Interactive Display Algorithm for Interactive Frame Rates During Visualization Of Complex Virtual Environments
Goal
Goal

- Interactive walkthrough
- Frame rate should be
  - Interactive
  - Consistent
Frame Render Time

a) No detail elision.
Level of Detail

- Can’t render all potentially visible geometry
- Trade visual quality for speed
Level of Detail

- How do we select the level of detail to render?
- Static
  - Distance, Approximate screen size
- Feedback
  - Update thresholds based on
Frame Render Time

\[ T(s) \]

\[ \text{Frames} \]

\[ 0 \quad 250 \]

b) *Static* algorithm.
Frame Render Time

c) Feedback algorithm.
Predictive vs. Reactive

- Feedback approach is better
- But varies, sometimes above goal render time
- We need to predict the performance instead of reacting to it
Approach - Definitions

- Object tuple $(O, L, R)$
  - $O$ - object
  - $L$ - level of detail
  - $R$ - rendering algorithm
- $\text{Cost}(O, L, R)$
- $\text{Benefit}(O, L, R)$
Approach

• Choose 1 object tuple for each object

• Maximize $\sum_{S} \text{Benefit}(O, L, R)$

• Subject to $\sum_{S} \text{Cost}(O, L, R) \leq \text{TargetFrameTime}$
Cost Heuristic

- Depends on hardware - measure and fit to model
- Model - assume per-primitive or per-pixel operations dominate frame time

\[
\text{Cost}(O,L,R) = \max( C_1 \text{Poly}(O,L) + C_2 \text{Vert}(O,L), C_3 \text{Pix}(O) )
\]
Benefit Heuristic

• Value of object to user
• Perceptual
• Heuristic could be application dependent
• Simple heuristic - approximate # of pixels
Benefit Heuristic

- **Accuracy** = 1 - Error = 1 - BaseError/Samples^m
- **Intuition** - type of shading indicates avg error, error decreases with more samples
- m and BaseError set arbitrarily based on shading
- **Samples** = # pixels, vertices, or polygons
Benefit Heuristic

• Semantics - application specific
• Focus - eye-tracking, middle of screen
• Motion Blur - screen space speed
• Hysteresis - changing LOD can be bothersome
Optimization

- Choose 1 object tuple for each object

- Maximize \( \sum_S \text{Benefit}(O, L, R) \)

- Subject to \( \sum_S \text{Cost}(O, L, R) \leq \text{TargetFrameTime} \)
Optimization

- Multiple Choice Knapsack Problem
- NP-Complete
- Approximate
Approximation

- Value = Benefit / Cost
- Consider tuples in order of descending value
  - If a tuple with same O is in S, keep the one with greater Benefit
  - Else add tuple to S
- Terminate when adding another tuple violates cost
Approximation

- $O(n \lg n)$ for $n$ potentially visible objects
- At least $1/2$ as good as optimal solution
Incremental Approximation

• Start with $S$ from last frame + lowest LOD for newly visible objects
• Increment LOD for object with highest subsequent Value
• Decrement LOD of lowest Value objects until Cost criteria satisfied
• Terminate when same object LOD is both increased and decreased
Incremental Approximation

- Finds same set as regular implementation if Value decreases monotonically for object LODs
- Worst case $O(n \lg n)$
- But usually better since initial set is usually close to selected set
- Rendering and optimization parallelized
Results

- For comparison, Benefit = Object size
Frame Render Time

d) *Optimization* algorithm.
Results

a) No detail elision.

b) Static algorithm.

c) Feedback algorithm.

d) Optimization algorithm.
## Results

<table>
<thead>
<tr>
<th>LOD Selection Algorithm</th>
<th>Compute Time</th>
<th></th>
<th>Frame Time</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Max</td>
<td>Mean</td>
<td>Max</td>
<td>StdDev</td>
</tr>
<tr>
<td>None</td>
<td>0.00</td>
<td>0.00</td>
<td>0.43</td>
<td>0.99</td>
<td>0.305</td>
</tr>
<tr>
<td>Static</td>
<td>0.00</td>
<td>0.01</td>
<td>0.11</td>
<td>0.20</td>
<td>0.048</td>
</tr>
<tr>
<td>Feedback</td>
<td>0.00</td>
<td>0.01</td>
<td>0.10</td>
<td>0.16</td>
<td>0.026</td>
</tr>
<tr>
<td>Optimization</td>
<td>0.01</td>
<td>0.03</td>
<td>0.10</td>
<td>0.13</td>
<td>0.008</td>
</tr>
</tbody>
</table>
Results
Results

b) Optimization algorithm (0.15 seconds)
c) Optimization algorithm (0.10 seconds)
Discussion

• Complementary to PVS, culling
• Flexible - applicable to other systems
• No results for more complicated Benefit heuristic
• Conservative Heuristics? Perceptually based?
• Accurate Heuristics?
Discussion

- Blending LODs
- Continuous Level of Detail (e.g. Progressive Meshes)
- Hierarchical Level of Detail
- This paper isn’t winning any awards for its title.
Questions?