Rendering: Next Event Estimation

Bernhard Kerbl

Research Division of Computer Graphics
Institute of Visual Computing & Human-Centered Technology
TU Wien, Austria

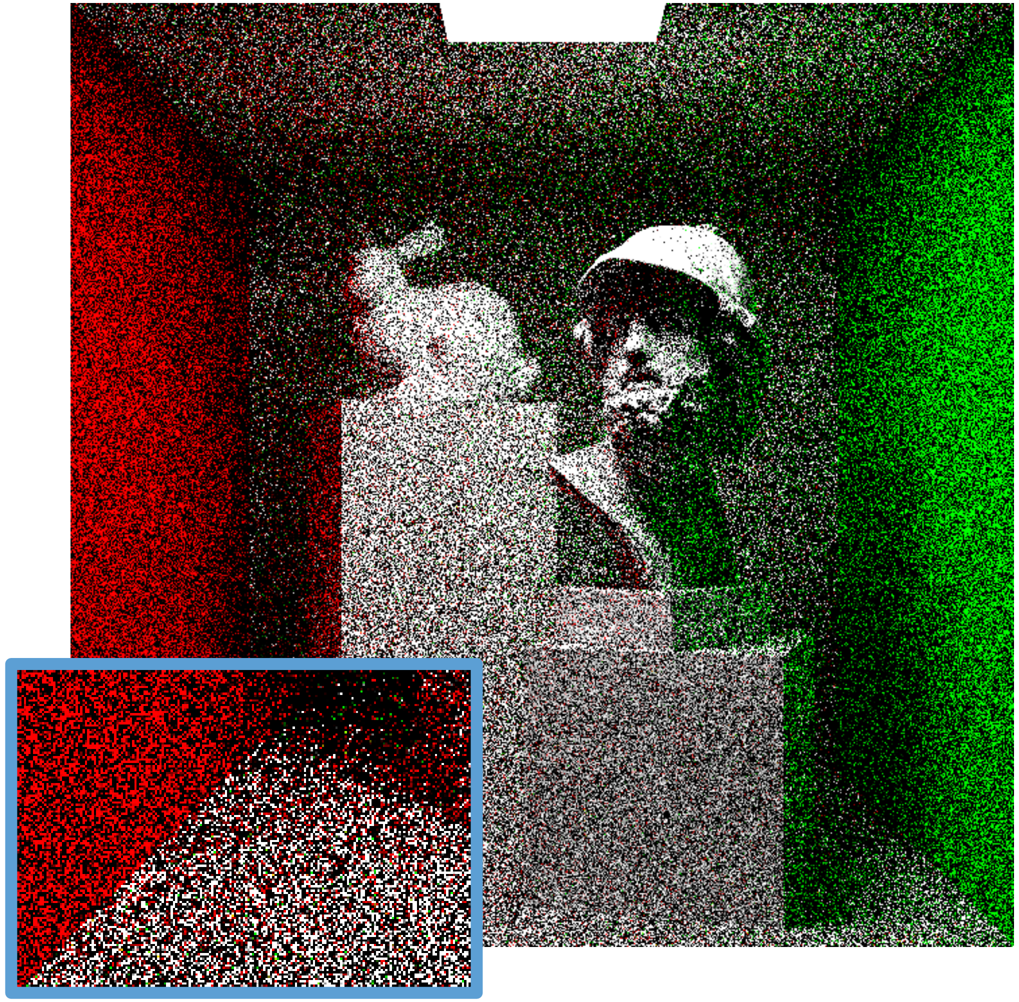
Accepting Importance Sampling as New Standard

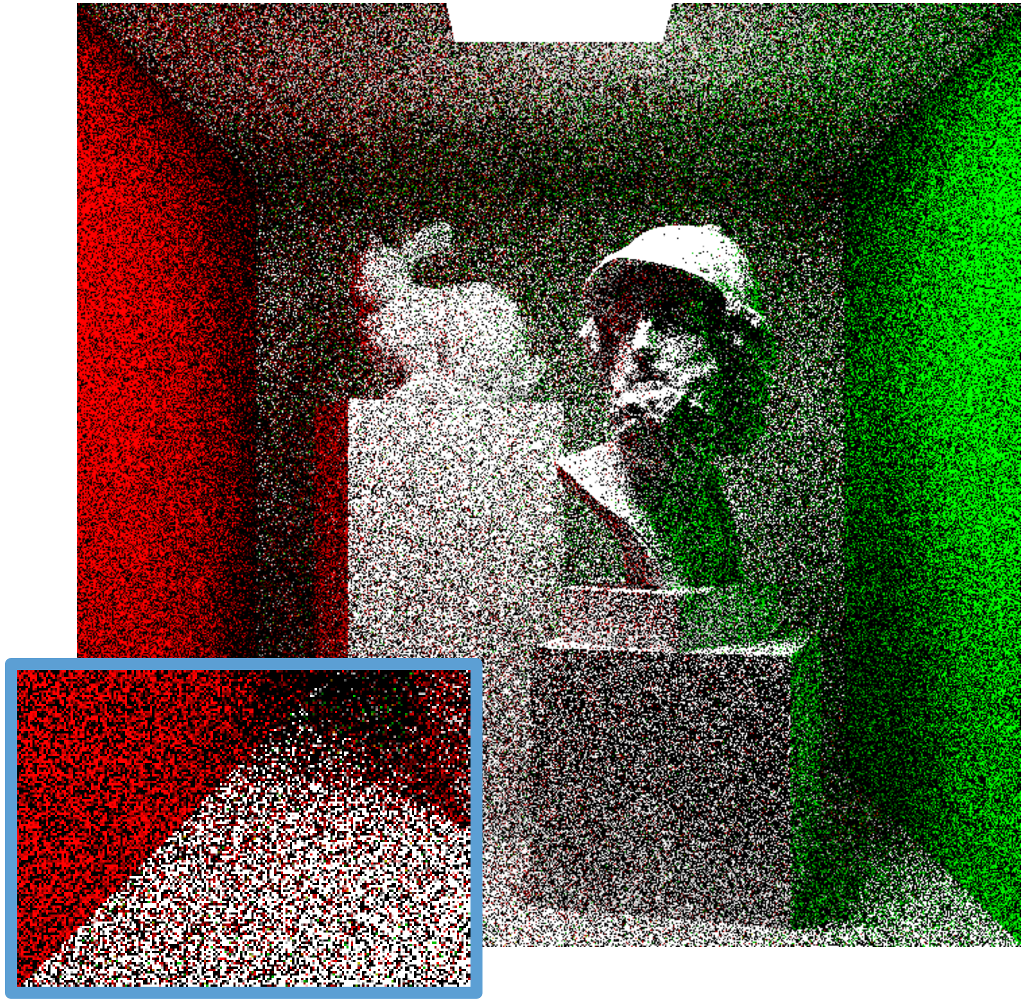
- At this point, we should mention that we won‘t be needing uniform hemisphere sampling anymore…

- We can replace it for most purposes with BRDF sampling, that is, importance sampling the hemisphere based on the material
  - E.g., for diffuse materials, cosine-weighted hemisphere sampling
  - We will see solutions for other materials soon!

- BRDF sampling usually improves quality in most cases (how much?)

Diffuse Path Tracing: Uniform vs BRDF Sampling





- Cosine-weighted hemisphere sampling “works”… can we do better?

How to Get Rid of All that Noise?

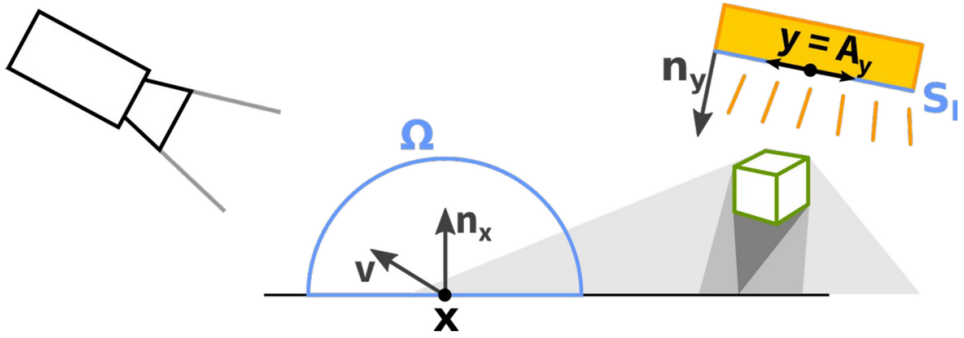
- Higher-dimensional path tracing is particularly prone to noise

- How can we reduce noise in our renderings?

- Common suggestions when looking for ways to reduce noise:
  - Use more samples (brute force, often takes too much time)
  - Use importance sampling (already applied)
  - Use today's technique, **Next Event Estimation (NEE)**
    - Based on something we saw before: light source sampling

Revisiting the Math Behind Light Source Sampling

Direct lighting (soft shadows) (usable for rendering)



light intensity at position y on the surface

emitter cos(θ)

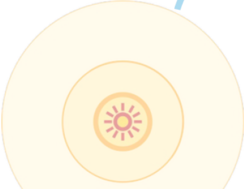
visibility (new, ray tracing)

$$L_e^{[l]}(x) = \int_{S_l} f_r(x, y \rightarrow v) L_e^{[l]}(y) V(x,y) \cos(\theta_x) \frac{\cos(\theta_y)}{r^2} \, \mathrm{d}A_y$$

Light going in direction v

Material, modelled by the BRDF

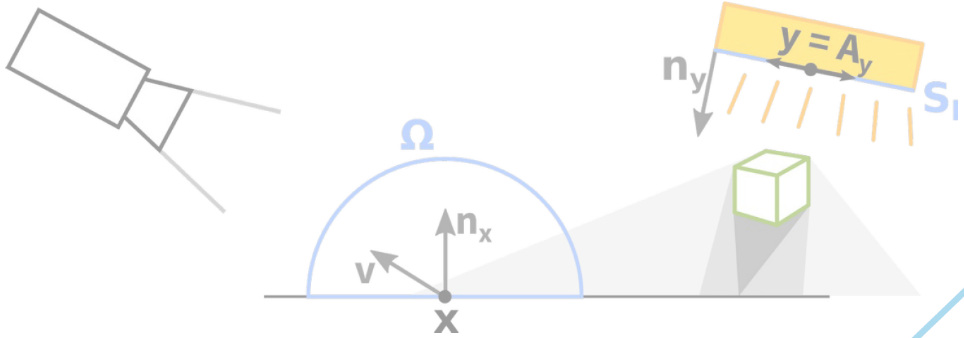
receiver cos(θ)



distance

Revisiting the Math Behind Light Source Sampling

Direct lighting (soft shadows) (the same, but more explicit)



light intensity at position y on the surface

visibility (new, ray tracing)

emitter cos(θ)

$$L_e^{[l]}(x) = \int_{S_l} f_r(x, y \to v) L_e^{[l]}(y) V(x,y) \left(\frac{y-x}{|y-x|} \cdot n_x\right) \frac{\left(\frac{x-y}{|x-y|} \cdot n_y\right)}{|x-y|^2} \mathrm{d}A_y$$

Light going in direction v

Material, modelled by the BRDF

receiver cos(θ)

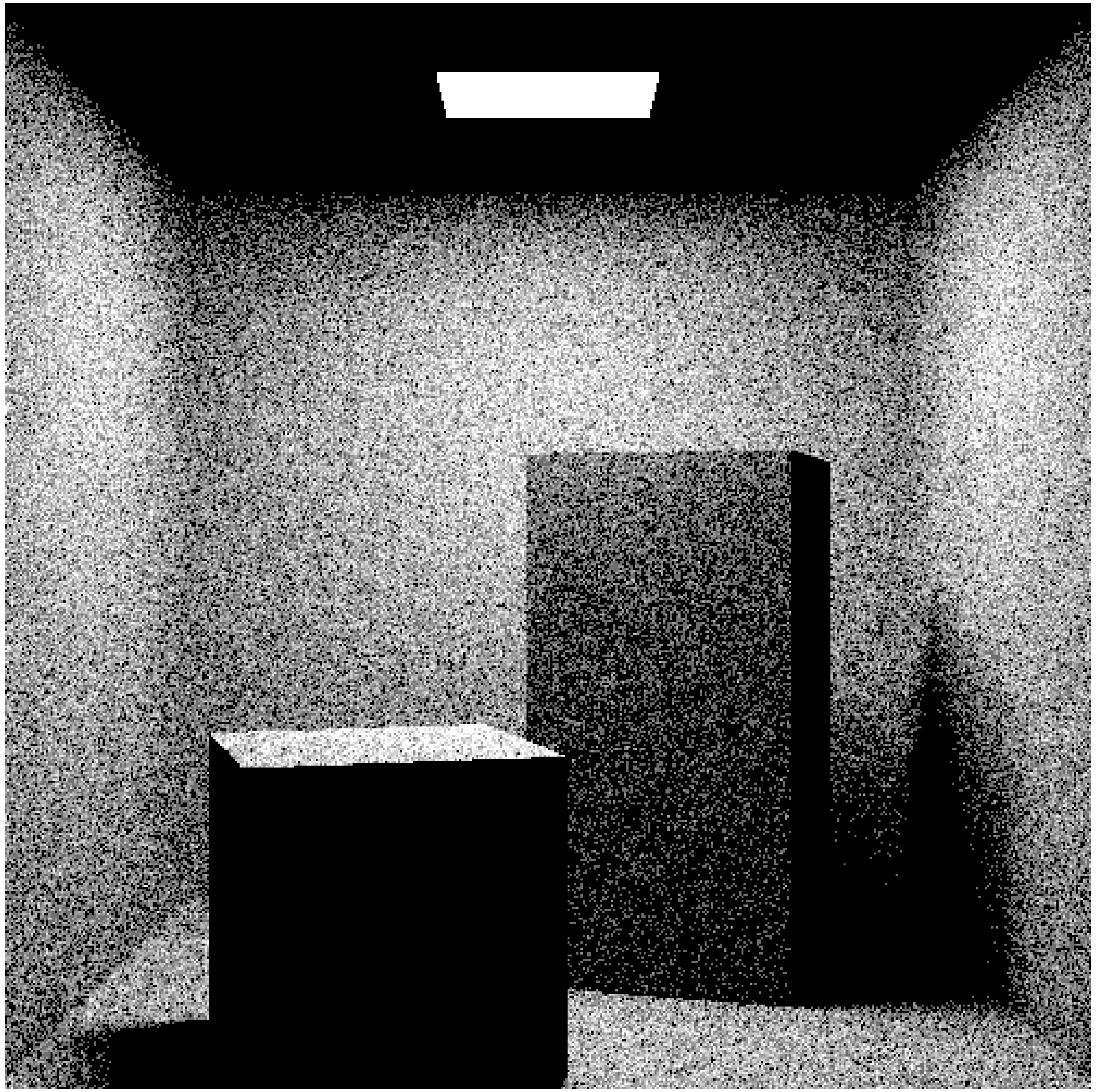
distance

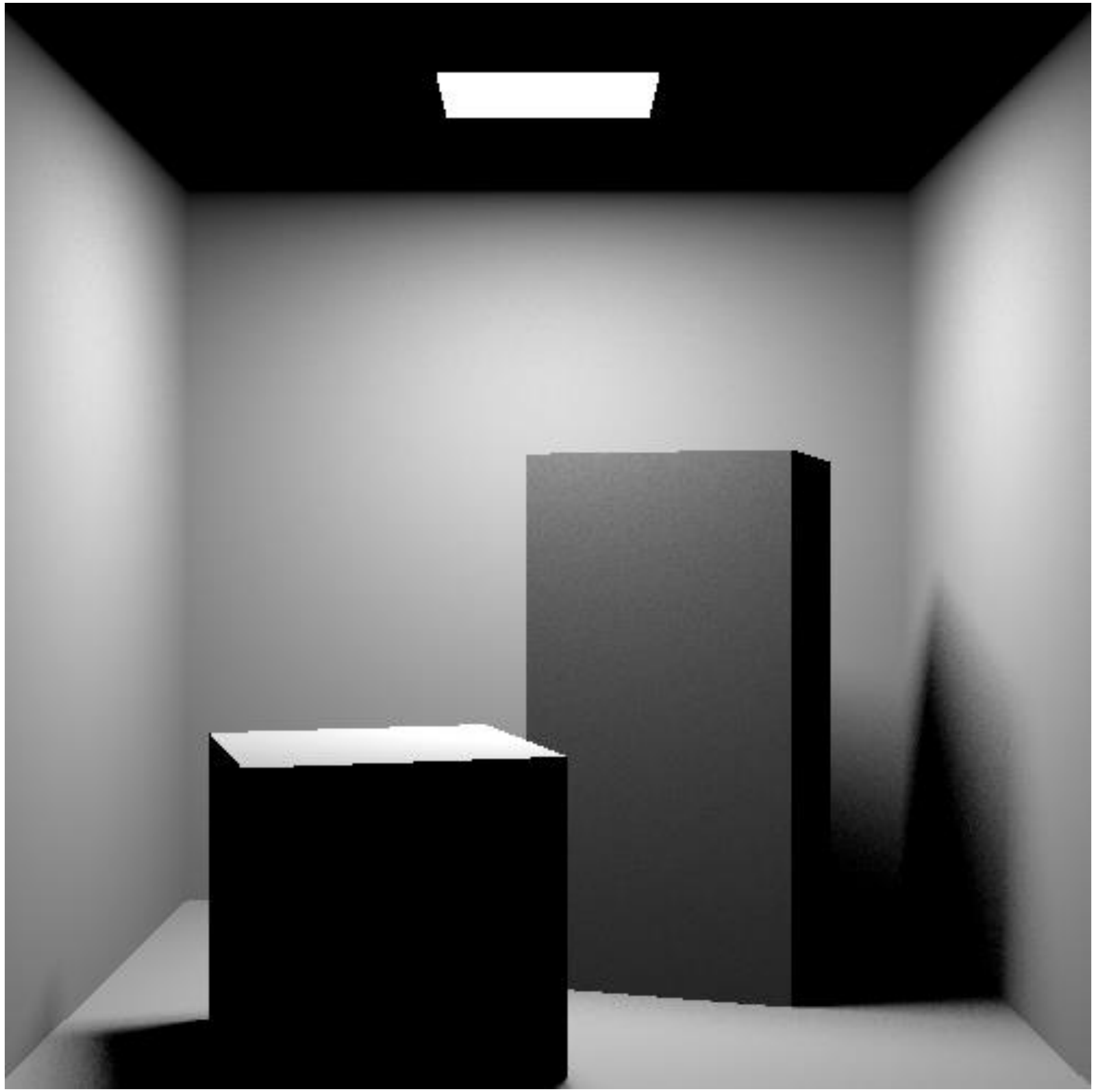
Direct Lighting with Light Source Sampling

$$f_r(x, \omega \to v) L_e^{[l]} \boxed{V(x,y)} \cos\theta \boxed{\frac{(-\omega \cdot n_y)}{|x-y|^2}} A^{[l]}$$

```
function Li(v_inv)
    x = scene.trace(v_inv)
    f = x.emit
    y, area = light_surface_uniform_world()
    omega = (y-x).normalised()
    r = make_ray(x, omega)
    v = 0
    if (scene.trace(r) == y)
        v = 1
    P = dot(y.normal, -omega) / dot(y-x, y-x)
    f = x.alb/pi * y.emit * v * dot(x.normal, omega) * P * area
return r
```

Direct Light: BRDF Sampling vs Light Source Sampling





Direct Light: BRDF Sampling vs Light Source Sampling

- In both cases, you want to find out from how many directions on the hemisphere a point does receive how much light
- With uniform or BRDF sampling, pick one direction for each sample, and pretend that this direction speaks for the entire hemisphere
  - But if you collect many of these estimates and average, you converge
- With light source sampling, pick a point y on the light, and pretend the evaluation (can y illuminate x?) holds for its entire surface area
  - But if you collect many of these estimates and average, you converge

Importance Sampling Direct Light: BRDF vs Light Source

- Both BRDF and Light Source Sampling perform importance sampling

- They selectively put samples at opportune places on the domain
  - BRDF: assuming uniform lighting, which directions contribute most?
  - Light source: knowing light locations, which directions may hit them?

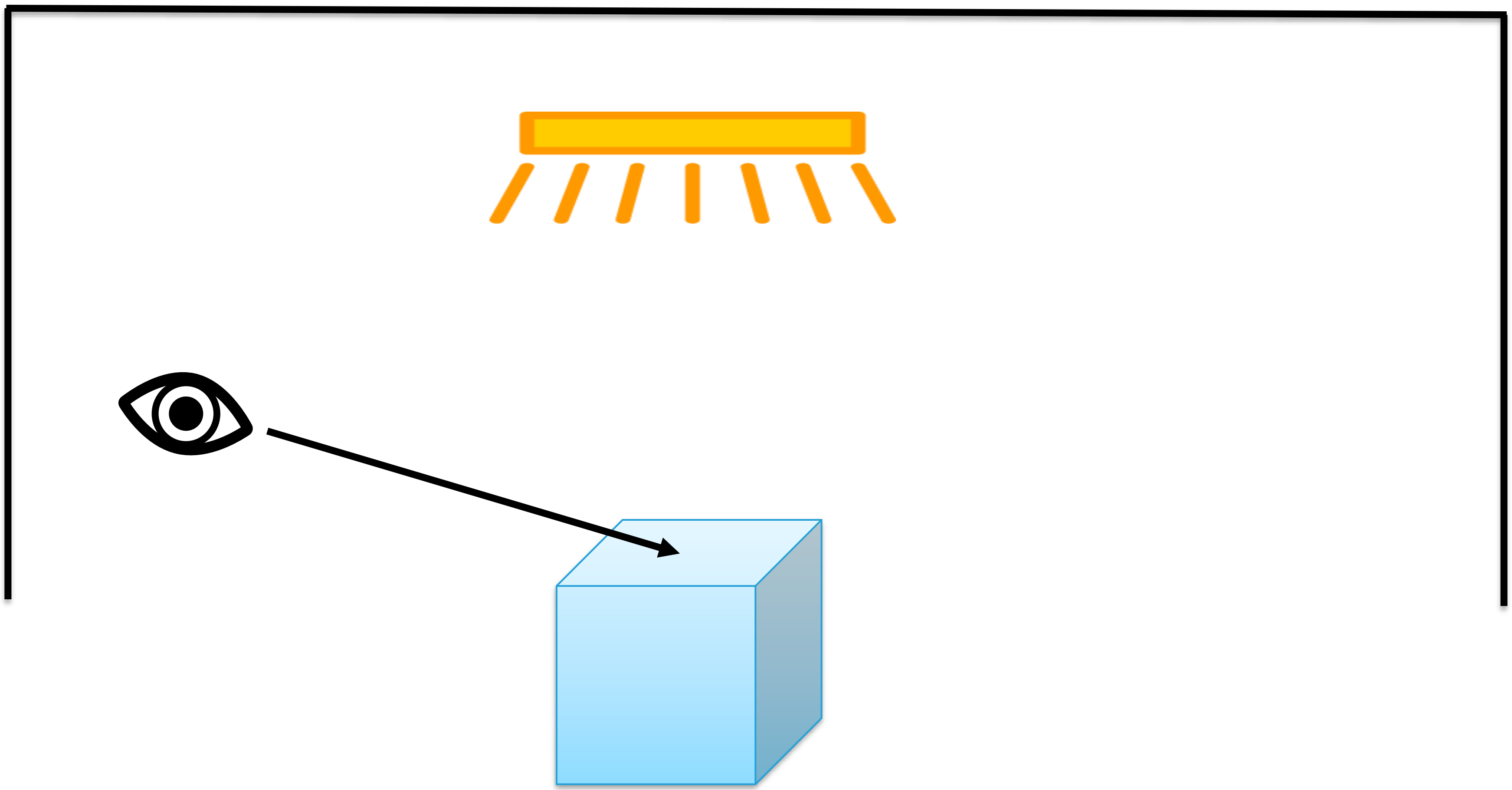
Exploiting Light Source Sampling for Path Tracing?

- Can light source sampling help us with path tracing?
  - Based on projecting all known light sources onto the hemisphere
  - Every surface in the scene is a potential source of (indirect) light!
  - If we treat every surface as a potential light source, we are back to randomly sampling the full hemisphere…

- Idea: follow each ray via multiple indirect bounces, but at each bounce, compute the direct lighting from light source surfaces!
  - Detected light at each bounce is no longer dependent on coincidence
  - This is what we refer to as ***Next Event Estimation***

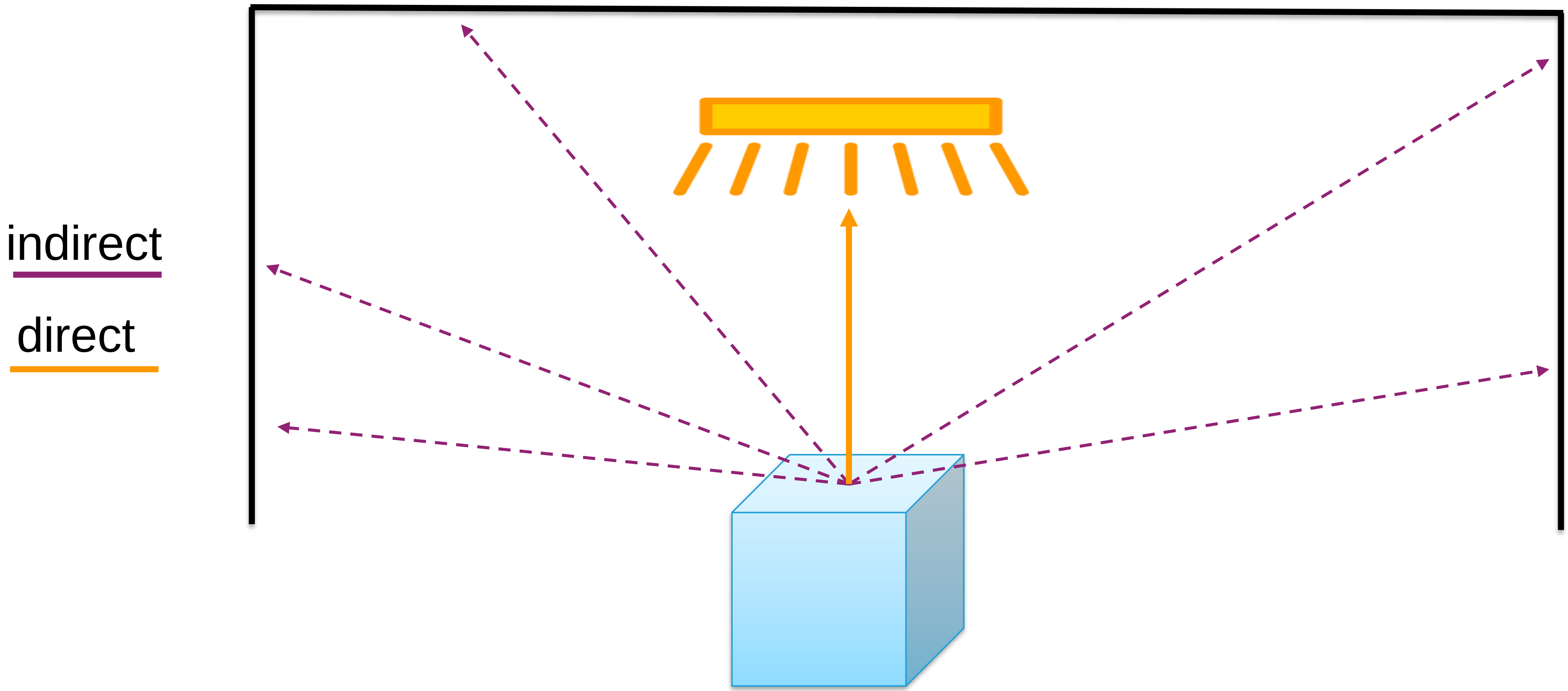
Next Event Estimation

- Builds on light source sampling. Think: where can light come from?



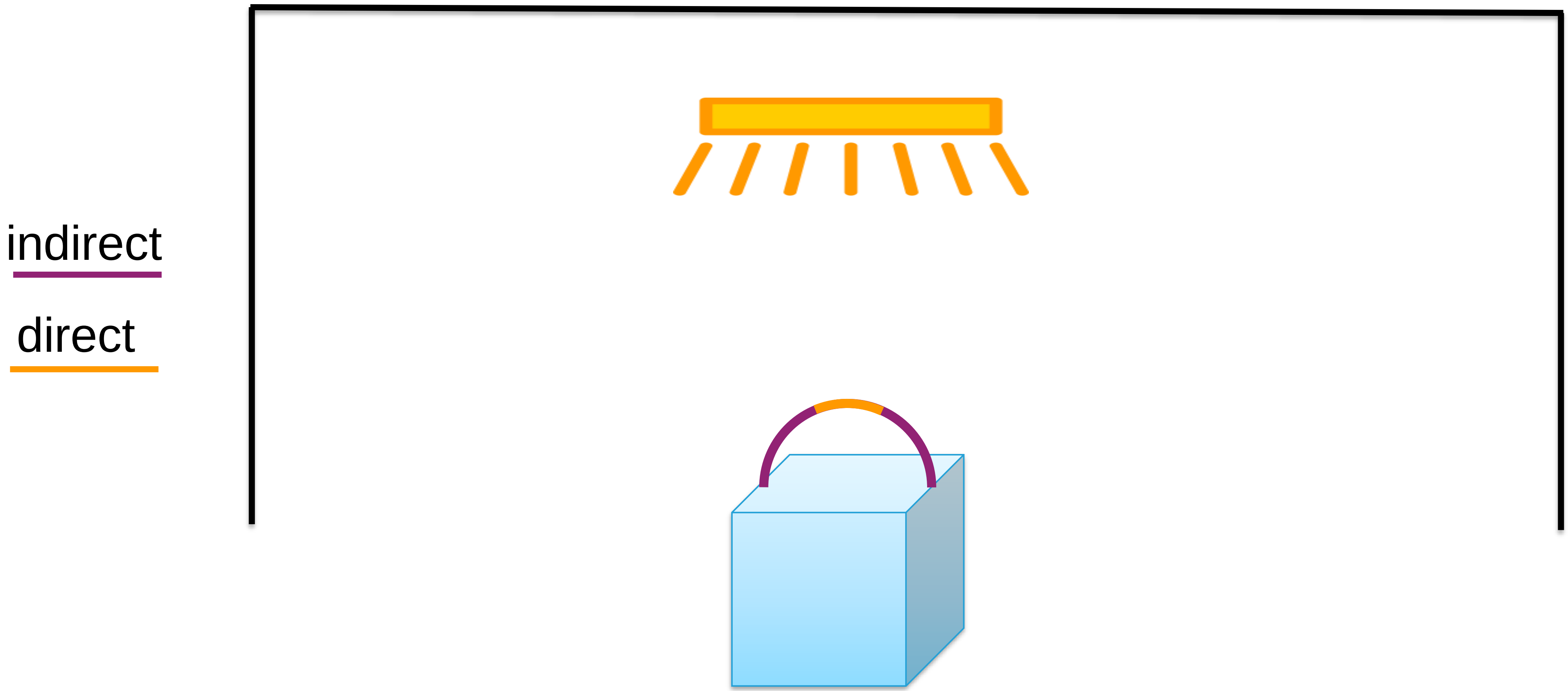
Next Event Estimation

- Builds on light source sampling. Think: where can light come from?



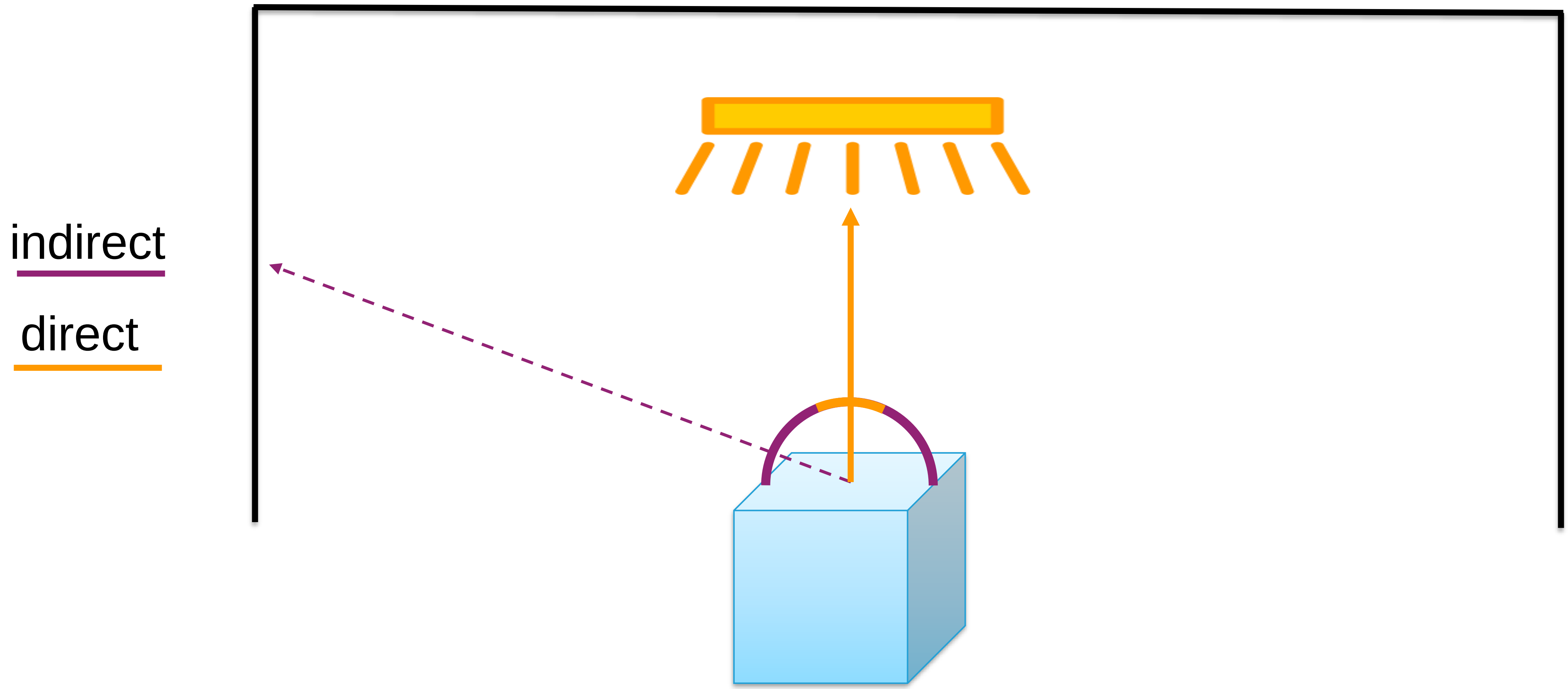
Next Event Estimation

- We can map out the full hemisphere and distinguish direct/indirect



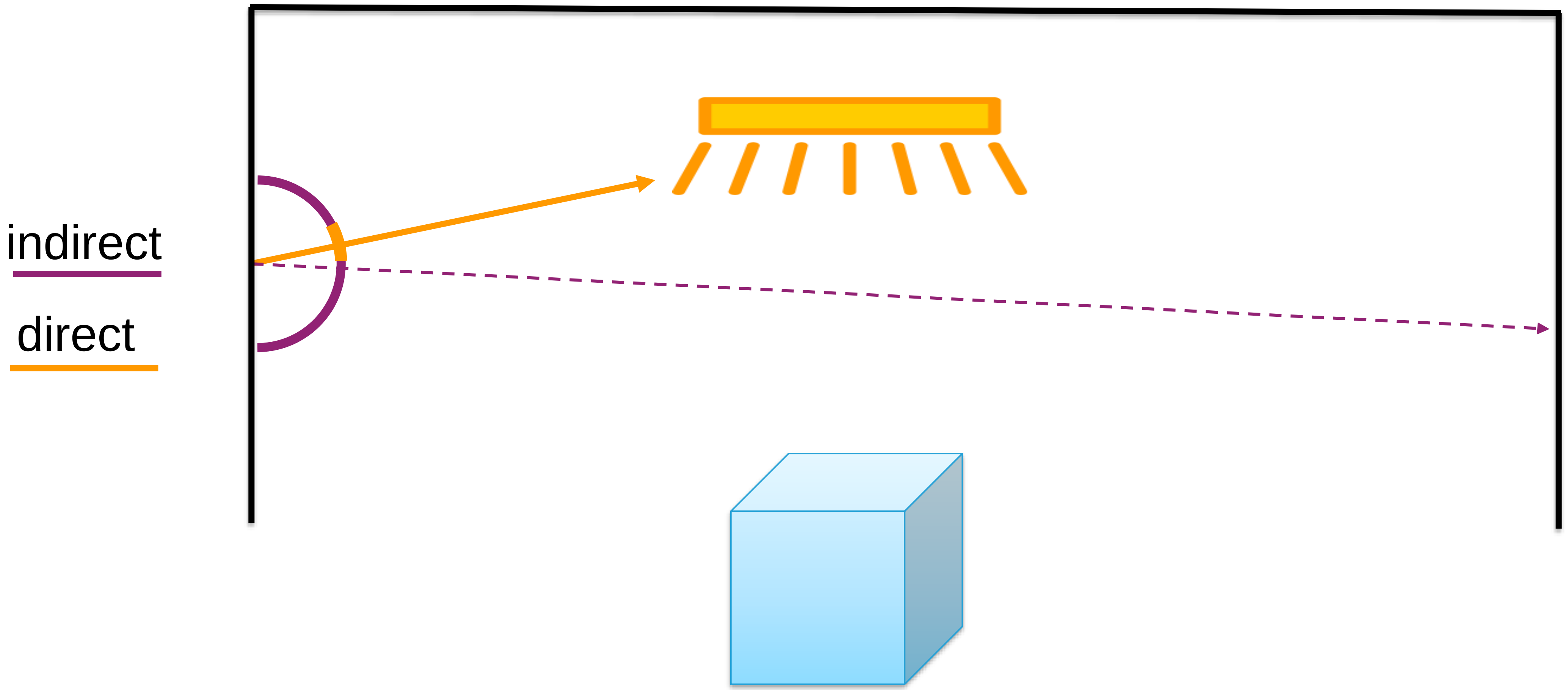
Next Event Estimation

- At each bounce, use light source sampling to get direct illumination
- Sample the BRDF to create direction for collecting indirect light



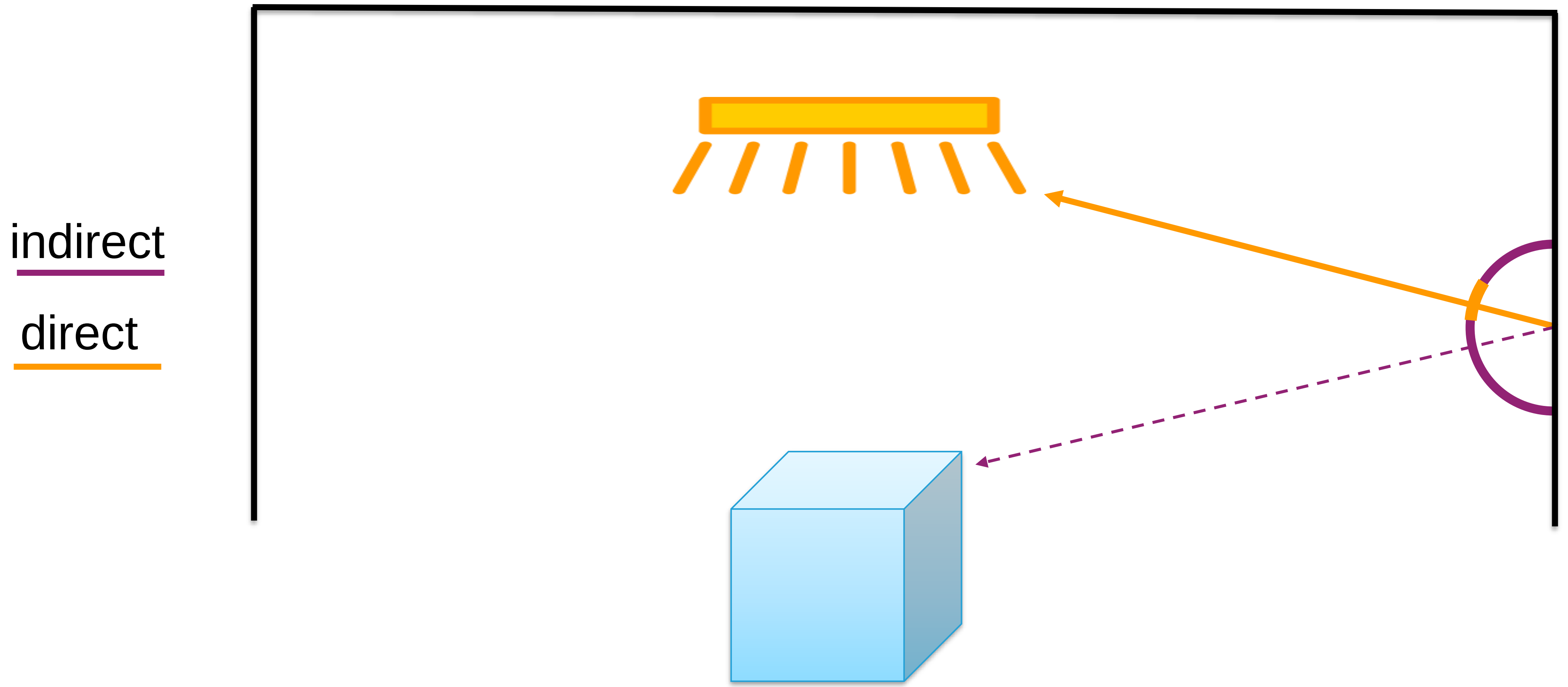
Next Event Estimation

- At each bounce, use light source sampling to get direct illumination
- Sample the BRDF to create direction for collecting indirect light



Next Event Estimation

- At each bounce, use light source sampling to get direct illumination
- Sample the BRDF to create direction for collecting indirect light

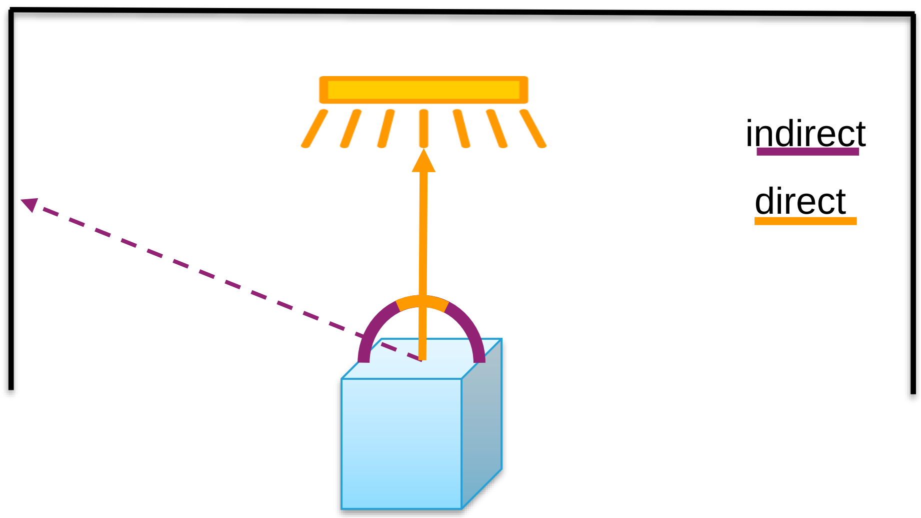


Divide and Conquer

- Light source sampling for direct light

  +

  BRDF sampling for indirect light



- Add them together to cover the hemisphere
  - Light source sampling to project light source onto hemisphere
  - Importance sampling of the hemisphere via the BRDF to generate next direction to collect potential indirect light from next hit point

First Attempt at Next Event Estimation

```
function Li(v_inv, D)
      …
      f = x.emit
      …
      // Apply Russian Roulette at some point
      …
      direct = <direct lighting with light source sampling>
      …
      indirect = <indirect light (recursive) with BRDF sampling>
      …
      f += (direct + indirect) / (rr_prob)
      return f
```

First Attempt at Next Event Estimation

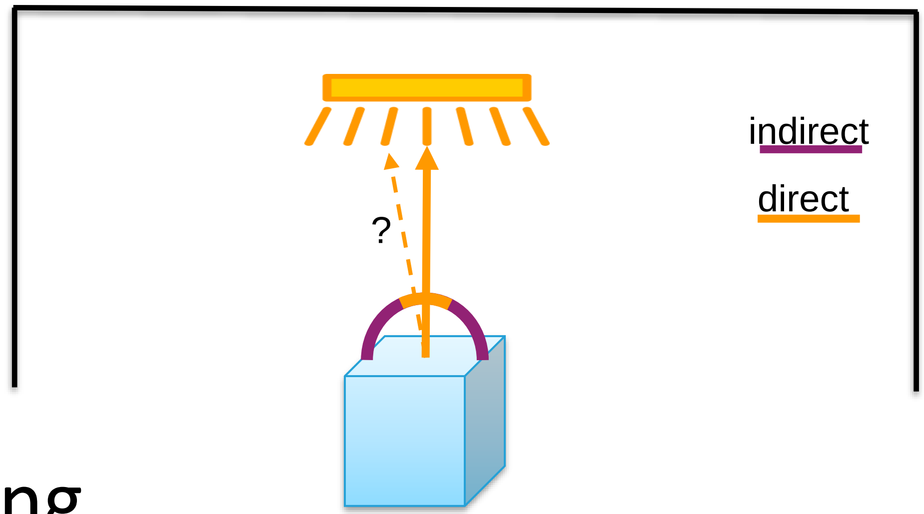
- Too bright! Better get some sunglasses to look at this…

BRDF Sampling

Current attempt

Revisiting our Approach

- Question: what happens if an indirect sample eventually hits a light?
- Indirect sample is accidentally direct,
  light is collected twice in same bounce!



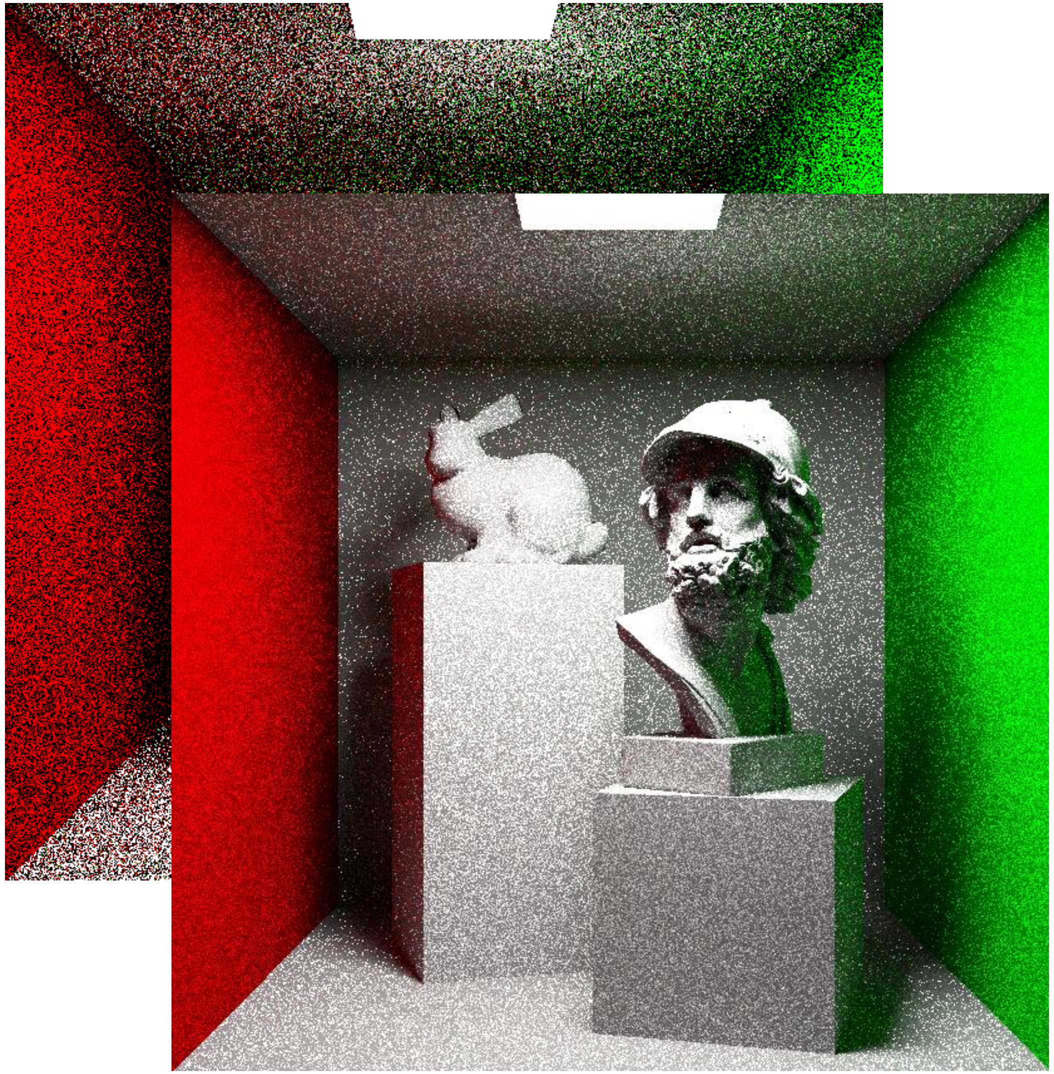
- Due to the compensation of BRDF sampling,
  we end up with double the amount of light we should have!
- Idea: if we have double the amount of light, can we just divide by 2?

Second Attempt at Next Event Estimation

```
function Li(v_inv, D)
        …
        f = x.emit
        …
        // Apply Russian Roulette at some point
        …
        direct = <direct lighting with light source sampling>
        …
        indirect = <indirect light (recursive) with BRDF sampling>
        …
        weight = (D == 0) ? 0.5 : 1 // halve indirect light after 1st bounce
        f += weight * (direct + indirect) / (rr_prob)
        return f
```

Averaging BRDF Sampling and Light Source Sampling

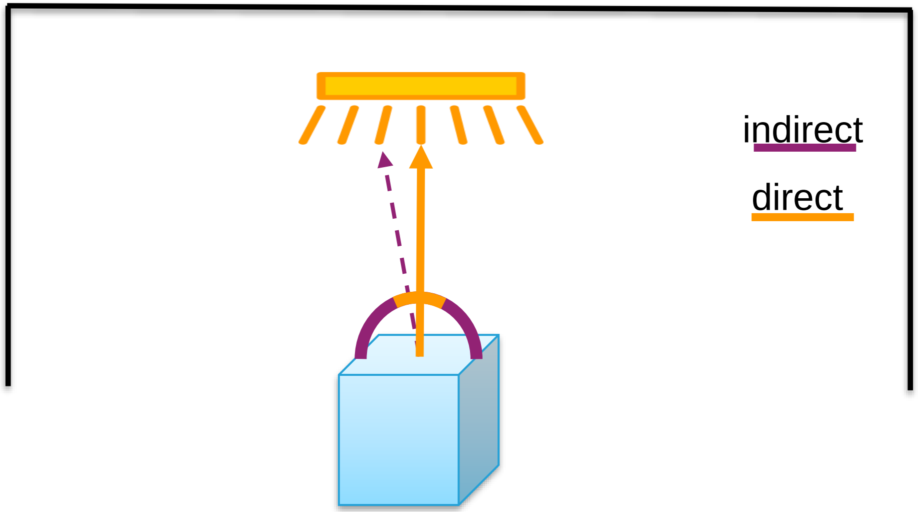
- The noise has significantly improved!
- Mixing several importance sampling techniques and weighting them...?
- It‘s multiple importance sampling!
- There are multiple ways to do MIS, let‘s quickly revisit some of them…



Multiple Importance Sampling Techniques Revisited

- Multiple MIS weightings to choose from:
  - Averaging: $w_i(x) = \frac{1}{N}$ for N techniques
  - 1 or 0, depending on each new sample
  - Balance heuristic (**Veach 1997**)

$$w_i(x) = \frac{p_i(x)}{\sum_{k=0}^{N} pk(x)}$$



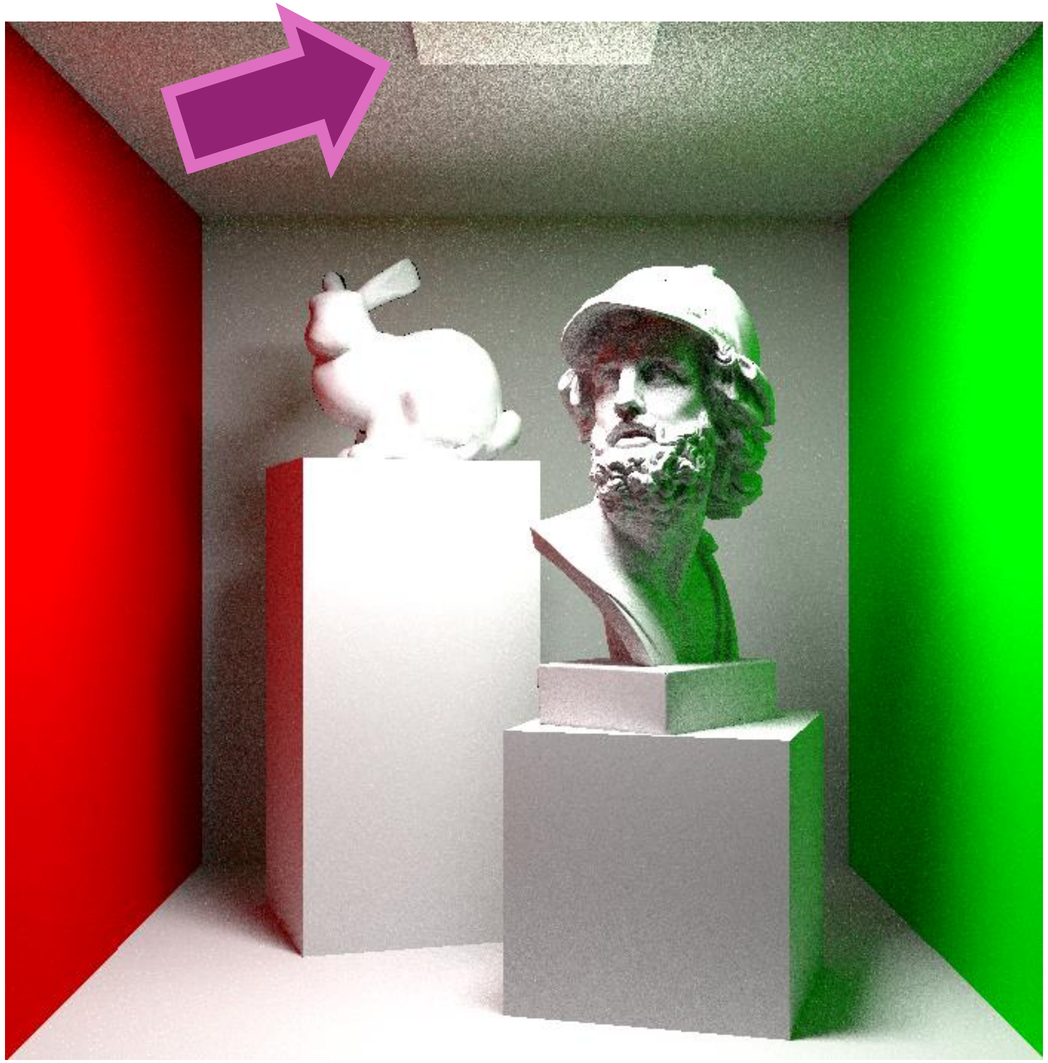
- Let's try something basic: assigning 0/1 weights to techniques
  - Assumption: light source sampling is better at **direct** light than BRDF
  - Keep BRDF sampling for indirect light, disable its direct light collection

Removing Surface Emittance from Path Tracing

```
function Li(v_inv, D)
    …
    // f = x.emit
    f = 0       // <-- 0 weight, removes direct light of BRDF sample at D-1!
    …
    // Apply Russian Roulette at some point
    …
    direct = <direct lighting with light source sampling>
    …
    indirect = <indirect light (recursive) with BRDF sampling> // unchanged
    …
    f += (direct + indirect) / (rr_prob) // 1 weight
    return f
```

Removing Surface Emittance from Path Tracing

- Looks better than averaging!
- But some information lost: light sources themselves don't seem to emit any light anymore…
- Logical, we removed emittance!
- It seems eliminating emittance altogether was too much…



Re-Enabling Emittance for Special Paths

- At the first bounce, there was no previous bounce for which we could compute the direct lighting with light source sampling

- I.e., we did not perform “next event estimation” at the 0th hit point, the camera (or viewpoint) itself

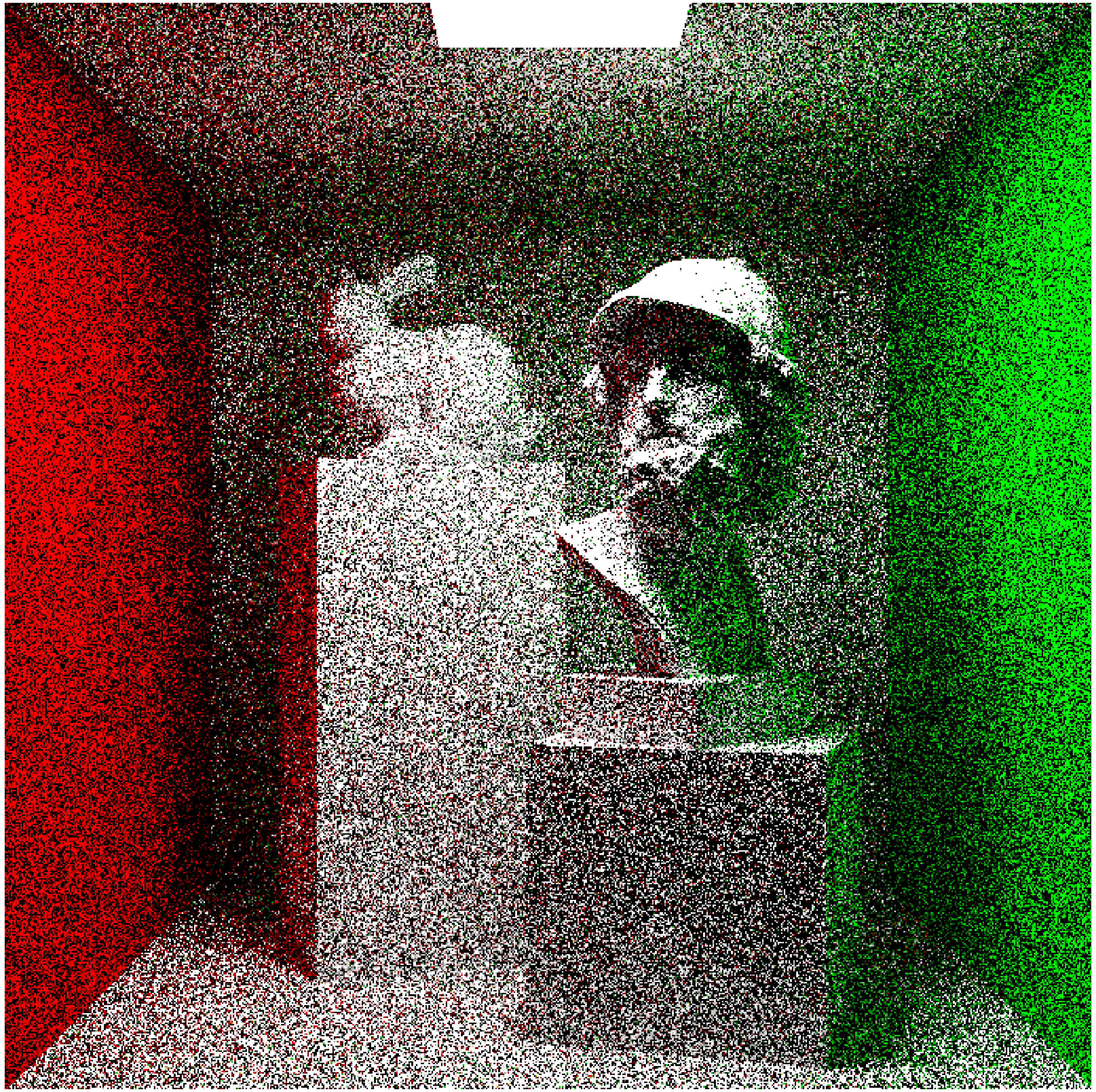
- Simple fix: actually, ignore emittance **most of the time**, except if the current hit point is the first hit after leaving the camera / eye

- Can use recursion depth to enable or disable emittance term

Re-Enabling Emittance for Special Paths

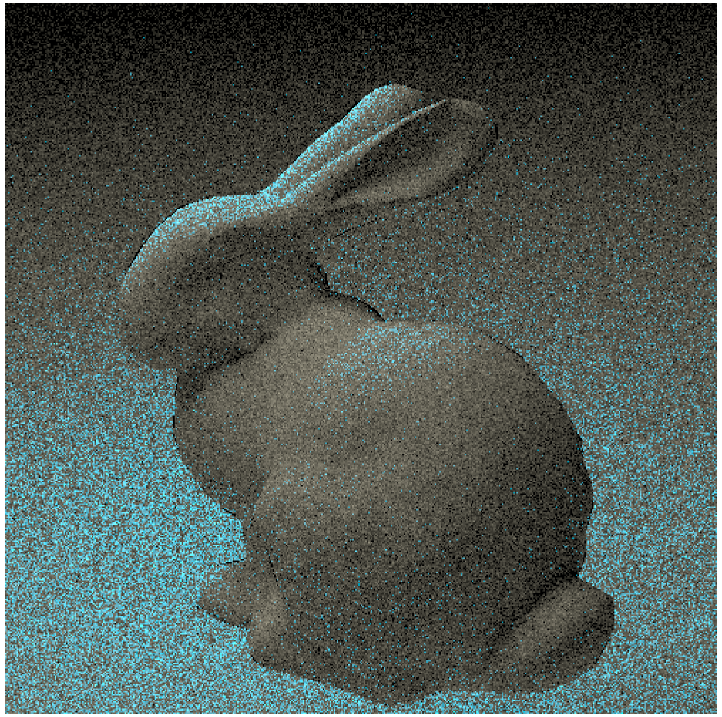
```
function Li(v_inv, D)
      …
      f = 0    // 0 weight, most of the time, except at first intersection
      if (D == 0)
            f = x.emit
      …
      // Apply Russian Roulette at some point
      …
      direct = <direct lighting with light source sampling>
      …
      indirect = <indirect light (recursive) with BRDF sampling>
      …
      f += (direct + indirect) / (rr_prob)
      return f
```

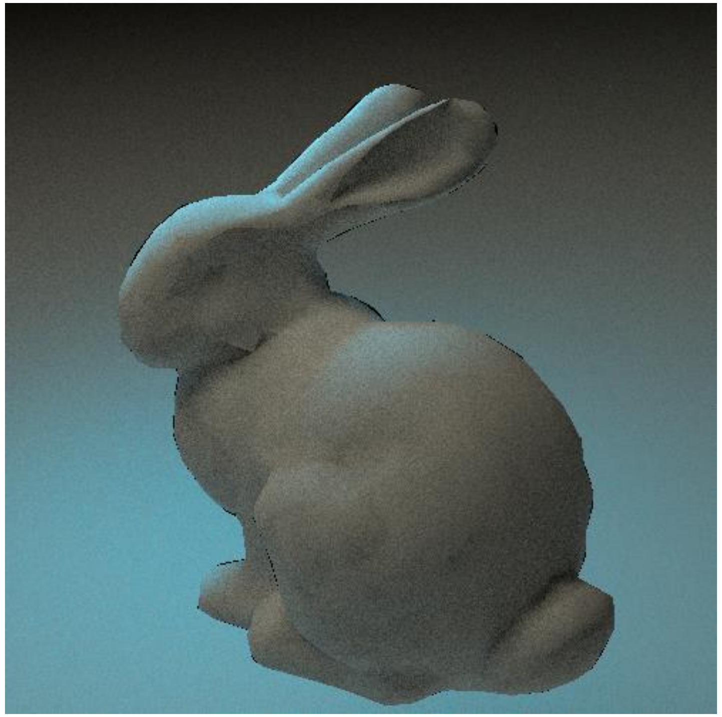
Diffuse Next Event Estimation – Unlocked!



More Diffuse Next Event Estimation Results

- BRDF importance sampling vs. next event estimation
  - In many cases, significant improvement of quality
  - Same number of samples, very similar runtime (NEE slightly slower)





More Diffuse Next Event Estimation Results

More Diffuse Next Event Estimation Results

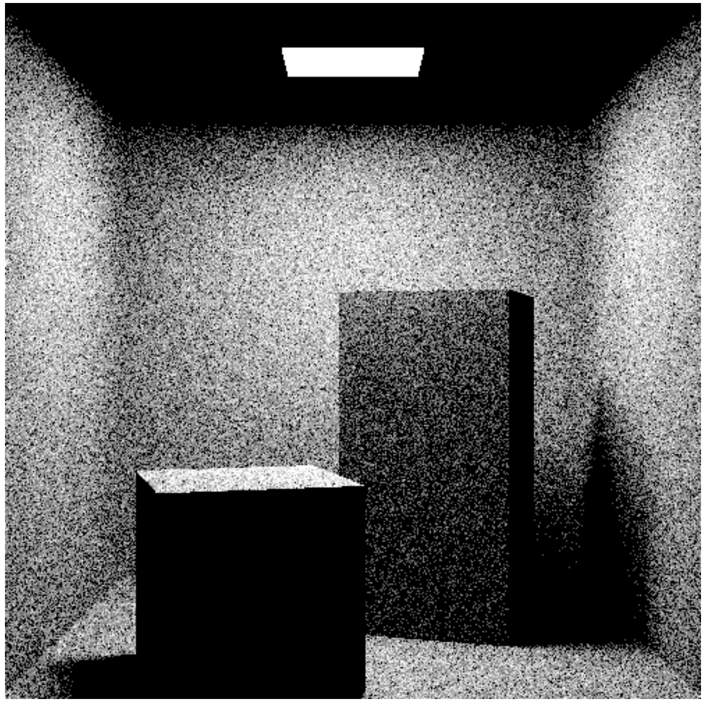
- Next event estimation is highly effective, but still no silver bullet

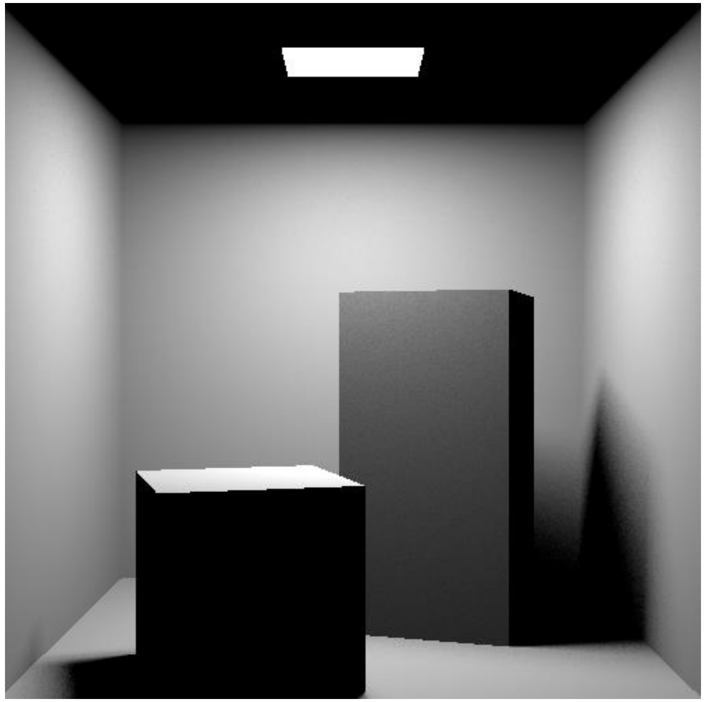
- Always a more challenging scene to push your renderer to its limit…

But is Using 0/1 Weights Really the Best Choice?

- Effectively, we now use light source sampling for all direct light
  - Using light source over BRDF sampling often improves direct lighting



White box, BRDF sampling



White box, light source sampling

  - But, as usual, the benefit very much depends on the input scene

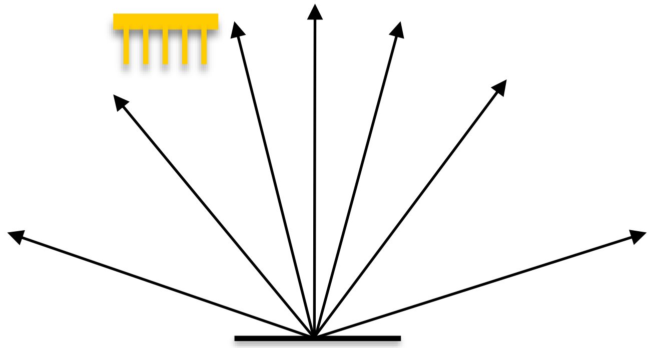
Direct Light: Light Source Sampling is Not Always Better…

- Scene with small and large area light: BRDF/light source sampling

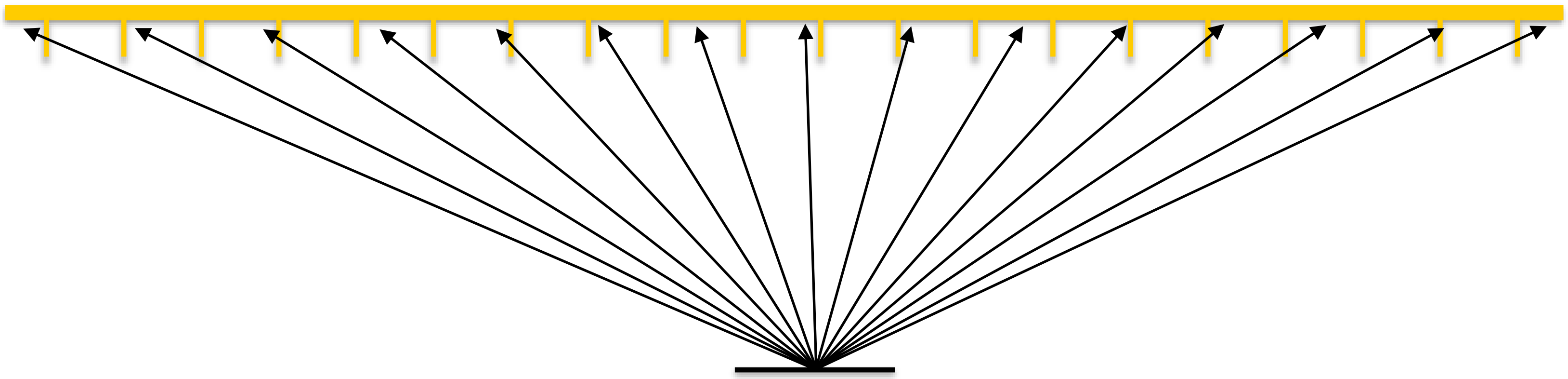
BRDF sampling: large grey light works well, small blue light causes a lot of noise

Light source sampling: small blue light works well, large grey light causes a lot of noise

How Sampling Methods Can Fail (Diffuse Materials)



Importance sampling the BRDF, we may miss the smaller light sources



Light source sampling a large, overhead light source (sky) concentrates more samples at flat angles – the opposite of importance sampling!

- Light source sampling can struggle even more with other BRDFs!
- But remember, there is another MIS option: the balance heuristic

Direct Lighting with the Balance Heuristic

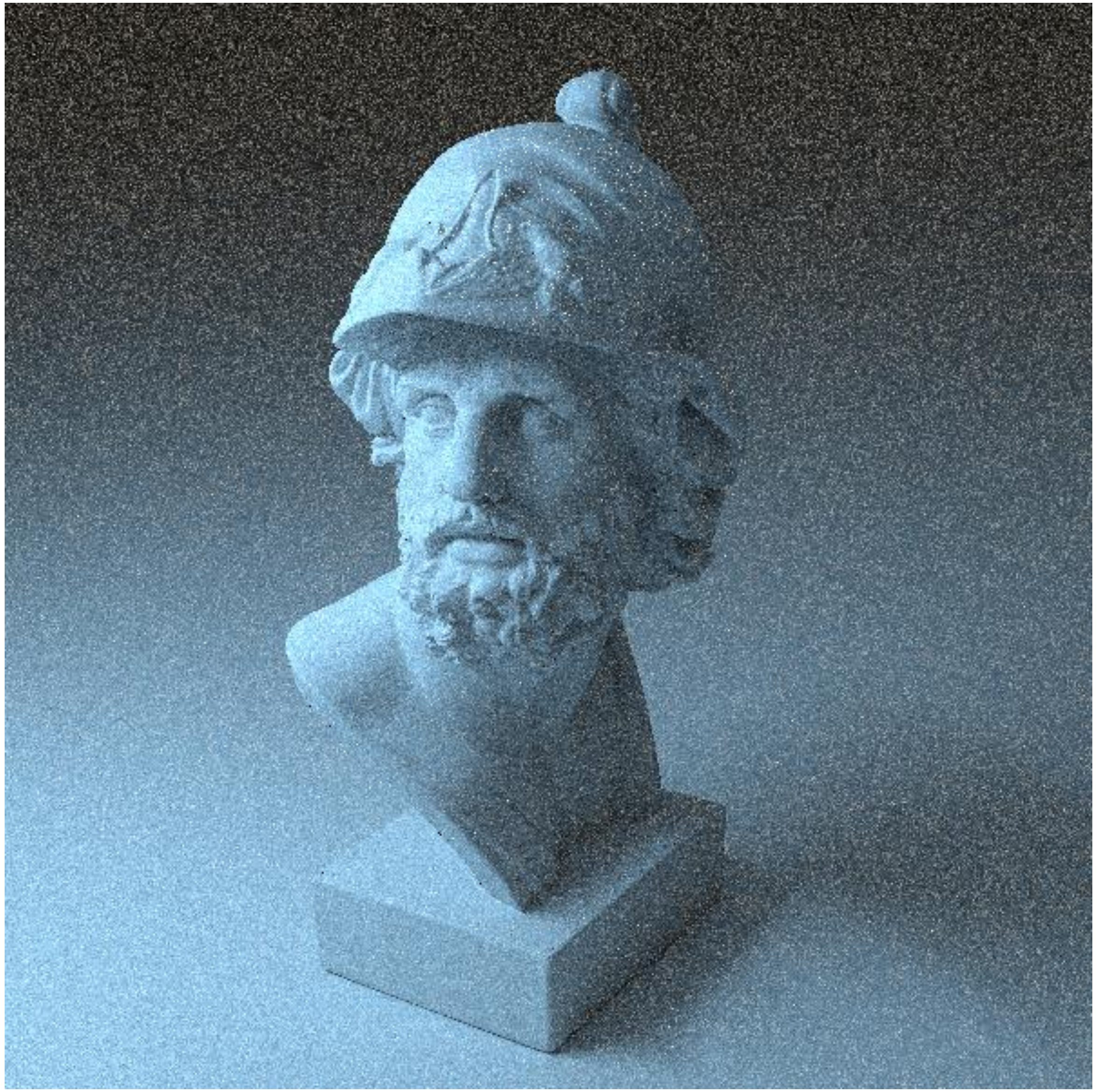
- Implementation in direct lighting integrator is not too complicated
  - Randomly choose technique and weight result using balance heuristic

BRDF sampling | Light source sampling | MIS, balance heuristic

Efficient Path Tracing (NEE) with the Balance Heuristic

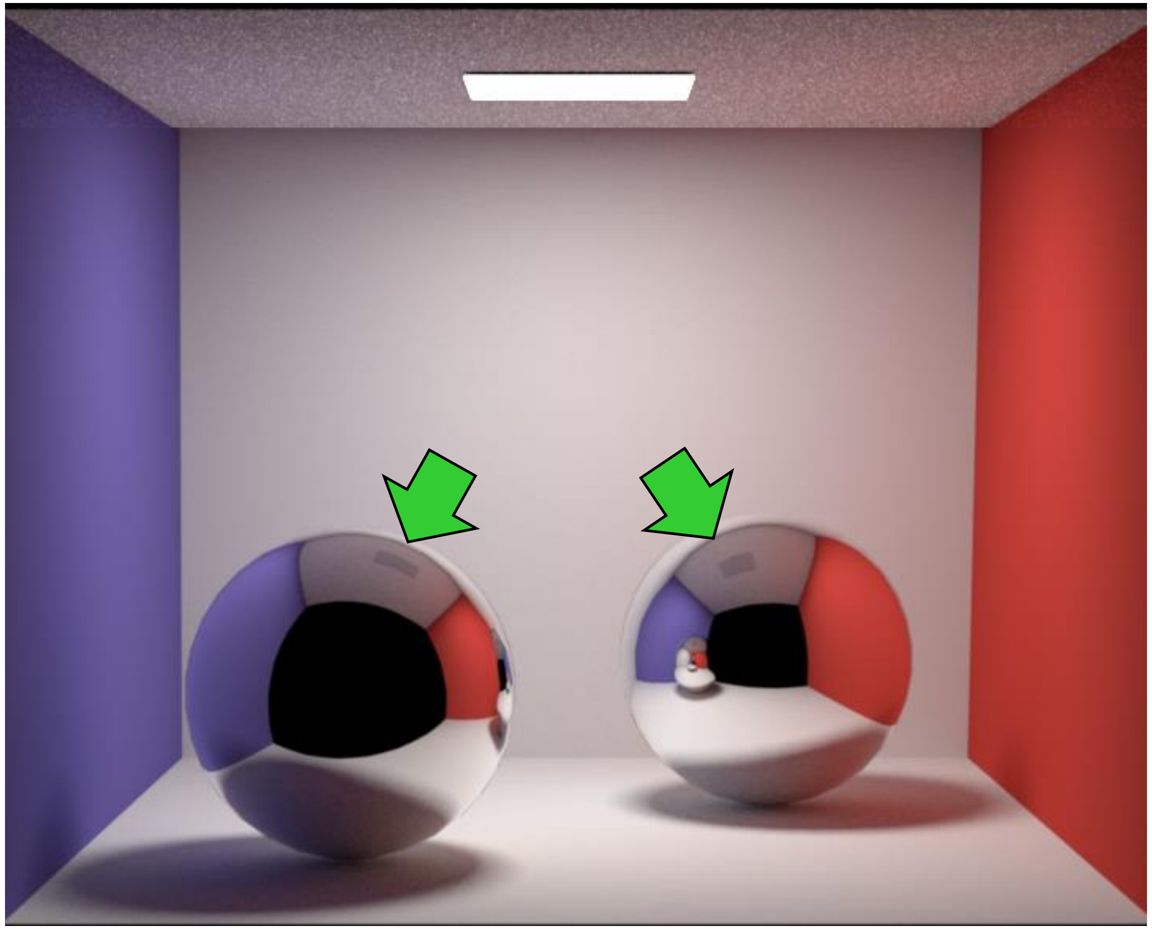
- Consider both light sampling methods in every bounce
- Weight on first intersection must still be 1 for BRDF sampling
- Compute probabilities for selected sample with both techniques, use the balance heuristic to compute adequate MIS weights
- Tricky: different sampling methods are evaluated at different times
  - BRDF sample contribution is found when hitting an emitting surface
  - Contribution of light source sampling is calculated one bounce prior

Path Tracing (NEE) with 0/1 Weights vs Balance Heuristic



Next Event Estimation meets Mirror Materials

- Soon, we will add some more exciting BRDFs for materials!
- Fully specular mirror materials are easy to simulate, however, they need extra care with NEE
- Naïve reflections can miss light!



- Why? Join us in the upcoming Materials lecture to find out…

References

- [1] Conquering noisy images in ray tracing with next event estimation: https://developer.nvidia.com/blog/conquering-noisy-images-in-ray-tracing-with-next-event-estimation/
- [2] Dittebrandt, A., Hanika, J., & Dachsbacher, C. (2020). Temporal Sample Reuse for Next Event Estimation and Path Guiding for Real-Time Path Tracing. In Eurographics Symposium on Rendering - DL-only Track. The Eurographics Association.
- [3] Guo, J., Eisemann, M., & Eisemann, E. (2020). Next Event Estimation++: Visibility Mapping for Efficient Light Transport Simulation. Computer Graphics Forum, 39(7), 205-217.
- [4] If you didn't like our explanations of next event estimation, perhaps Jacco Bikker can do a better job: http://www.cs.uu.nl/docs/vakken/magr/2015-2016/slides/lecture%2008%20-%20variance%20reduction.pdf