Acceleration Techniques for Parallel Real Time Ray Tracing

Programação 3D para Simulação de Jogos

Vasco Alexandre da Silva Costa
Mobile stream computing hardware has performance in the order of 100s of Gflops

Workstation stream computing hardware has performance around 1 Tflop

We need to be able to harness these capabilities

- AMD Vishera CPU
- NVIDIA Kepler GPU
- AMD Trinity APU
Acceleration Structures
Acceleration Structures

**BVHs** and **Grids** are the most popular subdivisions:

- Suitable for animated scenes due to respectively fast updates, rapid construction
- Easier to parallelize in stream computing architectures
- Array of primitive references

- Cells are ranges in the array
Grids

- Reduce to sorting

1. Write pairs of references and cell indices
2. Sort
3. Extract cell ranges
Grids

Input

Grid Cells

<table>
<thead>
<tr>
<th></th>
<th>[0,2)</th>
<th>[2,3)</th>
<th>[3,4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>[4,5)</td>
<td>[0,0)</td>
<td>[0,0)</td>
<td></td>
</tr>
</tbody>
</table>

|       | [0,0) | [0,0) |

Write Pairs

|       | 0A    | 1A    | 2A    | 0B    | 3B    |

(Cell ID, Prim ID)

Radix Sort

Sorted Pairs

|       | 0A    | 0B    | 1A    | 2A    | 3B    |

Javor Kalojanov, Markus Billeter and Philipp Slusallek, EG 2011
Prefix Sum aka Scan

For example, if $\oplus$ is addition, then the all-prefix-sums operation on the array

$$[3 \ 1 \ 7 \ 0 \ 4 \ 1 \ 6 \ 3]$$

would return

$$[0 \ 3 \ 4 \ 11 \ 11 \ 15 \ 16 \ 22].$$

Parallel Prefix Sum (Scan) with CUDA, GPU Gems 3
Compact Grids

Ares Lagae and Philip Dutré, EGSR 2008
Hashed Grids

Ares Lagae and Philip Dutré, EGSR 2008
Bounding Volume Hierarchy

Triangles

Leaf nodes

Triangles
Bounding Volume Hierarchy

- Axis-aligned bounding box (AABB)
- Internal nodes
- Groups of triangles
Bounding Volume Hierarchy

Recursive grouping
Bounding Volume Hierarchy

Descend from the root

Ray misses AABB → skip subtree

Ray intersection query
Bounding Volume Hierarchy

- Morton code assignment
- Triangle sorting
- Hierarchy generation
- AABB fitting

Triangle centroid
### Bounding Volume Hierarchy

#### Z-order and Morton code

<table>
<thead>
<tr>
<th>$y_2$</th>
<th>$0$</th>
<th>$1$</th>
<th>$2$</th>
<th>$3$</th>
<th>$4$</th>
<th>$5$</th>
<th>$6$</th>
<th>$7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y_1$</td>
<td>$000$</td>
<td>$001$</td>
<td>$010$</td>
<td>$011$</td>
<td>$100$</td>
<td>$101$</td>
<td>$110$</td>
<td>$111$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_2$</td>
<td>$0$</td>
<td>$000$</td>
<td>$001$</td>
<td>$010$</td>
<td>$011$</td>
<td>$100$</td>
<td>$101$</td>
<td>$110$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_1$</td>
<td>$000$</td>
<td>$00000$</td>
<td>$00010$</td>
<td>$00011$</td>
<td>$00100$</td>
<td>$00101$</td>
<td>$01000$</td>
<td>$01001$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$001$</td>
<td>$00010$</td>
<td>$00001$</td>
<td>$00100$</td>
<td>$00111$</td>
<td>$01001$</td>
<td>$01000$</td>
<td>$01011$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$010$</td>
<td>$00100$</td>
<td>$00101$</td>
<td>$00110$</td>
<td>$00111$</td>
<td>$01010$</td>
<td>$01011$</td>
<td>$01100$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$011$</td>
<td>$01000$</td>
<td>$01001$</td>
<td>$01010$</td>
<td>$01011$</td>
<td>$01100$</td>
<td>$01101$</td>
<td>$01110$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$100$</td>
<td>$10000$</td>
<td>$10001$</td>
<td>$10010$</td>
<td>$10011$</td>
<td>$11000$</td>
<td>$11001$</td>
<td>$11010$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$101$</td>
<td>$10010$</td>
<td>$10011$</td>
<td>$10100$</td>
<td>$10101$</td>
<td>$11010$</td>
<td>$11011$</td>
<td>$11100$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$110$</td>
<td>$10100$</td>
<td>$10101$</td>
<td>$10110$</td>
<td>$10111$</td>
<td>$11100$</td>
<td>$11101$</td>
<td>$11110$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$111$</td>
<td>$11000$</td>
<td>$11001$</td>
<td>$11010$</td>
<td>$11011$</td>
<td>$11110$</td>
<td>$11111$</td>
<td>$11111$</td>
</tr>
</tbody>
</table>
Bounding Volume Hierarchy

- Morton code assignment
- Triangle sorting
- Hierarchy generation
- AABB fitting

Space-filling curve → nearby triangles appear next to each other
Bounding Volume Hierarchy

- Morton code assignment
- Triangle sorting
- Hierarchy generation
- AABB fitting

Sorted Morton codes

Find the highest differing bit

Root covers all triangles
Bounding Volume Hierarchy

Morton code assignment

Triangle sorting

Hierarchy generation

AABB fitting

Split recursively
Bounding Volume Hierarchy

Morton code assignment

Triangle sorting

Hierarchy generation

AABB fitting

Split recursively
Bounding Volume Hierarchy

Morton code assignment

Triangle sorting

Hierarchy generation

AABB fitting

Split recursively
Bounding Volume Hierarchy

- Morton code assignment
- Triangle sorting
- Hierarchy generation
- AABB fitting
Bounding Volume Hierarchy

- Morton code assignment
- Triangle sorting
- Hierarchy generation
- AABB fitting

Union of child boxes

Triangle bounding boxes
Bounding Volume Hierarchy

- Morton code assignment
- Triangle sorting
- Hierarchy generation
- AABB fitting

Bottom-up order
Bounding Volume Hierarchy

Compute centroids

Sort along 1D Morton curve through a grid

Group by cell
Ray Reordering
Ray Reordering: Ray Hash Function

Figure 2: The quantization of ray origin and direction is used to compute a hash value for a given ray.
Ray Reordering: CSD

**Figure 4:** Compression example.
Ray Reordering: CSD

Figure 5: Decompression example.
Ray Reordering: CSD

Figure 3: The overall ray sorting scheme.
Ray Reordering: Frustums

Primary Rays
- Generate Rays
- Sort Rays
- Build Frustums
- Frustums Traversal
- Local Intersection Tests
- Accumulate Shading

Secondary Rays
Ray Reordering: Frustums Traversal

Figure 8: Four examples of node’s children ordering along the frustum are presented in a 2D projection.
References

- A Parallel Algorithm for Construction of Uniform Grids, Javor Kalojanov, Philipp Slusallek, HPG 2009

- Parallel Hierarchy Construction, Tero Karras, HPG 2013

- Fast Ray Sorting and Breadth-First Packet Traversal for GPU Ray Tracing, Kirill Garanzha, Charles Loop, EG 2010