컴퓨터그래픽스

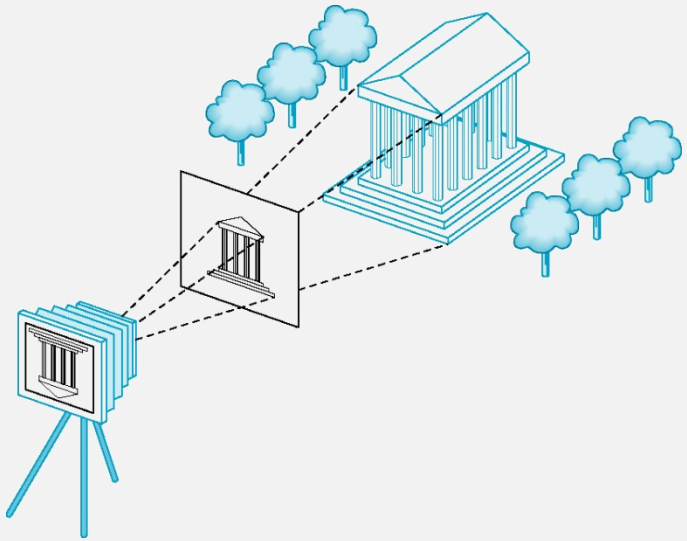
김준호
Visual Computing Lab.
국민대학교 소프트웨어학부

- Principles
- Extrinsic Parameters
- Intrinsic Parameters

Synthetic Camera

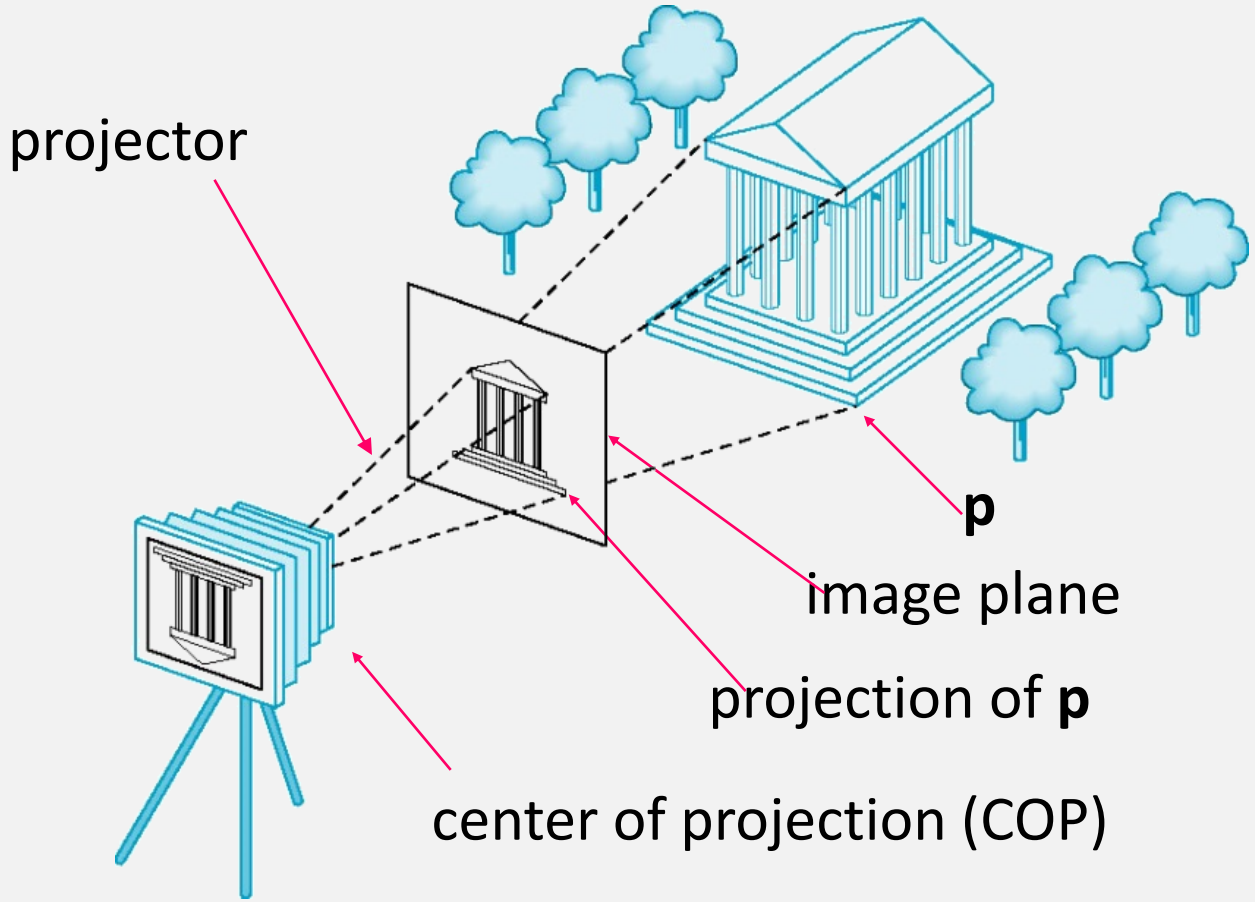
Elements of Image Formation

- Viewer (or camera)
  - Synthetic camera
- Objects
  - Synthetic objects
- Light source(s)
  - Synthetic lights
- Attributes
  - Material, surface normal for reflection model (i.e., light-material interaction)



Synthetic image formation
in Computer Graphics

Synthetic Camera Model

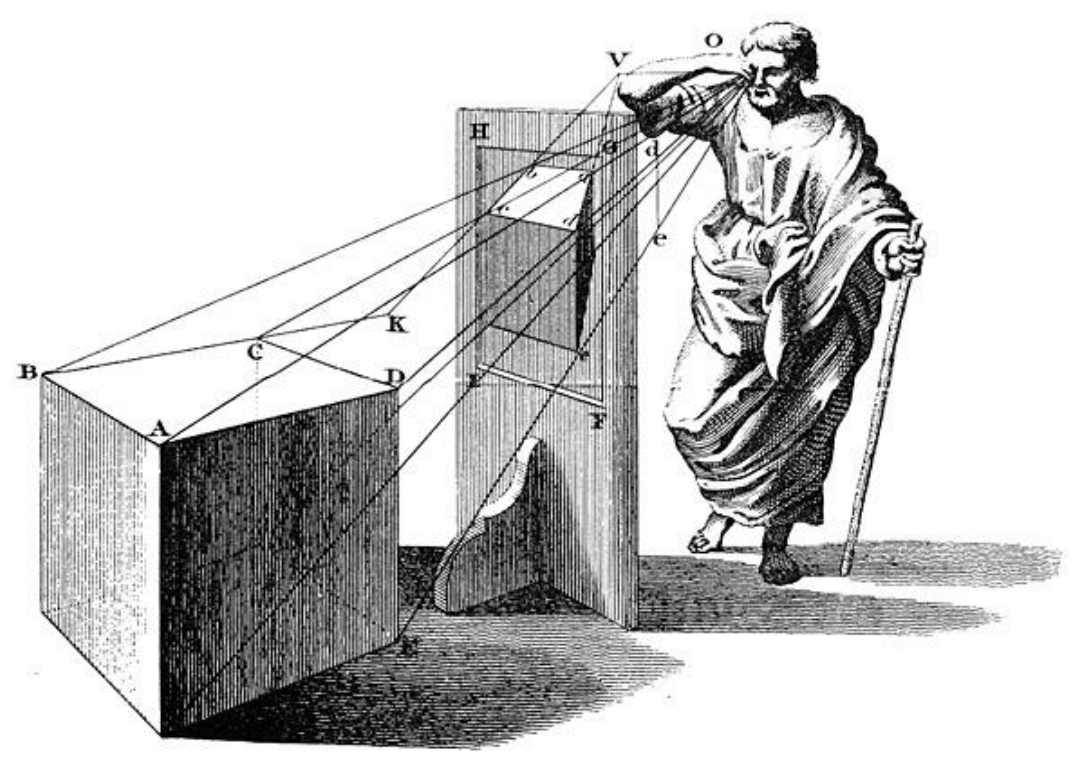


Seeing

본다는 것 – 선형 투사 기법

본다 (Seeing) v.s. 본다는 것을 이해했다

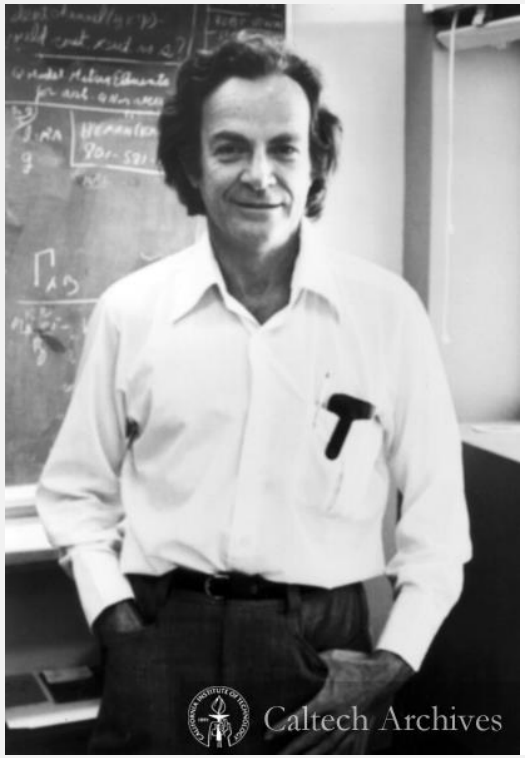
본다: 빛이 날아와 눈에 **맺히는 작용**

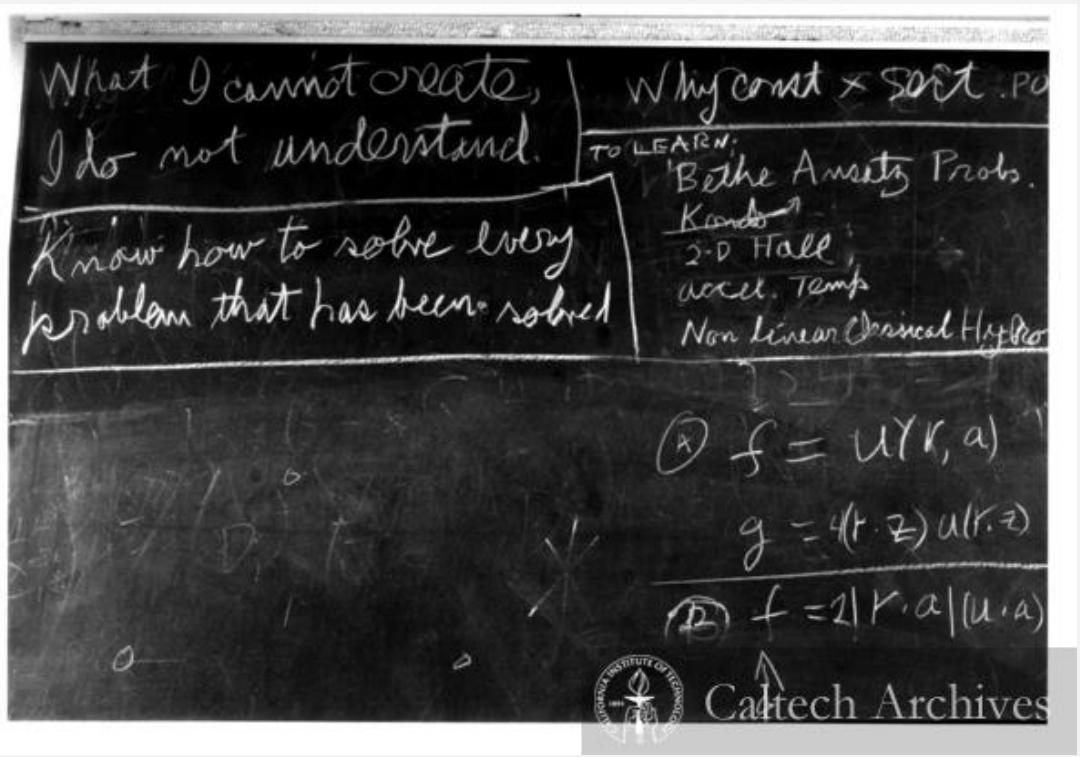


‘**본다**는 것을 **이해**했다’는 것은
‘**장면**에 대한 **묘사**가 **가능**’하다는 것

본다는 것을 이해했다

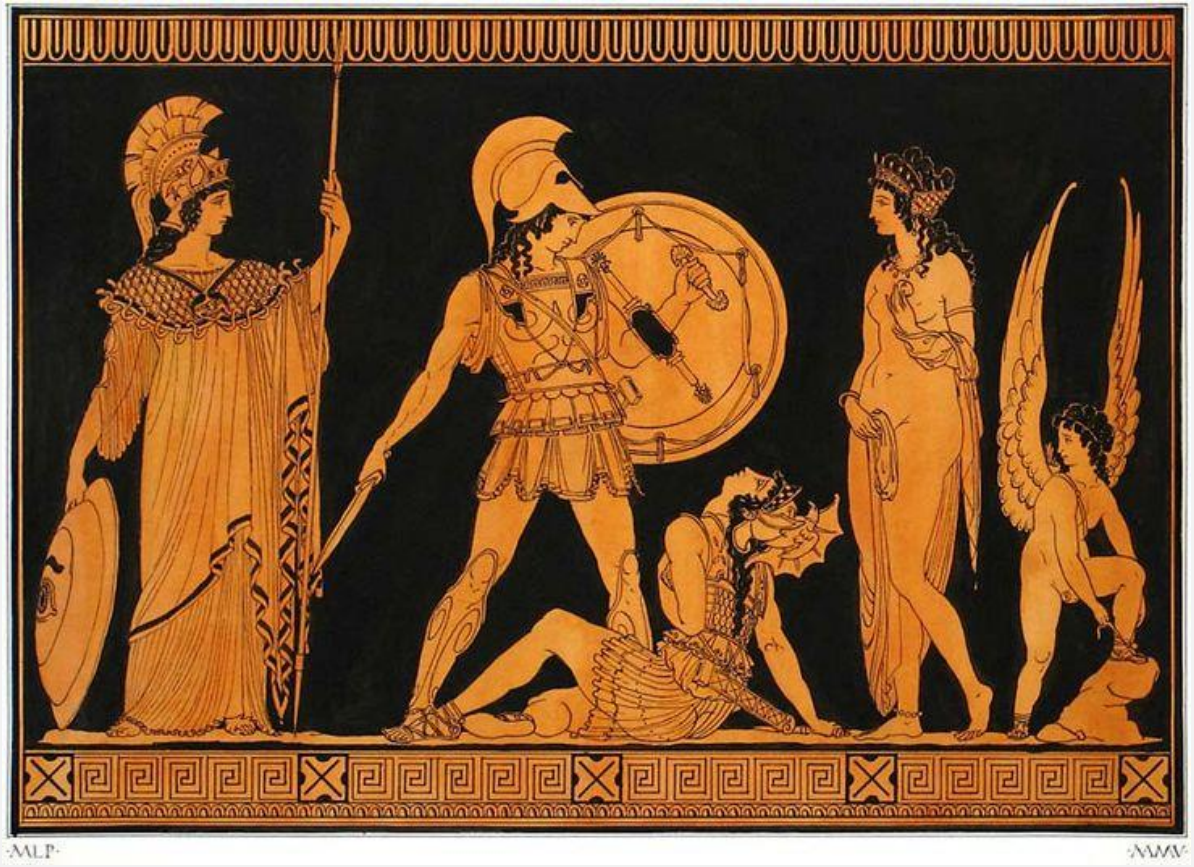
- Rechard Feynman's blackboard at time of his death (link)
  - **What I cannot create, I do not understand**
  - Know how to solve every problem that has been solved





장면 묘사 (Illustration)

[고대 이집트 벽화]

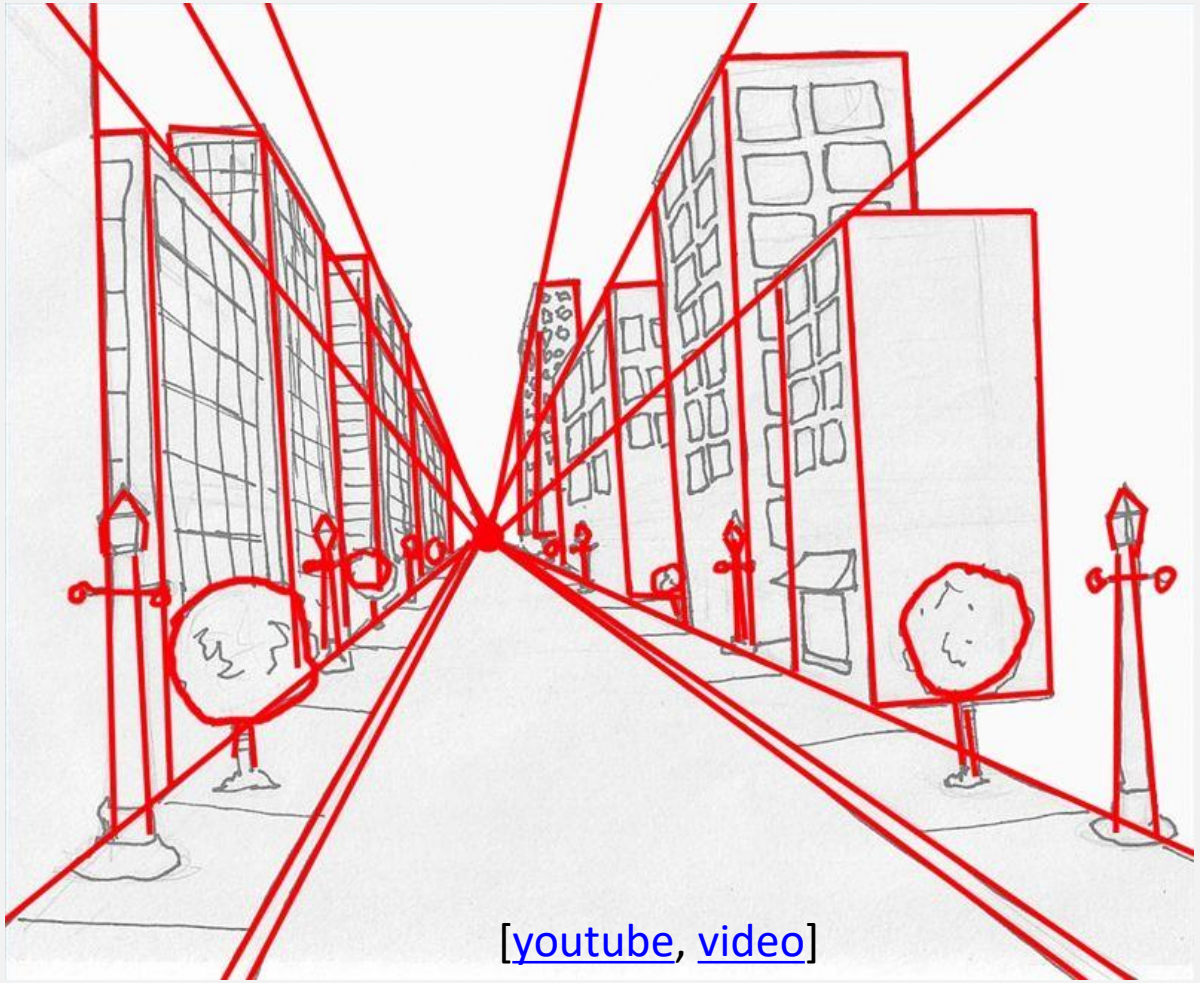


[고대 그리스 미술]

장면 묘사 (Illustration)

[현대 스케치 기법]

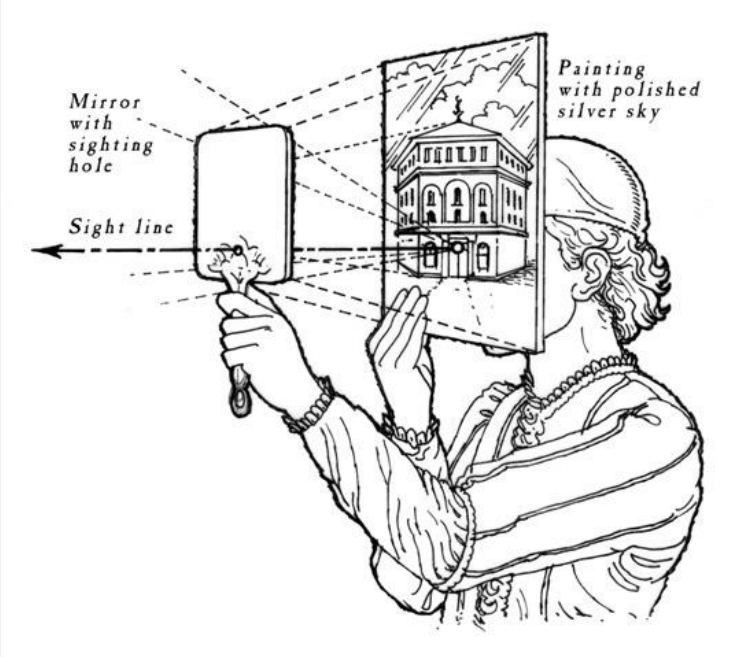
선형 투사 기법을 이용한 그리기 (Linear Perspective Drawing)



[youtube, video]

필리포 브루넬레스키 (1377 – 1446)

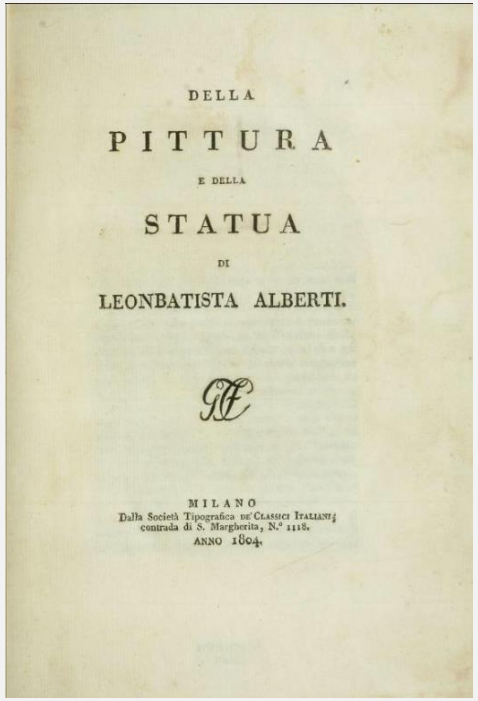
- 건축가, 조각가, 기계공학자
  - 선형 투사 기법 (linear perspective) 발견

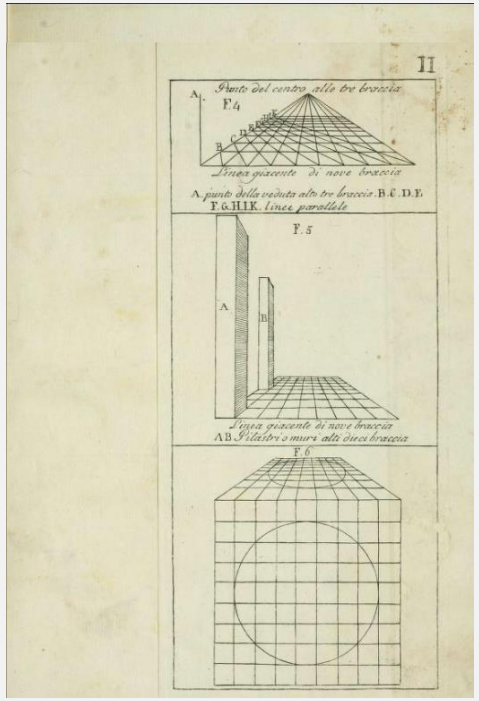


[youtube, video, 3:40 – 5:38]

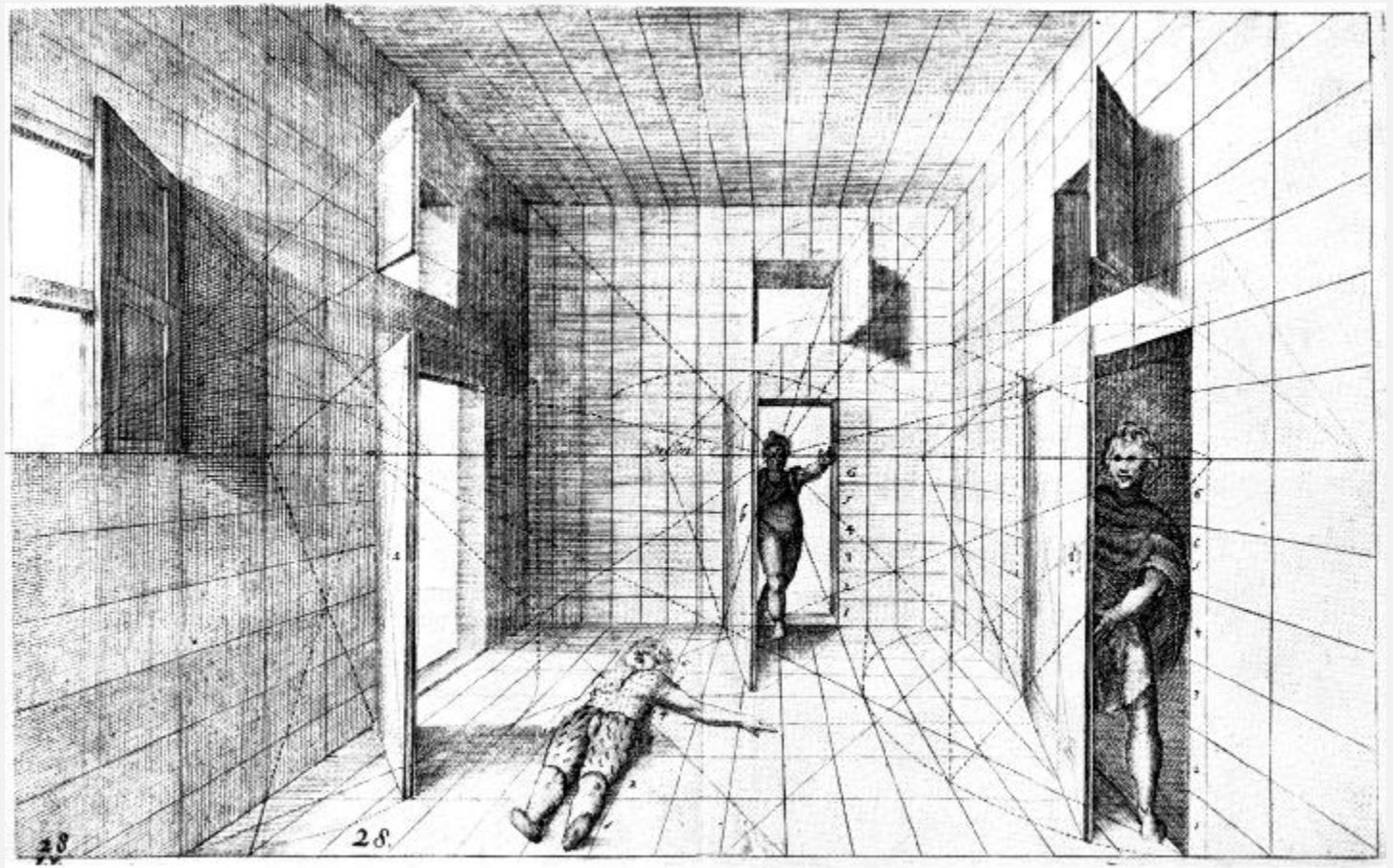
레온 바티스타 알베르티 (1404 – 1472)

- 철학자, 건축가
  - 선형 투사 기법 이론 정립
    - <Della pittura, 회화론> 저술 (link)





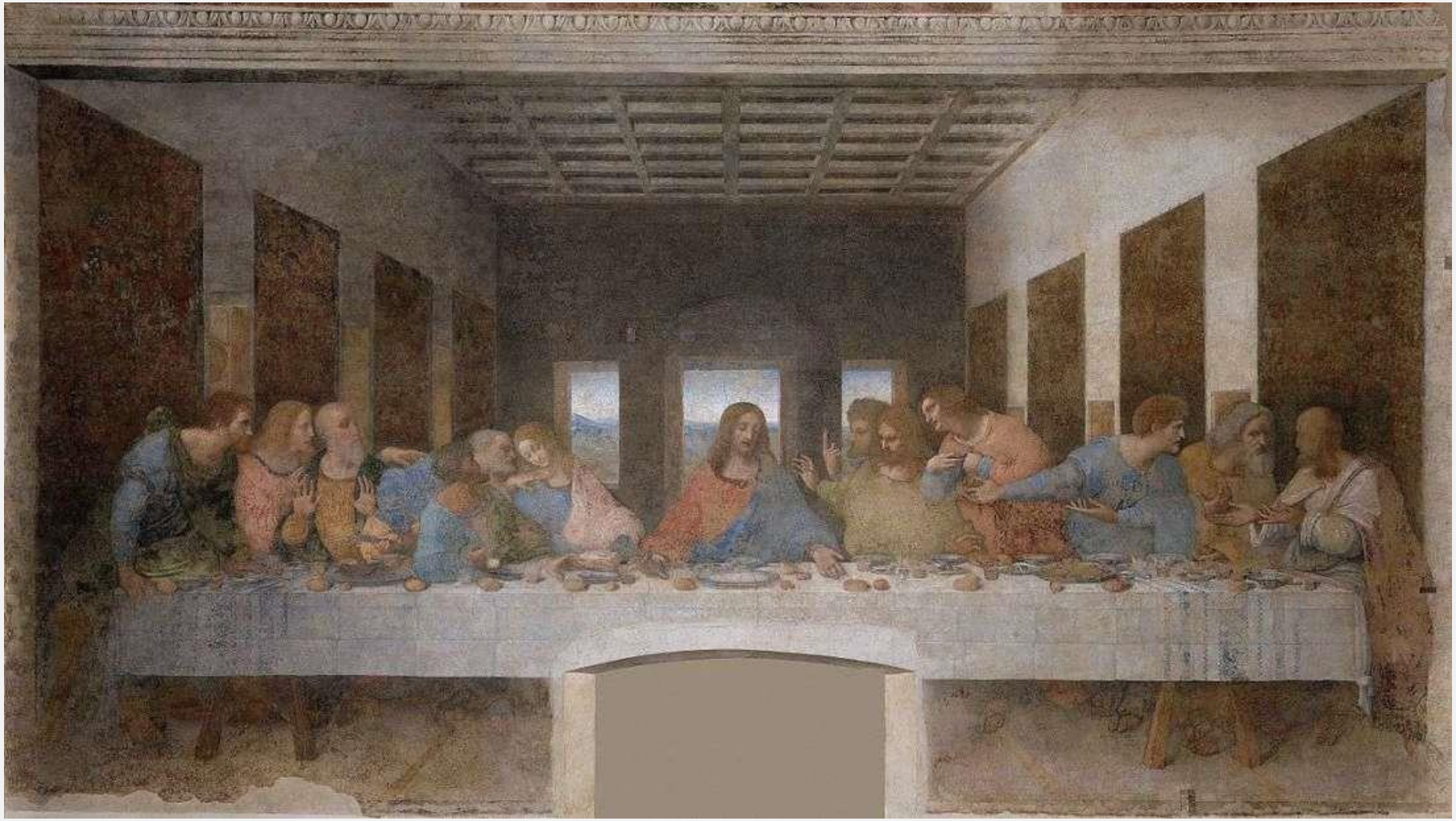
투사 기법에 의해 공간묘사의 절차화



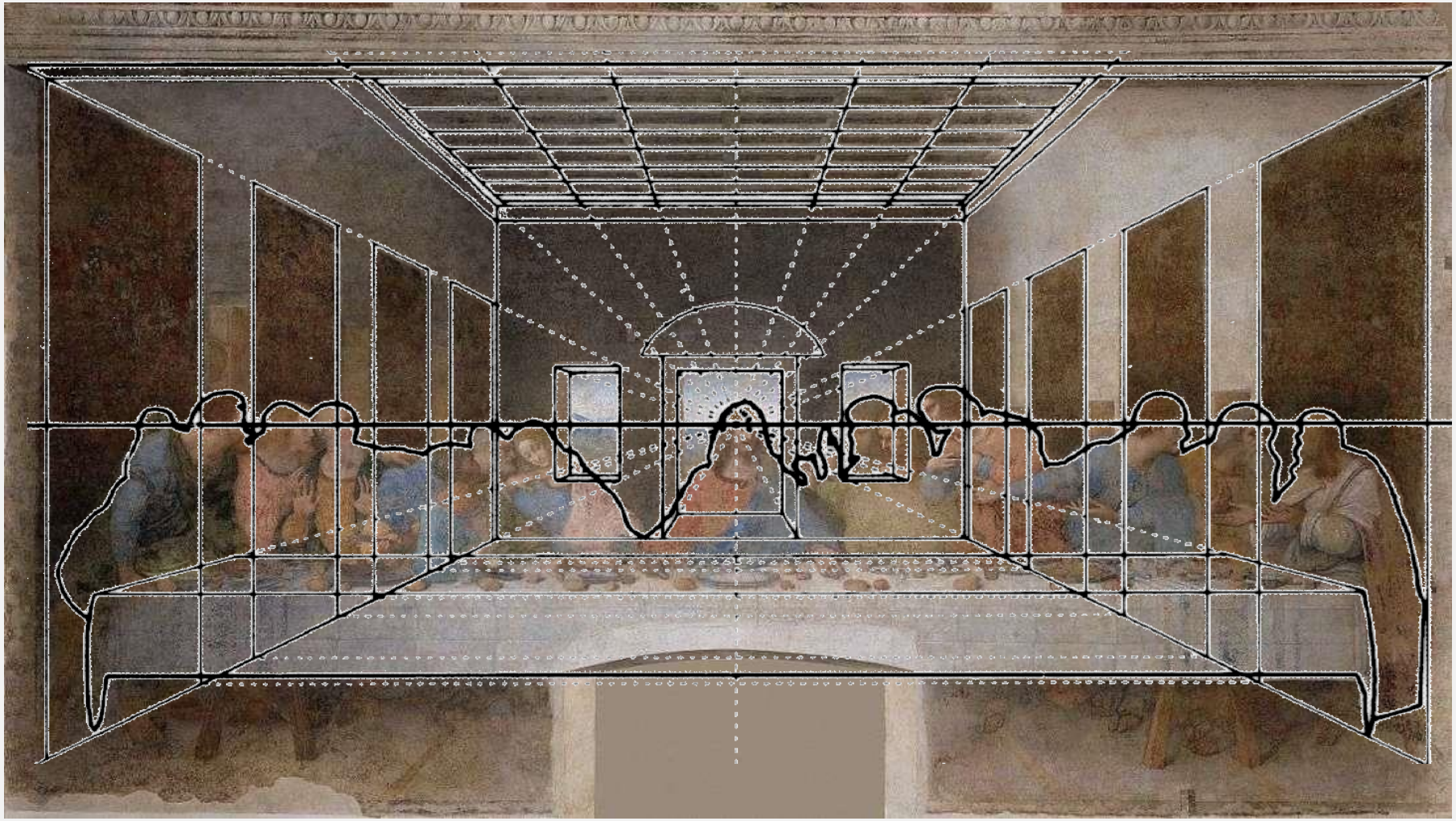
JAN VREDEMAN DE VRIES, *Perspective* (Leiden, 1604–5), plate 28. Courtesy, the Bancroft Library, Berkeley, California.

[youtube, video]

르네상스 시대 투사 기법 적용 사례 – 최후의 만찬

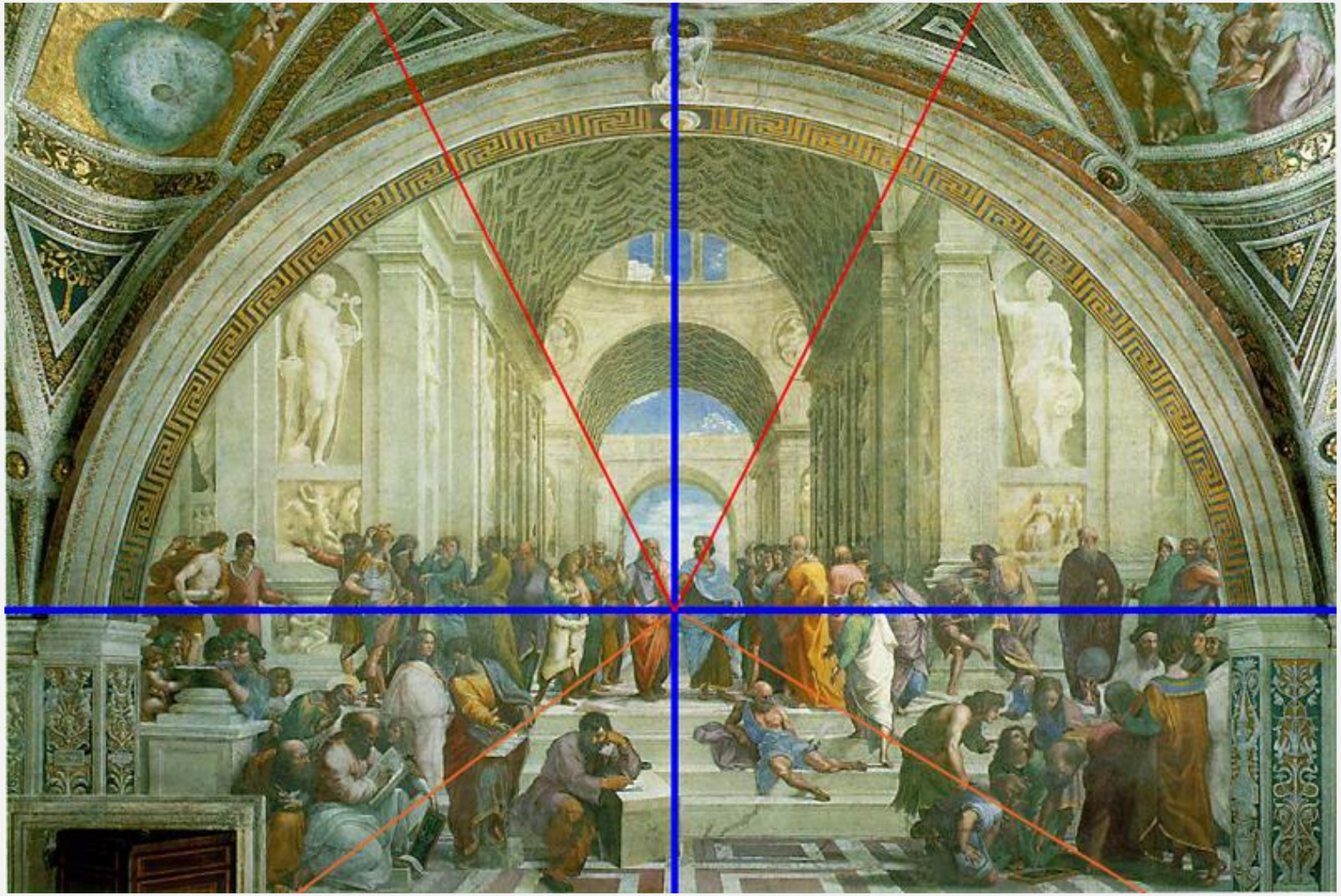


르네상스 시대 투사 기법 적용 사례 – 최후의 만찬



르네상스 시대 투사 기법 적용 사례 – 아테네 학당

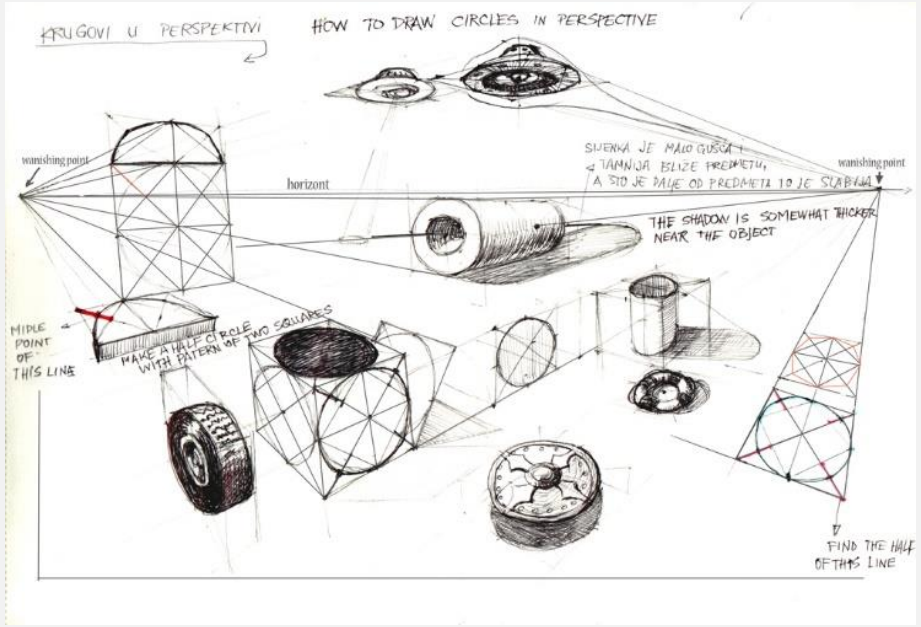
르네상스 시대 투사 기법 적용 사례 – 아테네 학당

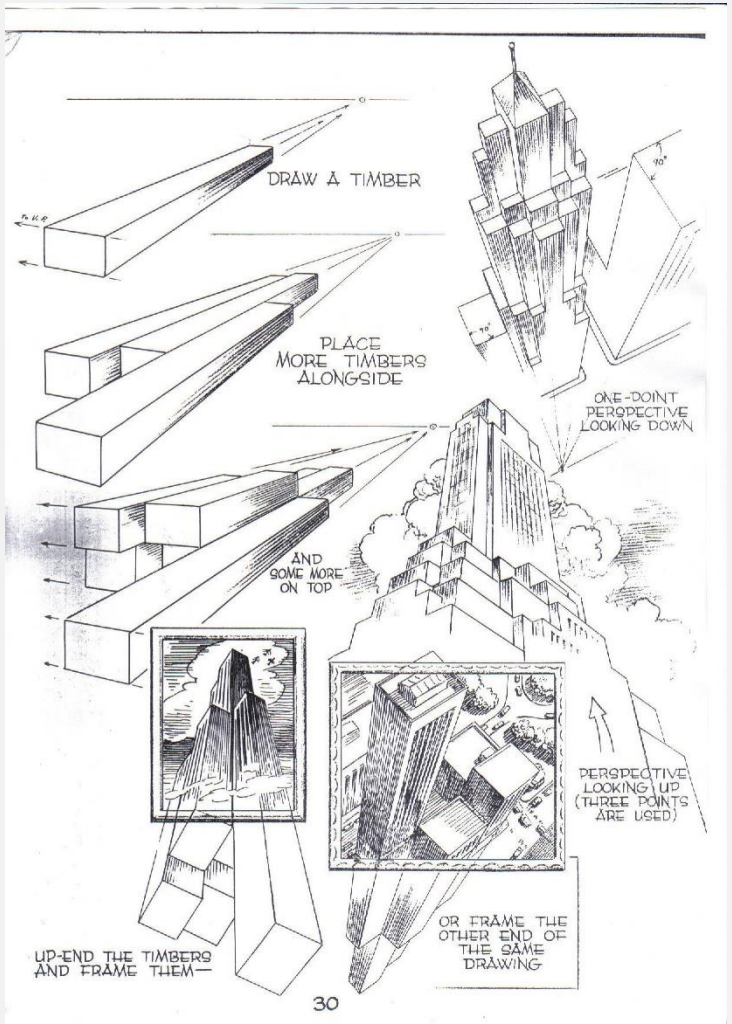


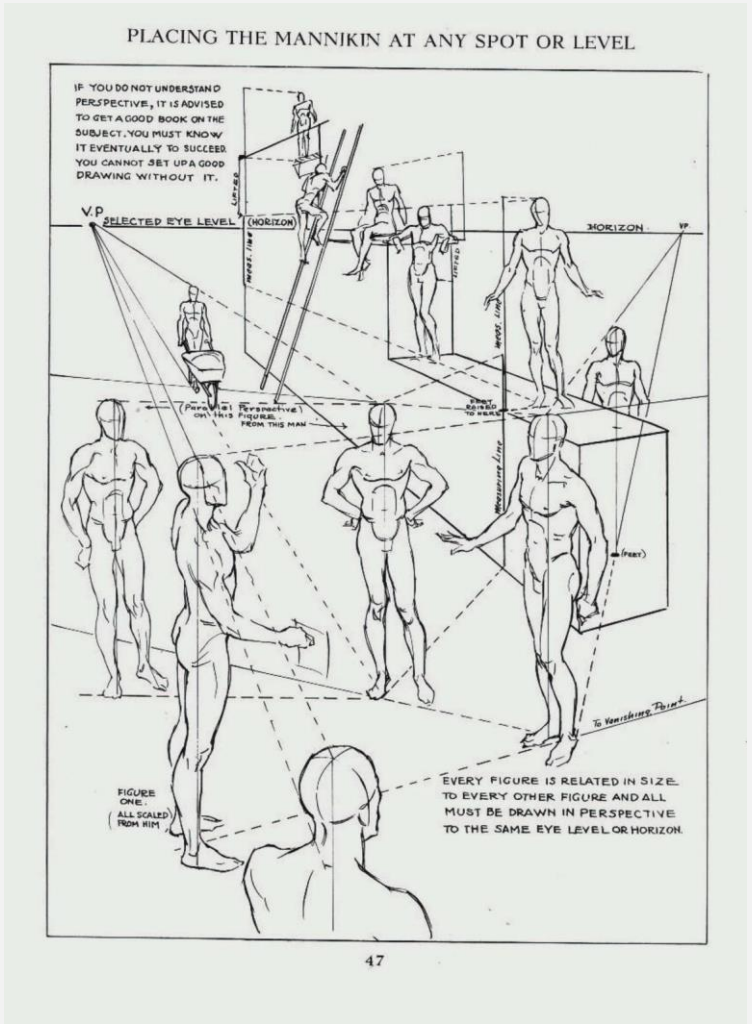
뒤러(1471 – 1528)의 투시도 기계

Durer’s Perspective Machine

오늘날 절차적 방식의 묘사







사영기하학(projective geometry) 발전

지라르 데자르그
(1591 – 1661)

블레즈 파스칼
(1623 – 1662)

카를 프리드리히 가우스
(1777 – 1855)

니콜라이 로바첩스키
(1792 – 1856)

베른하르트 리만
(1826 – 1866)

펠릭스 클라인
(1849 – 1925)

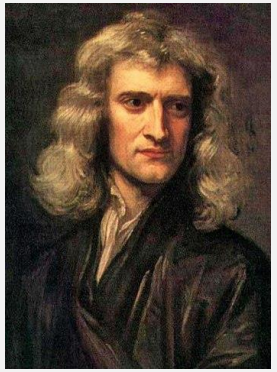
Non-Euclidean Geometry
(비유클리드 기하학)

Projective Geometry
(사영기하학)

사영기하학(projective geometry) 발전 (일시 침체기)

지라르 데자그르
(1591 – 1661)

블레즈 파스칼
(1623 – 1662)



아이작 뉴턴
(1643 – 1726)

카를 프리드리히 가우스
(1777 – 1855)

니콜라이 로바첸스키
(1792 – 1856)

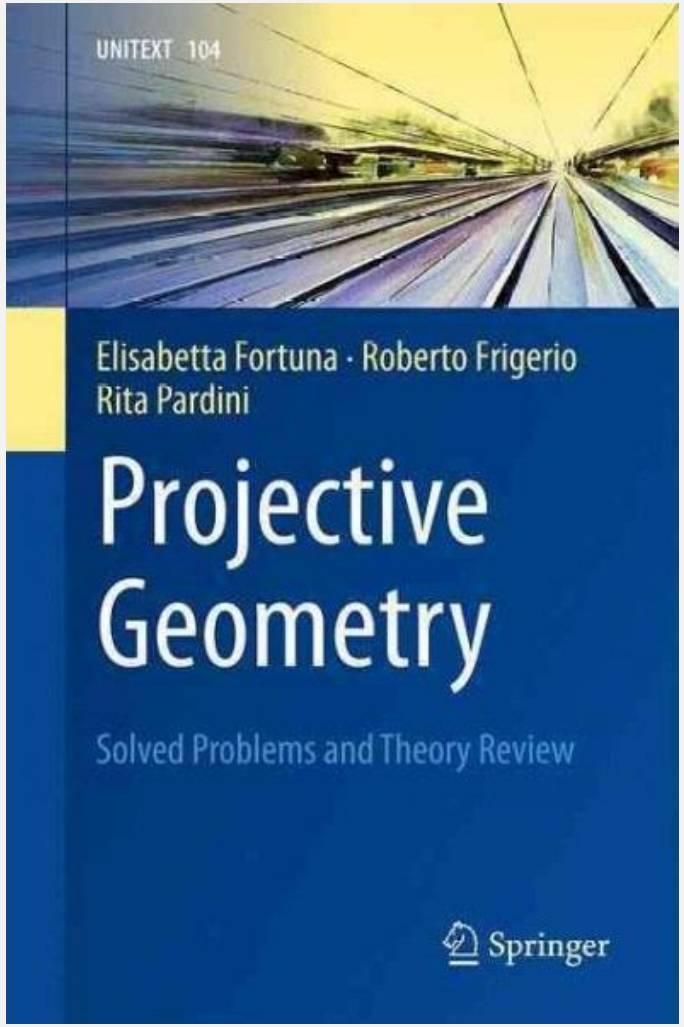
베른하르트 리만
(1826 – 1866)

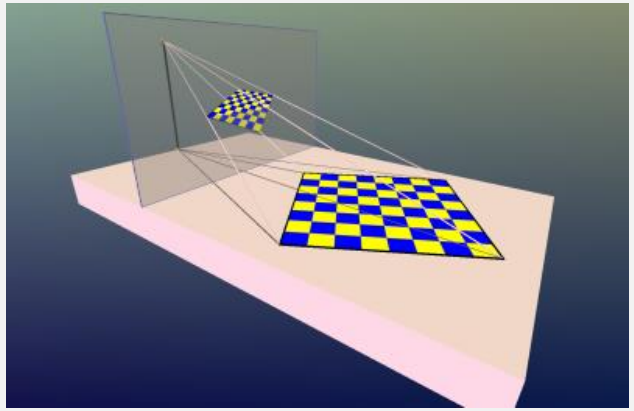
펠릭스 클라인
(1849 – 1925)

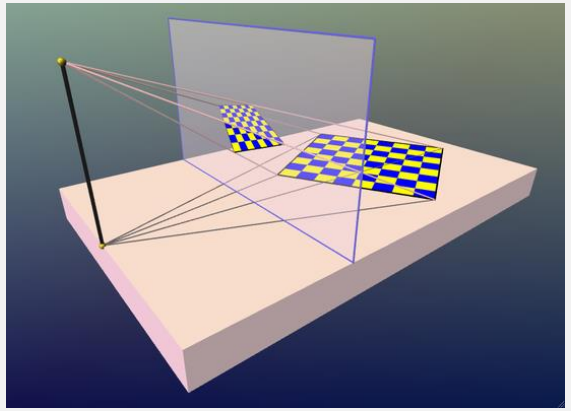
Non-Euclidean Geometry
(비유클리드 기하학)

Projective Geometry
(사영기하학)

사영기하학(projective geometry) 발전

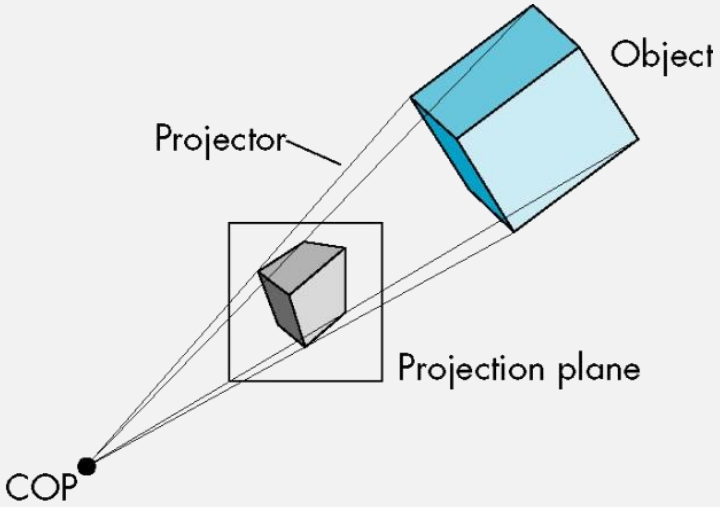




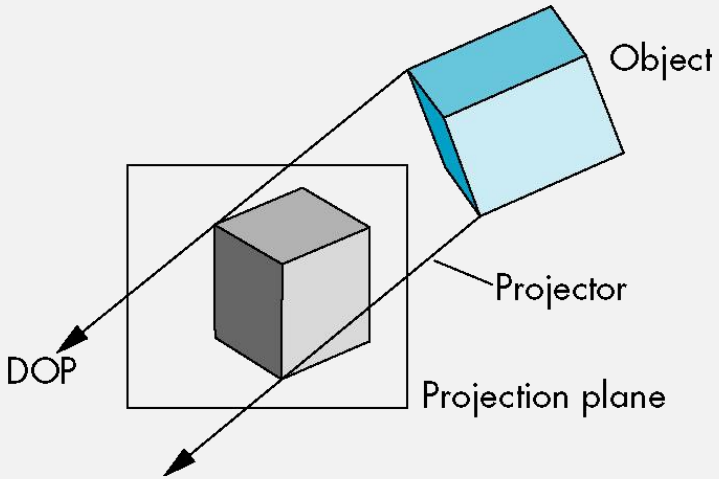


Camera Specification – Projection types

- Projection types
  - Perspective projection
  - Orthographic projection



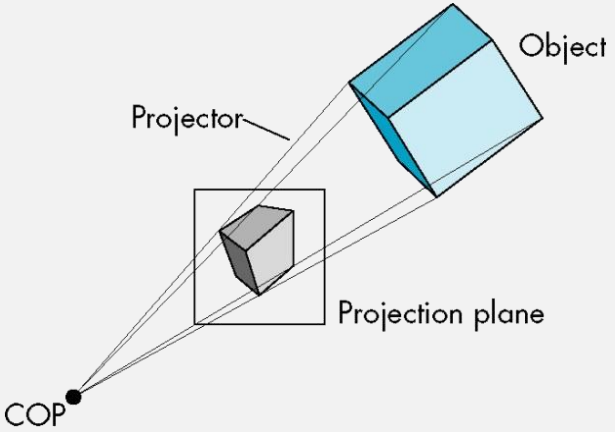
Perspective projection



Orthographic projection

Camera Specification – Projection types

- Projection types
  - Perspective projection
    - Parallel lines → Vanishing point
  - Orthographic projection

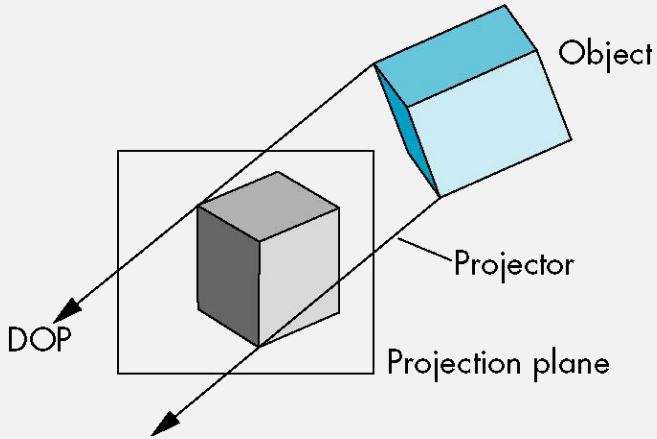


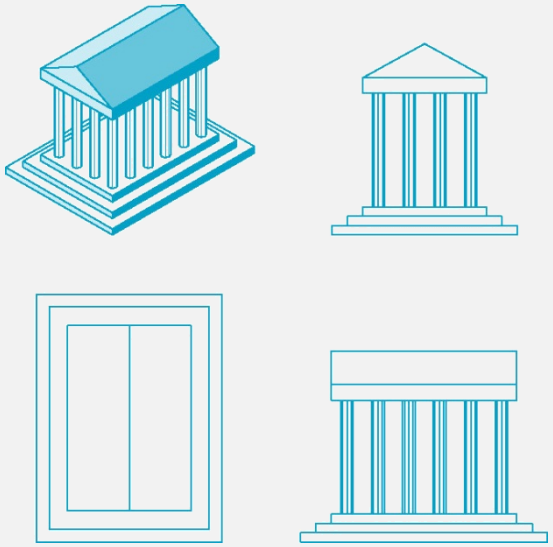
(a)

Eyes (or typical camera)

Camera Specification – Projection types

- Projection types
  - Perspective projection
  - Orthographic projection
    - Parallel lines → Parallel lines



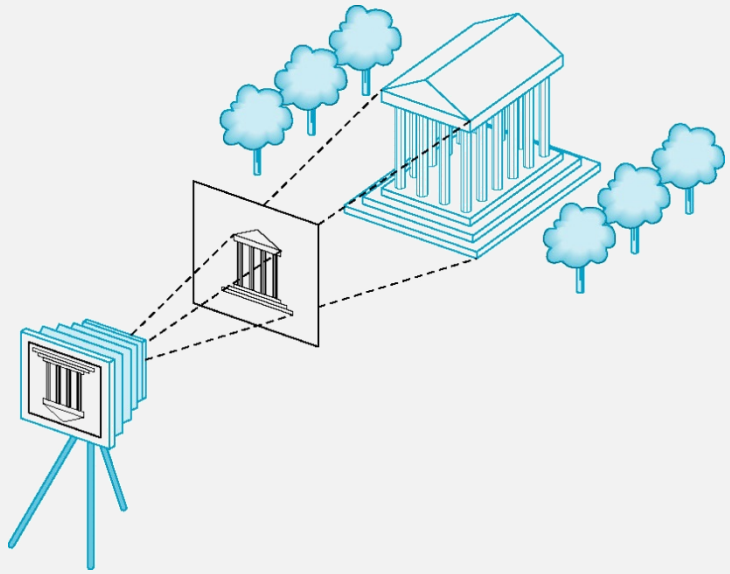


Multiview orthographic projections

Tilt-shift camera

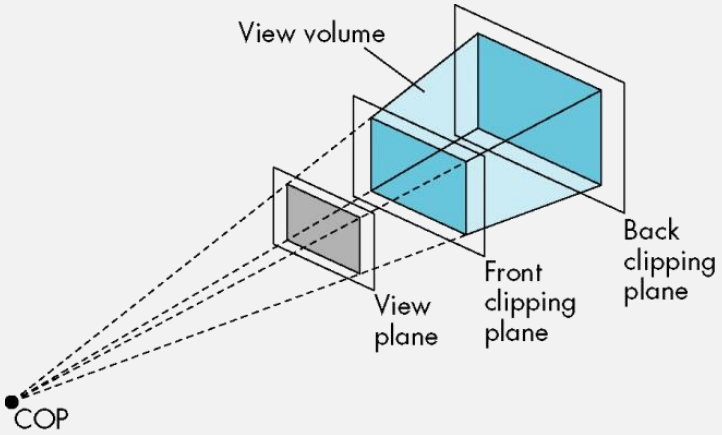
Camera Specification – Clipping Planes

- Clipping
  - Physically, a camera (or your eyes) cannot “see” the whole world
    - Objects that are not within the view volume are said to be clipped out of the scene
    - 4 clipping planes: left / right / top/ bottom



Camera Specification – Clipping Planes

- Clipping
  - In OpenGL, there are two additional clipping planes
    - 6 clipping planes: left / right / top/ bottom + front / back
    - Computer cannot process infinitely many objects



Camera Specification – Extrinsic parameters

- Extrinsic parameters: 6 degrees of freedom (DOF)
  - Position: 3DOF
    - Center of projection (COP): position of center of lens (x, y, z)
  - Orientation: 3 DOF
    - pitch(끄덕) — yaw(도리) — roll(갸웃)
  - In OpenGL, extrinsic parameters are handled by camera transformations
    - ~~OpenGL 1.x: simply use gluLookAt() in OpenGL Utility (GLU) library~~
    - OpenGL 2.x or higher: implement proper transformations by yourself

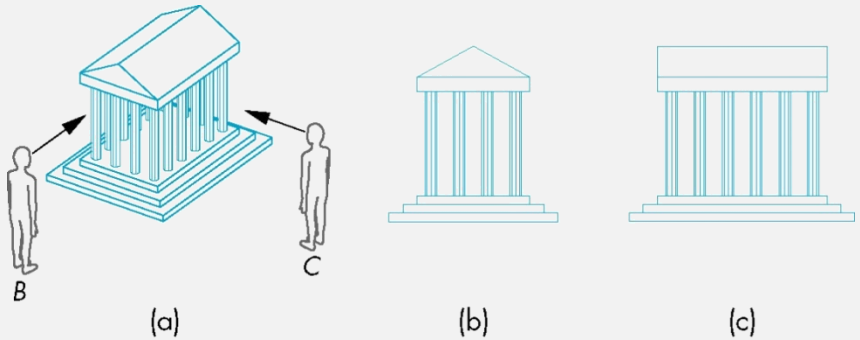
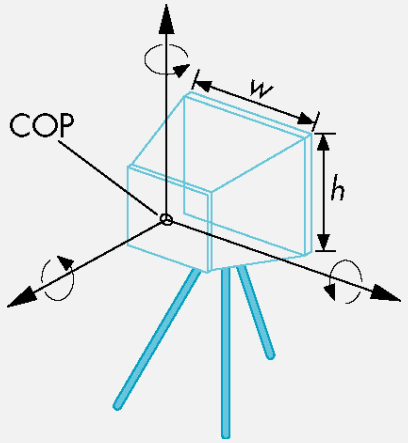


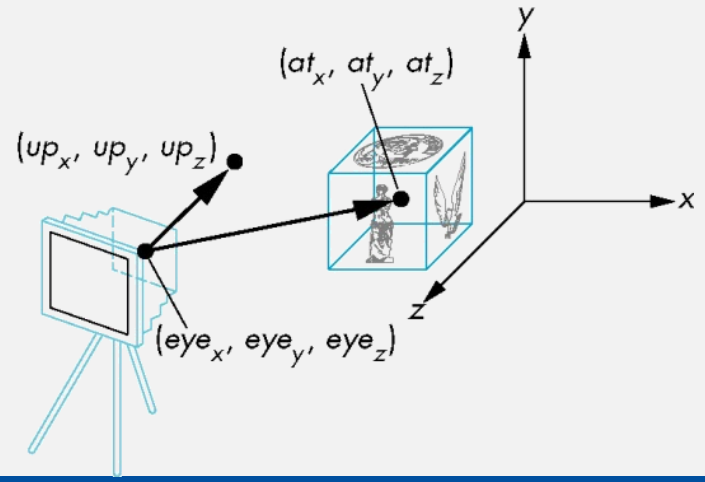
Image seen from different views



Camera Specification – Extrinsic parameters

- ~~gluLookAt() OpenGL 1.x~~
  - OpenGL utility (GLU) function for setting extrinsic parameters of OpenGL camera

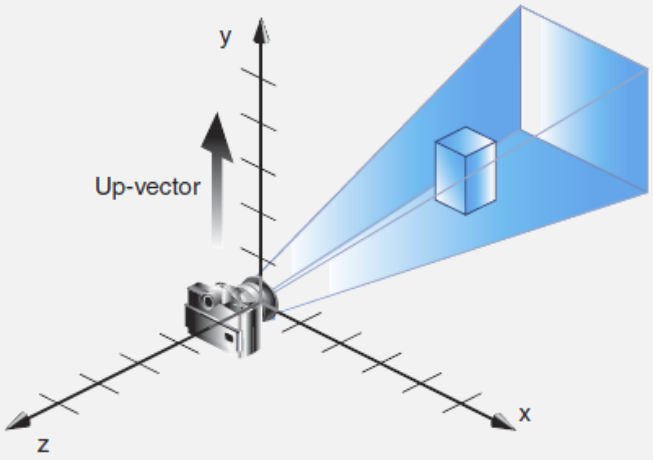
```
// OpenGL utility (GLU) function for specifying extrinsic parameters of the OpenGL camera
//
// The eyex, eyey, eyez arguments specifies the desired viewpoint
// The centerx, centery, centerz arguments any point along the desired line of sight
// The upx, upy, upz arguments indicate which direction is up

void gluLookAt(GLdouble eyex, GLdouble eyey, GLdouble eyez,
               GLdouble centerx, GLdouble centery, GLdouble centerz,
               GLdouble upx, GLdouble upy, GLdouble upz);
```



Camera Specification – Extrinsic parameters

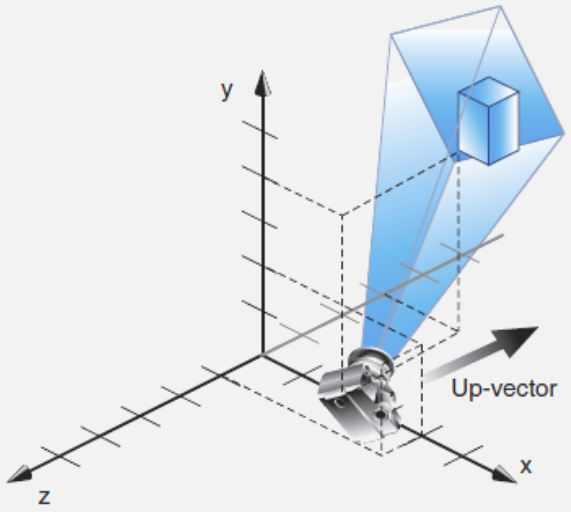
- ~~Example 1 (OpenGL 1.x)~~

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
gluLookAt(0.0, 0.0, 0.0,     // eye
          0.0, 0.0, -100.0, // center
          0.0, 1.0, 0.0);    // up
```



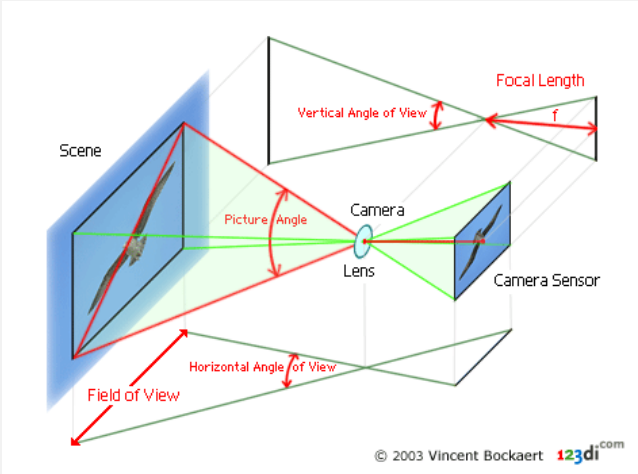
- ~~Example 2 (OpenGL 1.x)~~

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
gluLookAt(4.0, 2.0, 1.0,     // eye
          2.0, 4.0, -3.0,    // center
          2.0, 2.0, -1.0);   // up
```

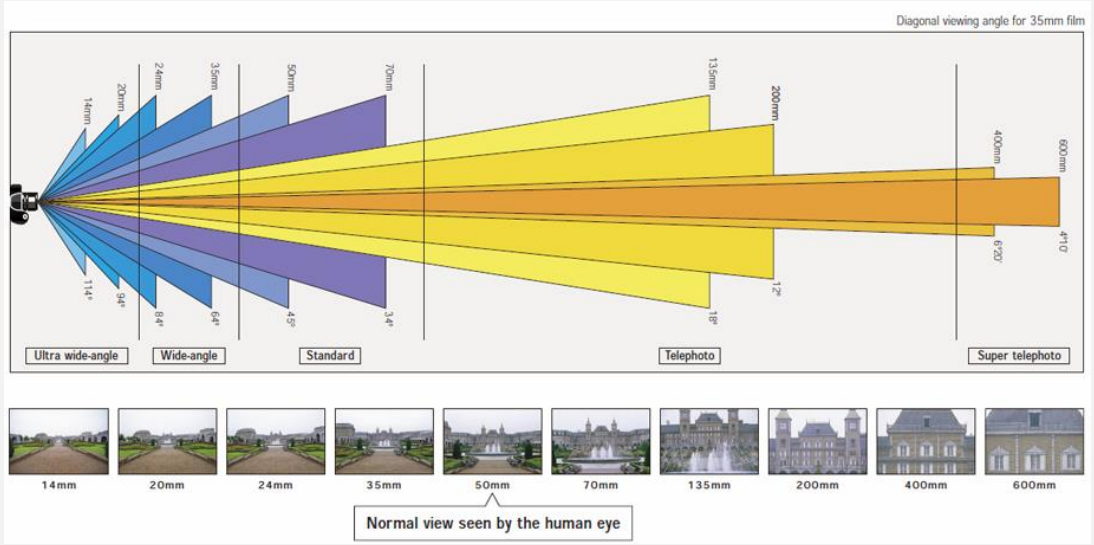


Camera Specification – Intrinsic parameters

- Intrinsic parameters
  - Focal length
    - Physical distance: lens – camera sensor
    - Zoom-in / zoom-out



http://www.dpreview.com/glossary/optical/focal-length

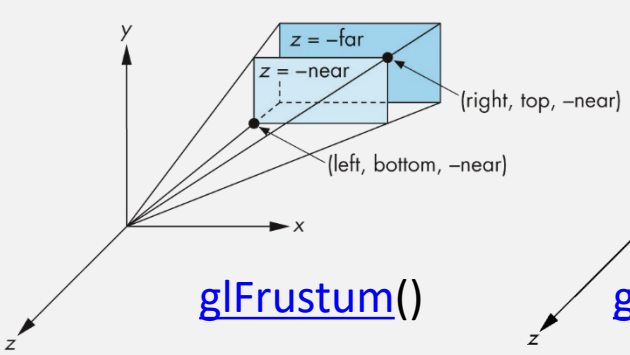


http://panasonic.jp/support/global/cs/dsc/knowhow/knowhow12.html

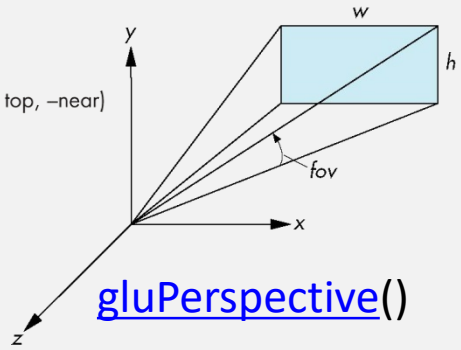
Camera Specification – Intrinsic parameters

- Intrinsic parameters
  - Focal length
    - In OpenGL, there is no physical meaning
  - Field of view (FOV)
    - In OpenGL, zoom-in/-out is handled by changing the field of view
      - Perspective projection: ~~glFrustum() (or gluPerspective() in GLU library): OpenGL 1.x~~
      - Orthographic projection: ~~glOrtho(): OpenGL 1.x~~

Perspective projection

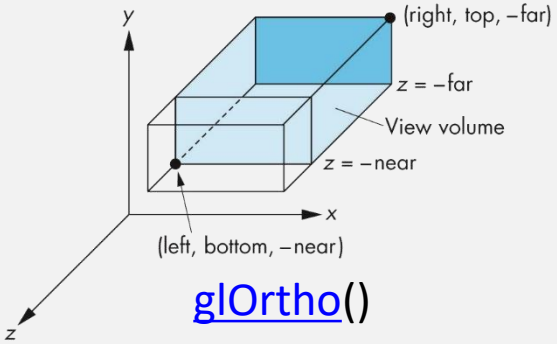


glFrustum()



gluPerspective()

Orthographic projection

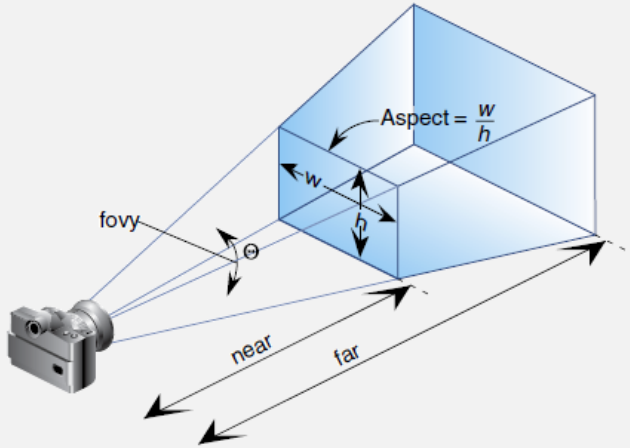


glOrtho()

Camera Specification – Intrinsic parameters

- ~~gluPerspective() - OpenGL 1.x~~
  - GLU function for setting intrinsic parameters of OpenGL perspective camera

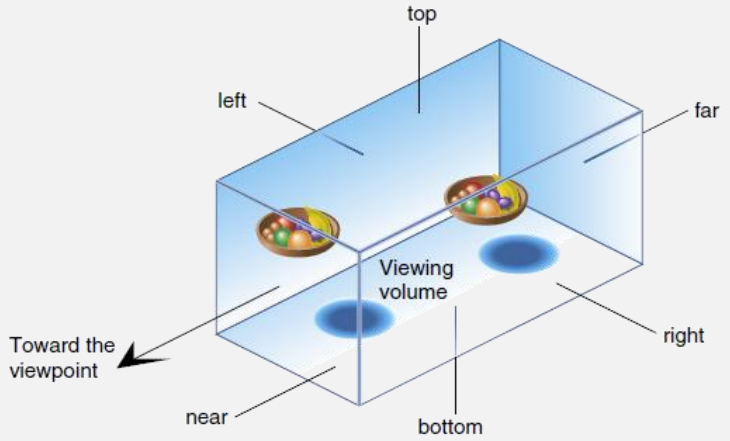
```
// OpenGL Utility (GLU) function for specifying intrinsic parameters of the OpenGL perspective camera
//
// fovy is the anlge of the field of view in yz-plane
// aspect is the aspect ratio of the frusutum, its width divided by its height (w/h)
// near and far values are the distances between the viewpoint and the clipping planes

void gluPerspective(GLdouble fovy, GLdouble aspect,
                    GLdouble near, GLdouble far);
```



Camera Specification – Intrinsic parameters

- ~~glOrtho() - OpenGL 1.x~~
  - OpenGL function for setting intrinsic parameters of OpenGL orthographic camera

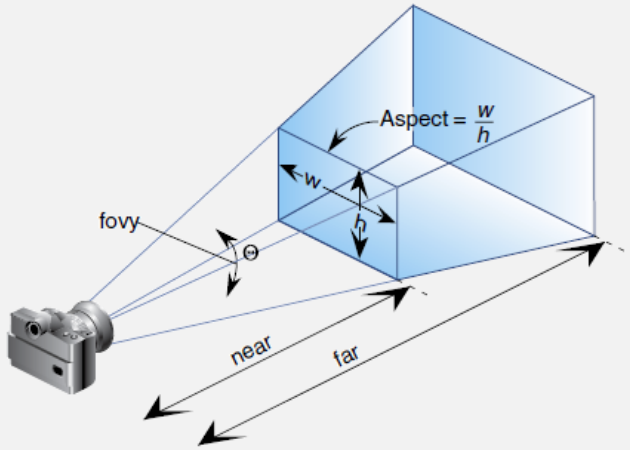
```
// OpenGL function for specifying intrinsic parameters of the OpenGL orthograpic camera
//
// left, right        specify the coordinates for the left and right vertical clipping planes
// bottom, top        specify the coordinates for the bottom and top horizontal clipping planes
// near and far       specify the distances to the nearer and farther depth clipping planes
                      (these values can be negative if plane is to be behind the viewer)

void glOrtho(GLdouble left, GLdouble right,
             GLdouble bottom, GLdouble top,
             GLdouble near, GLdouble far);
```



Camera Specification – Intrinsic parameters

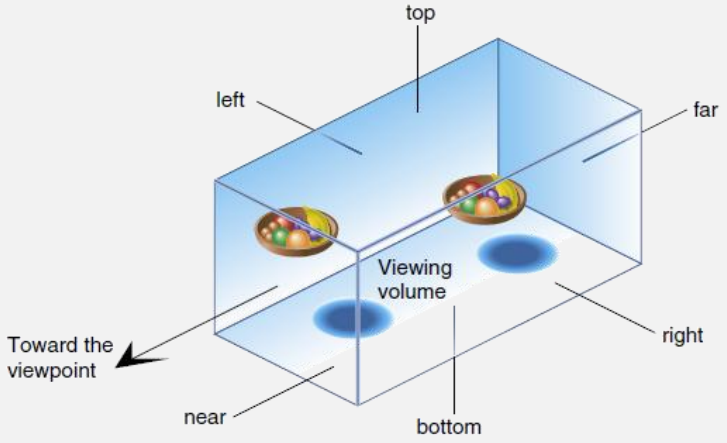
- Example of Perspective projection:
  ~~OpenGL 1.x~~

```
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluPerspective(60.0,          // fovy
               w/(GLdouble)h,// aspect
               0.1, 20.0);    // near,far
```



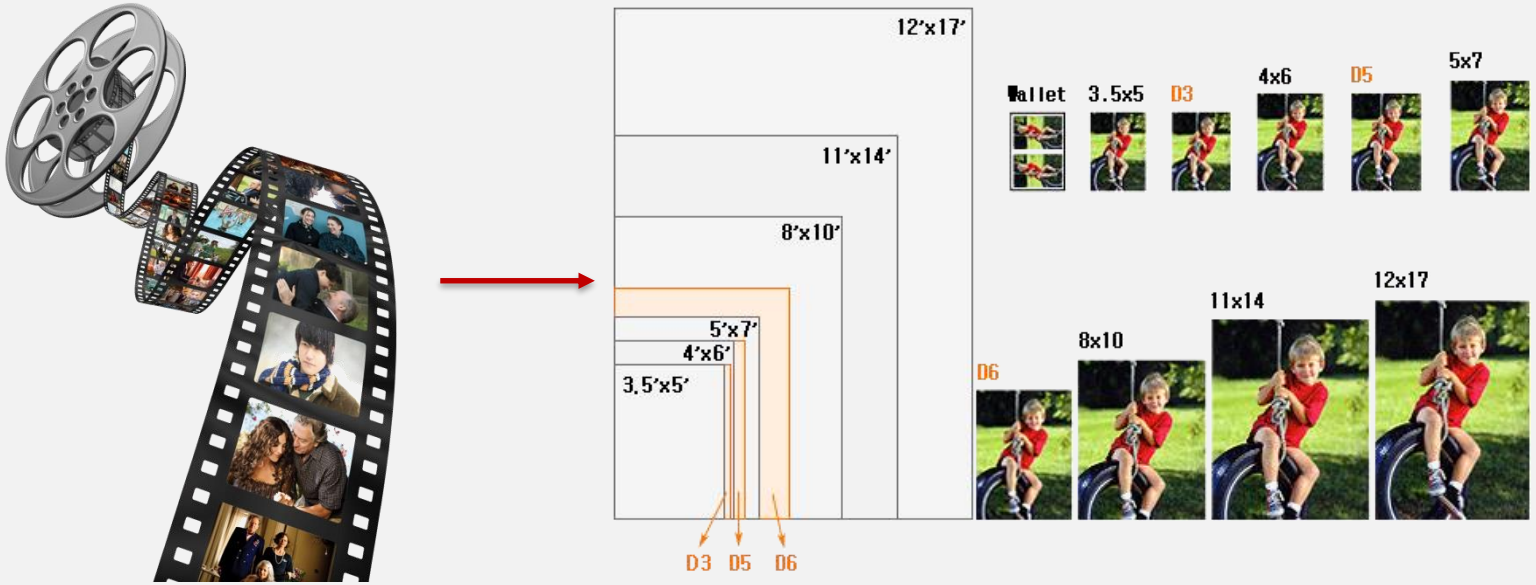
- Example of Orthograpic projection:
  ~~OpenGL 1.x~~

```
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
glOrtho(-1.5, 1.5,    // left, right
        -1.5, 1.5,    // bottom, top
        0.1, 20.0);   // near, far
```



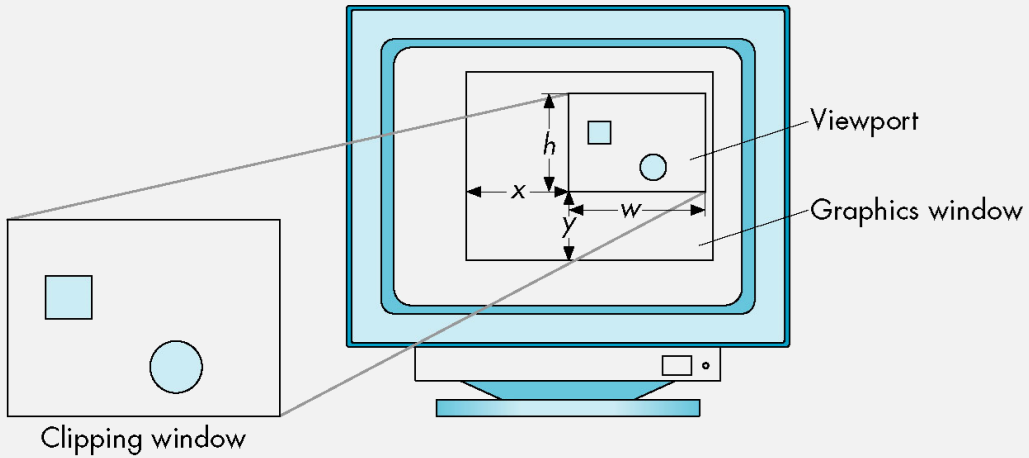
Camera Specification – Viewport

- Viewport
  - Similar to the size of photo printing
    - A film → Photos of different sizes
  - A rectangular area of the display window



Camera Specification – Viewport

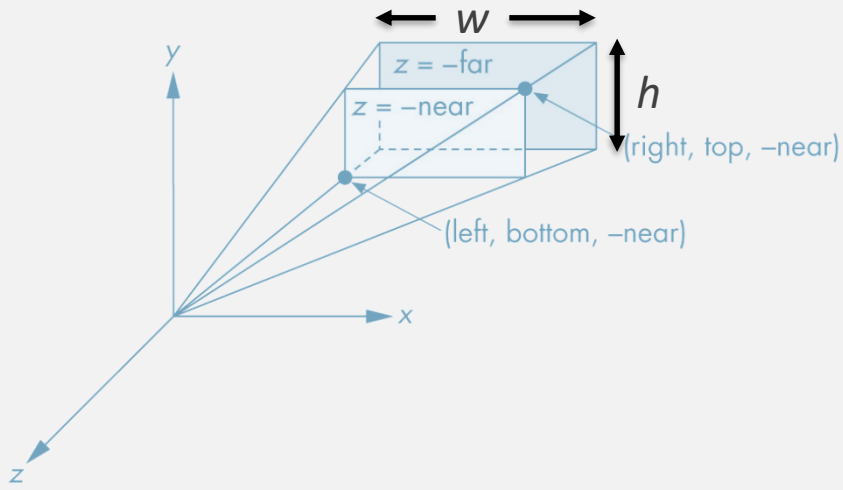
- Viewport
  - Similar to the size of photo printing
    - A film → Photos of different sizes
  - A rectangular area of the display window: x, y, w, h
    - (x, y): the lower-left corner of the viewport
    - w, h: the width and height of the viewport

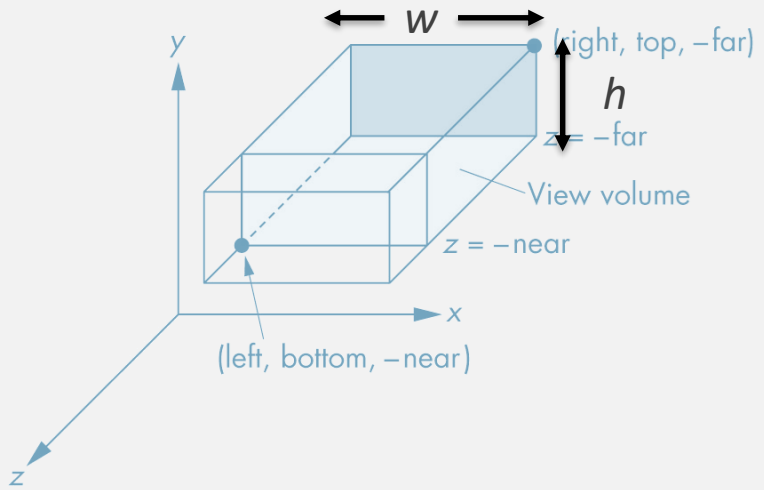


A mapping to the viewport

Camera Specification – Aspect ratio

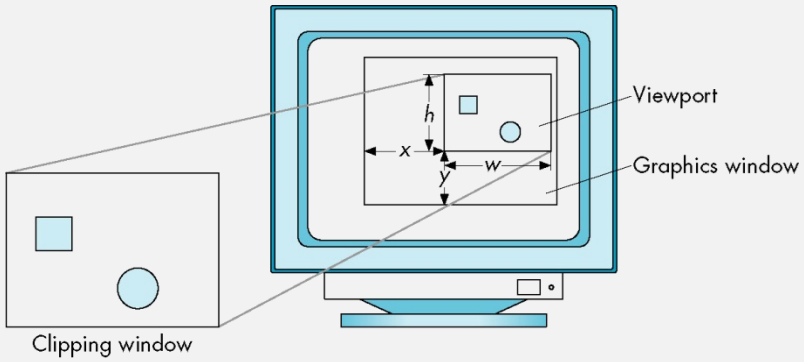
- Aspect ratio
  - width / height
    - For aspect ratio, absolute sizes of width & height are meaningless
    - Aspect ratio of display window (i.e., device screen) is important



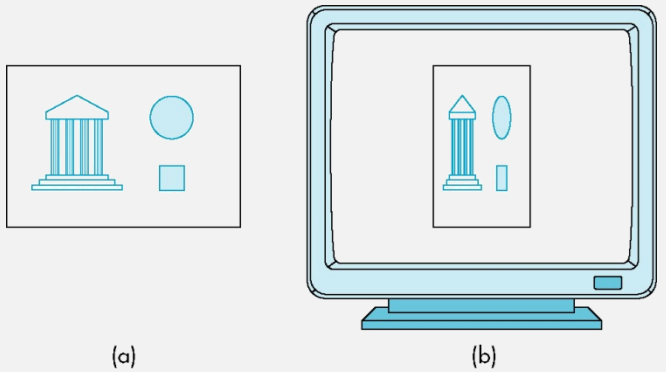


Camera Specification – Aspect ratio

- Aspect ratio
  - width / height
    - For aspect ratio, absolute sizes of width & height are meaningless
    - Aspect ratio of display window (e.g., device screen) is important



A mapping to the viewport



Aspect-ratio mismatch.
(a) viewing rectangle, (b) display window

Summary on Camera Specification – ~~OpenGL 1.x~~

Camera specification

- Viewport
  - Printing the frame buffer onto the screen
- Extrinsic parameters
  - 3D position & orientation (6DOF)
- Intrinsic parameters
  - Projection type: perspective or orthographic
  - Zoom-in / zoom-out: Field of view (FOV)
  - Aspect ratio

~~OpenGL 1.x codes~~

- glViewport()
- ~~glMatrixMode(GL_MODELVIEW)~~
  - ~~Changing extrinsic parameters~~
  - ~~Android: GLU.gluLookAt()~~
  - ~~iPhone: handling extrinsinc parameter of your camera by yourself~~
    - ~~gluLookAt() is officially not supported~~
- ~~glMatrixMode (GL_PROJECTION)~~
  - ~~Changing intrinsic parameters for zoom-in/-out & aspect ratio~~
  - ~~glFrustum()~~
    - ~~Android: GLU.gluPespective():~~
  - ~~glOrtho()~~

Summary on Camera Specification – Modern OpenGL

Camera specification

- Viewport
  - Printing the frame buffer onto the screen
- Extrinsic parameters
  - 3D position & orientation (6DOF)
- Intrinsic parameters
  - Projection type: perspective or orthographic
  - Zoom-in / zoom-out: Field of view (FOV)
  - Aspect ratio

Modern OpenGL codes

- glViewport()
- View Matrix (4x4 matrix)
  - **Generate 4x4 matrix by yourself**, similar to gluLookAt(), which explains the extrinsic parameters of your camera
- Projection Matrix (4x4 matrix)
  - **Generate 4x4 matrix by yourself**, similar to gluPerspective() or glOrtho(), which explains the intrinsic parameters of your camera

Demo with Cinematic Techniques

- Changing extrinsic parameters
  - Camera movements
    - 3D position: Truck / Pedestal / Dolly
    - Orientation: Tilt / Pan / Roll
- Changing intrinsic parameters
  - Camera settings
    - Projection type
    - Zoom-in/-out
    - Aspect ratio
- Viewport

Demo

- OpenGL demo for camera movements
  - http://www.songho.ca/opengl/gl_transform.html