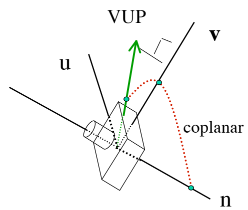
Computing (u,v,n) in world coordinates

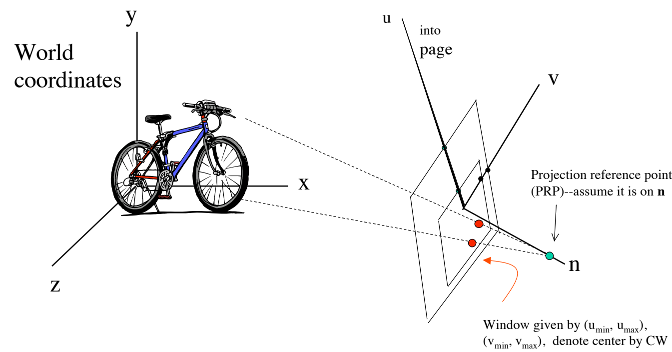
$$\mathbf{u} \parallel \mathrm{VUP} \times \mathbf{n}$$

$$\mathbf{u} = \frac{\mathrm{VUP} \times \mathbf{n}}{\left|\mathrm{VUP} \times \mathbf{n}\right|}$$

$$\mathbf{v} = \mathbf{n} \times \mathbf{u}$$



Specifying a camera



- VRP, VPN, VUP must be in world coords;
- PRP (focal point) could be in world coords, but more commonly, camera coords (which are the same scale as world coords)
- We will use camera coords, and further assume that PRP = (0,0,f).

Object in world coordinates (after modeling transforms)

↓

Transform object from world coordinates to standard camera coordinates

↓

Clip against canonical view frustum

↓

Project using standard camera model ✓

Transform object from world coords to camera coords

↓

Further transform so that frustum is canonical frustum.

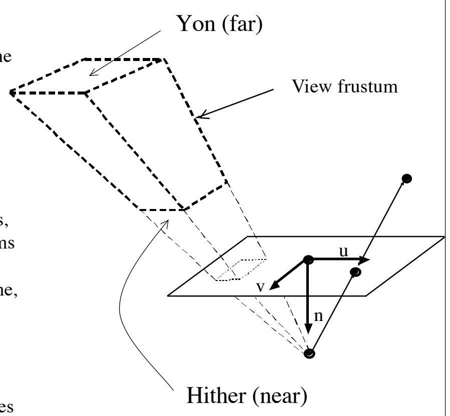
All in H&B chapter 7 "3D viewing"

u and **v** can be used to specify a window in the image plane; only this section of image plane ends up on the screen.

This window defines four planes; points outside these planes are not rendered.

Hither and yon clipping planes, which are always given in terms of camera coordinates, and always parallel to the film plane, give a volume - known as the view frustum.

Orthographic case: - view frustum is cuboid (i.e. all angles right angles, but edges not necessarily of equal length).



We will first map world coordinates to the camera coordinates (top figure)

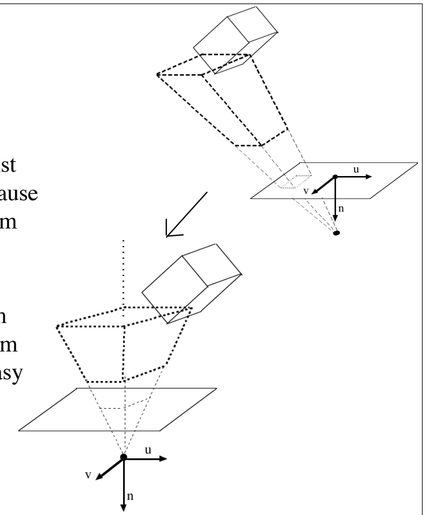
However, if clipping against the frustum is difficult because planes bounding the frustum have a complex form

Solution: further transform frustrum to a canonical form where clip planes are an easy form to deal with:

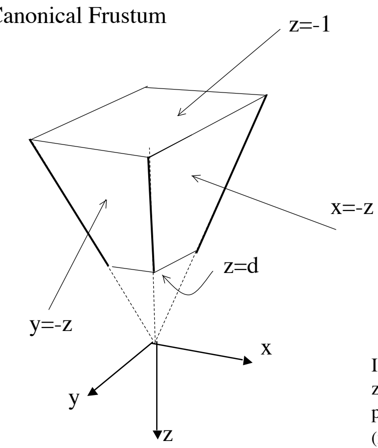
z=x, z=-x

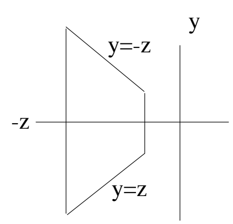
z=y, z=-y

z=-1, z=d



Canonical Frustum





If image plane transforms to z=-f' then in new frame, projection is easy:

(x, y, z) -> (-f'x /z, -f'y/z)

Transform object from world coords to camera coords

Step 1. Translate the camera at VRP to the world origin. Call this T_1.

Translation vector is simply negative VRP.

(We are changing the coordinate system of the world, which is the same thing mathematically as moving the camera. We want object world coordinates to **change** so that the camera location (VRP) **becomes** the origin).

Transform object from world coords to camera coords

Step 2. Rotate camera coordinate frame (in w.c.) so that so that **u** is **x**, **v** is **y**, and **n** is **z**. The matrix is ?

(We are changing the coordinate system of the world, which is the same thing mathematically as moving the camera. We want object world coordinates to **change** so that the camera axis **becomes** the standard axis—e.g, **u** becomes (1,0,0), **v** becomes (0,1,0) and **n** becomes (0,0,1)).

Transform object from world coords to camera coords

Step 2. Rotate camera coordinate frame (in w.c.) so that so that **u** is **x**, **v** is **y**, and **n** is **z**. The matrix is:

$$\begin{vmatrix} \mathbf{u}^T & & & 0 \\ \mathbf{v}^T & & & 0 \\ \mathbf{n}^T & & & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

(why?)

Transform object from world coords to camera coords

$$\begin{vmatrix} \mathbf{u}^T & & & 0 \\ \mathbf{v}^T & & & 0 \\ \mathbf{n}^T & & & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \mathbf{u} = \begin{vmatrix} 1 \\ 0 \\ 0 \\ 0 \end{vmatrix}$$

In the current coords (world shifted so that VPR is at origin): **u** maps into the X-axis unit vector (1,0,0,0) which is what we want.

(Similarly, **v**-->Y-axis unit vector, **n**-->Z-axis unit vector)

Object in world coordinates (after modeling transforms)

↓

Transform object from world coordinates to standard camera coordinates

↓

Clip against canonical view frustum

↓

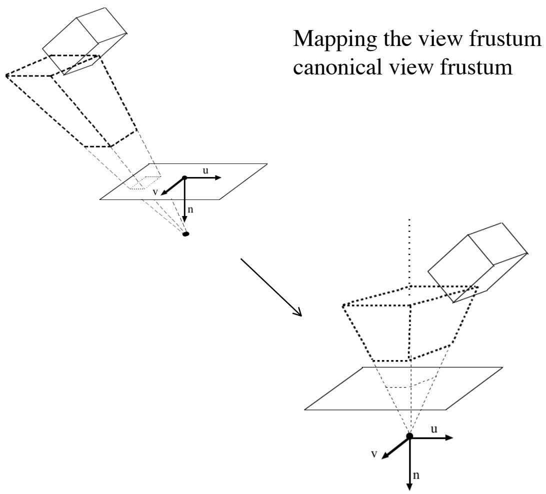
Project using standard camera model ✓

Transform object from world coords to camera coords ✓

↓

Further transform so that frustum is canonical frustum.

Mapping the view frustum to the canonical view frustum

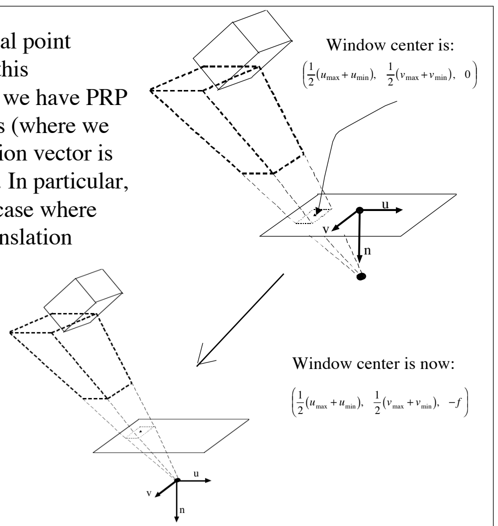


Further transform so that frustum is canonical frustum.

Since we are now in camera coordinates, we will often refer to them as (x,y,z) not (u,v,n).

1. Translate focal point to origin
2. Shear so that central axis of frustum lies along the z axis
3. Scale x, y so that faces of frustum lie on conical planes
4. Isotropic scale so that back clipping plane lies at z=-1

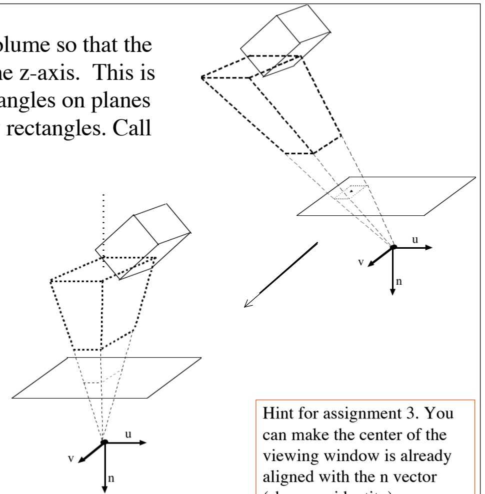
Step 1: Translate focal point (PRP) to origin; call this translation T_2. Since we have PRP in camera coordinates (where we now are), the translation vector is simply negative PRP. In particular, in the very common case where PRP is (0,0,f), the translation vector is (0,0,-f)



Step 1 is relatively straightforward, but notice that the location of the clipping planes also gets shifted.

So, before we had the back clipping plane at B (which is negative). Now it is at: B-f.

Step 2: Shear this volume so that the central axis lies on the z-axis. This is a shear, because rectangles on planes z=constant must stay rectangles. Call this shear S_1



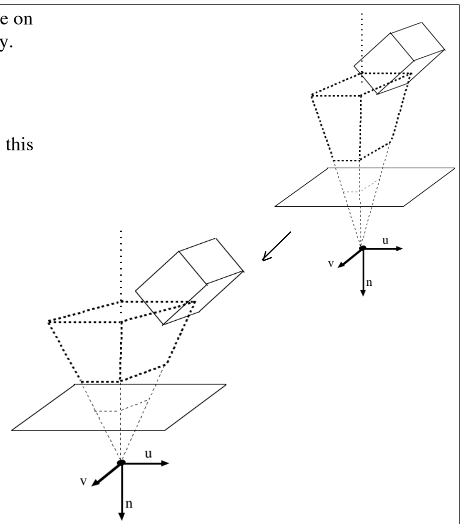
Hint for assignment 3. You can make the center of the viewing window is already aligned with the n vector (shear == identity).

Shear S_1 takes previous window midpoint $\left(\frac{1}{2}(u_{max}+u_{min}),\ \frac{1}{2}(v_{max}+v_{min}),\ -f\right)$ to (0, 0, -f) - this means that matrix is:

$$\begin{pmatrix} 1 & 0 & \frac{(u_{min}+u_{max})}{2f} & 0 \\ 0 & 1 & \frac{(v_{min}+v_{max})}{2f} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Note that the size of a rectangle in the image plane does not change.

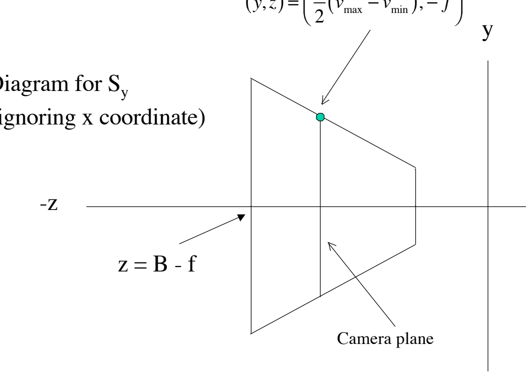
3. Scale x, y so that planes are on z=x, z=-x and z=y and z=-y. Call this scale Sc_1
4. Isotropic scale so that far clipping plane is z=-1; call this scale Sc_2



3. Scale x, y so that planes are on z=x, z=-x and z=y and z=-y. Call this scale Sc_1

$$(y,z)=\left(\frac{1}{2}(v_{max}-v_{min}),-f\right)$$

Diagram for S_y
(ignoring x coordinate)



4. Scale x, y so that planes are on z=x, z=-x and z=y and z=-y. Call this scale Sc_1

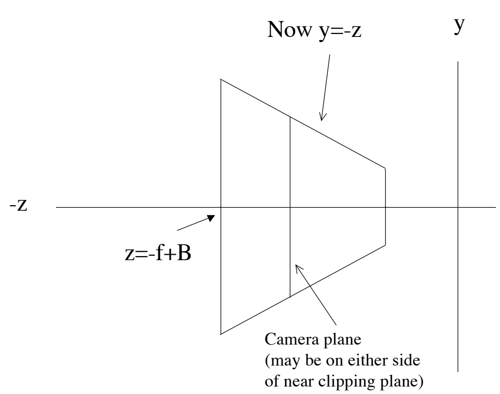
$$\left(\frac{1}{2}(v_{\max} - v_{\min}),\ -f\right) \Longrightarrow (f, -f) \quad \text{(because } y=-z\text{)}$$

$$k_y \frac{1}{2}(v_{\max} - v_{\min}) = f$$

$$k_y = \frac{2f}{(v_{\max} - v_{\min})} \qquad (k_y \text{ is y scale factor})$$

$$\mathbf{Sc}_1 = \begin{vmatrix} \frac{2f}{(u_{\max} - u_{\min})} & 0 & 0 & 0 \\ 0 & \frac{2f}{(v_{\max} - v_{\min})} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

5. Now isotropic scale so that far clipping plane is z=-1; call this scale Sc_2



5. Now isotropic scale so that far clipping plane is z=-1; call this scale Sc_2

Currently, at far clipping plane, z=-f+B

Want a factor k so that k(-f+B)=-1

So, k = -1 / (-f + B) = 1 / (f - B)

(Note that B is negative, and k is positive)

$$\mathbf{Sc}_2 = \begin{bmatrix} \frac{1}{f-B} & 0 & 0 & 0 \\ 0 & \frac{1}{f-B} & 0 & 0 \\ 0 & 0 & \frac{1}{f-B} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Note that the focal length, f, also gets transformed (needed for the perspective transformation coming up).

It is: $$f' = \frac{f}{f-B}$$

3D Viewing Pipeline

$$\begin{pmatrix} \text{Point in} \\ \text{canonical} \\ \text{camera} \\ \text{coordinates} \end{pmatrix} \quad Sc_2 Sc_1 S_1 T_2 R_1 T_1 \quad \begin{pmatrix} \text{Point in} \\ \text{world} \\ \text{coordinates} \end{pmatrix}$$

Then clip (more later), and then project