Hybrid Forward-Backward Reflection Rendering

Chunhui Mei

Mar 2006
Reflection Rendering

- Ray tracing – backward, slow
- Environment mapping – fast, not accurate
- Model morphing – forward, only for convex shape
- Approximation – billboard…
Reflection Rendering

- $N_{\text{pixel}} = 1M$
- $N_{\text{ver-d}} = 1K$
- $N_{\text{tri-d}} = 2K$
- $N_{\text{vex-r}} = 1K$
- $N_{\text{tri-r}} = 2K$
- $F_{\text{acc}} = 1/1000$
- $N_{\text{ray}} = 1M$
- $N_{\text{cluster}} = 2K$
Reflection Rendering

- Ray tracing

\[ N_{\text{pixel}} \times (N_{\text{tri-d}} \times F_{\text{acc}} \times T_{\text{inter}} + T_{\text{trav}}) \]

\[ 2M \times T_i + 1M \times T_t \]
Reflection Rendering

- Reflection morphing

\[ N_{\text{ver-d}}(N_{\text{ray}} \cdot F_{\text{acc}} \cdot T_{\text{inter}} + T_{\text{trav}}) \]

\[ 2M \cdot T_i + 1M \cdot T_t \]

\[ 1M \cdot T_i + 1K \cdot T_t \]
Reflection Rendering

- Ray tracing with clustering

\[ N_{\text{cluster}} \cdot T_{\text{trav}} + N_{\text{pixel}} \cdot N_{\text{tri-d}} \cdot F_{\text{acc}} \cdot T_{\text{inter}} \]

- \[ 2M \cdot T_i + 1M \cdot T_t \]
- \[ 1M \cdot T_i + 1K \cdot T_t \]
- \[ 2M \cdot T_i + 2K \cdot T_t \]
Reflection Rendering

• Hybrid backward forward rendering
  – Trace ray-cone in acceleration structure
  – Obtain the vertices which have potential projections
  – Project the vertices by general 3 ray camera
  – Generate reflection image using projected vertices
Reflection Rendering

- Hybrid approach

$$N_{\text{cluster}} \times (T_{\text{trav}} + N_{\text{ver-d}} \times F_{\text{acc}} \times T_{\text{proj}})$$

2M*Ti + 1M*Tt
1M*Ti + 1K*Tt
2M*Ti + 2K*Tt
2K*Tp + 2K*Tt
Hybrid Reflection Rendering

• Process
  – K-D tree construction of diffuse scene
  – For every triangle on reflective surface
    • Build general 3 ray camera
    • Generate bounding cone of 3 ray camera
    • Trace the cone into K-D tree
    • Obtain vertex set in leaf node intersect with the cone
    • Cull out vertices outside of cone
    • Project the vertices onto the reflective triangle
  – Render reflection image with projection of all vertices
Three Ray Camera

- Ray interpolation inside of triangle
Three Ray Camera

- Parallel 3 ray camera
- Projection using parallel 3 ray camera
- Discontinuity between neighboring parallel 3 ray cameras
Three Ray Camera

- General 3 ray camera
- Continuous between neighboring triangles
- Nonlinear projection
General 3 ray camera

- Bounding cone of 3 ray camera
- Convenient to tracing into K-D tree
- Fast culling for vertex outside of 3 ray camera
General 3 ray camera

• Projection

\[ \vec{V}_{AA} = l \times \vec{r}_A \]
\[ \vec{V}_{BB} = l \times \vec{r}_B \]
\[ \vec{V}_{CC} = l \times \vec{r}_C \]
\[ \vec{N} = \vec{V}_{AB} \times \vec{V}_{AC} \]
\[ \vec{V}_{AP} \cdot \vec{N} = 0 \]

\[ W_A = W_A = (\vec{V}_{BP} \times \vec{V}_{CP}) / (\vec{V}_{AB} \times \vec{V}_{AC}) \]
\[ W_B = W_B = (\vec{V}_{AP} \times \vec{V}_{CP}) / (\vec{V}_{AB} \times \vec{V}_{AC}) \]
\[ W_C = W_C = (\vec{V}_{BP} \times \vec{V}_{CP}) / (\vec{V}_{AB} \times \vec{V}_{AC}) \]

\[ Q = W_A \times \vec{V}_A + W_B \times \vec{V}_B + W_C \times \vec{V}_C \]

• Projection speed 3.5M/s
Hybrid Reflection Rendering

• Projection result
Hybrid Reflection Rendering

- Projection result
Hybrid Reflection Rendering

• Triangulation of projected vertices
  – Connect the nearest vertices for projected triangle
  – Avoid skinning triangles
  – Avoid holes
Hybrid Reflection Rendering
Hybrid Reflection Rendering

• Other approaches in process
  – Point based rendering
  – Clipping for general 3 ray camera
  – Filling the holes with ray tracing
Thank you