

Viewing in 3D

Taking a picture with a camera

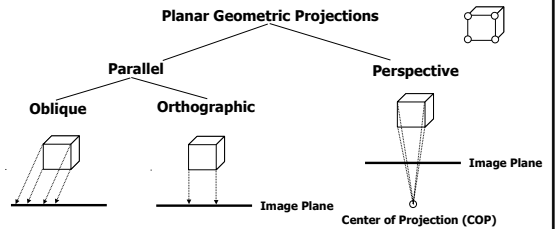
- Coordinates: Local, World, Viewing
- Rendering pipeline
- ModelView
 - Matrix operations on models
- World coordinates to Viewing coordinates
 - Matrix operations (models or cameras)
- Projection with a camera

Viewing in 3D

- Planar Geometric Projections
- Parallel Orthographic Projections
- Perspective Projections
- Projections in OpenGL
- Clipping

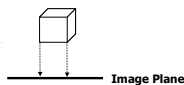
Planar Geometric Projections

- Maps points from camera coordinate system to the screen (image plane of the virtual camera).



Parallel Orthographic Projection

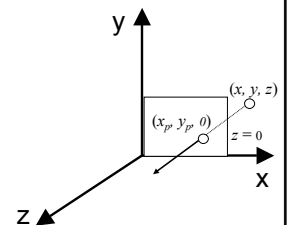
- Preserves X and Y coordinates.
- Preserves both distances and angles.



Parallel Orthographic Projection

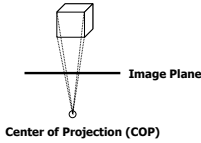
- $x_p = x$
- $y_p = y$
- $z_p = 0$

$$\begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

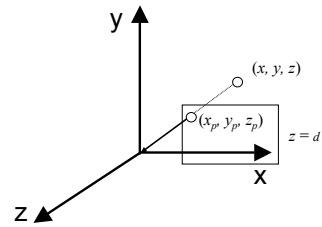


Perspective Projection

- Only preserves parallel lines that are parallel to the image plane.
- Line segments are shortened by distance.

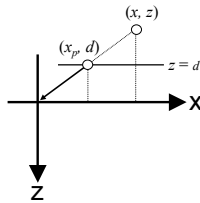


Perspective Projection



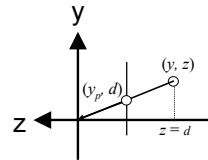
Perspective Projection

- $z_p = d$
- $x_p = (x \cdot d) / z$



Perspective Projection

- $z_p = d$
- $y_p = (y \cdot d) / z$



Perspective Projection

- $x_p = (x \cdot d) / z = x / (z/d)$
- $y_p = (y \cdot d) / z = y / (z/d)$
- $z_p = d = z / (z/d)$

Viewing in 3D

- Planar Geometric Projections
- Parallel Orthographic Projections
- Perspective Projections
- Projections in OpenGL

Viewing in 3D

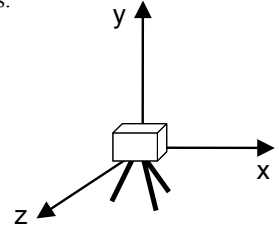
- Planar Geometric Projections
- Parallel Orthographic Projections
- Perspective Projections
- Projections in OpenGL
 - Positioning of the Camera
 - Define the view volume

CSC5870 Computer Graphics I

WISCONSIN STATE UNIVERSITY

Positioning the Camera

- By default, the camera is placed at the origin pointing in the negative z-axis.

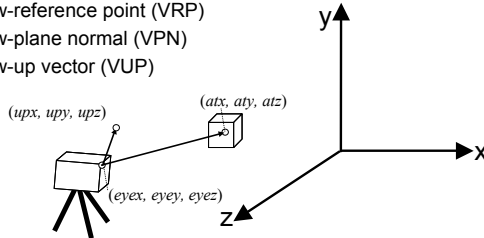


CSC5870 Computer Graphics I

WISCONSIN STATE UNIVERSITY

Positioning the Camera

- OpenGL Look-At Function
`gluLookAt(eyex, eyey, eyez, atx, aty, atz, upx, upy, upz)`
- View-reference point (VRP)
- View-plane normal (VPN)
- View-up vector (VUP)

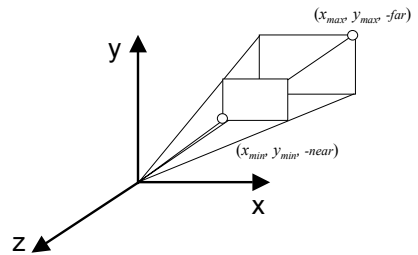


CSC5870 Computer Graphics I

WISCONSIN STATE UNIVERSITY

Defining the Perspective View Volume

`glFrustum(left, right, bottom, top, near, far)`

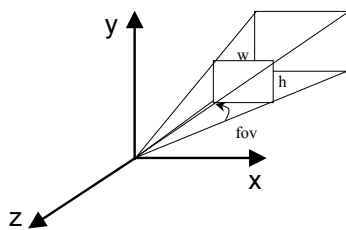


CSC5870 Computer Graphics I

WISCONSIN STATE UNIVERSITY

Defining the Perspective View Volume

`gluPerspective(fovy, aspect, near, far)`

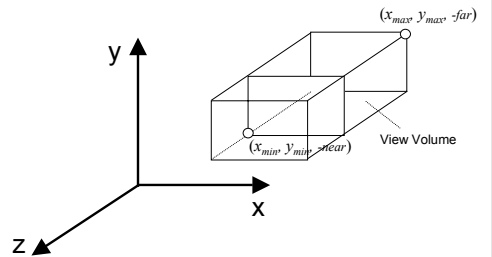


CSC5870 Computer Graphics I

WISCONSIN STATE UNIVERSITY

Defining the Parallel View Volume

`glOrtho(xmin, xmax, ymin, ymax, near, far)`



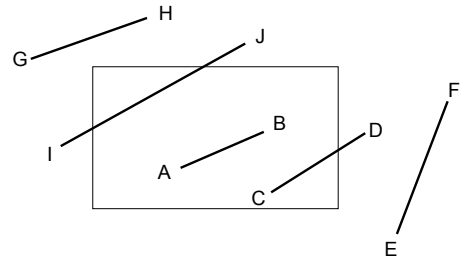
CSC5870 Computer Graphics I

WISCONSIN STATE UNIVERSITY

Clipping

2D Line Clipping

- Cohen-Sutherland algorithm



Cohen-Sutherland algorithm

- The space is divided into nine regions.
- Each region is assigned a unique 4-bit binary number: outcode

1001	1000	1010
0001	0000	0010
0101	0100	0110

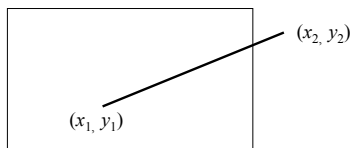
Cohen-Sutherland algorithm

- Bit 1: left
- Bit 2: right
- Bit 3: below
- Bit 4: above

1001	1000	1010
0001	0000	0010
0101	0100	0110

Cohen-Sutherland algorithm

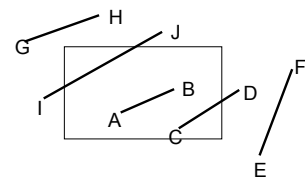
- Each line has two endpoints with two outcodes
 $o_1 = \text{outcode}(x_1, y_1)$ and $o_2 = \text{outcode}(x_2, y_2)$.
- We can decide base on these two outcodes.



Cohen-Sutherland algorithm

Four cases:

1. $o_1 = o_2 = 0$. Both endpoints are inside the window. Such as AB.
2. $o_1 \neq 0, o_2 = 0$, or vice versa. One endpoint is inside, the other is outside. Such as CD.



Cohen-Sutherland algorithm

Four cases:

- $o_1 \& o_2 \neq 0$. Whether or not the two endpoints lie on the same outside of the window. Such as EF.
- $o_1 \& o_2 = 0$. Both endpoints are outside, but there are on the outside of the different edges of the window. Such as GH and IJ.

