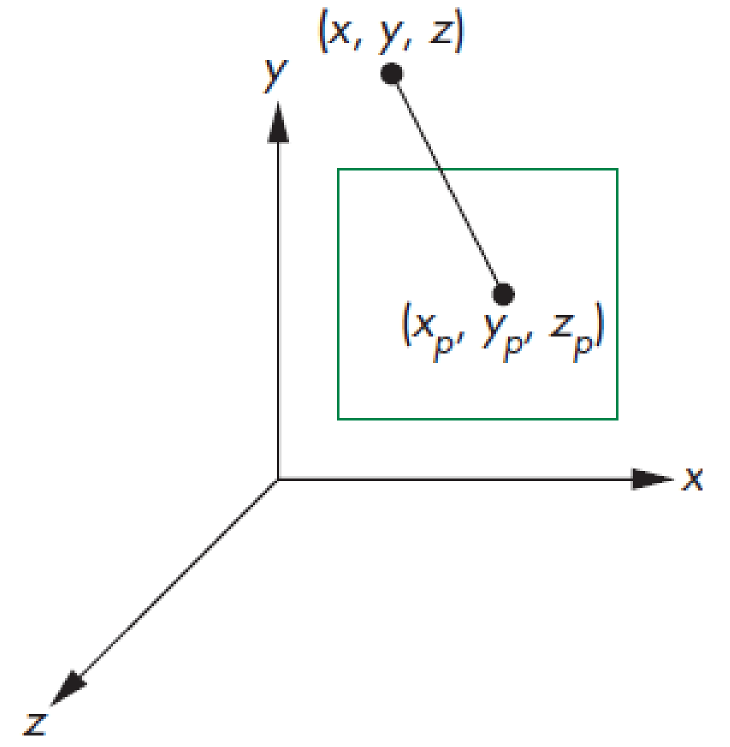
COSC 414/519I: Computer Graphics

2023W2

Shan Du

Parallel Projections

- Oblique Projections
  - An oblique projection can be characterized by the angle that the projectors make with the projection plane.



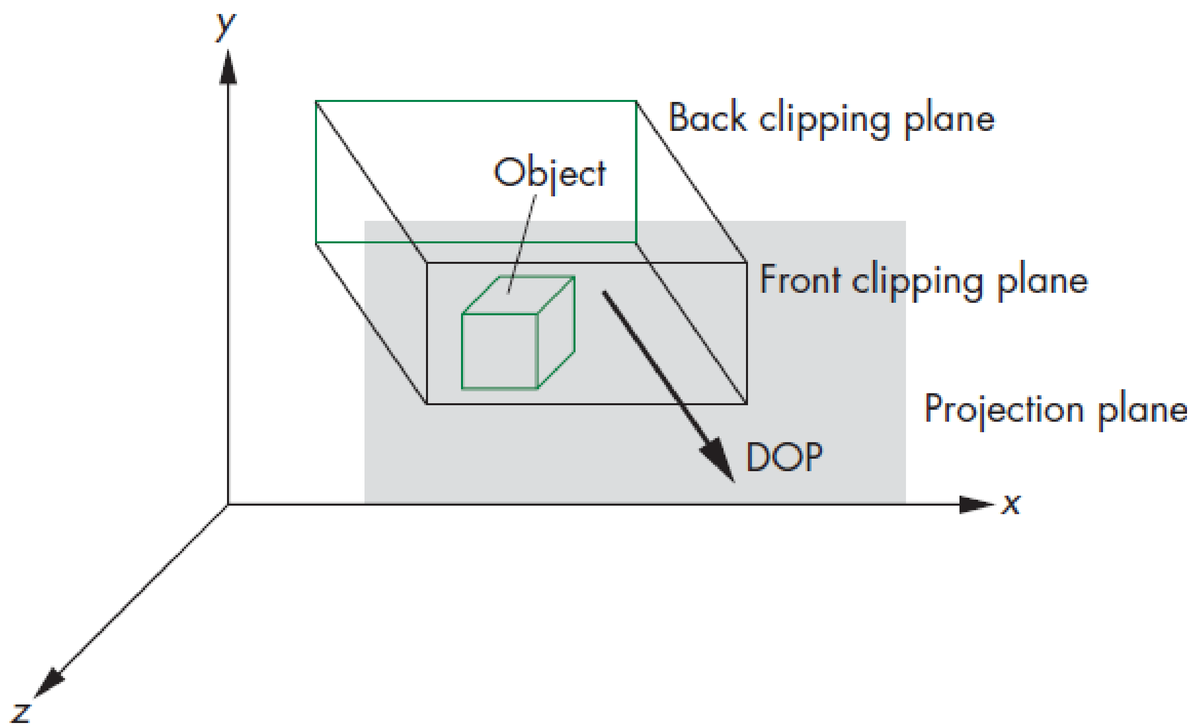
Oblique projection.

Parallel Projections

- The view volume for an oblique projection has the *near* and *far* clipping planes parallel to the view plane, and the right, left, top, and bottom planes parallel to the

direction of

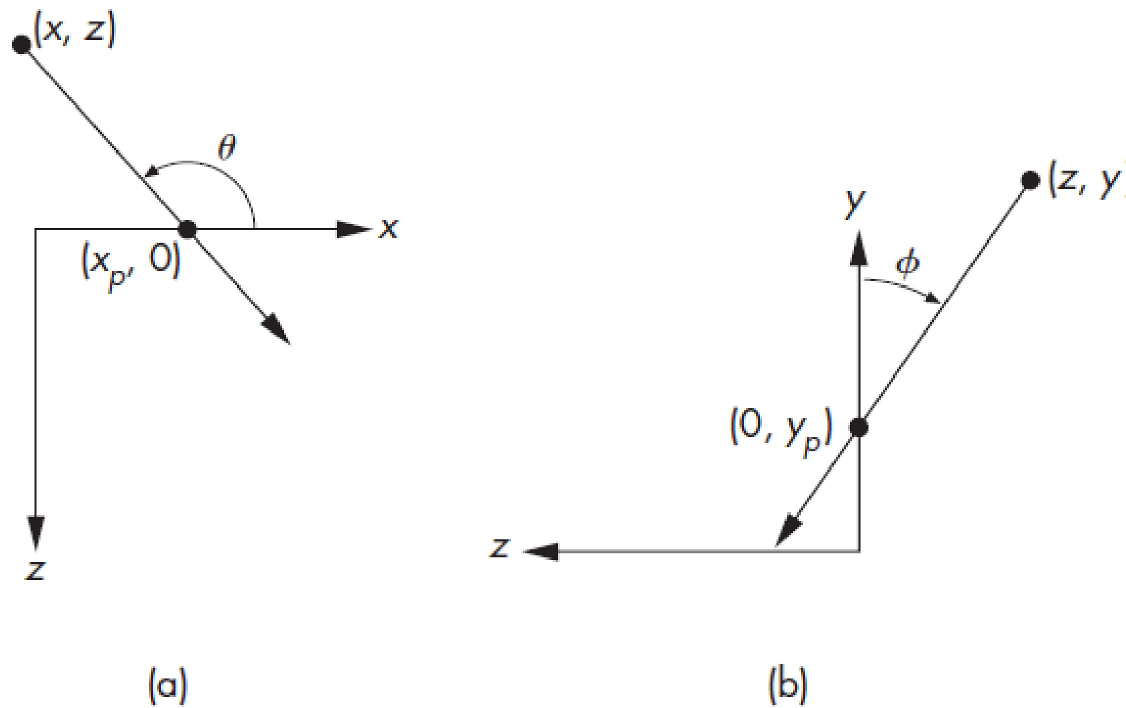
projection.



Oblique clipping volume.

Parallel Projections

- We can derive the equations by considering the top and side views. The angle θ and $\emptyset$ characterize the degree of obliqueness.



Oblique projection. (a) Top view. (b) Side view.

Parallel Projections

- We can find $x_p = x + zcot\theta$ and $y_p = y + zcot\emptyset$, and $z_p = 0$.
- The projection matrix is

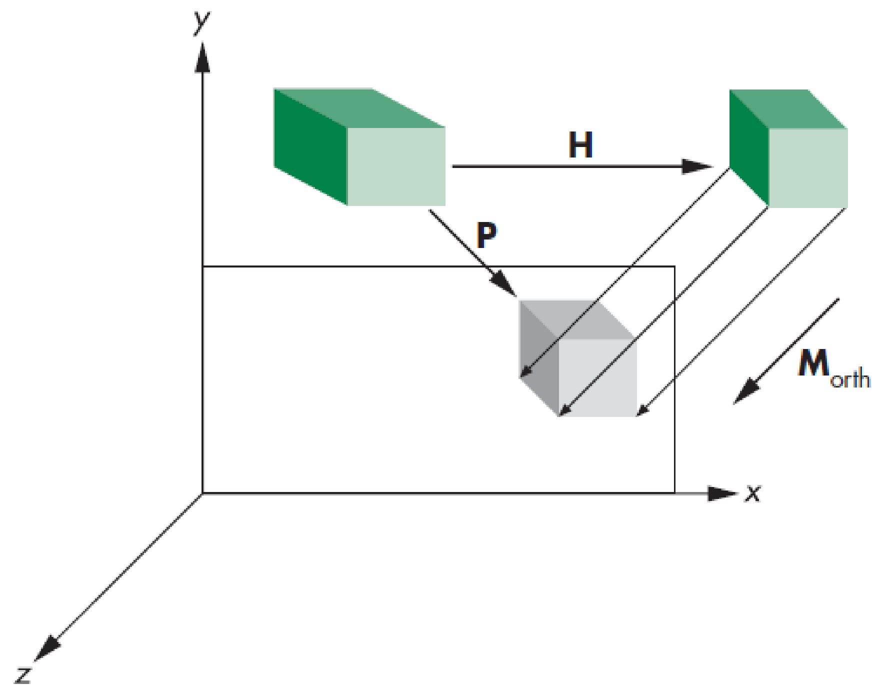
$$\mathbf{P} = \begin{bmatrix} 1 & 0 & \cot\theta & 0 \\ 0 & 1 & \cot\phi & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Break P into the product:

$$\mathbf{P} = \mathbf{M}_{\text{orth}}\mathbf{H}(\theta, \phi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & \cot\theta & 0 \\ 0 & 1 & \cot\phi & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Parallel Projections

- We can implement an oblique projection by first doing a shear of the objects by $H(\theta, \emptyset)$ and then doing an orthographic projection.



Effect of shear transformation.

Parallel Projections

- The orthographic projection of the distorted cube is identical to the oblique projection of the undistorted cube.
- The view volume created by the shear is not our canonical view volume. We have to apply the same scaling and translation matrices. Hence, the transformation

Parallel Projections

$$\mathbf{ST} = \begin{bmatrix} \frac{2}{right-left} & 0 & 0 & -\frac{right+left}{right-left} \\ 0 & \frac{2}{top-bottom} & 0 & -\frac{top+bottom}{top-bottom} \\ 0 & 0 & -\frac{2}{far-near} & -\frac{far+near}{near-far} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

must be inserted after the shear and before the final orthographic projection, so the final matrix is $N = M_{orth}STH$.

Perspective Projections

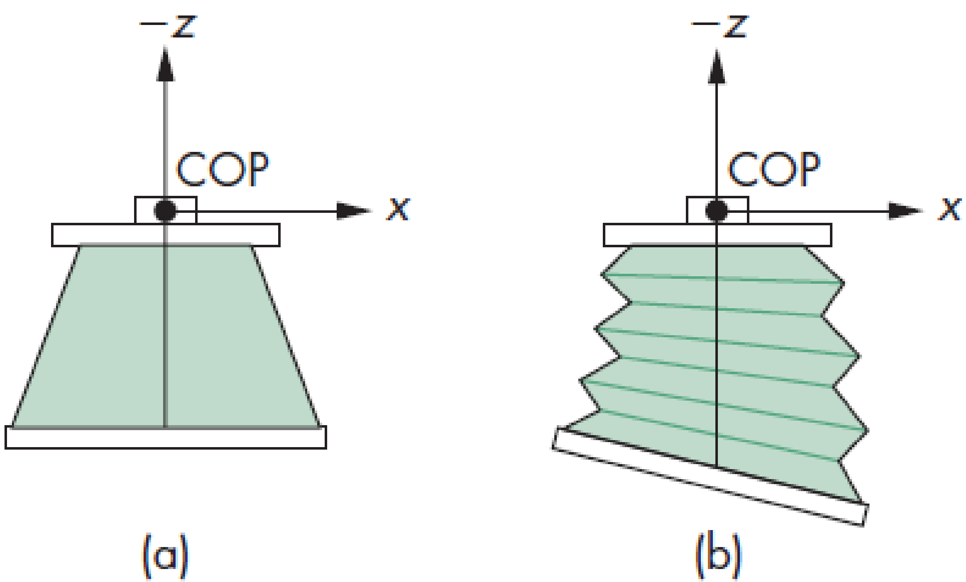
- As with parallel projections, we will separate perspective viewing into two parts: the positioning of the camera and the projection.
- Positioning will be done the same way and we can use *lookAt* function.
- The projection part is equivalent to selecting a lens for the camera.
- With a physical camera, a wide-angle lens gives the most dramatic perspectives, with objects near the camera appearing large compared to objects far from the lens. A telephoto lens gives an image that appears flat and is close to a parallel view.

Perspective Projections

- Simple Perspective Projections
  - Suppose that we are in the camera frame with the camera located at the origin, pointed in the negative z direction.
  - The back of the camera can be orthogonal to the z direction and be parallel to the lens or can have any orientation with respect to the front.

Perspective Projections

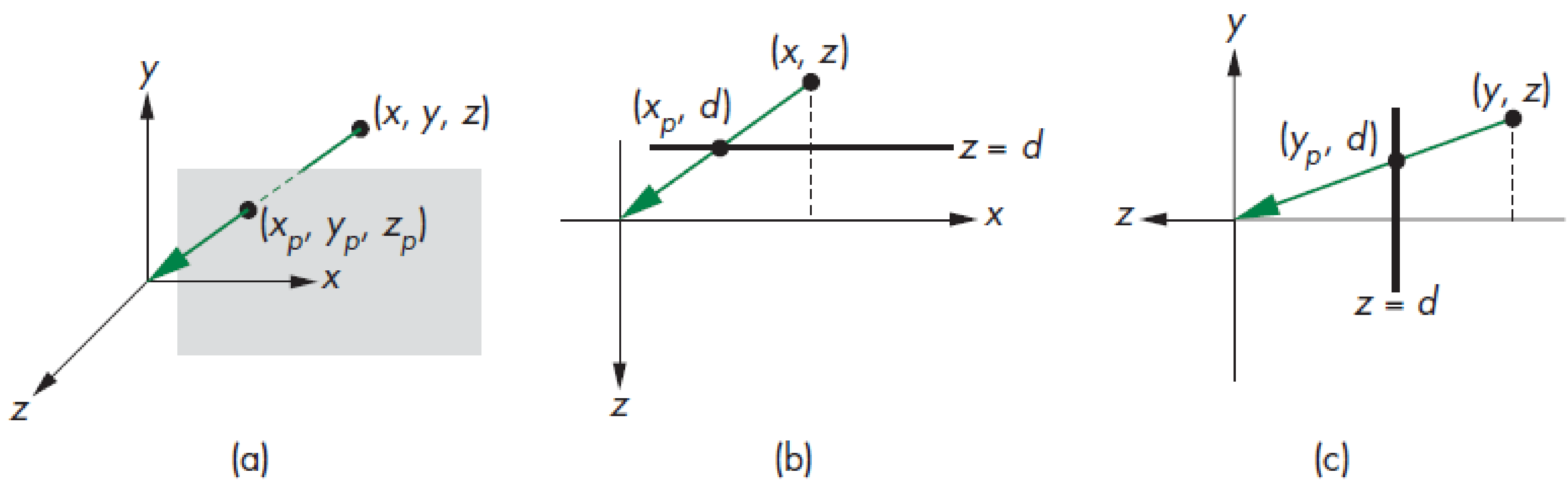
- Simple Perspective Projections



Two cameras. (a) Back parallel to front. (b) Back not parallel to front.

Perspective Projections

- Simple Perspective Projections



Three views of perspective projection. (a) Three-dimensional view. (b) Top view. (c) Side view.

Perspective Projections

- Simple Perspective Projections
  - We place the projection plane in front of the center of the projection.
  - A point in space (x, y, z) is projected along a projector into the point (x_p, y_p, z_p).

Then

$$x_p = \frac{x}{z/d}$$

$$y_p = \frac{y}{z/d}$$

$$z_p = d$$

Perspective Projections

- Simple Perspective Projections
  - Perspective transformation is not invertible since all points along a projector project into the same point, we cannot recover a point from its projection.
  - By using homogeneous coordinates, we represent a point in 3D (x, y, z) by the point $(x, y, z, 1)$ in 4D.
  - Now we replace with (wx, wy, wz, w), as long as $w \neq 0$, we can recover the 3D point by dividing the first three components by w.

Perspective Projections with WebGL

- Simple Perspective Projections
  - In this new form, points in 3D become lines through the origin in 4D.
  - The perspective projection matrix is

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix}$$

Perspective Projections with WebGL

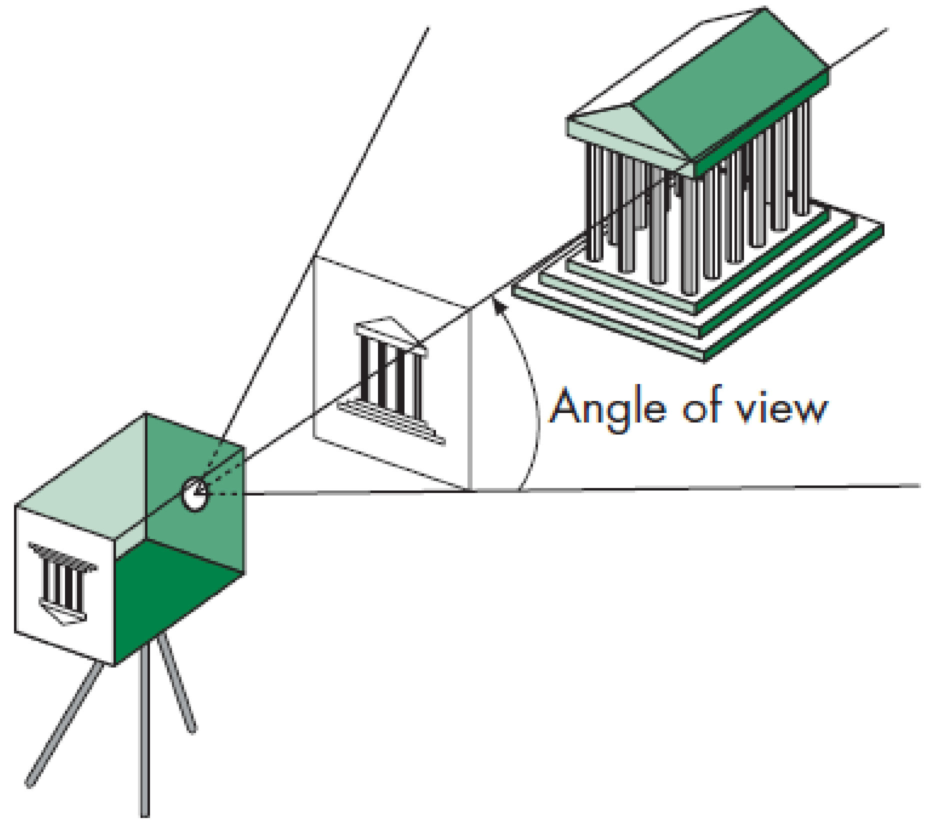
- Simple Perspective Projections

$$p = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad \mathrm{q} = \mathrm{Mp} = \begin{bmatrix} x \\ y \\ z \\ z/d \end{bmatrix} = \begin{bmatrix} \frac{x}{z/d} \\ \frac{y}{z/d} \\ \frac{z}{z/d} \\ \frac{z/d}{z/d} \end{bmatrix} = \begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix}$$

Projection pipeline.

Perspective Projections with WebGL

- In WebGL, only those objects that fit within the angle (or field) of view of the camera appear in the image.



Specification of a view volume.

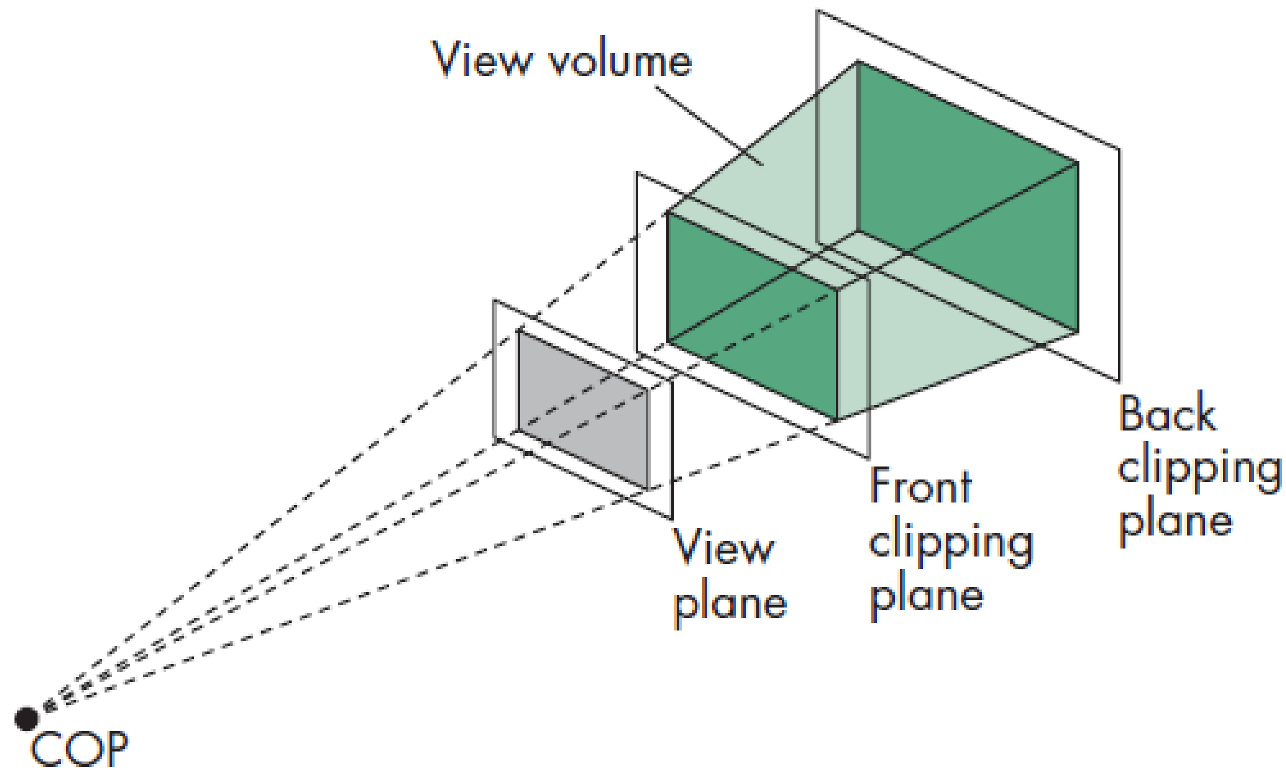
Perspective Projections with WebGL

- If the back of the camera is rectangular, only objects within an infinite pyramid – the view volume – whose apex is at the COP can appear in the image.
- Objects not within the view volume are said to be clipped out of the scene.
- Thus, we need to include the effects of clipping.

Perspective Projections with WebGL

- The application program specifies clipping parameters through the specification of a projection.
- The infinite pyramid becomes a finite clipping volume by adding front and back clipping plane, in addition to the angle of view.
- The resulting view volume is a frustum – a truncated pyramid.

Perspective Projections with WebGL



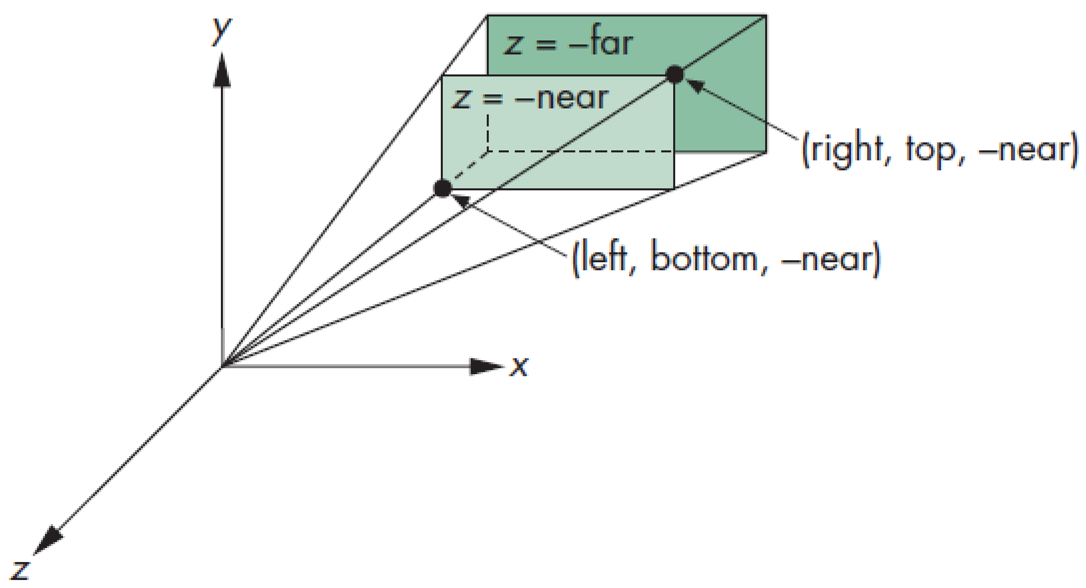
Front and back clipping planes.

Perspective Projections with WebGL

- Perspective functions:

```
frustum = function(left, right, bottom, top, near, far)
```

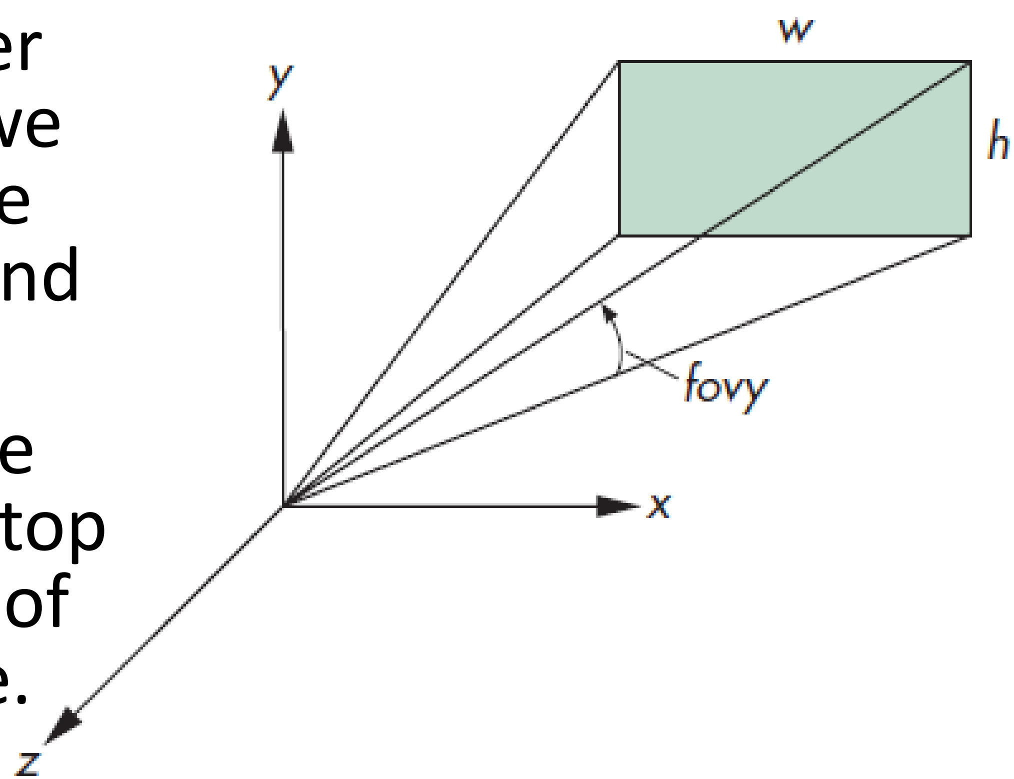
The left, right, top, and bottom values are measured in the near (front clipping) plane.



Specification of a frustum.

Perspective Projections with WebGL

- It is natural to specify the angle of view.
- If the projection plan is rectangular, rather than square, then we see a different angle of view in the top and side views.
- The angle *fovy* is the angle between the top and bottom planes of the clipping volume.




```
perspective = function(fovy, aspect, near, far)
```

Specification using the field of view.

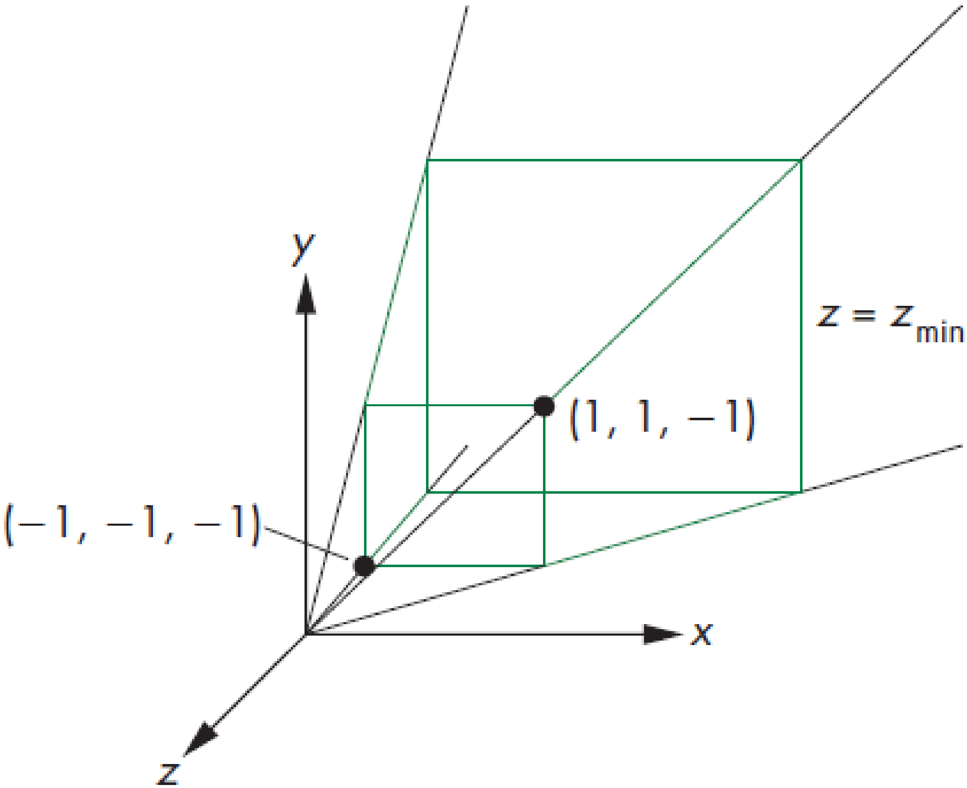
Perspective Projections Matrices

- We find a transformation that allows us, by distorting the vertices of our objects, to do a simple canonical projection to obtain the desired image.
- The transformation is called perspective normalization transformation that converts a perspective projection to an orthogonal projection.

Perspective Projections Matrices

- COP is the origin.
- The projection plane is at *z*=-1.
- The projection matrix is

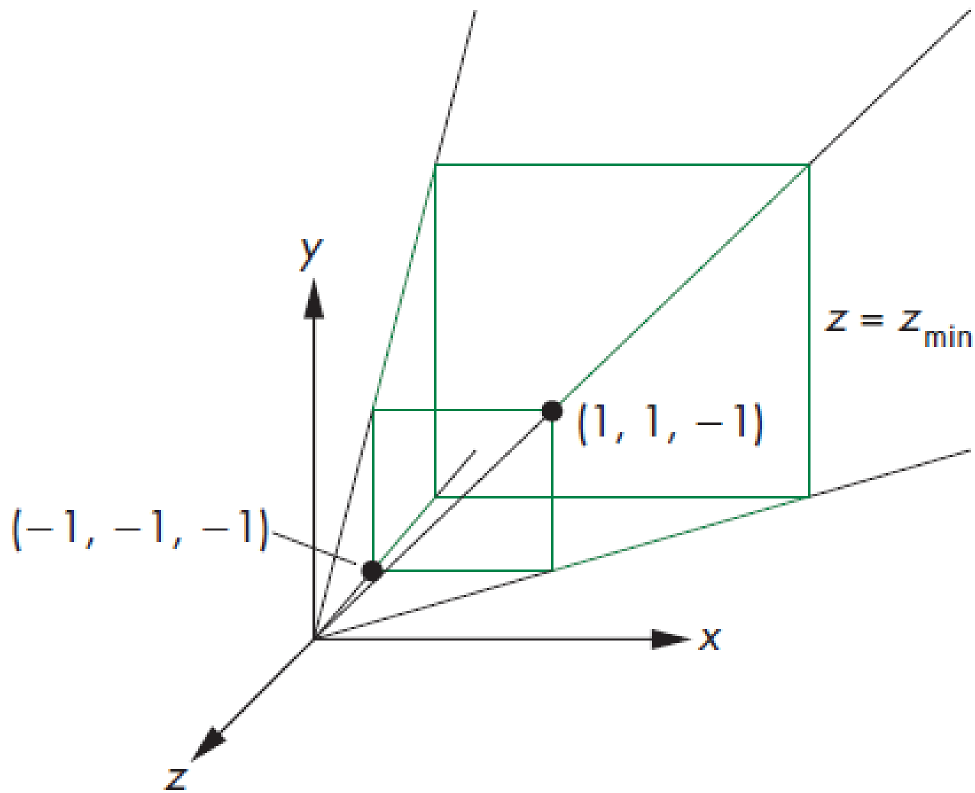
$$M = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix}$$



Simple perspective projection.

Perspective Projections Matrices

- To form an image, we also need to specify a clipping volume.
- Suppose we fix the angle of view at 90 degrees by making the sides of the viewing volume intersect the projection plane at a 45-degree angle.



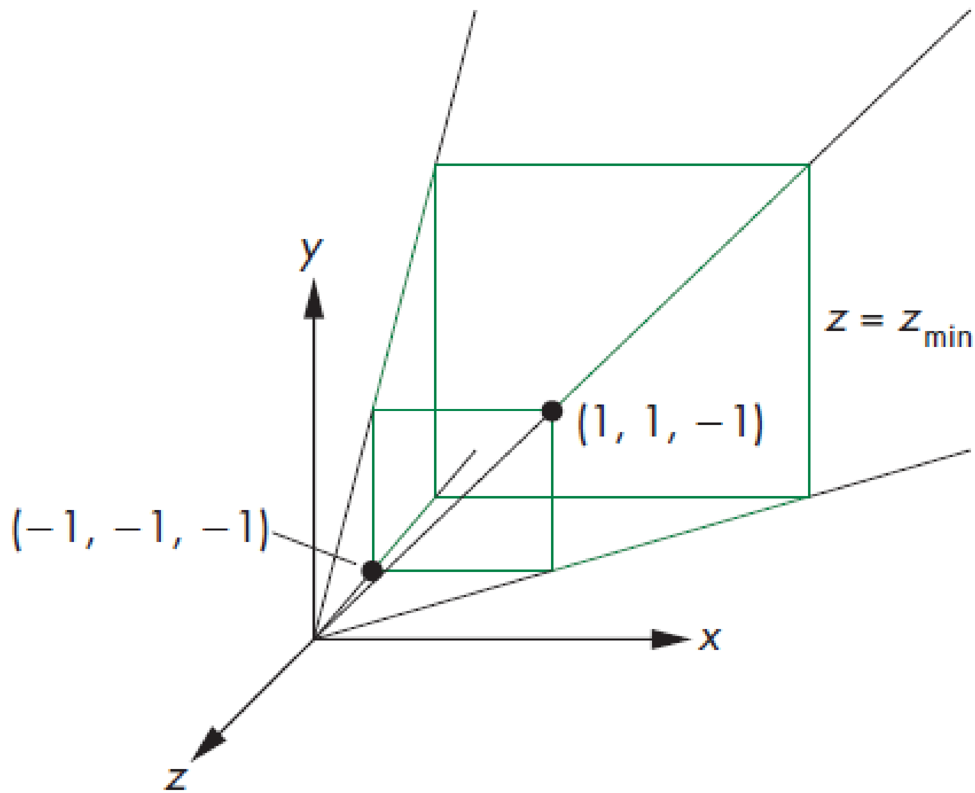
Simple perspective projection.

Perspective Projections Matrices

- Suppose matrix

$$N = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \alpha & \beta \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

we need to find α and β, to convert $\mathrm{p} = [x\ y\ z\ 1]^T$ to $\mathrm{q} = [x'\ y'\ z'\ w']^T$.



Simple perspective projection.

Perspective Projections Matrices

$$x' = x$$
$$y' = y$$
$$z' = \alpha z + \beta$$
$$w' = -z$$

After dividing by w', we have the 3D point

$$x'' = -\frac{x}{z}$$
$$y'' = -\frac{\mathrm{y}}{z}$$
$$z'' = -(\alpha + \frac{\beta}{z})$$

Perspective Projections Matrices

- If we apply an orthogonal projection along the *z*-axis to *N*, we obtain the matrix

$$\mathrm{M}_{\mathrm{orth}}\mathrm{N} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

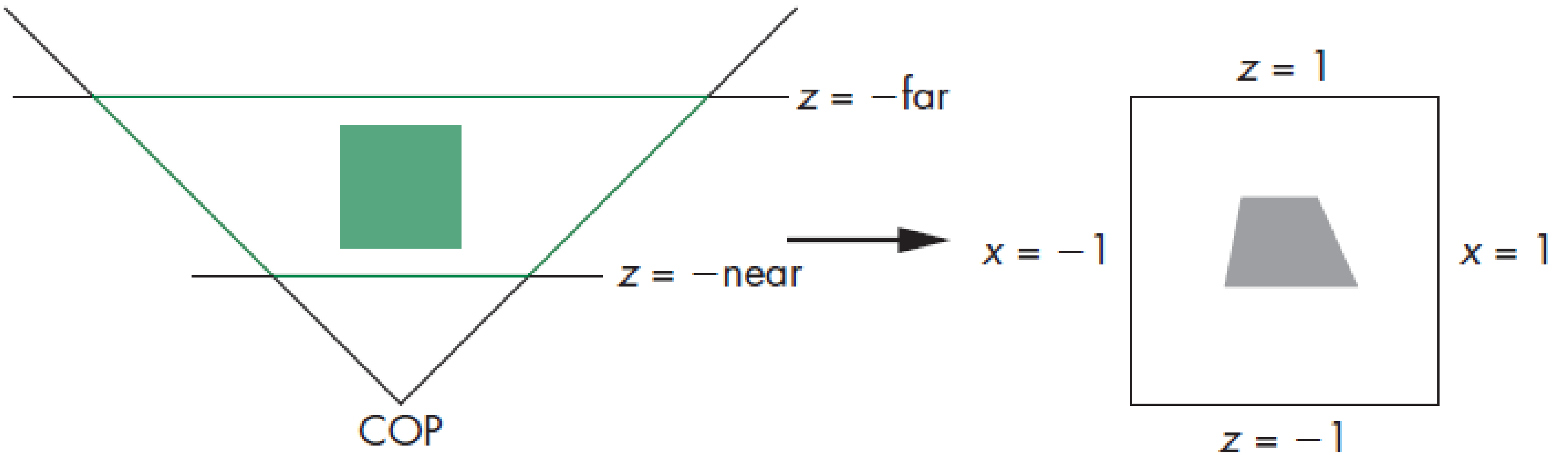
- Then

$$\alpha = -\frac{near + far}{near - far}$$

$$\beta = -\frac{2 * near * far}{near - far}$$

Perspective Projections Matrices

- Perspective Normalization



Perspective normalization of view volume.

Setting the Quadrangular Pyramid Viewing Volume

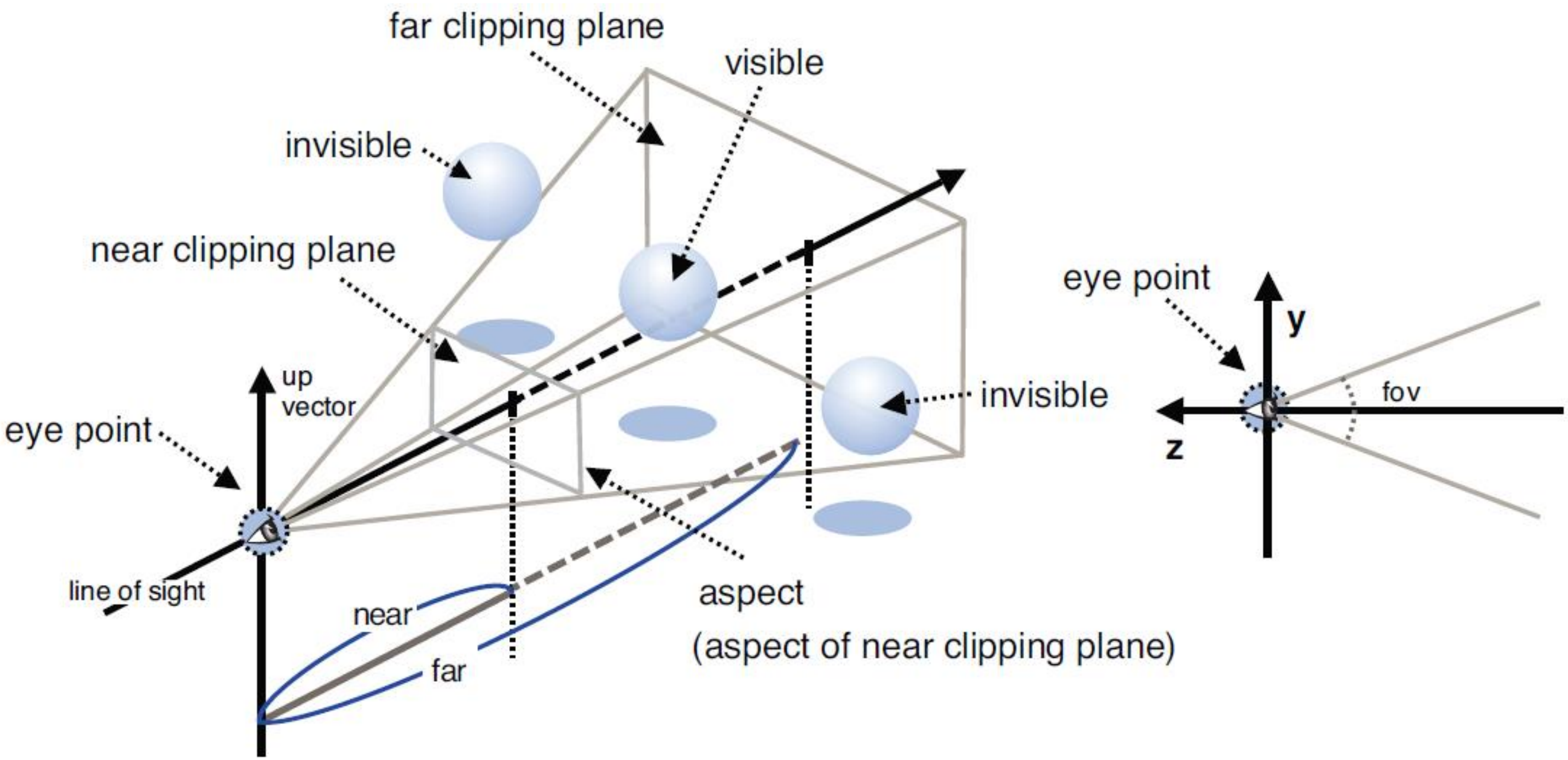


Figure 7.22 Quadrangular pyramid viewing volume

Setting the Quadrangular Pyramid Viewing Volume

`Matrix4.setPerspective(fov, aspect, near, far)`

Calculate the matrix (the perspective projection matrix) that defines the viewing volume specified by its arguments, and store it in `Matrix4`. However, the *near value must be less than the far value.*

Parameters	fov	Specifies field of view, angle formed by the top and bottom planes. It must be greater than 0.
	aspect	Specifies the aspect ratio of the near plane (width/height).
	near, far	Specify the distances to the near and far clipping planes along the line of sight (*near* > 0 and *far* > 0).
Return value	None	