STBVH: A Spatial-Temporal BVH for Efficient Multi-Segment Motion Blur

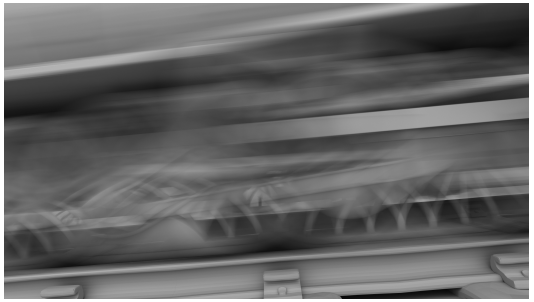
Sven Woop, Attila Áfra, Carsten Benthin

Intel Corporation

Motion Blur

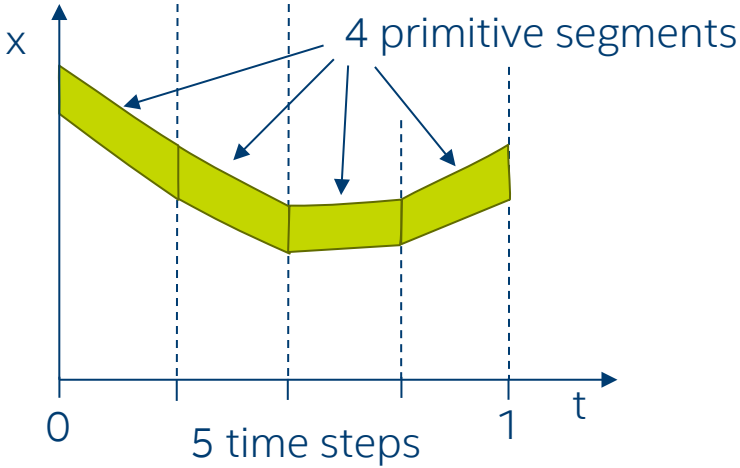
- Fast moving geometry gets blurred for long shutter times
- Often fast moving geometry moves on a straight line (linear motion blur)
- Sometimes fast curved motion (e.g. rotating wheel, fight scenes, spinning dancer, flying bird, etc.)

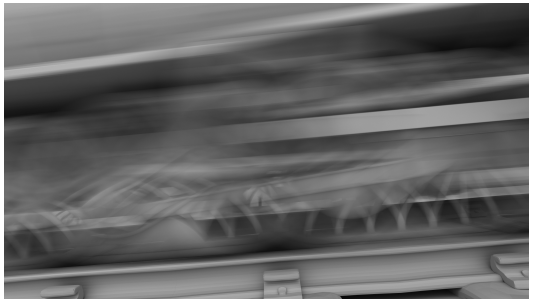
➔ Multi Segment Motion Blur required



Multi Segment Motion Blur

- Represent curved motion as sequence of time steps to be linearly interpolated
- Typically equidistant time steps and often different number of time steps per geometry





Previous Work

Linear Motion BVH using OBB Hyper-trapezoids [Hou et al. 2010]

- Works well for linear motion but inefficient for curved motion.

Multiple Linear Motion BVHs for sufficiently large number of time segments [Embree v2.12.0]

- High performance but memory consumption can be arbitrarily bad.

Sequence of AABBs per BVH node [Grünschloß et al. 2011]

- One BVH topology for entire motion, calculate segment to interpolate per traversal step, packet techniques have to gather bounds

4D kd-tree [Olsson 2007]

- Can shrink time range to simplify motion, kd-tree not good at bounding linear motion, no good build algorithm described

4D BVH using 12 fixed slab directions [Glassner 1988]

- Expensive to traverse (24 distance tests to mostly non-axis aligned planes), fixed directions do not align optimally with motion direction

Combining separate renderings for sufficiently many time segments

- No adaptive noise reduction possible, interactive preview not possible

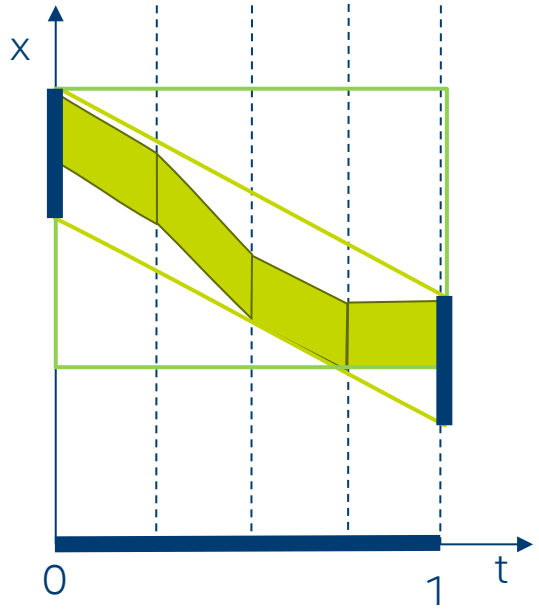
4D Spatial-Temporal BVH (STBVH)

- N-ary BVH (4 or 8 wide) [Ernst 2008, Dammertz 2008]
  - SOA layout allows efficient use of SIMD instructions during traversal
- Stores spatial *linear bounds* [Qiming Hou et al. 2010]
  - Pair of AABBs that bound the geometry for each time when linearly interpolated to the respective time
  - Efficient support for the common case of linear motion
- Stores temporal bounds as time range [Olsson 2007, Glassner 1988]
  - Efficient support of curved motion through time range reduction
- Two node types for improved performance
  - Spatial-temporal nodes (stores linear bounds and time range)
  - Spatial nodes (stores linear bounds only)

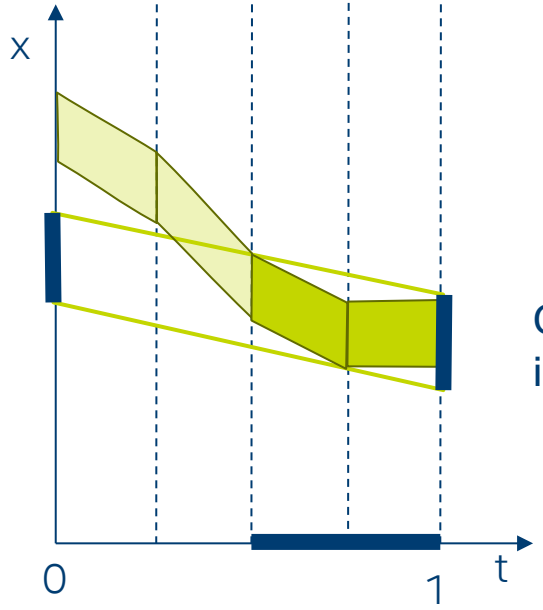
STBVH Advantages

- Efficient handling of different number of time steps per geometry
  - E.g. high temporal resolution possible for main character
  - Large memory savings compared to Embree 2.12.0 implementation
- Efficient handling of longer animations
  - Renderers with large setup times can render multiple frames with one STBVH
- Reduced memory consumption in case of unnecessarily high number of time steps
  - For these parts time ranges do not have to get reduced

Temporal Spatial Bounds Example 1



Temporal Spatial Bounds Example 2



Global linear bounds allow direct interpolation with ray time.

Minimal Traversal Changes

- Ray/box intersection with box interpolated to ray time
- Additional check for time bounds in case of spatial-temporal node

Motion Blur Surface Area Heuristic (MBSAH)

- Motion Blur Surface Area Heuristic
  - $C_{leaf}(X) = |X|_s \cdot C_I$
  - $C_{split}(X, X_0, X_1) = C_T + P(X_0|X) \cdot C_{leaf}(X_0) + P(X_1|X) \cdot C_{leaf}(X_1)$
- Where
  - X is the set of pairs of primitives and time ranges
  - $|X|_s$ is the sum of the number of *primitive segments* active in the time range
  - $P(Y|X) = \frac{SA'(Y)}{SA'(X)} \cdot \frac{T(Y)}{T(X)}$
  - $T(X)$ calculates the size of the merged time bounds over X
  - $SA'(X)$ calculates the surface area of the center time bounds of the linear bounds of X

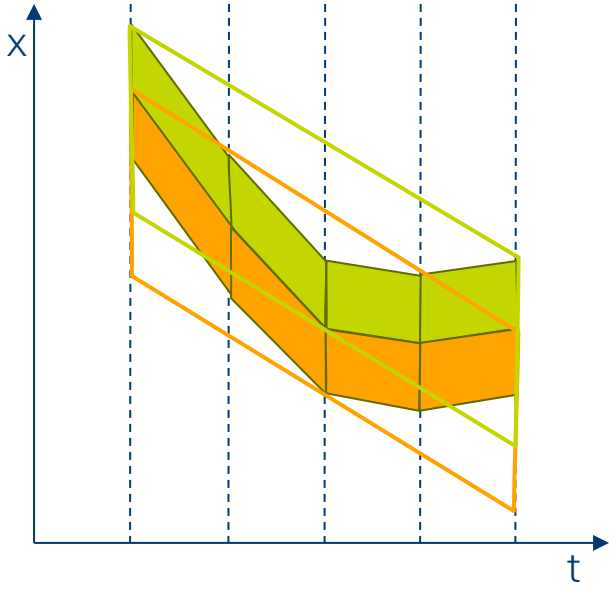
MBSAH Advantages

- Handling primitives plus time range
  - Makes time splits of primitives possible
- Counting primitive segments active in time range
  - Increases cost for geometries with many time steps
  - Splitting time at discrete time boundaries produces optimal SAH
- Surface area of linear bounds
  - More accurate than previous approaches
  - Up to 10% render performance improvement for some scenes

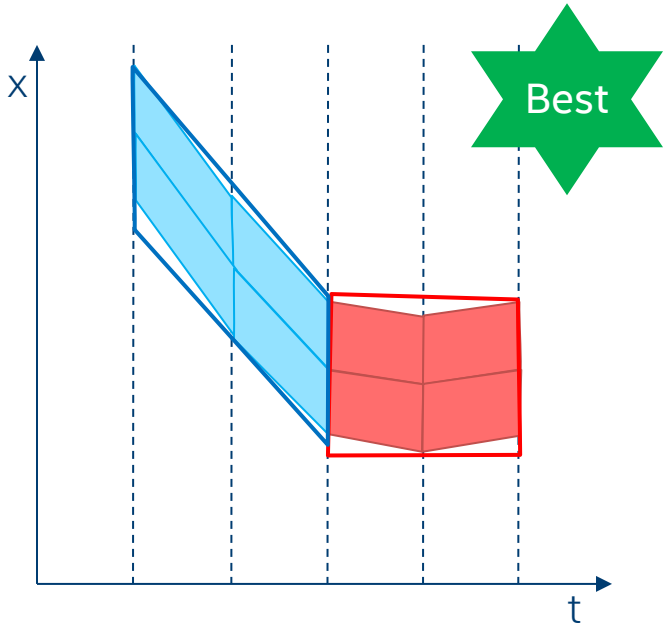
STBVH Build

- Top-down construction using MBSAH
- *Build primitive* represents primitive for current time range
  - Stores linear bounds and number of active primitive segments
- Object split
  - Bin build primitives in 3 dimensions using centroid of center time bounds
  - Splits build primitives into two disjoint sets with current time range unchanged
- Temporal split
  - Splits current time range at center time (adjusted to hit discrete time boundary)
  - Generate build primitives for both time ranges (most primitives valid in both time ranges)

MBSAH: Temporal Split

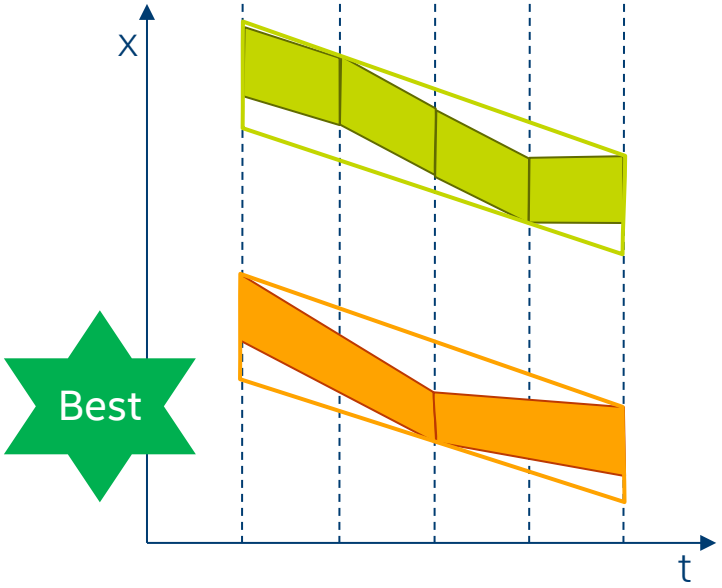


Object split

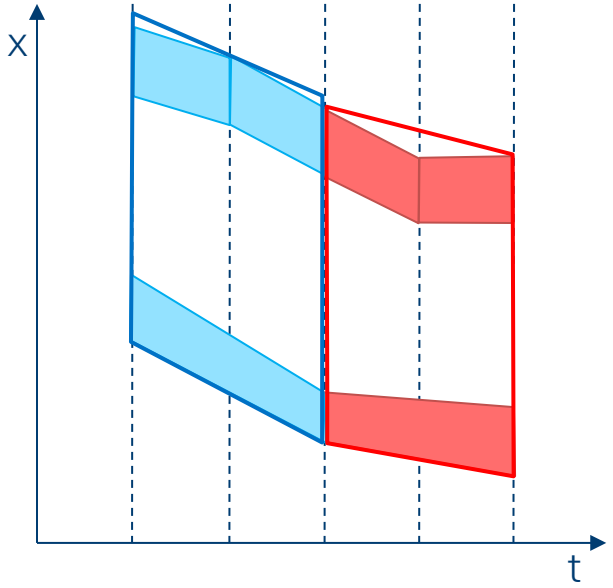


Temporal split

MBSAH: Spatial Split



Object split



Temporal split

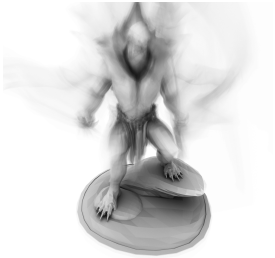
Results

Llama
3 time steps: 7M primitives
9 time steps: 1.7M primitives

Barbershop
3 time steps: 1.4M primitives
5 time steps: 2.8M primitives
9 time steps: 3.9M primitives

Train
3 time steps: 0.3M primitives
17 time steps: 2.0M primitives

Turtle Barbarian Crowd
2 time steps: 7.5M primitives
6 time steps: 2.8M primitives
15 time steps: 0.1M primitives



Turtle Barbarian
15 time steps: 0.1M primitives

Turtle Barbarian Rotate 0.5x
9 time steps: 0.1M primitives

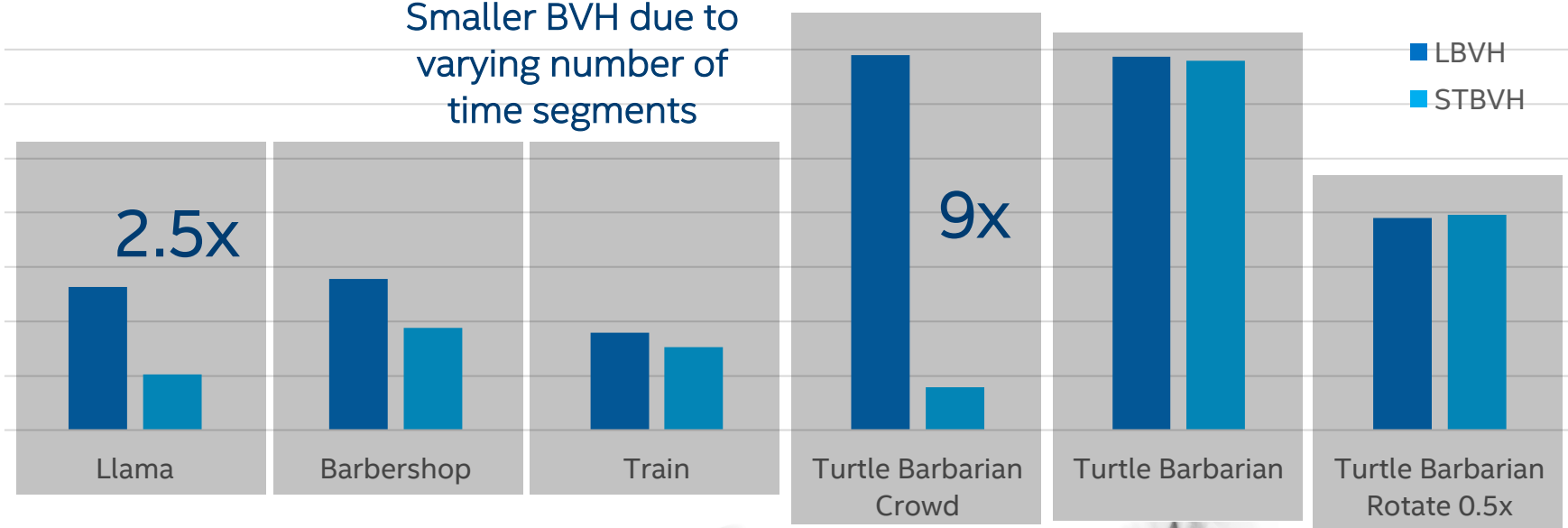
Results

- Comparing against LBVH of Embree 2.12.0
  - Separate Linear Motion BVHs for maximal number of linear time segments
  - We integrated our STBVH into Embree thus share algorithmic details of traversal and build
- Only motion blur geometry for benchmarks
- Intel® Xeon® E5-2699 v4 workstation (Broadwell 22 cores, 2.2 GHz)

Memory Consumption

Similar BVH size

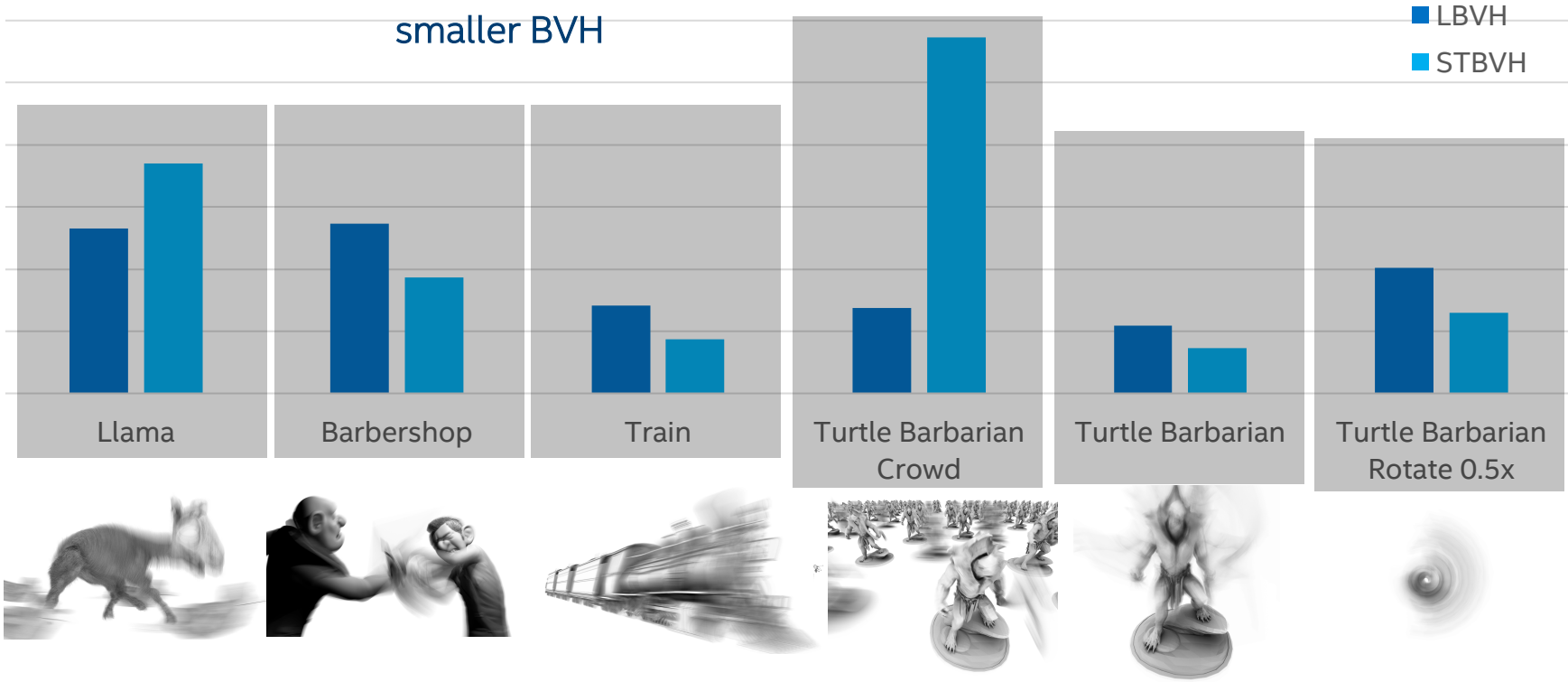
Smaller BVH due to varying number of time segments



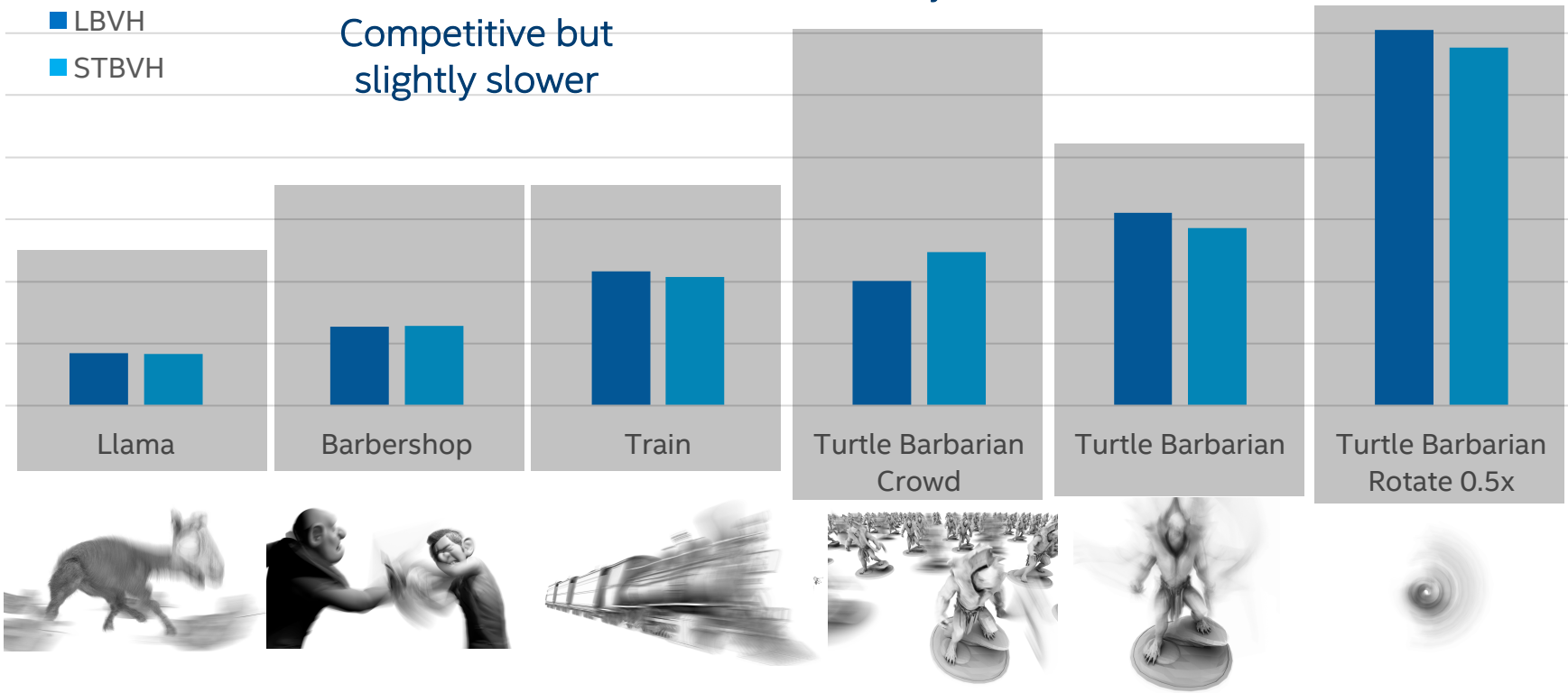
Build Performance

Faster due to smaller BVH

Slower due to linear bounds binning

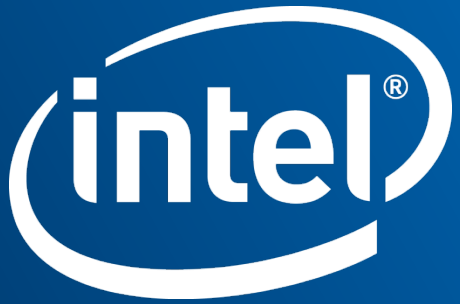


Render Performance



Questions?

“High Performance Rendering Appliance” demo
at Intel booth #807 at SIGGRAPH


intel®