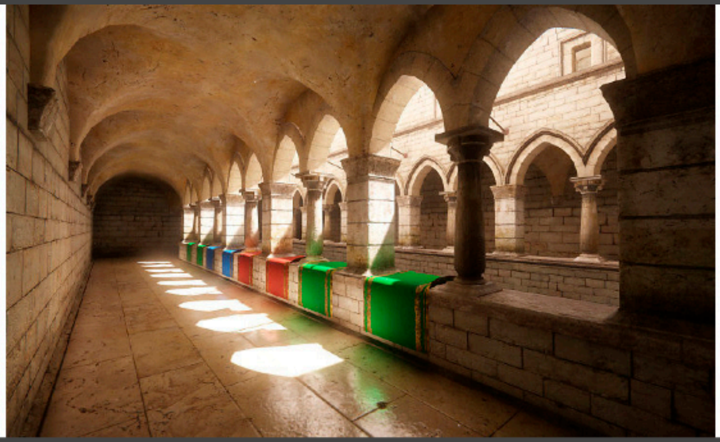
Real-Time High Quality Rendering

GAMES202, Lingqi Yan, UC Santa Barbara

Lecture 9:
Real-Time Global Illumination
(screen space cont.)

Announcements

- Happy Labor Day!
- Milestone: 2/3 contents covered after today's lecture!
- GAMES101 homework resubmission
  - Still recruiting graders!
  - I'm considering TAs instead of graders now
- GAMES202 homework late submission
  - Will start after HW2 is due
- I may suddenly cancel any lecture before May 20

Errata

- In RSM (Lecture 7)
  - Both my derivation and the equation from paper are CORRECT

$$
\begin{aligned}
L_o(\mathrm{p}, \omega_o) &= \int_{\Omega_{\text{patch}}} L_i(\mathrm{p}, \omega_i)\, V(\mathrm{p}, \omega_i)\, f_r(\mathrm{p}, \omega_i, \omega_o)\, \cos\theta_i \,\mathrm{d}\omega_i \\
&= \int_{A_{\text{patch}}} L_i(\mathrm{q} \to \mathrm{p})\, V(\mathrm{p}, \omega_i)\, f_r(\mathrm{p}, \mathrm{q} \to \mathrm{p}, \omega_o)\, \frac{\cos\theta_p \cos\theta_q}{\|q - p\|^2} \,\mathrm{d}A
\end{aligned}
$$

$$
E_p(x, n) = \Phi_p \frac{\max\{0, \langle n_p | x - x_p \rangle\} \max\{0, \langle n | x_p - x \rangle\}}{\|x - x_p\|^4}. \qquad (1)
$$

Last Lecture

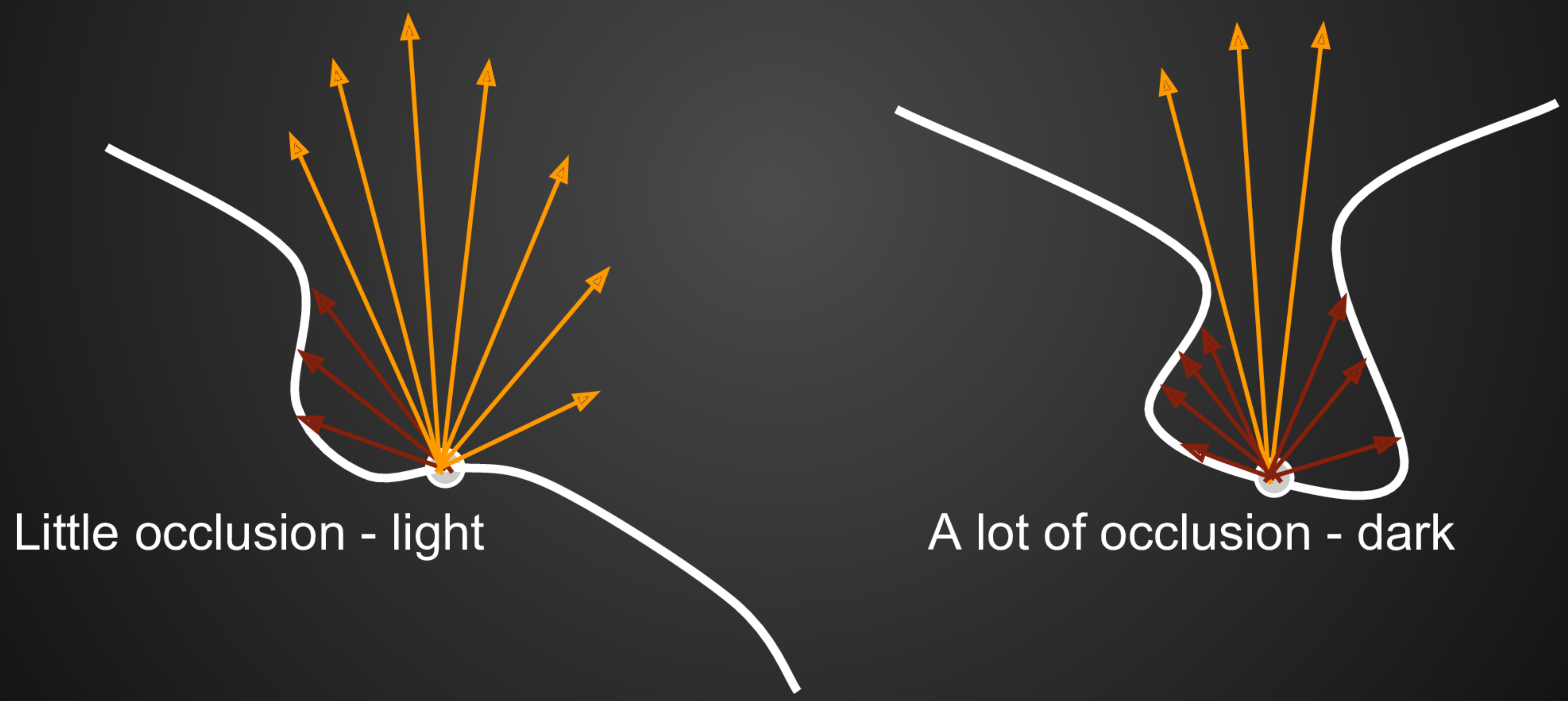
- Real-Time Global Illumination (3D space)
  - Light Propagation Volumes (LPV)
  - Voxel Global Illumination (VXGI)
- Real-Time Global Illumination (screen space)
  - Screen Space Ambient Occlusion (SSAO)
  - Screen Space Directional Occlusion (SSDO)
  - Screen Space Reflection (SSR)

Today

- Real-Time Global Illumination (screen space cont.)
  - Screen Space Directional Occlusion (SSDO)
  - Screen Space Reflection (SSR)
- Real-Time Physically-Based Materials

Recap: Screen Space Ambient Occlusion (SSAO)

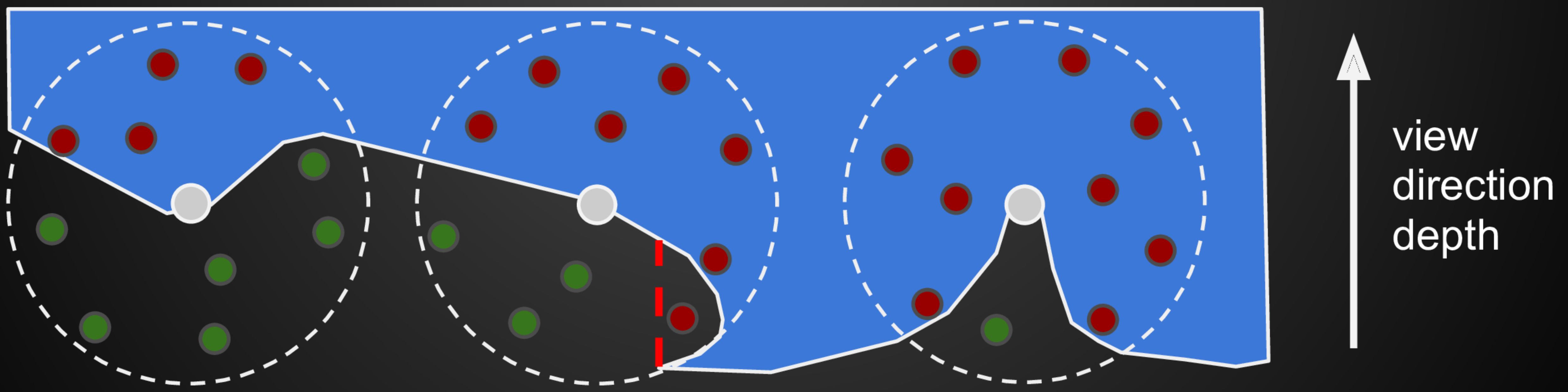
Ambient occlusion



SSAO:
Ambient occlusion using the z-buffer

Use the readily available depth buffer as an approximation of the scene geometry.

Take samples in a sphere around each pixel and test against buffer.

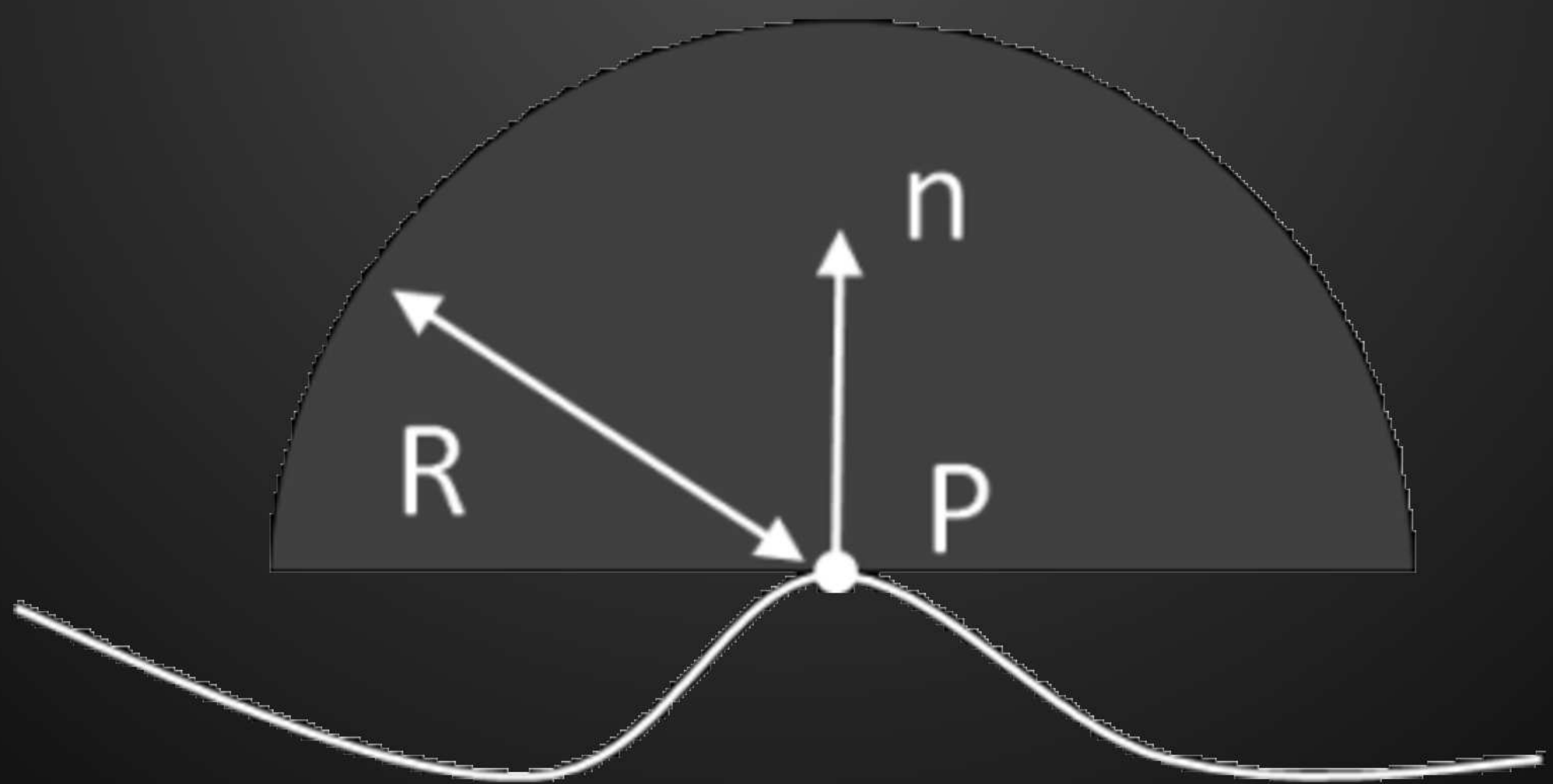


Horizon based ambient occlusion: HBAO

Also done in screen space.

Approximates ray-tracing the depth buffer.

Requires that the normal is known, and only samples in a hemisphere.



Screen Space Directional Occlusion (SSDO)

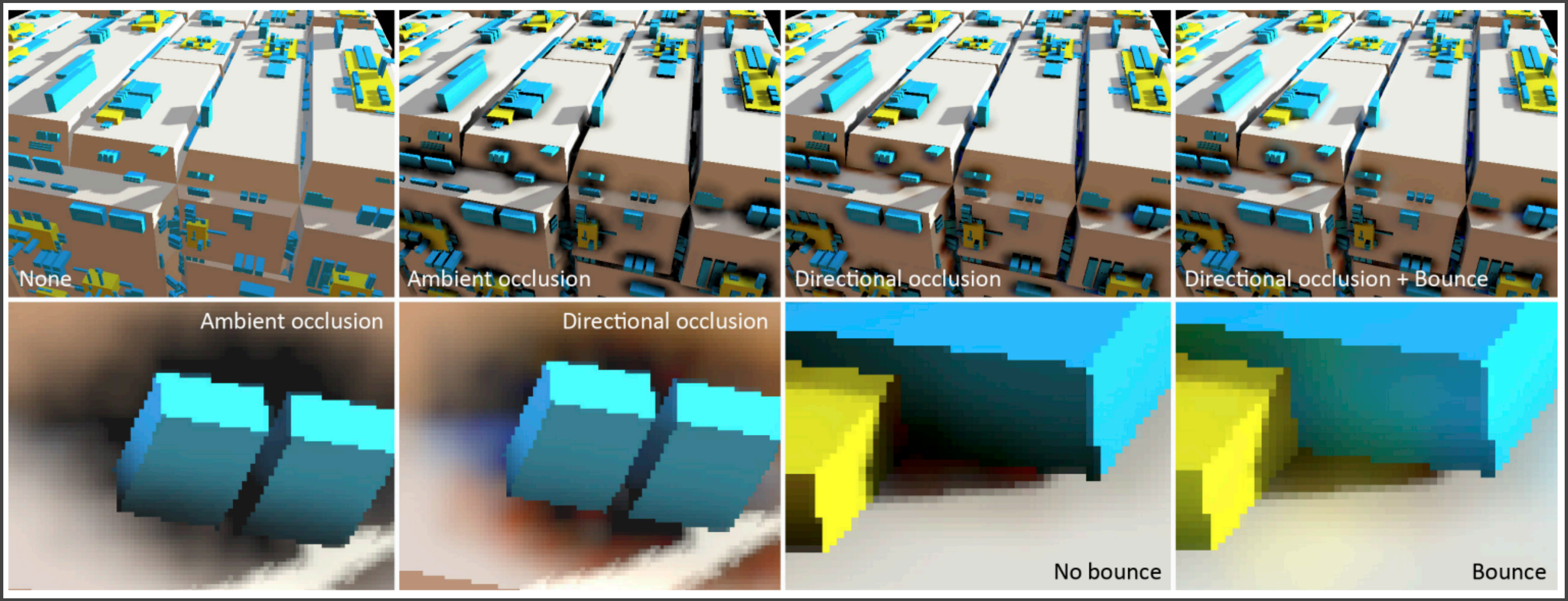
Screen Space Directional Occlusion

- What is SSDO?
  - An improvement over SSAO
  - Considering (more) actual indirect illumination

- Key idea
  - Why do we have to assume uniform incident indirect lighting?
  - Some information of indirect lighting is already known!
  - Sounds familiar to you?

Screen Space Directional Occlusion

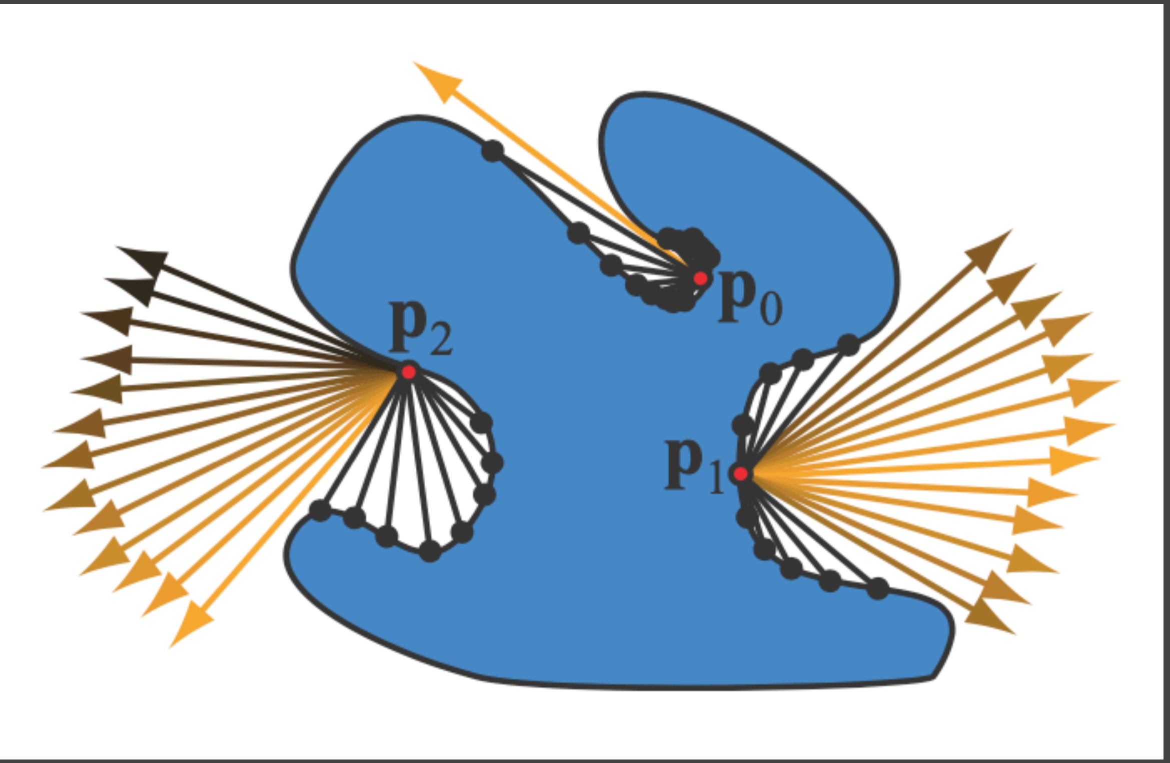
- SSDO exploits the rendered direct illumination
  - Not from an RSM, but from the camera



[Ritschel et al., Approximating Dynamic Global Illumination in Image Space]

Screen Space Directional Occlusion

- Very similar to path tracing
  - At shading point p, shoot a random ray
  - If it does not hit an obstacle, direct illumination
  - If it hits one, indirect illumination

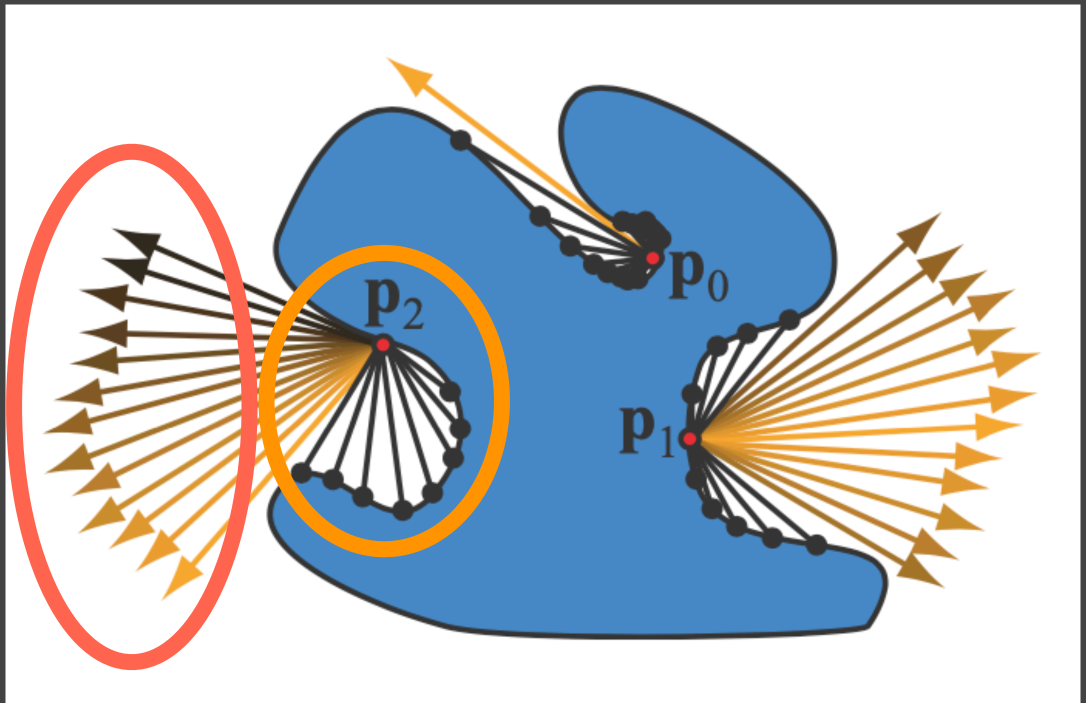


[From RTR4 book]

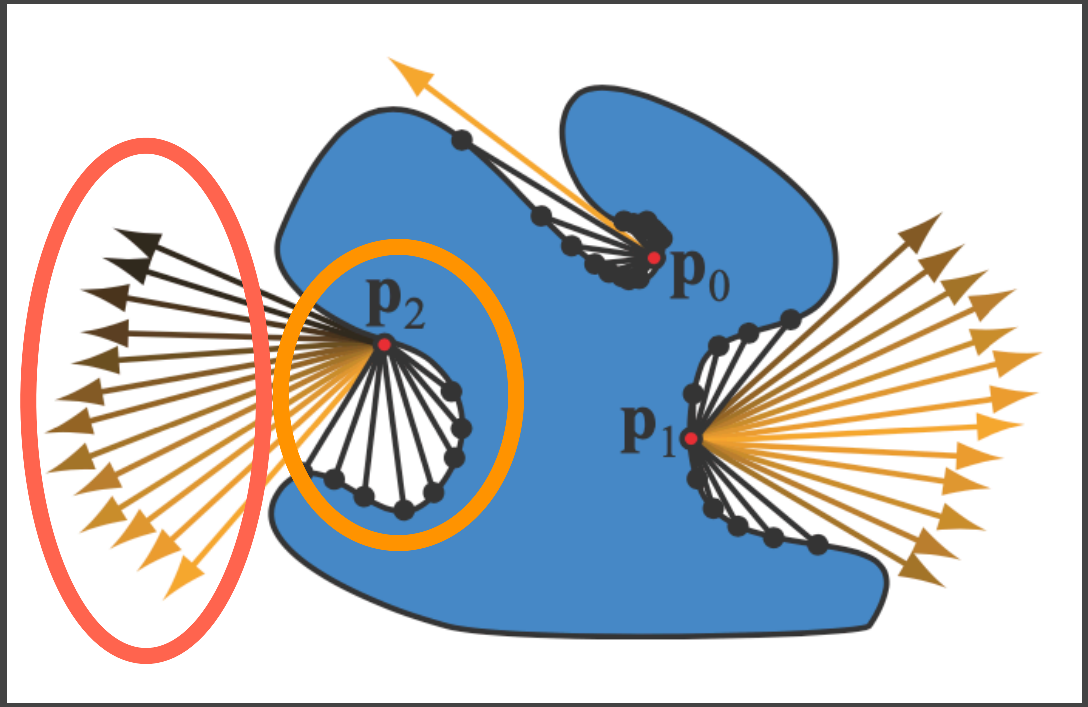
Screen Space Directional Occlusion

- Comparison w/ SSAO
  - AO: **indirect illumination** + **no indirect illumination**
  - DO: **no indirect illumination** + **indirect illumination** (same as path tracing)

[From RTR4 book]



Ambient Occlusion



Directional Occlusion

Screen Space Directional Occlusion

- Consider unoccluded and occluded directions separately

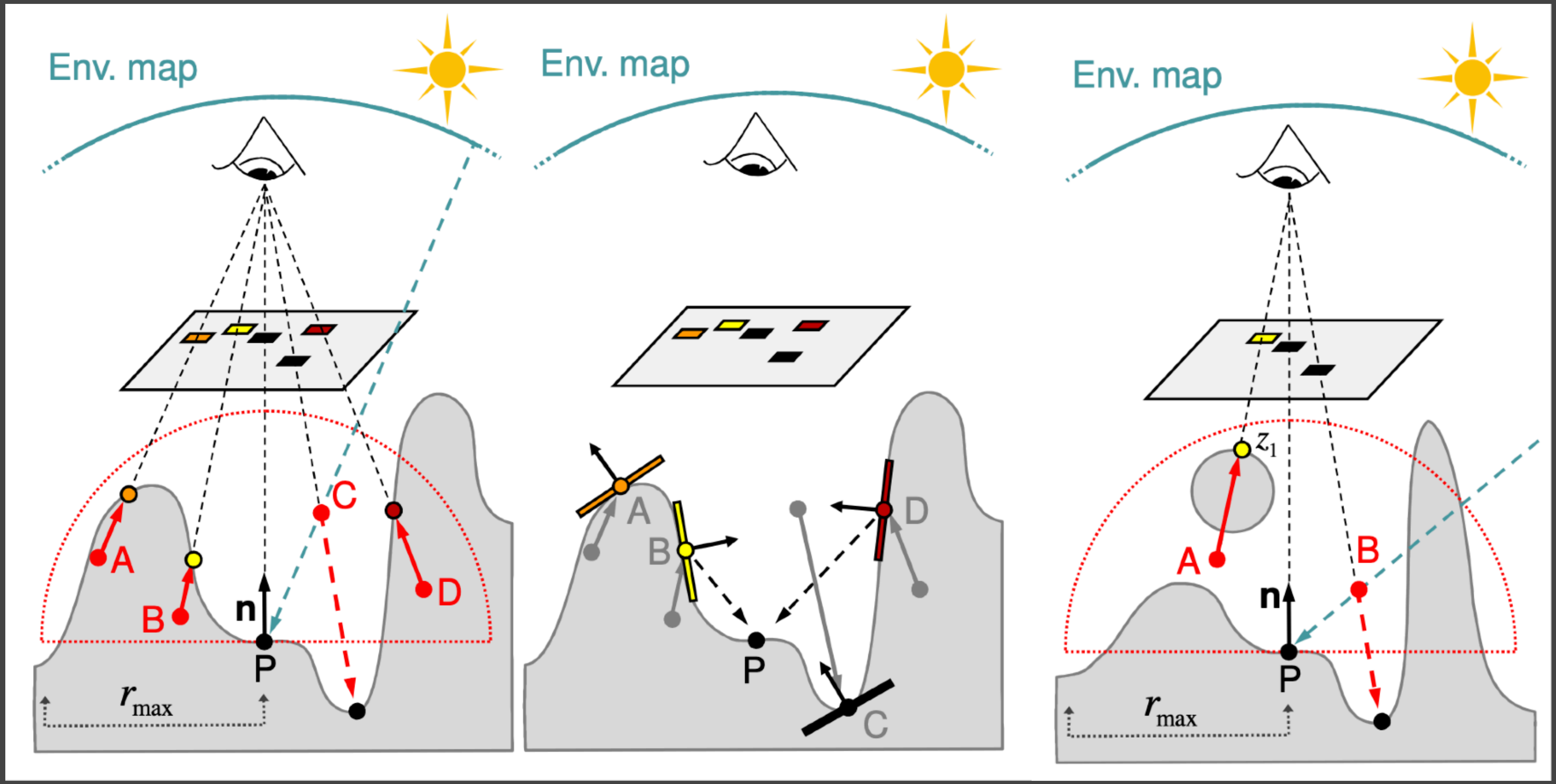
$$L_o^{\text{dir}}(\mathrm{p}, \omega_o) = \int_{\Omega^+, V=1} L_i^{\text{dir}}(\mathrm{p}, \omega_i) f_r(\mathrm{p}, \omega_i, \omega_o) \cos\theta_i \, \mathrm{d}\omega_i$$

$$L_o^{\text{indir}}(\mathrm{p}, \omega_o) = \int_{\Omega^+, V=0} L_i^{\text{indir}}(\mathrm{p}, \omega_i) f_r(\mathrm{p}, \omega_i, \omega_o) \cos\theta_i \, \mathrm{d}\omega_i$$

- Indirect illum from a pixel (patch) is derived in last lecture

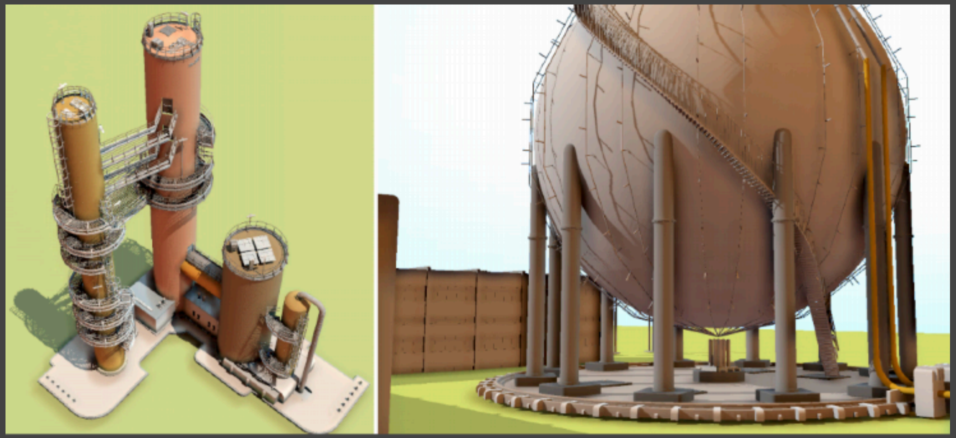
Screen Space Directional Occlusion

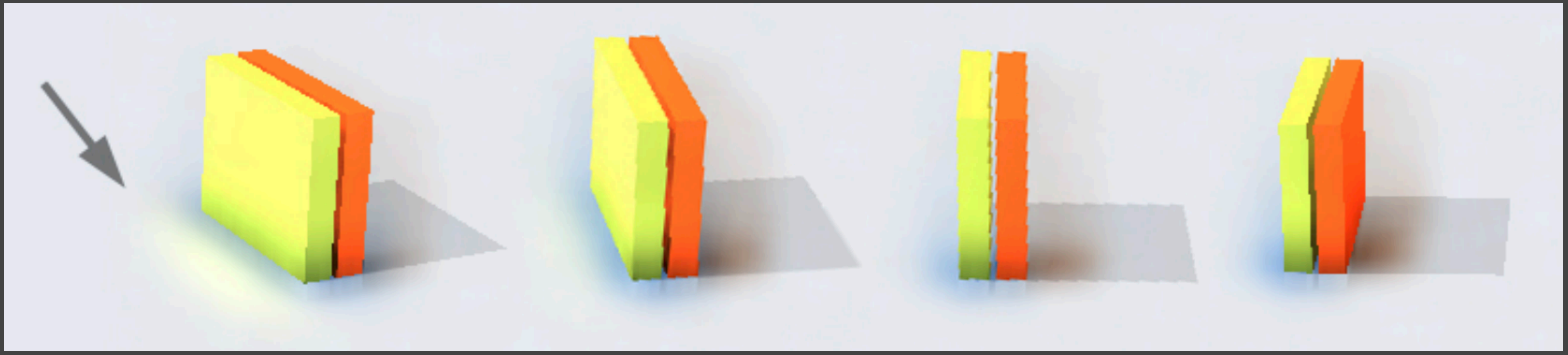
- Similar to HBAO, test samples' depths in local hemispheres



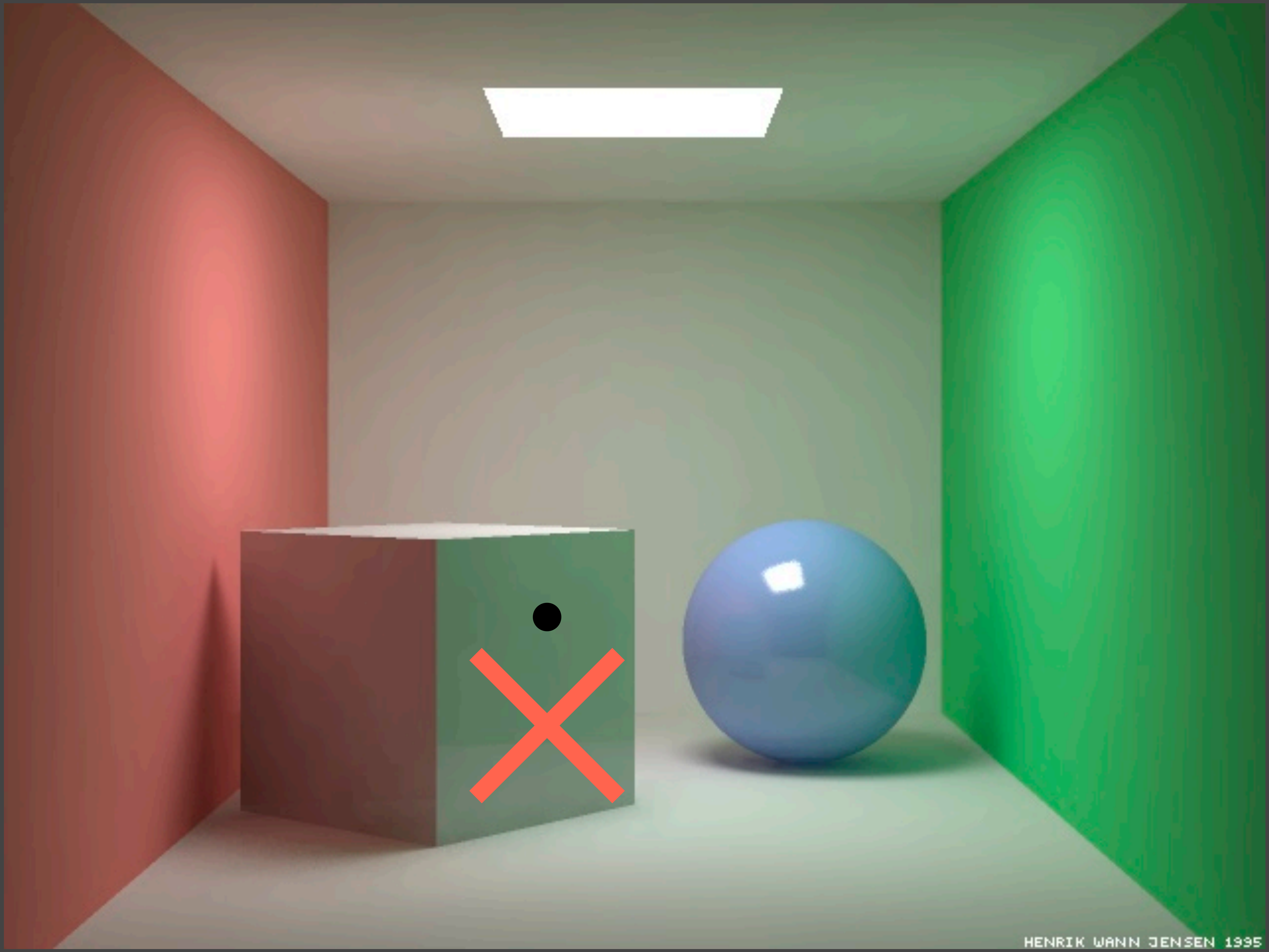
Screen Space Directional Occlusion

- SSDO: quality closer to offline rendering
- Issues?
  - Still, GI in a short range
  - Visibility
  - Screen space issue: missing information from **unseen** surfaces





SSDO: GI in a Short Range



Questions?

Screen Space Reflection (SSR)

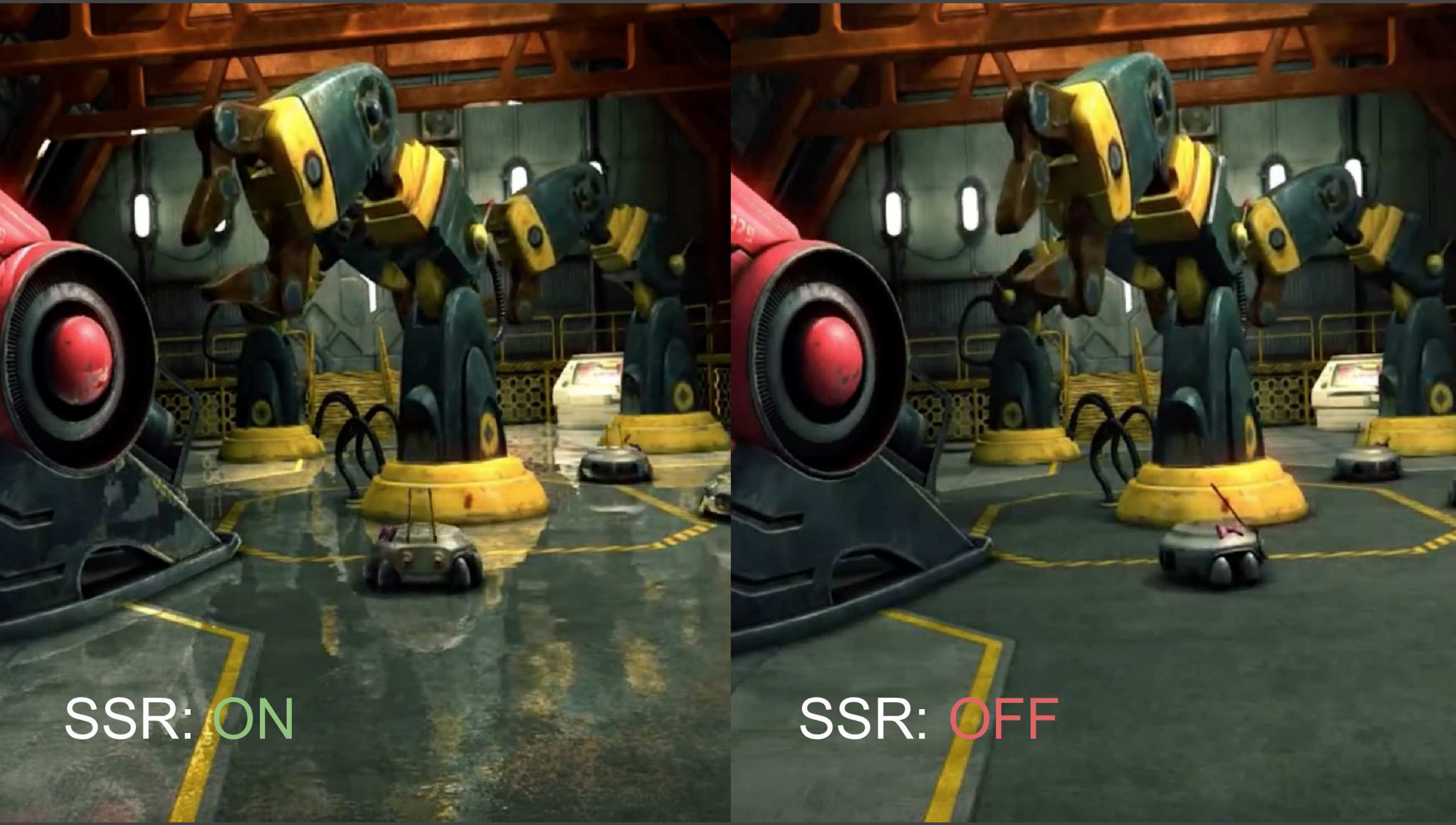
(Some slides from SIGGRAPH 2015 course:
Advances in Real-time Rendering)

Screen Space Reflection (SSR)

- What is SSR?
  - Still, one way to introduce Global Illumination in RTR
  - Performing ray tracing
  - But does not require 3D primitives (triangles, etc.)
- Two fundamental tasks of SSR
  - Intersection: between any ray and the scene
  - Shading: contribution from intersected pixels to the shading point

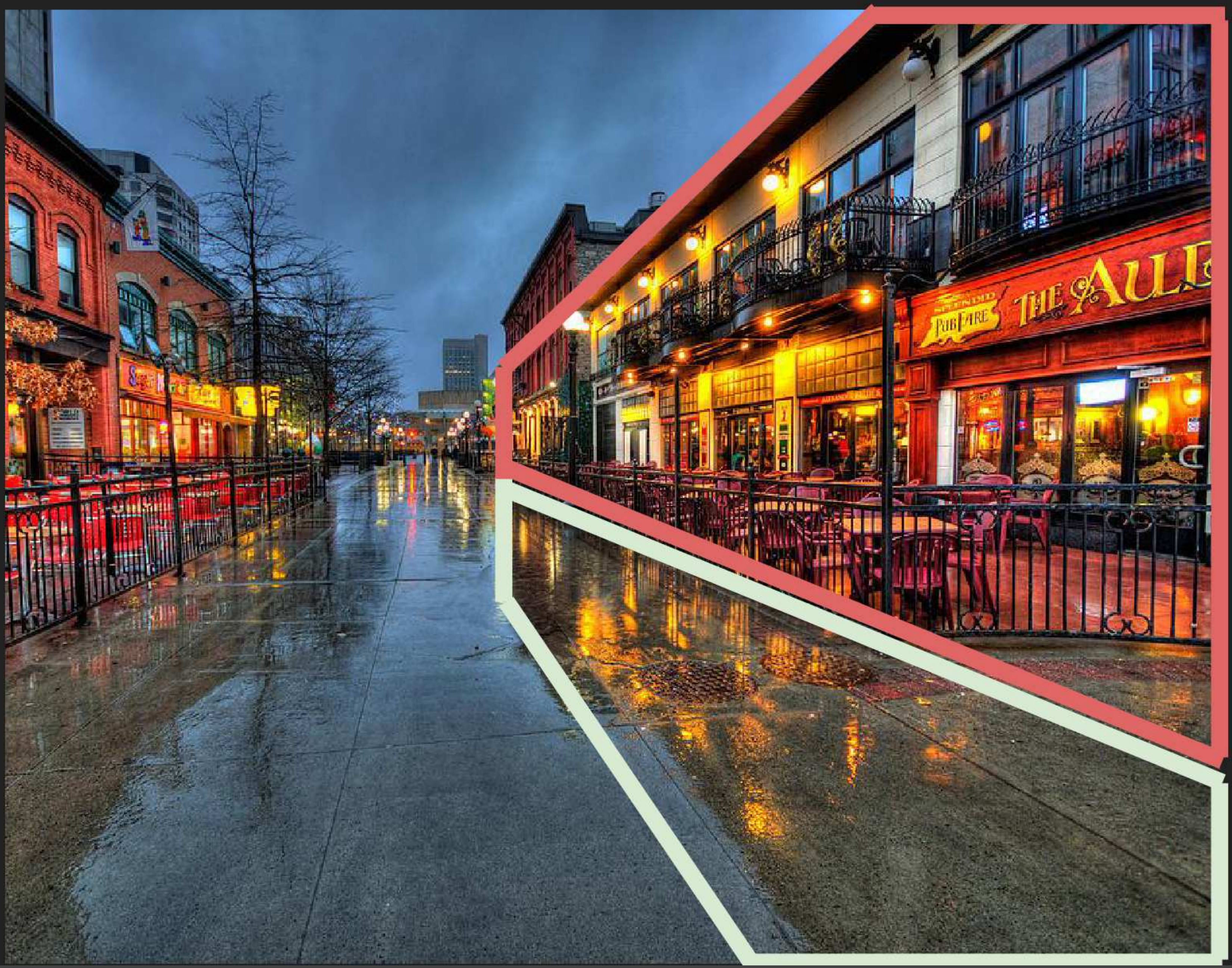
Screen-space Reflections

Motivation


SSR: ON
SSR: OFF

What can be exploited in scene?

Reuse screen-space data!

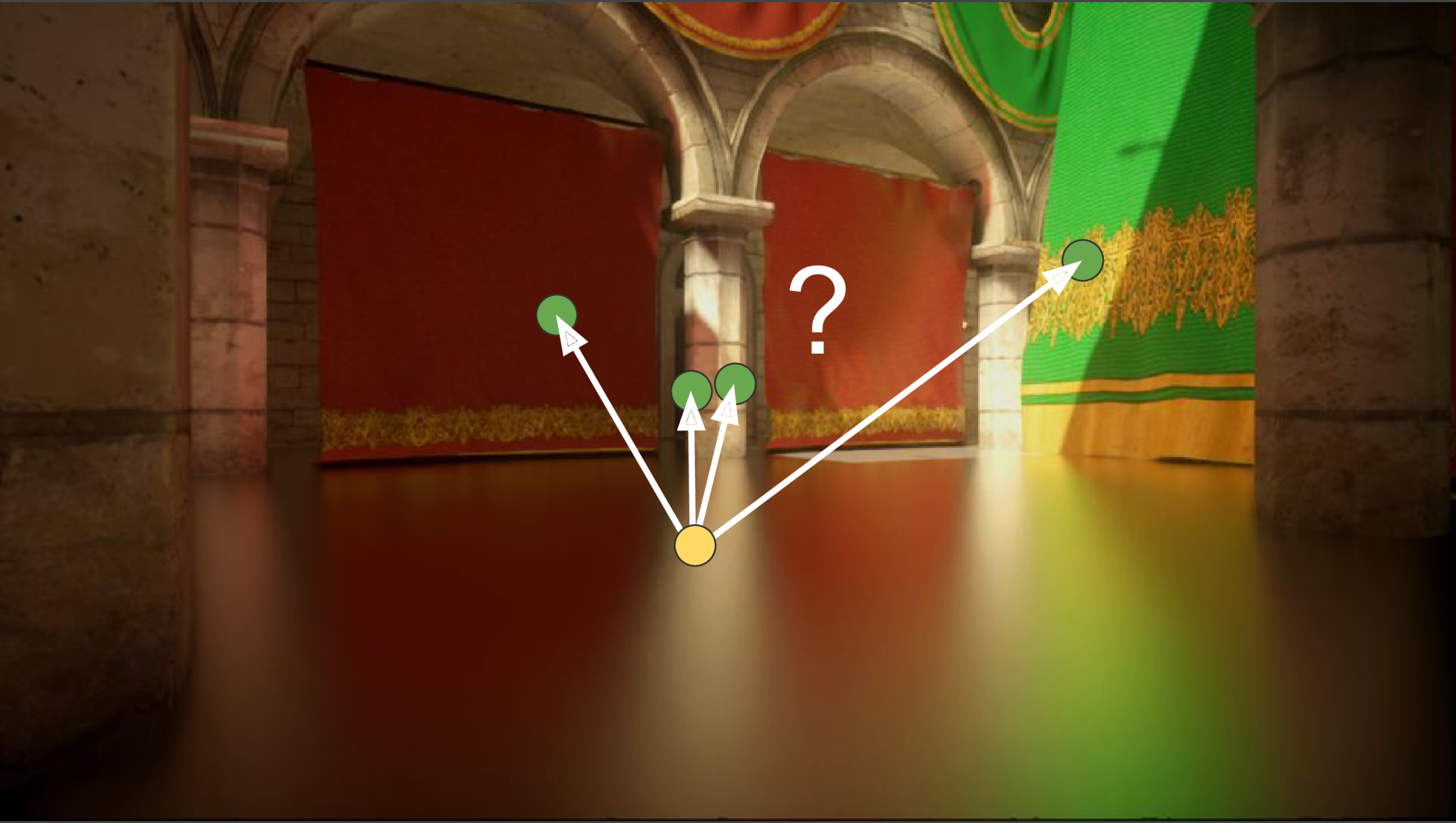


Basic SSR Algorithm - Mirror Reflection

- For each fragment
  - Compute reflection ray
  - Trace along ray direction (using depth buffer)
  - Use color of intersection point as reflection color

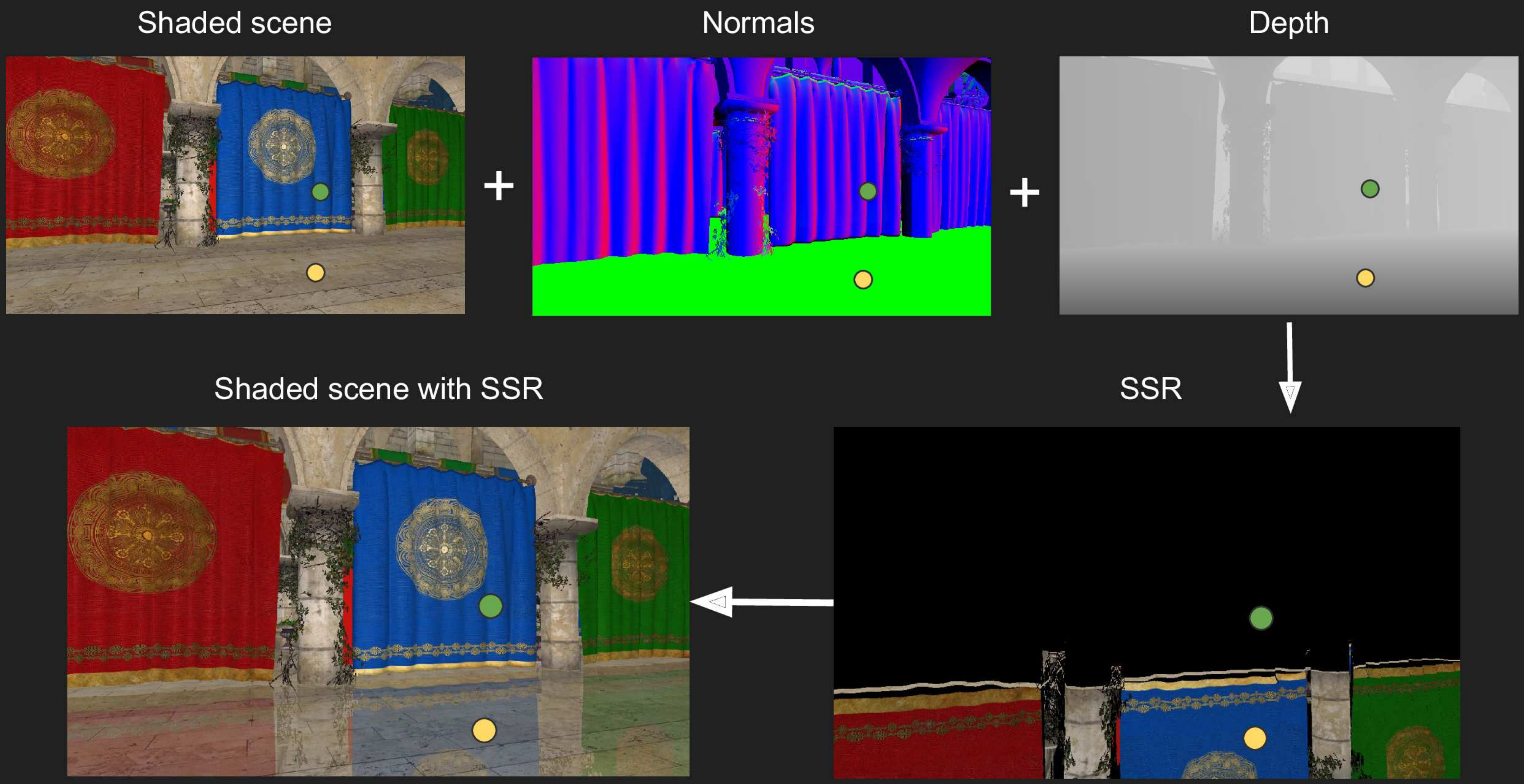
High smoothness


Medium smoothness




Medium smoothness + normals

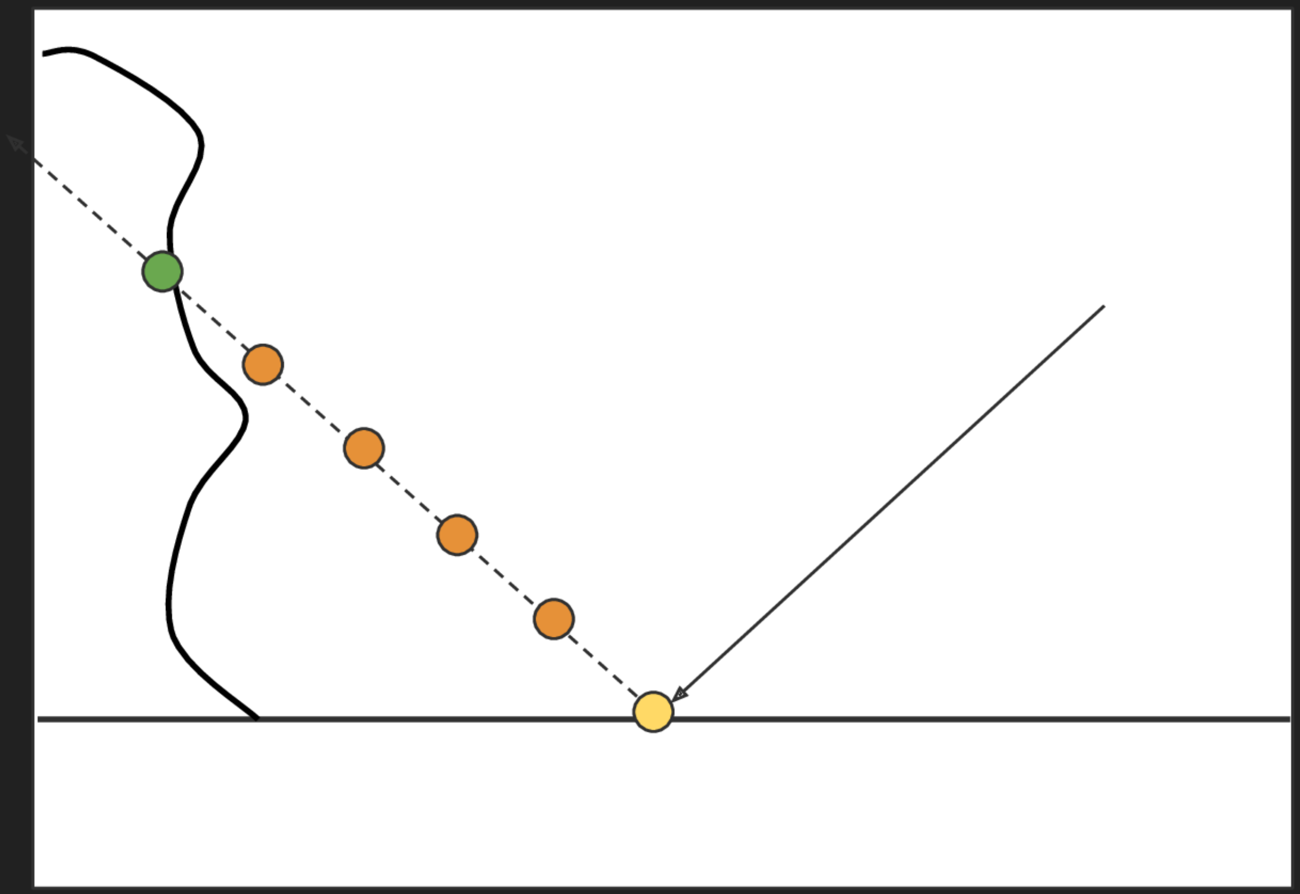
Variable smoothness


Shaded scene
+
Normals
+
Depth
Shaded scene with SSR
SSR

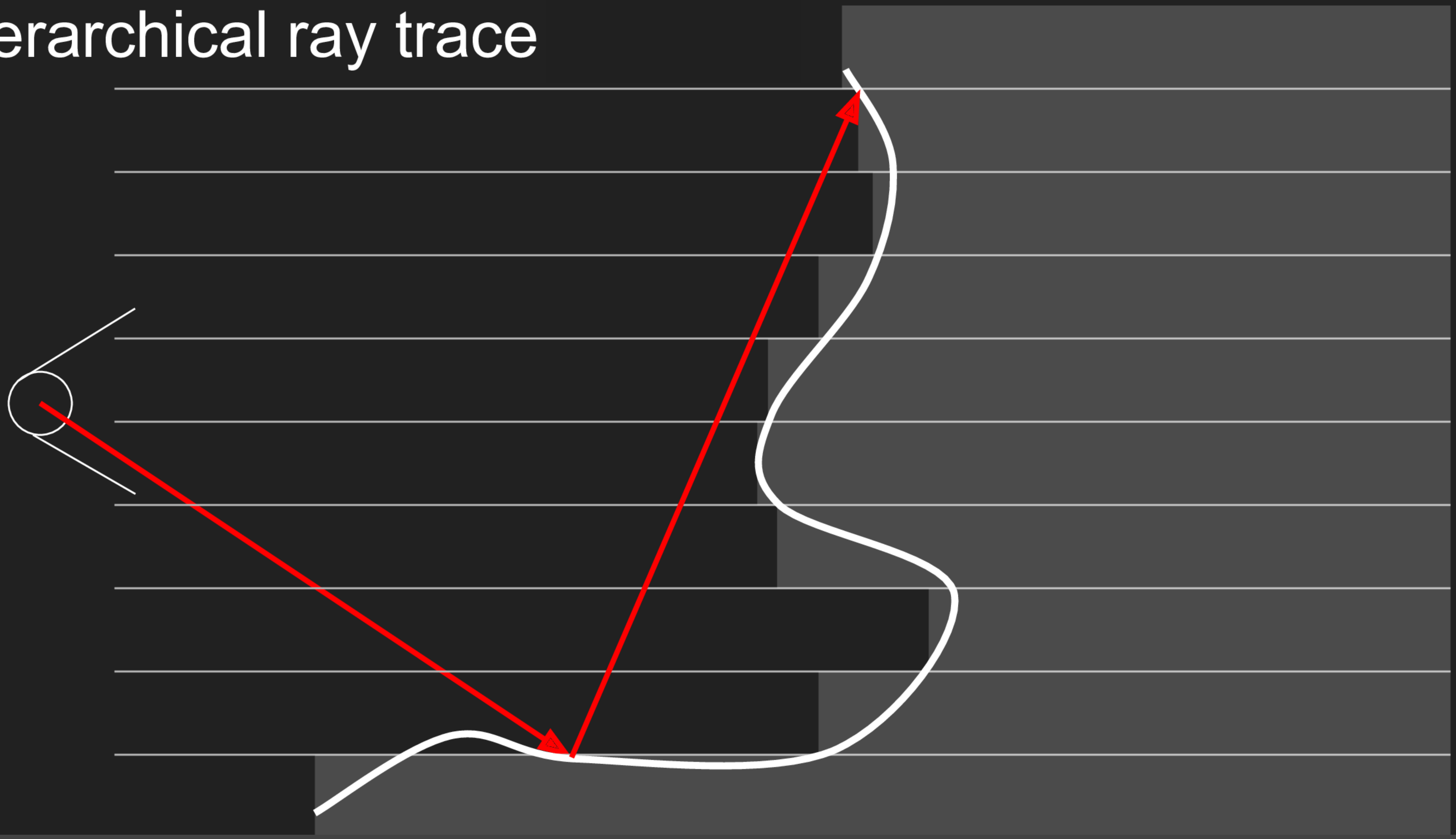
Linear Raymarch

Goal: Find intersection point

- At each step, check depth value
- Quality depends on step size
- Can be refined

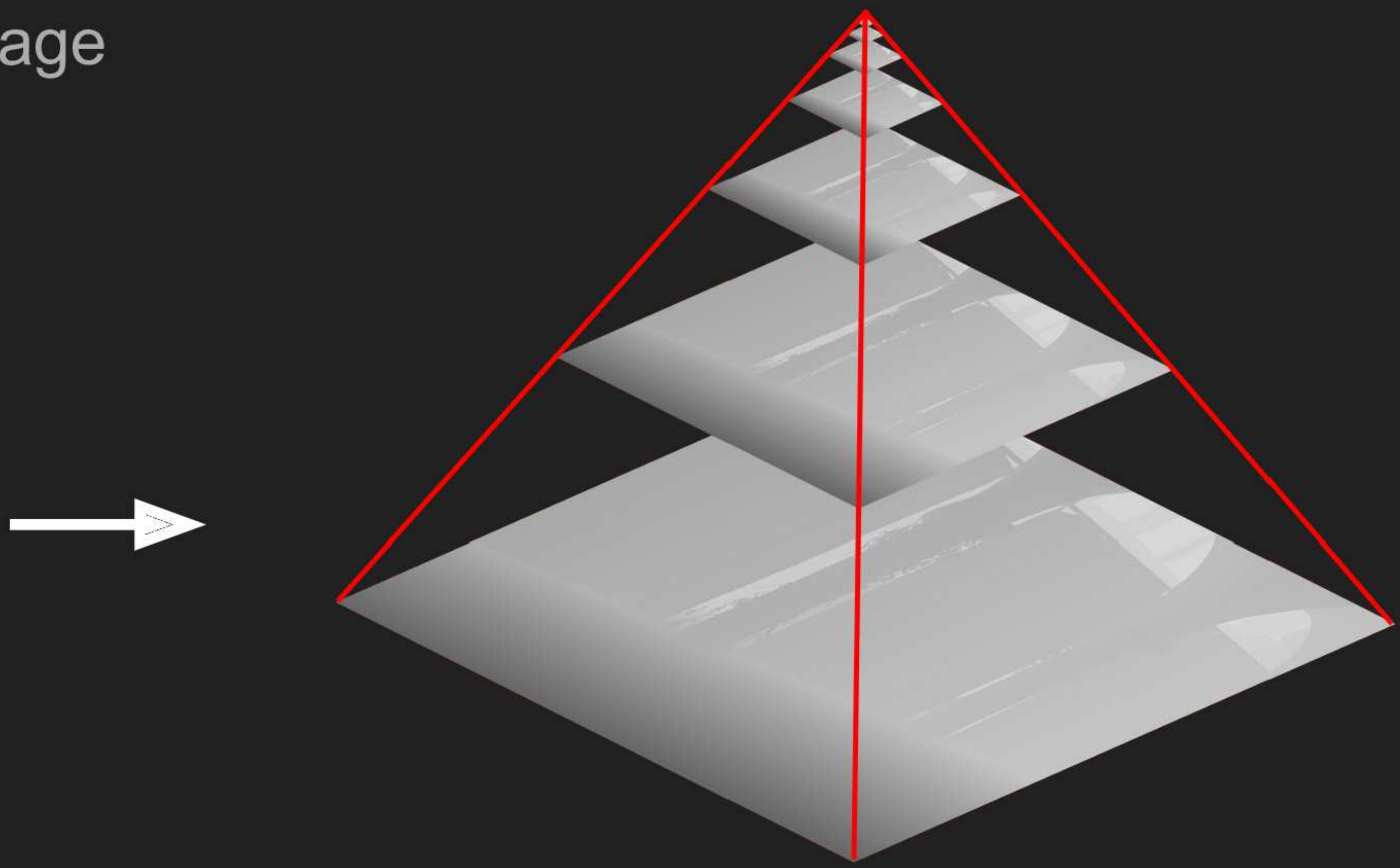


Hierarchical ray trace



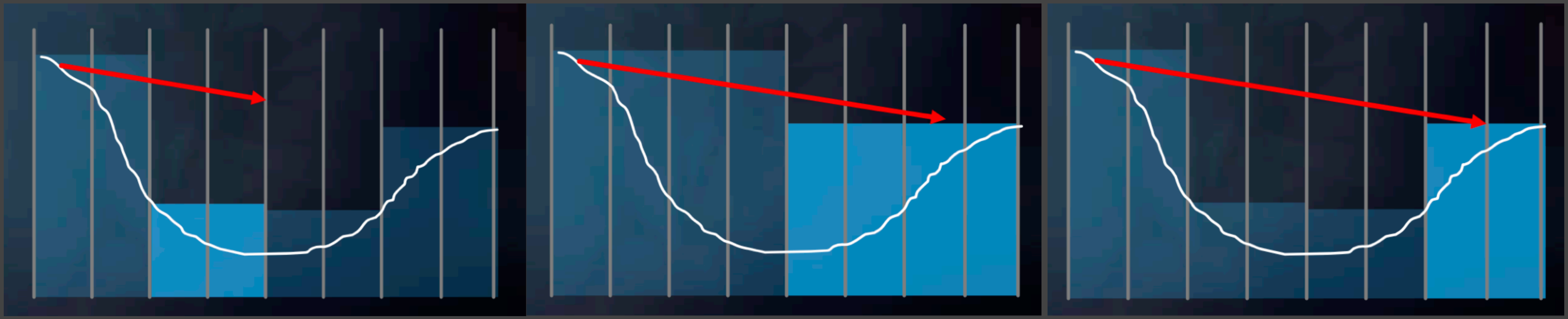
Generate Depth Mip-Map

- Use min values instead of average



Why Depth Mipmap

- Very similar to the hierarchy (BVH, KD-tree) in 3D
- Enabling faster rejecting of non-intersecting in a bunch
- The min operation guarantees a conservative logic
  - If a ray does not even intersect a larger node,
    it will never intersect any child nodes of it



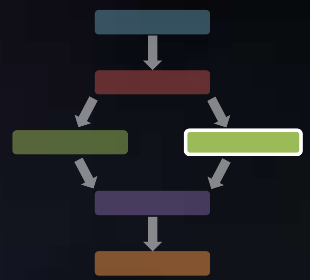
Hierarchical tracing

- Stackless ray walk of min-Z pyramid

```
mip = 0;
while (level > -1)
    step through current cell;
    if (above Z plane) ++level;
    if (below Z plane) --level;
```

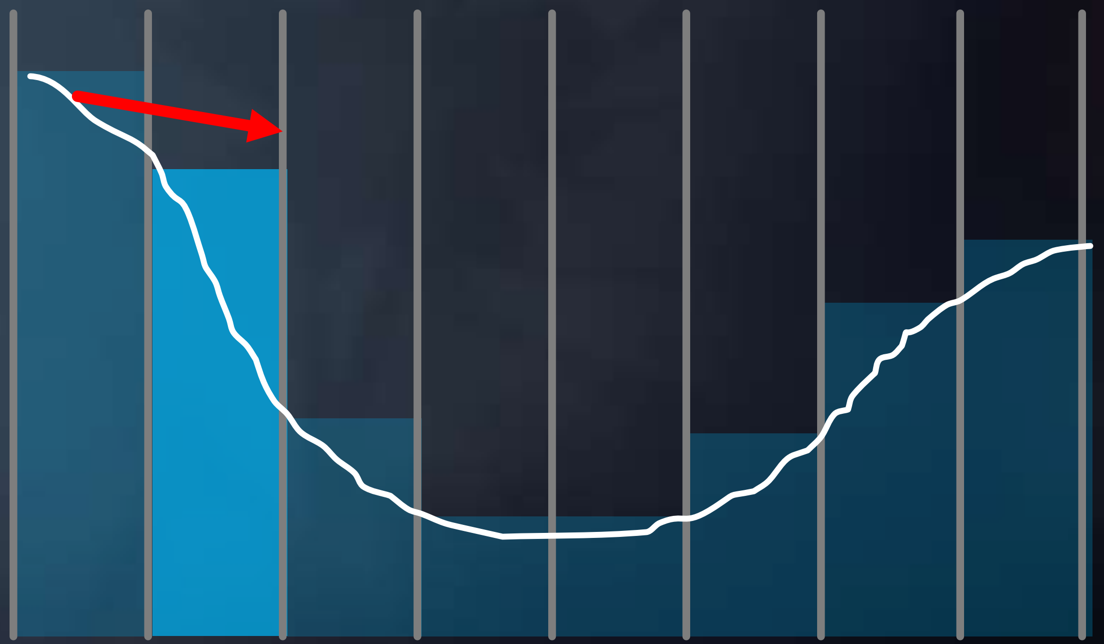
SIGGRAPH 2015: Advances in Real-Time Rendering course

Hierarchical tracing



- Stackless ray walk of min-Z pyramid

```
mip = 0;
while (level > -1)
    step through current cell;
    if (above Z plane) ++level;
    if (below Z plane) --level;
```



SIGGRAPH 2015: Advances in Real-Time Rendering course

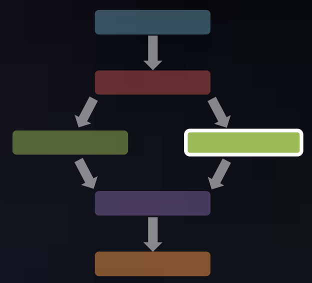
Hierarchical tracing

- Stackless ray walk of min-Z pyramid

```
mip = 0;
while (level > -1)
    step through current cell;
    if (above Z plane) ++level;
    if (below Z plane) --level;
```

SIGGRAPH 2015: Advances in Real-Time Rendering course

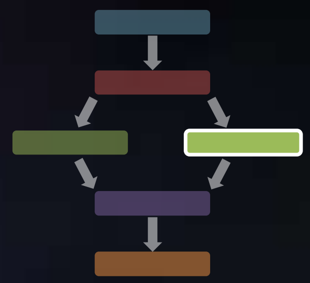
Hierarchical tracing



- Stackless ray walk of min-Z pyramid

```
mip = 0;
while (level > -1)
    step through current cell;
    if (above Z plane) ++level;
    if (below Z plane) --level;
```

SIGGRAPH 2015: Advances in Real-Time Rendering course



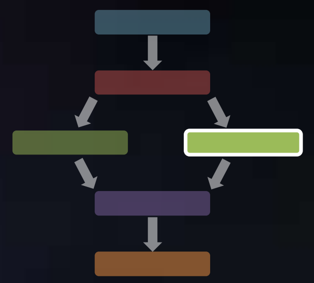
Hierarchical tracing

- Stackless ray walk of min-Z pyramid

```
mip = 0;
while (level > -1)
    step through current cell;
    if (above Z plane) ++level;
    if (below Z plane) --level;
```

SIGGRAPH 2015: Advances in Real-Time Rendering course

Hierarchical tracing



- Stackless ray walk of min-Z pyramid

```
mip = 0;
while (level > -1)
    step through current cell;
    if (above Z plane) ++level;
    if (below Z plane) --level;
```

SIGGRAPH 2015: Advances in Real-Time Rendering course

Hierarchical tracing

- Stackless ray walk of min-Z pyramid

```
mip = 0;
while (level > -1)
    step through current cell;
    if (above Z plane) ++level;
    if (below Z plane) --level;
```

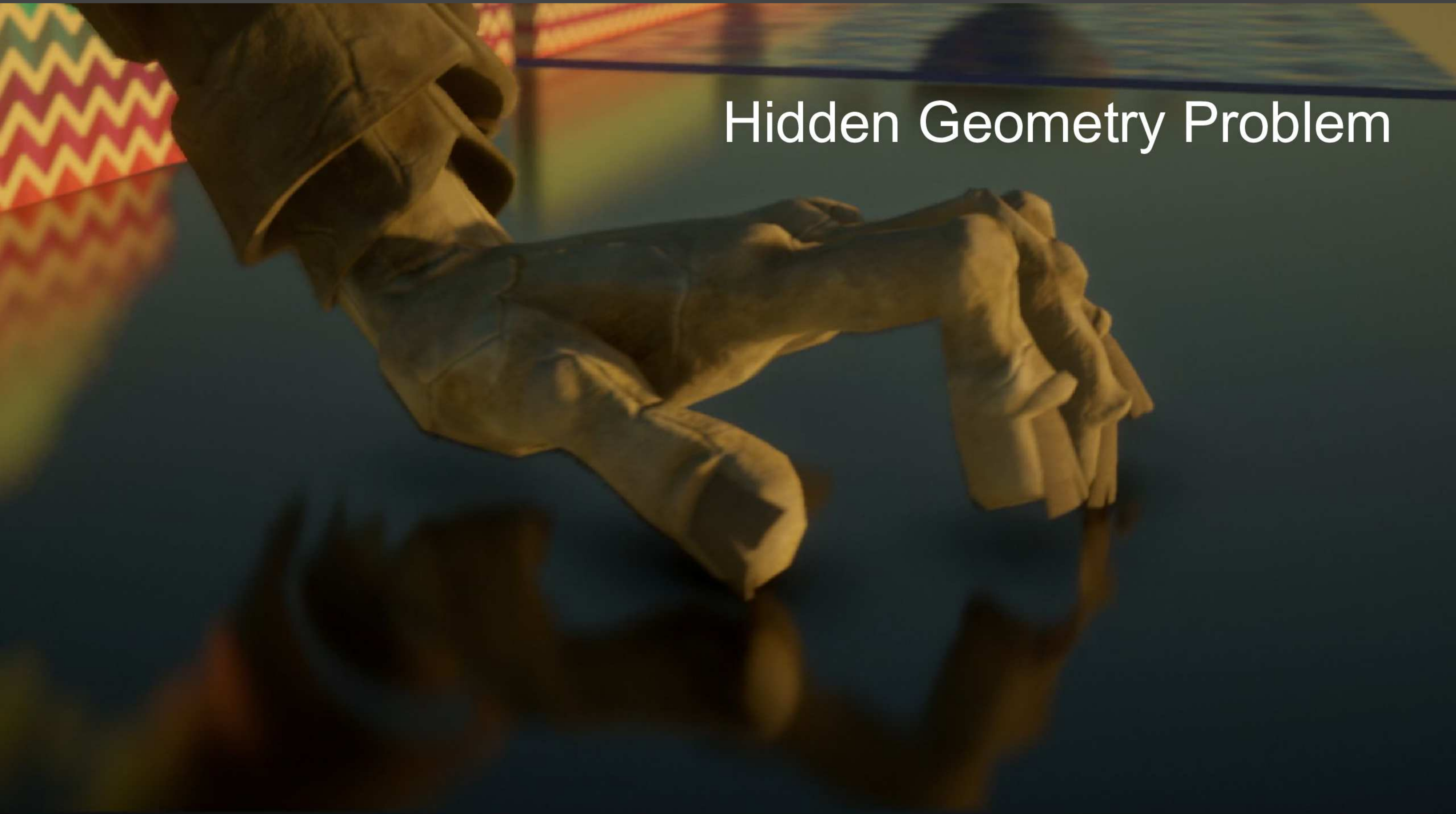
SIGGRAPH 2015: Advances in Real-Time Rendering course

Hierarchical tracing

- Stackless ray walk of min-Z pyramid

```
mip = 0;
while (level > -1)
    step through current cell;
    if (above Z plane) ++level;
    if (below Z plane) --level;
```

SIGGRAPH 2015: Advances in Real-Time Rendering course


Hidden Geometry Problem


Edge Cutoff

Edge Fading

Shading using SSR

- Absolutely no difference from path tracing
  - Just again assuming diffuse reflectors / secondary lights

$$L_o(\mathrm{p}, \omega_o) = \int_{\Omega^+} \underline{\underline{L_i(\mathrm{p}, \omega_i)}} f_r(\mathrm{p}, \omega_i, \omega_o) \cos\theta_i \, \mathrm{d}\omega_i$$

$$L_o(\mathrm{q}, q \to p)$$

- Questions
  - Does it introduce the square distance falloff?
  - Does it handle occlusions between the shading point and secondary lights?

Therefore,

Our requirements

- Sharp and blurry reflections
- Contact hardening
- Specular elongation
- Per-pixel roughness and normal

Therefore,

Our requirements

- Sharp and blurry reflections
- Contact hardening
- Specular elongation
- Per-pixel roughness and normal

Therefore,

Our requirements

- Sharp and blurry reflections
- Contact hardening
- Specular elongation
- Per-pixel roughness and normal

Therefore,

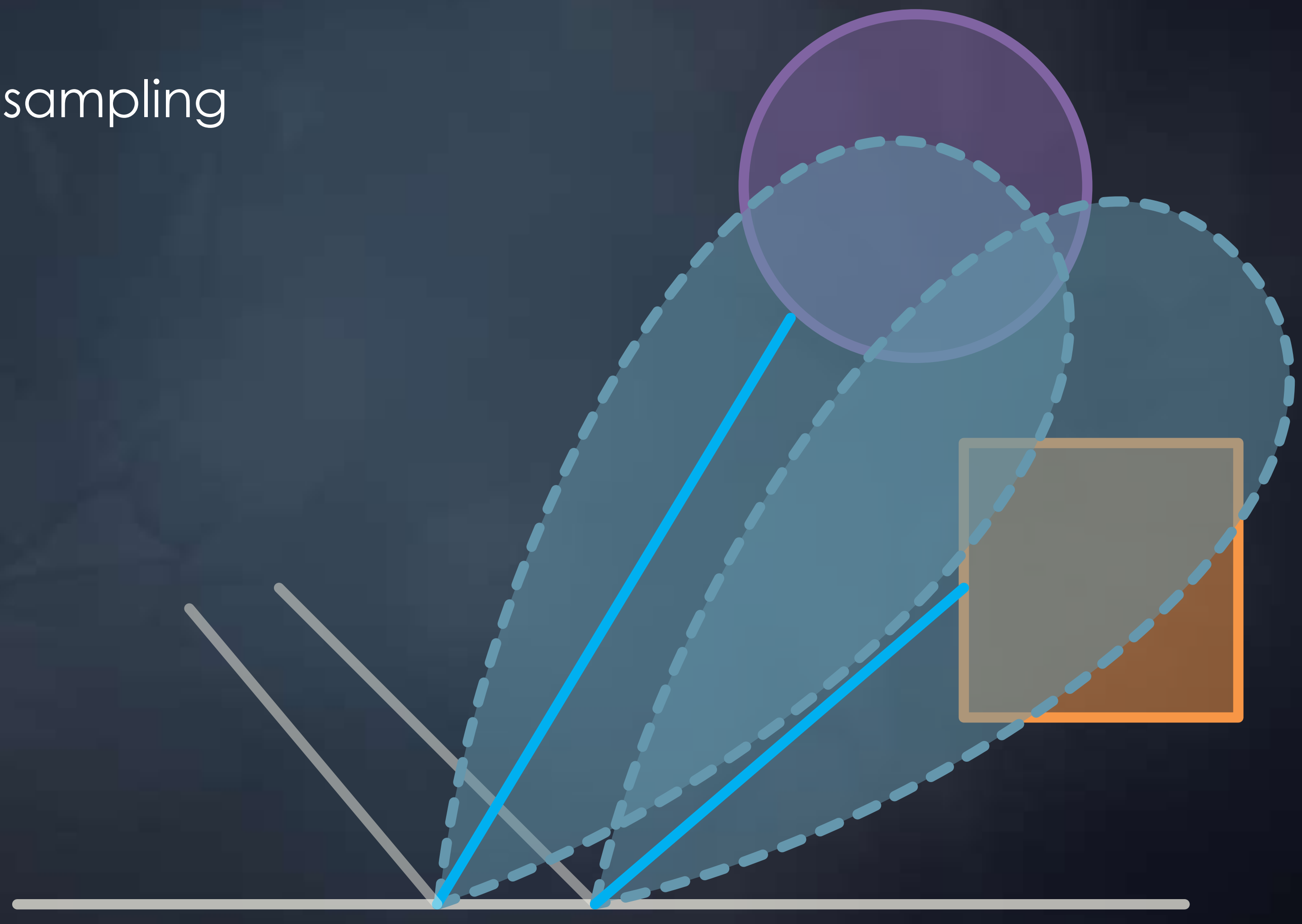
Our requirements

- Sharp and blurry reflections
- Contact hardening
- Specular elongation
- Per-pixel roughness and normal

Improvements

Our approach

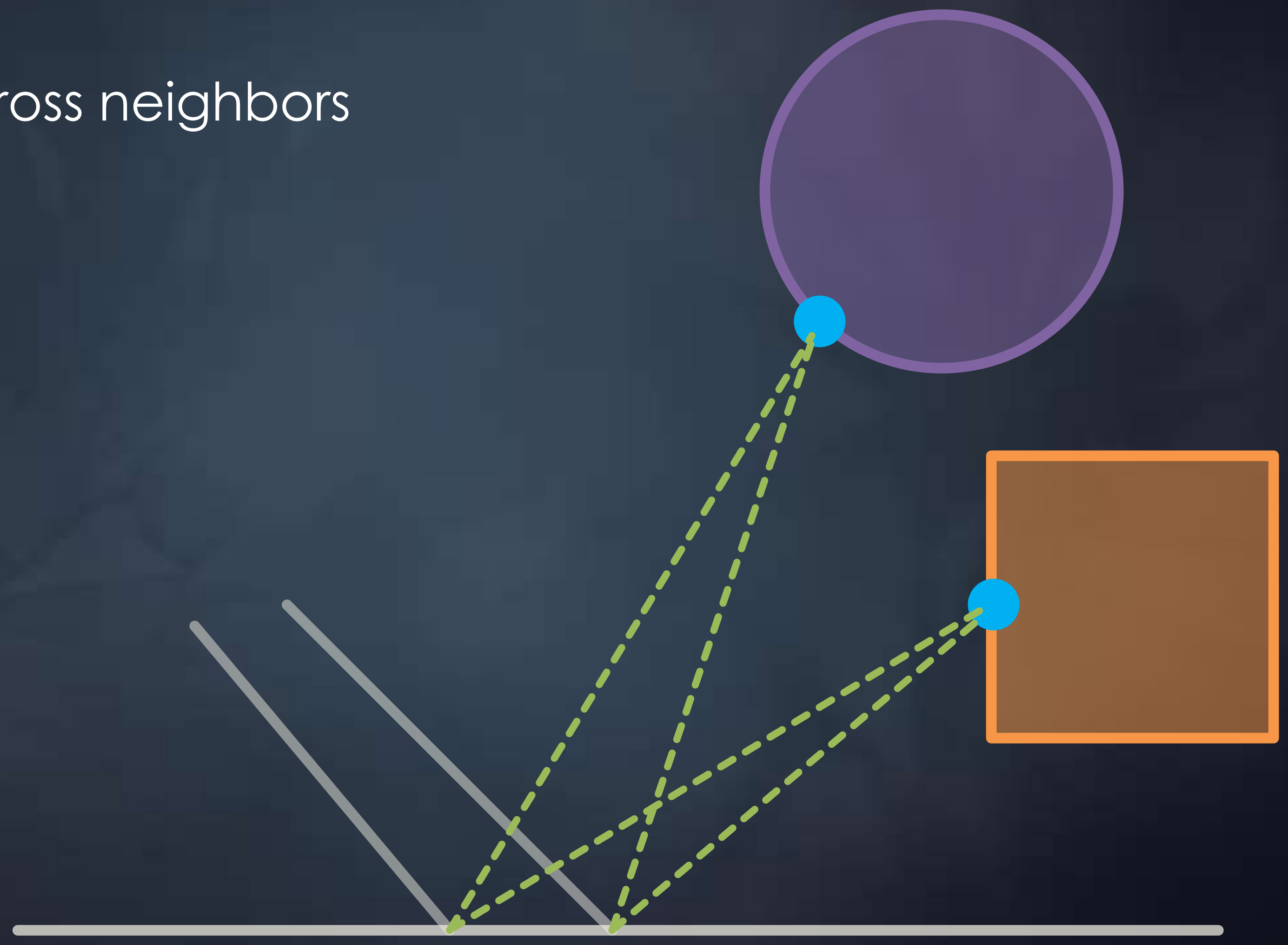
- BRDF importance sampling



Improvements

Our approach

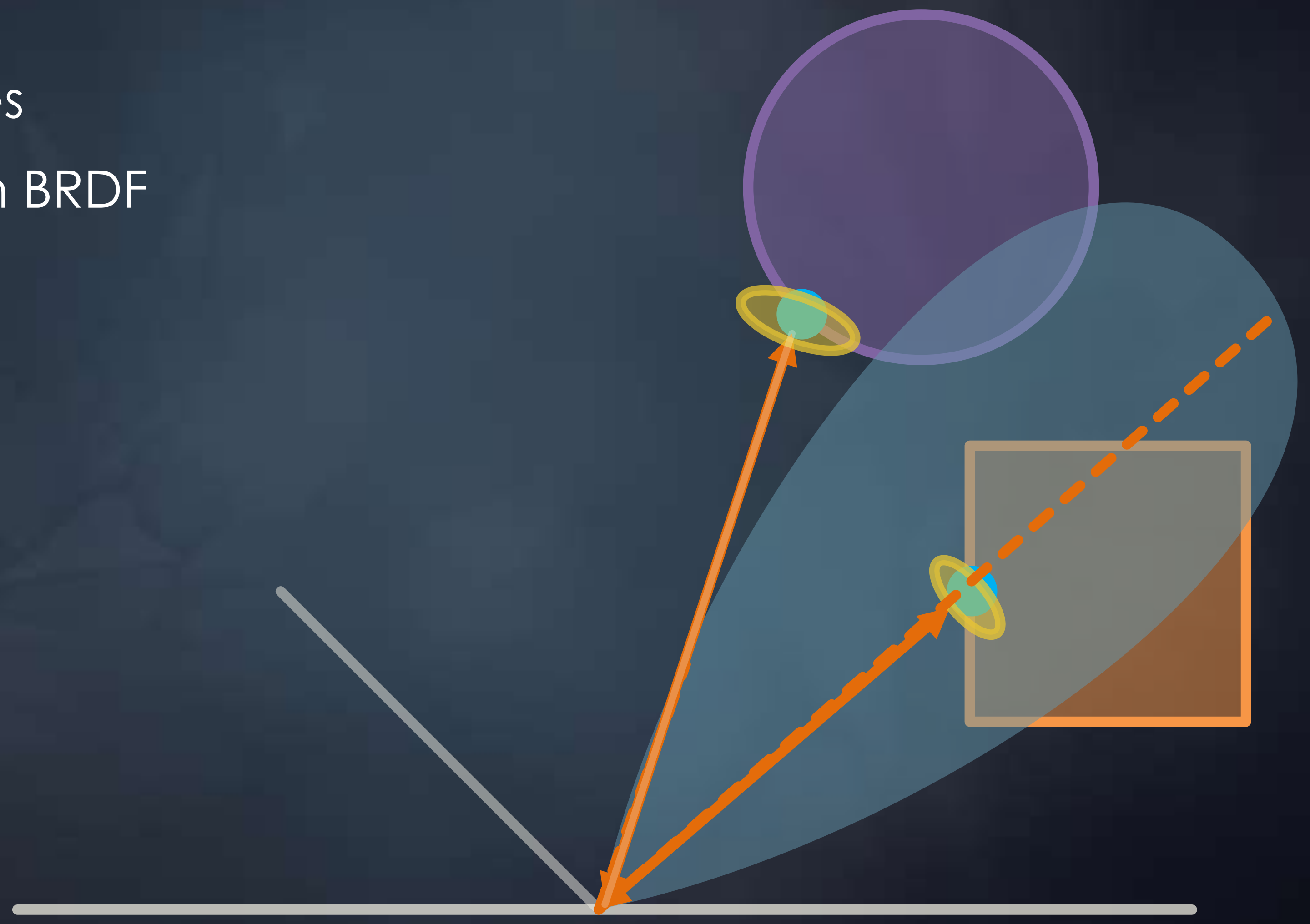
- Hit point reuse across neighbors



Improvements

Our approach

- Prefiltered samples
- Weighed by each BRDF



Summary of SSR

- Pros
  - Fast performance for glossy and specular reflections
  - Good quality
  - No spikes and occlusion issues
- Cons
  - Not as efficient in the diffuse case*
  - Missing information outside the screen

Questions?

Next Lecture

- Real-Time Physically-Based Materials

[Big Hero 6, Disney 2014]

Thank you!