GRTX

Efficient Ray Tracing for 3D Gaussian-Based Rendering

Junseo Lee Sangyun Jeon Jungi Lee Junyong Park Jaewoong Sim

Seoul National University

HPCA'26 | February 2026

Department of ECE
Seoul National University

3D Gaussian-Based Rendering

3D Gaussian Splatting (3DGS)



3D Gaussian-Based Rendering

Rasterization

3D Gaussian Splatting (3DGS)



3D Gaussians



Rendered Image

Ray-tracing

3D Gaussian Ray Tracing (3DGRT)

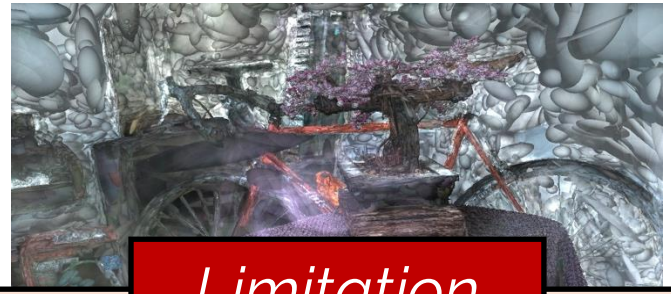
e.g., EVER (Google) [1], DSYG (Meta) [2], 3DGRT (NVIDIA) [3]

[1] Alexander et al., EVER: Exact Volumetric Ellipsoid Rendering for Real-time View Synthesis (ICCV'25)
[2] Jorge et al., Don't Splat your Gaussians: Volumetric Ray-Traced Primitives for Modeling and Rendering Scattering and Emissive Media (SIGGRAPH'25)
[3] Nicolas et al., 3D Gaussian Ray Tracing: Fast Tracing of Particle Scenes (SIGGRAPH Asia'24)

3D Gaussian-Based Rendering

Rasterization

3D Gaussian Splatting (3DGS)



Limitation

Rasterization cannot
model complex light transports



Rendered Image

Ray-tracing

3D Gaussian Ray Tracing (3DGRT)

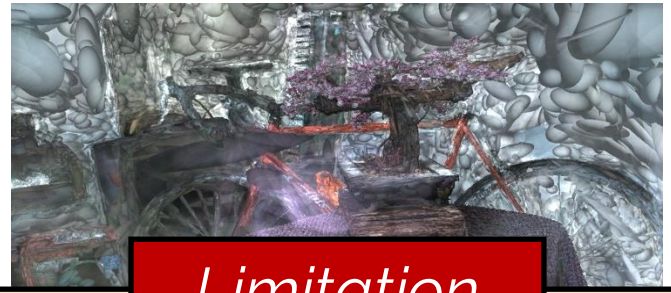
e.g., EVER (Google) [1], DSYG (Meta) [2], 3DGRT (NVIDIA) [3]

[1] Alexander et al., EVER: Exact Volumetric Ellipsoid Rendering for Real-time View Synthesis (ICCV'25)
[2] Jorge et al., Don't Splat your Gaussians: Volumetric Ray-Traced Primitives for Modeling and Rendering Scattering and Emissive Media (SIGGRAPH'25)
[3] Nicolas et al., 3D Gaussian Ray Tracing: Fast Tracing of Particle Scenes (SIGGRAPH Asia'24)

3D Gaussian-Based Rendering

Rasterization

3D Gaussian Splatting (3DGS)



Limitation

Rasterization cannot model complex light transports



Rendered Image

Ray-tracing

3D Gaussian Ray Tracing (3DGRT)

e.g., EVER (Google) [1], DSYG (Meta) [2], 3DGRT (NVIDIA) [3]



[1] Alexander et al., EVER: Exact Volumetric Ellipsoid Rendering for Real-time View Synthesis (ICCV'25)
[2] Jorge et al., Don't Splat your Gaussians: Volumetric Ray-Traced Primitives for Modeling and Rendering Scattering and Emissive Media (SIGGRAPH'25)
[3] Nicolas et al., 3D Gaussian Ray Tracing: Fast Tracing of Particle Scenes (SIGGRAPH Asia'24)

3D Gaussian-Based Rendering

Rasterization

3D Gaussian Splatting (3DGS)

Ray-tracing

3D Gaussian Ray Tracing (3DGRT)

e.g., EVER (Google) [1], DSYG (Meta) [2], 3DGRT (NVIDIA) [3]





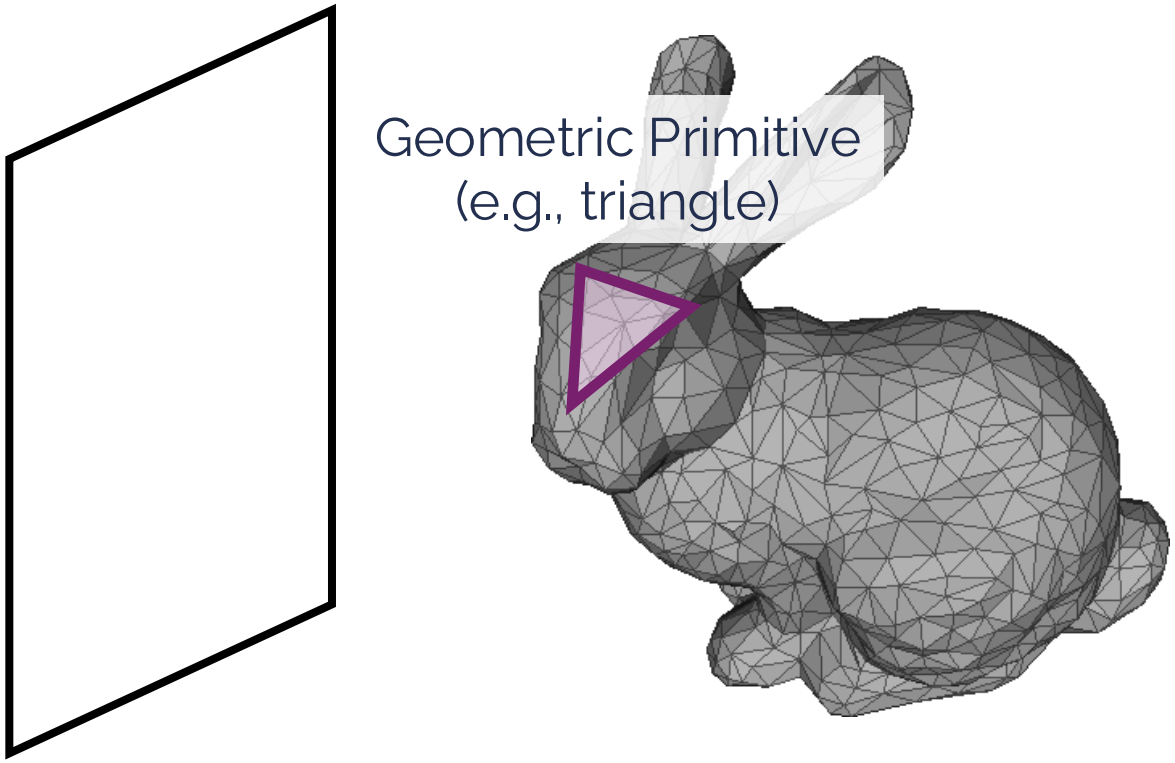
Q: What is the key difference between **rasterization** and **ray-tracing**?

Rendered Image

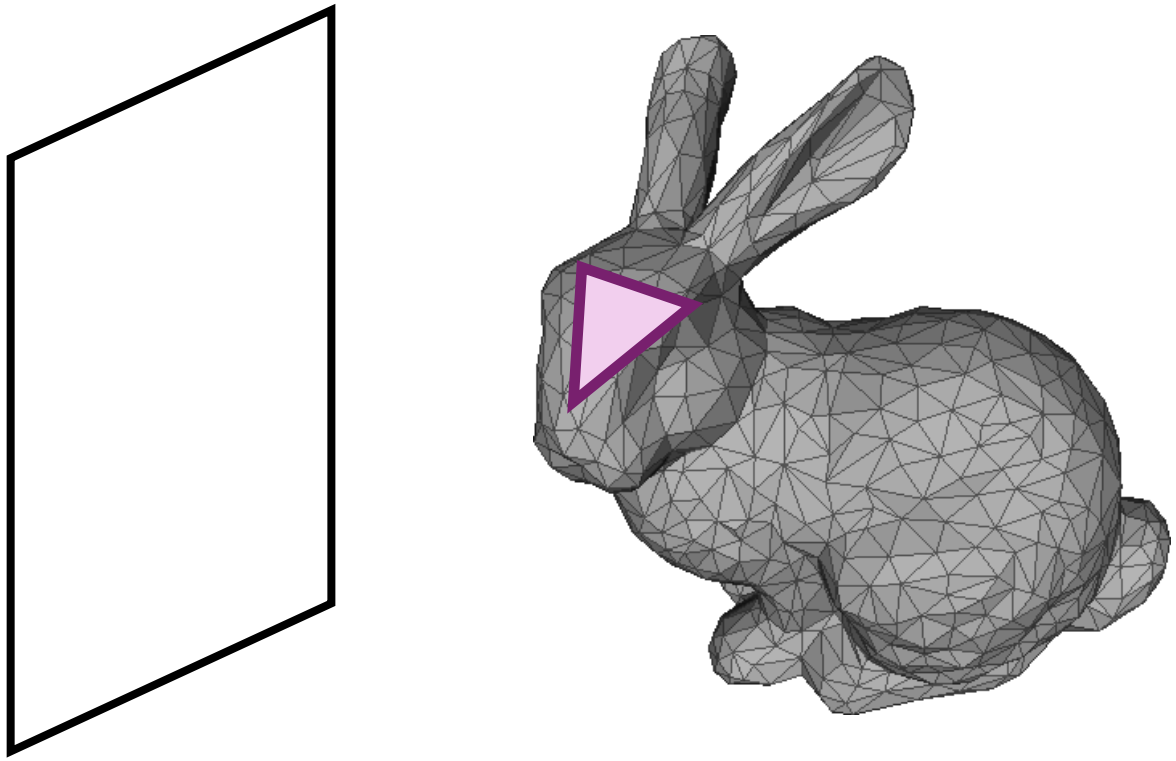
[1] Alexander et al., EVER: Exact Volumetric Ellipsoid Rendering for Real-time View Synthesis (ICCV'25)
[2] Jorge et al., Don't Splat your Gaussians: Volumetric Ray-Traced Primitives for Modeling and Rendering Scattering and Emissive Media (SIGGRAPH'25)
[3] Nicolas et al., 3D Gaussian Ray Tracing: Fast Tracing of Particle Scenes (SIGGRAPH Asia'24)

3D Gaussian-Based Rendering

Rasterization

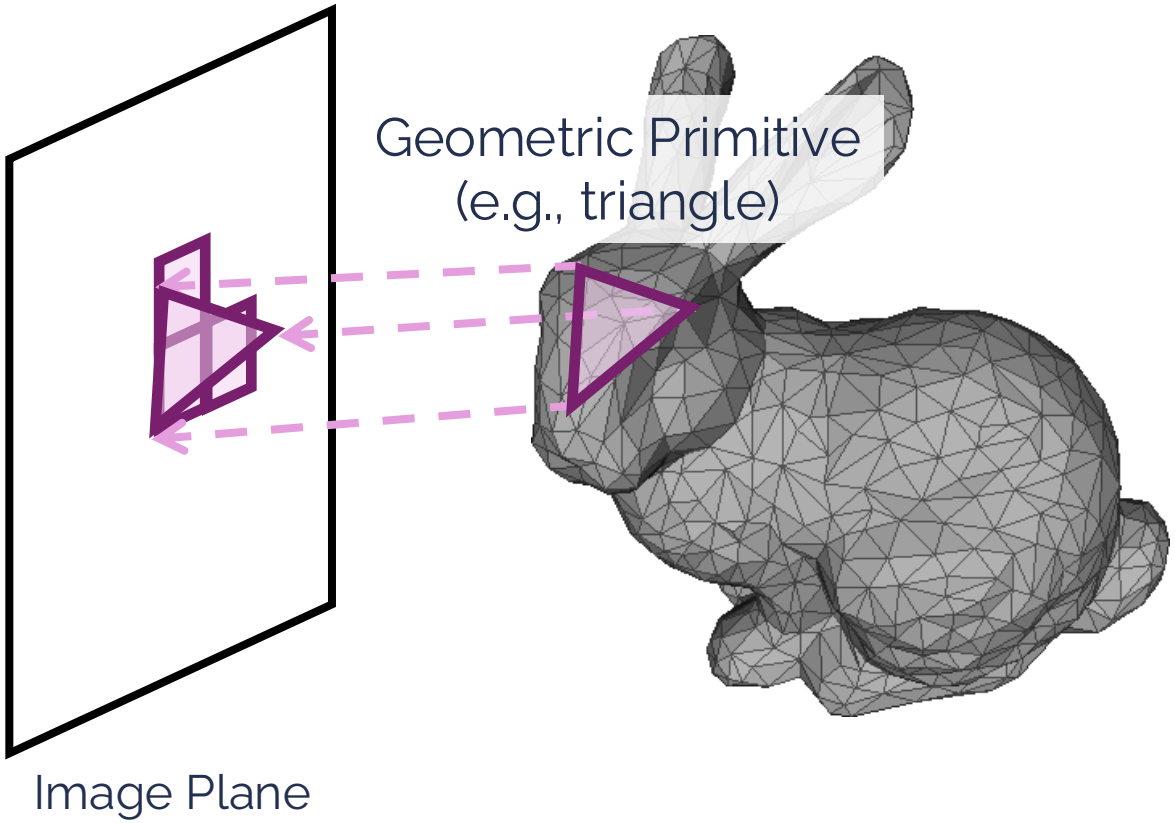


Ray-tracing

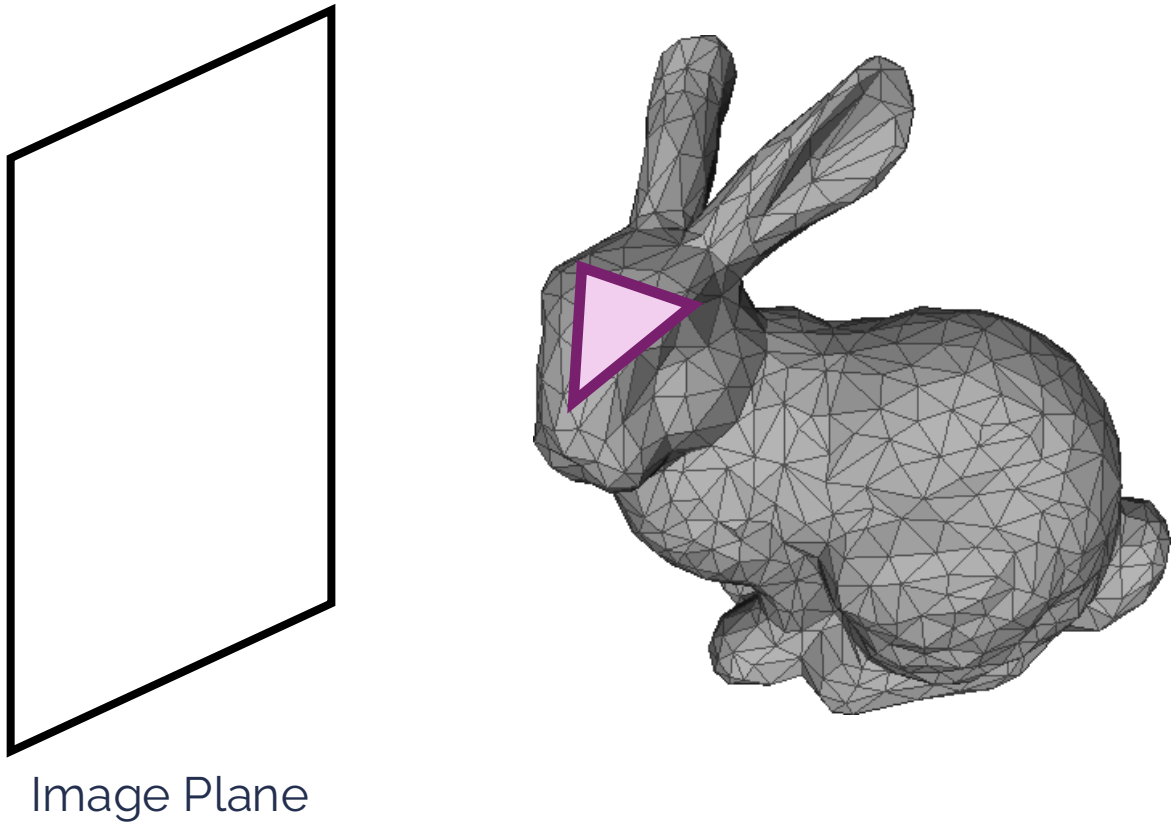


3D Gaussian-Based Rendering

Rasterization

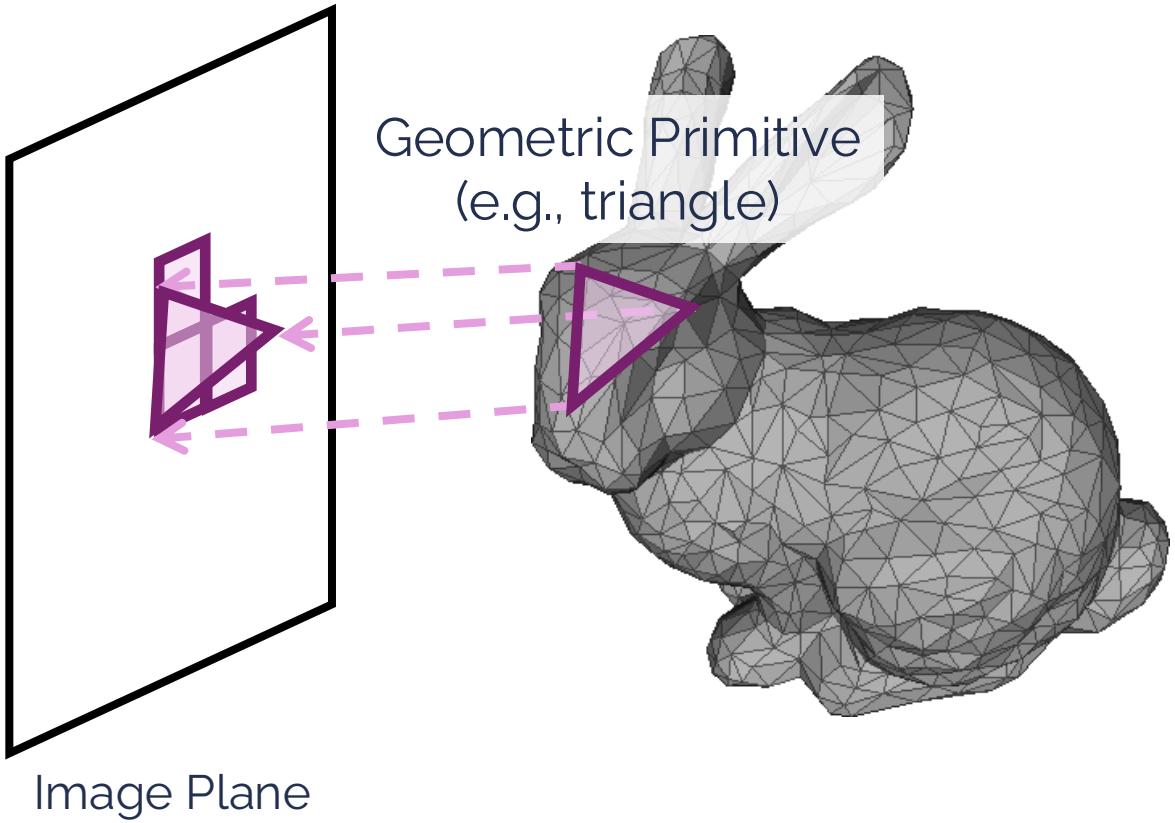


Ray-tracing

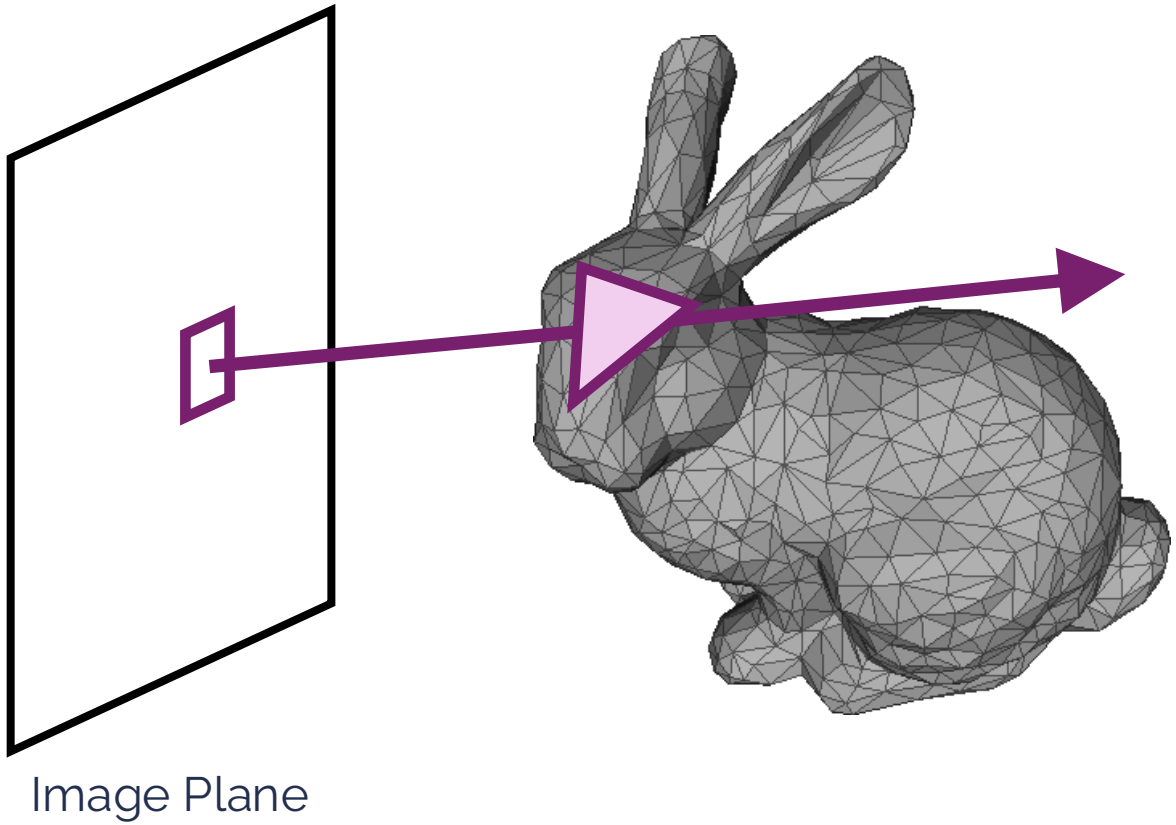


3D Gaussian-Based Rendering

Rasterization

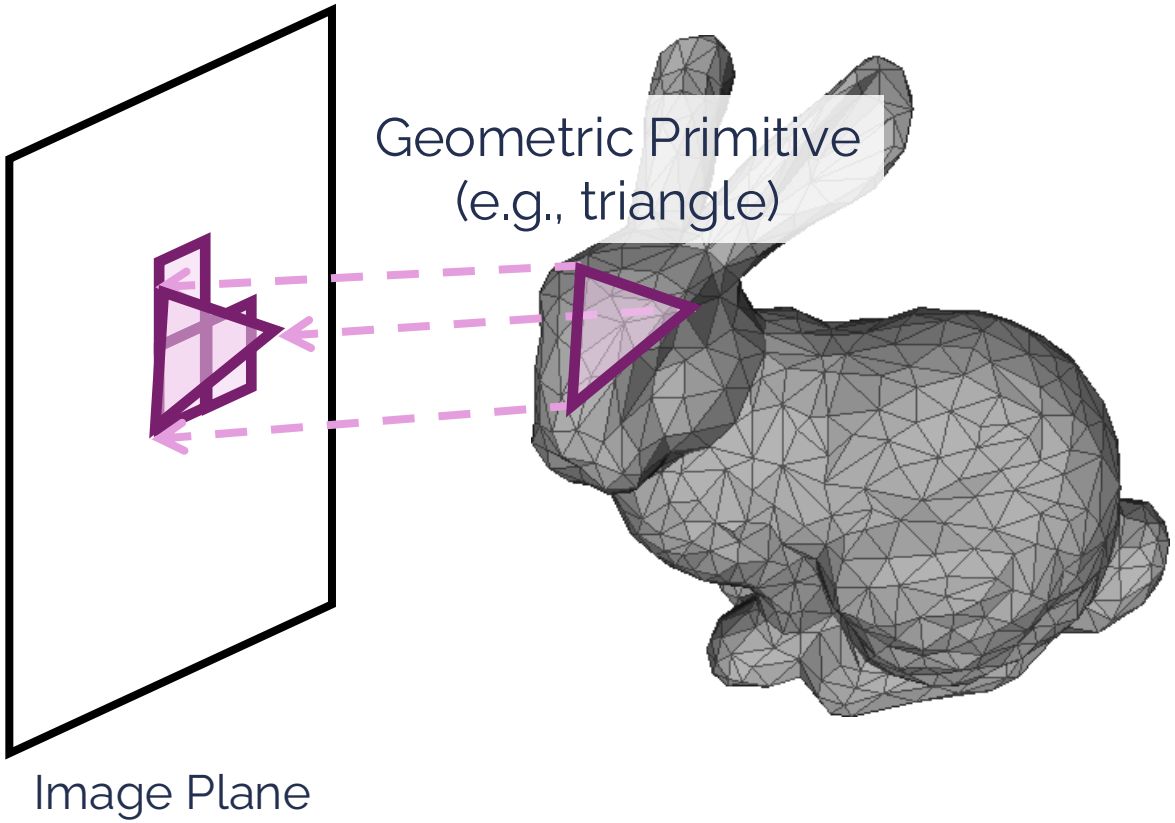


Ray-tracing

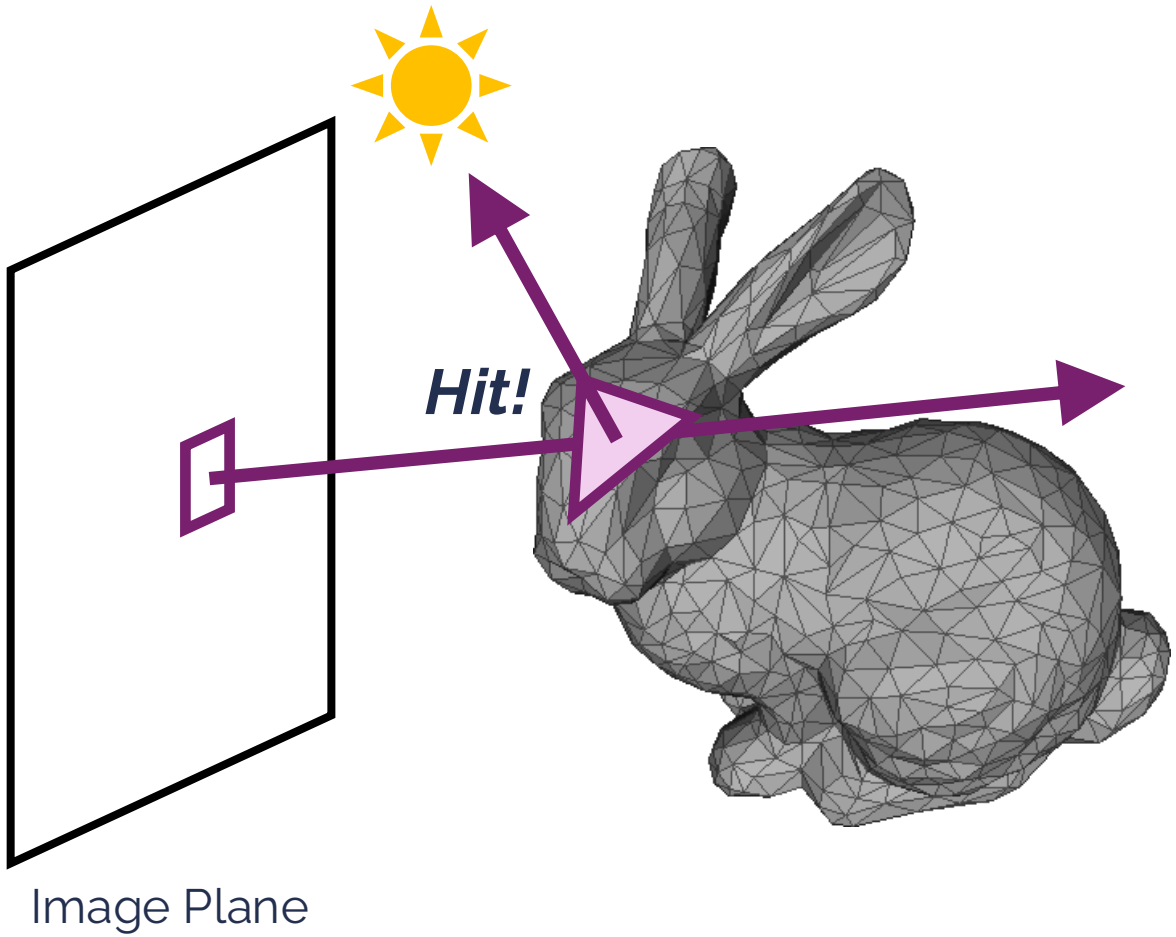


3D Gaussian-Based Rendering

Rasterization

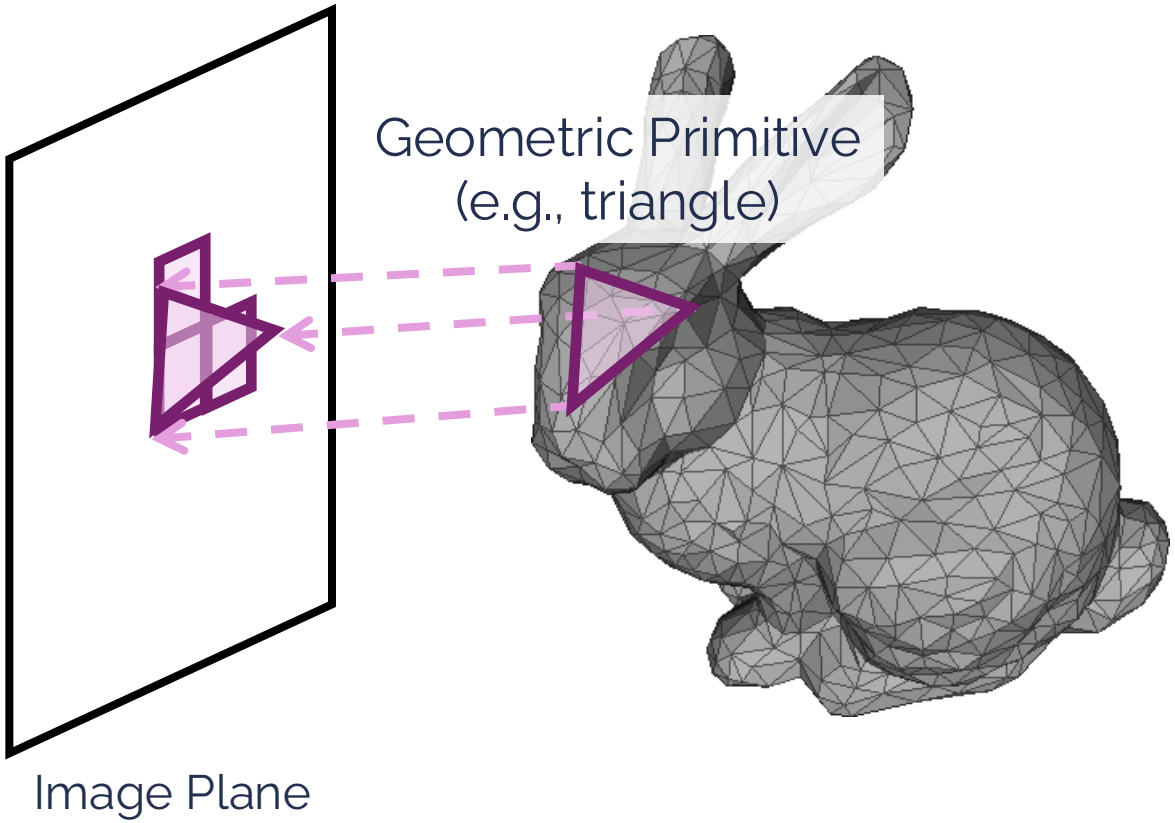


Ray-tracing

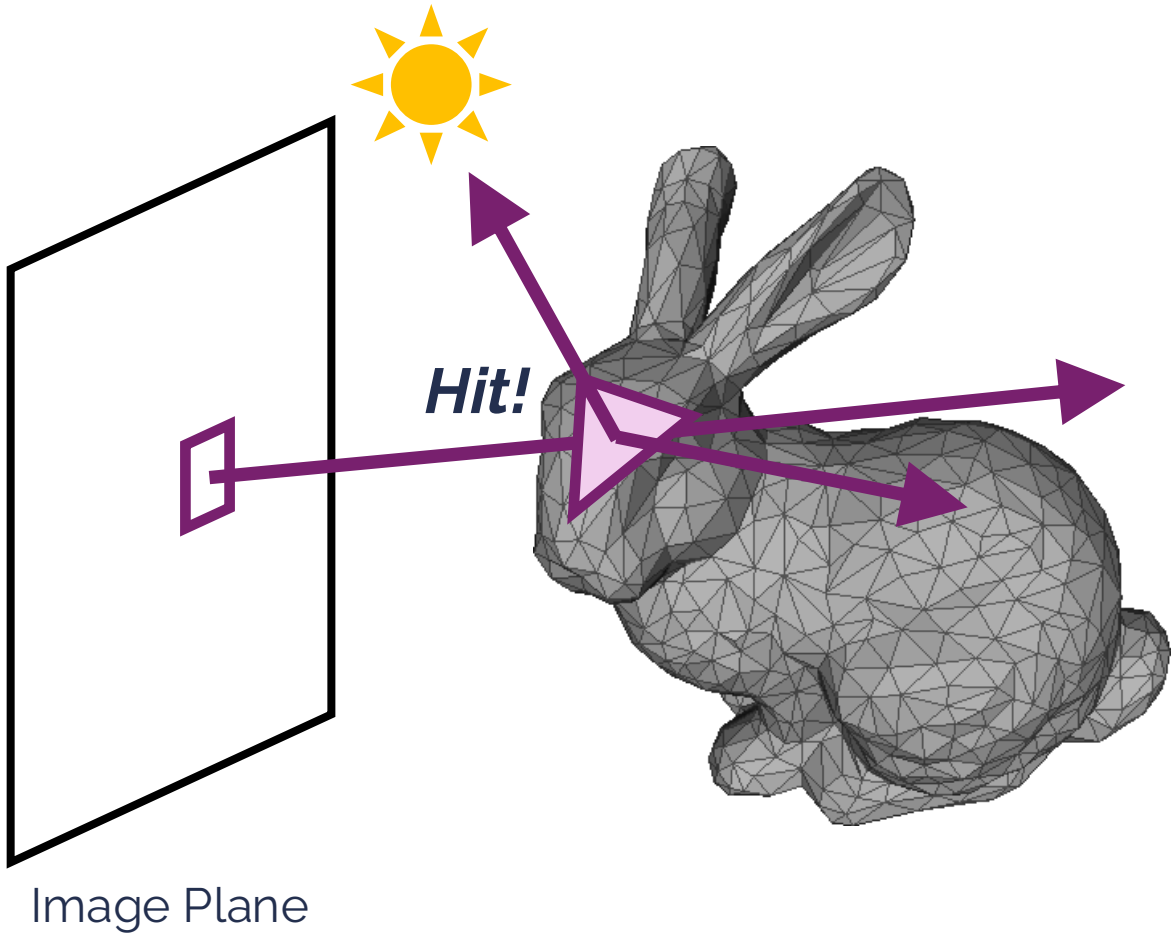


3D Gaussian-Based Rendering

Rasterization

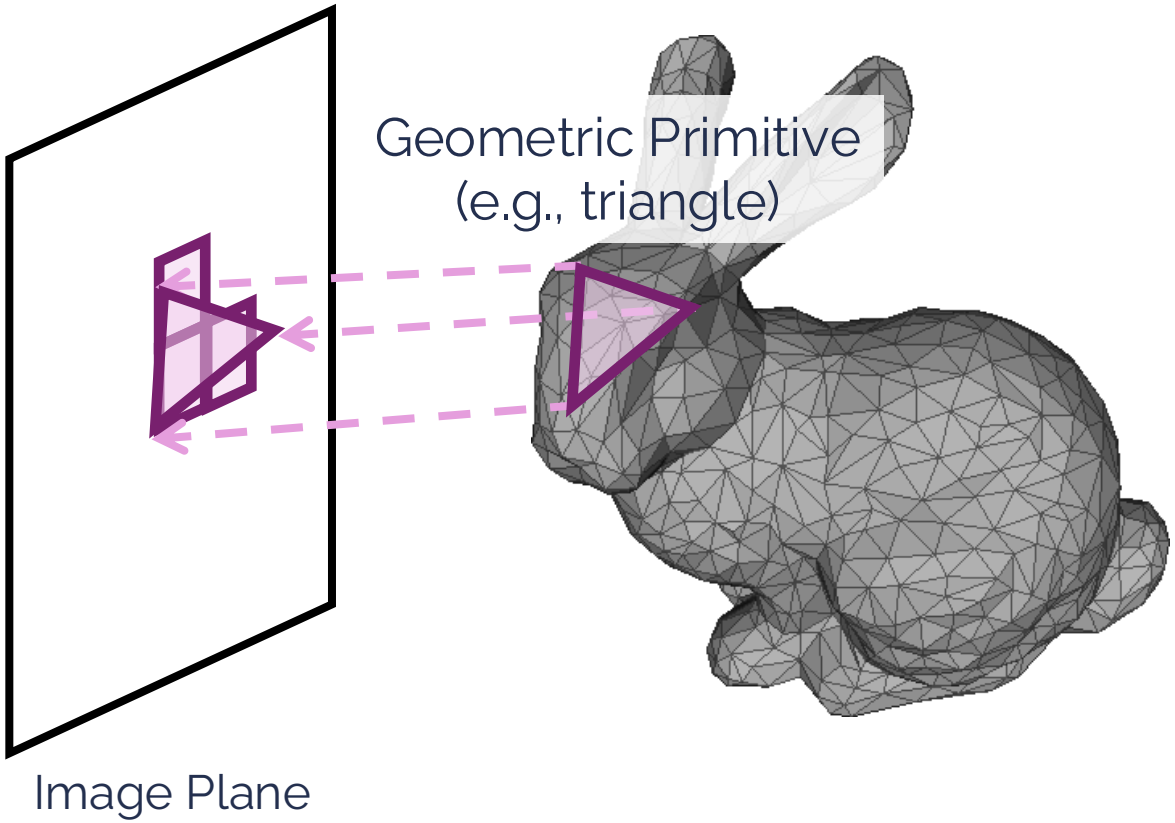


Ray-tracing



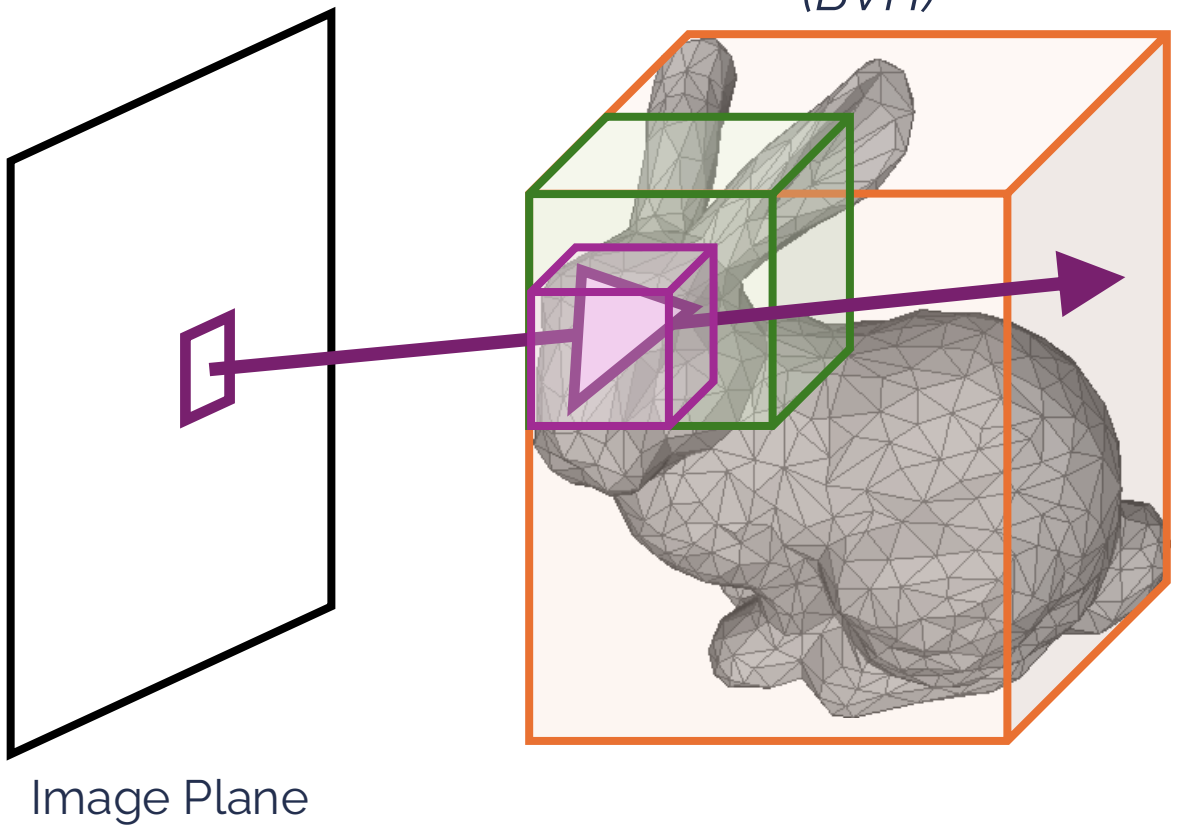
3D Gaussian-Based Rendering

Rasterization

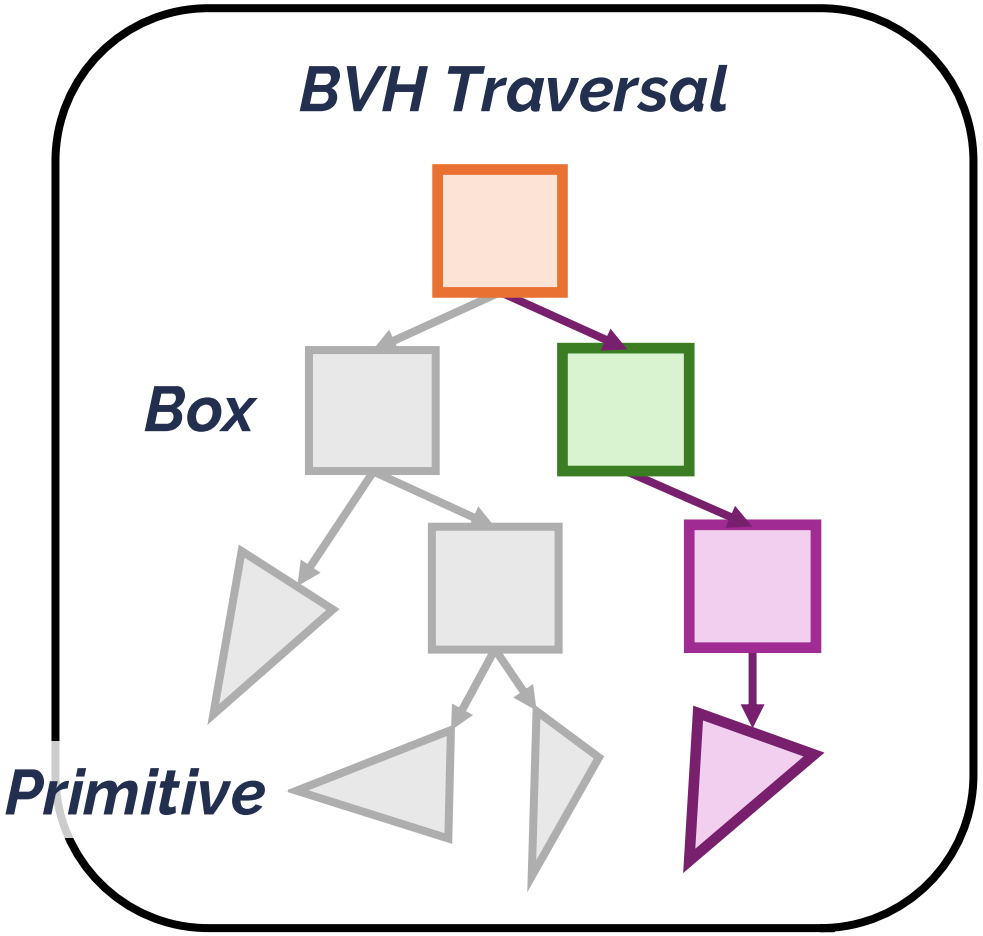


Ray-tracing

Bounding Volume Hierarchy (BVH)

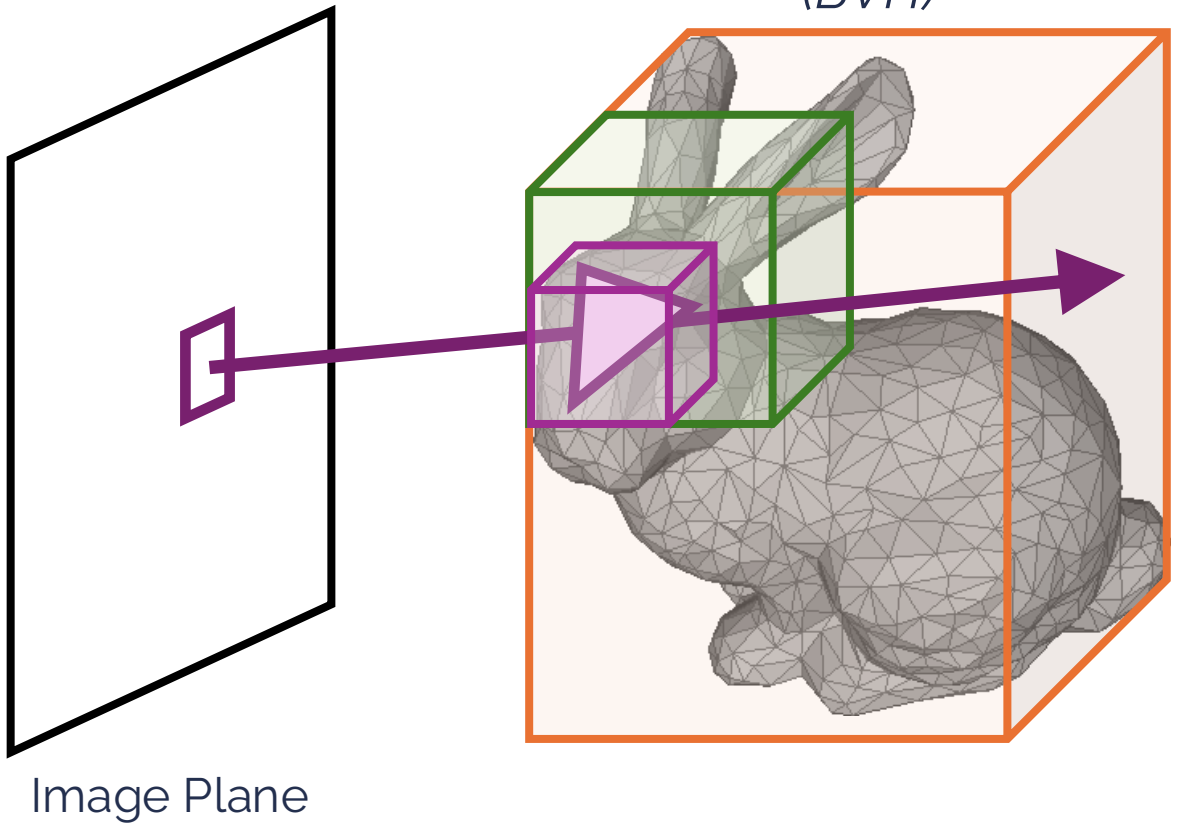


3D Gaussian-Based Rendering

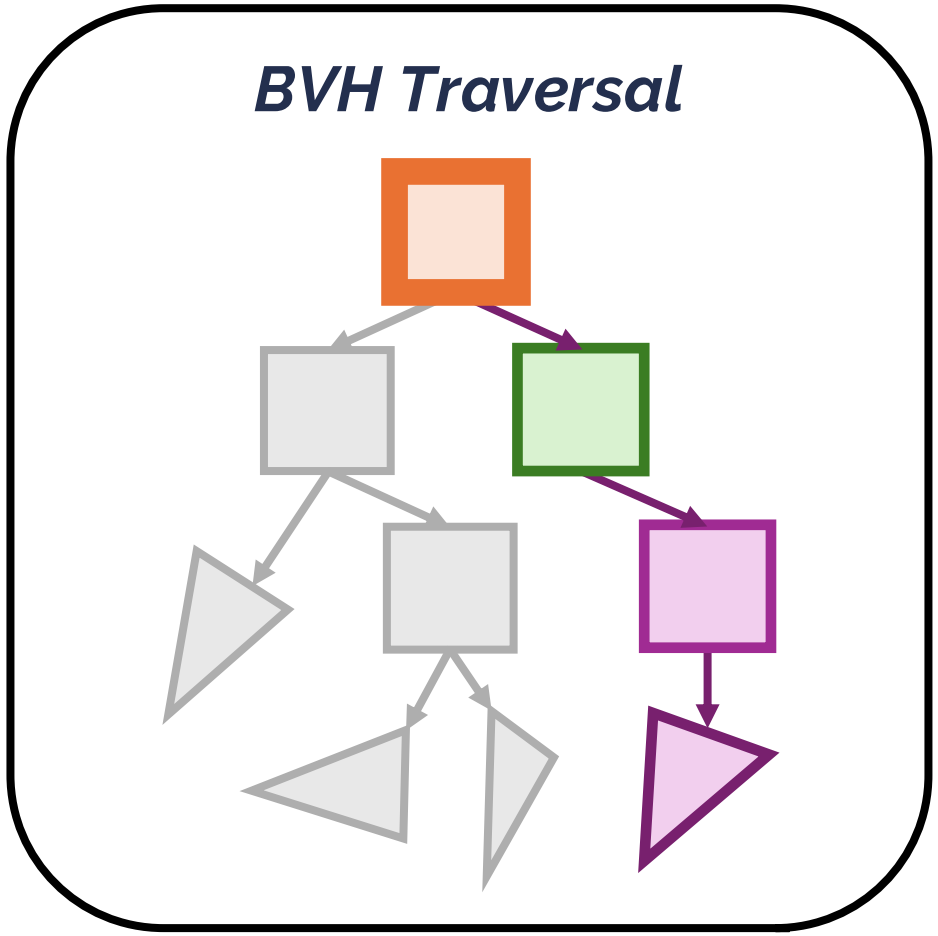


Ray-tracing

Bounding Volume Hierarchy (BVH)

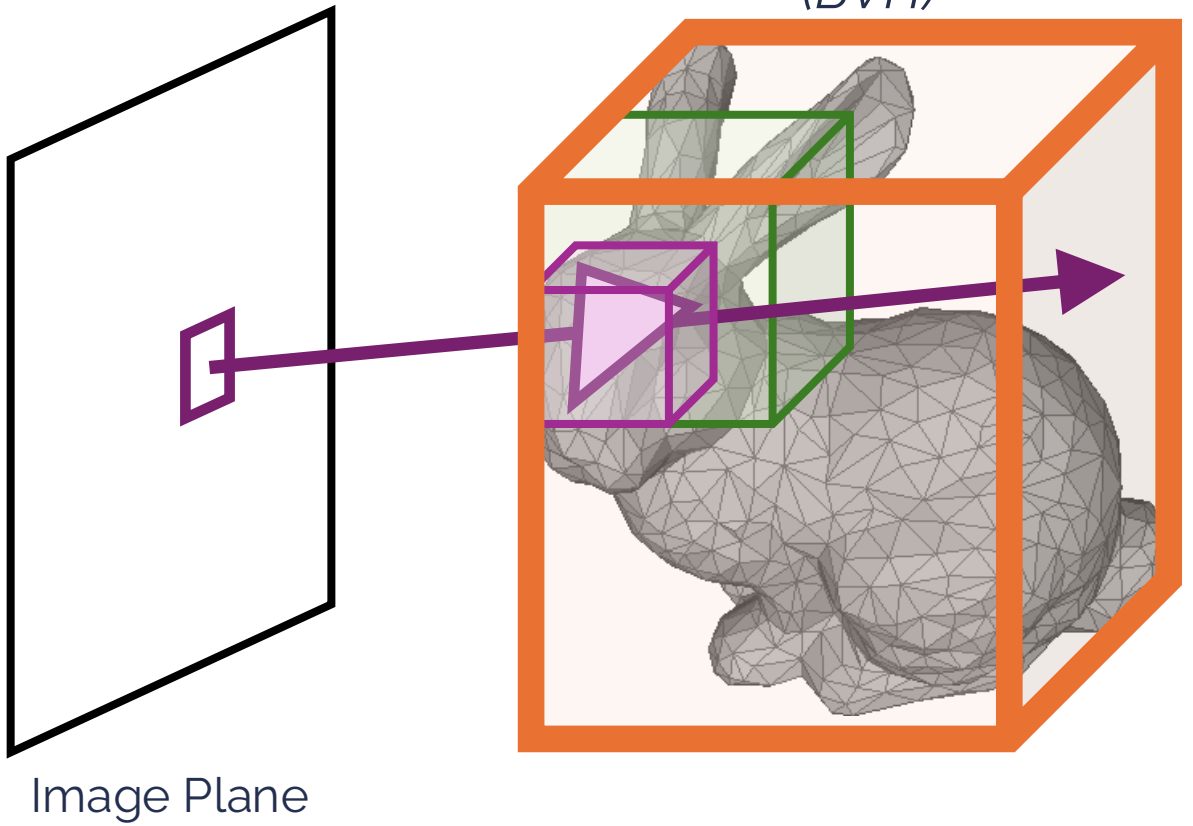


3D Gaussian-Based Rendering

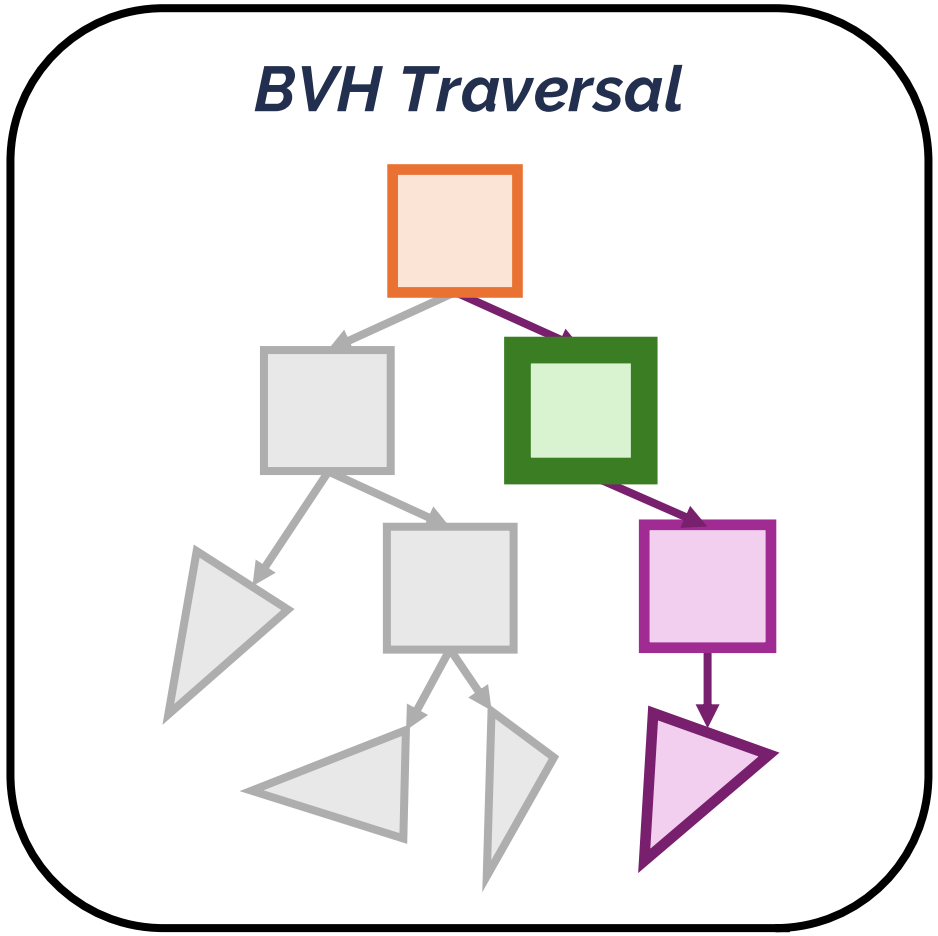


Ray-tracing

Bounding Volume Hierarchy (BVH)

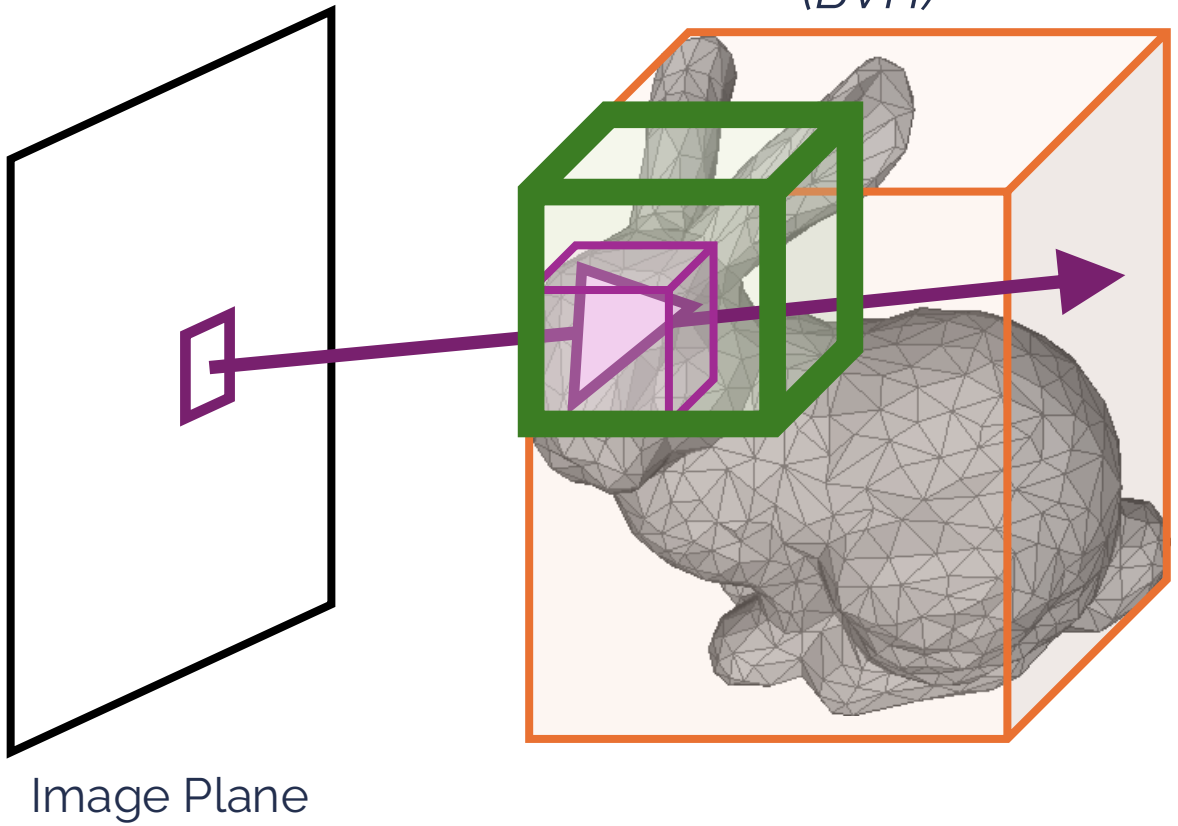


3D Gaussian-Based Rendering

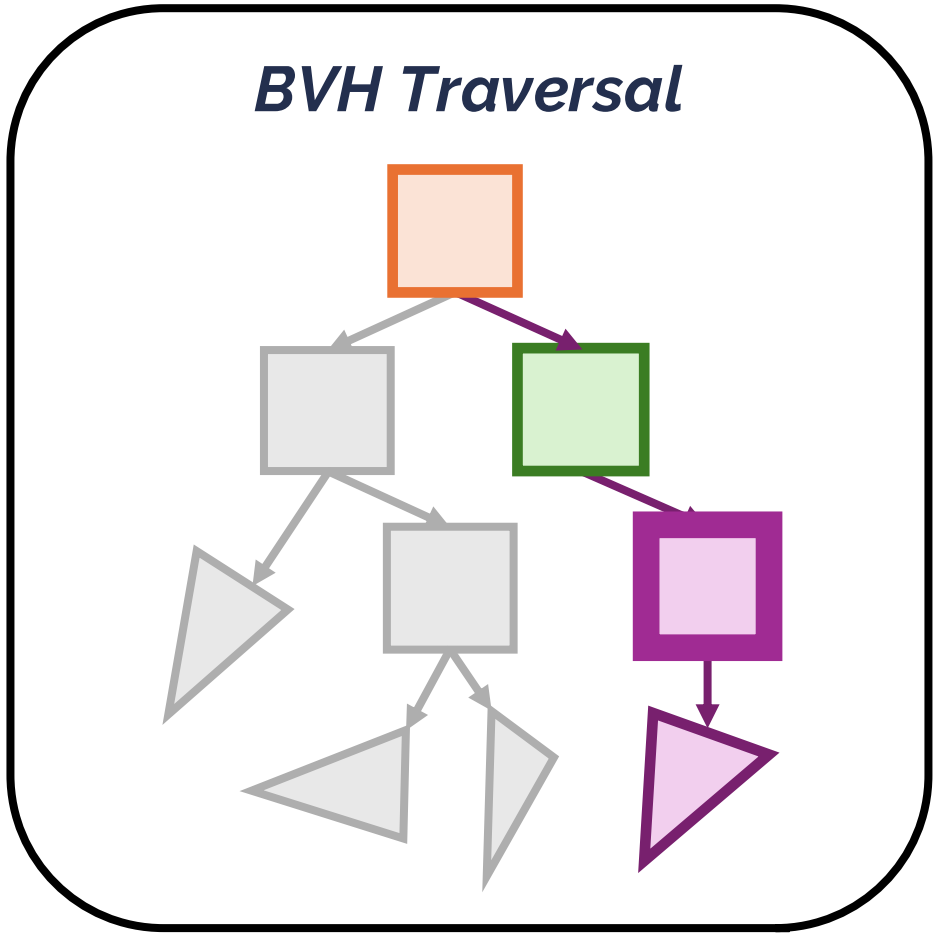


Ray-tracing

Bounding Volume Hierarchy (BVH)

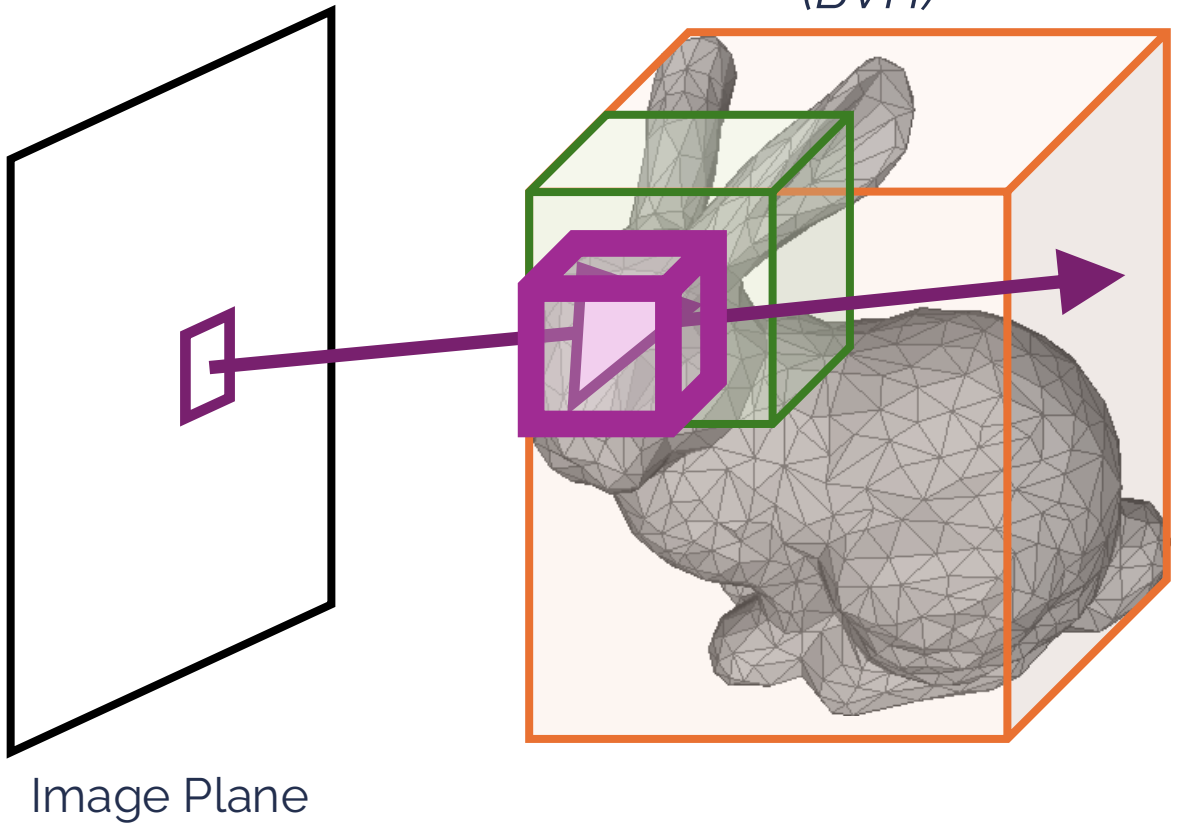


3D Gaussian-Based Rendering

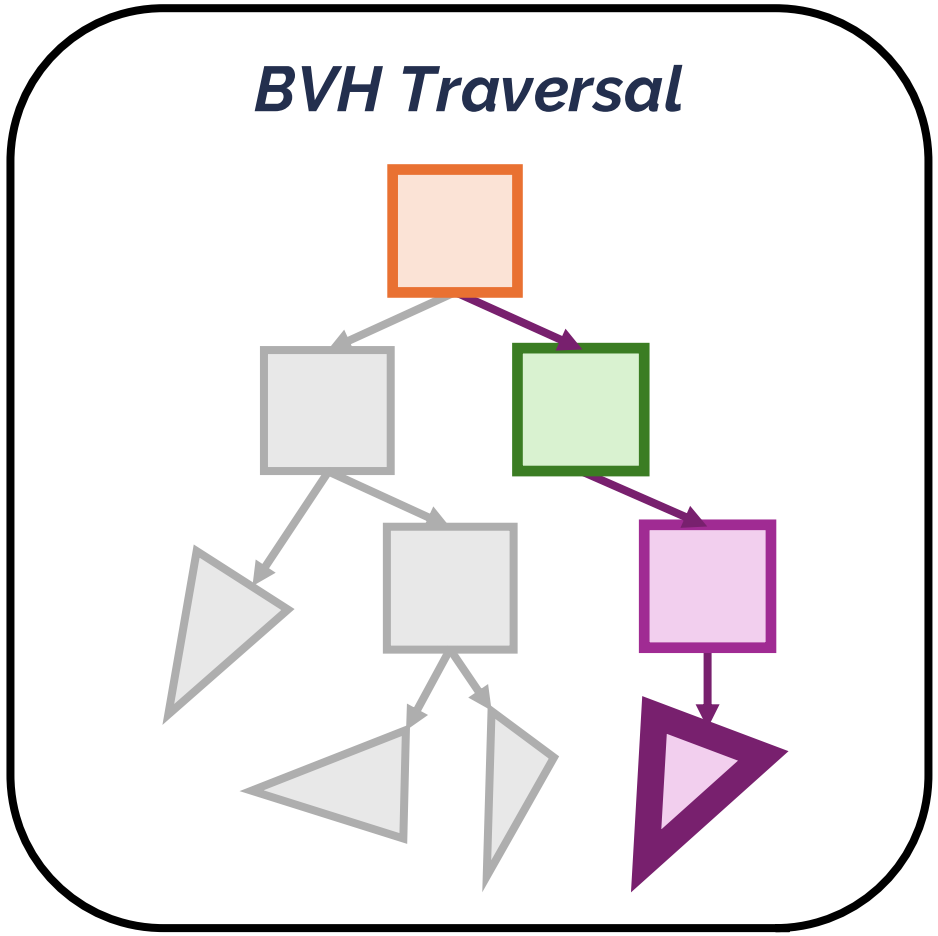


Ray-tracing

Bounding Volume Hierarchy (BVH)

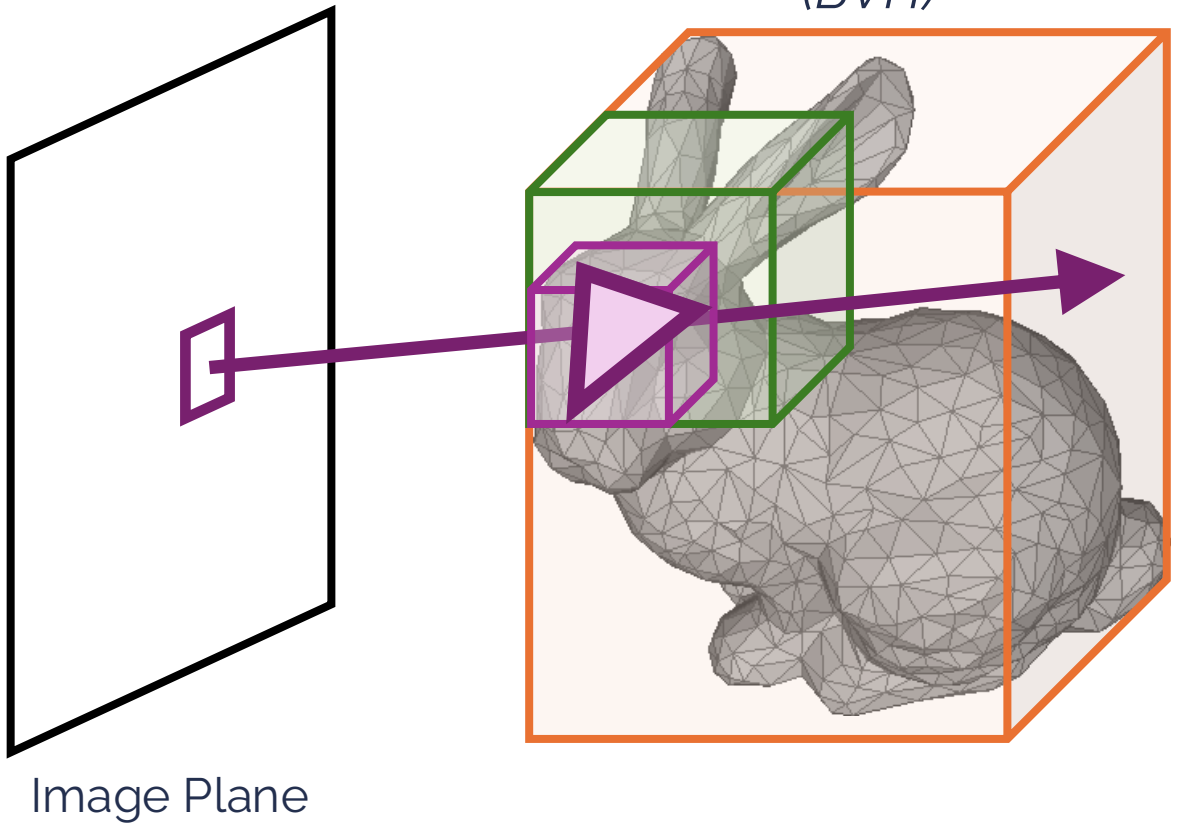


3D Gaussian-Based Rendering

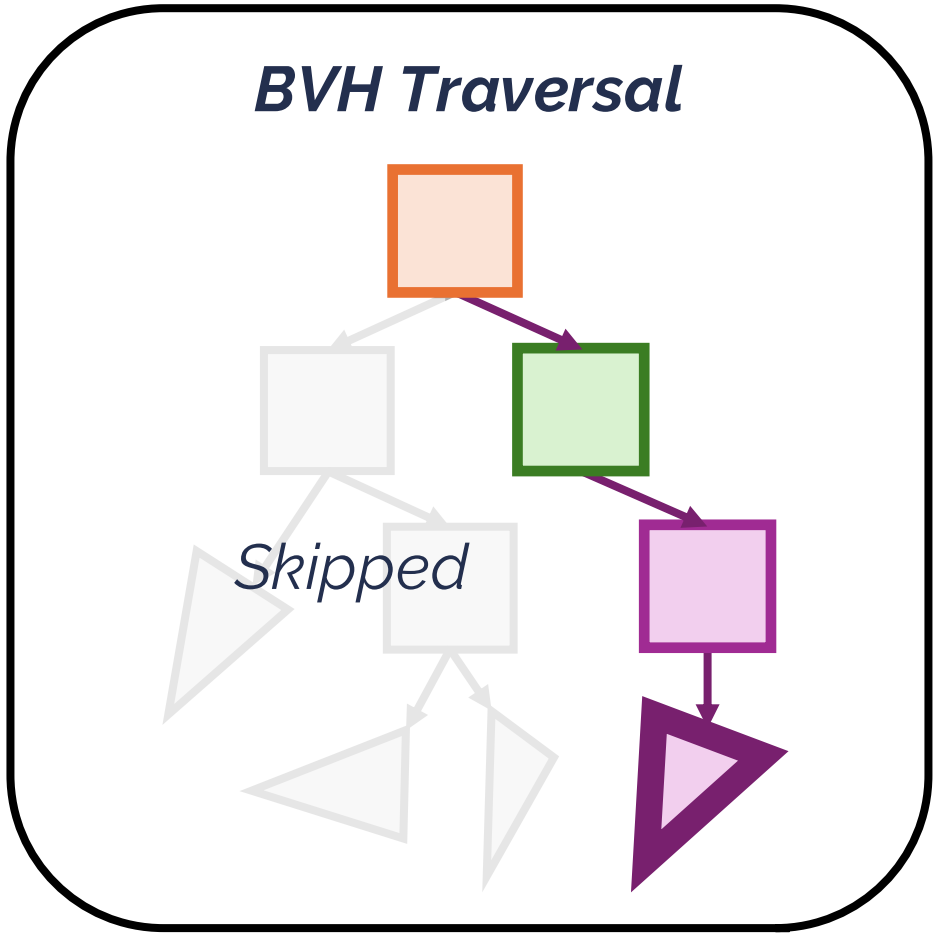


Ray-tracing

Bounding Volume Hierarchy (BVH)

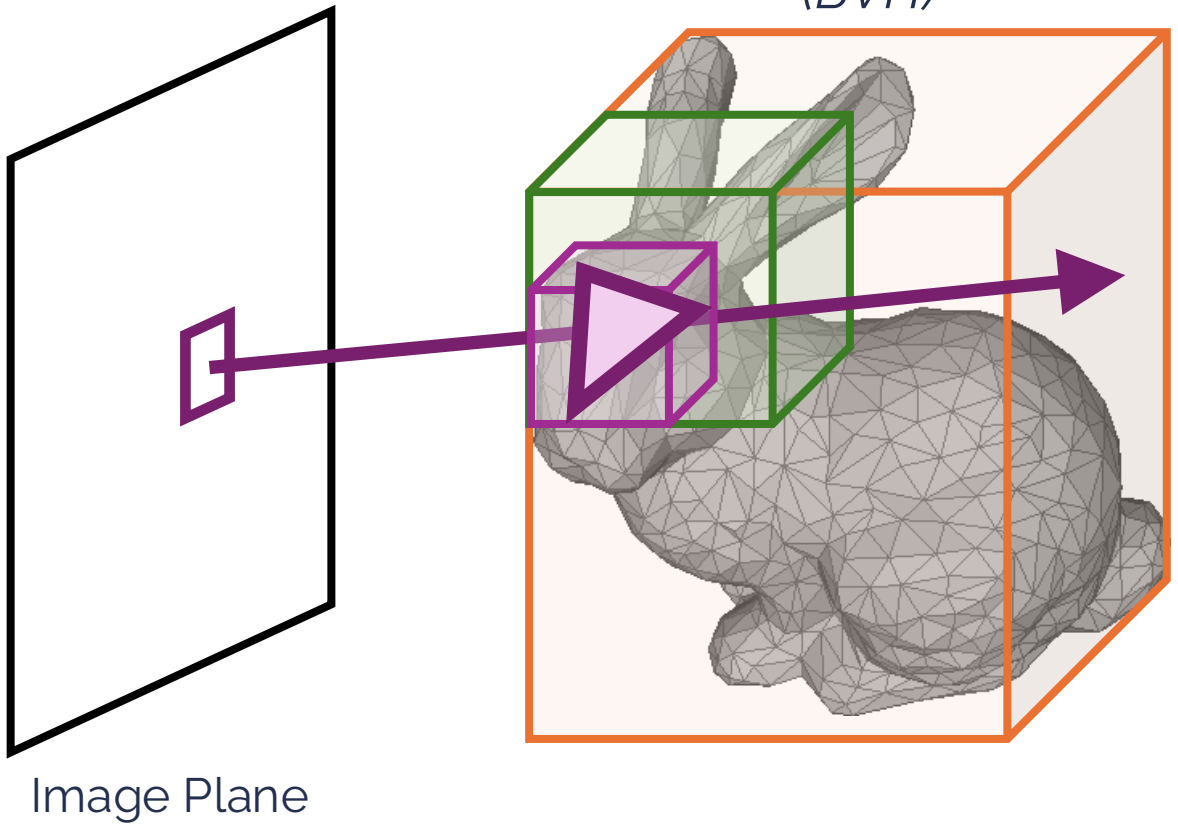


3D Gaussian-Based Rendering

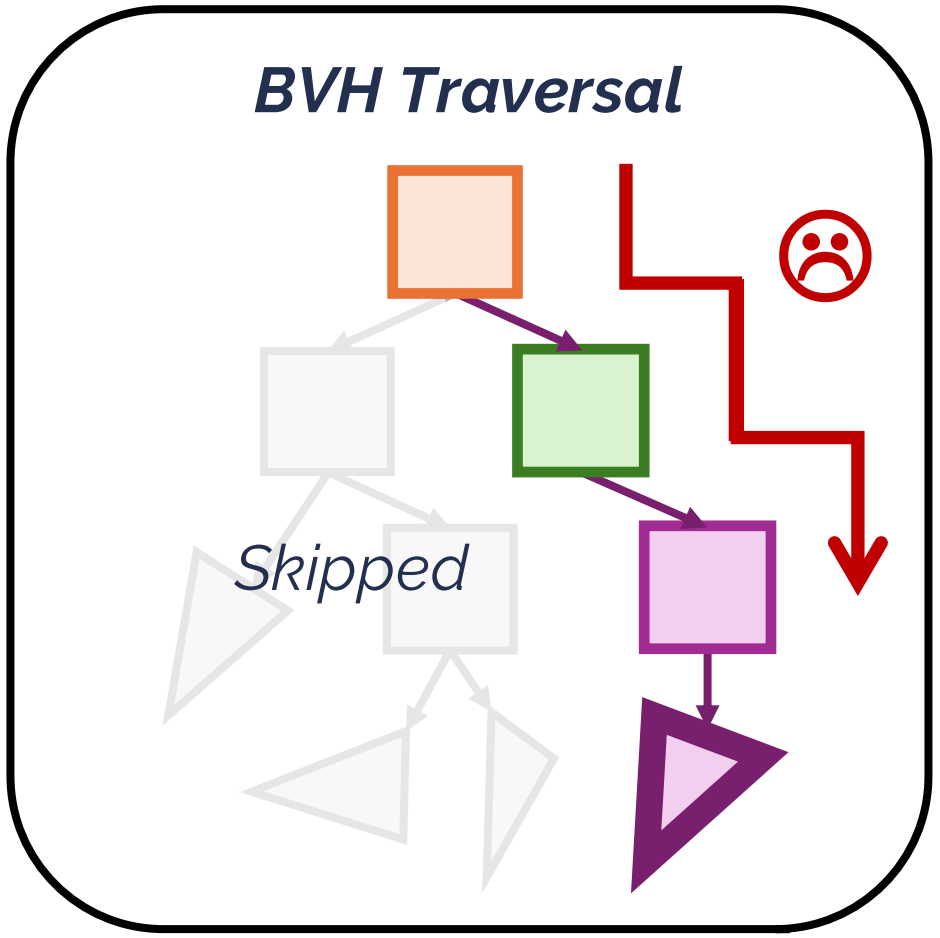


Ray-tracing

Bounding Volume Hierarchy (BVH)

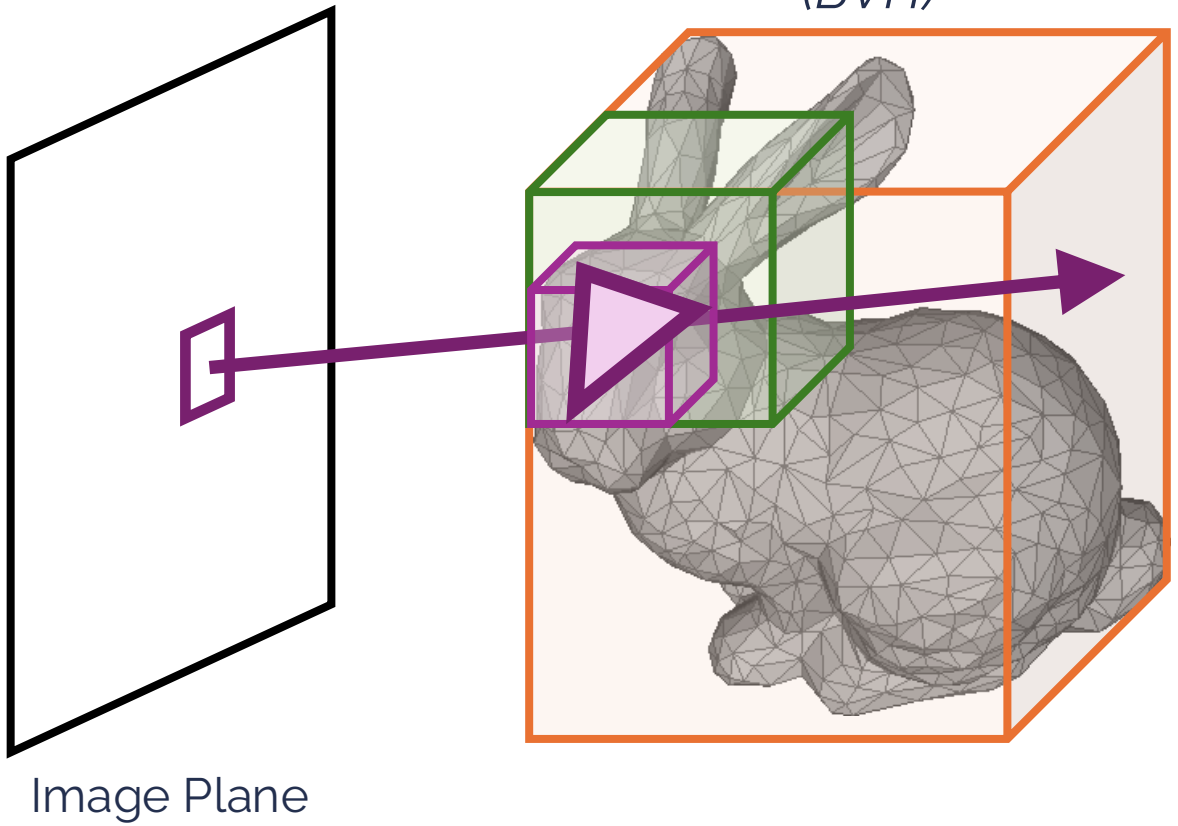


3D Gaussian-Based Rendering



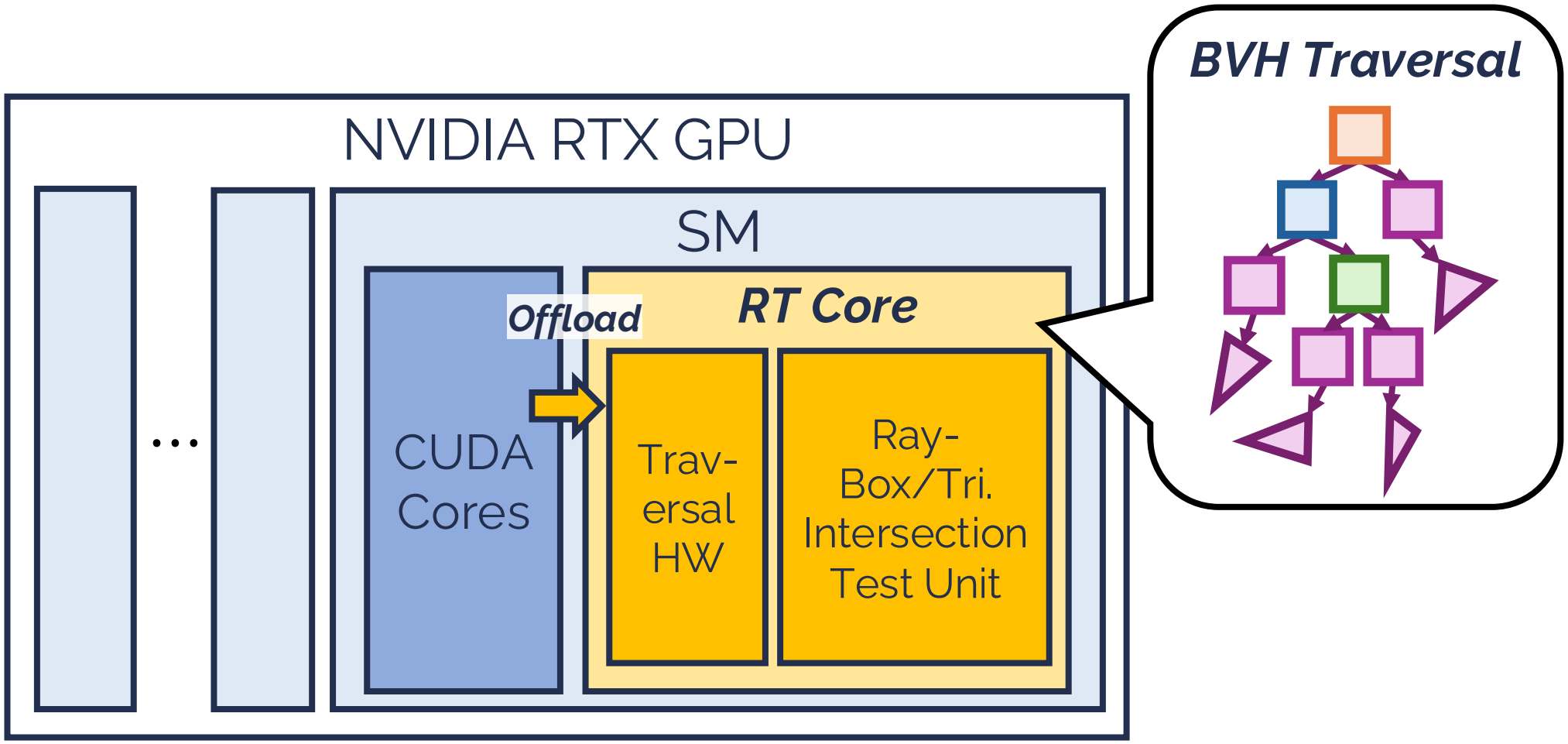
Ray-tracing

Bounding Volume Hierarchy (BVH)

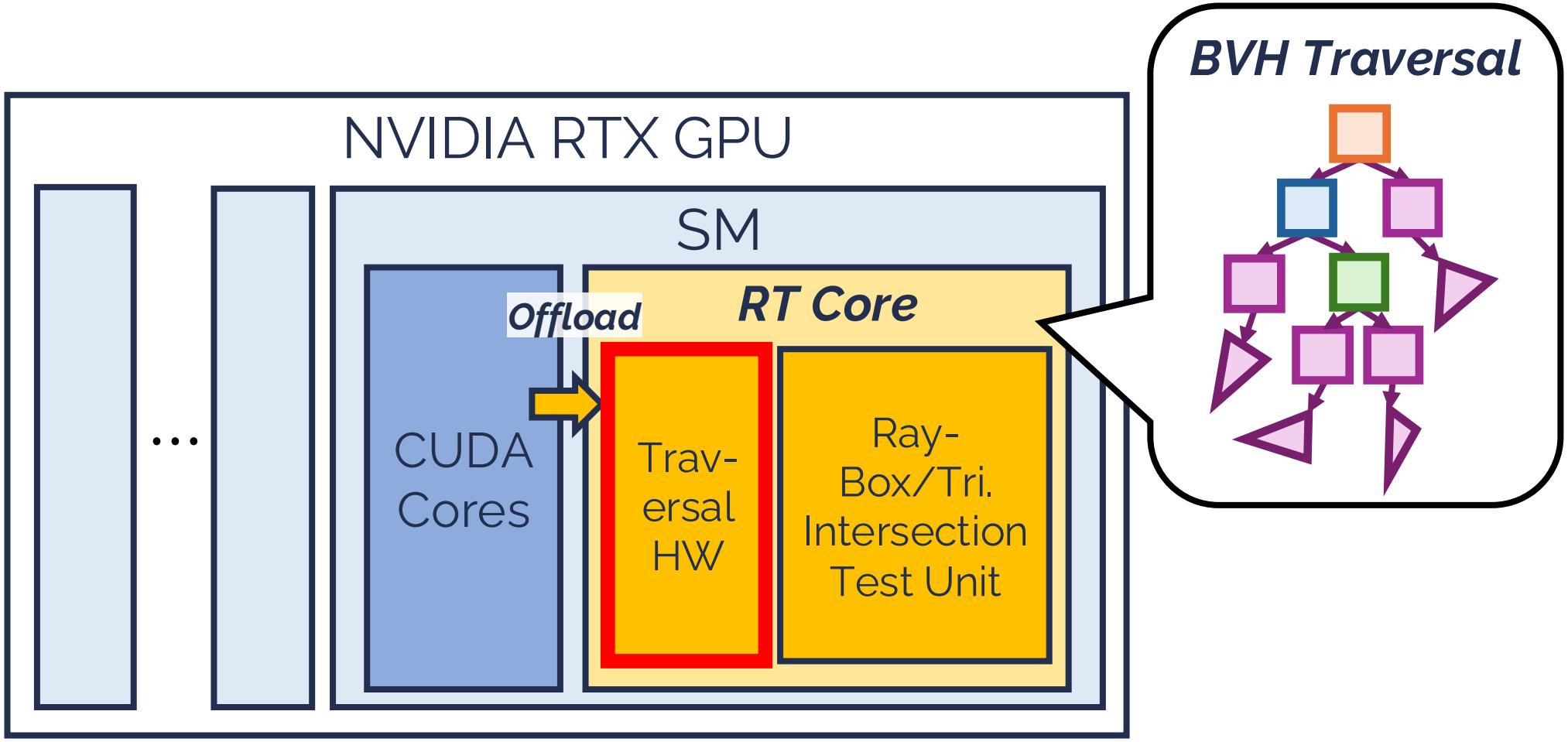


Ray Tracing Accelerators in GPUs

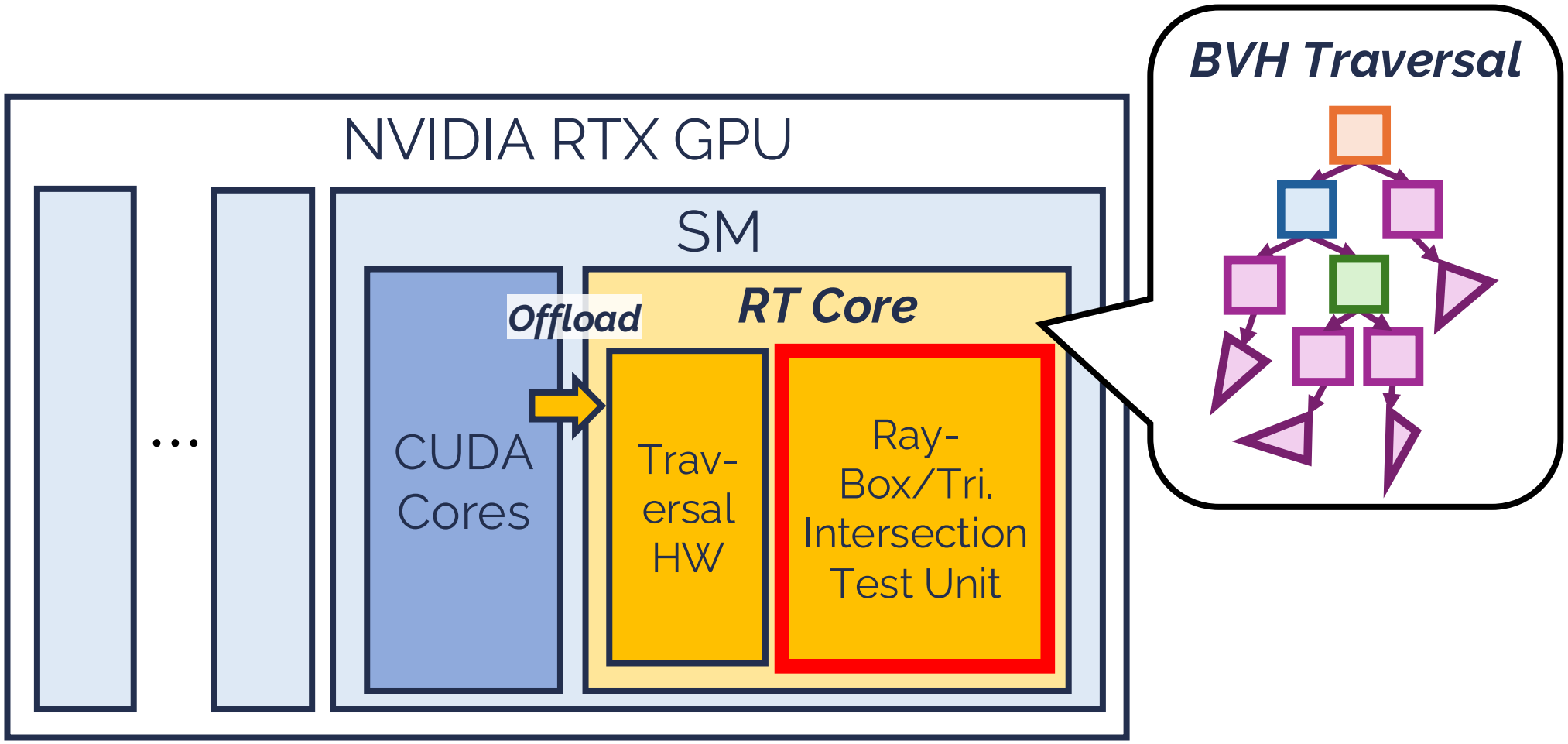
Ray Tracing Accelerators in GPUs



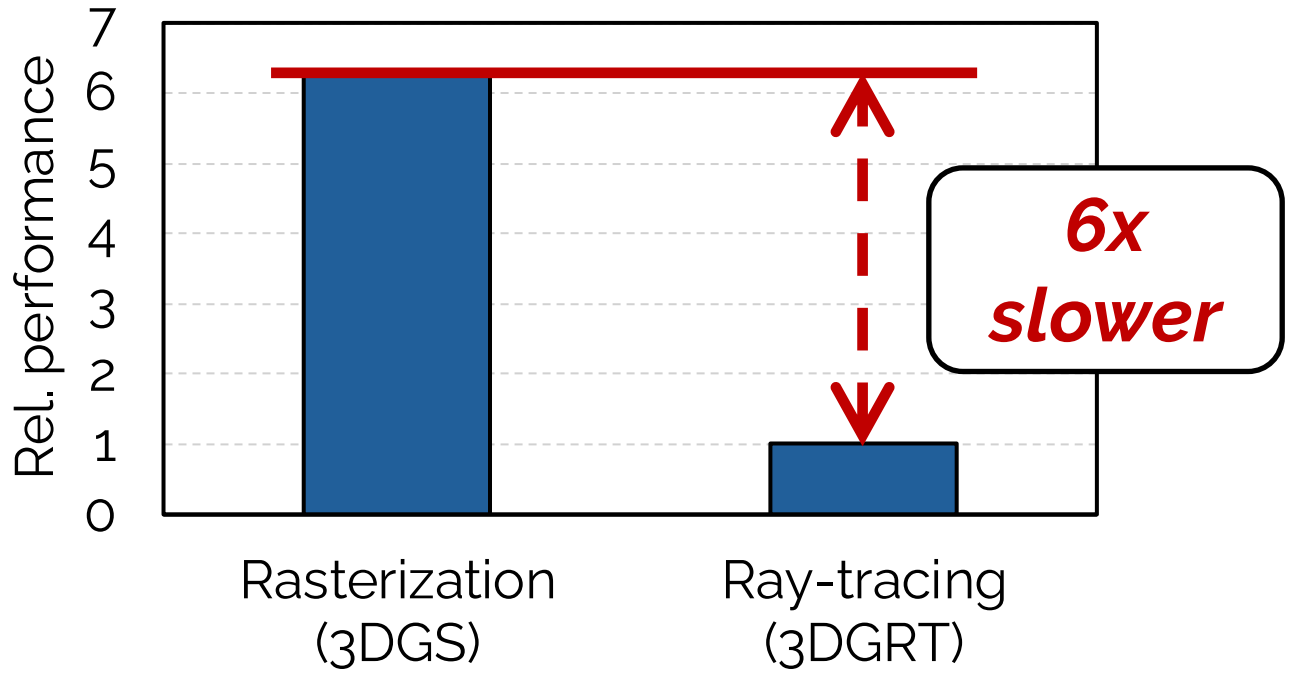
Ray Tracing Accelerators in GPUs



Ray Tracing Accelerators in GPUs



Goal of This Work



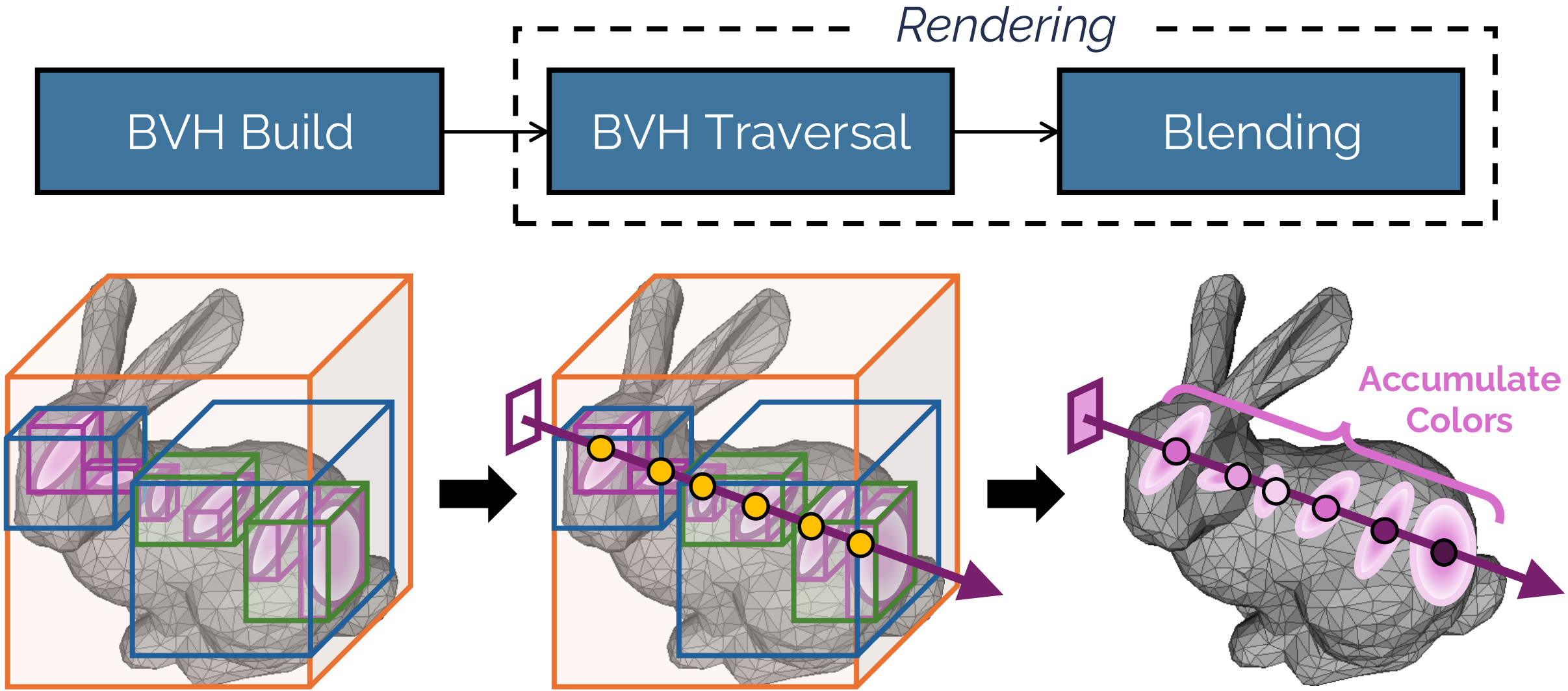
Goal

Reduce the performance gap
with **SW-HW optimizations for Gaussian ray-tracing**

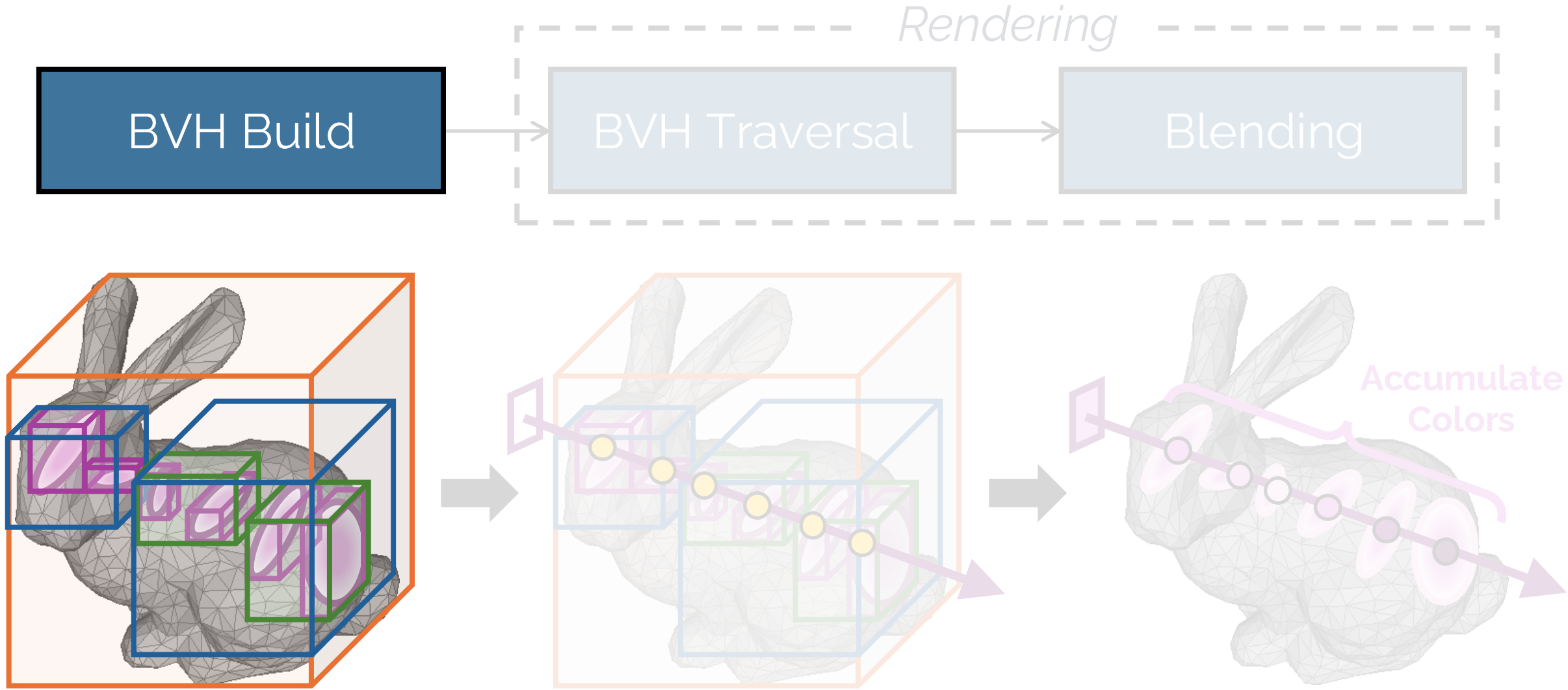
Outline

- **Background**
  - 3D Gaussian-based Rendering: Rasterization vs. Ray-tracing
  - Ray Tracing Accelerators in Modern GPUs
- **Gaussian RT Optimizations & Limitations**
- **GRTX: SW-HW Optimizations for Gaussian Ray Tracing**
  - GRTX-SW: Two-Level Acceleration Structure for Gaussian Primitives
  - GRTX-HW: HW Extension for Traversal Checkpointing and Replay
- **Evaluation**
- **Conclusion**

Overview of Gaussian Ray Tracing



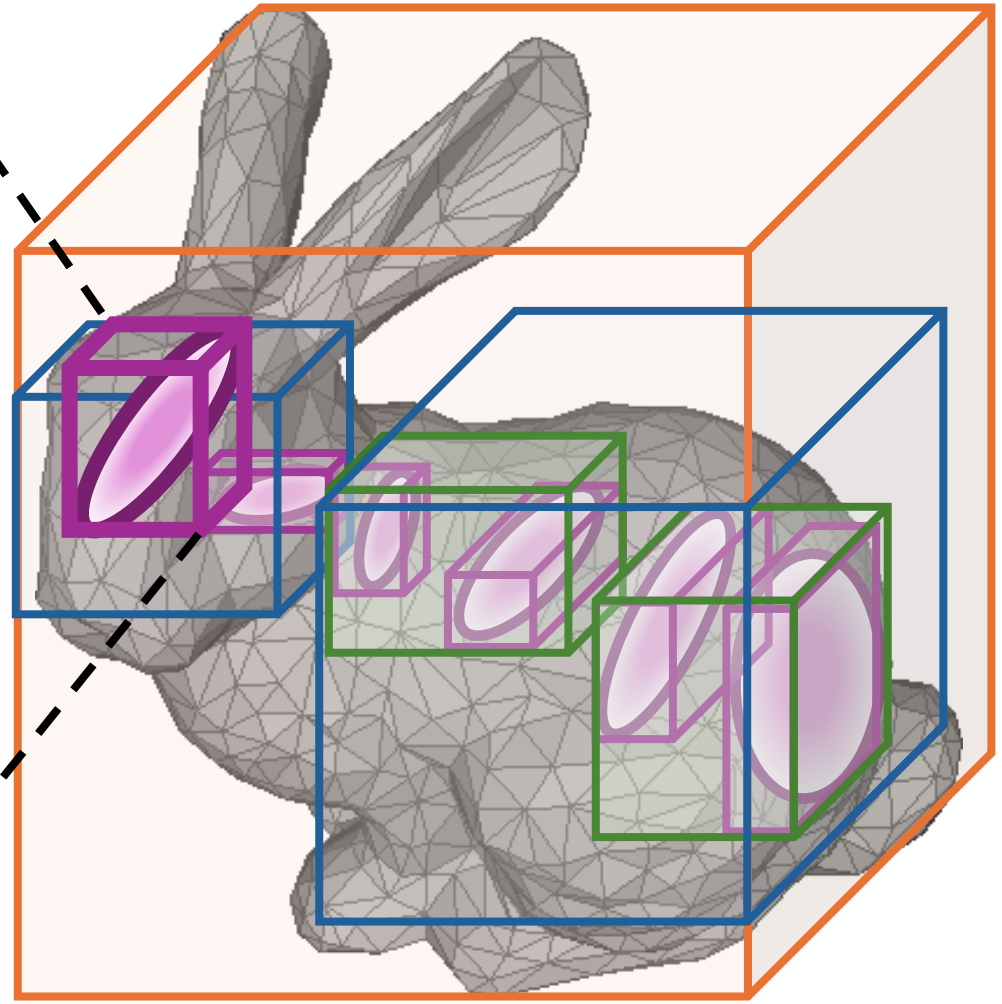
Overview of Gaussian Ray Tracing



Gaussian RT Optimizations & Limitations

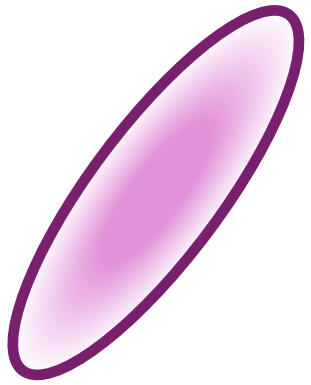
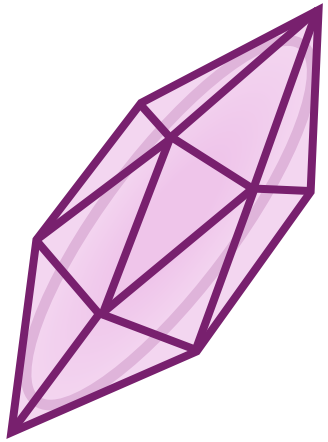
Primitive Types

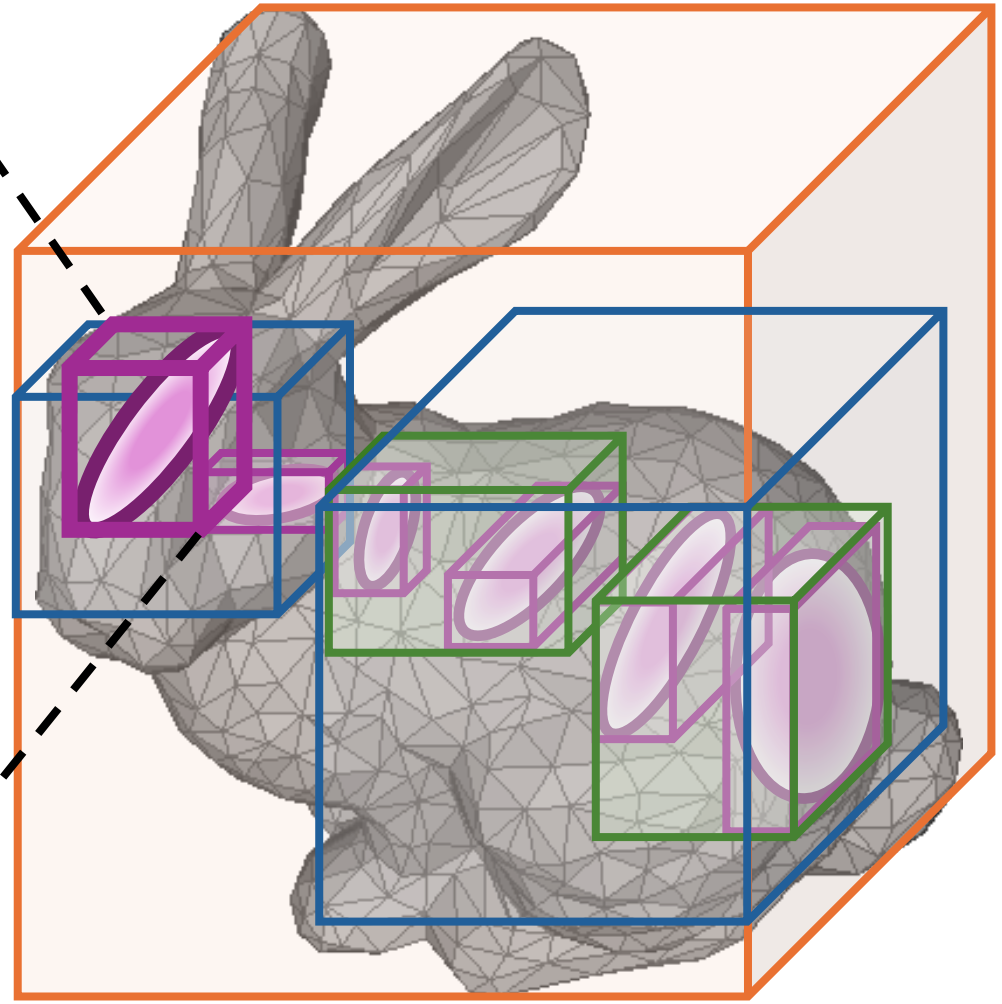
	Gaussian Primitive	Bounding Triangle Mesh
☺	Compact BVH (1 Gaus. = 1 Prim.)	HW-based intersection test
☹	**SW-based intersection test**	**Large BVH (1 Gaus. = 20 Prims.)**



Gaussian RT Optimizations & Limitations

Primitive Types

	 Gaussian Primitive	 Bounding Triangle Mesh
☺	Compact BVH (1 Gaus. = 1 Prim.)	HW-based intersection test
☹	**SW-based intersection test**	**Large BVH (1 Gaus. = 20 Prims.)**



Gaussian RT Optimizations & Limitations

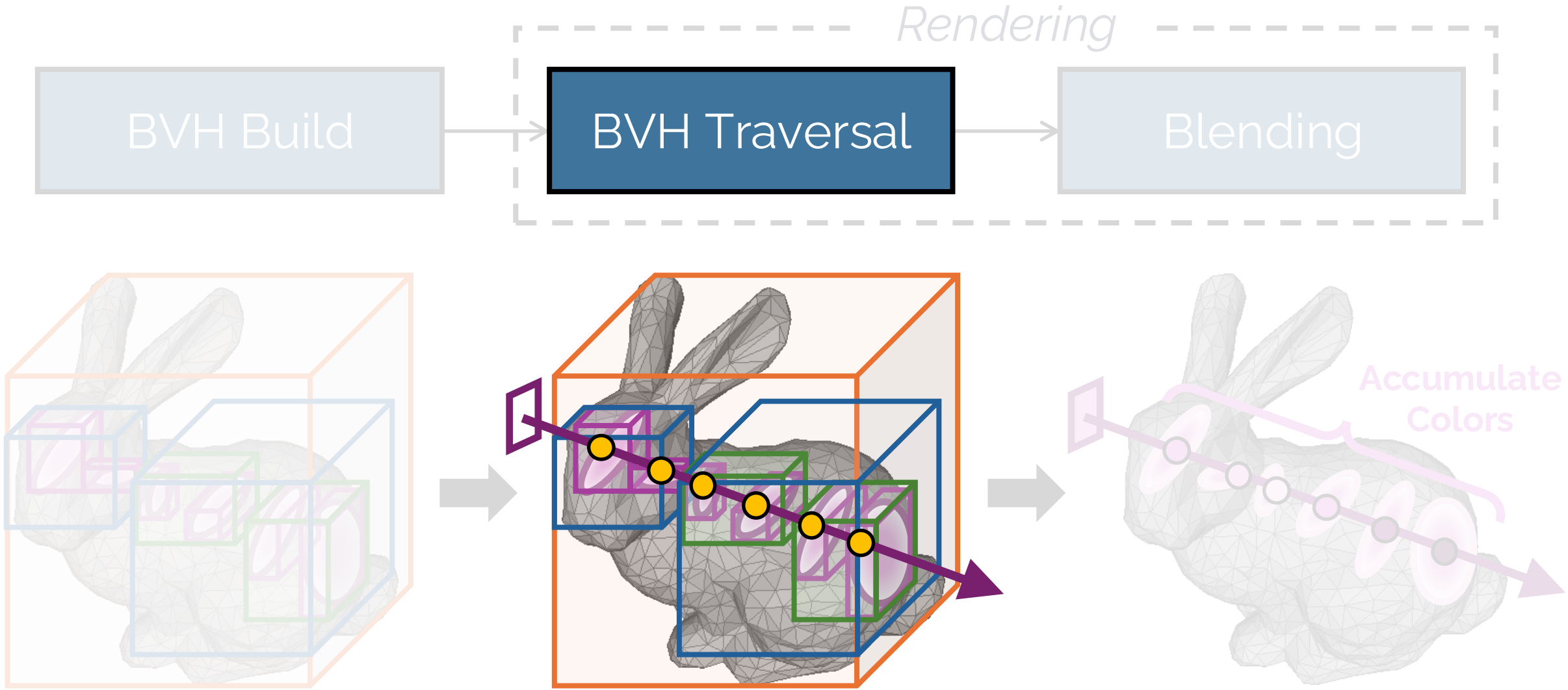
Primitive Types

	Gaussian Primitive	Bounding Triangle Mesh
☺	Compact BVH (1 Gaus. = 1 Prim.)	HW-based intersection test
☹	**SW-based intersection test**	**Large BVH (1 Gaus. = 20 Prims.)**

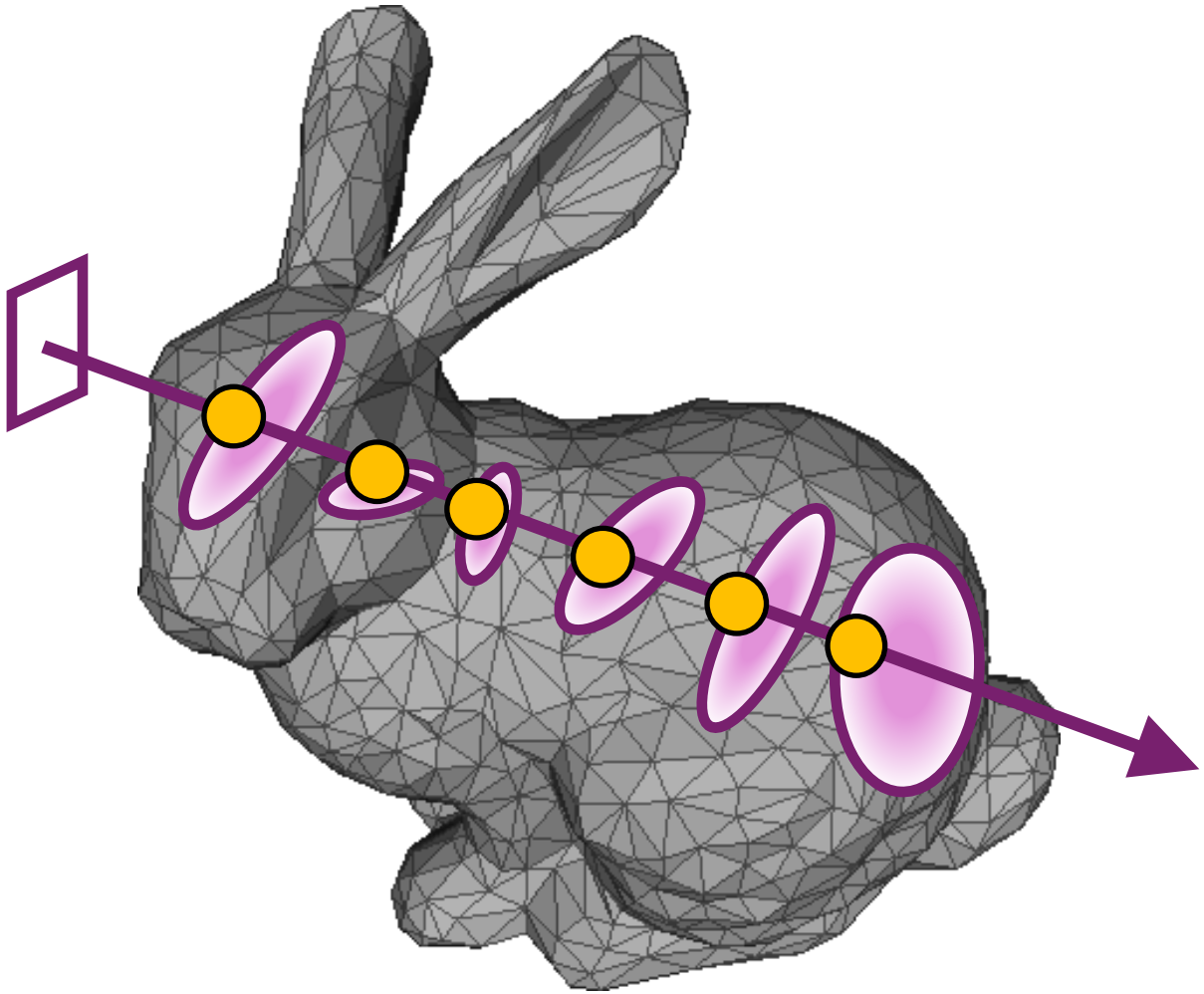
Problem 1

Bloated BVH size and increased **memory footprint**

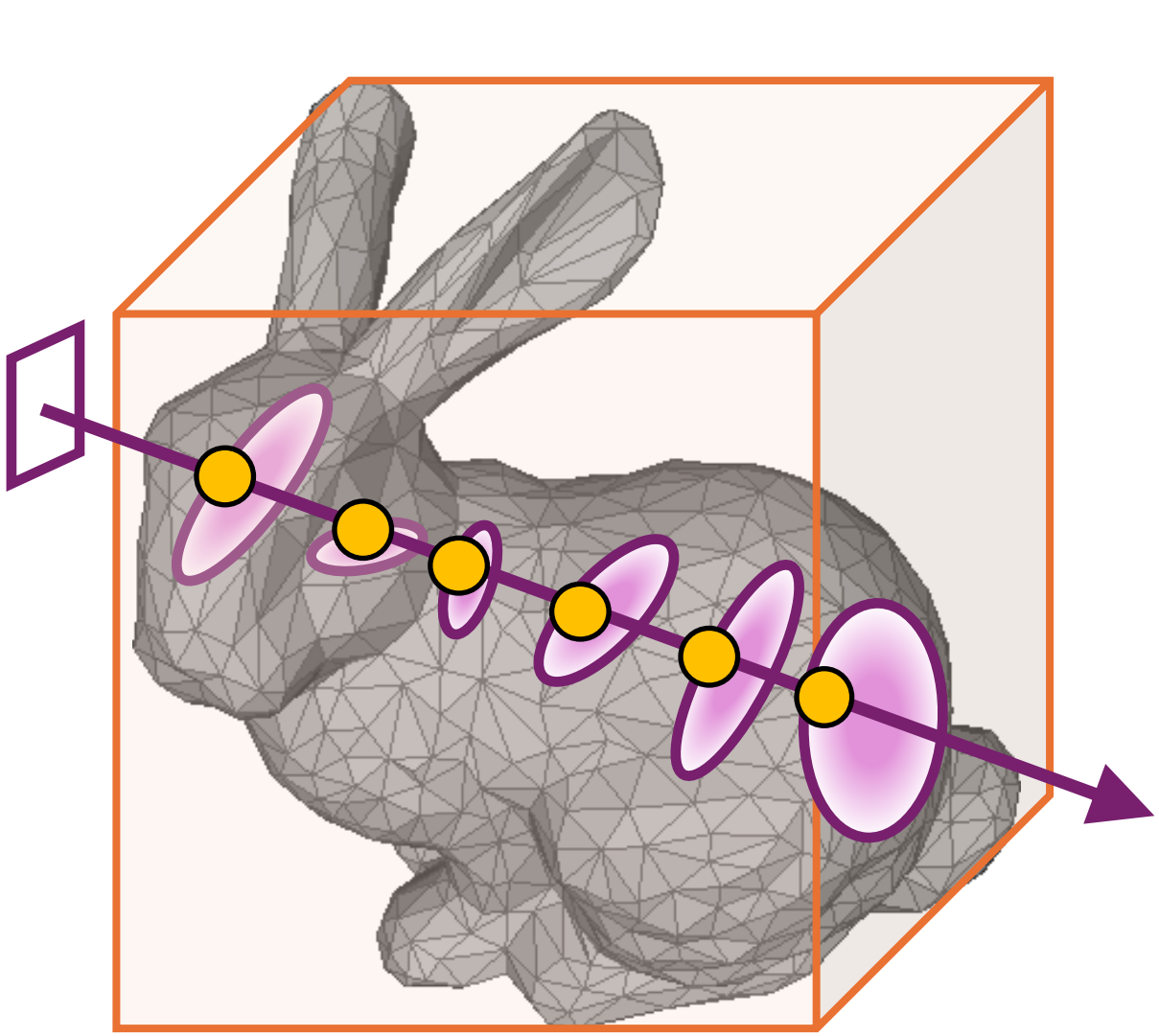
Gaussian RT Optimizations & Limitations



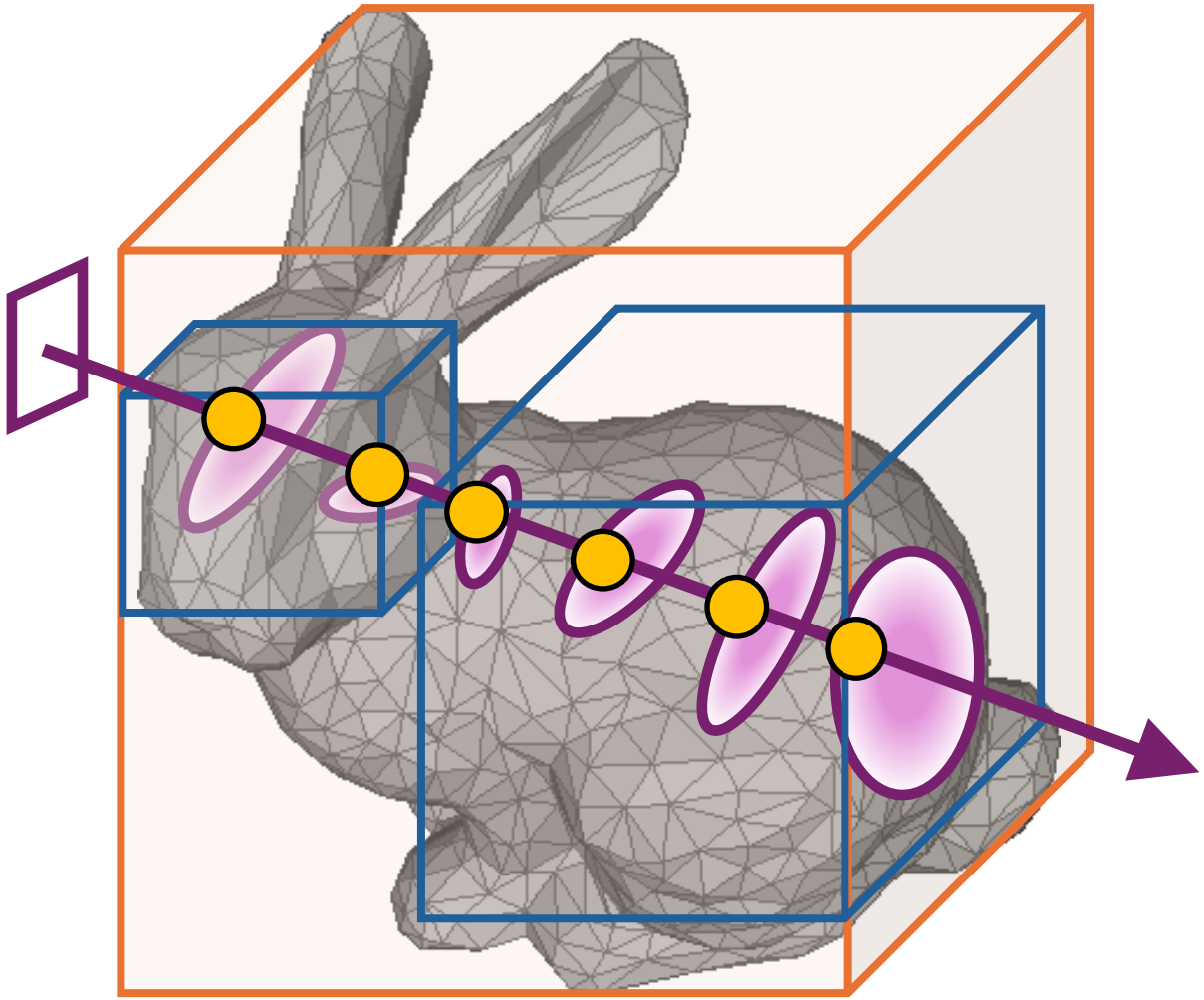
Gaussian RT Optimizations & Limitations

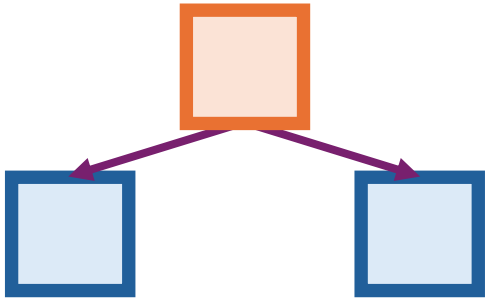


Gaussian RT Optimizations & Limitations

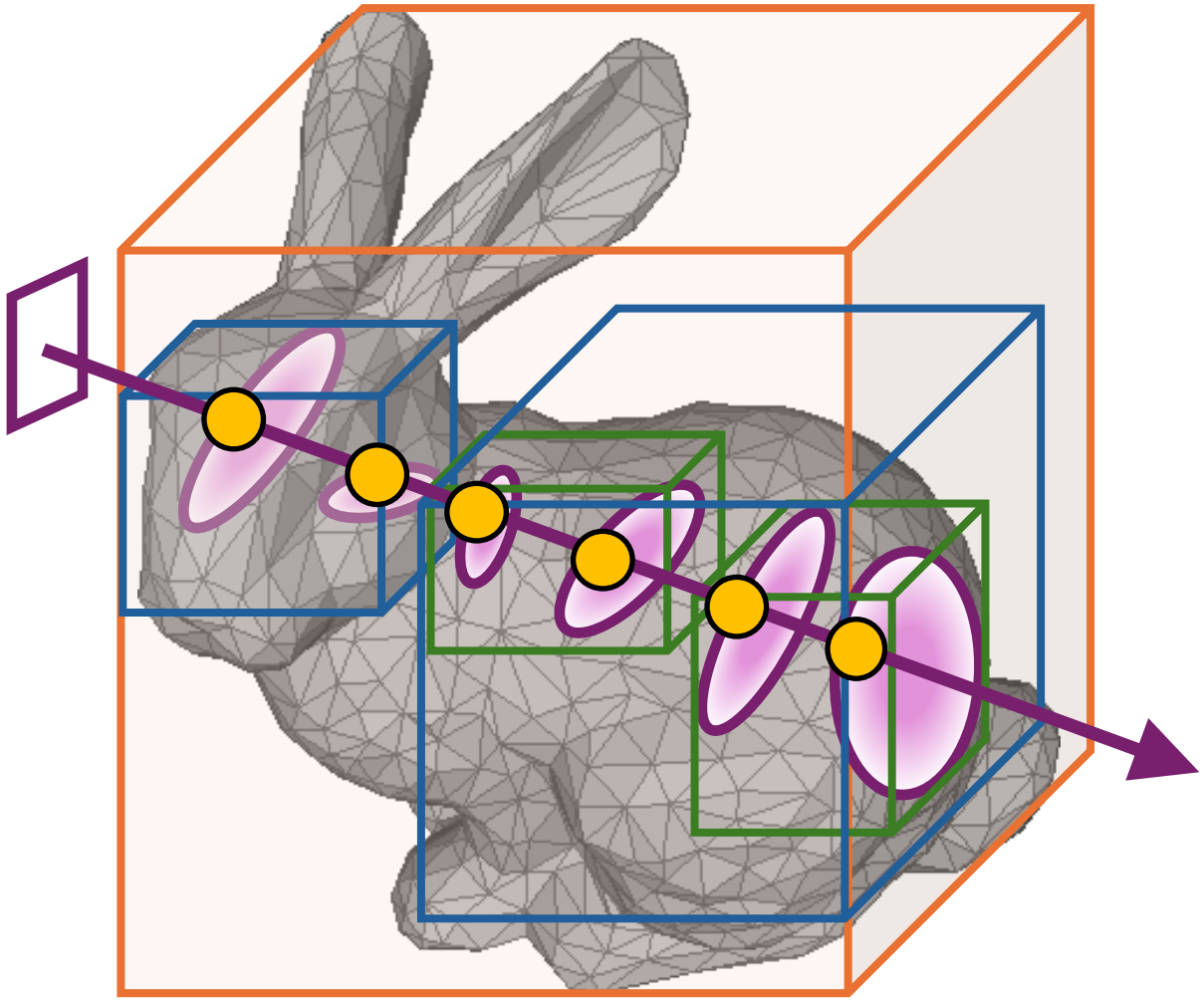


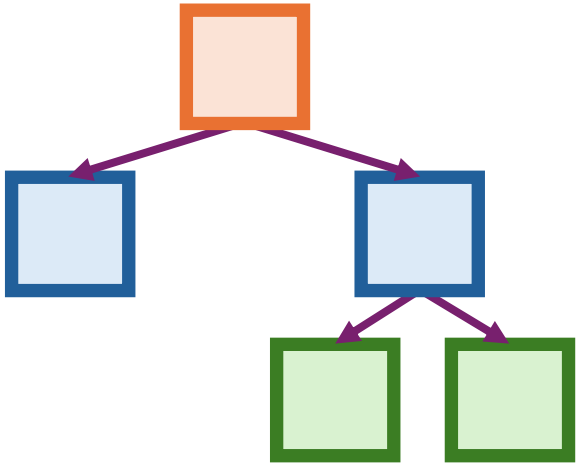
Gaussian RT Optimizations & Limitations



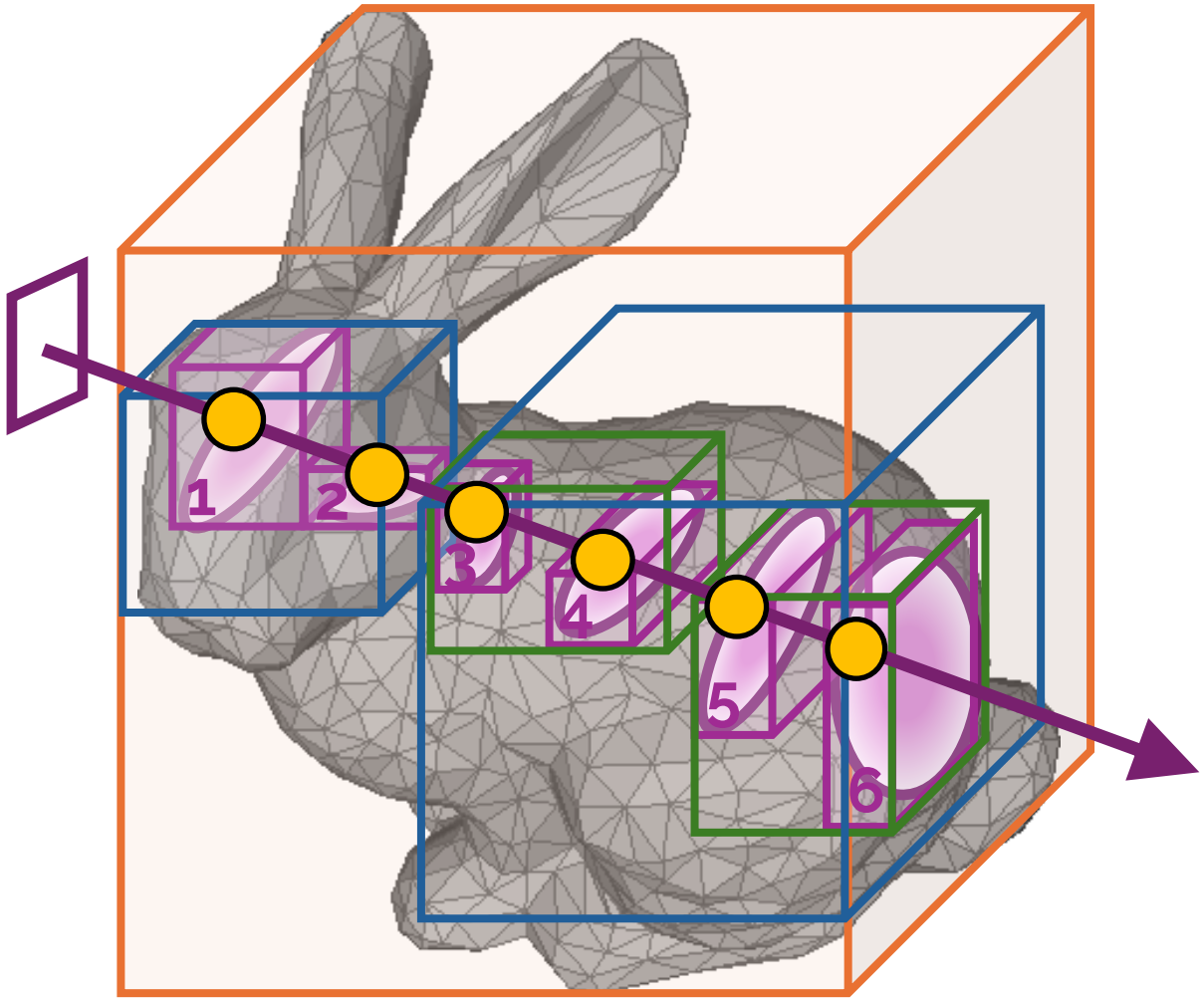


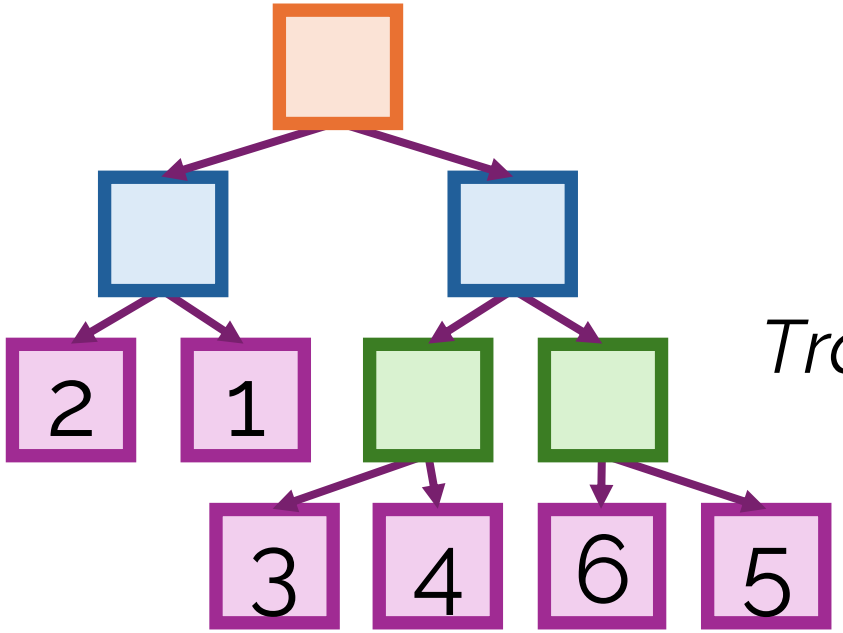
Gaussian RT Optimizations & Limitations





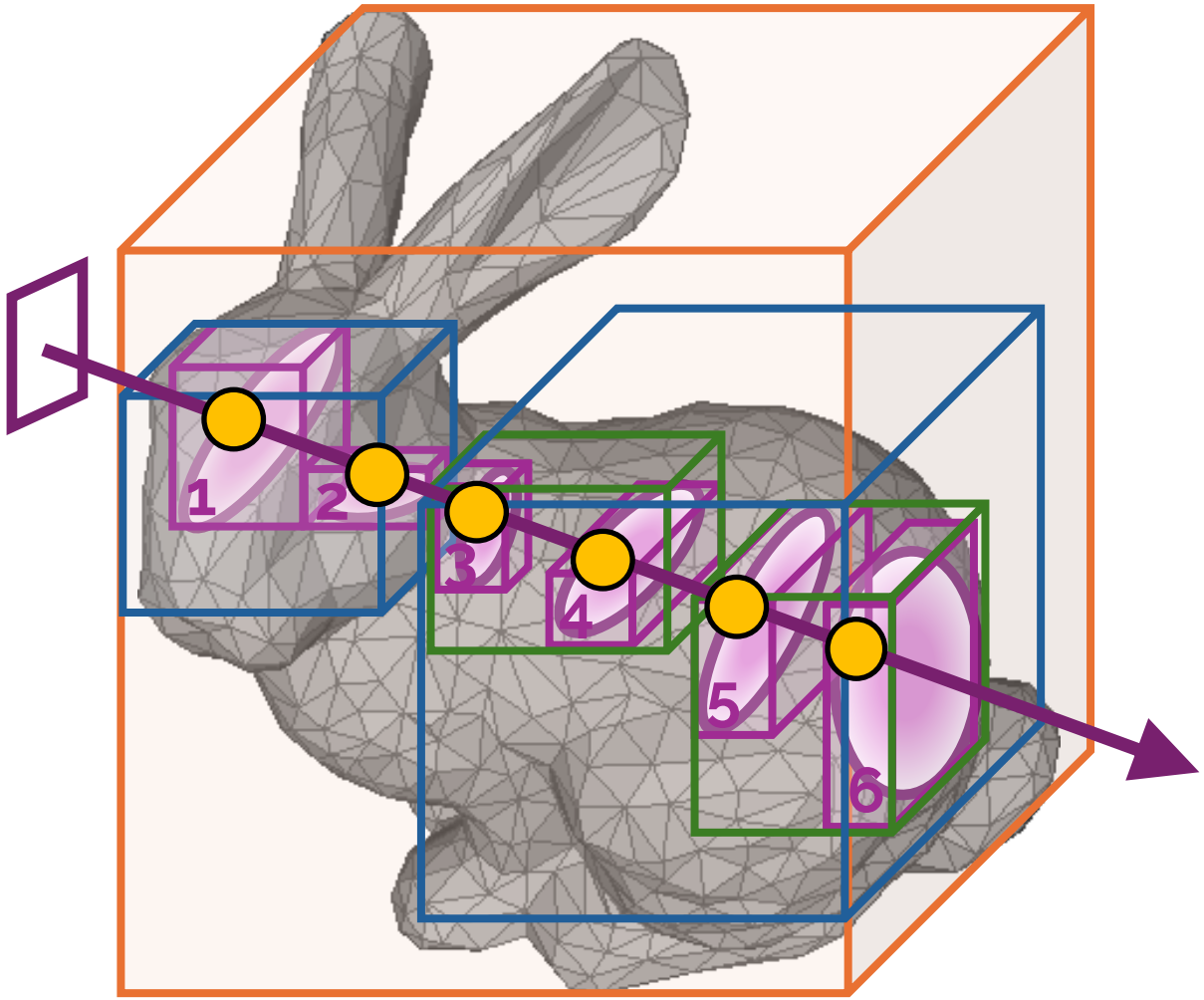
Gaussian RT Optimizations & Limitations

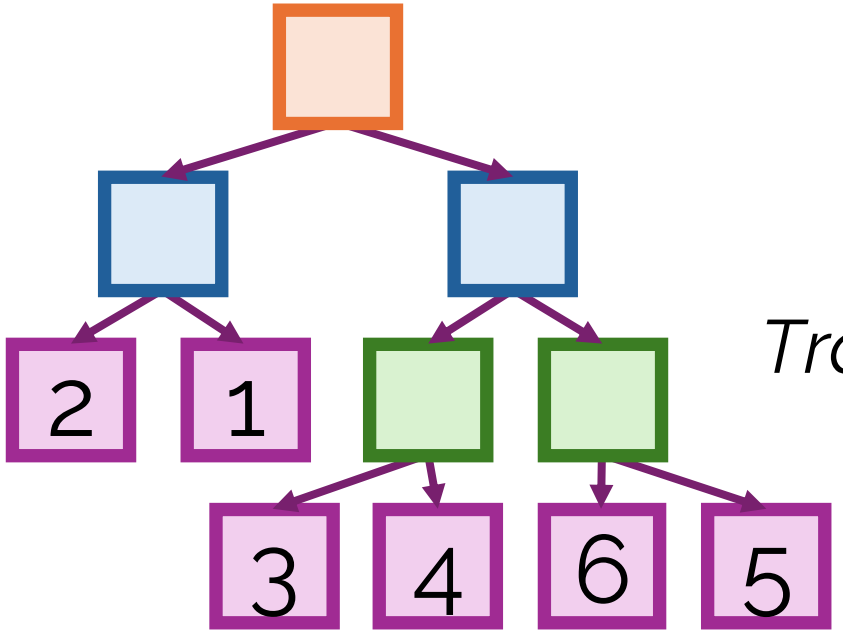




BVH Traversal

Gaussian RT Optimizations & Limitations



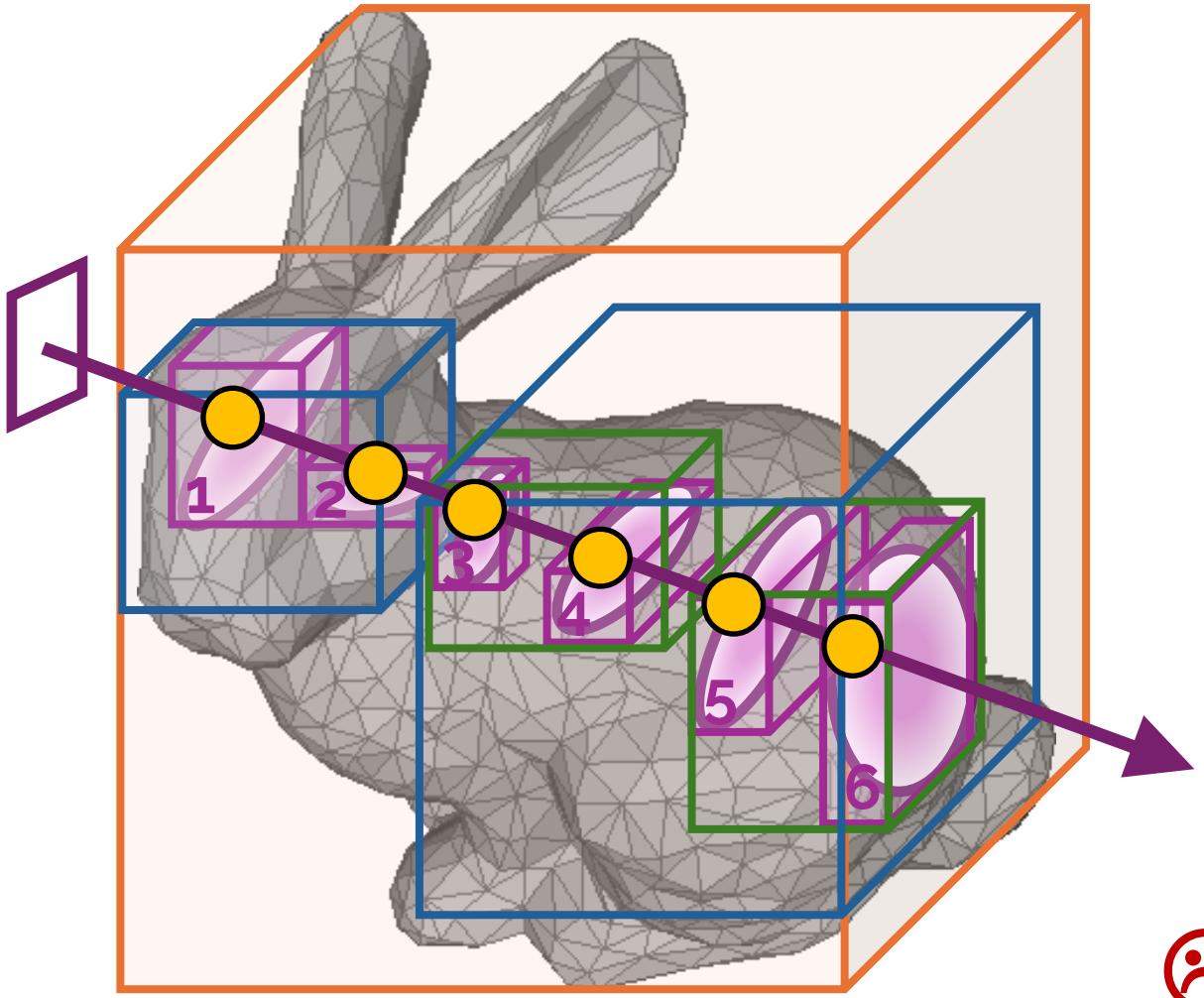


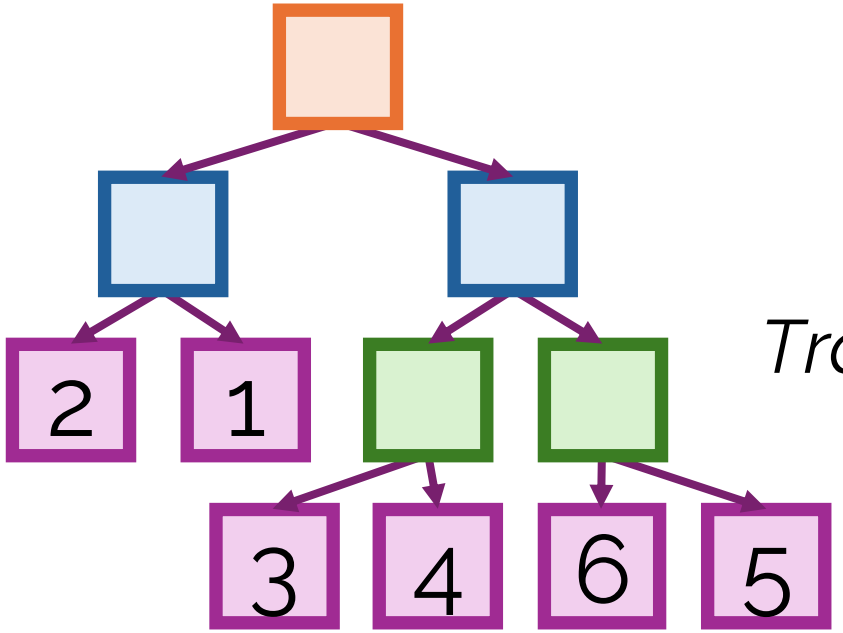
BVH Traversal

Front-to-Back Order

1 → 2 → 3 → 4 → 5 → 6

Gaussian RT Optimizations & Limitations





BVH Traversal

Front-to-Back Order

1 → 2 → 3 → 4 → 5 → 6

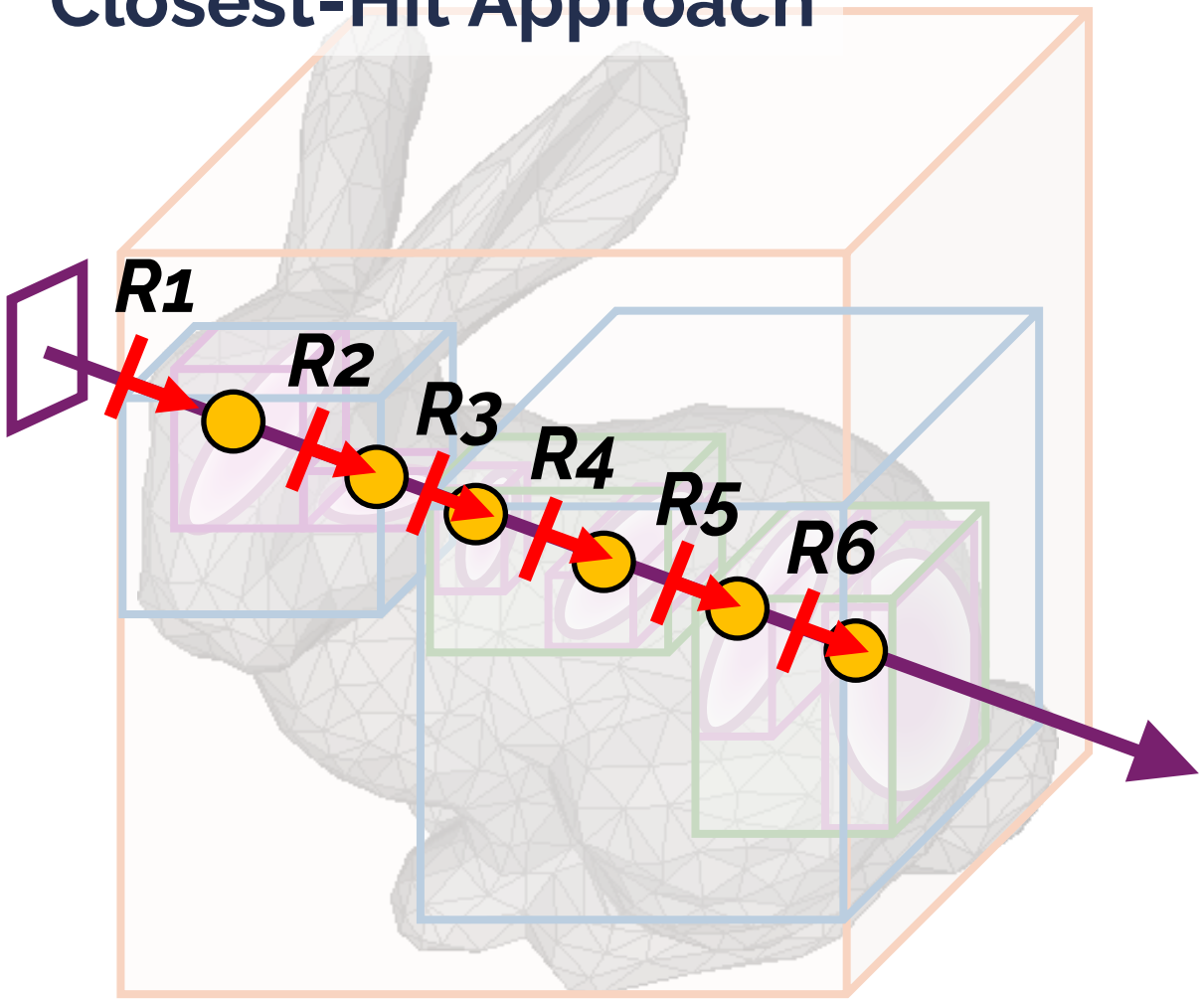
☹

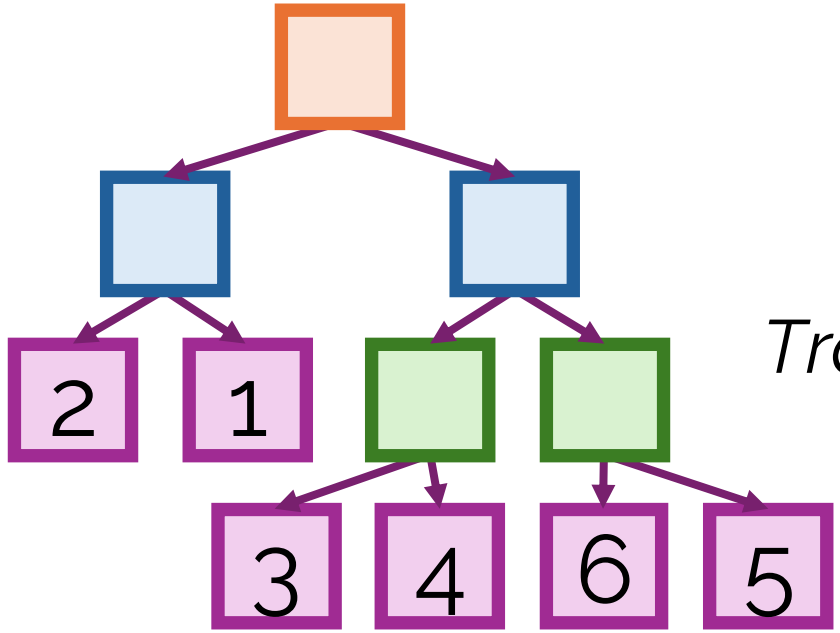
Traversed

2 → 1 → 3 → 4 → 6 → 5

Gaussian RT Optimizations & Limitations

Closest-Hit Approach

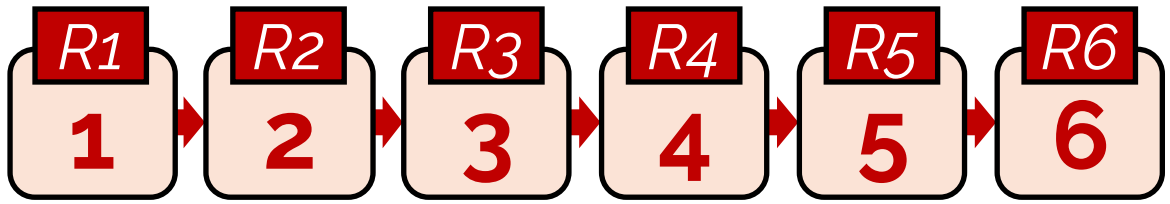




BVH Traversal

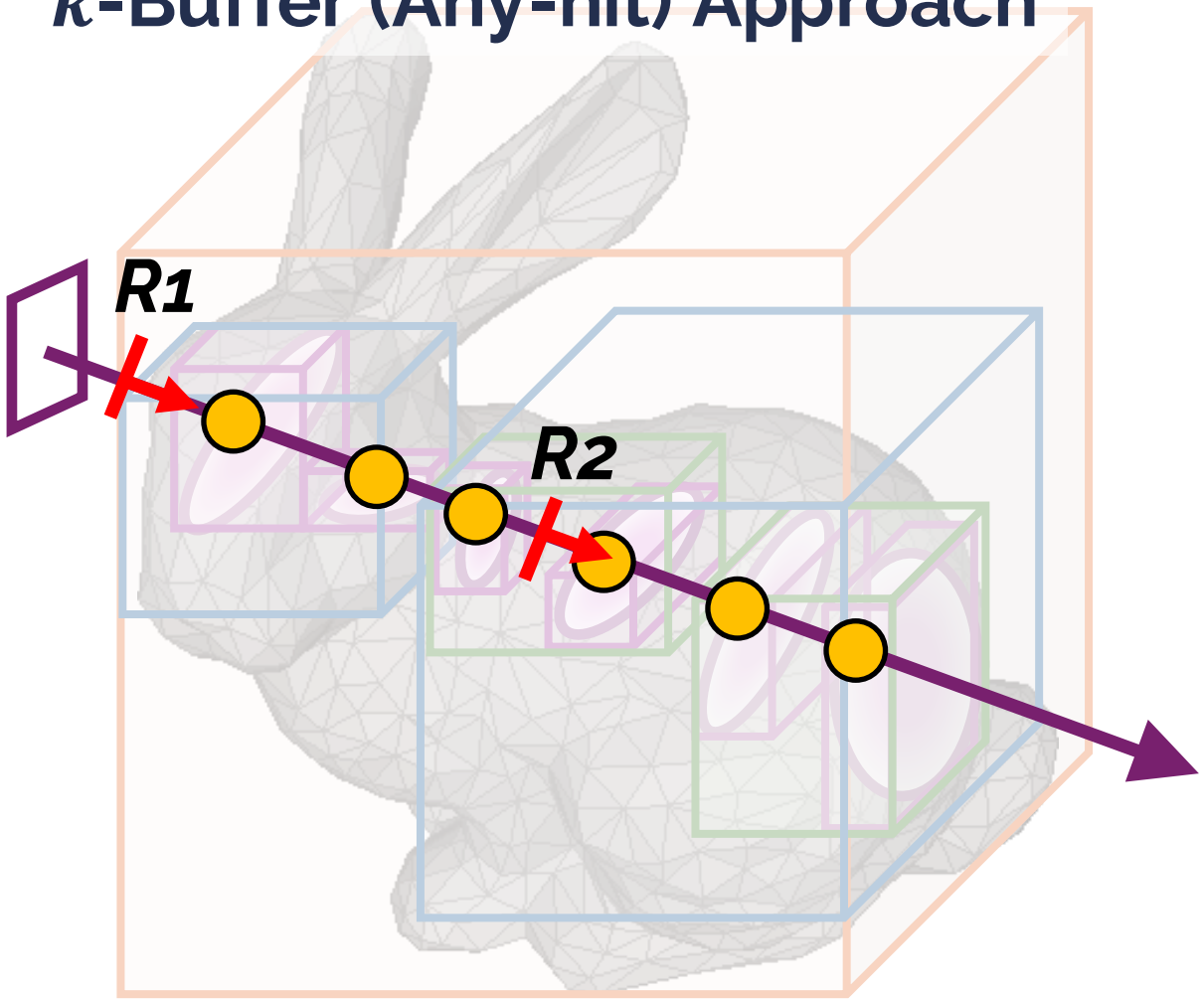
Front-to-Back Order

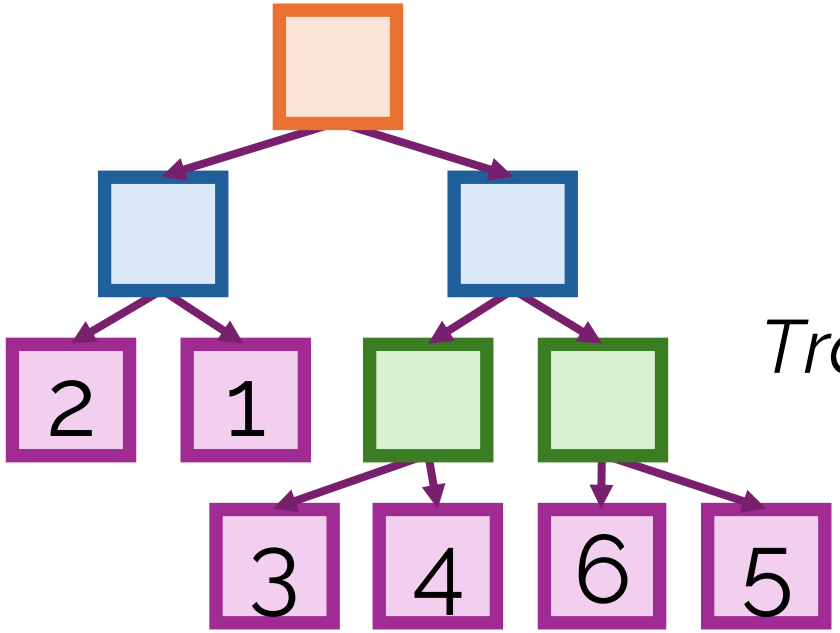
1 → 2 → 3 → 4 → 5 → 6



Gaussian RT Optimizations & Limitations

k-Buffer (Any-hit) Approach





BVH Traversal

Front-to-Back Order

1 → 2 → 3 → 4 → 5 → 6

R1

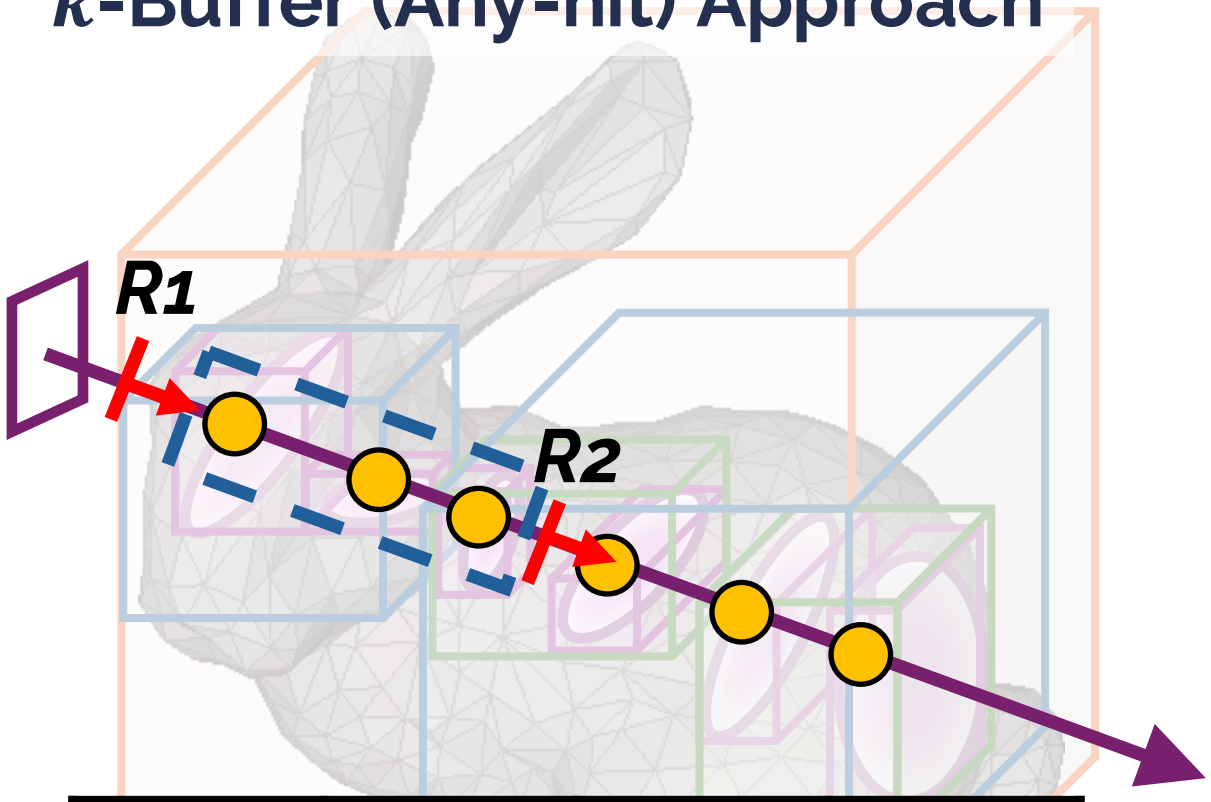
1 → 2 → 3

R2

4 → 5 → 6

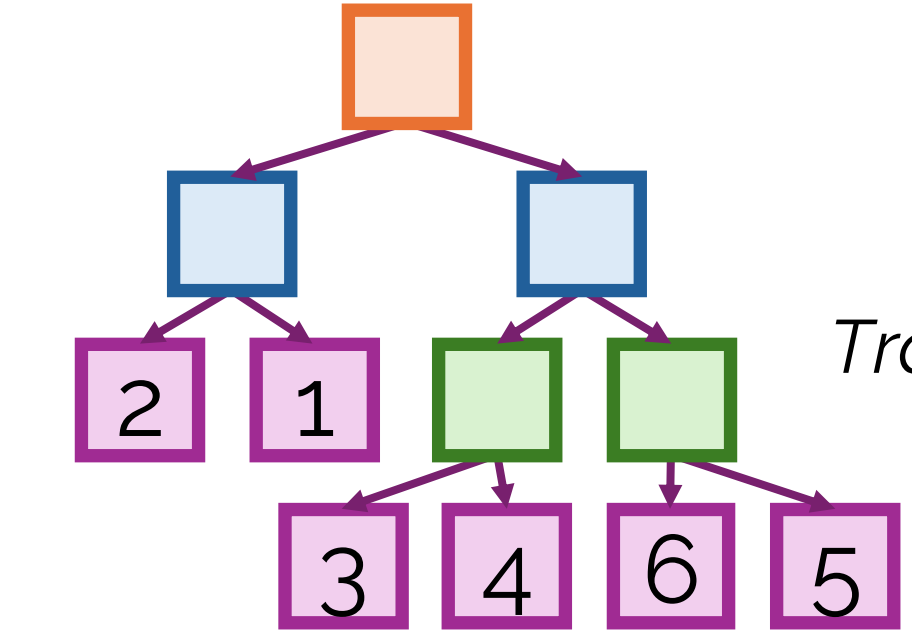
Gaussian RT Optimizations & Limitations

k-Buffer (Any-hit) Approach



Ray 1	1	2	3
Gaus. ID			
Distance			

k-buffer ($k = 3$)



BVH Traversal

Front-to-Back Order

1 → 2 → 3 → 4 → 5 → 6

R1

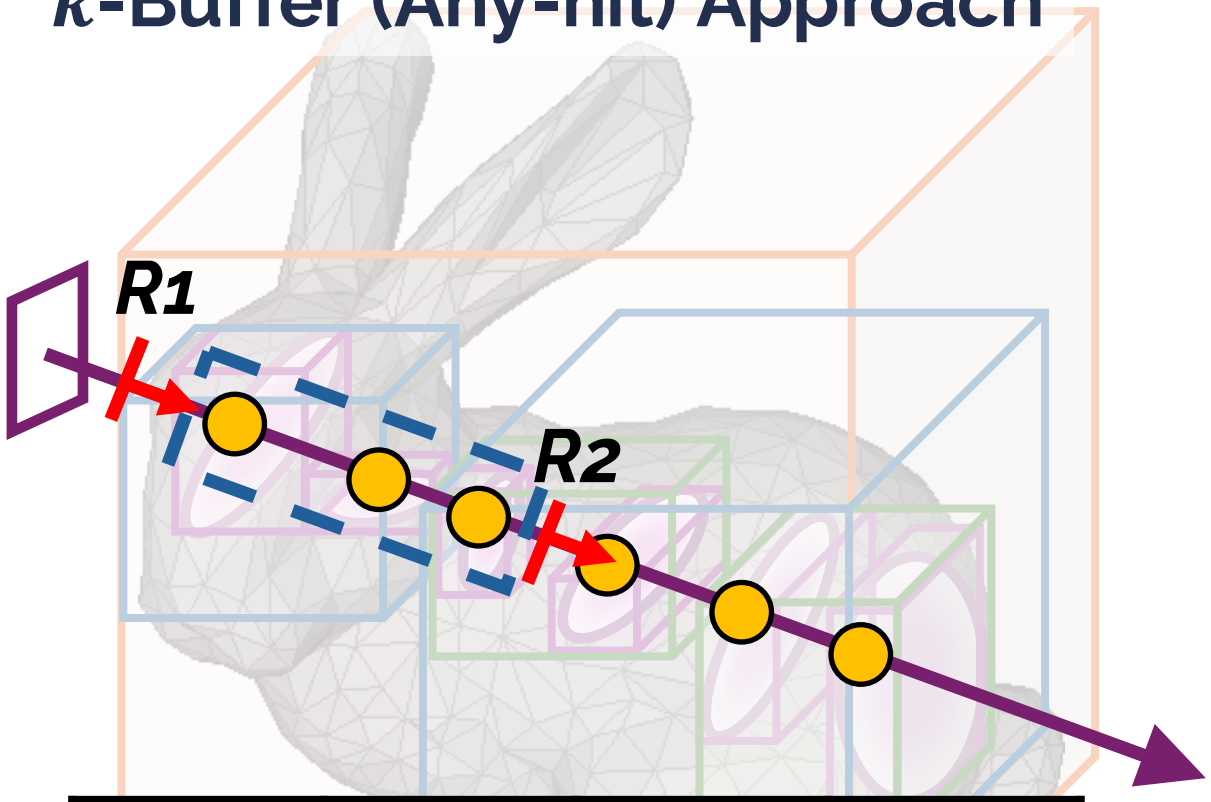
1 → 2 → 3

R2

4 → 5 → 6

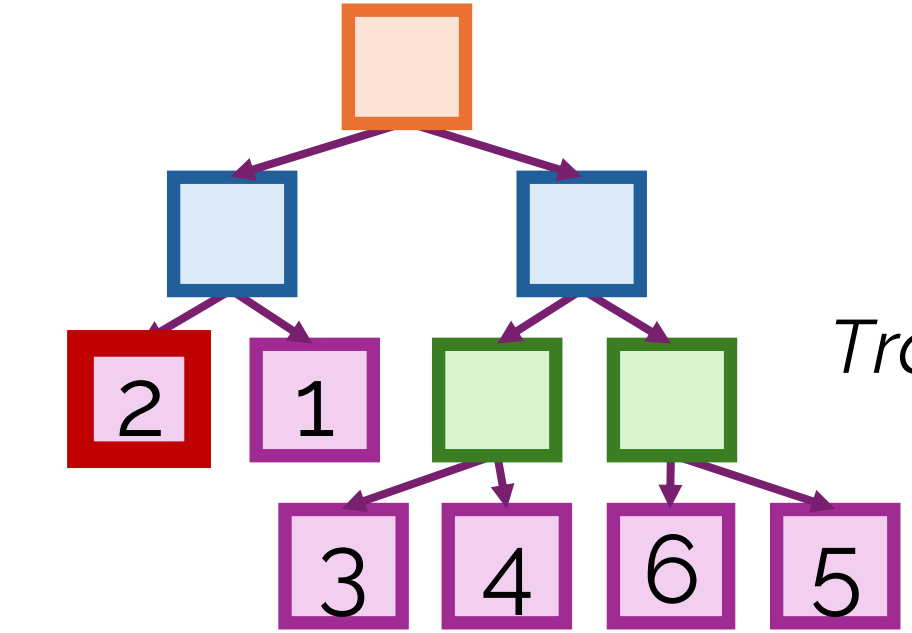
Gaussian RT Optimizations & Limitations

k-Buffer (Any-hit) Approach



Ray 1	1	2	3
Gaus. ID			
Distance			

k-buffer ($k = 3$)



BVH Traversal

Front-to-Back Order

1 → 2 → 3 → 4 → 5 → 6

R1

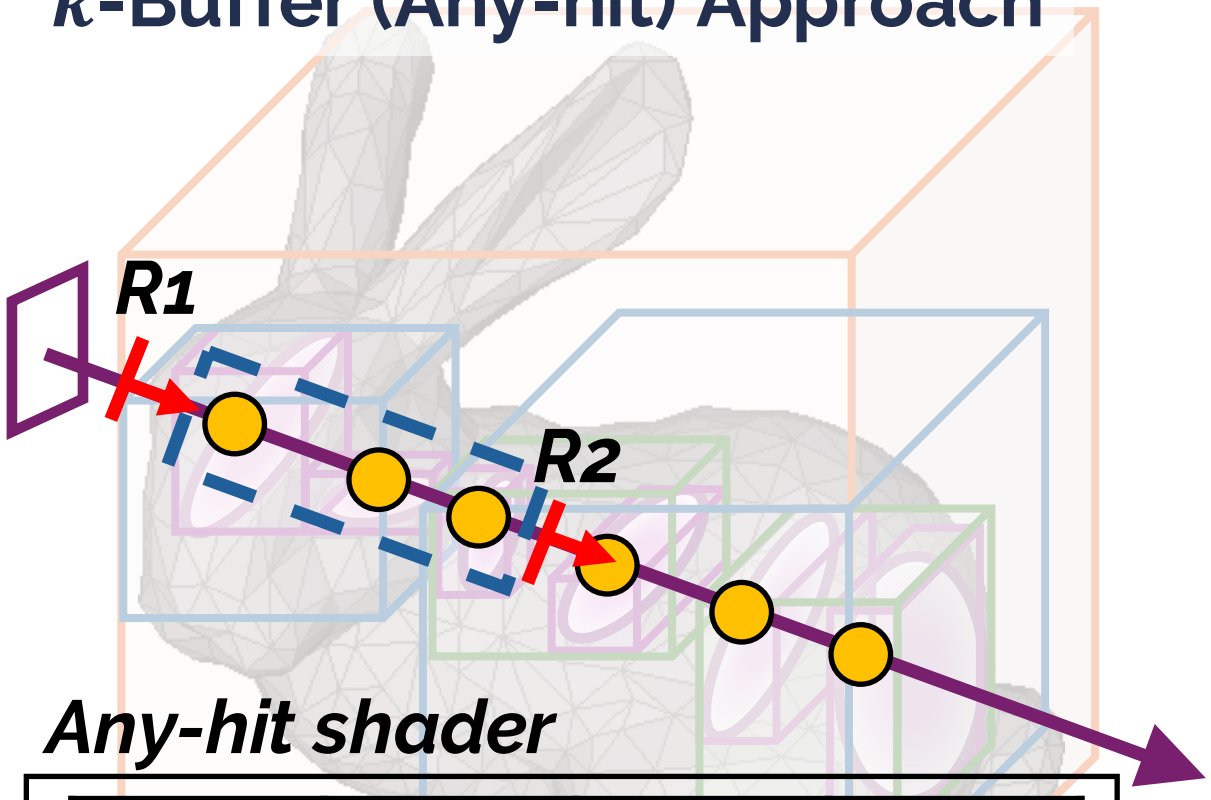
1 → 2 → 3

R2

4 → 5 → 6

Gaussian RT Optimizations & Limitations

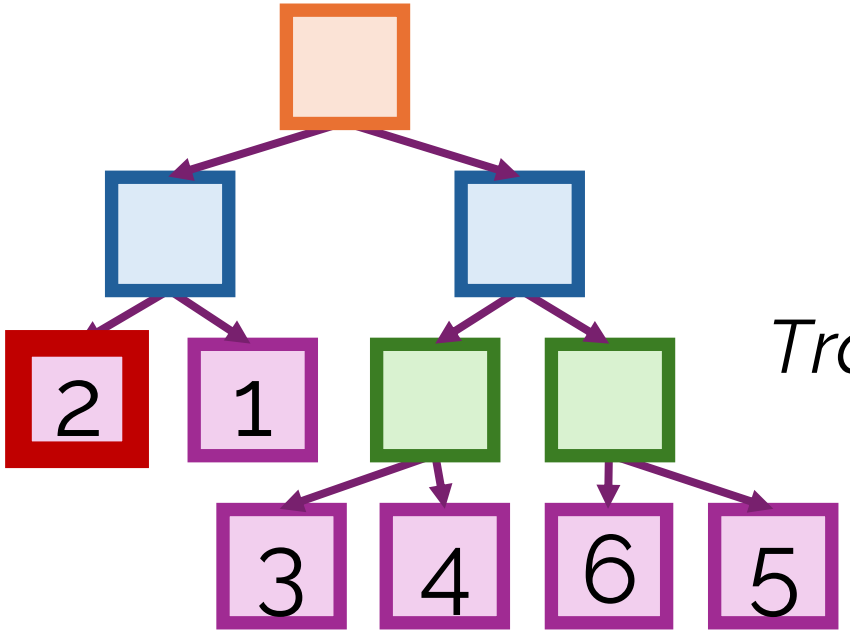
k-Buffer (Any-hit) Approach



Any-hit shader

Ray 1	1	2	3
Gaus. ID			
Distance			

k-buffer ($k = 3$)



BVH Traversal

Front-to-Back Order

1 → 2 → 3 → 4 → 5 → 6

R1

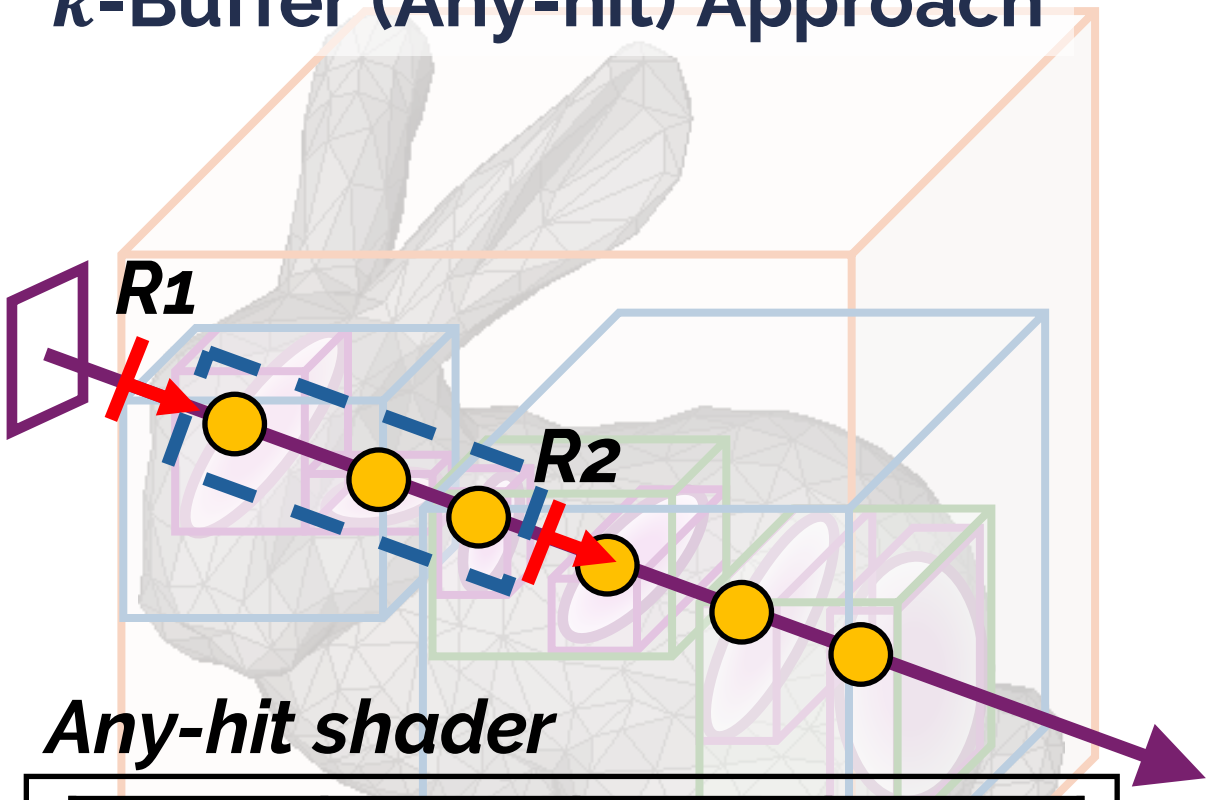
1 → 2 → 3

R2

4 → 5 → 6

Gaussian RT Optimizations & Limitations

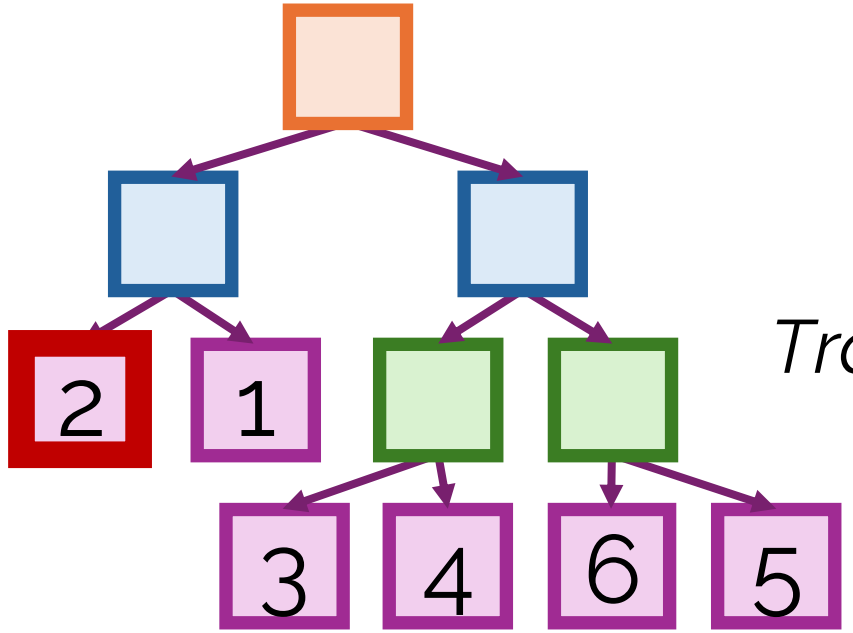
k-Buffer (Any-hit) Approach



Any-hit shader

Ray 1	1	2	3
Gaus. ID	2		
Distance	3.7		

k-buffer ($k = 3$)



BVH Traversal

Front-to-Back Order

1 → 2 → 3 → 4 → 5 → 6

R1

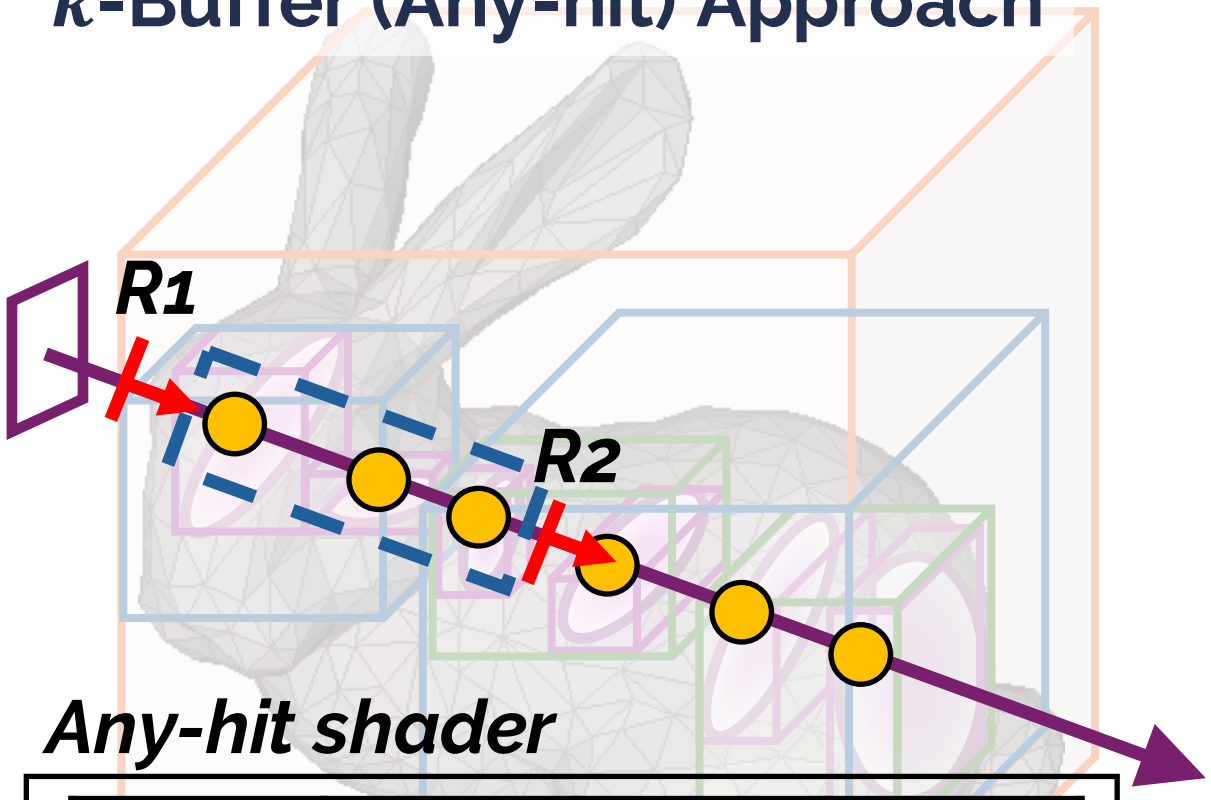
1 → 2 → 3

R2

4 → 5 → 6

Gaussian RT Optimizations & Limitations

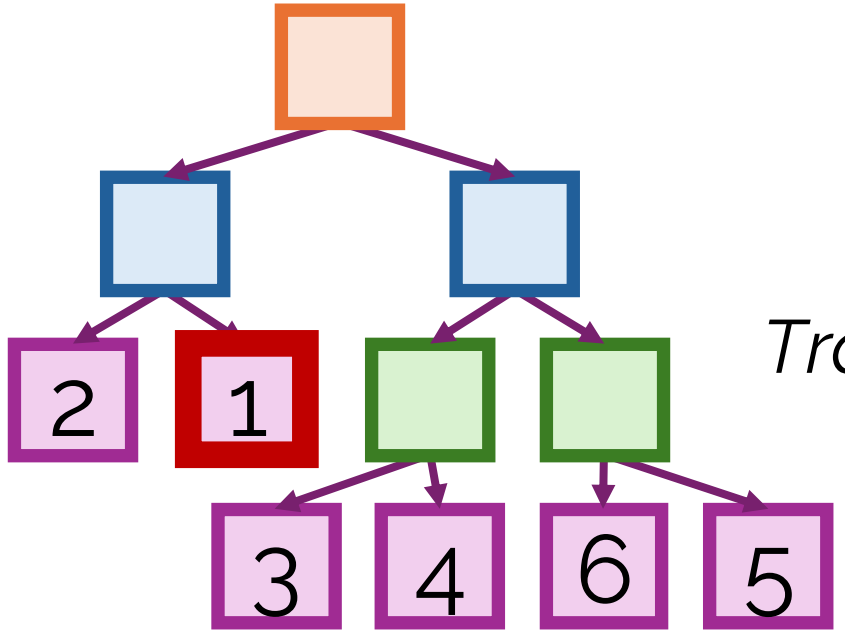
k-Buffer (Any-hit) Approach



Any-hit shader

Ray 1	1	2	3
Gaus. ID	**1**	2	
Distance	**1.4**	3.7	

k-buffer ($k = 3$)



BVH Traversal

Front-to-Back Order

1 → 2 → 3 → 4 → 5 → 6

R1

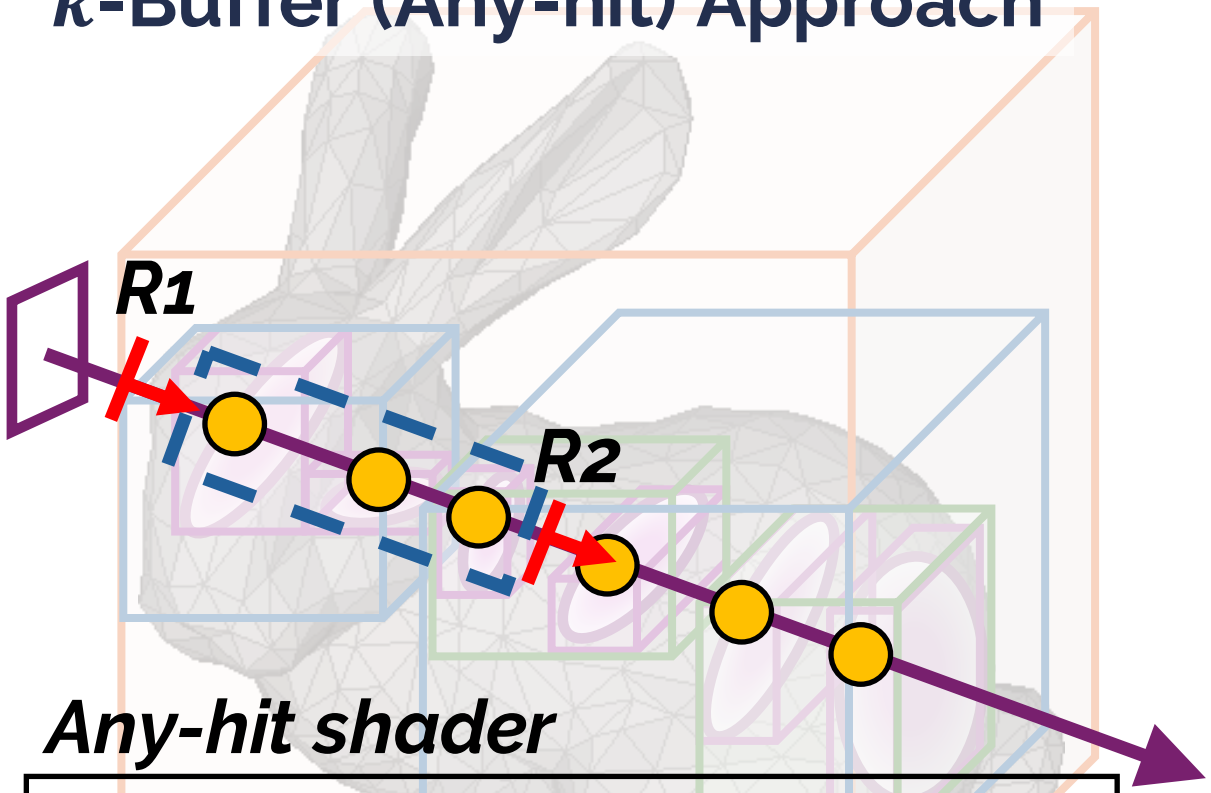
1 → 2 → 3

R2

4 → 5 → 6

Gaussian RT Optimizations & Limitations

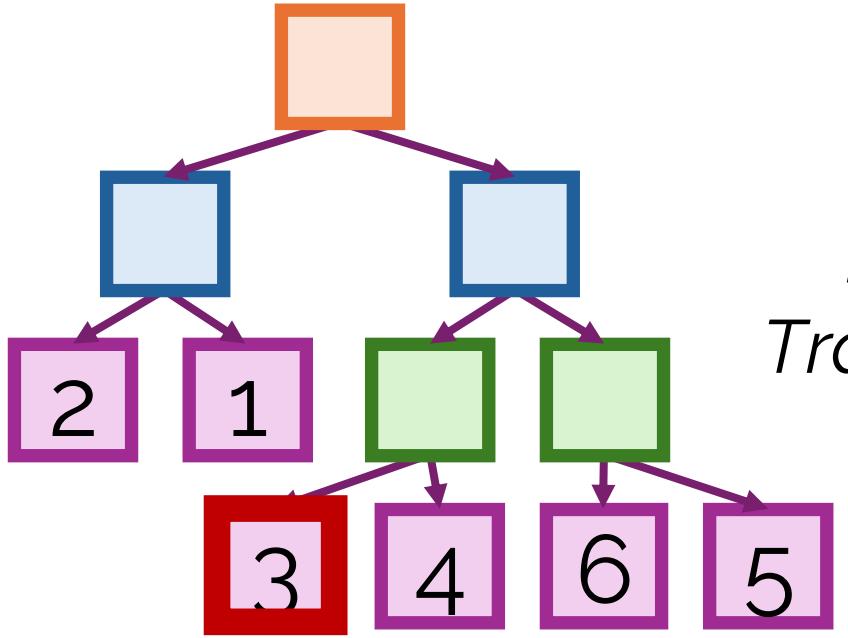
k-Buffer (Any-hit) Approach



Any-hit shader

Ray 1	1	2	3
Gaus. ID	1	2	**3**
Distance	1.4	3.7	**5.2**

k-buffer ($k = 3$)



BVH Traversal

Front-to-Back Order

1 → 2 → 3 → 4 → 5 → 6

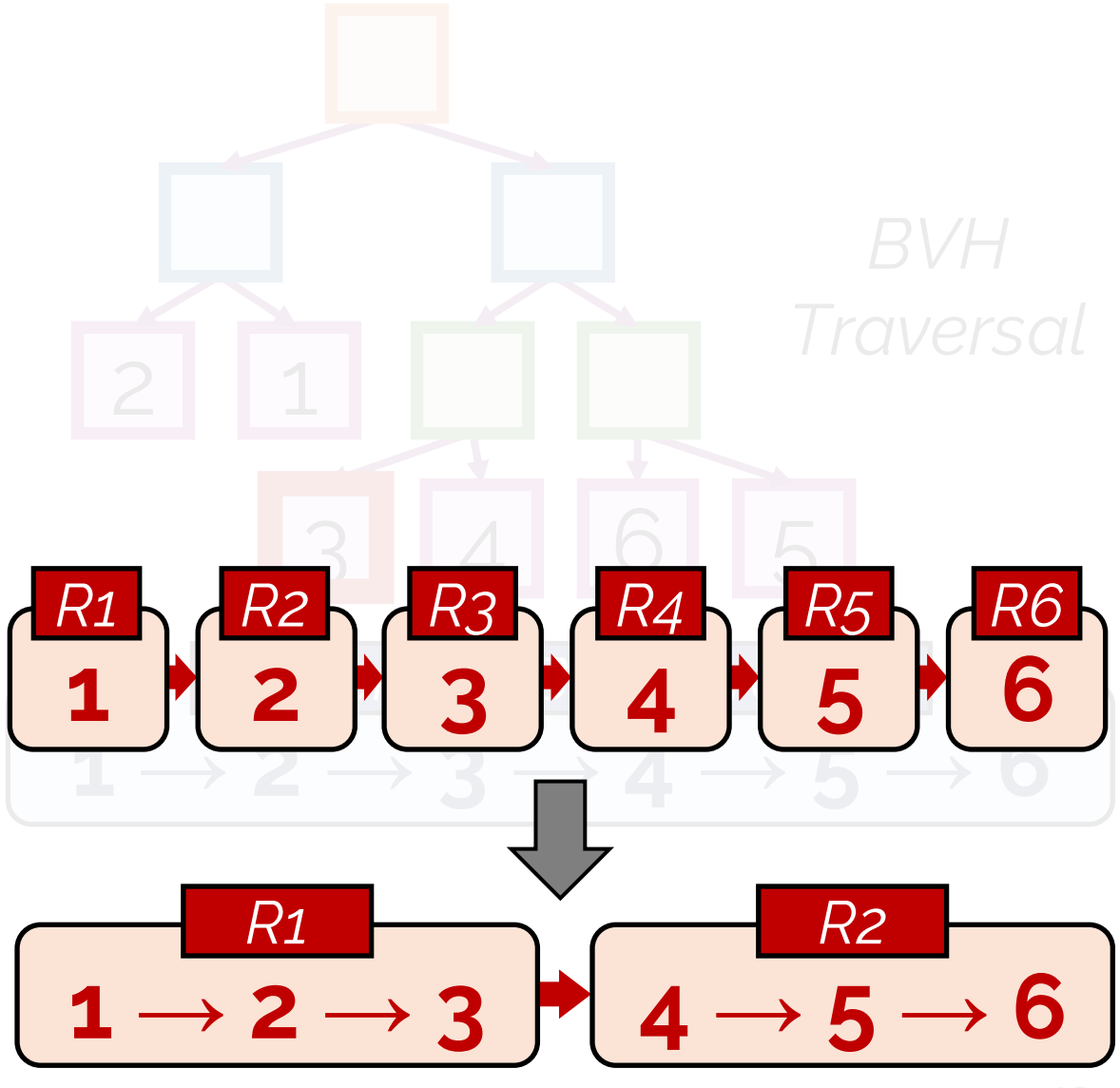
R1

1 → 2 → 3

R2

4 → 5 → 6

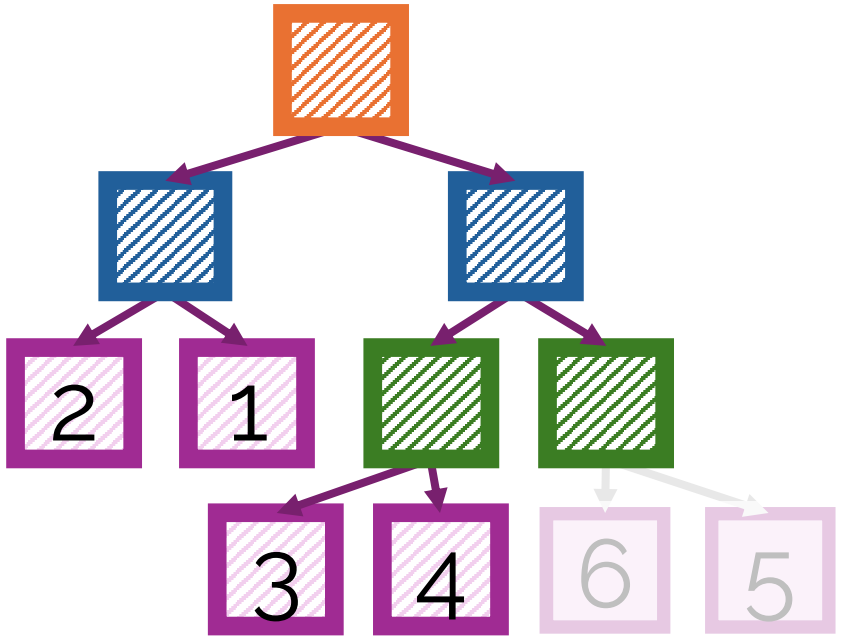
Gaussian RT Optimizations & Limitations



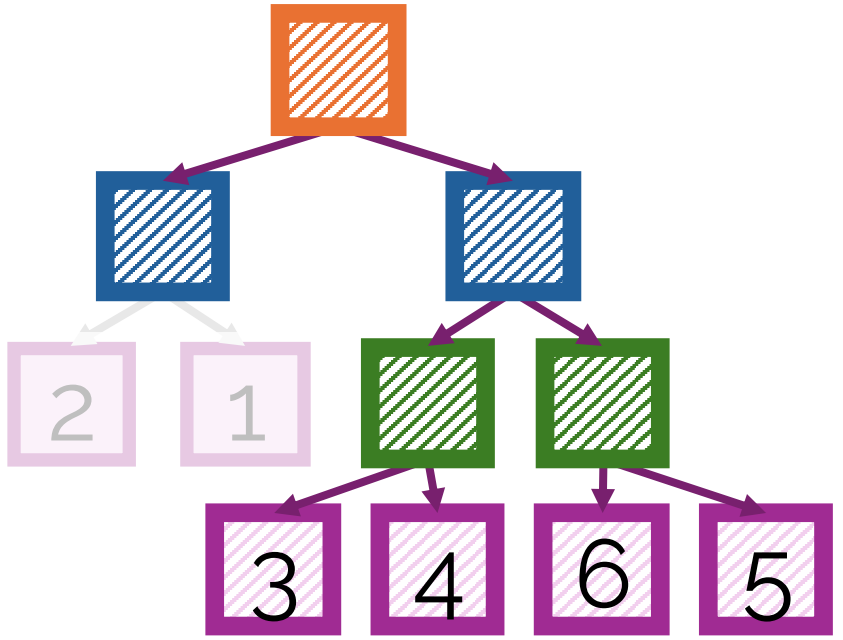
Gaussian RT Optimizations & Limitations

k-Buffer (Any-hit) Approach

Round 1



Round 2

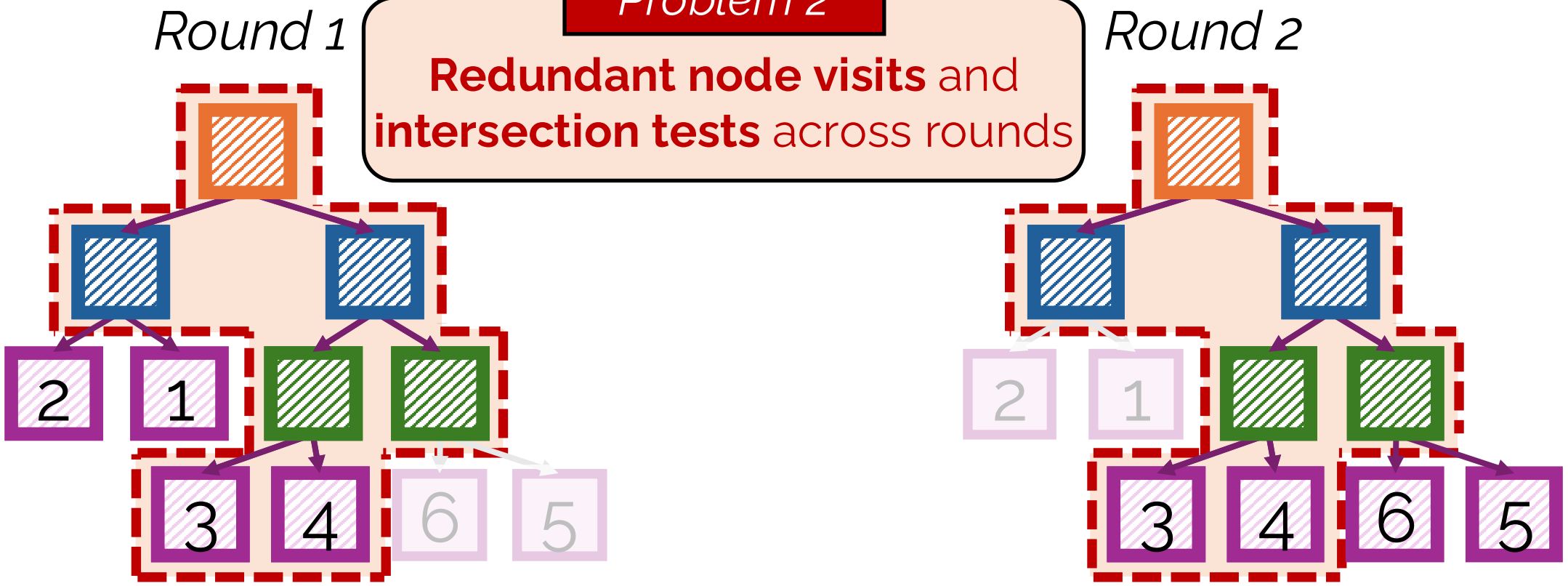


Gaussian RT Optimizations & Limitations

k-Buffer (Any-hit) Approach

Problem 2

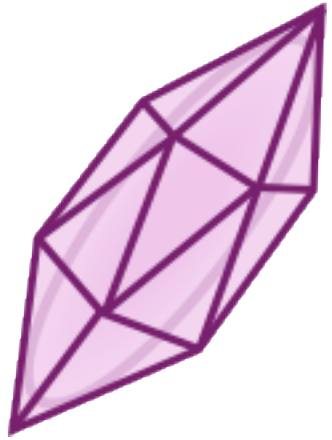
Redundant node visits and **intersection tests** across rounds



Gaussian RT Optimizations & Limitations

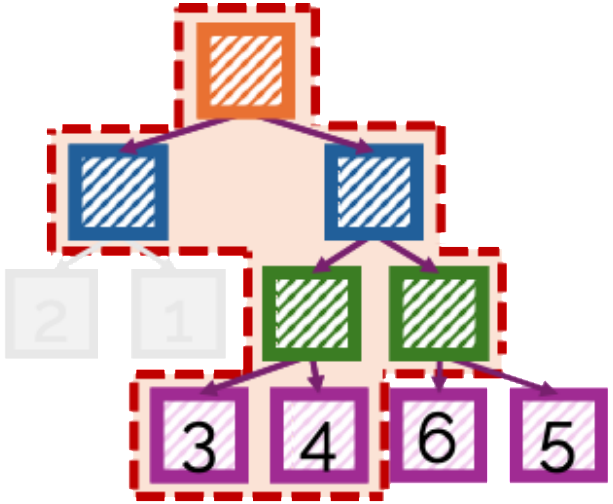
Problem 1

Bloated BVH size and
increased **memory footprint**



Problem 2

Redundant node visit and
intersection tests across rounds



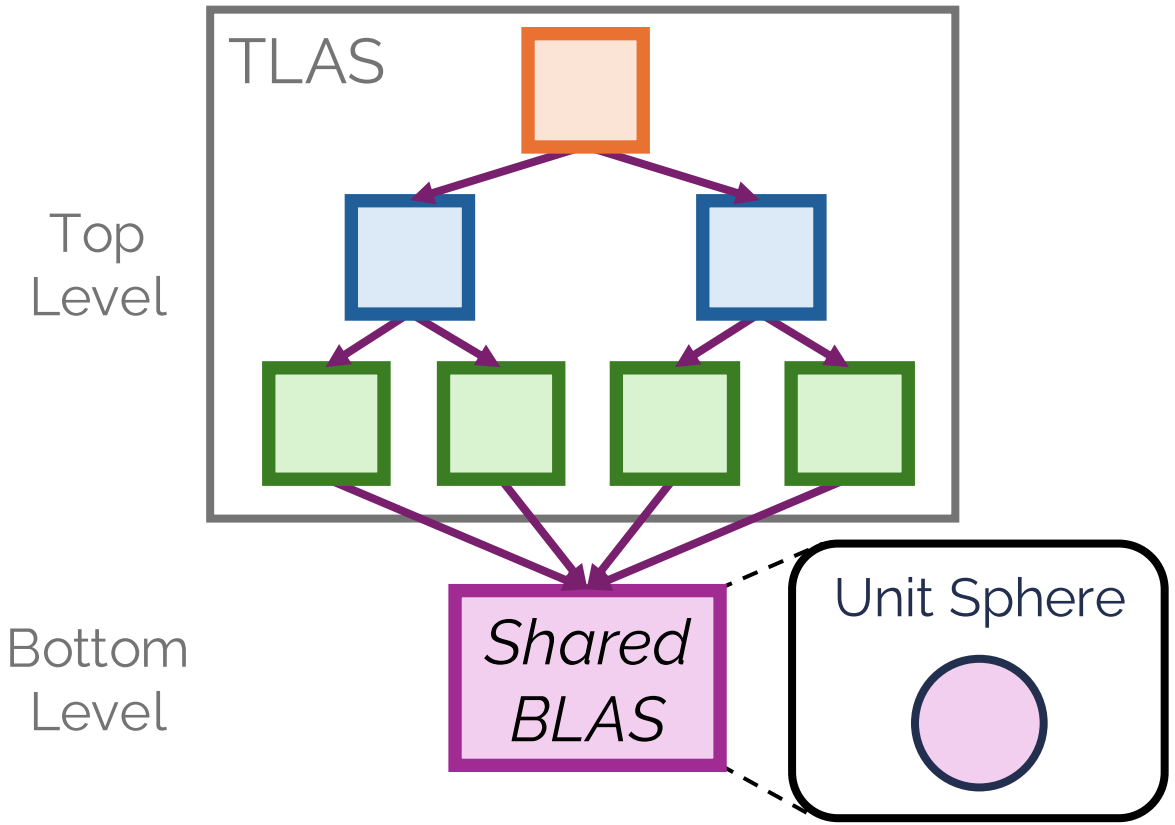
Outline

- **Background**
  - 3D Gaussian-based Rendering: Rasterization vs. Ray-tracing
  - Ray Tracing Accelerators in Modern GPUs
- **Gaussian RT Optimizations & Limitations**
- **GRTX: SW-HW Optimizations for Gaussian Ray Tracing**
  - GRTX-SW: Two-Level Acceleration Structure for Gaussian Primitives
  - GRTX-HW: HW Extension for Traversal Checkpointing and Replay
- **Evaluation**
- **Conclusion**

GRTX Overview

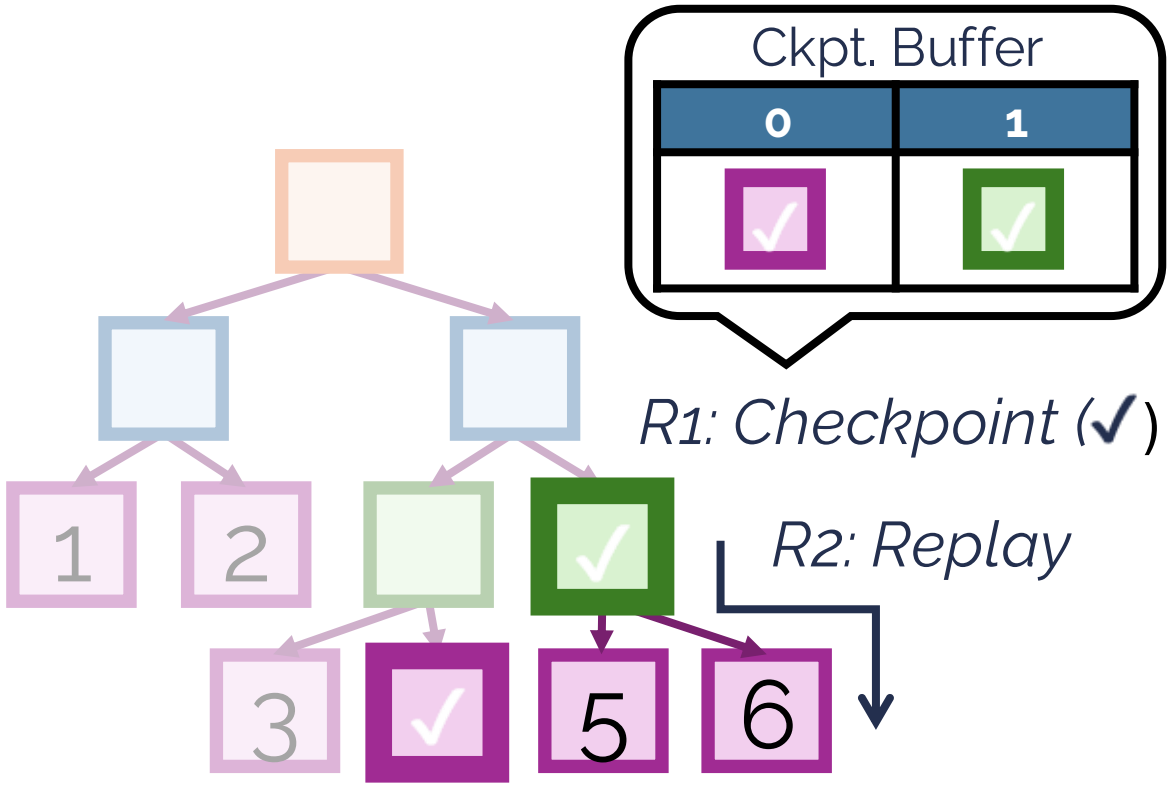
GRTX-SW

Two-level BVH with
a **single shared BLAS** for Gaussian



GRTX-HW

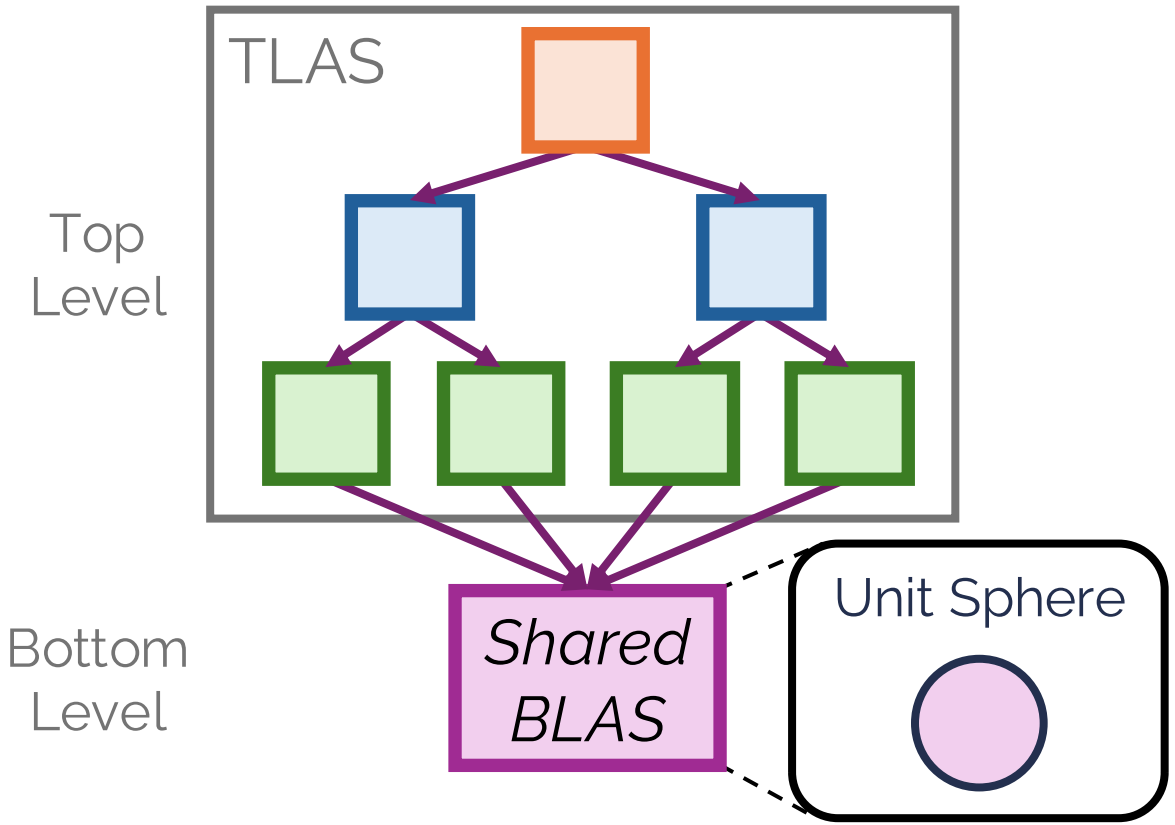
Hardware extension for
traversal checkpointing & replay



GRTX Overview

GRTX-SW

Two-level BVH with
a **single shared BLAS** for Gaussian



GRTX-HW

Hardware extension for
traversal checkpointing & replay

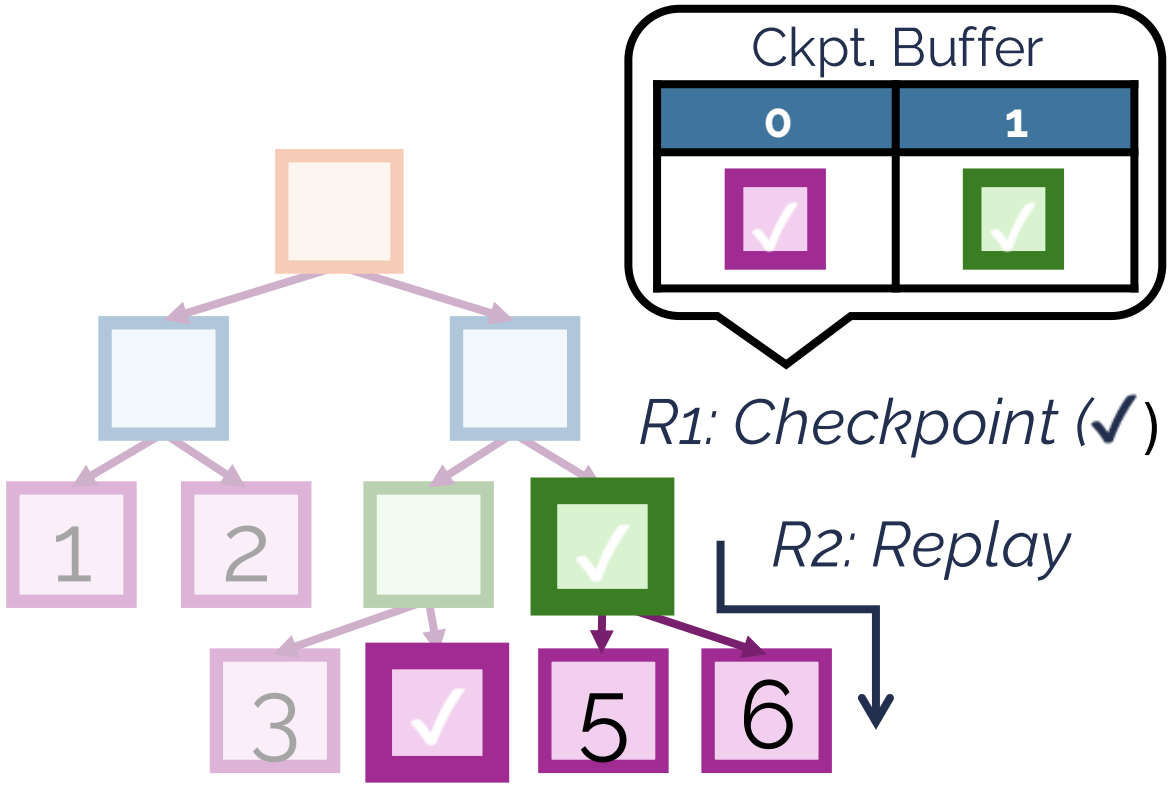
GRTX Overview

GRTX-SW

Two-level BVH with
a **single shared BLAS** for Gaussian

GRTX-HW

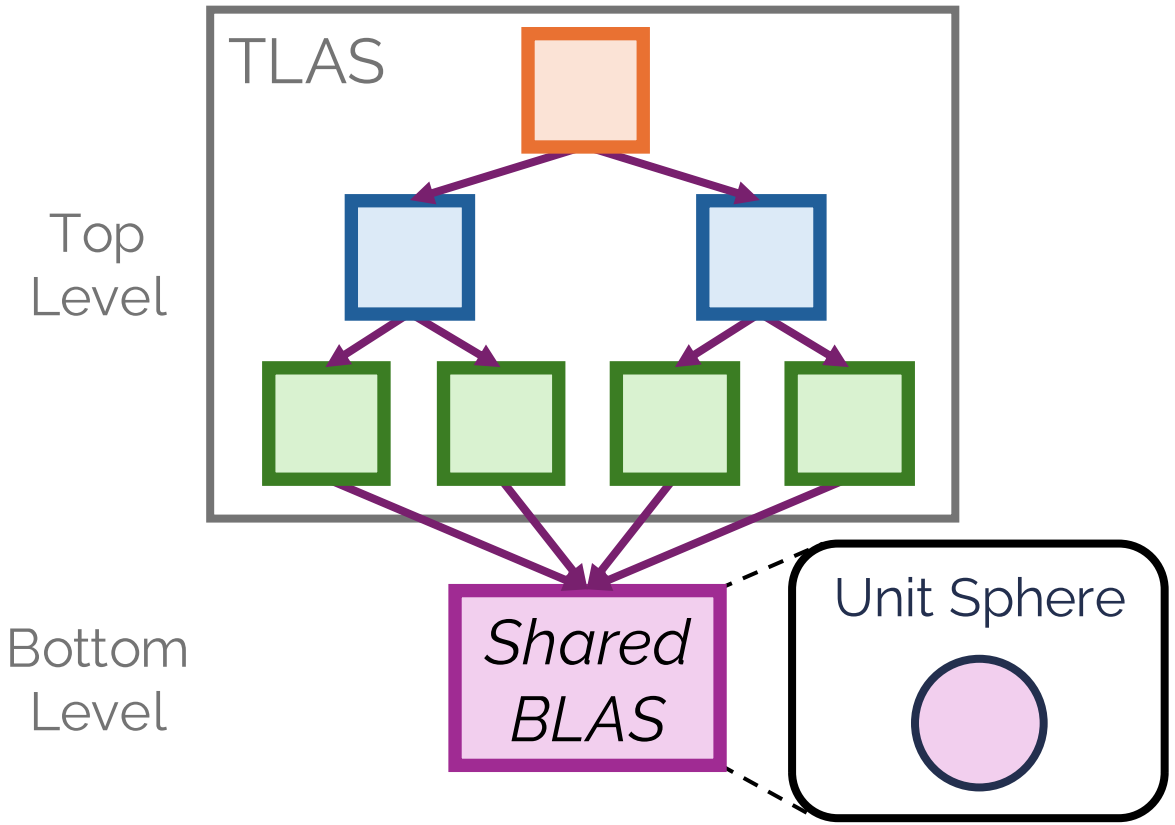
Hardware extension for
traversal checkpointing & replay



GRTX Overview

GRTX-SW

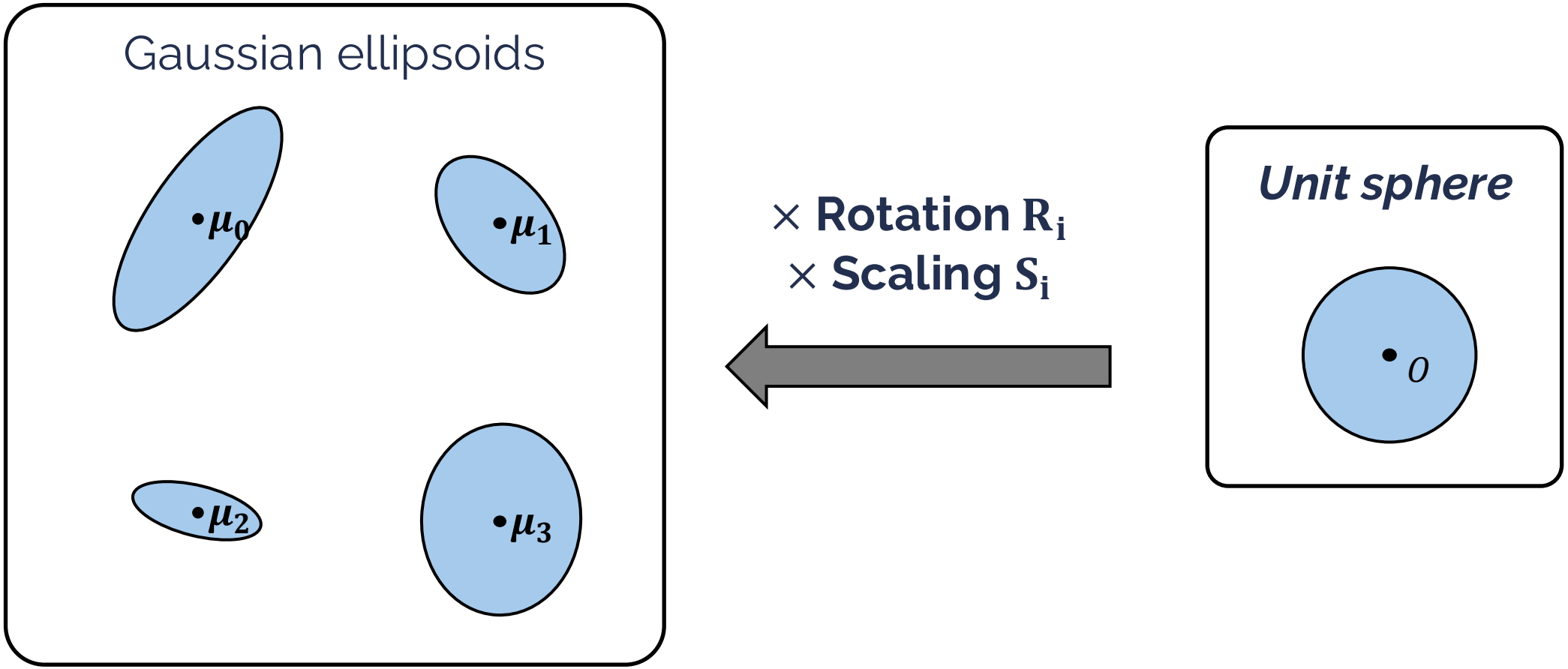
Two-level BVH with
a **single shared BLAS** for Gaussian



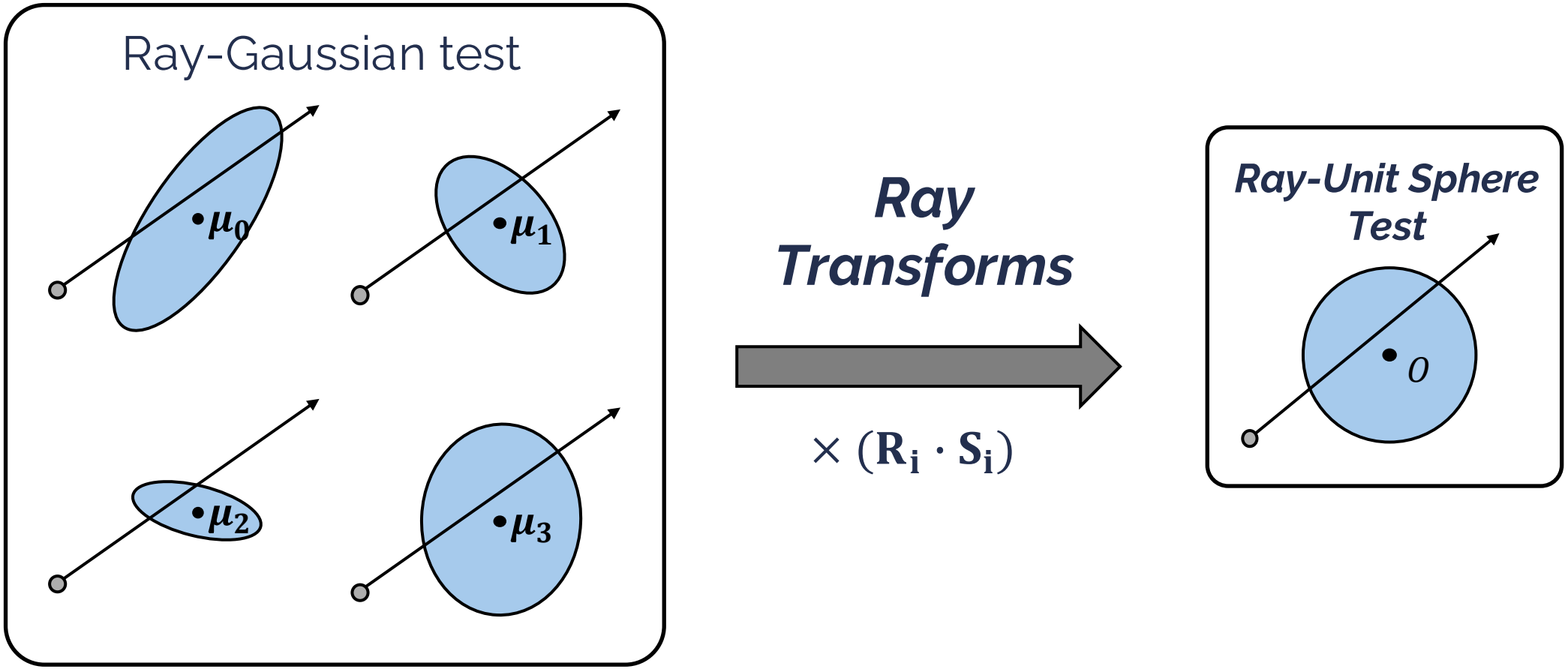
GRTX-HW

Hardware extension for
traversal checkpointing & replay

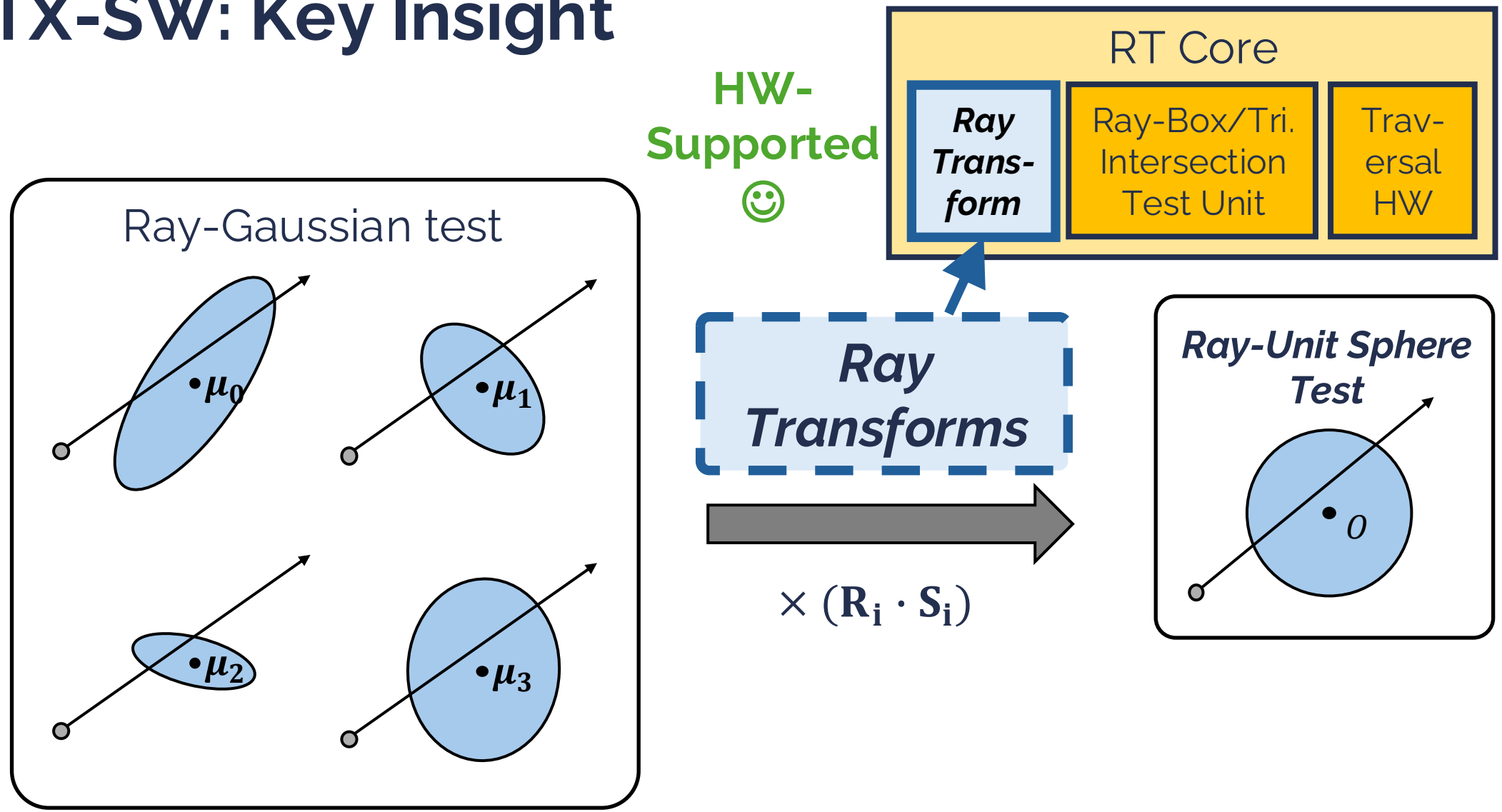
GRTX-SW: Key Insight



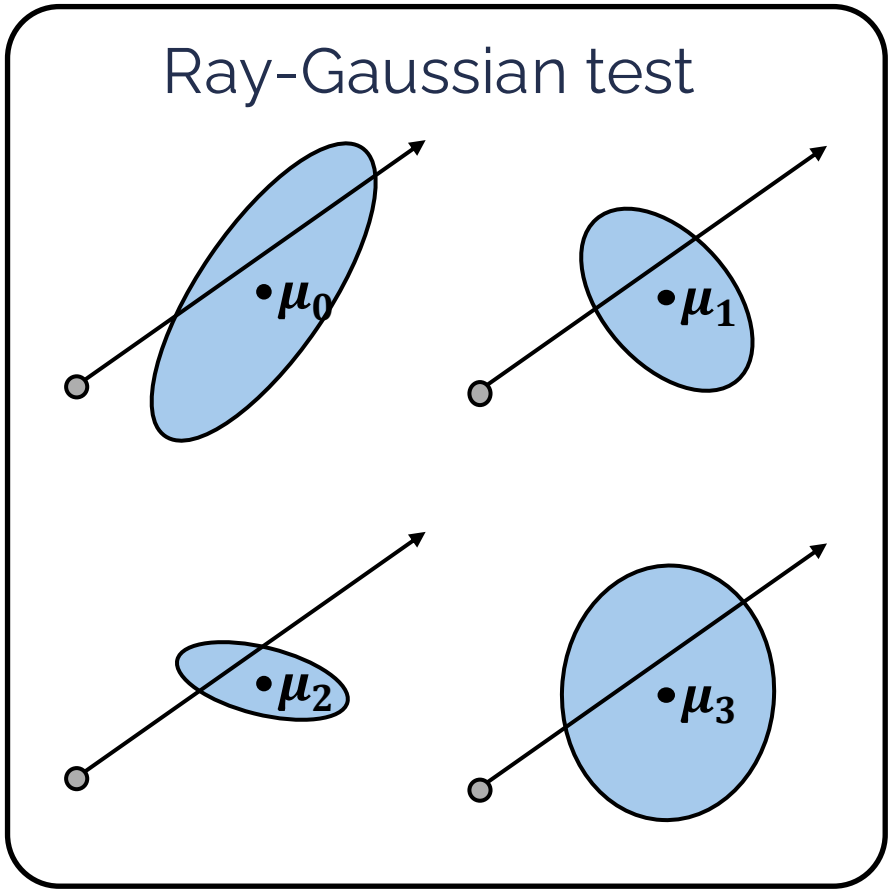
GRTX-SW: Key Insight



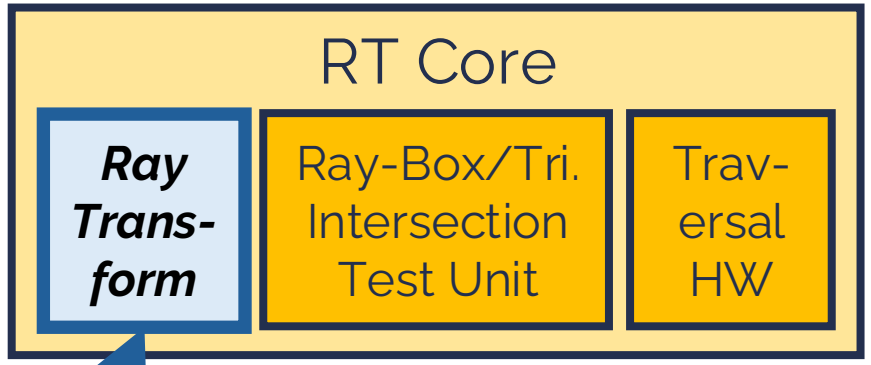
GRTX-SW: Key Insight

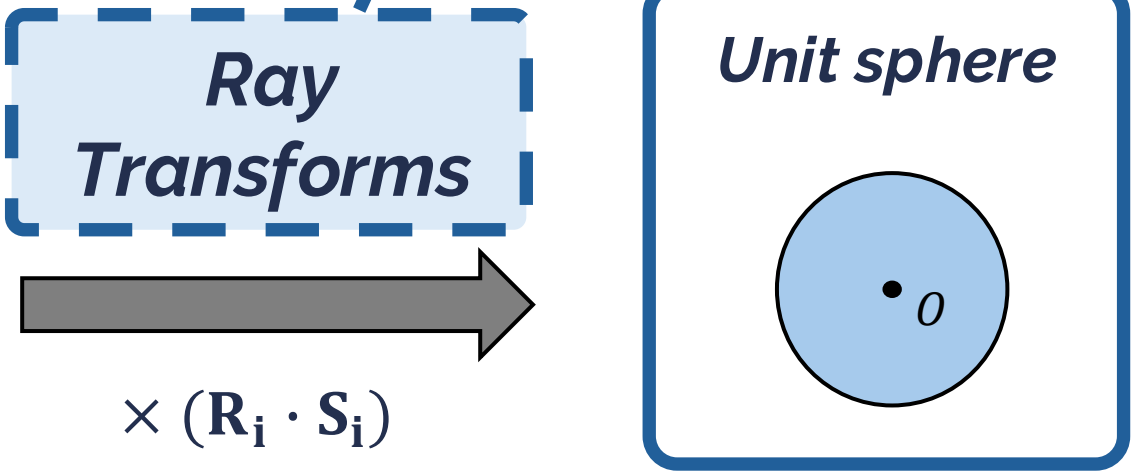


GRTX-SW: Key Insight



HW-Supported ☺



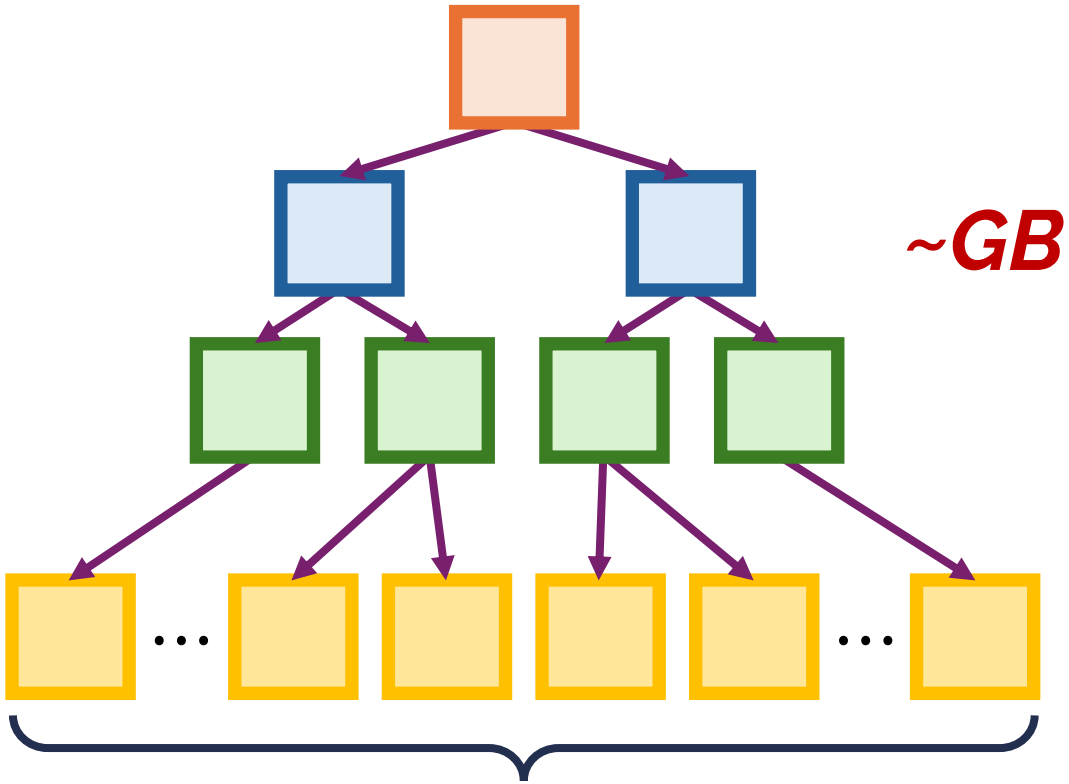


Idea

Use only **one shared BLAS** across all Gaussians

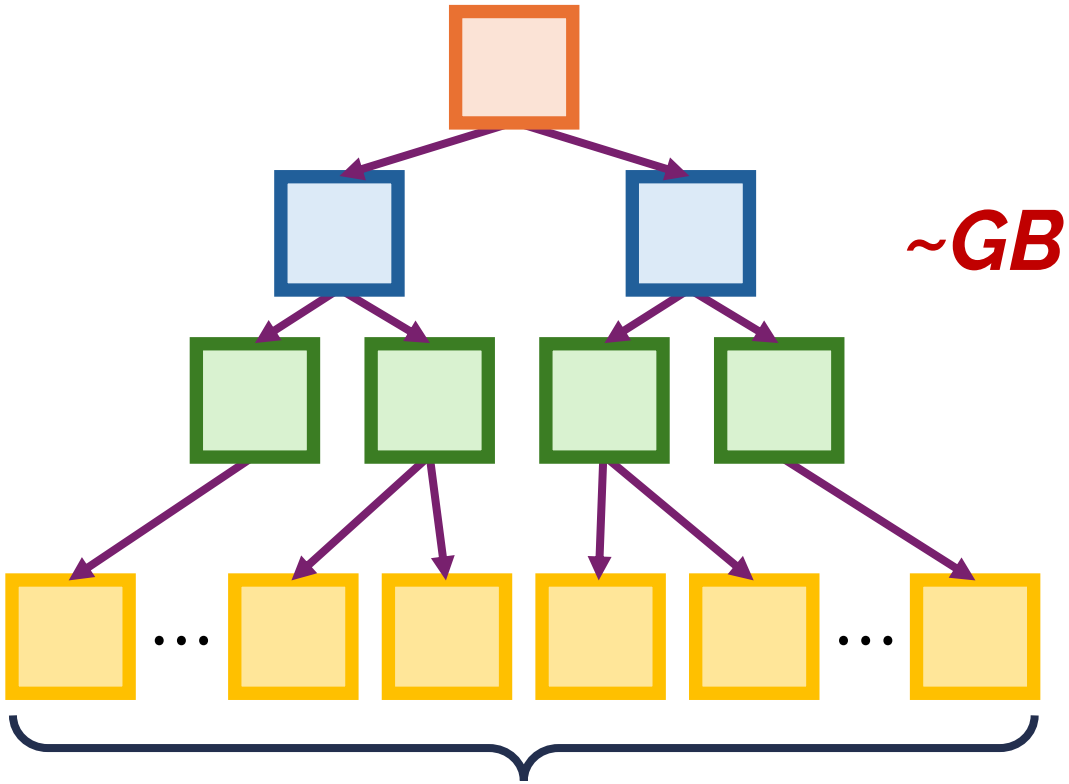
GRTX-SW: Two-Level BVH with a Shared BLAS

Prior Approaches

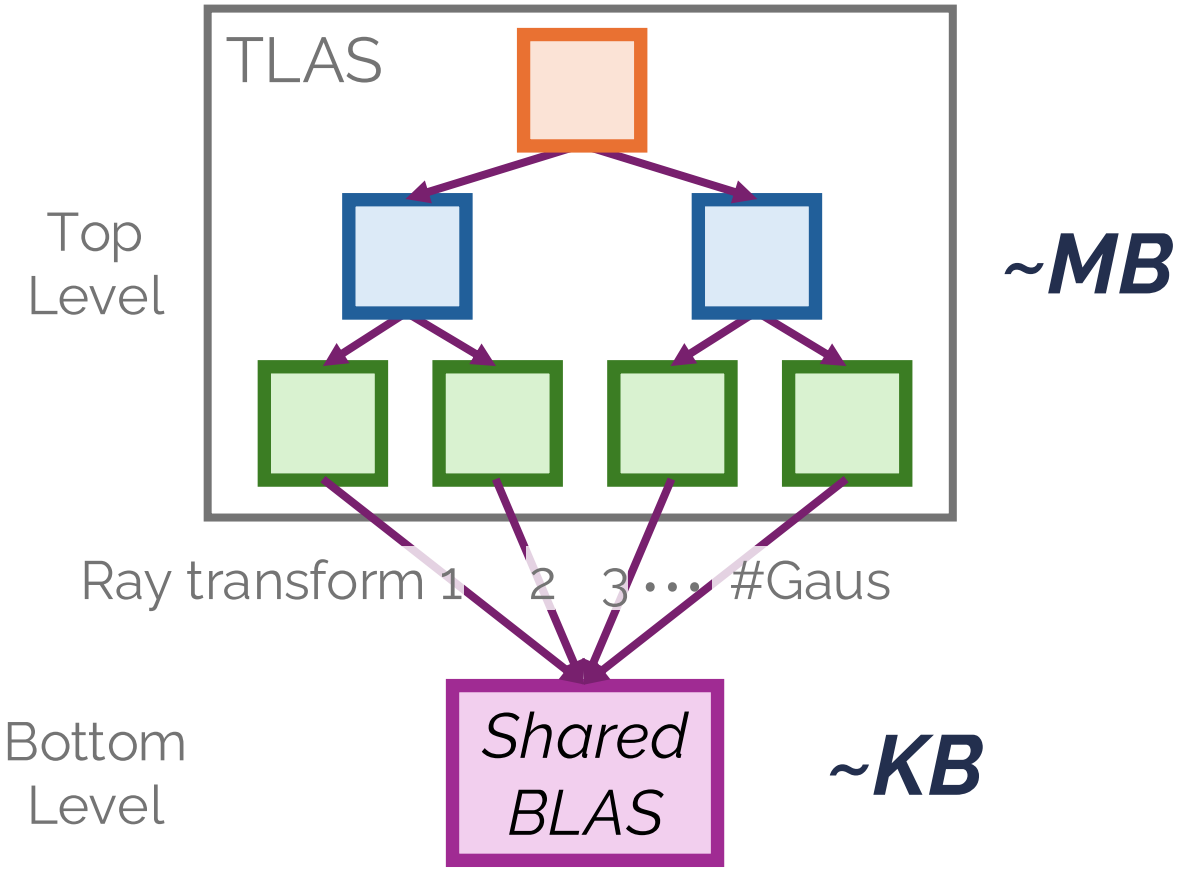


GRTX-SW: Two-Level BVH with a Shared BLAS

Prior Approaches

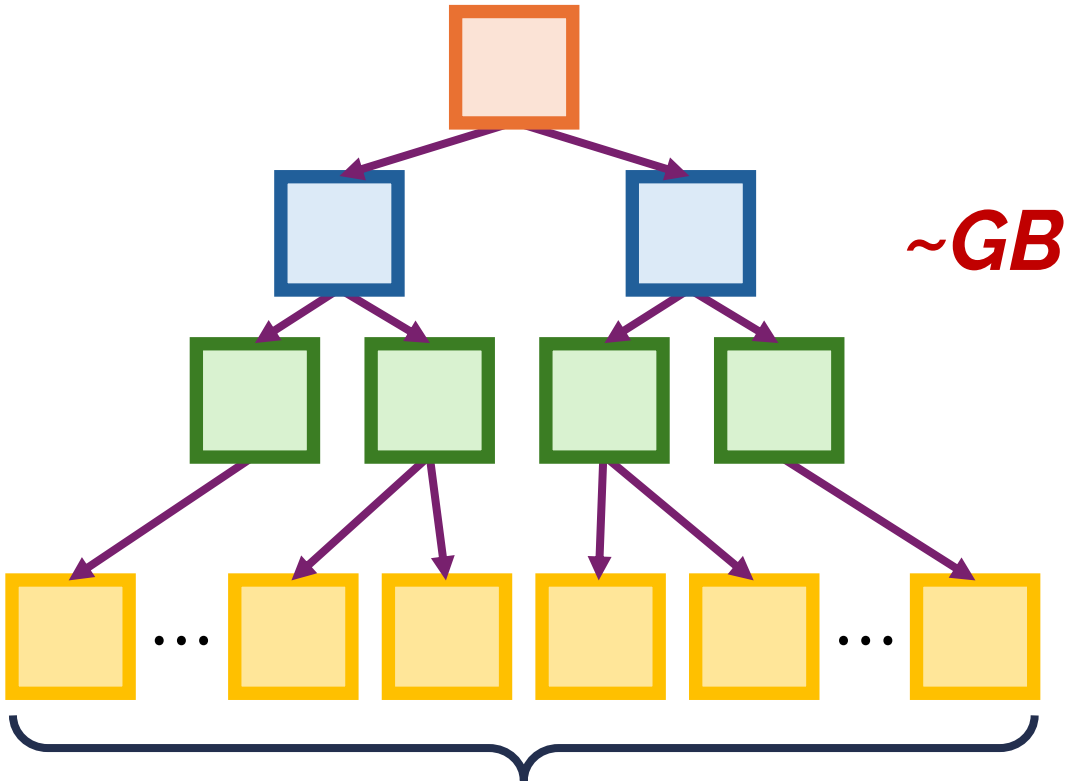


GRTX-SW

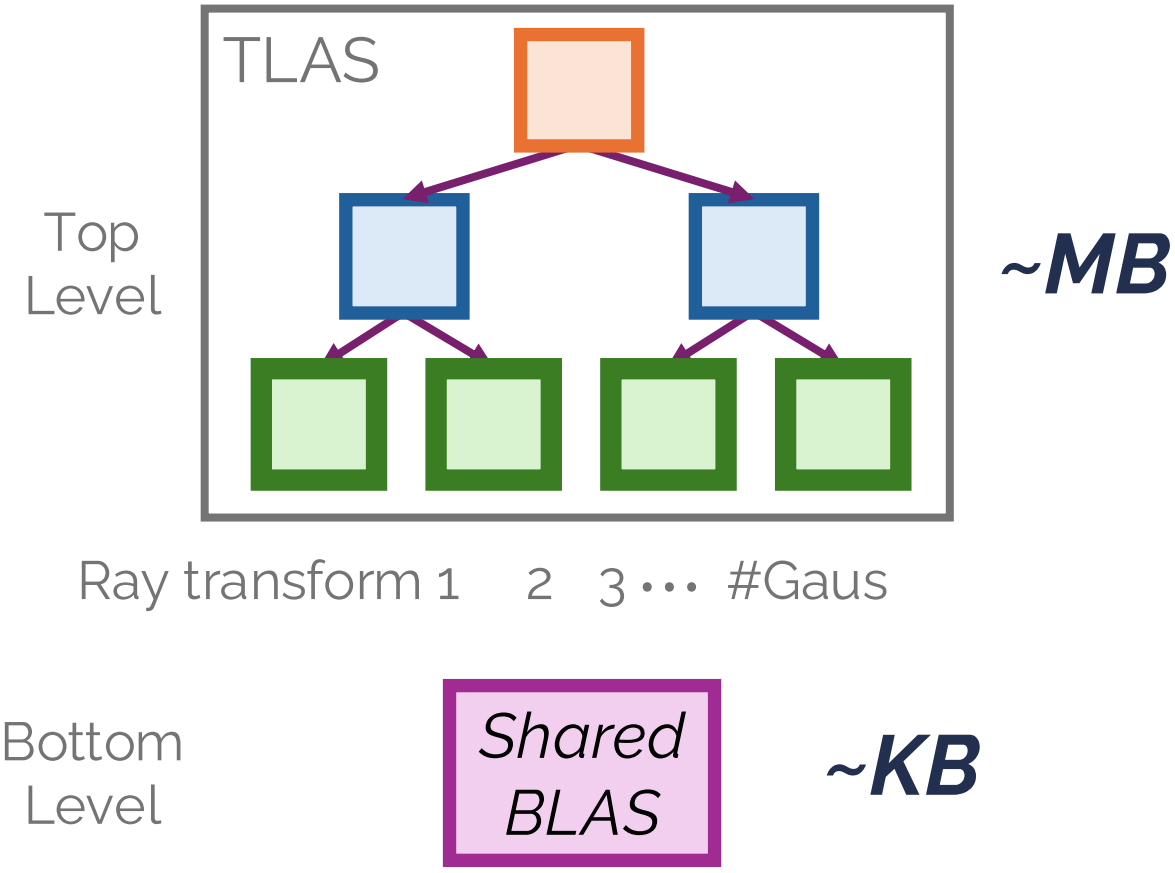


GRTX-SW: Two-Level BVH with a Shared BLAS

Prior Approaches

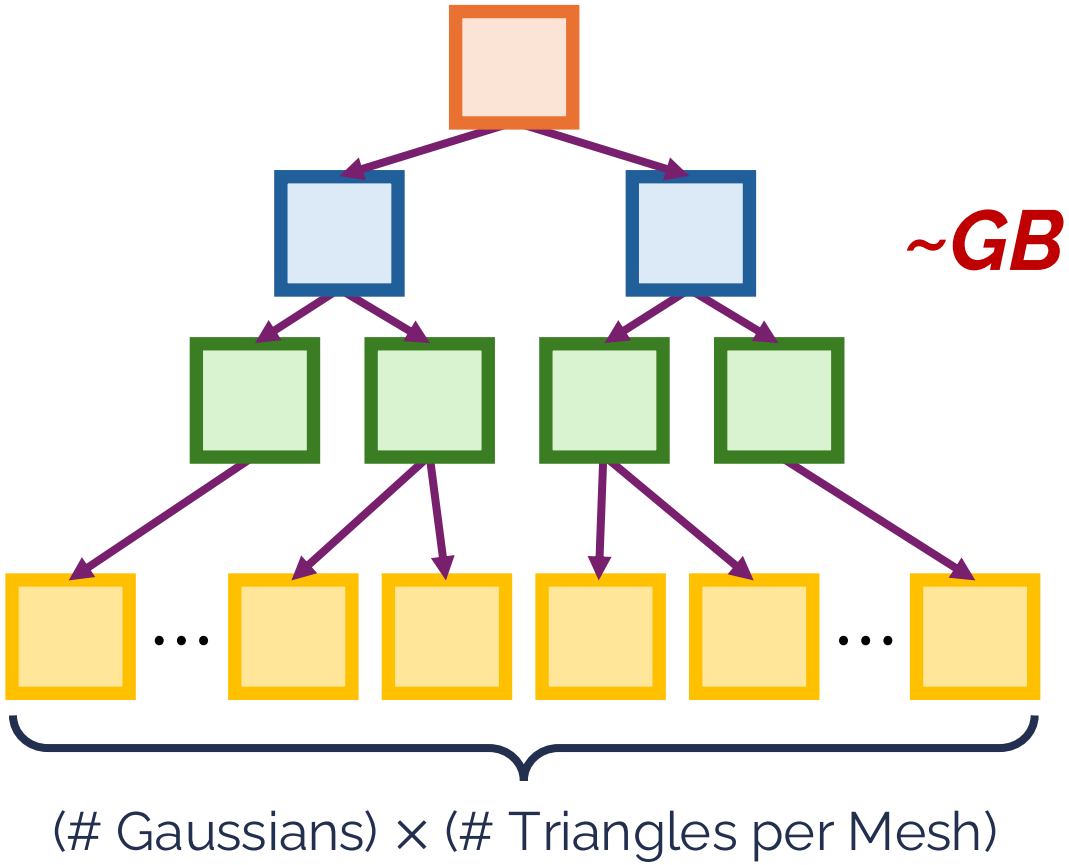


GRTX-SW

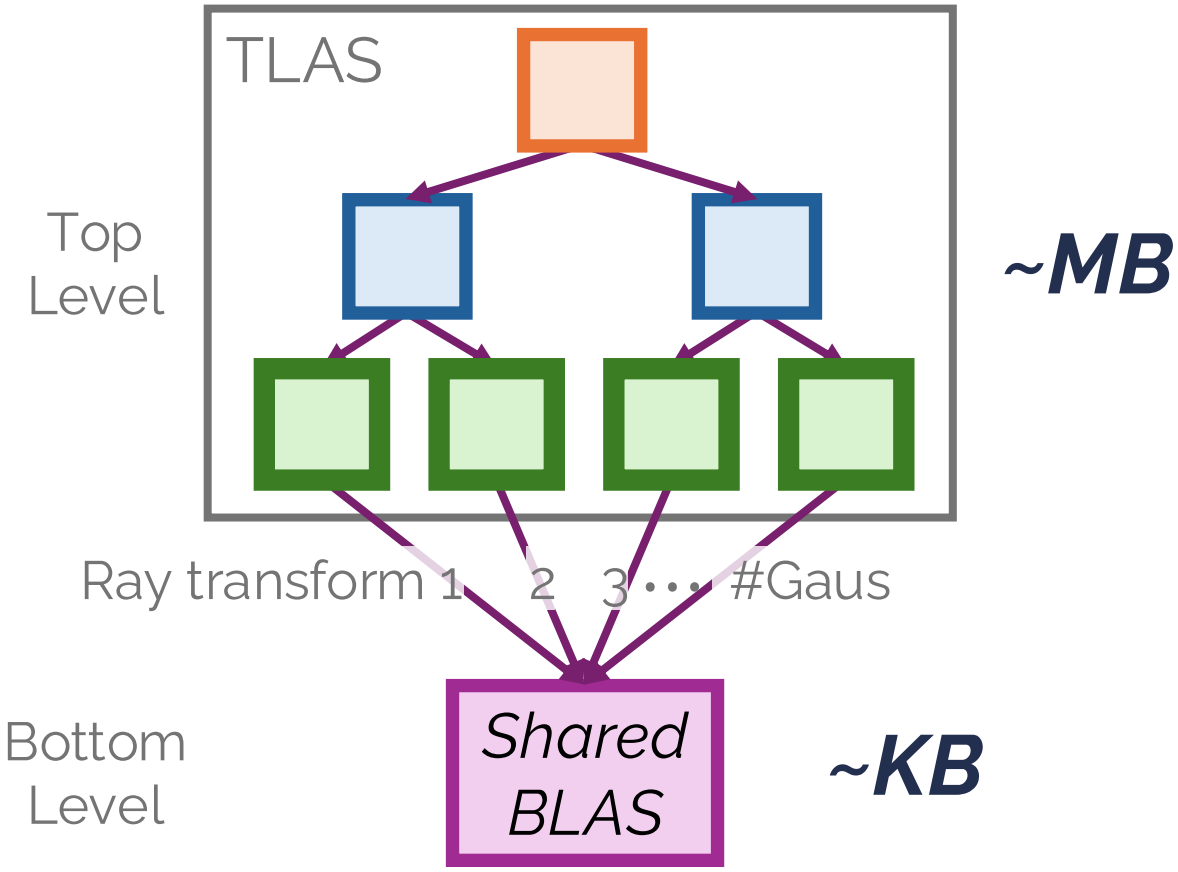


GRTX-SW: Two-Level BVH with a Shared BLAS

Prior Approaches



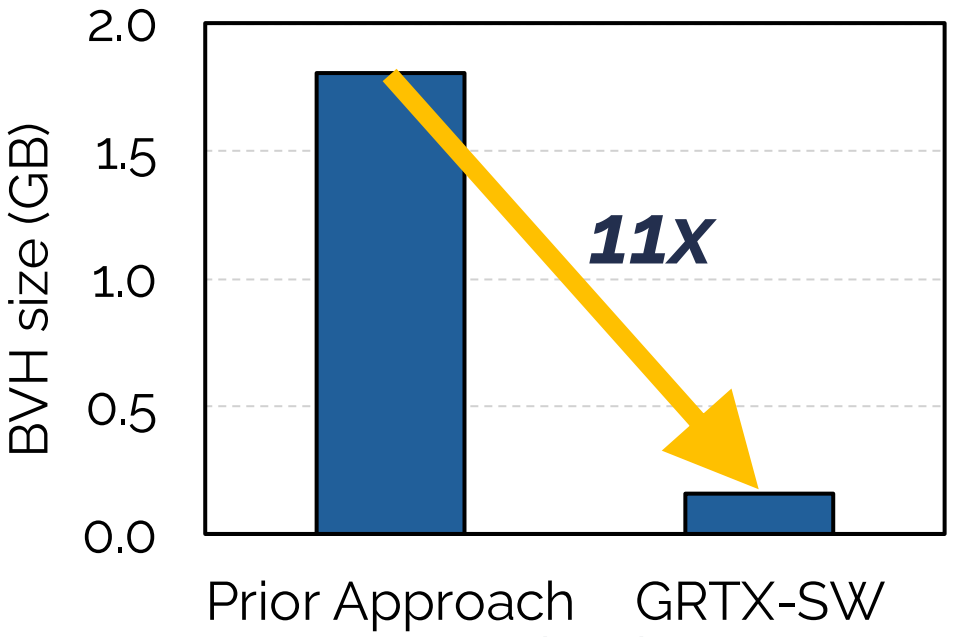
GRTX-SW

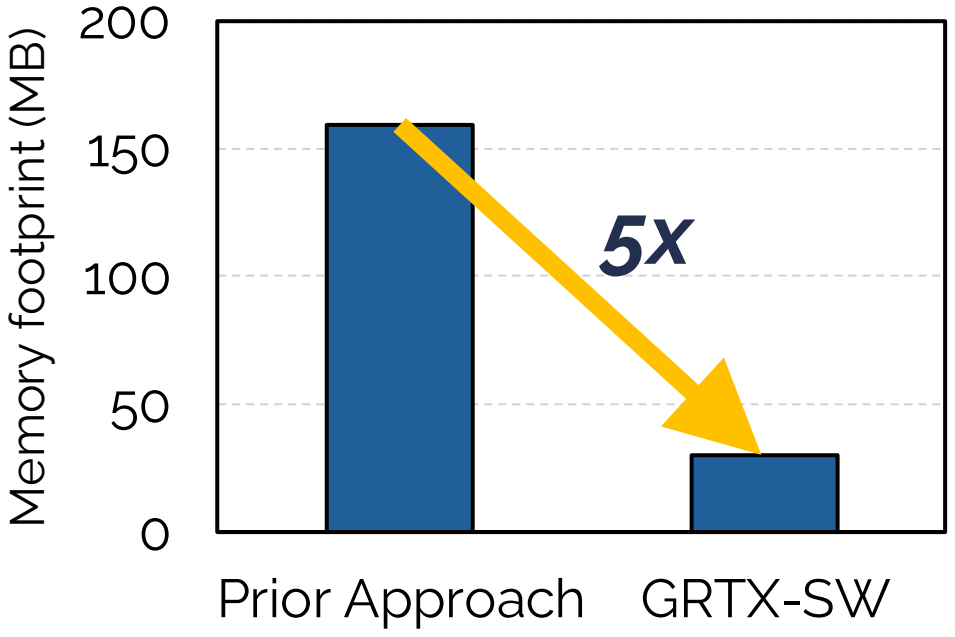


GRTX-SW: Two-Level BVH with a Shared BLAS

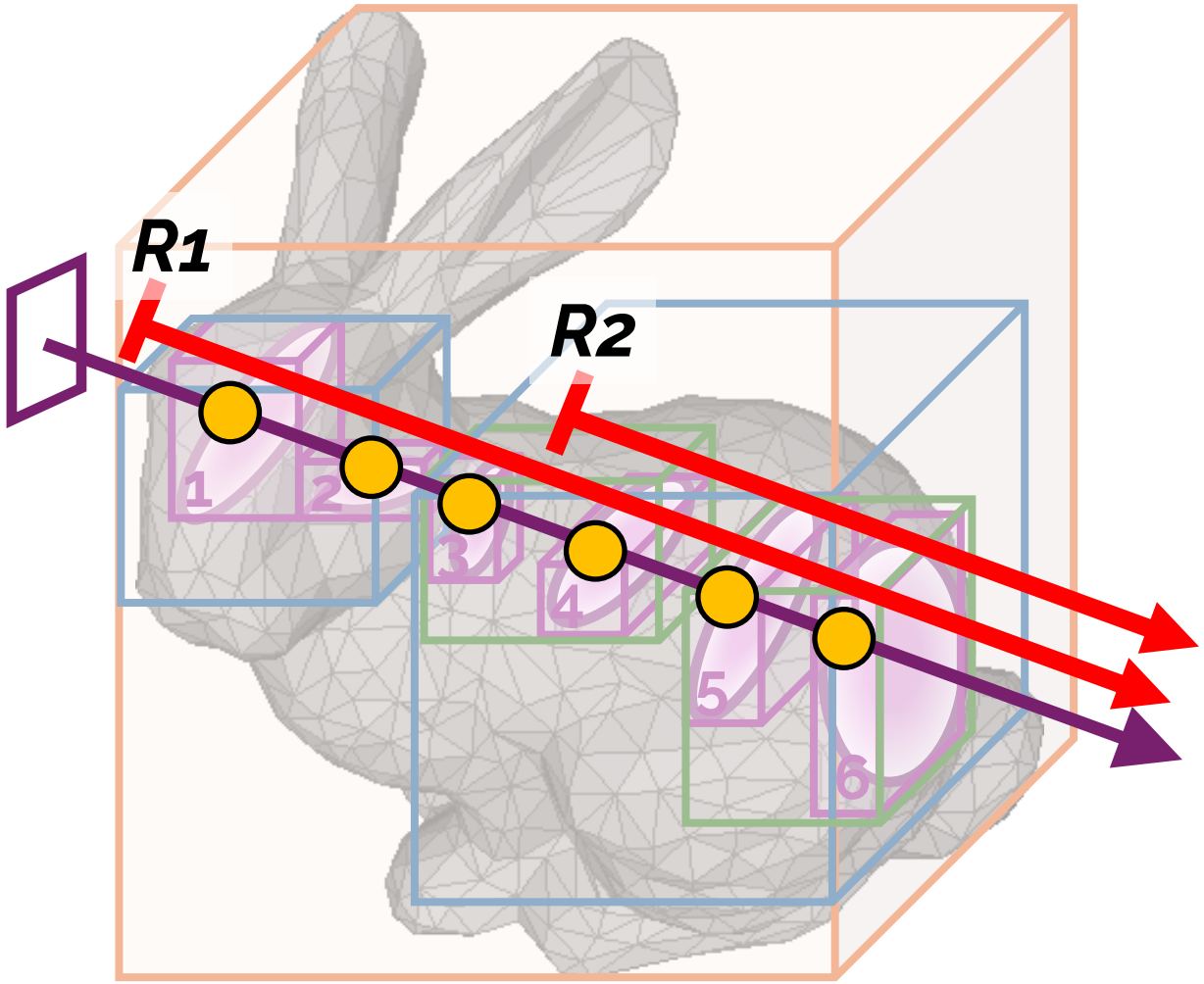
Advantage

Reduce BVH size & BVH memory footprint

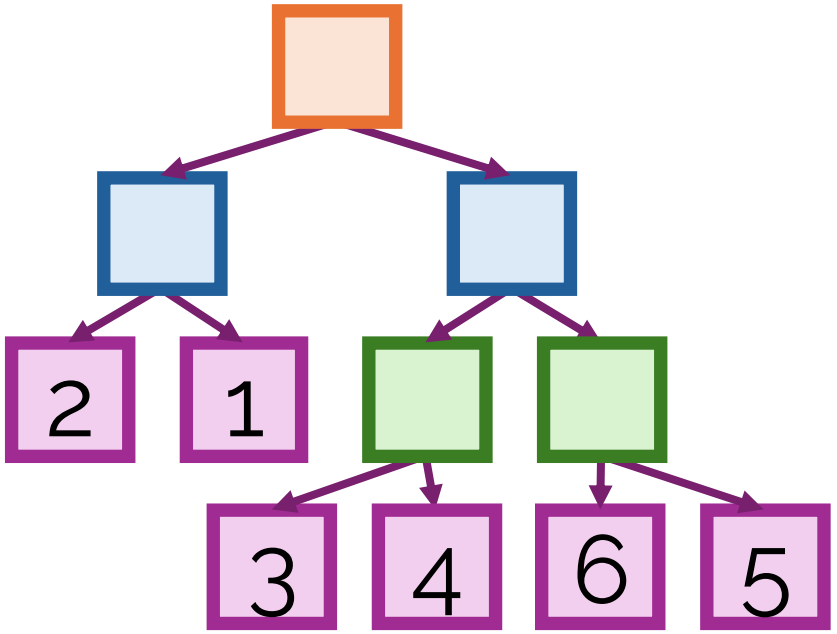




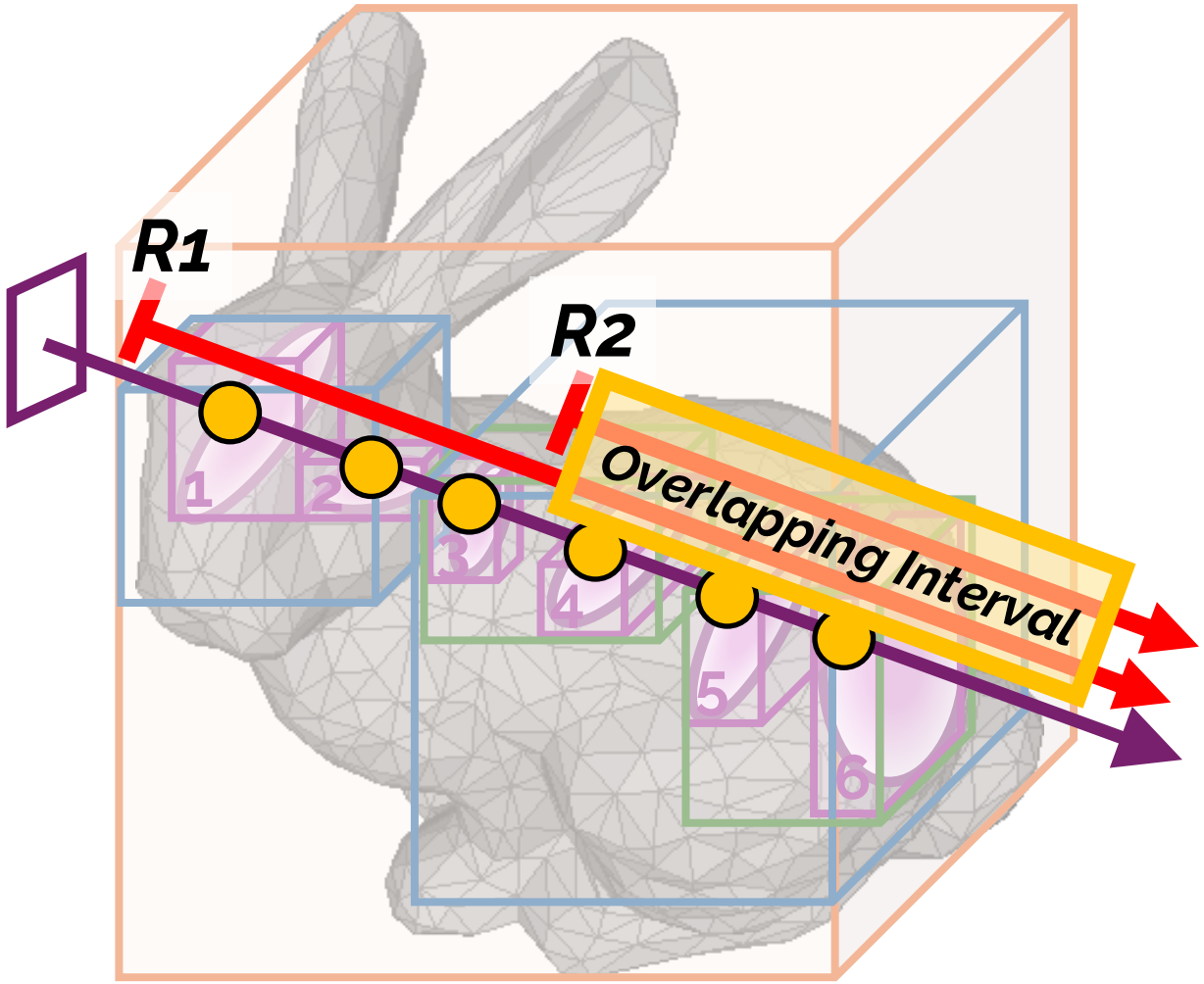
GRTX-HW: Traversal Checkpointing & Replay



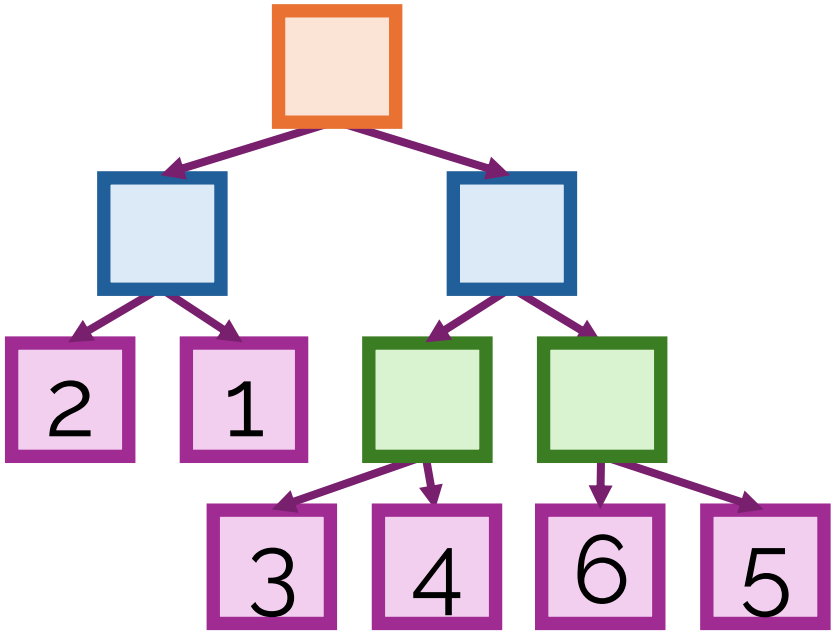
BVH traversal



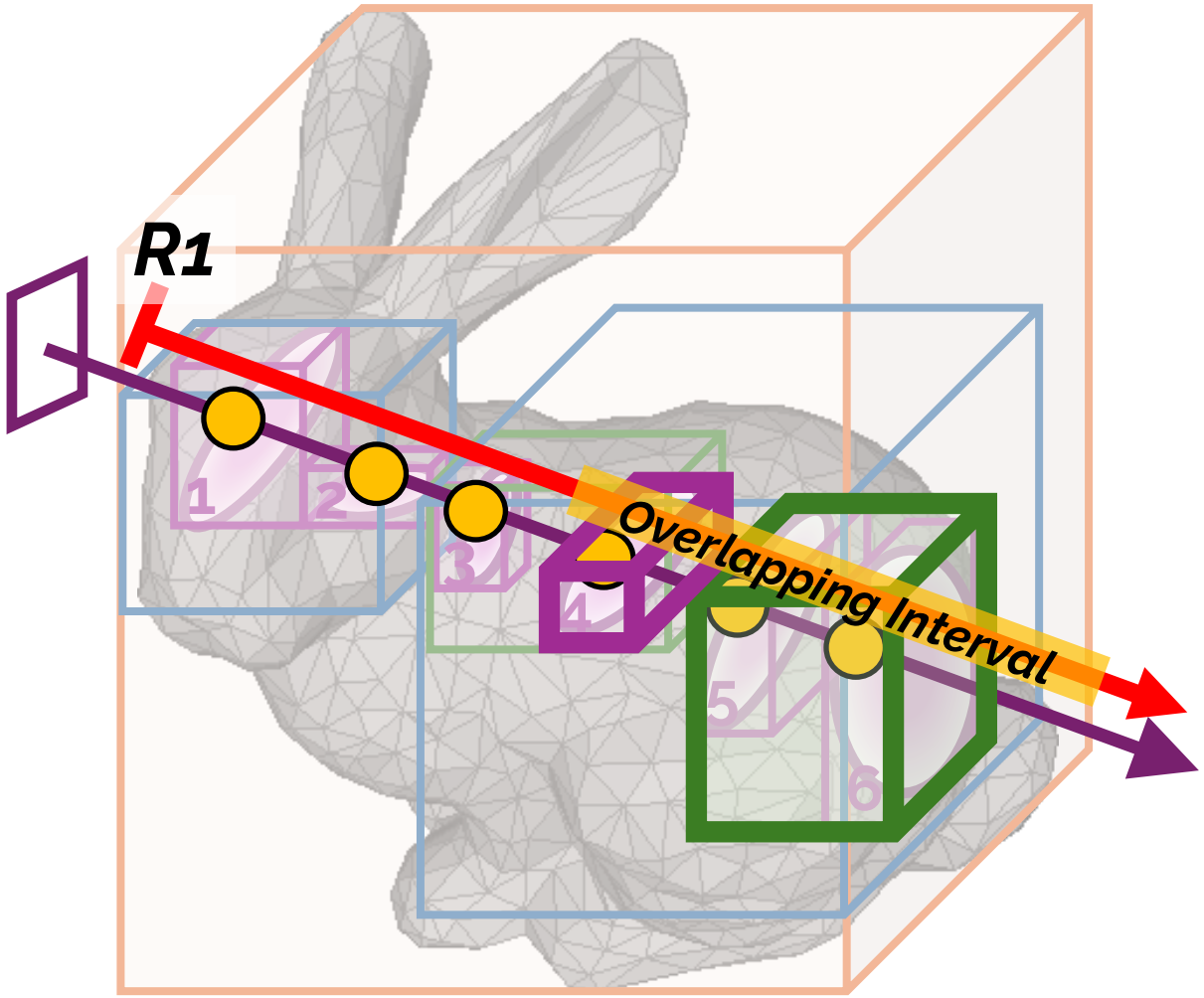
GRTX-HW: Traversal Checkpointing & Replay



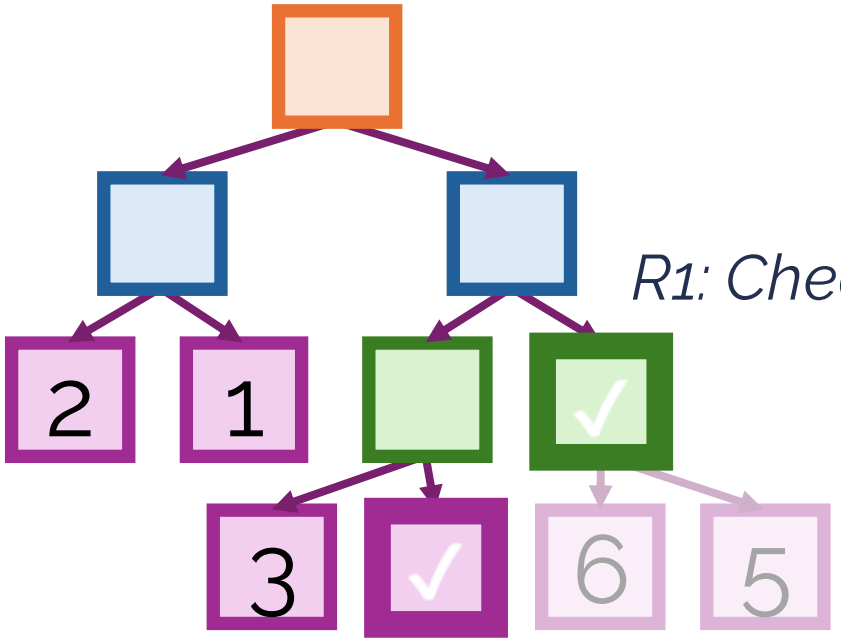
BVH traversal



GRTX-HW: Traversal Checkpointing & Replay

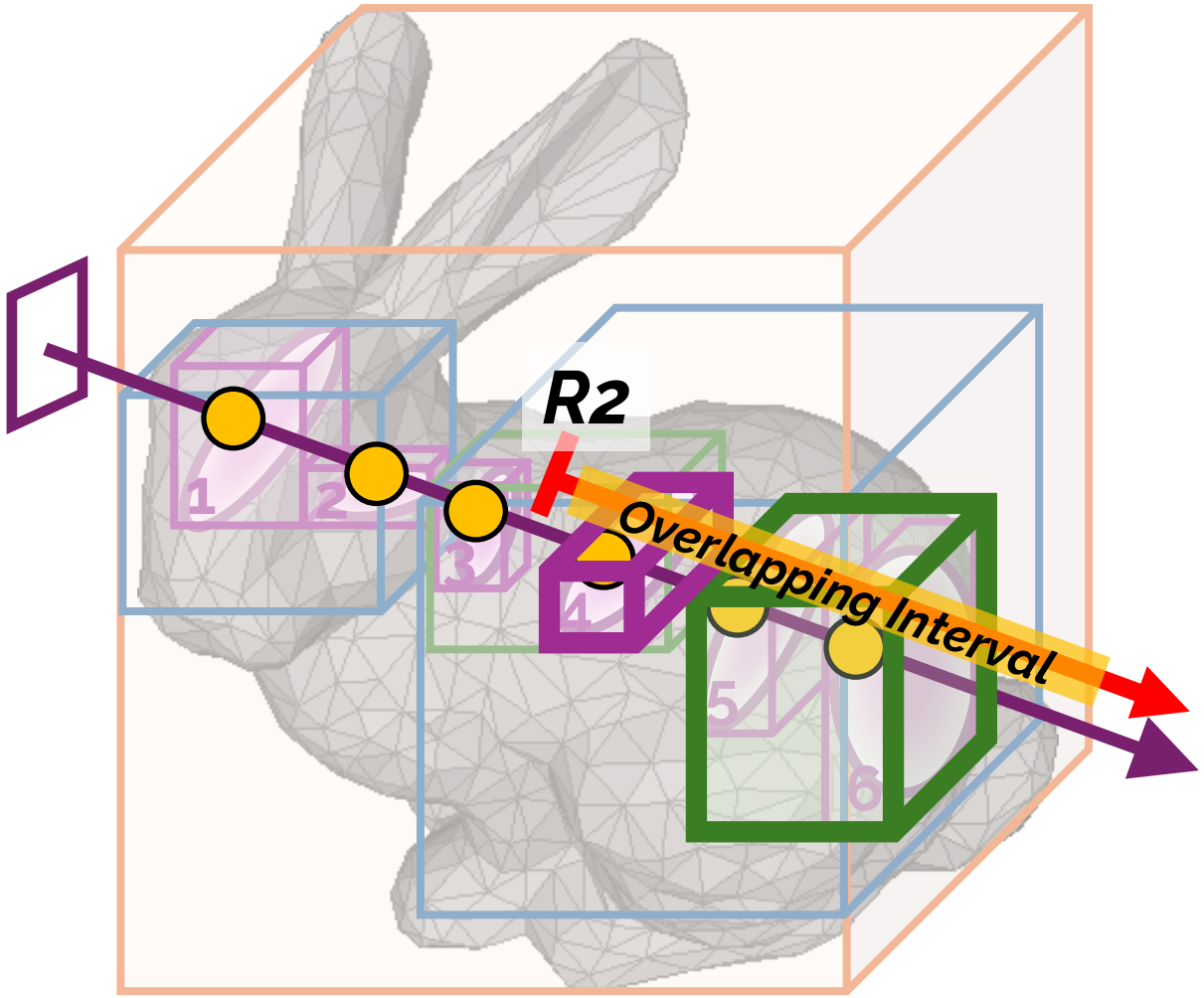


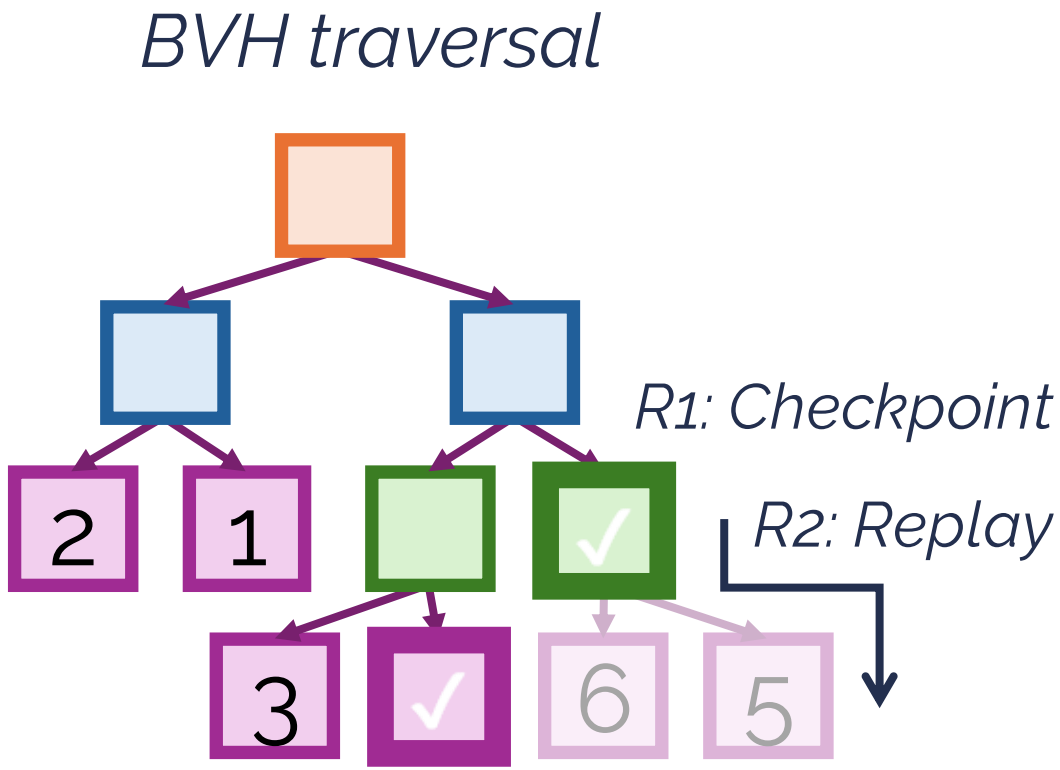
BVH traversal



R1: Checkpoint

GRTX-HW: Traversal Checkpointing & Replay

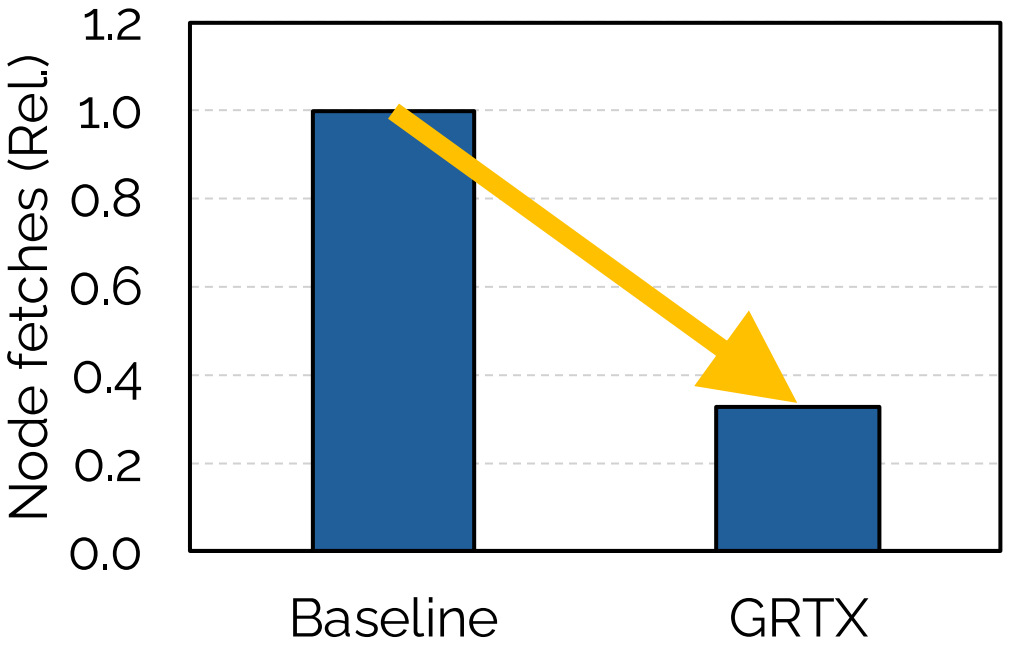




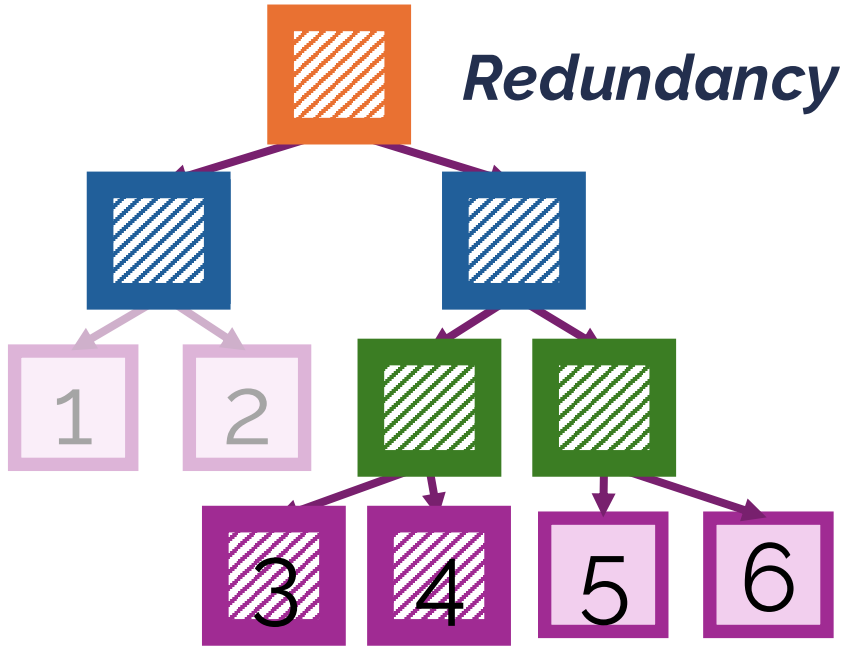
GRTX-HW: Traversal Checkpointing & Replay

Advantage

Eliminate redundant node visits & tests across rounds



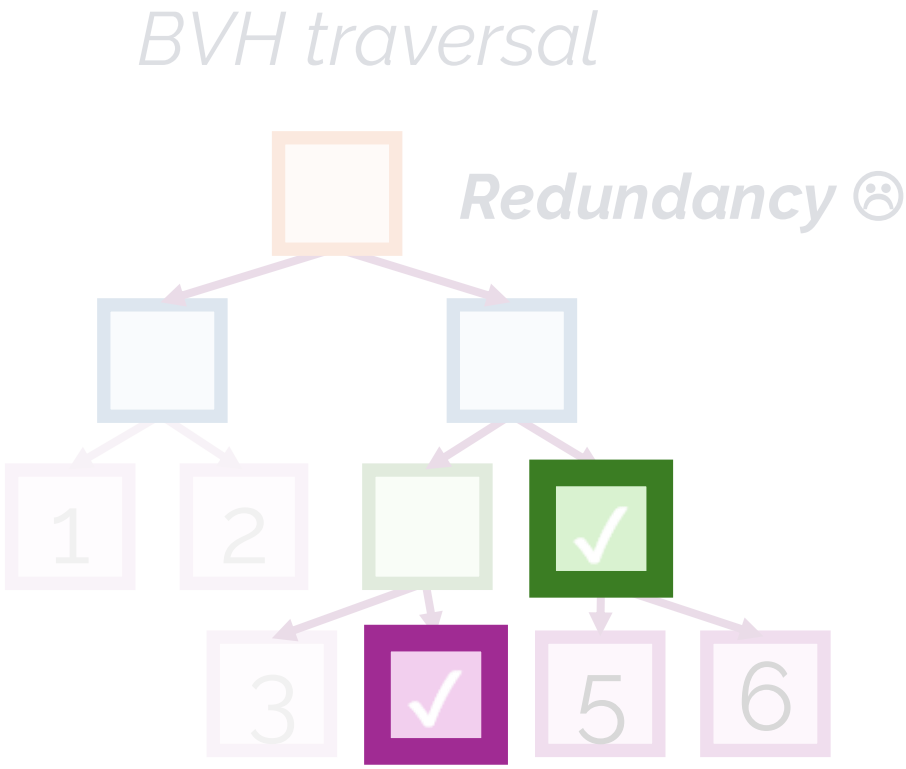
BVH traversal



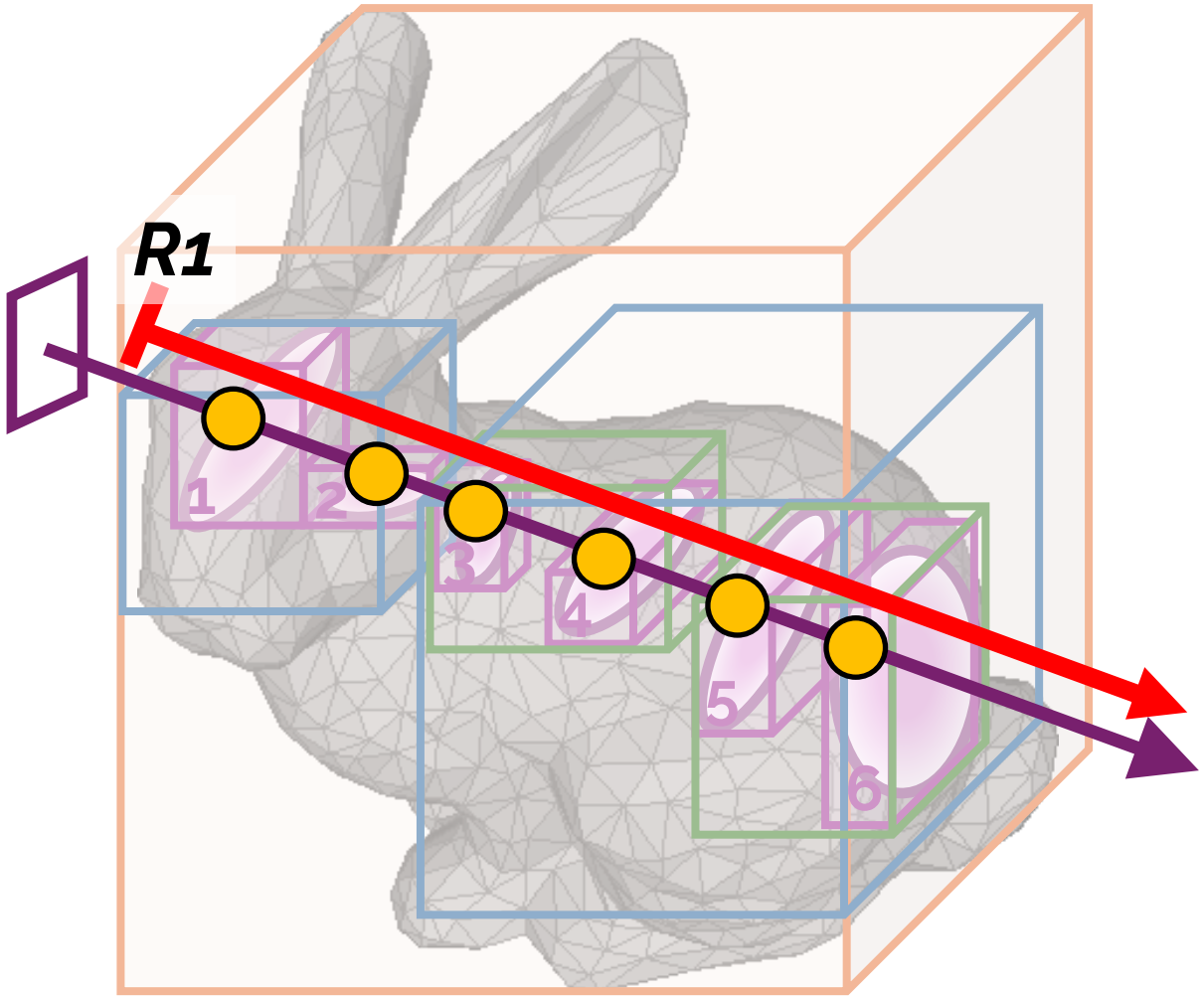
GRTX-HW: Traversal Checkpointing & Replay

Challange

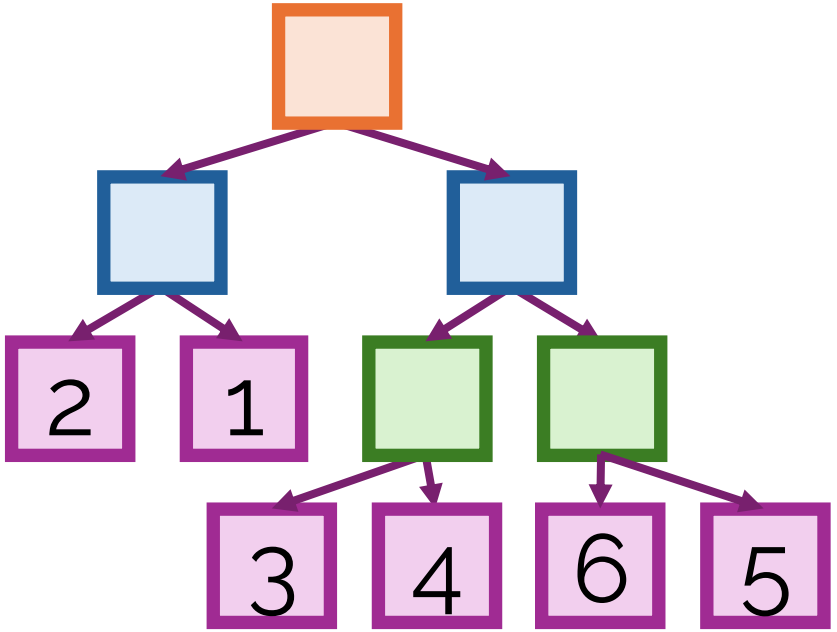
*Q: How can we **pinpoint** nodes that should be **checkpointed?***



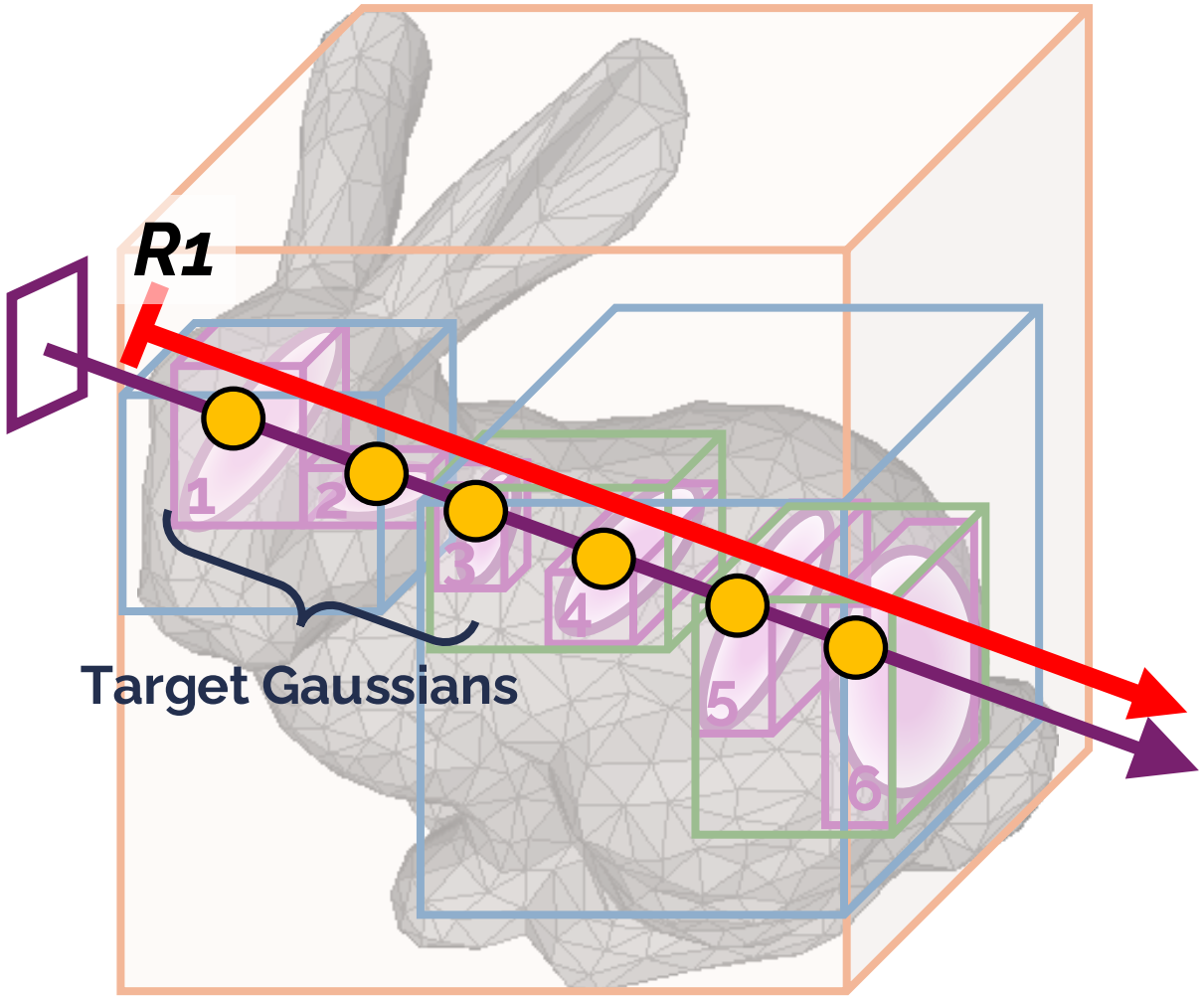
GRTX-HW: Traversal Checkpointing & Replay



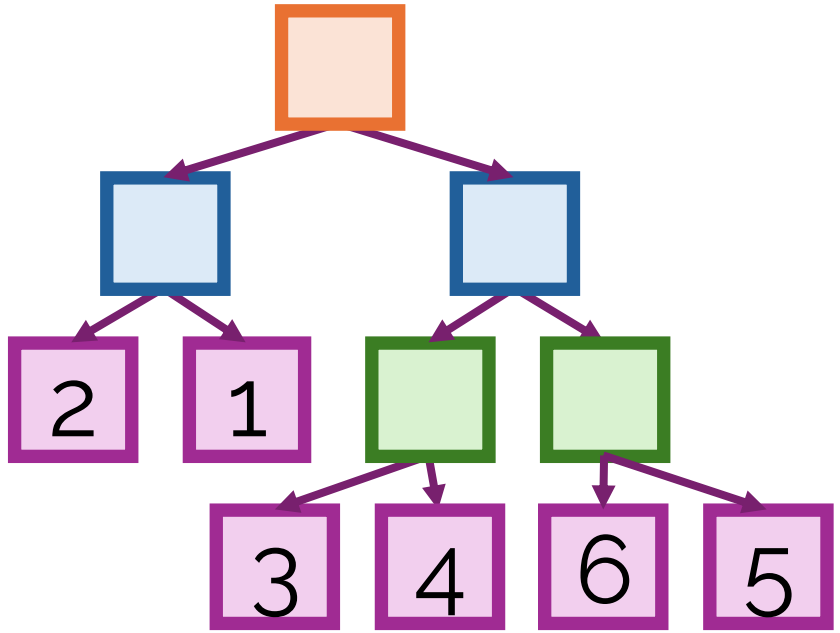
Round 1



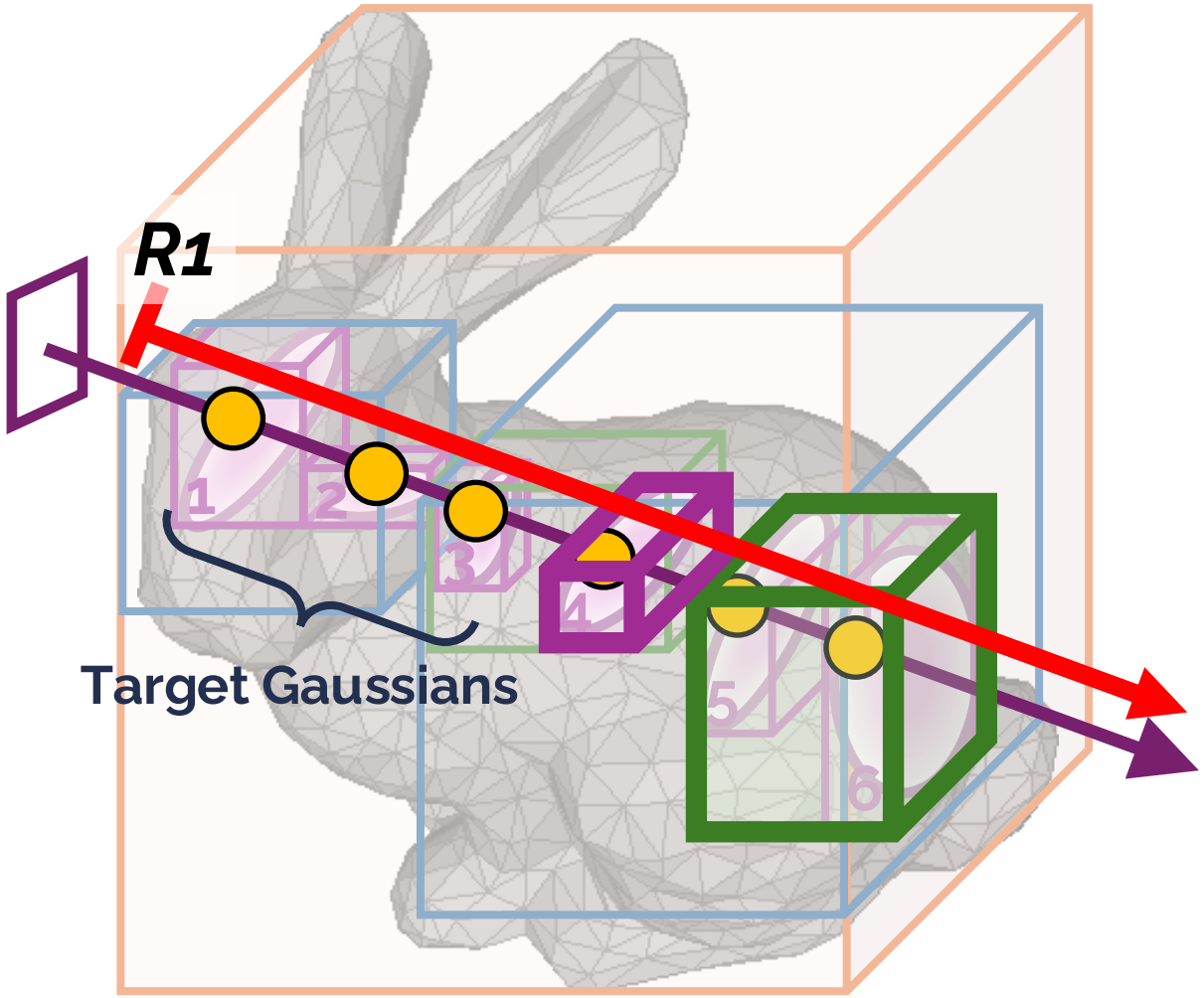
GRTX-HW: Traversal Checkpointing & Replay



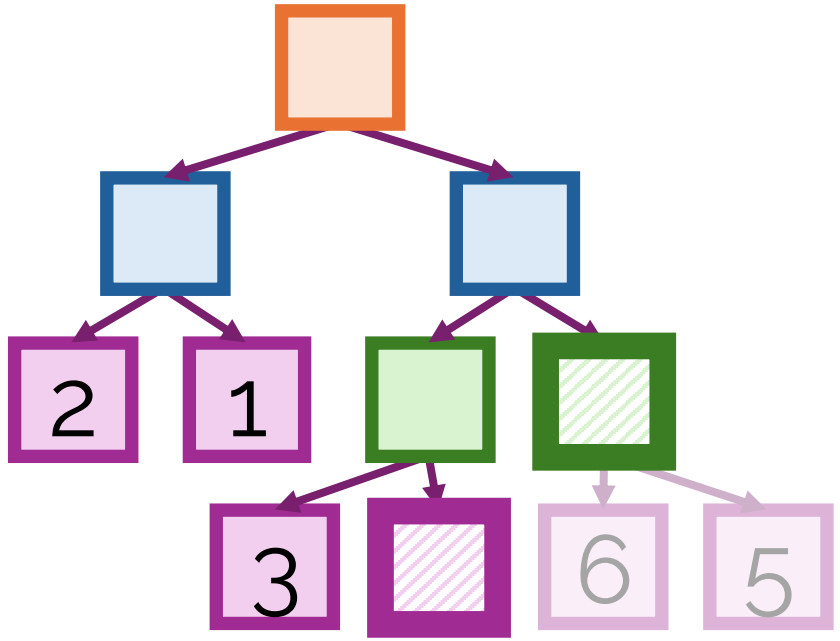
Round 1



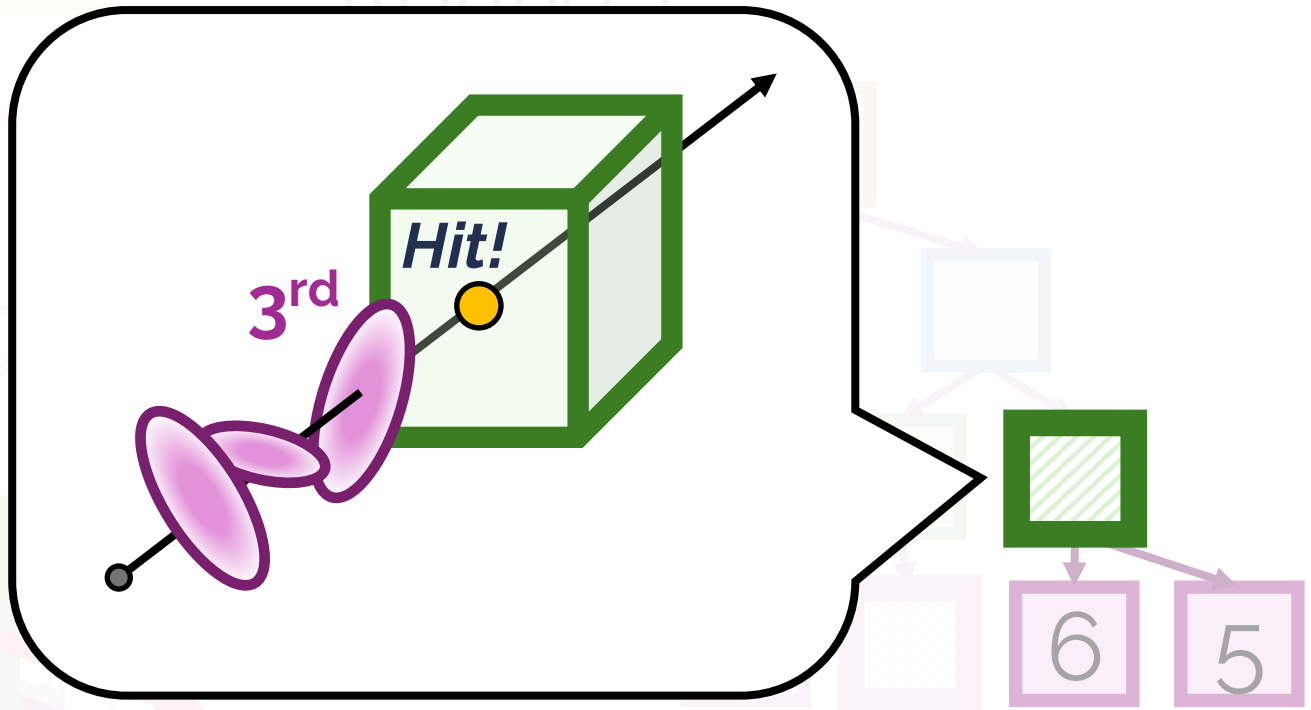
GRTX-HW: Traversal Checkpointing & Replay



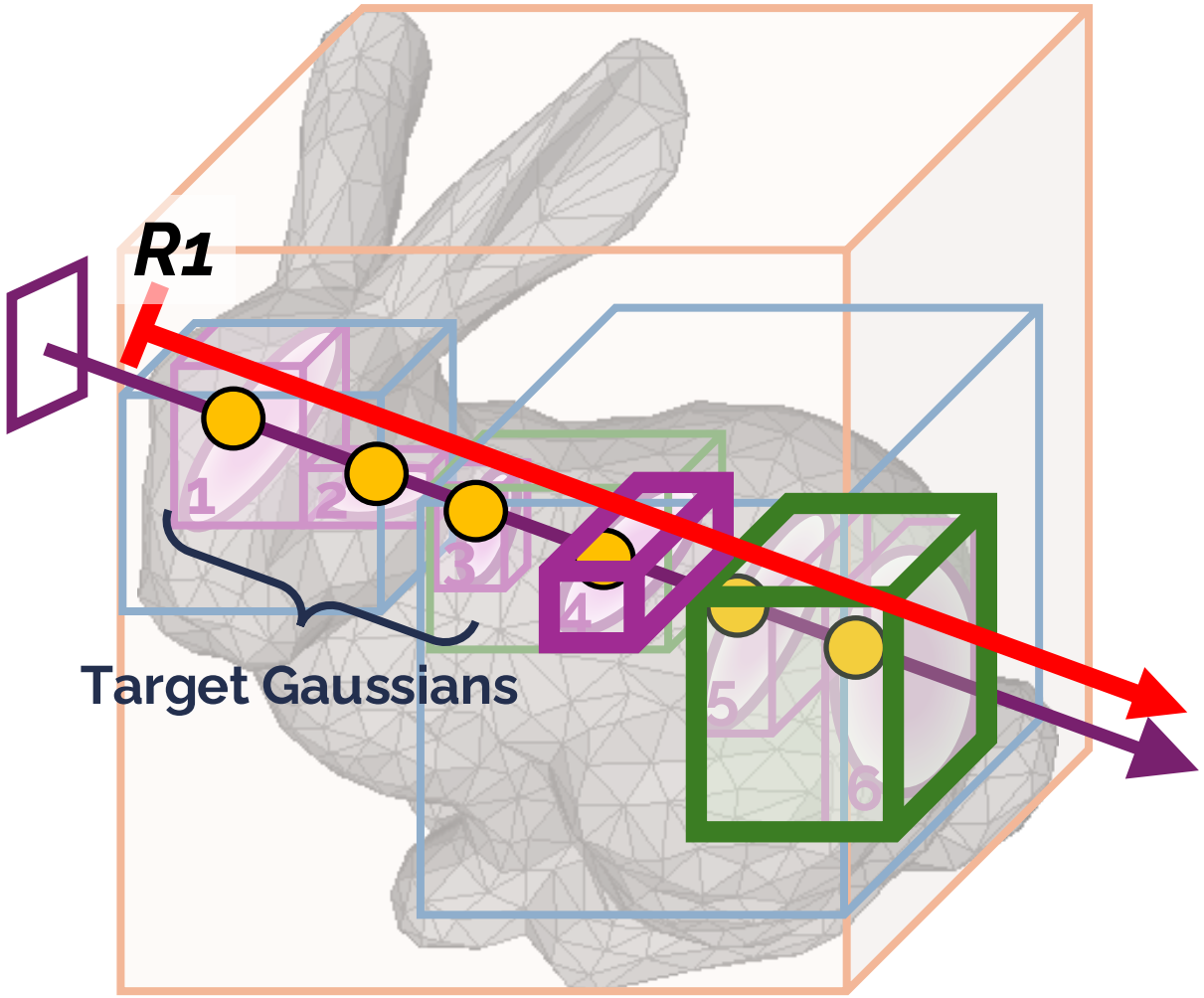
Round 1



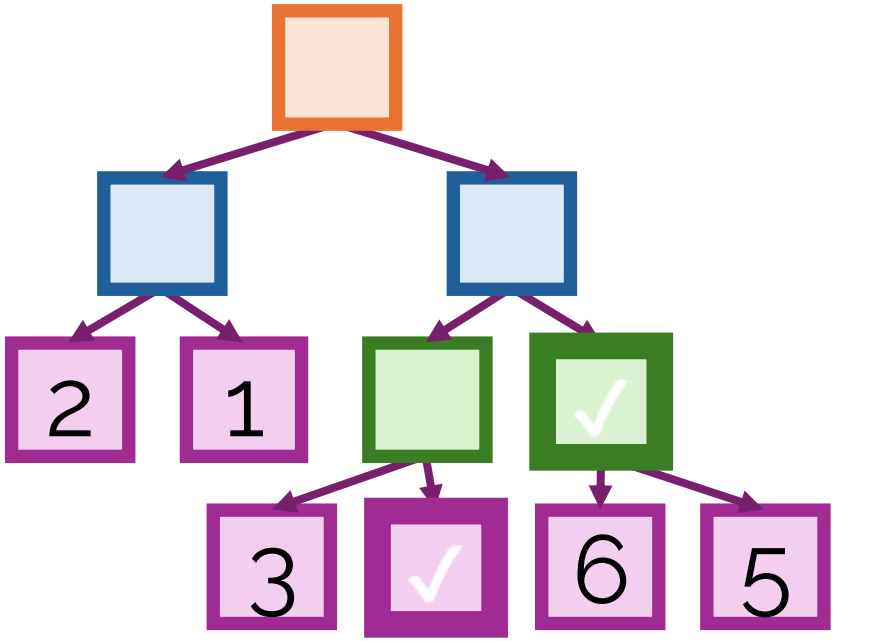
GRTX-HW: Traversal Checkpointing & Replay

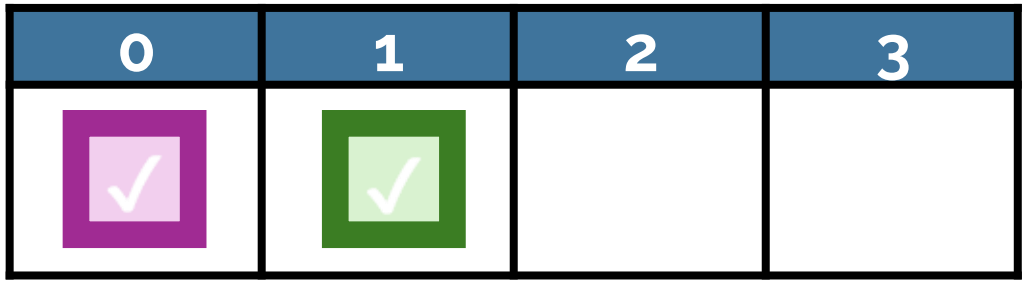


GRTX-HW: Traversal Checkpointing & Replay



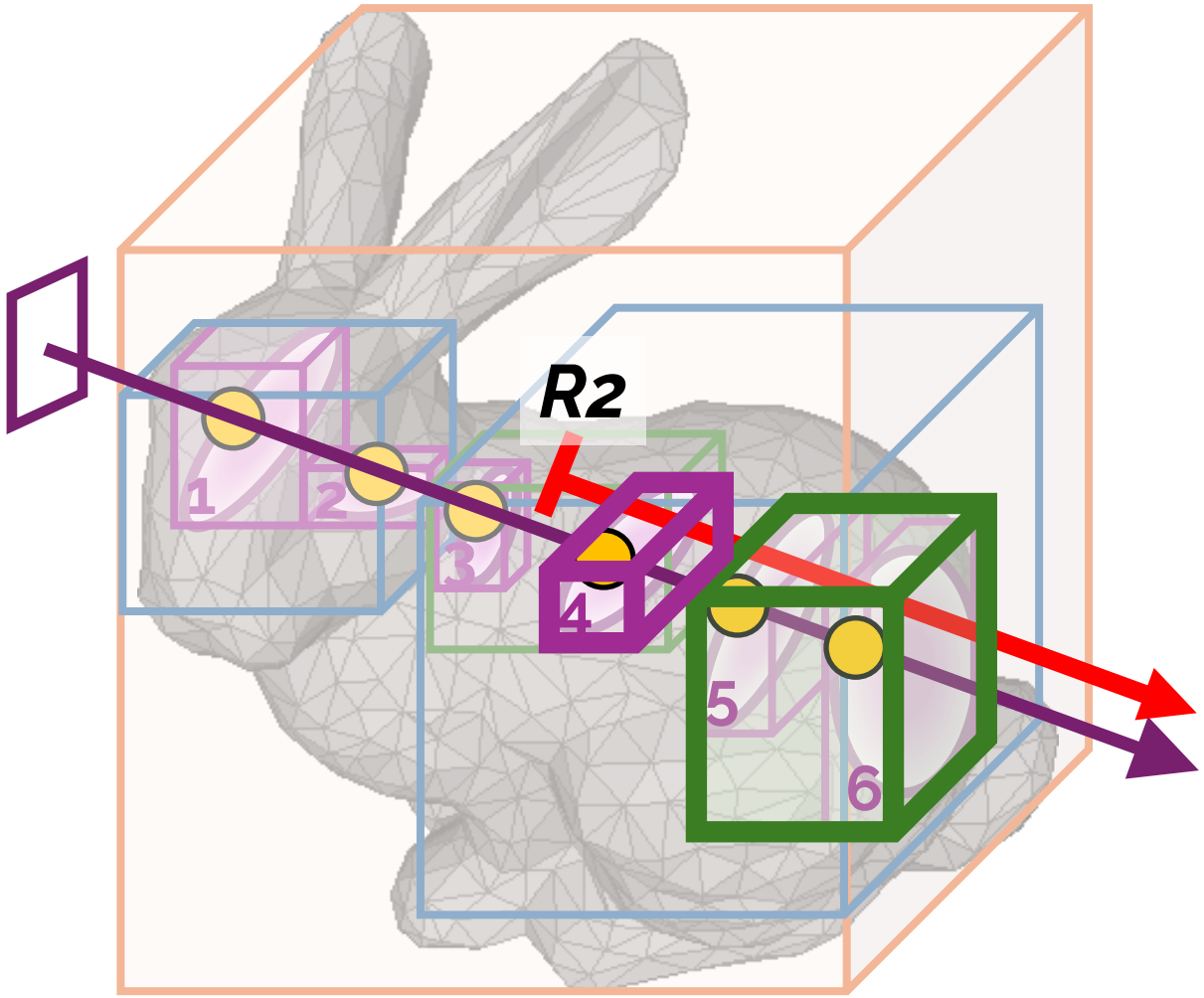
Round 1



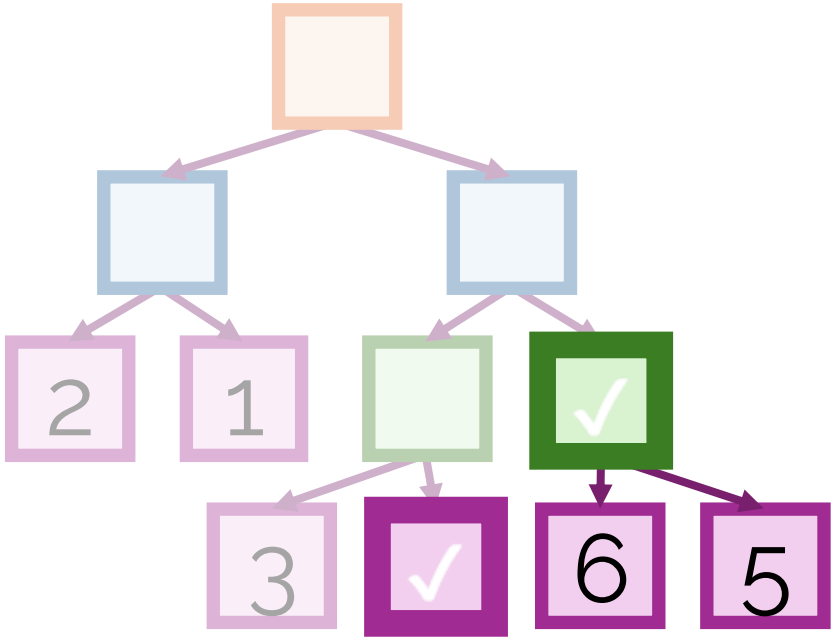


Checkpoint buffer

GRTX-HW: Traversal Checkpointing & Replay



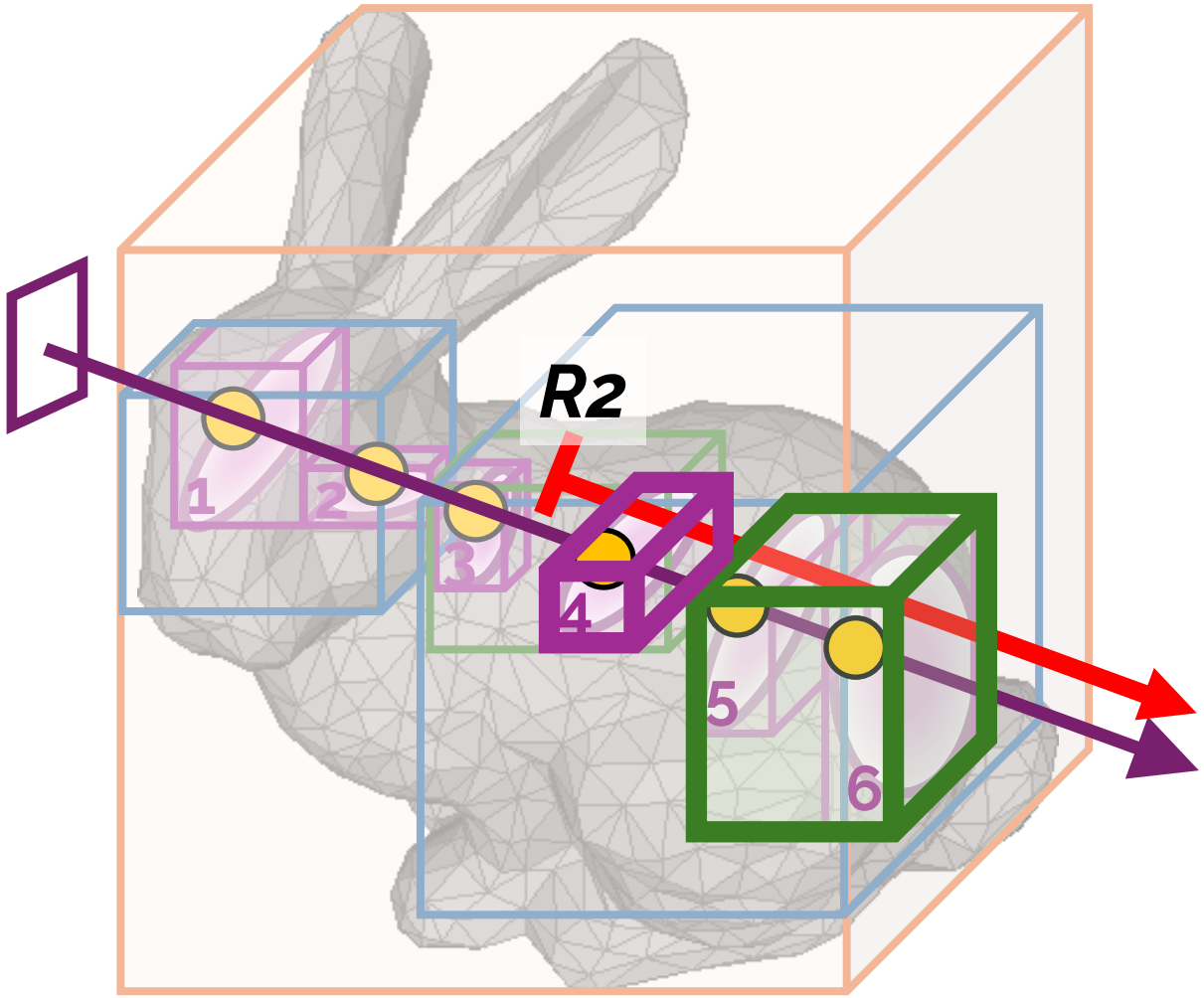
Round 2



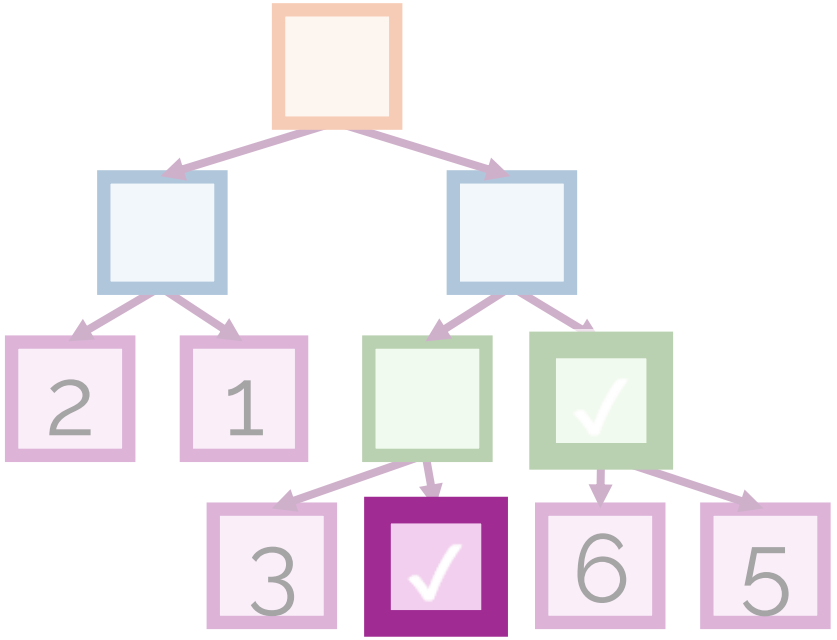
0	1	2	3
✓	✓		

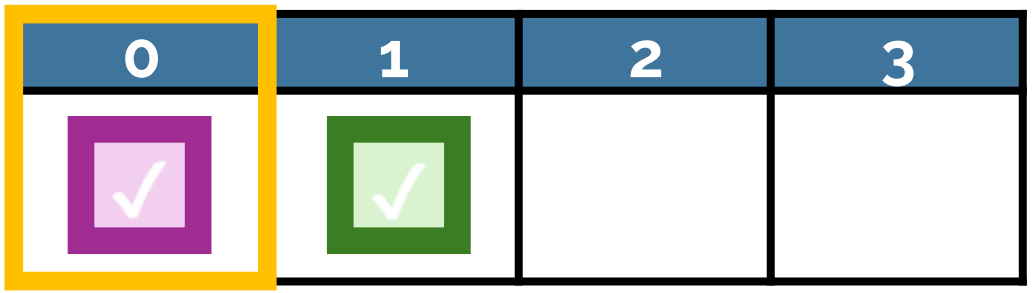
Checkpoint buffer

GRTX-HW: Traversal Checkpointing & Replay



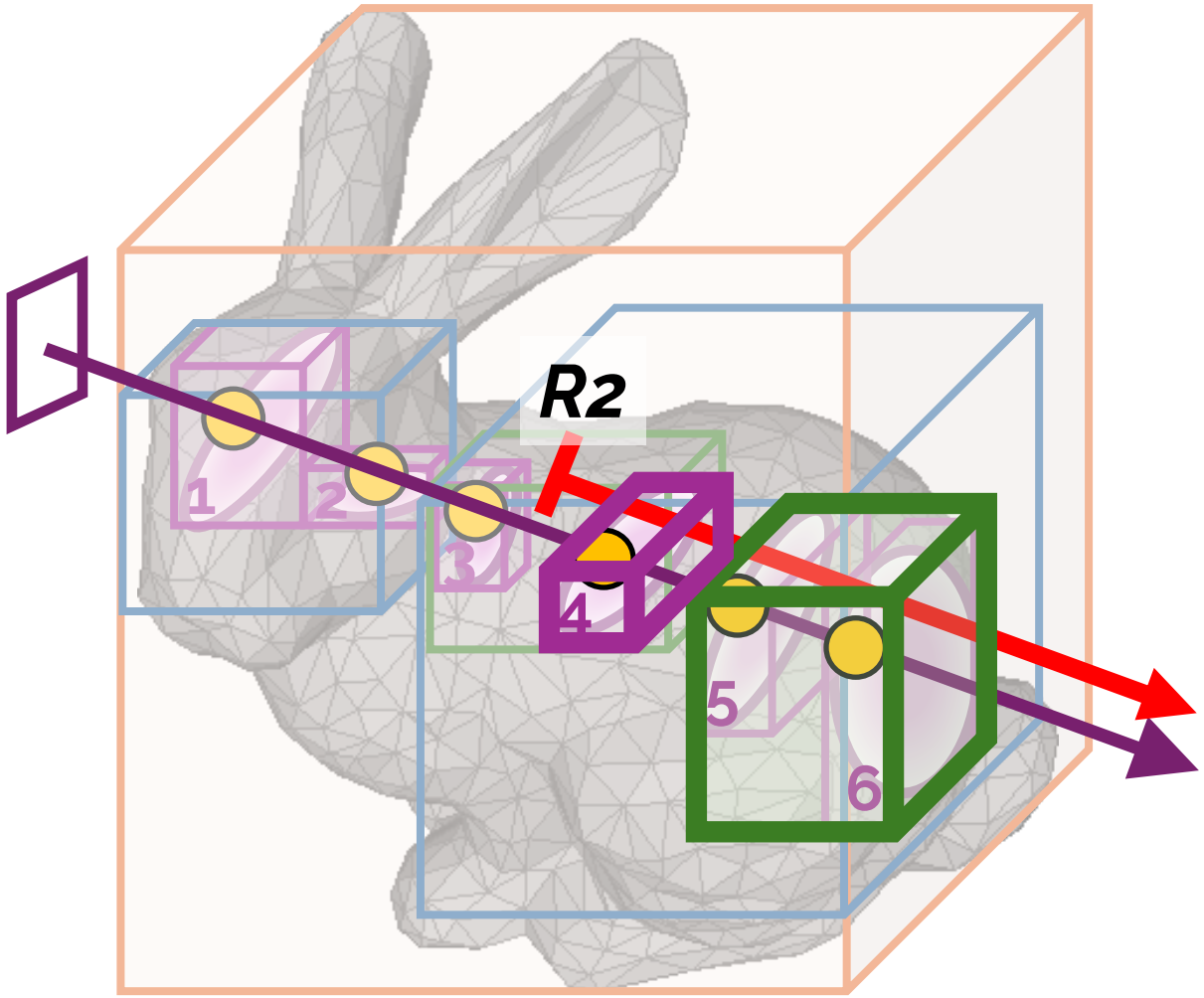
Round 2



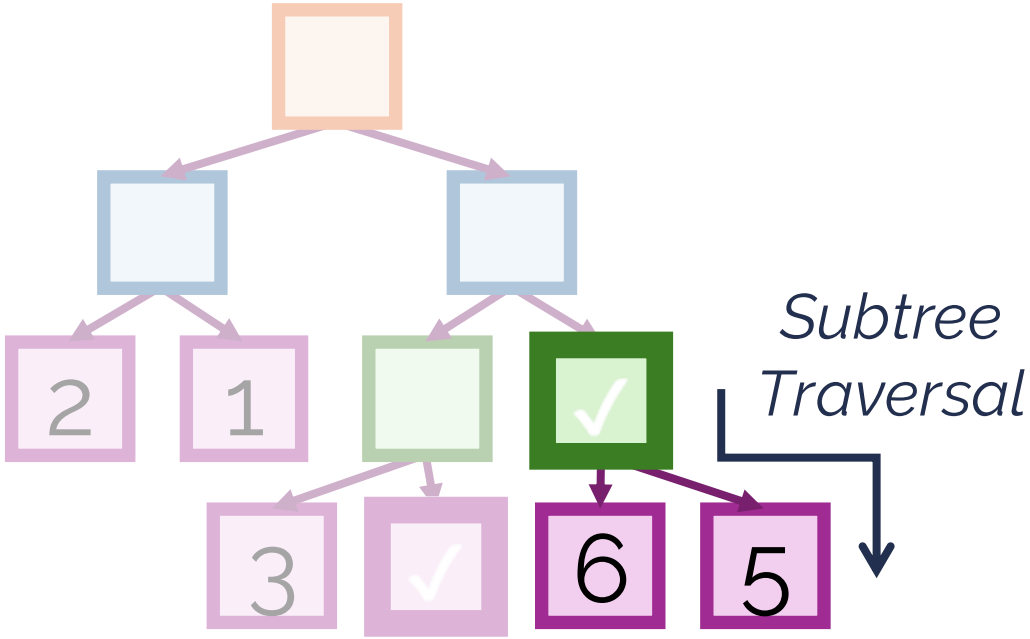


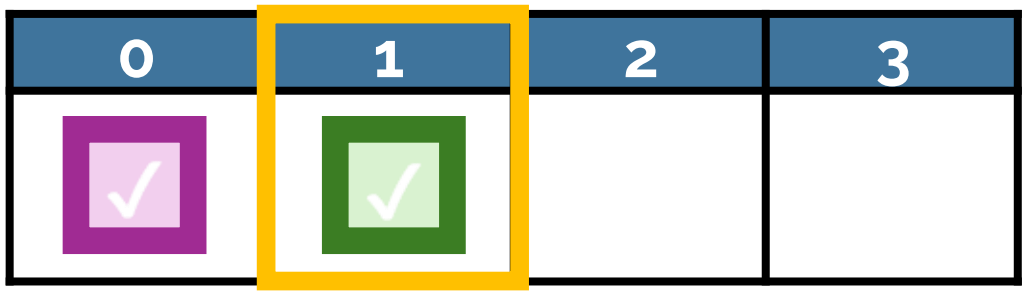
Checkpoint buffer

GRTX-HW: Traversal Checkpointing & Replay



Round 2





Checkpoint buffer

Outline

- **Background**
  - 3D Gaussian-based Rendering: Rasterization vs. Ray-tracing
  - Ray Tracing Accelerators in Modern GPUs
- **Gaussian RT Optimizations & Limitations**
- **GRTX: SW-HW Optimizations for Gaussian Ray Tracing**
  - GRTX-SW: Two-Level Acceleration Structure for Gaussian Primitives
  - GRTX-HW: Traversal Checkpointing and Replay
- **Evaluation**
- **Conclusion**

Experimental Setup

Performance Evaluation

- Vulkan-Sim (MICRO'22)
  - Cycle-level GPU simulator w/ ray-tracing accelerator modeling based on GPGPU-sim
  - With our **in-house ray tracing simulator** based on real-GPU analysis

Workloads

- Tanks & Temples: Train, Truck
- Mip-NeRF 360: Bonsai, Room
- Deep Blending: Drjohnson, Playroom

Baseline

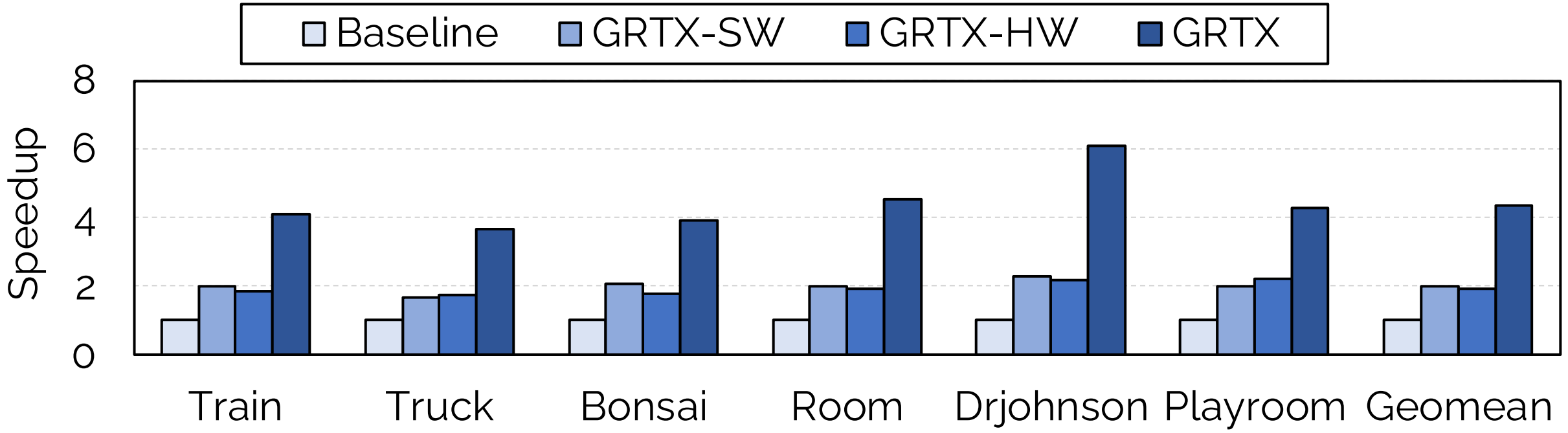
- Monolithic BVH w/ 20-faced polyhedron used in 3DGRT (NVIDIA) [1]

Baseline GPU Configuration

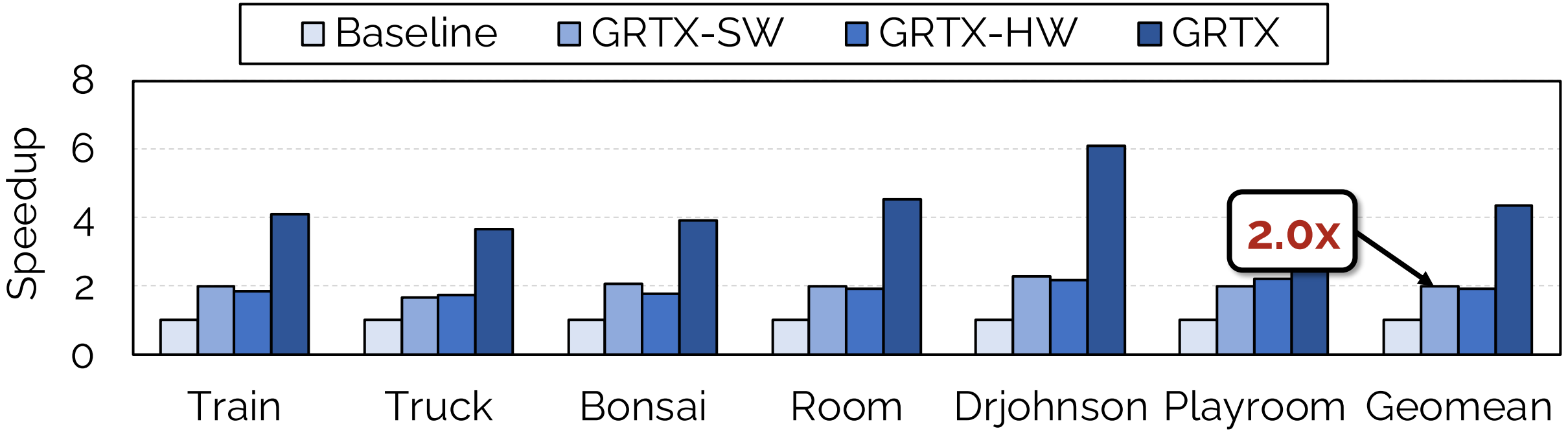
# GPC	1
# SMs	8
Core Frequency	1365 MHz
L1D Cache	128KB, 128B line (sectored)
L2 Cache	4MB, 128B line (sectored)
Memory Clock	3500MHz
# RT Units Per SM	1
Warp Buffer Size	8 (warps)

[1] Nicolas et al., 3D Gaussian Ray Tracing: Fast Tracing of Particle Scenes (SIGGRAPH Asia'24)

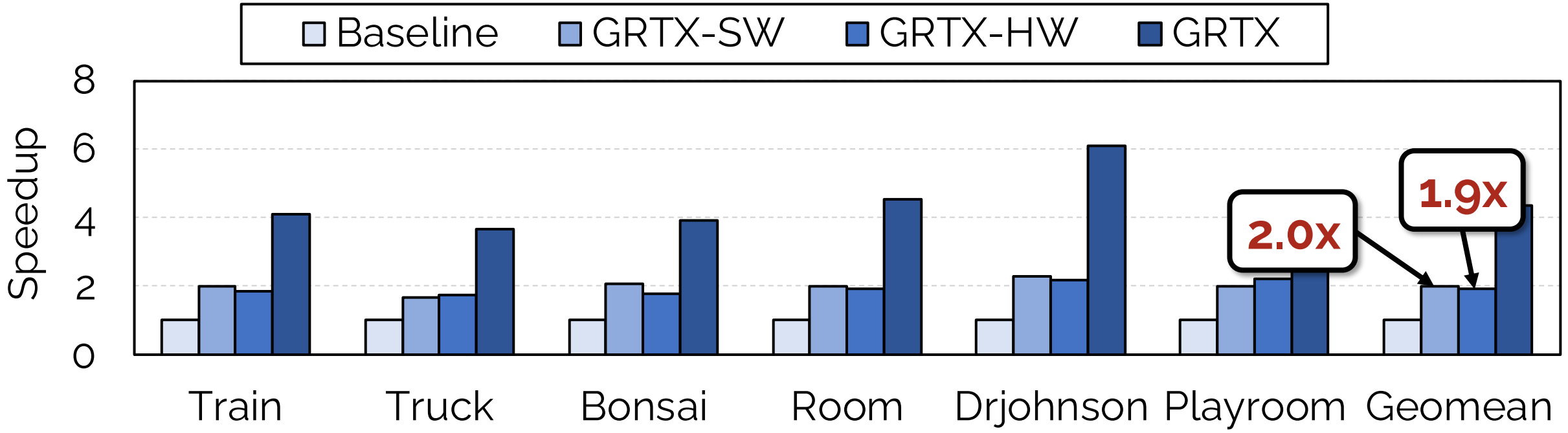
Performance



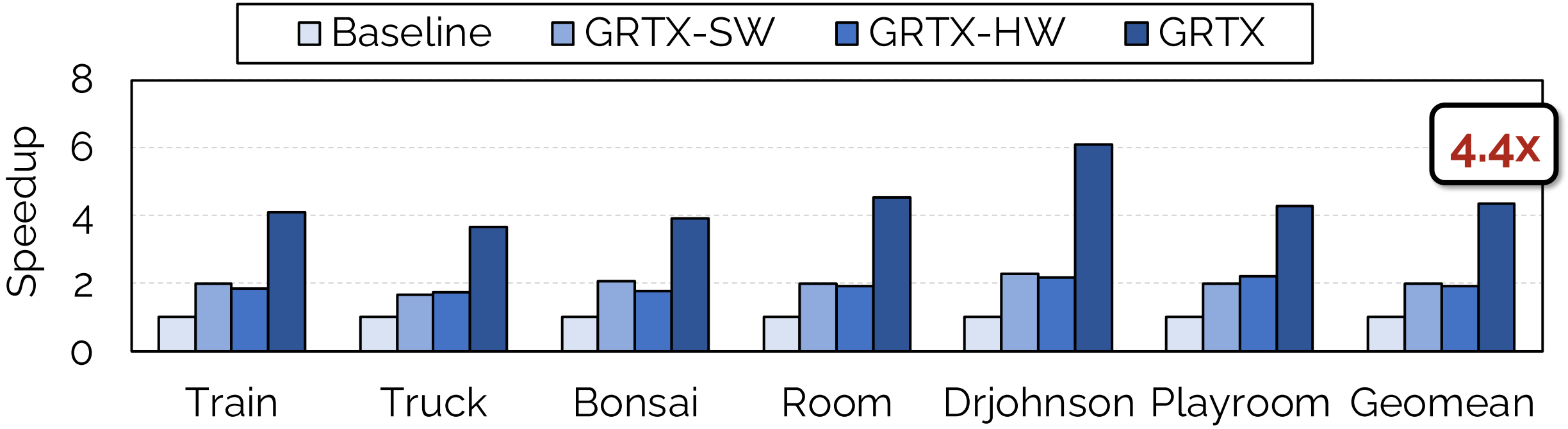
Performance



Performance



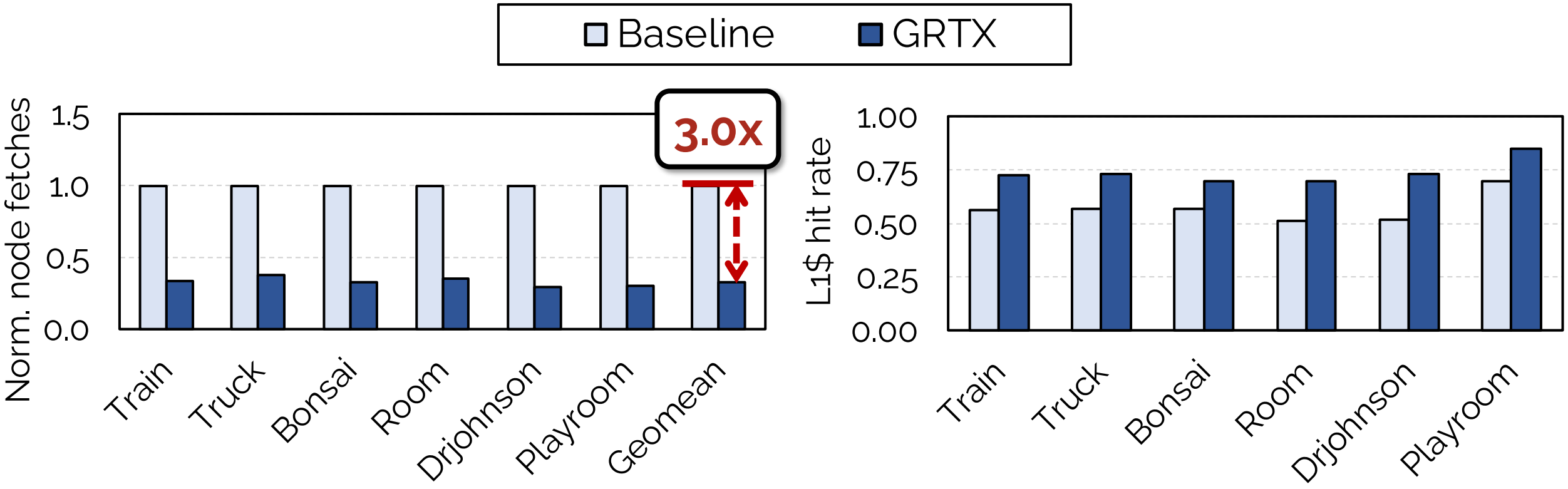
Performance



GRTX greatly improves rendering performance by 4.4x ☺

Source of Performance Gain

Number of Node Fetches & L1$ Hit Rate



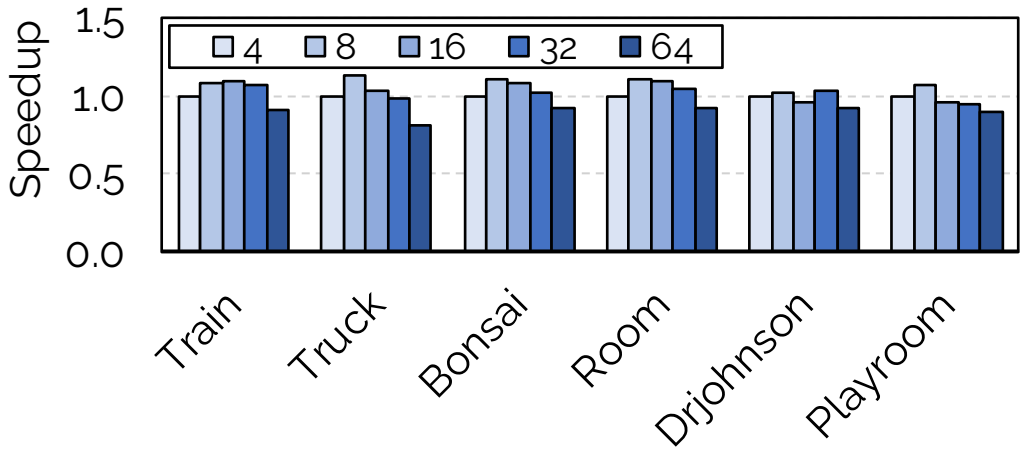
GRTX significantly **reduces the number of node fetches** while **increasing L1 cache hit rates**

More Details in Our Paper

- Support for dynamic and multi-object scenes
- GRTX-HW on secondary rays



- Sensitivity studies
  - k-buffer size
  - Varying resolutions and FoVs
- Cross-vendor applicability
- Others…



Conclusion

Problem

- Bloated BVH size and BVH memory footprint
- Redundant node visits and intersection tests across traversal rounds

Solution: **GRTX**, SW & HW optimizations for Gaussian ray-tracing

- **GRTX-SW**: Two-level BVH with a shared BLAS
- **GRTX-HW**: RT hardware extension with checkpointing and replay capabilities

Result

- GRTX shows **4.3x faster** ray-tracing performance over the baseline GPU with negligible hardware cost! ☺

Thank You!

GRTX

Efficient Ray Tracing for 3D Gaussian-Based Rendering

Junseo Lee
(junseo.lee@snu.ac.kr)

