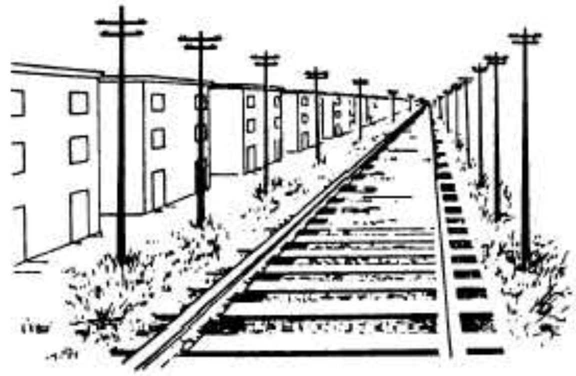
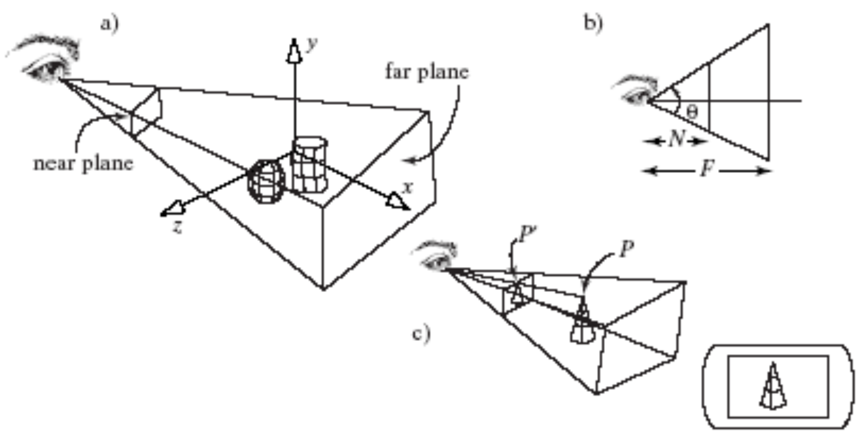
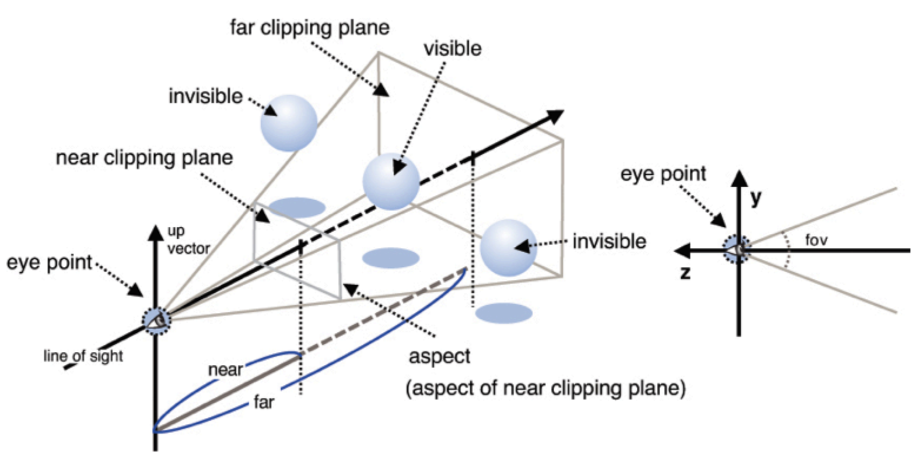
Perspective Projection
Projection from the 3D world on a 2D plane that maintains the appearance of depth.

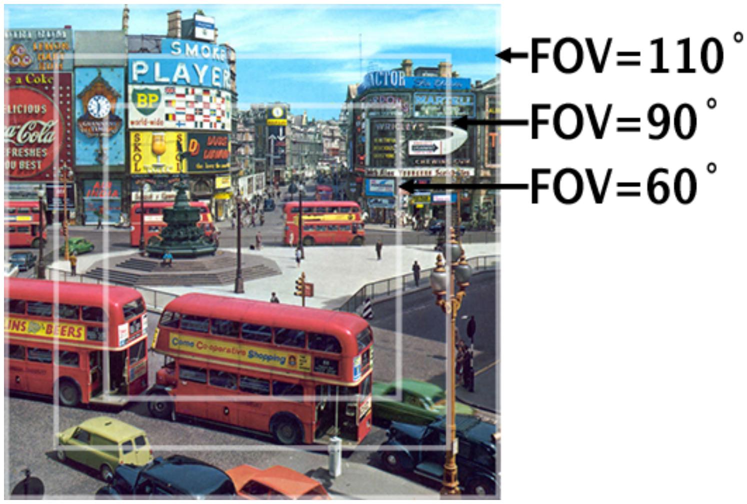


- Define the view volume
  - 2D image projected on the **near plane**
  - Two ways of representing the view volume
    - **frustum(left, right, bottom, top, near, far)**
      - eye as center of view projection (by default)
      - left, right, bottom, top (of the near plane)
      - near, far



    - **perspective(fov, aspect, near, far)**
      aspect = width / height of the viewing volume





- **These two representations are interchangable**
    - frustum → perspective
        - aspect = (right-left)/(top-bottom)
        - viewAngle=2* arctan(1/2*(top-bottom)/N)
    - perspective → frustum
        - top = N*tan(1/2* viewAngle*PI/180*)
        - bottom = -top
        - right = top*aspect
        - left = -right

Practice Question: Given a perspective viewing volume setup with :

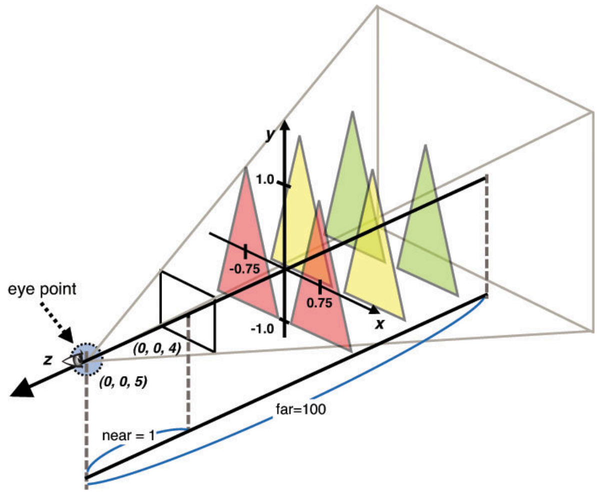
Perspective(60, 1.5, 2, 10);

What is the left and right side values of the near plane of this viewing volume (along x dimension)?

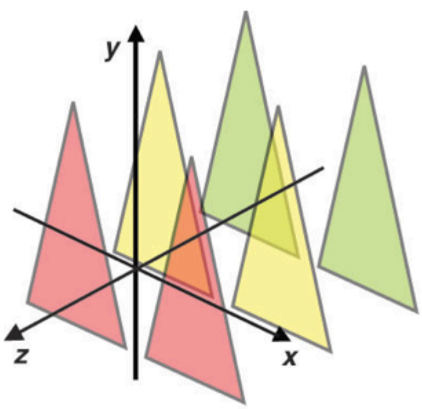
What is the bottom and top sides of the near plane?

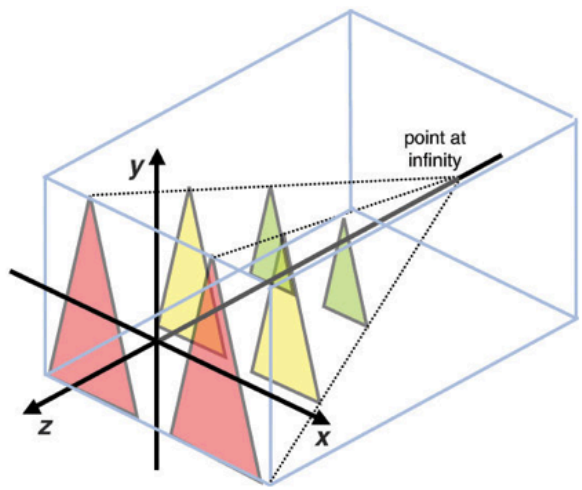
- **How does the perspective projection maintain the depth appearance of 3D objects?**

From this ➔



To this:

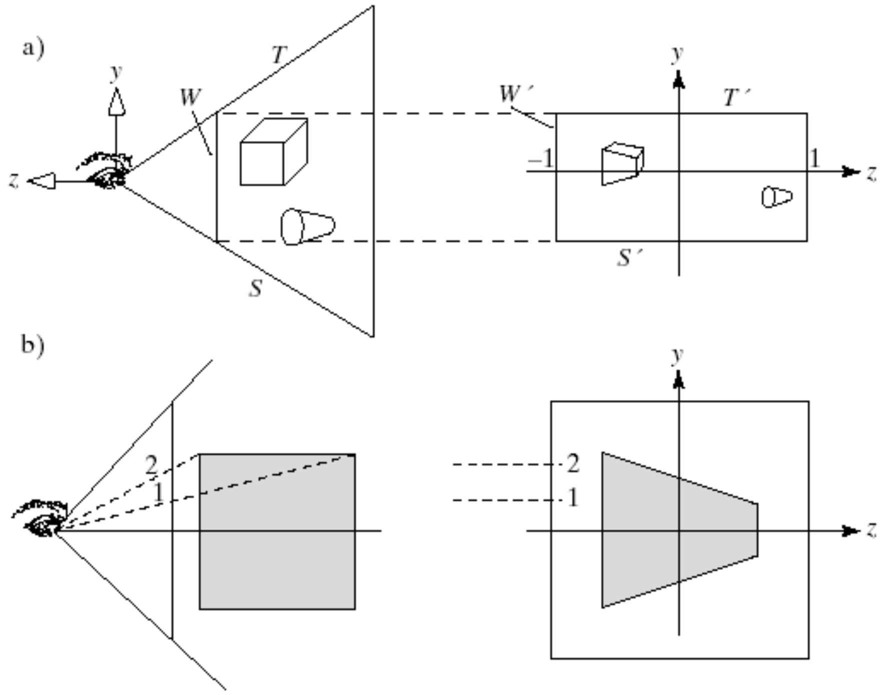




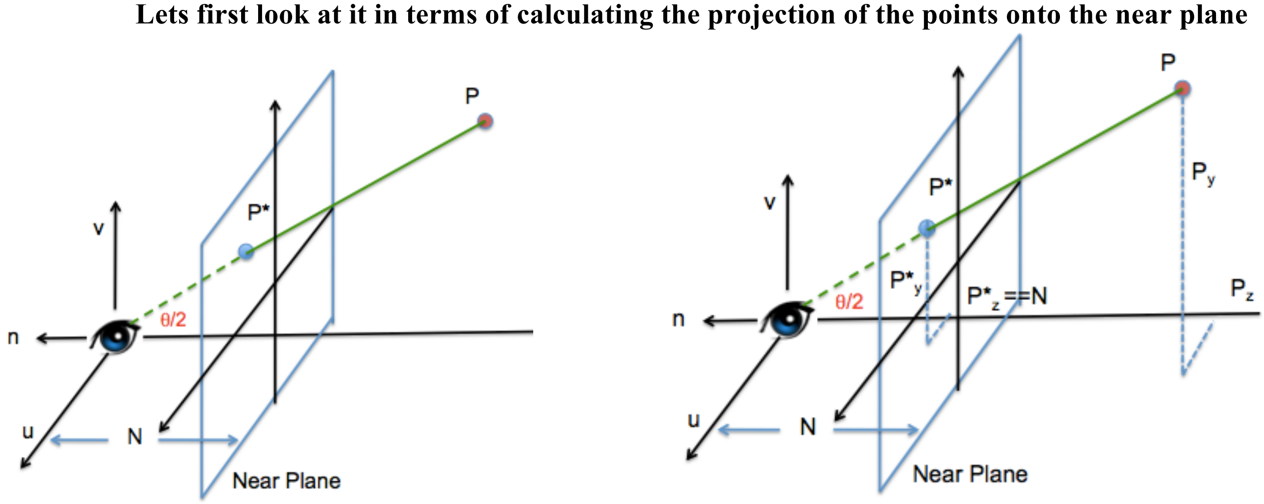
- Like in orthographic projection, the perspective projection will transform all the 3D objects within the viewing volume into the Canonical View Volume (CVV), but during this process, something different happened due to the difference in the shape of the viewing volume:

- First, I illustrate the part that is different from the orthographic projection. From the figure below, notice the following four changes that took place with this transformation (called the perspective transformation):

    1. transform the perspective viewing volume from the trapezoid shape in the left side figure into the rectangle shape in the right figure
    2. objects further away from the eye appears smaller than the ones close to the eye
    3. the center of the viewing volume is moved to the eye location, so the translation amount along x, y dimension (-(left+right)/2, -(bottom+top)/2, 0), z dimension is treated differently in a separate step
    4. the direction of the z axis in the eye coordinates is now changed to its opposite direction
    5. the near and far plane are scaled to be [-1, 1] along z



- **How are these perspective effects achieved?**
  Lets first look at it in terms of calculating the projection of the points onto the near plane



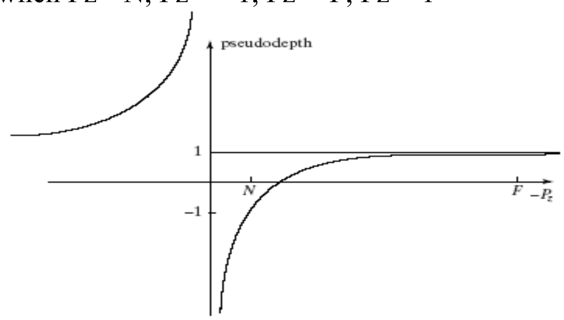
$$\frac{P_y*}{P_y} = \frac{N}{-P_z} \Rightarrow\Rightarrow P_y* = \frac{NP_y}{-P_z}, \quad \frac{P*_x}{P_x} = \frac{N}{-P_z} \Rightarrow\Rightarrow P*_x = \frac{NP_x}{-P_z}$$

Perspective transformation – how to perform this transformation using Matrix multiplication?

- The use of **homogeneous coordinates** **(wPx, wPy, wPz, w) ≡ (Px, Py, Pz)**
- Perspective transformation matrix

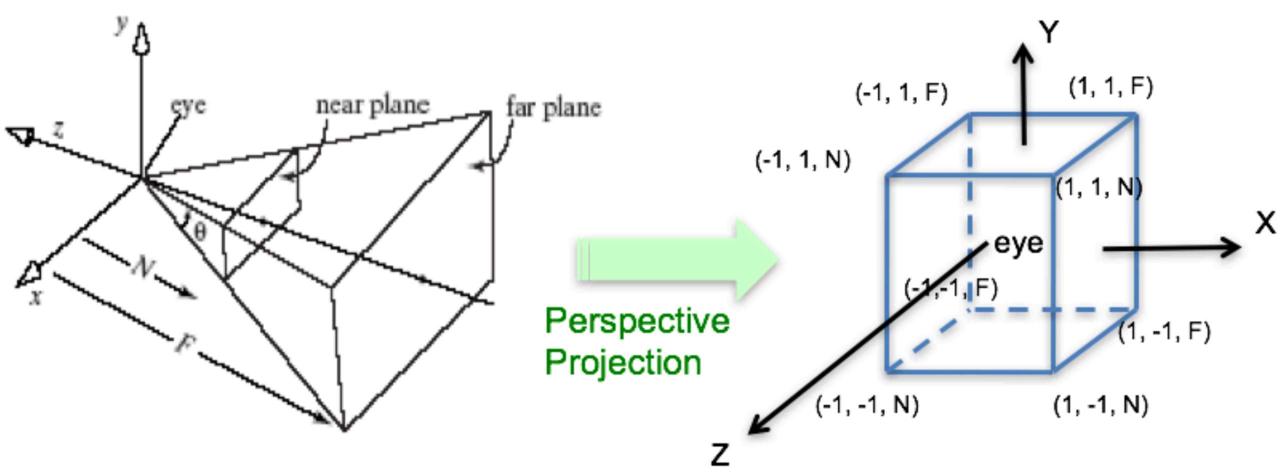
$$\begin{pmatrix} N & 0 & 0 & 0 \\ 0 & N & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & -1 & 0 \end{pmatrix}\begin{pmatrix} wP_x \\ wP_y \\ wP_z \\ w \end{pmatrix} = \begin{pmatrix} wNP_x \\ wNP_y \\ w(aP_z + b) \\ -wP_z \end{pmatrix}$$

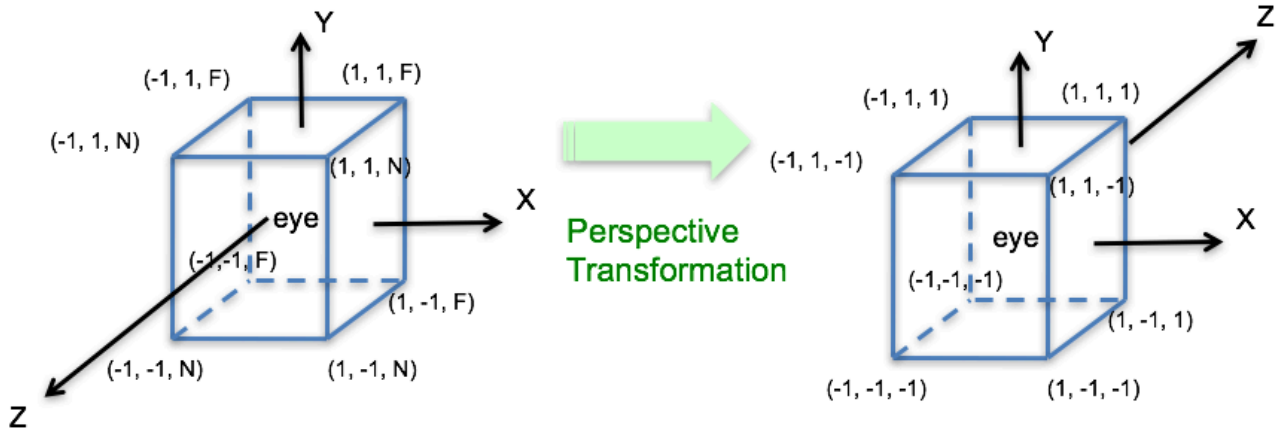
- Then, perform **perspective division**: $P = \begin{bmatrix} wNPx \\ wNPy \\ w(aPz+b) \\ -wPz \end{bmatrix} = \begin{bmatrix} N\frac{Px}{-Pz} \\ N\frac{Py}{-Pz} \\ \frac{aPz+b}{-Pz} \\ 1 \end{bmatrix}$
- Observations:
  - the last row of this matrix is different from all other matrices we have seen before
  - the new z-value?
  - what are the values of a and b?
    - they are values chosen to compute the pseudo depth (a=$-\frac{F+N}{F-N}$, b=$-\frac{2FN}{F-N}$)
    - when Pz=-N, Pz*= -1, Pz= -F, Pz*=1



- **Perspective Projection**

perspective projection = perspective transformation + perspective projection (modified orthographic projection)



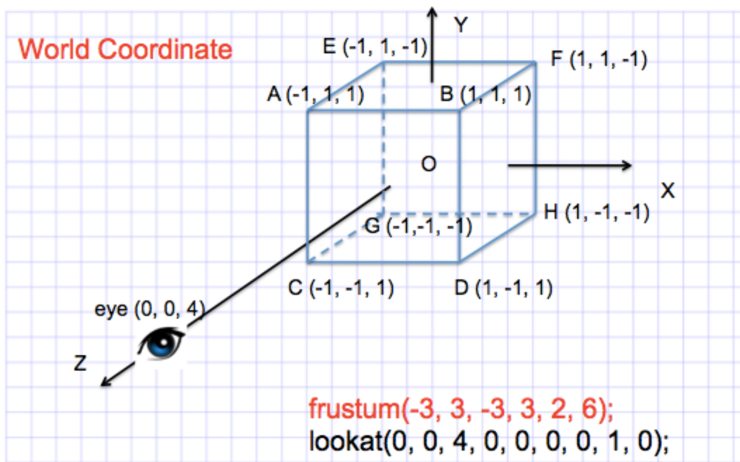


$$\begin{aligned}
&\textit{Perspective Projection Matrix}\\
&= \textit{Perspective Transformation} * \textit{Scale} * \textit{Translation}\\
&= \begin{pmatrix} N & 0 & 0 & 0 \\ 0 & N & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & -1 & 0 \end{pmatrix} * \begin{pmatrix} \frac{2}{right-left} & 0 & 0 & 0 \\ 0 & \frac{2}{top-bottom} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} 1 & 0 & 0 & -\frac{left+right}{2} \\ 0 & 1 & 0 & -\frac{top+bottom}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}\\
&= \begin{pmatrix} \frac{2N}{right-left} & 0 & \frac{right+left}{right-left} & 0 \\ 0 & \frac{2N}{top-bottom} & \frac{top+bottom}{top-bottom} & 0 \\ 0 & 0 & a & b \\ 0 & 0 & -1 & 0 \end{pmatrix}, \quad \textit{where} \ \ a = \frac{-(F+N)}{F-N} \ \ \textit{and} \ \ b = \frac{-2FN}{F-N}
\end{aligned}$$

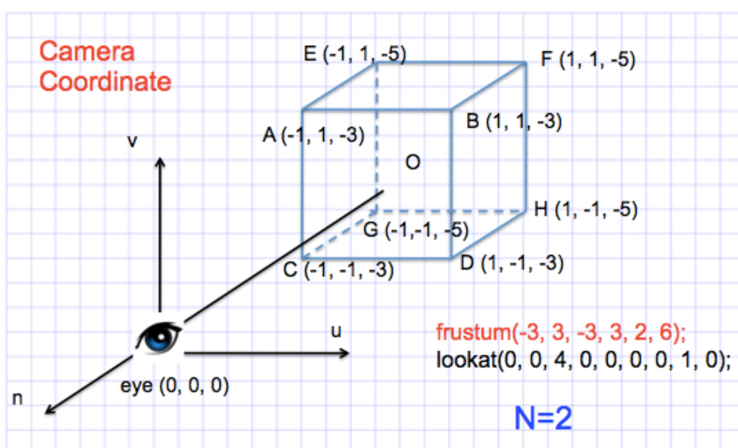
An equivalent projection matrix using the Frustum:

$$P= \begin{bmatrix} \frac{1}{aspect * \tan(\frac{fov}{2})} & 0 & 0 & 0 \\ 0 & \frac{1}{\tan(\frac{fov}{2})} & 0 & 0 \\ 0 & 0 & -\frac{far+near}{far-near} & -\frac{2*far*near}{far-near} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

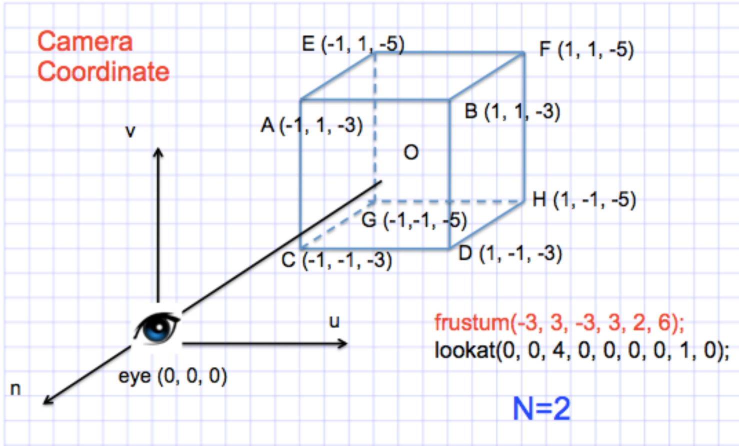
Practice Problem:



Change into eye/camera coordinates:



What will the cube look like when it is projected on the near plane? (i.e., displayed on screen?)



A=(-1, 1, -3) → A'=(2*-1/3, 2*1/3)
=(-2/3, 2/3)
B=(1, 1, -3) → B'=(2/3, 2/3)
C=(-1, -1, -3) → C'=(-2/3, -2/3)
D=(1,-1, -3) → D'=(2/3, -2/3)
E=(-1, 1, -5) → E'=(2*(-1)/(-(-5)), 2*(1)/(-(-5))) = (-2/5, 2/5)
F=(1, 1, -5) → F'=(2/5, 2/5)
G=(-1, -1, -5) → G'=(-2/5, -2/5)
H=(1, -1, -5) → H'=(2/5, -2/5)

The (wireframe) cube projected and displayed on the near plane: