Improving BVH Ray Tracing Speed Using the AVX Instruction Set

Attila T. Áfra
Budapest University of Technology and Economics, Hungary
Babeș-Bolyai University, Cluj-Napoca, Romania

- **SSE**: very popular SIMD instruction set, 128-bit, 4 floats *(1999)*
- **AVX**: introduced with Intel Sandy Bridge, 256-bit, 8 floats *(2011)*

- BVH ray packet traversal algorithms: **ranged**, **partition**, etc.
- Smallest ray primitive: **SIMD ray**
- **SSE**: 2×2 SIMD rays
- **AVX**: 4×2 SIMD rays
- Frustum culling: interval arithmetic (no SIMD) and corner rays (4-wide SIMD)

Rays are stored in **AoSoA** (array of structures of arrays) layout
- AoSoA combines the SIMD-friendliness of SoA with the locality of AoS
- **Example (3D vector, 4-wide SIMD):**

<table>
<thead>
<tr>
<th>X0</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>Y0</th>
<th>Y1</th>
<th>Y2</th>
<th>Y3</th>
<th>Z0</th>
<th>Z1</th>
<th>Z2</th>
<th>Z3</th>
<th>X4</th>
<th>X5</th>
<th>X6</th>
<th>X7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structure 0</td>
<td>Structure 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Performance**: AVX ~50% faster than SSE4.1
- Ranged traversal with frustum culling
- The sublinear speedup is due to larger SIMD rays with lower utilization and non-SIMD parts of the algorithm
- **Intel Core i5-2400** (4 cores, 4 threads, 3.1 GHz), 64-bit, Visual C++ 2010