Viewing Transformation

- World coordinate system
- Object (model) coordinate system
- In OpenGL: modelview matrix
  - modelview matrix is constructed in two steps
  - from ocs (mcs) to wcs
  - from wcs to vcs
- Viewing coordinate system
- A common way to define the camera position and orientation
  - eye position
  - a reference point
  - an up vector
-

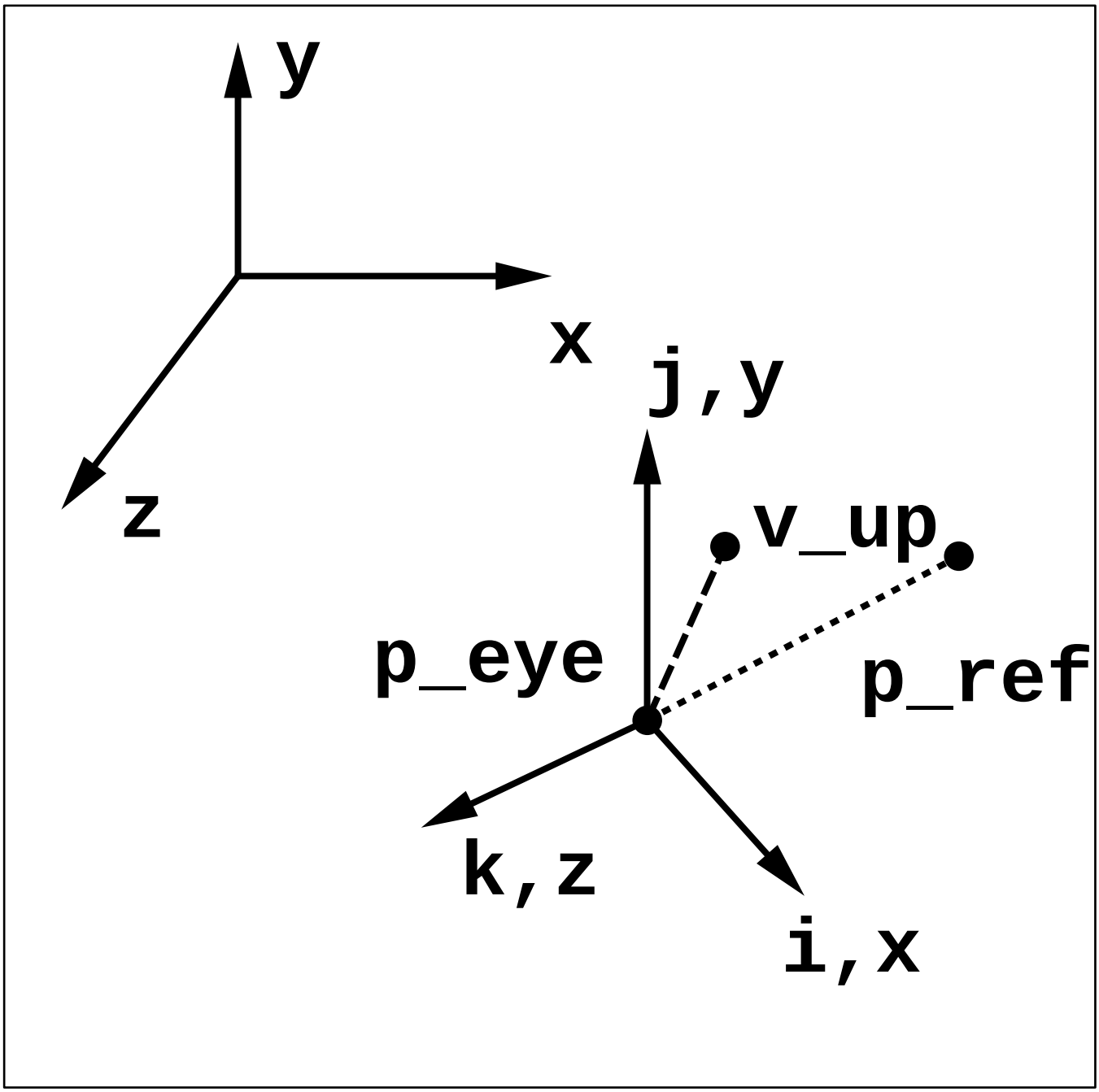
$$\mathrm{k} = \frac{\mathrm{p}_{eye} - \mathrm{p}_{ref}}{|\mathrm{p}_{eye} - \mathrm{p}_{ref}|}$$

$$\mathbf{I} = \mathbf{v}_{up} \times \mathbf{k}$$

$$\mathbf{i} = \frac{\mathbf{I}}{|\mathbf{I}|}$$

$$\mathbf{j} = \mathbf{k} \times \mathbf{i}$$

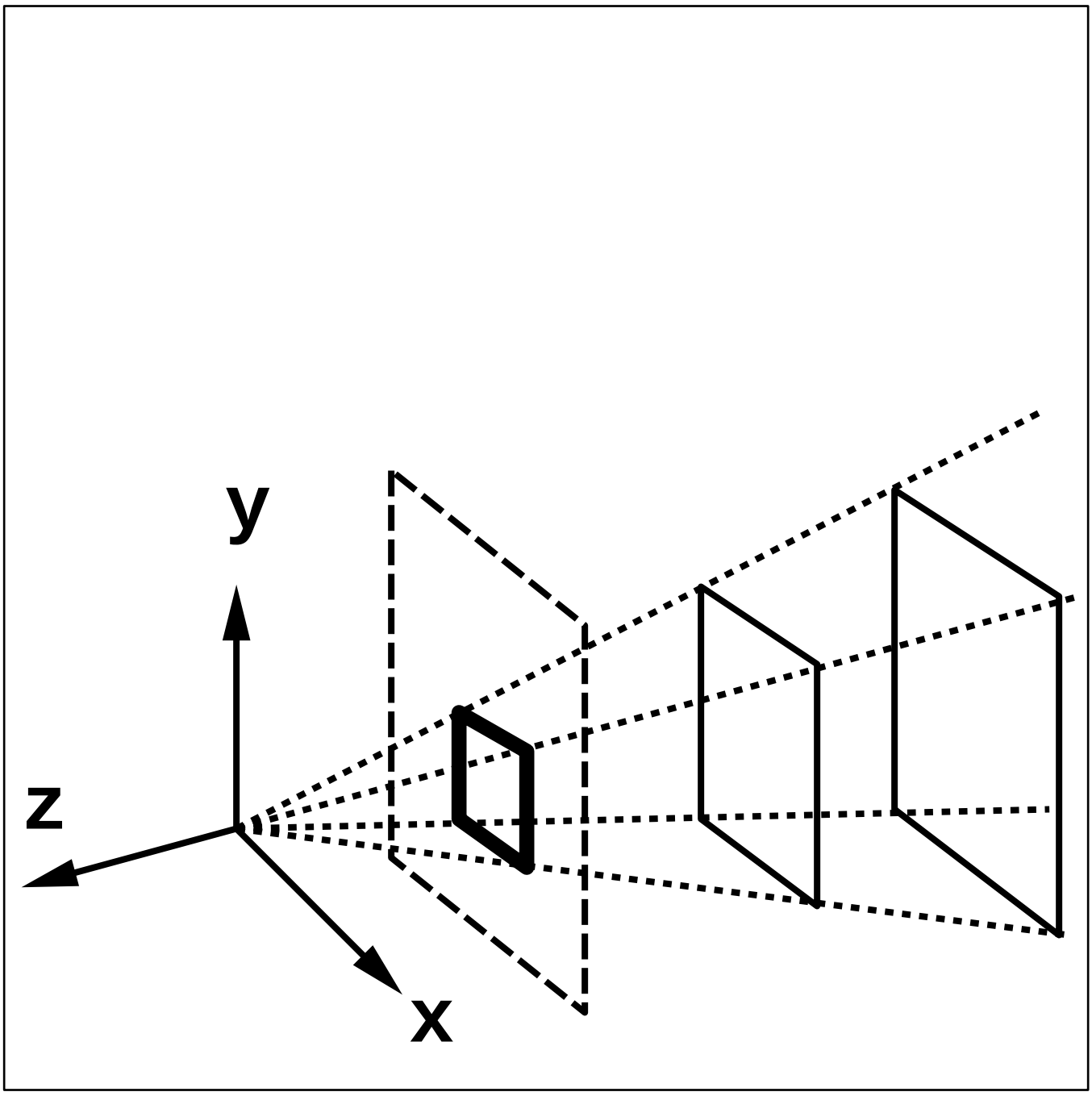
Viewing Coordinate System



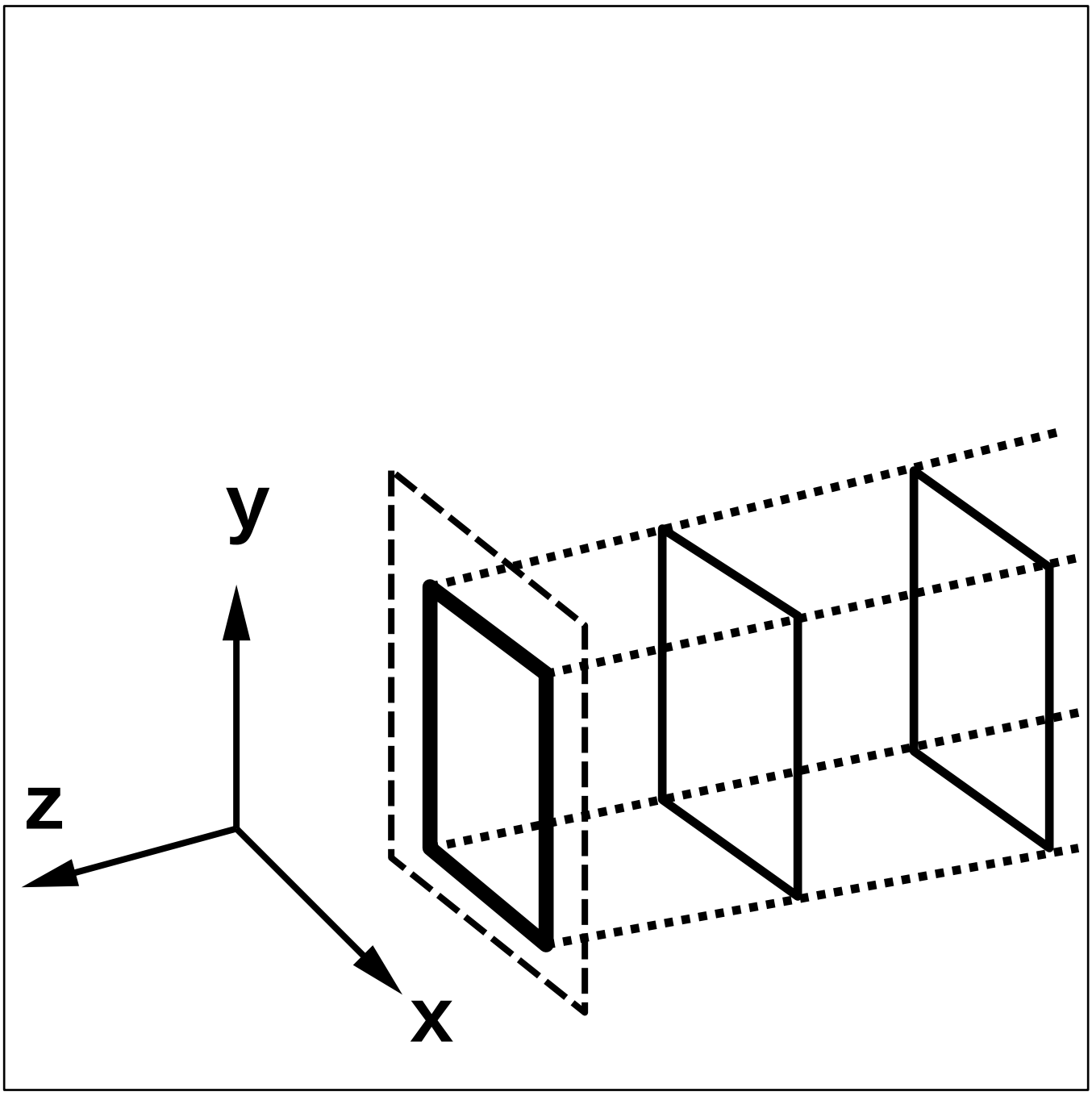
Viewing Volume

- Frustum
- Clip object which will not project on the image plane
- Restricting the domain of z for visibility calculation
- A perspective viewing volume
  - image plane
  - x = left
  - y = right
  - y = top
  - y = bottom
  - z = - near
  - z = - far

Viewing Volume



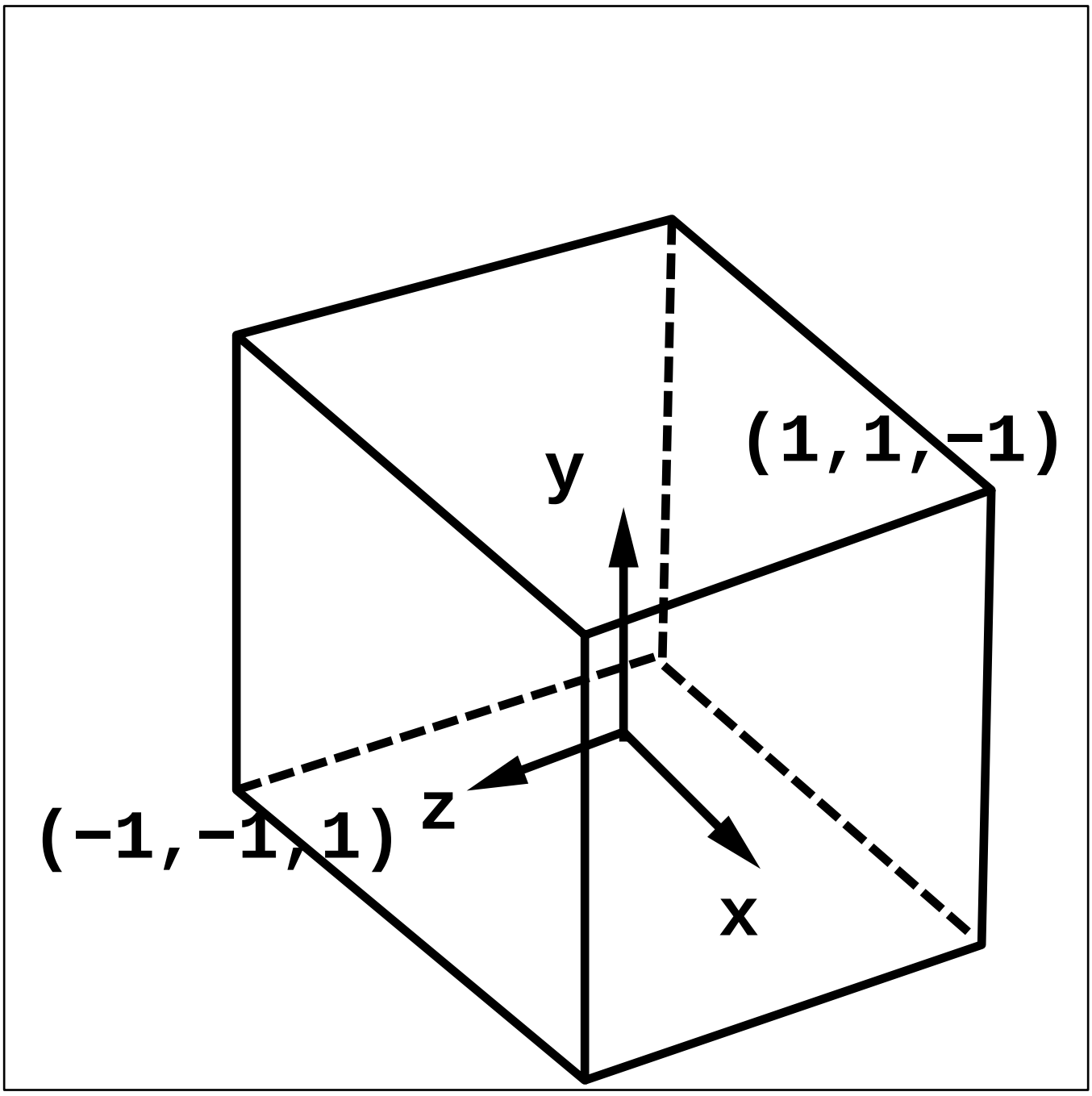
Orthographic View Volume



Orthographic View

- x = left
- x = right
- y = top
- y = bottom
- z = - near
- z = - far

Normalized View Volume



Projection Transformation

- From view volumes into canonical view volumes
- For orthographic projection
  (Scaling, translation)
- Clipping operations can be carried out
  after we map the existing view volume into
  "canonical or normalized view volume"
- This is because clipping operations are
  much simplified
- In OpenGL, normalized view volume is a cube of
  size 2 centered at origin !
- z-coordinates are retained for use in
  visibility calculations
- For perspective projection
  It is very complicated, deformation is involved!

3D Clipping

- Why 3D clipping
- Why not clipping in NDCS ?
- The transformation from VCS to NDCS is non-linear due to the division operation
- View volume clipping
- 2D algorithms can be generalized to 3D
  - Cohen-Sutherland line-clipping
  - Sutherland-Hodgeman algorithm
- Plane equations