Einführung in Visual Computing
186.822

Viewing

Werner Purgathofer
Viewing in the Rendering Pipeline

- **scene objects in object space**
- **transformed vertices in clip space**
- **scene in normalized device coordinates**
- **raster image in pixel coordinates**

### Vertex Stage
- **clipping + homogenization**
- **projection**
- **viewing**
- **modeling**

### Pixel Stage
- **viewport transformation**
- **rasterization**
- **shading**

- **object capture/creation**

**Vertex Stage**
- vertex stage ("vertex shader")

**Pixel Stage**
- pixel stage ("fragment shader")
From Object Space to Screen Space

modeling transformation

world space

camera transformation

camera space

projection transformation

clip space

viewport transformation

pixel space

“view frustum”
Viewport Transformation
From Object Space to Screen Space

- Modeling transformation
- Camera transformation

Object space → World space → Camera space

Projection transformation

Clip space → Viewport transformation → Pixel space

"View frustum"
assumption: scene is in clip space!

clip space = \([-1,1] \times [-1,1] \times [-1,1]\)

orthographic camera looking in \(-z\) direction

screen resolution \(n_x \times n_y\) pixels
Viewport Transformation

can be done with the matrix

\[
\begin{bmatrix}
    x_{\text{screen}} \\
    y_{\text{screen}} \\
    1
\end{bmatrix} = \begin{bmatrix}
    n_x/2 & 0 & n_x/2 \\
    0 & n_y/2 & n_y/2 \\
    0 & 0 & 1
\end{bmatrix} \cdot \begin{bmatrix}
    x \\
    y \\
    1
\end{bmatrix}
\]

(-1,-1) \rightarrow (0,0)
(1,1) \rightarrow (n_x, n_y)

this ignores the z-coordinate, but...
Projection Transformation
From Object Space to Screen Space

- **Object Space**
  - Modeling Transformation

- **World Space**
  - Camera Transformation

- **Camera Space**

- **Clip Space**
  - Projection Transformation
  - Viewport Transformation

- **Pixel Space**

"view frustum"
3 parallel-projection views of an object, showing relative proportions from different viewing positions
Parallel vs. Perspective Projection
For Now: Parallel (Orthographic) Projection

object space → world space → camera space

modeling transformation

world space → camera space

camera transformation

object space → clip space → screen space

projection transformation

screen space

„view frustum“
Projection Transformation (Orthographic)

- assumption: scene in box \([L, R] \times [B, T] \times [F, N]\)
- orthographic camera looking in \(-z\) direction
- transformation to clip space

\[(L, B, F) \rightarrow (-1, -1, -1)\]
\[(R, T, N) \rightarrow (1, 1, 1)\]
Projection Transformation (Orthographic)

\[(L,B,F) \rightarrow (-1,-1,-1)\]
\[(R,T,N) \rightarrow (1,1,1)\]

\[M_{\text{orth}} = \begin{bmatrix}
\frac{2}{R-L} & 0 & 0 & -\frac{R+L}{R-L} \\
0 & \frac{2}{T-B} & 0 & -\frac{T+B}{T-B} \\
0 & 0 & \frac{2}{N-F} & -\frac{N+F}{N-F} \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
L \\
B \\
F \\
1
\end{bmatrix} = \begin{bmatrix}
-1 \\
-1 \\
-1 \\
1
\end{bmatrix}\]
Parallel Projection Types

- **orthographic projection**
  - Different orientation of the projection vector \(-g\)

- **oblique projection**
  - Different orientation of the projection vector \(-g\)
Camera Transformation
For Now: Parallel (Orthographic) Projection

- object space
- world space
- camera space

- modeling transformation
- camera transformation

- projection transformation
- viewport transformation

- clip space
- pixel space

„view frustum“
coordinate reference for obtaining a selected view of a 3D scene
Viewing: Camera Definition

- similar to taking a photograph
- involves selection of
  - camera position
  - camera direction
  - camera orientation
  - “window” (zoom) of camera

world coordinates

camera coordinates
view reference point
- origin of camera coordinate system
- gaze direction or look-at point

right-handed camera-coordinate system,
with axes u, v, w, relative to world-coordinate scene
e ... eye position

\( g \) ... gaze direction (positive \( w \)-axis points to the viewer)

\( t \) ... view-up vector

\[
\begin{align*}
\mathbf{w} &= -\frac{\mathbf{g}}{|\mathbf{g}|} \\
\mathbf{u} &= \frac{\mathbf{t} \times \mathbf{w}}{|\mathbf{t} \times \mathbf{w}|} \\
\mathbf{v} &= \mathbf{w} \times \mathbf{u}
\end{align*}
\]
e ... eye position
g ... gaze direction (positive w-axis points to the viewer)
t ... view-up vector

\[ w = - \frac{g}{|g|} \]

\[ u = \frac{t \times w}{|t \times w|} \]

\[ v = w \times u \]
aligning viewing system with world-coordinate axes using translate-rotate transformations

\[ M_{\text{cam}} = R_z \cdot R_y \cdot R_x \cdot T \]
For Now: Parallel (Orthographic) Projection

- **object space**
- **world space**
- **camera space**

```
modeling transformation
```
```
camera transformation
```
```
”view frustum”
```

```
projection transformation
```
```
viewport transformation
```

```
clip space
```
```
pixel space
```
Viewing: Camera + Projection + Viewport

\[
\begin{bmatrix}
    x_{\text{screen}} \\
    y_{\text{screen}} \\
    z \\
    1
\end{bmatrix} = (M_{vp} \cdot M_{orth} \cdot M_{cam}) \cdot 
\begin{bmatrix}
    x \\
    y \\
    z \\
    1
\end{bmatrix}
\]

- pixels on the screen
- viewport transformation
- projection transformation
- camera transformation
- world coordinates
For Now: Parallel (Orthographic) Projection

- **modeling transformation**
- **world space**
- **clip space**
- **camera space**
- **projection transformation**
- **viewport transformation**
- **pixel space**

„view frustum“

"one matrix!"
Perspective Projection
perspective projection of equal-sized objects at different distances from the view plane
Parallel vs. Perspective Projection

**Parallel** projection: preserves relative proportions & parallel features (affine transform.)

**Perspective** projection: center of projection, realistic views
Perspective Transform
Perspective Transformation

view plane

\[ y_p = \frac{f}{z} \cdot y \]

O

definition of perspective transformation

\[ \frac{y_p}{y} = \frac{f}{z} \]

f … focal length
perspective transformation

\[
y_p = \frac{N}{z} y
\]

\[
x_p = \frac{N}{z} x
\]

derivation of perspective transformation

\[
\begin{bmatrix}
N & 0 & 0 & 0 & 0 \\
0 & N & 0 & 0 & 0 \\
0 & 0 & N+F & -F \cdot N & 0 \\
0 & 0 & 1 & 0 & 0
\end{bmatrix} = P
\]
Perspective Transformation

\[
P = \begin{bmatrix}
  N & 0 & 0 & 0 \\
  0 & N & 0 & 0 \\
  0 & 0 & N+F & -F \cdot N \\
  0 & 0 & 1 & 0 \\
\end{bmatrix} \cdot \begin{bmatrix}
  x \\
  y \\
  z \\
  1 \\
\end{bmatrix} = \begin{bmatrix}
  x \cdot N \\
  y \cdot N \\
  z \cdot (N+F) - F \cdot N \\
  z \\
\end{bmatrix}
\]

derivation of perspective transformation

homogenization: divide by \( z \)

\[
y_p = \frac{N}{z} y \\
x_p = \frac{N}{z} x
\]

\[
\begin{bmatrix}
  x \cdot N/z \\
  y \cdot N/z \\
  (N+F) - F \cdot N/z \\
  1 \\
\end{bmatrix}
\]
Example: (Right) Top Near Corner

\[
\begin{bmatrix}
N & 0 & 0 & 0 \\
0 & N & 0 & 0 \\
0 & 0 & N+F & -F \cdot N \\
0 & 0 & 1 & 0
\end{bmatrix}
\cdot
\begin{bmatrix}
R \\
T \\
N \\
1
\end{bmatrix}
= \begin{bmatrix}
R \cdot N \\
T \cdot N \\
N \cdot (N+F) - F \cdot N \\
N
\end{bmatrix}
\sim
\begin{bmatrix}
R \\
T \\
N \\
1
\end{bmatrix}
\]
Example: (Left) Top Far Corner

\[
\begin{bmatrix}
N & 0 & 0 & 0 \\
0 & N & 0 & 0 \\
0 & 0 & N+F & -F\cdot N \\
0 & 0 & 1 & 0
\end{bmatrix}
\cdot
\begin{bmatrix}
L\cdot F/N \\
T\cdot F/N \\
F \\
1
\end{bmatrix}
= \begin{bmatrix}
L\cdot F \\
T\cdot F \\
F\cdot (N+F) - F\cdot N \\
F
\end{bmatrix}
\sim
\begin{bmatrix}
L \\
T \\
F \\
1
\end{bmatrix}
\]
Perspective Transform
Nonlinear z-Behaviour
From Object Space to Screen Space

- **Object Space**
- **World Space**
- **Camera Space**
- **Clip Space**
- **Pixel Space**

- **Modeling Transformation**
- **Camera Transformation**
- **Projection Transformation**
- **Viewport Transformation**

"view frustum"
Viewing: Camera + Projection + Viewport

\[
\begin{bmatrix}
x_{\text{screen}} \\
y_{\text{screen}} \\
z' \\
1
\end{bmatrix} = (M_{\text{vp}} \cdot M_{\text{orth}} \cdot P \cdot M_{\text{cam}}) \cdot 
\begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix}
\]

pixels on the screen
projection transformation
viewport transformation
perspective transformation

world coordinates
From Object Space to Screen Space

- **Object Space**
- **Modeling Transformation**
- **World Space**
- **Camera Transformation**
- **Camera Space**
- **Projection Transformation**
- **Clip Space**
- **Viewport Transformation**
- **Pixel Space**

"view frustum"
z-Values Remain in Order

\[
\begin{bmatrix}
N & 0 & 0 & 0 \\
0 & N & 0 & 0 \\
0 & 0 & N+F & -F \cdot N \\
0 & 0 & 1 & 0
\end{bmatrix}
\cdot
\begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix}
\Rightarrow
\begin{bmatrix}
x \cdot N/z \\
y \cdot N/z \\
(N+F) - F \cdot N/z \\
1
\end{bmatrix}
\]

\[z_1, z_2, N, F < 0\]

\[z_1 < z_2\]

\[1/z_1 > 1/z_2\]

\[- F \cdot N/z_1 < - F \cdot N/z_2\]

\[(N+F) - F \cdot N/z_1 < (N+F) - F \cdot N/z_2\]
Perspective Projection Properties

- parallel lines parallel to view plane ⇒ parallel lines
- parallel lines not parallel to view plane ⇒ converging lines (vanishing point)
- lines parallel to coordinate axis ⇒ principal vanishing point (one, two or three)
Principle Vanishing Points

- 1-point perspective projection
- 2-point perspective projection
- 3-point perspective projection

vanishing point