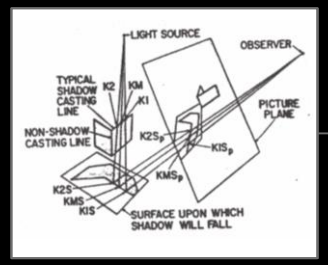
REAL-TIME RAYTRACING WITH NVIDIA RTX

Martin Stich, Director of Engineering

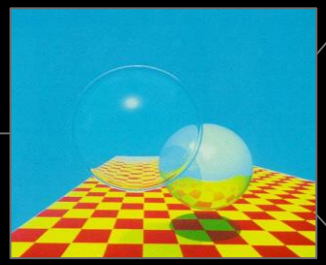
October 2018

GAMESCOM, KÖLN 2018

DECADES OF RAYTRACING RESEARCH



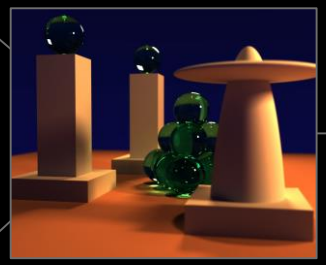
APPEL | 1968
"Some Techniques for Shading Machine Renderings of Solids"



TURNER WHITTED | 1979
"An Improved Illumination Model for Shaded Display"

COOK, PORTER & CARPENTER | 1984
"Distributed Ray Tracing"

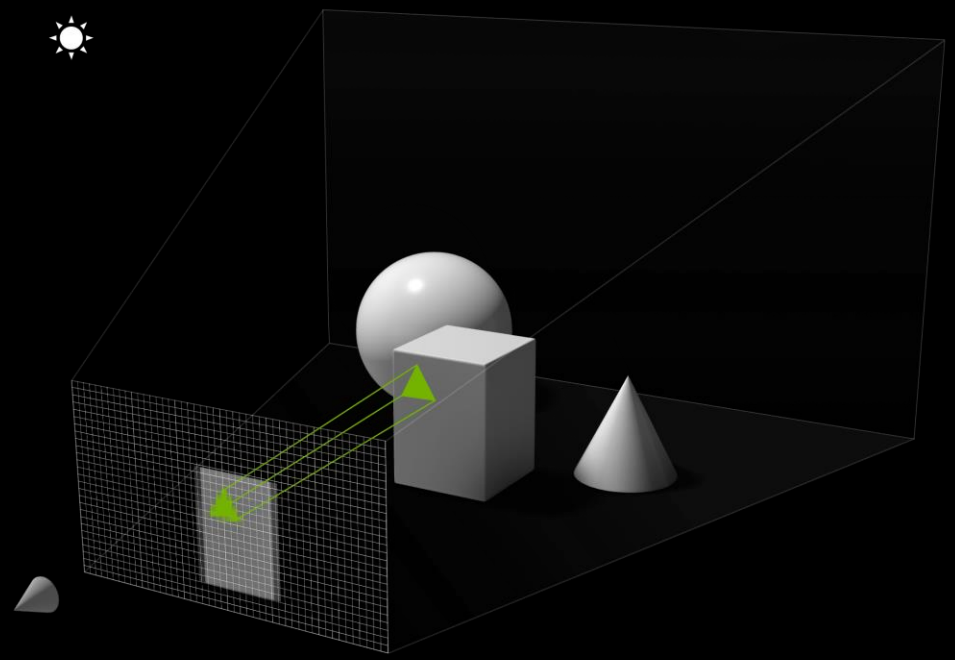
GORAL, TORRANCE & GREENBERG | 1984
"Modeling the Interaction of Light Between Diffuse Surfaces"



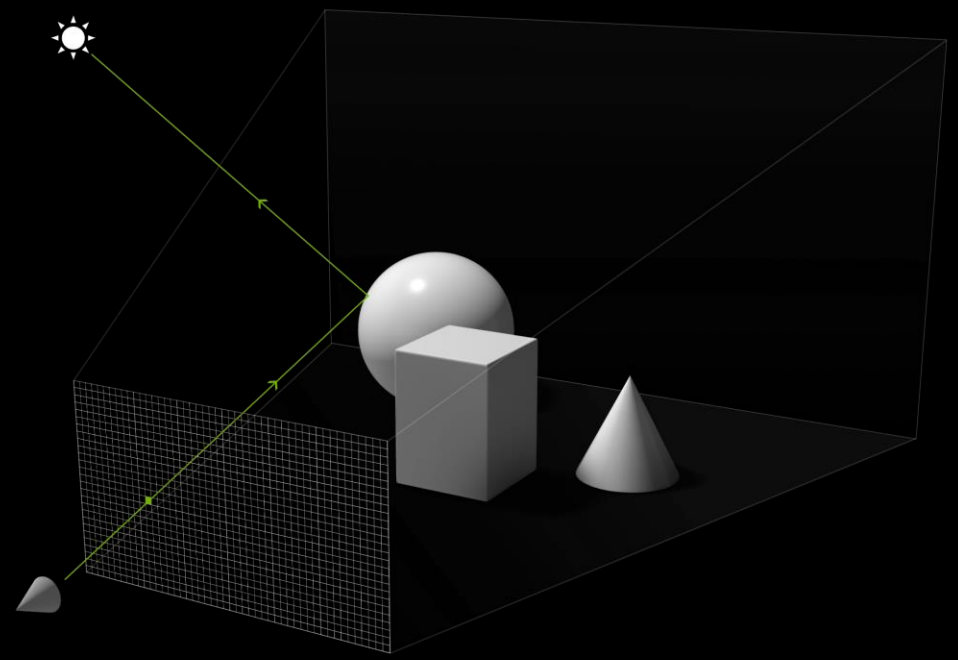
KAJIYA | 1986
"The Rendering Equation"

NVIDIA | 2008
SIGGRAPH

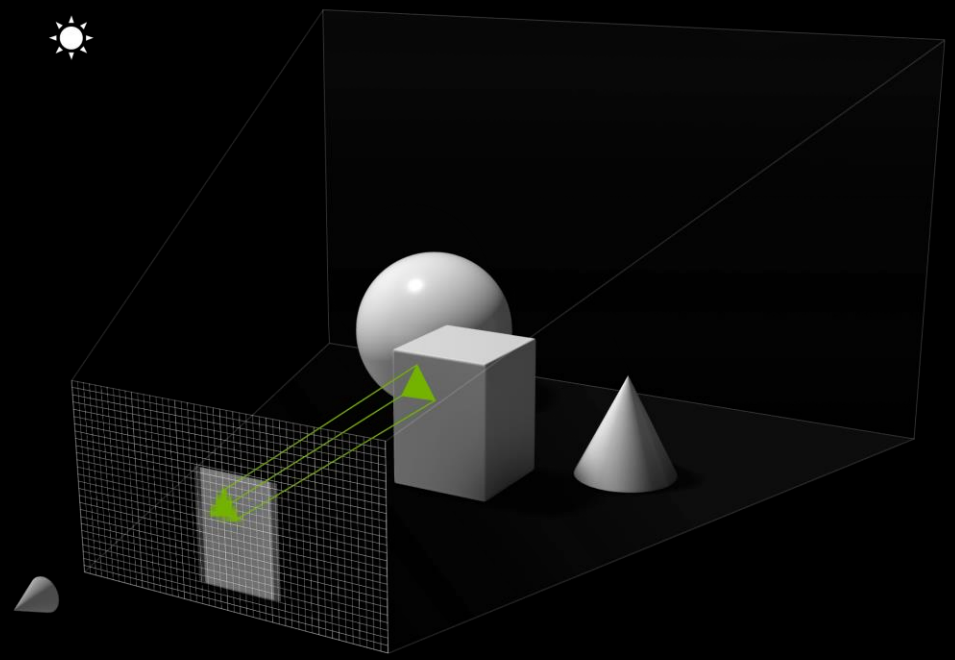
NVIDIA | 2018
SIGGRAPH + GAMESCOM



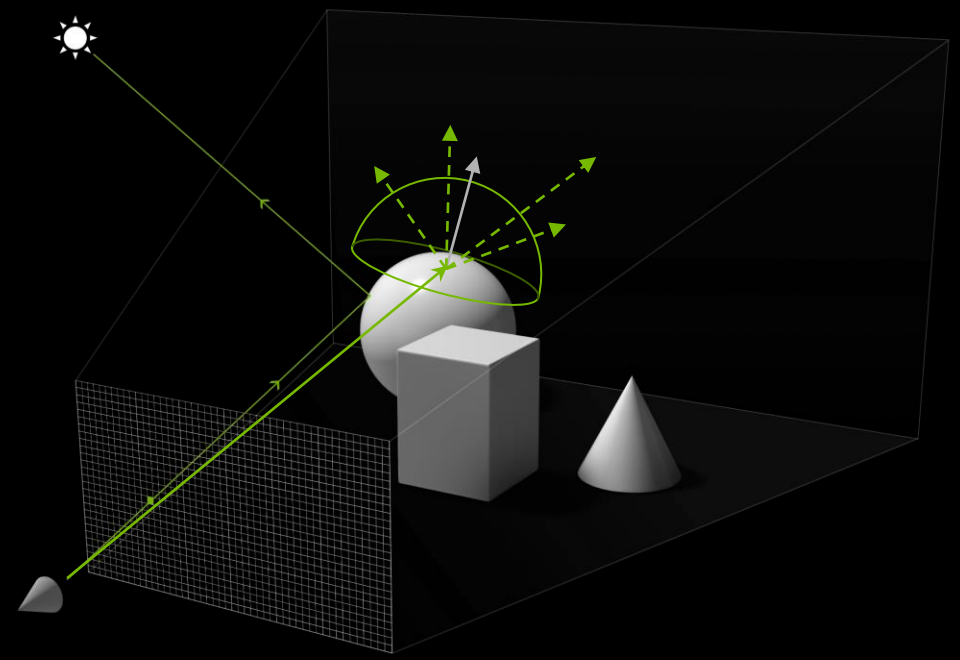
RASTERIZATION



RAYTRACING



RASTERIZATION



RAYTRACING

INFLUENCE OF RASTERIZATION-ONLY

A Bag of Tricks

Can I see the light?

PCF shadow maps, stencil shadows, perspective shadow maps, cascaded shadow maps, SDSMs, AVSMs, variance shadow maps, exponential shadow maps, exponential variance shadow maps, light probes, light maps, …

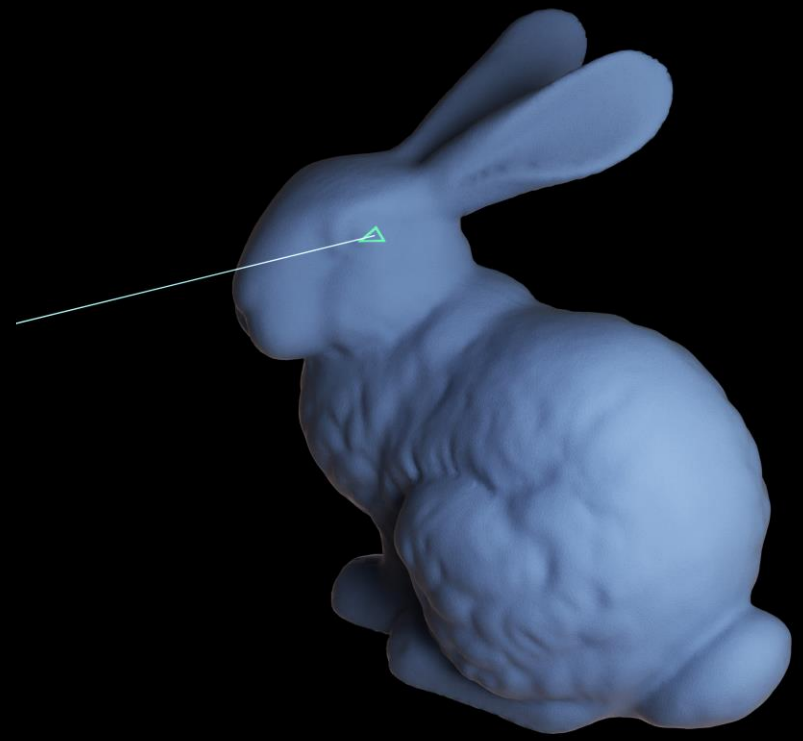
What's around me?

Environment maps, reflection maps, depth peeling, screen-space visibility (SSAO, SSR, etc.), sparse voxel octrees, stochastic transparency, billboard clouds, …

Epic Games | ILMxLAB | NVIDIA

THE RAYTRACING CHALLENGE

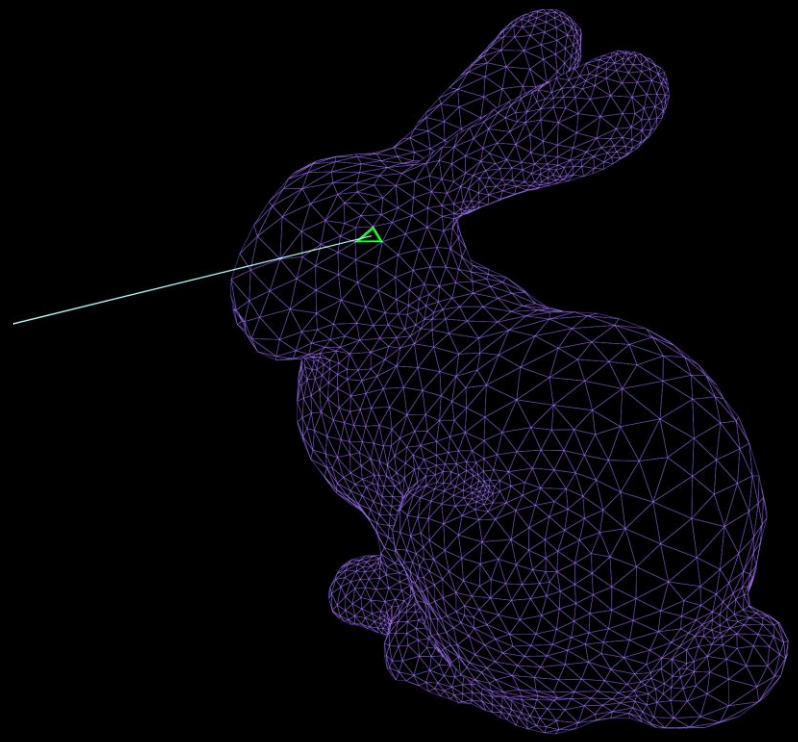
How to find the “needle” in the triangle data “haystack” ?



THE RAYTRACING CHALLENGE

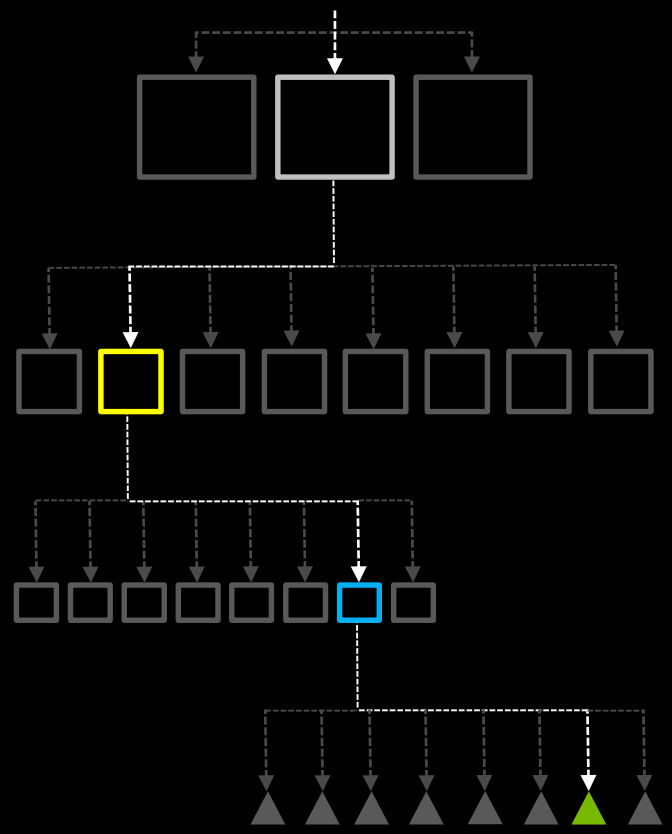
How to find the “needle” in the triangle data “haystack” ?

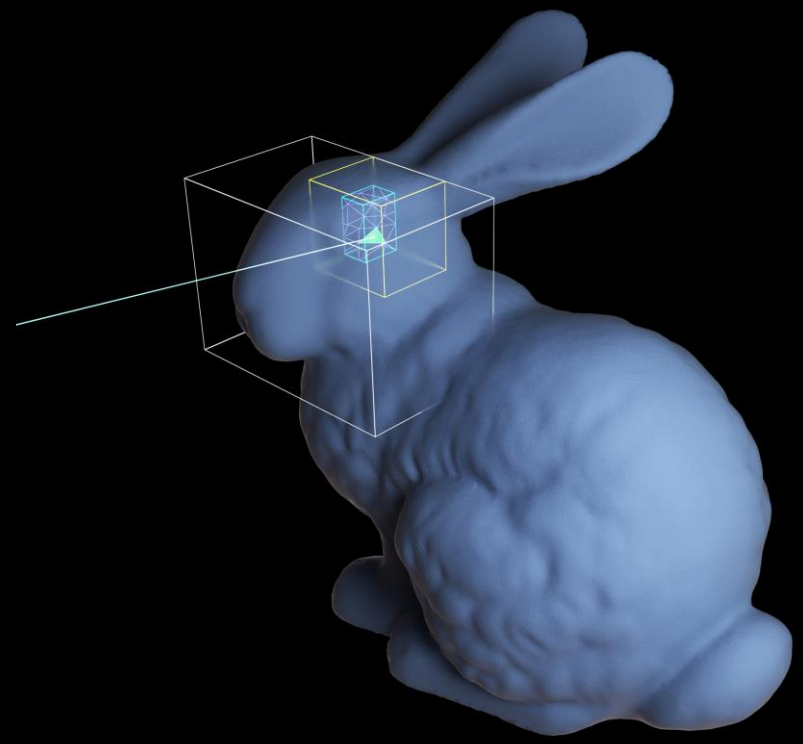
548.09	26.60	977.30	309.07	942.51	943.84	954.14	546.50	495.62
954.35	457.14	614.35	682.99	714.16	985.30	412.09	583.21	245.93
23.22	157.44	644.73	867.76	820.81	273.56	297.39	896.97	811.89
873.92	188.68	664.87	216.65	514.58	362.04	457.05	796.50	786.00
292.98	320.87	662.20	89.48	793.76	696.15	171.60	400.56	833.53
67.24	956.67	322.79	623.92	81.26	320.69	504.97	900.10	705.55
0.39	828.89	614.22	830.05	14.85	895.96	351.53	577.34	508.73
182.26	970.25	48.49	291.07	354.99	776.41	1.16	25.56	581.09
539.16	273.37	656.67	407.03	578.86	151.16	142.14	494.58	534.29
495.54	159.08	763.44	395.82	343.53	149.02	945.97	161.74	74.26
476.37	569.43	238.29	817.46	648.77	393.45	84.25	249.51	663.99
733.85	250.94	669.23	850.30	890.15	833.83	790.60	168.37	10.38
595.39	611.15	846.96	15.34	893.95	311.69	258.05	174.31	88.11
109.68	712.32	436.50	876.75	657.30	205.79	881.99	279.31	255.43
15.18	22.55	196.99	157.67	382.02	287.38	121.32	592.23	91.53
64.05	444.40	395.57	15.49	531.77	801.03	11.21	189.37	342.66
419.16	172.03	3.11	784.85	21.57	630.50	639.69	128.88	688.59
345.05	512.51	429.58	306.48	908.12	879.09	760.31	243.03	173.92
285.29	289.68	345.85	841.70	924.04	899.56	519.15	706.46	599.07
77.45	518.36	105.82	364.65	134.91	69.86	724.23	531.37	57.28
943.10	161.83	70.13	922.84	30.88	913.59	635.33	288.13	409.63
396.59	81.56	812.85	624.20	718.88	746.88	410.21	771.24	898.46
903.82	596.27	289.02	317.16	267.82	762.04	454.07	5.85	673.96
501.70	802.38	46.61	883.84	41.63	947.00	812.63	909.83	884.7



BVH ALGORITHM

Massive Improvement in Search Efficiency



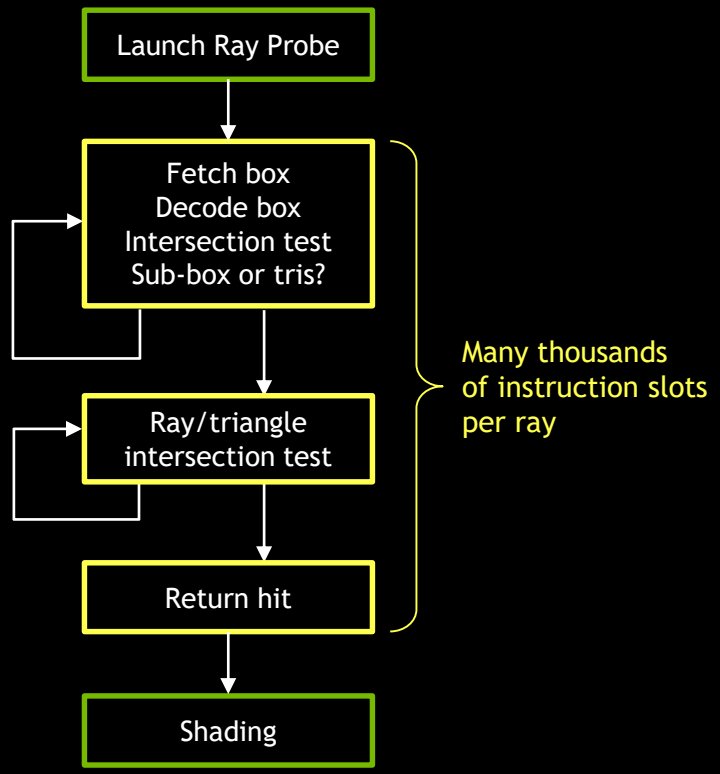


RAY TRACING BEFORE TURING

Software Emulation for BVH Search

Pascal SM

Shaders

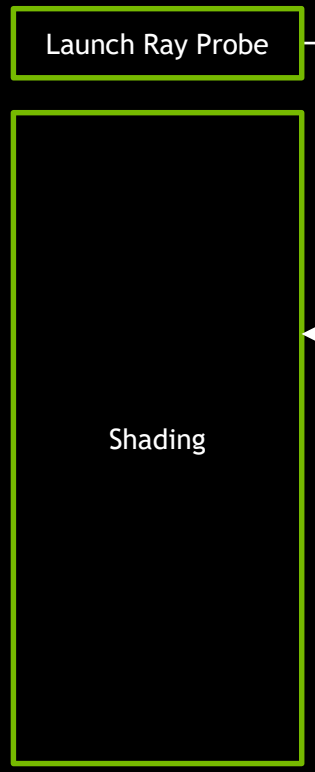


TURING RAYTRACING WITH RT CORES

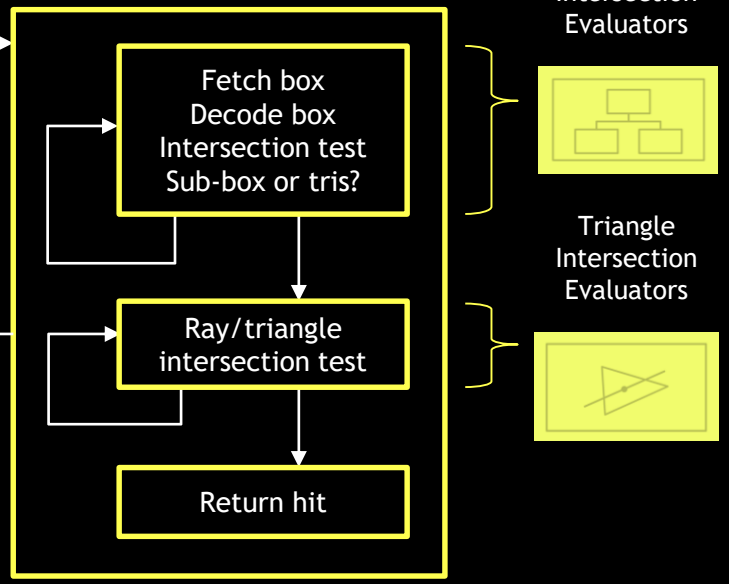
Hardware Acceleration Replaces Software Emulation

Turing SM

Shaders

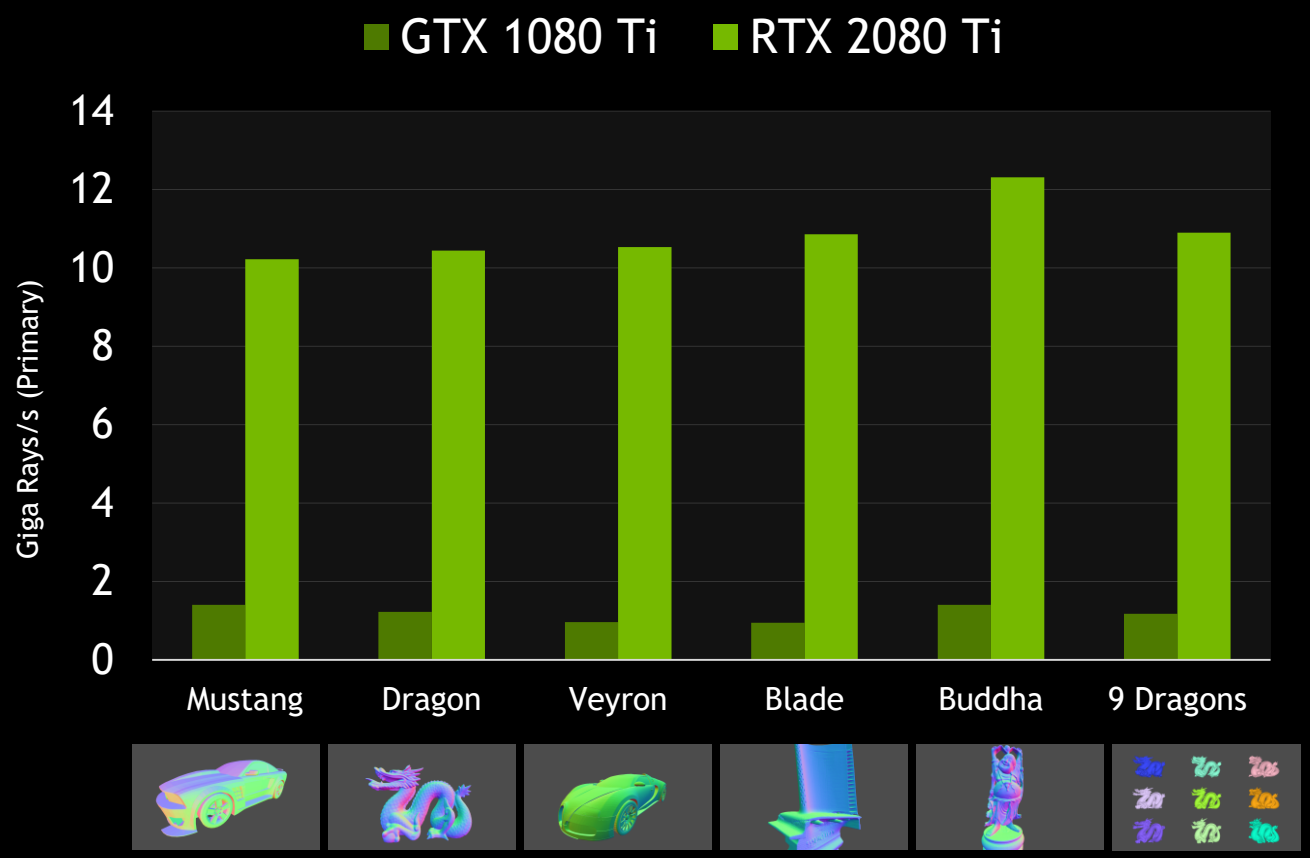


RT Core



TURING RAYTRACING PERFORMANCE

10 Giga Rays



GTX 1080 Ti	RTX 2080 Ti
11.3 TFLOPS	68 RT Cores
1.1 Giga Rays	10+ Giga Rays
10 TFLOPS / Giga Ray	~10X faster than 1080 Ti

RAYTRACING APIS

Applications

NVIDIA OPTIX™

Microsoft® DirectX®

Vulkan®

NVIDIA Driver with RTX Technology

RT Cores

NVIDIA Turing GPU

DXR API DESIGN

A First Class Member of the Real-Time Graphics Pipeline

Tight integration with raster and compute

Following the existing low-level design of DX12

All resources shared seamlessly with raster/compute

HLSL as shading language, extended with raytracing operations

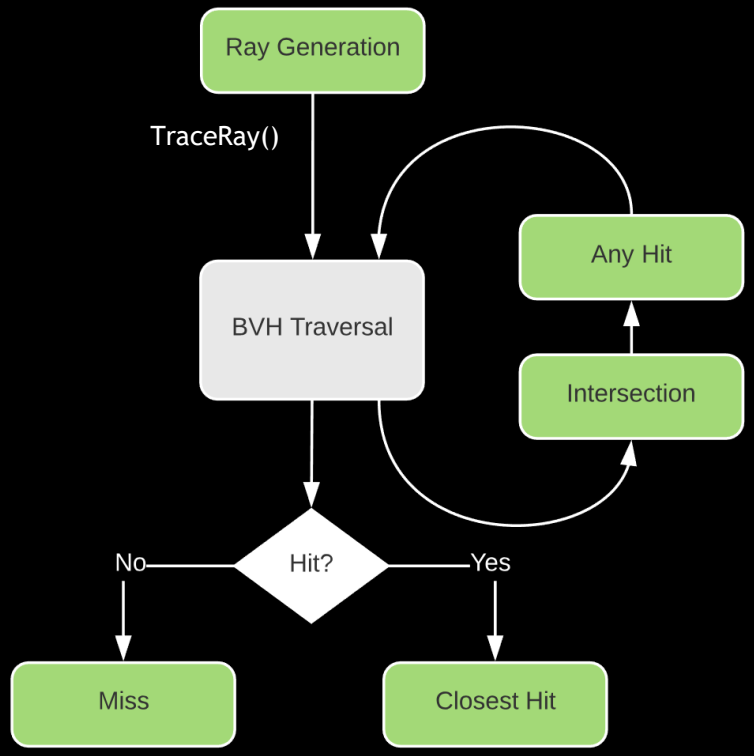
THE RAY TRACING PIPELINE

New shader types:

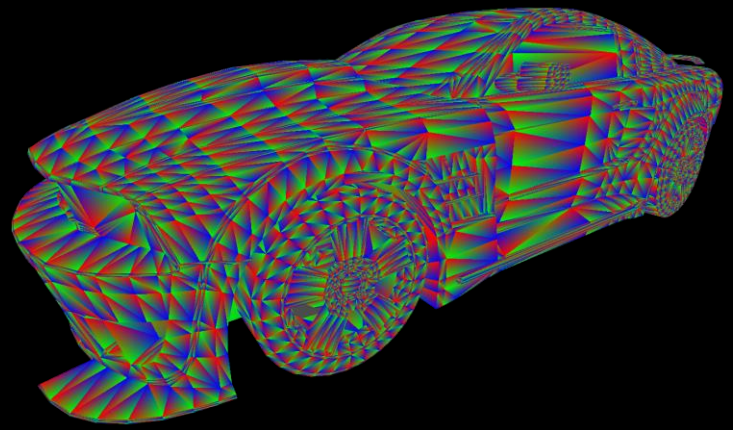
- Ray Generation: like CS, but can TraceRay()
- Intersection: custom primitives
- Any Hit: transparency
- Closest Hit: typically material shading

Single-ray model with recursion support

VSPS to auto-generate RT shaders from raster



```
struct Payload {
    float4 color;
};

RaytracingAccelerationStructure scene : register(t5);

[shader("raygeneration")]
void RayGenMain()
{
    uint2 launchIndex = DispatchRaysIndex();

    RayDesc ray;
    ray.Origin = SceneConstants.cameraPosition;
    ray.Direction = computeRayDirection( launchIndex );
    ray.TMin = 0;
    ray.TMax = 100000;

    Payload payload;
    TraceRay( scene, 0 /*flags*/, 0xFF /*mask*/, 0, 1, 0, ray, payload );

    outputTexture[launchIndex.xy] = payload.color;
}

[shader("closesthit")]
void ClosestHitMain( inout Payload payload, in Attributes attr )
{
    payload.color = float4( attr.barys.x, attr.barys.y, 1 - attr.barys.x - attr.barys.y, 1 );
}
```



DIRECTX & VULKAN

Raytracing APIs for Real-Time Applications

DirectX Raytracing (DXR)

- Final API out now! http://forums.directxtech.com
- OS support in Windows 10 RS5 (October Update)
- Full Turing/RTX support with NVIDIA Driver 416.16

Vulkan Raytracing Extensions

- Same abstraction level and functionality as DXR
- NVX_raytracing: shipping
- NV_raytracing: available shortly

RAYTRACING AT LOW SAMPLE COUNT

Hitting Realtime Realities

Even with Turing, we only have a budget of a few rays per pixel in real-time

Go hybrid: use rays where they matter most

Denoise!

- AI or traditional
- Many variants, lots of recent research
- See Edward Liu’s talks

1SPP RAY TRACED SHADOWS

1SPP RAY TRACED SHADOWS + DENOISING

RAY TRACED SHADOW GROUND TRUTH

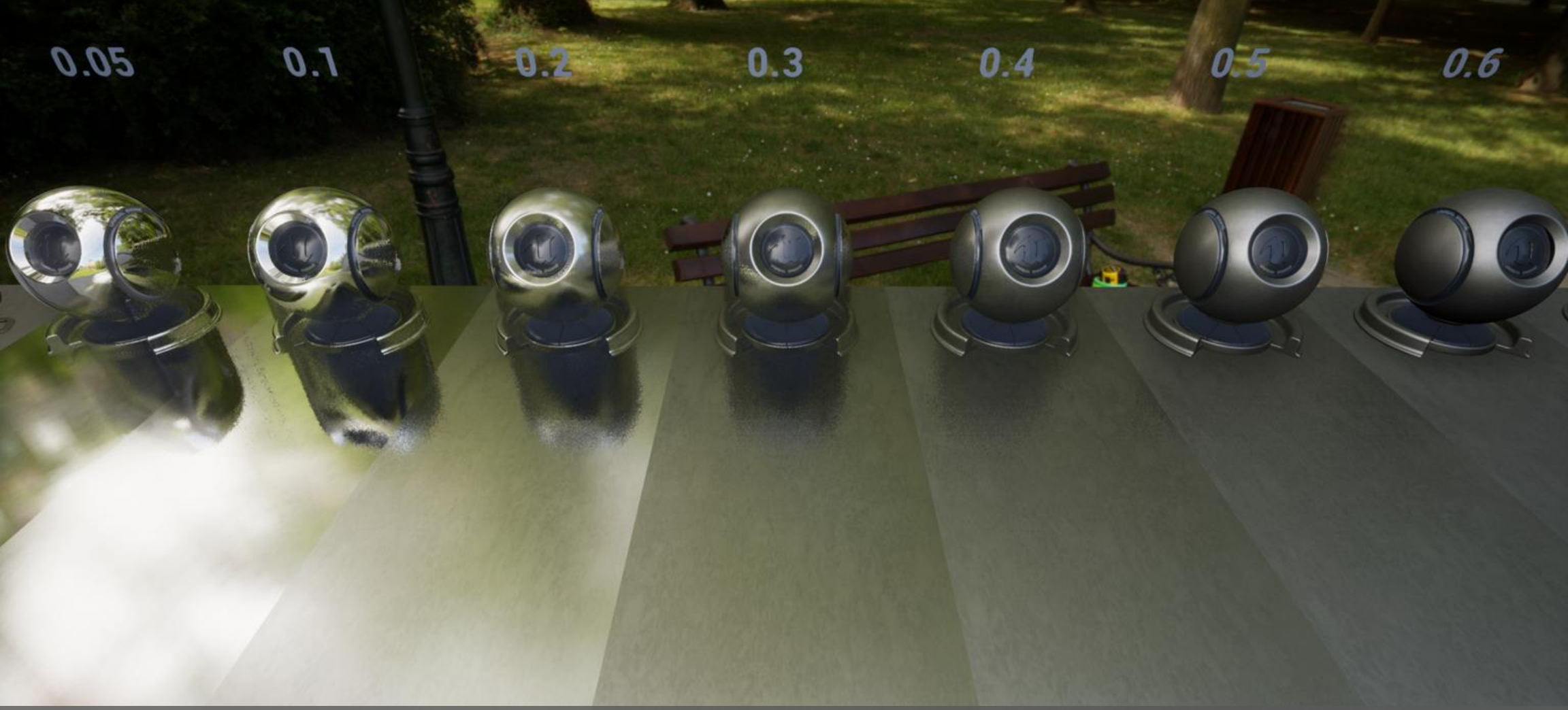
SHADOW MAPPING

SHADOW OF THE
TOMB RAIDER

1SPP RAY TRACED REFLECTIONS

1SPP RAY TRACED REFLECTIONS + DENOISING

GROUND TRUTH



STOCHASTIC SSR + REFLECTION CAPTURES

BATTLEFIELD V
NVIDIA
GEFORCE
RTX
DICE
EA

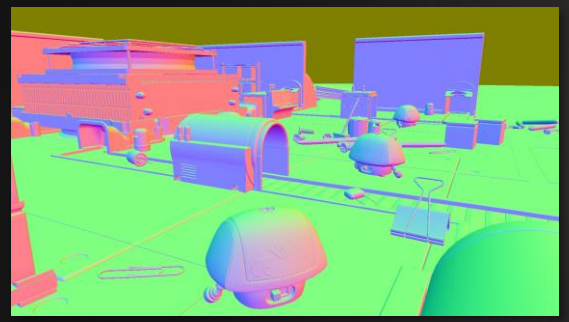
BATTLEFIELD V
NVIDIA
GEFORCE
RTX
EA

NVIDIA
GEFORCE
RTX

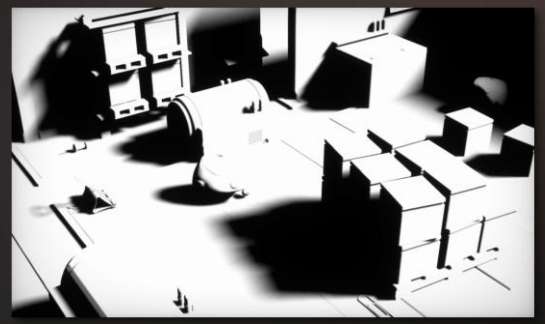
BATTLEFIELD V
NVIDIA
GEFORCE
RTX
DICE
EA

EA SEED Project PICA PICA
SEED

Hybrid Rendering Pipeline



Deferred shading
(raster)

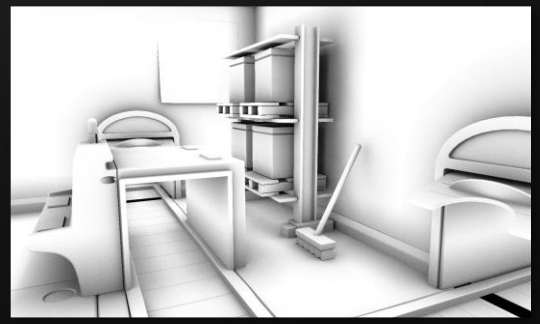


Direct shadows
(raytrace or raster)

Lighting
(compute + raytrace)

Reflections
(raytrace or compute)

Global Illumination
(compute and raytrace)

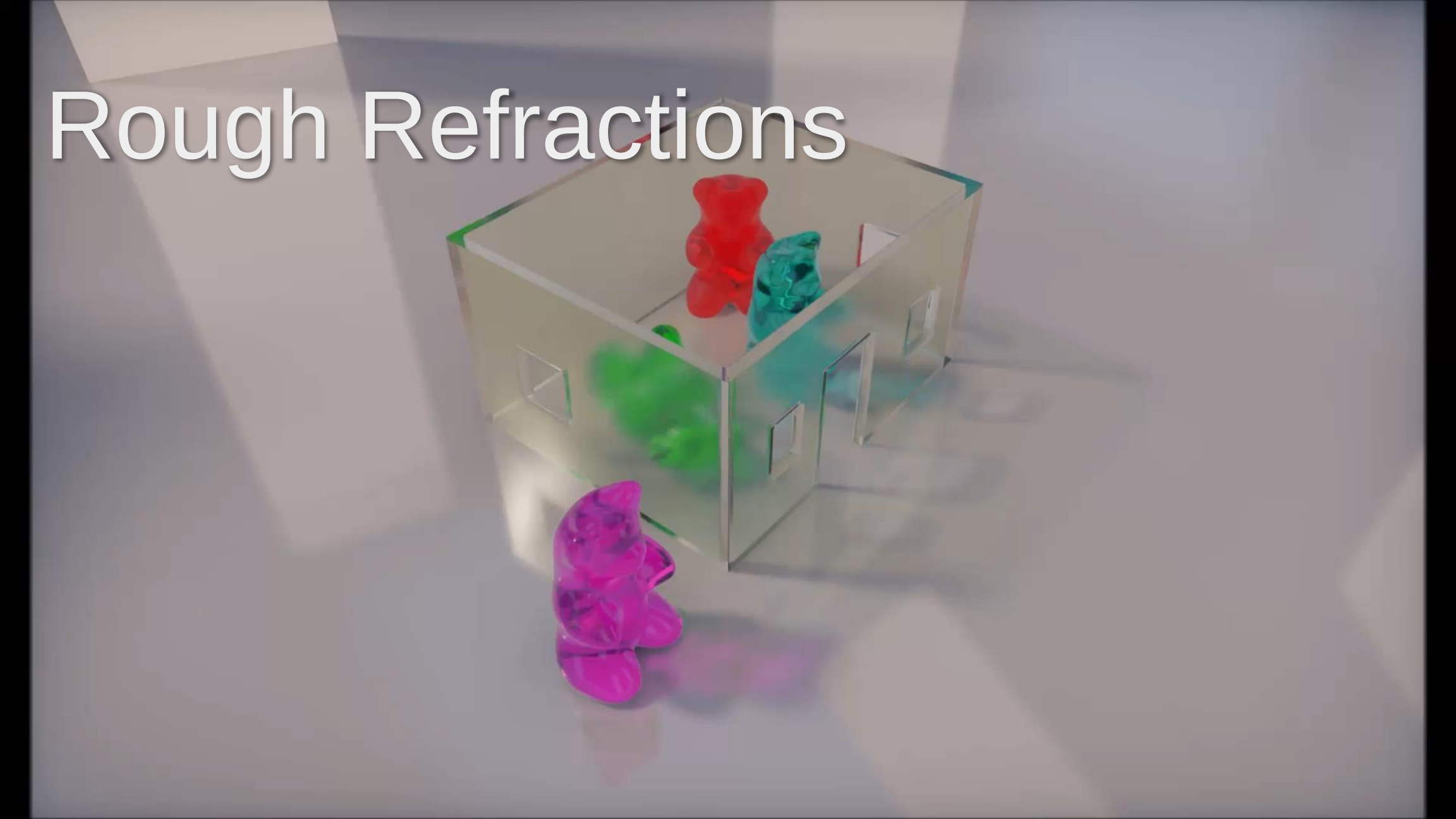


Ambient occlusion
(raytrace or compute)

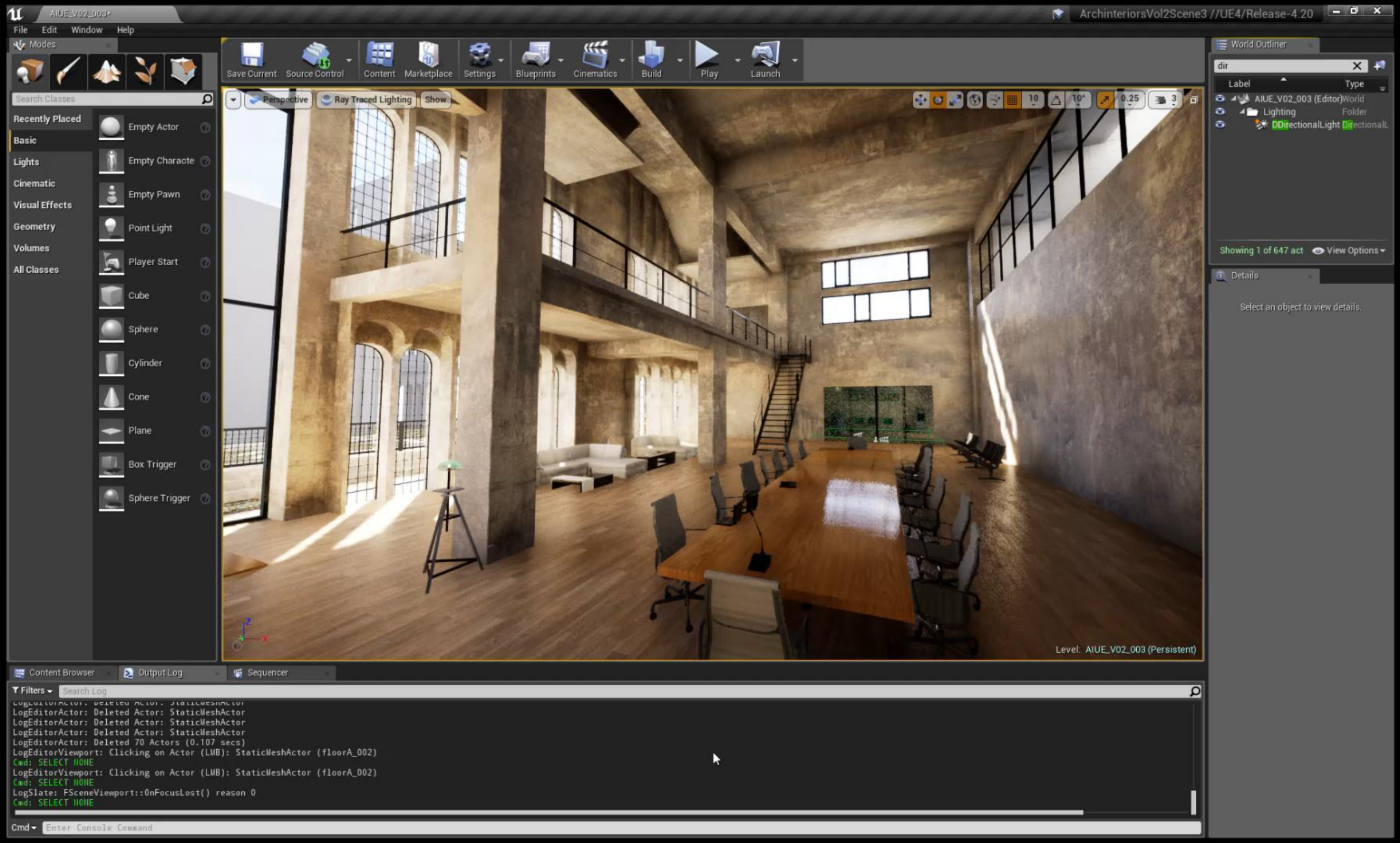
Transparency &
Translucency
(raytrace and compute)

Post processing
(compute)

Rough Refractions



Real-Time Lightmap Baking


AIUE_V02_003*
ArchinteriorsVol2Scene3 //UE4/Release-4.20
File
Edit
Window
Help
Modes
Save Current
Source Control
Content
Marketplace
Settings
Blueprints
Cinematics
Build
Play
Launch
Search Classes
Recently Placed
Basic
Lights
Cinematic
Visual Effects
Geometry
Volumes
All Classes
Empty Actor
Empty Characte
Empty Pawn
Point Light
Player Start
Cube
Sphere
Cylinder
Cone
Plane
Box Trigger
Sphere Trigger
Perspective
Ray Traced Lighting
Show
Level: AIUE_V02_003 (Persistent)
World Outliner
dir
Label
Type
AIUE_V02_003 (Editor)
World
Lighting
Folder
DDirectionalLight
DirectionalL
Showing 1 of 647 act
View Options
Details
Select an object to view details.
Content Browser
Output Log
Sequencer
Filters
Search Log
LogEditorActor: Deleted Actor: StaticMeshActor
LogEditorActor: Deleted Actor: StaticMeshActor
LogEditorActor: Deleted Actor: StaticMeshActor
LogEditorActor: Deleted 70 Actors (0.107 secs)
LogEditorViewport: Clicking on Actor (LMB): StaticMeshActor (floorA_002)
Cmd: SELECT NONE
LogEditorViewport: Clicking on Actor (LMB): StaticMeshActor (floorA_002)
Cmd: SELECT NONE
LogSlate: FSceneViewport::OnFocusLost() reason 0
Cmd: SELECT NONE
Cmd
Enter Console Command

UPCOMING RTX RAYTRACING TITLES

"DXR is one of the fastest adopted features that we've released in recent years." -Microsoft DirectX Team

Assetto Corsa Competizione from Kunos Simulazioni/505 Games

Atomic Heart from Mundfish

Battlefield V from *EA*/*DICE*

Control from Remedy Entertainment/505 Games

Enlisted from Gaijin Entertainment/Darkflow Software

Justice from NetEase

JX3 from Kingsoft

MechWarrior 5: Mercenaries from Piranha Games

Metro Exodus from 4A Games

ProjectDH from Nexon's devCAT Studio

Shadow of the Tomb Raider from Square Enix/Eidos-Montréal/Crystal Dynamics/Nixxes

THANK YOU!

SPECIAL THANKS

NVIDIA RTX, Driver, Arch, Devtech, Demo, and Research teams
Microsoft DirectX team
SEED, EA, Nixxes
Slide Content: Colin Barré-Brisebois, Edward Liu, Jacopo Pantaleoni, Brandon Lloyd, Steve Parker, Morgan McGuire, Chris Wyman, Matt Pharr

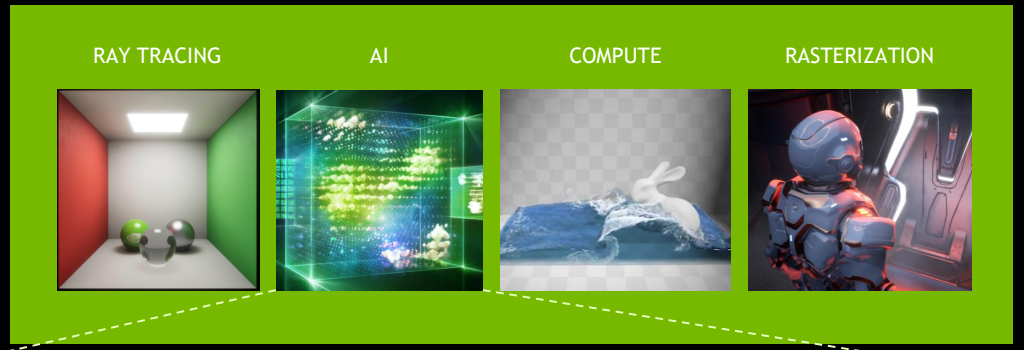
NVIDIA®

BACKUP

PICA PICA: https://www.youtube.com/watch?v=LXo0WdlELJk

EVENT COVERAGE

RTX: GRAPHICS REINVENTED



ANISOTROPIC SPATIAL KERNEL

NVIDIA DENOISERS

GameWorks Ray Tracing consists of a library of denoising techniques essential to increasing the speed and quality of real-time raytracing of:

Ray Traced Glossy Reflections

Offscreen reflections
Inter-object reflections
Mirror to Glossy

Ray Traced Shadows

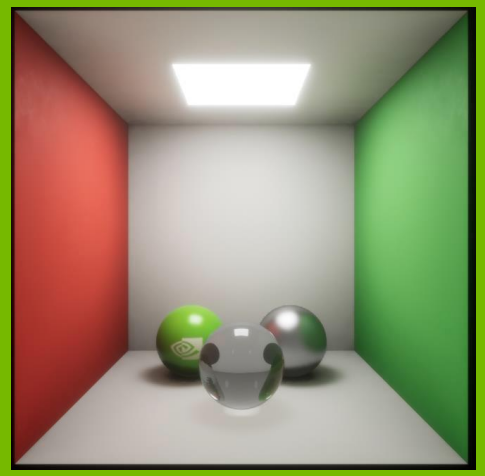
Spherical, rectangular and directional cone lights
Contact Hardening
Soft Shadows

Ray Traced Ambient Occlusion

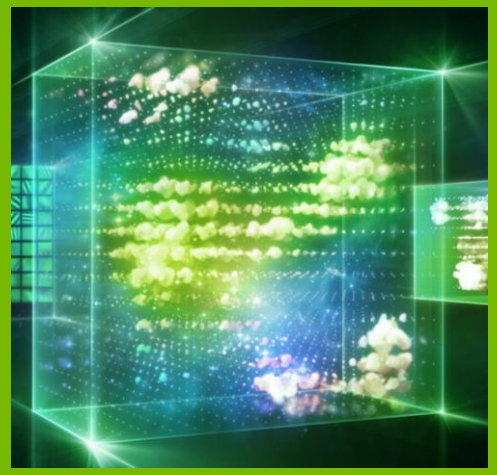
High quality contact hardening
Support for offscreen objects

NVIDIA RTX

RAY TRACING



AI



COMPUTE

RASTERIZATION

DXR SCENE REPRESENTATION

Acceleration Structures

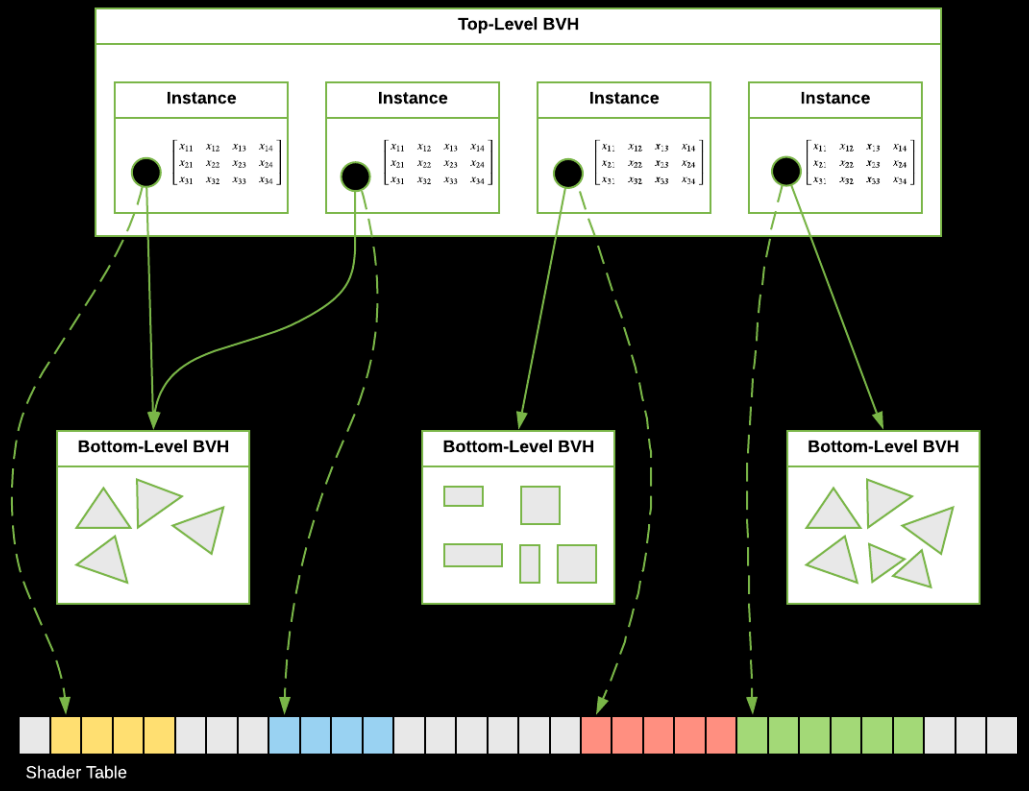
Two level hierarchy with instancing

- Instance = pointer to bottom-level BVH + pointer into shader table

BVHs are resources holding opaque data

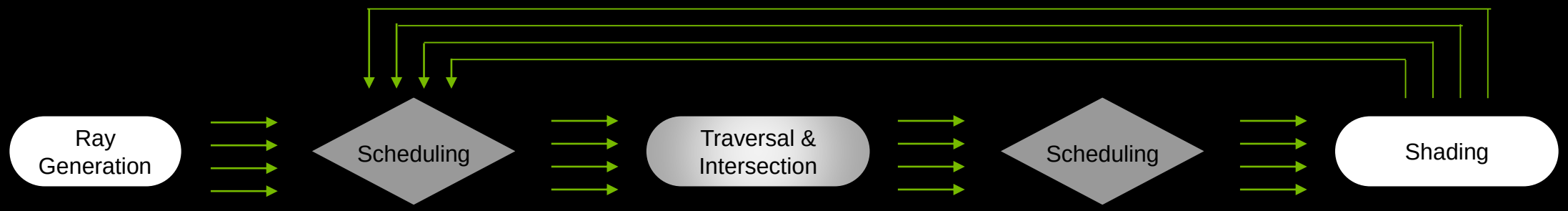
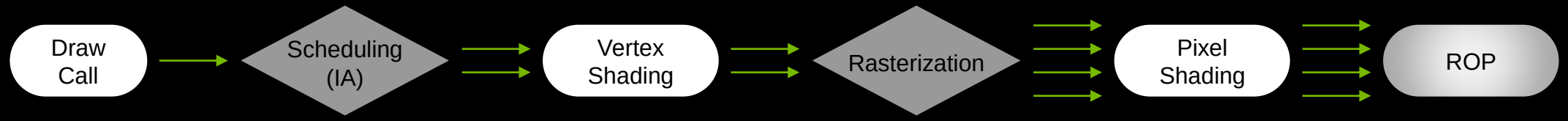
Materials defined in shader table

- Also just a memory resource, completely app-managed



HIGH LEVEL PIPELINES

Ray Tracing



Raster