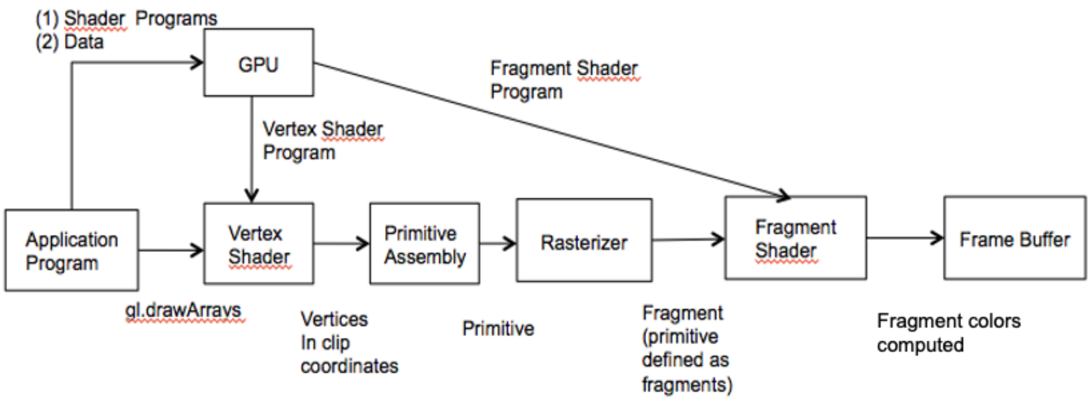
Orthographic Projection and Viewing

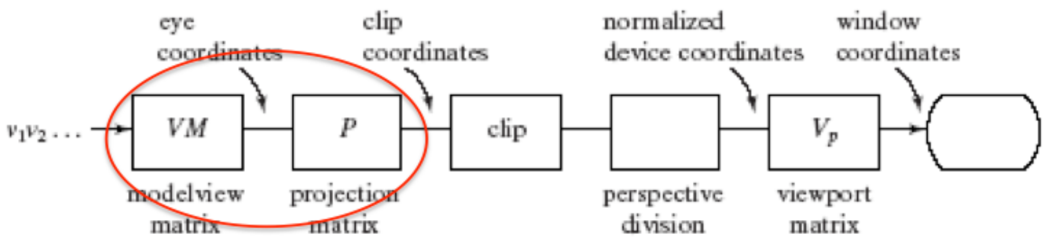
- **Where are we?**



Graphics execution model

The old Graphics pipeline:

- **Model matrix (transformations applied)**
  - What do the transformation matrices achieve?
  - Is there a change of frame involved?
- **View matrix (eye and look at direction)**
  - What does the view matrix achieve?
  - Is there a change of frame involved?
- **Projection matrix (viewing volume defined)**



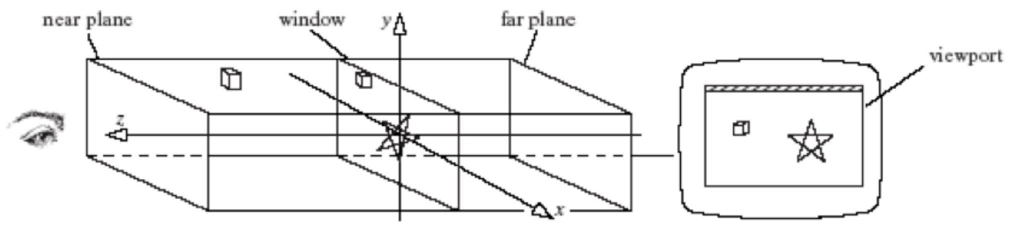
- **What is the purpose of a projection matrix?**
- **What are the clip-coordinates?**

- Where do these two steps (VM and P) go in terms of the "Execution model of graphics on GPU"?

Frame – Formed by a set of 3 orthogonal vectors and an origin.

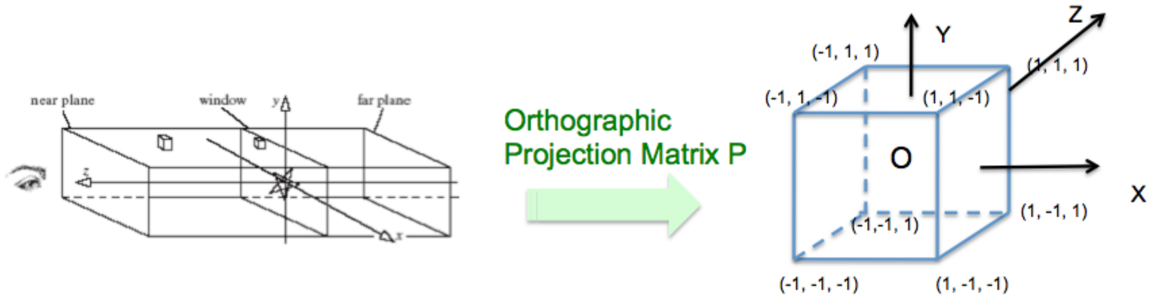
- **Orthographic Projection**
  ortho(left, right, bottom, top, near, far)

  Example1, projectionMatrix = ortho(-10, 10, -5, 5, -8, 8);
  Example2, projectionMatri = ortho(-2, 6, -4, 8, -1, 1);



- **The view volume is a rectangular parallelepiped**
  - Along the x-coordinate, and the y-coordinate
  - Along the z-coordinate

- **Projection Matrix**



  - **Canonical View Volume (CVV)** characteristics
    - A 2x2x2 cube aligned with the coordinate axes in viewing coordinates, and centered at origin
    - The idea is that no matter what the viewing conditions are, after a projection transformation is performed, all of the scene is transformed into this space, and anything in that scene that is not in the CVV after that transformation is ready to be clipped away easily in the next step, the clip step.
    - The origin of the frame is in the middle of the CVV
    - The Z axis is changed to the opposite direction from where it is in the viewing frame.

  - Clipping against CVV

  - What happens to the objects that were in the view volume?
    - The x-values and y-values of the points defining the primitives
      - Mapped onto the viewing plane
    - The z-values of the points defining the depth
      - Instead of Euclidean distance, a **pseudo-depth**, $-1 \leq P_z' \leq 1$ for $-N > P_z > -F$ is computed. This quantity is faster to compute than the Euclidean distance.

  - How to define the transformation matrix that will transform a view volume defined by a user into a CVV?
    - What will be the frame all the points are in?
    - Two transformations are needed:
      - **Step 1: translate** the frame to center at the origin
        - Where is the origin of the frame before projection? i.e., right after view matrix is applied?
        - Where is the center of the viewing volume?
        - What is the vector v that moves the origin of the frame from the eye location to the center of the viewing volume?

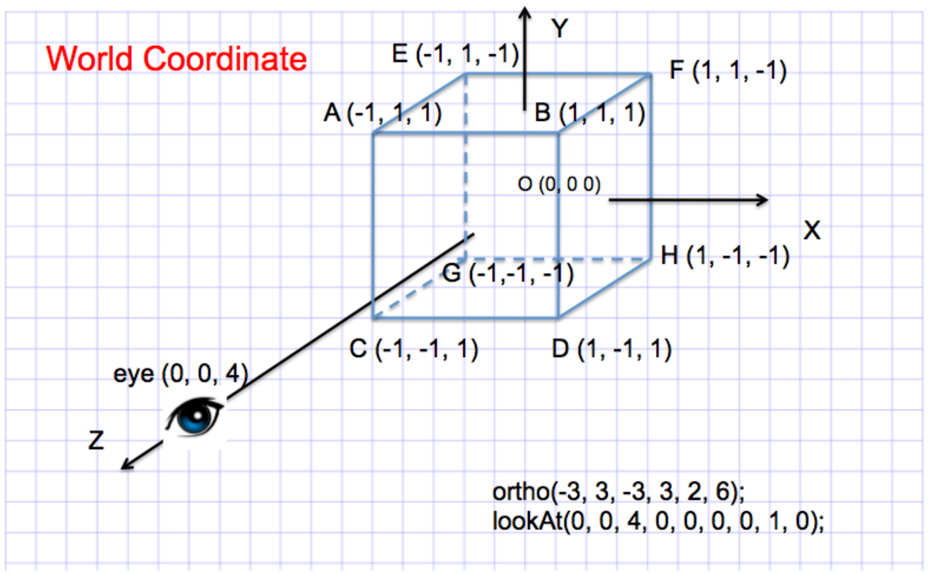
- **Step 2: scale** the viewing volume (already moved to center at **the new origin**) to be of size x:[-1, 1], y[-1, 1], and z[-1, 1], and with the direction of the z-axis flipped pointing to the opposite direction

$$P = \begin{bmatrix} \frac{2}{right-left} & 0 & 0 & 0 \\ 0 & \frac{2}{top-bottom} & 0 & 0 \\ 0 & 0 & -\frac{2}{far-near} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & -\frac{left+right}{2} \\ 0 & 1 & 0 & -\frac{bottom+top}{2} \\ 0 & 0 & 1 & \frac{near+far}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$P = \begin{bmatrix} \frac{2}{right-left} & 0 & 0 & -\frac{left+right}{right-left} \\ 0 & \frac{2}{top-bottom} & 0 & -\frac{bottom+top}{top-bottom} \\ 0 & 0 & \frac{-2}{far-near} & -\frac{near+far}{far-near} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

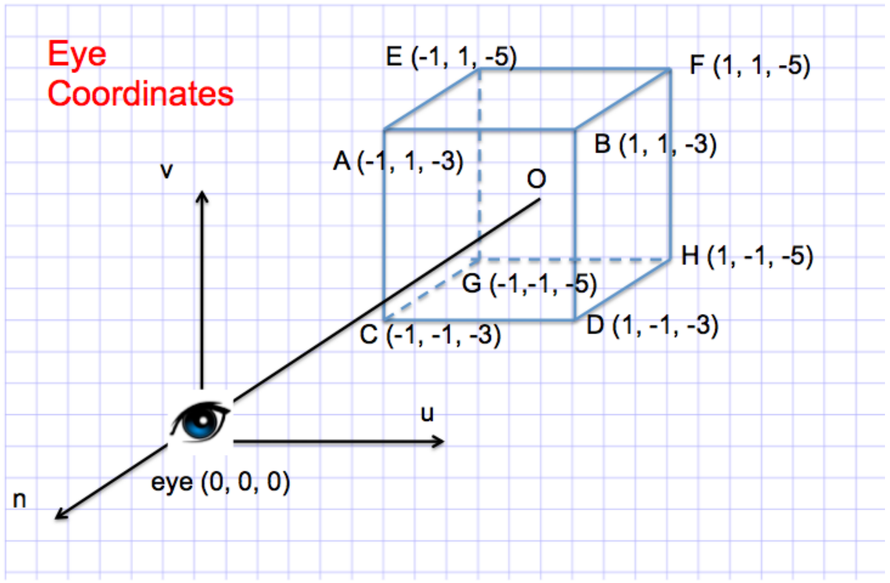
- MV.js code
- ch05/ortho.html, ortho.js
- swirl.html swirl.js

- **Putting it all together**
  Now consider how the location and direction of the eye/camera should be incorporated into the projection transformation:



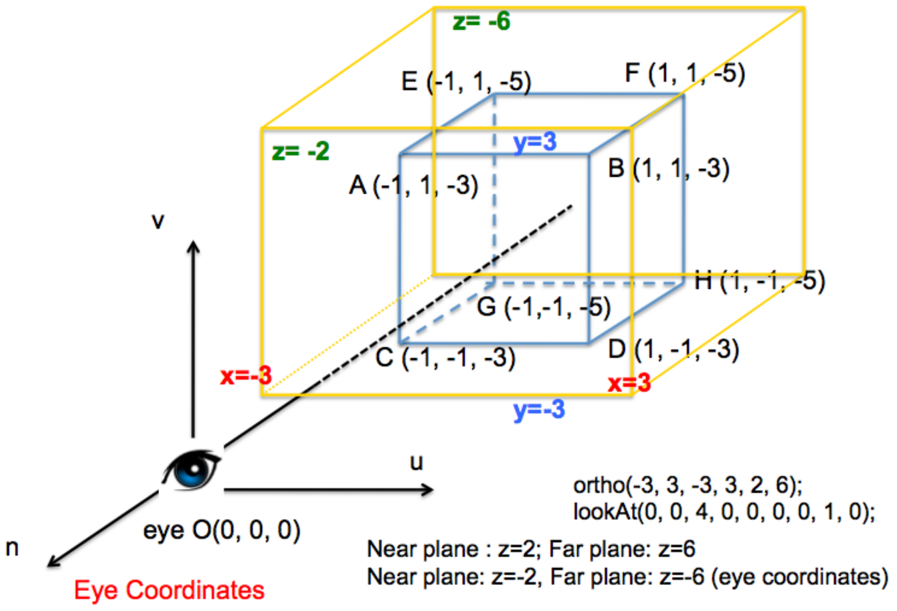
Given the coordinates of the points (A-H) defined in the world coordinates:
(to keep the problem simper, we assume that no model transformation (rotate, scale, translation, etc) is applied to these points.
  - Compute the View matrix
    - For a quick check, see if the coordinates in the eye coordinate system matches with the ones shown in the figure below.
  - [If there is model transformations, they would have been applied to the points]

- Compute the Projection matrix
- Compute the coordinates of the points in the CVV



$$V = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -4 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad C' = V * C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ -3 \\ 1 \end{bmatrix}$$

defining the view volume

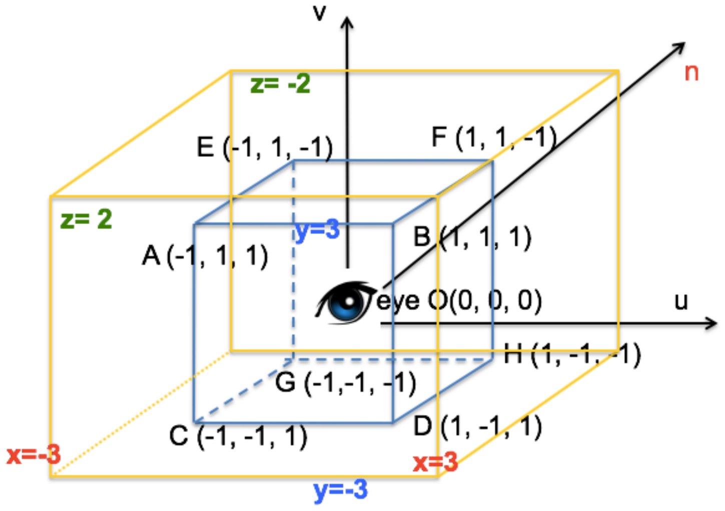


ortho(-3, 3, -3, 3, 2, 6);
lookAt(0, 0, 4, 0, 0, 0, 0, 1, 0);
Near plane : z=2; Far plane: z=6
Near plane: z=-2, Far plane: z=-6 (eye coordinates)

Applying the projection matrix (converting into CVV)

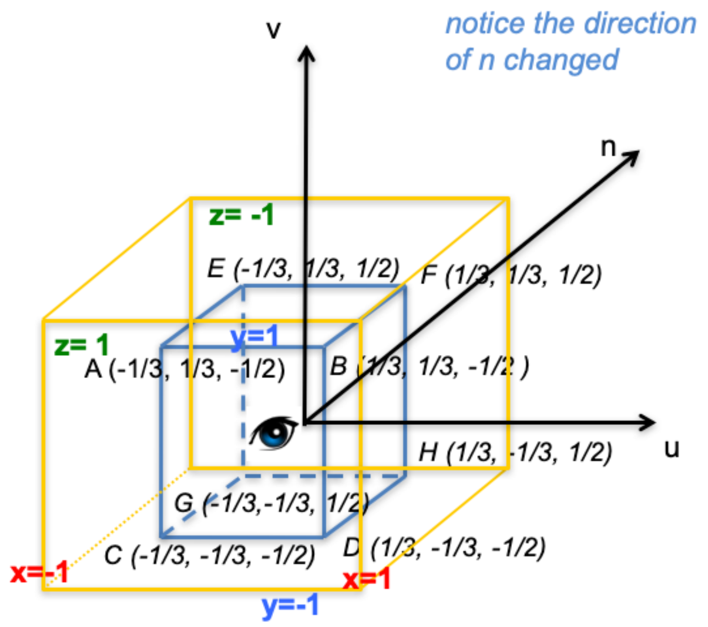
- **Step 1: Translate the view volume – the center of the view volume is translated to the eye location (O). The translation vector**

  $V = (0, 0, 0) - \left(\frac{left+right}{2}, \frac{bottom+top}{2}, -\frac{near+far}{2}\right) = \left(-\frac{left+right}{2}, -\frac{bottom+top}{2}, \frac{near+far}{2}\right)$



$$T = \begin{bmatrix} 1 & 0 & 0 & -\dfrac{left+right}{2} \\ 0 & 1 & 0 & -\dfrac{bottom+top}{2} \\ 0 & 0 & 1 & \dfrac{near+far}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- **Step 2: Scale this view volume to CVV shape x:[-1,1], y:[-1,1], z[-1,1]**



$$S = \begin{bmatrix} \frac{2}{right-left} & 0 & 0 & 0 \\ 0 & \frac{2}{top-bottom} & 0 & 0 \\ 0 & 0 & \frac{2}{far-near} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The projection matrix is the combination of these two steps: P=S*T

$$P = \begin{bmatrix} \frac{2}{right-left} & 0 & 0 & 0 \\ 0 & \frac{2}{top-bottom} & 0 & 0 \\ 0 & 0 & \frac{-2}{far-near} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & -\frac{left+right}{2} \\ 0 & 1 & 0 & -\frac{bottom+top}{2} \\ 0 & 0 & 1 & \frac{near+far}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$P = \begin{bmatrix} \frac{2}{right-left} & 0 & 0 & -\frac{left+right}{right-left} \\ 0 & \frac{2}{top-bottom} & 0 & -\frac{bottom+top}{top-bottom} \\ 0 & 0 & \frac{-2}{far-near} & -\frac{near+far}{far-near} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

What are the coordinates of point C after the cube is converted into the Canonical View Volume (CVV)?

$$P = \begin{bmatrix} 1/3 & 0 & 0 & 0 \\ 0 & 1/3 & 0 & 0 \\ 0 & 0 & -1/2 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix}, V = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -4 \\ 0 & 0 & 0 & 1 \end{bmatrix}, PV = \begin{bmatrix} 1/3 & 0 & 0 & 0 \\ 0 & 1/3 & 0 & 0 \\ 0 & 0 & -1/2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

For the point C=(-1, -1, 1), C'=P*V*C= $\begin{bmatrix} 1/3 & 0 & 0 & 0 \\ 0 & 1/3 & 0 & 0 \\ 0 & 0 & -1/2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} -1 \\ -1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} -1/3 \\ -1/3 \\ -1/2 \\ 1 \end{bmatrix}$