Adjoint-Driven Russian Roulette and Splitting in Light Transport Simulation

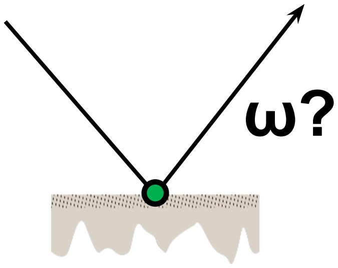
J. Vorba, J. Krivanek [SIGGRAPH 2016]

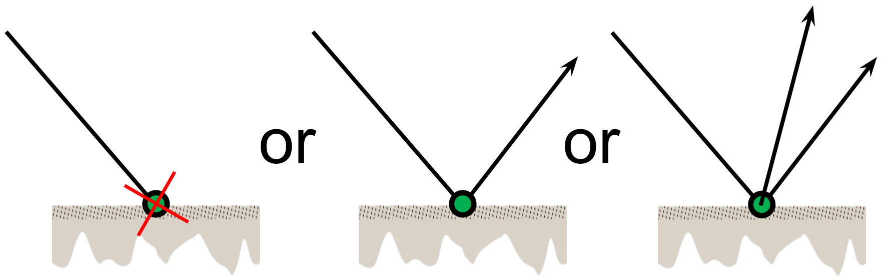
Presenter: Eunhyouk Shin

Last presentation:
Interactive Sound Propagation with Bidirectional Path Tracing

- Acoustic transport equation
- Solution in Neumann series form
- Estimate terms using BDPT & multiple importance sampling
- Caching for interactive performance
- SNR metric for evaluating stochastic sound propagation
- Compared to visual rendering, temporal dimension quality is important rather than spatial resolution

Recall: Probabilistic Aspects of MCRT





Direction sampling

Russian roulette / splitting

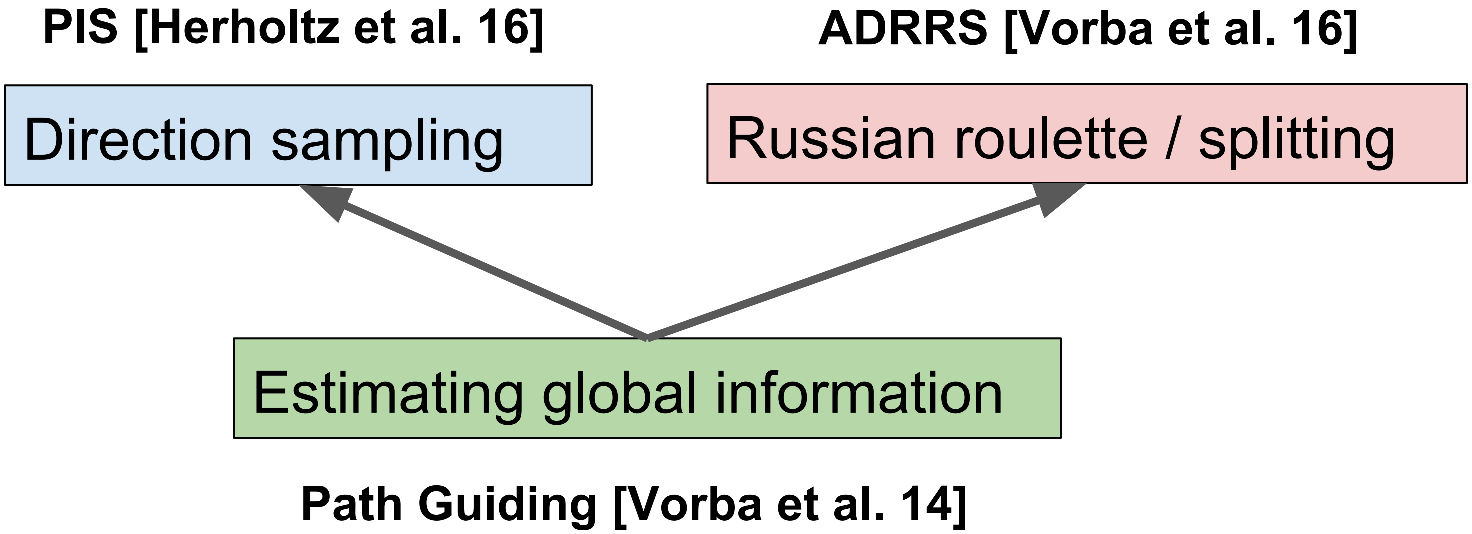
Recall: Importance Sampling

To reduce variance (noise),

we must **make more path that contributes more**.

But how do we know beforehand how much it will contribute?

Recent Techniques using Global Estimation

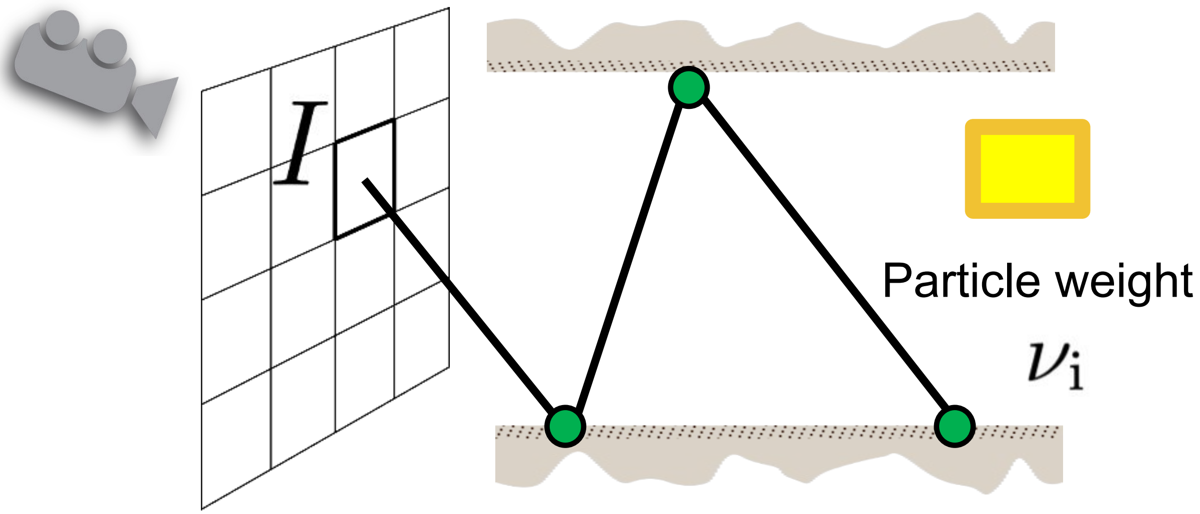


Contents

***Figures in the slides is from the authors [Vorba et al. 16]**

Particle Tracing

Particle Tracing

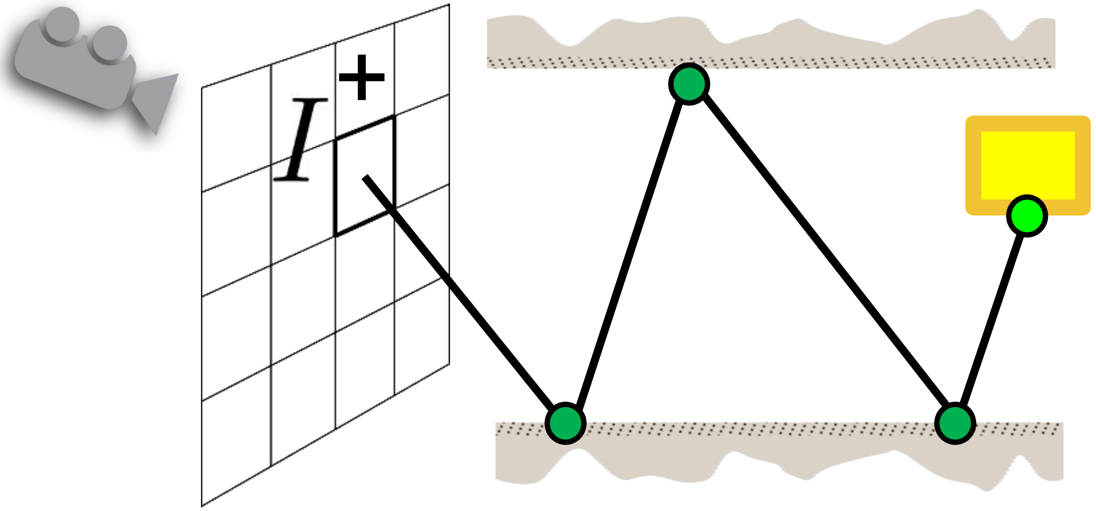


- Trace weighted particles to get an unbiased estimate of the pixel value

Particle Tracing

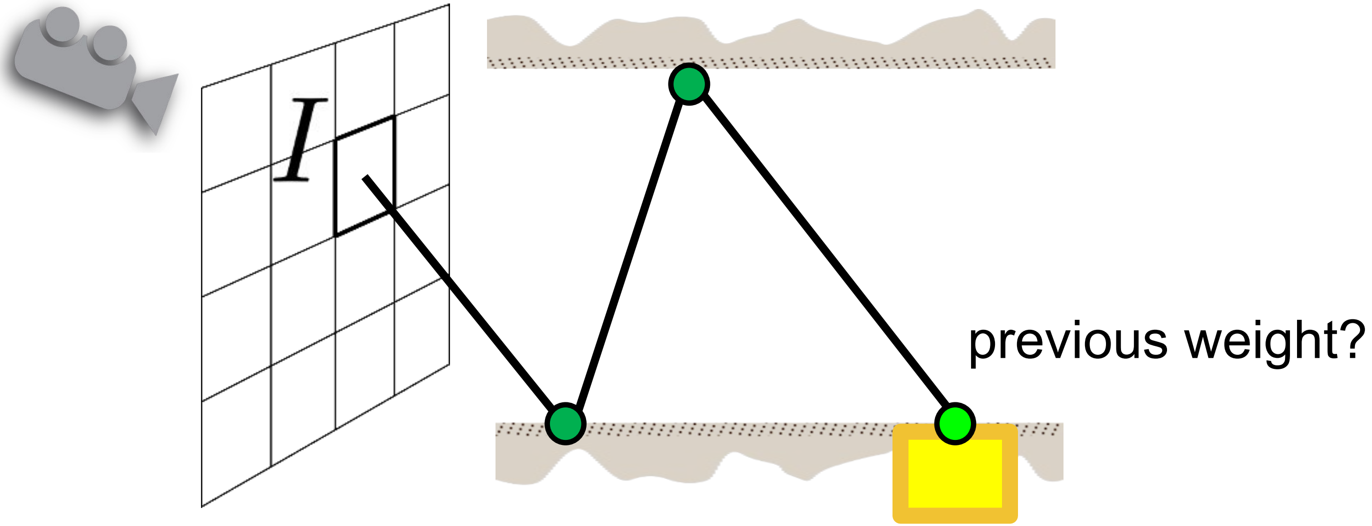
Weight Emission

$$+\ \nu_{\mathrm{i}}(\mathbf{y}_k, \omega_k)\, L_{\mathrm{o}}^{\mathrm{e}}(\mathbf{y}_k, \omega_k)$$



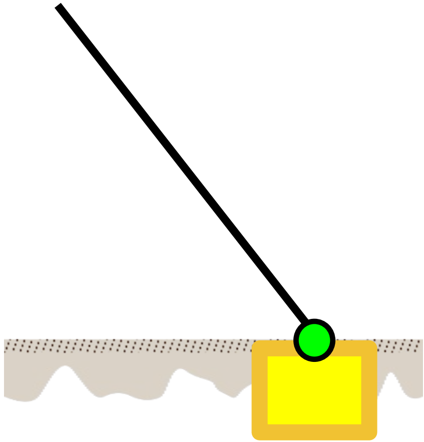
- When a particle collides to a light source, it contributes to the estimation according to the weight.

Particle Tracing



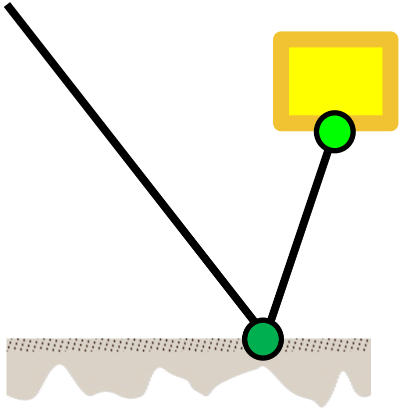
- Contribution must be different for before/after a bounce

Weight Update: Bounce



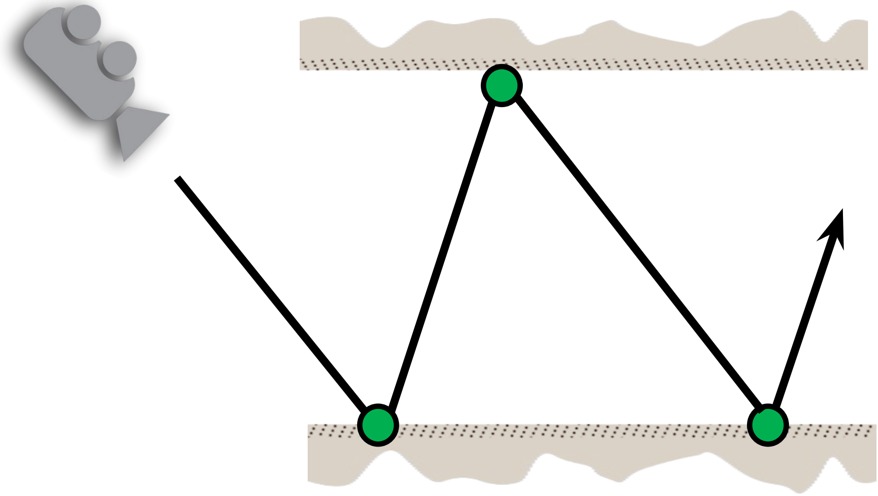
particle weight update

$$\times \; \frac{f_s(\mathbf{y}, \omega_o \to \omega_i)\,|\cos\theta_o|}{p(\omega_o \,|\, \mathbf{y})}$$



- Incremental weight update by BRDF, geometric term, sampling distribution

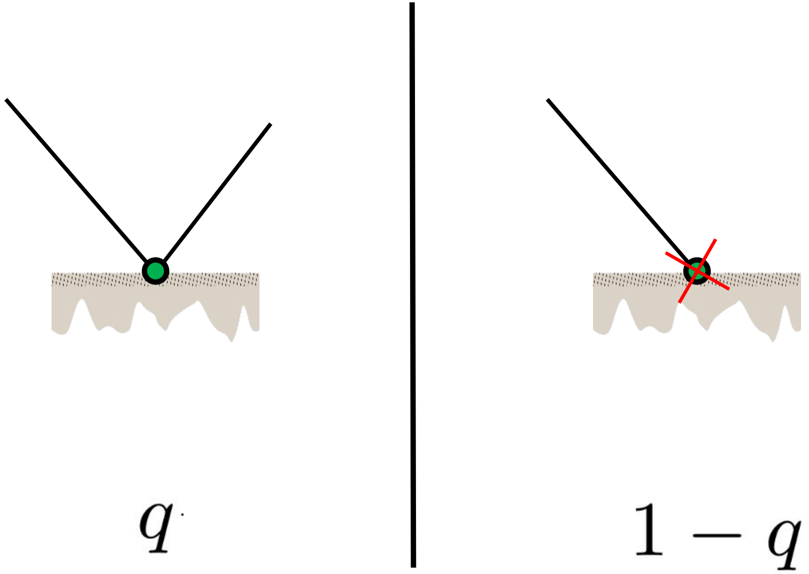
Particle Tracing



We have an unbiased estimator,
But when do we terminate?

Russian Roulette & Splitting

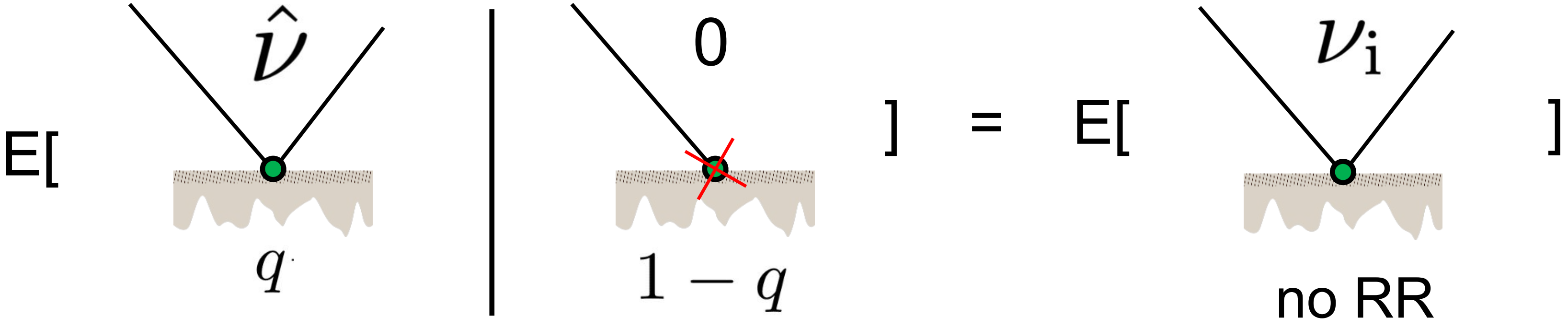
Russian Roulette



- Upon collision, the particle survives with probability q

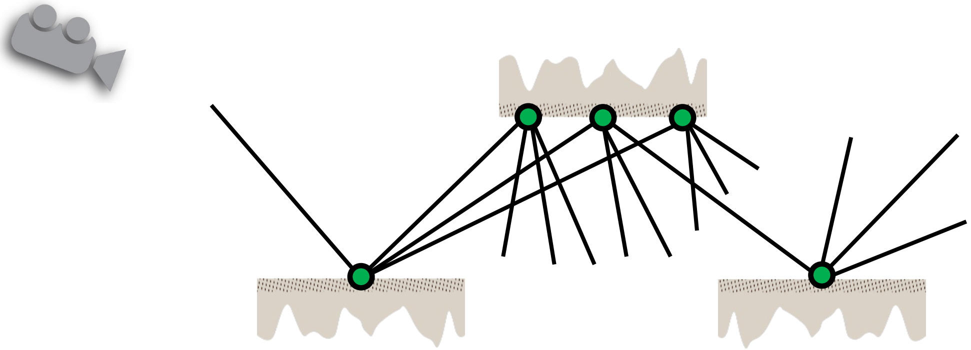
Weight Update: Russian Roulette

- **Key: Expected contribution must remain unchanged**



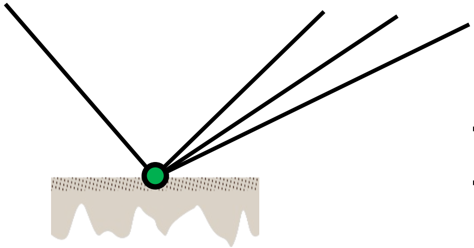
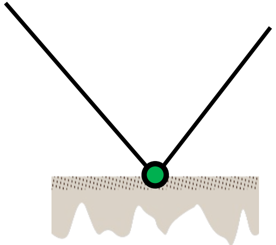
$$\hat{\nu} = \frac{\nu_{\mathrm{i}}}{q}$$: Weight of survived particle gets ‘boosted’

Splitting



Weight Update: Splitting

- **Key: Expected contribution must remain unchanged**

E[] = E[]

split to **q** particles

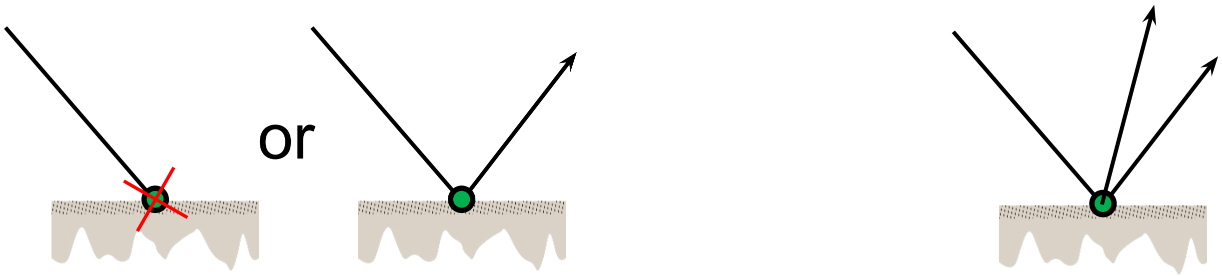
no splitting

$$\hat{\nu} = \frac{\nu_{\mathrm{i}}}{q}$$

: Splitted particles share the weight evenly

Unified RR/Splitting

- RR and splitting has same weight update formula: $\hat{\nu} = \frac{\nu_{\mathrm{i}}}{q}$



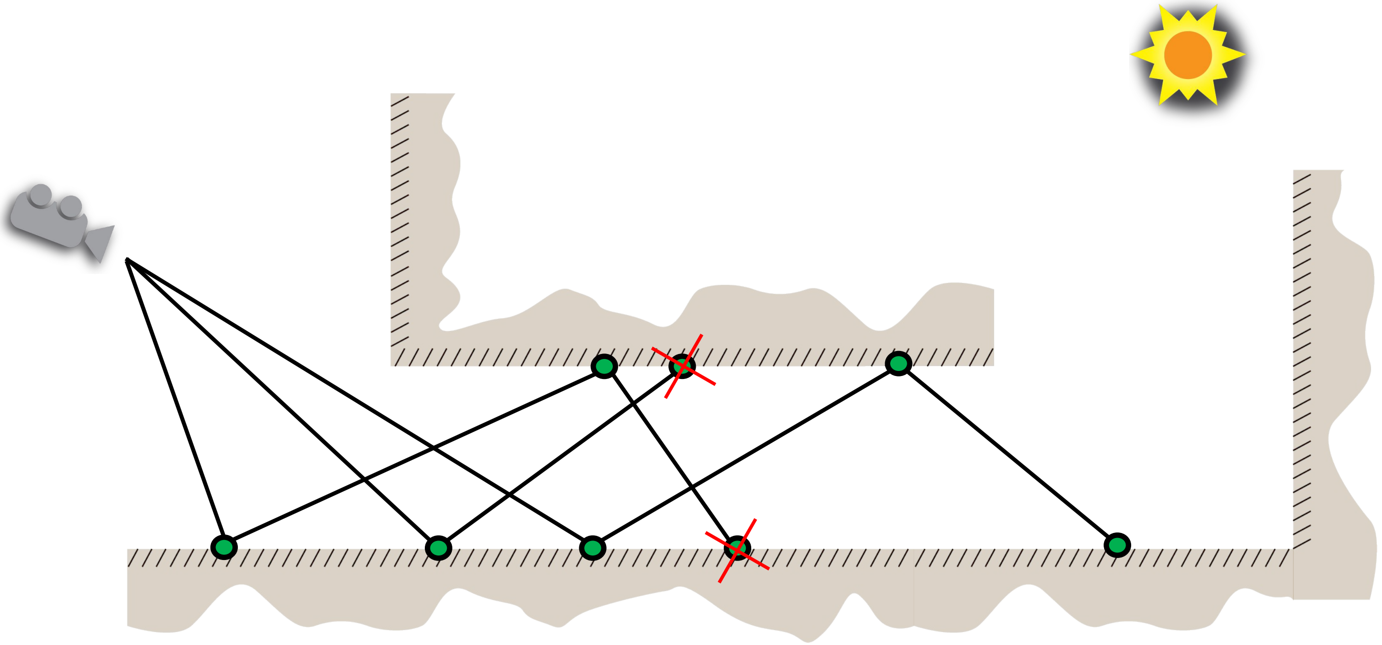
RR: $q < 1$

Split: $q > 1$

- Make q particles in expectation sense
    - Extends to non-integer splitting

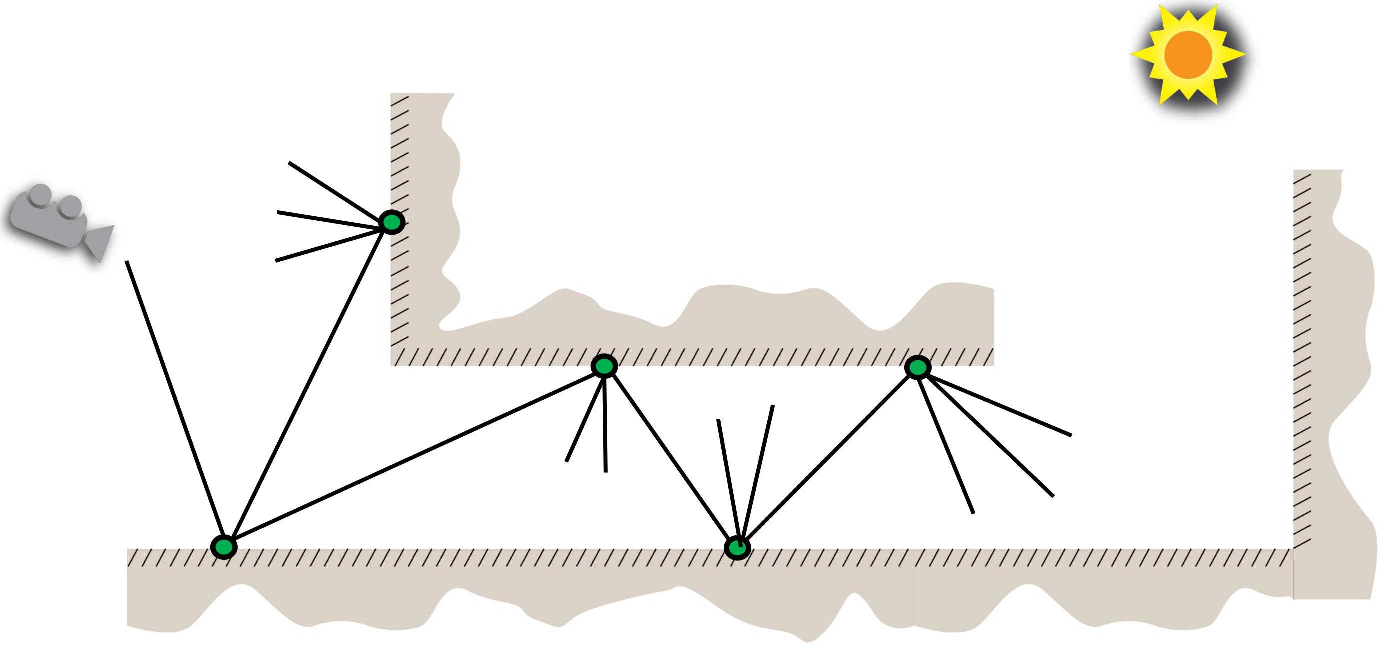
Determining the RR/Splitting factor

Issues



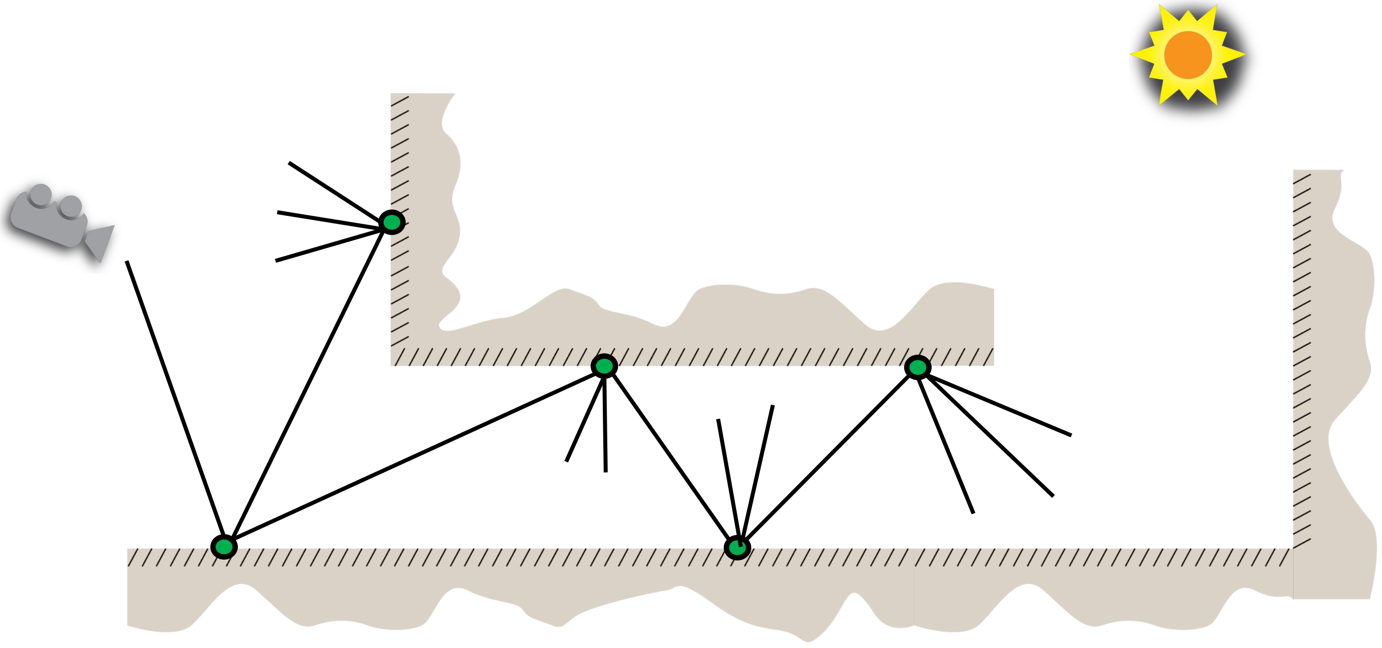
- If q is too low, only a small fraction of the particles reach the light source

Issues



- If q is too high, we are wasting too much resource for particles with low contribution

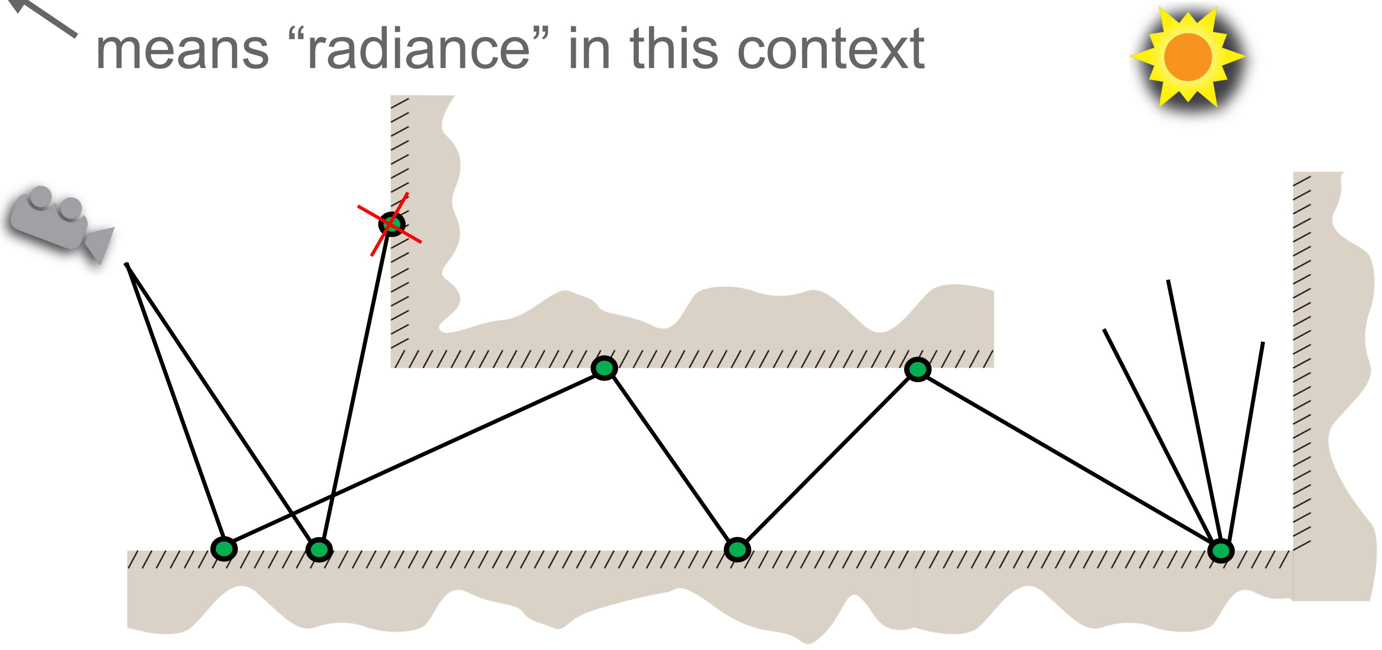
Issues



- Existing approaches set q according to the local properties (e.g. reflectance) of the surface, but it's still far from optimal

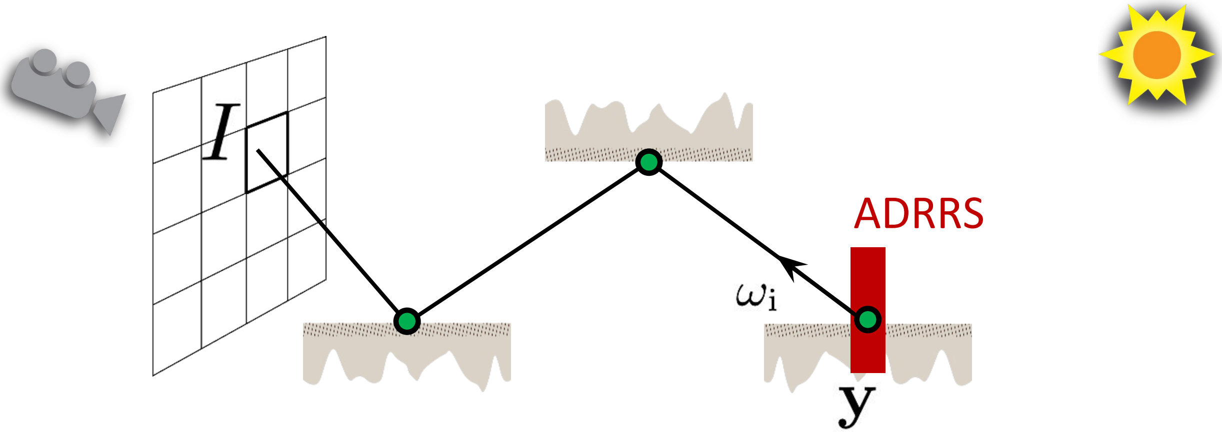
Adjoint Driven Russian Roulette and Splitting

means "radiance" in this context



- New approach uses global information of radiance

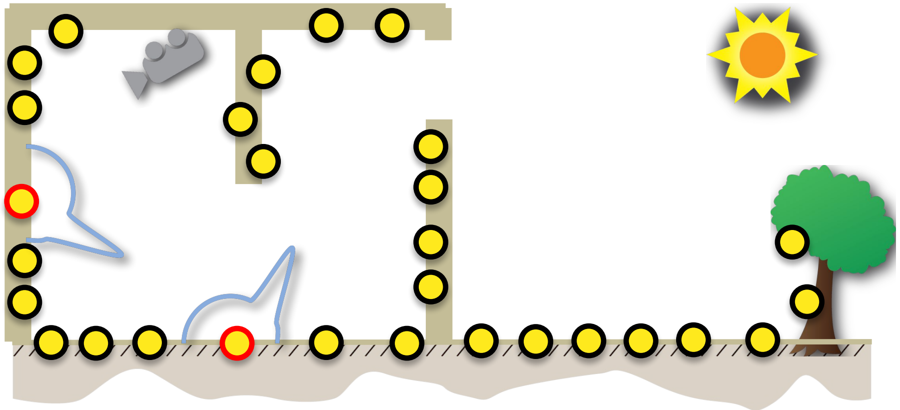
Adjoint Driven Russian Roulette and Splitting

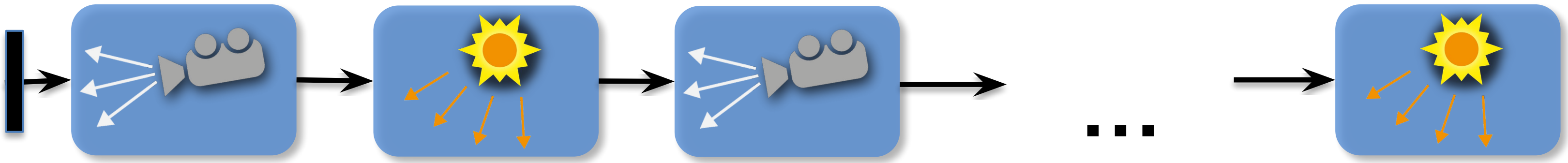


$$q(\mathbf{y}, \omega_\mathrm{i}) = \frac{\nu(\mathbf{y}, \omega_\mathrm{i}) L_\mathrm{o}^\mathrm{r}(\mathbf{y}, \omega_\mathrm{i})}{I}$$

- Set q as expected contribution of outgoing particles
- How to estimate pixel value and reflected radiance?

Path Guiding [Vorba et al. 2014]



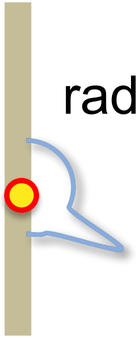


- Precomputed cache of estimated radiance distribution

Estimated Radiance

Summary: Determining the RR/splitting Factor

$$q(\mathbf{y}, \omega_\mathrm{i}) = \frac{\nu(\mathbf{y}, \omega_\mathrm{i}) L_\mathrm{o}^\mathrm{r}(\mathbf{y}, \omega_\mathrm{i})}{I}$$



- Want lower variance in rendered image
- Should make more particles for high-contribution paths
- We query the precomputed radiance cache to compute the adjoint-driven RR/splitting factor

Weight Window

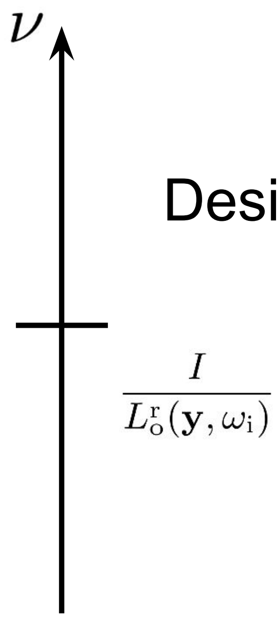
Weight Invariance

RR/Splitting factor

$$q(\mathbf{y}, \omega_\mathrm{i}) = \frac{\nu(\mathbf{y}, \omega_\mathrm{i}) L_\mathrm{o}^\mathrm{r}(\mathbf{y}, \omega_\mathrm{i})}{I}$$

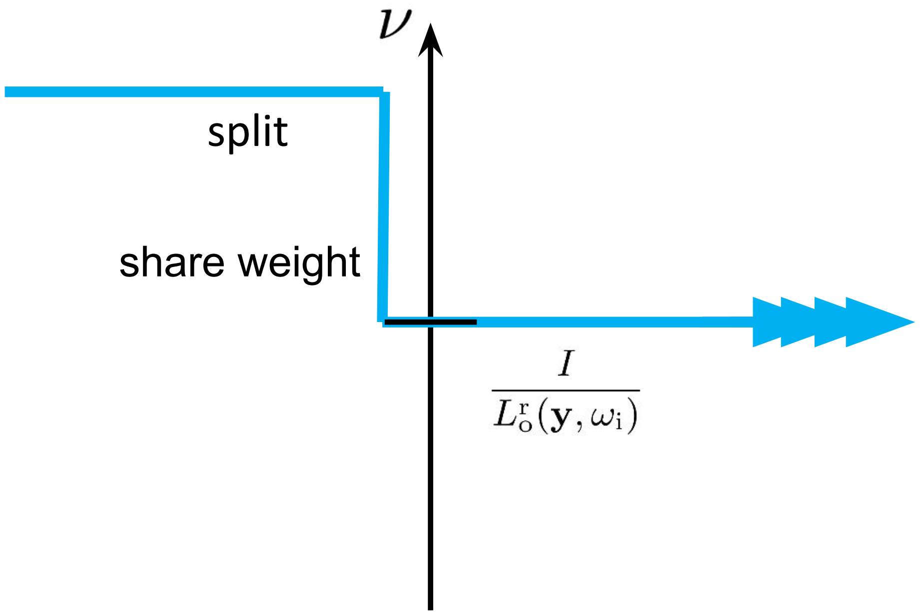
Weight after RR/split

$$\hat{\nu}(\mathbf{y}, \omega_\mathrm{i}) = \frac{\nu(\mathbf{y}, \omega_\mathrm{i})}{q(\mathbf{y}, \omega_\mathrm{i})} = \boxed{\frac{I}{L_\mathrm{o}^\mathrm{r}(\mathbf{y}, \omega_\mathrm{i})}}$$

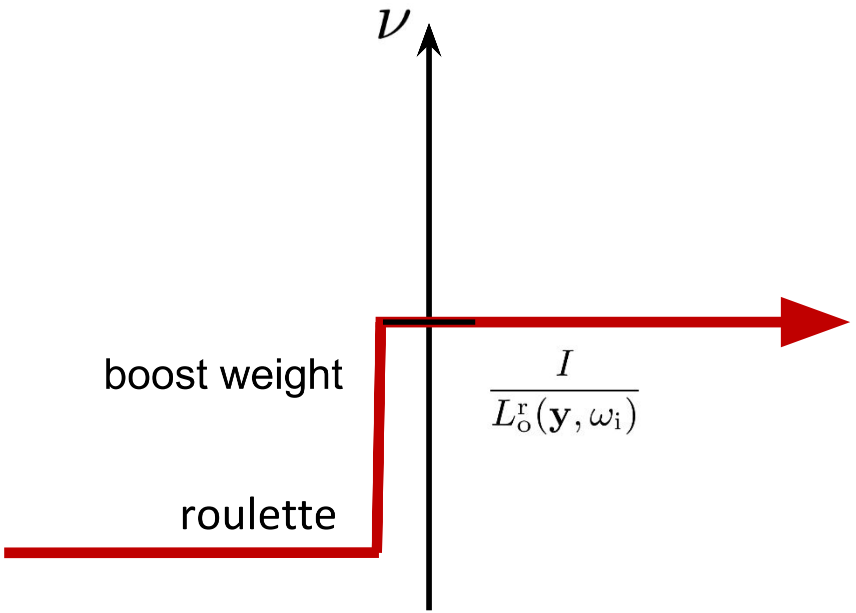


Desired weight after RR/splitting

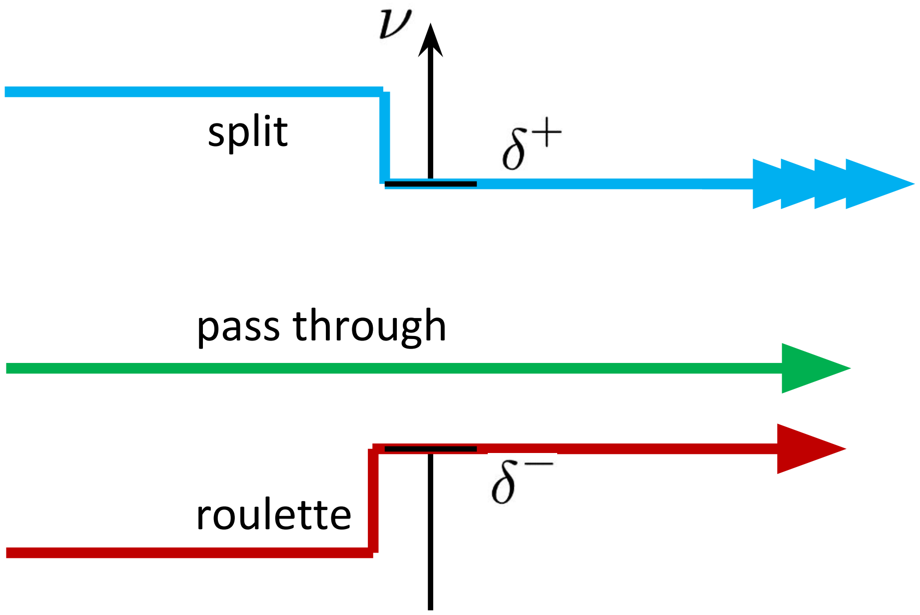
Weight Invariance



Weight Invariance

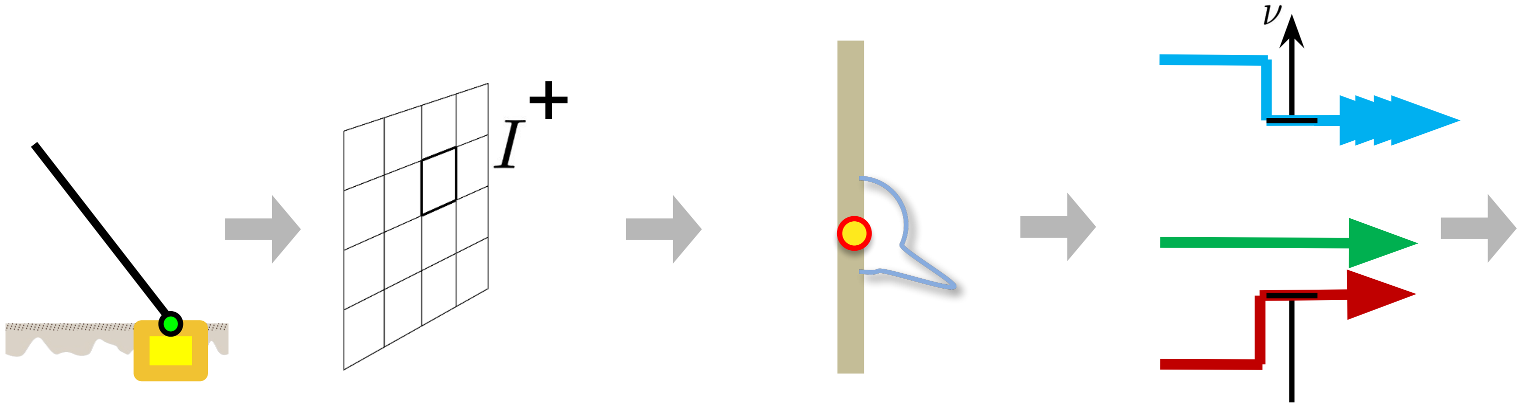


Weight Window



- More robust by loosening the RR/splitting

ADRRS Summary



make 0+ new particles

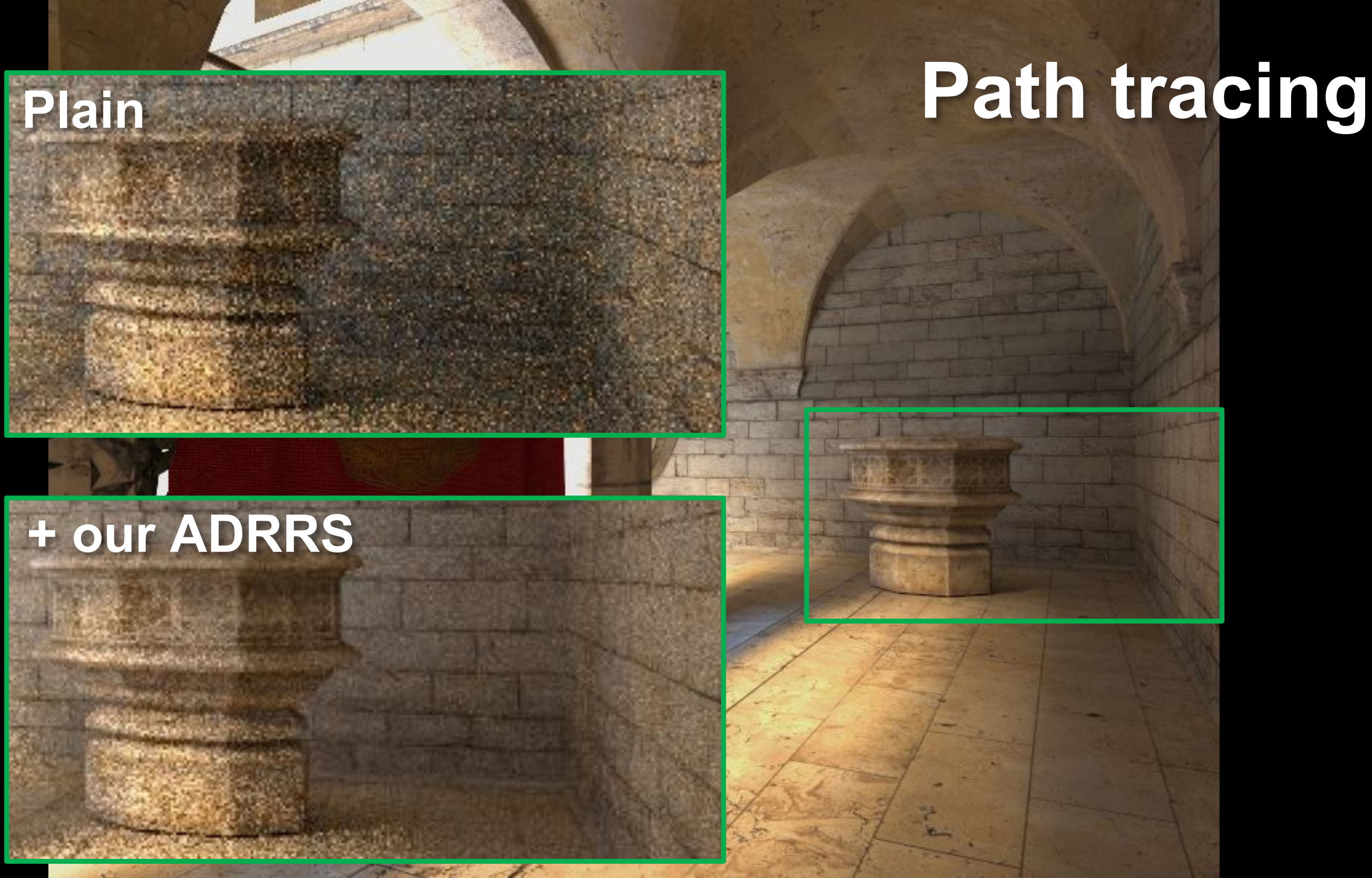
Collision

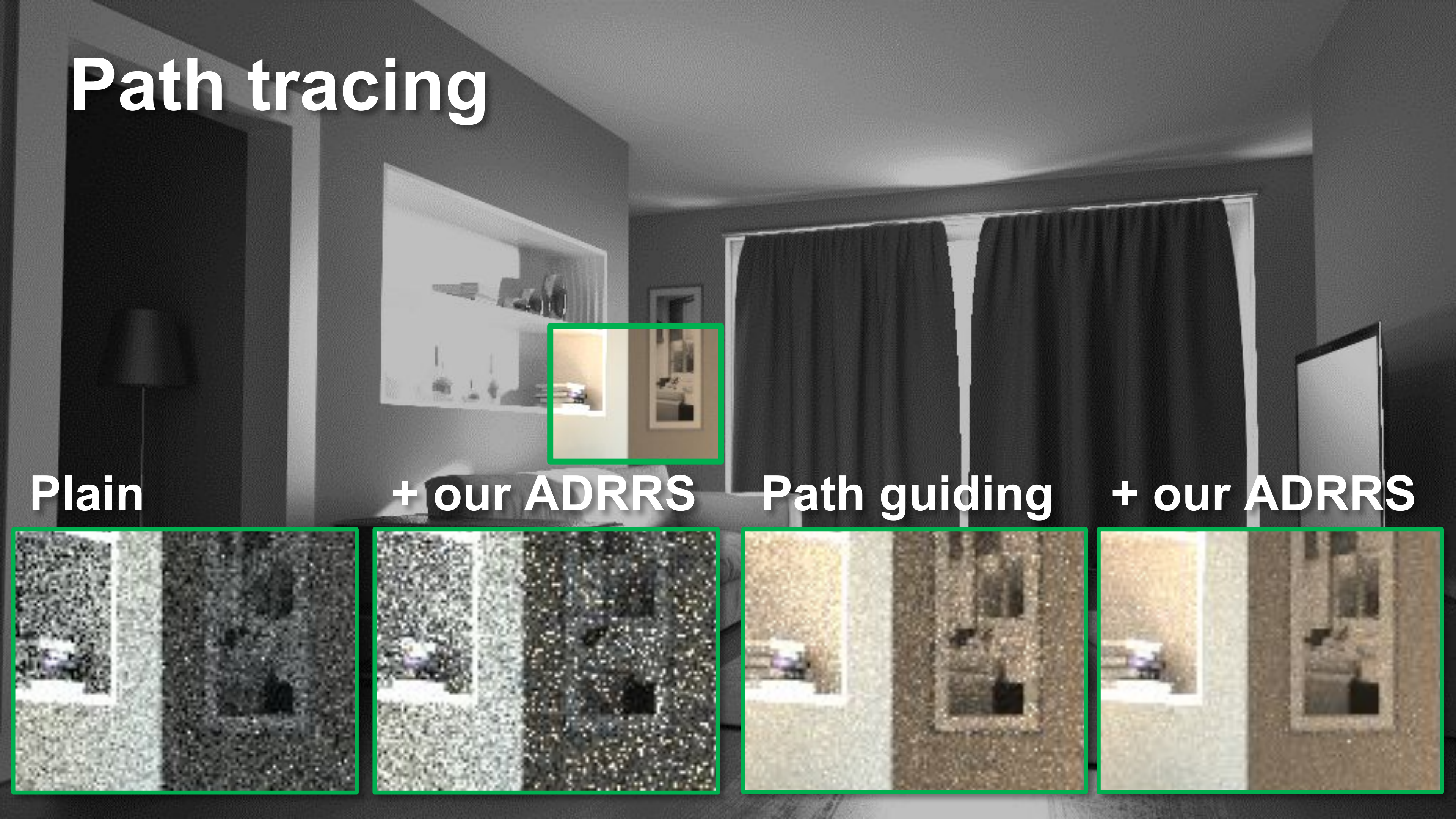
Contribute to pixel estimate

Query radiance cache to compute desired weight

Apply weight window (RR/splitting)

Results


Path tracing
Plain
+ our ADRRS


Path tracing
Plain
+ our ADRRS
Path guiding
+ our ADRRS

Conclusion

- Use precomputed radiance (adjoint) estimation to compute expected path contribution
- Set RR/splitting factor as the expected contribution
- Able to make more particles for high-contribution path
- Reduced variance in rendered image