Computer Graphics

1 Pipeline

IN Application
- Vertex Processing
- Triangle Processing
- Rasteriztion
- Fragment Processing
- Framebuffer Operations

OUT Display

2 Transformation

Translation

$$\mathbf{T}(t_x,t_y,t_z) = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotation

$$\mathbf{R}(\phi) = \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix}$$

Scale

$$\mathbf{S}(s_x,s_y,s_z) = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Shear

$$\mathbf{H}_{xz}(s) = \begin{pmatrix} 1 & 0 & s & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

General Transformation (TRaSh)

TRSO

Point

$$(x,y,z,1)^T$$

Normalized Device Coordinates (NDC)

$$(x,y,z,w)^T = (\frac{x}{w}, \frac{y}{w}, \frac{z}{w})^T$$

Vector

$$(x,y,z,0)^T$$

Euler Angles (roll, pitch, yaw)

$$\mathbf{R}_{xyz}(\alpha,\beta,\gamma) = \mathbf{R}_x(\alpha)\mathbf{R}_y(\beta)\mathbf{R}_z(\gamma)$$

Rodrigues' Rotation Formula: Rotation by angle α around axis $\mathbf{n}$.

$$\begin{aligned} \mathbf{R}(\mathbf{n},\alpha) \\ &= \cos(\alpha)\mathbf{I} \\ &\quad + (1-\cos(\alpha))\mathbf{n}\mathbf{n}^T \\ &\quad + \sin(\alpha) \begin{pmatrix} 0 & -n_z & n_y \\ n_z & 0 & -n_x \\ -n_y & n_x & 0 \end{pmatrix} \end{aligned}$$

View Transformation (Book p. 67)

$$M_{\text{view}} = R_{\text{view}} T_{\text{view}}$$

$$T_{\text{view}} = \begin{pmatrix} 1 & 0 & 0 & -x_e \\ 0 & 1 & 0 & -y_e \\ 0 & 0 & 1 & -z_e \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$R_{\text{view}} = \begin{pmatrix} x_{\hat{g}\times\hat{t}} & y_{\hat{g}\times\hat{t}} & z_{\hat{g}\times\hat{t}} & 0 \\ x_t & y_t & z_t & 0 \\ x_{-g} & y_{-g} & z_{-g} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Orthographic Projection (Book p. 94)

$$\begin{aligned} \mathbf{P}_o &= \mathbf{S}(\mathbf{s})\mathbf{T}(\mathbf{t}) \\ &= \begin{pmatrix} \frac{2}{r-l} & 0 & 0 & 0 \\ 0 & \frac{2}{t-b} & 0 & 0 \\ 0 & 0 & \frac{2}{n-f} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ &\begin{pmatrix} 1 & 0 & 0 & -\frac{r+l}{2} \\ 0 & 1 & 0 & -\frac{t_b}{2} \\ 0 & 0 & 1 & -\frac{n+f}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & \frac{2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{aligned}$$

Perspective Transform Matrix (Book p. 99)

$$\mathbf{P}_p = \begin{pmatrix} \frac{n}{r} & 0 & 0 & 0 \\ 0 & \frac{n}{t} & 0 & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

Canonical Cube to Screen: transform $[-1,1]^2$ to $[0,w]\times[0,h]$.

$$\mathbf{M}_{\text{viewport}} = \begin{pmatrix} \frac{w}{2} & 0 & 0 & \frac{w}{2} \\ 0 & \frac{h}{2} & 0 & \frac{h}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

3 Shading Basics

Gooch Shading Model (Book p. 104)

$$\begin{aligned} \mathbf{c}_{\text{shaded}} =& s\mathbf{c}_{\text{hightlight}} \\ &+ (1-s)(t\mathbf{c}_{\text{warm}} + (1-t)\mathbf{c}_{\text{cool}}) \end{aligned}$$

Transparency (**over** operator)

$$\mathbf{c}_o = \alpha_s \mathbf{c}_s + (1-\alpha_s)\mathbf{c}_d$$

4 Illumination

Diffuse Shading (**Lambertian**)

$$L_d = k_d \frac{I}{r^2} \max(0, \mathbf{n}\cdot\mathbf{l})$$

Specular Shading (**Blinn-Phong**)

$$L_s = k_s \frac{I}{r^2} \max(0, \mathbf{n}\cdot\mathbf{h})^p$$

where $\mathbf{h} = \text{bisector}(\mathbf{v},\mathbf{l})$ (see p. 336).
Ambient Shading

$$L_a = k_a I_a$$

Blinn-Phong Reflection Model

$$\begin{aligned} L &= L_a + L_d + L_s \\ &= k_a I_a + k_d \frac{I}{r^2} \max(0,\mathbf{n}\cdot\mathbf{l}) \\ &+ k_s \frac{I}{r^2} \max(0,\mathbf{n}\cdot\mathbf{h})^p \end{aligned}$$

Flat Shading shade each triangle/-face.
Gouraud Shading shade each vertex.
Phong Shading shade each pixel.
MSAA p. 139

5 Texture

Corresponder function
- wrap
- mirror
- clamp
- border

Barycentric Coordinates

$$(x,y) = \alpha A + \beta B + \gamma C$$

where $\alpha + \beta + \gamma = 1$.

$$\alpha = \frac{S_{BxC}}{S}$$
$$\beta = \frac{S_{AxC}}{S}$$
$$\gamma = \frac{S_{AxB}}{S}$$

Bump Mapping Access a texture to modify the surface normal instead of using a texture to change a color component in the illumination equation. Store three vectors: vertex normal $\mathbf{n}$, tangent $\mathbf{t}$, and bitangent $\mathbf{b}$.

$$\begin{pmatrix} t_x & t_y & t_z & 0 \\ b_x & b_y & b_z & 0 \\ n_x & n_y & n_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

A problem with bump and normal mapping is that the bumps never shift location with the view angle, nor ever block each other.

Parallax Mapping take an educated guess of what should be seen in a pixel by examining the height of what was found to be visible. The bumps are stored in a heightfield texture.

Environment Map A function from the sphere to colors, stored as a texture.

6 Geometry

Represent Geometry

- Implicit
  - Level set
  - Algebraic surface $f(\mathbf{p}) < 0$ Inside
  - Distance functions
- Explicit
  - Point cloud
  - Polygon mesh
  - Subdivision, NURBS (p. 781)

Bézier Curve (p. 720)

$$\mathbf{b}_0^1(\mathbf{t}) = (1-t)\mathbf{b}_0 + t\mathbf{b}_1$$
$$\mathbf{b}_0^2(\mathbf{t}) = (1-t)^2\mathbf{b}_0 + 2t(1-t)\mathbf{b}_1 + t^2\mathbf{b}_2$$
$$\cdots$$
$$\mathbf{b}_0^n(t) = \sum_{j=0}^{n} B_j^n(t)\mathbf{b}_j$$

where Bernstein polynomials

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

de Casteljau Algorithm (p. 737)

7 Mesh

Local Mesh Operations

Edge Flip Change the division method of triangles

Edge Split Get more triangles

Edge Collapse Replace edge with a single vertex

Global Mesh Operations

Mesh Subdivision upsampling

Mesh Simplification downsampling — Quadric Error Metrics (p. 708)

Mesh Regularization same number of triangles

Loop Subdivision Split each triangle into four (p. 758)

Catmull-Clark Subdivision Regular Quad Mesh (p. 762)

8 Raytracing

Ray Casting : Perform shading calculation here to computer color of pixel (e.g. Blinn Phong model)

Recursive Ray Tracing (Whitted-Style)

1. Trace secondary rays recursively until hit a non specular surface (or max desired levels of recursion)
2. At each hit point, trace shadow rays to test light visibility (no contribution if blocked)
3. Final pixel color is weighted sum of contributions along rays, as shown
4. Gives more sophisticated effects (e.g. specular reflection, refraction, shadows), but we will go much further to derive a physically based illumination model

Ray Equation (p. 943)

$$\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$$

Plane Equation

$$(\mathbf{p} - \mathbf{p}') \cdot \mathbf{N} = 0$$

AABB (Axis Aligned Bounding Box) p. 944

OBB (Oriented Bouding Box) p.945

Ray intersection with AABB

$$t_{\text{enter}} = \max t_{\min}$$
$$t_{\text{exit}} = \min t_{\max}$$
$$\begin{cases} t_{\text{enter}} < t_{\text{exit}} \\ t_{\text{exit}} \geq 0 \end{cases}$$

Spatial Partitioning

- Oct-Tree p. 824
- KD-Tree p. 822
- BSP-Tree p. 823