포트의 개념)
- 윈도우-뷰포트 좌표 변환

뷰잉 파이프라인(윈도우와 뷰포트의 개념)

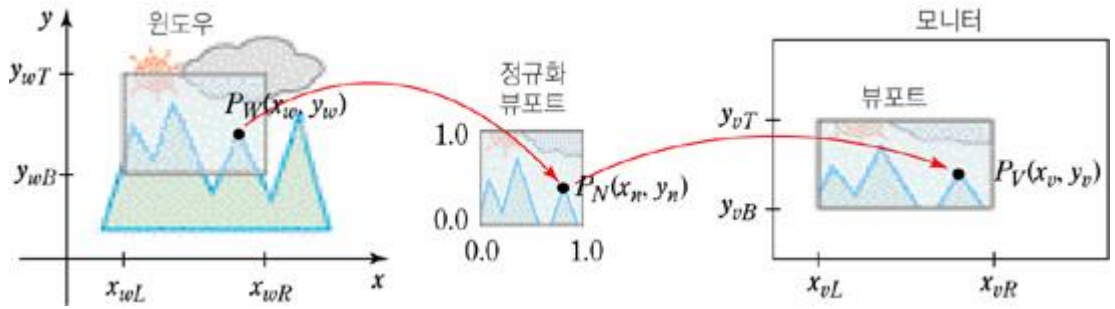
- 2차원 그래픽스의 뷰잉 과정:
  - 모델좌표계 : 개별 객체의 표현
  - 월드좌표계 : 통합된 그림, 윈도우 설정
  - 뷰잉좌표계 : 뷰포트 내의 좌표
  - 장치좌표계 : 출력될 화면, 뷰포트 설정



윈도우 – 뷰포트 변환

윈도우-뷰포트 좌표 변환

- 월드 좌표계에서 뷰잉 좌표계로의 변환



  - 이동 및 신축변환 적용

$$x_v = x_{vL} + (x_w - x_{wL}) \frac{x_{vR} - x_{vL}}{x_{wR} - x_{wL}}$$

$$y_v = y_{vB} + (y_w - y_{wB}) \frac{y_{vT} - y_{vB}}{y_{wT} - y_{wB}}$$

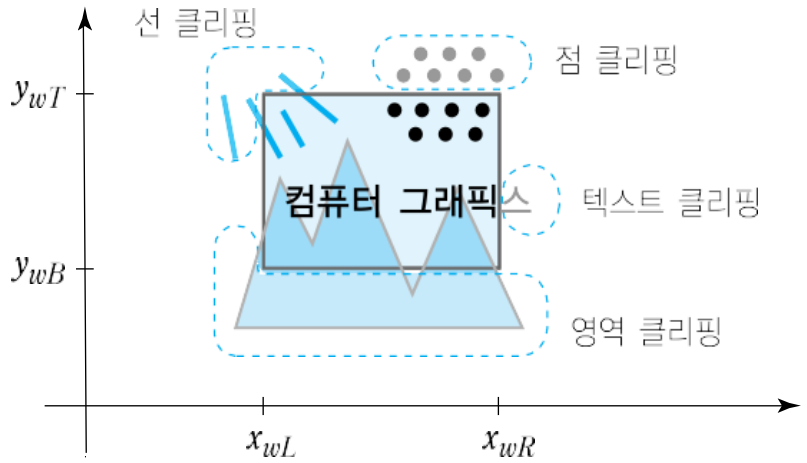
클리핑 알고리즘(Clipping Algorithm)

- 클리핑의 종류
- 점 클리핑
- 선 클리핑
- 영역 클리핑
- 텍스트 클리핑

클리핑의 종류

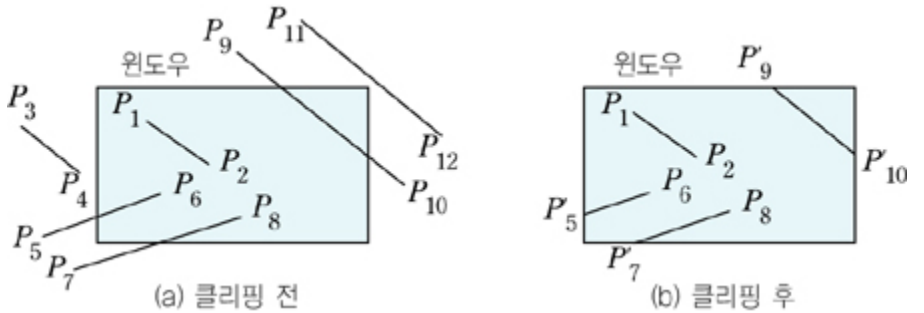
- 월드 좌표 클리핑과 뷰포트 클리핑
- 점, 선, 영역, 텍스트 클리핑
- 점 클리핑
  - 다음 식을 만족하는 점 P(x, y)만 그린다.

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



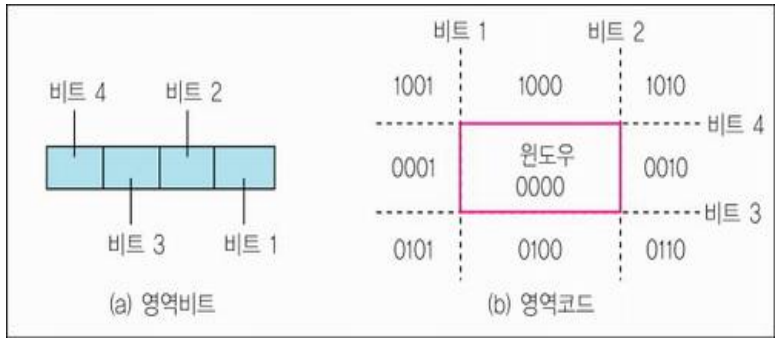
클리핑의 종류

선 클리핑

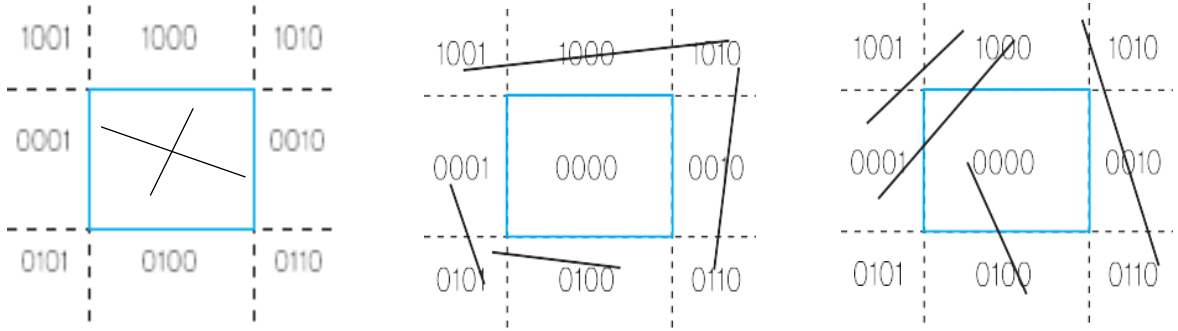


윈도우와 선분의 위치 관계 및 선 클리핑

- Cohen-Sutherland 알고리즘
  - (1) 선분의 양끝점의 영역코드를 정함
  - (2) 양끝점의 영역코드가 모두 0000 인 경우 → Accept
    양끝점의 영역코드의 logical AND 가 0000 이 아닌 경우 → Reject
  - (3) 윈도우의 경계와 교차하는 경우 ((2)의 경우에 해당하지 않으면)
    윈도우 경계에서 수직 또는 수평 분할
    수직경계에 걸치는 경우: $x = x_L$ 또는 x_R에 대해 $y = y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x - x_1)$
    수평경계에 걸치는 경우: $y = y_B$ 또는 y_T에 대해 $x = x_1 + \frac{x_2 - x_1}{y_2 - y_1}(y - y_1)$
  - (4) 과정 (1), (2)를 반복.

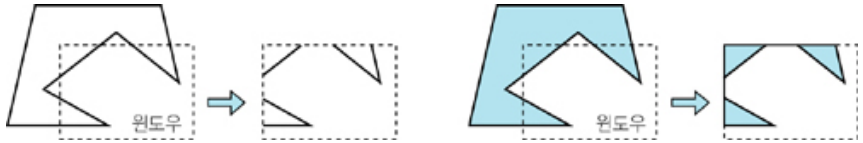


선 클리핑을 위한 영역 비트와 영역 코드



윈도우와 선분의 3가지 위치 관계

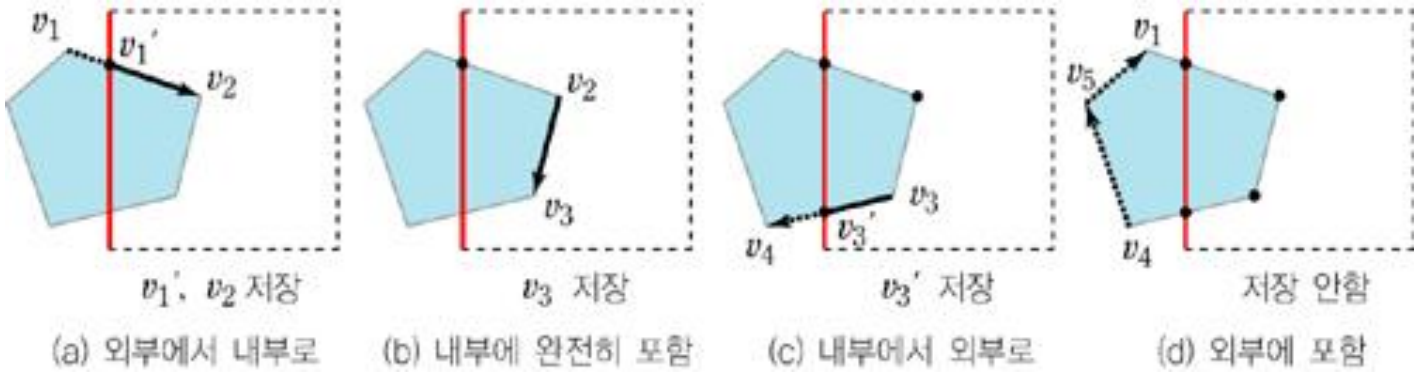
영역 클리핑



다각형 클리핑의 문제점

- 속이 빈 다각형(Hollow polygon) : 선 클리핑 알고리즘 적용
- 속이 찬 다각형 : 몇 개의 Closed filled polygon 생성
  - 선 클리핑 적용 후 닫힌 다각형(들)로 구성, 색을 채워야 함
  - Sutherland-Hodgman 알고리즘

(1)각 윈도우 경계(상하좌우)에 대하여 아래 알고리즘을 적용
(2) Filled polygon 을 라인들로 표현
(3) 4 가지 경우로 구분하여 다각형 꼭지점을 재구성 : 시계 방향으로 다각형의 에지와 윈도우 경계가 만나는 점을 구한다.
(4) 각 윈도우 경계 (left, right, top, bottom) 에 위의 과정을 반복 적용



텍스트 클리핑

- 전체 텍스트 클리핑
- 전체 문자 클리핑
- 개별 문자 클리핑

(a) 원래 텍스트의 위치 (b) 전체 텍스트 클리핑 (c) 전체 문자 클리핑 (d) 개별 문자 클리핑