2023-2 Computer Graphics PI 1차시

Basic Concept & 2D Transformation

AM11:00에 시작됩니다.

입장 후 채팅창에 학번/이름 작성 부탁드립니다.

PI 수업이란?

PI 수업이란?

- Peer Instructor가 수업 복습, 기출 문제 풀이, 질의 응답 등을 진행

- 입장 후 채팅창에 학번/이름 작성 부탁드립니다.

- 질문

카카오톡 오픈 채팅방에서 미리 받습니다.
원활한 진행을 위해 사전에 받은 질문만 답해드리는 점 양해 부탁드립니다.
https://open.kakao.com/o/glocN1Hf

- 수업 자료 드라이브

https://drive.google.com/drive/folders/1nuPE0OuIQfwnm5AJ5tgIQK5zAALWx1fg?usp=share_link

- 자세한 사항은 공지사항 참고

차시	임연우 PI	김유민 PI	학 습 내 용
1	10/6/금 11:00~12:15	10/11/수 20:00~21:15	Basic Concepts, 2D Geometric Transformations
2	10/20/금 11:00~12:15	10/23/월 20:00~21:15	3D Geometric Transformations, OpenGL
3	10/27/금 11:00~12:15	10/30/월 20:00~21:15	3D viewing (Orthographic, Perspective)
4	11/3/금 11:00~12:15	11/6/월 20:00~21:15	Clipping Algorithm, Rasterization (Line, Polygon)
5	11/10/금 11:00~12:15	11/13/월 20:00~21:15	Illumination Models, 중간고사 기출 풀이
6	11/24/금 11:00~12:15	11/27/월 20:00~21:15	Ray Tracing
7	12/1/금 11:00~12:15	12/4/월 20:00~21:15	Texture Mapping, Visible Surface Determination
8	12/8/금 11:00~12:15	12/11/월 20:00~21:15	3D Object Representation, 기말고사 기출 풀이

Computer Graphics
VS Computer Vision

Computer Graphics VS Computer Vision

Computer Graphics

3D object 들을 여러 transformation 을 거치고

rasterization을 거쳐

2D 화면으로 rendering 하는 것

Computer Vision

주로 CG와 반대로

2D input (이미지)를 가지고 여러 작업을 거쳐 feature 들을 뽑아내는 것

cf) 현실세계는 3D. 그것을 카메라로 찍으면 2D. 1D가 축소되며 정보 손실 발생.

ex) edge detect, corner detect, feature matching …

Computer Graphics 수업

- **수업 :** 수학적인 것에 집중
  (중간 범위)
  3D scene이 무엇으로 구성 되어있고
  그 각각을 수학적으로 어떻게 표현해서 (**coordinate system 좌표계**)
  어떤 연산을 통해 (**transformation**)
  2D image로 나타내고 (**rendering**)
  그것을 어떻게 pixel로 바꿔서 display에 나타내는가 (**rasterization**)

  (기말범위)
  그리고 rasterization된 pixel의 색상은 어떻게 결정해야
  빠른 속도로, 현실감 있게 나타낼 수 있을까 (**illumination model** & **rendering model**)

- **과제**
  수업에서 배운 내용을 실제로 어떻게 코딩할까.
  Open GL 활용.

Basic Concept

이 부분은 앞으로 배울 내용의 전체 개요 정도로 보면 된다!
지금 당장 다 이해할 필요 없음!

Basic Concept

1. Rendering이란?
2. Graphics Rendering Pipeline
3. Basic Rendering Algorithms
4. Display

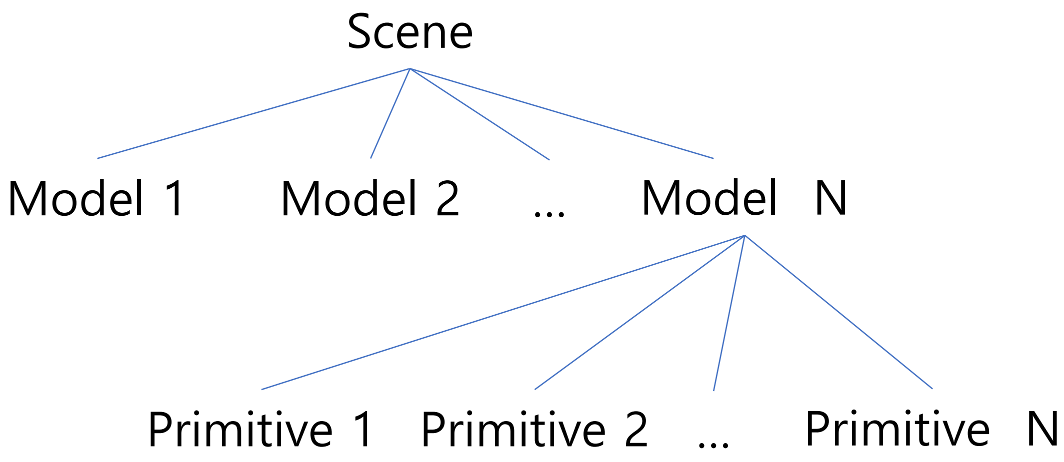
1. Rendering이란?

- **CG의 3대 분야**
  Modeling : 수학 (기하) 기반
  Rendering : 물리학 (광학) 기반
  Animation

- **CG의 3대 분야인 modeling, rendering, animation 중에 rendering에 집중해보자.**

1. Rendering이란?

- **Scene, Model, Primitive**



1. Rendering이란?

- **Geometric Primitive**
  - **GPU 로 바로 render 가능한 primitive**
    - 점
    - 선
    - 면

  - **GPU 로 바로 render 불가능한 primitive**. 위의 것들로 근사해서 render.
    - 곡선, 곡면
    - Voxel

Point = 점 = 1D (크기 없)
Pixel = picture element = 2D (크기 있)
Voxel = volume element = 3D (부피 있)

1. Rendering이란?

- **Rendering Pipeline**

  3D scene을 2D image로 바꾸기 위한 일련의 과정

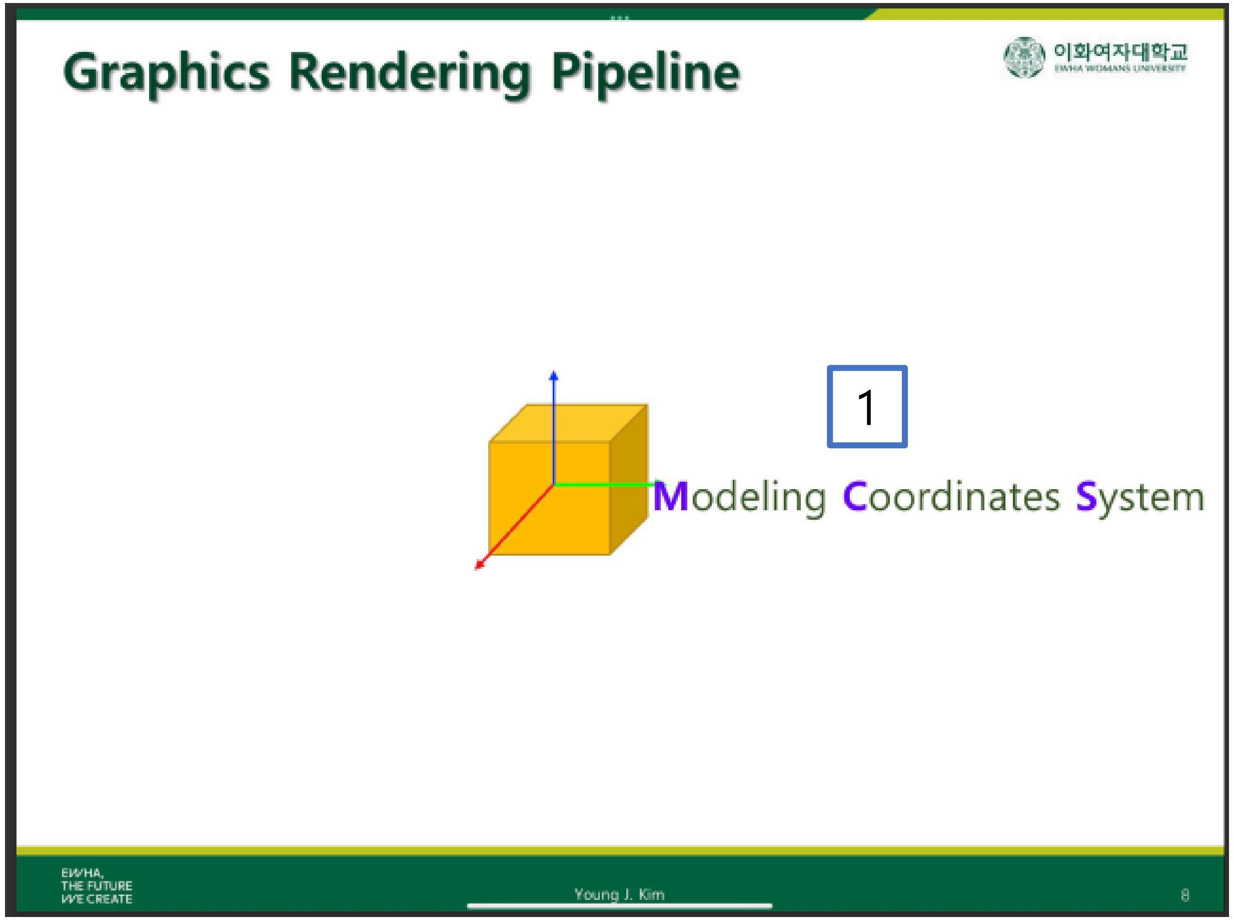
  model → scene → image

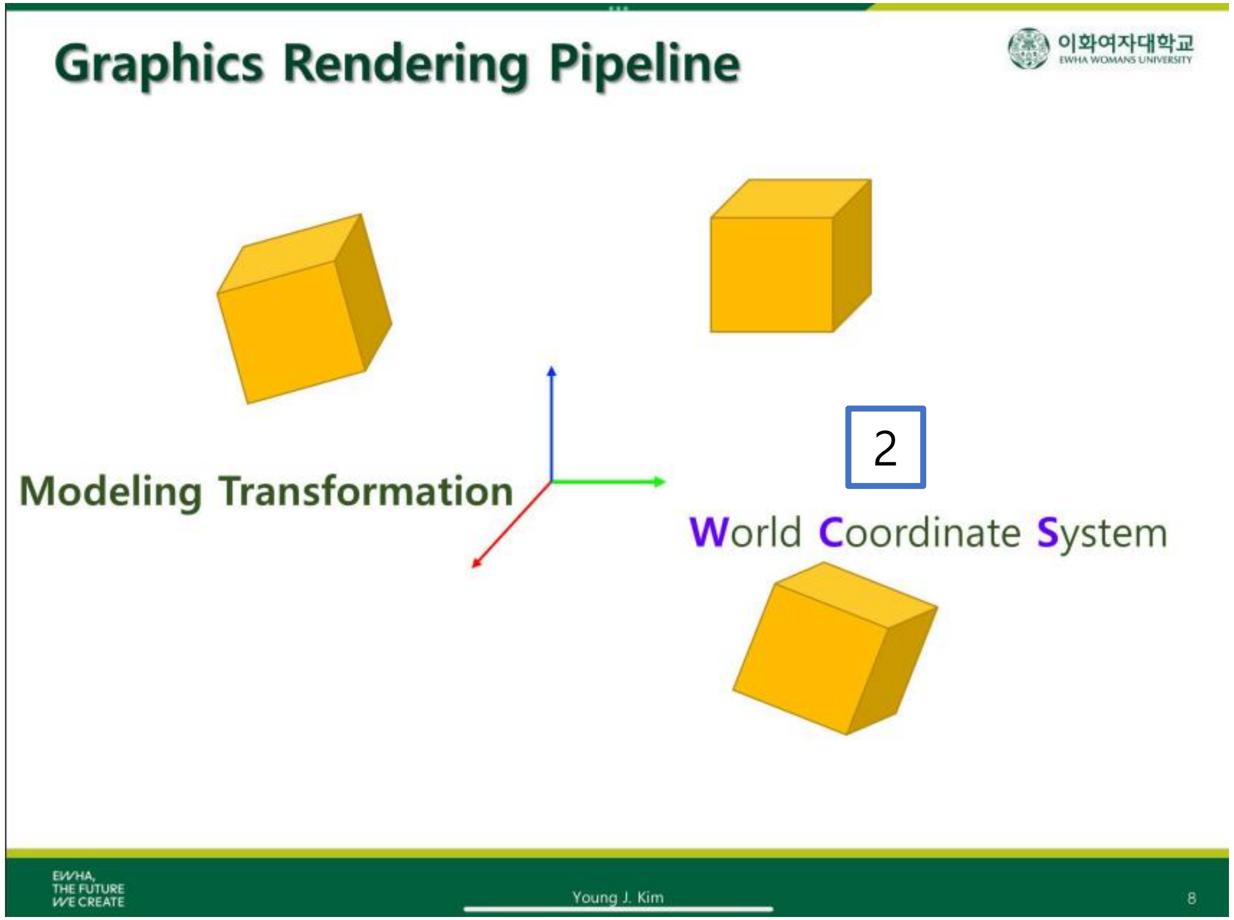
  transformation의 반복 + 마지막에 shading

  이 pipeline 자체가 GPU에 구현 되어있음
  이 pipeline 자체가 어떻게 작동하는지 몰라도 OpenGL 로 작동 가능함.
  그러나 우리는 작동 방법도 배움.

2. Graphics Rendering Pipeline

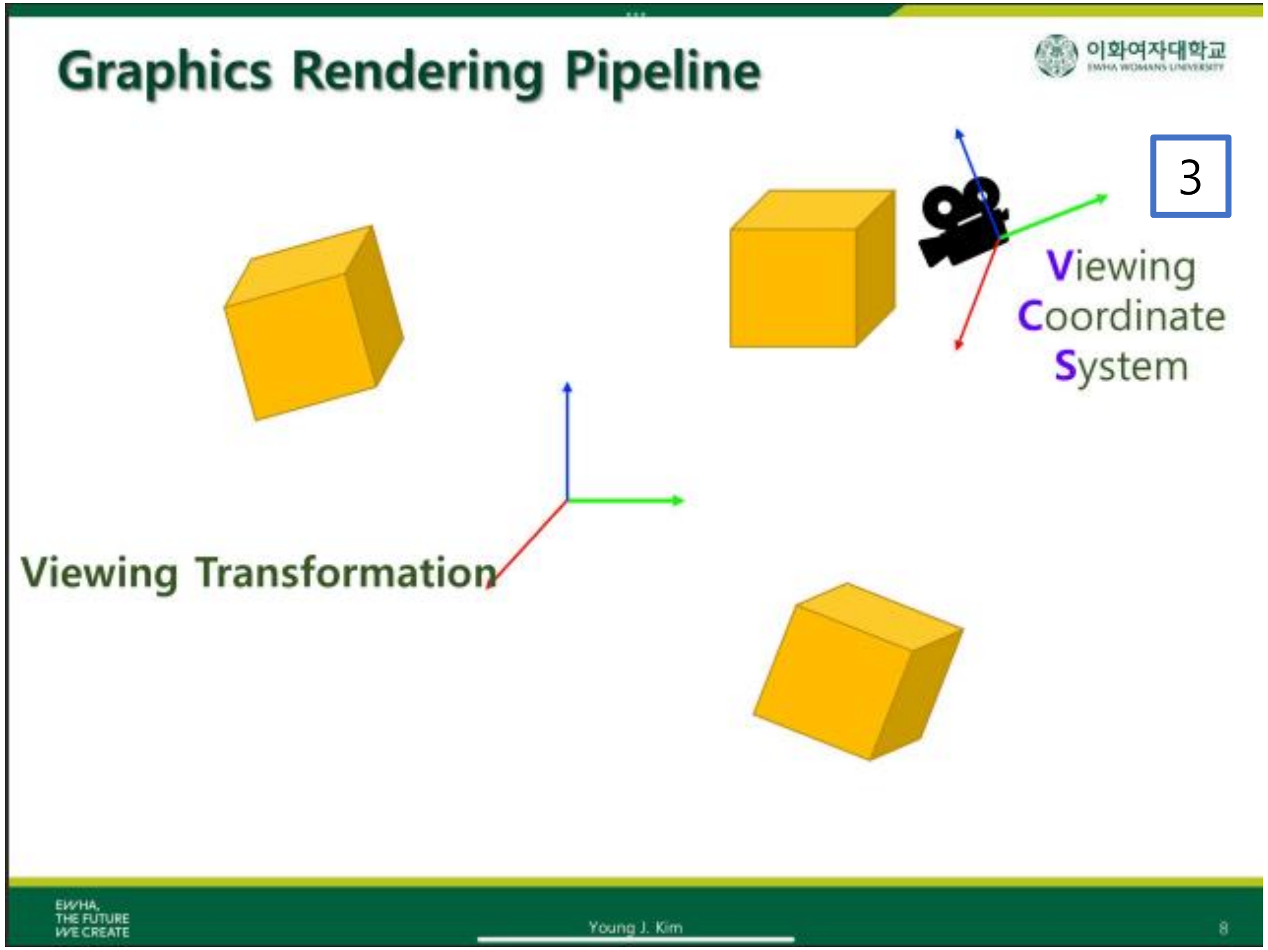
- 우선 흐름만 이해. 요약 슬라이드 뒤에 있음.
- 그래서 transformation 을 어떻게 하는지 자세한 방법은 나중에 배움.





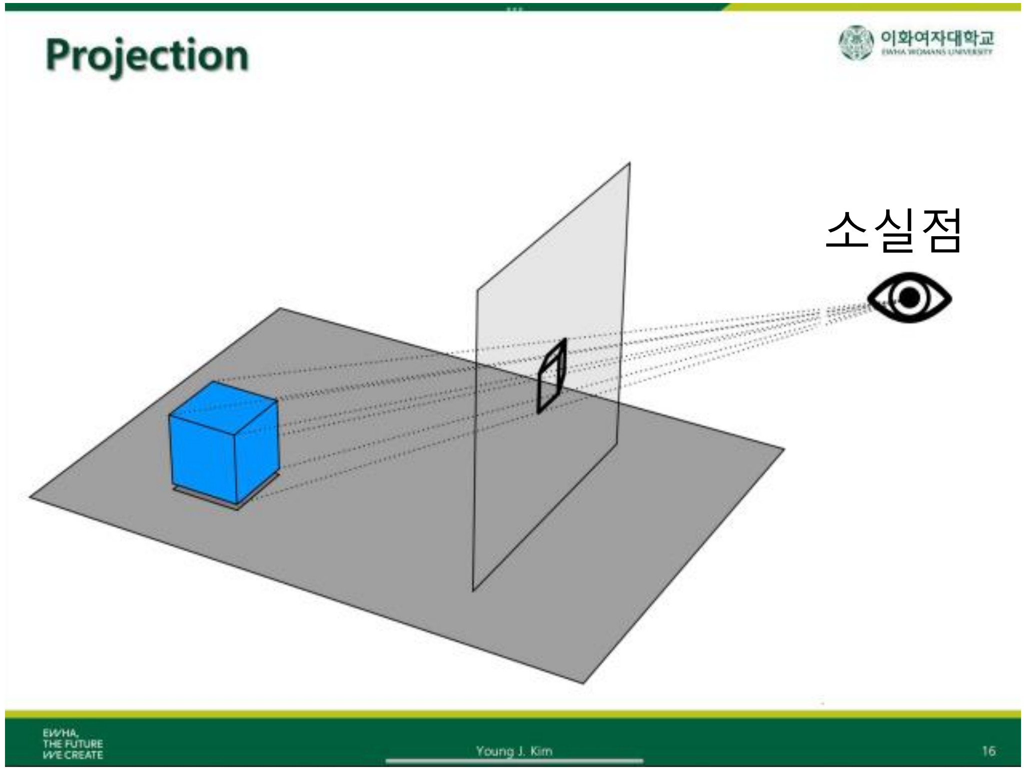
비로소 scene이 결정됐다.
모든 model이 하나의 wcs 에 존재.

2. Graphics Rendering Pipeline



카메라 위치 결정.
카메라 위치 따라 model 좌표 변환됨.

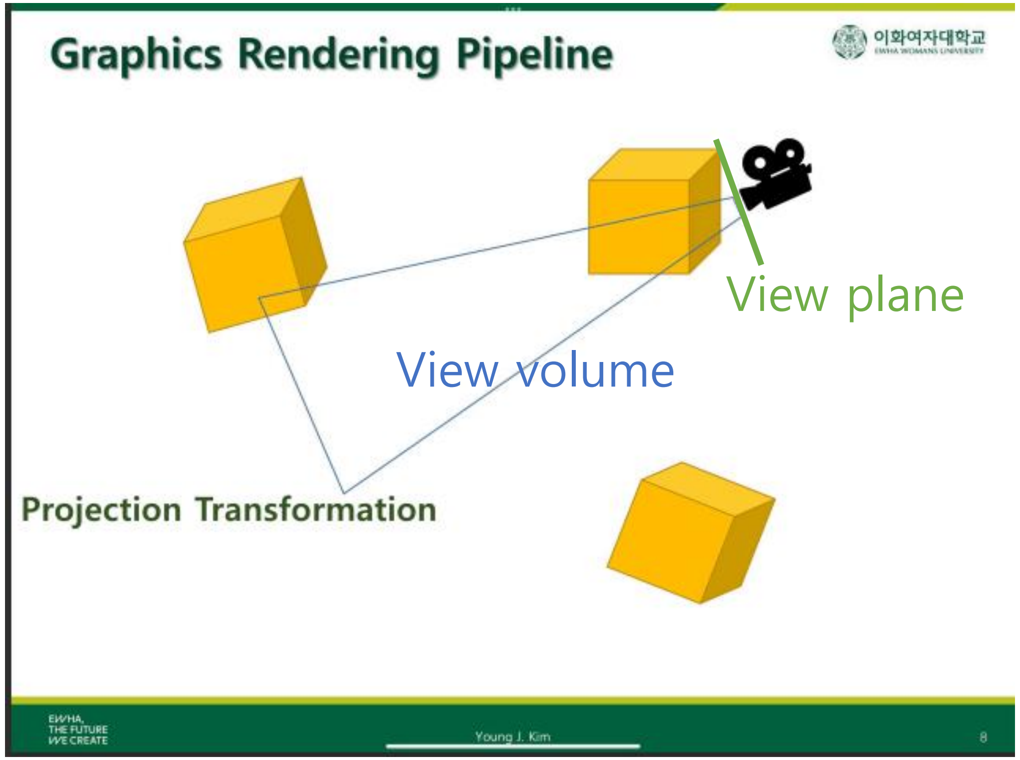
2. Graphics Rendering Pipeline

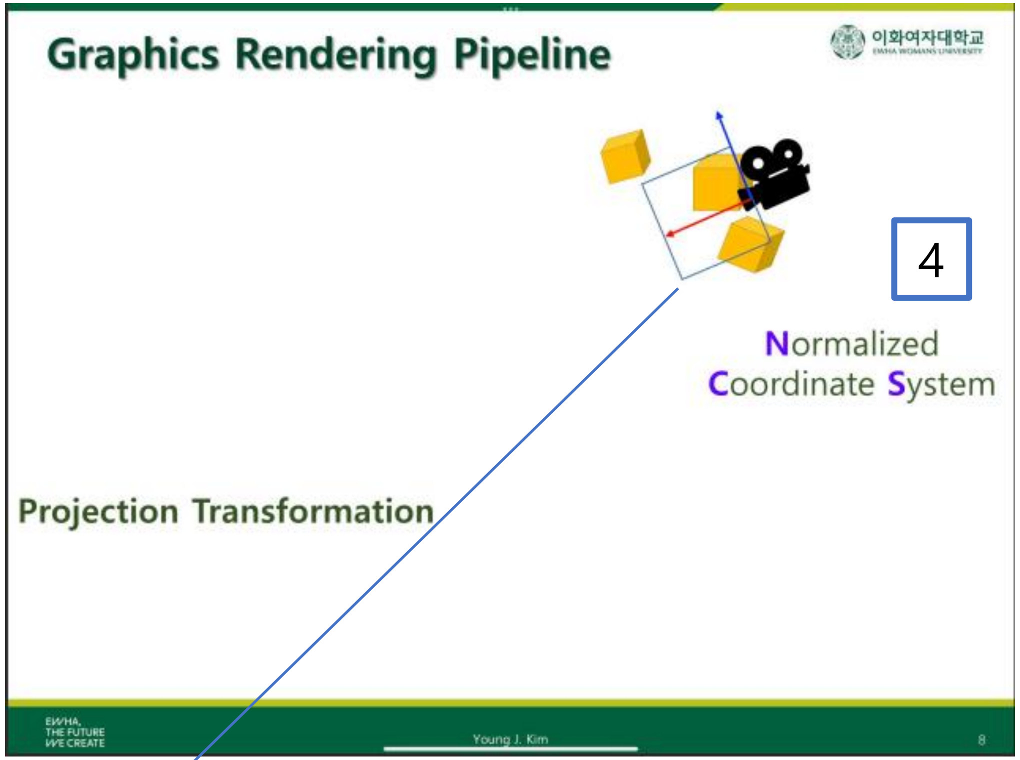


cf) Perspective projection

: 원근감 있는 projection
: 가까운건 크게, 멀리 있는건 작게
: 다른 종류의 projection 도 있음 (나중에 배움)

2. Graphics Rendering Pipeline

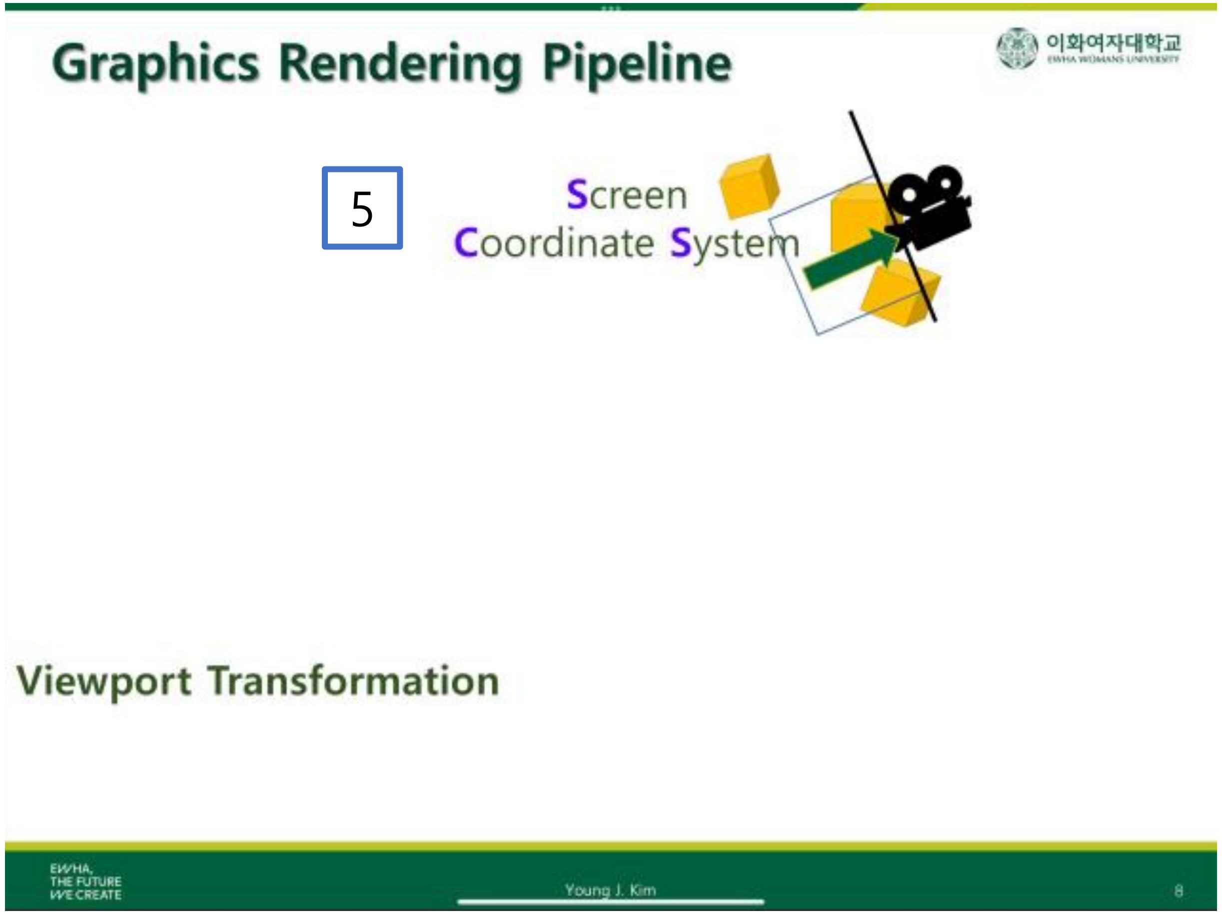




2차원으로 만들기 위한 준비.
3차원의 물체를 카메라 앞에 놓여있는 view plane 쪽으로 projection 하기 위한 transformation.

Viewing volume (VV)가 2x2x2 정육면체가 된다.
그래서 "normalized" (정규화) 된 좌표계라고 부름.

2. Graphics Rendering Pipeline



1) x, y, z 중에 z를 날려서
3D를 2D로 만든다.

2) display screen에 맞게 scaling.
(각 display 마다 해상도가 다르다)

2. Graphics Rendering Pipeline

- 해당 coordinate system 에 맞게 각 transformation으로 좌표체계 바꿔주는 것
- 각 transformation 은 4x4 행렬 (나중에 배움)
- 아래 표가 rendering pipeline 전부는 아님. Shading 등의 작업 남음.

Coordinate System	Transformation	한마디 정리	세부 설명
modeling coordinate system			각 모델마다 다름. 여러 개 존재.
world coordinate system	modeling transformation	모델 위치 결정	하나의 scene안에 있는 모든 모델이 하나의 wcs에 위치함.
viewing coordinate system	viewing transformation	카메라 위치 결정	카메라 위치가 vcs의 원점 & xyz축 결정
normalized coordinate system	projection transformation	3D → 2D 준비	1) Viewing volume 결정 2) Projection type 결정
screen coordinate system	viewport transformation	3D → 2D	1) Z 좌표값 날리기 2) Display screen 사이즈에 맞게 scaling

2. Graphics Rendering Pi

1) **viewing volume 결정**
clipping = vv 밖 날린다
vv가 2x2x2 정육면체가 된다

2) **projection type 결정**
z좌표 날리기 전 x,y 값 조정을 어떻게 할지 결정하는 것
(a) parallel → orthographic, oblique
(b) perspective

- 해당 coordinate system 에 맞게 각 transformation으로 좌표체계 바
- 각 transformation 은 4x4 행렬
- 아래 표가 rendering pipeline 전부는 아님. Shading 등의 작업 남음.

Coordinate System	Transformation	한마디 정리	세부 설명
modeling coordinate system			각 모델마다 다름. 여러 개 존재.
world coordinate system	modeling transformation	모델 위치 결정	하나의 scene안에 있는 모든 모델이 하나의 wcs에 위치함.
viewing coordinate system	viewing transformation	카메라 위치 결정	카메라 위치가 vcs의 원점 & xyz축 결정
normalized coordinate system	projection transformation	3D → 2D 준비	1) Viewing volume 결정 2) Projection type 결정
screen coordinate system	viewport transformation	3D → 2D	1) Z 좌표값 날리기 2) Display screen 사이즈에 맞게 scaling

3. Basic Rendering Algorithms

- 위의 표 말고 rendering pipeline에는 어떤 것들이 더 있는가.
- **(1) Transformation**

  위의 표에서 봤던 행렬 연산들

- **(2) Clipping/ Visible Surface determination**

  Clipping = viewing volume 밖은 잘라버린다.

  Visible Surface determination = vv 안에 있더라도 어떤 면은 내 눈에 안 보일 수도 (앞 물체에 가려서)

- **(3) Rasterization**

  continuous한 object 를 pixel로 바꿔줘야 함. 그 과정을 rasterization 알고리즘을 통해 수행.

- **(4) Shading & Illumination : 각 pixel의 color 결정**

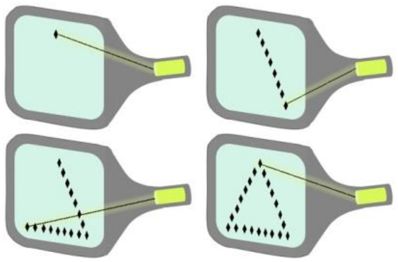
  shading = 음영처리 = 빛을 향하는 부분은 밝고 아닌 부분은 어둡게

  반사, 굴절, color bleeding 등도 고려

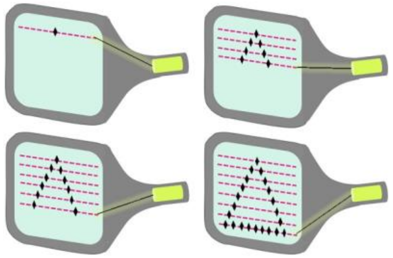
4. Display

- Display 장비에 따라서 display하는 방식이 다르다.

- 시간순으로 display 방식 보면,
- **(1) Vector Display** : 꼭짓점을 따라서 쭉 디스플레이
  Primitive 가 너무 많은데 그걸 다 따라가기가 힘들어서 요새는 사용하지 않음.



- **(2) Raster Display** : rasterization 이 선행되고 디스플레이
  - **Rasterization** : vector description(연속적인 것) → set of pixels
  - Rasterization된 pixel이 저장 되어있는 것을 frame buffer라 함.



4. Display

- **Frame buffer**
  : 2D array of pixels
  : graphics H/W 인 GPU 내부에 존재
  : 각 행(row)는 "**scan line**", "**raster line**"이라 부름

  : 한 픽셀은 R, G, B로 구성된다.

  : (예전) R, G, B 값 각각 **8 bit**
  각각 2^8 개 = 256개 = 0~255
  0~255 사이 integer
  R 2^8 개, G 2^8 개, B 2^8 개 => 총 $2^8 \times 2^8 \times 2^8 = 2^{24}$ 개의 색 표현

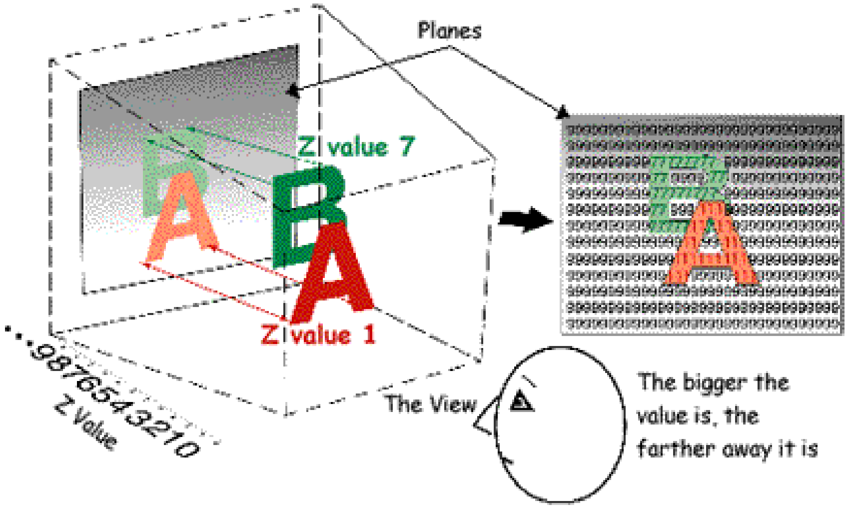
  : (최근) R, G, B 값 각각 **32bit (4byte)**
  색상을 0~255 사이 integer 로 표현하지 않고 **0.0~1.0 사이 floating point** 로 표현
  floating point 가 4byte.

4. Display

- **Alpha Channel, A, 투명도**
  - 0 은 투명, 1은 불투명
  - A 도 32bit, 0~1 사이 floating point
  - $Out_{RGB} = Current_{RGB} * Current_{A} + Existing_{RGB} * (1 - Current_{A})$

새로 그리려는 것의 RGB 이미 그려둔 것의 RGB

- **Z buffer, Depth duffer, D or Z**
  - Depth 도 32bit, 0~1 사이 floating point
  - 눈으로부터 거리가 더 가까운 곳이 더 작은 수
  - Depth 가 작은 것만 render 한다



4. Display

- **Double buffering**
  - 애니메이션을 flicking 없게 (그리는 과정은 눈에 안보이게) 하는 것
  - Buffer 두개 이용 (front buffer, back buffer)
  - front buffer 만 보여준다.
  - back buffer 에 그리고 swapping.

정리하자면,

rasterization 을 마친 후

각 픽셀마다 **[RGB A D]**를 **frame buffer** 에 저장.

그것을 raster line 따라 display 하는 방법을 **raster display** 라 한다.

똑같은 frame buffer 두개를 이용해 (front buffer, back buffer)

그리는 과정은 눈에 안보이게 하는 것을 **double buffering** 이라 한다.

QUIZ - Basic Concept

문제 1
정답
총 1.00 점에서 1.00 점 할당
문제 표시

Order the transformation steps correctly:

하나를 선택하세요.
- a. Modeling -> Viewing -> Viewport -> Projection
- b. Modeling -> Viewing -> Projection -> Viewport ✔
- c. Viewing -> Modeling -> Viewport -> Projection
- d. Viewing -> Viewport -> Modeling -> Projection

정답 : Modeling -> Viewing -> Projection -> Viewport

문제 2
정답
총 1.00 점에서 1.00 점 할당
문제 표시

What do you call the process of "converting a projected screen primitive to a set of pixels"?

하나를 선택하세요.
- a. Shading
- b. Clipping
- c. Rasterization ✔
- d. Transformation

정답 : Rasterization

문제 3
정답
총 1.00 점에서 1.00 점 할당
문제 표시

Find which of the following primitives can not be directly rendered by GPU.

하나를 선택하세요.
- a. Voxels ✔
- b. Line segments
- c. Points
- d. Polygons

정답 : Voxels

2D Geometric Transformations

Basic 2D Transformations

- Translation
- Rotation
- Scaling

Homogeneous Coordinates

Matrix Representations

-Inverse

-Composite

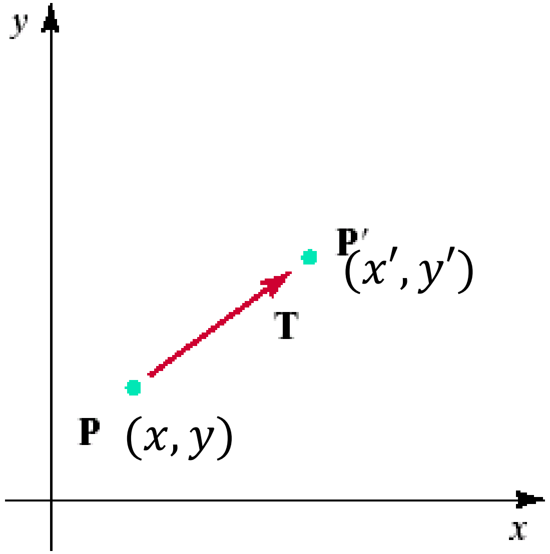
-Shear

-Reflection

Basic Two-Dimensional Transformations

Translation

- 평행이동
- 점 (x,y)를 x축으로 tx, y축으로 ty만큼 평행이동 시켰을 때의 점이 (x′,y′)



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

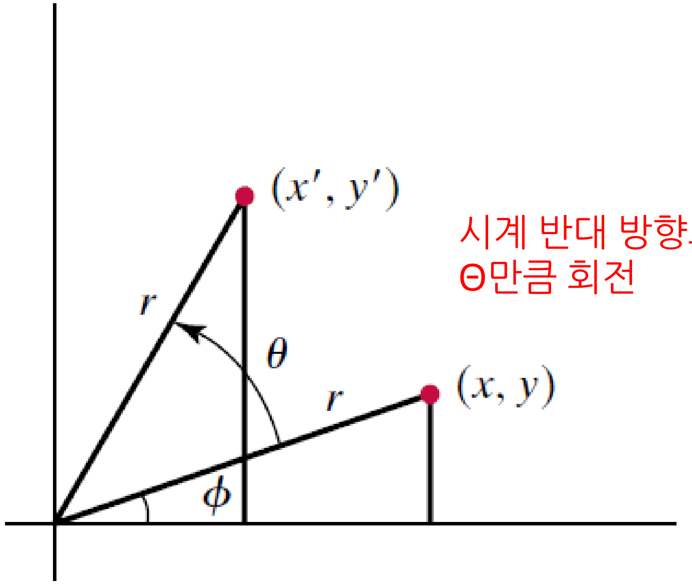
$$\mathbf{P} = \begin{bmatrix} x \\ y \end{bmatrix}, \qquad \mathbf{P}' = \begin{bmatrix} x' \\ y' \end{bmatrix}, \qquad \boxed{\mathbf{T} = \begin{bmatrix} t_x \\ t_y \end{bmatrix}}$$

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

Basic Two-Dimensional Transformations

Rotation

- 2차원 평면에서 회전
- 양의 회전 방향 = 시계 반대 방향
- 점 (x,y)를 원점 중심으로 Θ만큼 회전 이동했을 때의 점이 (x′,y′)



시계 반대 방향으로
Θ만큼 회전

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

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

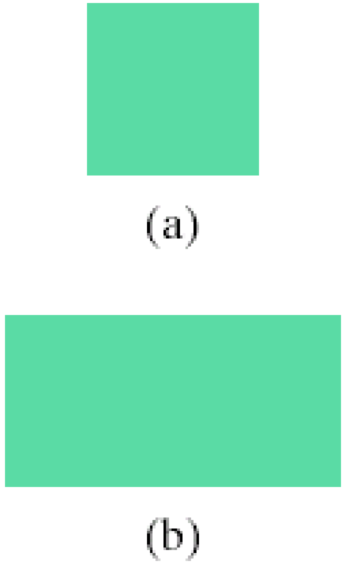
$$\boxed{\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}}$$

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

Basic Two-Dimensional Transformations

Scaling

- x축, y축 방향으로 늘리거나 줄이는 것
– s_x : x축으로 얼마나 scaling할지 나타내는 수
– s_y : y축으로 얼마나 scaling할지 나타내는 수
– s_x , s_y가 1보다 크면 scale up, 1보다 작으면 scale down



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

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

=S

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

Translation $\xleftrightarrow{\text{구조 다름}}$ **Rotation** $\xleftrightarrow{\text{구조 비슷}}$ **Scaling**

$$\mathbf{T} = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

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

$$\mathbf{S} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$$

2x1 행렬

2x2 행렬

2x2 행렬

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

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

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

- translation만 행렬 차원이 2x1로 다름
- translation만 연산이 x가 아닌 +연산

➡ consistent하지 못하면 여러 문제가 발생할 수 있음

Uniform한 형태로 어떻게 통일할 수 있는가?

=> Homogenous Coordinate 사용

Homogeneous Coordinates

P' = MP

로 모양 통일

Homogeneous Coordinates
- h를 추가하여 1차원을 높임 (3차원으로)
- Map $(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}^2$ to $(\boldsymbol{x_h}, \boldsymbol{y_h}, \boldsymbol{h}) \in \mathbb{RP}^3$, where $\boldsymbol{x} = \frac{\boldsymbol{x_h}}{\boldsymbol{h}}, \boldsymbol{y} = \frac{\boldsymbol{y_h}}{\boldsymbol{h}}$
- 일반적으로 $\boldsymbol{h} = \mathbf{1}$ (i.e. $\boldsymbol{x} = \boldsymbol{x_h}, \boldsymbol{y} = \boldsymbol{y_h}$)
  => (x,y) -> (x,y,1) 이므로 변환이 쉽기 때문

Matrix Representations

Homogeneous coordinate로 변경해서 사용

- Translation $T(t_x, t_y)$

결과에서 첫번째, 두
번째 값 가져가면 됨

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

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

- Rotation $R(\theta)$

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

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

- Scaling $S(s_x, s_y)$

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

$$x' = xs_x$$
$$y' = ys_y$$

Inverse Transformations

- 역행렬

Translation

$$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

tx->-tx, ty->-ty

$$\mathbf{T}^{-1} = \begin{bmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

θ->-θ

$$\mathbf{R}^{-1} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Scaling

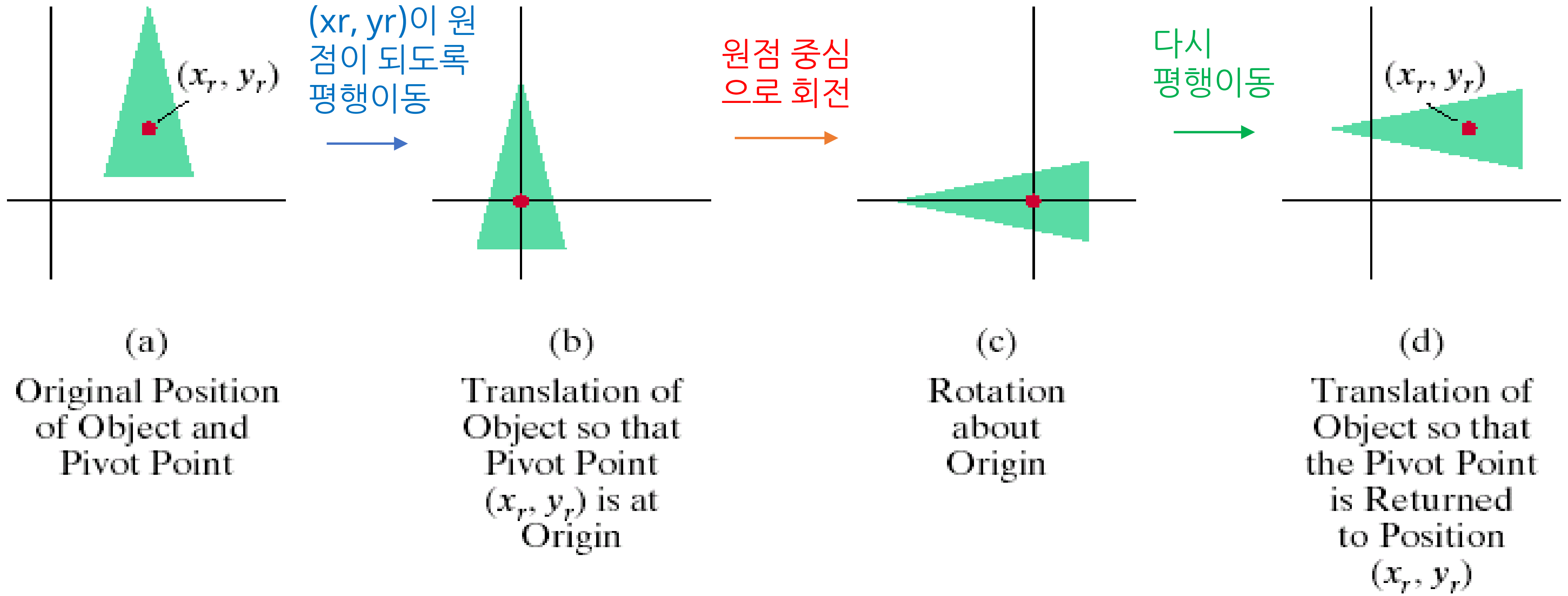
$$\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Sx->1/sx, sy→1/sy

$$\mathbf{S}^{-1} = \begin{bmatrix} \frac{1}{s_x} & 0 & 0 \\ 0 & \frac{1}{s_y} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Composite Transformations

- 임의의 2D point (x_r, y_r)에서 회전



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

진행순서 연산순서 반대

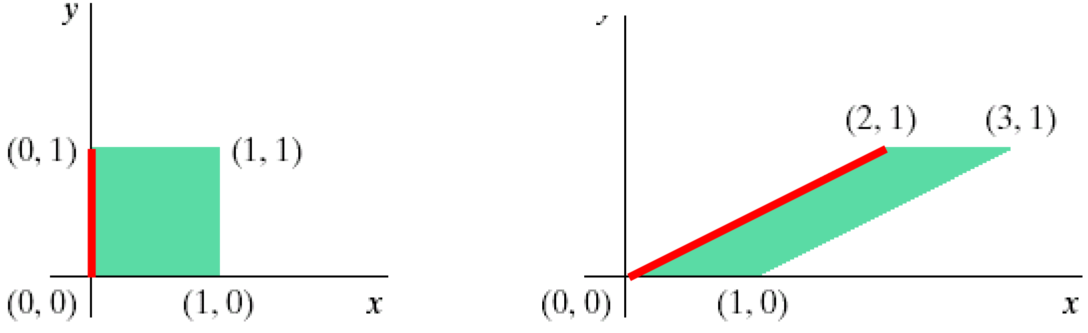
Shear (Other Transformations)

- 모양을 왜곡 (x나 y방향)
- sh_x, sh_y : 왜곡 양 결정

shear along *x*

$$\begin{bmatrix} 1 & sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{matrix} x \\ y \\ 1 \end{matrix} = \begin{matrix} x' \\ y' \\ 1 \end{matrix}$$

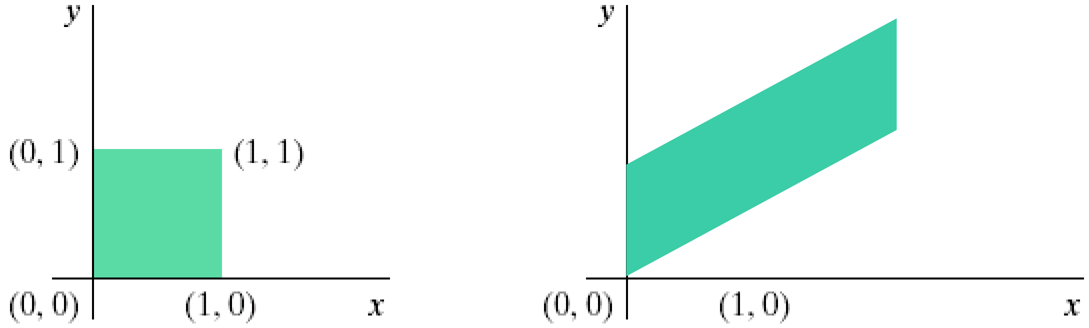
$$x' = x + sh_x \cdot y, \qquad y' = y$$



(a) (b)

shear along *y*

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

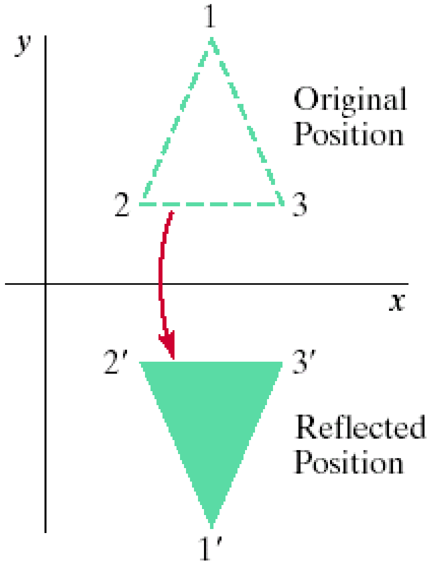


(a) (b)

Reflection (Other Transformations)

- 대칭이동

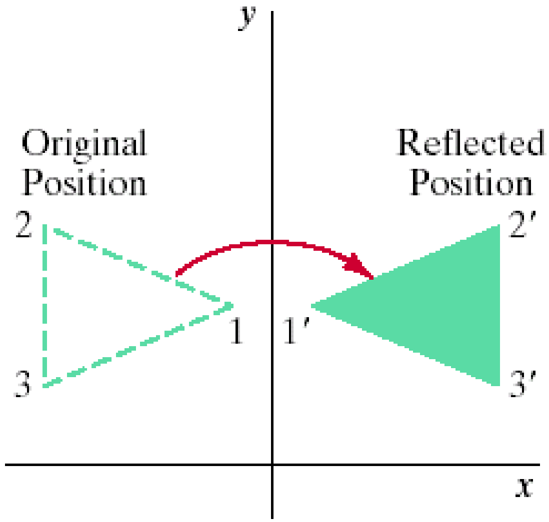
X축



$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{matrix} x \\ y \\ 1 \end{matrix} = \begin{matrix} x' \\ y' \\ 1 \end{matrix}$$

x'=x y'=-y

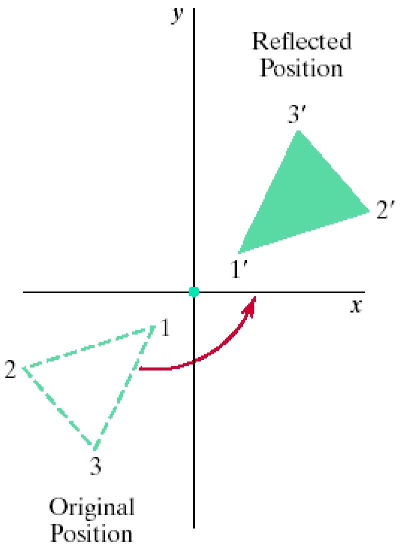
Y축



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

x'=-x y'=y

원점



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

x'=-x y'=-y

Quiz – 2D Transformations

문제 1

Which of the following buffers is used for visual surface determination in 3D?

하나를 선택하세요.

- a. Double buffer
- b. Alpha channel
- c. Color buffer
- d. Z-buffer ✔

Quiz – 2D Transformations

문제 2

Rotate (1,0) by 120 degrees around the origin.

하나를 선택하세요.

a. $(-\frac{1}{2}, \frac{\sqrt{3}}{2})$

b. $(\frac{1}{2}, -\frac{\sqrt{3}}{2})$

c. $(\frac{1}{2}, \frac{\sqrt{3}}{2})$

d. $(-\frac{1}{2}, -\frac{\sqrt{3}}{2})$

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

$x' = 1 \cdot \cos 120^\circ - 0 \cdot \sin 120^\circ$

$= -\frac{1}{2}$

$y' = 1 \cdot \sin 120^\circ - 0 \cdot \cos 120^\circ$

$= \frac{\sqrt{3}}{2}$

$\therefore (-\frac{1}{2}, \frac{\sqrt{3}}{2})$

Quiz – 2D Transformations

문제 3

Which of the following homogeneous coordinates in 3D projective space corresponds to $(1, 2)$ in 2D Euclidean coordinate?

하나를 선택하세요.

- a. (2, 4, 1)
- b. (2, 4, 0)
- c. (2, 4, 2)
- d. $(2, 4, \frac{1}{2})$

Homogeneous Coordinates

- Map $(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}^2$ to $(\boldsymbol{x_h}, \boldsymbol{y_h}, \boldsymbol{h}) \in \mathbb{RP}^3$,

where $\boldsymbol{x} = \frac{\boldsymbol{x_h}}{\boldsymbol{h}}, \boldsymbol{y} = \frac{\boldsymbol{y_h}}{\boldsymbol{h}}$

x=2/2=1
y=4/2=2
-> (1,2)