3-D Viewing Continued

Examples of 3-D Viewing

Must first specify the type of projection desired

When use parallel projections?

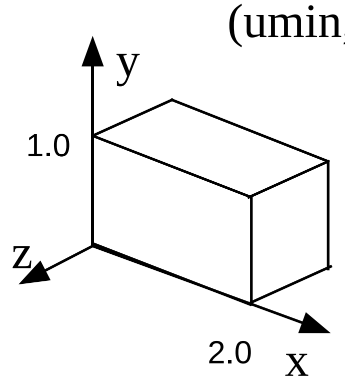
For technical drawings, etc.

When distance between points in the scene are small compared with distance between "camera" and scene

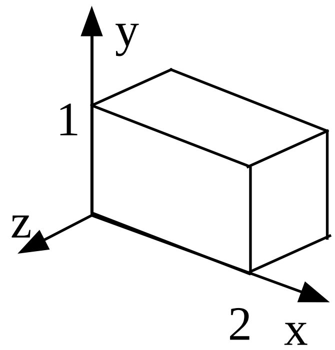
Perspective Projections

Specify the viewing parameters

Viewing Parameter	Value
VRP(WC)	(0, 0, 0)
(defines one point of the VP)	
VPN(WC)	(0, 0, 1)
(now VP is defined)	
VUP(WC)	(0, 1, 0)
(now u,v,n (VRC) coordinate system defined)	
PRP(VRC)	(0.5, 0.5, 1.0)
(defines the center of projection)	
window(VRC)	(-1, 3,-1,3)
	(umin, umax, vmin, vmax)



How get a two- point perspective view of this scene?



Need to have view plane cut two axes (say x and z)

VRP(WC) (2, 0.5, 0)
(to get view plane in "front" of scene)
VPN(WC) (1, 0, 1)
(to get view plane to cut x and z axes)
VUP(WC) (0, 1, 0)
(to specifiy the window orientation)
PRP(VRC) (0, 0, 10)
(to get center of projection)
window(VRC) (-10, 20, -10, 20)
(to specify view volume)

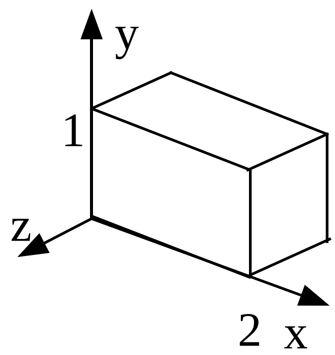
What does projection look like?

How to get a larger image?

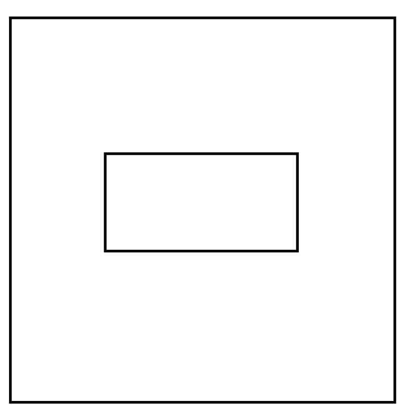
How know how to specify window?
Can do it interactively.

CDQ: What if cut y and z axes?

Example Parallel Projection



VRP(WC) (1, 0.5, 0)
(to get view plane reference point)
VPN(WC) (0, 0, 1)
(to get view plane to cut z axes)
VUP(WC) (0, 1, 0)
(to specifiy the window orientation)
PRP(VRC) (0, 0, 10)
(to projection direction parallel to z axis)
window(VRC) (-2, 2, -2, 2)
(to specify view volume)



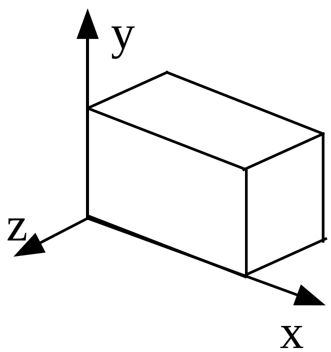
How do we implement projections in our graphics systems?

We'll see we can do it with a 4x4 projection matrix

Look at the basic mathematics of planar projections

Parallel projection

start by assuming VP lies in xy plane

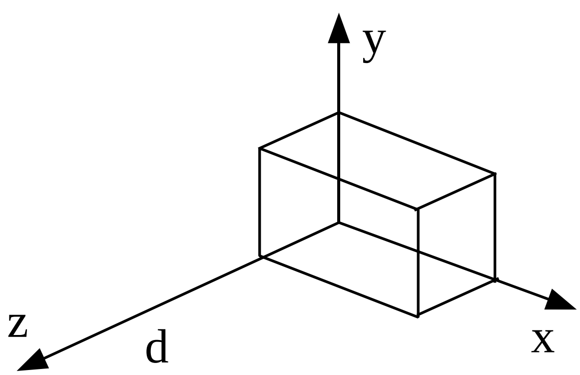


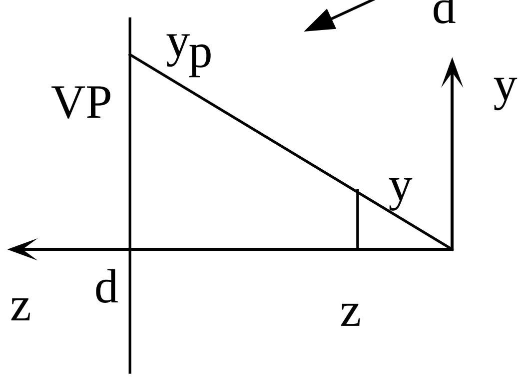
if VRP = (0, 0, 0)
  VPN = (0, 0, 1)
  VUP = (0, 1, 0)
given (x, y, z) --> (xp, yp)
  yp =?
  xp = ?

Perspective projection

start by assuming VP is normal to z axis at z = d

and center of projection = (0, 0, 0)





$y_p = y / (z / d)$

$x_p =$

One-point perspective projection

$y_p = y / (z/d)$

$xp = x / (z/d)$

Thus d is simply a scale factor,
and the division by z gives the foreshortening

How express this as a transformation matrix?

$$\begin{bmatrix} X \\ Y \\ Z \\ W \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$X = x, \; Y = y, \; Z = z, \; W = z/d$

As homogeneous coordinates, divide by W
to get another equal homogeneous point

$x_p = X/W = x / (z/d)$

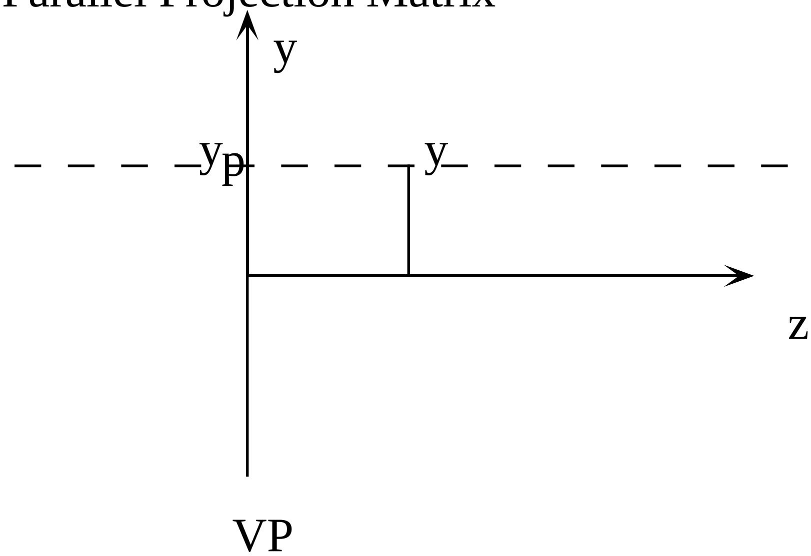
$y_p = y / (z/d)$

$z_p =$

$$\begin{bmatrix} X \\ Y \\ Z \\ W \end{bmatrix} = \begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix}$$

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

Parallel Projection Matrix



$y_p =$

$x_p =$

$z_p =$

$$\begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$M_{ort} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

M_{per} only good for center of projection at origin

M_{ort} only good for VP in xy plane at origin

Class Discussion Question:

How to get two-point perspective from M_{per}?

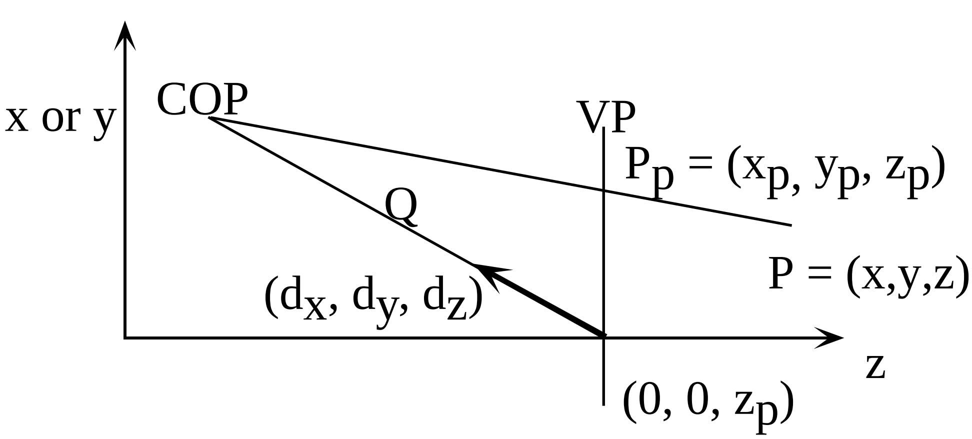
Need to have VP cut two axes (preferably x and z)

How do we need to transform VP?

General Projection Matrix

$$M_{gen} = \begin{bmatrix} 1 & 0 & -(d_x/d_z) & z_p(d_x/d_z) \\ 0 & 1 & -(d_y/d_z) & z_p(d_y/d_z) \\ 0 & 0 & -z_p/(Q\, d_z) & z_p + (z_p^2/(Q\, d_z)) \\ 0 & 0 & -1/(Q\, d_z) & 1 + (z_p/(Q\, d_z)) \end{bmatrix}$$

where Q is the distance from the center of the projection to the point $(0, 0, z_p)$, which is the intersection of the z axis and the projection plane (which is normal to the z axis) and the direction from $(0, 0, z_p)$ to the center of projection is given by the normalized direction vector, (d_x, d_y, d_z)



Implementation of Planar Geometric Projections

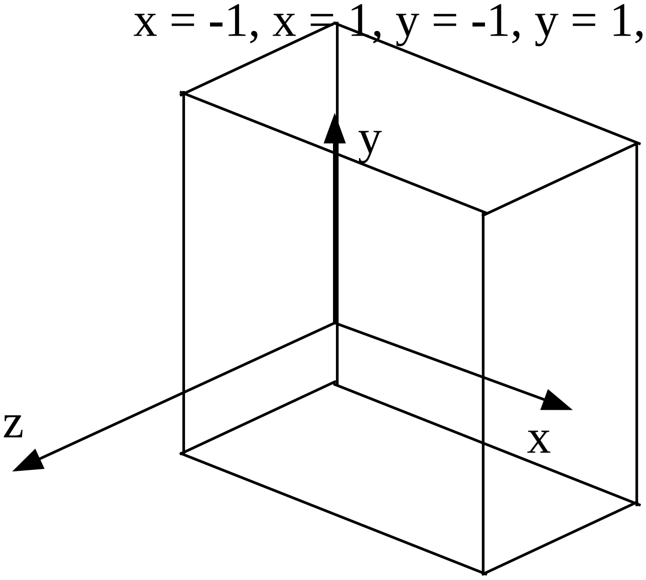
What needs to be done:

Clip scene by view volume
Transform 3-D world coordinates to 2-D device coordinates

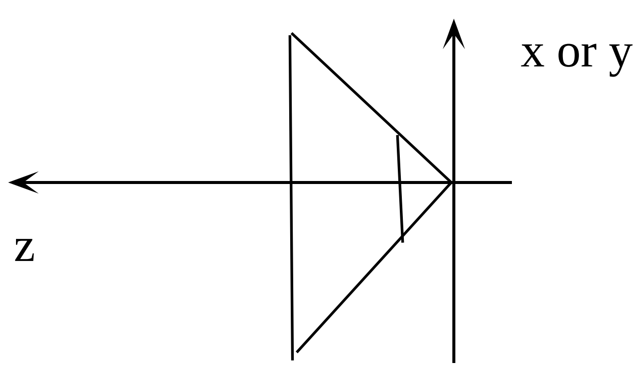
First step is hard, so transform it into an easier problem

Clip scene by easy canonical view volume
Six easy planes

Parallel projections:
$x = -1,\ x = 1,\ y = -1,\ y = 1,\ z = 0,\ z = -1$



Perspective Projection
$x = z,\ x = -z,\ y = z,\ y = -z,\ z = -z_{min},\ z = -1$



Find normalizing transformations , N_{par} and N_{per}, that transform arbitrary view volumes into the canonical ones

So normalize in 3-D

Then clip in 3-D

Then apply simple projection matrices to get 2-D

Then tranform into device coordinates

Which can be composed?

Trade-off of clipping more and not composing all versus simple clipping

Derive N_{par}

Translate VRP to origin

Rotate VRC such that the n axis (VPN) lies on the z axis, the u axis lies on the x axis and the v axis lies on the y axis

Shear such that the DOP is parallel to the z axis

Translate and scale into the canonical view volume

Derive N_{per}

Translate VRP to the origin

Rotate VRC such that n axis lies on z axis, u axis lies on x, and v axis becomes the y axis

Translate such that PRP is at the origin

Shear such that the center line of the view volume becomes the z axis

Scale such that the view volume becomes the canonical perspective view volume

Clipping against the canonical view volumes

Modify the Cohen-Sutherland Clipping algorithm

six bit outcode for each end of a line
eg for parallel cononical view volume
bit 1 = 1 if $y > 1$
bit 2 = 1 if $y < -1$
bit 3 = 1 if $x > 1$
bit 4 = 1 if $x < -1$
bit 5 = 1 if $z < -1$
bit 6 = 1 if $z > 0$
Trivially accept if both outcodes all zeros
Trivially reject if logical and of the codes is not all zeros
Else subdivide line and retest
Use parametric representation of line to compute intersections

Why go through all this complication to get projections?

Can get neat effects by changing just a few parameters:

Move VRP to acheive a "fly-by" or a "walk-through"

"Look around" by changing VPN to face diferent directions

"Tilt your head" by changing VUP