NVIDIA

A FAST FORWARD THROUGH *RAY TRACING GEMS*

Eric Haines, Distinguished Engineer | March 20, 2019 | Booth S466 | South Hall

There is an old joke that goes, "Ray tracing is the technology of the future, and it always will be!"

– David Kirk, March 2008

RAY TRACING GEMS

http://raytracinggems.com -
Table of Contents, links, and what this talk is mostly about.

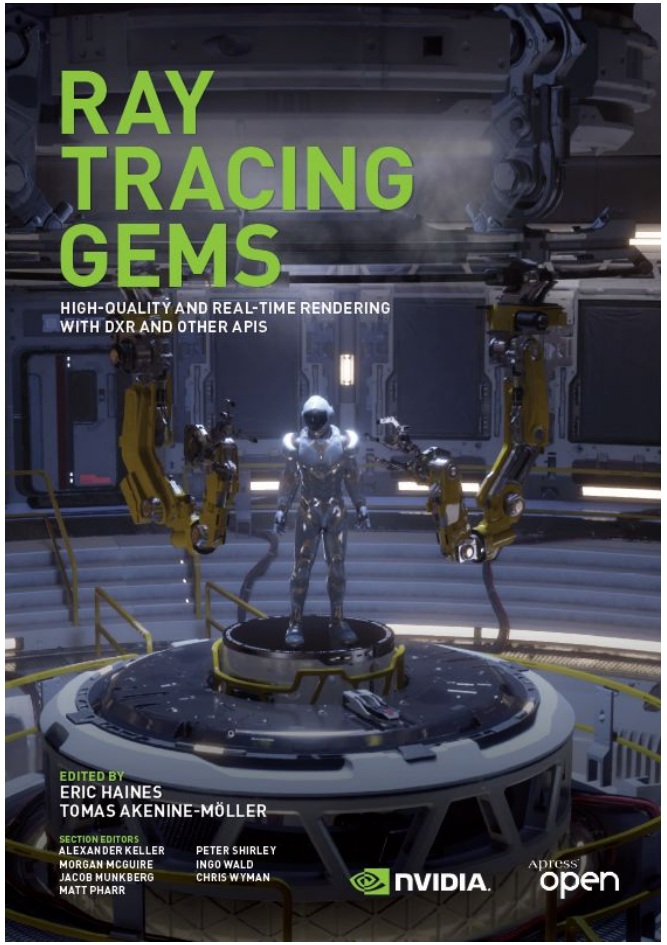
Proposed by Tomas Akenine-Möller, co-editor, in Spring 2018.

Like other “Gems” books: provide tools and case studies.

32 papers accepted, 64 authors, 652 pages.

Tight schedule: papers received October 15, finished book proof done February 12. 121 days.

Open Access! Articles can be freely redistributed (non-commercial, and attribute the source).

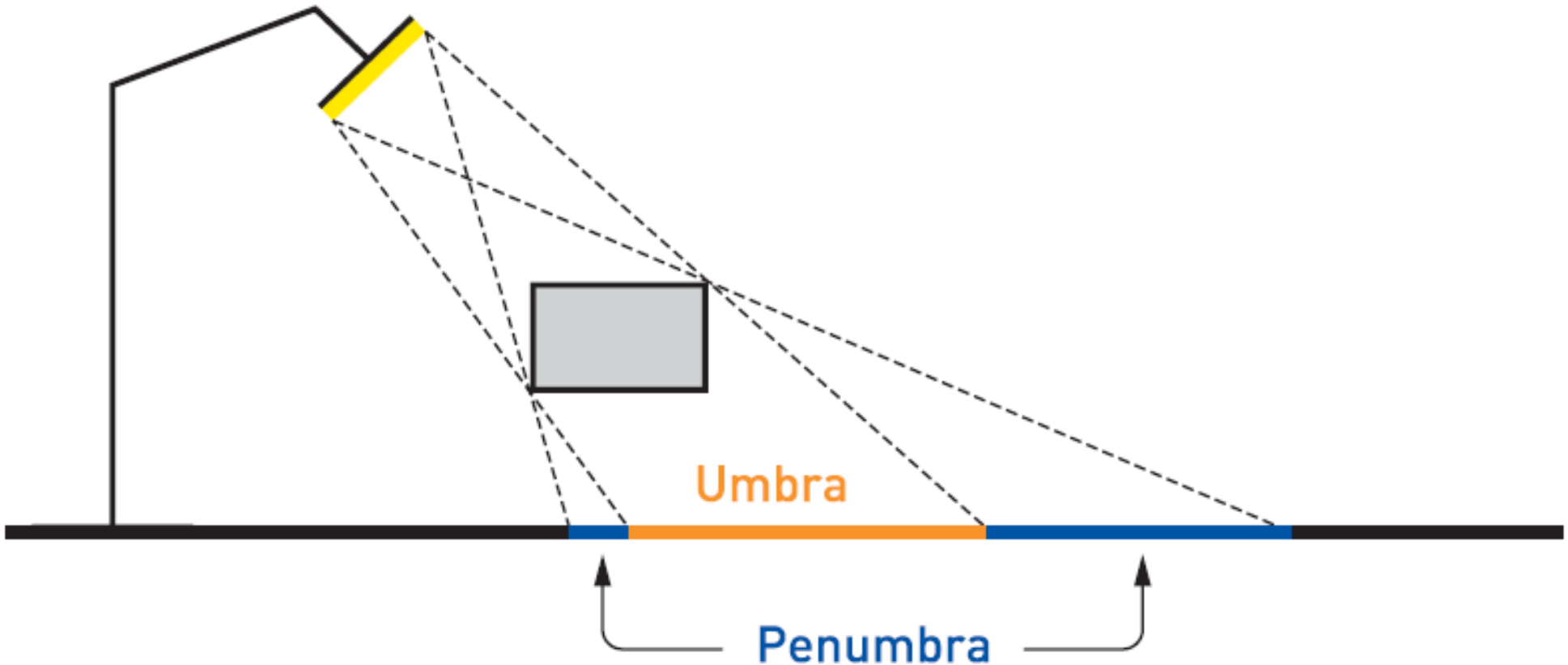


PART I
RAY TRACING BASICS

Edited by Chris Wyman

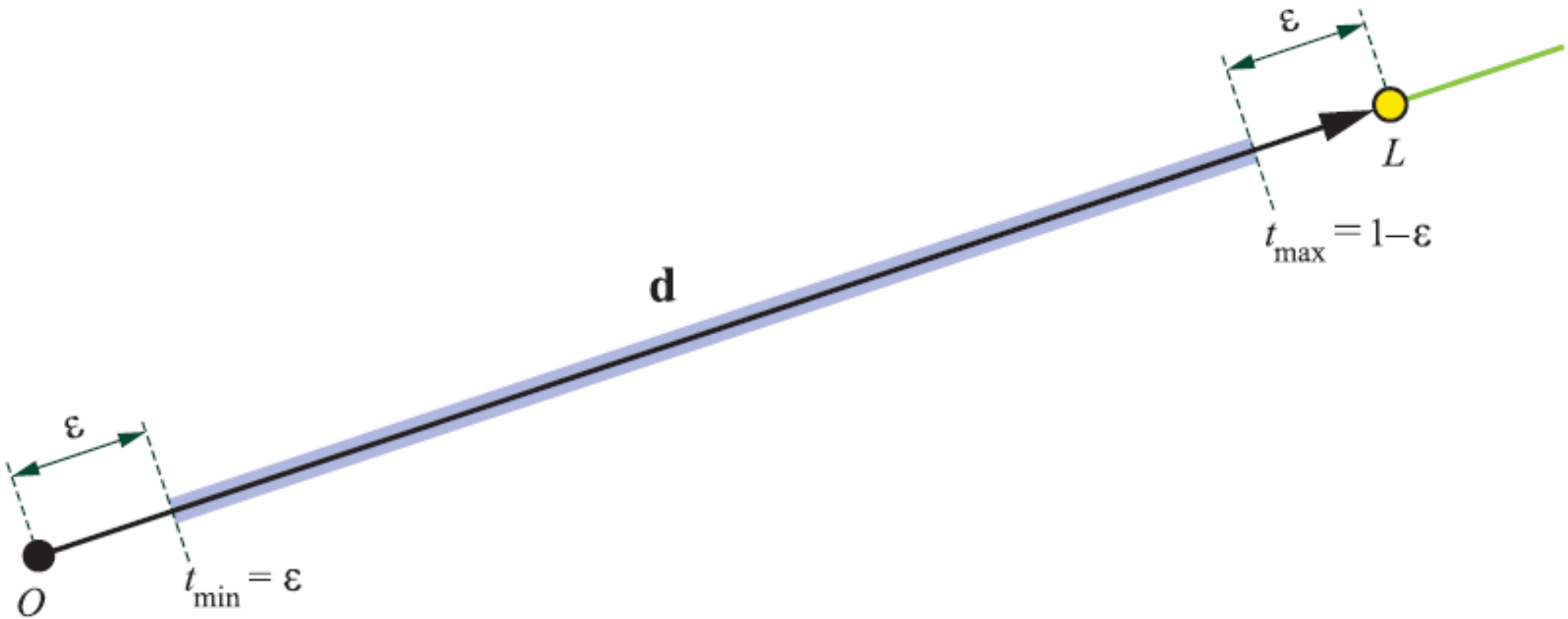
RAY TRACING TERMINOLOGY

by Eric Haines and Peter Shirley



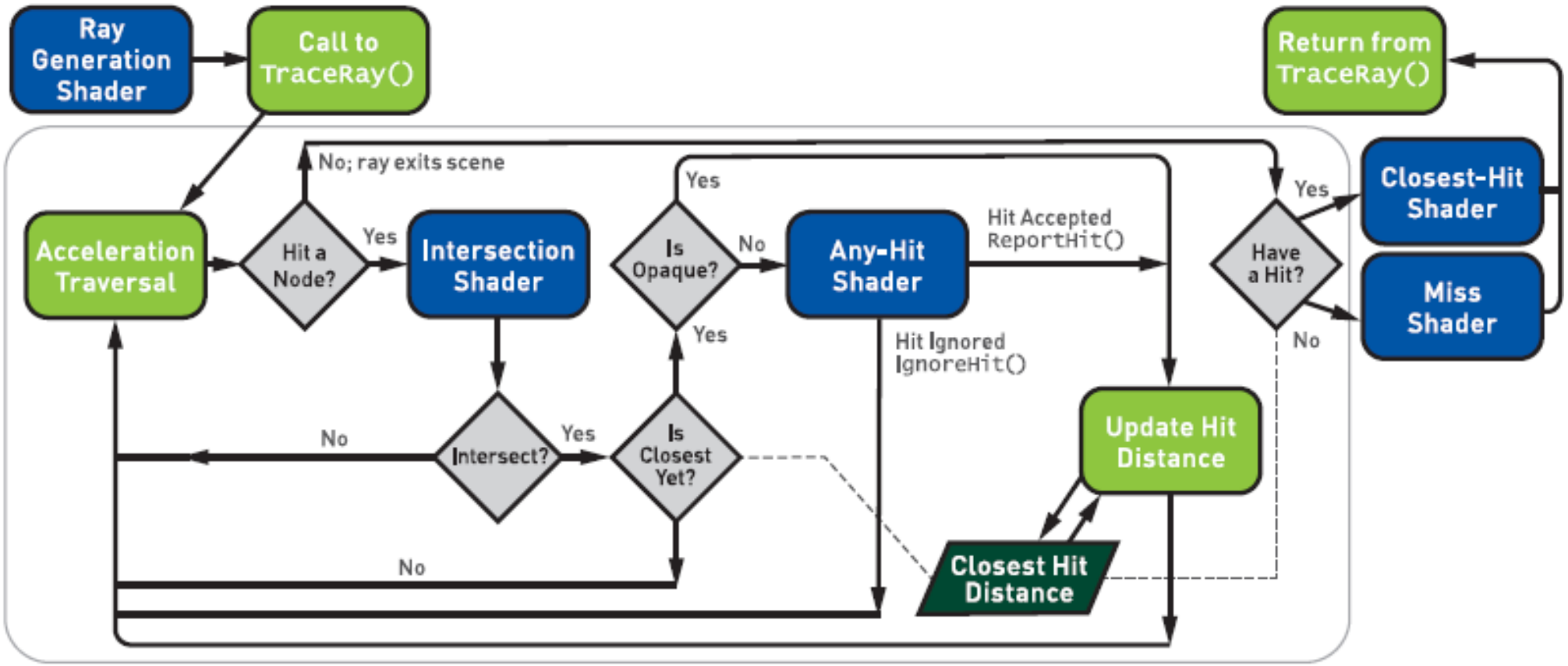
WHAT IS A RAY?

by Peter Shirley, Ingo Wald, Tomas Akenine-Möller, and Eric Haines



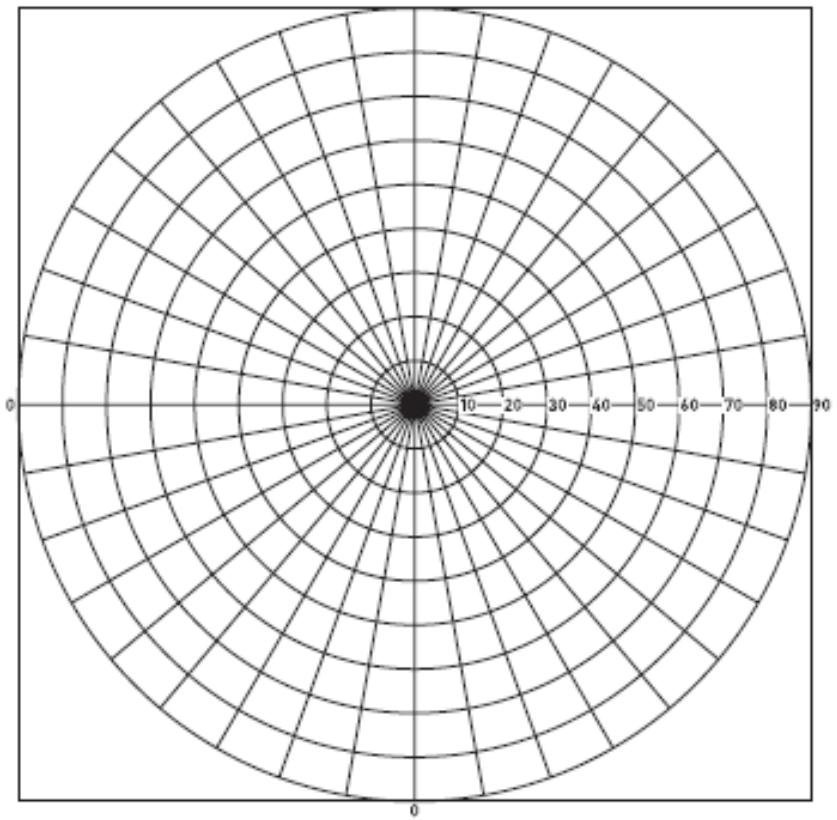
INTRODUCTION TO DIRECTX RAYTRACING

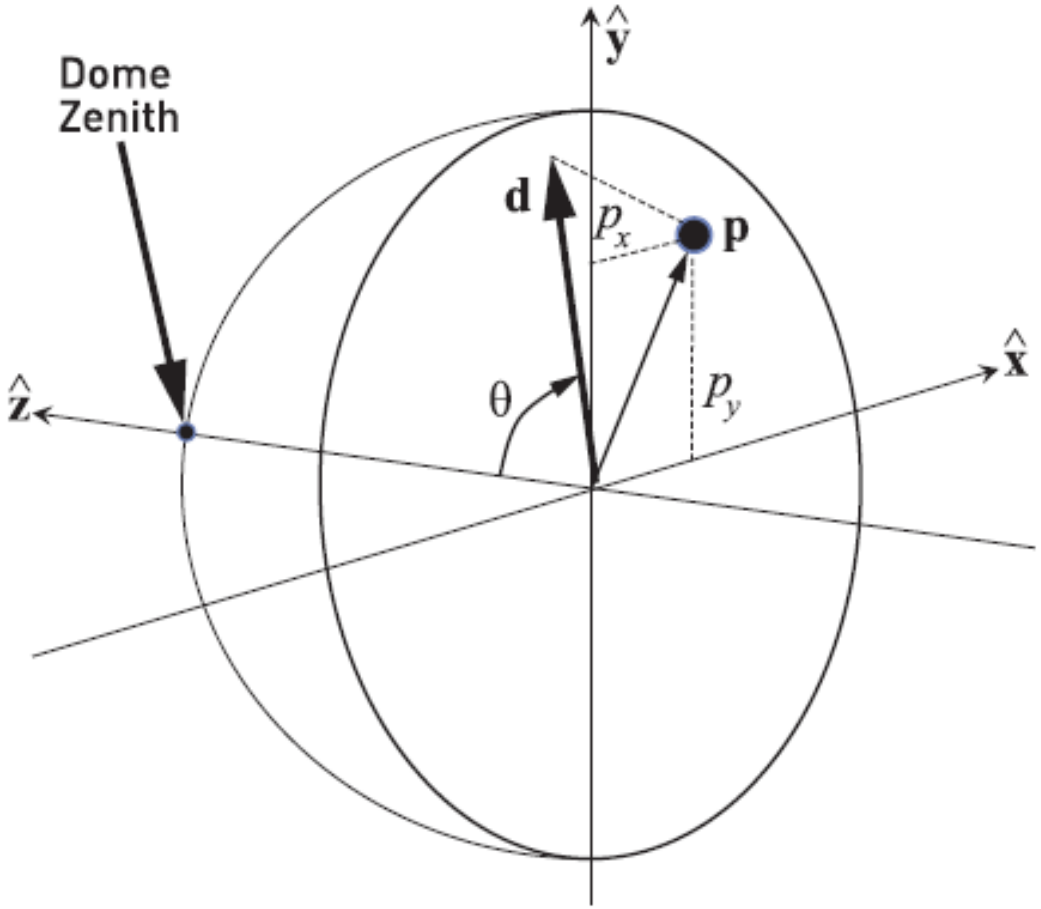
by Chris Wyman and Adam Marrs



A PLANETARIUM DOME MASTER CAMERA

by John E. Stone

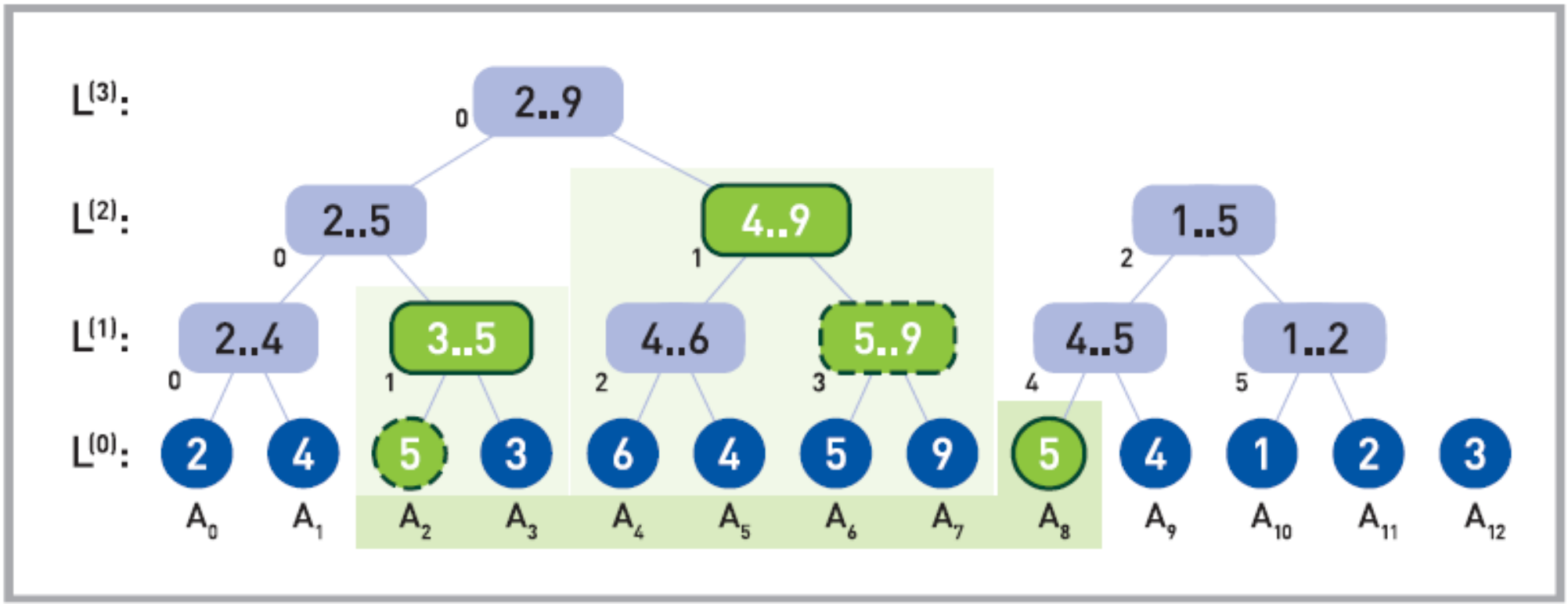




Used in "The Birth of Planet Earth"

COMPUTING MINIMA AND MAXIMA OF SUBARRAYS

by Ingo Wald

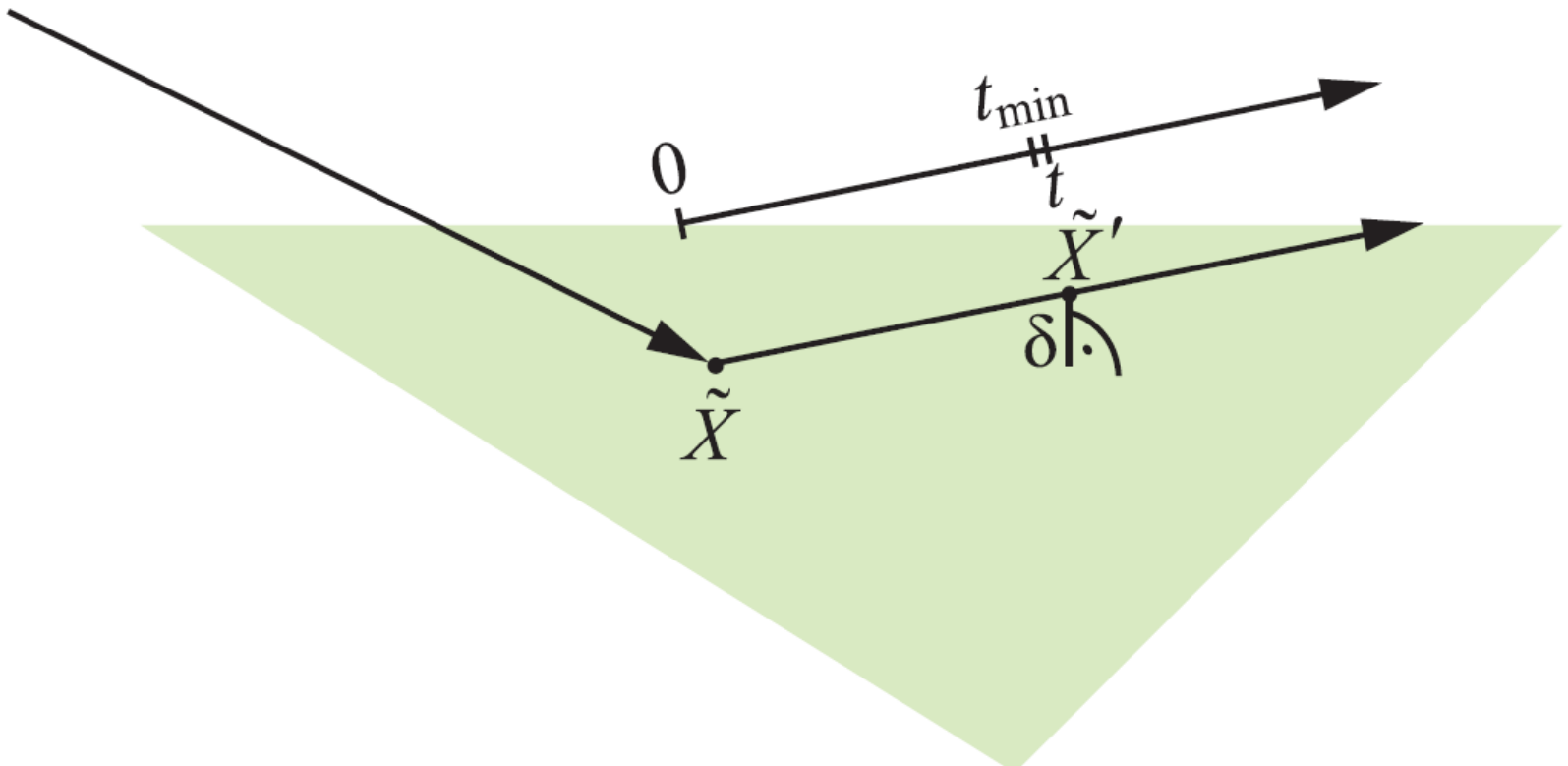


PART II
INTERSECTIONS AND EFFICIENCY

Edited by Ingo Wald

A FAST AND ROBUST METHOD FOR AVOIDING SELF-INTERSECTION

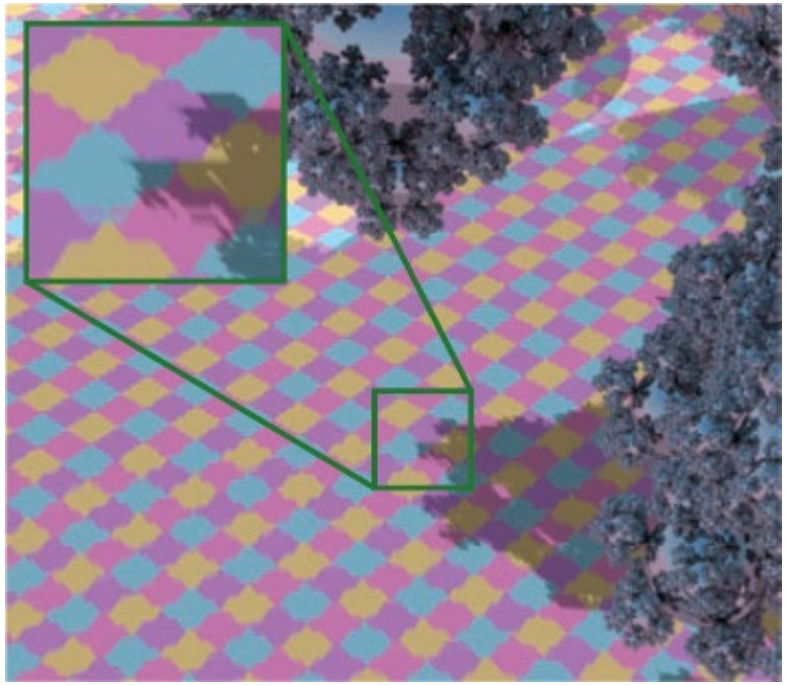
by Carsten Wächter and Nikolaus Binder



PRECISION IMPROVEMENTS FOR RAY/SPHERE INTERSECTION

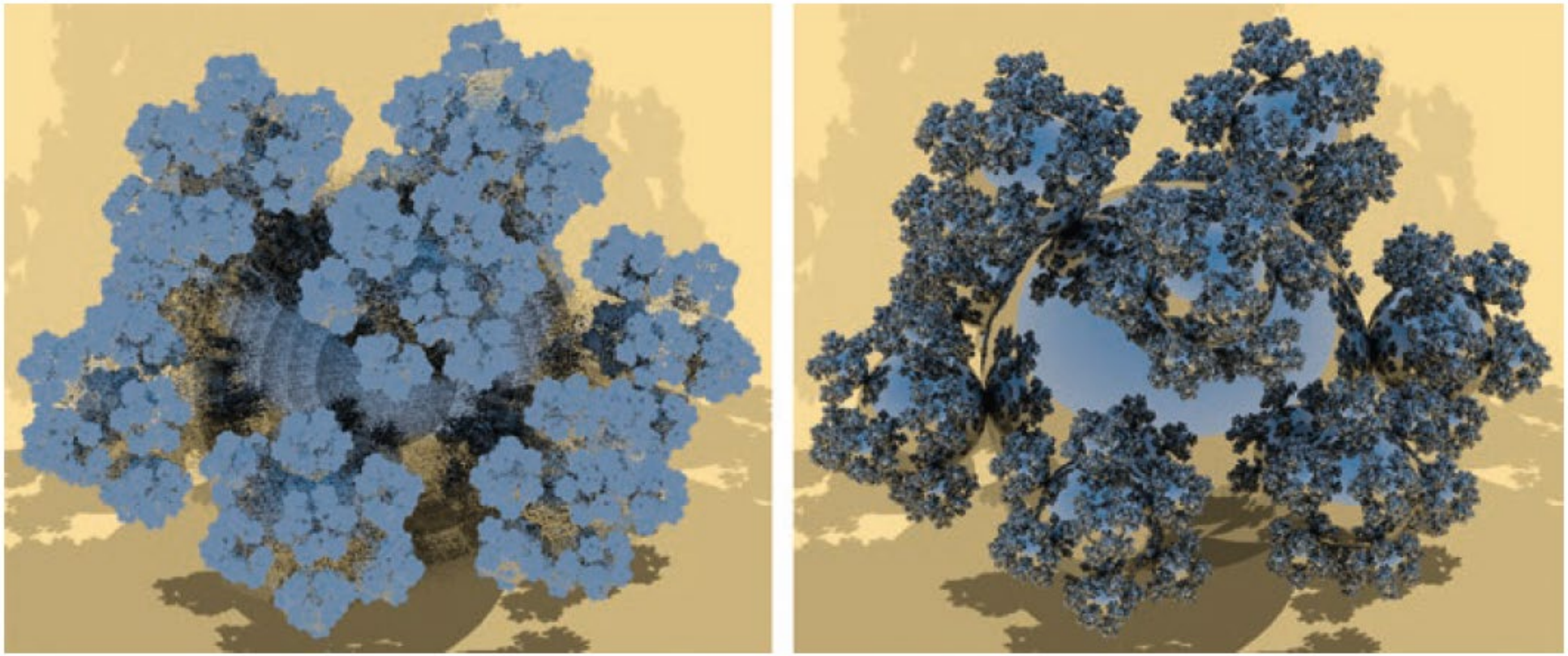
by Eric Haines, Johannes Günther, and Tomas Akenine-Möller

With a trembling arm shoot an arrow at a coin - so are ray and sphere.



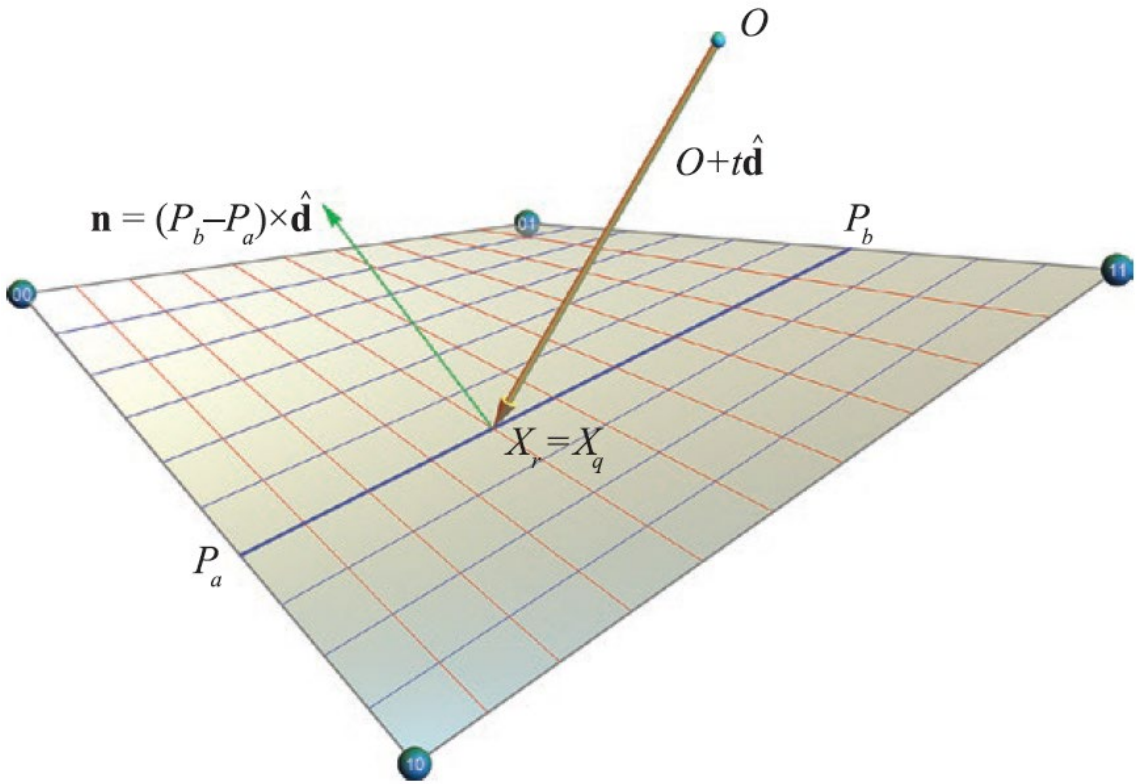
PRECISION IMPROVEMENTS FOR RAY/SPHERE INTERSECTION

by Eric Haines, Johannes Günther, and Tomas Akenine-Möller



COOL PATCHES: A GEOMETRIC APPROACH TO RAY/BILINEAR PATCH INTERSECTIONS

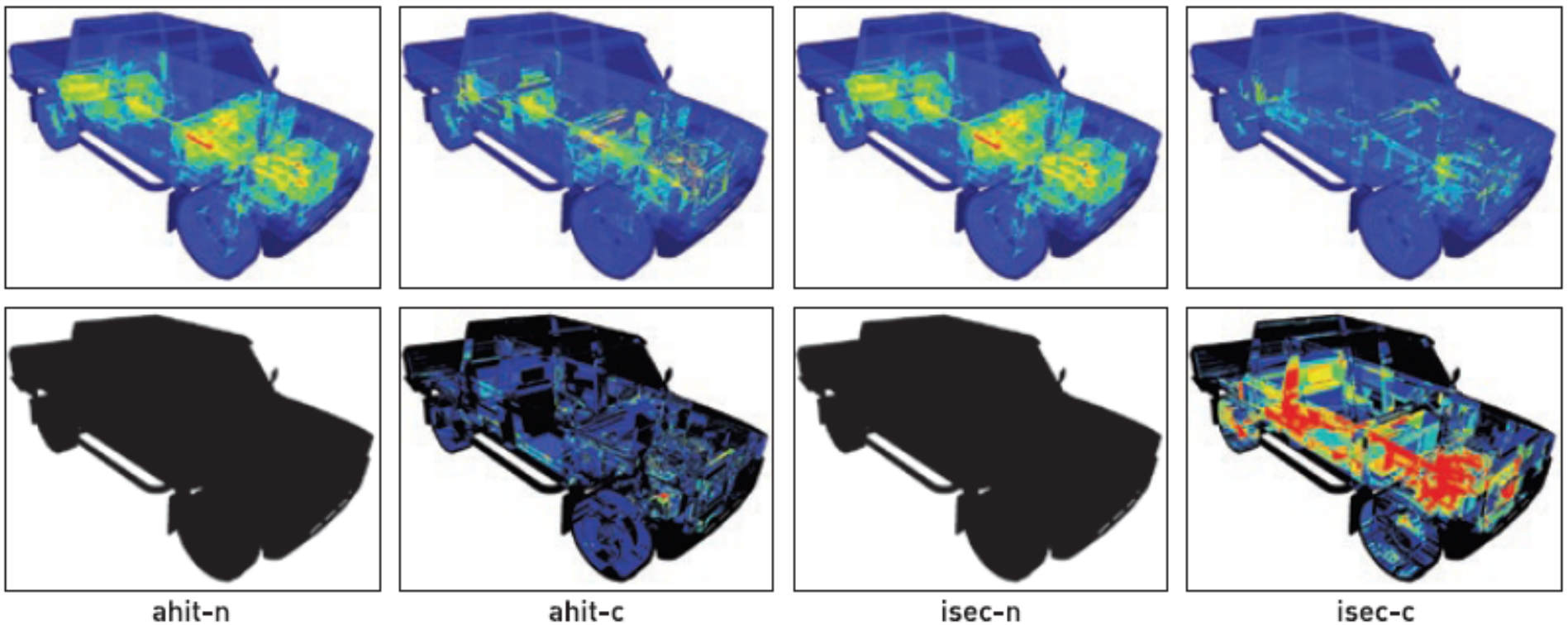
by Alexander Reshetov




```
1 RT_PROGRAM void intersectPatch(int prim_idx) {
2   // ray is rtDeclareVariable(Ray, ray, rtCurrentRay,) in OptiX
3   // patchdata is optix::rtBuffer
4   const PatchData& patch = patchdata[prim_idx];
5   const float3* q = patch.coefficients();
6   // 4 corners + "normal" qn
7   float3 q00 = q[0], q10 = q[1], q11 = q[2], q01 = q[3];
8   float3 e10 = q10 - q00; // q01 ----------- q11
9   float3 e11 = q11 - q10; // |                 |
10  float3 e00 = q01 - q00; // | e00         e11 |  we precompute
11  float3 qn = q[4];       // |       e10       |  qn = cross(q10-q00,
12  q00 -= ray.origin;      // q00 ----------- q10             q01-q11)
13  q10 -= ray.origin;
14  float a = dot(cross(q00, ray.direction), e00); //  the  equation  is
15  float c = dot(qn, ray.direction);               // a + b u + c u^2
16  float b = dot(cross(q10, ray.direction), e11); // first compute
17  b -= a + c;                                     // a+b+c and then b
18  float det = b*b - 4*a*c;
19  if (det < 0) return;      // see the right part of Figure 5
20  det = sqrt(det);          // we -use_fast_math in CUDA_NVRTC_OPTIONS
21  float u1, u2;             // two roots(u parameter)
22  float t = ray.tmax, u, v;  // need solution for the smallest t > 0
23  if (c == 0) {                          // if c == 0, it is a trapezoid
24    u1 = -a/b; u2 = -1;                  // and there is only one root
25  } else {                               // (c != 0 in Stanford models)
26    u1 = (-b - copysignf(det, b))/2;     // numerically "stable" root
27    u2 = a/u1;                           // Viete's formula for u1*u2
28    u1 /= c;
29  }
30  if (0 <= u1 && u1 <= 1) {               // is it inside the patch?
31    float3 pa = lerp(q00, q10, u1);       // point on edge e10 (Fig. 4)
32    float3 pb = lerp(e00, e11, u1);       // it is, actually, pb - pa
33    float3 n = cross(ray.direction, pb);
34    det = dot(n, n);
35    n = cross(n, pa);
36    float t1 = dot(n, pb);
37    float v1 = dot(n, ray.direction);     // no need to check t1 < t
38    if (t1 > 0 && 0 <= v1 && v1 <= det) {  // if t1 > ray.tmax,
39        t = t1/det; u = u1; v = v1/det;    // it will be rejected
40    }                                      // in rtPotentialIntersection
41  }
```

MULTI-HIT RAY TRACING IN DXR

by Christiaan Gribble



A SIMPLE LOAD-BALANCING SCHEME WITH HIGH SCALING EFFICIENCY

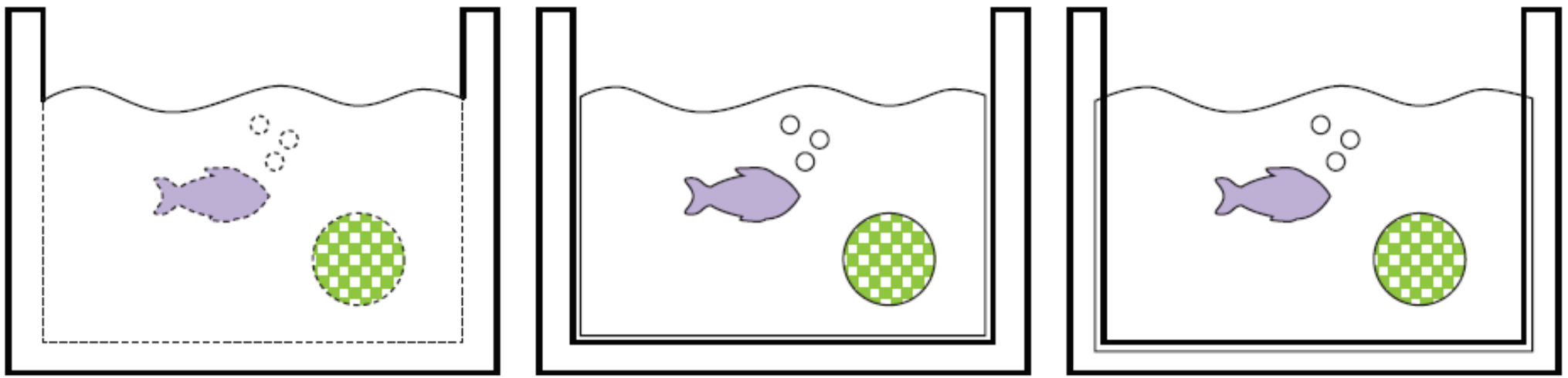
by Dietger van Antwerpen, Daniel Seibert, and Alexander Keller

PART III
REFLECTIONS, REFRACTIONS, AND SHADOWS

Edited by Peter Shirley

AUTOMATIC HANDLING OF MATERIALS IN NESTED VOLUMES

by Carsten Wächter and Matthias Raab



A MICROFACET-BASED SHADOWING FUNCTION TO SOLVE THE BUMP TERMINATOR PROBLEM

by Alejandro Conty Estevez, Pascal Lecocq, and Clifford Stein

RAY TRACED SHADOWS: MAINTAINING REAL-TIME FRAME RATES

by Jakub Boksansky, Michael Wimmer, and Jiri Bittner

RAY-GUIDED VOLUMETRIC WATER CAUSTICS IN SINGLE SCATTERING MEDIA WITH DXR

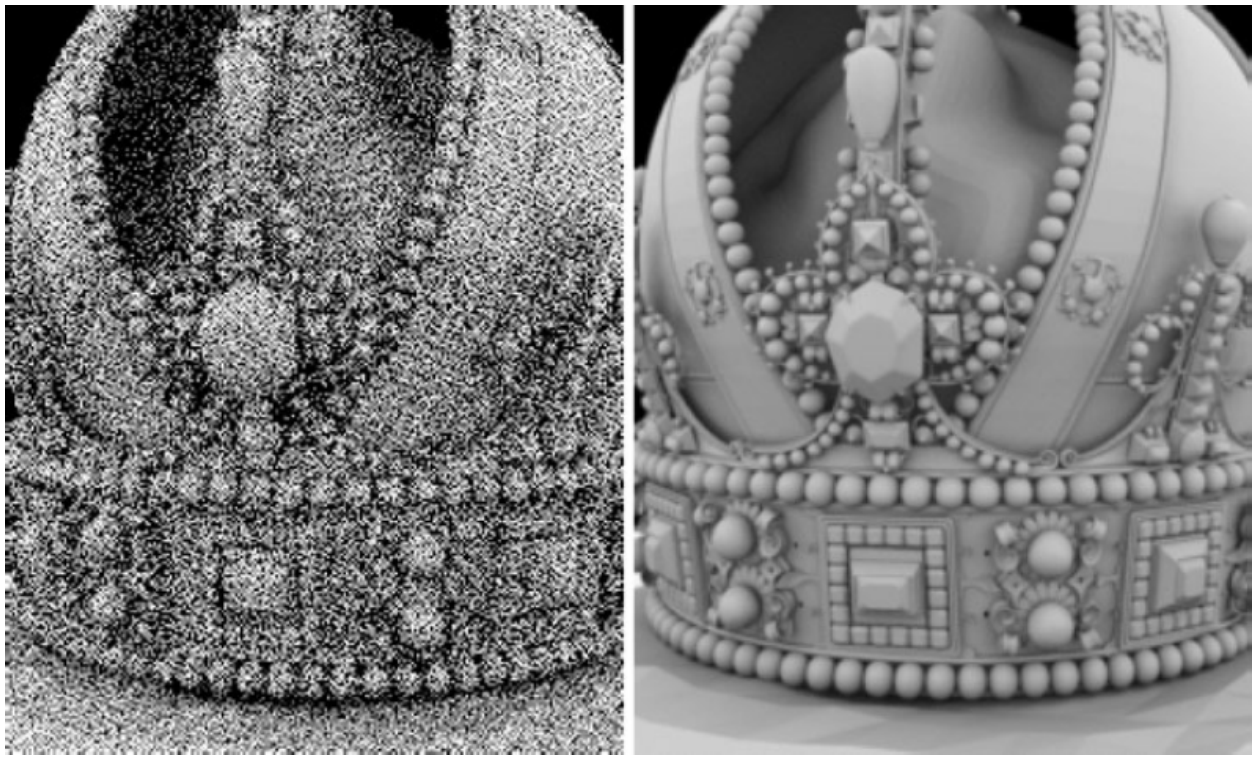
by Holger Gruen

PART IV
SAMPLING

Edited by Alexander Keller

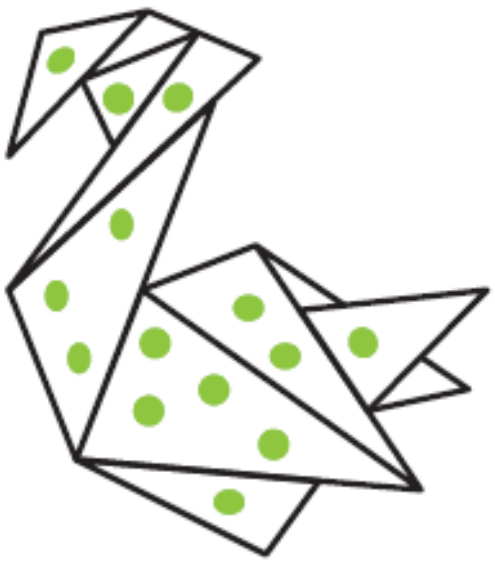
ON THE IMPORTANCE OF SAMPLING

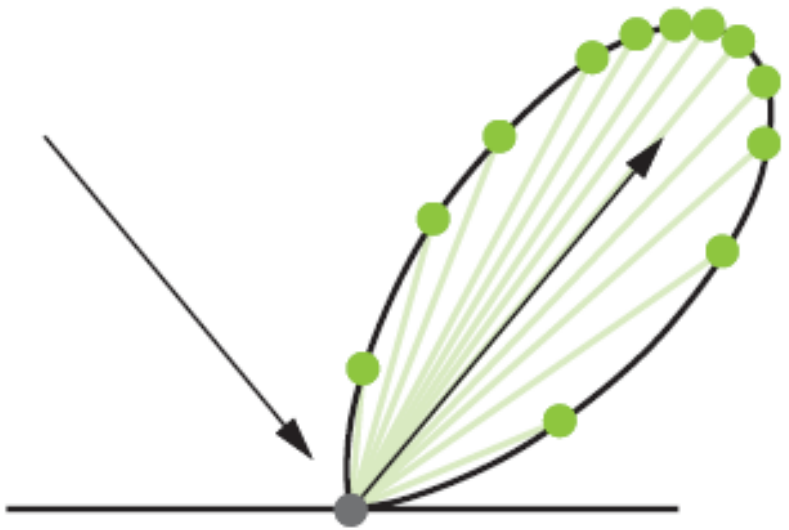
by Matt Pharr



SAMPLE TRANSFORMATIONS ZOO

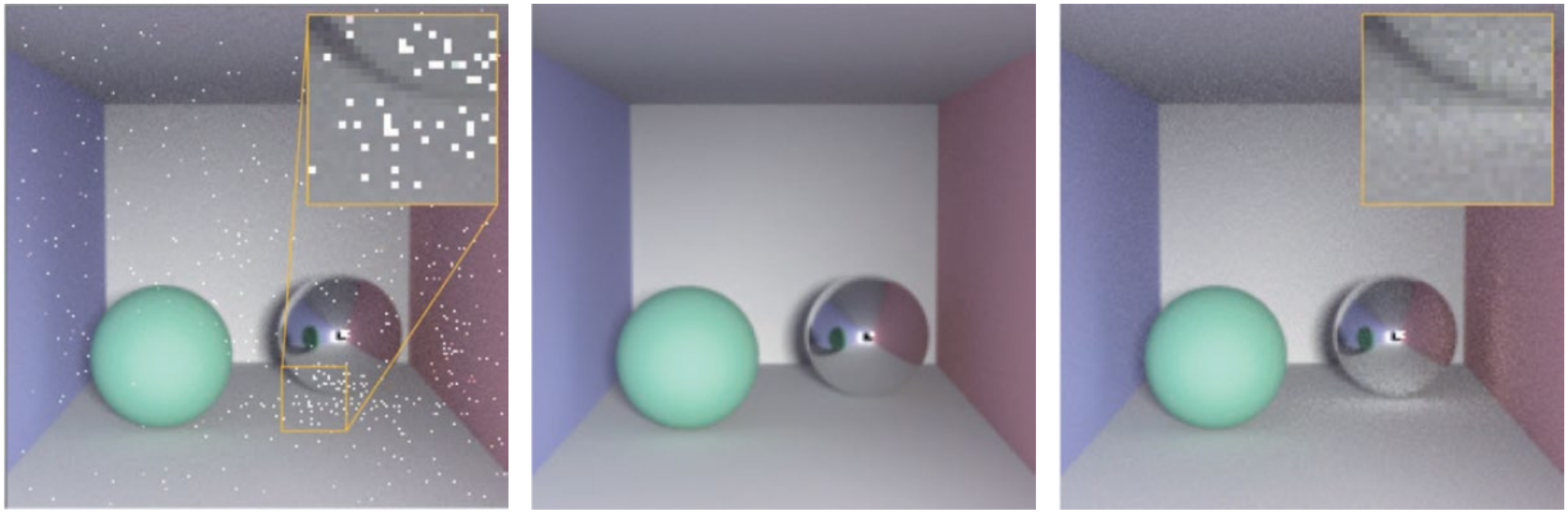
by Peter Shirley, Samuli Laine, David Hart, Matt Pharr,
Petrik Clarberg, Eric Haines, Matthias Raab, and David Cline





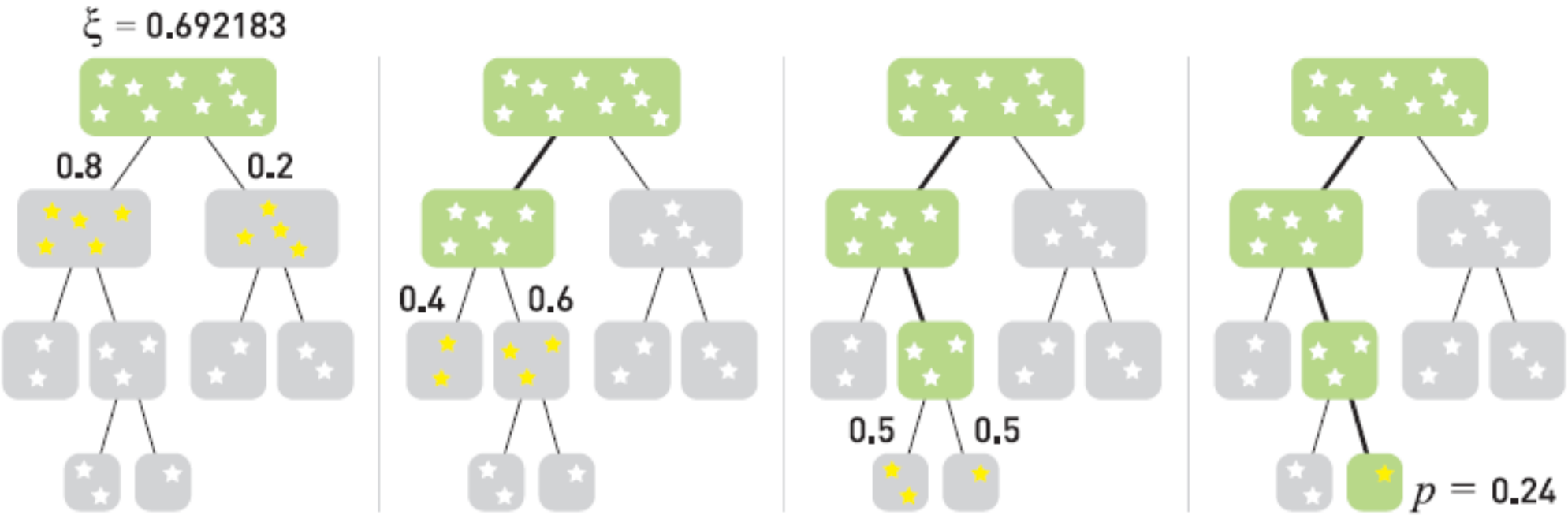
IGNORING THE INCONVENIENT WHEN TRACING RAYS

by Matt Pharr



IMPORTANCE SAMPLING OF MANY LIGHTS ON THE GPU

by Pierre Moreau and Petrik Clarberg



PART V
DENOISING AND FILTERING

Edited by Jacob Munkberg

CINEMATIC RENDERING IN UE4 WITH REAL-TIME RAY TRACING AND DENOISING

by Edward Liu, Ignacio Llamas, Juan Cañada, and Patrick Kelly

(a) Ray traced shadows

(b) Shadow maps

CINEMATIC RENDERING IN UE4 WITH REAL-TIME RAY TRACING AND DENOISING

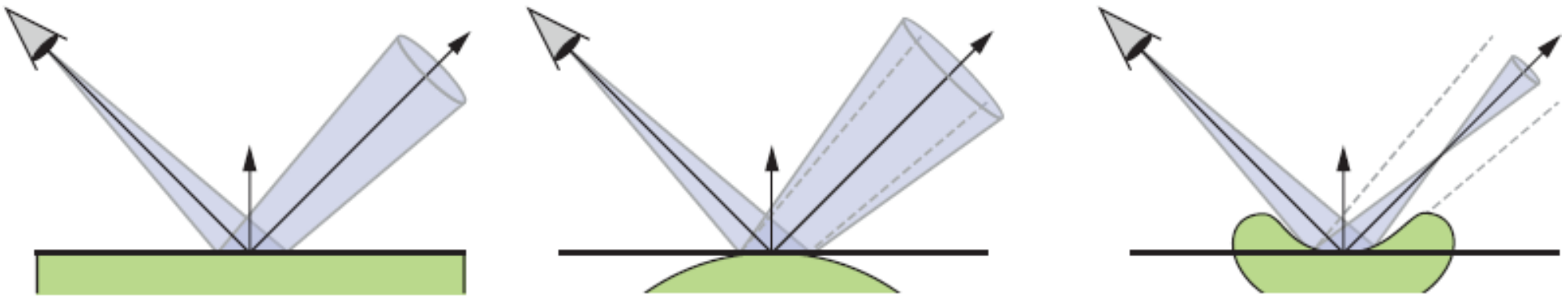
by Edward Liu, Ignacio Llamas, Juan Cañada, and Patrick Kelly

(a) Noisy input (1 spp)

(b) Our spatial denoiser

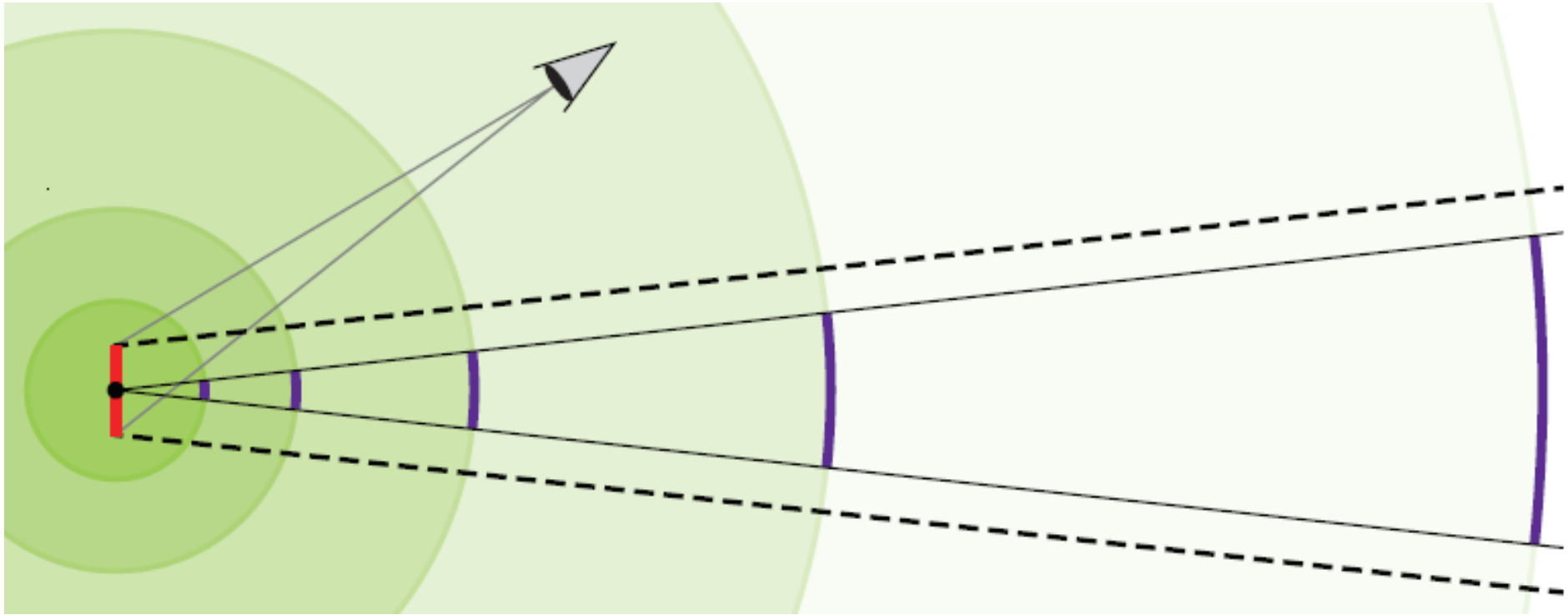
TEXTURE LEVEL OF DETAIL STRATEGIES FOR REAL-TIME RAY TRACING

by Tomas Akenine-Möller, Jim Nilsson, Magnus Andersson,
Colin Barré-Brisebois, Robert Toth, and Tero Karras



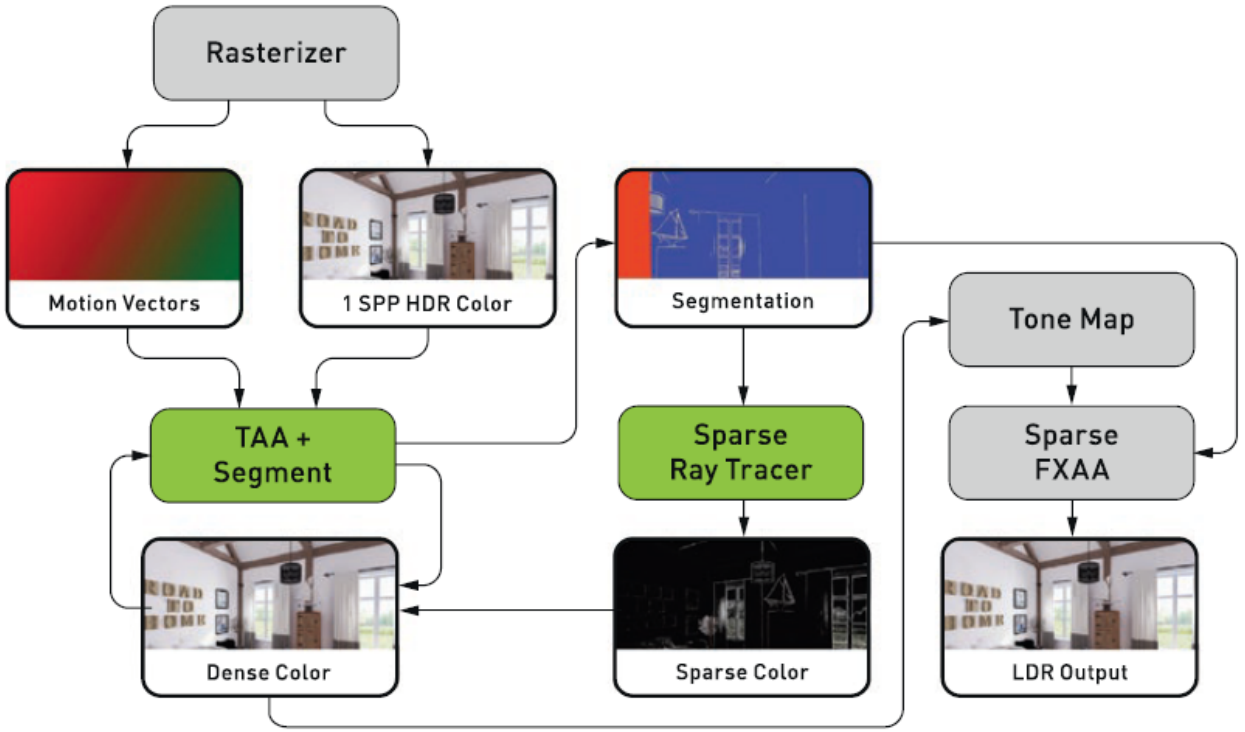
SIMPLE ENVIRONMENT MAP FILTERING USING RAY CONES AND RAY DIFFERENTIALS

by Tomas Akenine-Möller and Jim Nilsson



IMPROVING TEMPORAL ANTIALIASING WITH ADAPTIVE RAY TRACING

by Adam Marrs, Josef Spjut, Holger Gruen, Rahul Sathe, and Morgan McGuire



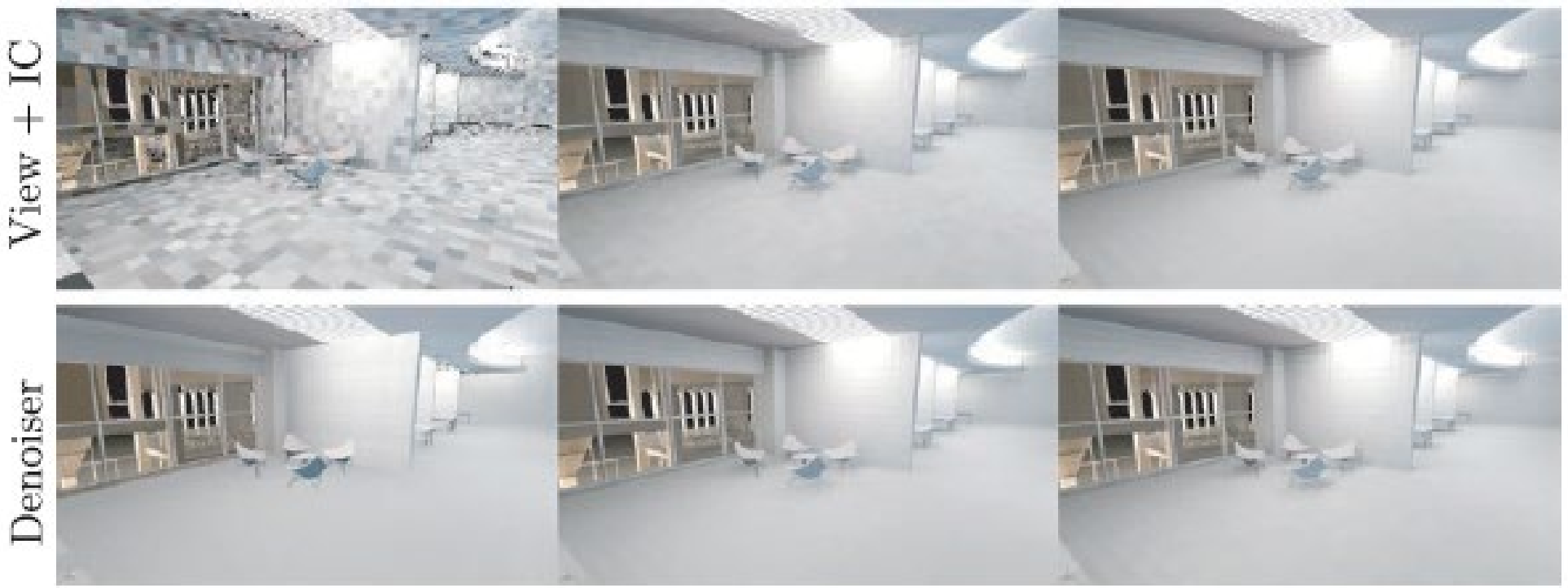
	SSAA	ATAA-C	Speedup
2×	6.30	1.47	4.29×
4×	12.60	2.70	4.67×
8×	25.20	5.32	4.74×

PART VI
HYBRID APPROACHES AND SYSTEMS

Edited by Morgan McGuire

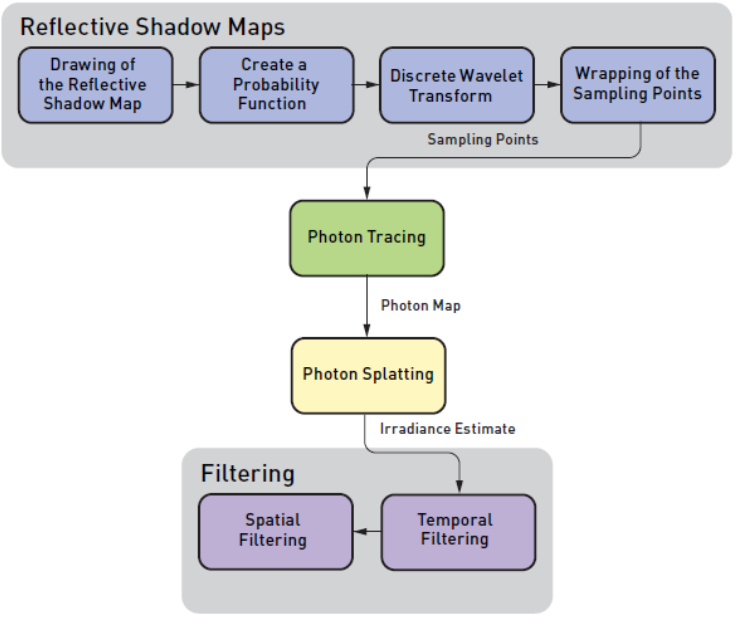
INTERACTIVE LIGHT MAP AND IRRADIANCE VOLUME PREVIEW IN FROSTBITE

by Diede Apers, Petter Edblom, Charles de Rousiers, and Sébastien Hillaire



REAL-TIME GLOBAL ILLUMINATION WITH PHOTON MAPPING

by Niklas Smal and Maksim Aizenshtein



HYBRID RENDERING FOR REAL-TIME RAY TRACING

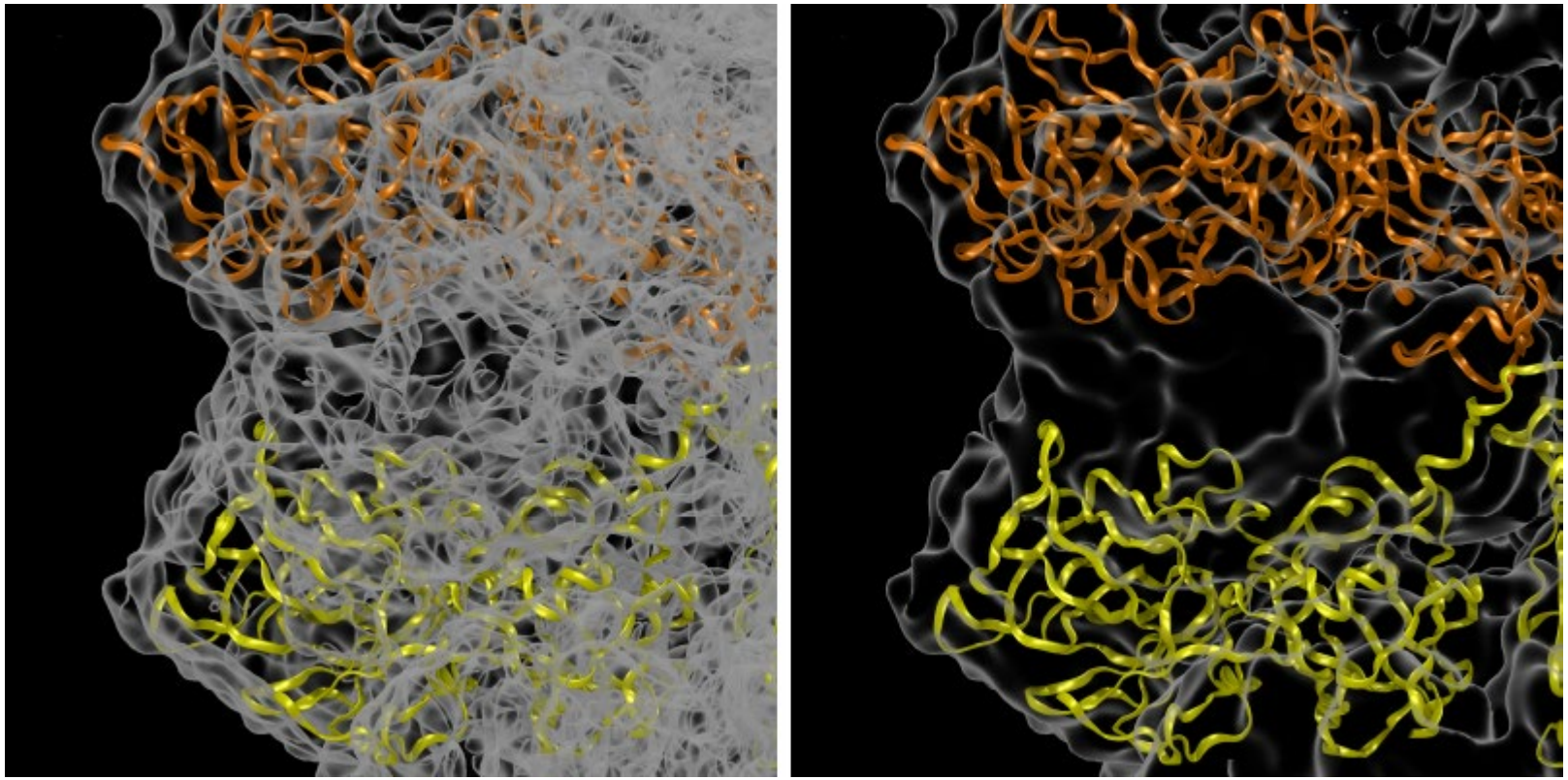
by Colin Barré-Brisebois, Henrik Halén, Graham Wihlidal, Andrew Lauritzen, Jasper Bekkers, Tomasz Stachowiak, and Johan Andersson

DEFERRED HYBRID PATH TRACING

by Thomas Schander, Clemens Musterle, and Stephan Bergmann

INTERACTIVE RAY TRACING TECHNIQUES FOR HIGH-FIDELITY SCIENTIFIC VISUALIZATION

by John E. Stone



PART VII
GLOBAL ILLUMINATION

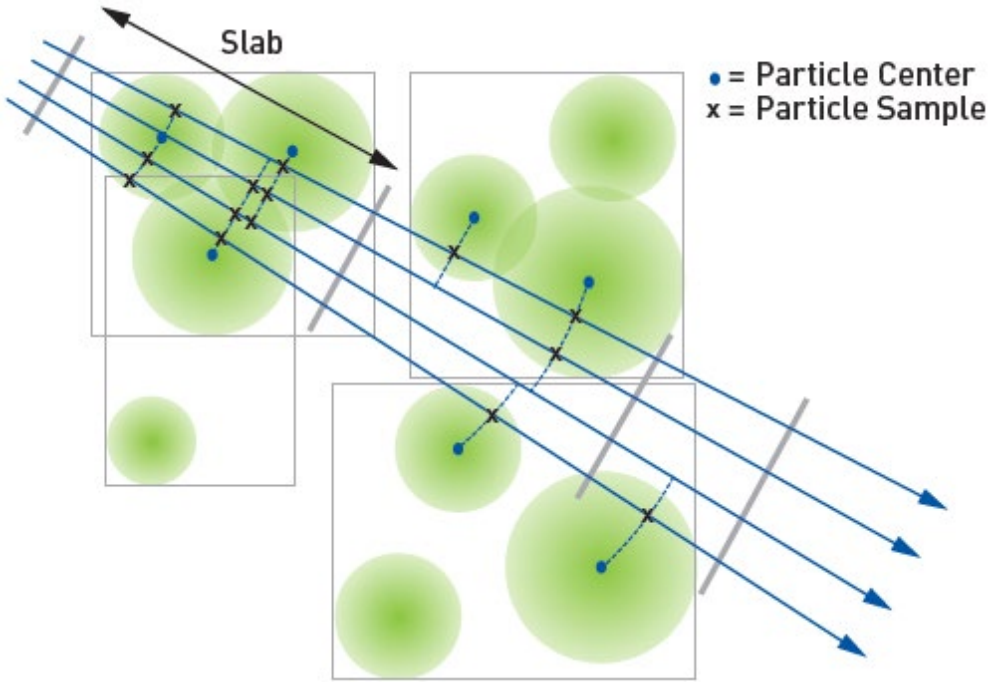
Edited by Matt Pharr

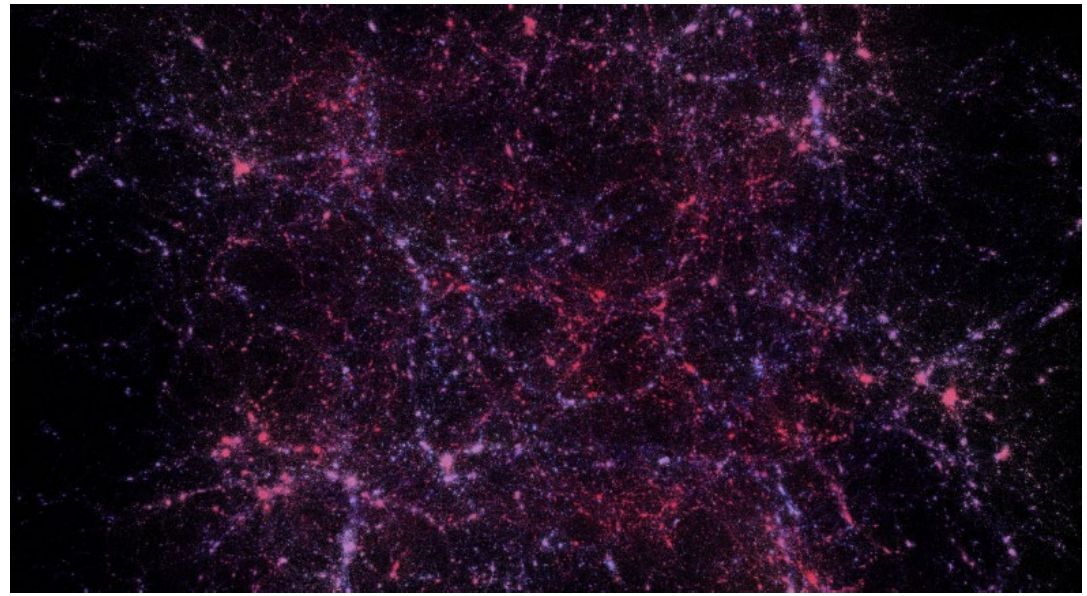
RAY TRACING INHOMOGENEOUS VOLUMES

by Matthias Raab

EFFICIENT PARTICLE VOLUME SPLATTING IN A RAY TRACER

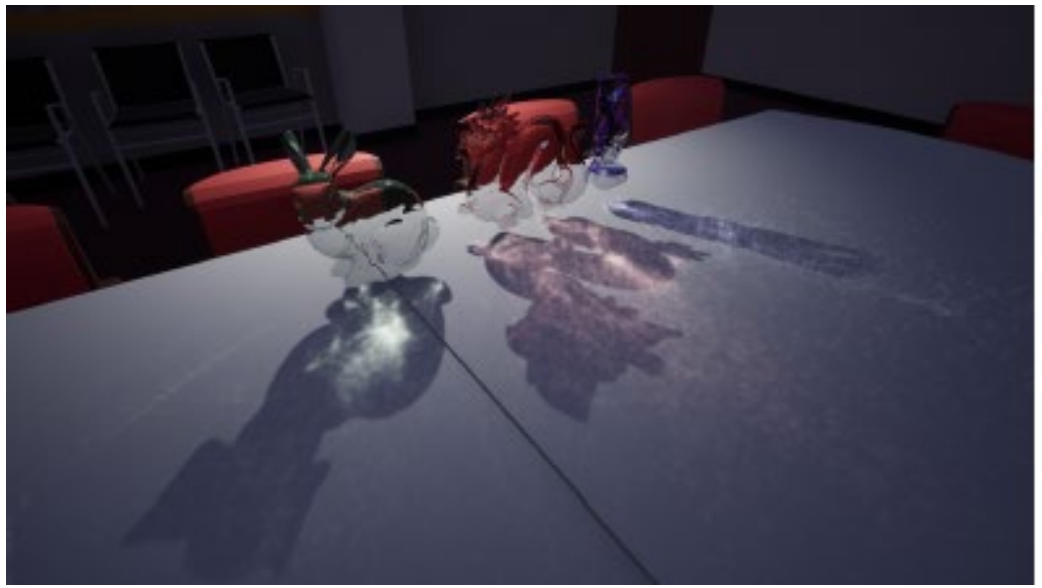
by Aaron Knoll, R. Keith Morley, Ingo Wald, Nick Leaf, and Peter Messmer





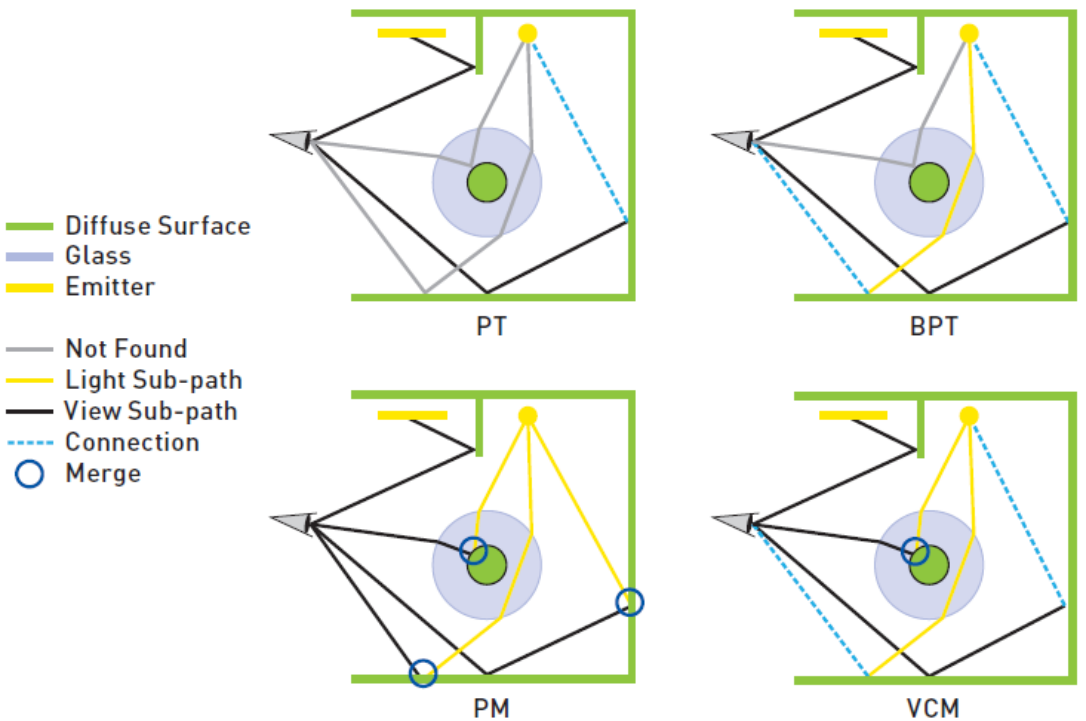
CAUSTICS USING SCREEN SPACE PHOTON MAPPING

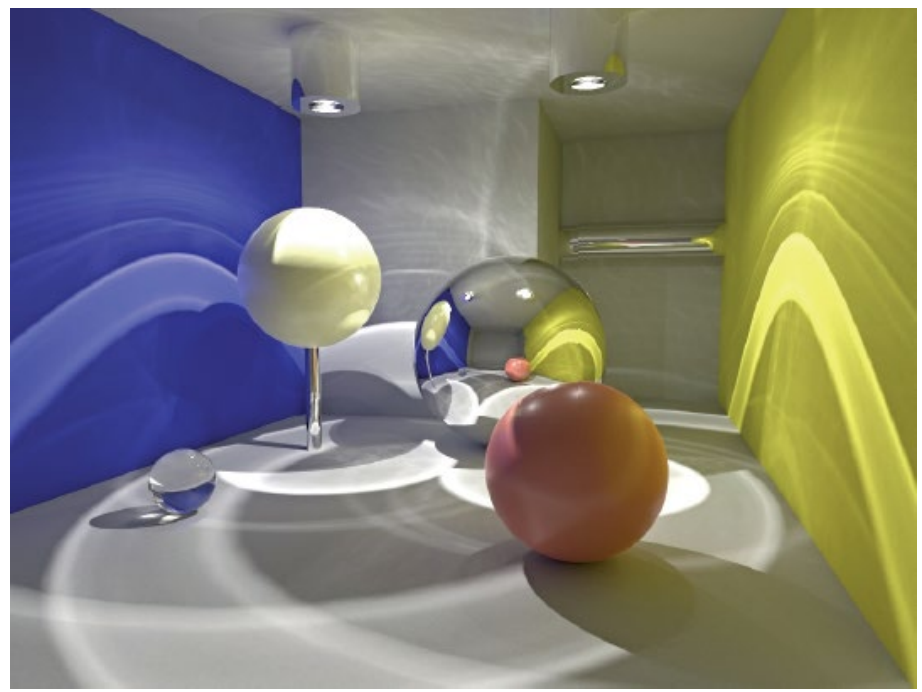
by Hyuk Kim



VARIANCE REDUCTION VIA FOOTPRINT ESTIMATION IN THE PRESENCE OF PATH REUSE

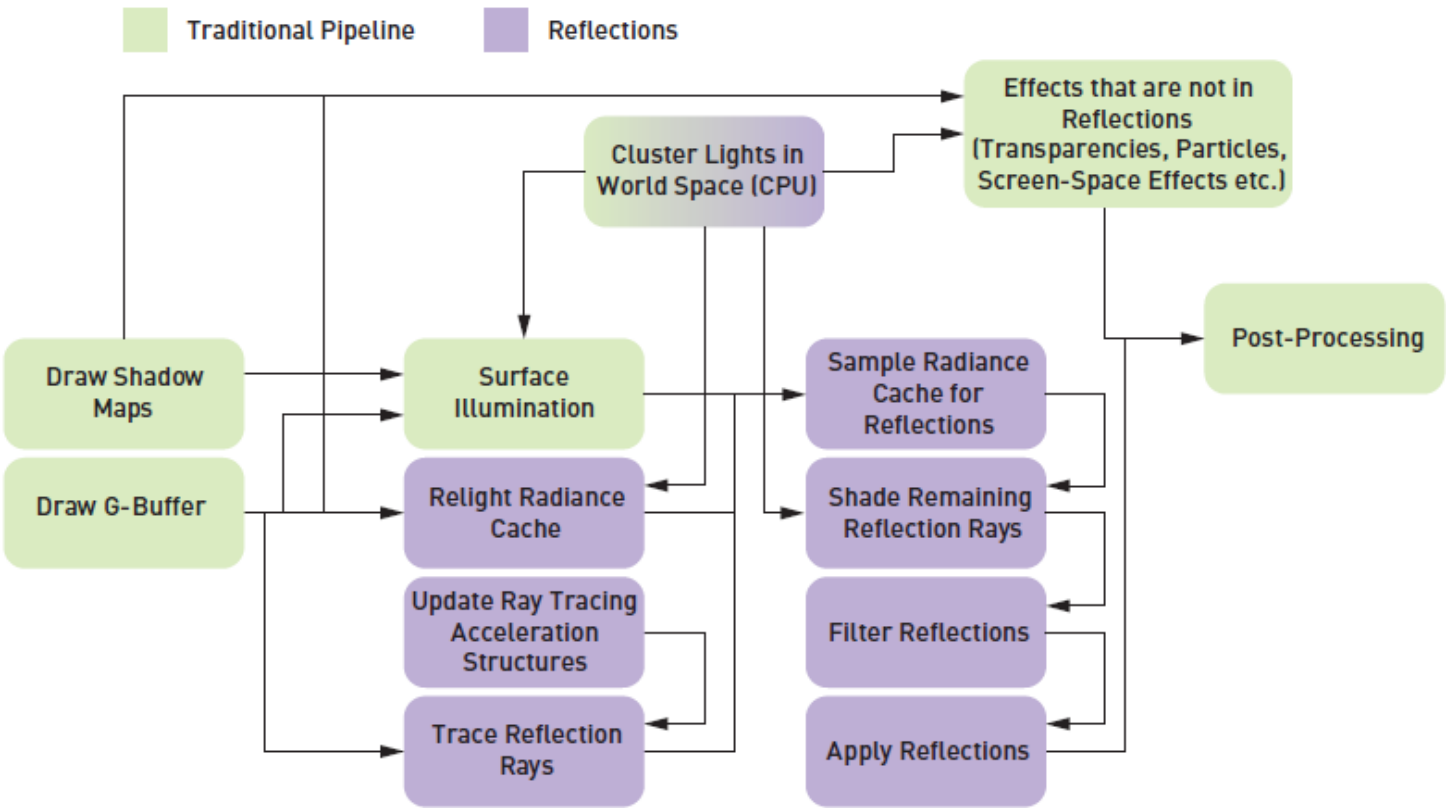
by Johannes Jendersie





ACCURATE REAL-TIME SPECULAR REFLECTIONS WITH RADIANCE CACHING

by Antti Hirvonen, Atte Seppälä, Maksim Aizenshtein, and Niklas Smal



ACCURATE REAL-TIME SPECULAR REFLECTIONS WITH RADIANCE CACHING

by Antti Hirvonen, Atte Seppälä, Maksim Aizenshtein, and Niklas Smal

“RT is the future of gaming, so the main focus is now on RT either way.”

- Ben Archard, *Metro Exodus* programmer

Ray Tracing Gems 2?

raytracinggems.com

THE DANGERS OF RAY TRACING

CAUTION:
OBJECT
CONTAINS
CAUSTICS

QUESTIONS?

Eric Haines| ehaines@nvidia.com