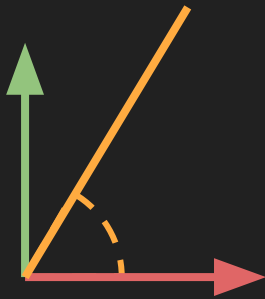


Fundamentals of Computer Graphics and Image Processing

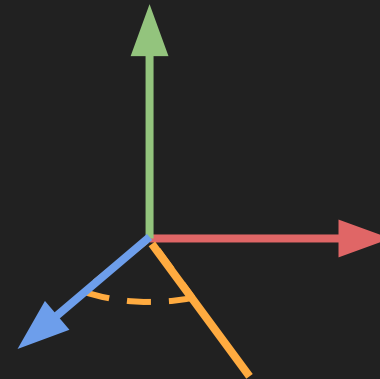
Computer Graphics - Exercise #02

Right-handed coordinate system

All rotational transformations are performed using right-hand rule



2D rotation
(around Z axis)



3D rotation
around Y axis

Transformation Matrices

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Identity

$$\begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Translation

$$\begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Scaling

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \varphi_x & -\sin \varphi_x & 0 \\ 0 & \sin \varphi_x & \cos \varphi_x & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotation X

$$\begin{pmatrix} \cos \varphi_y & 0 & \sin \varphi_y & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \varphi_y & 0 & \cos \varphi_y & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotation Y

$$\begin{pmatrix} \cos \varphi_z & -\sin \varphi_z & 0 & 0 \\ \sin \varphi_z & \cos \varphi_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

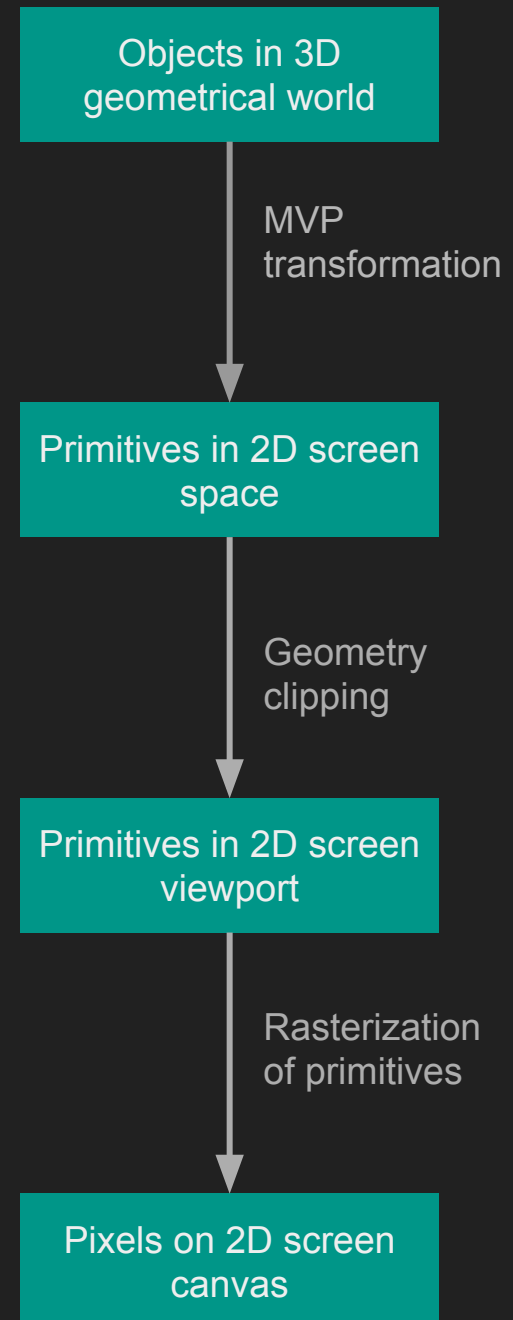
Rotation Z

Rendering Pipeline

Process of getting 3D models to screen pixels

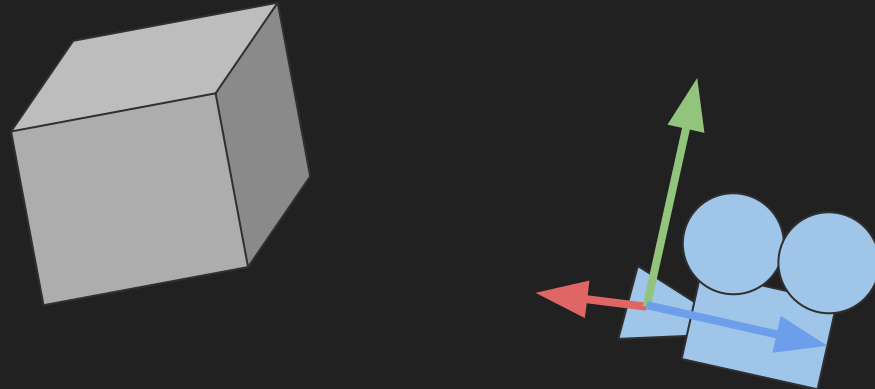
Each step takes processing power (data filtering)

At each step, there can be additional actions performed



Projection

Points in camera coordinate system are transformed to screen



Local Coordinates

Model Matrix

Global Coordinates

View Matrix

Camera Coordinates

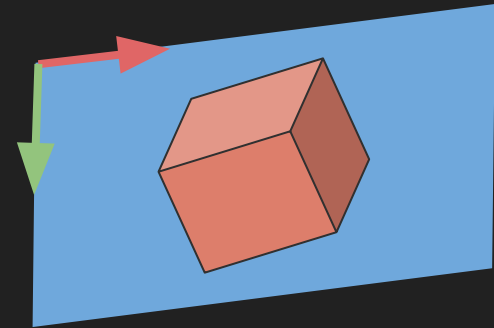
Projection Matrix

Screen Coordinates

Projection

Points in camera coordinate system are transformed to screen

3D geometry space is transformed to 2D screen space



Local Coordinates

Model Matrix

Global Coordinates

View Matrix

Camera Coordinates

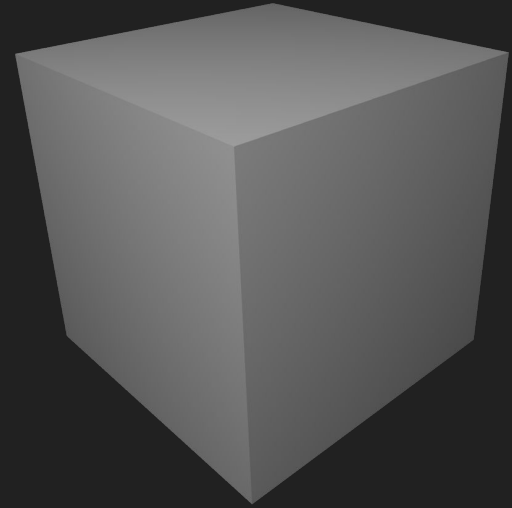
Projection Matrix

Screen Coordinates

Projection

Perspective

Distant objects appear smaller. Eye-like projection, looks natural for human observers.

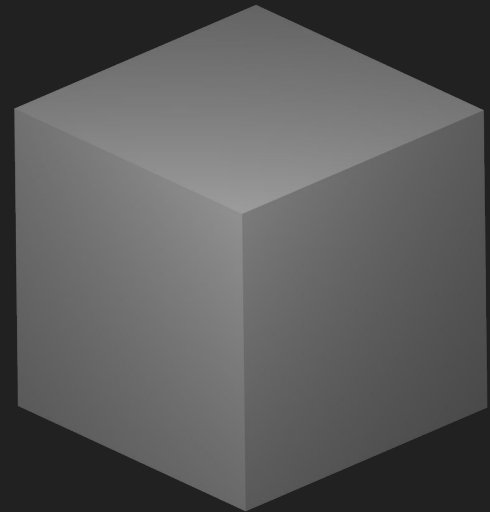


Orthographic

Parallel lines stay parallel, distant objects keep their size. Good for examining geometry.

Isometric

Special case of orthographic projection, where world axes X,Y,Z hold 120° in screen space.



Orthographic Projection

Geometry primitive in camera coordinate system = Z-axis vector and view direction are collinear

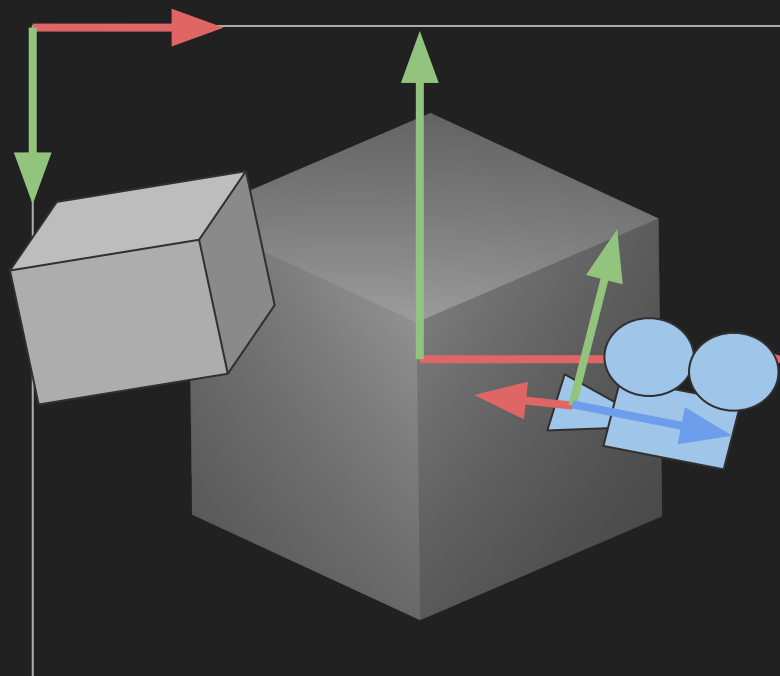
Ignoring Z coordinate performs an orthographic projection

Normalized screen space is independent of screen resolution

Additional transformation, of translation and scaling, is used to get object to viewport coordinates

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

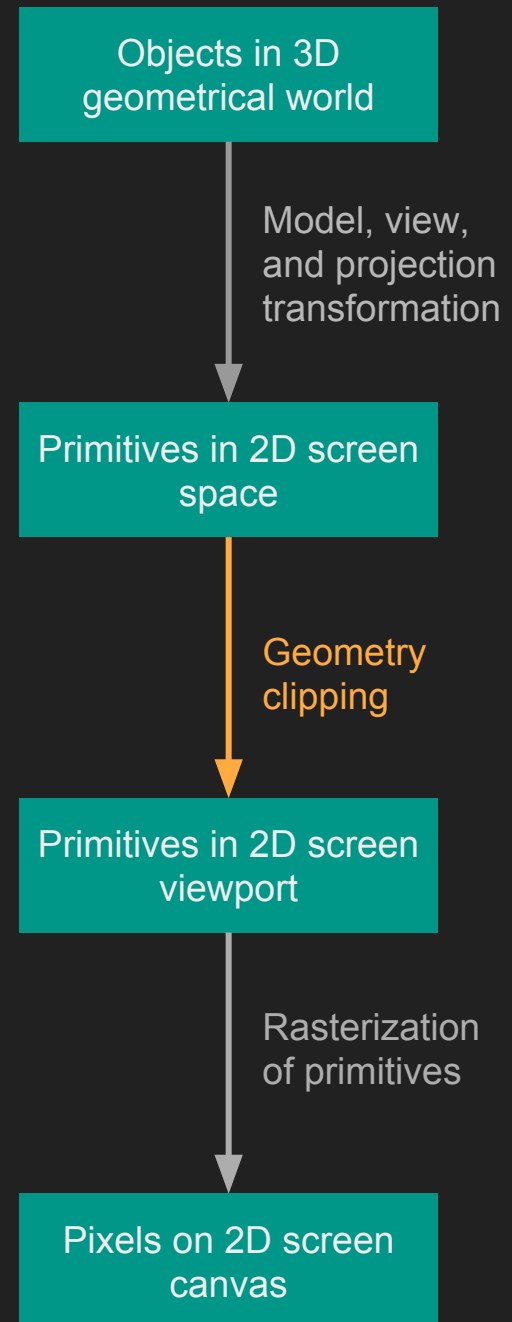
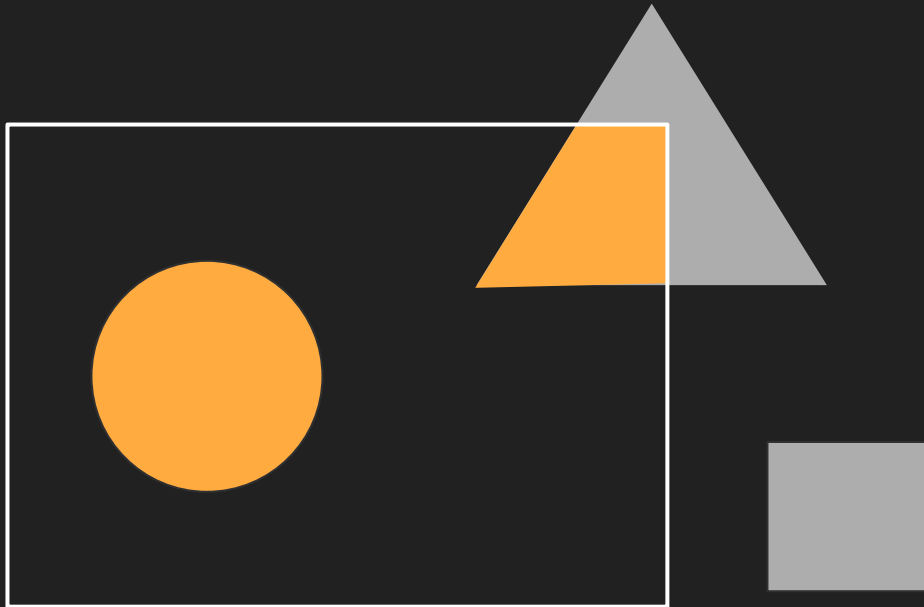
Orthographic projection matrix



Normalized screen coordinates

Clipping

Determines parts of geometry which are truly visible, therefore should be rendered



Cohen-Sutherland Line Clipping Algorithm

Works only with rectangular clip regions (viewports)

- 1) Create binary code for both ending points of a line (b_1, b_2) using comparison to 4 sides of clip region ($x_{\min}, y_{\min}, x_{\max}, y_{\max}$)

$$y > y_{\max} \mid y < y_{\min} \mid x > x_{\max} \mid x < x_{\min}$$

- 2) Compare both codes using binary operations

$b_1 \text{ OR } b_2 = 0$

~ line is whole inside viewport (draw!)

$b_1 \text{ AND } b_2 \neq 0$

~ line is whole outside viewport (don't draw!)

ELSE

~ line may be partially in viewport (continue to point 3)

- 3) Get a point which binary code is not equal to 0 and correct it by finding intersection with clip region sides.

Take a new/corrected point and continue from point 1