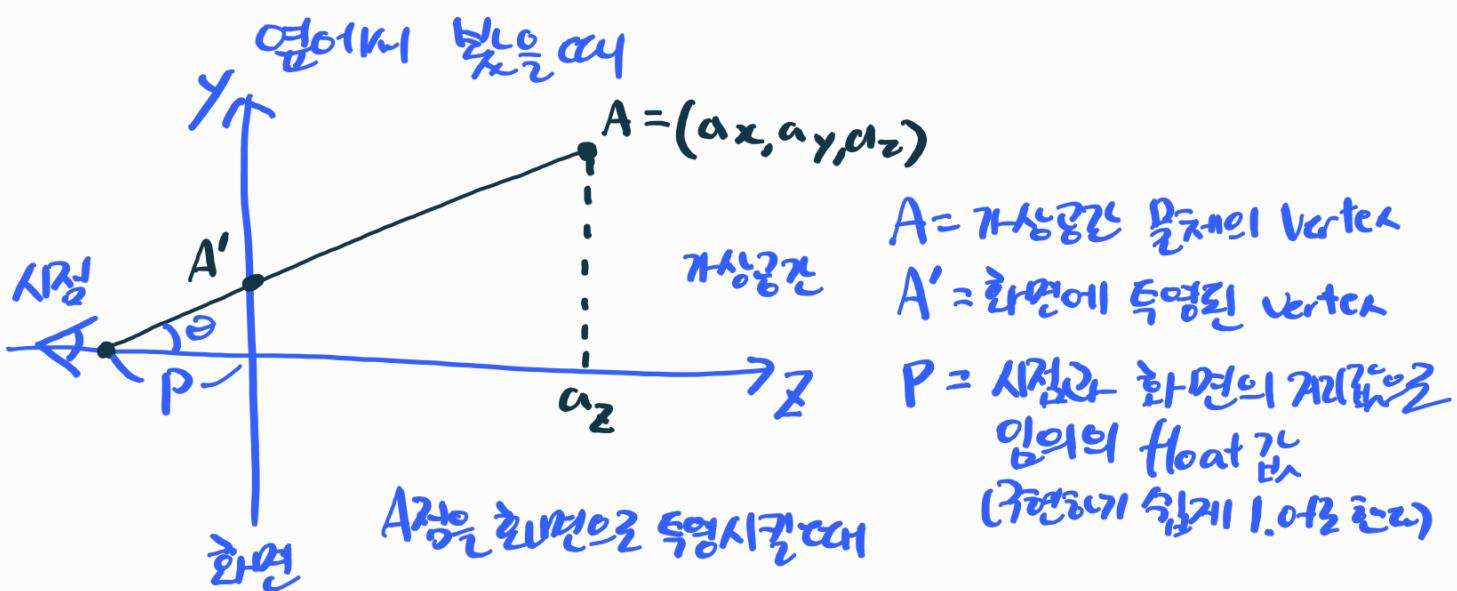
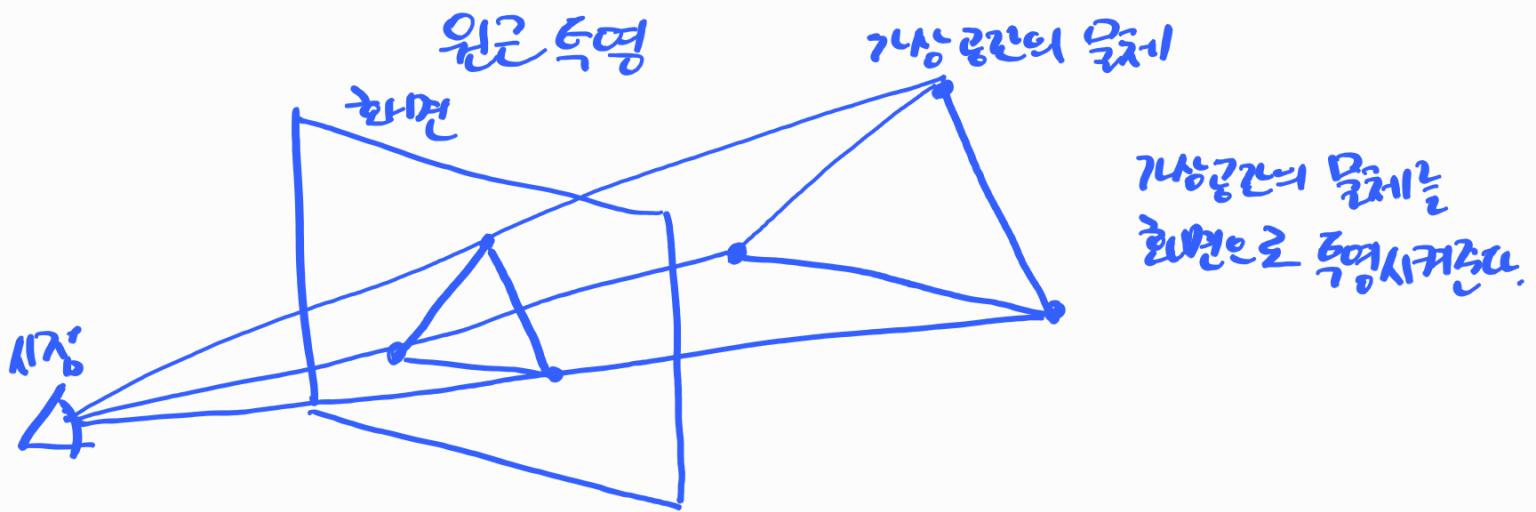


원근 투영(Perspective Projection)

Rasterization은 Raytracing과 마찬가지로 물체를 화면에 투영하는 데
간접방법은 냉식이므로 정투영(Orthographic Projection)을 원근투영으로
바꿔서 구현은 어렵지 않으나 시각적으로 왜곡이 일어난다.
그래서 원근 투영을 할 때는 왜곡을 수정해 줄 수 있어 제대로 간접방법이다.



$$A' = (a'_x, a'_y, a'_z), \text{ 투영된 점}$$

$$a'_z = 0, \text{ 화면위의 한 점으로}$$

$$\frac{a'_y}{P} = \frac{a_y}{P+a_z}, \text{ 비례식}$$

$$a'_y = \frac{a_y}{P+a_z} \times P$$

a'_x 도 위에서 관계식을 봤을 때 비례식
 a'_y 를 찾았을 때와 같이 구할 수 있다

$$\frac{a'_x}{P} = \frac{a_x}{P+a_z}, \text{ 비례식}$$

$$a'_x = \frac{a_x}{P+a_z} \times P$$

원근 투영

1) 가상의 물체의 vertex를 화면에 투영한다

$$\text{const float scale} = p \times (p + a_z);$$

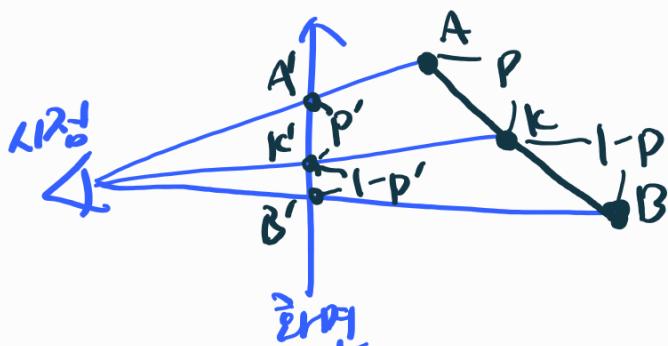
$$\text{vec2 projPt} = \text{vec2}(\text{worldPt}.x, \text{worldPt}.y) \times \text{scale};$$

2) NDC로 변환 $[-1, 1] \times [-1, 1]$

3) NDC \rightarrow 2NDC 좌면좌표계로 변환

4) 렌더링 (화면 챕팅)

외곽이 생기는 이유 - 원근 투영



Rasterization에서 물체를 투영할 때 Pixel의 color, uv, depth 등을 interpolation 한다.

K' 같은 A' 와 B' 의 interpolation에서 3차원 공간상에서의 K' 같은 2D 원자리는 실제로 원하는 같은 3차원 공간상에서의 K' 같은이다.
3차원 공간상의 A 와 B 를 이용해 K 를 3차원상에서 찾는 것이다.

문제는 A 와 B 의 Z (화면으로부터의 거리)가 다르다는 점이다.

그래서 $P' \neq P$, $1-p' \neq 1-p$ 가 된다.

정밀영에서는 그간의 거리를 신경하지 않아서 상관없지만 원근 투영은 상관 있다
기 때문에 $P' \neq P$, $1-p' \neq 1-p$ 인 경우에 A' 와 B' 를 interpolation 하면
정밀한 같은 interpolation하게 된다.

! 가상화(시점)와 vertex의 거리와 depth(depth)는 같다.

depth Buffer에 저장된 그림의 거리를 계산하는 것이다. (화면상의 역행렬)

depth = 현재 Pixel의 깊이값, 거리 = 시점과 vertex의 Z 거리값

외적 해결 방법

Bary-centric coordinates(무게중심좌표)를 기준에 대해 보정한다.
정확히는 가중치(w_0, w_1, w_2)에 대해 보정해준다.

$$\text{Area} = \frac{w_0}{\zeta_0} + \frac{w_1}{\zeta_1} + \frac{w_2}{\zeta_2}, \quad \text{보정하는 값}$$

$$w'_0 = \frac{w_0 / \zeta_0}{\text{Area}}$$

w_0, w_1, w_2 = 원래 무게중심좌표의 계산

$$w'_1 = \frac{w_1 / \zeta_1}{\text{Area}}$$

w'_0, w'_1, w'_2 = 보정한 가중치

$$w'_2 = \frac{w_2 / \zeta_2}{\text{Area}}$$

$$\begin{aligned}\zeta_i(\text{제타}) &= \text{시점과 Vertex } V_i \text{의} \\ &\text{Z축 위에서의 거리} \\ &= V_i.Z + P\end{aligned}$$

보정한 가중치들을 사용해 color, uv, depth 등의 Interpolation 값을 구한다.

원근 투영

1) 가상의 물체의 vertex를 화면에 투영한다

$$\text{const float scale} = P \times (P + a_Z);$$

$$\text{vec2 projPt} = \text{vec2}(\text{worldPt}.x, \text{worldPt}.y) \times \text{scale};$$

2) NDC로 변환 $[-1, 1] \times [-1, 1]$

3) NDC \rightarrow 2NDC 좌면좌표계로 변환

4) Backface Culling

5) Bounding Box

6) 무게중심좌표의 가중치 계산

7) 외적 해결을 위해 가중치값 보정

8) depth, color, uv를 구한뒤 앞면과 뒷면에 있는 물체 렌더링