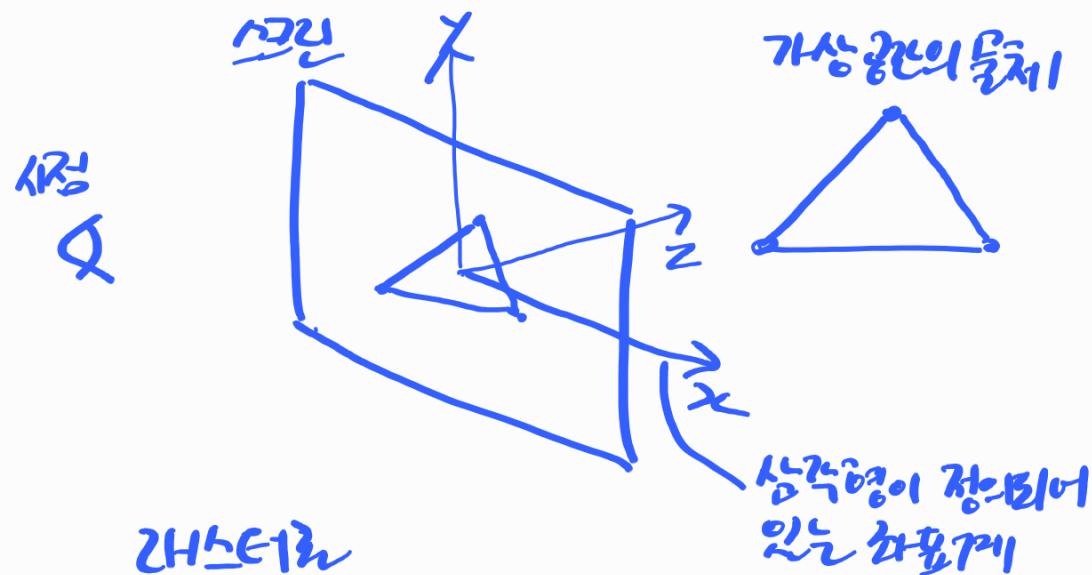


# 삼각형 렌더링 - 원소 전송계 (DirectX가 사용)

## 전송계



- 1) 3차원 공간의 물체를 2차원으로 투영(Projection)한다.  
 ↗ 삼각형으로 물체를 표현하고자 시장점(Vertices)만 투영하니 삼각형을 투영한것과 같다.  
 ↗ 좌표계 변화를 예룬다.

World Coordinate  $\rightarrow$  Screen Coordinate

3차원 월드 좌표계  $\rightarrow$  2차원 스코린 좌표계

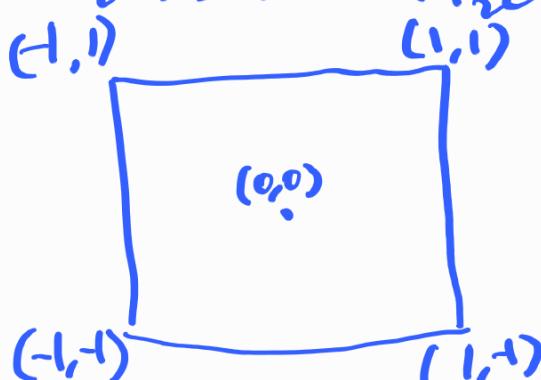
2D는 월드 좌표계의 원점이 화면의 중심이라고 가정한다.

## NDC (Normalized Device Coordinate, 정규화된 좌표계)

- Orthographical Projection(평행영)으로 가정하기

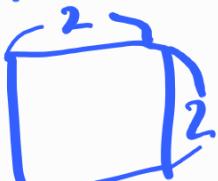
좌표계는 절대적이 나름인데 NDC는 DirectX에서 스코린 좌표계의 범위를  $[-1, 1] \times [-1, 1]$ 로 정의 것이다.

실제 모니터의 비율은 16:9, 4:3 등 각각 다른 비율로 C언어에



실제 모니터의 케이지 NDC를 정의해 유타임 좌표계로 사용하는

기준 가로:세로, 1:1 비율이 되도록.



각 픽셀의 비율

$$xscale = 2.0f / width$$

$$yscale = 2.0f / height$$

3차원 좌표계  $\rightarrow$  2차원 좌표계 (NDC) - Orthographic Projection

NDC  $[-1, 1] \times [-1, 1]$ , vec3 point = 3차원 좌표

aspect(화면 비율) = width / height

NDC의 화면 비율은 1:1이거나 aspect를 통해 나눠 비율을 맞춘다.

정수화하기에 Z좌표를 없애는 것만으로 화면으로 끌어 가는데

vec2 pointNDC = vec2(point.x/aspect, point.y), 1:1 비율

NDC의 Z는 0.0f 이므로 정수화 가능

가로: 세로 1:1 이기 때문에 각 폭/높이의 비율은

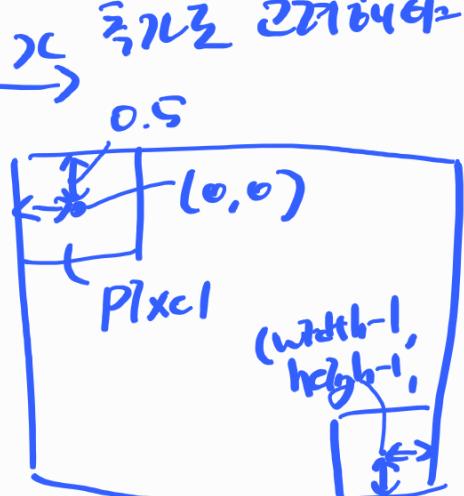
$$xscale = 2.0f / width$$

$$yscale = 2.0f / height$$

NDC  $\rightarrow$  2D 화면 좌표계로 다시 변환(픽셀의 좌표계)

2D 화면 좌표계  $[-0.5, width-1+0.5] \times [-0.5, height-1+0.5]$

2D 화면 좌표계는 픽셀의 좌표계와 같은데 픽셀의 x는 픽셀의  
중앙에 있다. 그래서 으깨운 절차를 고려해보면 0.5 벡터를  
추가로 고려해보자는데



$$\begin{aligned} -0.5 \leq x \leq 1 &\Rightarrow 0 \leq x + 1 \leq 2 \Rightarrow 0 \leq \frac{x+1}{xscale} \leq width \\ &\Rightarrow -0.5 \leq \frac{x+1}{xscale} - 0.5 \leq width - 0.5 \end{aligned}$$

$$vec2.x = (NDC.x + 1.0f) / xscale - 0.5f$$

$$vec2.y = (vec3.y + 1.0f) / yscale - 0.5f$$

화면 좌표의 y는 맨 위를 0으로 가는 방향으로  
증가하는 경우이다.

$$vec2.y = (1.0f - NDC.y) / yscale - 0.5f$$

NDC  $\rightarrow$  2D 화면 좌표계

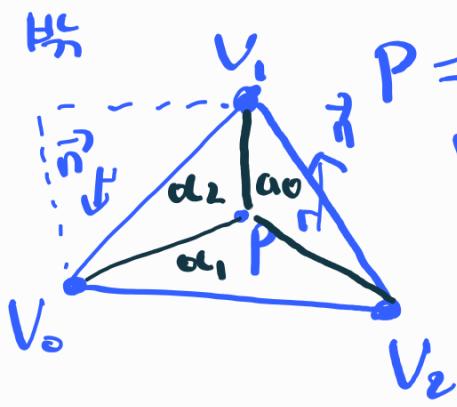
$[-1, 1] \times [-1, 1] \rightarrow [-0.5, width-1+0.5] \times [-0.5, height-1+0.5]$

vec2 ScreenPoint = Vec2((pointNDC.x + 1.0f) / xscale - 0.5f,  
 $(1.0f - pointNDC.y) / yscale - 0.5f)$

## 2차원 삼각형

2) 스코프 진동계에서 특정한 물체의 색을 결정하는.

삼각형 정리 - 목재정상화



$\alpha_0, \alpha_1, \alpha_2$  모두

0 이상이면 삼각형 내부  
(3차원에서 2 vector의 방향이  
같아지면 두 벡터의 내적부분이)

$$P = w_0 V_0 + w_1 V_1 + w_2 V_2, \text{Affine Combination}$$

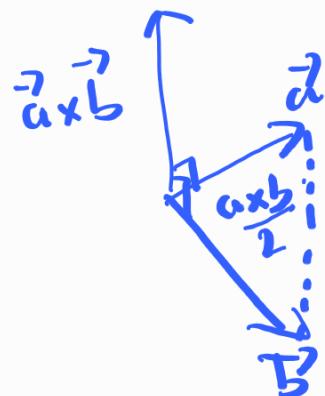
$$w_0 + w_1 + w_2 = 1$$

$$w_0 = \frac{\alpha_0}{\alpha_0 + \alpha_1 + \alpha_2}, \alpha_0, \alpha_1, \alpha_2 \text{ 삼각형의 넓이}$$

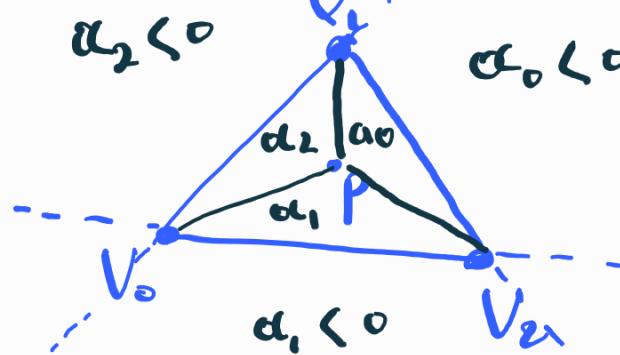
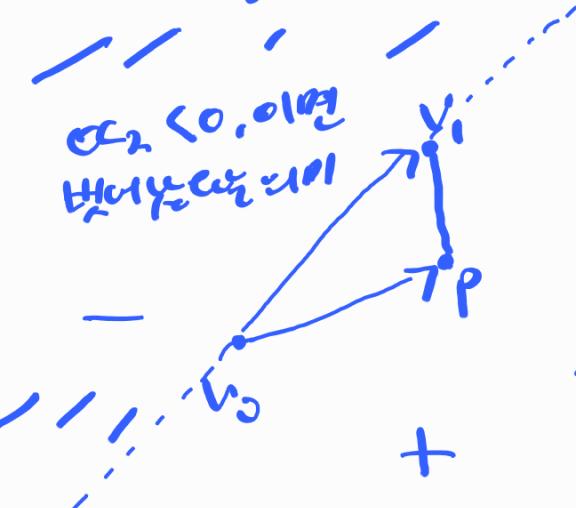
$$w_1 = \frac{\alpha_1}{\alpha_0 + \alpha_1 + \alpha_2}$$

$$w_2 = \frac{\alpha_2}{\alpha_0 + \alpha_1 + \alpha_2}$$

Edge Function - 삼각형 넓이



$$\alpha_2 = (V_1 - V_0) \times (P - V_0) / 2$$



$\vec{v}_c$  와 외적  $\vec{a} = (ax, ay, az), \vec{b} = (bx, by, bz)$

$$\vec{a} \times \vec{b} = (ay \times bz - az \times by, az \times bx - ax \times bz, ax \times by - ay \times bx)$$

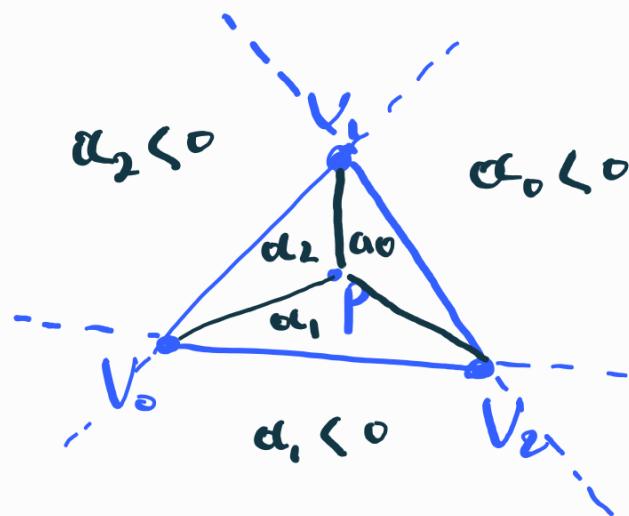
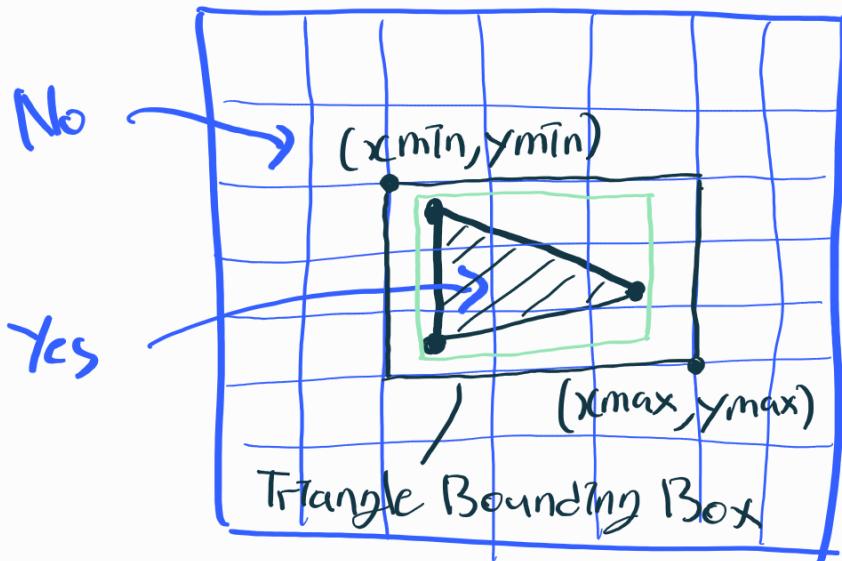
3차원에서 2차원으로 변환하기에 고려해야 한다.

$$\therefore \vec{a} \times \vec{b} = (0, 0, ax \times by - ay \times bx) \text{로}$$

2차원 삼각형의 넓이는  $\frac{\vec{a} \times \vec{b}}{2} = \frac{ax \times by - ay \times bx}{2}$  가 된다.

계산 시 음수, 양수로 나누는 일면적으로 2로 나누기를 필요로 한다.

친적화를 위해 Bounding Box를 통해 이면의 Pixel들의  
좌표들만 들면 스크린의 모든 Pixel을 들지 않아도 된다.



$$x_{\min}, y_{\min} = \text{clamp}\left(\text{floor}(m \ln(v_0, v_1, v_2)), 0, \frac{\text{width}-1}{\text{height}-1} \text{ or } \right)$$

이면을  
구하기

$$x_{\max}, y_{\max} = \text{clamp}\left(\text{ceil}(\max(v_0, v_1, v_2)), 0, \frac{\text{width}-1}{\text{height}-1} \text{ or } \right)$$

이면  
구하기

for ( $y_{\min} \sim y_{\max}$ )

for ( $x_{\min} \sim x_{\max}$ )

vec2 point( $x, y$ )

If ( $\alpha_0 > -1 \text{ || } \alpha_1 > -1 \text{ || } \alpha_2 > -1$ ), 삼각형 연인지 확인  
삼각형 Pixel 색칠 = 물체 중심좌표를 이용해 색칠장

$$C = w_0 C_0 + w_1 C_1 + w_2 C_2$$

이전의 사용 Perspective Projection으로 depth 값을 고려하여 하는  
Orthographic Projection은 물체 중심좌표를 이용해 구현된다.