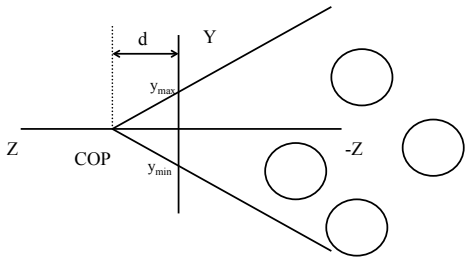
General Camera

Overview

- *Simple camera is limiting and it is necessary to model a camera that can be moved*
- *We will define parameters for a camera in terms of where it "is", the direction it points and the direction it considers to be "up" on the image*

Simple Camera (Cross Section)



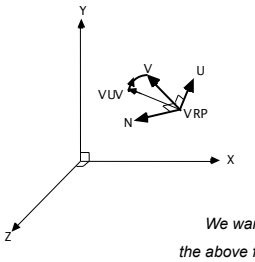
General Camera

- View Reference Point (VRP)
  - where the camera is
- View Plane Normal (VPN)
  - where the camera points
- View Up Vector (VUV)
  - which way is up to the camera
- X (or U-axis) forms LH system

UVN Coordinates

- View Reference Point (VRP)
  - origin of VC system (VC=View Coordinates)
- View Plane Normal (VPN)
  - Z (or N-axis) of VC system
- View Up Vector (VUV)
  - determines Y (or V-axis) of VCS
- X (or U-axis) forms Left Hand system

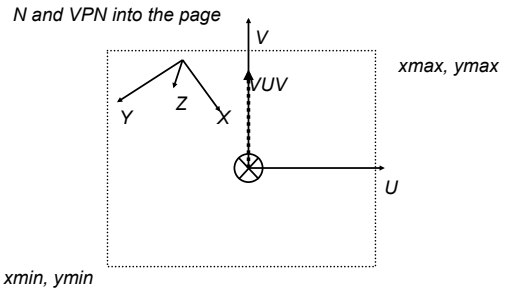
World Coords and Viewing Coords



$$M = \begin{pmatrix} R_1 & R_2 & R_3 & 0 \\ R_4 & R_5 & R_6 & 0 \\ R_7 & R_8 & R_9 & 0 \\ T_1 & T_2 & T_3 & 1 \end{pmatrix} \text{(EQ1)}$$

We want to find a general transform of the above form (EQ1) that will map WC to VC

View from the Camera



Finding the basis vectors

- Step 1 - find n $n = \frac{VPN}{|VPN|}$
- Step 2 - find u $u = \frac{n \times VUV}{|n \times VUV|}$
- Step 3 - find v $v = u \times n$

Finding the Mapping (1)

- u,v,n must rotate under R to i,j,k of viewing space

$$\begin{pmatrix} u \\ v \\ n \end{pmatrix} \begin{pmatrix} \quad \\ R \\ \quad \end{pmatrix} = \begin{pmatrix} \quad \\ I \\ \quad \end{pmatrix}$$

- Both basis are normalised so this is a pure rotation matrix
  - recall in this case $R^T = R^{-1}$

$$R = \begin{pmatrix} u_1 & v_1 & n_1 \\ u_2 & v_2 & n_2 \\ u_3 & v_3 & n_3 \end{pmatrix}$$

Finding the Mapping (2)

- *In uvn system VRP (q) is (0 0 0 1)*
- *And we know from EQ1* $qR + t = 0$ *so*

$$t = -qR$$
$$= -\left(\sum_{i=1}^{3} q_i u_i \;\; \sum_{i=1}^{3} q_i v_i \;\; \sum_{i=1}^{3} q_i n_i \right)$$

Complete Mapping

- *Complete matrix*

$$M = \begin{pmatrix} u_1 & v_1 & n_1 & 0 \\ u_2 & v_2 & n_2 & 0 \\ u_3 & v_3 & n_3 & 0 \\ -\sum_{i=1}^{3} q_i u_i & -\sum_{i=1}^{3} q_i v_i & -\sum_{i=1}^{3} q_i n_i & 1 \end{pmatrix}$$

For you to check

- *If* $M = \begin{pmatrix} R & 0 \\ -qR & 1 \end{pmatrix}$
- *Then* $M^{-1} = \begin{pmatrix} R^T & 0 \\ q & 1 \end{pmatrix}$

Using this for Ray-Casting

- *Use a similar camera configuration (COP is usually, but not always on -n)*
- *To trace object must either*
  - *transform spheres into VC*
  - *transform rays into WC*

Ray-casting

- *Transforming rays into WC*
  - *Transform end-point once*
  - *Find direction vectors through COP as before*
  - *Transform vector by* $\begin{pmatrix} R^T & 0 \\ q & 1 \end{pmatrix}$
  - *Intersect spheres in WC*

Ray-casting

- *Transforming spheres into VC*
  - *Centre of sphere is a point so can be transformed as usual (WC to VC)*
  - *Radius of sphere is unchanged by rotation and translation (and spheres are spheroids if there is a non-symmetric scale)*

Tradeoff

- *If more rays than spheres do the former*
  - *transform spheres into VC*
- *For more complex scenes e.g. with polygons*
  - *transform rays into WC*

Alternative Forms of the Camera

- *Simple "Look At"*
  - *Give a VRP and a target (TP)*
  - *VPN = TP-VRP*
  - *VUV = (0 1 0) (i.e. "up" in WC)*
- *Field of View*
  - *Give horizontal and vertical FOV or one or the other and an aspect ratio*
  - *Calculate viewport and proceed as before*

Animated Cameras

- *Animate VRP (observer-cam)*
- *Animate VPN (look around)*
- *Animate TP (track-cam)*
- *Animate COP*
  - *along VPN - zoom*
  - *orthogonal to VPN - distort*

Recap

- *We created a more general camera which we can use to create views of our scenes from arbitrary positions*
- *Formulation of mapping from WC to VC (and back)*