컴퓨터 그래픽스 (목) PI 1차시

BASIC CONCEPT & 2D Transformation

5:00PM에 시작됩니다

입장 후 채팅창에 학번/이름 작성 부탁드립니다.

감사합니다.

BASIC CONCEPT

Rendering이란?

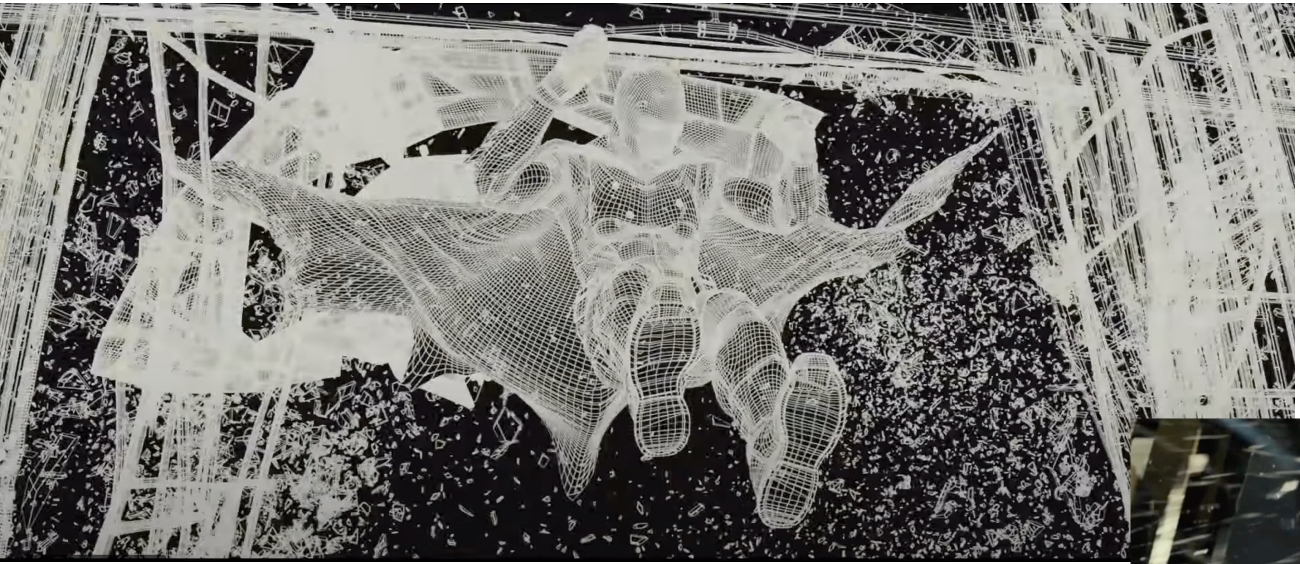
Graphics rendering pipeline(==Rendering STEP)

기본적인 Rendering 알고리즘

Rendering이란?

Rendering?

3차원 scene을 2차원의 이미지로 만들어가는 과정



Rendering?

scene

Model

...

...

...

Model

primitives

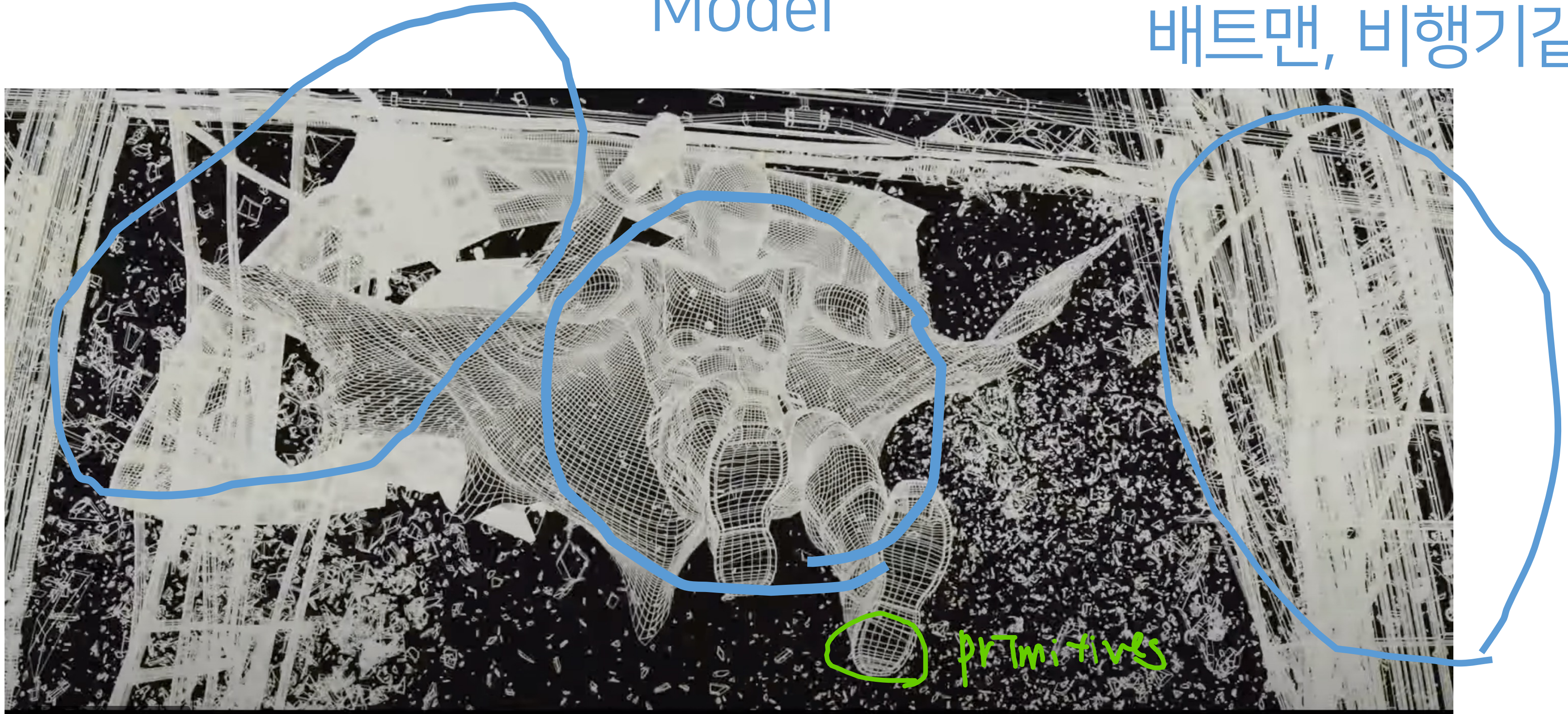
...

...

...

primitives

배트맨, 비행기같은 거, 창문

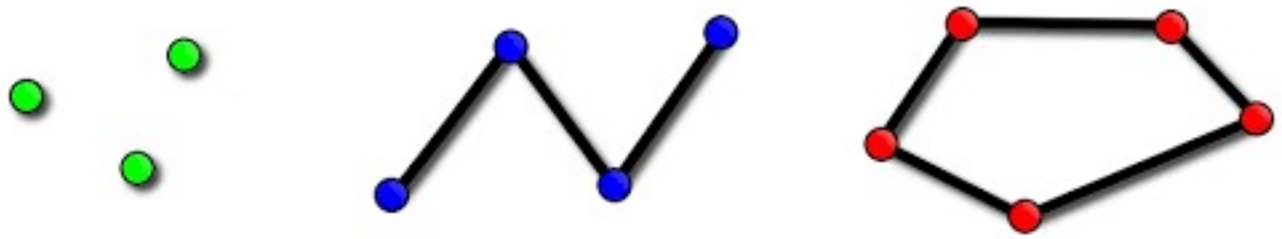


Primitives?

컴퓨터 그래픽스에서 그래픽스 프로그램(open gl등)에 의해 개별적인 실체로 그려지고,

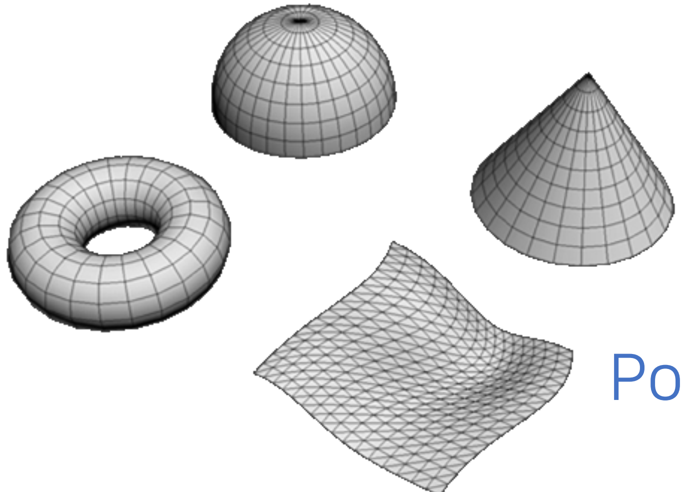
저장 조작될수 있는 선 / 원 / 곡선 / 다각형과 같은 그래픽 디자인을 창작하는데 필요한 요소

하드웨어가 바로 만들어 주는 것들



Polygon(다각형)

아닌것들..



Polygon을 근사하는 방법 사용

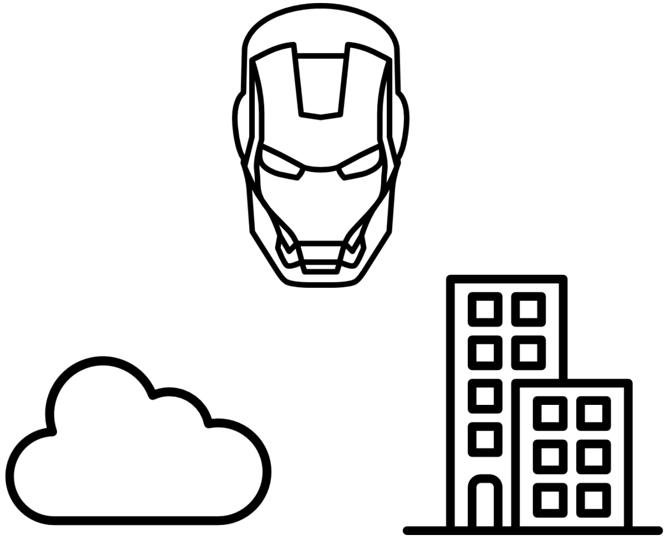
Graphics Rendering Pipeline

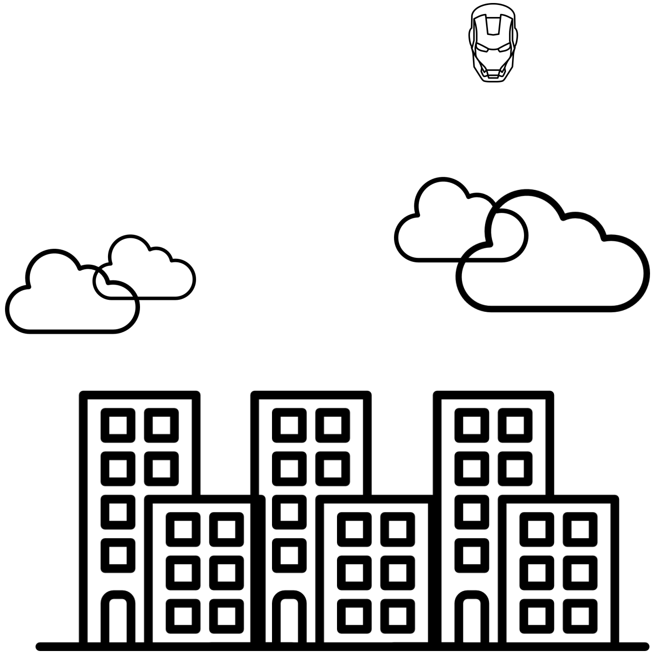
정의

구분

정의

Rendering을 위한 step





Model 을 Scene으로, scene을 image로 convert하는 과정

구분

<연산>

Modeling Transformation

Viewing Transformation

Projection Transformation

Viewport Transformation

변화가 발생할 때 일어난다!

<좌표계>

Modeling Coordinate System

World Coordinate System

Viewing Coordinate System

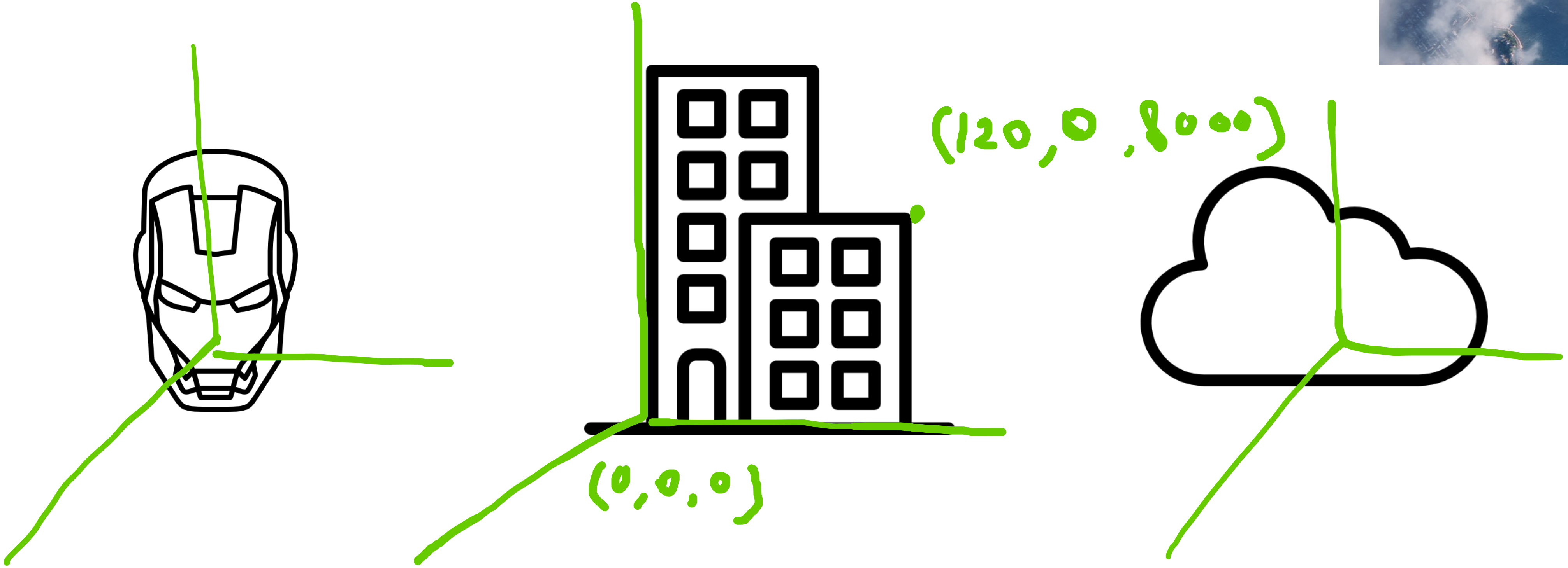
Normalized Coordinate System

Screen Coordinate System

변화는? 기준(좌표계)가 바뀌어야 할 때!

Model

Modeling Coordinate System



각자 좌표계를 가짐

Scene

Modeling Transform

(128, 80, 90000)

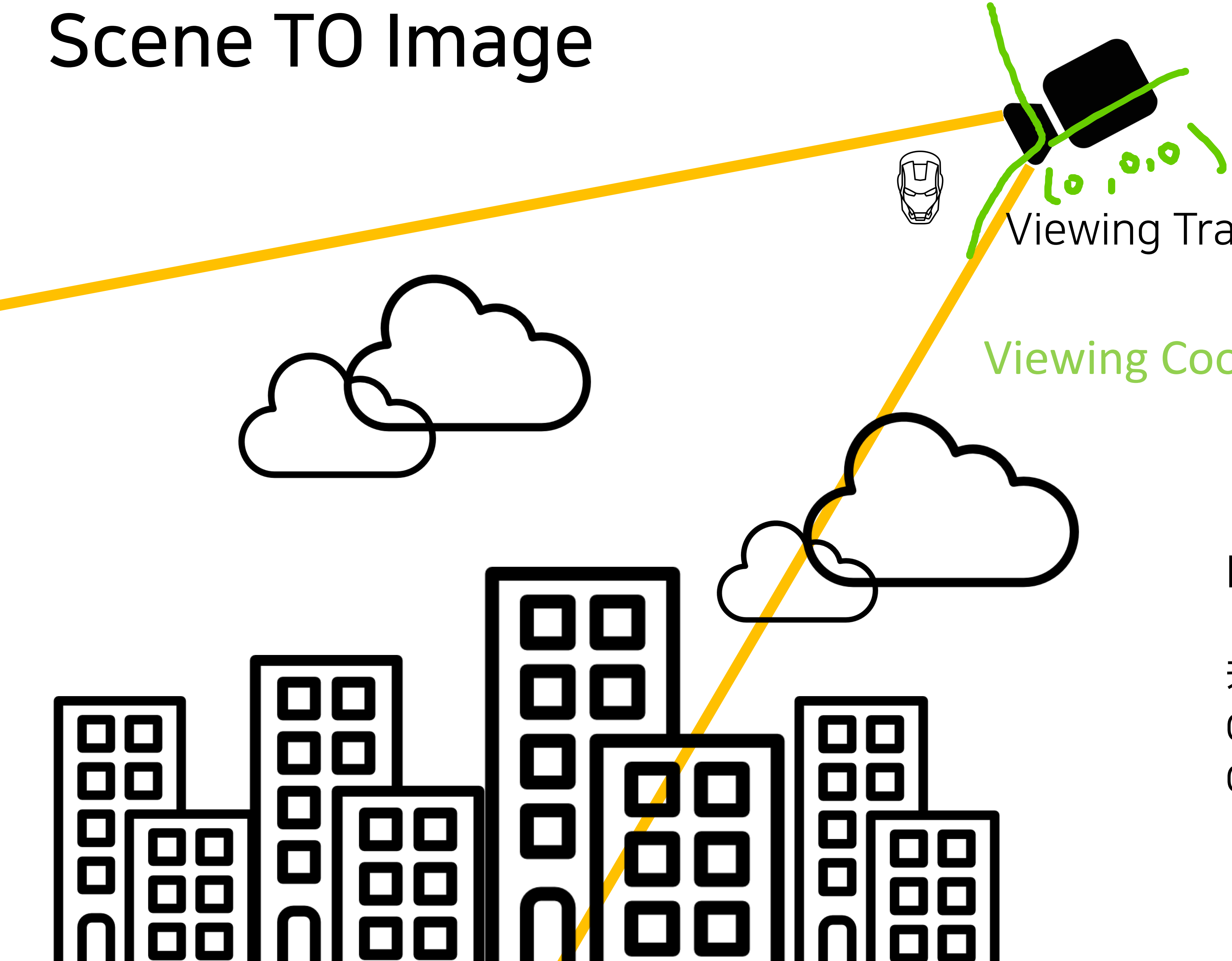
(60, 170, 810, 247)

World Coordinate System

공통된 화면에 배치

공통된 좌표계 필요

Scene TO Image



Viewing Transform

Viewing Coordinate System

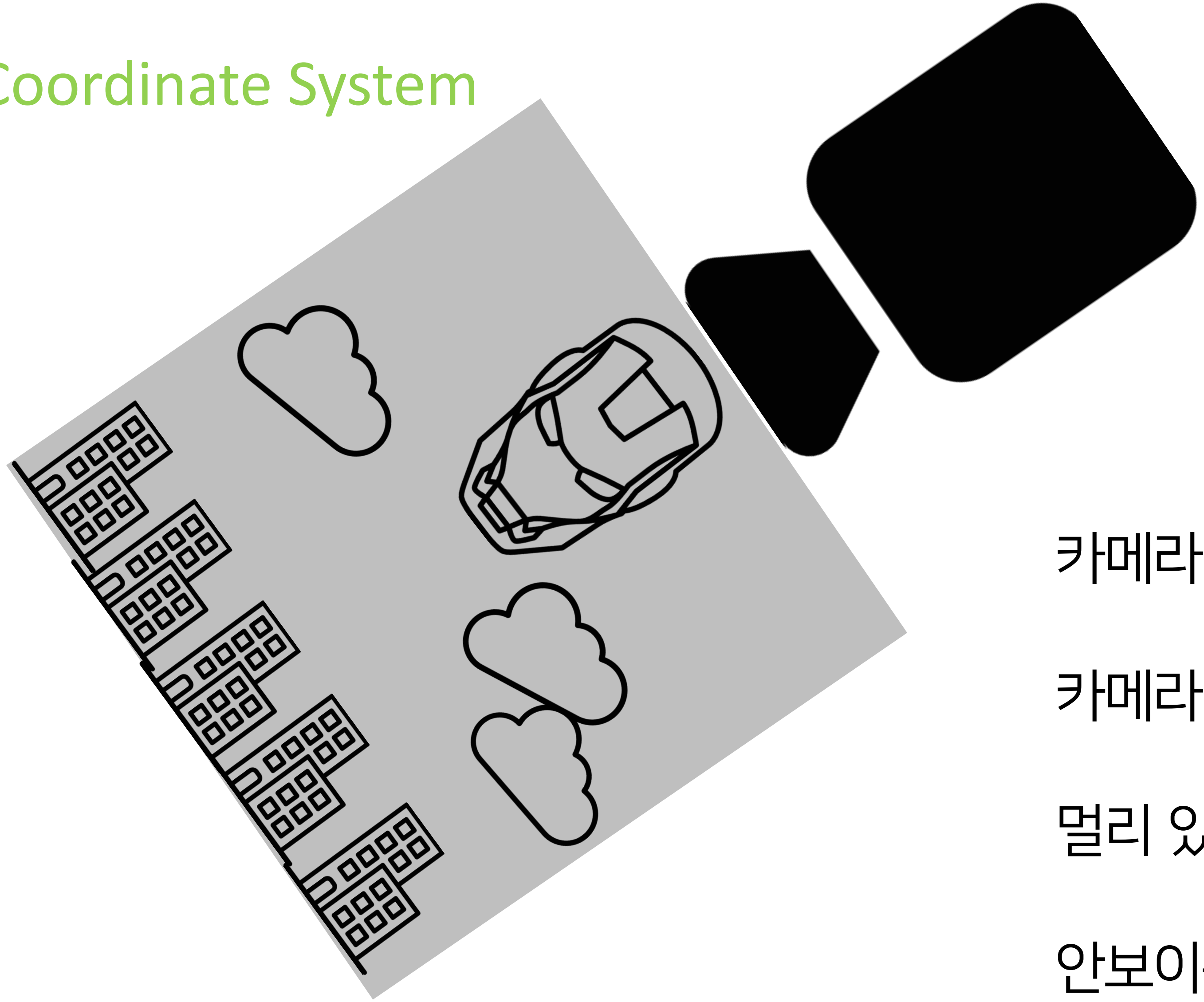
Image => 카메라 필수!

카메라 어디에 둘까?
어떤 각도로 둘까?
어느 방향으로 할까?

Scene TO Image

Projection Transform

Normalized Coordinate System



카메라가 담을 수 있는 것만 중요하다

카메라 기준 가까운 건 크게

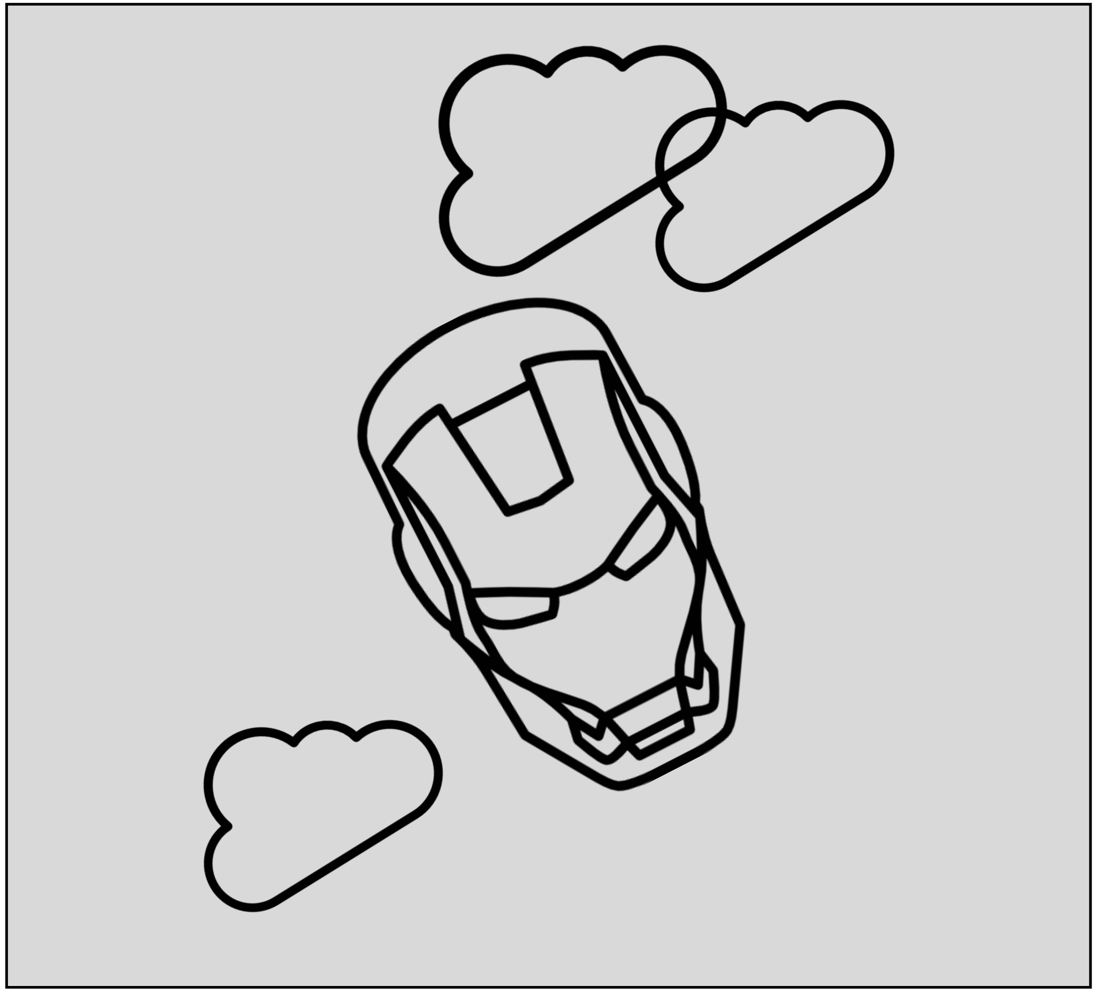
멀리 있는 건 작게

안보이는 부분은? 알 빠 아님!

Image

Screen Coordinate System

Viewport Transform



2차원 z축이 필요하지 않음

기본적인

Rendering

알고리즘

Basic Rendering Algorithm

- Transformation
- Clipping / hidden surface removal
- Rasterization
- Shading & Illumination

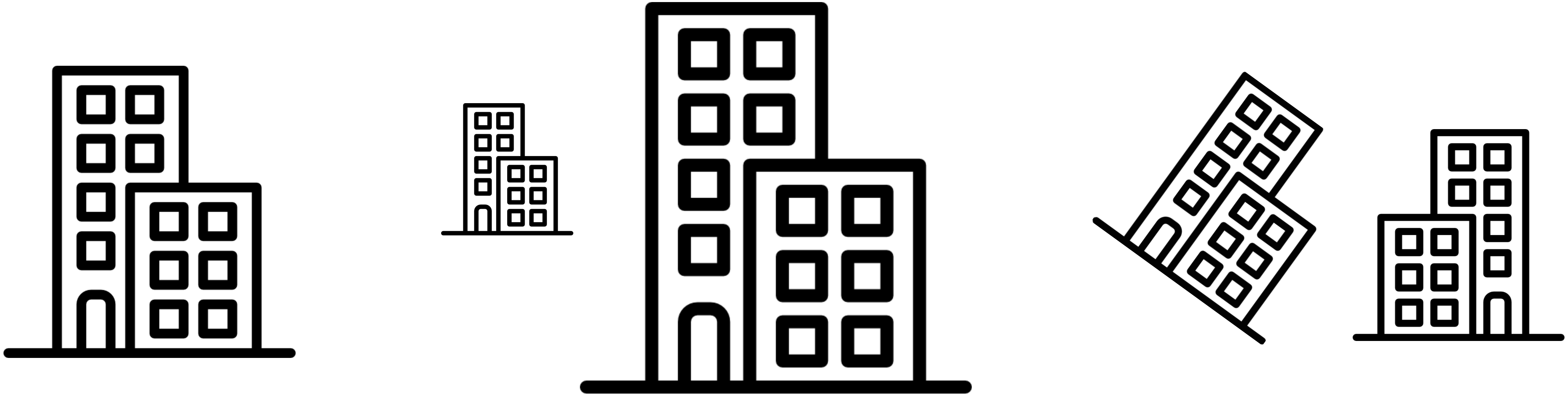
앞으로 뭘 배울 지에 대한 예고편 지금은 정확히 몰라도 된다

(정의 + 대충 이런거 구나)

Transformation

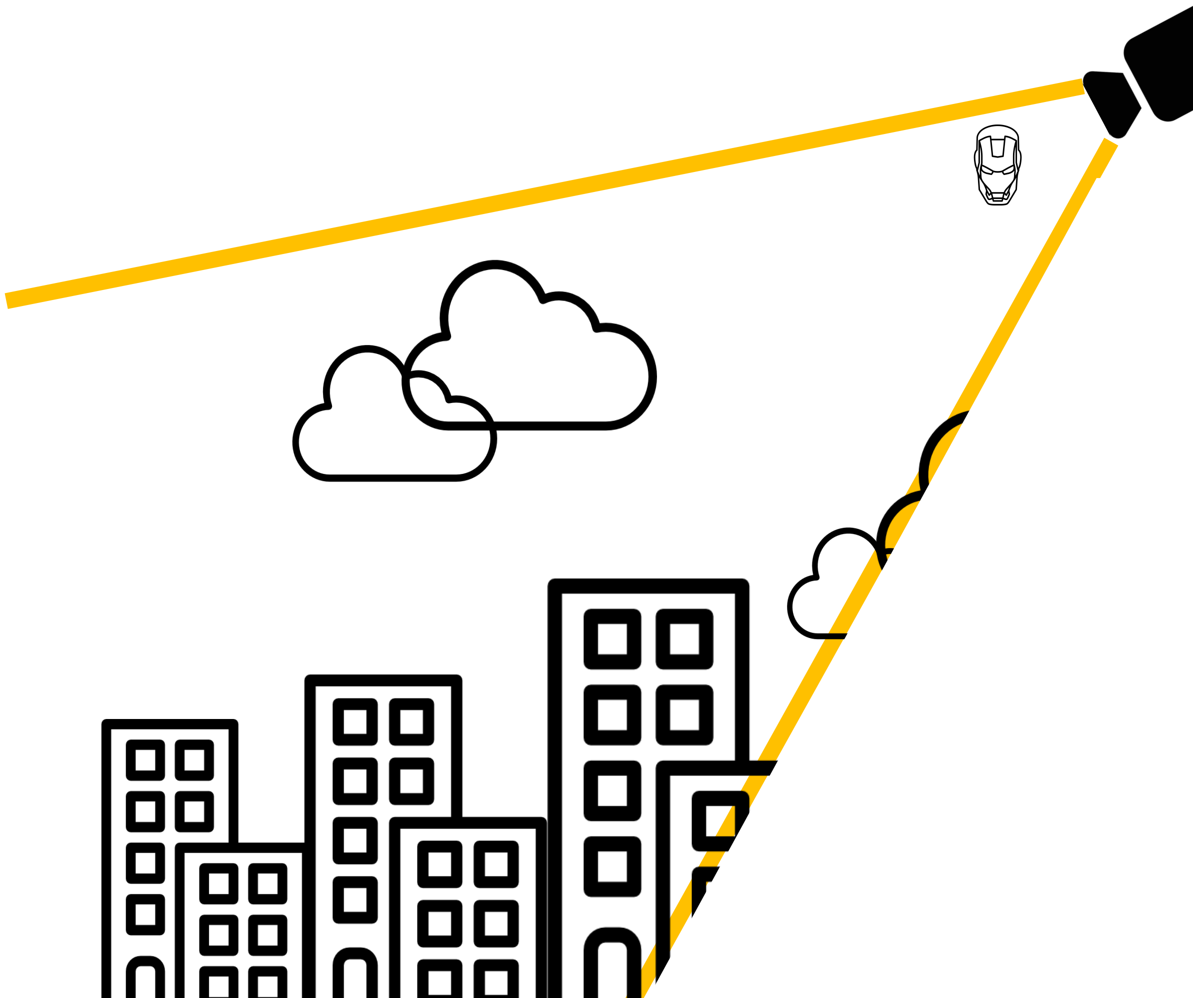
2D/ 3D
Transformation

작게? 크게? 뒤집기? 회전? 원하는 대로



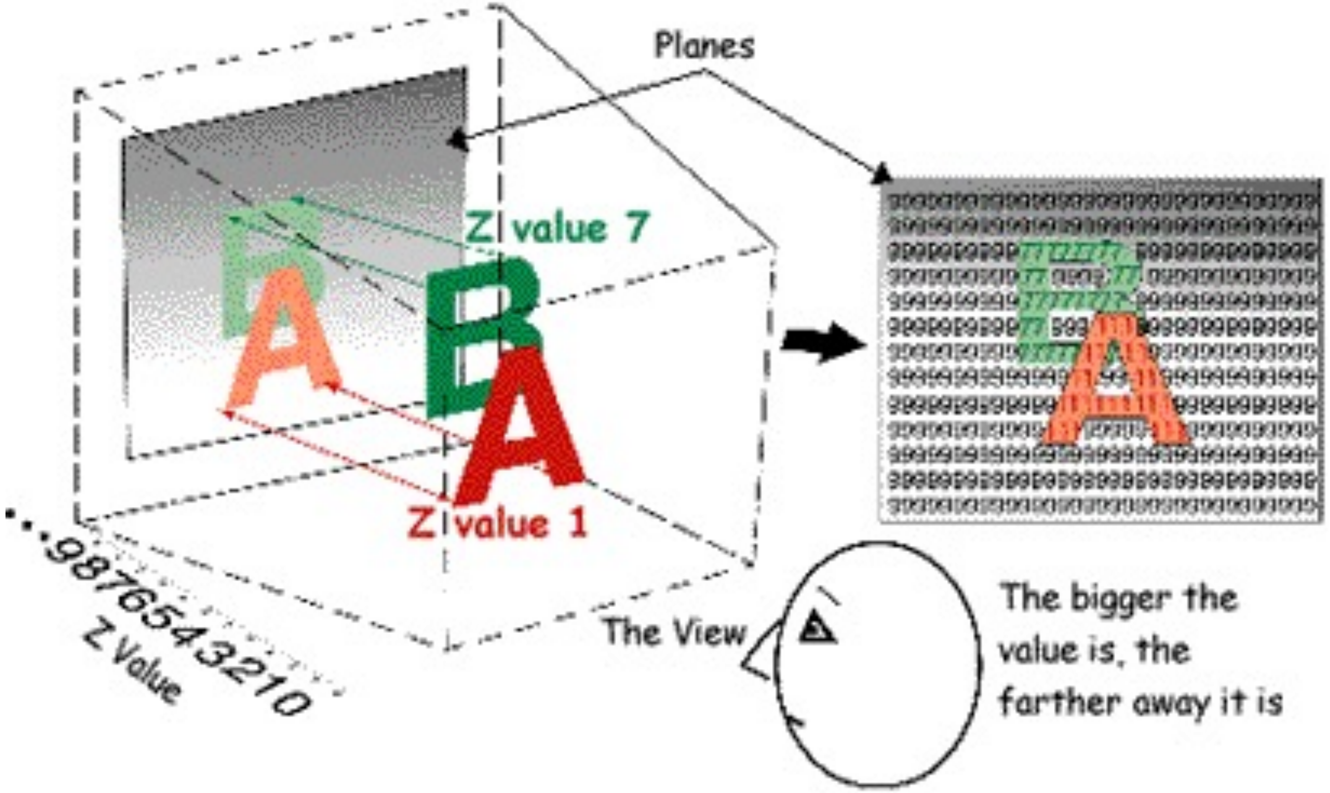
Clipping

카메라 범위(view volume) 내에 없는 것은 버린다



Hidden surface removal

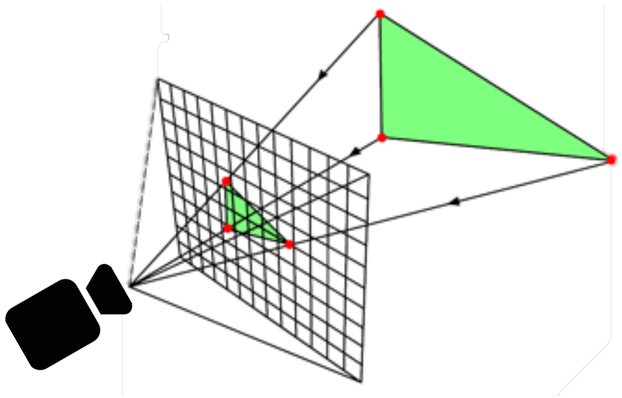
가려진 것에 대해서는 굳이 고려하지 말자

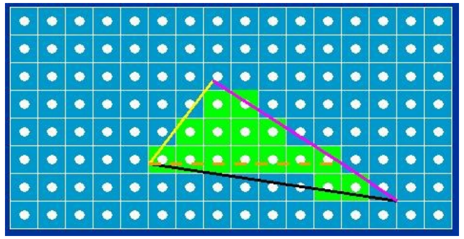


Rasterization시 각 pixel당 depth값을 hold(눈까지의 거리)
Bigger z가 smaller z에 가려지도록 설정

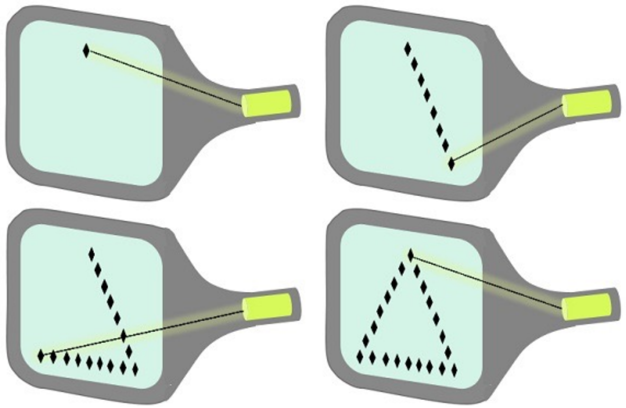
Rasterization(래스터화=픽셀화)

2차원에 projection된 screen space primitive(연속적)들을 set of pixel(불연속적)로 바꾸는 과정

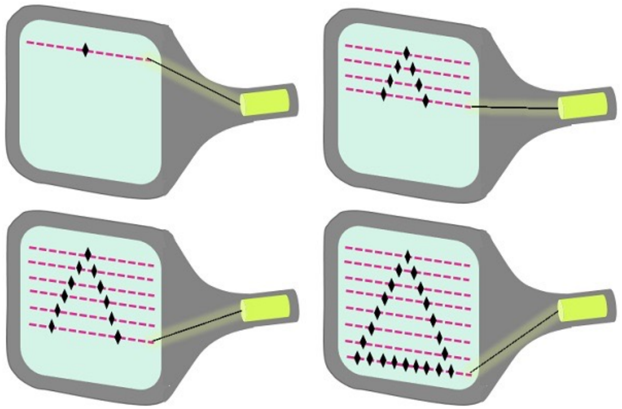




Rasterization의 결과는 frame buffer에 저장 후 다음과 같은 방식으로 보여짐(display)



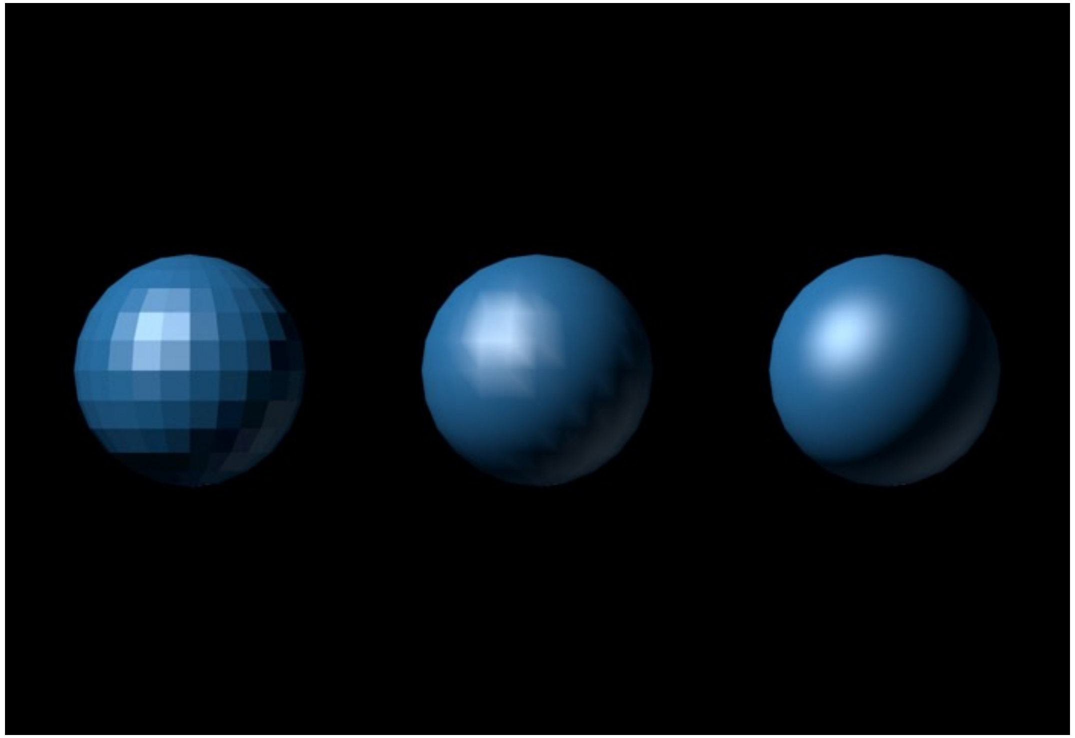
Vector display

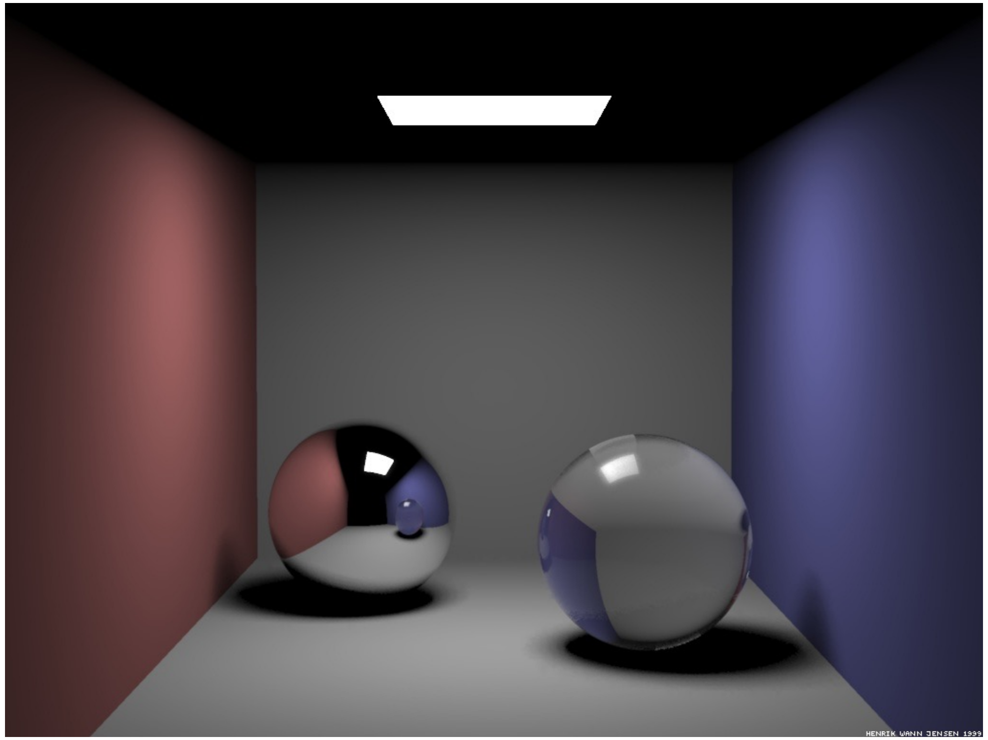


Raster display

Shading & Illumination

빛을 고려해서 해당 픽셀의 color를 결정하자





2D
Transformation

기본적인 2D Transformation

Homogenous Coordinate

Matrix Representations

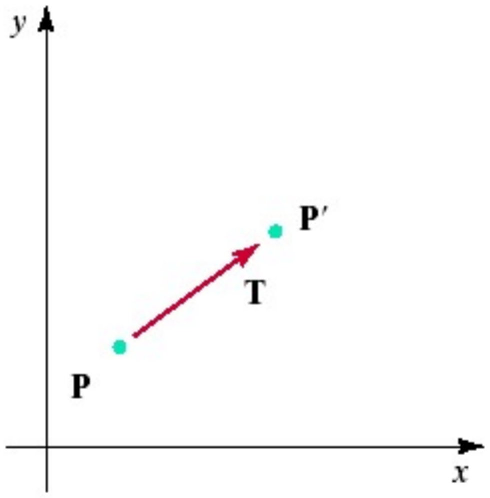
Basic 2D Transformation

Translation

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

$$\mathbf{P} = \begin{bmatrix} x \\ y \end{bmatrix}, \qquad \mathbf{P}' = \begin{bmatrix} x' \\ y' \end{bmatrix}, \qquad \mathbf{T} = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$\mathbf{P}' = \mathbf{P} + \mathbf{T}$$

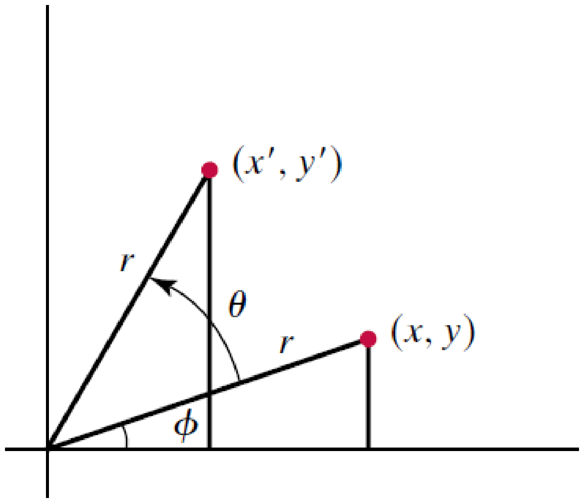


Rotation

$$x' = x\cos\theta - y\sin\theta$$
$$y' = x\sin\theta + y\cos\theta$$

$$\mathbf{P}' = \mathbf{R} \cdot \mathbf{P}$$

$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

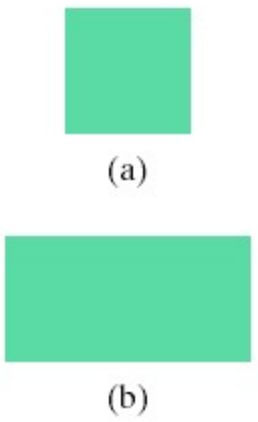


Scaling

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

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\mathbf{P}' = \mathbf{S} \cdot \mathbf{P}$$



Homogenous Coordinates

왜 쓸까?

Translation

$P' = P + T$

Rotation

$P' = R \cdot P$

Scaling

$P' = S \cdot P$

$$P' = M \cdot P$$

로 모양 통일하고 싶어서

Homogenous Coordinates

$$P' = M \cdot P$$

- Translation $T(t_x, t_y)$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{matrix} x + t_x \\ y + t_y \\ 1 \end{matrix}$$

- Rotation $R(\theta)$

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

$$\begin{matrix} x\cos\theta - y\sin\theta \\ x\sin\theta + y\cos\theta \\ 1 \end{matrix}$$

- Scaling $S(s_x, s_y)$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{matrix} x \cdot s_x \\ y \cdot s_y \\ 1 \end{matrix}$$

Matrix Representation

Inverse : 지금 까지 한 것에 역 연산

Composite : 지금 까지의 연산은 원점기준 , 원점이 아닌 임의의 점기준으로 연산

Shear 왜곡

Reflection : 반전

$$P' = M \cdot P$$

Inverse

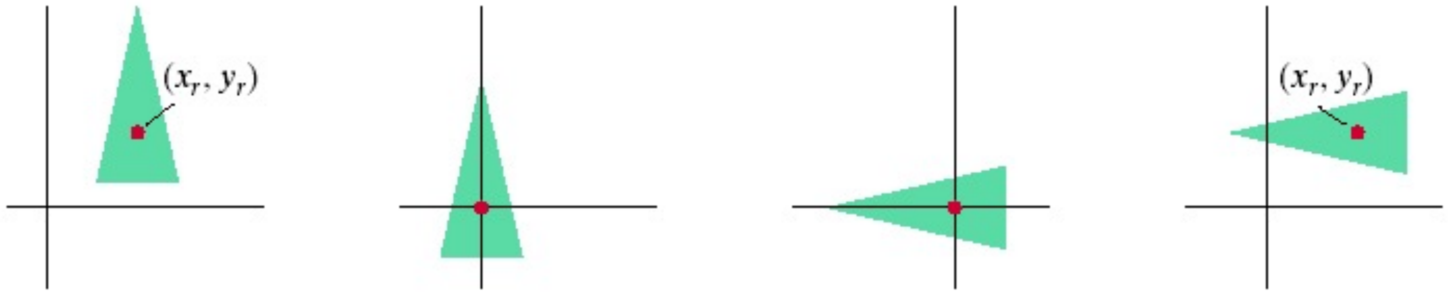
Original

$$\begin{matrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{matrix} \qquad \begin{matrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{matrix} \qquad \begin{matrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{matrix}$$

Inverse

$$\mathbf{T}^{-1} = \begin{bmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{R}^{-1} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{S}^{-1} = \begin{bmatrix} \frac{1}{s_x} & 0 & 0 \\ 0 & \frac{1}{s_y} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Composite



(a)
Original Position of Object and Pivot Point

(b)
Translation of Object so that Pivot Point (x_r, y_r) is at Origin

(c)
Rotation about Origin

(d)
Translation of Object so that the Pivot Point is Returned to Position (x_r, y_r)

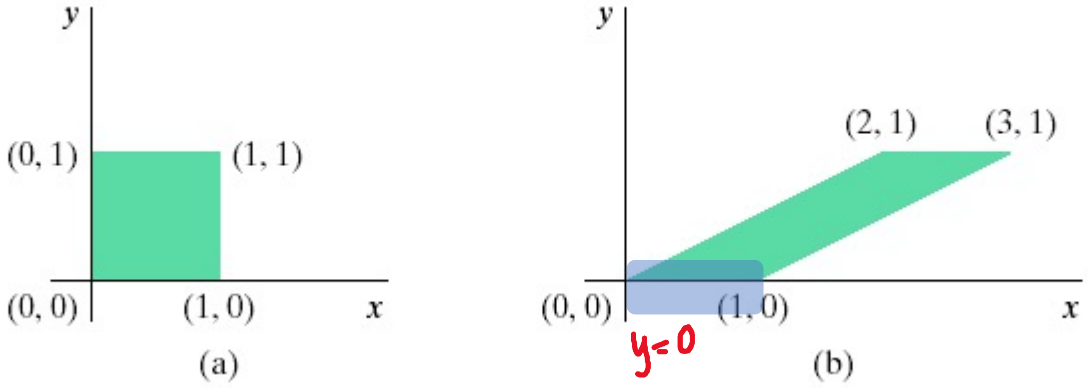
기준점

$$\mathbf{T}(x_r, y_r) \cdot \mathbf{R}(\theta) \cdot \mathbf{T}(-x_r, -y_r) = \mathbf{R}(x_r, y_r, \theta)$$

진행순서 연산순서 반대

Shear

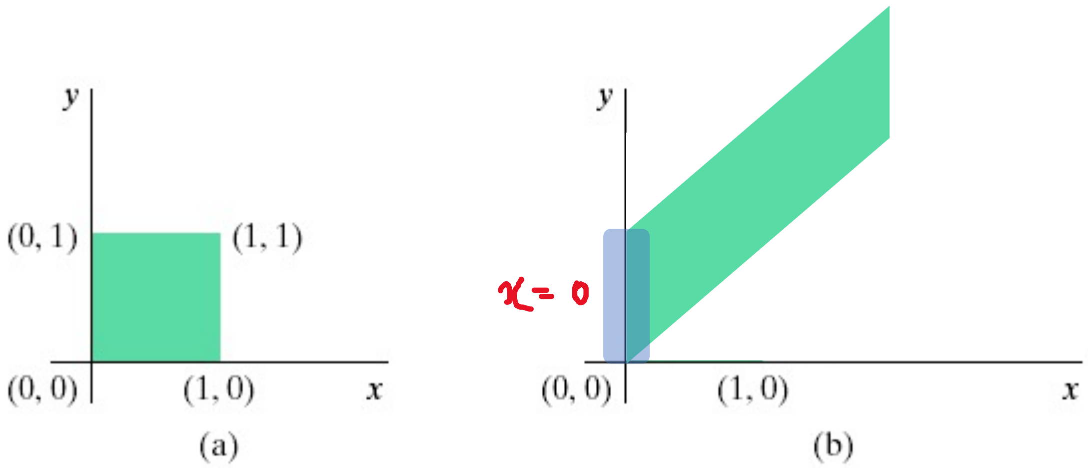
x-shearing



$$x' = x + sh_x \cdot y, \qquad y' = y$$

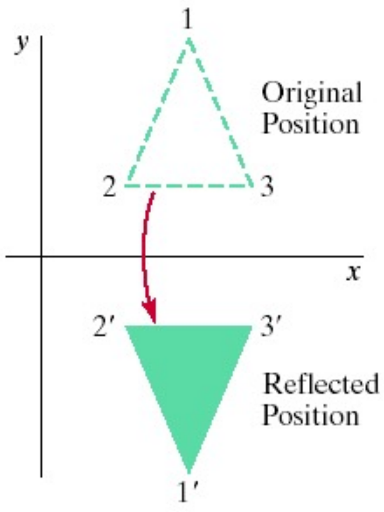
$$\begin{bmatrix} 1 & sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

y-shearing

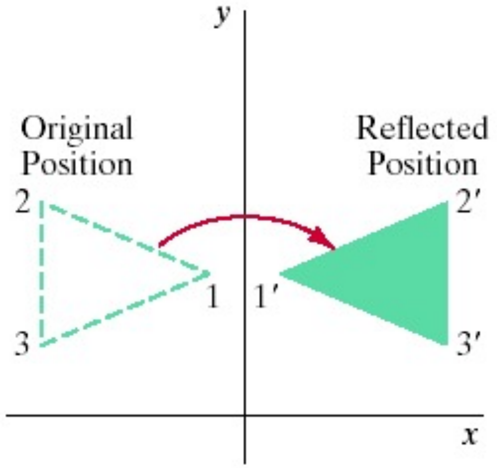


$$\begin{matrix} 1 & 0 & 0 \\ sh_y & 1 & 0 \\ 0 & 0 & 1 \end{matrix}$$

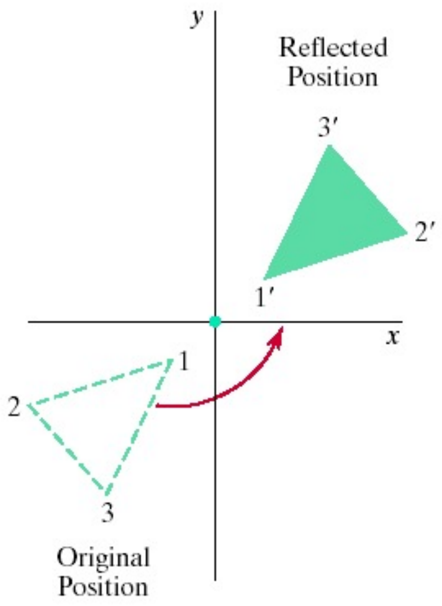
Reflection



$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

X-reflection

Y-reflection

Z-buffer ?

Vector display vs Raster display?

Modeling Coordinate System vs World Coordinate System?

Alpha channel?

하나의 scene에 대하여 MCS는 여러개 있어도 되는 반면에, WCS는 하나만 존재해야 한다.

Vector display는 현재 가장 잘 사용되는 display system으로, 이는 벡터를 효과적으로 다룰 수 있게 해준다.

Homogeneous transformation은 항상 matrix를 이용하여 나타내질 수 있다.

Z-buffer ?

Vector display vs Raster display?

Modeling Coordinate System vs World Coordinate System?

Alpha channel? 투명도

모델별로

하나의 scene에 대하여 MCS는 여러개 있어도 되는 반면에, WCS는 하나만 존재해야 한다. T

~~Vector~~ display는 현재 가장 잘 사용되는 display system으로, 이는 벡터를 효과적으로 다룰 수 있게 해준다. Raster F

Homogeneous transformation은 항상 matrix를 이용하여 나타내질 수 있다. T

Q & A

사전 질문은 여기에 올려주시면 감사하겠습니다