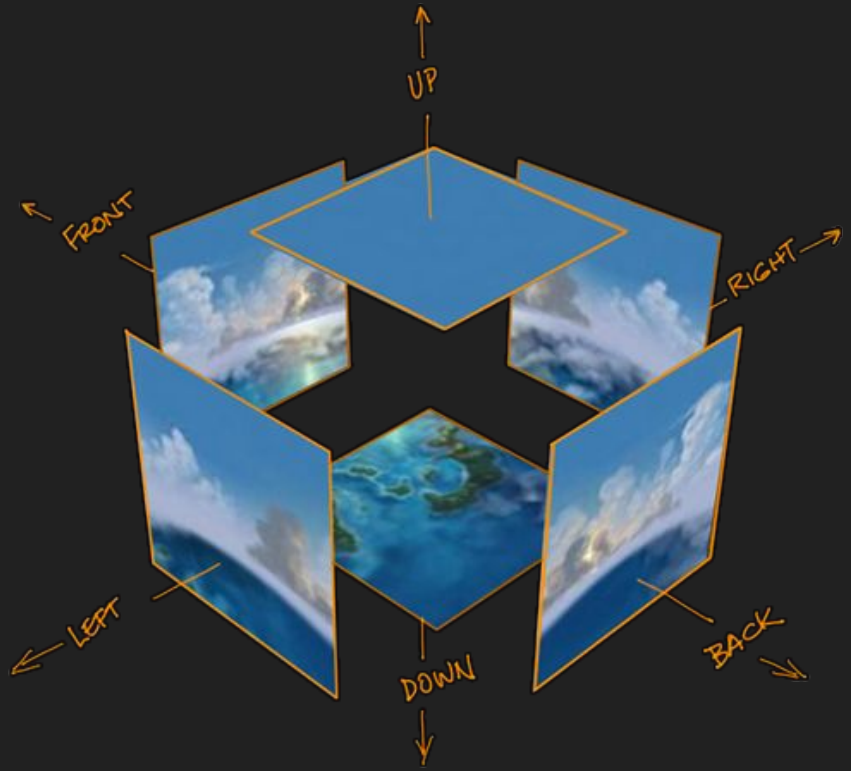
Screen-space Reflections


SSR: ON
SSR: OFF

What about cubemaps?



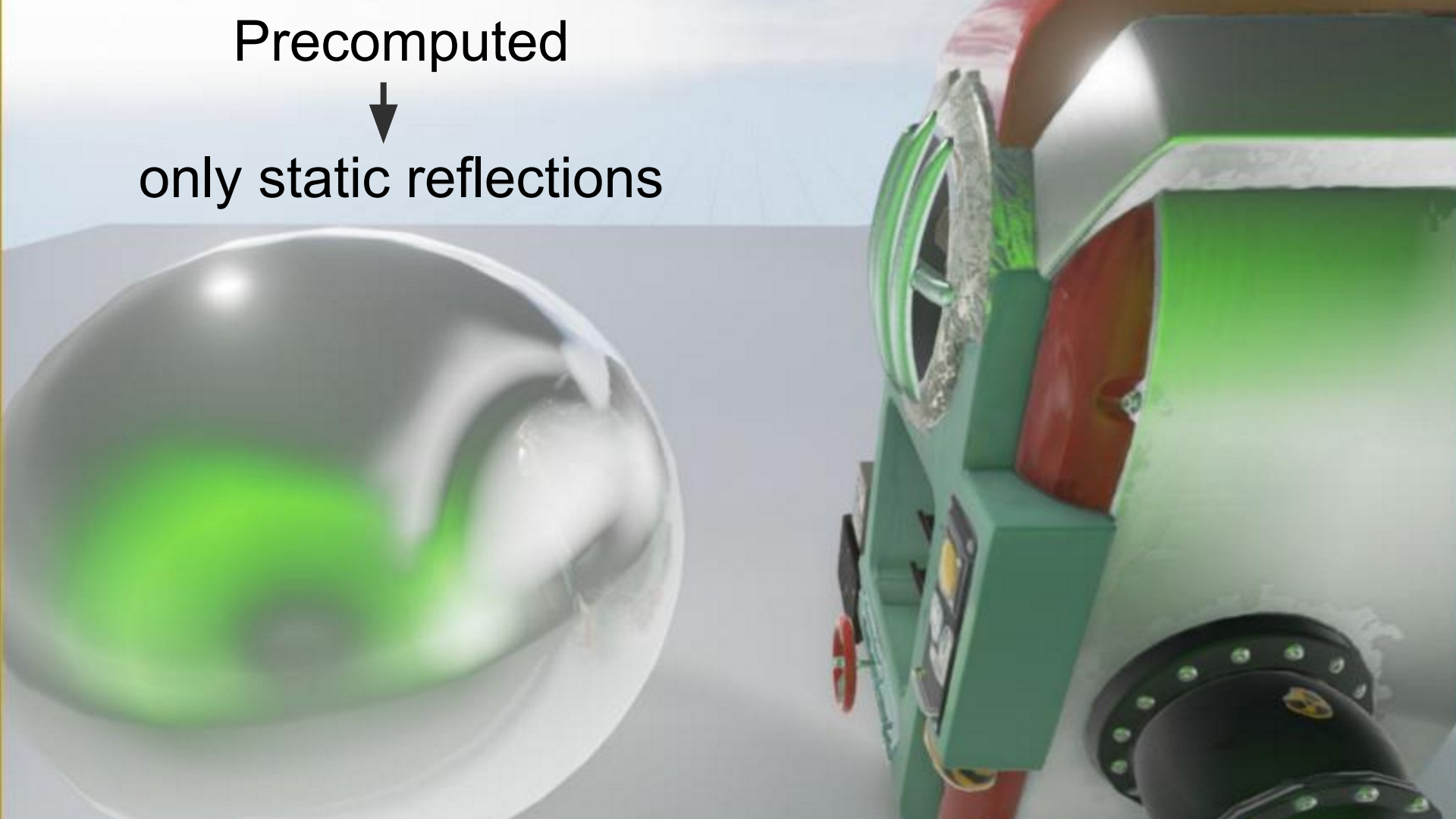

Sometimes work well


But no self reflection

Precomputed

↓

only static reflections

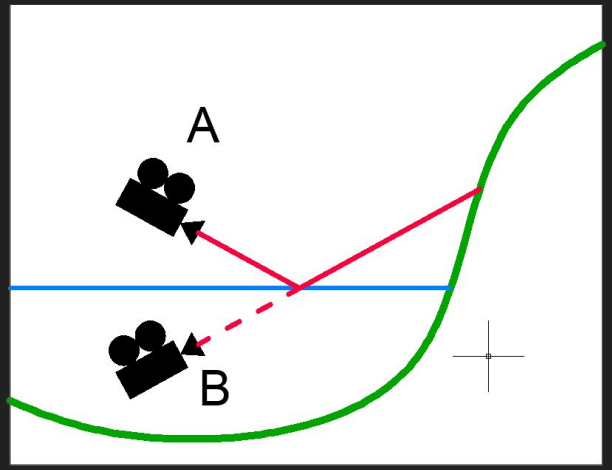


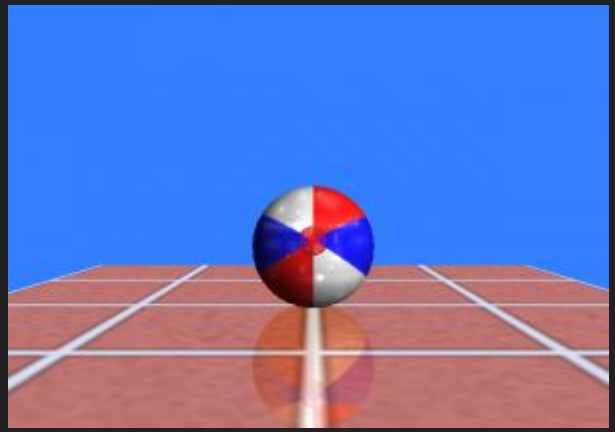
Planar Reflections


With cube map, reflections out of sync

Rerender Solution

- Render again from reflected viewpoint
- Complexity scales with scene
- Normally for perfectly planar surfaces





What can be exploited in scene?

Reuse screen-space data!

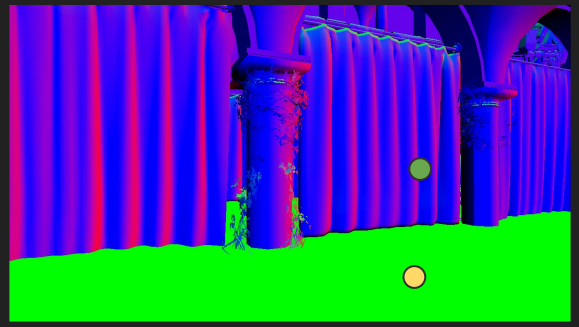
Basic SSR Algorithm - Mirror Reflection

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

Shaded scene

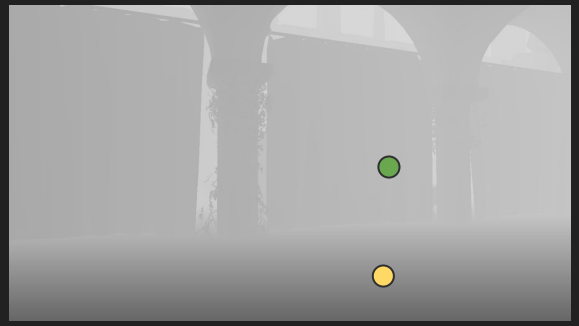
+

Normals

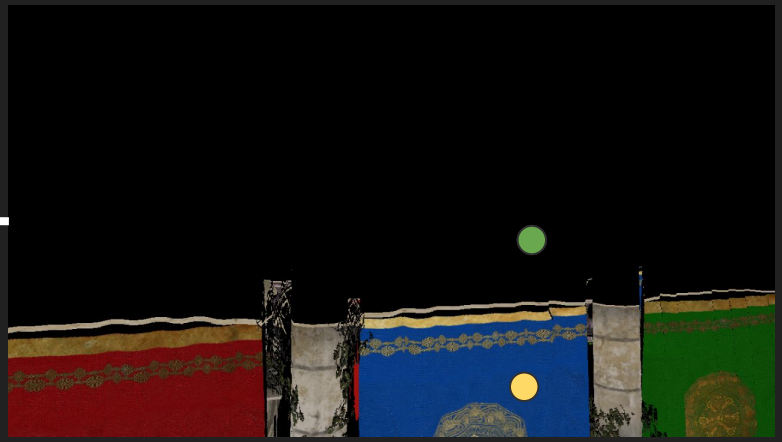


+

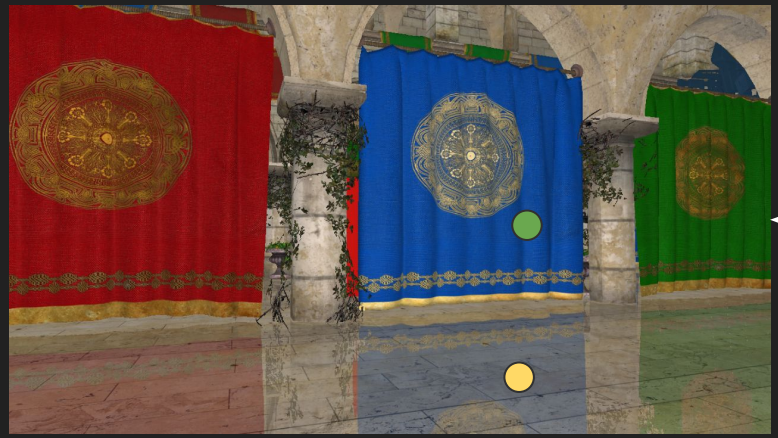
Depth



SSR



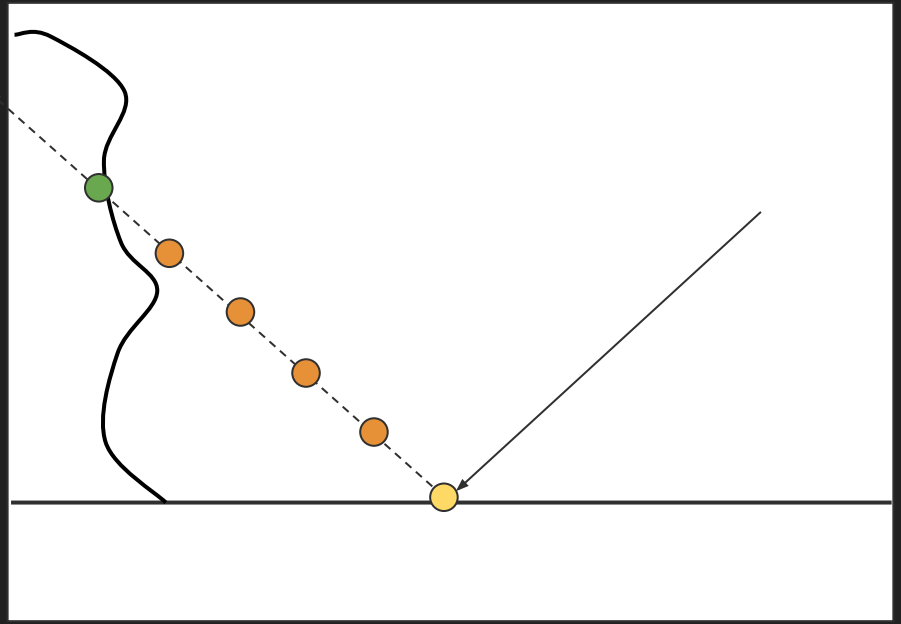
Shaded scene with SSR



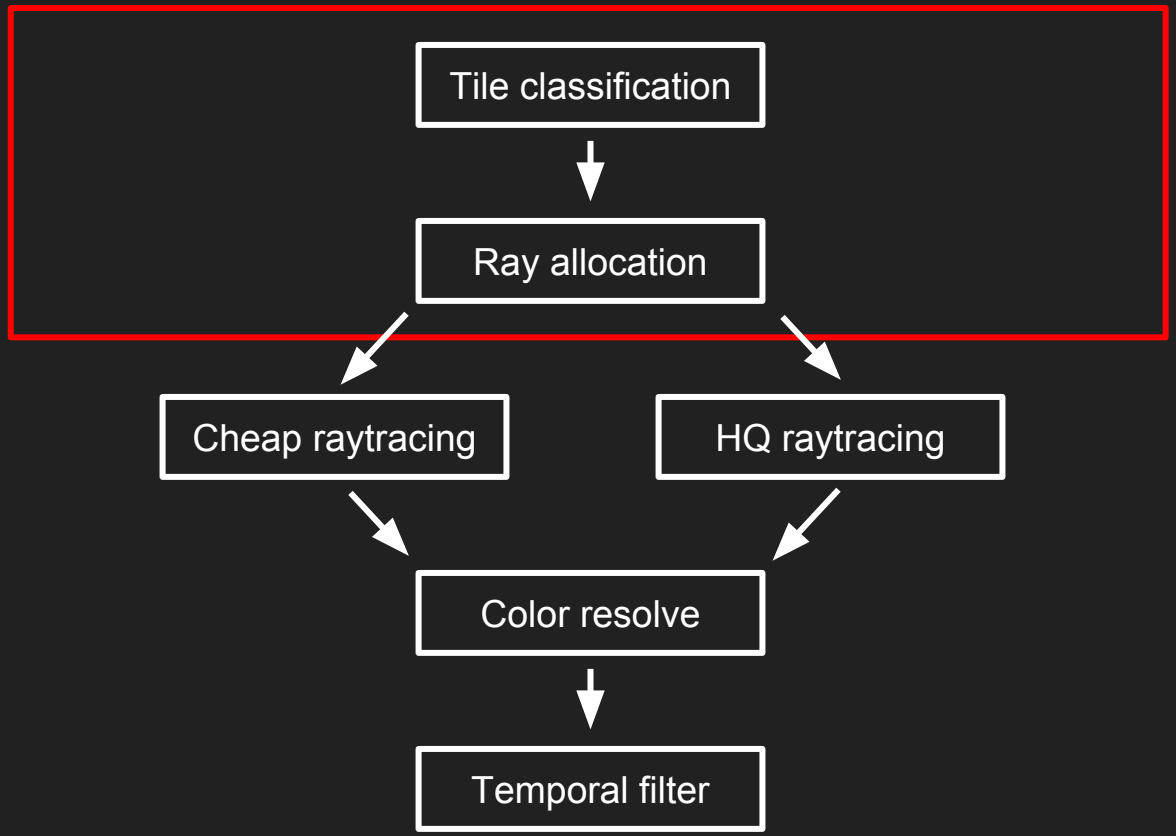
Linear Raymarch

Goal: Find intersection point

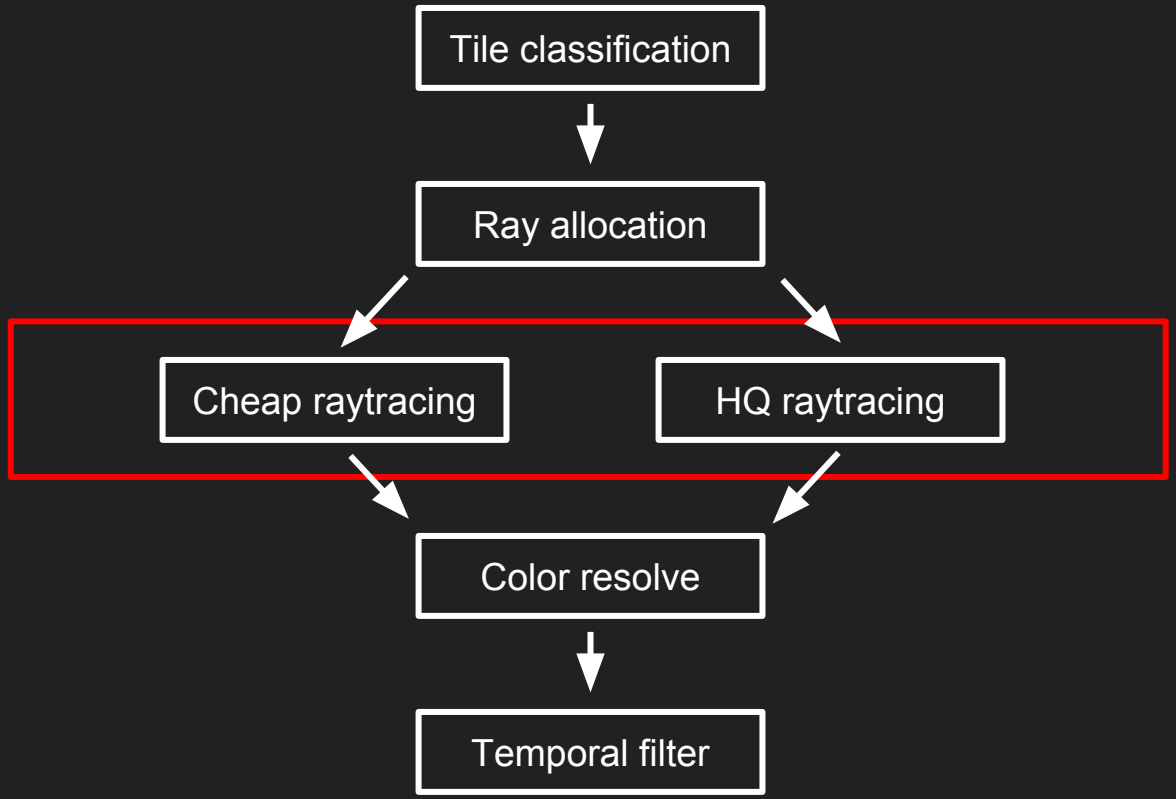
- At each step, check depth value
- Quality depends on step size
- Can be refined

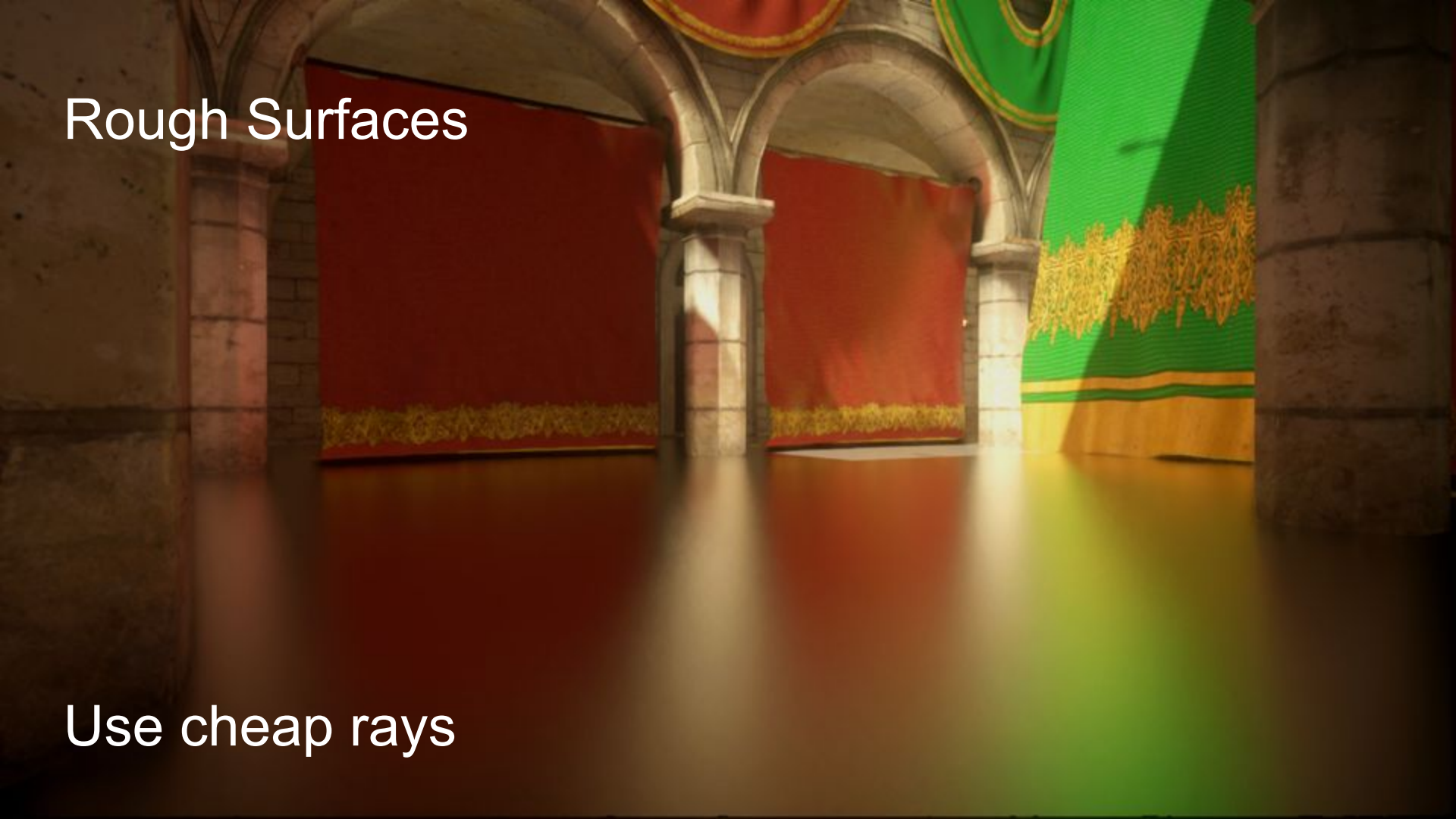


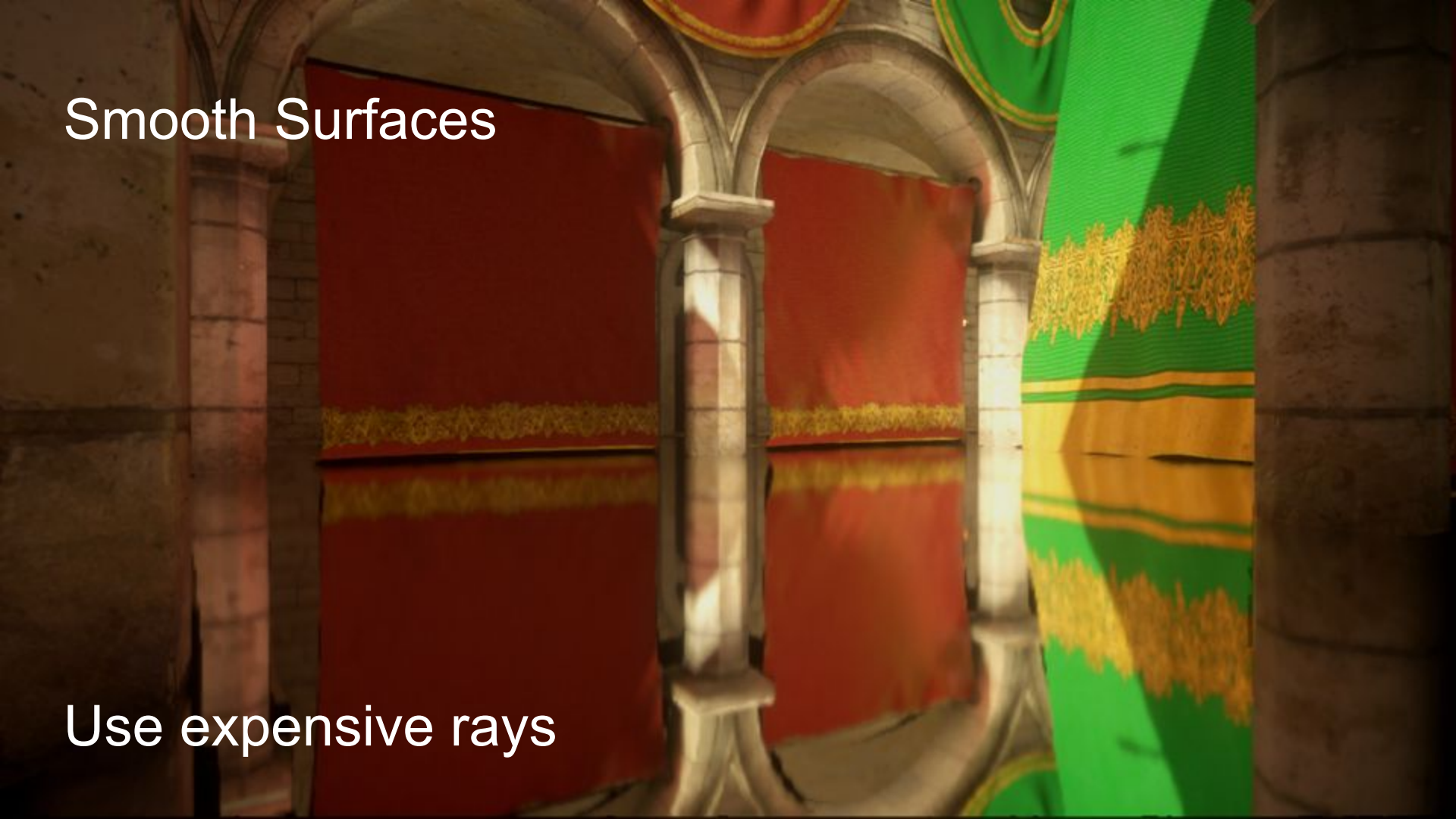
SSR in Frostbite



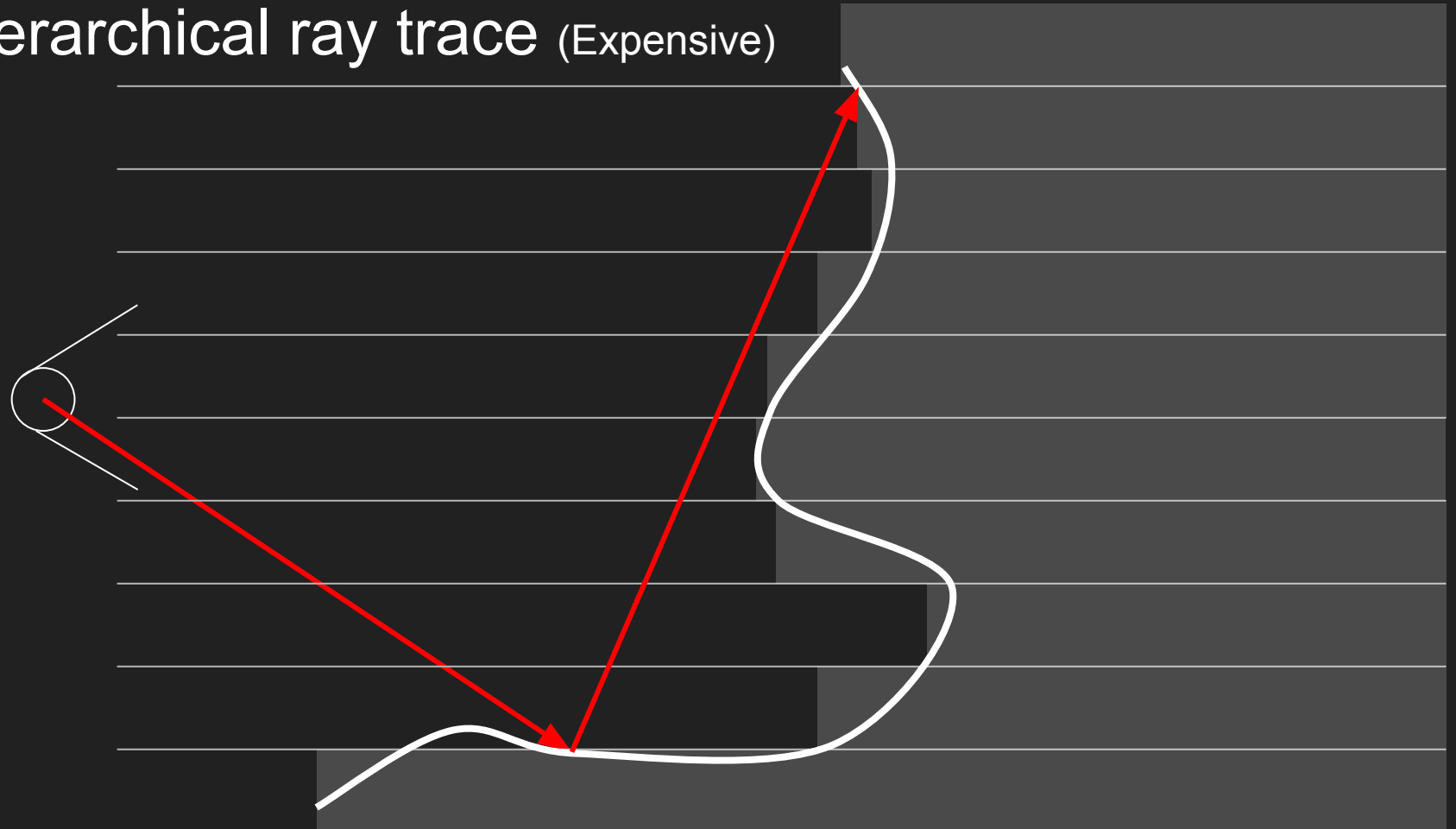
SSR in Frostbite




Rough Surfaces
Use cheap rays

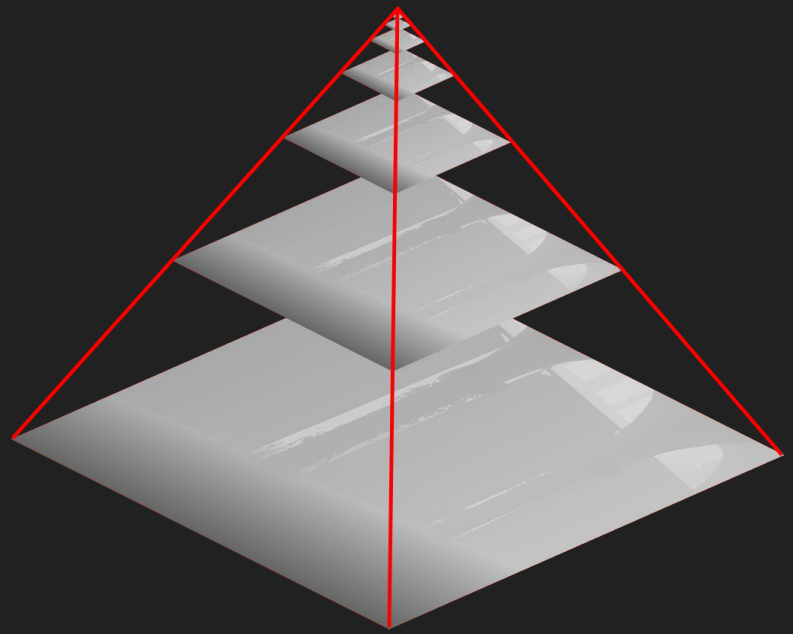

Smooth Surfaces
Use expensive rays

Hierarchical ray trace (Expensive)

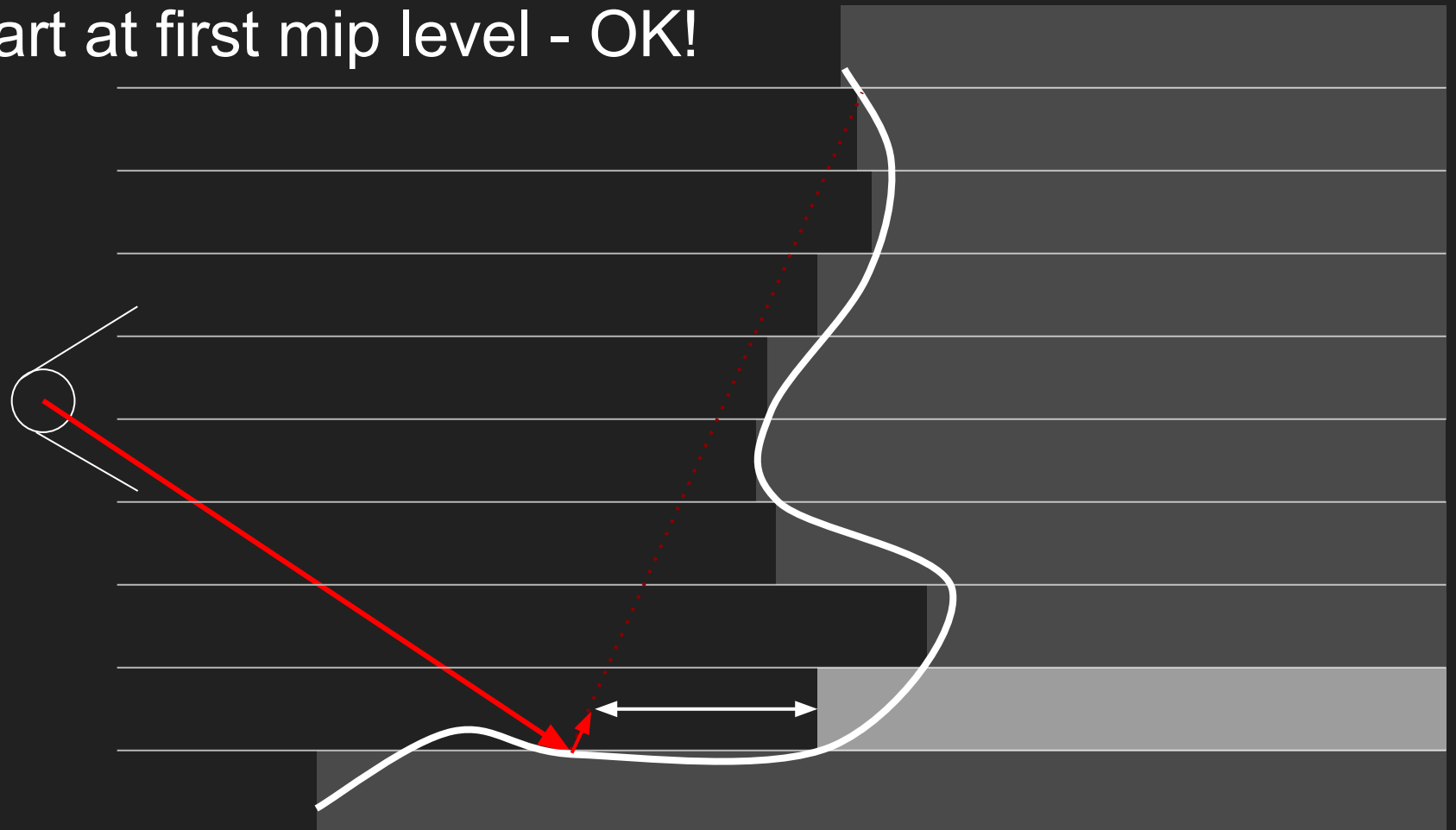


Generate Depth Mip-Map

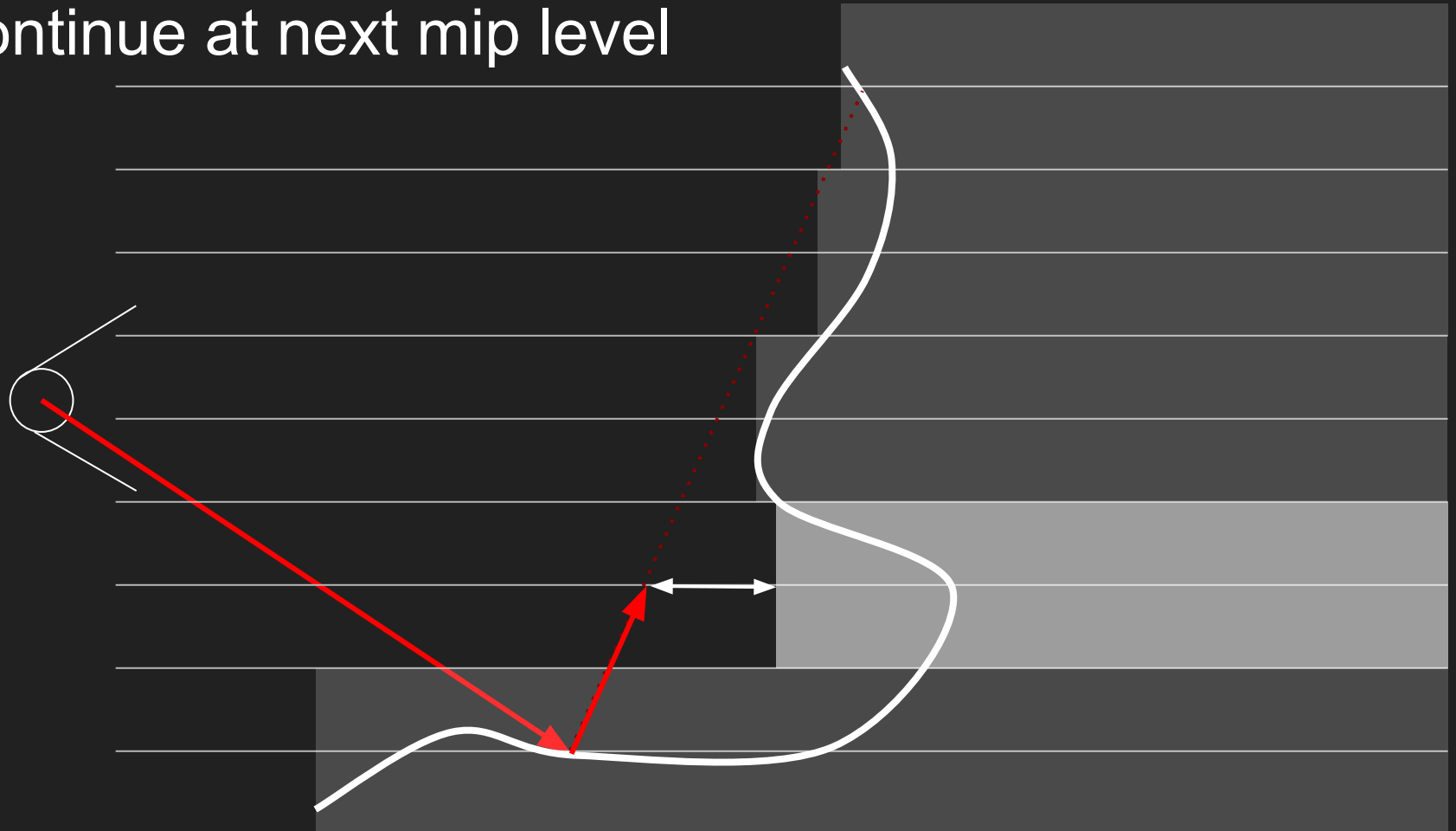
- Use min values instead of average



Start at first mip level - OK!

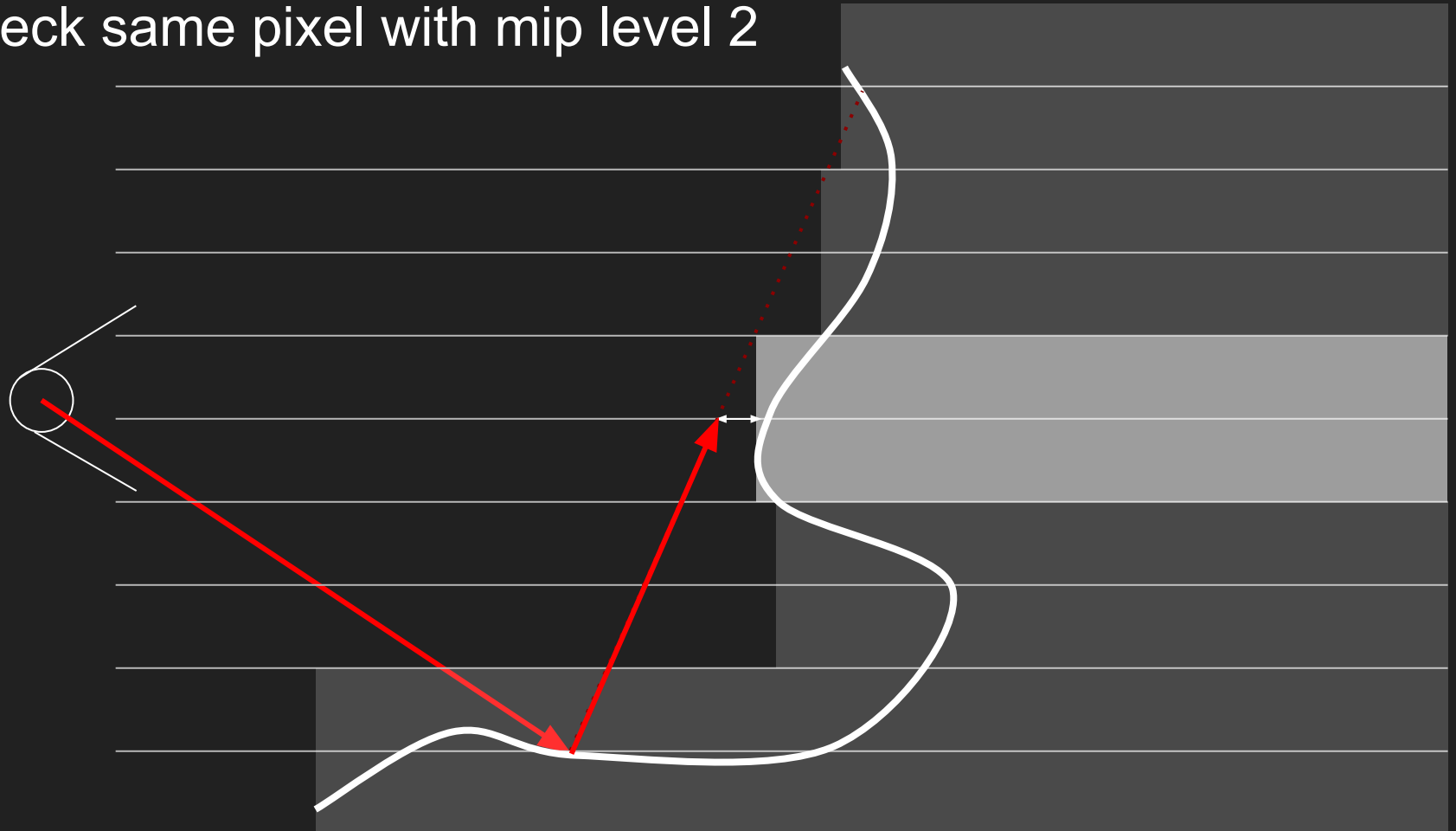


Continue at next mip level

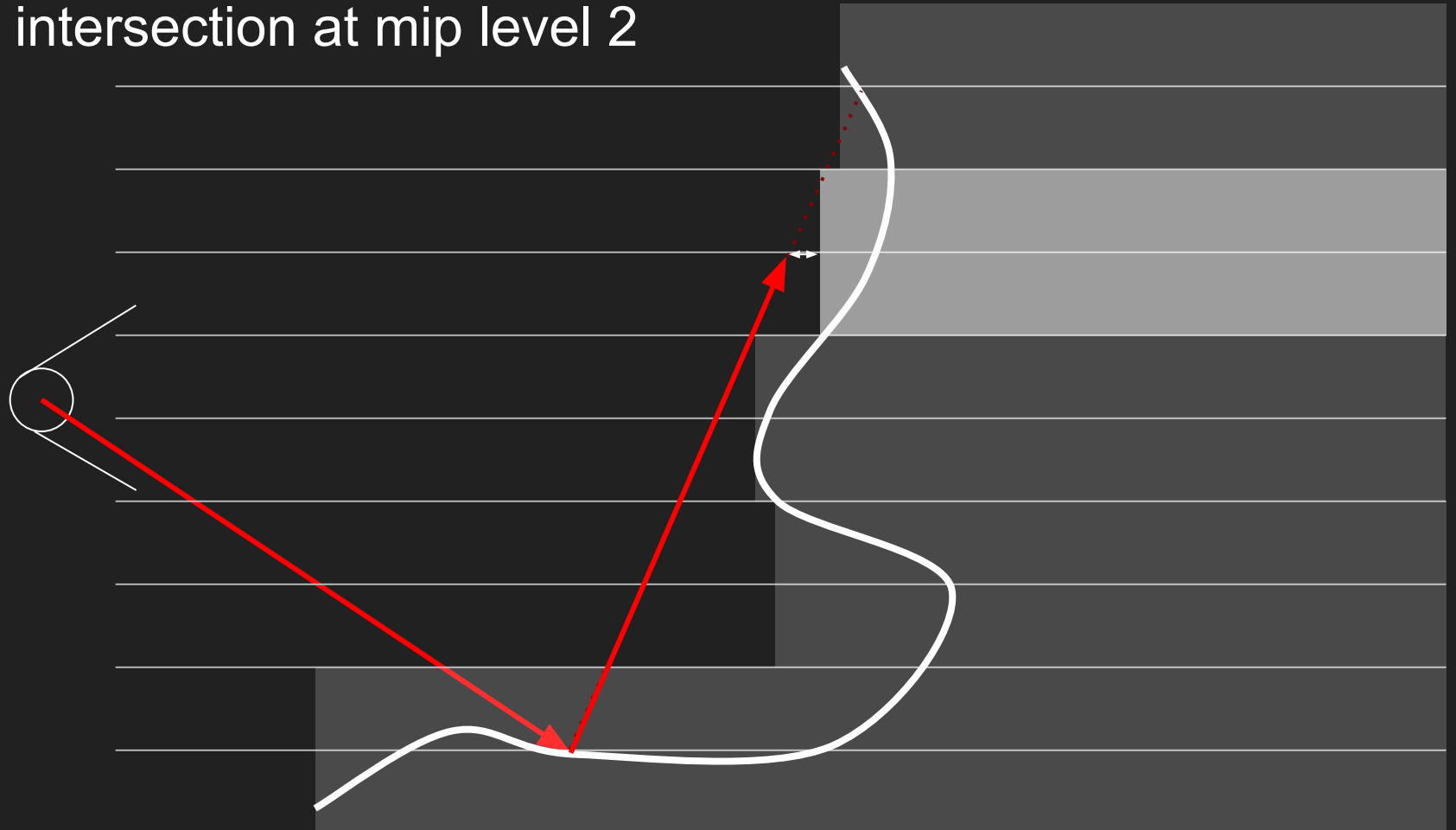


Intersection at 3 mip level!

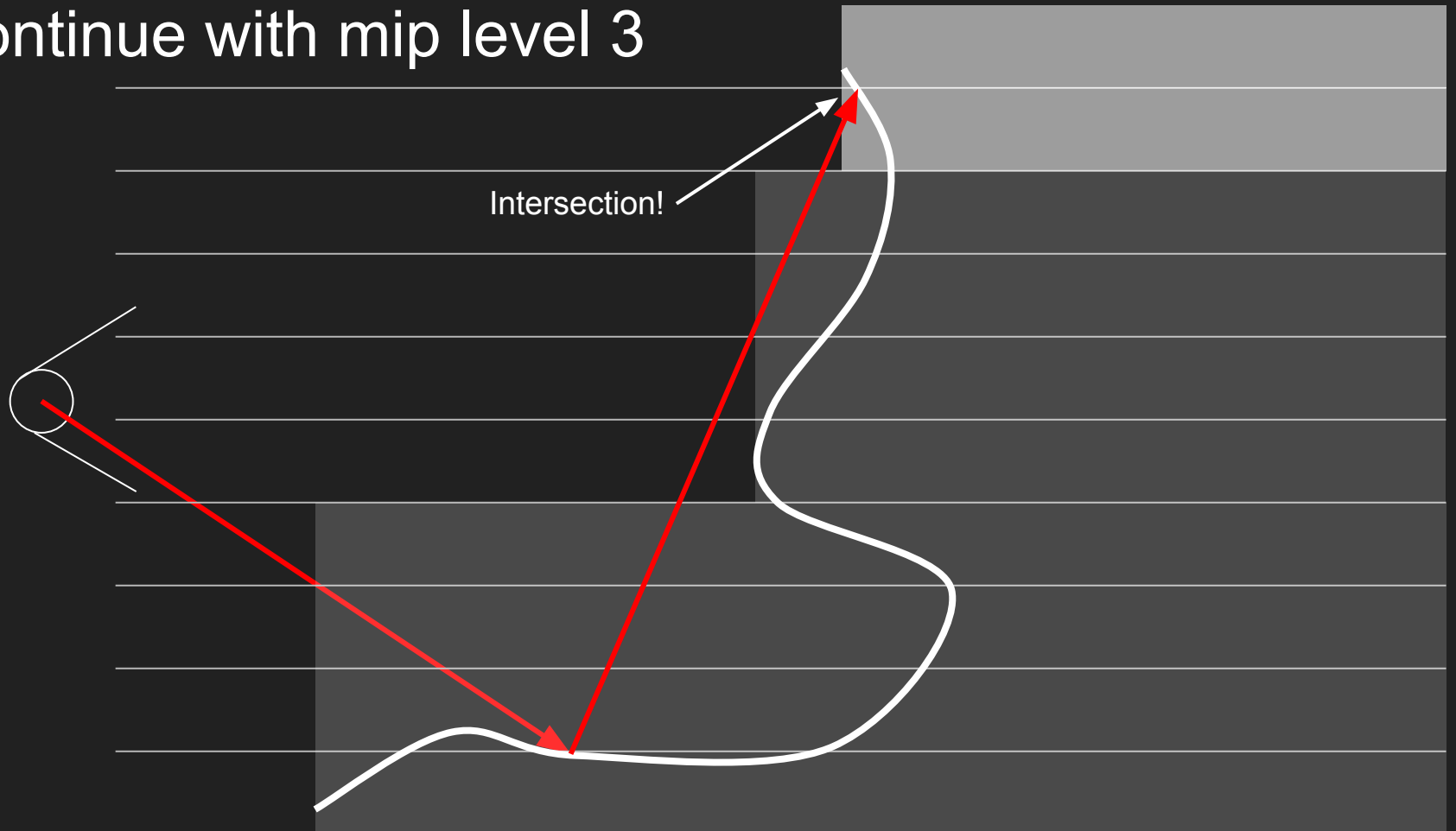
Check same pixel with mip level 2



No intersection at mip level 2



Continue with mip level 3

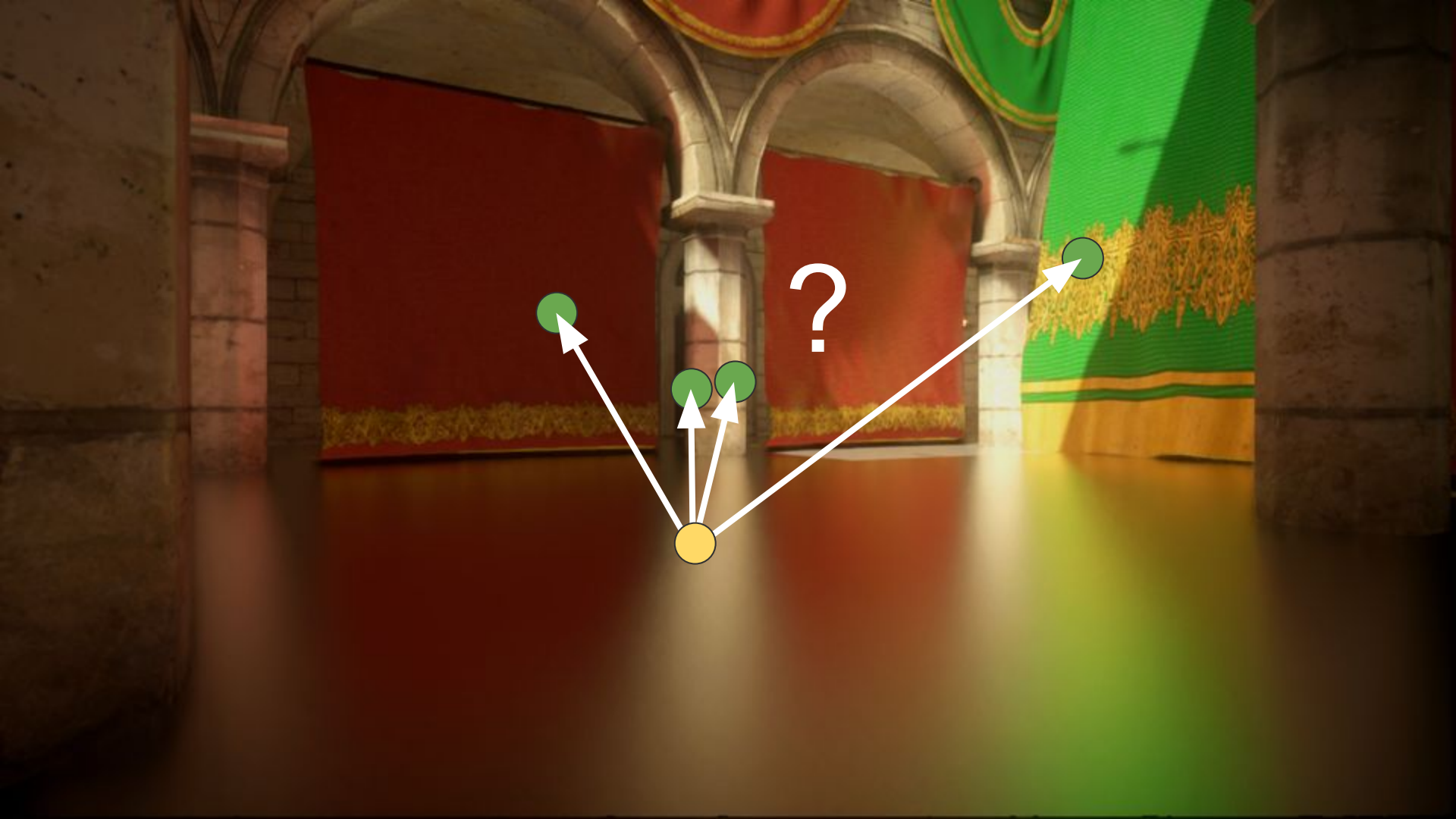


Check mip level 2

Intersection at mip level 1 ➔ DONE

What's the difference?

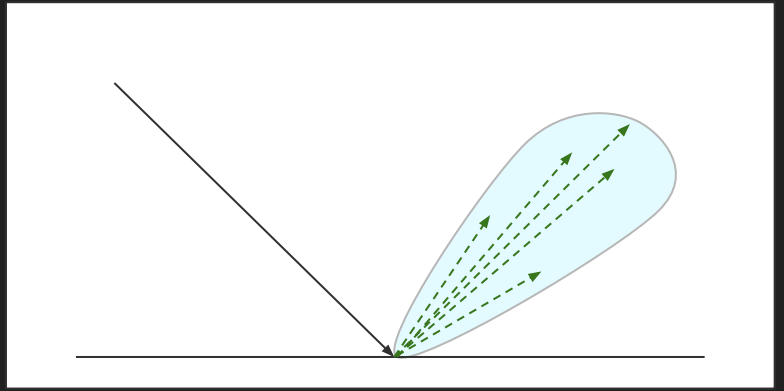
- Cannot miss tiny geometry
- Mip-map hierarchy
- More expensive

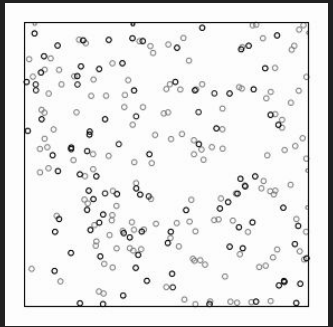


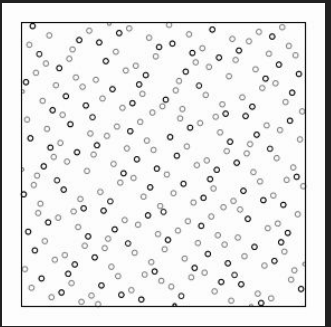
Reflection direction to use

Importance Sampling:

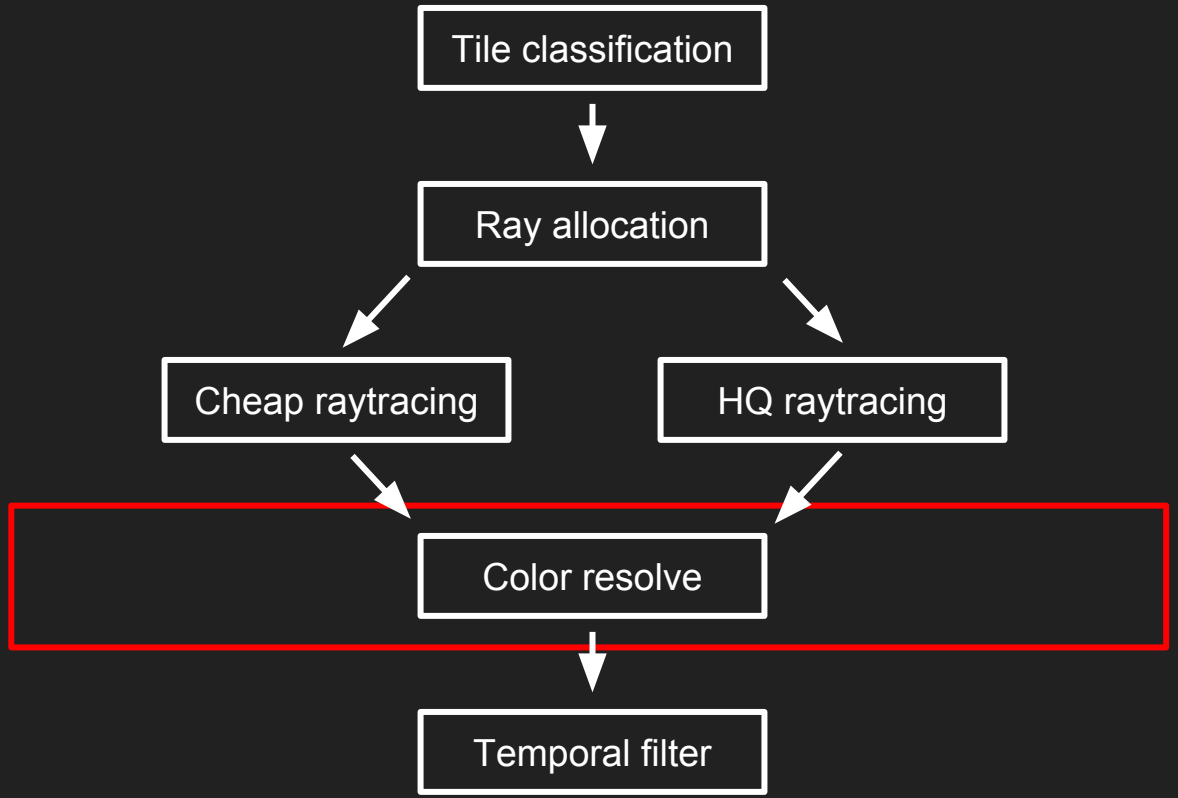
- BRDF
- Halton sequence





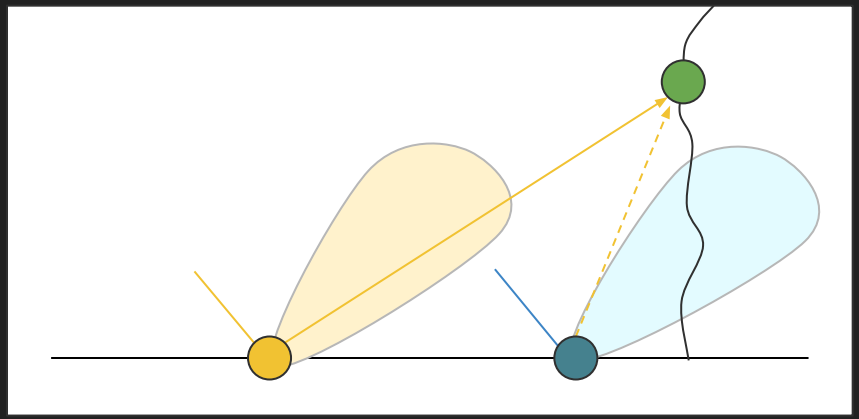


SSR in Frostbite



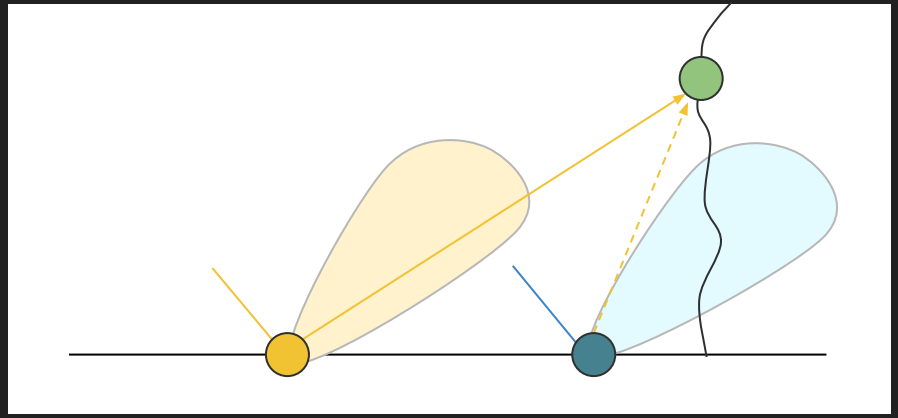
Neighbor Rays

- Nearby pixels likely to give similar reflection
- Reuse nearby intersection points
- Need to weight accordingly



Weighing Neighbors

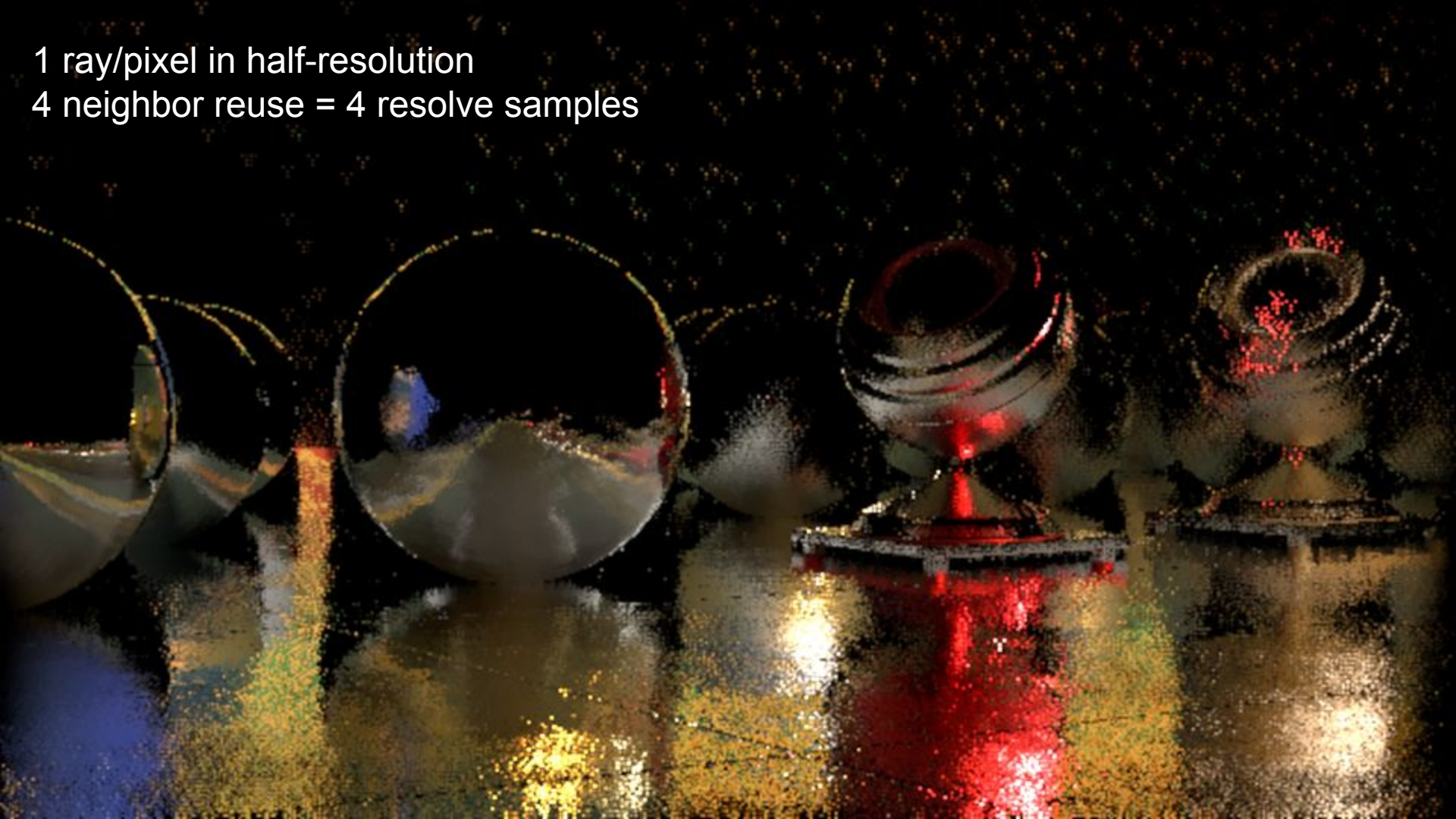
- Green in formula is important part
- "Variance reduction"
- For each nearby hit
  - weight = f_s(hit) / p_k
  - contribution = color(hit) * weight
- FG is precomputed BRDF factor
- They use $N = 4$



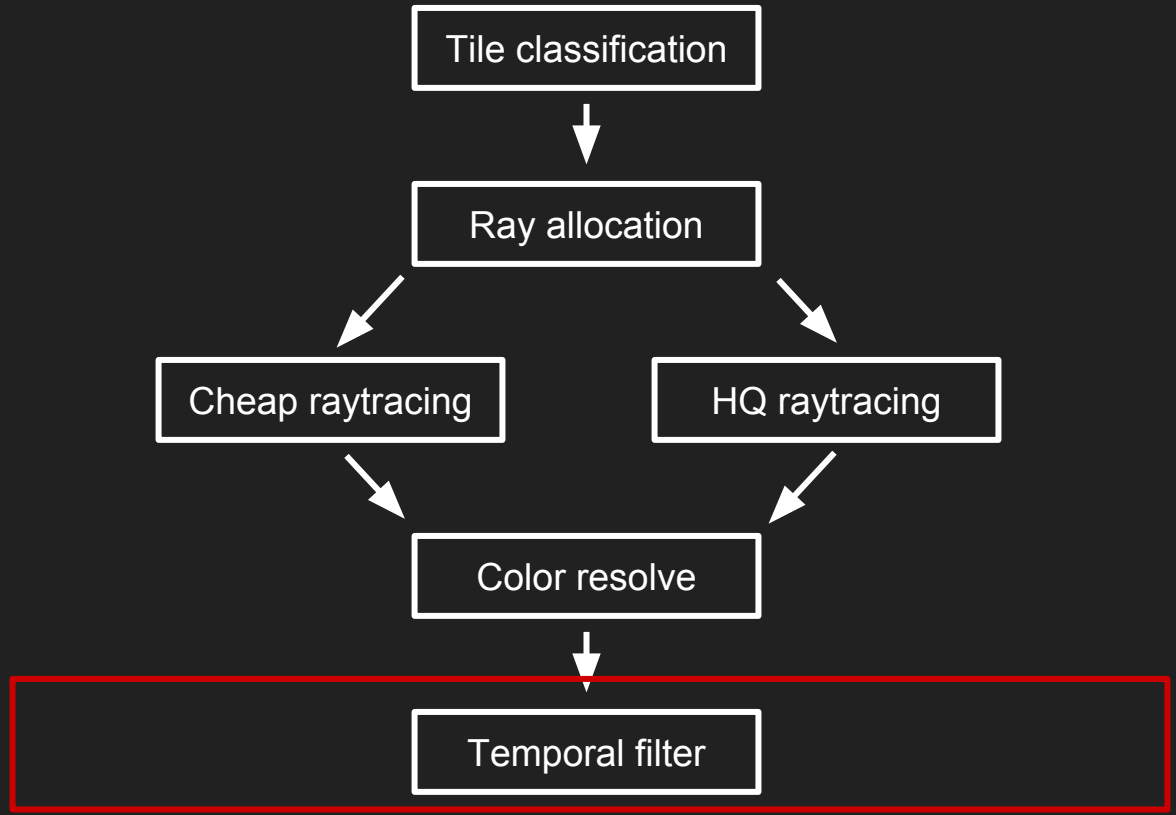
$$L_0 \approx \frac{\sum_{k=1}^{N} \frac{L_i(l_k) f_s(l_k \rightarrow v) \cos\theta_{ik}}{p_k}}{\sum_{k=1}^{N} \frac{f_s(l_k \rightarrow v) \cos\theta_{ik}}{p_k}} FG$$


1 ray/pixel in half-resolution
No ray reuse

1 ray/pixel in half-resolution
4 neighbor reuse = 4 resolve samples



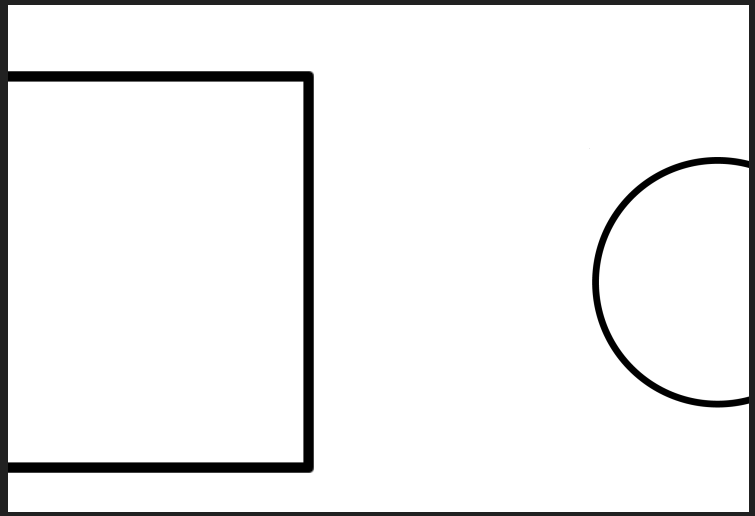
SSR in Frostbite

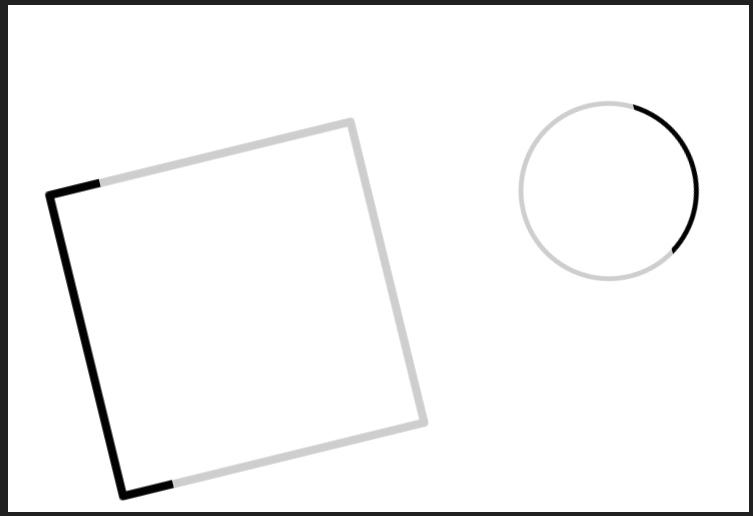


Temporal Reprojection (filtering)

Reproject last frame into current frame

$$\vec{x}_t = P_t V_t (P_{t-1} V_{t-1})^{-1} \vec{x}_{t-1}$$





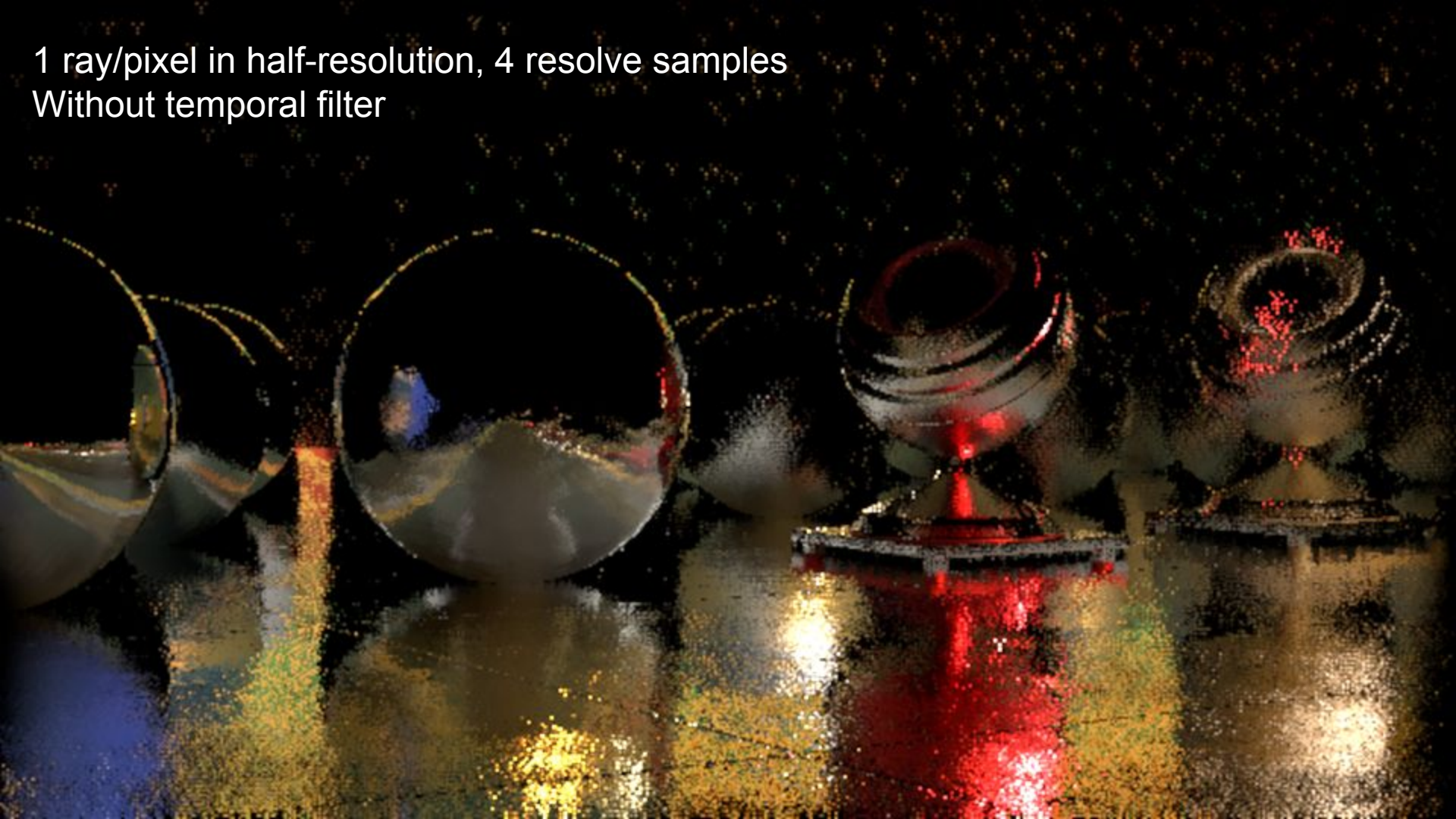
Temporal Reprojection (filtering)

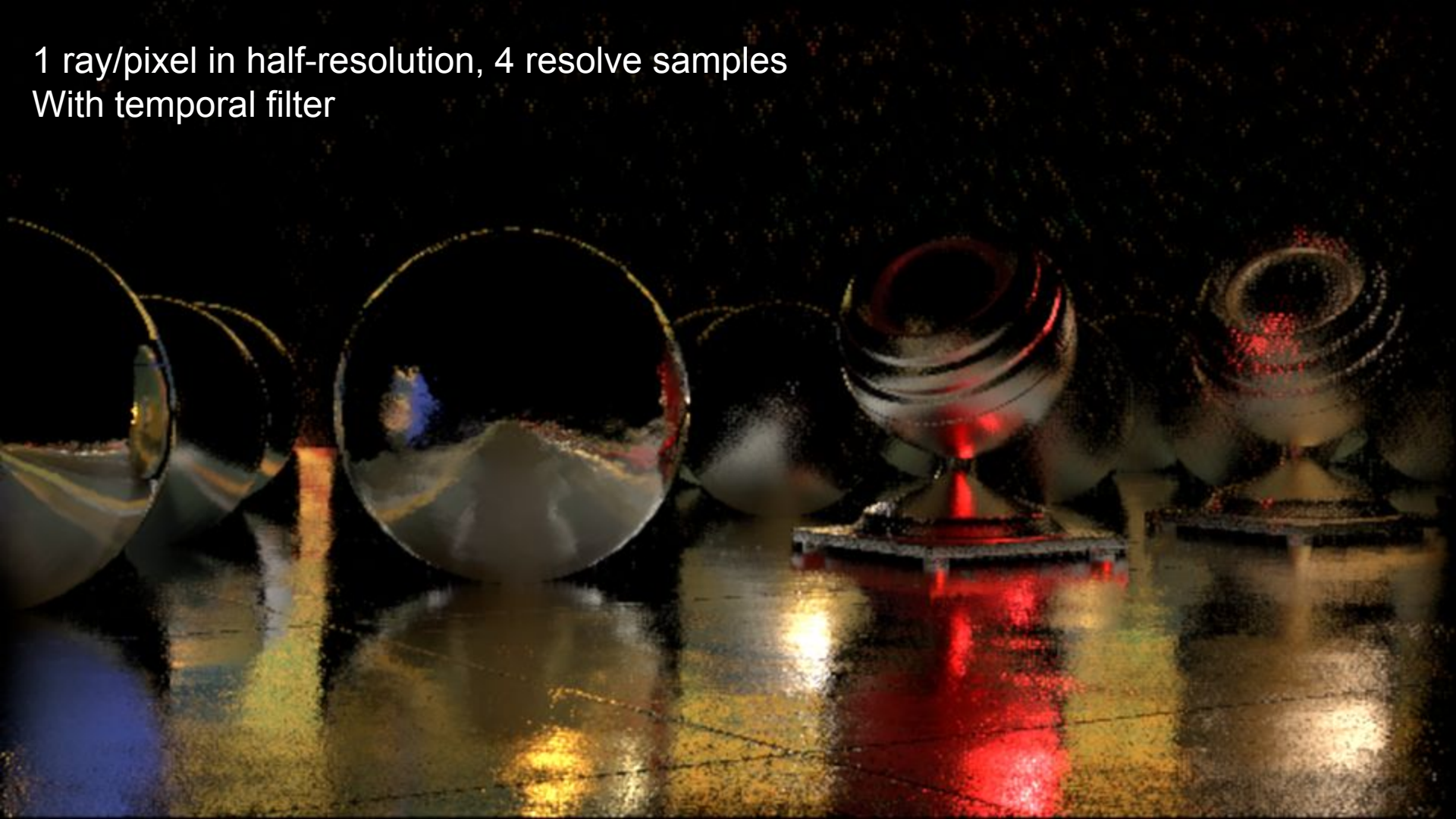
- Problems: smeared reflections when moving camera
- Reproject ray intersection location instead of reflected surface location

Camera

Surface location

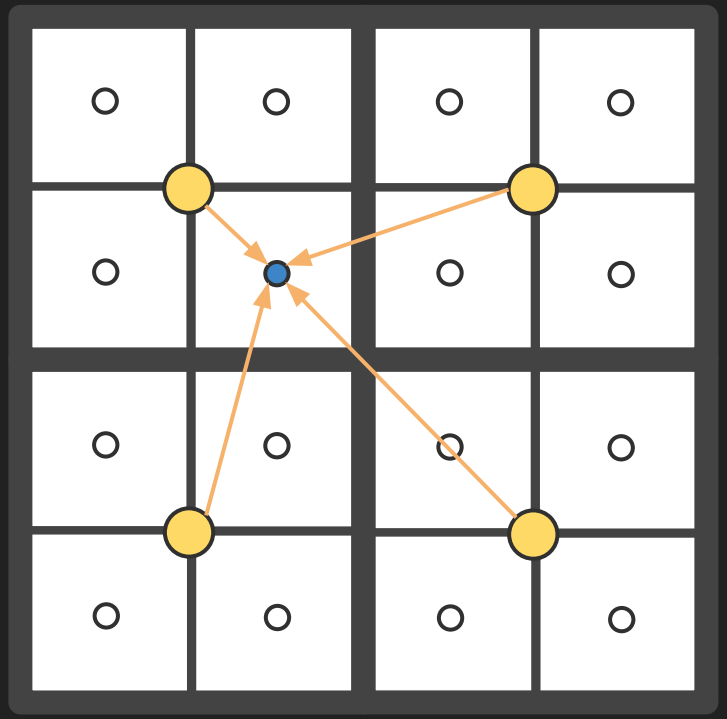
Ray intersection


1 ray/pixel in half-resolution, 4 resolve samples
Without temporal filter


1 ray/pixel in half-resolution, 4 resolve samples
With temporal filter

Sparse Ray Tracing

- Decouple ray tracing from color resolve
- Only do ray tracing in half resolution
- Promote color resolving to full resolution
- They still use four nearby in resolve

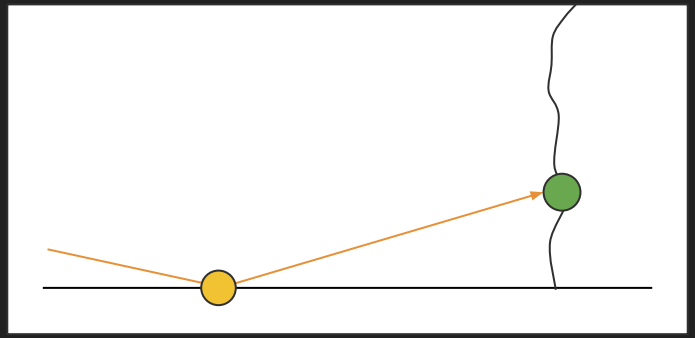


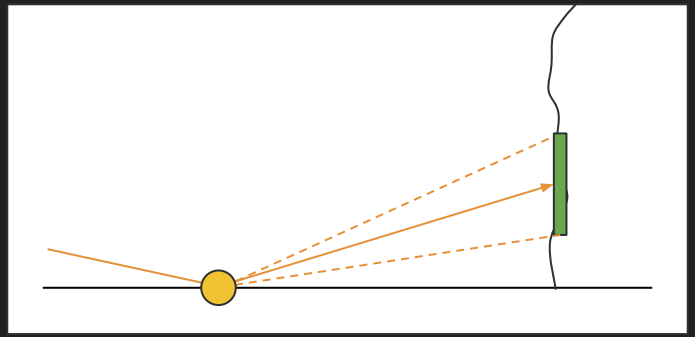
Temporal filter +
1 ray/pixel in half resolution +
4 resolve samples in **half** resolution

Temporal filter +
1 ray/pixel in half resolution +
4 resolve samples in **full** resolution

Filtered Importance Sampling

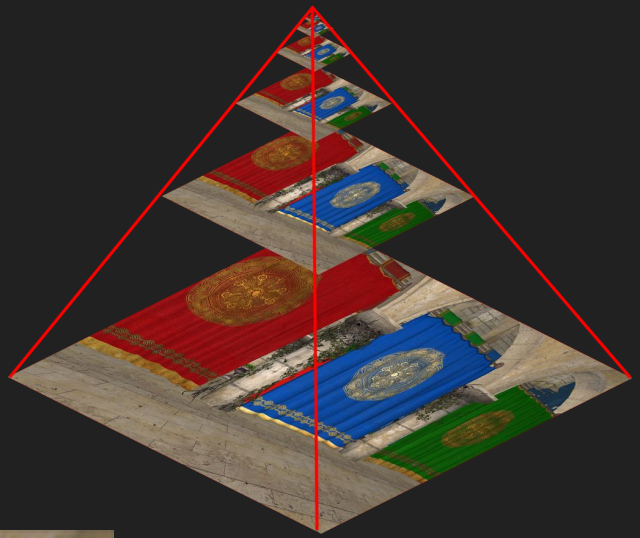
- Pretend rays are cones
- Wider intersection point

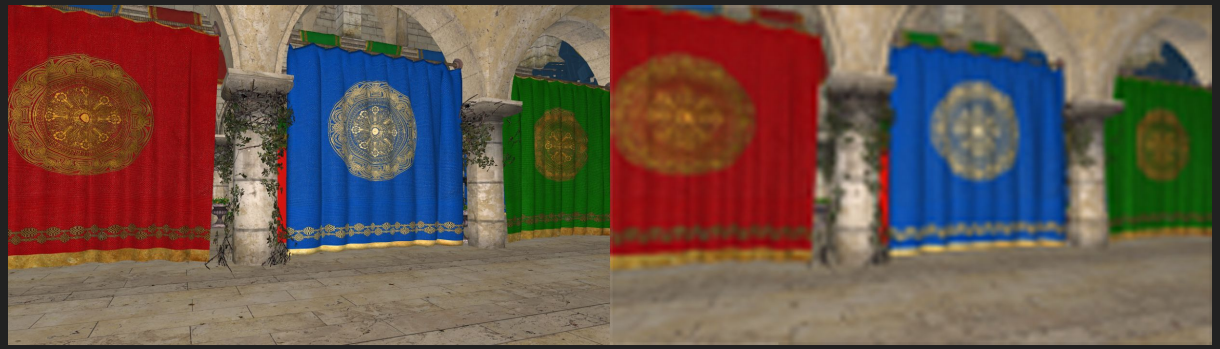




Filtered Importance Sampling

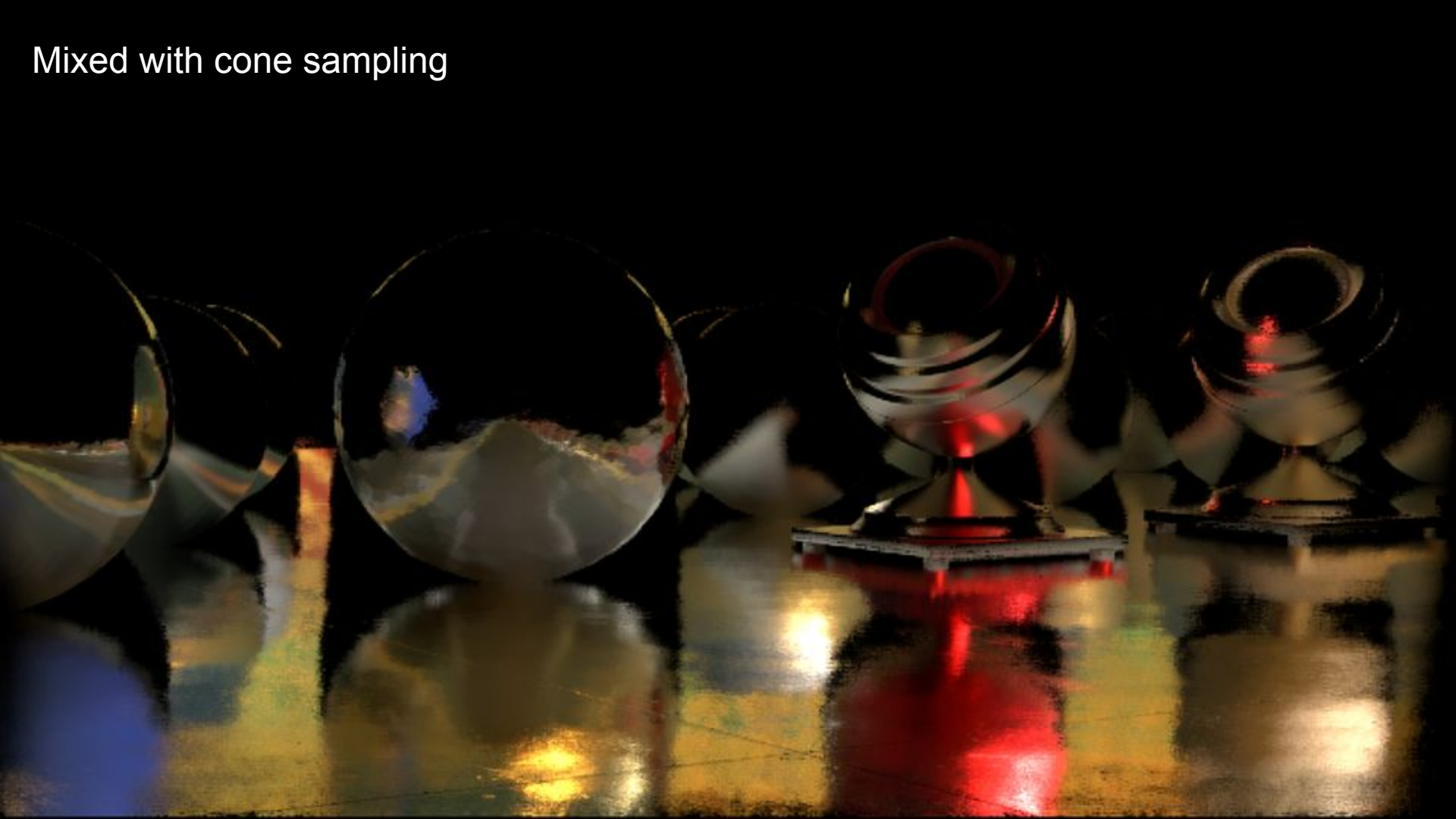
- Use mipmapped values in resolve
- Mip level depend on
  - Roughness
  - Distance to hit
- Mix with monte carlo approach





Previous result (Monte carlo)

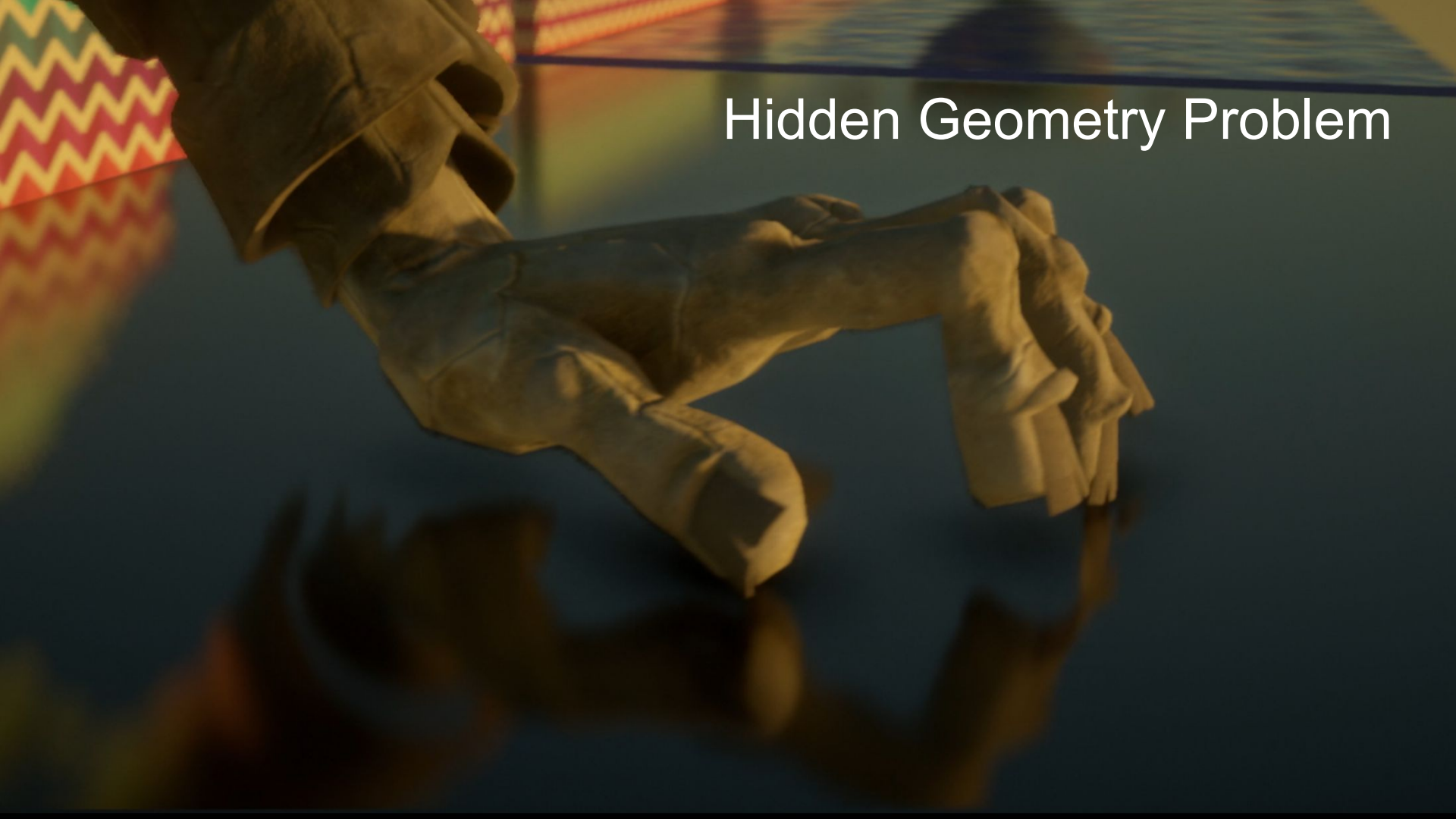
Mixed with cone sampling



Performance

- PS4, 1600x900
- Benchmark parameters
  - Ray trace once per half pixel
  - Use four nearby rays in color resolve
  - <20% HQ rays
- Takes 2.19 ms
  - Most expensive step: 0.8 ms in color resolving

However, screen space reflection is not perfect


Hidden Geometry Problem


Edge Cutoff


Edge Fading

Summary

- Limited data, make the best of it
- Many tricks and optimisations
- Hiding some artifacts, can achieve useful result

Video

Bonus Slides!


Usage Today
Just Cause 3


Just Cause 3


Metal Gear Solid V: The Phantom Pain


Metal Gear Solid V: The Phantom Pain

Crysis 2 (introduced technique)


Crysis 2 (introduced technique)
RLR off


Fallout 4


Fallout 4


Mirror’s Edge Catalyst (pre-release)


Mirror’s Edge Catalyst (pre-release)