Effect of Perspective Projection on Lines

Line equations (again)

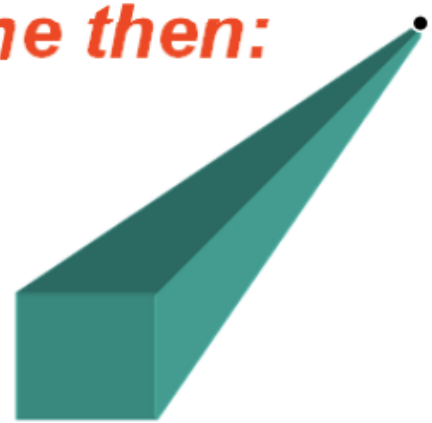
$$\text{Original:}\quad L(t) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} P_x + v_x t \\ P_y + v_y t \\ P_z + v_z t \end{bmatrix}$$

$$\text{Projected:}\quad L'(t) = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} -Nx/z \\ -Ny/z \\ -N \end{bmatrix} = \begin{bmatrix} -N(P_x + v_x t)/(P_z + v_z t) \\ -N(P_y + v_y t)/(P_z + v_z t) \\ -N \end{bmatrix}$$

If lines are not parallel to the image plane then:

$$v_z \neq 0 \rightarrow \lim_{t \to \infty} L'(t) = \begin{bmatrix} -Nv_x/v_z \\ -Nv_y/v_z \\ -N \end{bmatrix}$$

Lines converge to a **vanishing point**



But is There a Difference?

Original: $$L(t) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} P_x + v_x t \\ P_y + v_y t \\ P_z + v_z t \end{bmatrix}$$

Projected: $$L'(t) = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} -Nx / z \\ -Ny / z \\ -N \end{bmatrix} = \begin{bmatrix} -N(P_x + v_x t) / (P_z + v_z t) \\ -N(P_y + v_y t) / (P_z + v_z t) \\ -N \end{bmatrix}$$

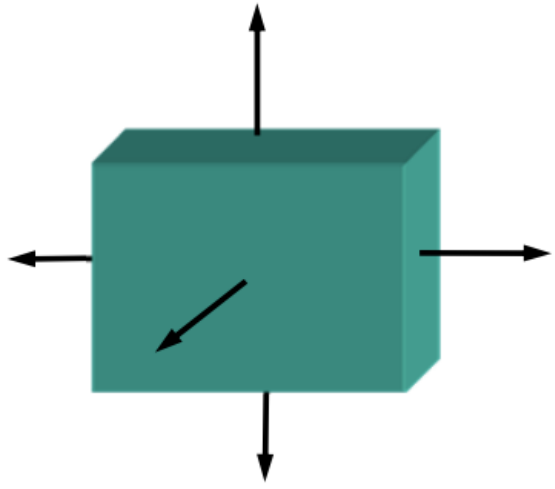
Polygon Attributes

Per vertex

- Color
- Texture coordinates

Per vertex or per face

- Color
- Normal



Rule of Thumb

Transforming a point P:

Transformations: T_1, T_2, T_3

Matrix: $\mathbf{M} = \mathbf{M}_3 \mathbf{M}_2 \mathbf{M}_1$

Point is transformed by $\mathbf{M}P$

Each transformation happens with respect to the **same** coordinate system

Transforming a coordinate system:

Transformations: T_1, T_2, T_3 (not generally the same as the ones above)

Matrix: $\mathbf{M} = \mathbf{M}_1 \mathbf{M}_2 \mathbf{M}_3$

A point has coordinates $\mathbf{M}P$ in the original coordinate system

Each transformation happens with respect to **previous** coordinate system

Properties of Affine Transformations

1. *Affine transformations are composed of elementary ones*
2. *Preservation of affine combinations of points*
3. *Preservation of lines and planes*
4. *Preservation of parallelism of lines and planes*
5. *Relative ratios are preserved*

Preview:
Raytracing Algorithm

for each pixel on screen

determine ray from eye through pixel

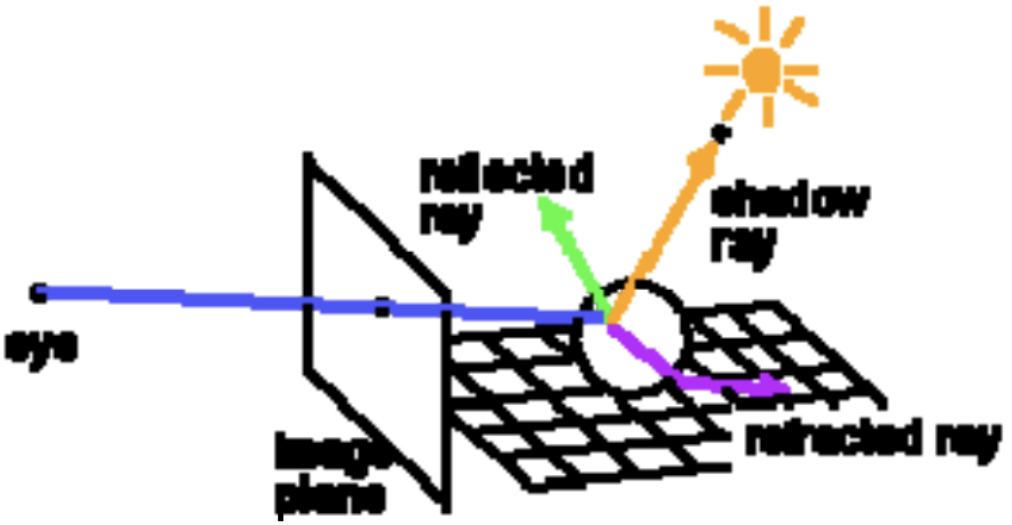
find closest intersection of ray with an object

cast off reflected and refracted ray, recursively

calculate pixel color

draw pixel

end



Preview:
Z-Buffer Algorithm

set pixels to background color and z-buffer to maximum z-values

for each polygon in model

  project vertices of polygon onto viewing plane

  for each pixel inside the projected polygon

    calculate pixel color

    calculate pixel z-value

    if z-value is less than z-value stored for pixel in z-buffer

      set pixel to color and store z-value into z-buffer

  end

end