Projective texture mapping.

Shadow maps

Mapping from desired to reference image

\[
\begin{align*}
\mathbf{C}_1 + (\mathbf{C}_1 + u_1 \mathbf{a}_1 + v_1 \mathbf{b}_1)w_1 &= \mathbf{C}_1 + (\mathbf{C}_2 + u_2 \mathbf{a}_2 + v_2 \mathbf{b}_2)w_2 \\
[\mathbf{w}_1 \mathbf{u}_1 \mathbf{v}_1 \mathbf{w}_2] &= \mathbf{C}_1 + (\mathbf{C}_2 + [\mathbf{u}_2 \mathbf{v}_2 \mathbf{w}_2])w_2 \\
[\mathbf{w}_1 \mathbf{u}_1 \mathbf{v}_1 \mathbf{w}_2] &= \mathbf{C}_1 + (\mathbf{C}_2 + [\mathbf{u}_2 \mathbf{v}_2 \mathbf{w}_2])w_2 \\
[\mathbf{w}_1 \mathbf{u}_1 \mathbf{v}_1 \mathbf{w}_2] &= \mathbf{C}_1 + (\mathbf{C}_2 + [\mathbf{u}_2 \mathbf{v}_2 \mathbf{w}_2])w_2 \\
\end{align*}
\]

where

\[
\begin{align*}
\mathbf{u}_2 &= \frac{q_{a2} + q_{b2} + q_{c2} + q_{d2}}{w_1} \\
\mathbf{v}_2 &= \frac{q_{a2} + q_{b2} + q_{c2} + q_{d2}}{w_1} \\
\mathbf{w}_1 &= \frac{q_{a1} + q_{b1} + q_{c1} + q_{d1}}{w_1} \\
\frac{1}{w_1} &= A_{11} + B_{12} + C \\
\end{align*}
\]

\( (C_2, a_2, b_2, c_2) \) – reference view
\( (u_2, v_2) \) – reference pixel coordinates
\( (C_1, a_1, b_1, c_1) \) – desired view
\( (u_1, v_1) \) – desired view pixel coordinates
\( (u_2, v_2, w_2) \) – unknowns
\( w_1 \) – computed using triangle \( V_0V_1V_2 \)
Shadow Maps

• Efficient implementation of shadows
• Essentially a zbuffer rendered from the light
  – if a point is behind the map as seen from the light, it is in shadow
  – the z-values model the first-surfaces seen from the light

Shadow Map Implementation

• Step 1: construction
  – one per light
  – updated when light or objects move
  – does not need to be updated when only the camera moves
  – resolution according to
    • scene geometry
    • desired image resolution
    • desired shadow quality
    • budget
  – view should
    • cover all light rays
    • cover all scene (cube maps if needed)
    • near / far plane according to scene bounding box
Shadow Map Implementation

- Step 2: shadow computation
  - project scene point visible at current pixel onto shadow map(s)
  - if hidden, pixel is in shadow
  - else light contributes to pixel color
- soft shadows
  - pixels close to the shadow border are partially in shadow (penumbra)
  - implemented by testing neighborhood in the shadow map
    - if all samples of neighborhood are in shadow -> shadow
    - if all samples of neighborhood are in the light -> light
    - if \( k \) of \( n \) samples are in the light -> penumbra (light contribution \( k/n \))