idc16

Imagination Developers Connection

Real-time ray tracing techniques for developers

March 2016

James Rumble

Overview

Ray Traced Techniques!

- Traditional (Hybrid) Ray Tracing – Shadows, Reflections, Refractions.
- Non-traditional Ray Tracing – Collision Detection.

Shadows, Reflections, Refractions.

Shadows

Why are shadows important?

Shadows provide visual cues for relative object positions.

Shadows help us understand size and shape of objects.

Shadows provide visual cues for light sources.

Shadows add realism.

Rasterized Shadows

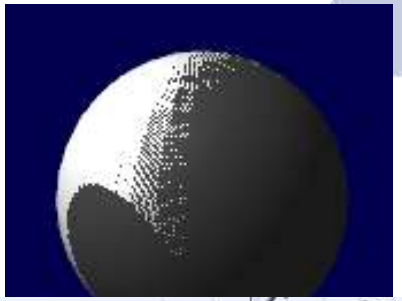
How to Rasterize Shadows

1). Render the scene from the light's point of view.

2). Render the scene from the camera's point of view.

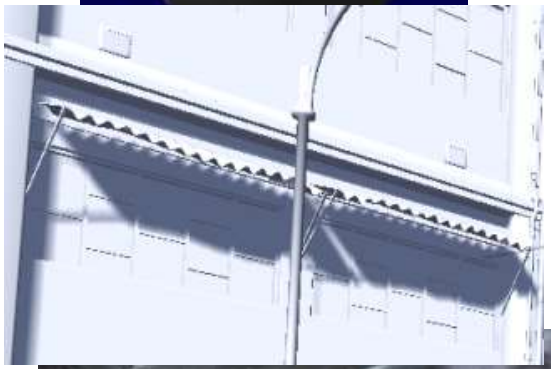
Rasterized Shadows

Common Problems/Fixes



Shadow Acne

- Add bias when sampling shadow maps.

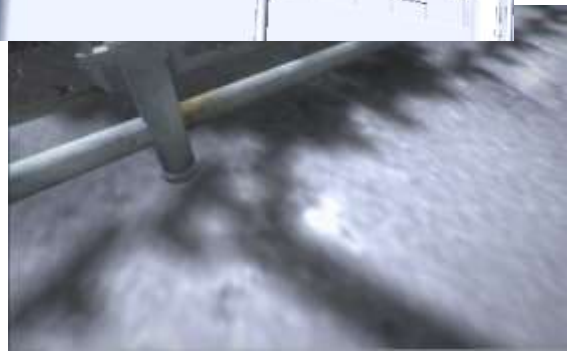


Peter Panning

- How large should this bias be?

Aliasing

- Differing sampling rates across scene.

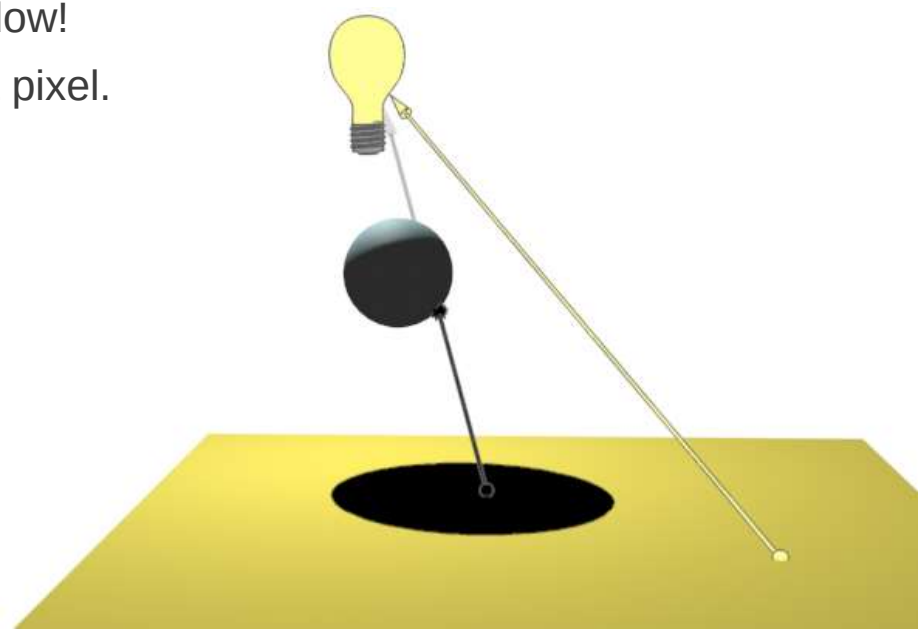


Ray Traced Shadows

How to Ray Trace Shadows (Hybrid)

For each pixel shoot a ray towards the light source:

- If the ray hits anything then do nothing – shadow!
- If the ray reaches the light then illuminate that pixel.
- Use a default ray shader!



Properties

- No Aliasing issues – Screen Space.
- No Shadow acne.

Soft Shadows

Shadows don’t have perfectly sharp edges.

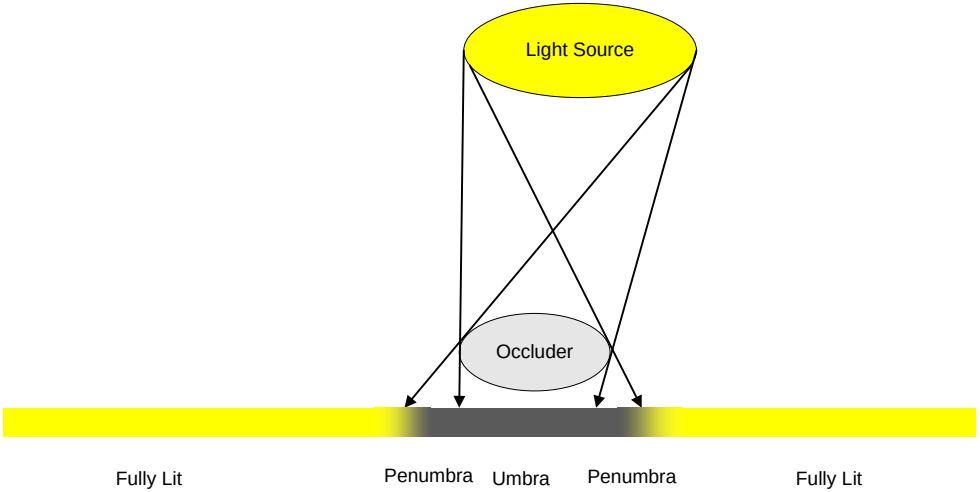
Shadow have penumbra.

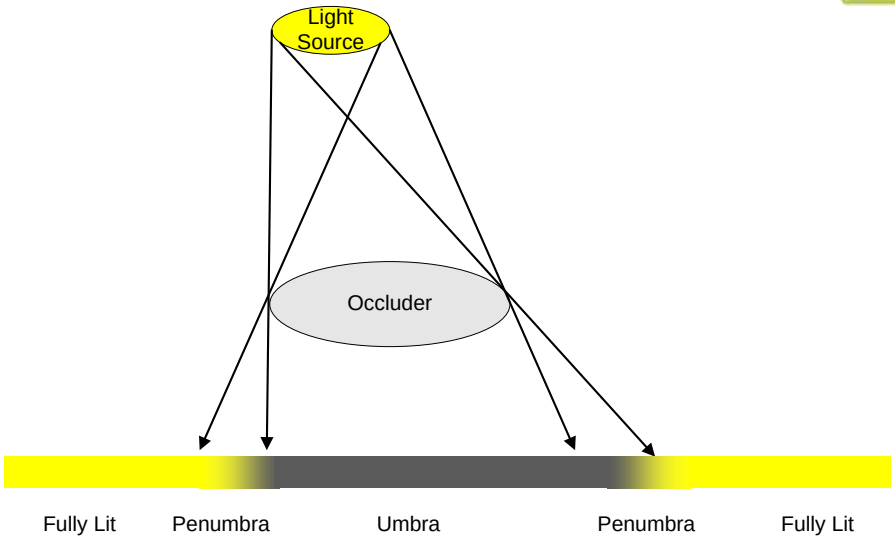
Why?

- Light sources are not infinitely small points.
- Scattering of light.

Soft Shadows

Calculating penumbra size using a simplified model of Soft Shadows





Rasterized Soft Shadows

How to Rasterize Soft Shadows

PCF (Fixed Penumbra)

PCSS (Variable Penumbra)

Variance Shadow Maps

Exponential Shadow Maps

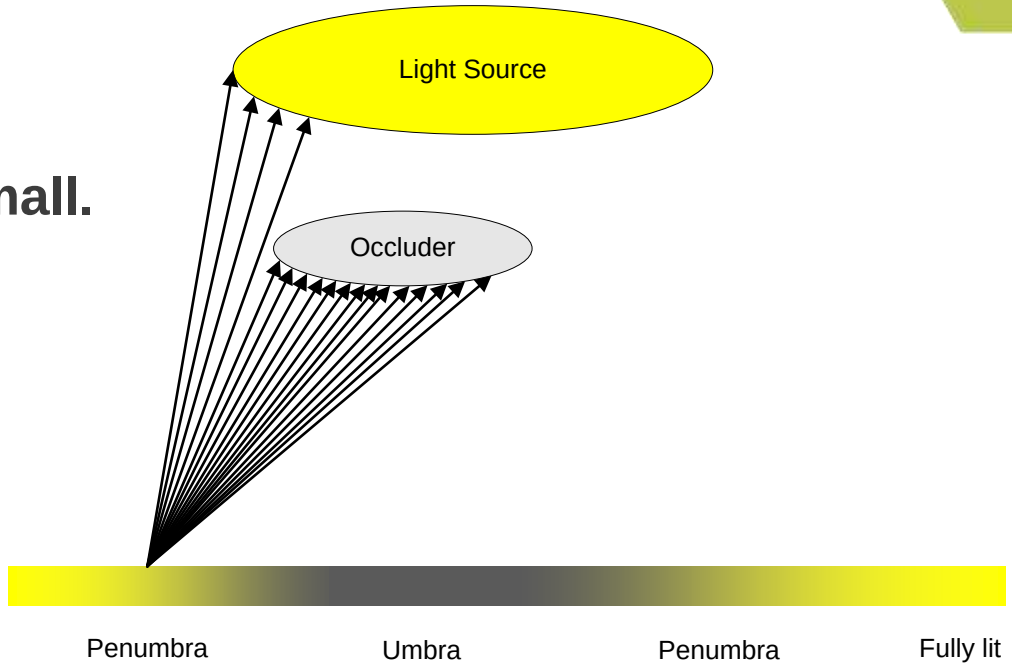
Summed Area Tables

Ray Traced Soft Shadows

How to Ray Trace Soft Shadows (naïve)

Light sources are not infinitely small.

Model an area light source.



Easy but we can do better…

Ray Traced Soft Shadows

How to Ray Trace Real Time Soft Shadows

Use Importance Sampling.

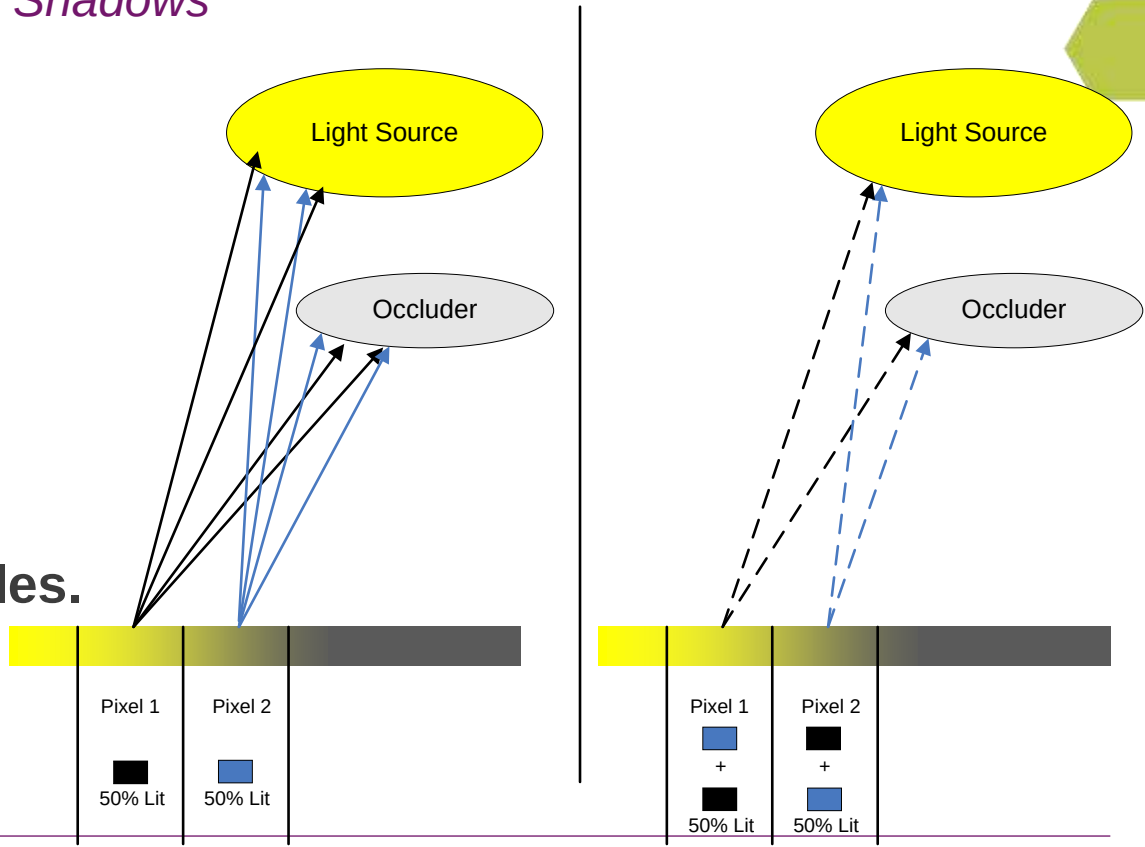
+

Reuse neighbour samples.

=

Fewer rays.

Illusion of using more samples.



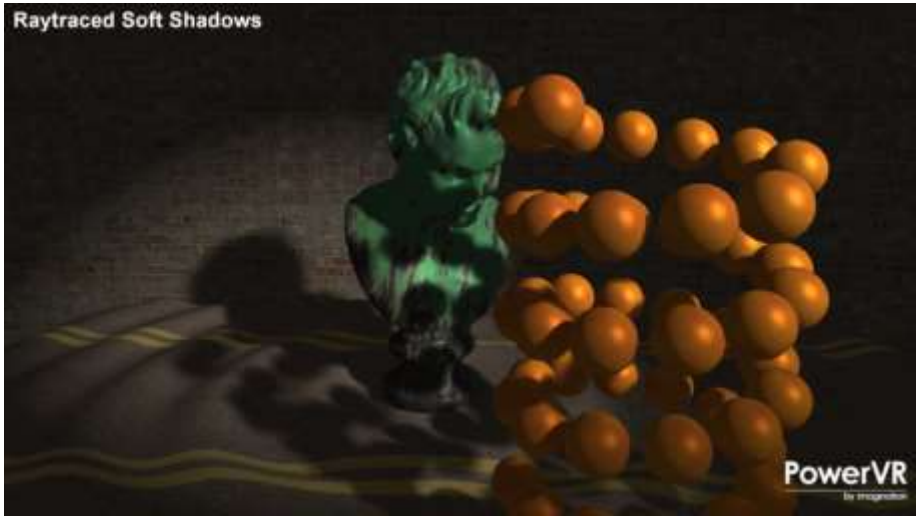
Ray Traced Soft Shadows

How to Ray Trace Real Time Soft Shadows

Reuse concepts introduced by PCSS!

Real-time Ray Traced Soft Shadows

- 1). Generate “noisy” shadow map storing distance to occluder.
- 2). Penumbra size estimation.
- 3). Rotated Poisson Disc filter.



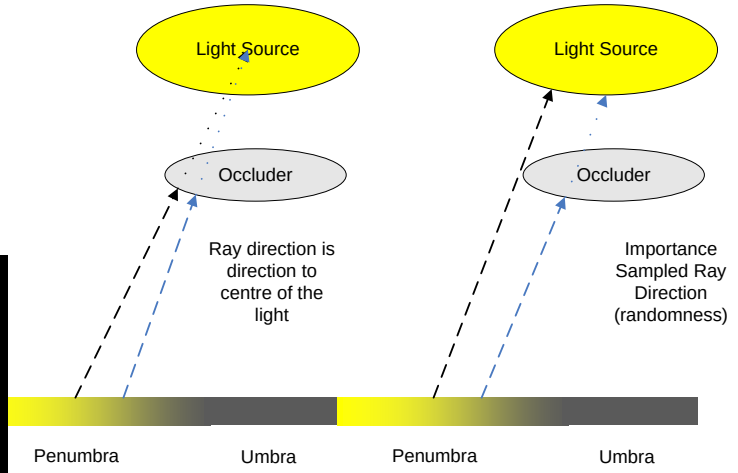
Ray tracing is well suited to tackling step 1!

How to Ray Trace Real Time Soft Shadows

1). Generate “noisy” shadow map storing distance to occluder.

Use pseudo-randomly offset directions.

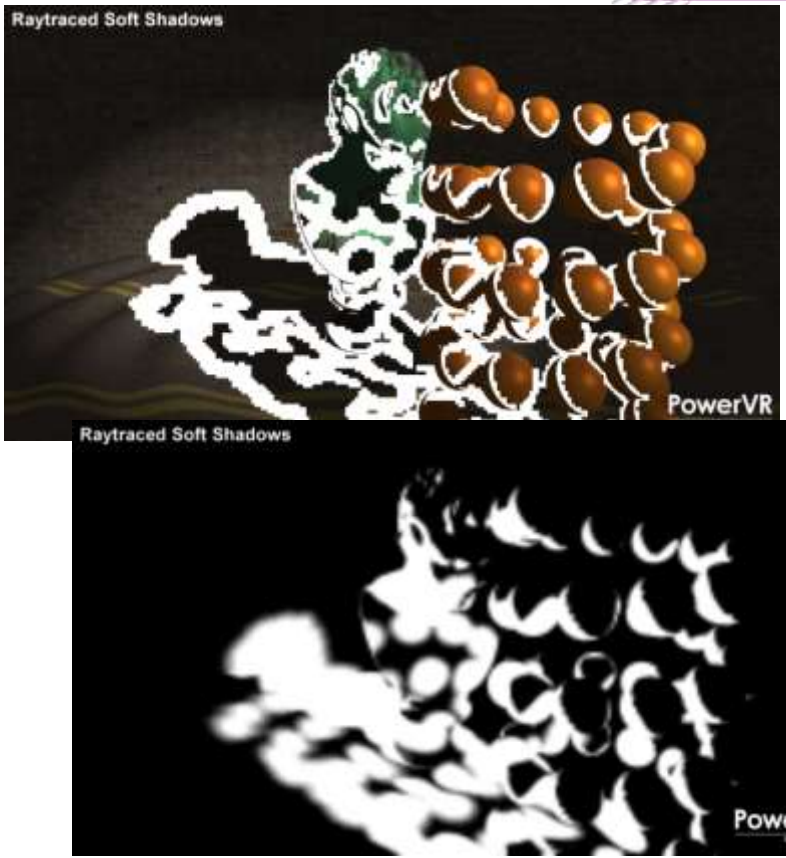
- Makes the world of difference.
- Produces convincing results using only 1 ray per pixel
- The result is undoubtedly “noisy” - addressed later.



How to Ray Trace Real Time Soft Shadows

Use Mip Maps!

Mip Mapped shadows!



Purpose is two fold:

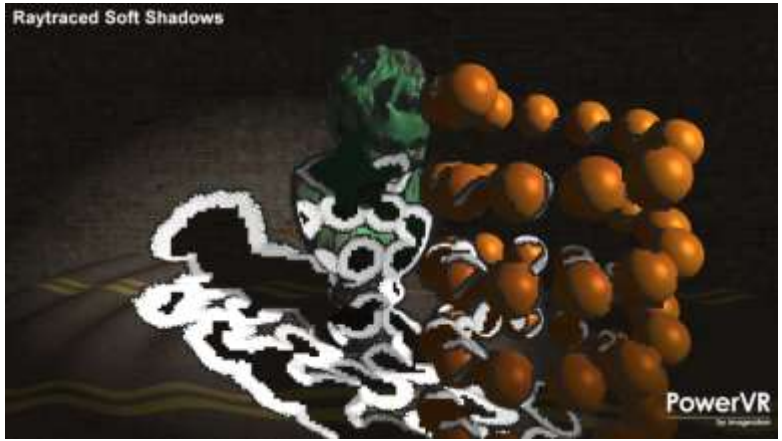
- 1). Helps isolate areas of penumbra requiring additional filtering.
- 2). Mip mapped shadow map can be used directly with additional filtering.

How to Ray Trace Real Time Soft Shadows

2). Penumbra size estimation

Determine screen space filter kernel size using the distance to the occluder, distance to light source and light radius.

- $kernelSize = \frac{lightRadius * distanceToOccluder}{distanceToLight}$

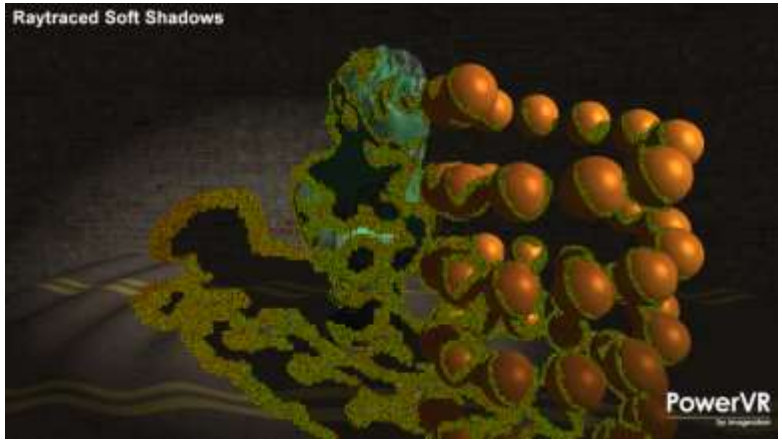


How to Ray Trace Real Time Soft Shadows

3). Rotated Poisson Disc filter

Use a rotated Poisson Disc Filter – more randomness!

- Randomness used to rotate the Poisson Disc.
- Based on the world space position again!
- Introduces additional (desirable) noise.



Reflections and Refractions

When light hits a surface one or more of three things can happen:

- Reflected from surface.
- Transmitted through the surface (Refractions).
- Absorbed by the surface.

Reflections

Reflections are everywhere!

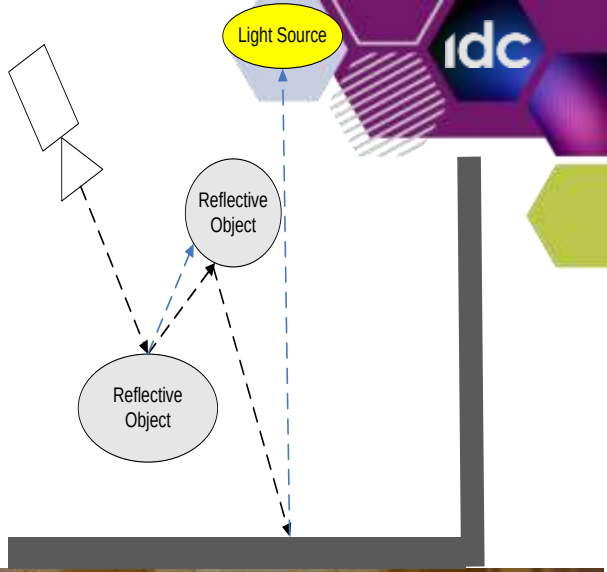
All objects have some reflectivity.

Almost all light perceived from reflections.

Reflections are important for realism.

Focus is on specular reflections today!

How to Ray Trace Reflections



When a ray intersects a reflective object:

- Fire a reflection ray in the reflection direction calculated using the incident direction and the normal.
- Fire shadow ray to determine whether the intersected position is lit.

Easy to integrate Fresnel reflections!

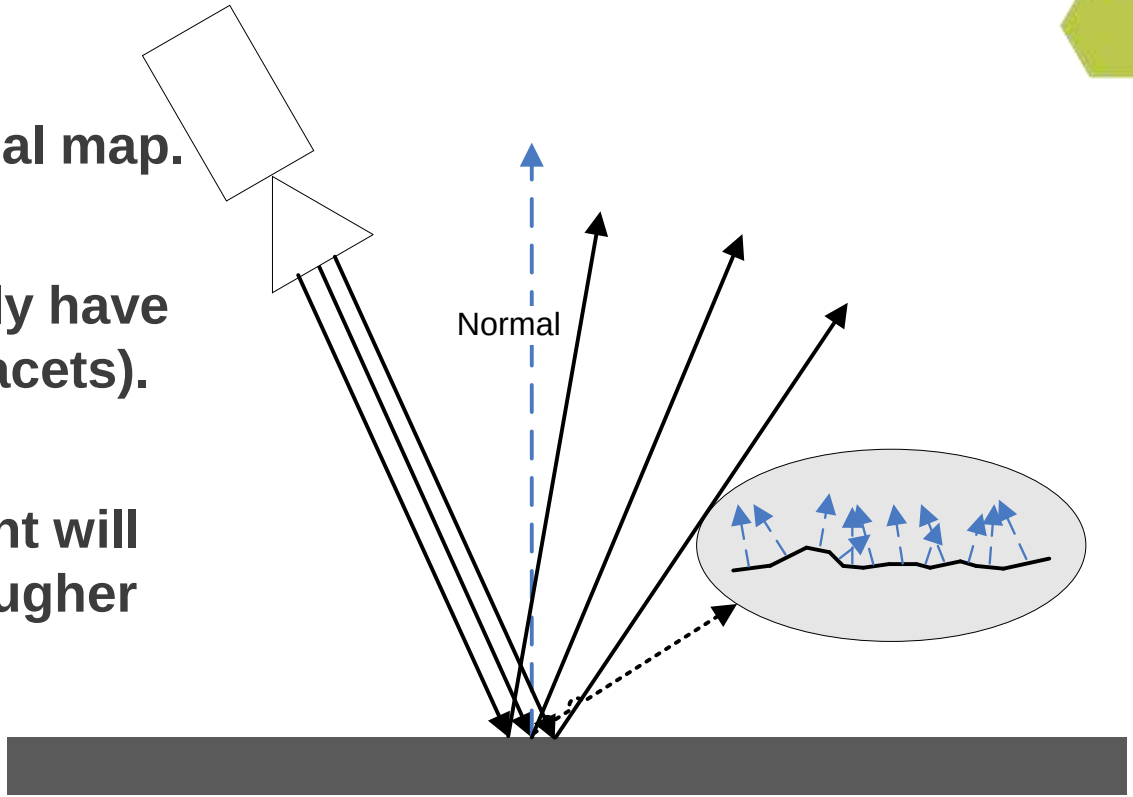
How to Ray Trace Reflections

Microfacets

Reflection direction usually calculated using mesh/normal map.

Real world surfaces generally have many imperfections (microfacets).

Parallel incoming rays of light will diverge when reflecting a rougher surface.



How to Ray Trace Reflections

Glossy Reflections

Mirror reflections not that common!

Realistic reflections:

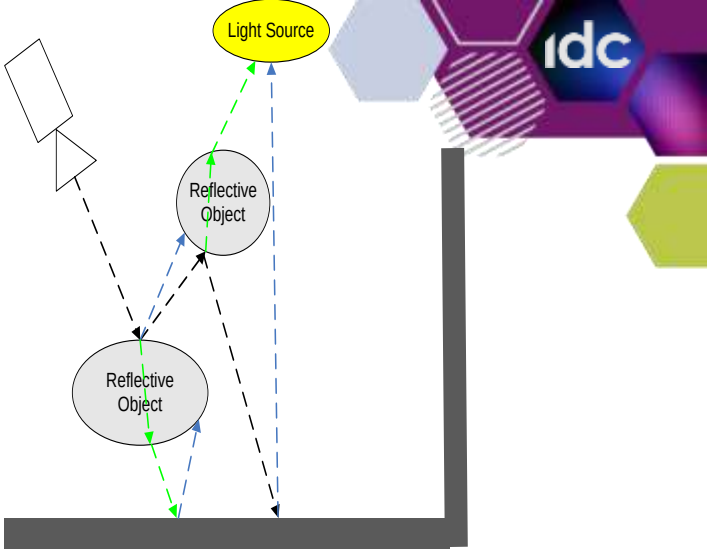
- Contact hardening.
- Roughness of an objects causes blurry reflections.

Similar to Soft shadows:

- Pseudo-random offset derived from roughness.
- Distance to reflected object controls blurring.

Refractions

Light Transmission!



Fresnel term determines the amount of reflectivity vs refractivity.

- Object index of refraction used in Fresnel calculation.

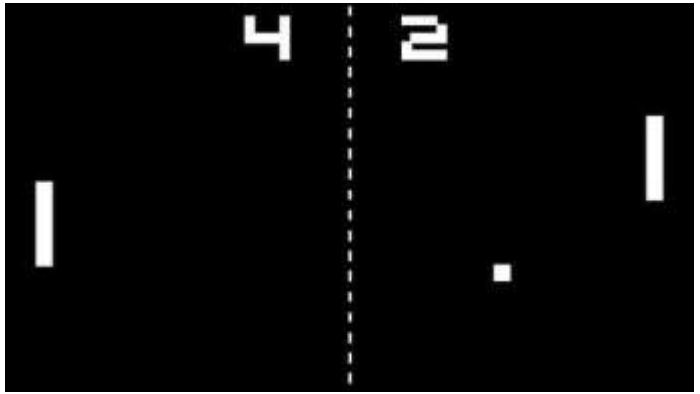
Easy to handle order independent transparency.

Non-Traditional Ray Tracing

Collision Detection

What is Collision Detection?

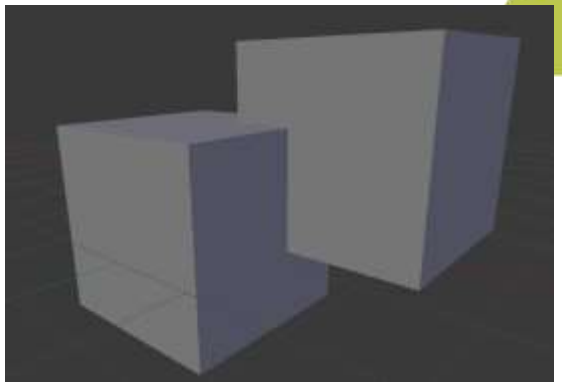
- Detecting whether two objects have collided.



Some amount of collision detection used in all modern games (and some older).

Generally broken into Broad and Narrow phases of collision detection.

Broad phase Collision Detection



What is Broad phase collision detection?

- Identifies pairs of objects with a strong possibility of colliding.

How is it used?

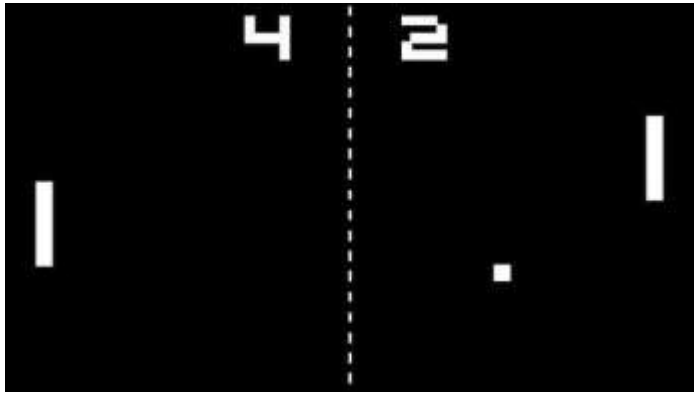
- Represent objects using bounding boxes – commonly AABBs.
- Testing for overlap between two boxes is easy – just a bunch of comparisons.
- N^2 comparisons is a bad idea though… $(x(x-1)\ /\ 2)$

Needs something smarter…

Collision Detection

How can Ray Tracing help?

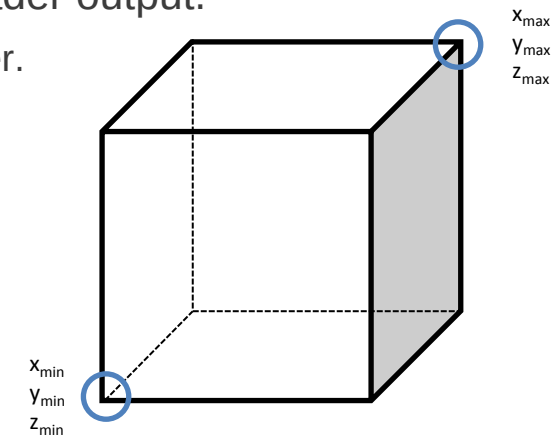
- Ray tracing gives us the ability to determine whether a ray intersects an object/triangle.
- If we can use this logic to support object vs object collisions we are sorted.



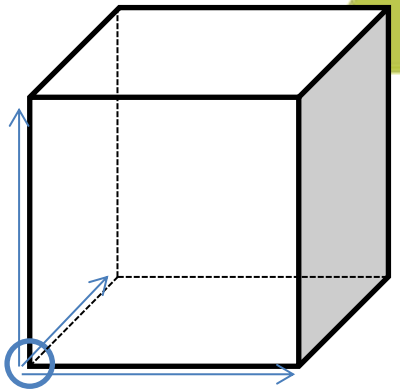
Ray Traced Broad Phase Collision Detection

Our implementation uses (1)Compute, (2)Ray tracing SHG (3) Ray Tracing RTU) and (4)Rasterization.

- (1) Calculate AABB world space min/max positions using Compute shader.
  - Recalculate bounding box from cached bounding box!
- (2) Rebuild bounding box Scene Hierarchy using Compute shader output.
  - Compute shader output feeds as input to Bounding Box Vertex shader.

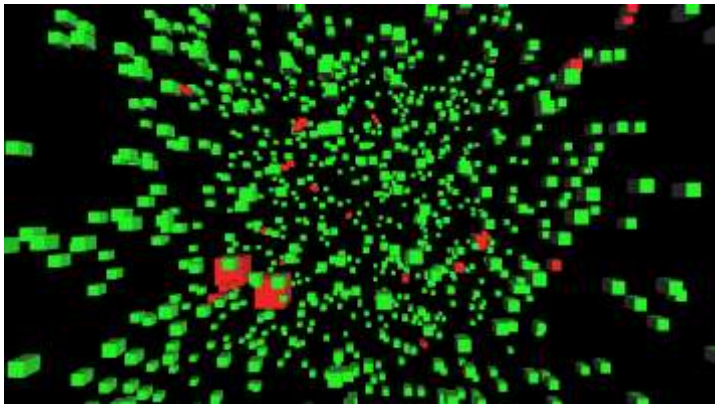


Ray Traced Broad Phase Collision Detection



Our implementation uses (1)Compute, (2)Ray tracing SHG (3) Ray Tracing RTU) and (4)Rasterization.

- (3) Use min/max positions in Frame shader to fire 3 rays per AABB from min along axis of the box (x,y,z).
  - Fixed function hardware takes care of the hard work.
  - Ray shaders handle collection of colliding pairs.
- (4) Render AABBs using instancing and color depending collision result.



Future Work

Extend to off-axis boxes.

Ray Traced Narrow phase.

Conclusion

Traditional (hybrid) graphical effects.

Non-traditional ray tracing.

Coming up

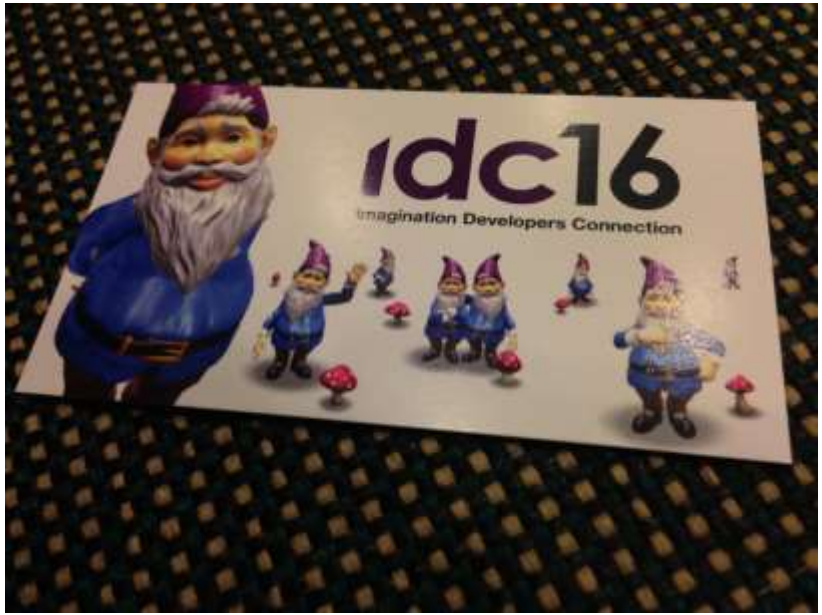
- 12:20-12:50 So you want to build a lightmapper?
- 12:50-1:20 Advanced Topics for PowerVR Ray Tracing

Questions?

Get in touch:

- *James.Rumble@imgtec.com*
- *@powervrinsider*

Please come and visit us at booth #1902 in the South Hall to see our demos and to collect your very own Vulkan™ Gnome t-shirt!



www.imgtec.com/idc