



# Embree Ray Tracing Kernels: *Overview and New Features*

Sven Woop, Attila Áfra, Ingo Wald, Carsten Benthin

Intel Corporation

# Embree Overview

Embree API

Advanced Features

Embree Performance

# Usage of Ray Tracing Today

- Special effects in movies (better image quality, faster feedback)
- High quality rendering for product visualization
- Provides higher fidelity for automotive rendering, architectural design, etc.
- Various kind of simulations (lighting, sound, particles, collision detection, etc.)
- Prebaked lighting in games



# Fast Ray Tracing Challenges

## Multi-threading

- Easy for rendering but difficult for hierarchy construction

## Vectorization

- Efficient use of SIMD units, different ISAs (SSE, AVX, AVX2, AVX-512)

## Domain knowledge

- Many different data structures and algorithms to choose from

## Support for different CPUs

- Different ISAs/CPU types favor different data structures, data layouts, and algorithms

## Different usage scenarios

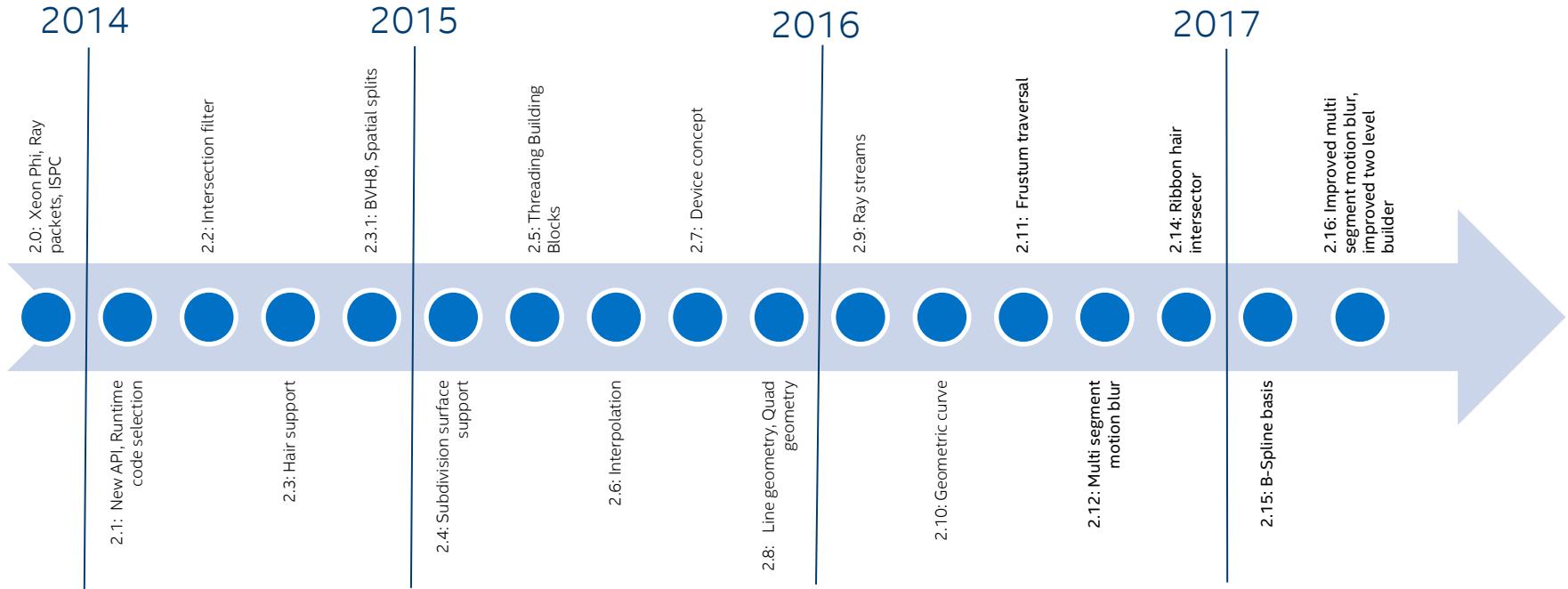
- e.g. large model visualization favors memory conservative algorithms

# Embree Ray Tracing Kernels

- Provides highly optimized and scalable ray tracing kernels
  - Focus on acceleration structure build and ray traversal
- Highest ray tracing performance on CPUs
  - 1.5–6× speedup reported by users
- Support for latest CPUs and ISAs (e.g. AVX-512)
- Targets professional rendering applications
- API for easy integration into applications
- Free and Open Source under Apache 2.0 license
  - <http://embree.github.com>



# Embree Timeline



# Embree Features

- Find closest hit (`rtcIntersect`), find any hit (`rtcOccluded`)
- Single rays, ray packets (4, 8, 16), ray streams (N)
- High-quality and high-performance BVH builders
- Triangles, quads, subdivs + displacement, curves, instances, user defined geometries
- Multi segment motion blur
- Intel® SPMD Program Compiler (ISPC) support
- Intel® Threading Building Blocks (TBB) support

# Embree System Overview

## Embree API (C and ISPC)

### Ray Tracing Kernel Selection

#### Acceleration Structures

bvh4.triangle4  
bvh8.triangle4  
bvh4.quad4v  
...

#### Builders

SAH Builder  
Spatial Split Builder  
Morton Builder  
BVH Refitter

#### Traversal

Single Ray  
Packet/Hybrid  
Ray Stream

#### Intersection

Möller-Trumbore  
Plücker  
Bézier Curve  
Line Segment  
Triangle Grid

#### Subdiv Engine

B-Spline Patch  
Gregory Patch  
Tessellation Cache  
Displ. Mapping

### Common Vector and SIMD Library

(Vec3f, Vec3fa, vfloat4, vfloat8, vfloat16, ..., SSE2, SSE4.1, AVX, AVX2, AVX-512)

Embree Overview

# Embree API

Advanced Features

Embree Performance

# Embree API Overview

- Version 2 of the Embree API (version 3 in progress)
- C and ISPC version
- Object oriented
- Easy to use
- Hides implementation details
- For details visit <https://embree.github.io/api.html>

# Example: Scene creation

Scene is a container for a set of geometries

Scene flags passed at creation time

- Static scene
- Dynamic scene
- etc.

Scene geometry changes have to get committed (`rtcCommit`), which triggers BVH build

```
// include Embree headers
#include <embree2/rtcore.h>

int main()
{
    // create Embree device at application startup
    RTCDevice device = rtcNewDevice ();

    // create scene
    RTCScene scene = rtcDeviceNewScene
        (device, RTC_SCENE_STATIC,
         RTC_INTERSECT1);

    // add geometries
    ... later slide ...

    // commit changes
    rtcCommit(scene);

    // trace rays
    ... later slide ...

}
```

# Example: Triangle Mesh creation

Triangle mesh contains vertex and index buffers

Number of triangles and vertices set at creation time

Shared buffers of flexible layout (offset + stride) supported

```
// application vertex and index layout
struct Vertex { float x, y, z, s, t; };
struct Triangle { int materialID, v0, v1, v2; };

// add mesh to scene
unsigned int geomID = rtcNewTriangleMesh
    (scene, RTC_STATIC_GEOMETRY,
     numTriangles, numVertices, 1);

// set data buffers
rtcSetBuffer(scene, geomID, RTC_VERTEX_BUFFER,
             vertexPtr, 0, sizeof(Vertex));
rtcSetBuffer(scene, geomID, RTC_INDEX_BUFFER,
             indexPtr, 4, sizeof(Triangle));

// add more geometries
...

// commit changes
rtcCommit(scene);
```

# Intel® SPMD Program Compiler (ISPC)

- C-based language plus vector extensions
- Simplifies writing vectorized renderer
- Scalar looking code that gets vectorized automatically
- Guaranteed vectorization
- Compilation to different vector ISAs (SSE, AVX, AVX2, AVX-512)
- Available as Open Source from <http://ispc.github.com>

# Example: Rendering using ISPC

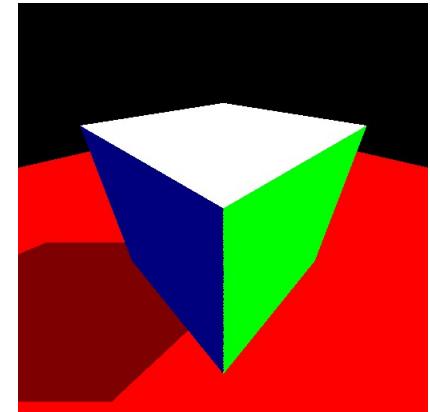
```
// loop over all screen pixels
foreach (y=0 ... screenHeight-1, x=0 ... screenWidth-1) {

    // create and trace primary ray
    RTCRay ray = make_Ray(p, normalize(x*vx + y*vy + vz), eps, inf);
    rtcIntersect(scene, ray);

    // environment shading
    if (ray.geomID == RTC_INVALID_GEOMETRY_ID) {
        pixels[y*screenWidth+x] = make_Vec3f(0.0f); continue;
    }

    // calculate hard shadows
    RTCRay shadow = make_Ray(ray.org+ray.tfar*ray.dir, neg(lightDir), eps, inf);
    rtcOccluded(scene, shadow);

    if (shadow.geomID == RTC_INVALID_GEOMETRY_ID)
        pixels[y*width+x] = colors[ray.primID]*(0.5f + clamp(-dot(lightDir, normalize(ray.Ng)), 0.0f, 1.0f));
    else
        pixels[y*width+x] = colors[ray.primID]*0.5f;
}
```



Embree Overview

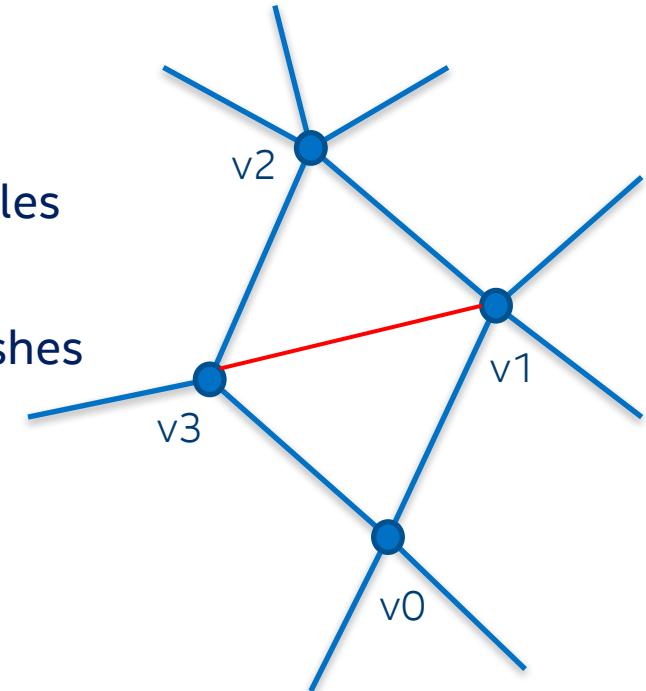
Embree API

# Advanced Features

Embree Performance

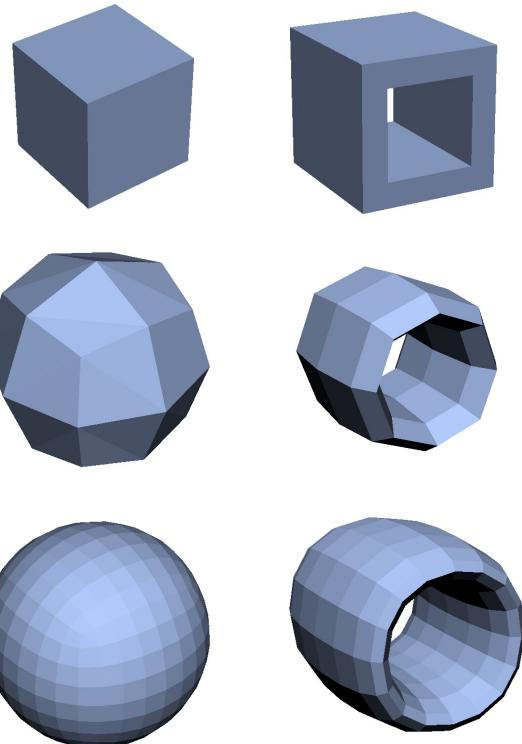
# Quad Meshes

- Quad rendered as pairs of triangles ( $v0, v1, v3$  and  $v2, v3, v1$ )
- Mixed Triangle/Quad mesh supported as triangles can also get encoded using quads ( $v0, v1, v3, v3$ )
- Most 3D modeling packages produce quad meshes
- Lower memory consumption
- Faster BVH building
- Ray Tracing slightly slower than for triangles



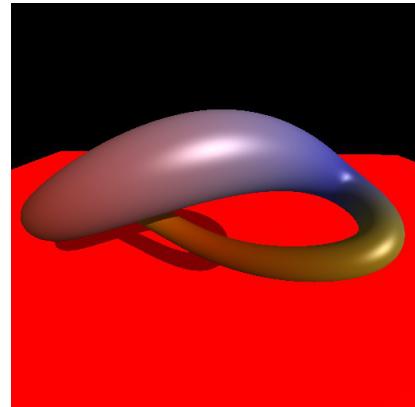
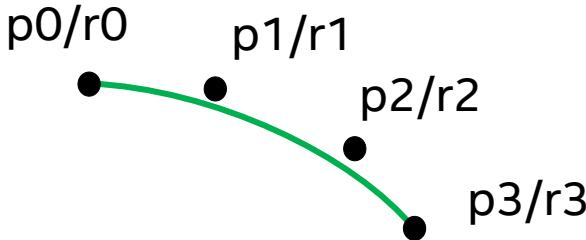
# Catmull Clark Subdivision Surfaces

- Converts coarse mesh into smooth surface by subdivision (C<sub>2</sub> continuous almost everywhere)
- Support for arbitrary topology (generalization of B-spline surface, no trimming required as with NURBS)
- Established as standard in movie production
- Embree implementation compatible with OpenSubdiv 3.0 (creases, boundary modes, etc.)
- Vector displacement mapping supported



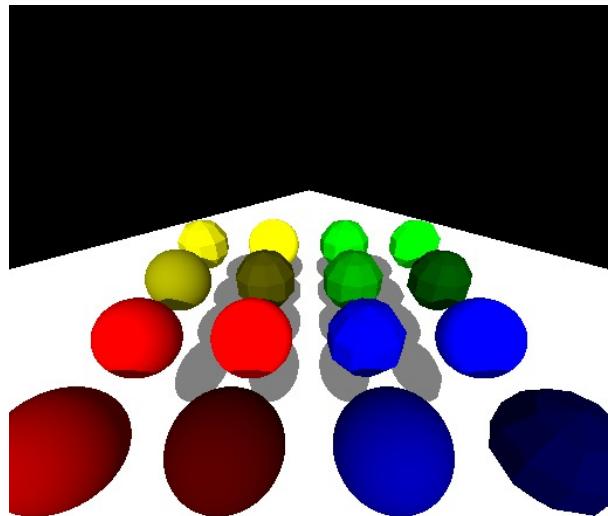
# Cubic Spline Curves

- Cubic polynomial curves
  - Bézier basis, B-spline basis, and line segments
  - Varying radius along the curve
- Two accuracies (close vs. distant curves):
  - Sweep surface of a circle along curve
  - Ray oriented ribbon primitive
- High performance through use of oriented bounding boxes [Woop et al. 2014]
- Low memory consumption through direct ray/curve intersection



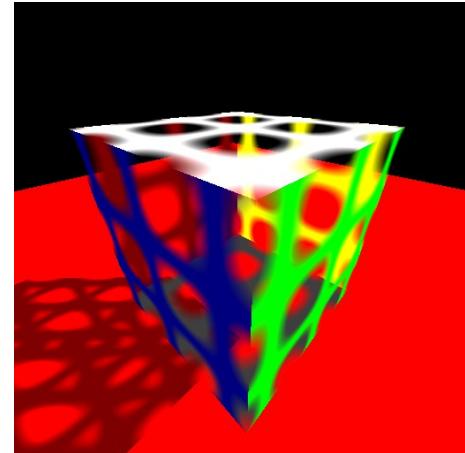
# User Defined Geometries

- Enables implementing custom primitives and features not provided by Embree
  - e.g., sphere, disk, multi level instancing, rotation motion blur, etc.
- User provides:
  - Bounding function
  - Intersect and Occluded functions



# Intersection Filter Functions

- Per geometry callback that is called during traversal for each primitive intersection
- Callback can **accept** or **reject** hit
- Can be used for:
  - Trimming curves (e.g. modeling tree leaves)
  - Transparent shadows (reject and accumulate)
  - Find all hits (reject and collect)
  - Advanced self intersection avoidance



# Multi Segment Motion Blur

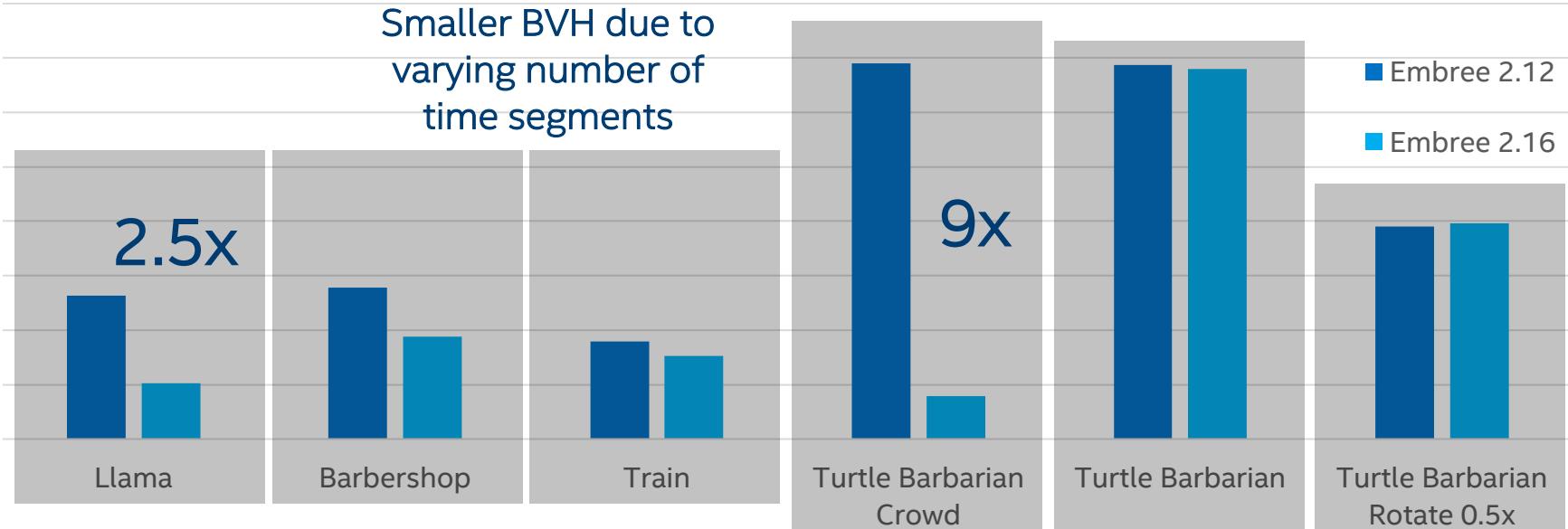
- Important to render fast curved motion (e.g. rotating wheel, fight scenes, spinning dancer, etc.)
- Sequence of time steps to be linearly interpolated provided to renderer.
- Typically equidistant time steps and often different number of time steps per geometry.



# Multi Segment Motion Blur Implementation

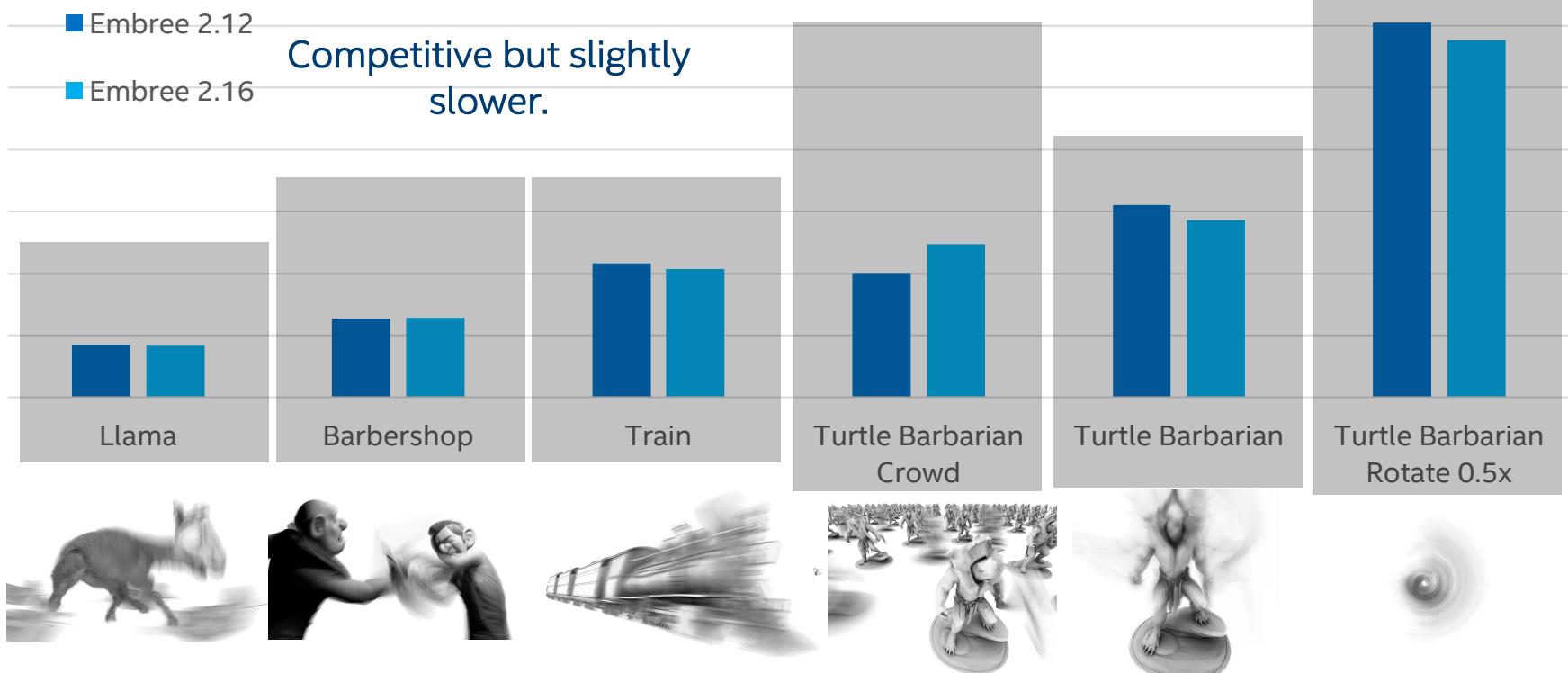
- 4D-BVH which stores linear spatial and temporal bounds
  - BVH can spatially separate geometries
  - BVH can reduce time ranges where required
- High temporal resolution for parts of the scene supported efficiently
- Longer animations efficiently supported, e.g. to render multiple frames using single geometry setup
- Large memory savings compared to Embree v2.12 implementation

# Memory Consumption



# Render Performance

Faster due to  
less memory traffic



# Multi Segment Motion Blur Implementation

“STBVH: A Spatial-Temporal BVH for Efficient Multi-Segment Motion Blur” Sven Woop, Attila T. Afra, Carsten Benthin, High Performance Graphics 2017

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

# Embree Dynamic Scene Support

- Two level BVH for optimal build performance
  - only changed geometries have to get updated
- Traditional two level build causes suboptimal render performance
  - multiple geometries traversed at overlapping region
  - wrong traversal order at overlapping region

# Embree Improved Top Level Build

- Top level BVH built using novel approach
  - Exploit available BVH of geometries
  - Open large BVH nodes of geometries during build
  - Disable opening when single object isolated
- Slightly more expensive BVH build
- Up to 2x improvement of render performance of dynamic BVH

# Embree Improved Top Level Build

“Improved Two-Level BVHs using Partial Re-Braiding”, Carsten Benthin, Sven Woop, Ingo Wald, Attila T. Afra, High Performance Graphics 2017

“Embree Ray Tracing” demo at Intel booth #807

Embree Overview  
Embree API  
Advanced Features

# Embree Performance

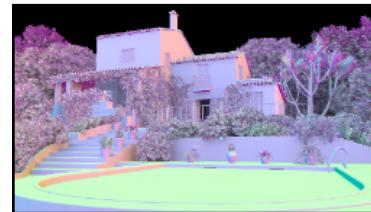
# Diffuse Path Tracing Performance

- Simple illumination effect to measure pure ray tracing performance
- Highest quality BVH build for all platforms
- Embree v2.16.0 performance measured on:
  - Dual socket Intel® Xeon® Platinum 8180 Processor (2x28 cores @ 2.5 GHz, AVX-512)
  - Intel® Xeon Phi™ 7250 Processor (68 cores @ 1.4 GHz, AVX-512)
- Comparing against state of the art GPU methods using:
  - OptiX™ Prime 4.0.2 and CUDA® 8.0.44
  - NVIDIA Tesla P100 Coprocessor (3584 CUDA cores @ 1.175 GHz, Pascal)

# 3D Models used for Benchmarking



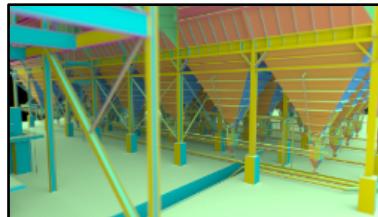
Mazda  
5.7M triangles



Villa  
37.7M triangles



Art Deco  
10.7M triangles

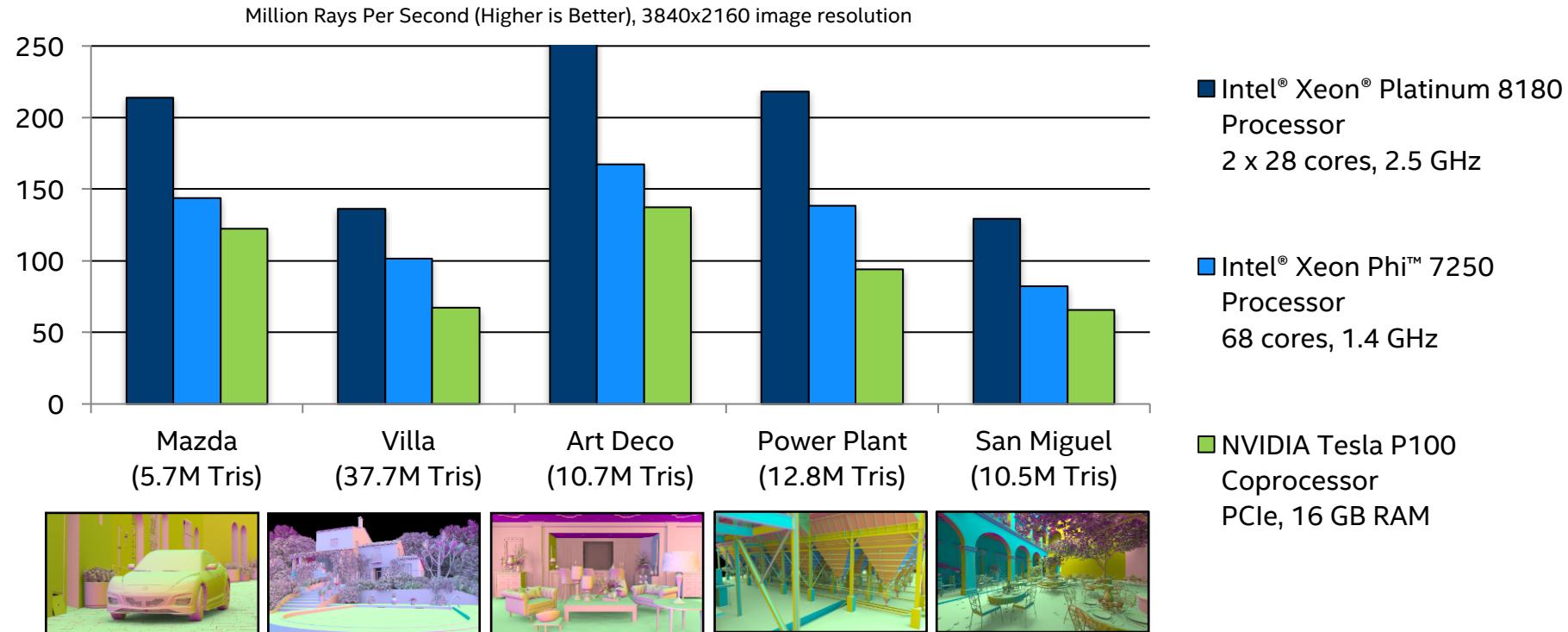


Power Plant  
12.8M triangles



San Miguel  
10.5M triangles

# Diffuse Path Tracing Performance

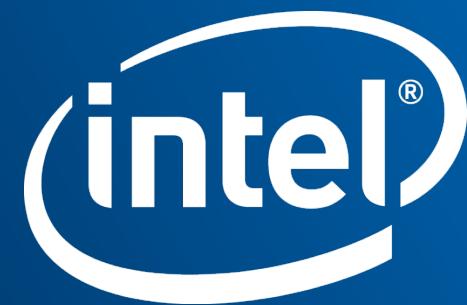


Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark® and MobileMark®, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more information go to <http://intel.com>

# Questions?

<https://embree.github.io>  
[embree@googlegroups.com](mailto:embree@googlegroups.com)

Visit the Intel booth #807 for a live Embree demo!



# Legal

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice.

All products, dates, and figures specified are preliminary based on current expectations, and are subject to change without notice.

Intel processors, chipsets, and desktop boards may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Optimized Intel® HD Graphics P3000 only available on select models of the Intel® Xeon® processor E3 family. To learn more about Intel Xeon processors for workstation visit [www.intel.com/go/workstation](http://www.intel.com/go/workstation). HD Graphics P4000 introduces four additional execution units, going from 8 in the HD P3000 to 12 in the HD P4000. Optimized Intel® HD Graphics P4000 only available on select models of the Intel® Xeon® processor E3-1200 v2 product family. For more information, visit <http://www.intel.com/content/www/us/en/architecture-and-technology/hdgraphics/hdgraphics-developer.html>

Iris™ graphics is available on select systems. Consult your system manufacturer.

Any code names featured are used internally within Intel to identify products that are in development and not yet publicly announced for release. Customers, licensees and other third parties are not authorized by Intel to use code names in advertising, promotion or marketing of any product or services and any such use of Intel's internal code names is at the sole risk of the user.

Intel product plans in this presentation do not constitute Intel plan of record product roadmaps. Please contact your Intel representative to obtain Intel's current plan of record product roadmaps.

Performance claims: Software and workloads used in performance tests may have been optimized for performance only on Intel® microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more information go to : <http://www.intel.com/performance>

# Legal Disclaimer and Optimization Notice

INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS". NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Copyright © , Intel Corporation. All rights reserved. Intel, the Intel logo, Xeon, Core, VTune, and Cilk are trademarks of Intel Corporation in the U.S. and other countries.

## Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804

