Irradiance & Light field Probes with Visibility

Morgan McGuire (@CasualEffects) NVIDIA GDC 2017

Based on Real-Time Global illumination using Precomputed Light Field Probes
by McGuire, Mara, Nowrouzezahrai, and Luebke I3D 2017

http://bit.ly/2mQYlwG

State of the Art

GLOBAL ILLUMINATION

Mirror reflections: screen-space ray cast + environment probes

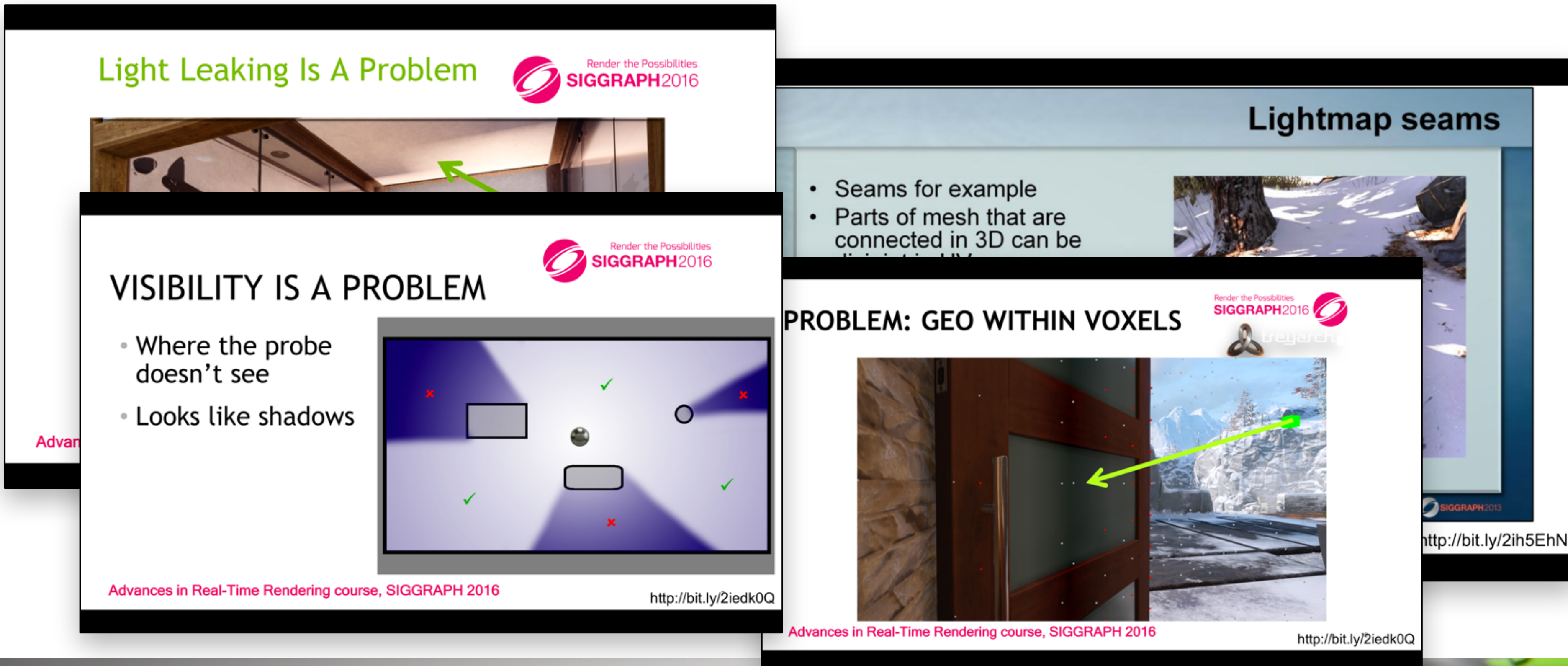
Glossy reflections: distorted preconvolved environment map probes

Matte reflections: light maps or irradiance/voxel probes

Transmission: blending or screen-space distortion

State of the Art

GLOBAL ILLUMINATION



[Iwanicki 2013, Hooker 2016]

TODAY'S TALK

1. Irradiance Probes with Visibility

(*Deployable now*)

Extend existing irradiance tech.

Fixes light leaks: no per-probe artist time

0.35 ms/ frame @ 1080p on GeForce 1080

2. Light Field Probes

(*Preview of ongoing R&D*)

Extend screen-space ray tracing tech.

Fixes all SSR problems

10 ms/ frame @ 1080p on GeForce 1080

Irradiance Probes with Visibility

History of

PRECONVOLVED IRRADIANCE PROBES

1970s Constant ambient

1990s Hemisphere ambient

1990s IBL

Circa 2000 Preconvolved irradiance cube & SH maps

(ATI cubemapgen/RTR2)

Grid of irradiance maps

Depth proxy geometry

New: automatic leak prevention and smoothing

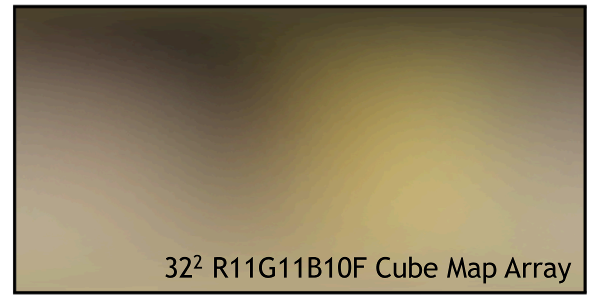
Far Cry 3

PREFILTERED VISIBILITY

Prefiltered irradiance probes are a common trick…but leak light. Adding visibility tests creates hard shadow line errors.

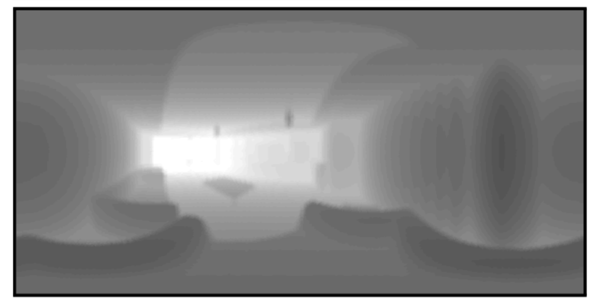
Following variance shadow maps [Donnelly & Lauritzen], we store the first two moments of a depth distribution and perform a prefiltered Chebyshev depth test.

$E = \int_\Omega L \ \omega \cdot n$

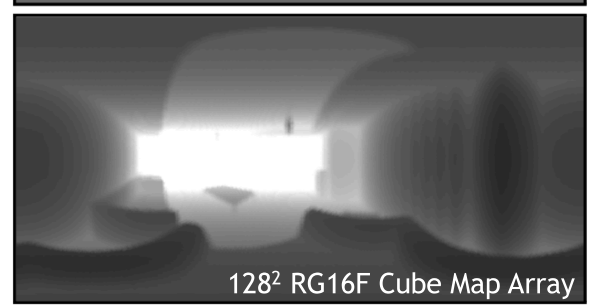


32^2 R11G11B10F Cube Map Array

$\int_G r$

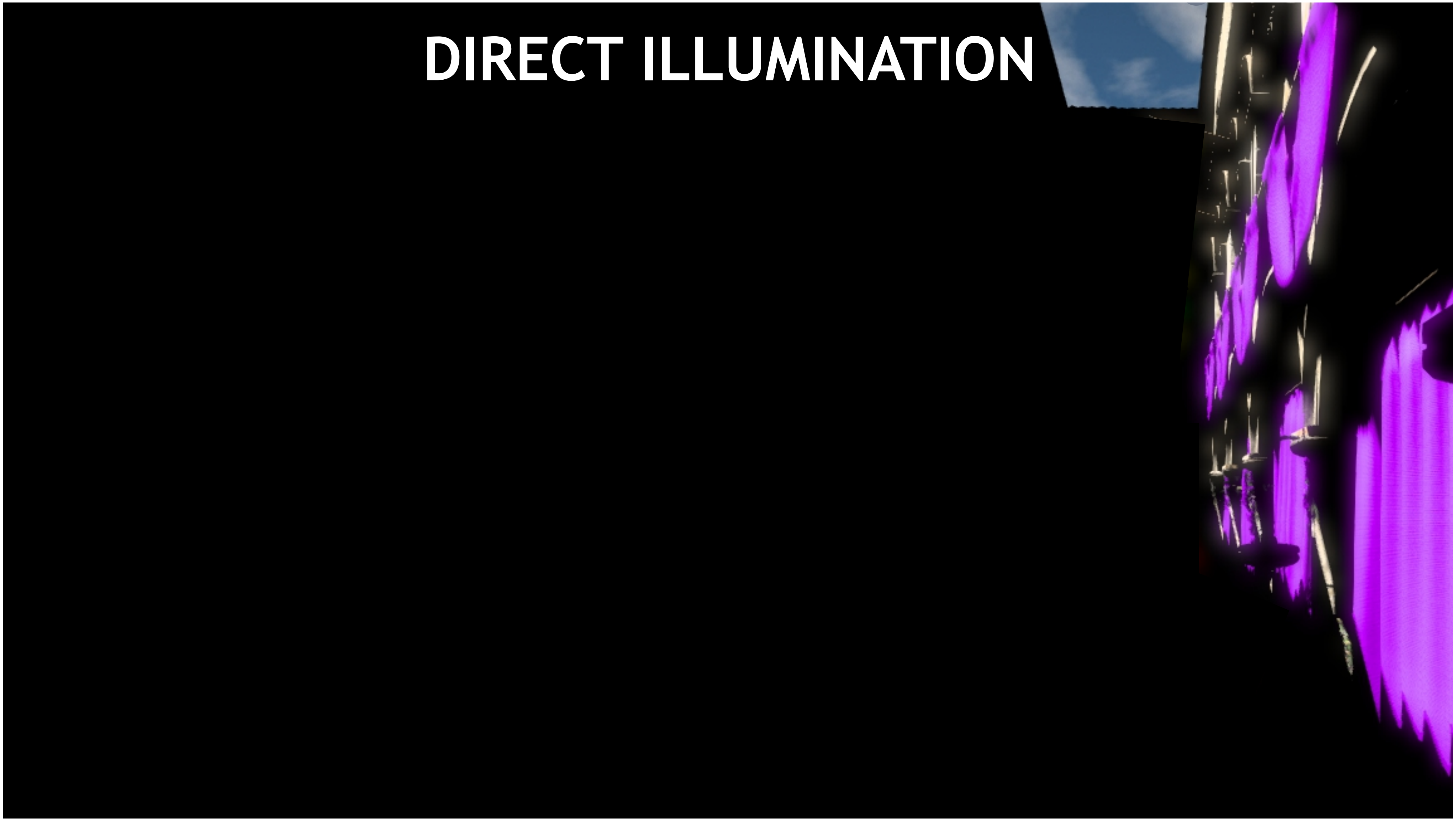


$\int_G r^2$



128^2 RG16F Cube Map Array

DIRECT ILLUMINATION



GLOBAL ILLUMINATION

IRRADIANCE PROBE WEIGHTS

Smoothly fade out backfaces

$$w = \mathrm{max}(\mathrm{trilinear}, \varepsilon) \cdot \mathrm{max}(\hat{n} \cdot v / \|\vec{v}\|, \varepsilon) \cdot \mathrm{max}(\sigma^2 / (\sigma^2 + (\|\vec{v}\| - m^2), \varepsilon)$$

Transition to nearest probe

Chebyshev: Fraction of [weighted] sphere that is visible

where m = mean radius = interpolate(r), s = mean squared radius = interpolate(r^2),

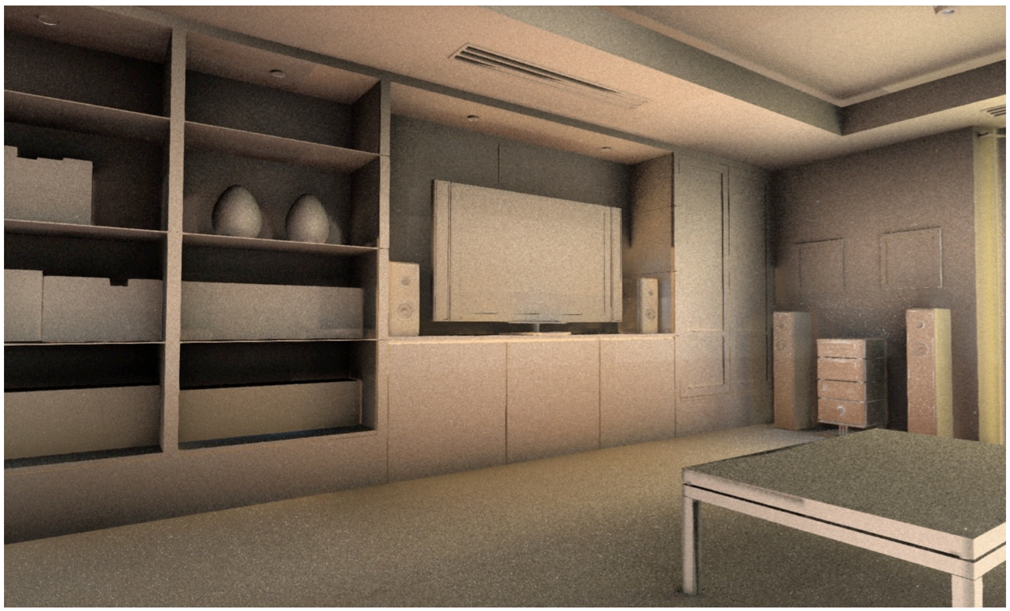
$\hat{n}$ = surface normal, $\|\vec{v}\|$ = vector to probe, $\sigma^2 = |m^2 - s|$

SHADER IMPLEMENTATION

```
for (int i = 0; i < 8; ++i) {
    int3  offset = ivec3(i, i >> 1, i >> 2) & ivec3(1, 1, 1);
    int3  probeGridCoord = clamp(baseGridCoord + offset, int3(0, 0, 0), int3(lightFieldSurface.probeCounts - 1));
    int p = gridCoordToProbeIndex(lightFieldSurface, probeGridCoord);

    float3 probePos = gridCoordToPosition(lightFieldSurface, probeGridCoord);
    float3 probeToPoint = wsPosition - probePos;
    float3 dir = normalize(-probeToPoint);
    float distToProbe = length(probeToPoint);

    // Trilinear and smooth backface weights
    float3 trilinear = lerp(1.0 - alpha, alpha, offset);
    float weight = trilinear.x * trilinear.y * trilinear.z * max(0.005, dot(dir, wsN));

    // Chebychev weight
    float2 temp = texture(lightFieldSurface.meanMeanSquaredProbeGrid.sampler, vec4(-dir, p)).rg;
    float mean = temp.x + lightFieldSurface.irradianceDistanceBias;
    float variance = abs(square(temp.x) - temp.y) + lightFieldSurface.irradianceVarianceBias;
    float chebyshevWeight = variance / (variance + square(distToProbe - mean));

    // Increase contrast in the weight
    chebyshevWeight = max(square(chebyshevWeight) - lightFieldSurface.irradianceChebyshevBias, 0.0) / (1.0 - lightFieldSurface.irradianceChebyshevBias);
    weight = max(0.00001, weight * ((distToProbe <= mean) ? 1.0 : chebyshevWeight));

    sumWeight += weight;
    sumIrradiance += weight * texture(lightFieldSurface.irradianceProbeGrid.sampler, float4(normalize(irradianceDir), p)).rgb;
}

E_lambertianIndirect = 0.5 * pi * sumIrradiance / sumWeight;
```

APPROXIMATION QUALITY

True ray traced irradiance

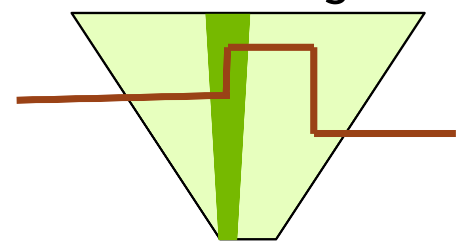


Probe w/ visibility approximation

WHAT ABOUT LEAKING FROM VSM?

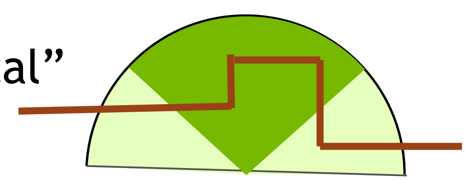
VSM leaks light & shadow with point sources:

- Point light shadow texels see bimodal depth distributions: 2 moments not enough
- Single shadow map for entire scene
- Chebyshev test is very conservative…leaks



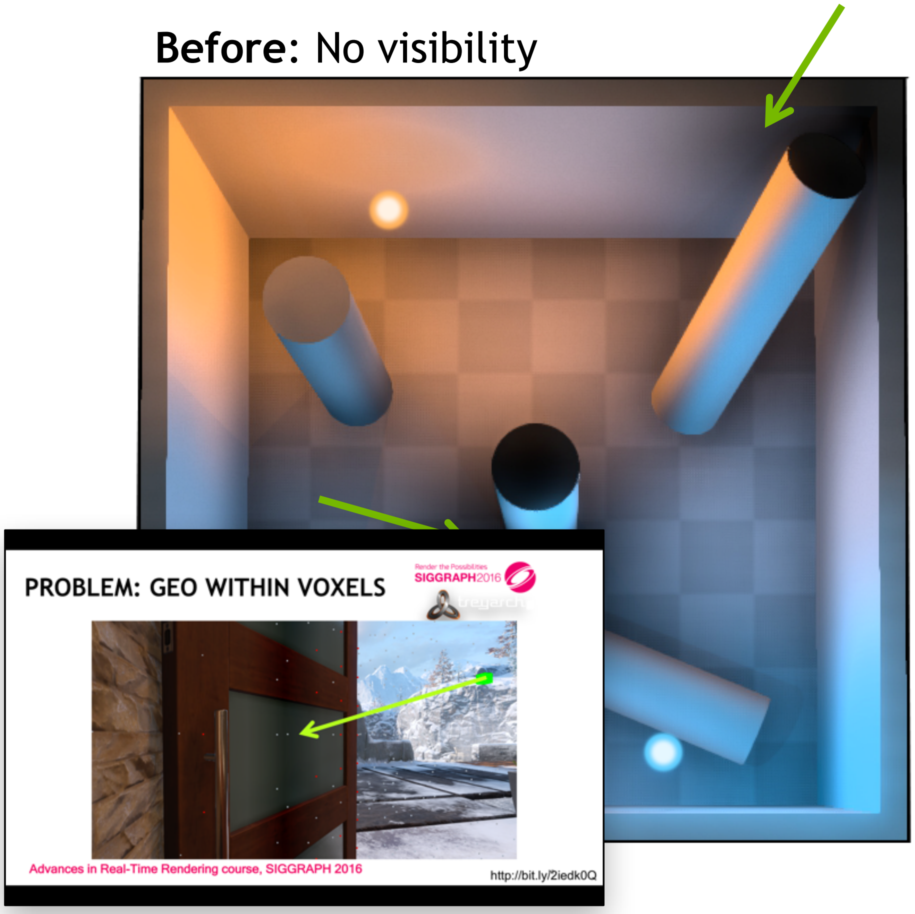
VSM fits irradiance probes well:

- Irradiance shadows integrate ¼ cosine-weighted sphere: smoother distribution
- Switch shadow maps every 2m and clamp depth, so always “local”
- Additional backface and trilinear terms for proximity

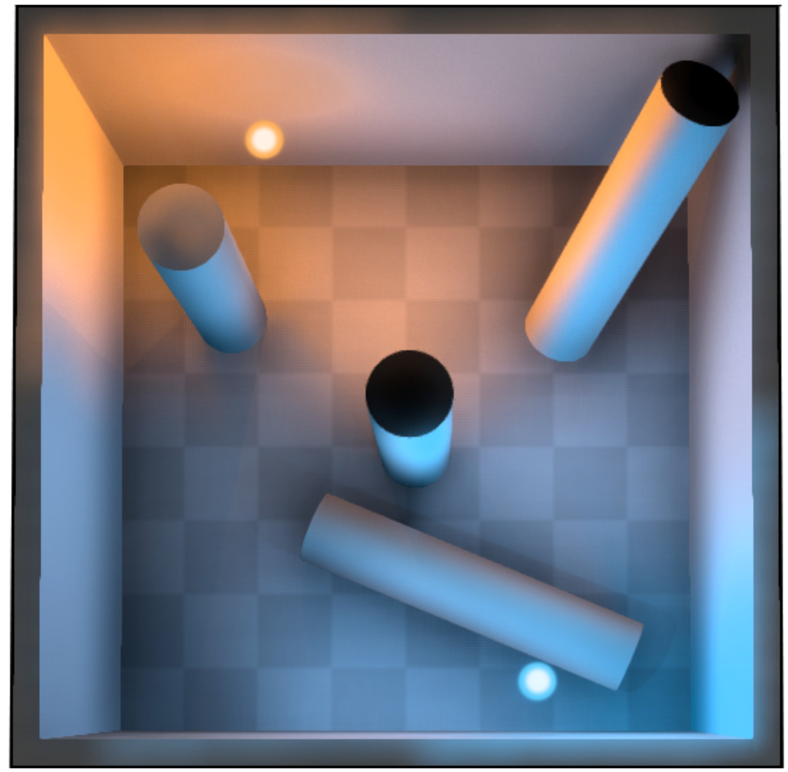


IRRADIANCE PROBES INSIDE GEOMETRY

Before: No visibility

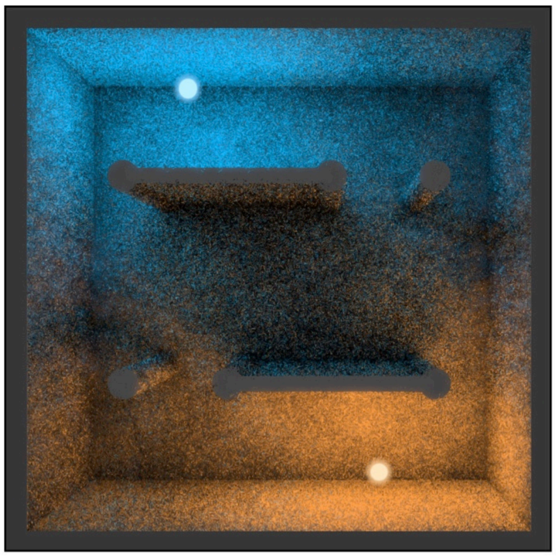


After: Our prefiltered visibility



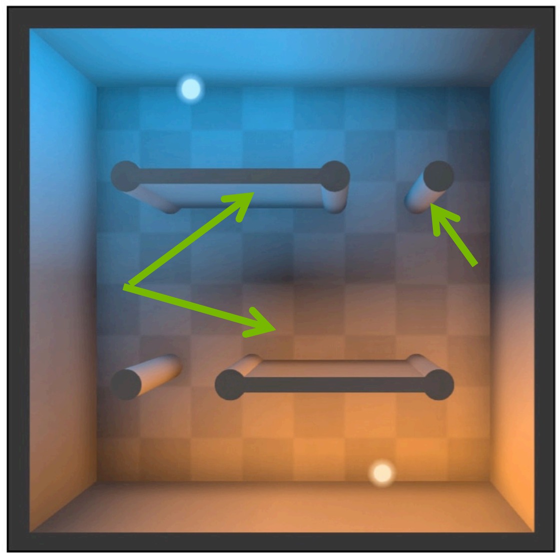
IRRADIANCE PROBE INDIRECT SHADOWING

Ray traced (13ms)

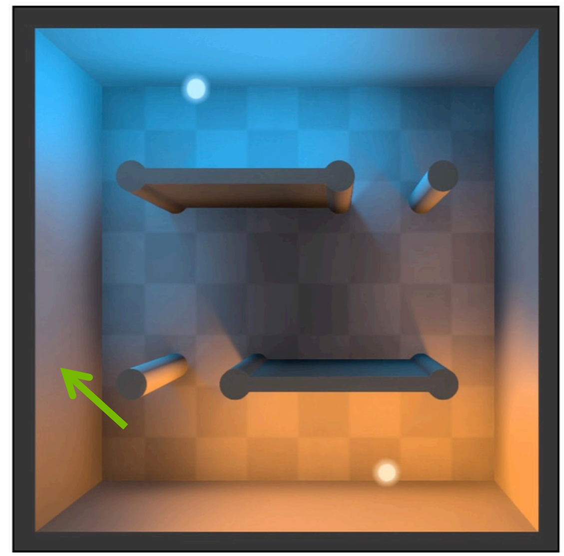


4 Probes

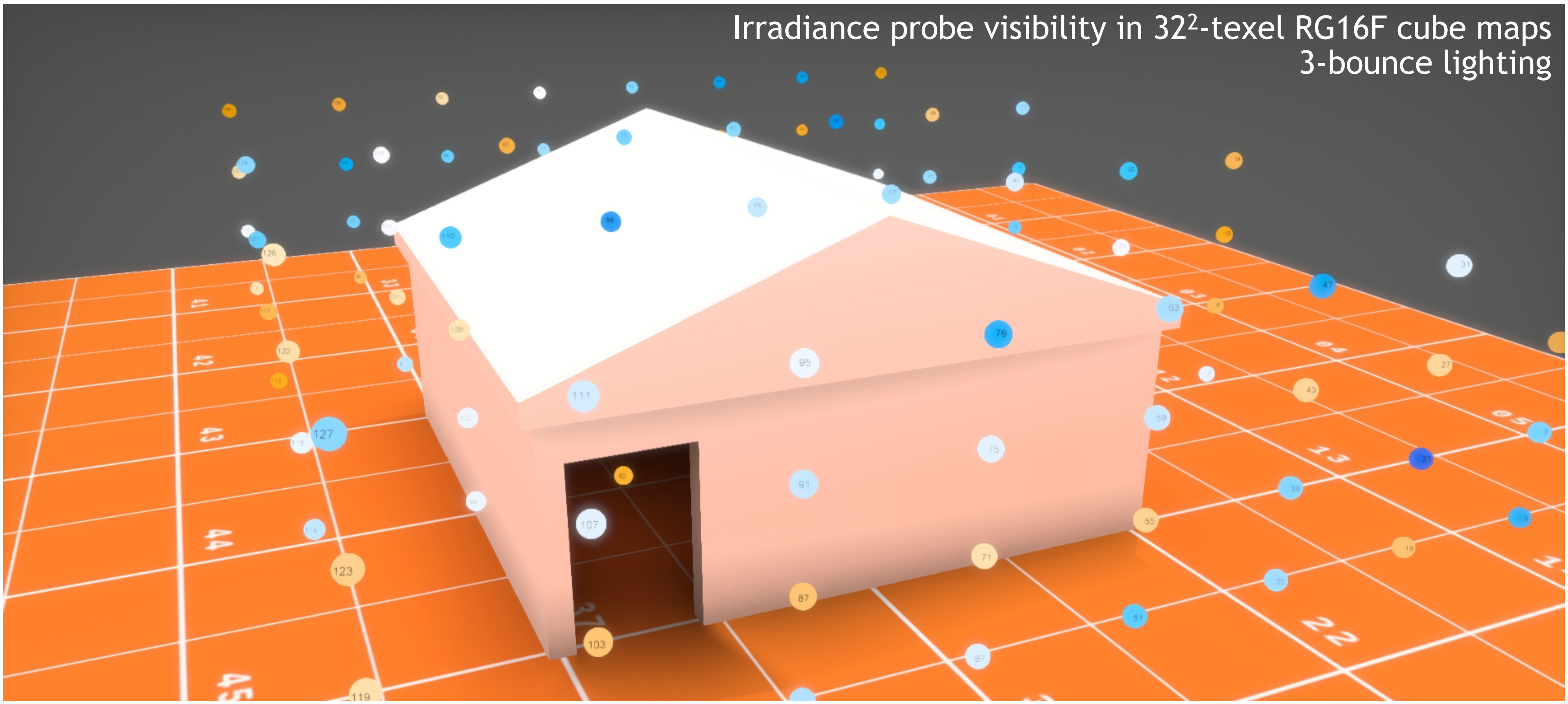
Before



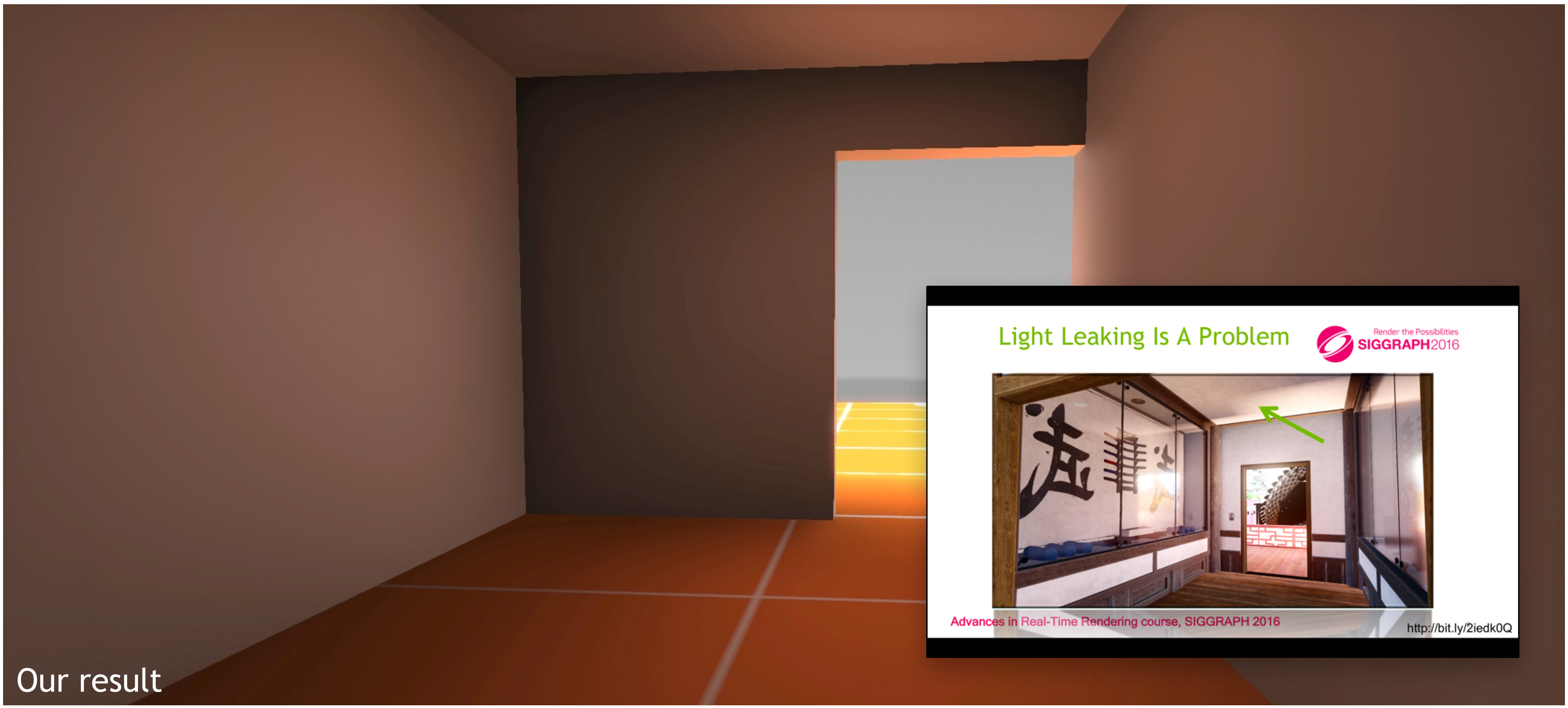
After: Prefiltered vis. (0.2ms)



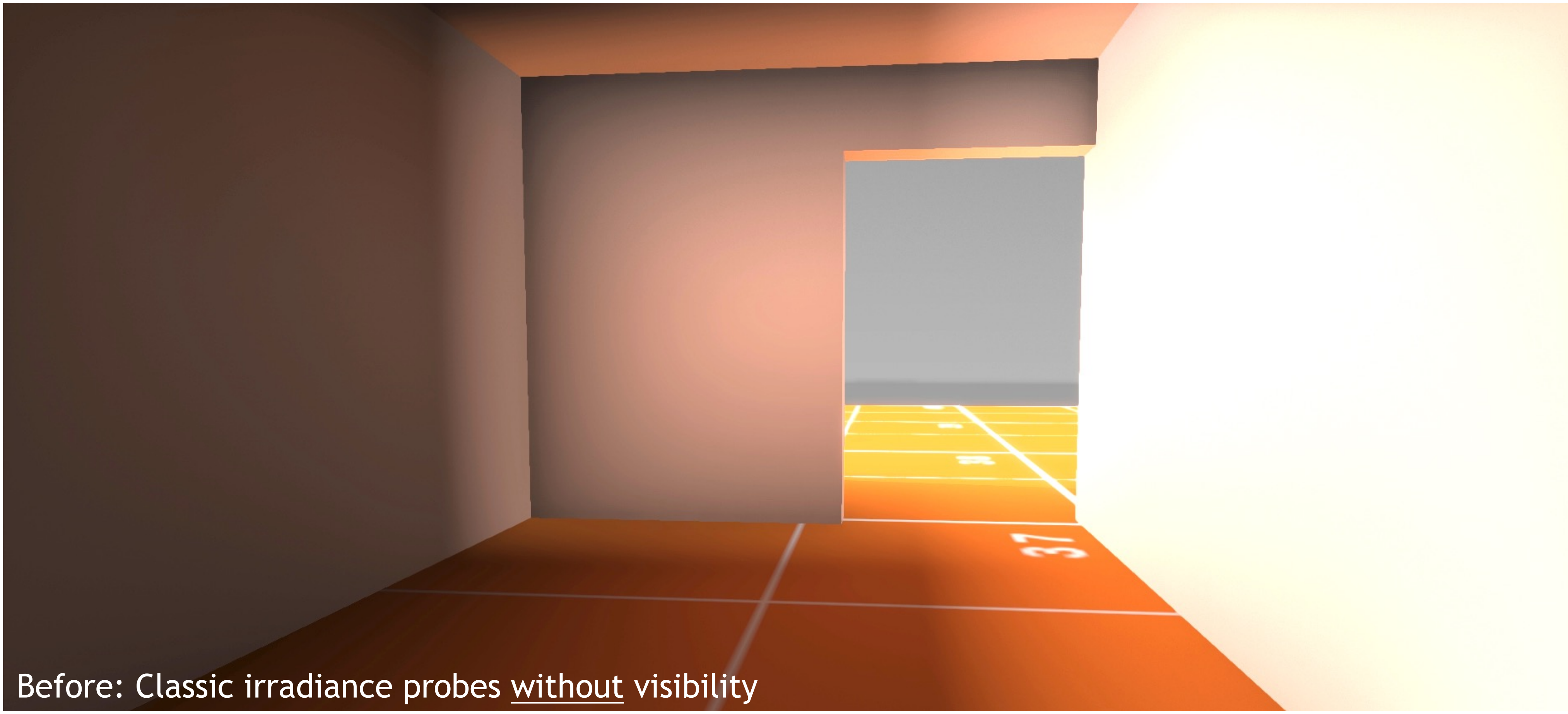
VISIBILITY TEST CASE



VIEW FROM INSIDE

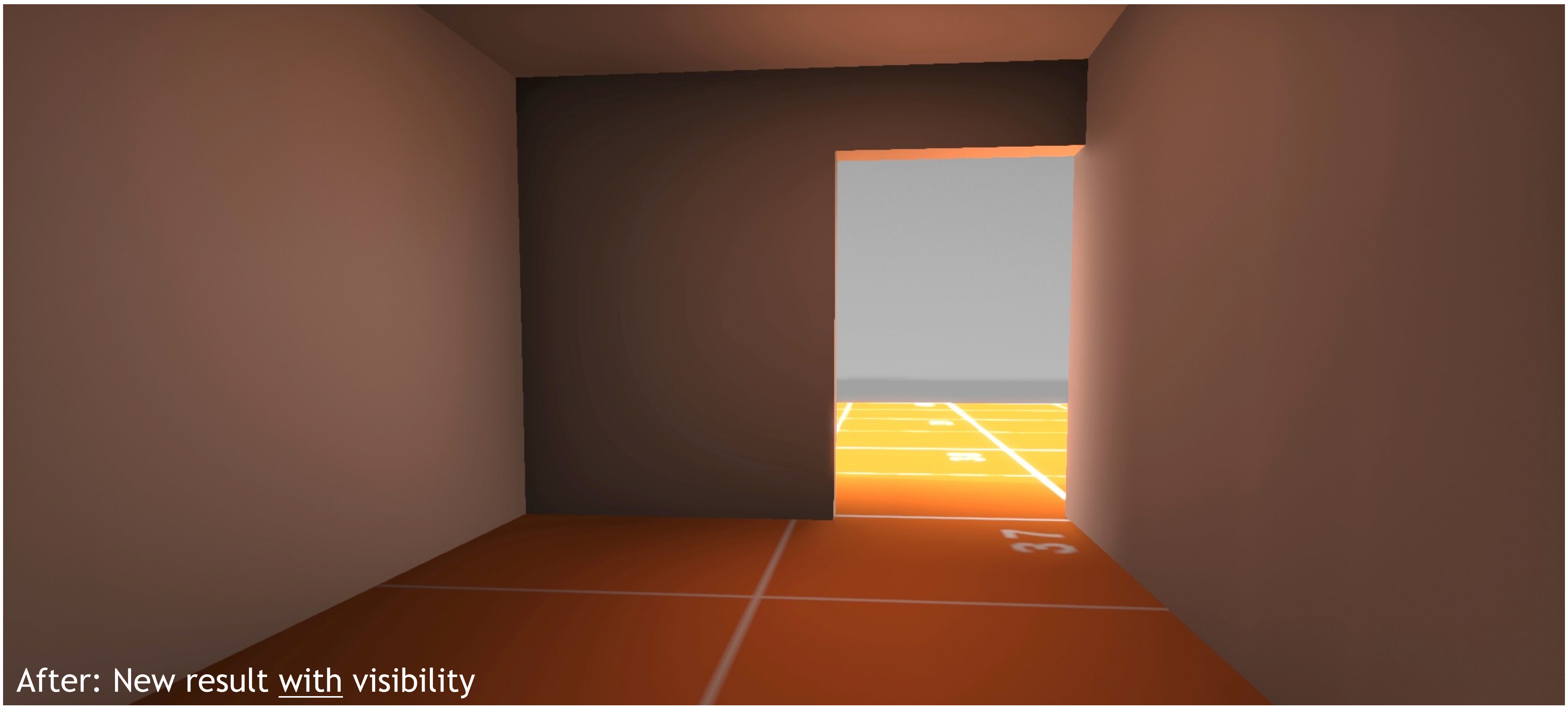


VIEW FROM INSIDE



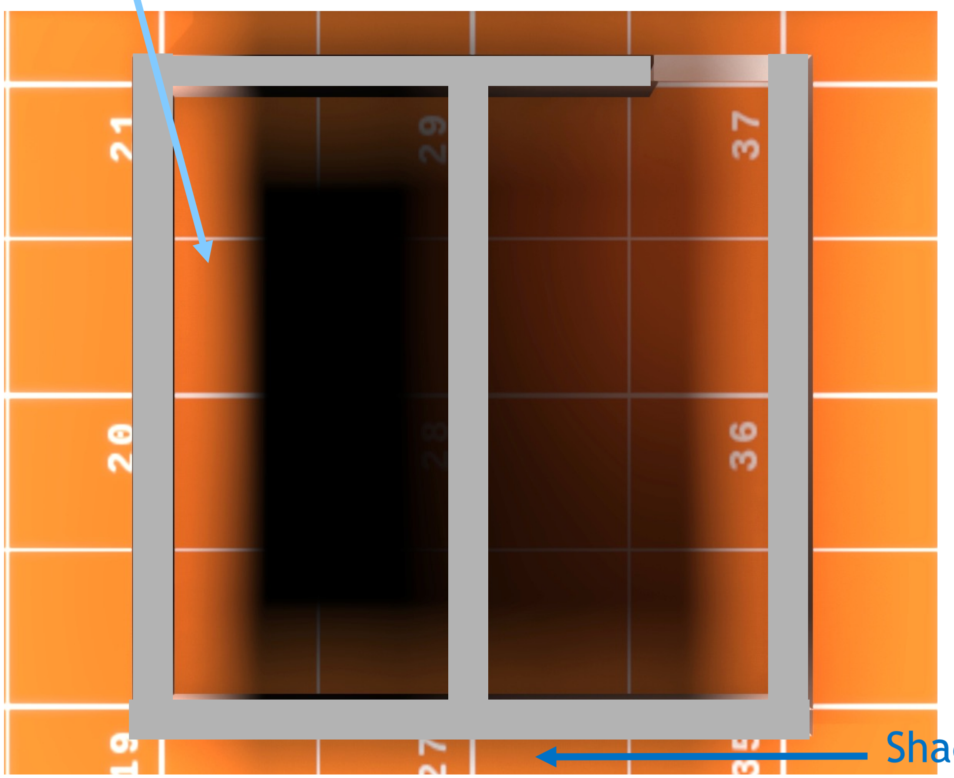
Before: Classic irradiance probes without visibility

VIEW FROM INSIDE

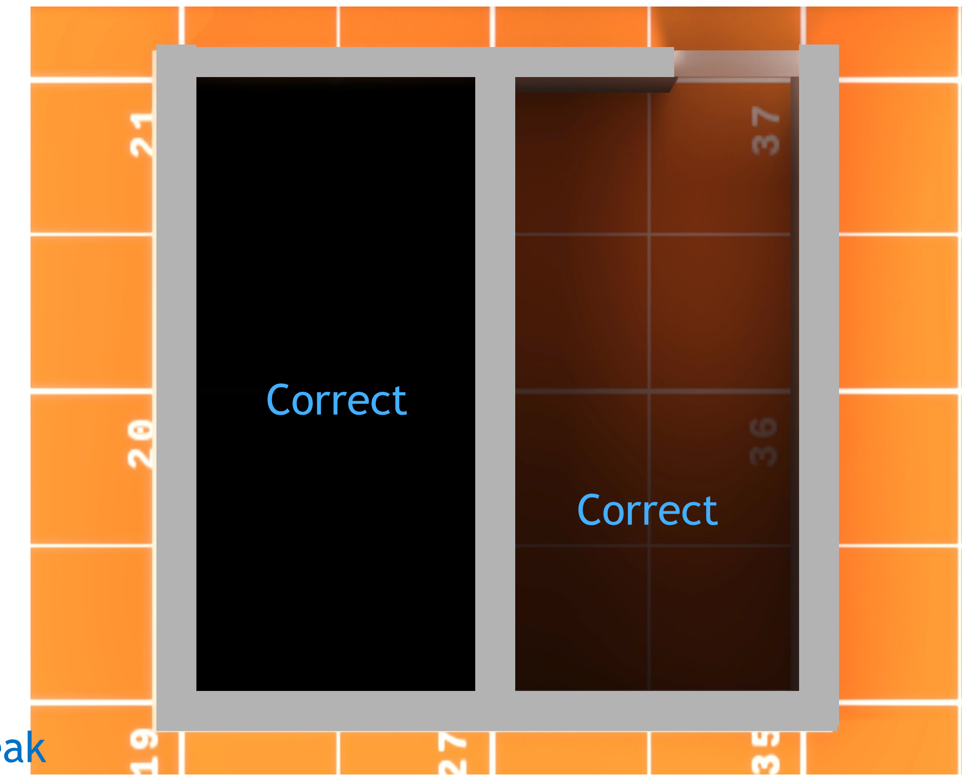


TOP VIEW CUTAWAY

Without visibility



Our result




Before: Without visibility, 3-bounce lighting
NVIDIA


After (Max quality): 128² visibility, 3-bounce lighting
NVIDIA


Before: Without visibility, 3-bounce lighting


After (Max quality): 128² visibility, 3-bounce lighting

After (Medium Quality): $\underline{32}^2$ visibility, 3-bounce lighting

After (Minimum memory): $\underline{2}^2$ visibility, constant variance, 3-bounce lighting

Before: 2^2 no visibility, 3-bounce lighting

IRRADIANCE RESOLUTION REGIMES

$128^2 \times 6 \times$ RG16F: Robust to leaks and probe positions, great indirect shadows

$32^2 \times 6 \times$ RG16F: Robust to leaks, shadowing biased by probes

$4^2 \times 6 \times$ R16F: Some leaking, but better than state of the art at low memory

VARIATIONS

Use a **sparse lookup texture** to map grid points to probes and remove probes from the center of rooms, like sparse oct-tree

Hard-code variance (shown here) when the probes are smaller than 8^2 texels per face

Use **screen-space blurring** (shown here) to further smooth transitions a low resolution

Combine with scalable AO and deep G-buffer **AO** (shown here) to restore high-frequency occlusion

Avoid the fetch for probes that fail the **backface test.** Saves 4 fetches/pix on average

Tetrahedral grid halves the average number of fetches, but harder to filter and index

Light Field Probes

Upcoming tech in development
4-10 ms @ 1920x1080 on GeForce 1080

History of

REFLECTION PROBES

Environment maps - Blinn & Newell 1976

Cube maps – Greene 1986

Various **blended cube** map grid solutions, e.g., Source engine

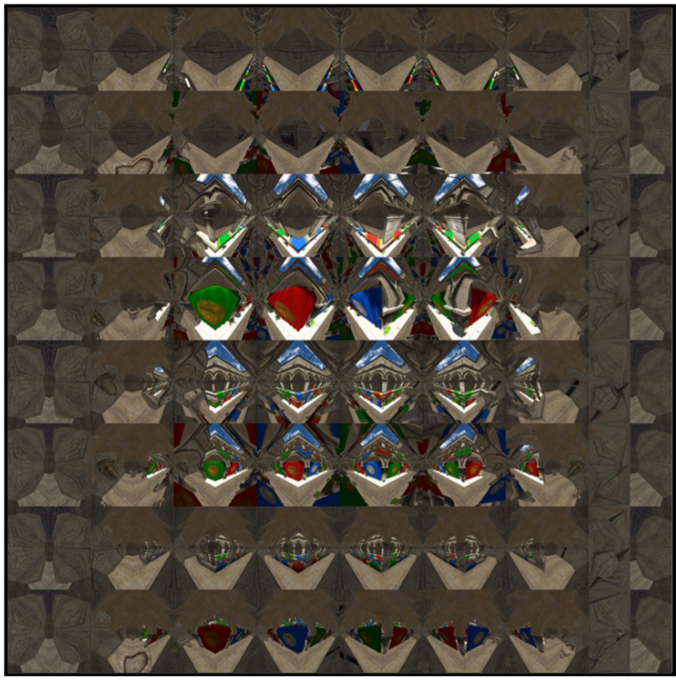
Cube depth proxy – Bjorke 2004, Sebastien and Zanuttini 2012, Lagarde 2013, et al.

Polyhedral depth proxy - Szirmay-Kalos 2005

Heightfield depth - Evangelakos 2015, Donow 2016

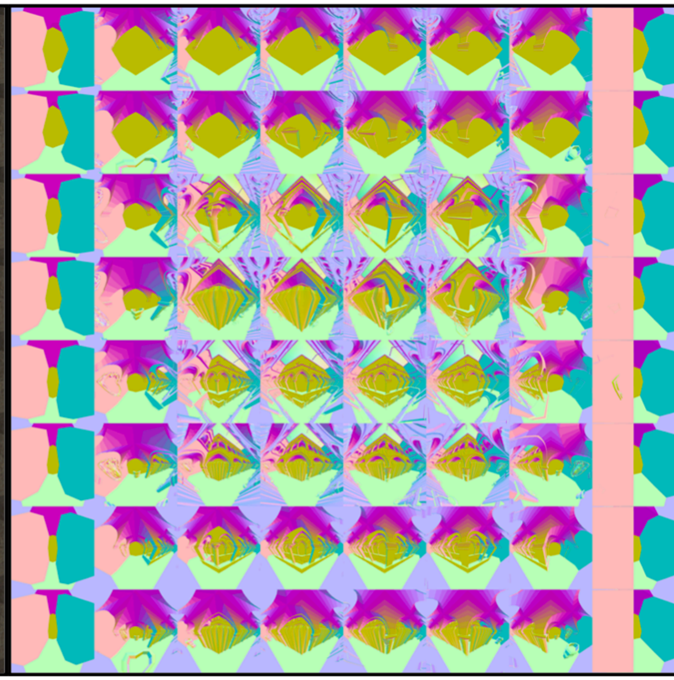
Light Field Probes

DATA STRUCTURE



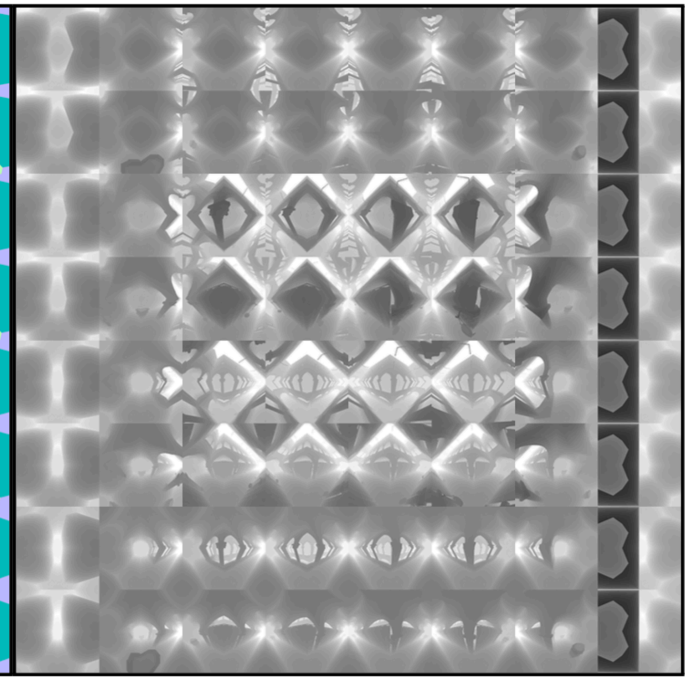
HDR Radiance

R11G11B10F, BC6H



Compressed Surface Normals

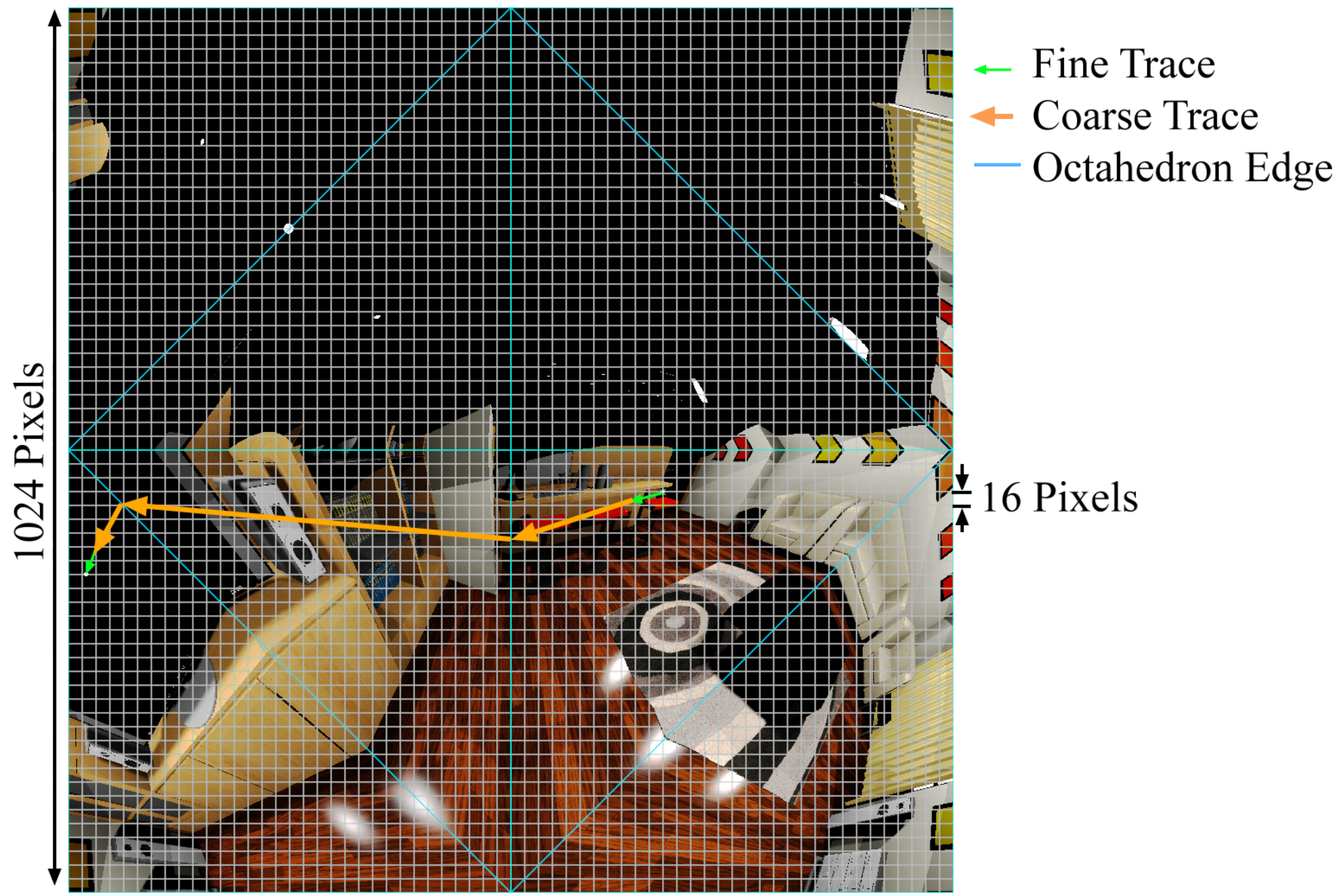
RG8



Radial Distance

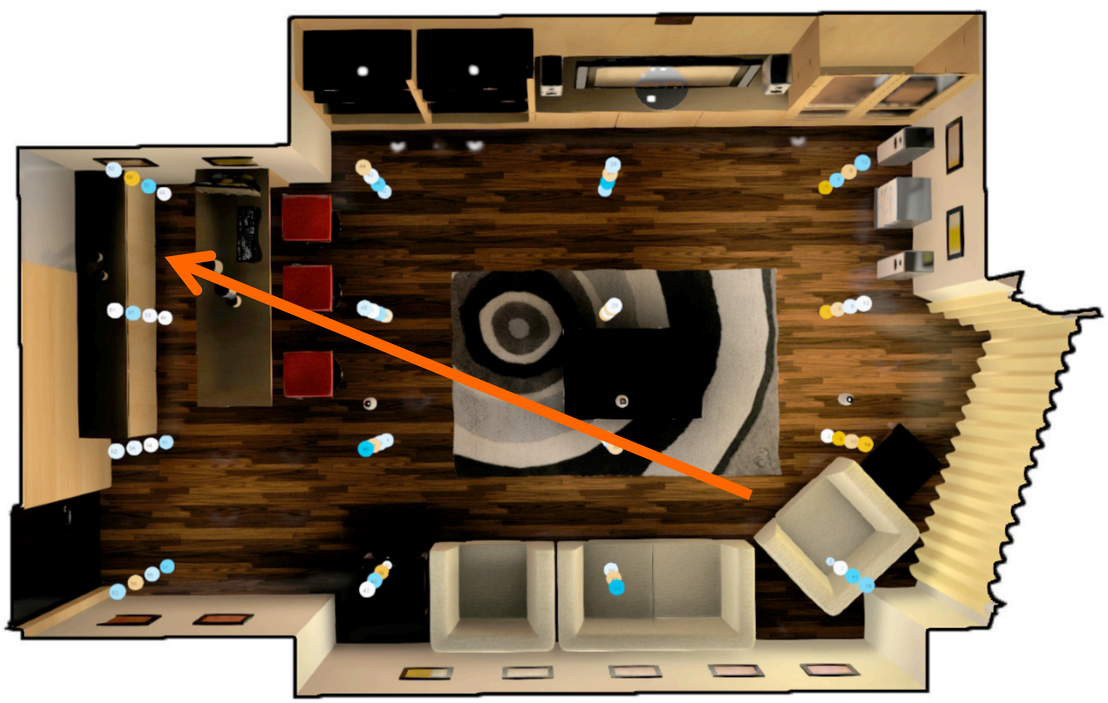
R16F

TRACING THROUGH ONE PROBE

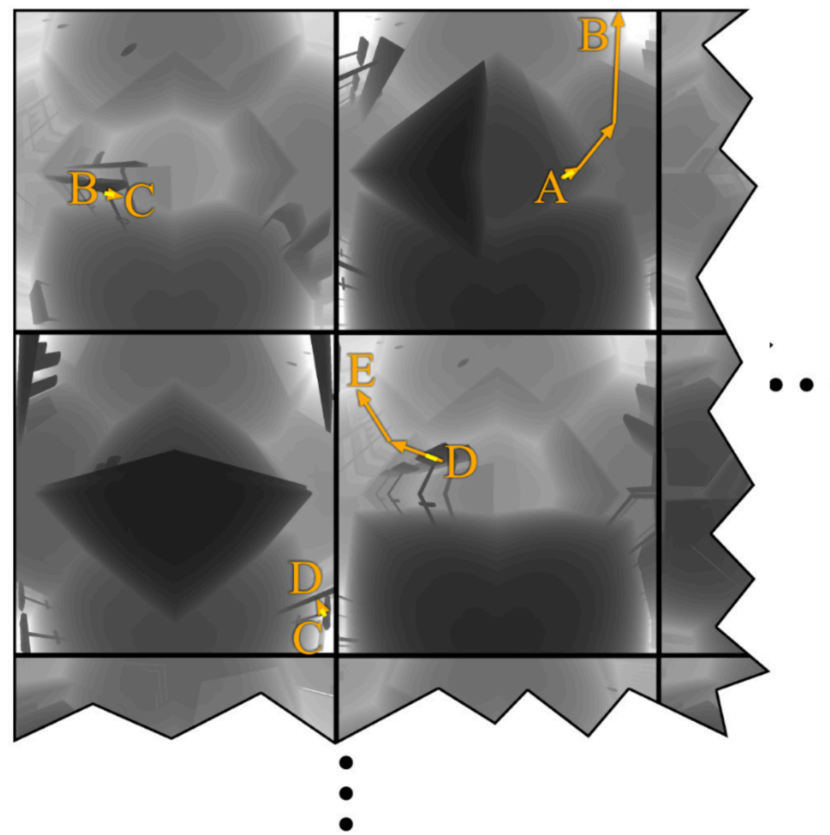


Light Field Probes

TRACING ACROSS MULTIPLE PROBES



Light Probe Locations in Top View

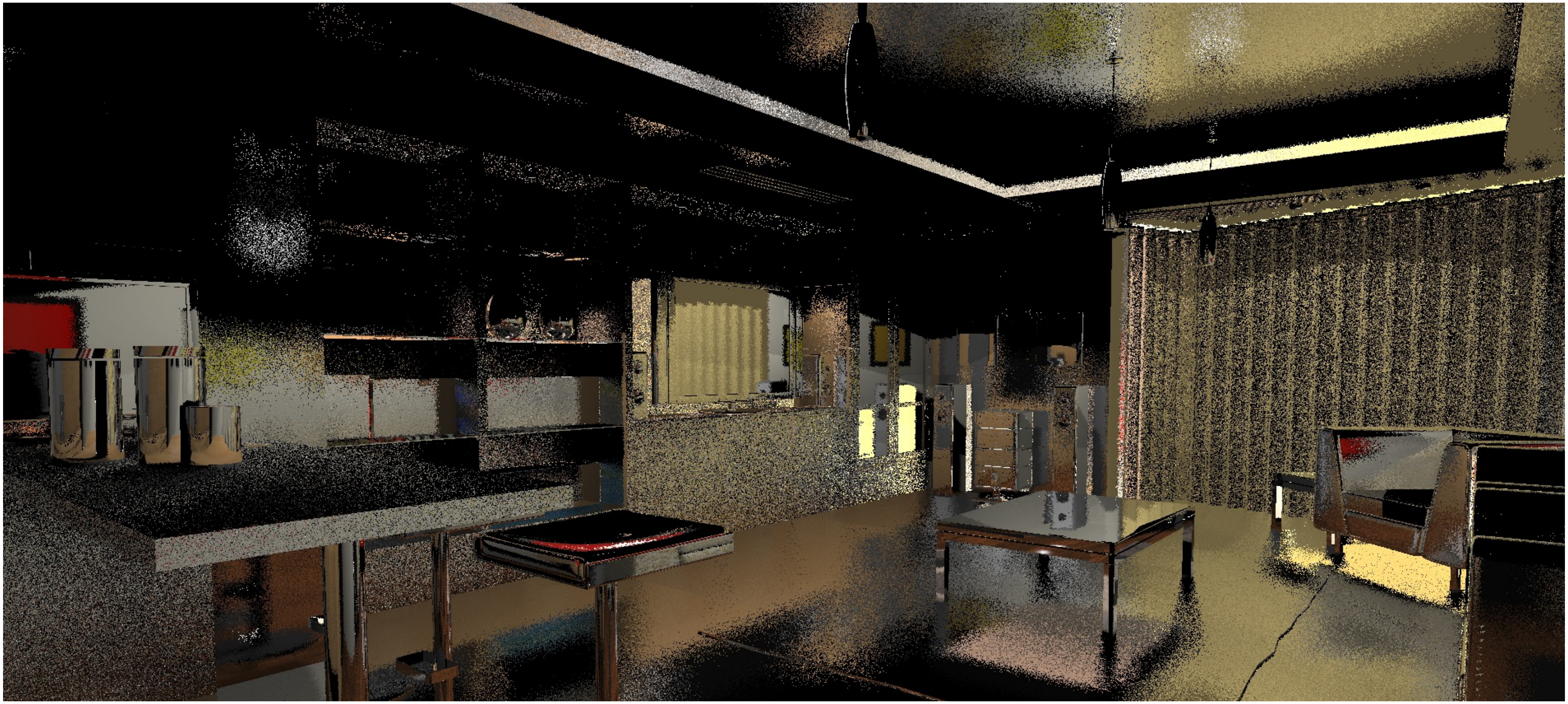


Path of one Ray Through Light Field

```
def singleProbeTrace(ray, probe):
   compute the four 2D polyline segments
   for each polyline segment:
     for each 2D pixel and corresponding 3D point on the segment:
       compare the voxel in the radial distance texture to the ray:
         if hit: return (HIT, point)
         if hidden behind surface: return (UNKNOWN, point)
         # (otherwise, keep iterating)
  return (MISS, last polyline endpoint) # Reached the end of the line

def lightFieldTrace(ray):
  result = UNKNOWN
  while result == UNKNOWN:
     choose the next probe
     (result, endpoint) = singleProbeTrace(ray, probe)
     ray.origin = endpoint # Advance the ray to the last point checked
  return result
```

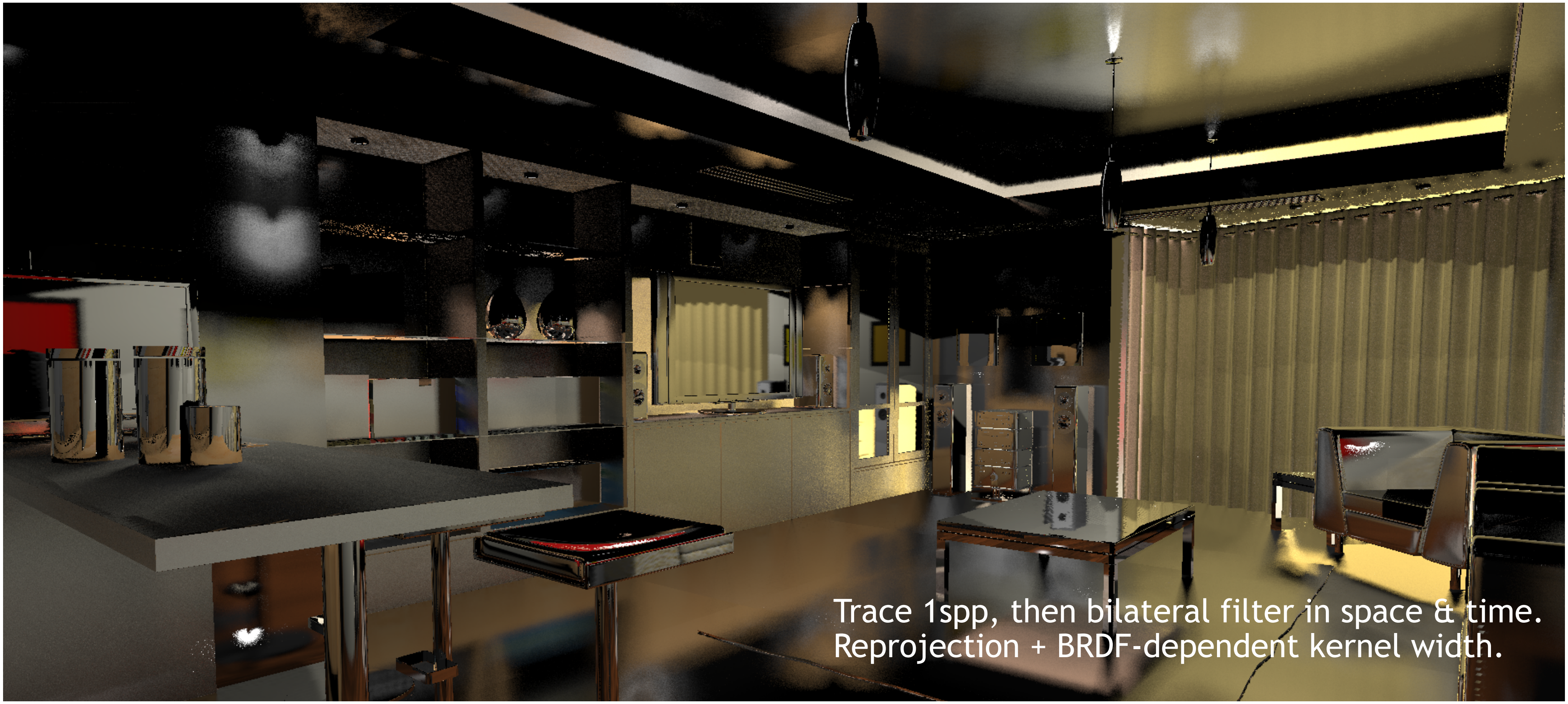
Denoising Example #1

IMPORTANCE-SAMPLED RADIANCE @ 1SPP



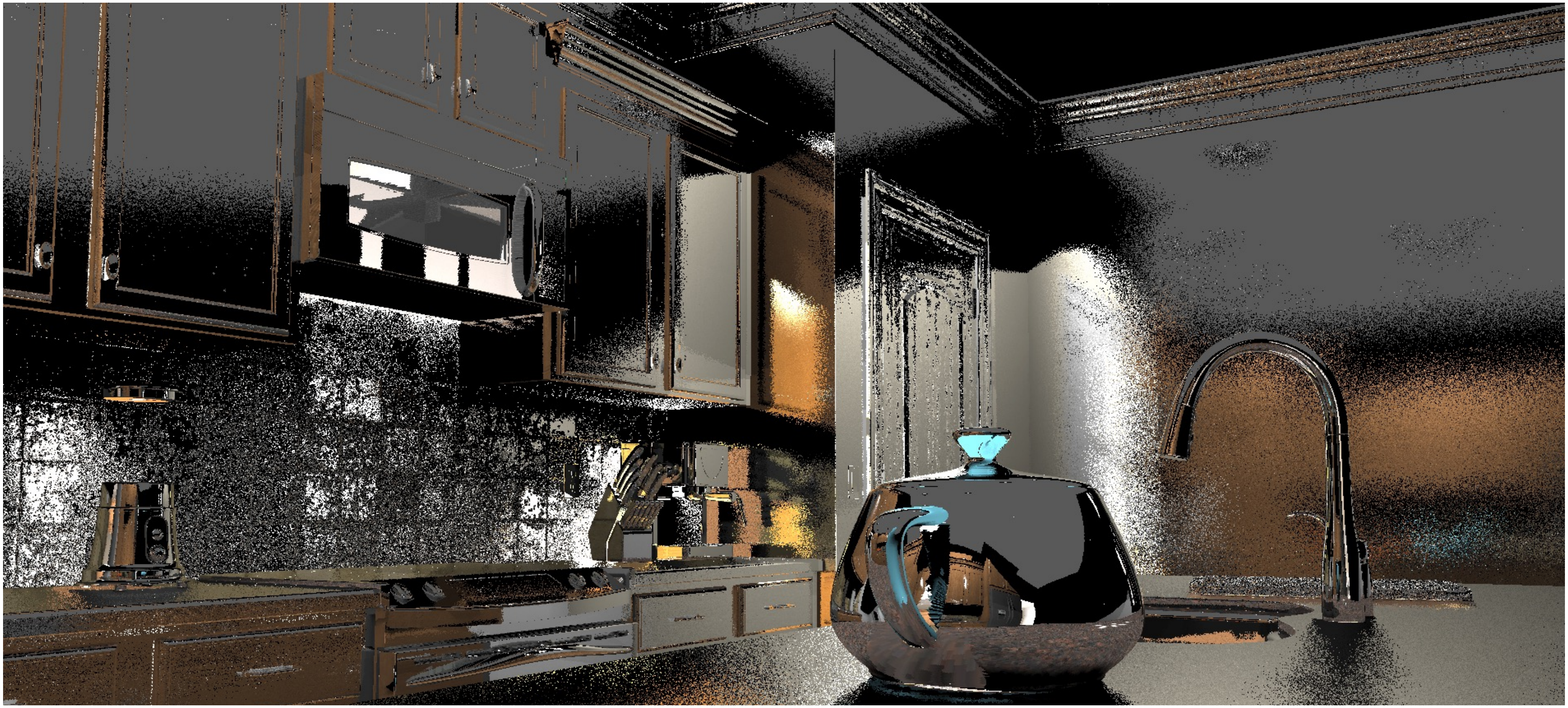
Denoising Example #1

FILTERED RADIANCE



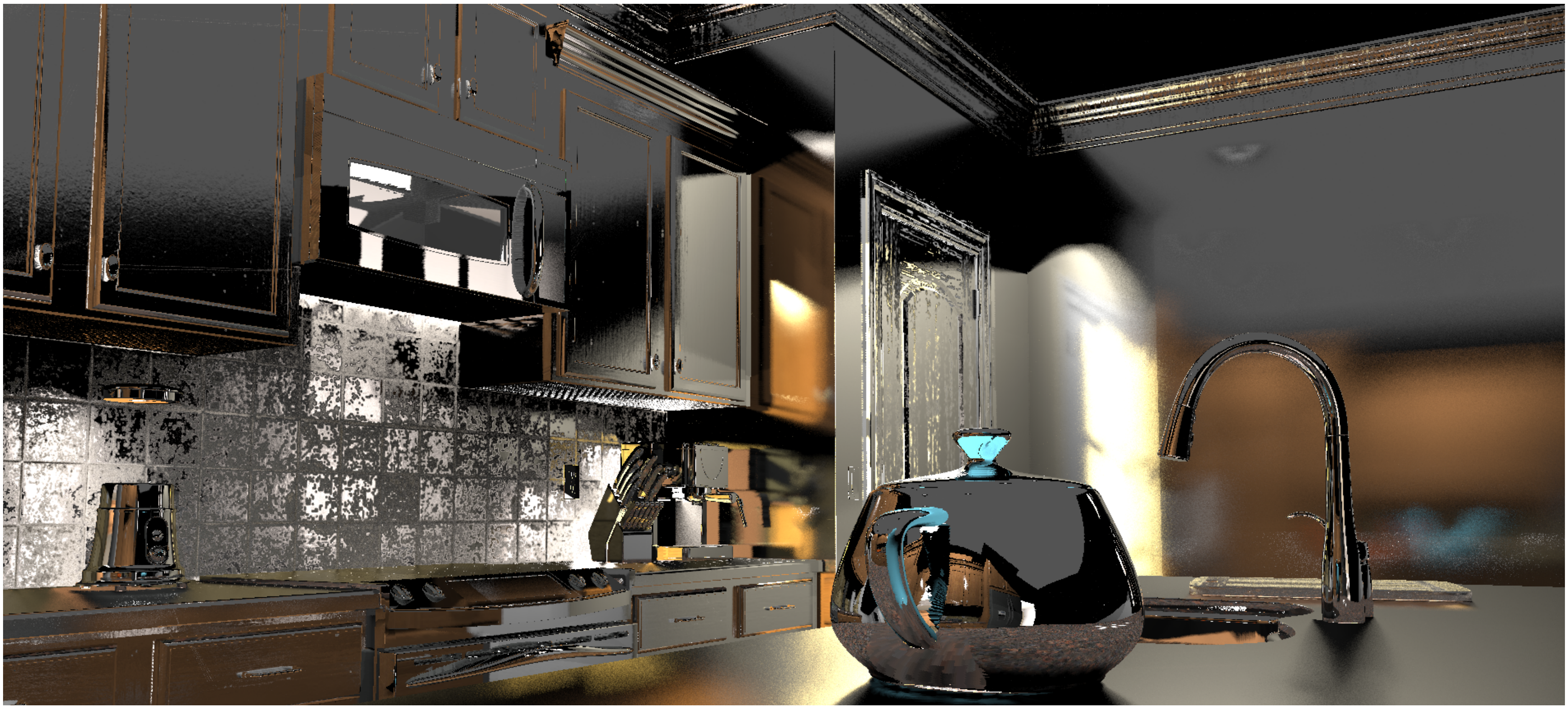
Denoising Example #2

IMPORTANCE-SAMPLED RADIANCE @ 1SPP



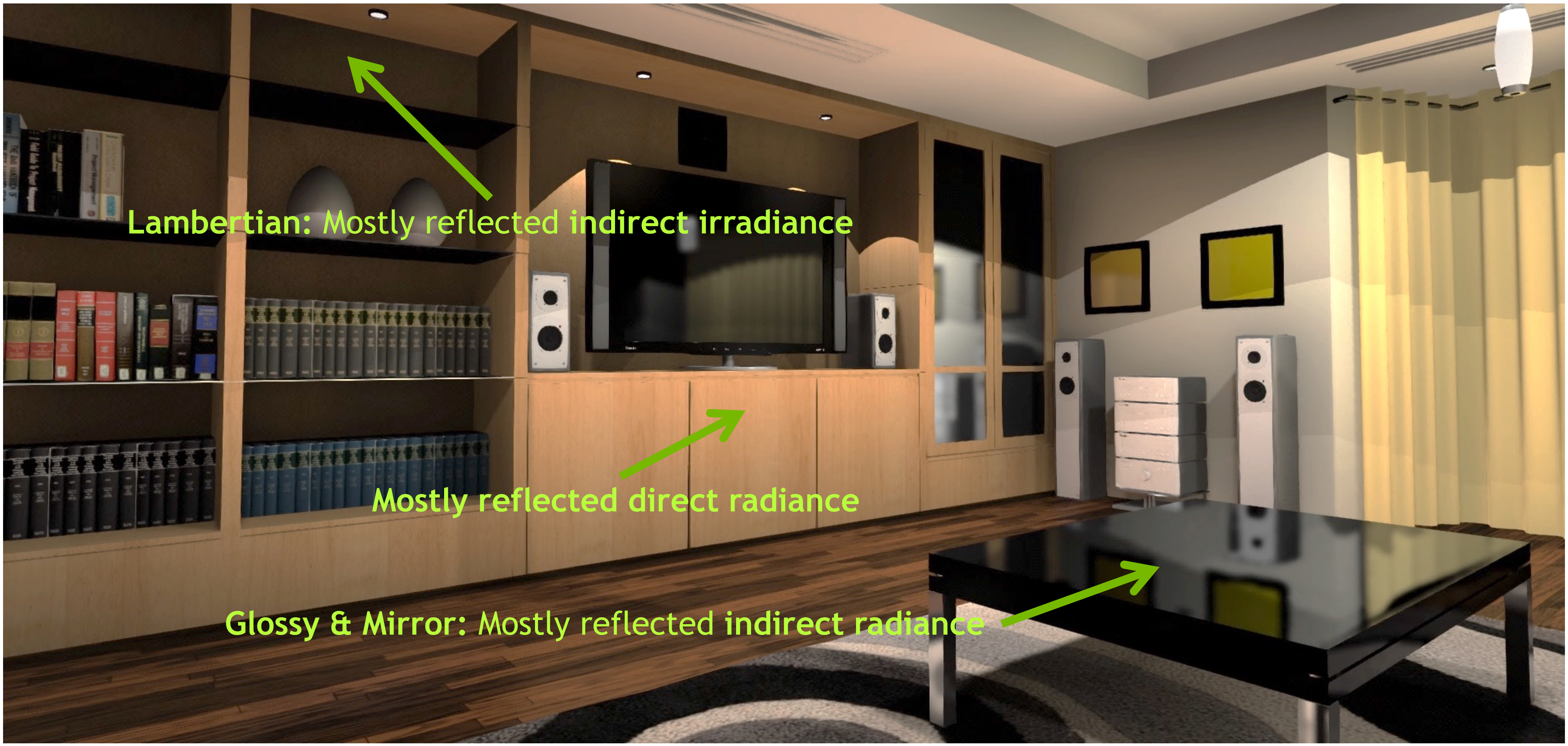
Denoising Example #2

FILTERED RADIANCE

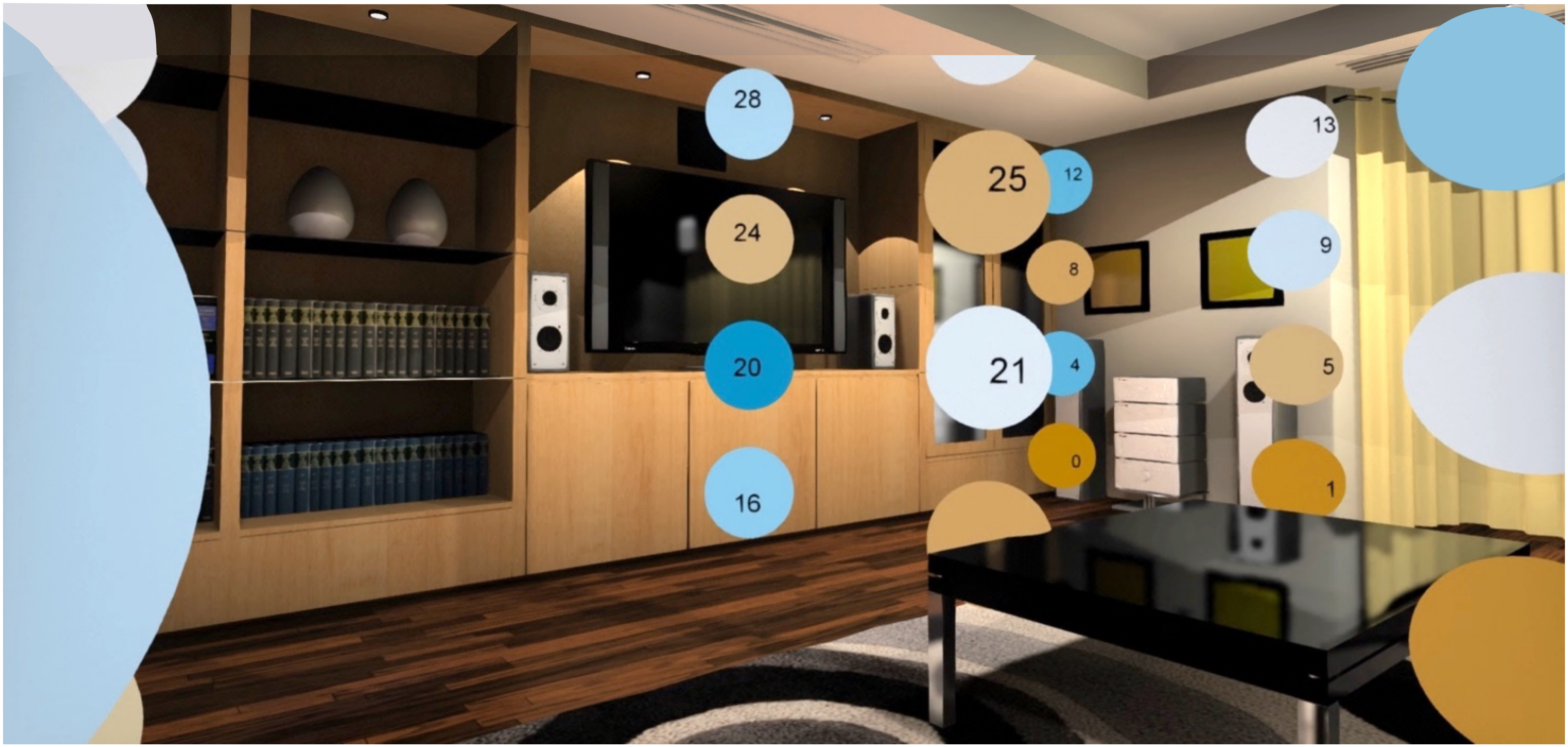


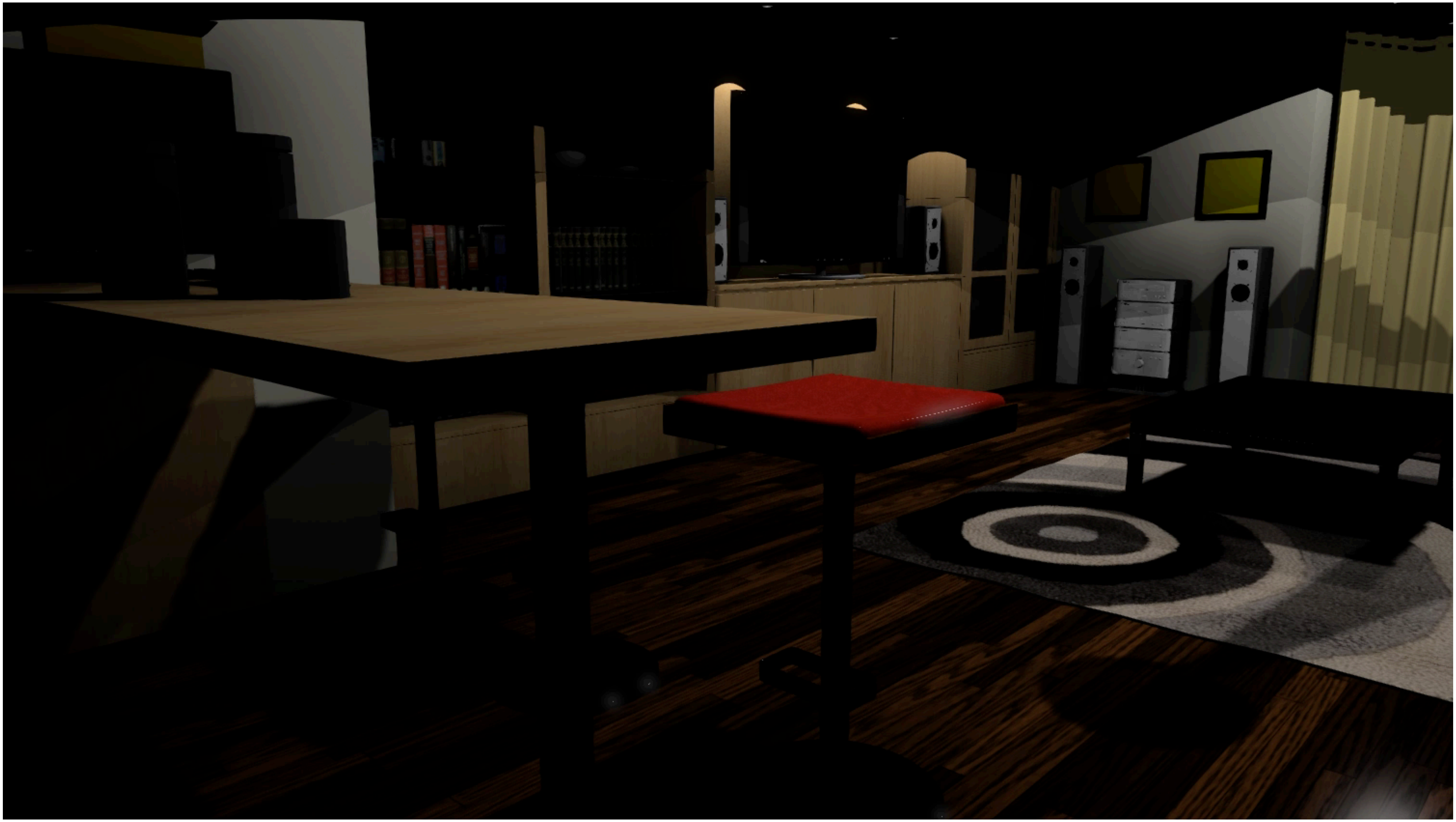
DIRECT ILLUMINATION ONLY

OUR GLOBAL ILLUMINATION RESULT

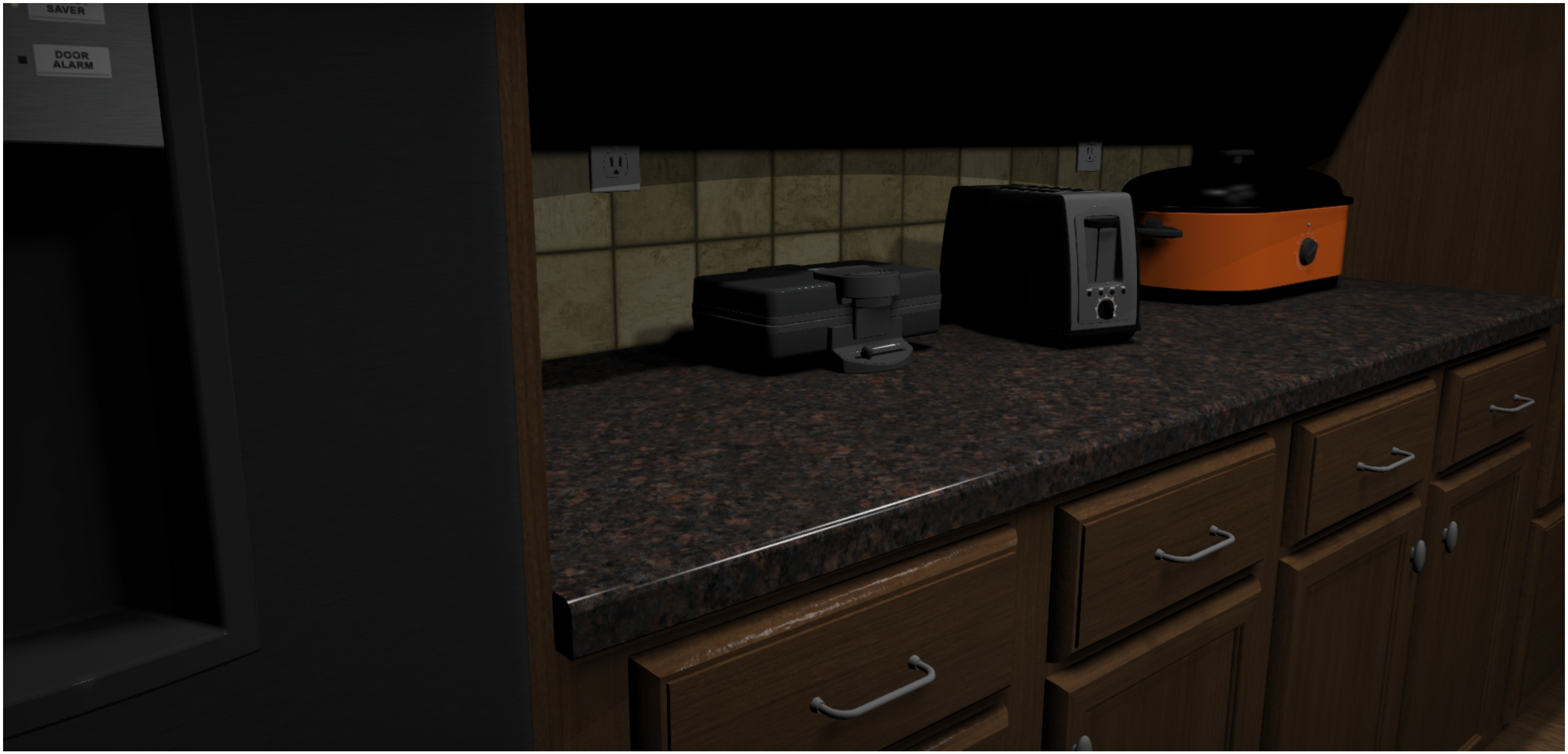


LIGHT FIELD PROBE GRID





DIRECT ILLUMINATION

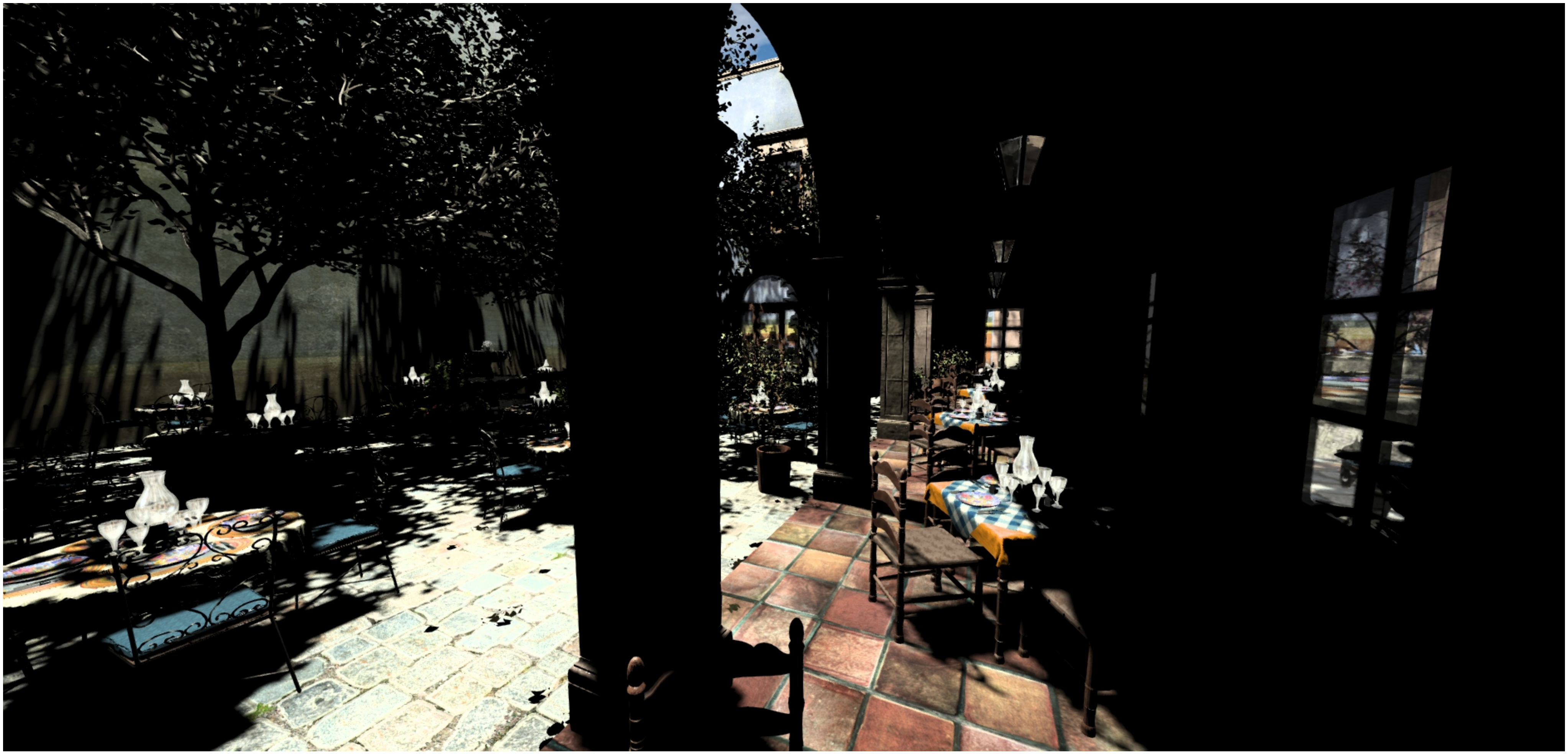


GLOBAL ILLUMINATION

DIRECT ILLUMINATION

GLOBAL ILLUMINATION

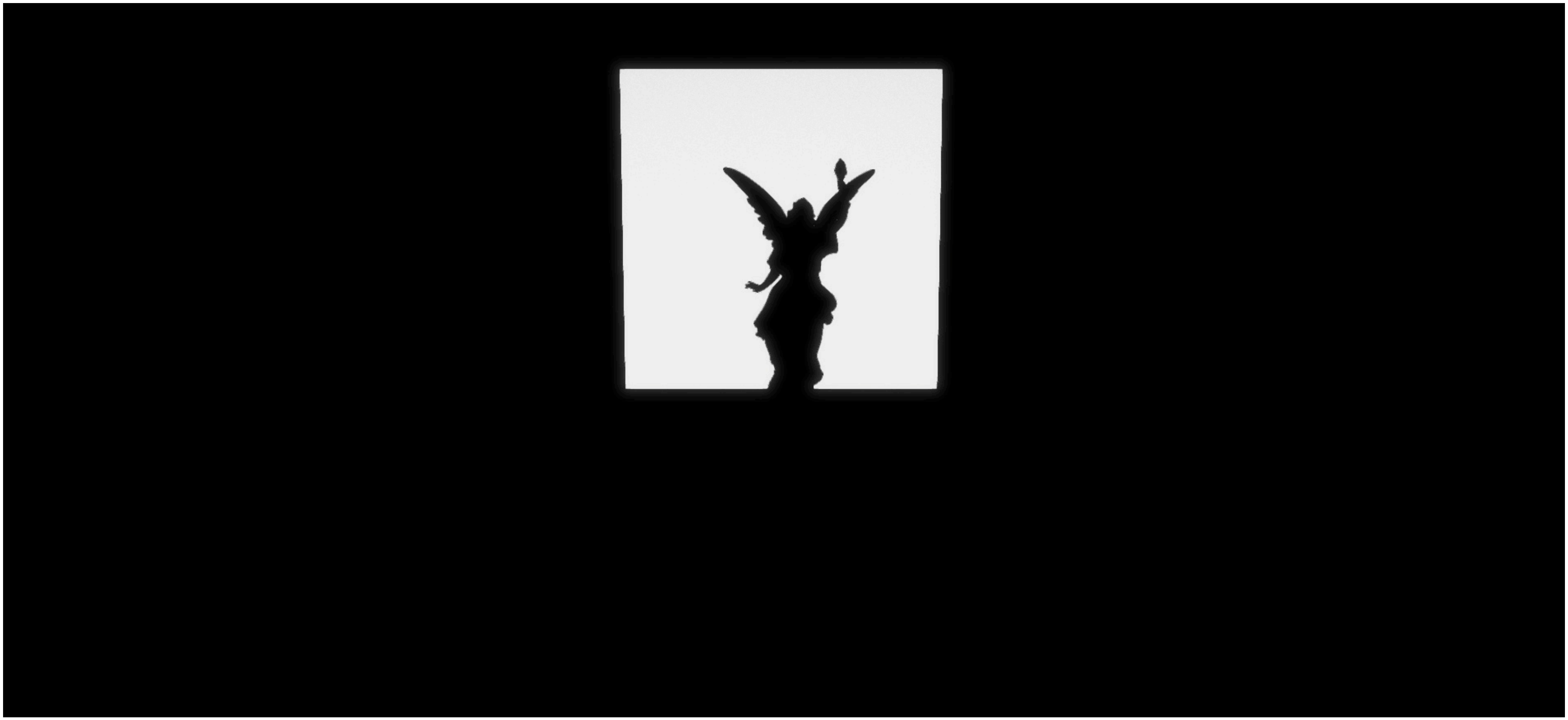
DIRECT ILLUMINATION



GLOBAL ILLUMINATION

MIXED STATIC AND DYNAMIC RECEIVERS

DIRECT [POINT SOURCE] ILLUMINATION



GLOBAL ILLUMINATION WITH AREA & SKY LIGHT SHADOWS

NEXT STEPS

Dynamic probe scheduling

Inspired by Martin and Einarsson 2012

For low-res irradiance probes, just EWMA-filter + real-time ray trace

Light field compression

Inspired by Chang et al. 2006, Hurlburt & Geldreich 2017 [Basis]

SUMMARY

1. Irradiance Probes with Visibility

(*Deployable now*)

Extend existing irradiance tech.

Fixes light leaks: no per-probe artist time

0.35 ms/ frame @ 1080p on GeForce 1080

2. Light Field Probes

(*Preview of ongoing R&D*)

Extend screen-space ray tracing tech.

Fixes all SSR problems

10 ms/ frame @ 1080p on GeForce 1080

CONCLUSIONS

Addressed real-world problems:

- Light & shadow leaks
- Discontinuities & occlusions
- Authoring time/cost

Robust, filterable pixel-shader ray cast reflections ← longer-term significance

Irradiance probes without leaks ← shorter-term significance

Spatio-temporal denoising ← great for all stochastic effects

Code online at http://bit.ly/2mQYlwG

THANKS

David Lubke & Marco Salvi (NVIDIA)

Michael Mara (Stanford)

Derek Nowrouzezahrai (McGill)

Michał Iwanicki (Activision)

Vicarious Visions Visual Alchemy team

BIBLIOGRAPHY

Bjorke, Image-based lighting, GPU Gems 2004

Blinn & Newell, Texture and reflection in computer generated images, Comm. ACM, 1976

Cigolle et al., A survey of efficient representations for independent unit vectors, JCGT 2014

Greene, Environment mapping and other applications of world projections CG&A, 1986

McGuire & Mara, Efficient GPU screen-space ray tracing, JCGT 2014

Musgrave, Grid tracing: fast ray tracing for height fields, Yale tech report 1990

Praun & Hoppe, Spherical parameterization and remeshing, ToG, 2003

Ritschel et al., Imperfect shadow maps for efficient computation of indirect illumination, ToG 2008

Sebastien and Zanuttini, Local image-based lighting with parallax-corrected cube maps, SIGGRAPH 2012 Talk

Lagarde & Seymour, Game environments Part 1: Rendering Remember Me, FX Guide, 2013

Debevec, Image-based lighting, 2006

Donnelly and Lauritzen, Variance shadow maps, I3D 2006

Donow, Light probe selection algorithms for real-time rendering of light fields, Williams College Thesis 2016

Evangelakos, A light field representation for real time global illumination, 2015

Szirmay-Kalos et al., Approximate ray-tracing on the GPU with distance impostors, CGF 2005

Widmer et al., An adaptive acceleration structure for screen-space ray tracing, HPG 2015

Duchamp et al., View-based rendering: visualizing real objects from scanned range and color data, EGSR 1990

Wood et al., Surface light fields for 3D photograph, SIGGRAPH 2000

Nguyen et al., Material editing in complex scenes by surface light field manipulation and reflectance optimization, EG 2014

Scherzer et al, Pre-convolved radiance caching, EGSR 2012

Ritschel et al., Approximating dynamic global illumination in image space, I3D 2009

Wyman, An approximate image-space approach for interactive refraction,, SIGGRAPH 2005

Gilabert, Deferred radiance transfer volumes, GDC 2012

Martin and Einarsson, A real time radiosity architecture for video games, SIGGRAPH 2012 Courses

Chang et al., Light field compression using disparity-compensated lifting and shape adaptation, Trans. Image. Processing, 2006

Bitterli et al., Nonlinearly weighted first-order regression for denoising monte carlo renderings. EGSR 2016

Hurlburt & Geldreich, http://binomial.info/blog/2017/2/23/introducing-texture-array-support-in-basis, 2017

Flynn et al., DeepStereo: Learning to Predict New Views from the World's Imagery, CVPR 2016

Valient, Taking Killzone Shadow Fall image quality into the next generation, GDC 2014

Mara et al., Deep G-Buffers for Stable Global Illumination Approximation, HPG 2016

Arikan, Fast and Detailed Approximate Global Illumination by Irradiance Decomposition, 2005

Shanmugam & Arikan, Hardware accelerated ambient occlusion techniques on GPUs, 2007

Mittring, Finding next gen – CryEngine 2, SIGGRAPH 2007b Courses

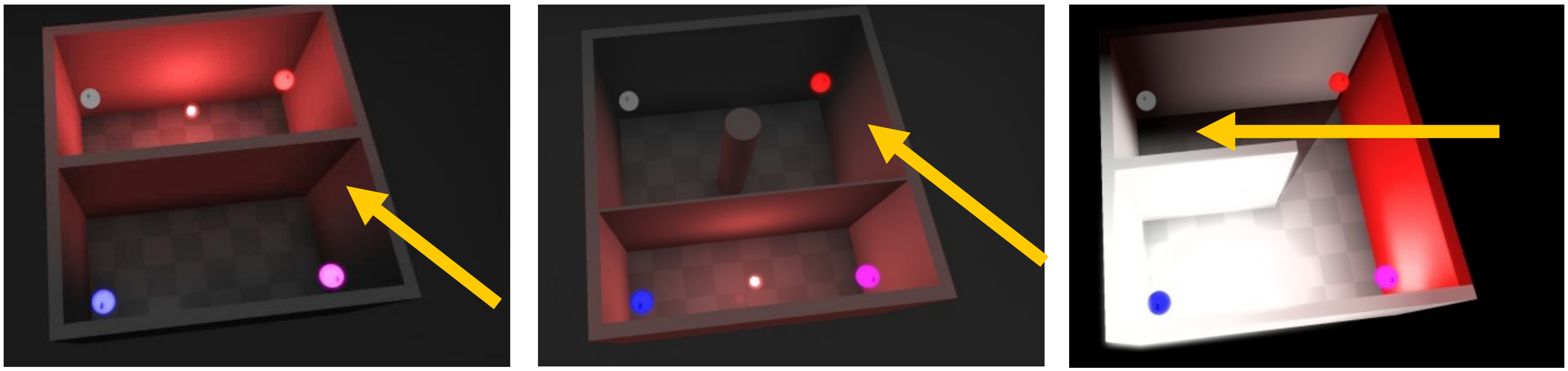
Filion, Starcraft II Effects and Techniques, SIGGRAPH 2008 Courses

Iwanicki, Lighting technology of “The Last of Us”, SIGGRAPH 2013 Courses

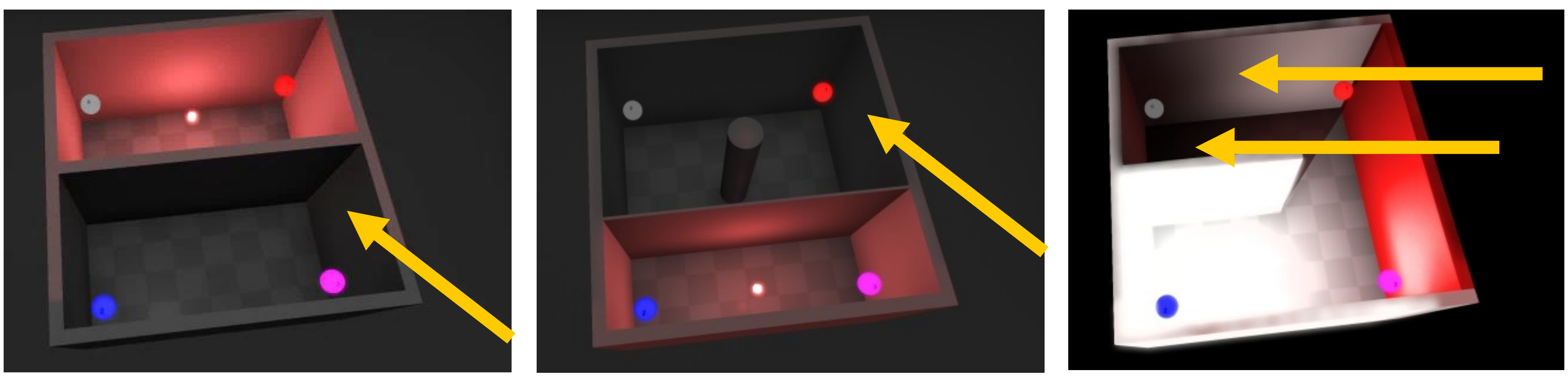
JT Hooker, Volumetric global illumination at Treyarch, SIGGRAPH 2016 Courses

PREFILTERED VISIBILITY MINIMIZES LEAKS

Before: No visibility



After: Our prefiltered visibility



IRRADIANCE

20 years ago, games added "ambient light" and "environment map reflections" to keep areas in shadow from being completely black.

Today, most game engines instead use indirect light equations similar to*

Complicated math, environment probes, and screen-space ray tracing

Material Lambertian color fetched from texture

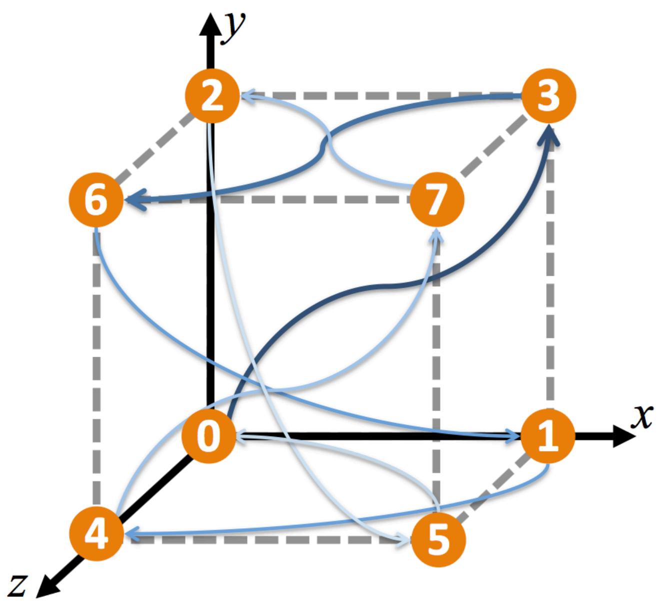
Fresnel coefficient varies with view angle

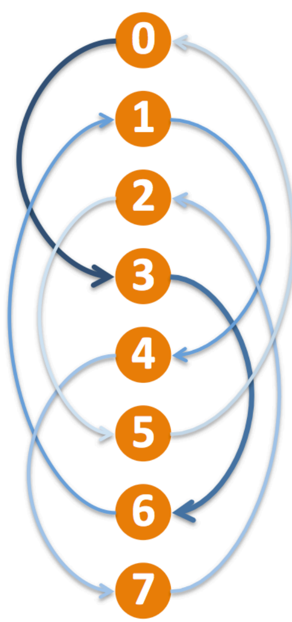
$$\text{shadeIndirect}(\ldots) = \text{lerp}(\text{microfacetStuff}, \boxed{E(X,\hat{n})} \cdot \rho_L, F) / \pi$$

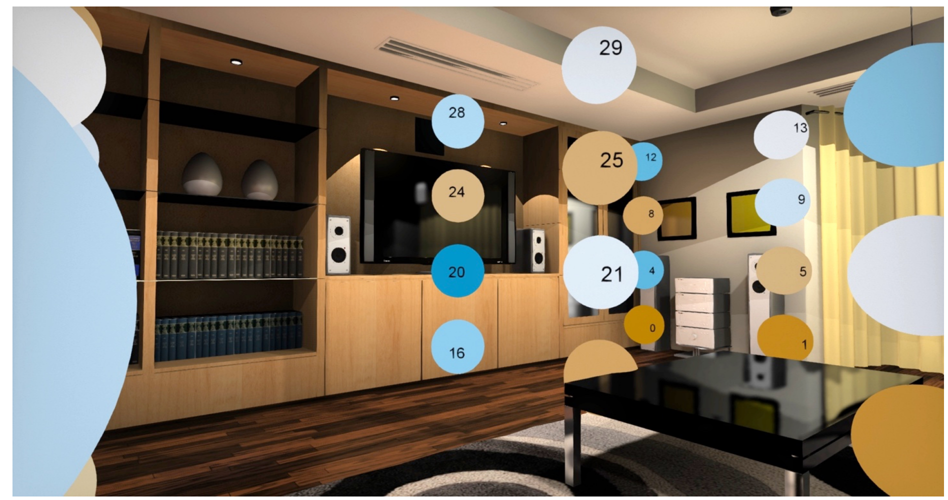
Irradiance: weighted average of incoming indirect light from all directions. Changes (very slowly) with position *X* and surface normal *n*.

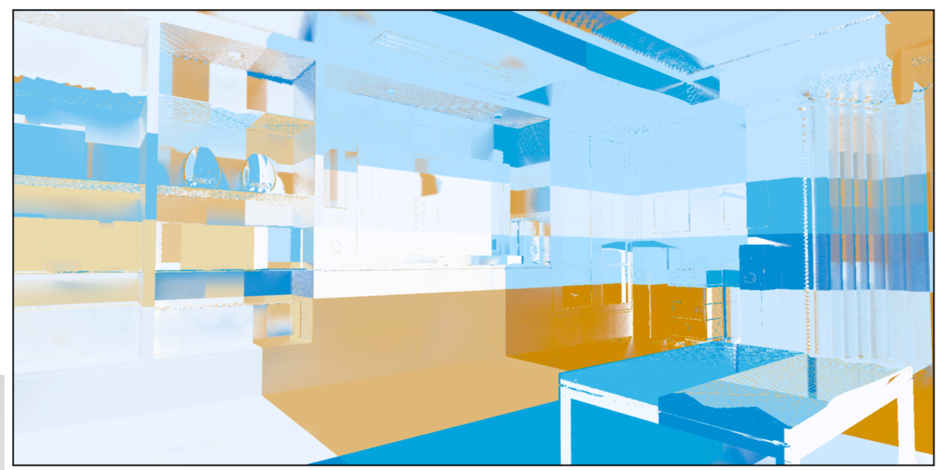
* They are actually factored into lookup textures of precomputed integrals in most engines, but that's not important for today

PROBE SELECTION HEURISTIC









ENVIRONMENT MAP

LIGHT FIELD PROBE RAY TRACE