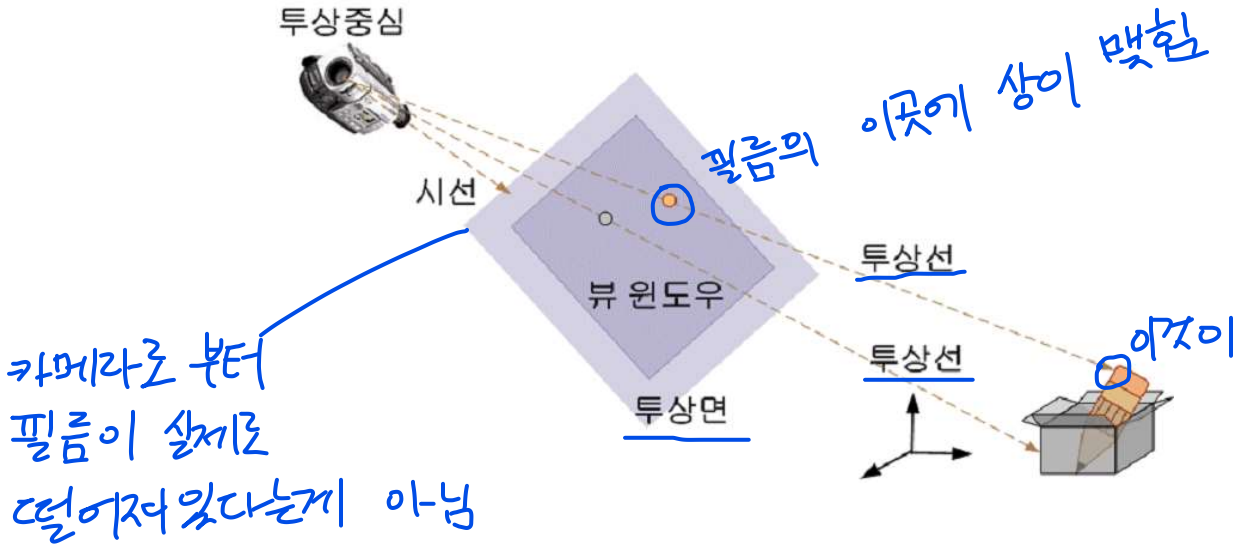
7장. 투상변환과 뷰포트변환

학습목표

- 평행투상과 원근투상의 차이점을 이해한다.
- 가시부피 설정방식을 이해한다.
- glOrtho( ) 함수와 gluPerspective( ) 함수 파라미터를 이해한다.
- 전방 절단면을 되도록 시점에서 멀리 가져가는 이유를 이해한다.
- 시점좌표, 절단좌표, 정규화 장치좌표, 화면좌표로의 변환과정을 이해한다.

투상

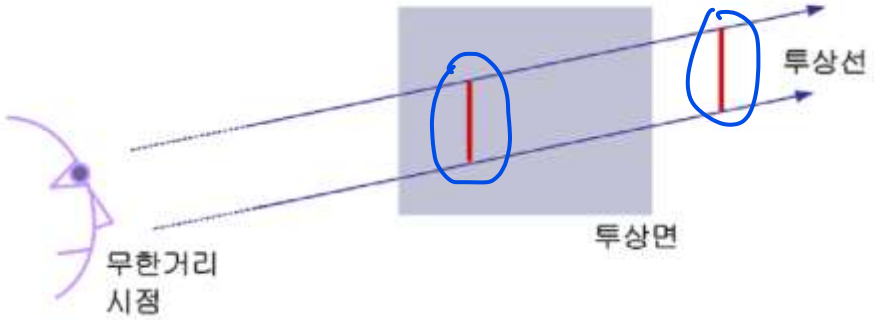
- 투상(Projection) = 가시변환(Viewing Transformation)



  - 투상면(View Plane, Projection Plane)
  - 관찰자 위치(View Point, Eye Position)
    = 카메라 위치(Camera Position) = 투상중심(COP: Center of Projection) = 시점좌표계 원점(Origin of VCS)
  - 투상선(Projectors): 물체 곳곳을 향함
  - 시선(Line of Sight) : WCS원점 또는 초점을 향함
  - 투상면(Projection Plane, View Plane)

평행투상(Parallel Projection)

원근투상에서 시점이 끝없이 뒤로가면 이것과 같다



장점 : 물체의 실제크기가 화면에 반영
= 원근법을 고려하지 않는다

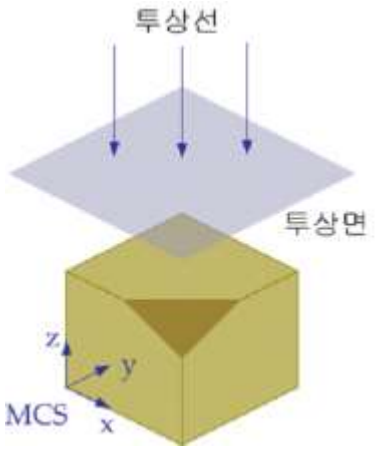
- 시점이 물체로부터 무한대의 거리에 있다고 간주
  - 투상선이 평행
  - 원래 물체의 평행선은 투상 후에도 평행
  - 시점과의 거리에 무관하게 같은 길이의 물체는 같은 길이로 투상

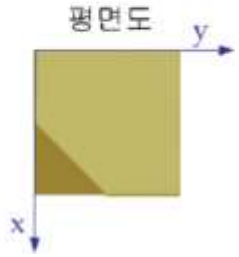
- 정사투상, 축측투상, 경사투상 등으로 분류

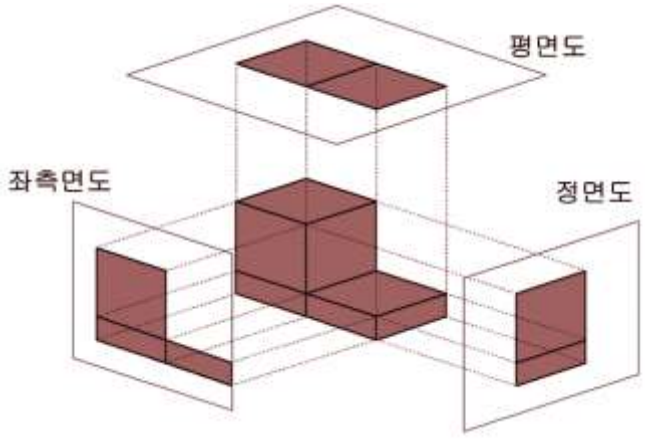
별로 안중요

정사투상(Orthographic Projection)

- 평면도, 입면도, 측면도 등
  - 주 평면(Principal Plane): MCS 주축인 x, y, z에 의해 형성되는 x-y, y-z, z-x
  - 투상면은 주 평면 중 하나와 평행
- 투상선은 투상면과 직교
  - 원래 물체의 길이를 정확히 보존. 공학도면에 사용
  - 투상선이 반드시 투상면과 직교-> 시점위치가 제한됨.





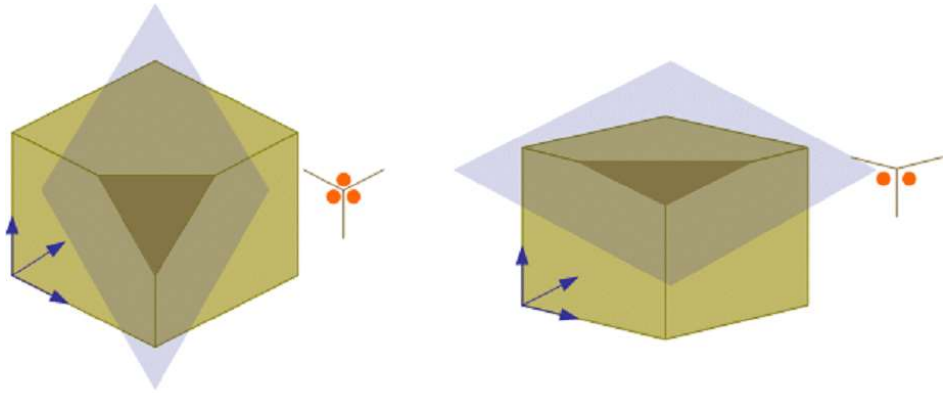


그냥 수직으로 바라보는 것

축측투상(Axonometric Projection)

- 한꺼번에 여러 면을 보여줌
  - 투상선은 투상면과 직교. 투상면이 주평면들과 평행하지 않음
- 축 방향으로 서로 다른 축소율(cf. 정사투상)
  - 삼각(삼중형, Trimetric)
    - 투상면이 임의의 위치.
  - 양각(이중형, Dimetric)
    - 투상면이 2 개의 주 평면에 대해서 대칭적.
    - 2개의 축 방향에 대해 동일 축소율
  - 등각(동형, Isometric)
    - 투상면이 3 개의 주 평면이 만나는 모서리에서 모든 평면에 대해 대칭적으로 놓일 때. 3개의 축 방향에 대해 동일 축소율

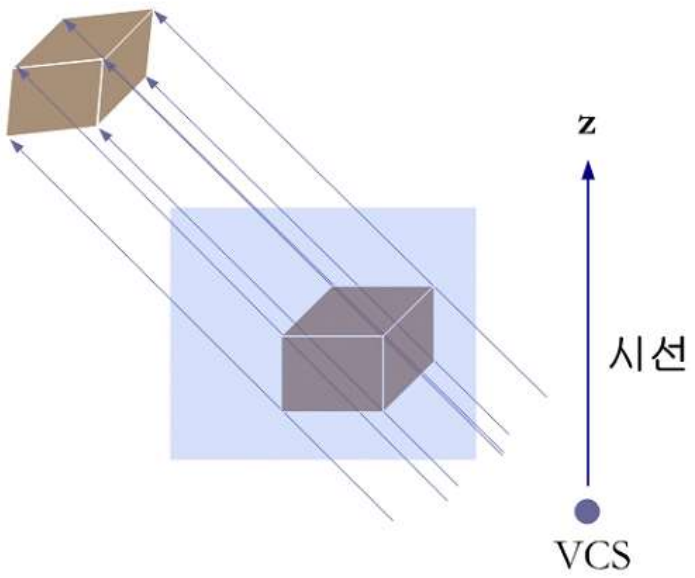
- Ex. 등각, 양각



(a) (b)

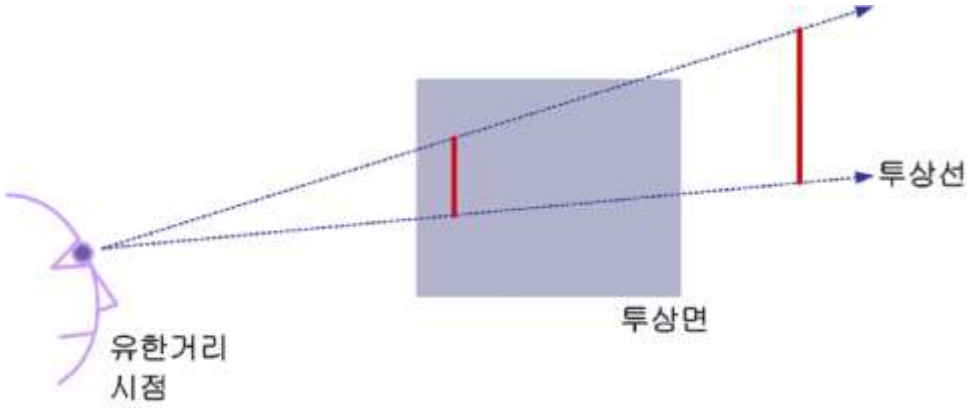
경사투상(Oblique Projection)

- 투상선끼리는 평행
- 투상면은 시선에 수직이지만 투상선과 직교하지 않음.
- 고개는 돌리지 않고 눈동자만 돌려서 보는 것과도 흡사



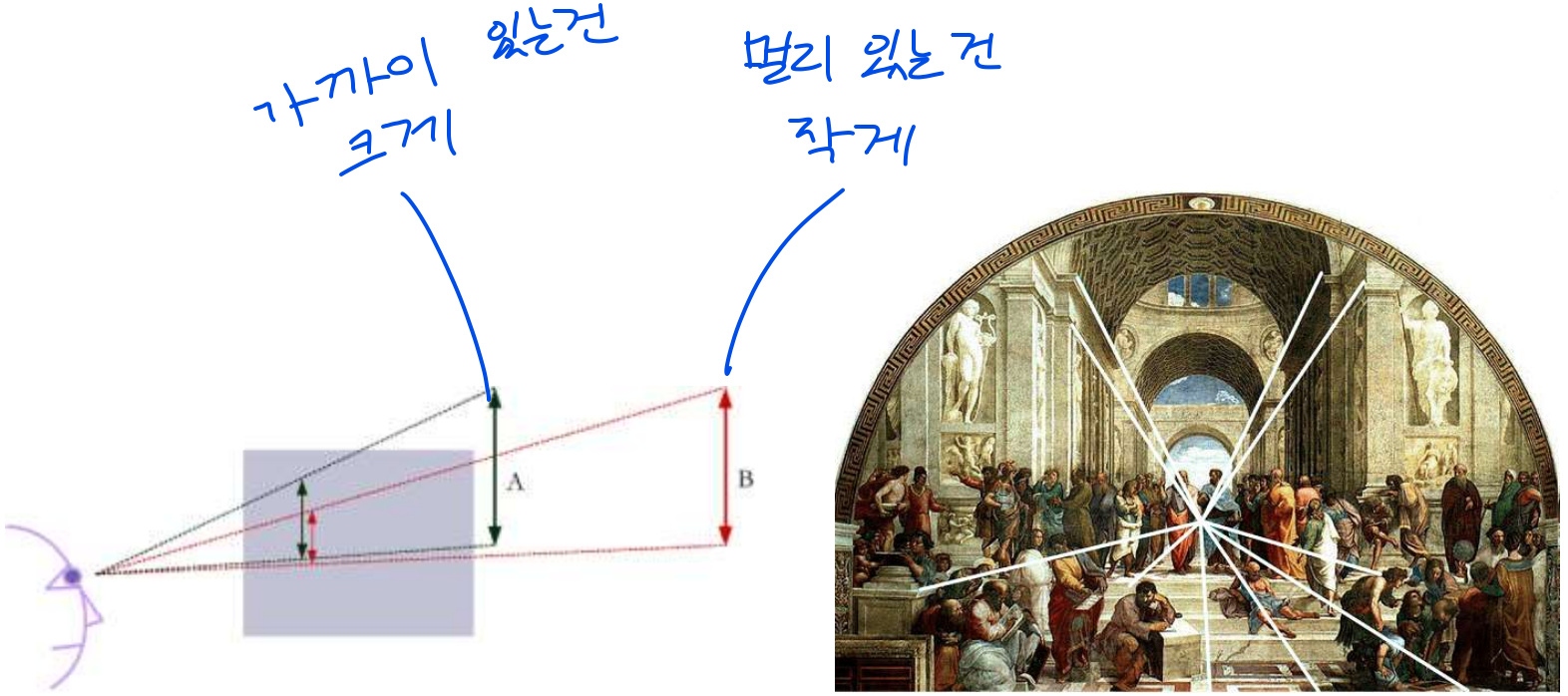
원근투상(Perspective Projection)

- 시점이 물체로부터 유한한 거리에 있다고 간주
- 투상선이 시점에서 출발하여 방사선 모양으로 퍼져감.
- 카메라나 사람의 눈이 물체를 포착하는 방법



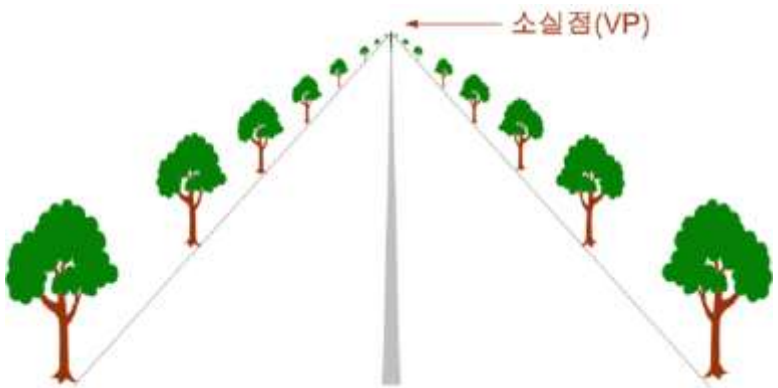
원근투상(Perspective Projection)

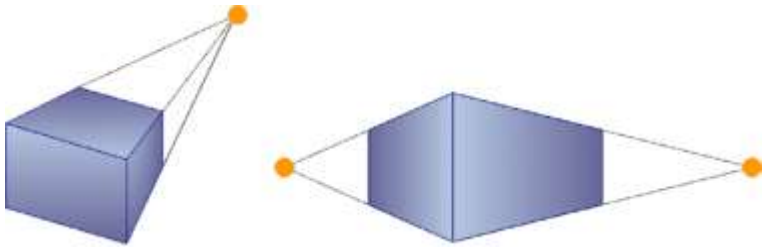
- 원근감(Depth Feeling)
  - 동일한 크기의 물체라도 시점으로부터 멀리 있는 것은 작게 보이고 가까운 것은 크게 보임



원근투상(Perspective Projection)

- 소실점(VP: Vanishing Point)
  - 원근투상 결과 평행선이 만나는 점(시점 높이)
  - 소실점의 수
    - 일점투상(One-point Projection), 이점투상(Two-point Projection), 삼점투상(Three-point Projection)
- 원근변환(Perspective Transformation)
  - 직선->직선, 평면->평면
  - 물체 정점간의 거리에 대한 축소율이 달라짐. (cf. 어파인 변환)





지엘의 평행투상

- 투상: void glMatrixMode(GL_PROJECTION);

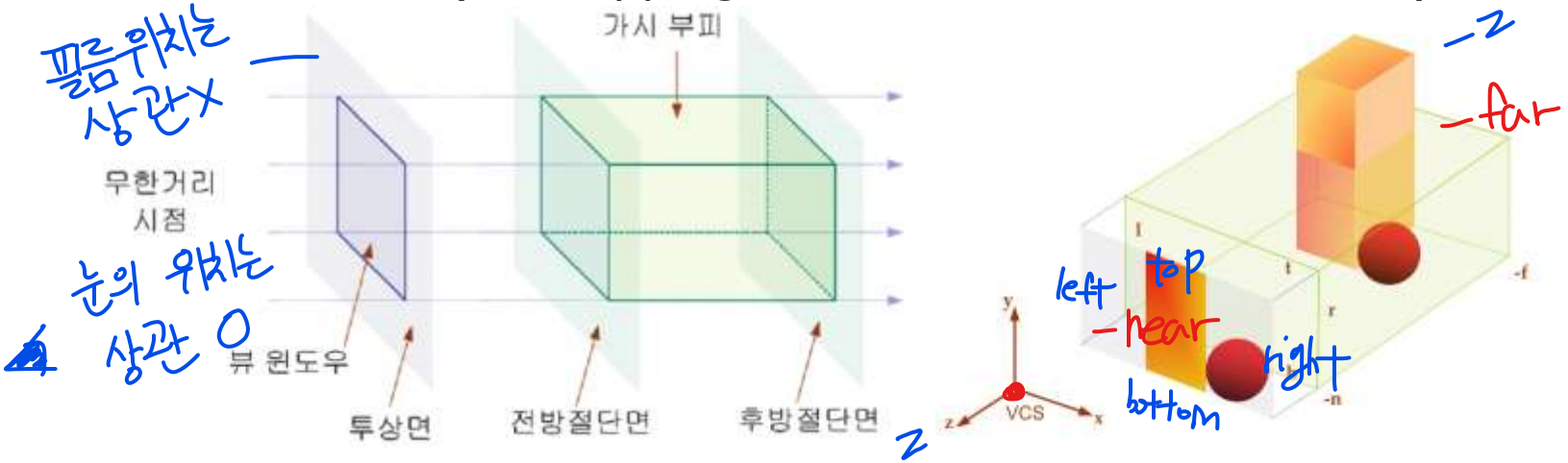
$$P' = M_{parallel} \cdot P$$

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -d \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ -d \\ 1 \end{pmatrix}$$

- 기본 평행투상
  - 모델 좌표, 전역 좌표, 시점 좌표 순서로 변환된 상태
  - P, P'은 시점 좌표계 기준의 좌표. 거리 d에 무관하게 동일한 모습
  - 특이변환(Singular Transformation): 역변환이 없는 변환
    - (x, y, z, 1)에서 (x, y)만 읽어내면 그것이 투상된 2차원 좌표
    - 나중에 깊이 정보를 활용하기 위해서 지엘은 이러한 변환을 가하지는 않으므로, 개념적 설명임
    - 투상결과 여전히 3차원 좌표가 유지.
  - 눈이 원점에 있다고 가정하고 –z 방향으로 관찰하고 있다고 생각한다.

가시부피에 의한 평행투상

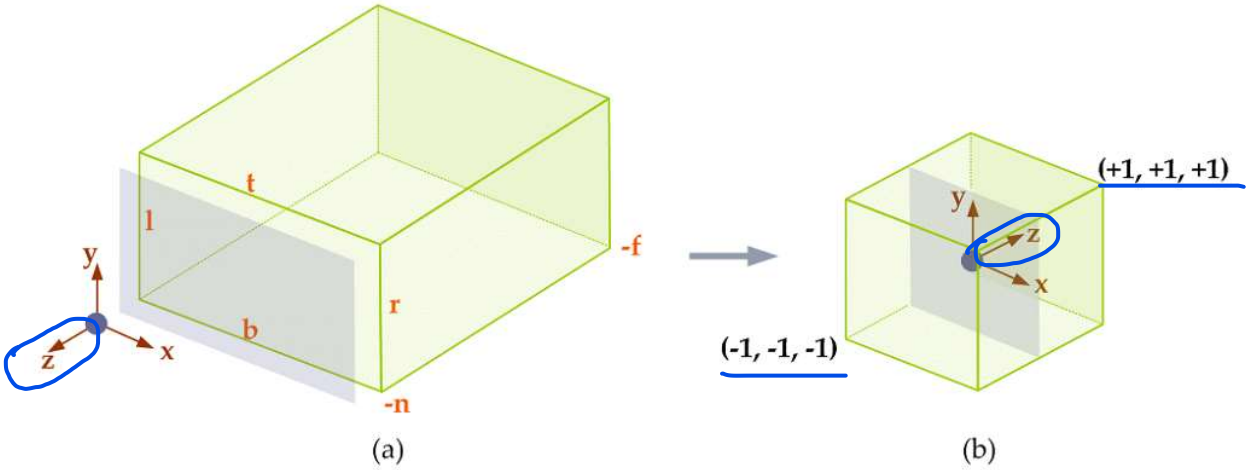
- 장면의 범위를 지정할 필요성: 가시부피(View Volume)
  - 전방 절단면(Near Clipping Plane, Near Plane, Front Plane)
  - 후방 절단면(Far Clipping Plane, Far Plane, Back Plane)



- void glOrtho(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble near, GLdouble far);
- 단, left, right, near, far는 카메라 기준 좌표계의 좌표이며
- near, far는 역사적 이유로 [-near, -far] 까지의 범위이다.
- 관찰 방향 (–z)을 고려하면, 이해 가능

정규화 가시부피

- 정규화 가시부피(CVV: Canonical View Volume)
  - 가로, 세로, 높이가 2인 정육면체로 투상
  - 정규화 변환(Normalization Transformation)
  - 참고) 그림에서 z축 방향이 변환 후 반대가 됨 (계산의 편리)
- 이유
  - 평행투상, 원근투상을 동일한 모습의 정규화 가시부피로 변형
    - 동일 파이프라인 사용
  - 정육면체를 기준으로 하면 연산이 간단함.
  - 다양한 해상도의 화면 좌표계로 변환하기가 간단함.



기초적 선형 변환

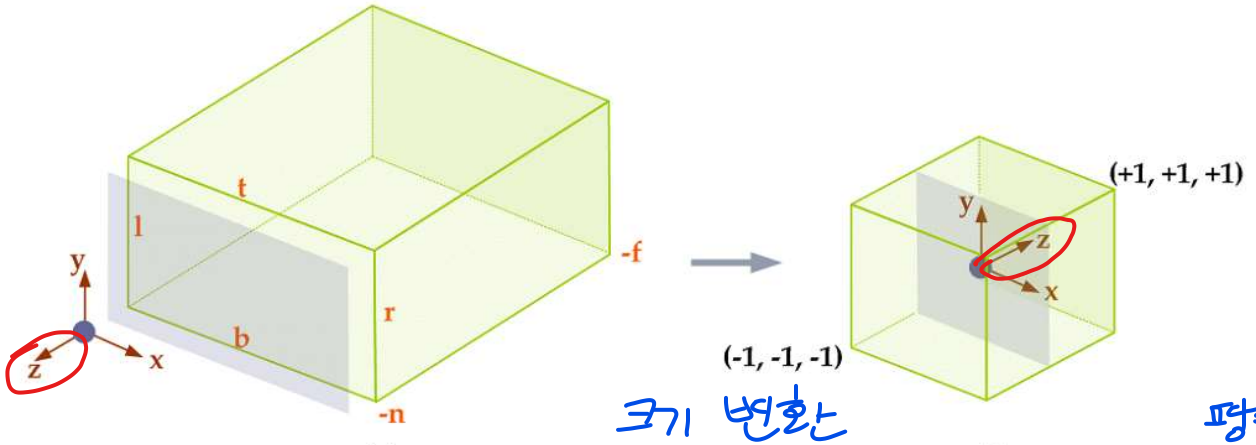
- [2, 8] 범위의 선분이 [-1,1] 구간으로 변형된다면,
- 범위내의 한 점 x는, 어떤 점으로 움직이는가?

- 중심점 (2+8)/2 = 5 가 원점으로 이동하는 평행이동
- y = x – 5
- [2, 8] 범위의 점은 [-3, +3]범위로 변경됨

- 구간의 폭 6(=8-2)이 2로 변경되도록 축소변환
- z = y * (2/6)

- 행렬 공식으로 변형,
- x축만 구간 [a, b]가 [-1, 1]로 되도록 (y, z는 변화 없음)
- P' = S(2/(b-a), 1, 1) * T(-(a+b)/2, 0, 0) * P

정규화 가시부피 변환



크기 변환

평행 이동

$$N = R \cdot S \cdot T$$

③ $$R = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

② $$S = \begin{pmatrix} \frac{2}{r-l} & 0 & 0 & 0 \\ 0 & \frac{2}{t-b} & 0 & 0 \\ 0 & 0 & \frac{2}{f-n} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

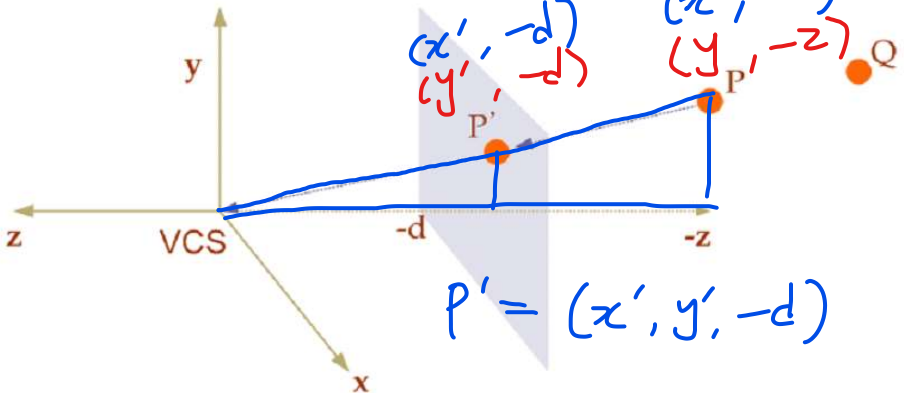
① $$T = \begin{pmatrix} 1 & 0 & 0 & -\frac{r+l}{2} \\ 0 & 1 & 0 & -\frac{t+b}{2} \\ 0 & 0 & 1 & \frac{f+n}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} l \\ b \\ -n \\ 1 \end{pmatrix}$$

- 예제) 점(l,b,-n,1), (r,t,-f,1)의 변환 후 좌표를 구하시오.
- 물체에 대한 이동, 크기조절, 반사변환으로 간주
- Reflection: 정규화 가시부피는 왼손좌표계 (near=-1, far=+1)
- 결과적인 좌표계 = 절단 좌표계(CCS: Clip Coordinate System)

실제로 사용하는건 이것

지엘의 원근투상

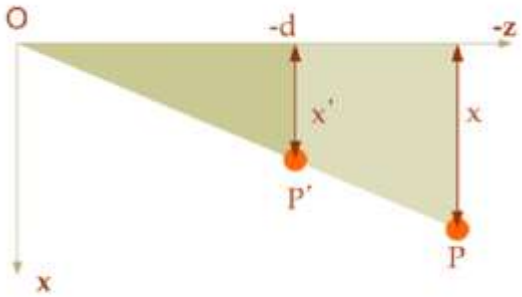
- 기본 원근투상
  - x' : (-d) = x : (-z), x' = ?
  - y' : (-d) = y : (-z), y' = ?



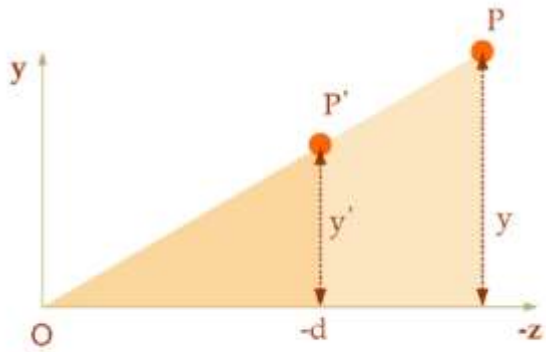
$$P' = \begin{pmatrix} x' \\ y' \\ -d \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{x}{z/d} \\ \frac{y}{z/d} \\ -d \\ 1 \end{pmatrix}$$

$$P' = \begin{pmatrix} x' \\ y' \\ -d \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ -z \\ z/d \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 1/d & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$



(a)



(b)

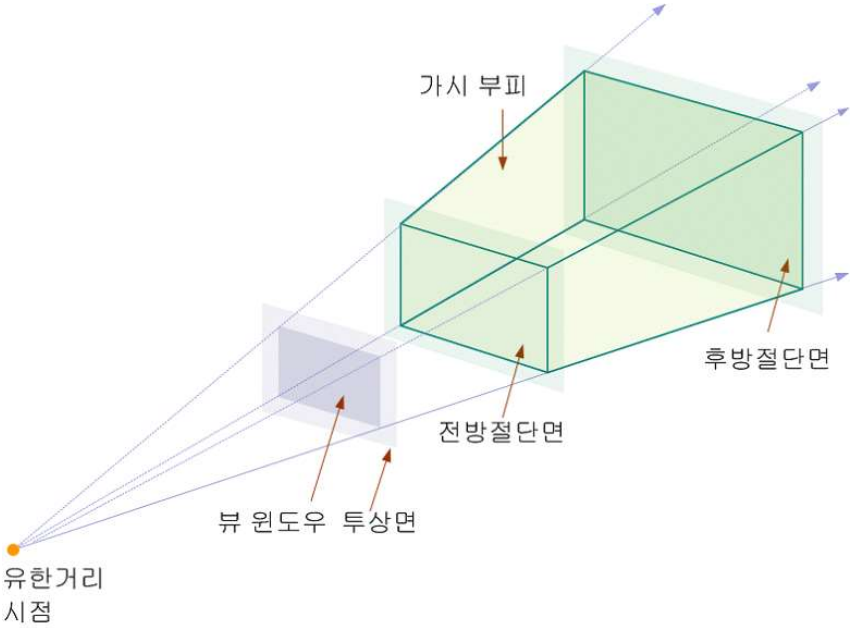
원근변환

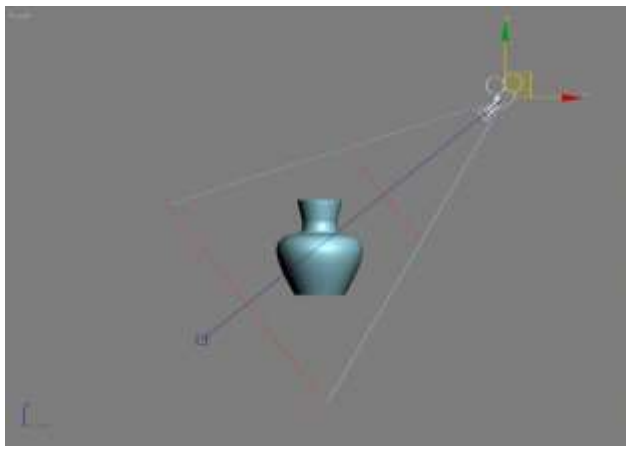
- 원근분할(Perspective Division, Homogenization) 원래대로 돌아가기?
  - 동차좌표의 마지막 요소로 이전 요소를 나누는 작업
  - 실제 삼차원 공간의 x, y, z 좌표를 계산
  - 원근 분할은 늦게 수행됨. 절단(8장)을 동차좌표에서 수행할 예정.
  - 절단 작업 이후 원근분할 수행됨
  - $\begin{pmatrix} x \\ y \\ -z \\ z/d \end{pmatrix} \rightarrow \begin{pmatrix} \frac{x}{z/d} \\ \frac{y}{z/d} \\ -d \\ 1 \end{pmatrix}$
- 원근변환
  - 어파인 변환이 아님: 마지막 행이 (0, 0, 0, 1)이 아님
  - 3차원 좌표관점: x' = x/(z/d): 비선형 변환
  - 4차원 동차좌표 관점: 선형변환

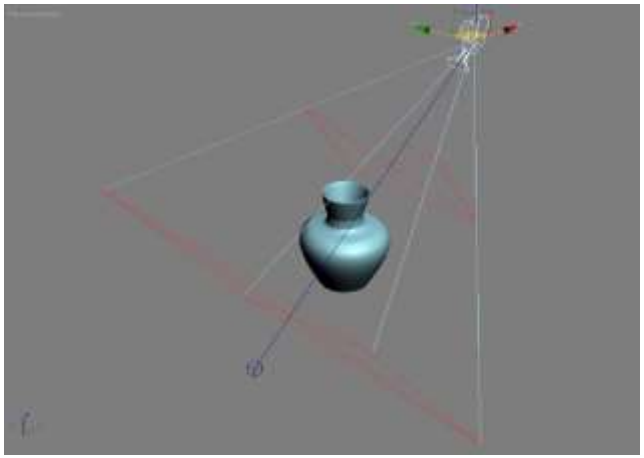
$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 1/d & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \text{Mperspective} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

정규화 가시부피에 의한 원근투상

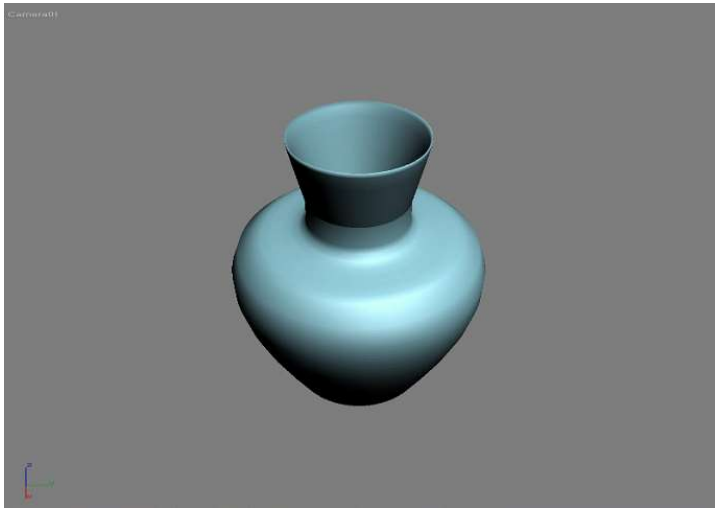
- 절단 사각뿔(Frustum) = 절두체

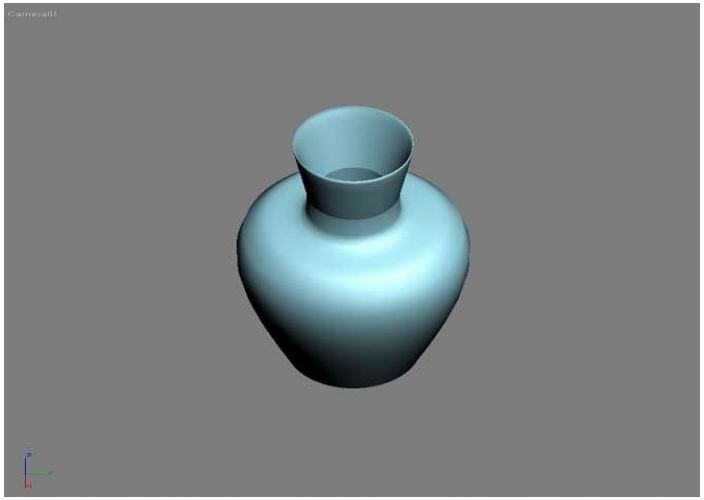






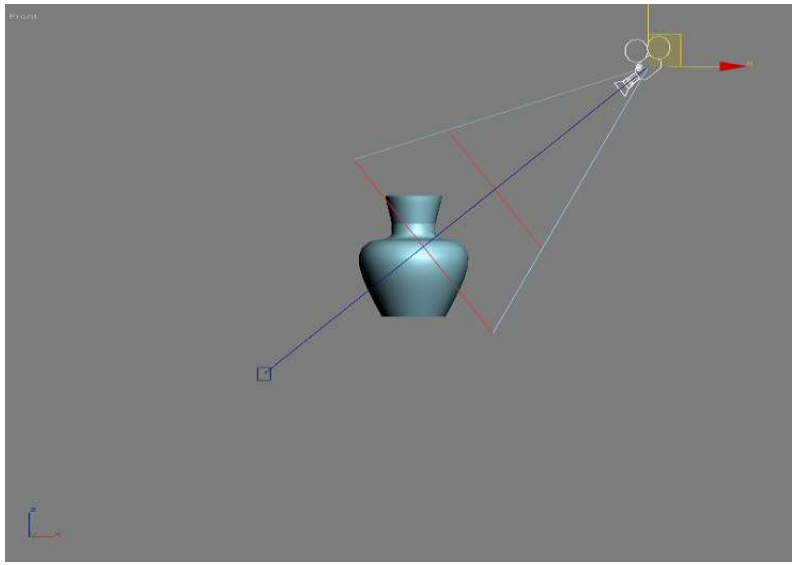
원근투상과 평행투상

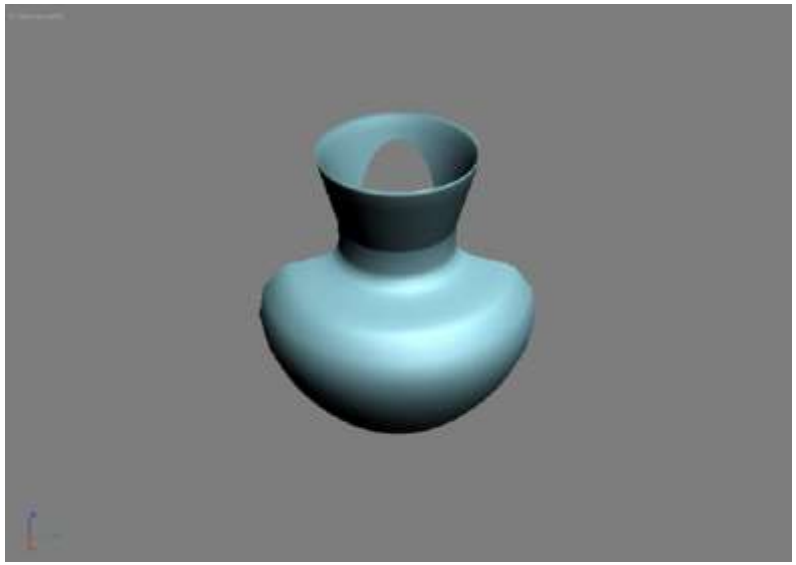




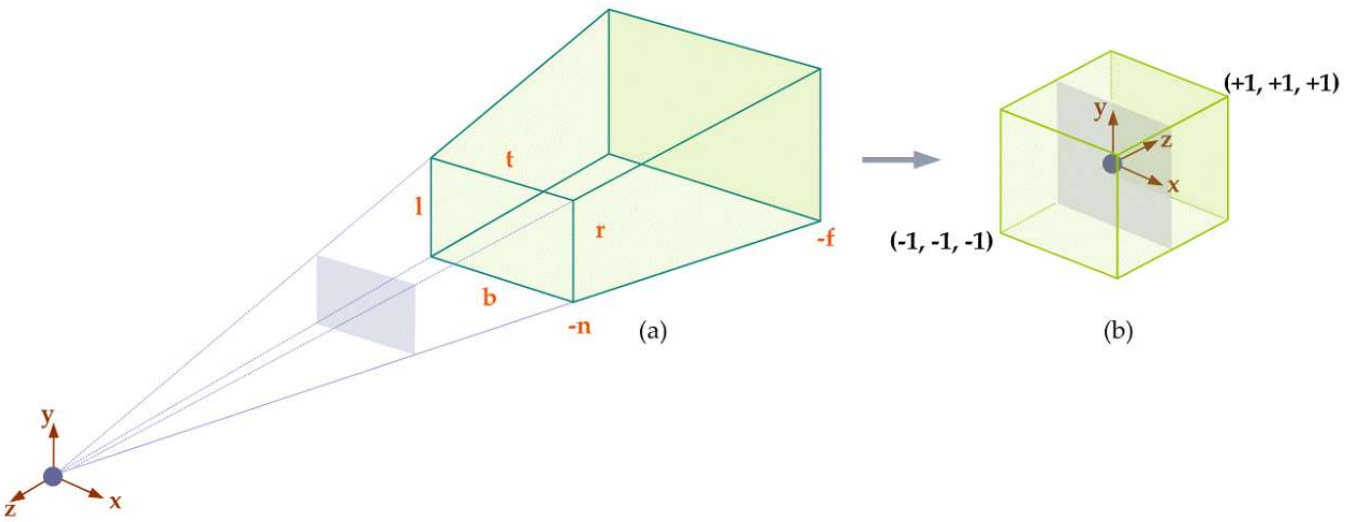
가시부피

- 가시부피 설정에 의한 절단



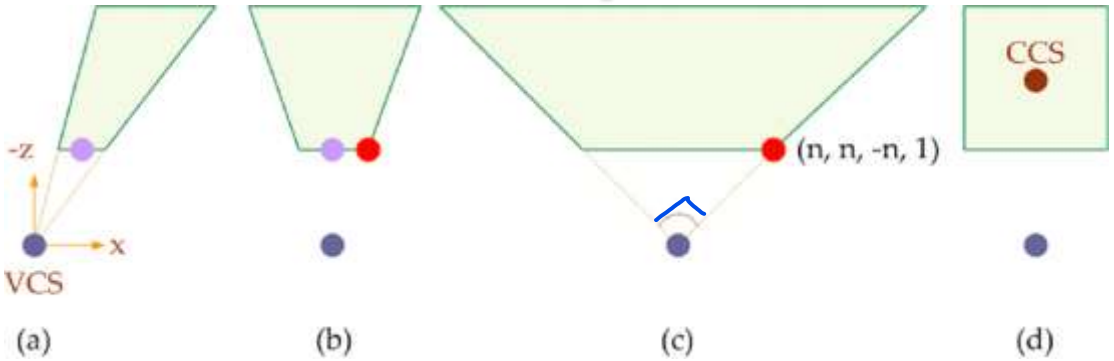


원근투상의 정규화 가시부피



원근투상의 정규화 가시부피

- 일반적 형태의 가시부피: void glFrustum(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble near, GLdouble far);



$$N = T \cdot S \cdot Sh$$

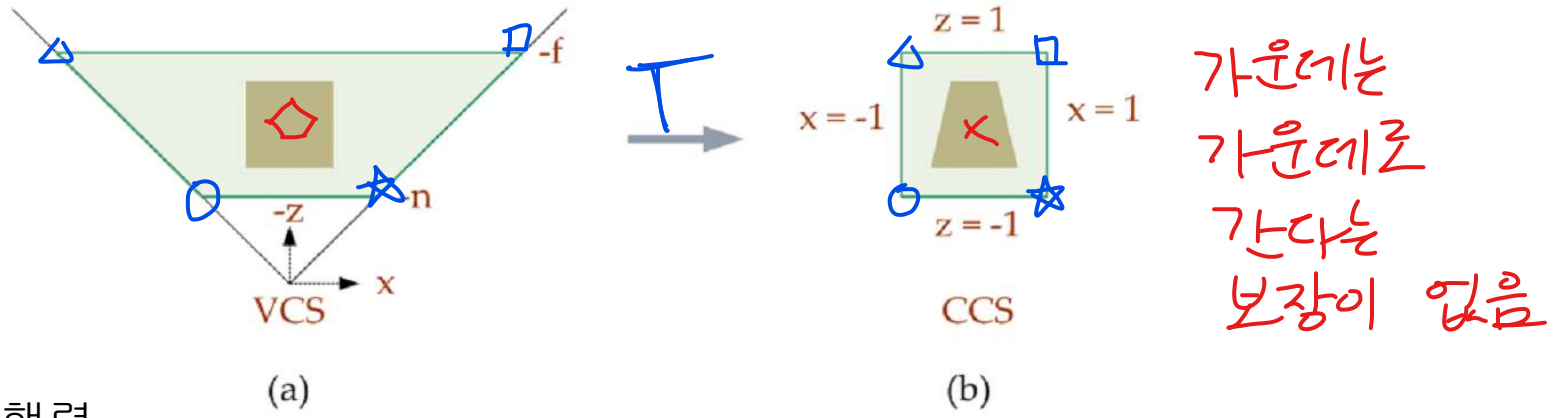
$$= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} \frac{2n}{r-l} & 0 & 0 & 0 \\ 0 & \frac{2n}{t-b} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & \frac{r+l}{2n} & 0 \\ 0 & 1 & \frac{t+b}{2n} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

실제로 사용하는 것은 이것

원근투상의 정규화 가시부피

- 전방 절단면에 비해 후방 절단면 면적이 줄어든다.
  - 멀리 있는 것이 작게 보여야 함.



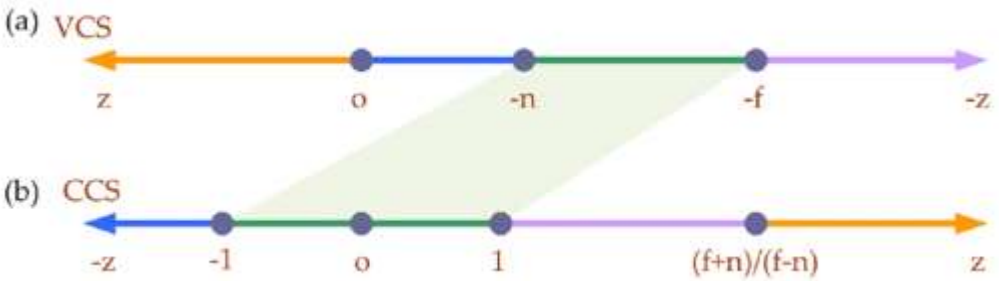
가운데는 가운데로 간다는 보장이 없음

- T 행렬
  - Z 값에 영향을 미침: 원래의 물체 정점의 깊이 z와 정규화 변환 후의 물체 정점의 깊이 z'의 관계
  - 예제) 점 (0,0,-n,1)과 (0,0,-f,1)의 변환 후 위치를 계산하라

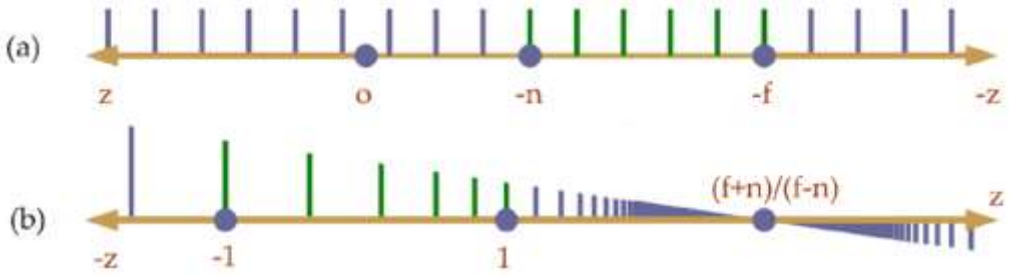
$$P' = \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

행렬을 외우지 말고 연관관계 알기

시점 좌표계에서 절단 좌표계로



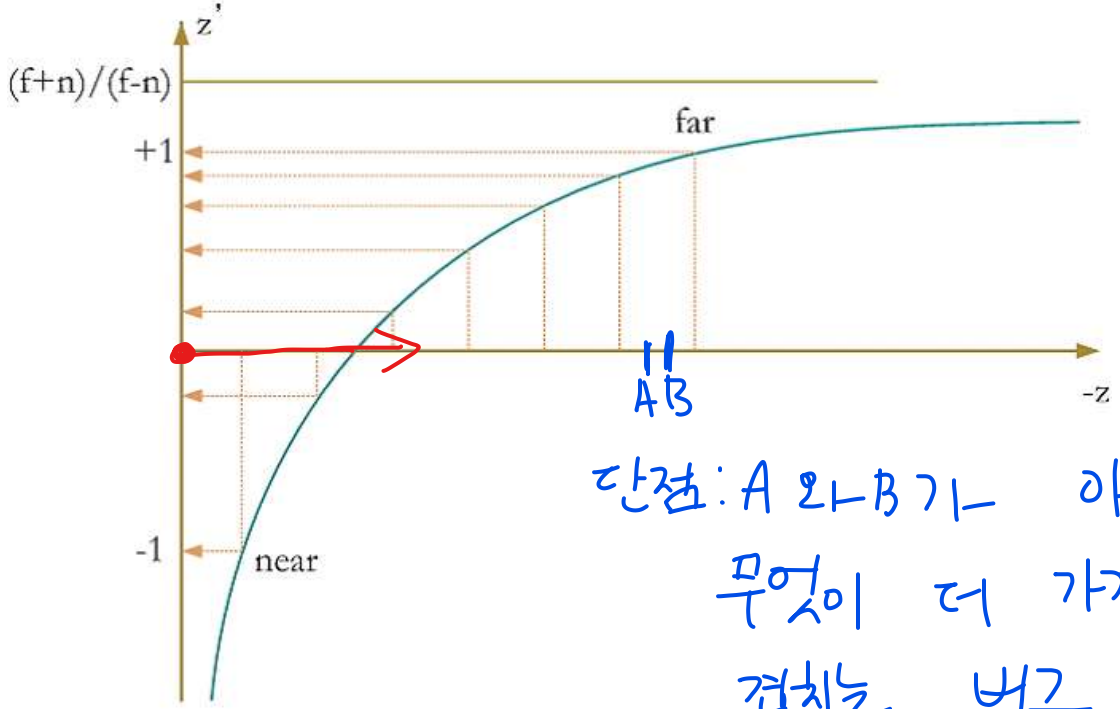
- [-n, -f]의 범위를 [-1,1]로 옮기는 과정에서 분수함수가 도입
- 분수함수의 side effect로, 관심 외 구간(주황색)의 좌표가 크게 변형됨



- 높이: 멀어질수록 전봇대 높이가 낮아짐(원근변환)

시점 좌표계에서 절단 좌표계로

- 간격:더욱 촘촘해 짐에 유의(비선형 변환)



$$z' = \frac{f+n}{f-n} - \frac{2fn}{f-n}\frac{1}{(-z)}$$

단점: A와 B가 아주 멀면 (0.99999999, 1)
무엇이 더 가까운지 몰라서,
겹치는 버그 발생

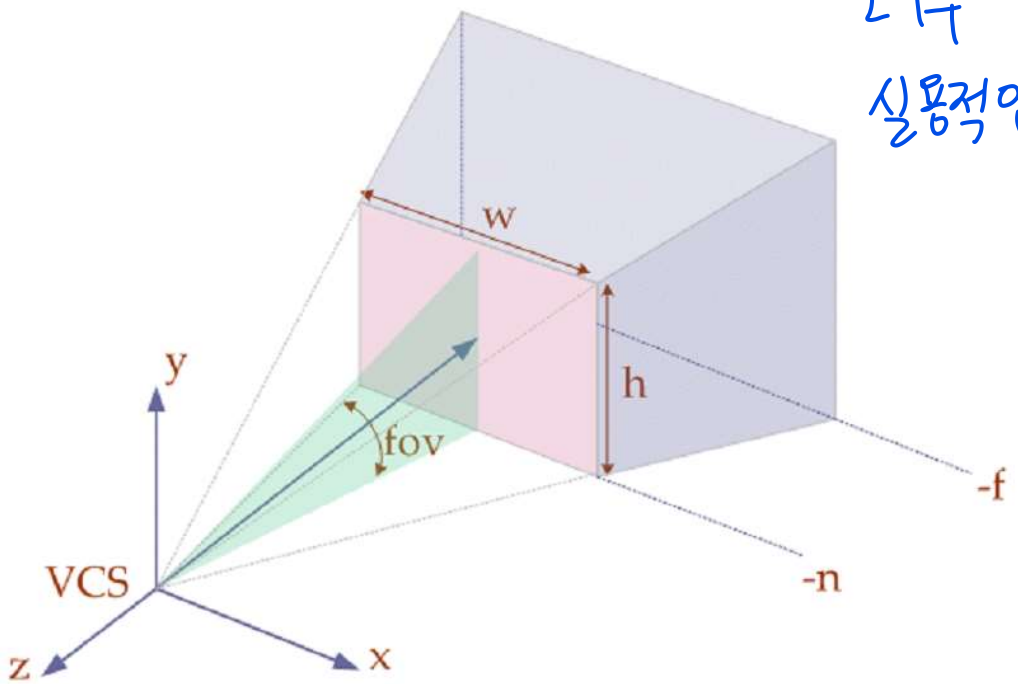
- 전방 절단면: 시점에서 멀리, 물체에 최대한 근접시켜 설정
  - 물체간격이 상대적으로 보존, 지-버퍼 처리에 유리

이쪽에 있는 것을 오른쪽으로 옮기면 보완 가능

대칭적 원근투상

- void gluPerspective(GLdouble fov, GLdouble aspect, GLdouble near, GLdouble far);

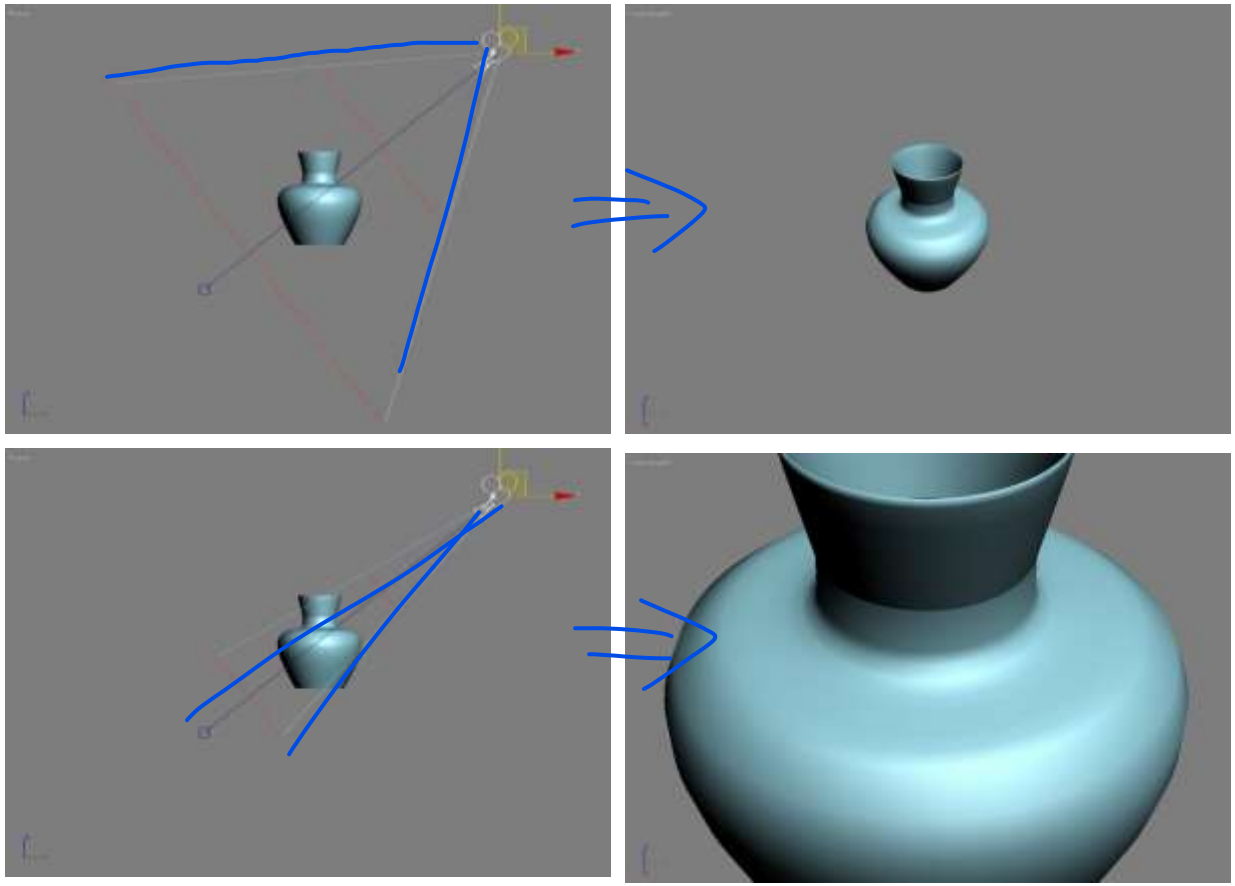
glFrustum은 가능이 너무 많아서, 더 가볍고 실용적인 이것을 쓸 것이다.



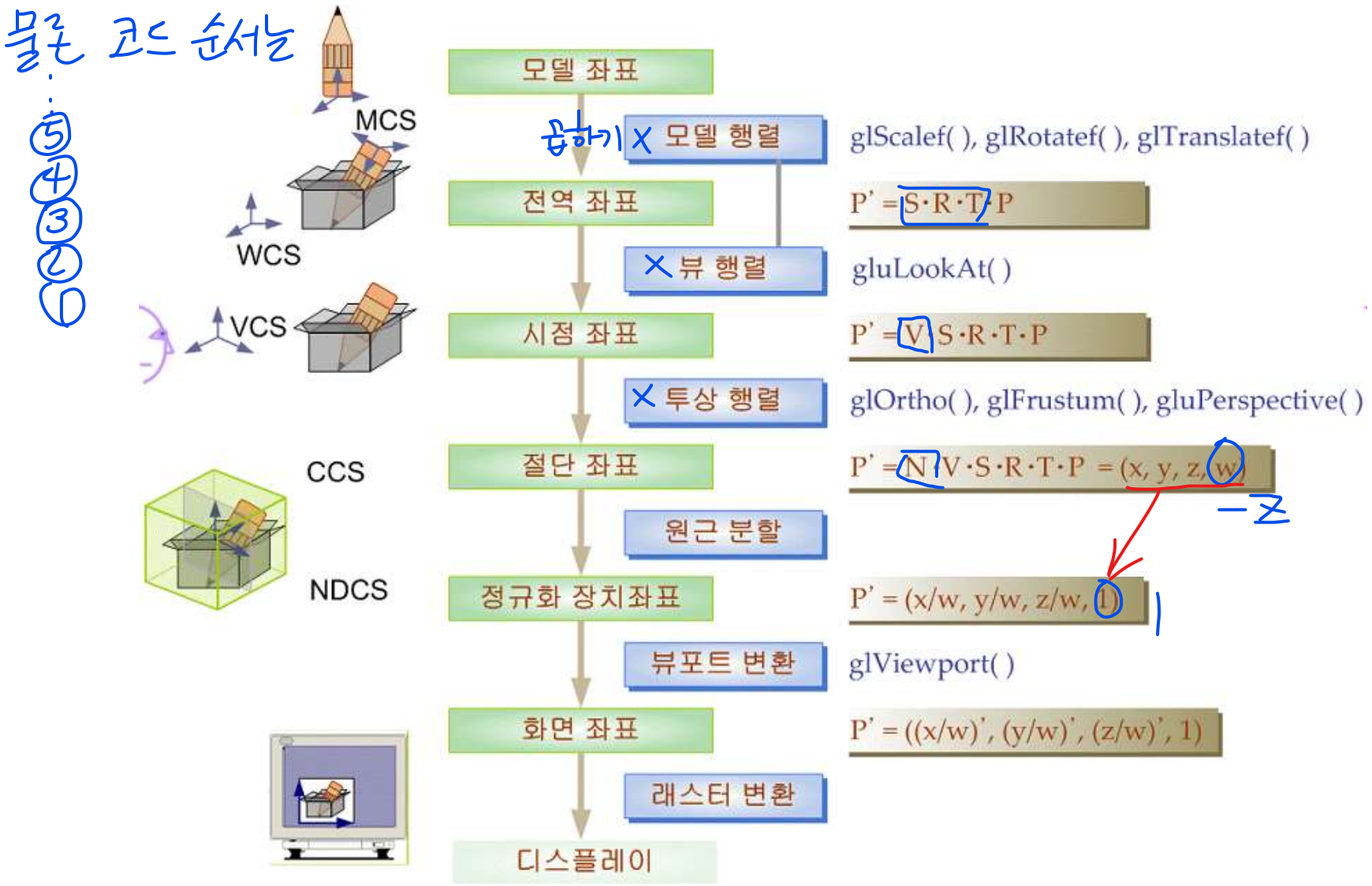
- 시야(FOV: Field of View):상하 y축 방향의 시야각(0 -180도)
- X축 방향의 시야는 종횡비(Aspect Ratio)에 의해 결정됨(폭 / 높이)

시야각과 카메라 렌즈

- 초점 거리 50mm 기준
  - 광각렌즈(Wide Angle Lens) : 50보다 작음
  - 망원렌즈(Telescope Lens) : 50보다 큼
  - Ex. 20mm = 85도 시야각, 85mm = 24도 시야각

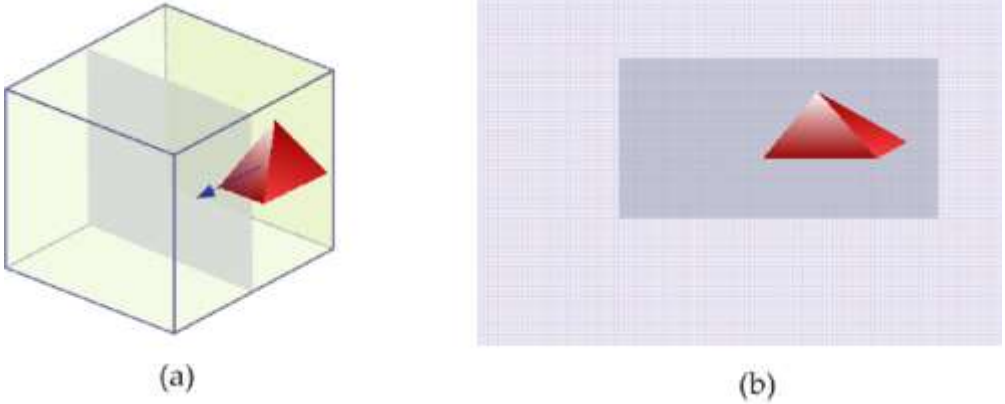


지엘 파이프라인


모델 좌표
× 모델 행렬
glScalef( ), glRotatef( ), glTranslatef( )
전역 좌표
P' = S·R·T·P
× 뷰 행렬
gluLookAt( )
시점 좌표
P' = V·S·R·T·P
× 투상 행렬
glOrtho( ), glFrustum( ), gluPerspective( )
절단 좌표
P' = N·V·S·R·T·P = (x, y, z, w)
원근 분할
정규화 장치좌표
P' = (x/w, y/w, z/w, 1)
뷰포트 변환
glViewport( )
화면 좌표
P' = ((x/w)', (y/w)', (z/w)', 1)
래스터 변환
디스플레이
MCS
WCS
VCS
CCS
NDCS

뷰포트 변환

- 정규화 장치좌표계(NDCS: Normalized Device Coordinate System)
  - 절단 이후 원근분할에 의해 물체 정점을 3차원 좌표로 변환한 것
  - (x', y', z', 1) = (x/w, y/w, z/w, 1)
- 뷰포트 변환(Viewport Transformation)
  - 정규화 장치좌표계에서 화면 좌표계로 가는 작업
  - 화면 좌표계(SCS: Screen Coordinate System), 뷰포트 좌표계(Viewport Coordinate System), 윈도우 좌표계(Window Coordinate System)



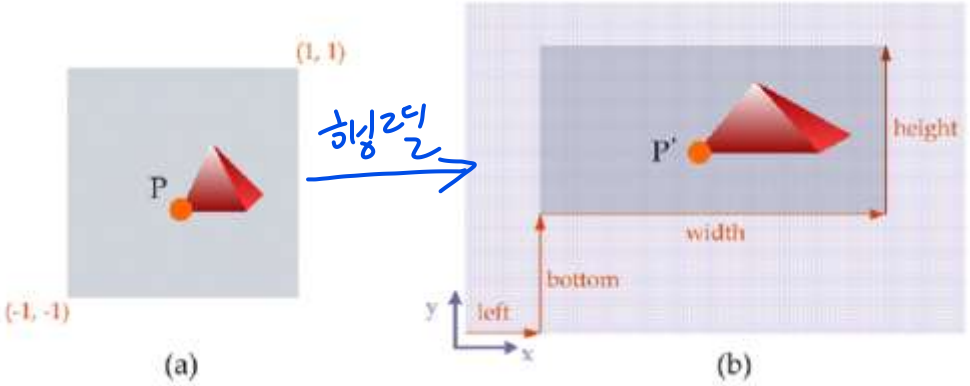
(a) (b)

뷰포트 설정

- void glViewport(GLint left, GLint bottom, GLsizei width, GLsizei height);

$$x' = \frac{x-(-1.0)}{(1.0)-(-1.0)} \times width + left$$

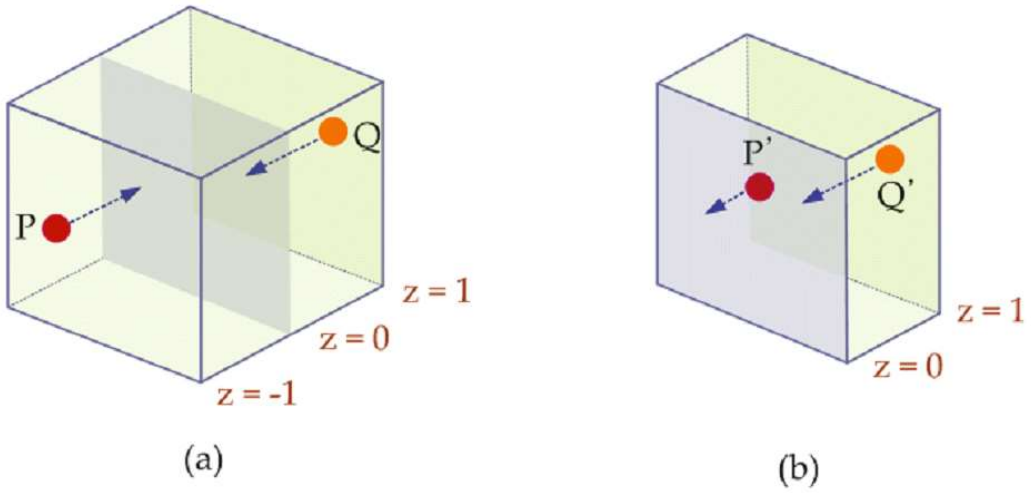
$$y' = \frac{y-(-1.0)}{(1.0)-(-1.0)} \times height + bottom$$



$$P' = \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{width}{2} & 0 & 0 & left + \frac{width}{2} \\ 0 & \frac{height}{2} & 0 & bottom + \frac{height}{2} \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

- P' = T(left+width/2, bottom+height/2)
  * S(width/2, height/2, 1)
  * P

z 값의 재 정규화



- 정규화 가시부피에서의 z 값의 범위인 [-1, +1] 사이를 [0, 1] 사이로 사상
  - 정점 사이의 상대적인 깊이는 유지
  - 정규화 가시부피를 원점을 중심으로 해서 z축 방향으로 1/2 만큼 크기조절 변환을 가한 후, z 방향으로 1/2 만큼 평행 이동.
  - P' = T(0, 0, 0.5) * S(1,1, 0.5) * P

가시부피와 뷰포트