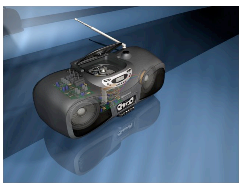
Projection: Completing the Camera Model

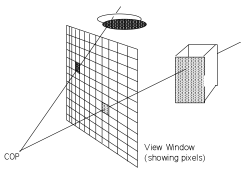
Summary

- Up to now we saw how to create images using ray tracing
- Now we want to speed this up using a number of accelerations
- Today we speed process up but break several properties
- Will spend a few lectures "fixing" problems we generated

Ray Tracing

- Very 'realist' images but not real-time





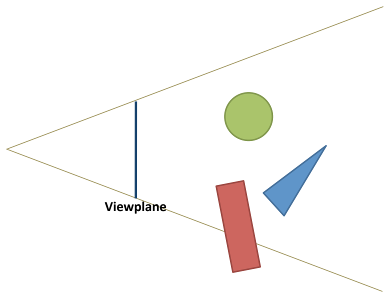
Let's accelerate by simplifying

- Drop the global illumination part
  - i.e no recursion
- Drop the lighting, just ambient
- Assume only polygons
- Instead of tracing rays to each pixel, just trace them to the vertices and fill the space in-between (rasterization)
- Instead of tracing the vertices, project them

Rasterization
(Projection-Based Rendering)

- Although much faster, it creates several new challenges:
  - Projecting the vertices
  - Clipping to the view volume
  - Visible surface determination
  - Rendering a polygon in 2D (rasterization)
  - Lighting
  - Shadows
  - [Global illumination (Radiosity)]

Pipeline

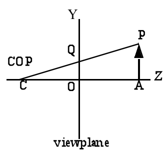


Full camera specification

- We have already seen:
  - VRP, VPN, VUV, COP, view plane window
- Some more parameters:
  - Viewplane Distance
  - Type of projection
    - Perspective – all rays converge to the COP
    - Parallel – parallel rays from points in the scene (DOP)
  - Front and back clip planes
  - View plane window

Perspective Projection

- In order to get a natural looking image we need the perspective
- For a simple arrangement it is easy to find the projection



We know: QO/CO = PA/CA

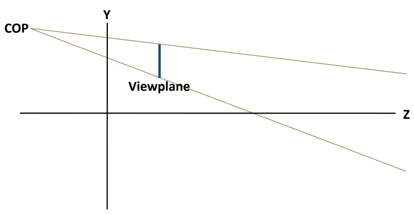
With coordinates (x for example):
Def: x' = QO, x = PA, d = CO, z = AO

x'/d = x/(d+z) → x' = x*d / (d+z)

E.g. for d = 1: x' = x/(z+1)

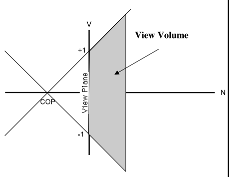
General Cameras

- How to generalize to other cameras?



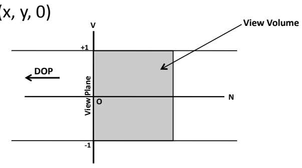
Canonical Frames

- We use *canonical frames* as intermediate stages from which we know how to proceed
- Canonical Frame for Perspective Projection:
  - Cop at (0,0,-1)
  - Viewplane coincident with U-V plane
  - Viewplane window bounded by –1 to +1

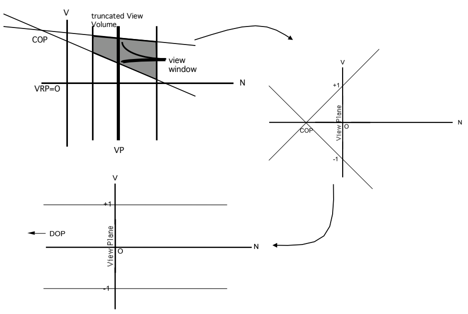


Canonical Frame for Parallel Projection

- Orthographic parallel projection
- Direction of projection (DOP) is (0,0,-1)
- View volume bounded by –1 and +1 on U and V
- And by 0 and 1 on the N axis
- p' = (x, y, 0)



General Perspective to Canonical Parallel

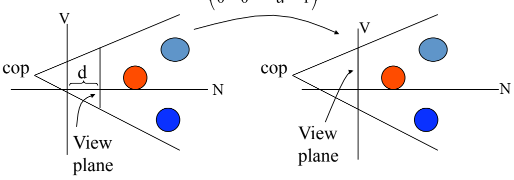


General Perspective to Canonical Perspective

- We will apply a set of transformation matrices (4 in total)
- Each one 'corrects' a particular aspect of the projection
- Then we put them all together to get one matrix

Step 1: move the view plane to the UV plane (n = 0)

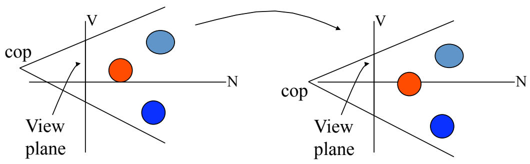
$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -d & 1 \end{pmatrix}$$



Step 2: translate the COP so that it lies on the N axis

cop = (cx,cy,cz)

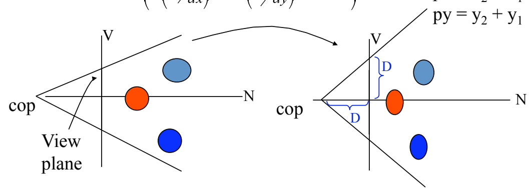
$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -c_x & -c_y & 0 & 1 \end{pmatrix}$$



Step 3: change the view volume into a regular pyramid

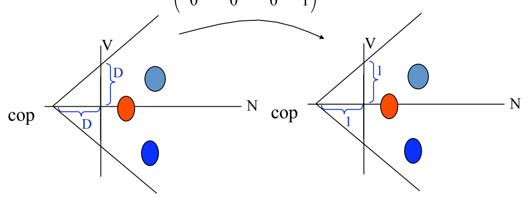
$$\begin{pmatrix} \frac{2D}{dx} & 0 & 0 & 0 \\ 0 & \frac{2D}{dy} & 0 & 0 \\ -\frac{px}{dx} & -\frac{py}{dy} & 1 & 0 \\ -\left(\frac{px}{dx}\right)D & -\left(\frac{py}{dy}\right)D & 0 & 1 \end{pmatrix}$$

Where
$D = d - c_z$
$dx = x_2 - x_1$
$dy = y_2 - y_1$
$px = x_2 + x_1$
$py = y_2 + y_1$



Step 4: Scale by 1/D

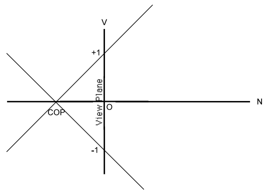
$$\begin{pmatrix} \frac{1}{D} & 0 & 0 & 0 \\ 0 & \frac{1}{D} & 0 & 0 \\ 0 & 0 & \frac{1}{D} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



We multiply all 4 matrices together to get (Q):

$$Q = \begin{pmatrix} \frac{2}{dx} & 0 & 0 & 0 \\ 0 & \frac{2}{dx} & 0 & 0 \\ -\left(\frac{px}{dx}\right)\frac{1}{D} & -\left(\frac{py}{dy}\right)\frac{1}{D} & \frac{1}{D} & 0 \\ -c_x\left(\frac{2}{dx}\right)+\left(\frac{1}{D}\right)\left(\frac{px}{dx}\right)c_z & -c_y\left(\frac{2}{dy}\right)+\left(\frac{1}{D}\right)\left(\frac{py}{dy}\right) & -\left(\frac{d}{D}\right) & 1 \end{pmatrix}$$

Canonical Viewing Space (Canonical Perspective)



We can now compute T = MQ, where

- M maps WC to VC (see lecture on camera)
- Q maps VC to canonical VC

The composite matrix T = MQ

$$\begin{pmatrix} \frac{2u_1D-n_1(px)}{Ddx} & \frac{2v_1D-n_1(py)}{Ddy} & \frac{n_1}{D} & 0 \\ \frac{2u_2D-n_2(px)}{Ddx} & \frac{2v_2D-n_2(py)}{Ddy} & \frac{n_2}{D} & 0 \\ \frac{2u_3D-n_3(px)}{Ddx} & \frac{2v_3D-n_3(py)}{Ddy} & \frac{n_3}{D} & 0 \\ -\frac{2(qu)D-(qn)(px) + 2c_xD-(px)c_z}{Ddx} & -\frac{2(qv)D-(qn)(py) + 2c_yD-(py)c_z}{Ddy} & -\frac{(qn)+d}{D} & 1 \end{pmatrix}$$

Where the symbols mean:

$D = d - c_z$

$x_i = U_i - c_x$ $(i = 1,2)$

$y_i = V_i - c_y$ $(i = 1,2)$

$dx = x_2 - x_1 = U_2 - U_1$

$dy = y_2 - y_1 = V_2 - V_1$

$px = x_1 + x_2 = U_1 + U_2 - 2c_x$

$py = y_1 + y_2 = V_1 + V_2 - 2c_y$

(q_1, q_2, q_3) is the VRP

$$qu = \sum_{i=1}^{3} q_i u_i$$

$$qv = \sum_{i=1}^{3} q_i v_i$$

$$qn = \sum_{i=1}^{3} q_i n_i$$

Canonical Viewing Space to Canonical Parallel

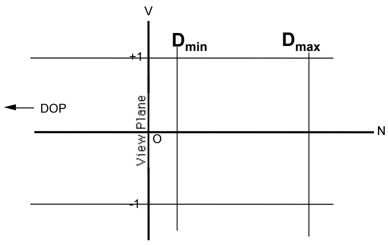
$$P = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Given a point in CVS $(x, y, z, 1)$
- Result in CPS is

$$(x, y, z, z+1) = \left(\frac{x}{z+1}, \frac{y}{z+1}, \frac{z}{z+1}, 1\right)$$

- This is **perspective projection**

Towards Canonical Parallel



Front and Back Clipping Planes

- Actually want Z between 0 and 1
- Replace P with

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{D_{max}+1}{D_{max}-D_{min}} & 1 \\ 0 & 0 & -\frac{D_{min}\cdot(D_{max}+1)}{D_{max}-D_{min}} & 1 \end{pmatrix}$$

Little Exercise

- Given the following camera:
  - COP=(1,0,-1)
  - Viewplane is at d = +1
  - Viewplane boundaries are U=[-2,2] and V=[-2,2]
  - Dmin = 0 and Dmax = +2
- Compute
  - The projection of the point p = (1,0,2) into
    - Canonical viewing space (using matrix Q)
    - Canonical parallel space (using matrix P)

Recap

- Moving away from Ray-Tracing to projection
- Finalised a camera specification and looked at mapping
  - General perspective
  - Canonical perspective
  - Canonical parallel
- We'll spend next couple of weeks tidying up problems!
  - Clipping, lighting, visibility, etc…