OpenGL 8

가상현실론

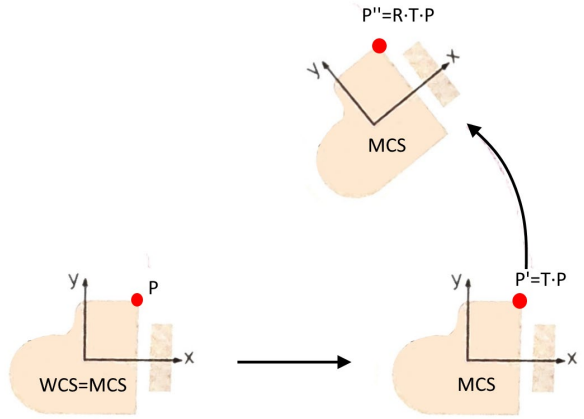
2021/03/22

목차

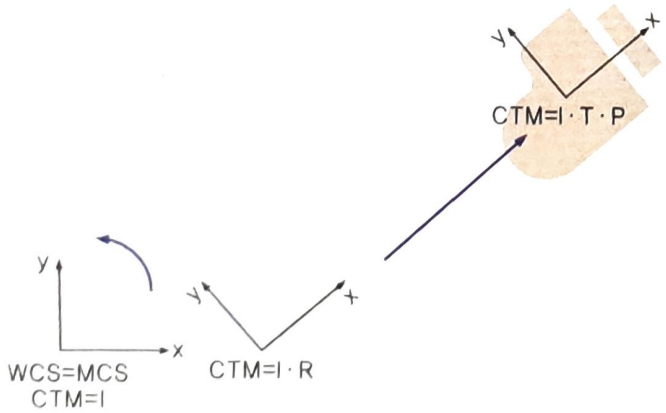
- 복습
  - 복합변환에 의한 모델링
  - 행렬 스택 (Matrix Stack)
  - GL의 시점변환
- 투상
- GL의 투상 변환

복습

- 복합변환에 의한 모델링

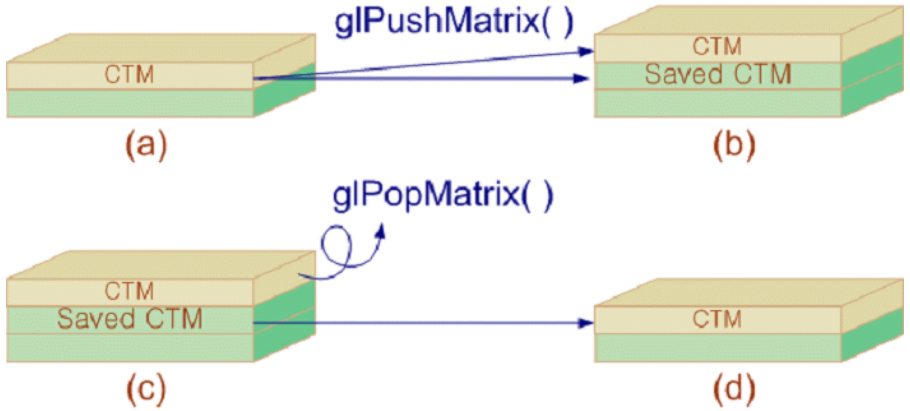


- 전역좌표계 기준으로 물체 이동
- Pre-multiplication
- 물체 자체를 이동시킨 후 회전
  - P’=T·P
  - P’’=R·P’=R·T·P



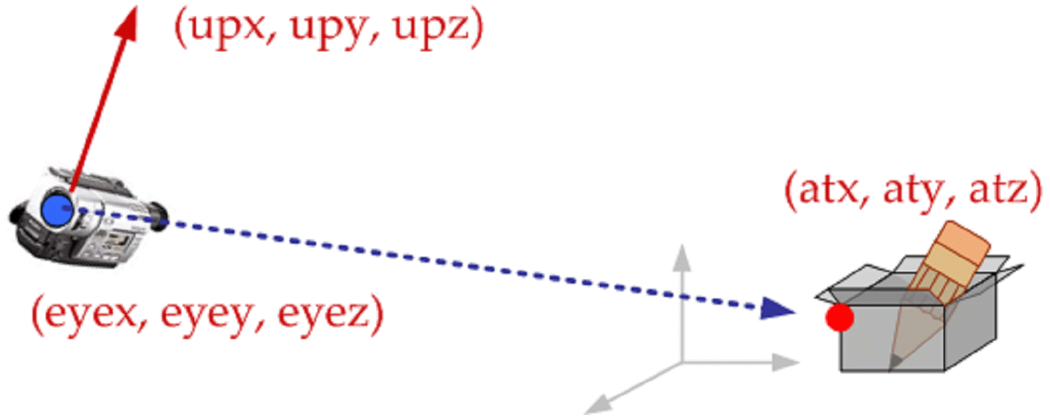
- 모델 좌표계 자체를 이동
- Post-mulitiplication
  - CTM=I·R
  - CTM= I· R·T
  - P’’=I· R·T·P= R·T·P

복습



- 행렬스택
  - GL은 행렬 스택을 제공. 이를 사용하여 좌표계가 변화한 과정을 파악할 수 있음.
  - 스택은 LIFO (Last-In-First-Out) 방식의 자료 구조
  - 모델 뷰 행렬 스택의 깊이는 최소 32개, 투상 행렬 스택의 깊이는 최소 2개. 행렬 스택을 조작하기 위한 함수는
  - void glPushMatrix();
    - 현 변환행렬(CTM)을 스택 top에 삽입하여 그것이 새로운 현 변환행렬이 되게 함.
  - void glPopMatrix();
    - 스택 top의 행렬을 삭제함으로써 바로 아래에 있던 이전의 현 변환 행렬을 현 변환 행렬로 복원함.

복습

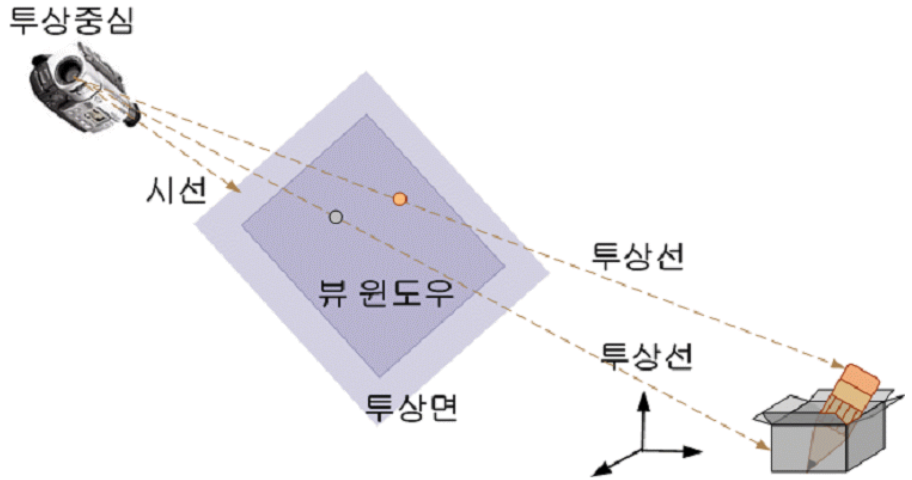


- GL의 시점 변환
  - GL의 시점좌표계는 다음 3가지 요소에 의해서 정의
    - 카메라 위치
    - 카메라가 바라보는 점, 즉 초점의 위치
    - 카메라 기울기(orientation)
  - void gluLookAt(GLdouble eyex, GLdouble eyey, GLdouble eyez, GLdouble atx, GLdouble aty, GLdouble atz, GLdouble upx, GLdouble upy, GLdouble upz);
    - GL의 시점 좌표계 설정
    - (eyex, eyey, eyez)에 있는 카메라가 (atx, aty, atz)에 있는 초점을 바라보게 하라는 것.
    - (upx, upy, upz)는 카메라의 상향벡터.
    - 9개의 파라미터를 통해 카메라 위치, 초점, 기울기 정보를 도출함

투상 변환과 뷰 포트 변환

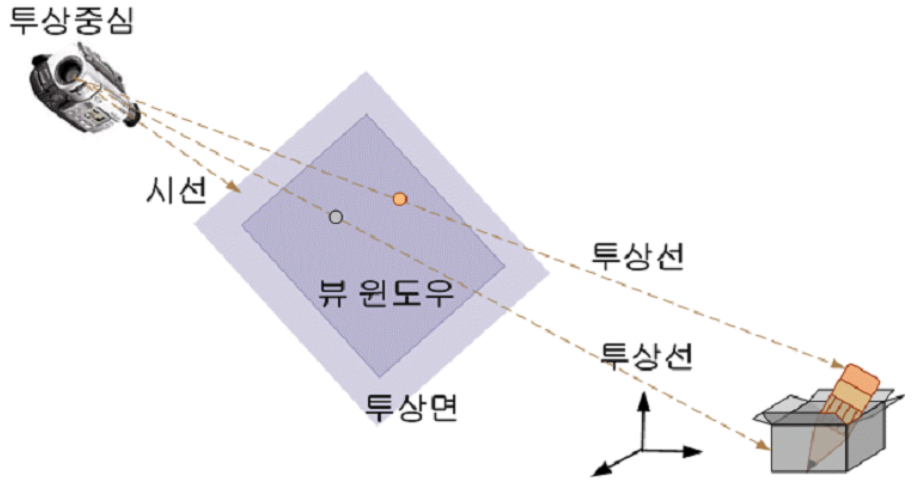
OpenGL

투상 (Projection)



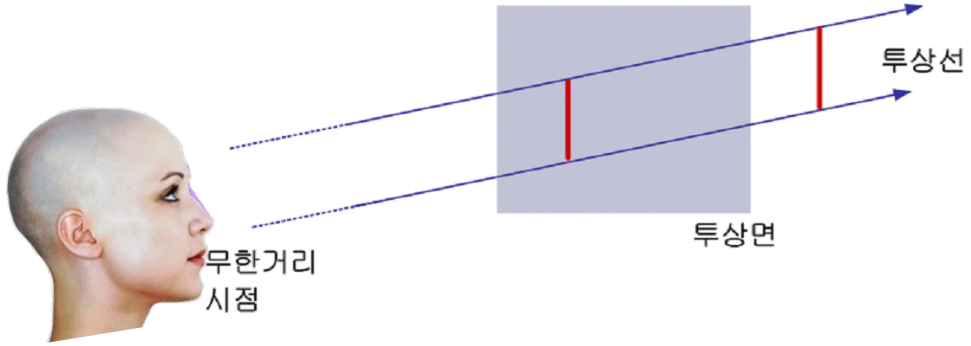
- 투상 (Projection): 3차원 물체를 2차원 화면으로 사상하기 위한 작업. 모델 좌표계, 전역 좌표계, 시점 좌표계를 순차적으로 거친 다각형 정점 좌표를 2차원 투상면(View Plane, Projection Plane)으로 사상시키는 과정
- 관찰자 위치(View Point, Eye Position, Camera Position): 관찰자(카메라)의 눈의 위치 (시점좌표)
- 투상 중심(COP: Center of Projection): 시점좌표계 원점
- 투상선 (Projectors): 투상 중심과 물체를 연결한 선

투상 (Projection)



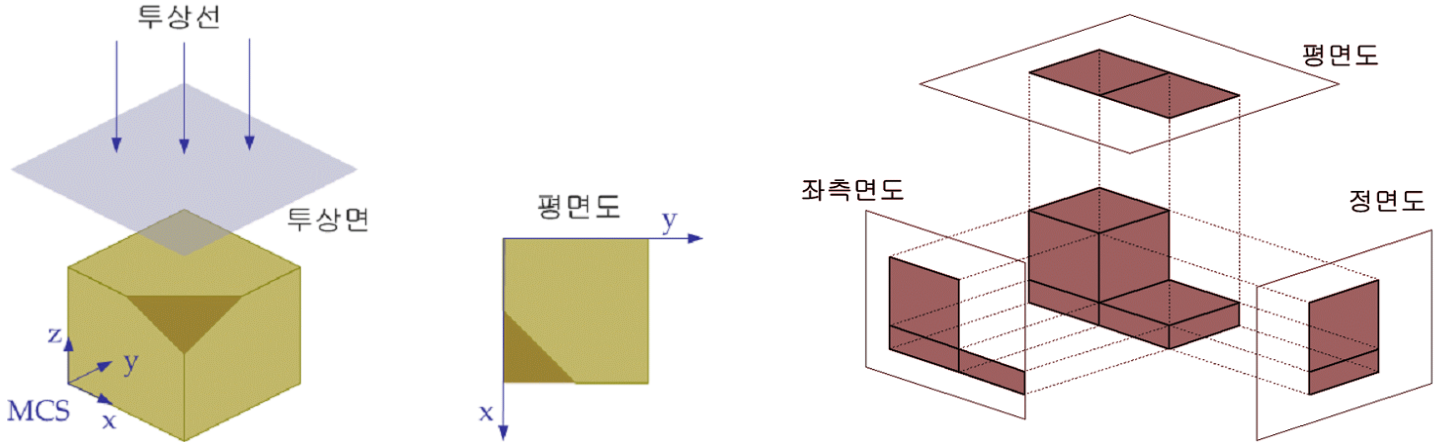
- 시선 (Line of Sight): 카메라가 바라보는 방향. 카메라 설정 방법에 따라서 시선은 전역좌표계 원점을 향하기도 하고, 임의 위치의 초점을 향하기도 함
- 투상면 (Projection Plane, View Plane): 물체 영상이 맺히는 화면 (영화관 스크린과 같은 역할). 일반적으로 시선에 수직으로 놓인다.
- 투상중심에서 물체 정점을 향한 투상선이 투상면과 만나는 곳에 해당 정점이 투상(Projection)됨.
- 투상은 평행 투상과 원근 투상으로 구분

평행 투상 (Parallel Projection)



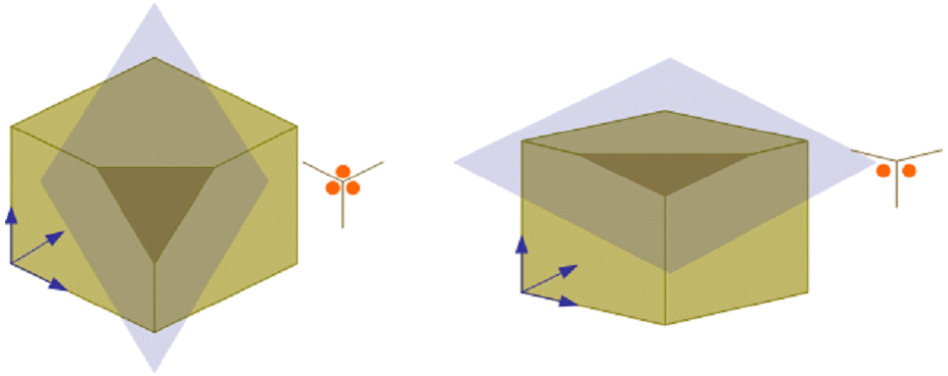
- 평행 투상(Parallel Projection): 시점이 물체로부터 무한대의 거리에 있다고 간주하여 투상선을 나란히 가져가는 투상법.
- 태양이 지구로부터 매우 먼 거리에 위치해있기 때문에 태양에서 출발한 빛은 지구에 도착할 때 평행광선이라는 것에 비유할 수 있음.
- 평행 투상에서 원래 물체의 평행선은 투상 후에도 평행.
- 평행투상은 정사 투상, 축측 투상, 경사 투상으로 구분됨

평행 투상-정사투상



- 정사투상(Orthographic Projection): 평면도(상하, Top/Bottom View), 입면도 (전후, Front/Rear View), 측면도 (좌우, Left/Right View) 등을 말함.
- 모델 좌표계의 주축 (Principal Axes)인 x, y, z에 의해 형성되는 x-y, y-z, z-x를 주 평면(Principal Plane)이라고 하면, 정사 투상의 투상면은 주 평면 중 하나와 놓이게 됨.
- 정사 투상에서 투상선은 투상면과 직교함.
- Projection된 상이 원래 물체의 길이를 정확히 보존하기 때문에 정사 투상은 정확성을 요하는 공학 도면에 주로 사용.
- 투상선이 반드시 투상면과 직교해야 하므로 정사 투상에서 시점 위치가 제한됨.

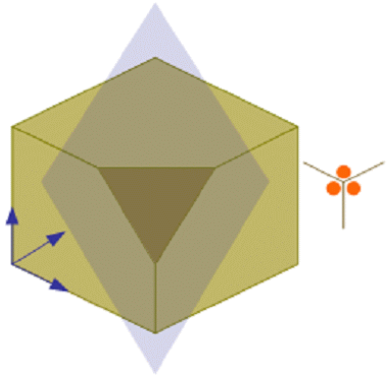
평행투상-축측투상



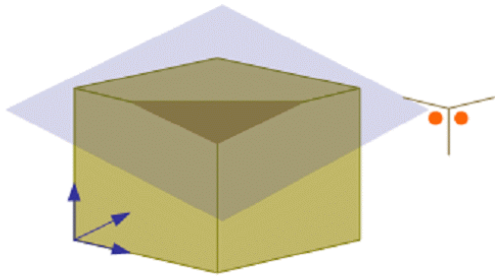
등각투상 양각투상

- 축측 투상(Axometic Projection): 한꺼번에 여러 면(주평면)을 보여줌
- 투상선이 투상면과 직교한다는 면에서 정사 투상과 같지만 투상면이 반드시 주 평면들과 나란하지 않음
- 투상면이 임의의 위치에 놓일 때를 삼각(삼중형, Trimetric) 투상, 2개의 주 평면에 대해서 대칭적으로 놓일 때를 양각(이중형, Dimetric) 투상, 3개의 주 평면이 만나는 모서리에서 모든 평면에 대해 대칭적으로 놓일 때를 등각(동형, Isometric) 투상이라고 함

평행투상-축측투상



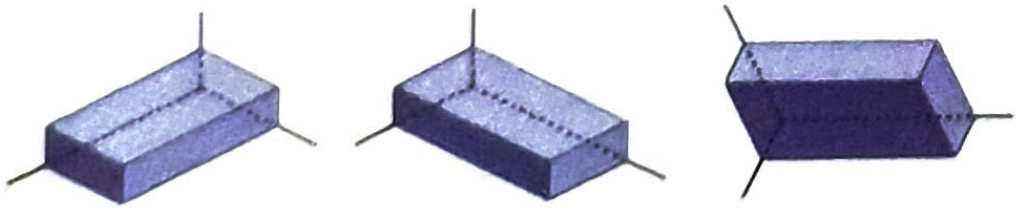
등각투상



양각투상

- 등각투상에서는 3개의 주평면이 대칭인 곳에 위치하는 것을 알 수 있음.
- 모서리 부근에서 3개의 주축이 서로 120도를 유지함. 따라서 정삼각형의 단면은 그대로 정삼각형으로 투상.
- 양각투상에서는 투상면이 2개의 주평면에 대칭인 곳에 놓임.
- 정삼각형의 단면이 이등변 삼각형으로 투상. 이 경우 교차하는 3개의 주축이 이루는 각도 중 2개만 동일해짐.

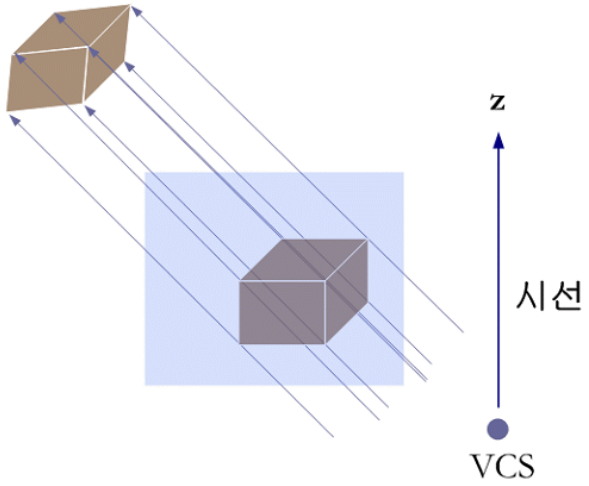
평행투상-축측투상



등각 투상의 대칭 축

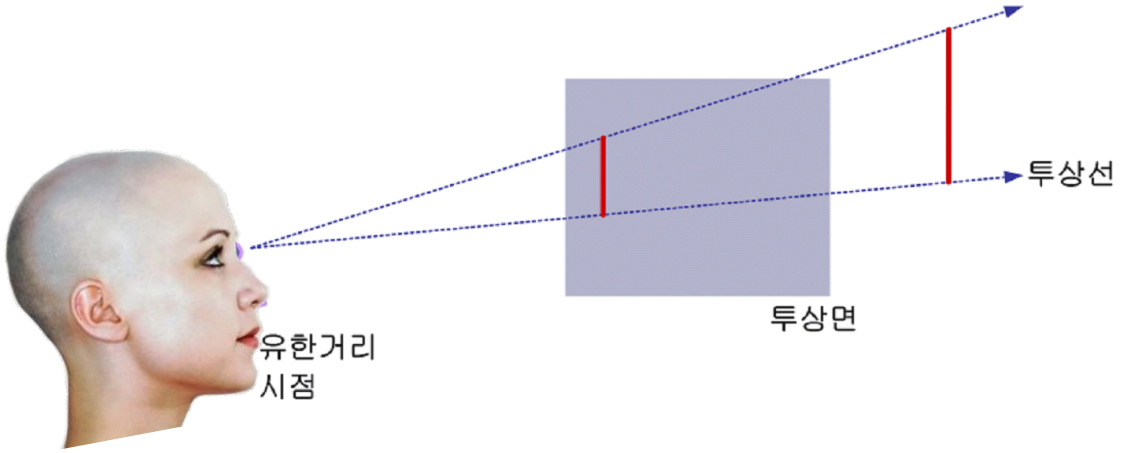
- 축측 투상의 결과는 일반적으로 물체의 실제 모습이 아님. 즉, 실제 길이가 보존되지 않으면 각 축의 방향으로 서로 다른 축소율(Foreshrotening Factor)를 보임.
- 세축 방향 모두 서로 다른 축소율을 보이는 것이 삼각 투상, 두 축에 대해서 동일한 축소율을 보이는 것이 양각 투상, 세 축 모두 동일한 축소율을 보이는 것이 등각 투상

평행투상-축측투상



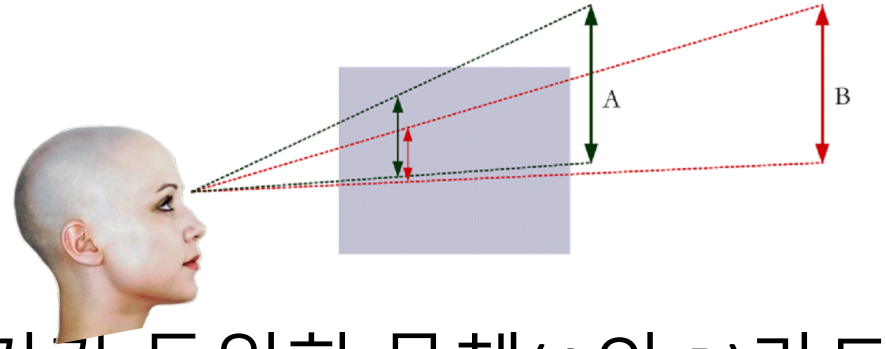
- 경사투상(Oblique Projection): 투상선이 나란하다는 점에서는 평행 투상에 속하지만 투상선이 투상면과 직교하지 않음
- 투상면은 시선에 수직이지만 투상면의 위치가 그림처럼 좌우로 비켜 서 있음.
- 고개를 돌리지 않고 눈동자만 돌려서 왼쪽 및 오른쪽 보는 것과 유사.

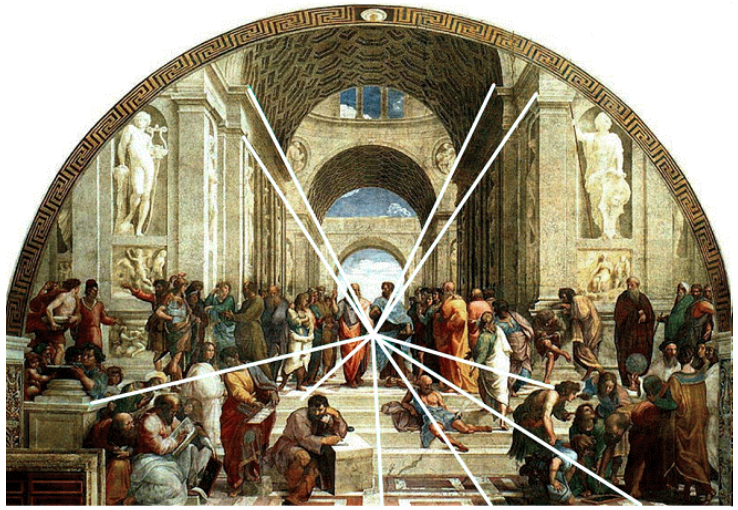
원근투상 (Perspective Projection)



- 원근투상 (Perspective Projection): 시점이 물체로부터 유한한 거리에 있다고 간주하여 모든 투상선이 시점에서 출발하여 방사선 모양으로 퍼져가는 방법.
- 카메라나 사람의 눈이 실제로 물체를 포착하는 방법

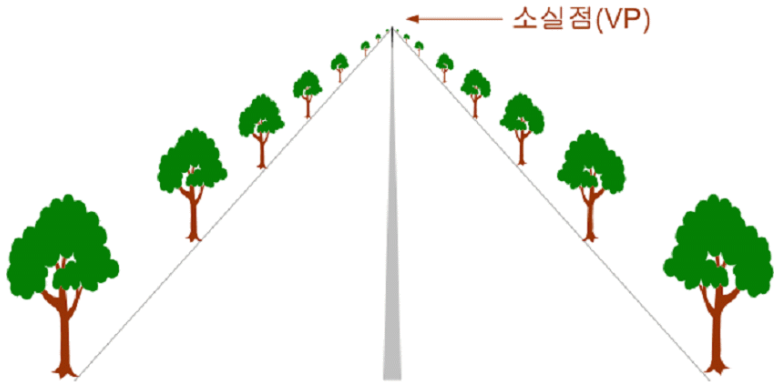
원근투상 (Perspective Projection)





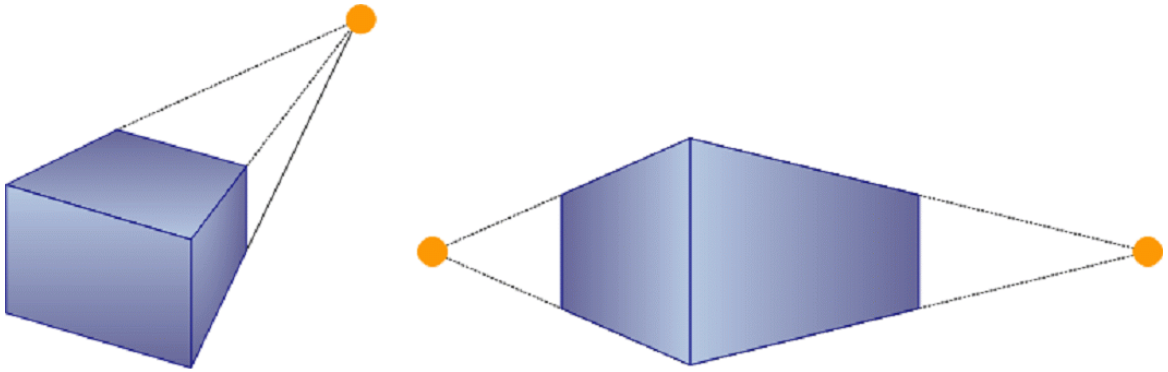
- 크기가 동일한 물체(A와 B)라도 시점으로부터 멀면(B) 작게 보이고 가까우면(A) 크게 보임.
- (중세~르네상스 시대 회화에서 이론이 정립)
- 이를 통해서 물체의 원근감(Depth Feeling)이 나타남.

원근투상 (Perspective Projection)



- 나무의 높이가 모두 동일하더라도 시점과의 거리가 멀 수록 작게 보임. 도로의 폭 역시 멀리 있는 것은 좁아보임.
- 곧게 뚫린 도로를 길 가운데에 서서 바라보면 멀리 한 점에서 만나듯이 원근투상의 결과 실제 평행한 선이 만나게 됨.
- 소실점(VP: Vanishing Point): 시점 높이에서 평행선이 만나는 점

원근투상 (Perspective Projection)



- 소실점 수에 따라 원근 투상을 ㅈ일점투상 (One-point Projection), 이점 투상 (Two-point Projection), 삼점투상(Three-point Projection) 등으로 나누기도 함.

GL의 투상변환

변환 종류	상태 변후 값(상수)	목적
모델 변환 시점 변환	GL_MODELVIEW	물체에 대한 기하변환 카메라 위치 및 방향에 대한 기하변환
투상 변환	GL_PROJECTION	촬영 방법 및 렌즈 선택에 대한 기하변환

- 모델 뷰 행렬에 의해 카메라의 위치와 방향이 설정되면 이를 실제로 찍는 작업이 필요함. 즉, 3차원 물체를 2차원 필름에 투상해야 함.

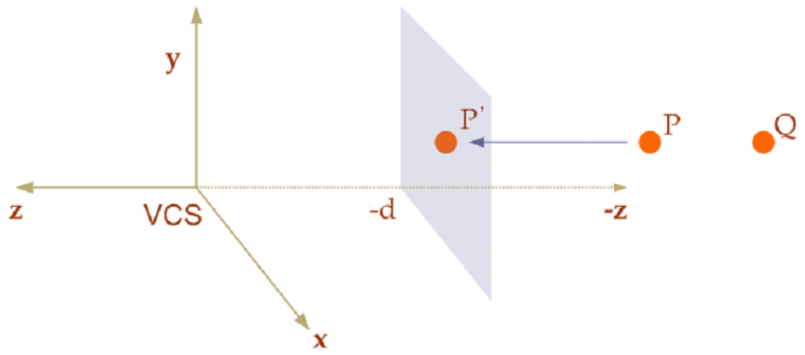
GL의 투상변환

- 투상행렬(Projection Matrix): 투상 변환을 행렬로 표시한 것 (모델 변환과 시점 변환을 행렬로 표시한 것은 모델 뷰 행렬)
- void glMatrixMode(GL_PROJECTION)
  - 행렬 모드를 투상 행렬로 설정하는 함수
- 행렬 모드를 설정하지 않을 경우의 기본값은 모델 뷰 행렬 (GL_MODELVIEW). 투상을 가하기 위해서는 반드시 이 함수를 호출해야 함.
- 호출 이후 다시 행렬 모드를 바꾸기 전까지 제시되는 모든 변환 함수는 투상 행렬에 반영.

투상 행렬 스택

- Projection matrix은 model-view matrix와 마찬가지로 matrix stack으로 구성.
- 투상 행렬 스택(Projection matrix stack)은 최소한 2개 이상. 스택의 맨 위에 있는 것이 현 변환 행렬(CTM_P: Current Transformation matrix). Model-view matrix의 CTM_M과는 완전히 다른 것임에 유의
- void glGetFloatv(GLenum pname, GLfloat *params);
  - 현 투상 행렬 CTM_P를 검색할 때 사용.
  - 만약 GLfloat MyMatrix[16]으로 선언한 변수를 입력 파라미터로 실행하면 (glGetFloatv(GL_PROJECTION_MATRIX, MyMatrix))projection matrix stack의 stack top에 있는 현 투상 행렬의 값이 반환.
  - 해당 행렬은 model-view matrix에도 동일하게 적용 (glGetFloatv(GL_MODEIVIEW_MATRIX, MyMatrix)).

GL의 평행 투상

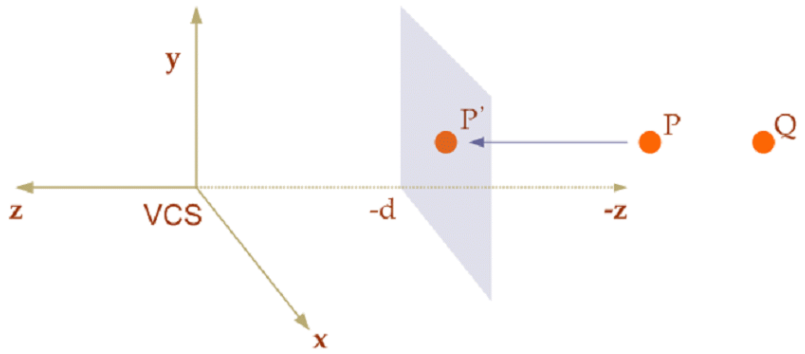


- GL의 카메라는 기본적으로 –z방향을 바라봄.
- 평행 투상에서 투상선은 z축과 나란하므로 그림의 점 P는 P’으로 투상.
- 좌표계 원점으로부터 –z 방향으로 d만큼 떨어진 곳에 투상면이 있다면,

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

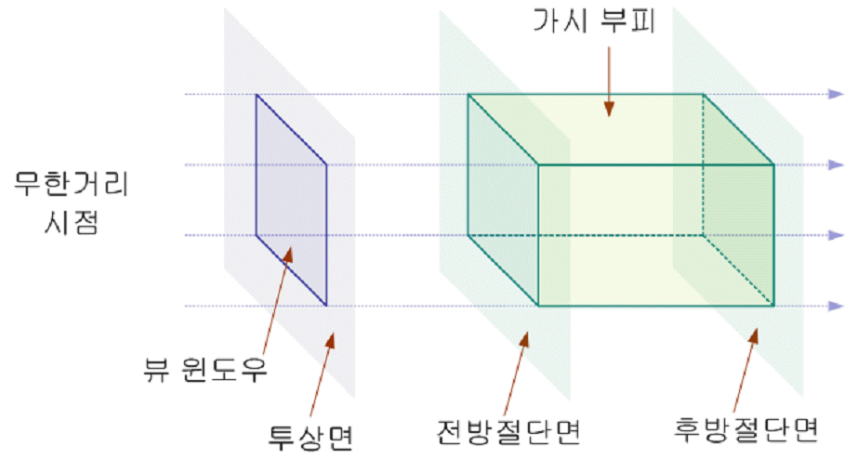
$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -d \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ -d \\ 1 \end{pmatrix}$$

GL의 평행투상



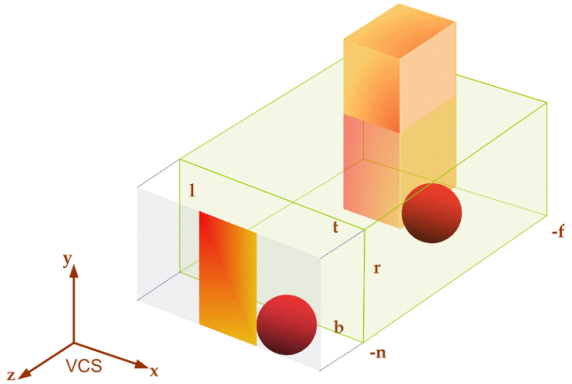
- P, P'은 시점좌표계를 기준으로 함.
- z축 방향의 평행 투상으므로 x-y 평면상의 좌표는 보존됨.
- 그림의 투상면이 z축을 따라 어디로 움직이더라도 영상은 동일. 따라서 투상면에 맺히는 영상은 거리 d와 무관.
- z 값이 클수록 시점에서 멀어진다는 점에서 z를 '물체의 깊이(Depth) ' 라고도 함.

정규화 가시 부피에 의한 투상



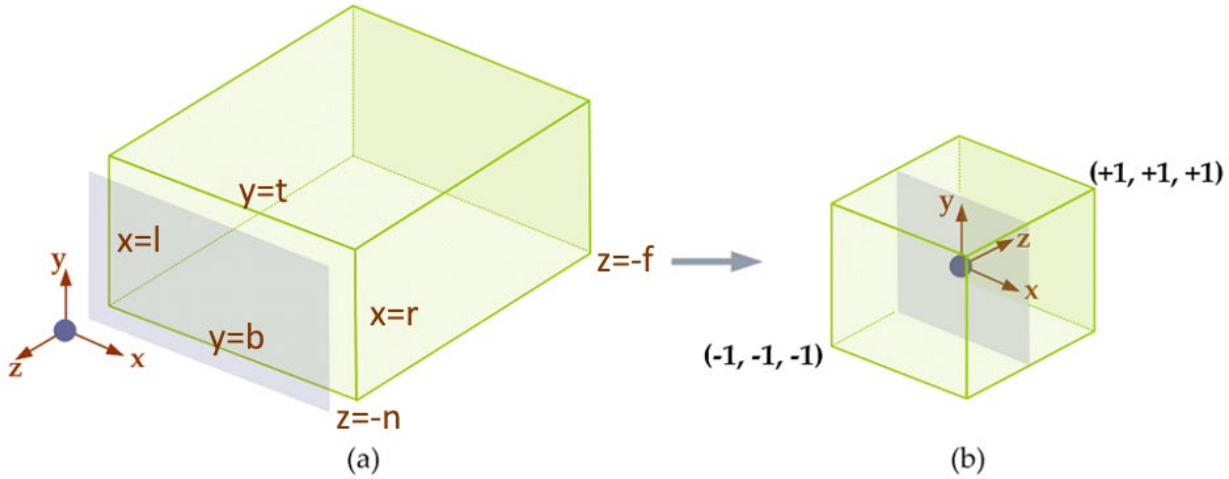
- 가시부피 (View Volume): 투상하는 범위. 제한도니 크기의 뷰 윈도우에 모든 물체의 영상을 맺히게 할 수 없기 때문에 설정 필요.
- 그림에서 시점에 가까운 쪽을 전방절단면(Near Clipping Plane, Near Plane, Front Plane, Hither), 먼 쪽을(Far Clipping Plane, Far Plane, Back Plane, Yon)으로 부름.

정규화 가시 부피



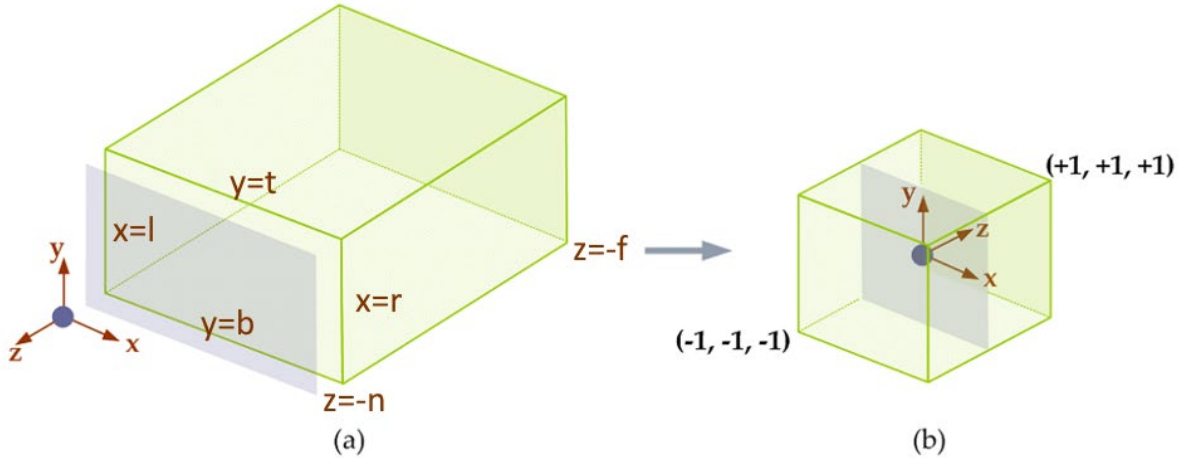
- 가시 부피 안에 있는 물체만 투상
- void glOrtho(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble near, GLdouble far);
  - 평행 투상의 가시 부피를 설정. 정사 투상보다는 일반적인 평행 투상을 위한 함수 투상면이 물체의 주 평면과 반드시 나란할 필요는 없기 때문.
  - 파라미터는 부피의 좌, 우, 하, 상, 전, 후를 뜻함.
  - near와 far값은 반드시 **양수**로 나타내야 함. 시점으로부터 절단면까지의 거리이기 때문. 다만 실제 z좌표는 -1를 곱해야 함 (예) 전방 절단면 우상단 좌표 (right, top, -near))

정규화 가시 부피



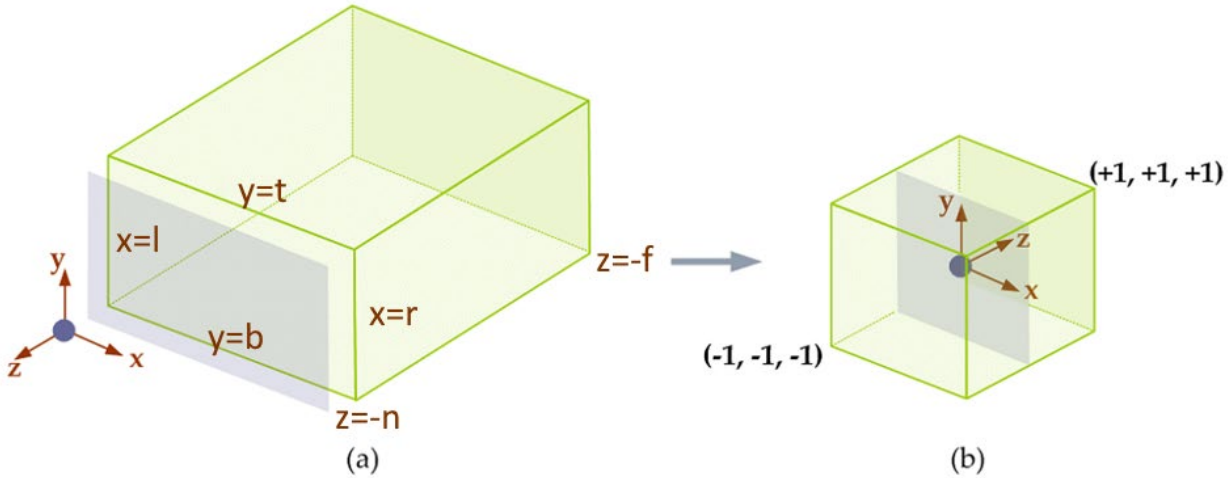
- 정규화 가시부피 (CVV: Canonical View Volume)
  - 가로, 세로, 높이가 2인 정육면체로 투상
  - 정규화 변환 (Normalization Transformation)

정규화 가시 부피



- 제한된 크기의 가시 부피를 제한된 크기의 뷰 윈도우로 투상하기 위해서 GL은 정규화 변환(Normalization Transformation)을 통해서 정규화.
- GL이 가시 부피 그대로를 사용하지 않는 이유는
  - 평행 투상, 원근 투상 모두 동일한 모습의 정규화 가시 부피를 사용함으로써 파이프라인 처리 구조가 동일
  - 가시 부피 밖의 물체를 절단하는 데 있어서 정규화 가시 부피인 정육면체를 기준으로 하는 것이 원래의 가시 부피를 그대로 절단하는 것보다 단순
  - 시점 좌표계 원점을 중심으로 가로, 세로 길이를 1로 정규화함으로써 화면 좌표계로 변환하기가 수월

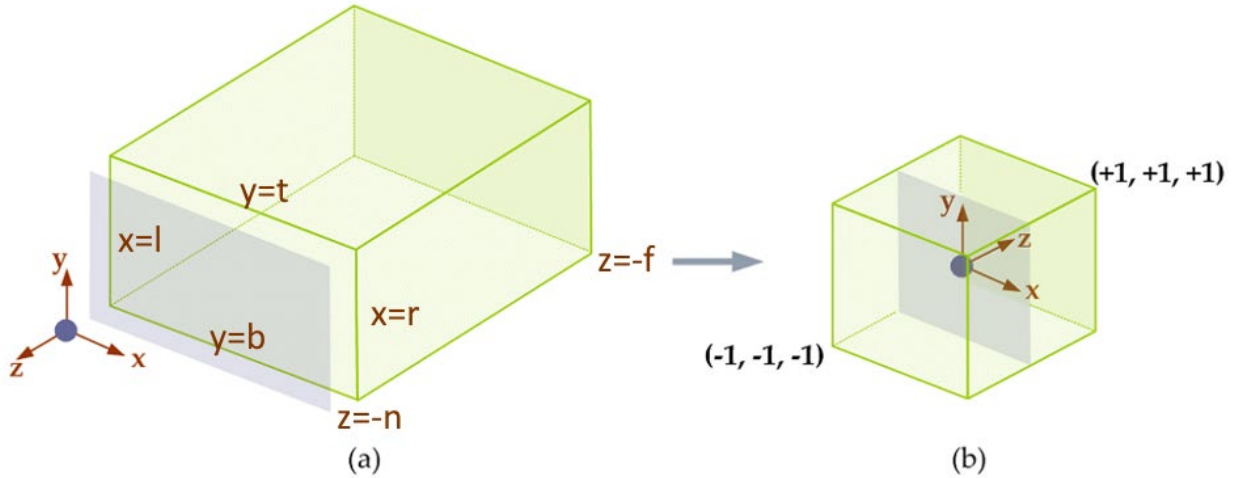
정규화 가시 부피



- 좌측의 가시 부피를 우측의 시점 좌표계 원점으로 이동하기 위한 이동 변환 행렬(translational transformation matrix)

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

정규화 가시 부피



- 이동된 가시부피에 크기조절을 가하여 길이 2인 정육면체를 만들기 위한 크기 조절 변환 S
- 왼손 좌표계로 변경함에 따른 z축 방향 반전을 위한 반사 변환 R

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