Accelerated Ray Tracing of Constructive Solid Geometry

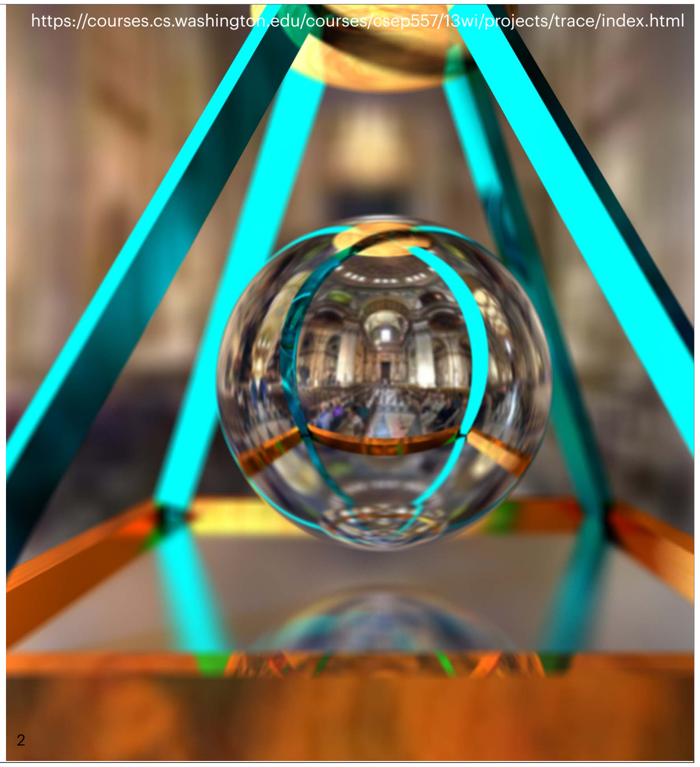
BSc. Thesis in Computer Science

Presenter: Otmane Sabir

Supervisor: Prof. Dr. Sergey Kosov

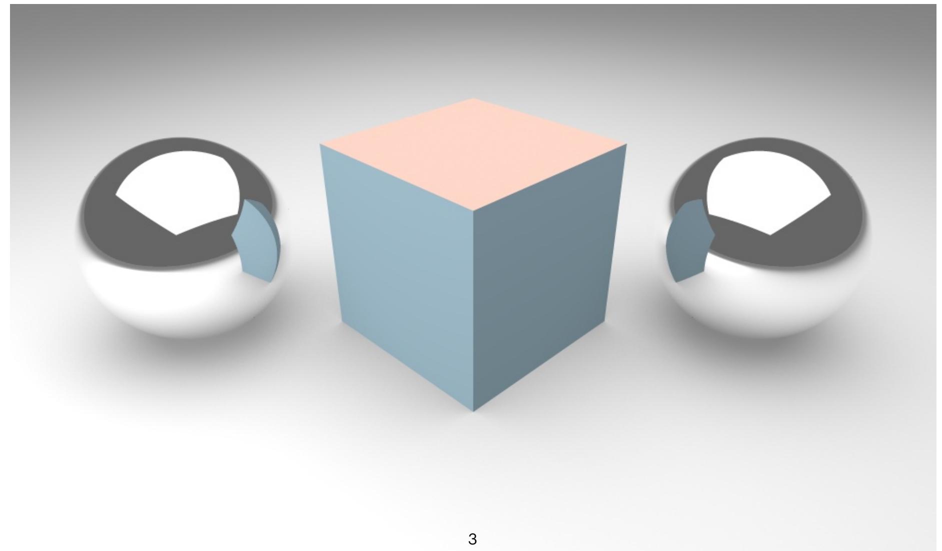
Objective

- Modeling complex solids using implicit and explicit geometries.
- Fast evaluation of geometries to enable faster ray tracing.
- Robust, reliable, and expandable approach.



https://courses.cs.washington.edu/courses/csep557/13wi/projects/trace/index.html

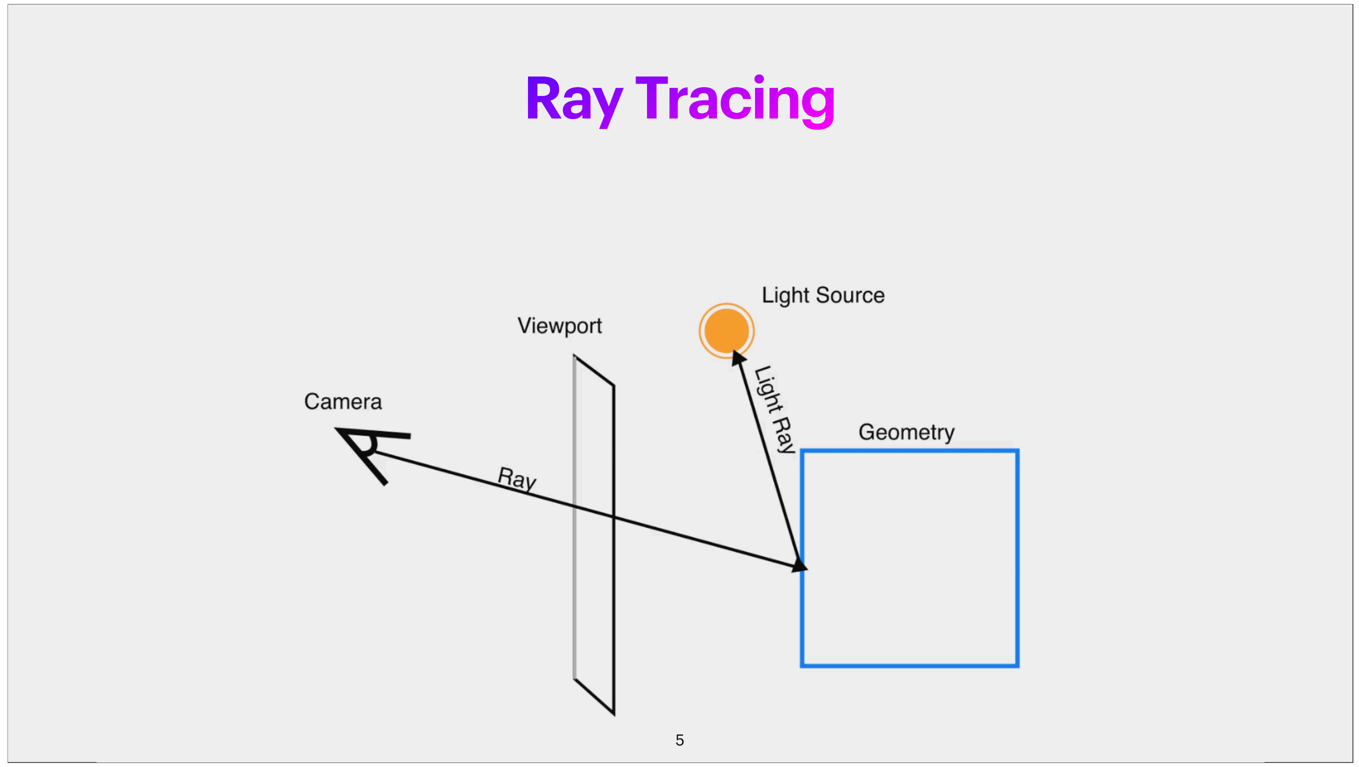
openRT



OpenRT is used to perform our investigations. OpenRT is a C++ free open source ray tracing library.
OpenRT has all the functionality we need to describe geometry, shaders, lights, cameras, and samplers.

Agenda

1. Ray Tracing
2. Constructive Solid Geometry
3. Minimal Hit Classification
4. Binary Space Partitioning
5. Accelerated CSG Ray Tracing
6. Results
7. Limitations



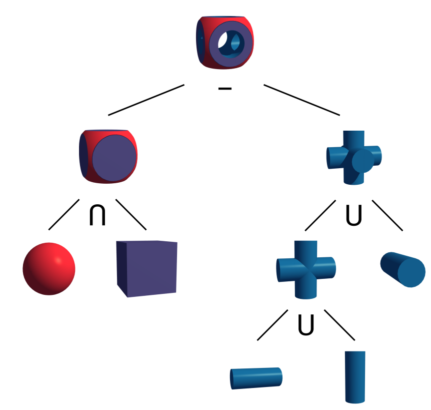
Ray tracing is essentially reversing the photographic process.
Instead of receiving rays of light to our lens or eyes, we shoot a ray towards some object and evaluate whether or not it intersects our scene.
4 primary components (Cameras, Shaders, Lights, Geometry)

Ray Tracing Advantages

- Ray tracing parallelizes efficiently and trivially.
- Modularity of components (Cameras, Shaders, Light sources).
- Highly photorealistic results when coupled with other techniques.

1) It takes advantage of the ever-increasing computational power of the hardware.
2) Various results depending on the needs and the case scenario.
3) Photon Mapping, Path Tracing, Models for accurate estimation of the rendering equation.

Constructive Solid Geometry



Constructive solid geometry takes basis on the fundamental premise that any complex physical object is obtainable from a set of primitive geometries and the base boolean operations. In the general constructive solid geometry description, the solids are put in a binary tree, referred to as the CSG tree. The root node is the complete composite geometry. The leaf nodes depict the base geometries (cubes, spheres, cylinders, tori, cones, and polygon meshes) used in the composition. Every node in the tree, besides the leaf nodes, expresses another complete solid and contains information of the set operation of that node.

CSG is radically different from other geometrical description as it does not collect any topological information but instead evaluates the geometries as needed by the case scenario.

Advantages:
* there is no explicit description of the boundary of the solid.
* a high-level specification of the objects in space
* permits significantly more straightforward modification and manipulation

Constructive Solid Geometry

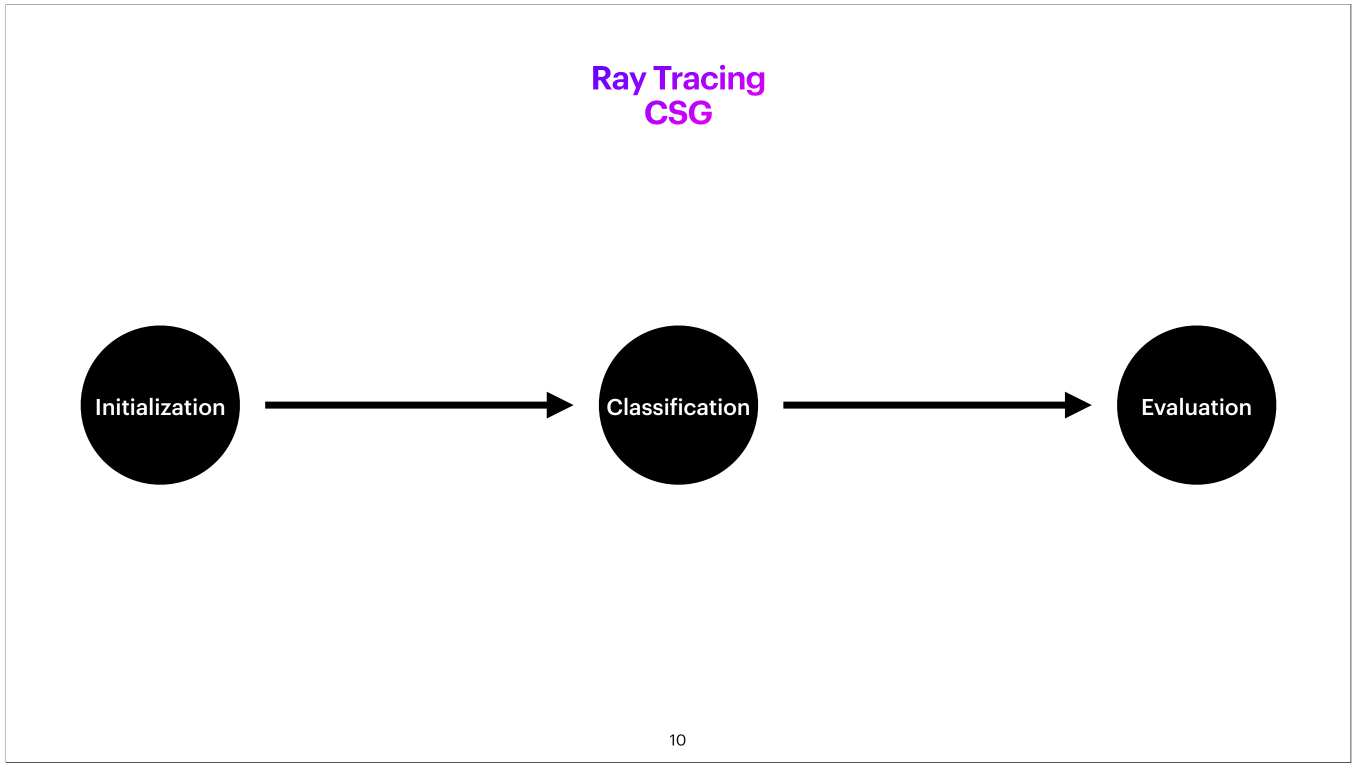
$$M[X,S] = \begin{cases} eM(X,S), & \text{if } S \subset A \\ g(M[X,\text{l-subtree}(S)], M[X,\text{r-subtree}(S)], \text{root}(S)), & \text{otherwise} \end{cases}$$

S = The regular reference set.
X = The candidate regular set.
eM = The primitive evaluation function.
A = The set of all allowed primitives.
g = The combine function.
l-subtree = The left subtree.
r-subtree = The right subtree.
root = The operation type.

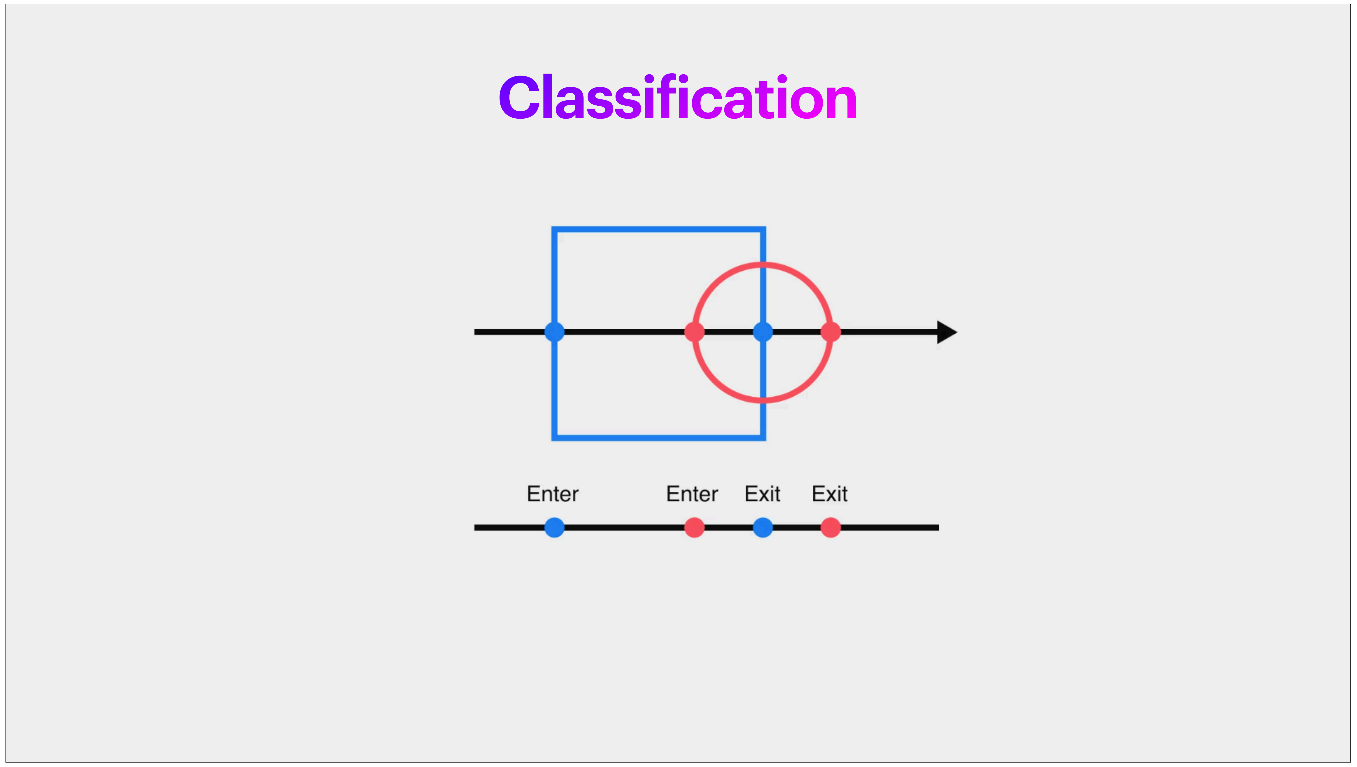
Constructive solid geometry takes basis on the fundamental premise that any complex physical object is obtainable from a set of primitive geometries and the base boolean operations. In the general constructive solid geometry description, the solids are put in a binary tree, referred to as the CSG tree. The root node is the complete composite geometry. The leaf nodes depict the base geometries (cubes, spheres, cylinders, tori, cones, and polygon meshes) used in the composition. Every node in the tree, besides the leaf nodes, expresses another complete solid and contains information of the set operation of that node.

CSG is radically different from other geometrical description as it does not collect any topological information but instead evaluates the geometries as needed by the case scenario.

Advantages:
* there is no explicit description of the boundary of the solid.
* a high-level specification of the objects in space
* permits significantly more straightforward modification and manipulation

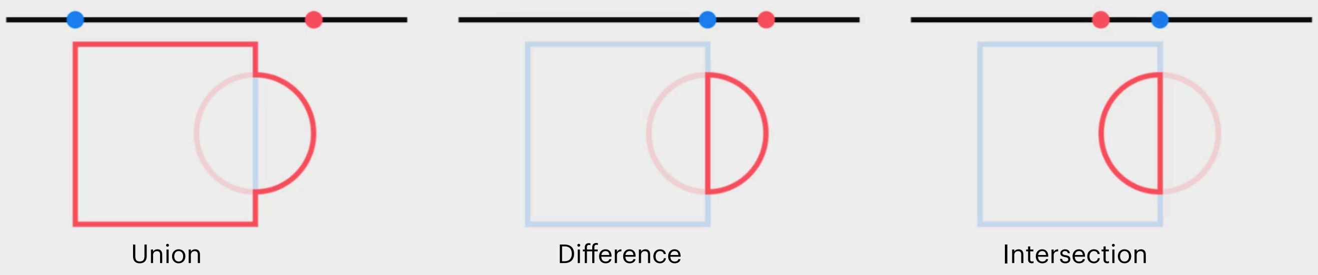


Initialization is taken care of by the rendering engine.
Classification -> Evaluation are based on the divide-and-conquer paradigm.



* Gather all intersections
* Sorting them by distance to ray origin
* Classification of intersection points to either entering or exiting.

Evaluation

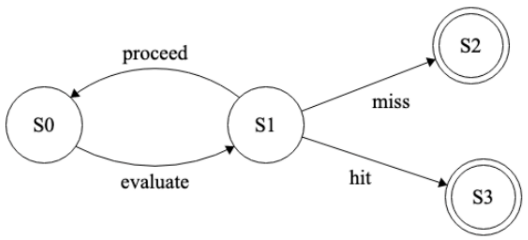


Performance Downfalls

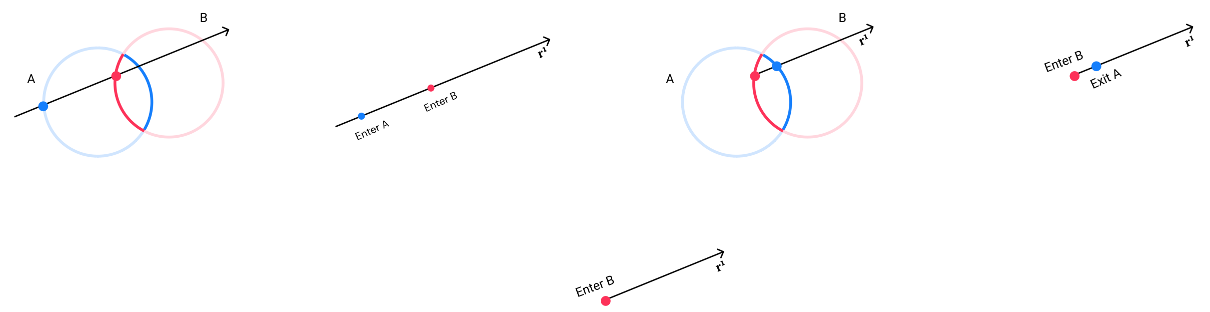
- Memory usage when classifying and evaluating complex geometries.
- Adds time complexity of sorting the intersections by distance and a minimum of 2 scans per array.
- More complex when evaluation BREPs instead of implicitly defined primitives.

Minimal Hit Classification

- Only look for intersections as needed.
- Evaluate a pair of classifications at a time.



Example - Intersection

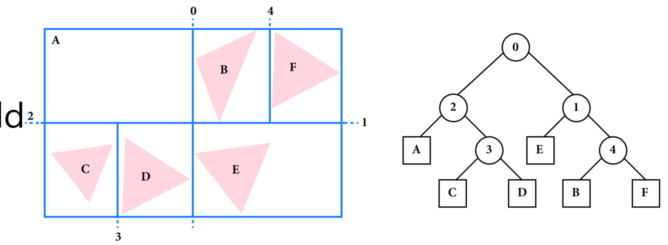


Advantages of MHC

- Only checks for needed intersections.
- Less memory load on each step.
- Faster evaluation than classical algorithm when used on implicit geometry.
- No sorting or array scans needed.
- Compatibility with spatial indexing structures.

Binary Space Partitioning

- Hierarchical data structures using binary space subdivision.
- Relies on two major components: build and traversal logic.
- Quick evaluation of ray-intersection tests using bounds around geometries.



1) One of the most fundamental concepts in ray tracing is spatial or hierarchical data structures built using binary space subdivision to efficiently search for objects in the scene.
2) A predominant concept in these data structures is binary space partitioning which refers to the successive subdivision of a scene's bounding box with planes until we reach termination criteria.
3) The resulting data structure is called a binary space partition tree or a BSP tree. BSP trees offer the flexibility of using arbitrarily oriented planes to accommodate complex scenes and uneven spatial distributions.

Accelerated CSG Ray Tracing

- Wrap the entire scene in a BSP tree.
- Each node in the composite solid is described using the hierarchal data structure.
- Efficient evaluation using the minimal hit algorithm.

1) The accelerated algorithm leverages the building blocks that we have established before.
2) -> Results

Results

3 Variants

I	II	III
(NaiCSG)	(BinCSG)	(OptimCSG)
No Acceleration	BSP Acceleration on the scene level.	BSP on the scene level. BSP on the composite level.

3 Tests

I

Geometric Complexity

Assess how the rendering time develops to the complexity of the geometry.

II

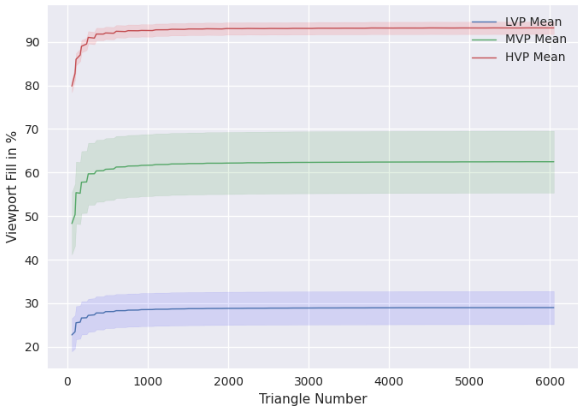
Ray-Geometry Test Counts

Count the number of intersection tests performed by each variant at each pixel

III

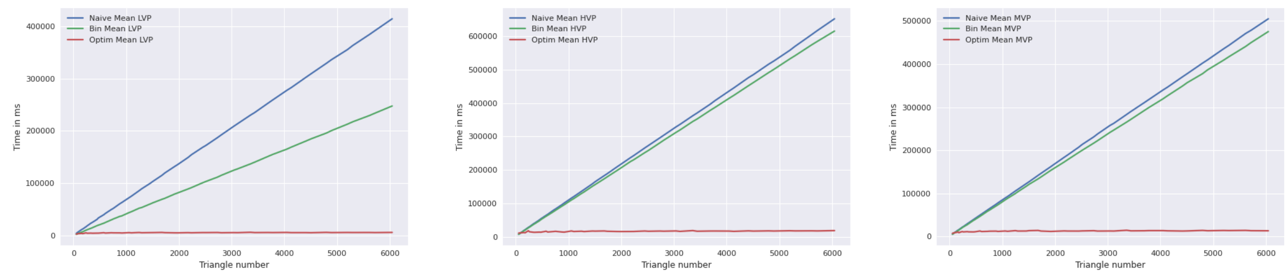
Nesting Test

Different number of nested geometries on the various algorithms while maintaining a relevantly similar viewport fill rate.



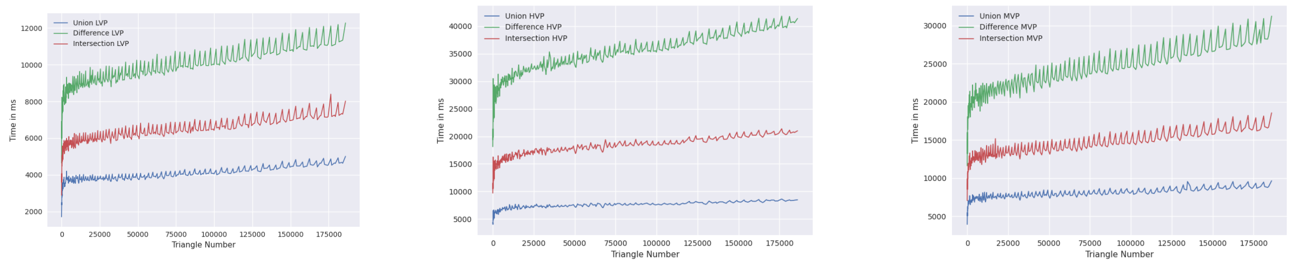
Range of view port rates on which the tests are conducted as the complexity of the geometry increases.
The area around the curve signifies the error by which the rate fluctuates.

Geometric Complexity



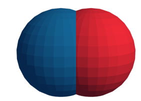
Rendering time with respect to gradual increases in geometry complexity in a scene filling various rates of the view port.

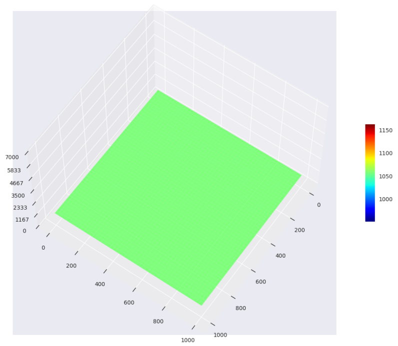
Geometric Complexity



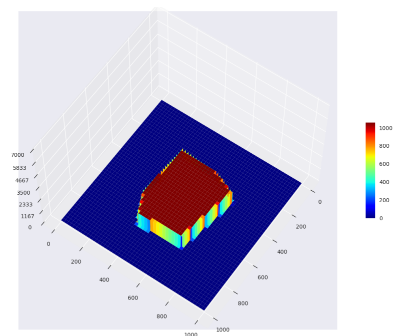
OptimCSG rendering time of different operations with respect to gradual increases in geometry complexity.

Ray-Primitive Checks

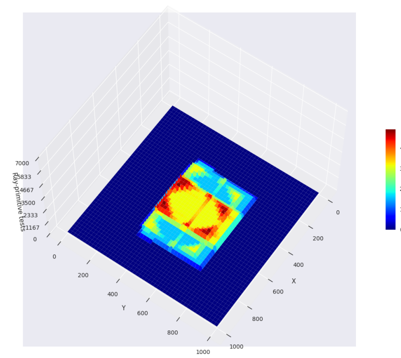




NaiCSG



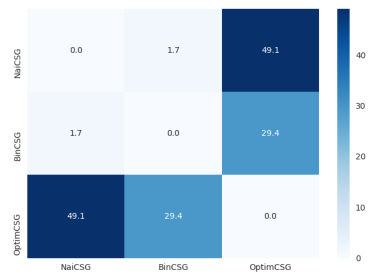
BinCSG



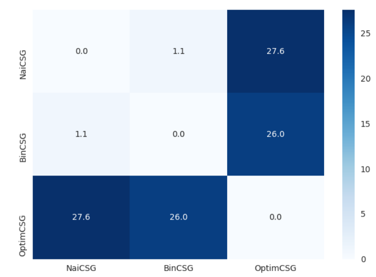
OptimCSG

Surface plot showing the number of the performed ray-primitive tests on each pixel.

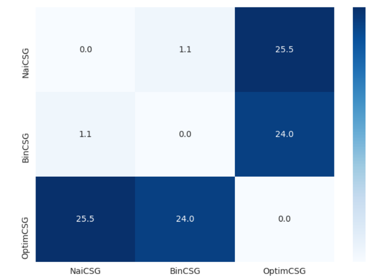
Speedup of Geo Complexity



Low viewport rate.



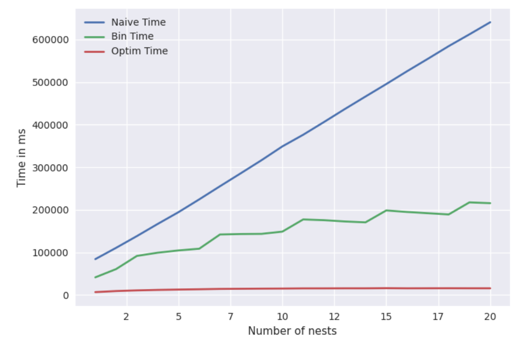
Mid viewport rate.

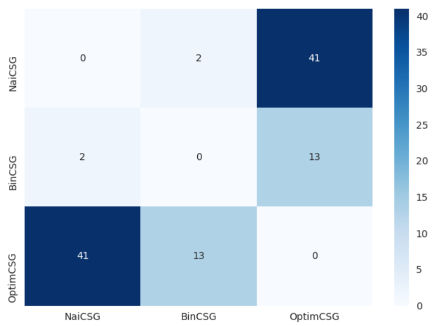


High viewport rate.

Achieved speedup of the different variants.

Composite Nesting Tests





Limitations

- Artifacts created by the numeric stability issues in classification.
- Limitation of consistently oriented normals to classify intersections.

1) When classifying using the surface normals, we will run into the issue where the dot product of the ray direction and the normal is near zero (vectors are orthogonal). Hence, leading to certain artifacts emerging near the boundaries of the geometries of edges. The described issue can be solved using a sampler which would increase the number of rays shot in the neighborhood of a pixel and estimate a better shading result. This solution is already plausible in OpenRT.
2) While all primitives and solids constructed inside OpenRT guarantee this property, meshes imported from the outside could potentially lead to is-sues. However, one can solve this by implementing an extra scan when constructing solids or passing them to a composite to verify and modify the surface normals when needed. Algorithms that allow for fast checks of consistent normal orientation are readily available.

Questions?

[1] https://courses.cs.washington.edu/courses/csep557/13wi/projects/trace/index.html
[2] https://www.youtube.com/watch?v=b2WOjo0C-xE
[3] https://deepnote.com/@otmane-sabir/OpenRT-CSG-Results-Main-TSrcMFOhS_GGcF8N1bgH1Q#