Viewing Transformation

COLLEGE OF COMPUTING
HANYANG ERICA CAMPUS
Q YOUN HONG (홍규연)

3D물체를 어떻게 2D 화면에 그리는가?

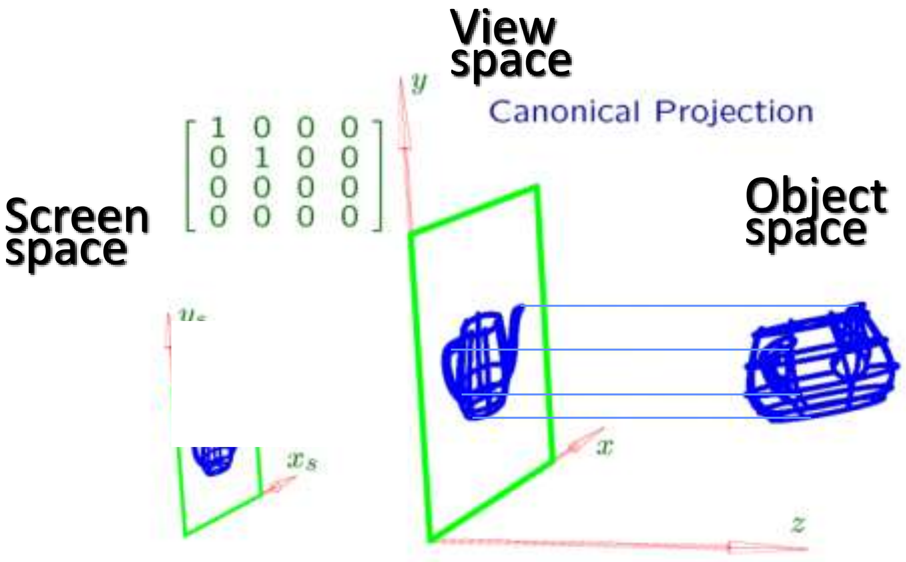
“거울 앞 소녀”, 파블로 피카소, 1932

어안렌즈 이미지

3D물체를 어떻게 2D 화면에 그리는가?

- 3D 물체는 image plane에 투영(projection)됨

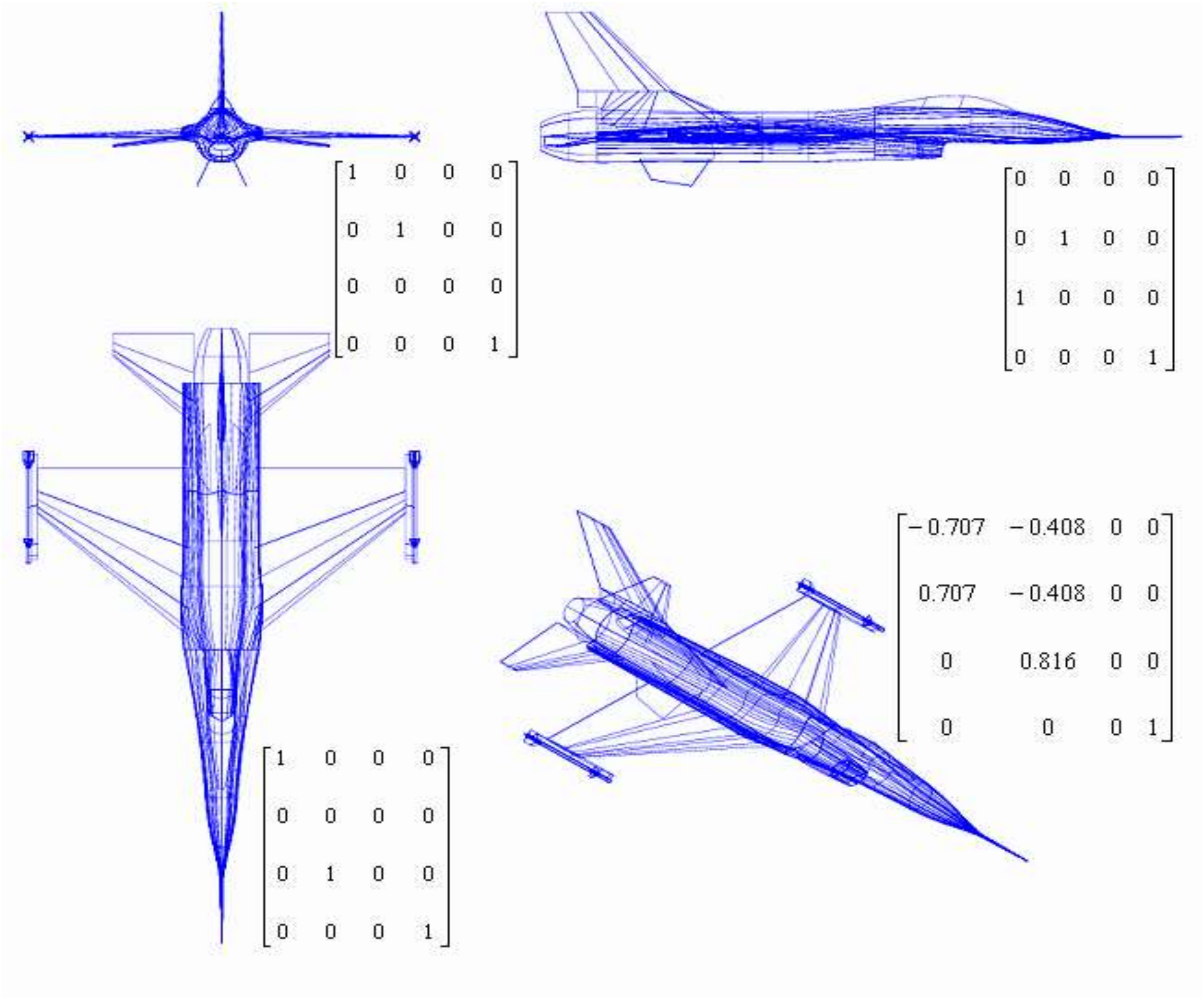
  (직선이 직선으로 투영됨)



투영의 한 예시: 3D 물체를 xy-plane에 투영
(이때, z값은 무시됨)

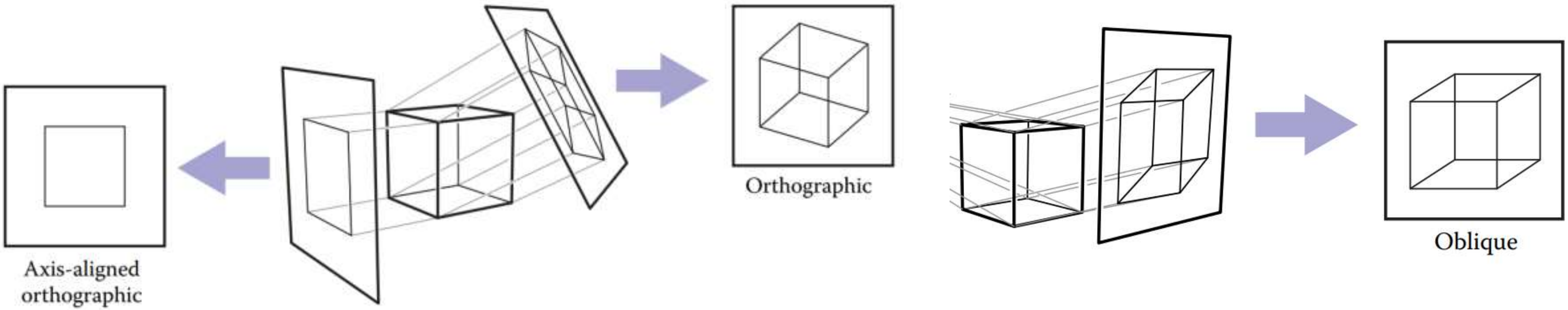
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

투영(Projection)의 예시



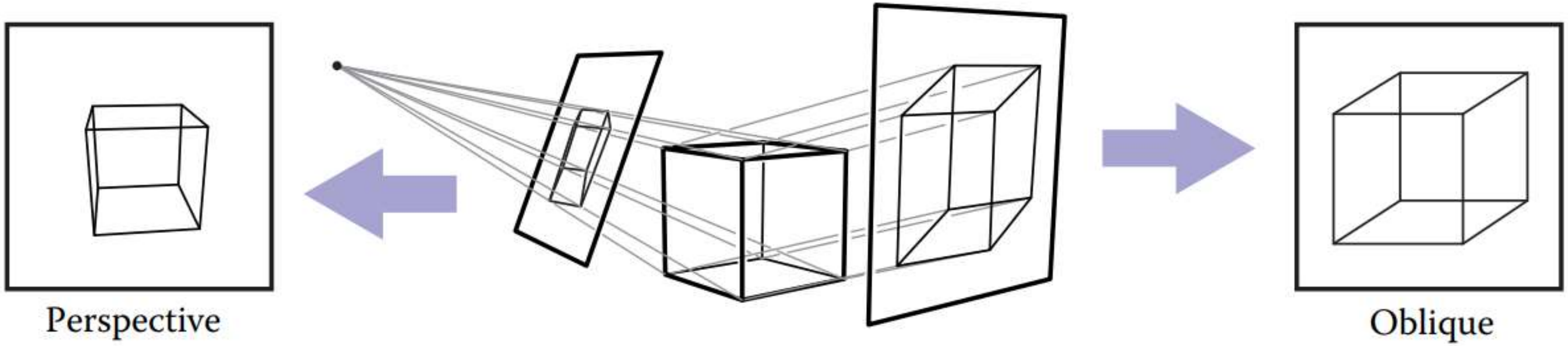
Projection의 종류

- Parallel projection (평행 투영)
  - 투영 직선 (projection line)들이 모두 평행임
  - Orthographic projection (정사 투영): 투영 직선들이 이미지에 수직
  - Oblique projection (경사 투영): 투영 직선들이 이미지에 비스듬히 만남



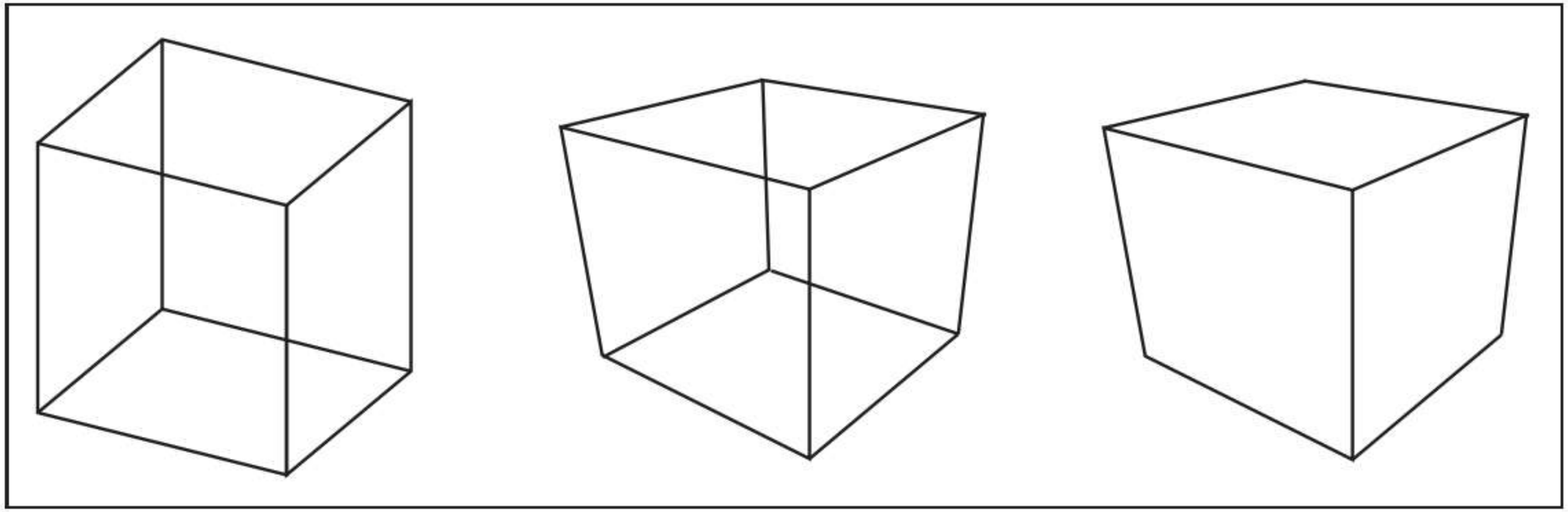
Projection의 종류

- Perspective projection (원근 투영)
  - 투영 직선이 한점(viewpoint)를 지남
  - 3D 물체에서 viewpoint를 잇는 투영 직선들과 image plane의 교점들로 2D 이미지 생성됨



Projection의 종류

정사 투영(Orthographic)과 원근 투영(Perspective)의 비교

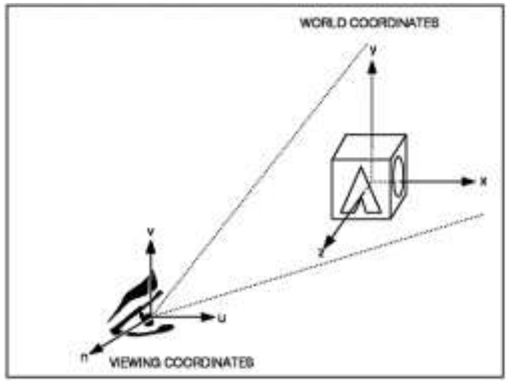


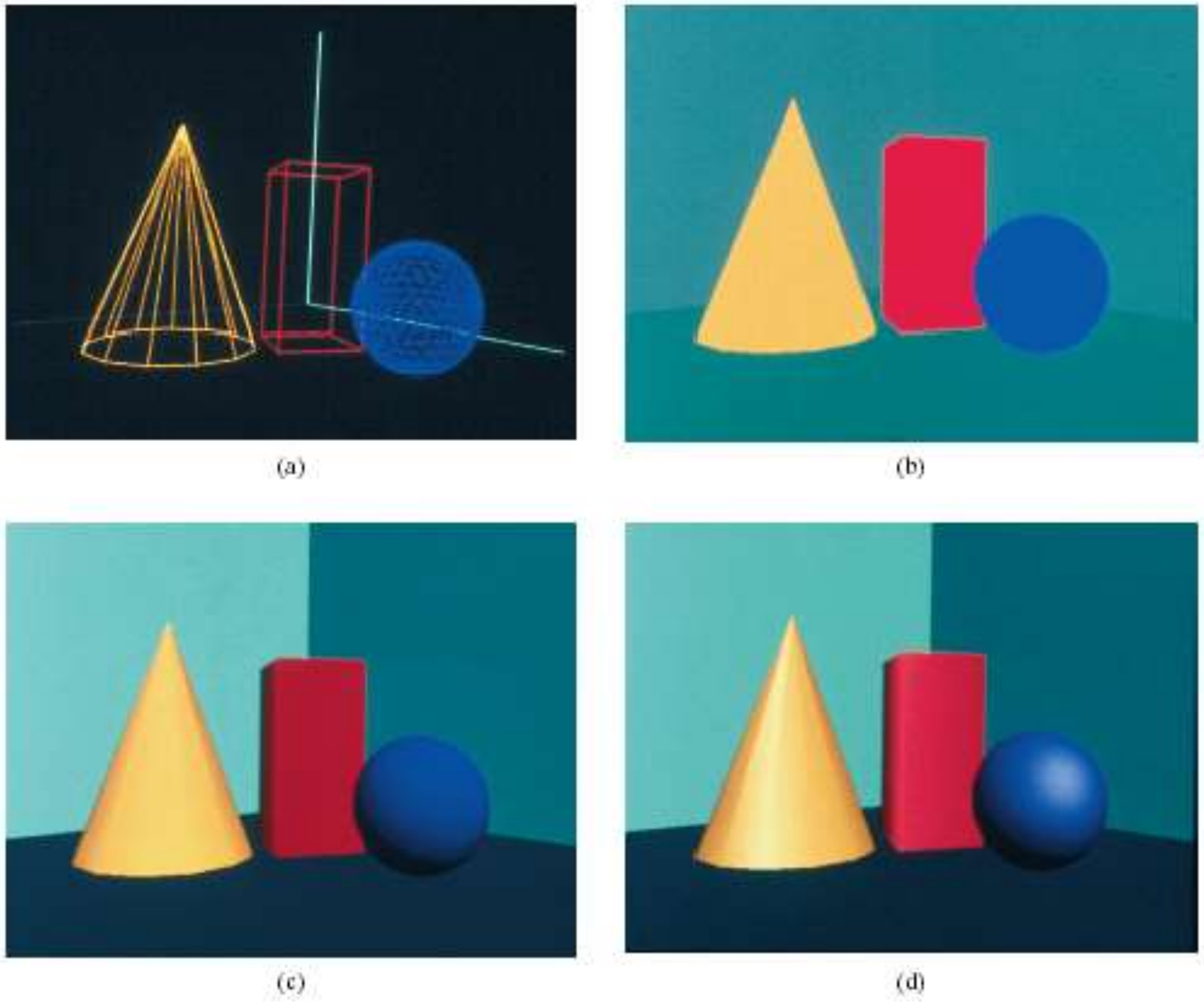
Orthographic projection

Perspective projection

Perspective projection
(with hidden edges)

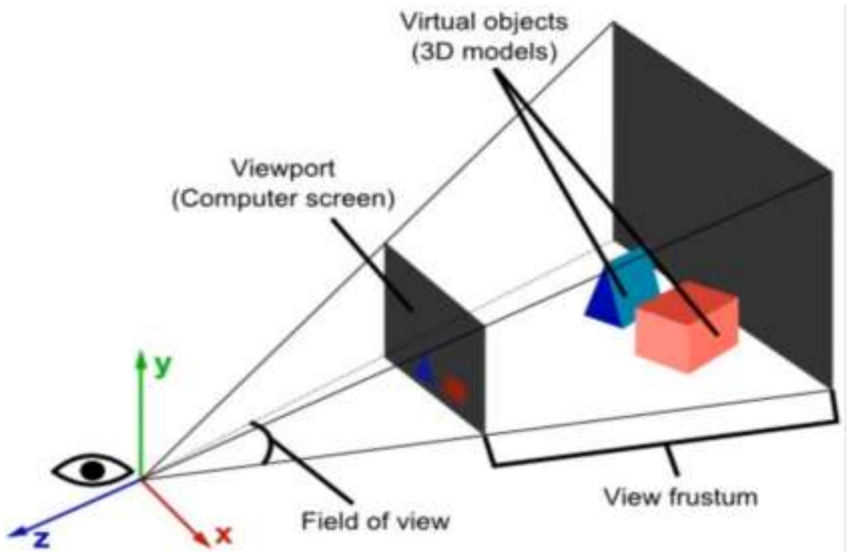
뷰잉 변환(Viewing Transformation)

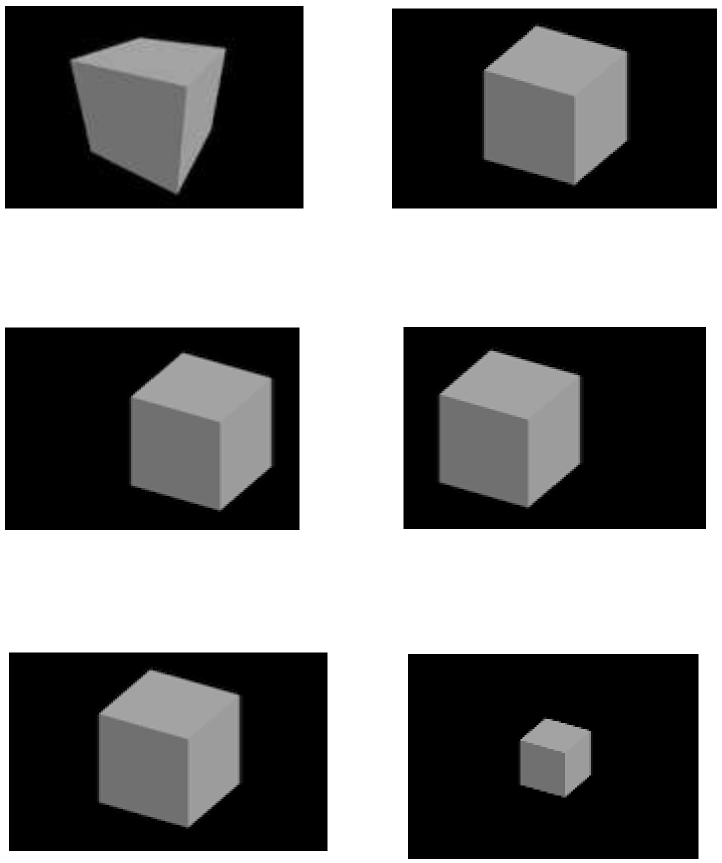




뷰잉 변환(Viewing Transformation)

- 3D Scene의 뷰잉 변환은 사진을 찍는 과정과 비슷함
  - 카메라 위치
  - 카메라 방향(orientation)
  - Viewing window (zooming)





그래픽스에서의 뷰잉 변환

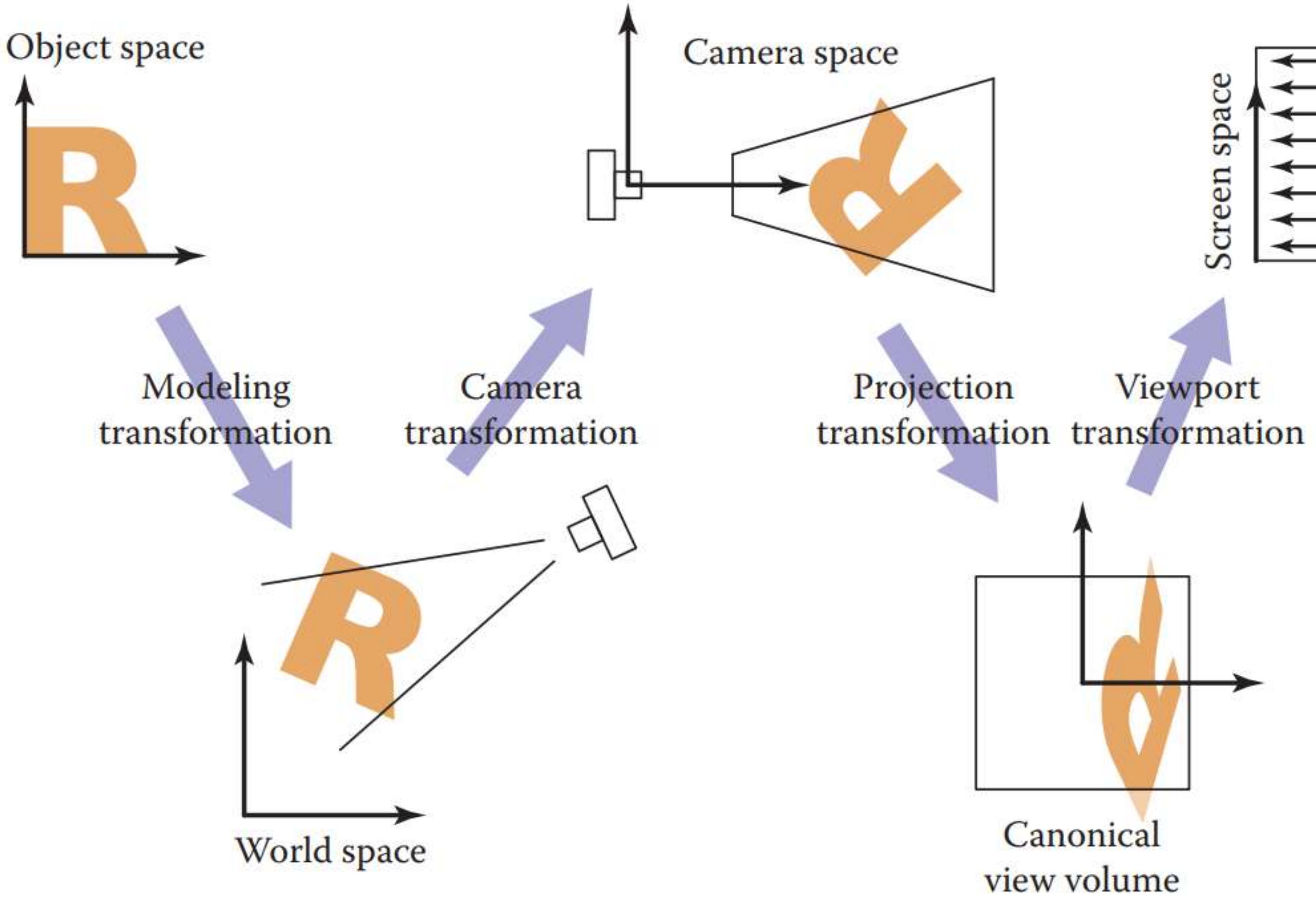
- 정준좌표계(canonical coordinate system)에서 (x,y,z) 로 표현하는 3D 점들의 이미지 위에서의 위치를 pixel 단위로 계산
- 아래의 변환들의 결합

① Camera transformation(eye transformation)
- 카메라를 원점에 위치시키는 강체 변환(rigid body transformation)
- 카메라의 위치와 방향에 따라 달라짐

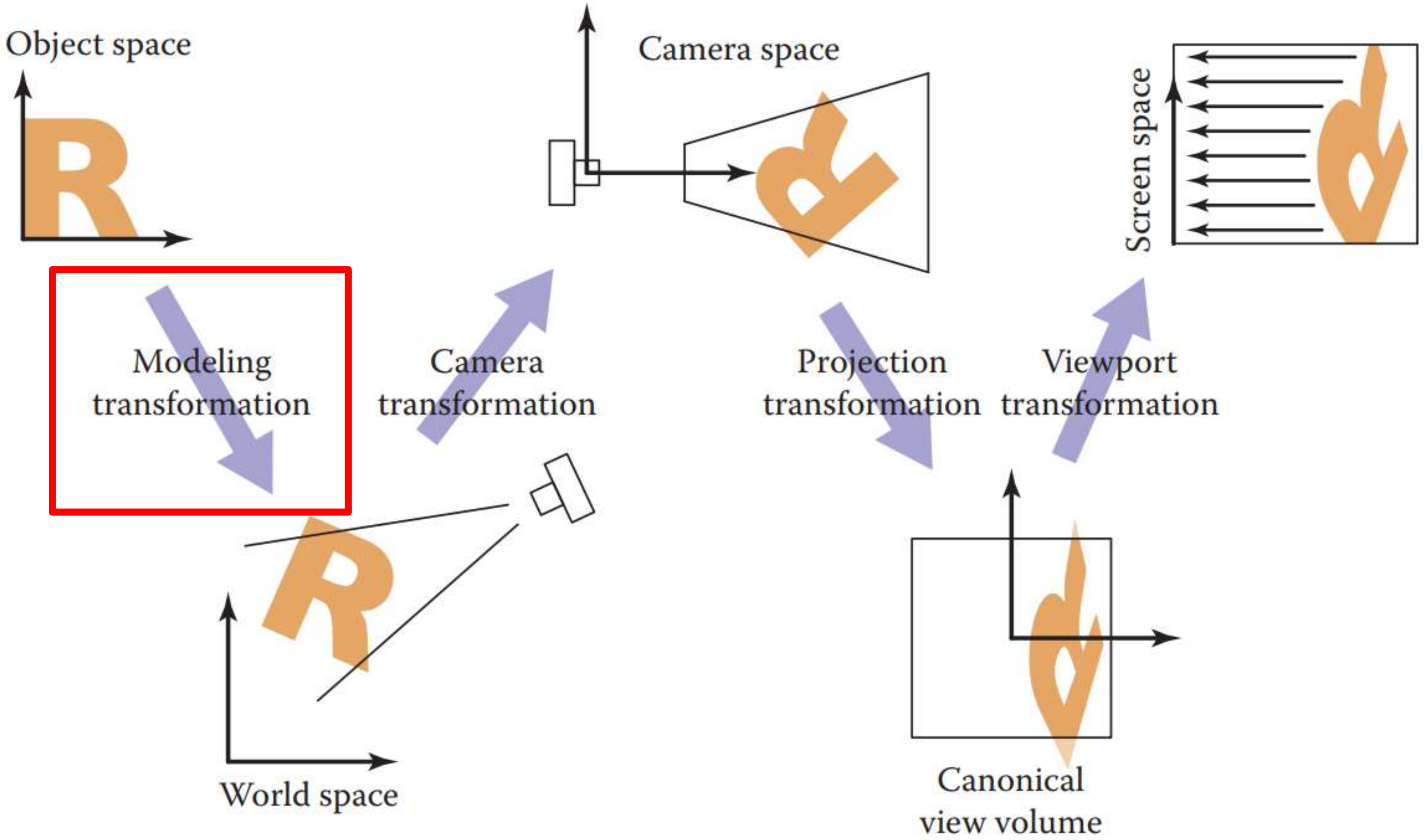
② Projection transformation
- 카메라 좌표계의 모든 점들의 x,y를 [-1,1]x[-1,1]에 위치시킴
- 투영의 종류에 따라 달라짐

③ Viewport transformation(windowing transformation)
- 이미지를 screen coordinate (pixel 좌표계)로 맵핑시킴
- 출력 이미지 (screen)의 크기와 위치에 따라 달라짐

뷰잉 변환(Viewing Transformation)

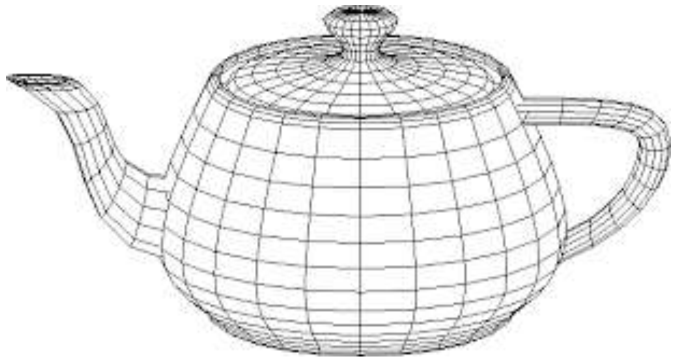


뷰잉 변환(Viewing Transformation)

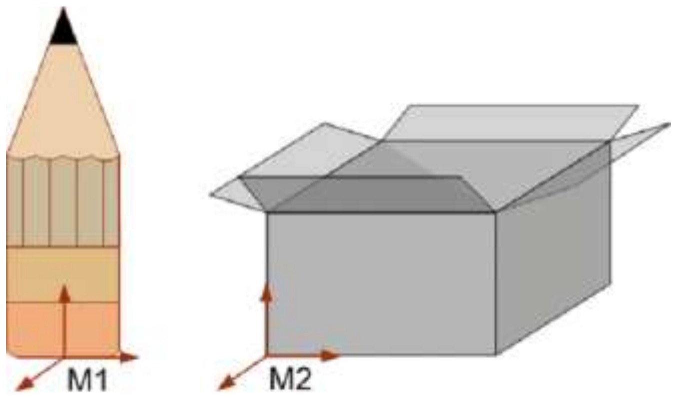


Modeling Coordinate(MC)

- Modeling
  - 3D 물체들을 점들의 집합으로 표현
  - 3D 물체들의 면을 mesh로 정의



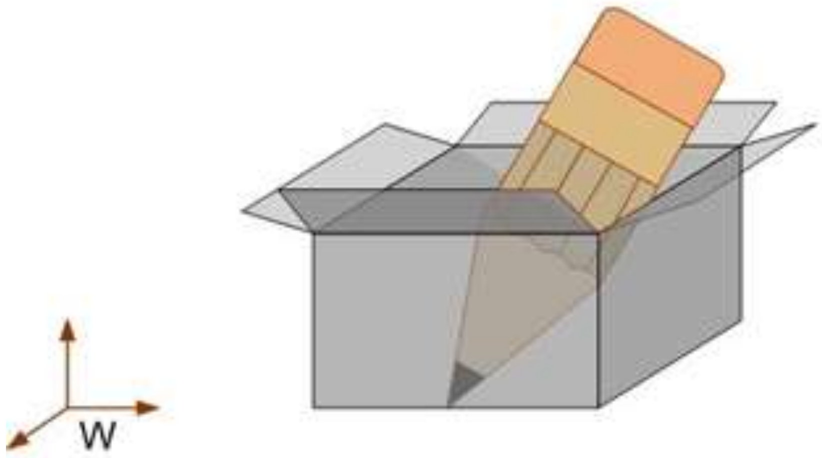
- Modeling(Object/Local) Coordinate System
  - 3D 물체들을 각각의 지역 좌표계(Local coordinate system)에서 표현
  - 각각의 물체가 다른 원점과 축을 가지고 있음



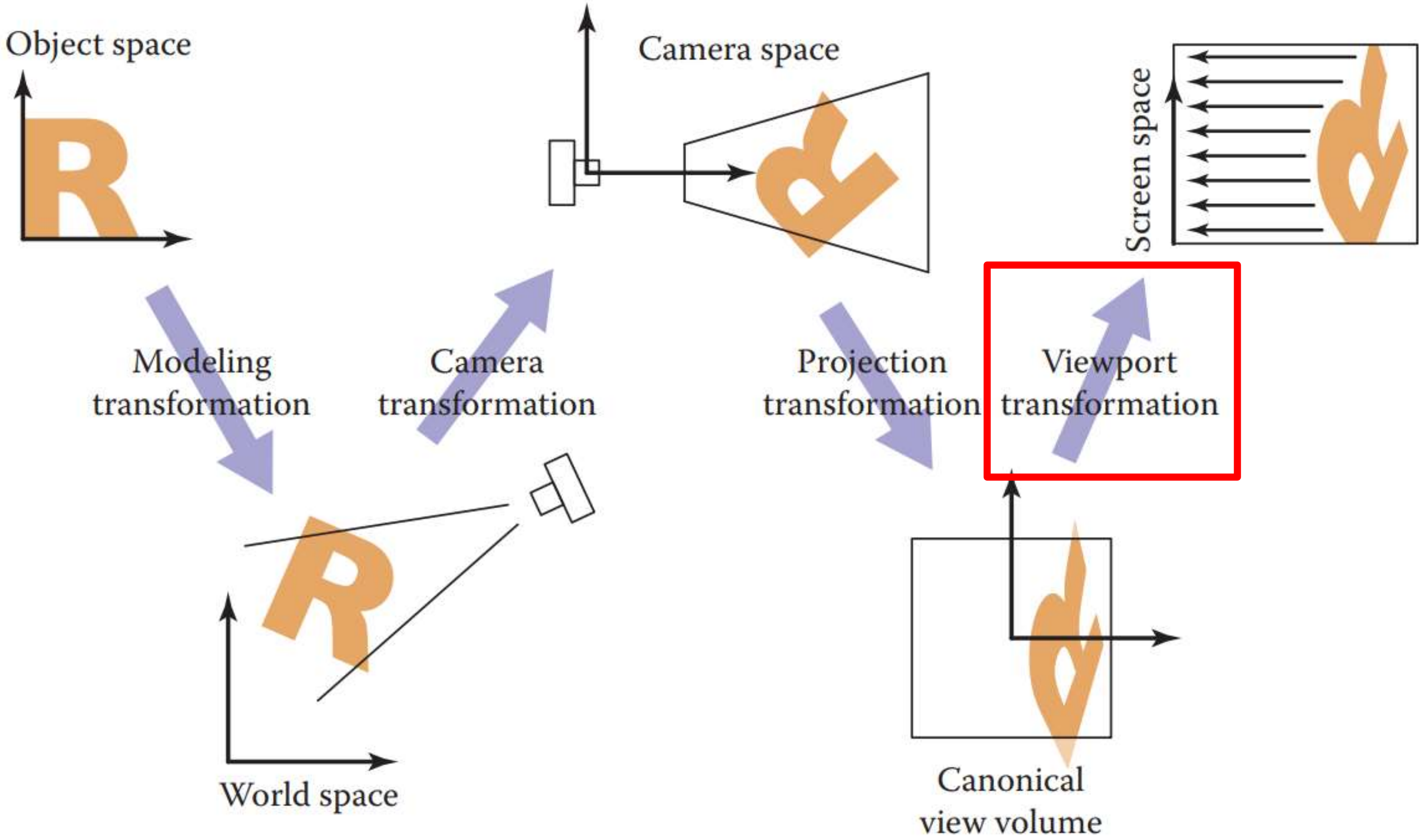
World Coordinate(WC)

- 3D World
  - 모든 물체들을 하나의 좌표계에서 표현
  - canonical coordinate system = reference coordinate system = world coordinate system

- World Coordinate System (WCS)
  - 3차원 상의 임의의 원점과 축을 가짐

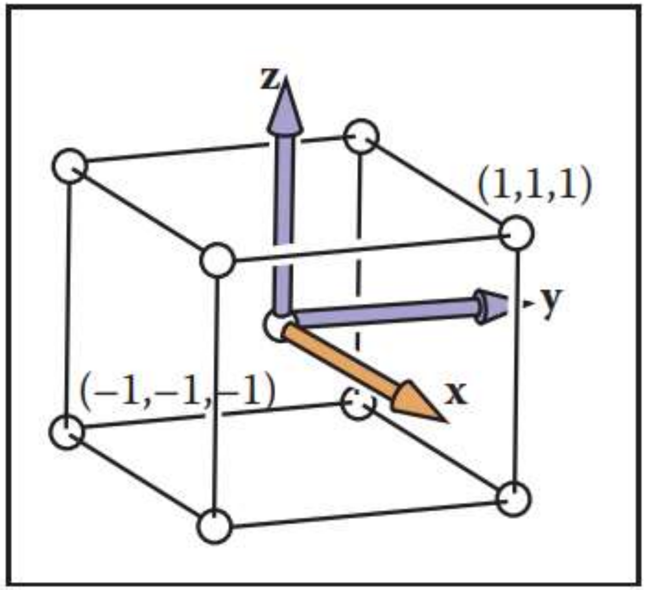


뷰잉 변환(Viewing Transformation)

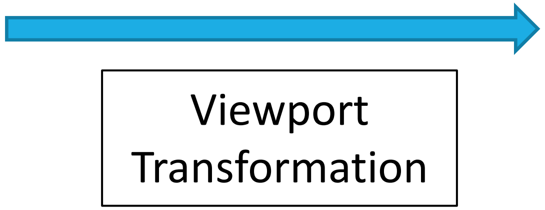


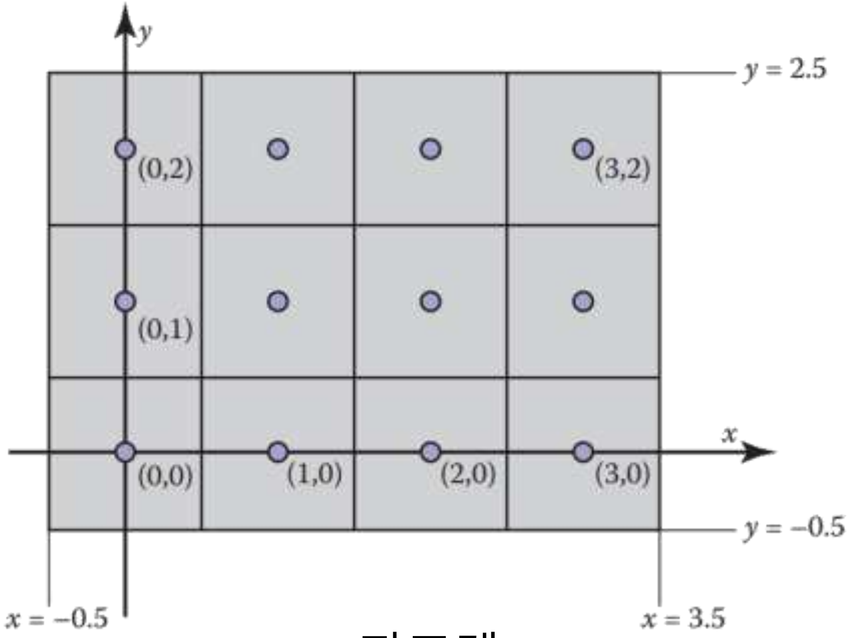
Viewport Transformation

- Canonical View Volume에 있는 모든 3D 좌표를 스크린위의 pixel 좌표로 변환
  - Canonical View Volume: (x,y,z) 좌표가 [-1,1]x[-1,1]x[-1,1]인 3D 큐브
  - Pixel 좌표계: $[-0.5, n_x - 0.5] \times [-0.5, n_y - 0.5]$ (n_x, n_y: x,y 방향 pixel 개수)



Canonical View Volume





Pixel 좌표계

Viewport Transformation

- Viewport transformation: 2D window 변환의 일종

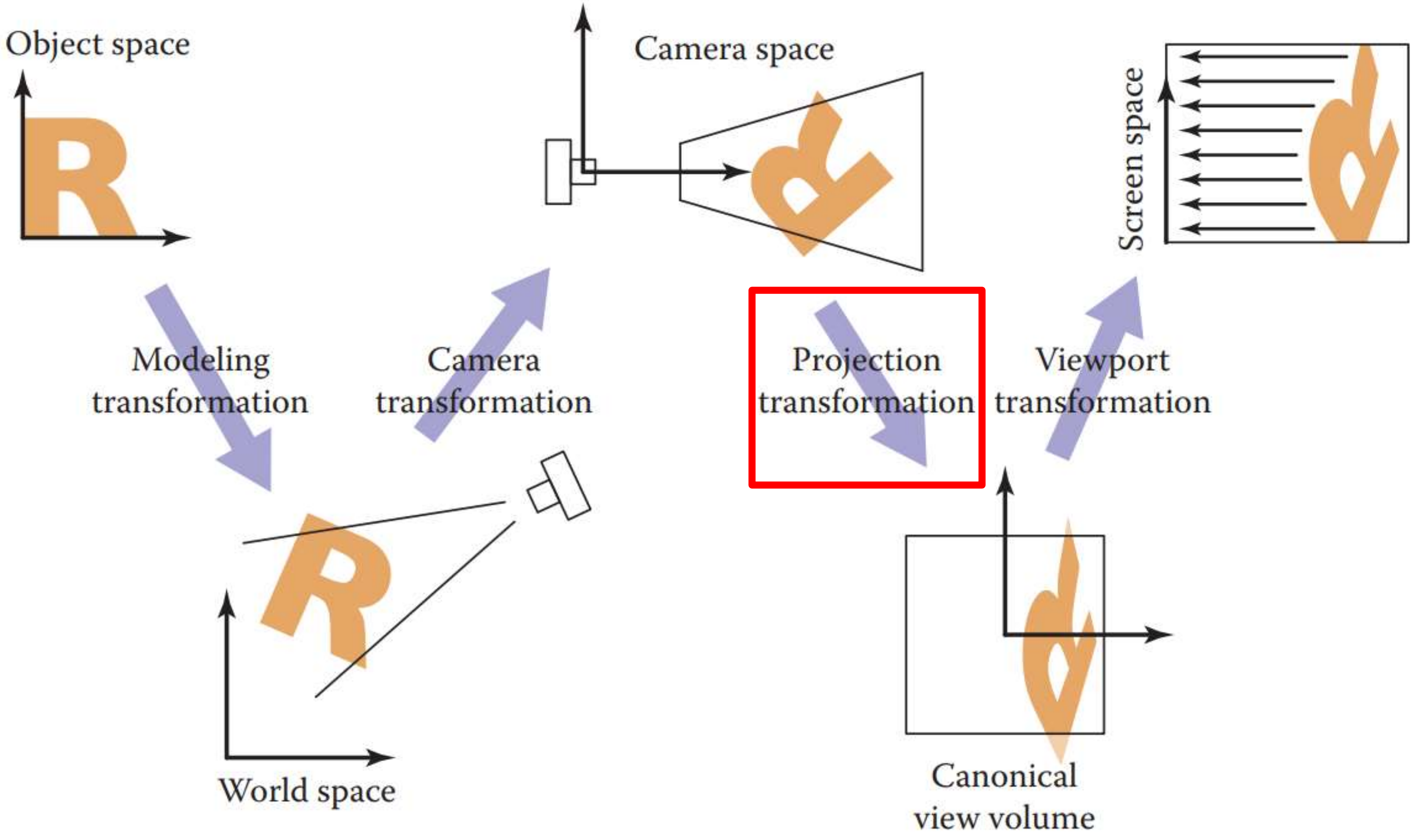
$\Rightarrow$ Translate(1,1) $\rightarrow$ Scale($\frac{n_x}{2}, \frac{n_y}{2}$) $\rightarrow$ Translate(-0.5, -0.5)

$$\begin{bmatrix} x_{\text{screen}} \\ y_{\text{screen}} \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{n_x}{2} & 0 & \frac{n_x-1}{2} \\ 0 & \frac{n_y}{2} & \frac{n_y-1}{2} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{\text{canonical}} \\ y_{\text{canonical}} \\ 1 \end{bmatrix}$$

- Z 값을 유지시키는 3D viewport transformation matrix

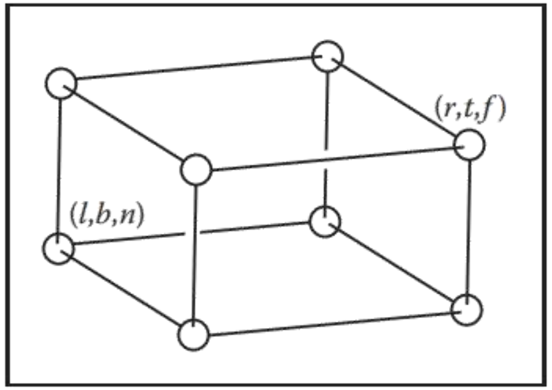
$$M_{\text{vp}} = \begin{bmatrix} \frac{n_x}{2} & 0 & 0 & \frac{n_x-1}{2} \\ 0 & \frac{n_y}{2} & 0 & \frac{n_y-1}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

뷰잉 변환(Viewing Transformation)



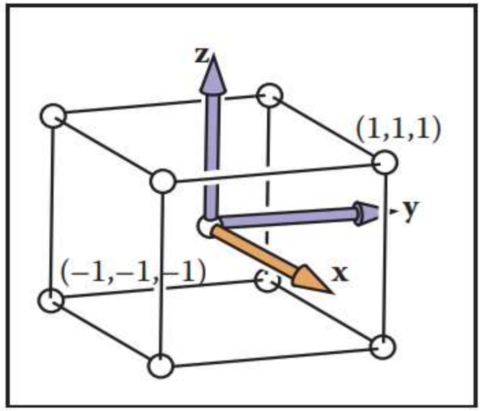
Orthographic Projection Transformation (직교 투영 변환)

- (Canonical view volume이 아닌) 임의의 view volume 그리기?
  - Viewing direction: -z axis
  - Up direction: +y axis
  - (x,y)의 범위는 [-1,1]x[-1,1]이 아닌 임의의 직사각형
- Orthographic Projection Transformation



Orthographic View Volume

Orthographic Projection Transformation

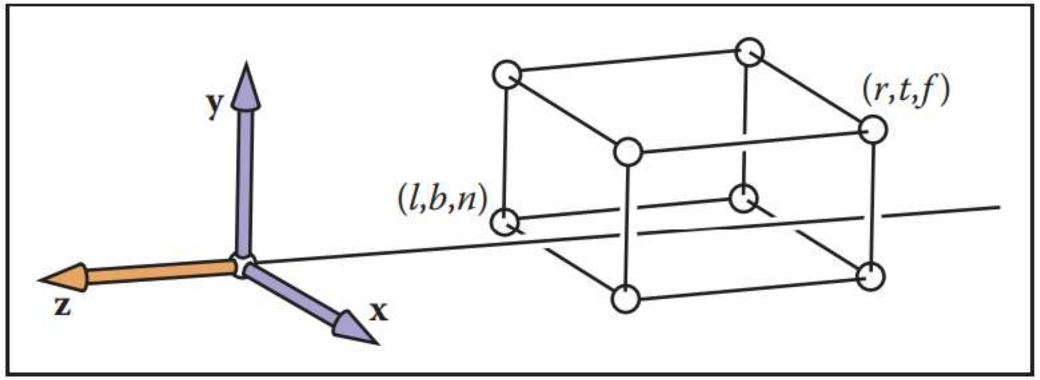


Canonical View Volume

Orthographic Projection Transformation

- Orthographic View Volume
  - -Z 방향으로 바라봄(viewing direction)
  - +y이 항상 위 (up direction)
  - Orthographic view volume: [l,r]x[b,t]x[f,n]

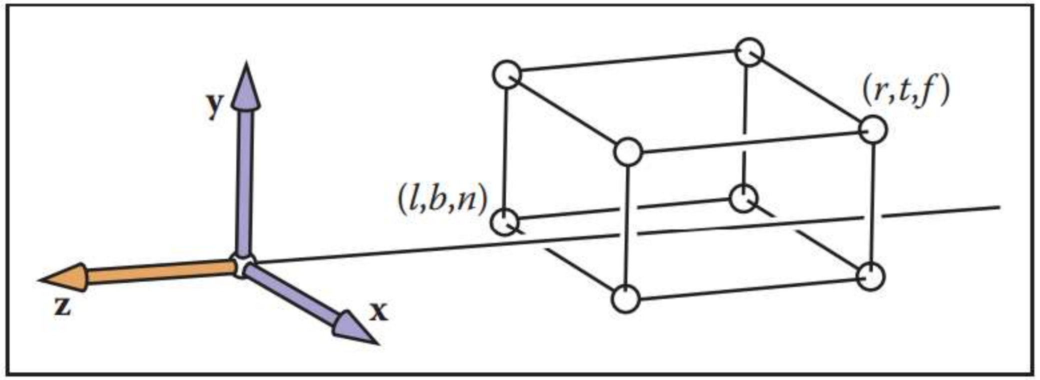
$$
\begin{aligned}
x = l &\equiv \text{left plane},\\
x = r &\equiv \text{right plane},\\
y = b &\equiv \text{bottom plane},\\
y = t &\equiv \text{top plane},\\
z = n &\equiv \text{near plane},\\
z = f &\equiv \text{far plane}.
\end{aligned}
$$



Orthographic Projection Transformation

- Orthographic Projection Transformation
  - $\Rightarrow$ [l,r]x[b,t]x[f,n]를 [-1,1]x[-1,1]x[-1,1]로 변환
  - $\Rightarrow$ Translate(-l, -b, -f) → Scale($\frac{2}{r-l}, \frac{2}{t-b}, \frac{2}{n-f}$) → Translate(-1, -1, -1)

$$\mathbf{M}_{\text{orth}} = \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & \frac{2}{n-f} & -\frac{n+f}{n-f} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Orthographic Projection Transformation

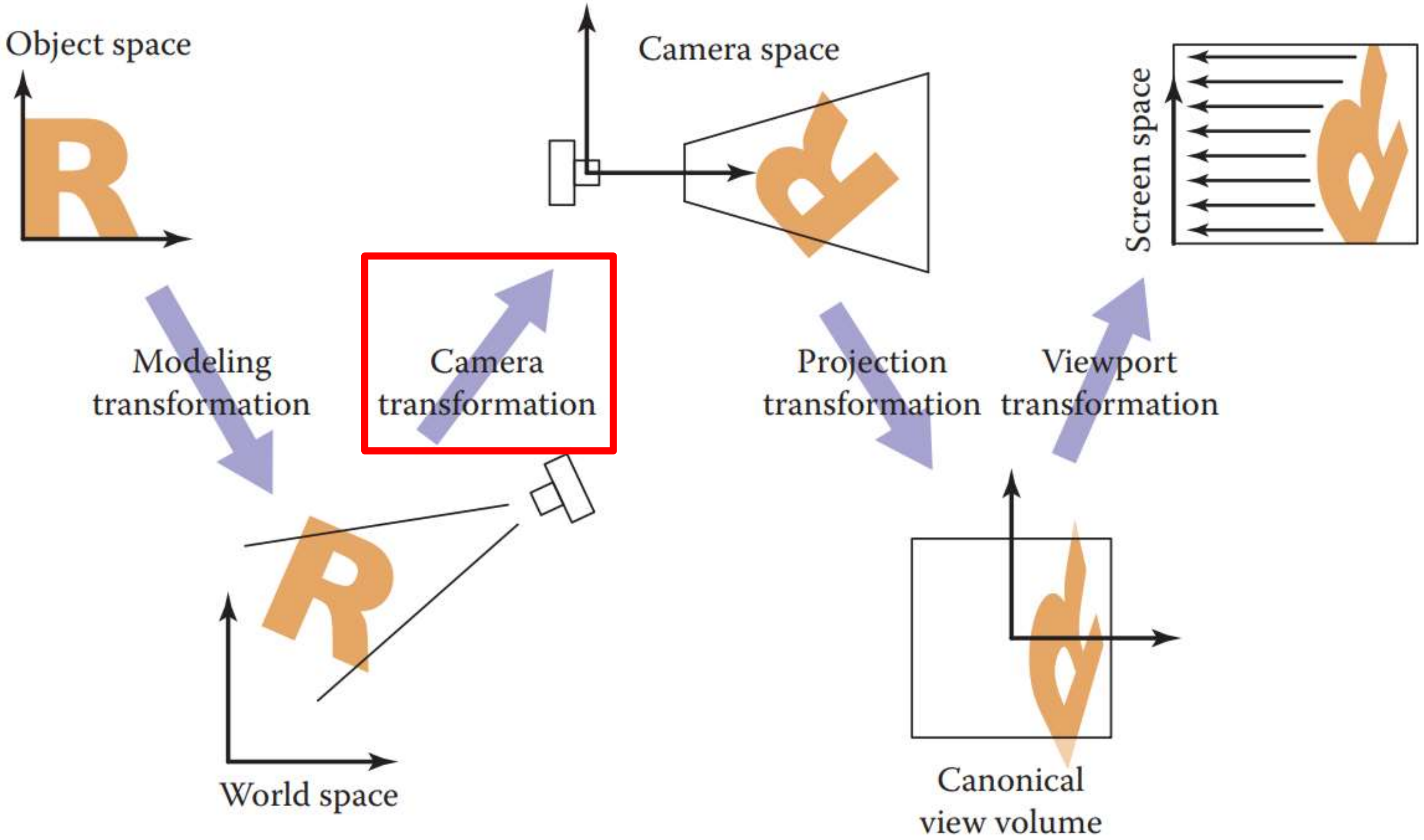
- Orthographic projection + viewport transformation

$$\begin{bmatrix} x_{\text{pixel}} \\ y_{\text{pixel}} \\ z_{\text{canonical}} \\ 1 \end{bmatrix} = (\mathbf{M}_{\text{vp}}\mathbf{M}_{\text{orth}}) \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- To draw many 3D lines with (a_i, b_i)

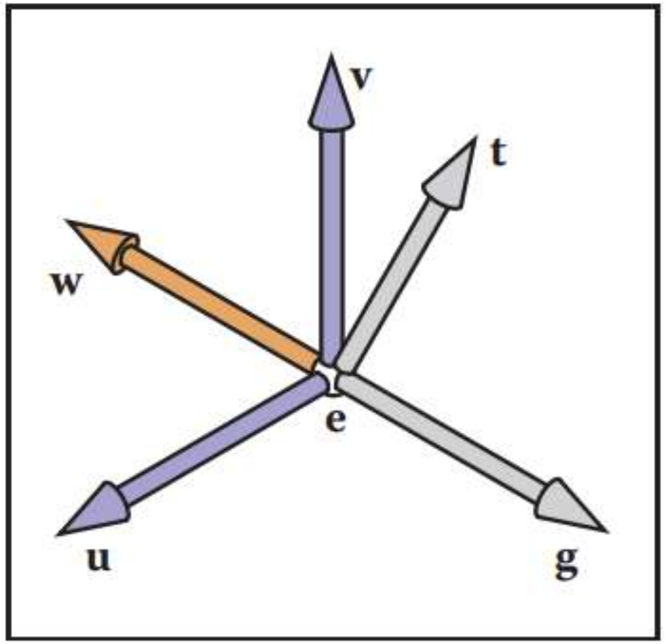
construct $\mathbf{M}_{\text{vp}}$
construct $\mathbf{M}_{\text{orth}}$
$\mathbf{M} = \mathbf{M}_{\text{vp}}\mathbf{M}_{\text{orth}}$
for each line segment $(\mathbf{a}_i, \mathbf{b}_i)$ **do**
    $\mathbf{p} = \mathbf{M}\mathbf{a}_i$
    $\mathbf{q} = \mathbf{M}\mathbf{b}_i$
    drawline(x_p, y_p, x_q, y_q)

뷰잉 변환(Viewing Transformation)



Camera Transformation

- Viewing direction, 카메라 위치가 임의인 scene 그리기?



- e: 카메라의 위치
- g: 카메라가 바라보는 방향
  (viewing direction = gazing direction)
- t: 카메라의 위쪽 방향 (view-up vector)

카메라 좌표계를 표현하는 직교좌표계 uvw 찾기

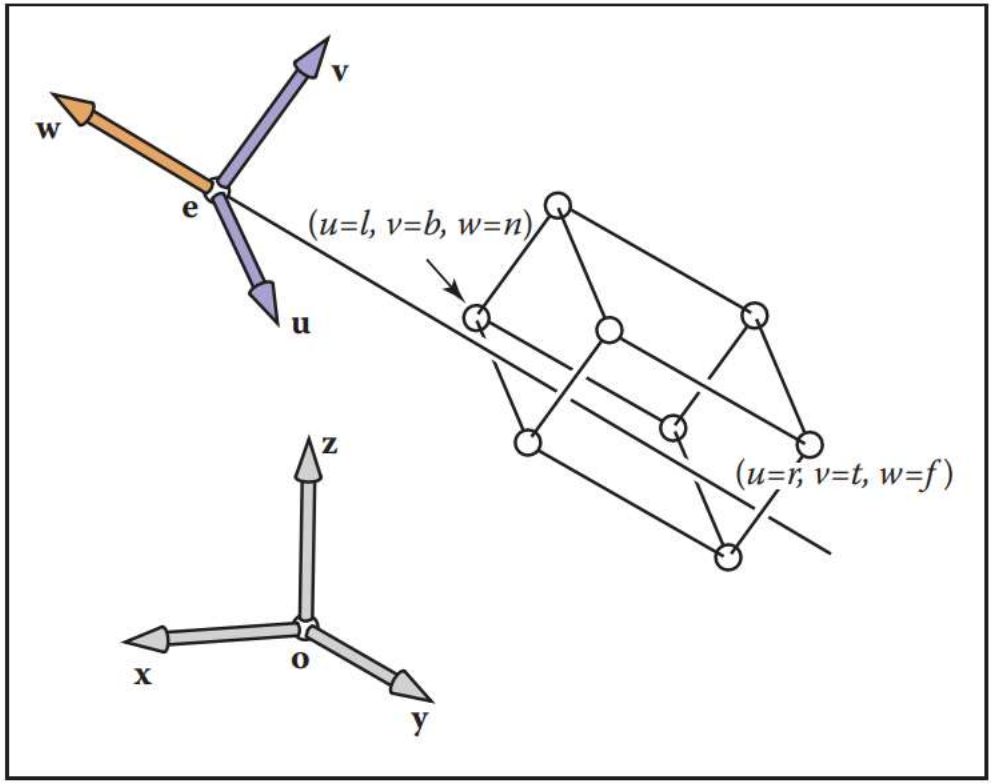
Camera Transformation

- 카메라 직교좌표계: 카메라의 위치(e)를 원점으로 하고 uvw를 세 축으로 하는 직교좌표계

$$\mathbf{w} = -\frac{\mathbf{g}}{\|\mathbf{g}\|},$$

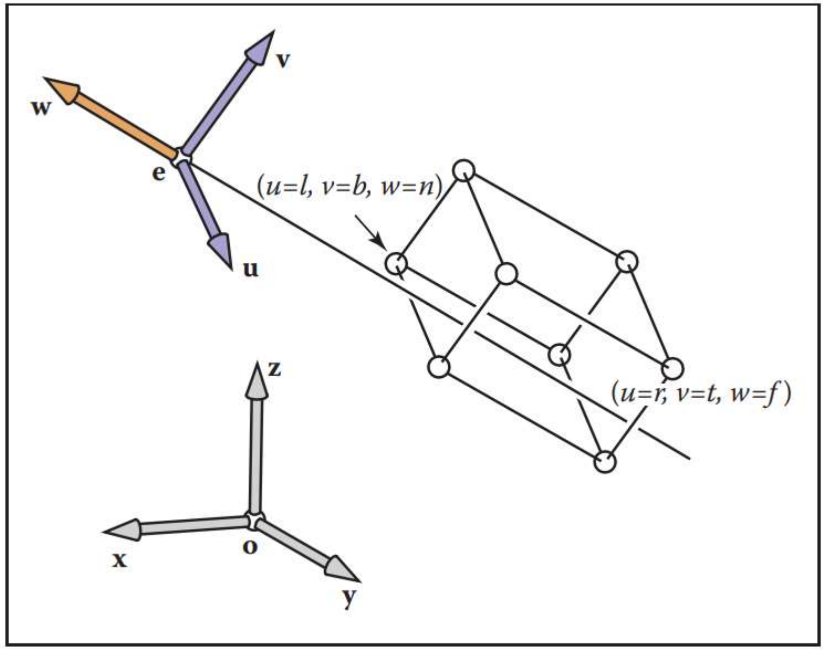
$$\mathbf{u} = \frac{\mathbf{t} \times \mathbf{w}}{\|\mathbf{t} \times \mathbf{w}\|},$$

$$\mathbf{v} = \mathbf{w} \times \mathbf{u}.$$

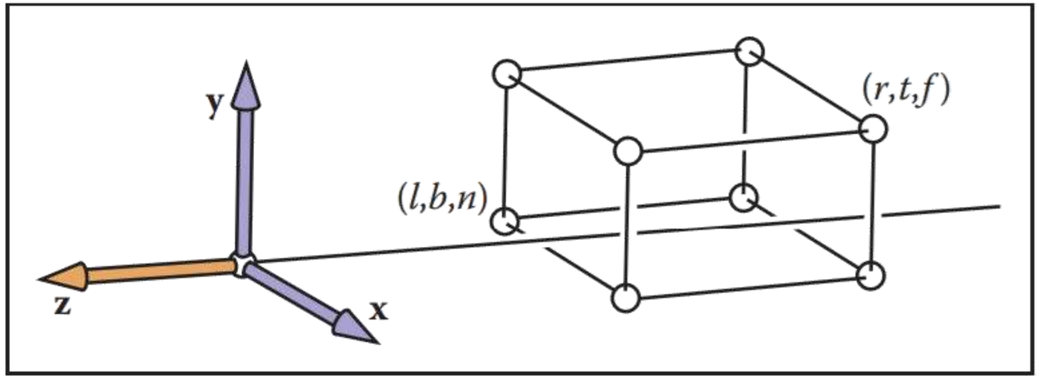


Camera Transformation

- Camera Transformation: 카메라 좌표계를 이용, 임의의 view volume을 orthographic view volume으로 변환



Camera Transformation



$$\mathbf{M}_{\text{cam}} = \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{w} & \mathbf{e} \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} x_u & y_u & z_u & 0 \\ x_v & y_v & z_v & 0 \\ x_w & y_w & z_w & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -x_e \\ 0 & 1 & 0 & -y_e \\ 0 & 0 & 1 & -z_e \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(e를 원점 o로 이동하고, uvw를 xyz로 정렬시킴)

Camera Transformation

- 임의의 view volume을 화면에 그리기

  ⇒ Camera transformation → Orthographic projection transformation → Viewport transformation

construct $\mathbf{M}_{\mathrm{vp}}$

construct $\mathbf{M}_{\mathrm{orth}}$

construct $\mathbf{M}_{\mathrm{cam}}$

$\boxed{\mathbf{M} = \mathbf{M}_{\mathrm{vp}}\mathbf{M}_{\mathrm{orth}}\mathbf{M}_{\mathrm{cam}}}$

for each line segment $(\mathbf{a}_i, \mathbf{b}_i)$ **do**

  $\mathbf{p} = \mathbf{M}\mathbf{a}_i$

  $\mathbf{q} = \mathbf{M}\mathbf{b}_i$

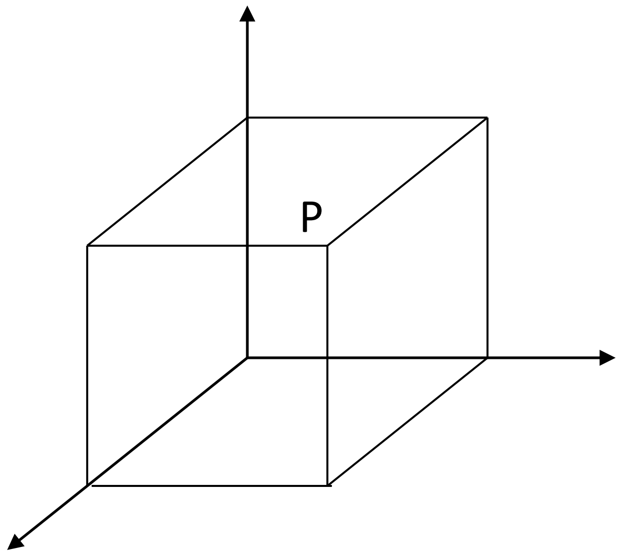
  drawline(x_p, y_p, x_q, y_q)

Example: Camera Transformation

- Q) Example
  - Unit cube located at the origin (0,0,0)
  - Camera(eye) position = (2,2,2)
  - Viewing direction: 카메라는 점 P(1,1,1)을 바라보고 있음
  - Up direction: (0,1,0)

$\Rightarrow M_{cam}$?

$\Rightarrow$ P점의 Camera coordinate



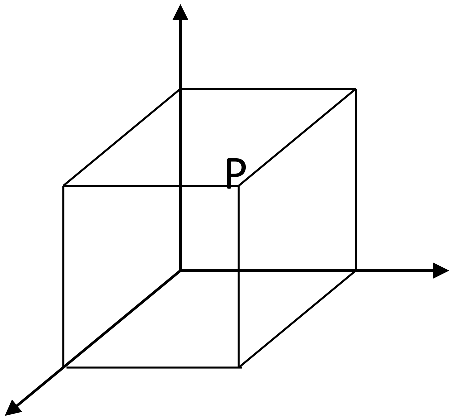
Example: Camera Transformation

Answer)

- e = (2,2,2)

  g = (1,1,1) – (2,2,2) = (-1,-1,1)

  t = (0,1,0)



$$\mathbf{w} = -\frac{\mathbf{g}}{\|\mathbf{g}\|},$$

$$\mathbf{u} = \frac{\mathbf{t}\times\mathbf{w}}{\|\mathbf{t}\times\mathbf{w}\|},$$

$$\mathbf{v} = \mathbf{w}\times\mathbf{u}.$$

$$w = -\frac{g}{\|g\|} = \frac{1,1,1}{\|(1,1,1)\|} = (\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}})$$

$$u = \frac{t\times w}{\|t\times w\|}, \quad t\times w = \begin{vmatrix} i & j & k \\ 0 & 1 & 0 \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \end{vmatrix} = (\frac{1}{\sqrt{3}}, 0, -\frac{1}{\sqrt{3}})$$

$$u = \frac{t\times w}{\|t\times w\|} = (\frac{1}{\sqrt{3}}, 0, -\frac{1}{\sqrt{3}}) / \left\| (\frac{1}{\sqrt{3}}, 0, -\frac{1}{\sqrt{3}}) \right\| = (\frac{1}{\sqrt{2}}, 0, -\frac{1}{\sqrt{2}})$$

$$v = w\times u = \begin{vmatrix} i & j & k \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} \end{vmatrix} = (-\frac{1}{\sqrt{6}}, \frac{2}{\sqrt{6}}, -\frac{1}{\sqrt{6}})$$

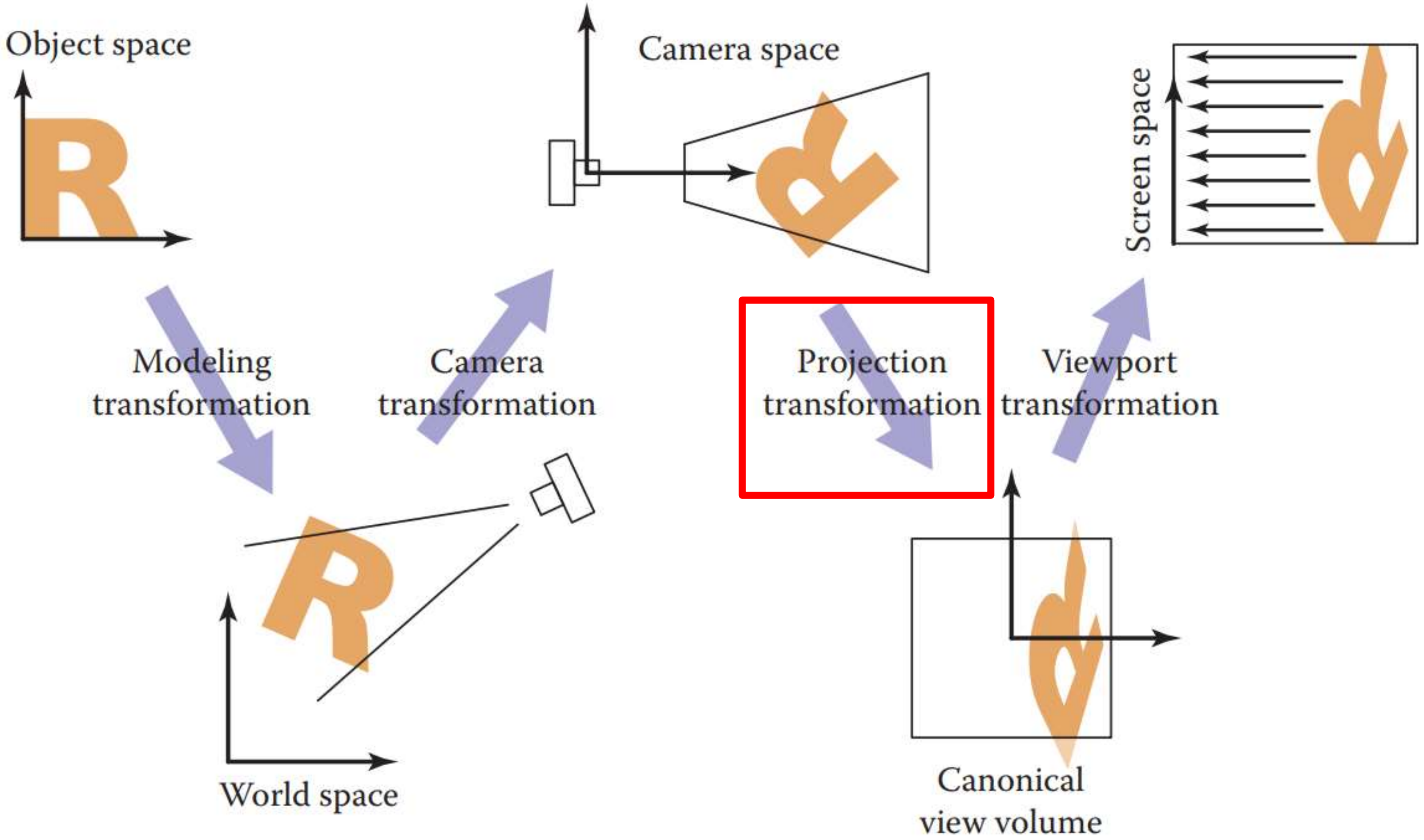
Example: Camera Transformation

$$\mathbf{M}_{\text{cam}} = \begin{bmatrix} \mathbf{u} & \mathbf{v} & \mathbf{w} & \mathbf{e} \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} x_u & y_u & z_u & 0 \\ x_v & y_v & z_v & 0 \\ x_w & y_w & z_w & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -x_e \\ 0 & 1 & 0 & -y_e \\ 0 & 0 & 1 & -z_e \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathrm{M}_{cam} = \begin{bmatrix} 1/\sqrt{2} & 0 & -1/\sqrt{2} & 0 \\ -1/\sqrt{6} & 2/\sqrt{6} & -1/\sqrt{6} & 0 \\ 1/\sqrt{3} & 1/\sqrt{3} & 1/\sqrt{3} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -2 \\ 0 & 1 & 0 & -2 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

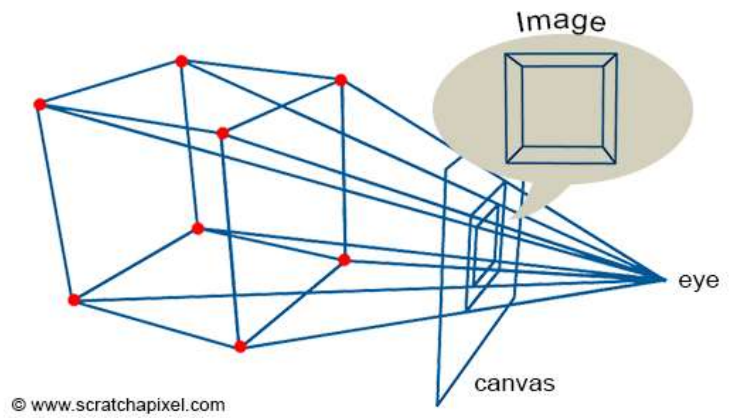
$$\mathrm{M}_{\mathrm{cam}}P = M_{cam} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -\sqrt{3} \\ 1 \end{bmatrix} \sim (0,0,-\sqrt{3})$$

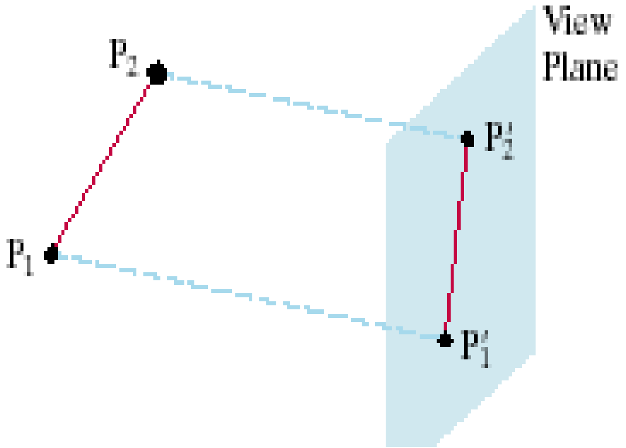
뷰잉 변환(Viewing Transformation)

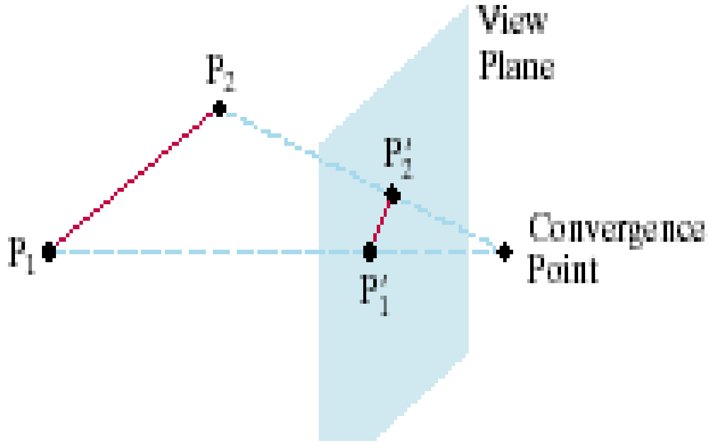


Projection Transformation

- Projection Transformation: 3D objects => 2D scene
  - Parallel Projection (평행 투영)
  - Perspective Projection (원근 투영)





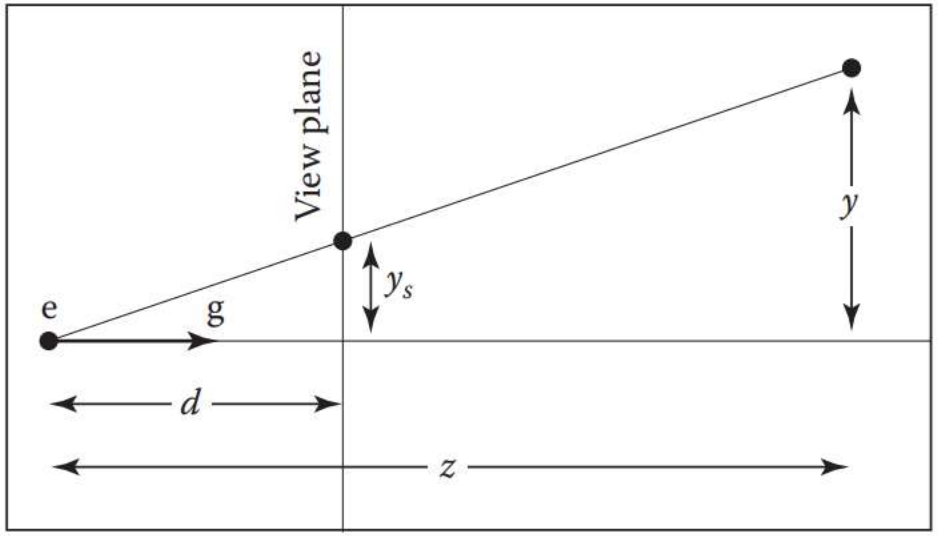


Projection Transformation

- Projection Transformation에서 y-좌표는 1/z에 비례함

$$y_s = \frac{d}{z} y$$

=> 행렬 연산으로 어떻게 표현하는가?



Homogeneous Coordinates

- Homogeneous coordinate system에서의 3D point 표현

$[x\ y\ z\ w]^T \sim \left[\frac{x}{w}\ \frac{y}{w}\ \frac{z}{w}\ 1\right]^T \sim \left(\frac{x}{w}, \frac{y}{w}, \frac{z}{w}\right)$ (non-homogeneous coord.)

- 다음과 같이 표현하는 것도 가능

$$x' = \frac{a_1 x + b_1 y + c_1 z + d_1}{ex + fy + gz + h},$$

$$y' = \frac{a_2 x + b_2 y + c_2 z + d_2}{ex + fy + gz + h},$$

$$z' = \frac{a_3 x + b_3 y + c_3 z + d_3}{ex + fy + gz + h}.$$

$\boxed{w = ex + fy + gz + h}$

Homogeneous Coordinates

- Homogeneous coordinate system은 linear rational 변환을 표현하는 것이 가능

$$x' = \frac{a_1 x + b_1 y + c_1 z + d_1}{ex + fy + gz + h},$$

$$y' = \frac{a_2 x + b_2 y + c_2 z + d_2}{ex + fy + gz + h},$$

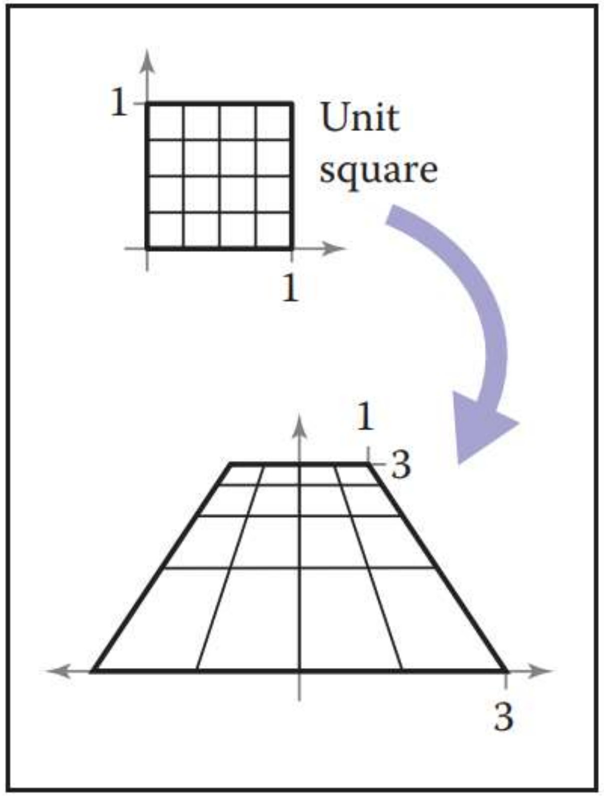
$$z' = \frac{a_3 x + b_3 y + c_3 z + d_3}{ex + fy + gz + h}.$$

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ a_3 & b_3 & c_3 & d_3 \\ e & f & g & h \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$(x', y', z') = (\tilde{x}/\tilde{w}, \tilde{y}/\tilde{w}, \tilde{z}/\tilde{w}).$$

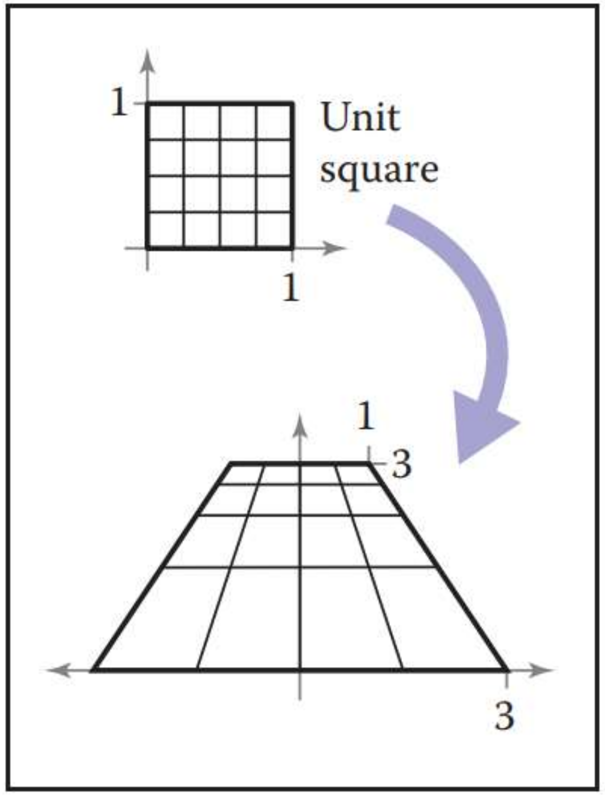
Example: 2D Projection Matrix

Q) [0,1]x[0,1]을 그림의 사변형과 같이 투영변환시키는 행렬은?



Example: 2D Projection Matrix

Q) [0,1]x[0,1]을 그림의 사변형과 같이 투영변환시키는 행렬은?



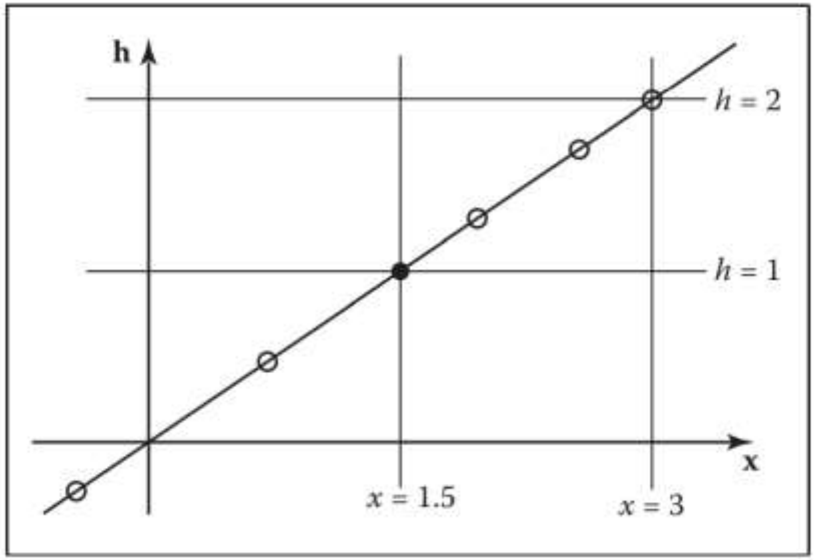
Answer) $M = \begin{bmatrix} 2 & 0 & -1 \\ 0 & 3 & 0 \\ 0 & \frac{2}{3} & \frac{1}{3} \end{bmatrix}$

- $[1\ 0\ 1]^T \rightarrow \left[1\ 0\ \frac{1}{3}\right]^T \sim (3\ 0)$ 으로 변환
- cM 변환은 M 변환과 동일

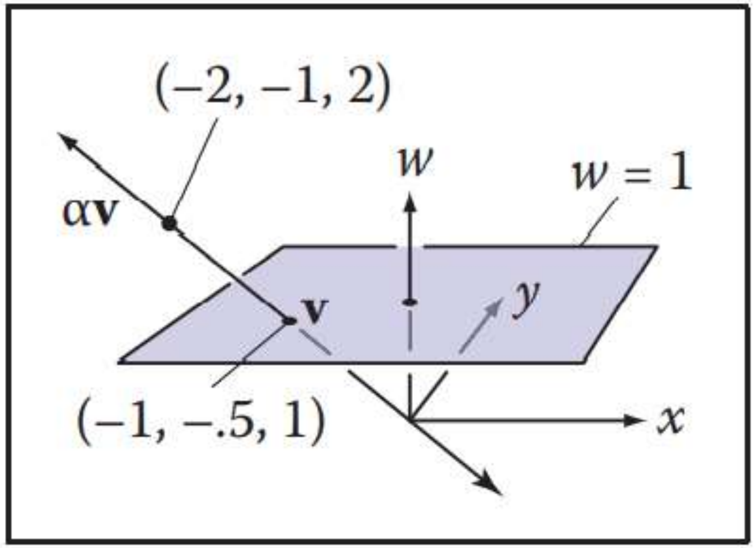
Projection Transformation

- Homogeneous coordinate system을 이용하면 nD 투영 변환 (projection transformation)을 (n+1)D 행렬 연산으로 표현 가능

$$\boldsymbol{x} \sim \alpha \boldsymbol{x} \quad \text{for } \alpha \neq 0$$



1D Homogeneous coordinate system for x=1.5



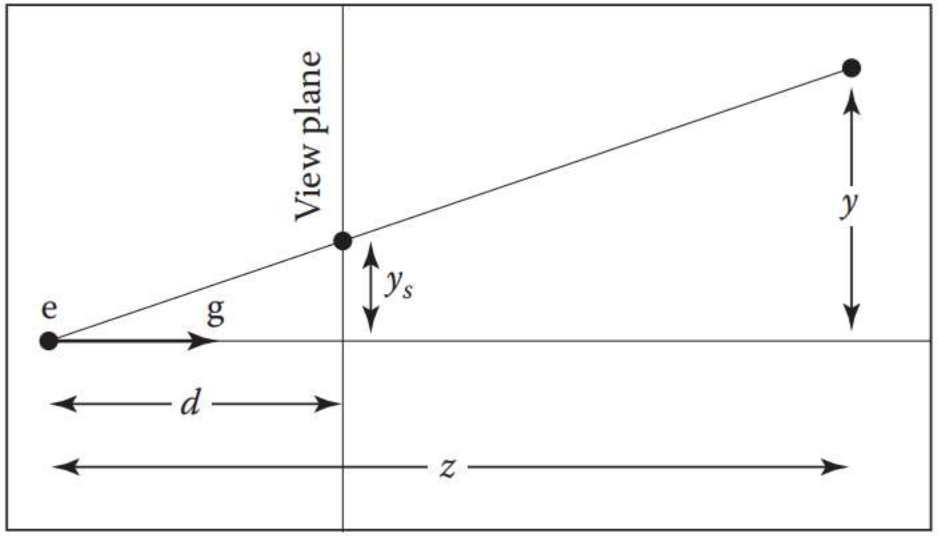
2D Homogeneous coordinate system for (-1, -0.5)

Perspective (Projection) Transformation (원근 투영 변환)

- y_s계산:

$$\begin{bmatrix} y_s \\ 1 \end{bmatrix} \sim \begin{bmatrix} d & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} y \\ z \\ 1 \end{bmatrix}$$

$\Rightarrow$ 2D homogeneous vector $[y\ z\ 1]^T$을
1D homogeneous vector $[dy\ z]^T$(점 dy/z)로 변환



- x_s계산:

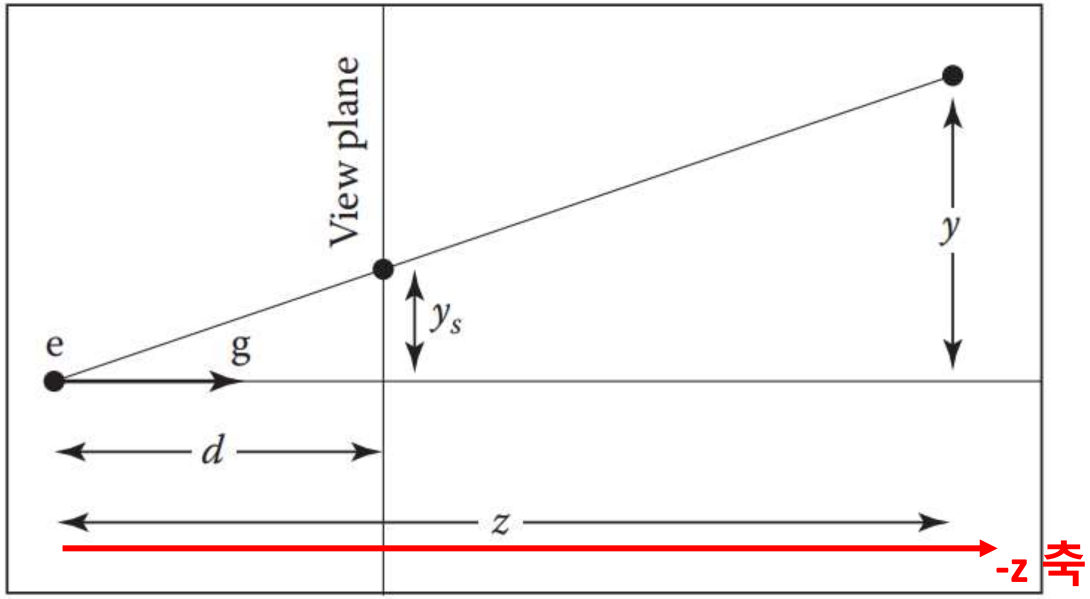
$$\begin{bmatrix} x_s \\ 1 \end{bmatrix} \sim \begin{bmatrix} d & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ z \\ 1 \end{bmatrix}$$

$\Rightarrow$ 2D homogeneous vector $[x\ z\ 1]^T$을
1D homogeneous vector $[dx\ z]^T$(점 dx/z)로 변환

Perspective Transformation

- Q) 3D 점 $[x\ y\ z\ 1]^T$은 perspective transformation에 의해 어떻게 변환되는가?

$$\mathbf{P} = \left[\begin{array}{cccc} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ \multicolumn{4}{c}{\boxed{\quad ? \quad}} \\ 0 & 0 & 1 & 0 \end{array}\right]$$



d = -n: distance to near plane
(view plane = near plane)

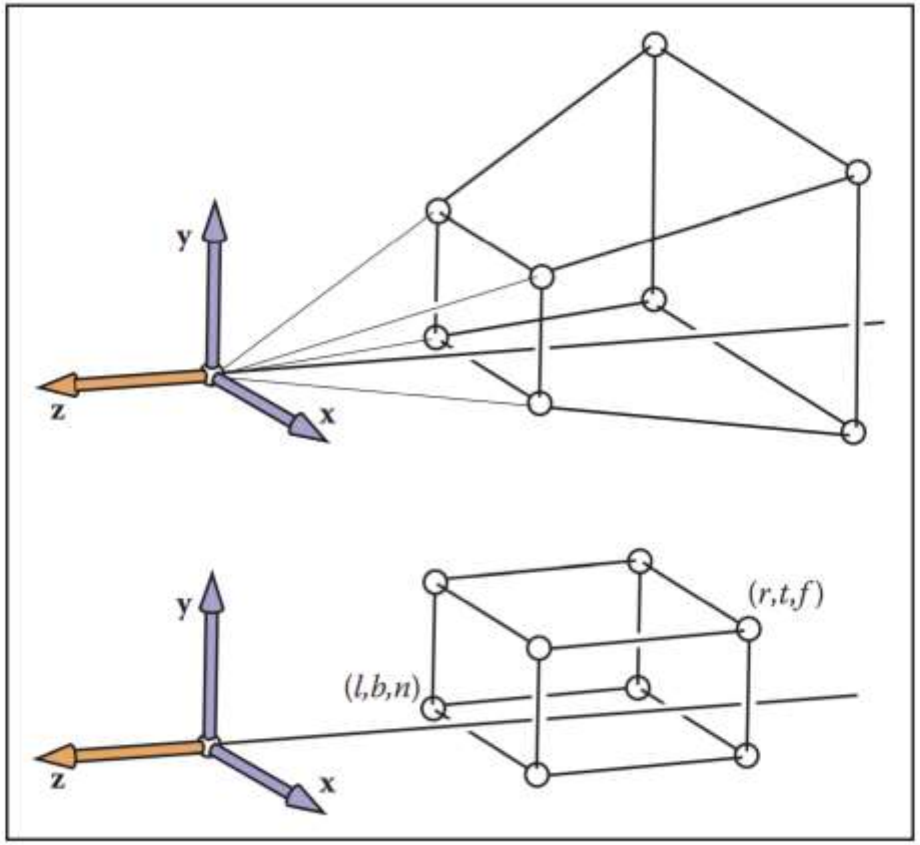
Perspective Transformation

- Perspective transformation후에도 z값의 ordering을 유지
  - z값은 hidden surface removal에 유용함
  - Non-homogeneous coordinate로 바꾸는 과정에서 z값이 바꾸는 것을 막을 수 없음

$\Rightarrow$ Near/far plane 근처에서는 z값이 바뀌지 않도록 P 구성

$$\mathbf{P} = \begin{bmatrix} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ 0 & 0 & n+f & -fn \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Perspective Transformation



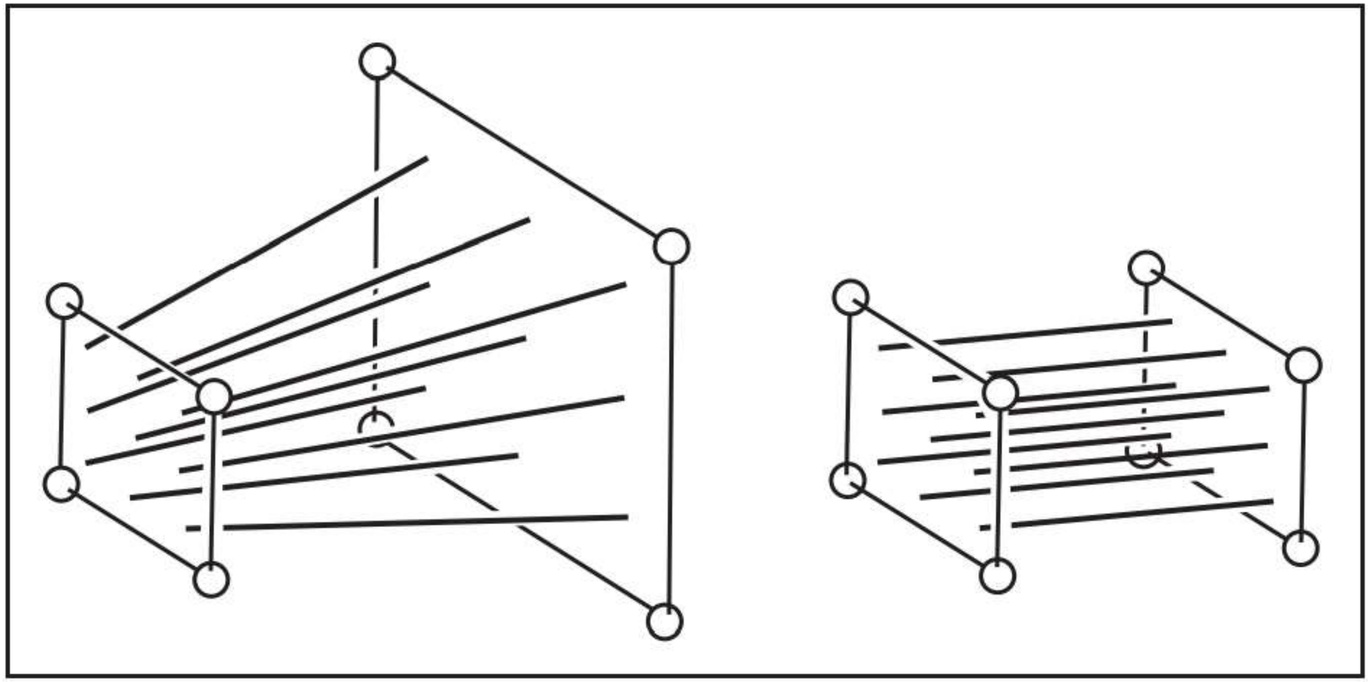
$$\mathbf{P} = \begin{bmatrix} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ 0 & 0 & n+f & -fn \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\mathbf{P}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} nx \\ ny \\ (n+f)z - fn \\ z \end{bmatrix} \sim \begin{bmatrix} \frac{nx}{z} \\ \frac{ny}{z} \\ n+f-\frac{fn}{z} \\ 1 \end{bmatrix}$$

- z = n일 때: x, y의 값은 변하지 않는다
- z = f일 때: x, y의 값이 near plane과 같은 크기를 가지는 작은 직사각형 안으로 축소됨

Perspective Transformation

- Perspective transformation후의 view volume은 orthographic view volume으로 변환
  - 눈(카메라와)와 각 점을 잇는 직선이 평행한 직선으로 변환



$$\mathbf{P} = \begin{bmatrix} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ 0 & 0 & n+f & -fn \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Perspective Transformation

- Inverse of perspective transformation: 원근 투영 변환으로 그려진 화면에서 스크린 coordinate + z 값을 투영 전으로 변환
  - 3D picking 등에 사용

$$\mathbf{P}^{-1} = \begin{bmatrix} \frac{1}{n} & 0 & 0 & 0 \\ 0 & \frac{1}{n} & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{1}{fn} & \frac{n+f}{fn} \end{bmatrix} \sim \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 0 & fn \\ 0 & 0 & -1 & n+f \end{bmatrix}$$

Perspective Transformation Matrix

- Perspective transformation matrix

$$\mathbf{M}_{\text{per}} = \mathbf{M}_{\text{orth}}\mathbf{P}$$

$$\mathbf{M}_{\text{per}} = \begin{bmatrix} \frac{2n}{r-l} & 0 & \frac{l+r}{l-r} & 0 \\ 0 & \frac{2n}{t-b} & \frac{b+t}{b-t} & 0 \\ 0 & 0 & \frac{f+n}{n-f} & \frac{2fn}{f-n} \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Perspective Transformation

- Perspective transformation summary
  - 원근 투영으로 투영된 점을 화면에 표시하는 변환

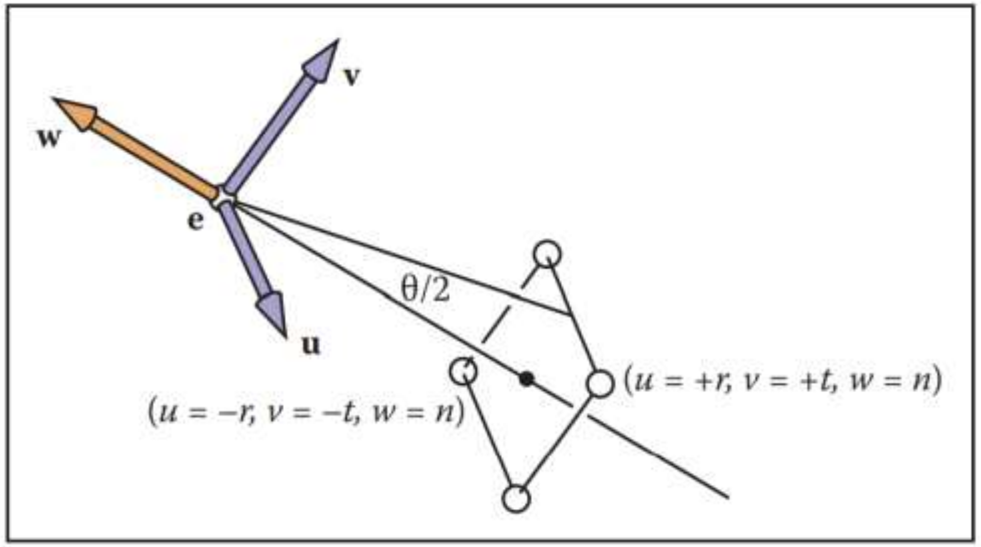
$$\mathbf{M} = \mathbf{M}_{\mathrm{vp}}\mathbf{M}_{\mathrm{orth}}\mathbf{P}\mathbf{M}_{\mathrm{cam}}$$

compute $\mathbf{M}_{\mathrm{vp}}$
compute $\mathbf{M}_{\mathrm{per}}$
compute $\mathbf{M}_{\mathrm{cam}}$
$\mathbf{M} = \mathbf{M}_{\mathrm{vp}}\mathbf{M}_{\mathrm{per}}\mathbf{M}_{\mathrm{cam}}$
for each line segment $(\mathbf{a}_i, \mathbf{b}_i)$ **do**
  $\mathbf{p} = \mathbf{M}\mathbf{a}_i$
  $\mathbf{q} = \mathbf{M}\mathbf{b}_i$
  drawline$(x_p/w_p, y_p/w_p, x_q/w_q, y_q/w_q)$

Field-of-view (FOV)

- Near plane의 중점을 원점으로 지정하면
  - l = -r, b = -t
  - 이때 distortion을 최소화하기 위해서는 $\frac{n_x}{n_y} = \frac{r}{t}$ 여야 함 (aspect ratio)
- Field-of-view (화면의 zoom을 결정)

$$\tan\frac{\theta}{2} = \frac{t}{|n|}$$



Field-of-view

- Zoom in/out 효과
  - FOV 각도 감소

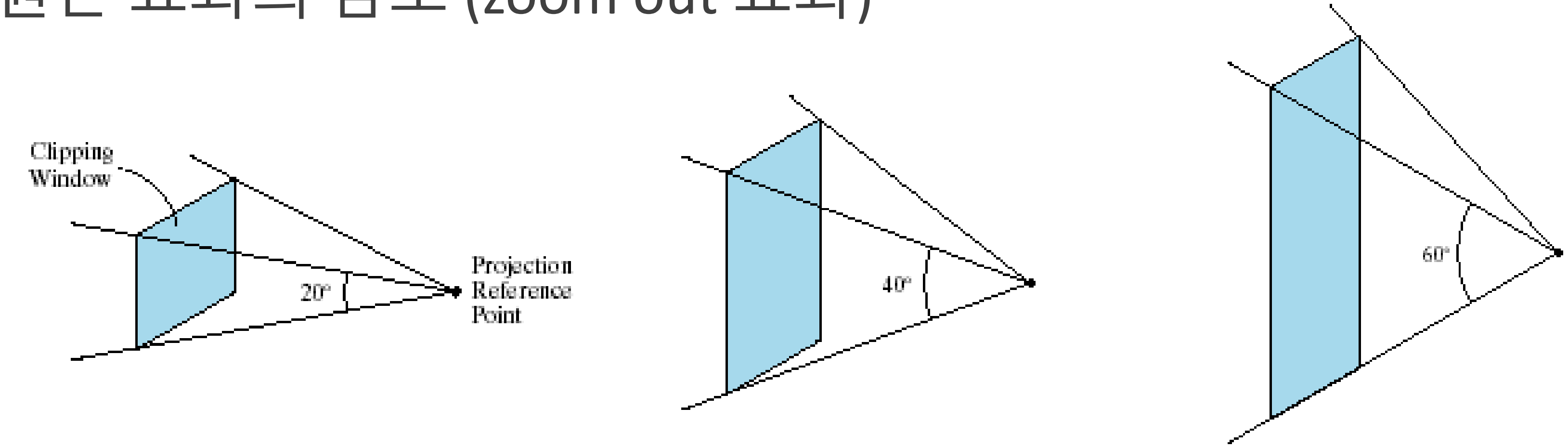
⇒ 카메라(눈)의 위치를 멀리 이동하는 것과 같은 효과

⇒ 원근 효과의 감소 (zoom in 효과)

  - FOV 각도 증가

⇒ 카메라(눈)의 위치를 view plane 쪽으로 이동하는 것과 같은 효과

⇒ 원근 효과의 감소 (zoom out 효과)



뷰잉 변환(Viewing Transformation)