Computer Graphics Charles University

weta DIGITAL

Adjoint-Driven Russian Roulette and Splitting in Light Transport Simulation

Jiří Vorba[1,2] Jaroslav Křivánek[1]

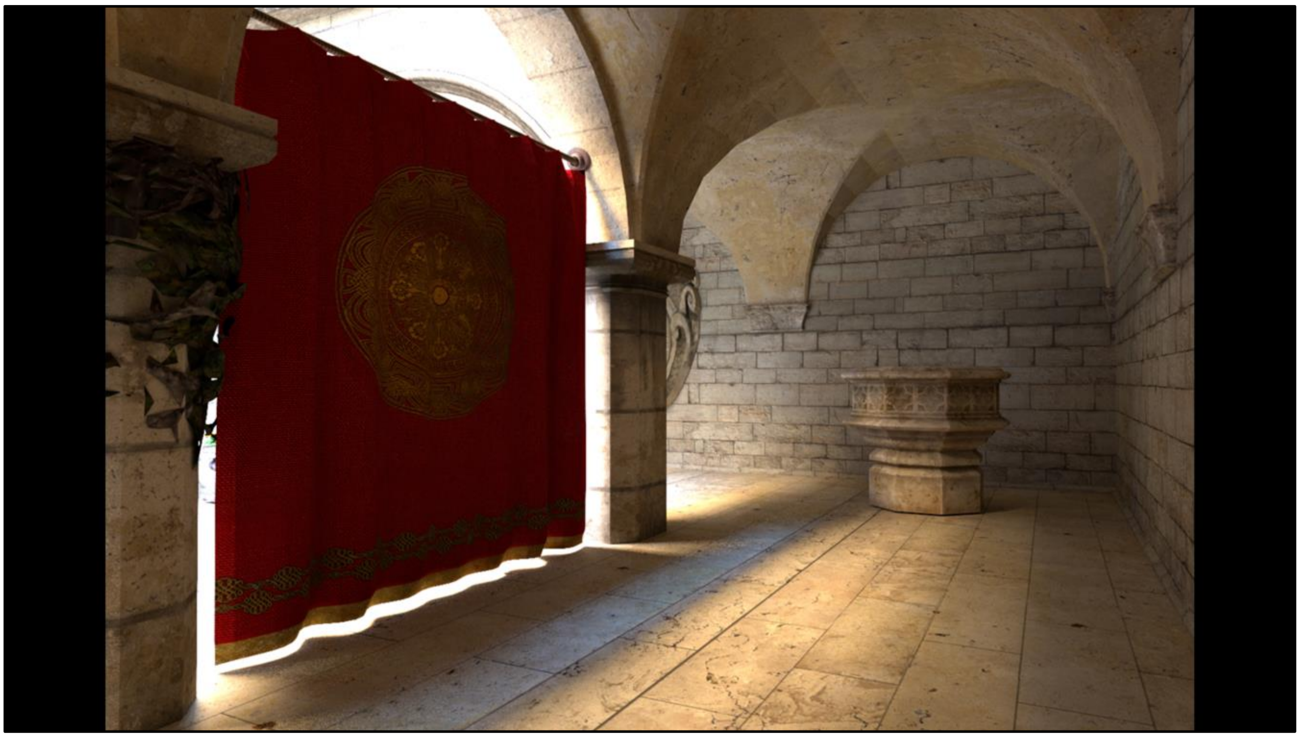
1) Charles University in Prague
2) Weta Digital, Wellington

I present Adjoint-Driven Russian Roulette and Splitting for efficient light transport simulation which is a part of my PhD. work at Charles University in Prague.

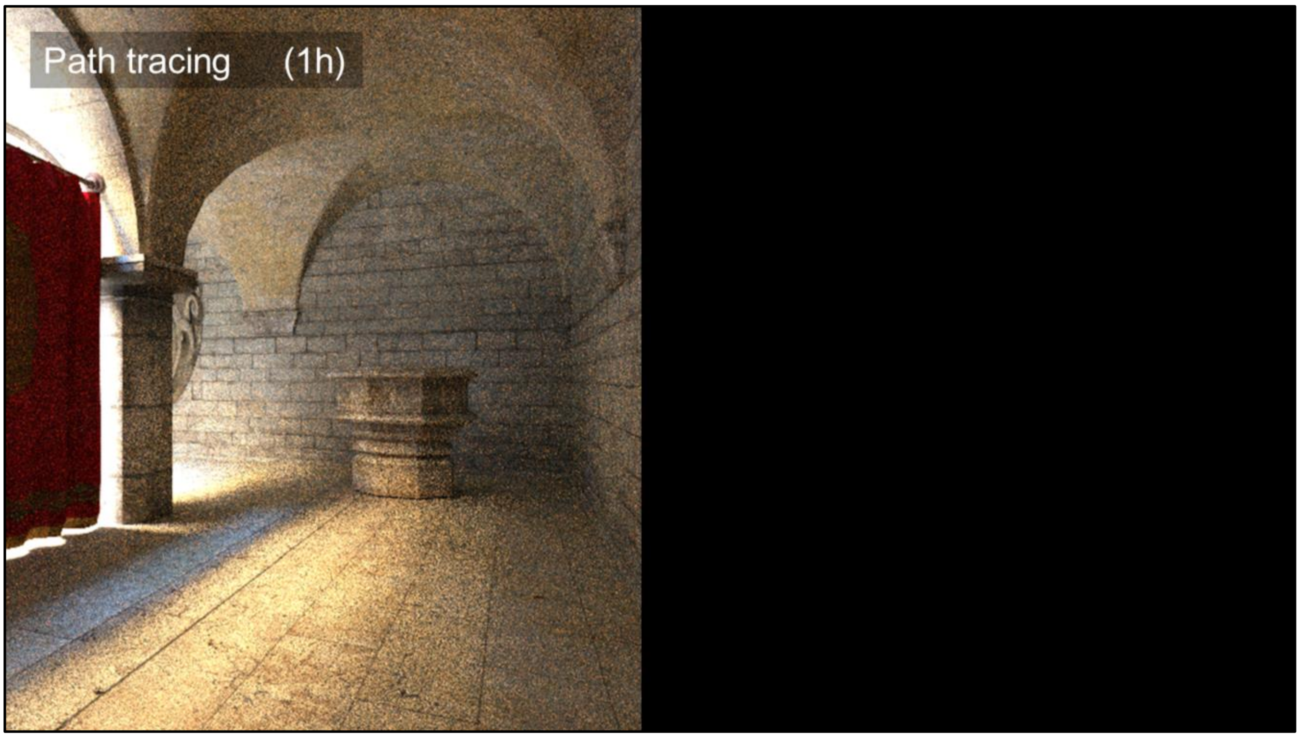
Motivation

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 2

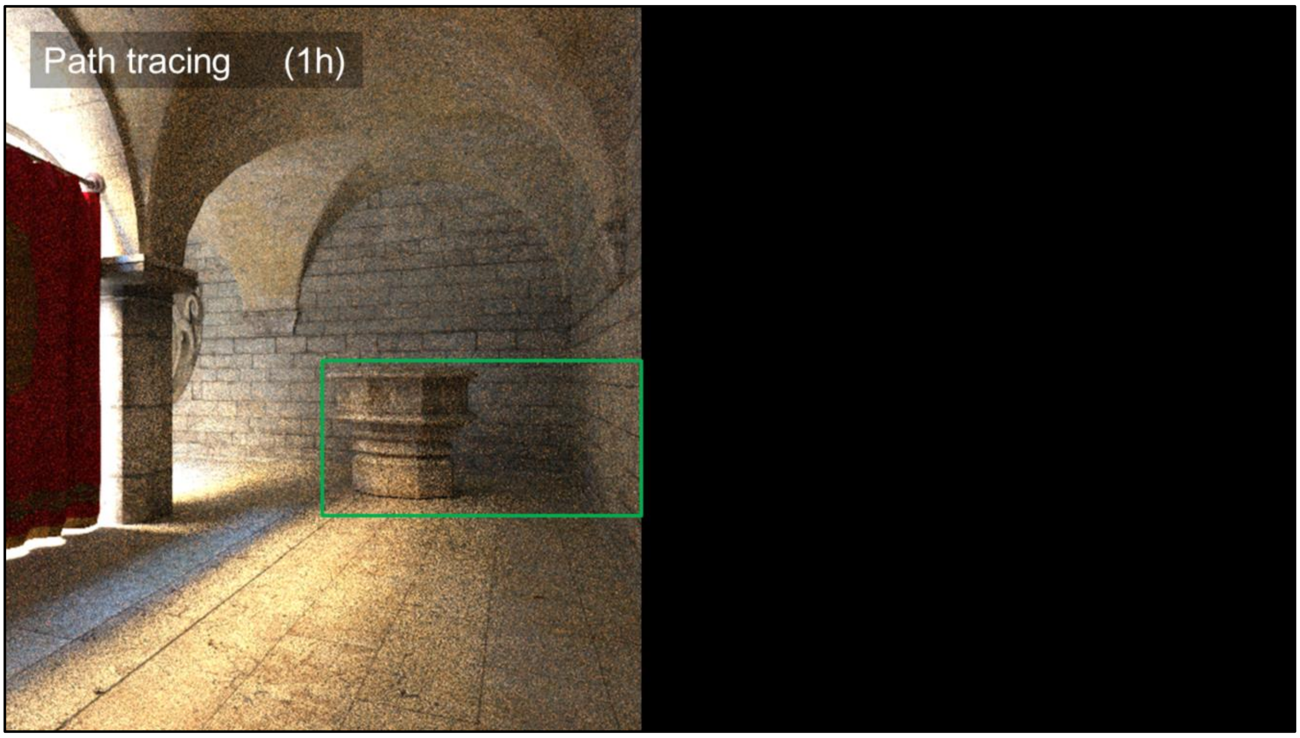
The simulation becomes expensive especially in the presence of indirect illumination.



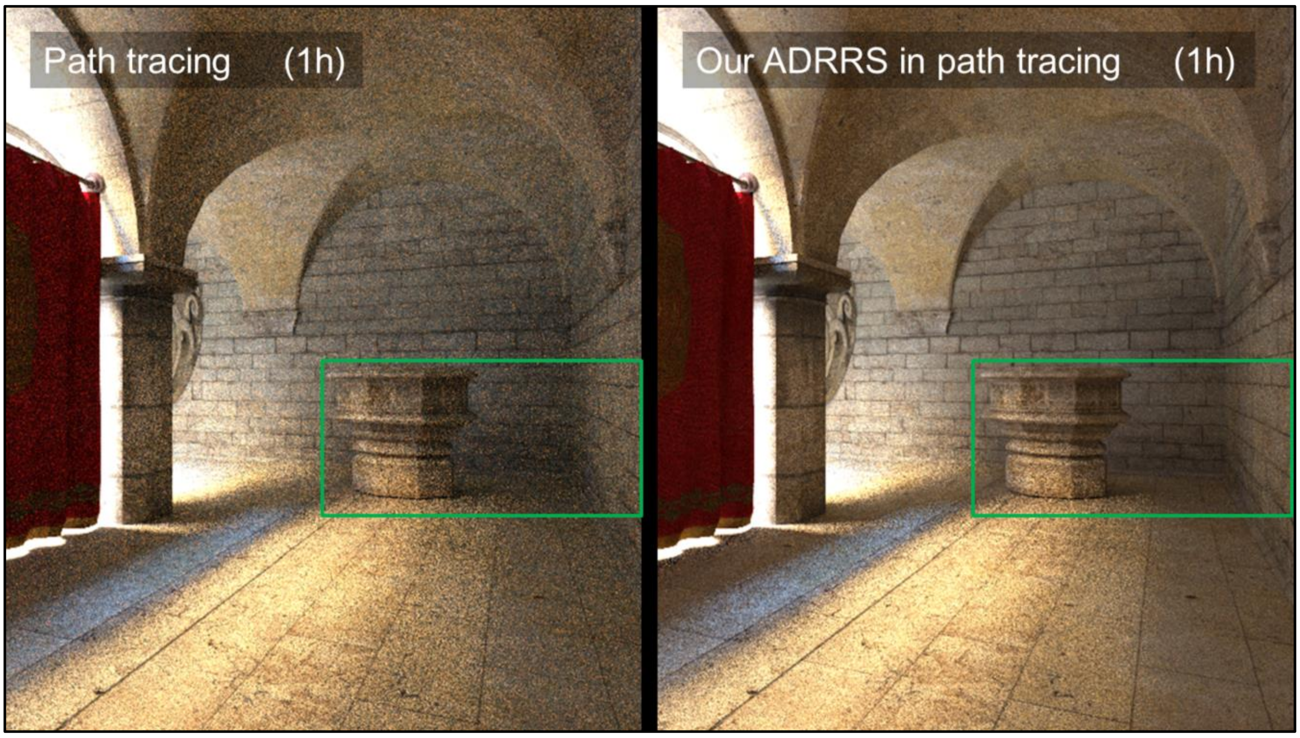
In this Crytek recreation of the famous Sponza scene, the sun shines into the atrium behind the curtain and the shot is lit mostly indirectly.



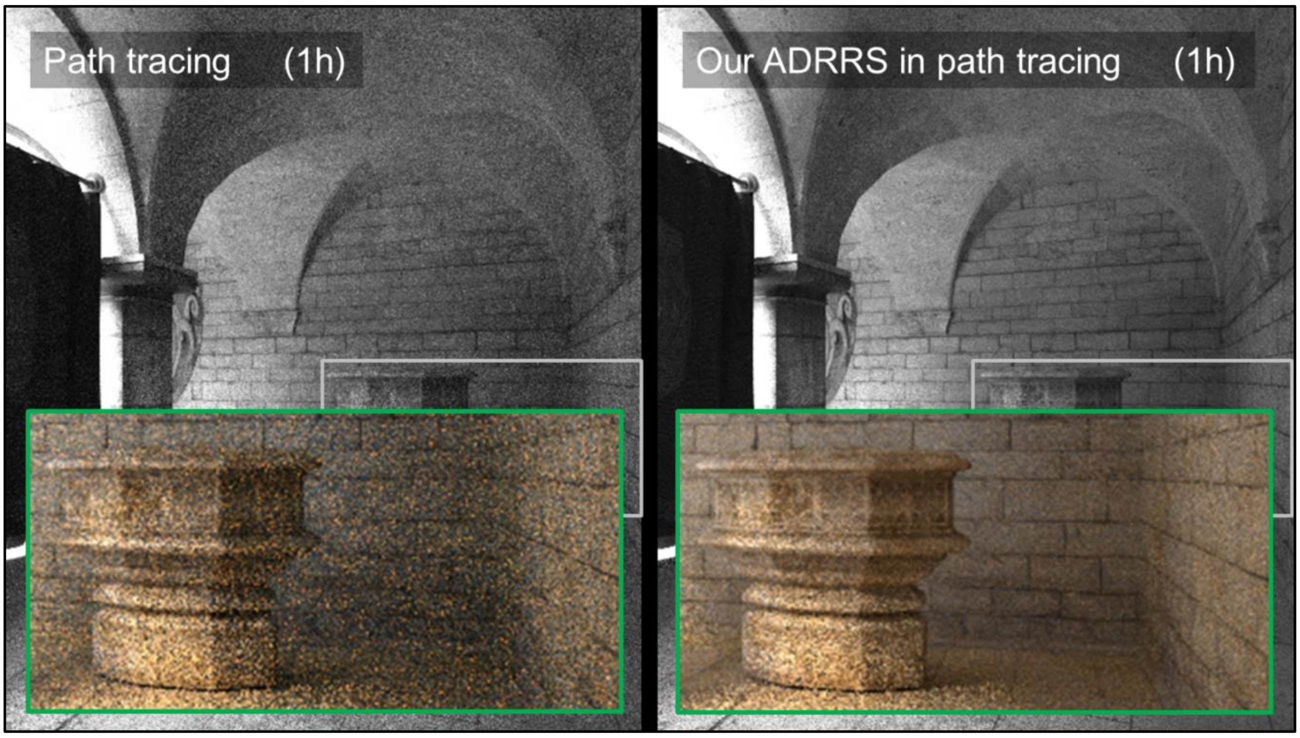
When we render the scene with path tracing that utilizes classic albedo-driven Russian roulette, we can notice that the image is especially noisy in areas …



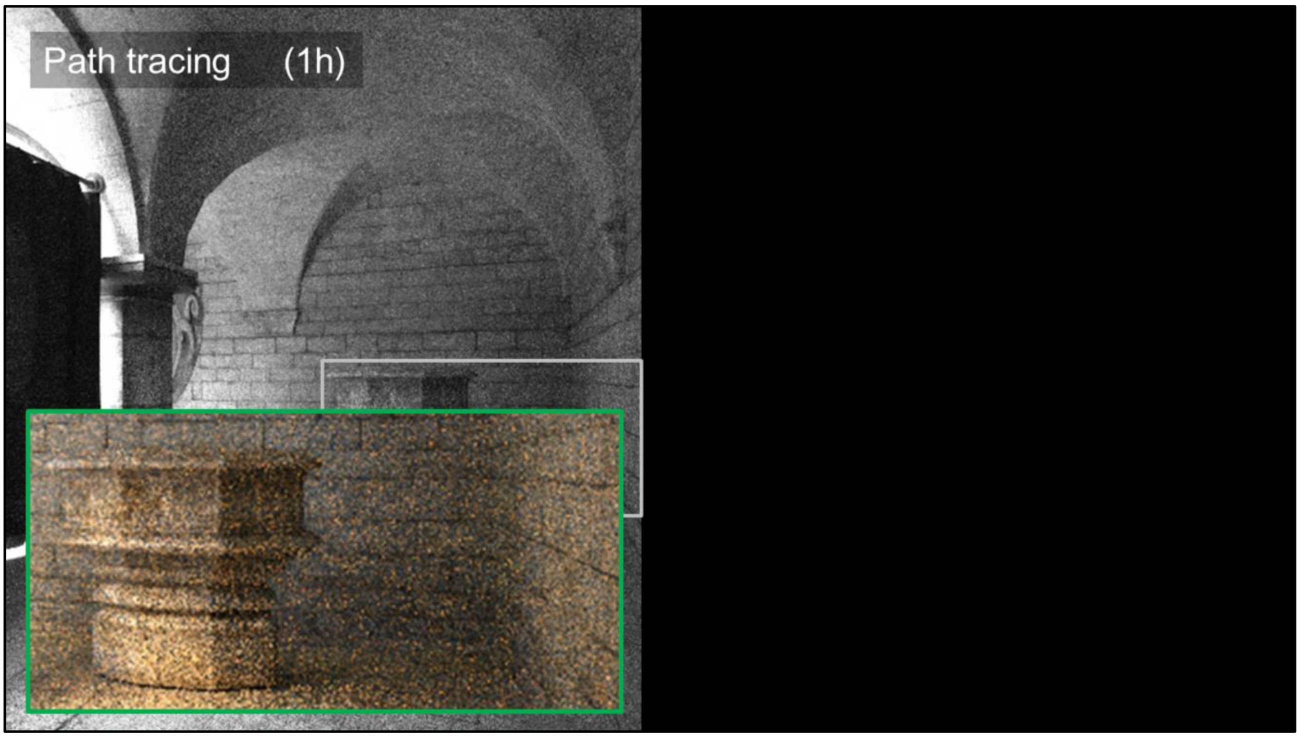
… where light is transported over long paths.

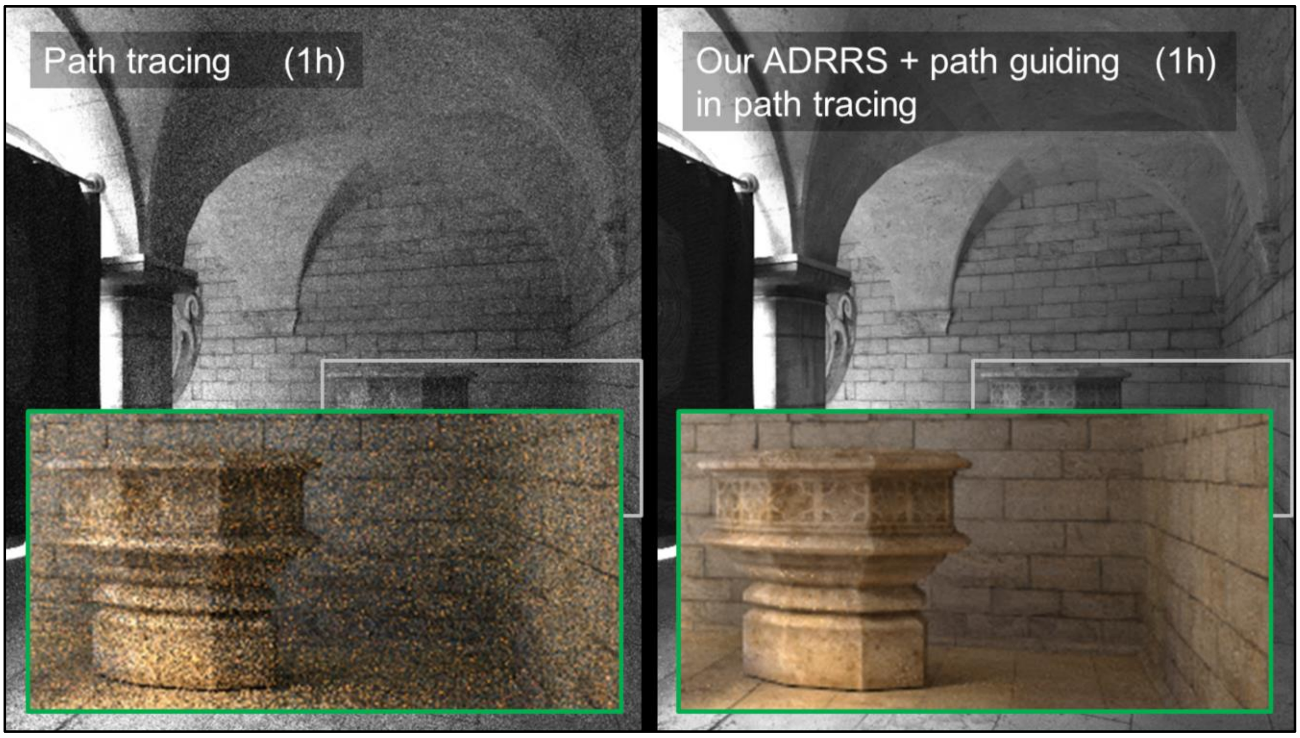


We show that our Adjoint-Driven Russian roulette and Splitting achieves better importance sampling and results in more efficient simulations.

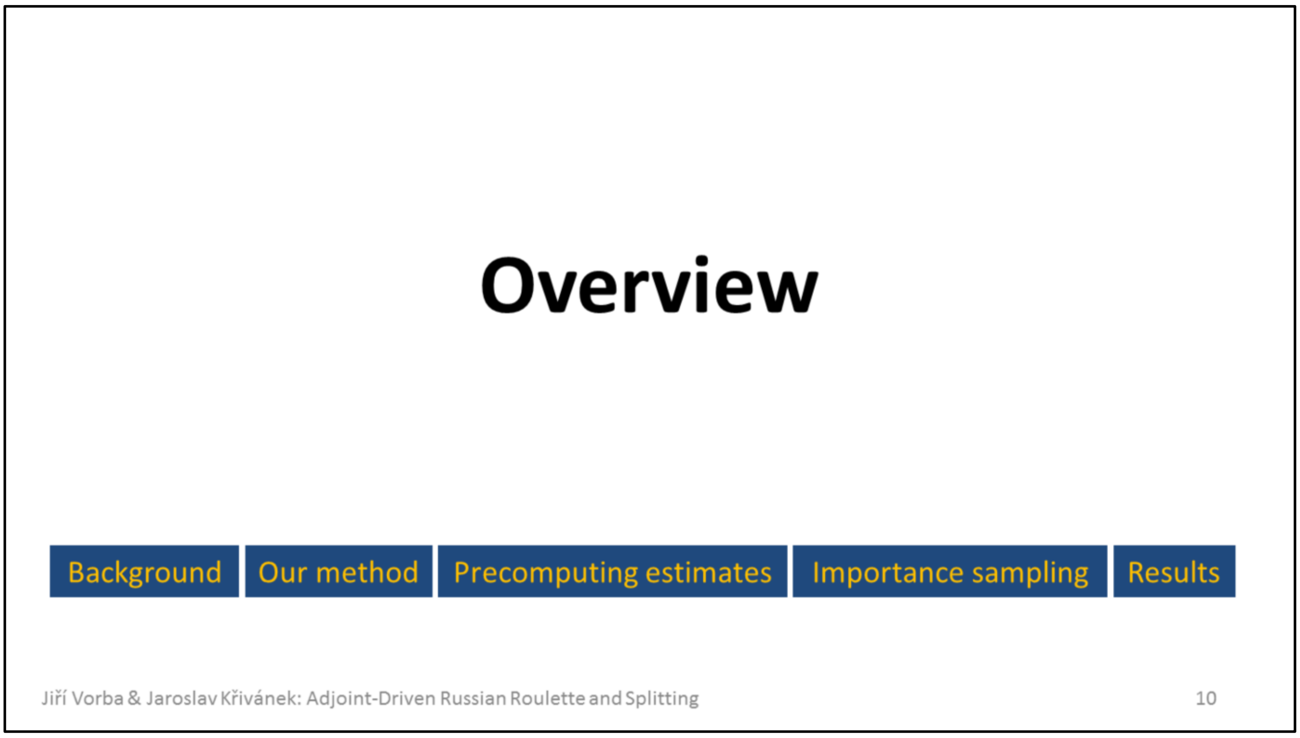


Recently, we introduced a path guiding method for efficient sampling of indirect illumination.


Path tracing (1h)



It turns out, that our new ADRRS method further improves the efficiency of path guiding and that these two different importance sampling schemes work in synergy.



Overview

Background | Our method | Precomputing estimates | Importance sampling | Results

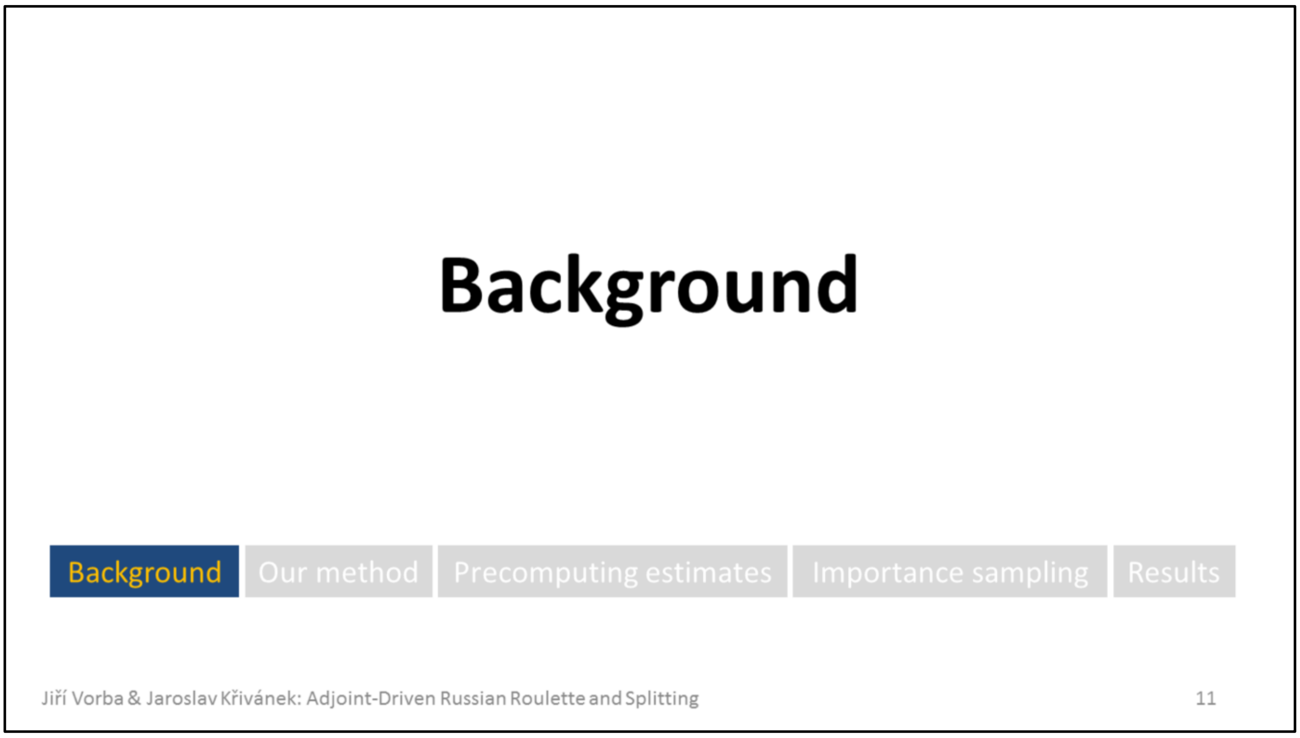
I start by providing background on Russian roulette and splitting.

In the next section I will explain the core of our method that lies in estimating expected contribution of each path.

To estimate the expected contribution we need to precompute estimates of radiance field and of our pixels which is the topic of the following section.

In the last but one section I will present our theoretical contribution that explains importance sampling properties of our method. It also shows why it works in synergy with path guiding.

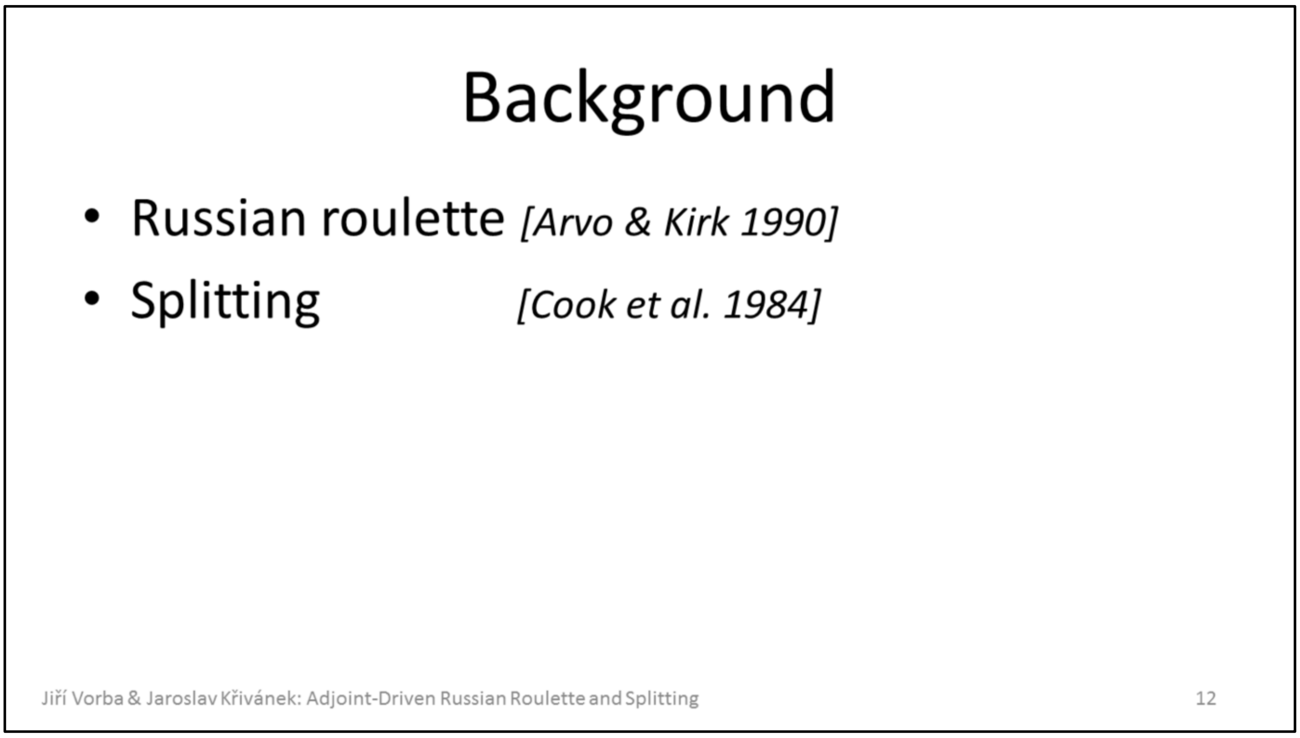
In the end I will present some of our results.



Background

Background | Our method | Precomputing estimates | Importance sampling | Results

Lets start with the background.



Background

- Russian roulette *[Arvo & Kirk 1990]*
- Splitting *[Cook et al. 1984]*

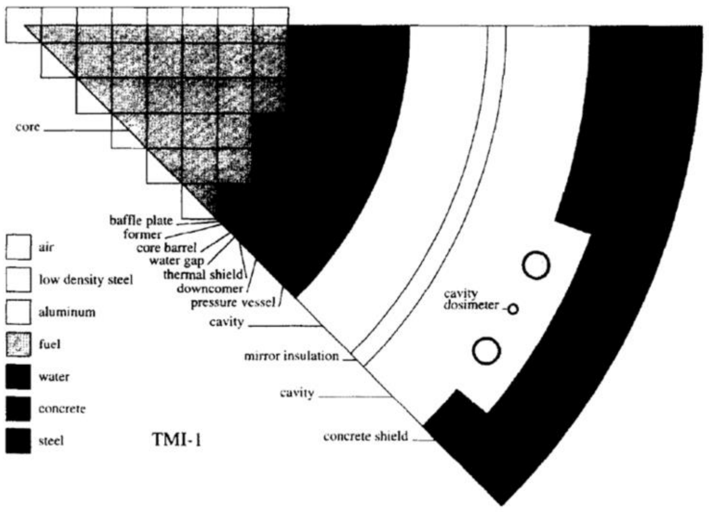
Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 12

Since Arvo & Kirk adopted Russian roulette from neutron transport simulations it has been used in the same form ever since.

They also noticed that splitting of path trajectories is used in neutron transport as well and that in computer graphics it is known as distributed ray tracing.

We believe that the main contribution of our work is the revision of the Russian roulette and splitting in light transport simulation.

People in neutron transport apply Russian roulette and splitting successfully on remarkably difficult transport problems.



One octant of the TMI-1 reactor

(Courtesy of Haghighat & Wagner)

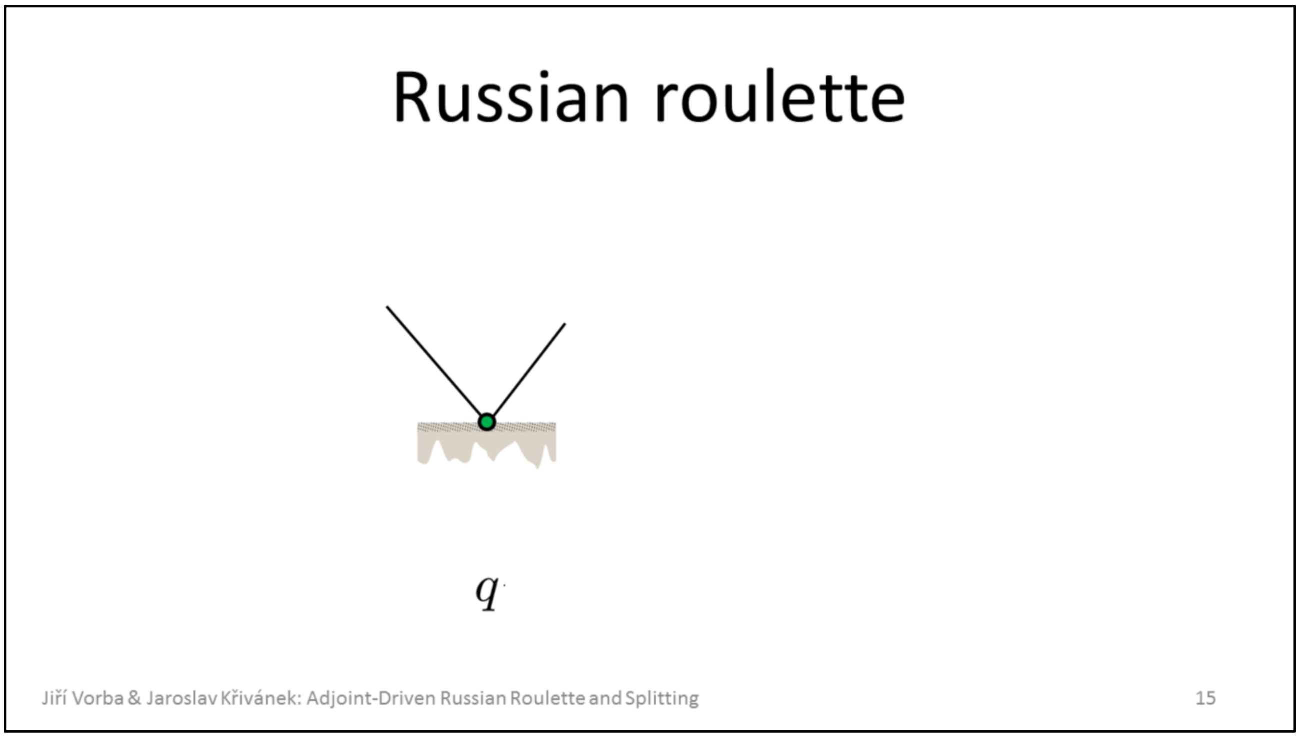
Imagine that we want to measure radiation that escapes through a shield of a nuclear reactor.

If the shield is designed well then roughly 1 in a billion or even less particles that were emitted in the reactor core will make it out through the shield.

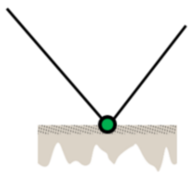
Well, that sounds as a mission impossible for MC estimators.

Yet if they are given a rough estimate of how much are various parts of the shield important with respect to the computed estimate they can use Russian roulette and splitting as powerful importance sampling techniques.

This has directly inspired our approach as difficult visibility and multi-bounce indirect illumination cause similar problems to MC estimators.



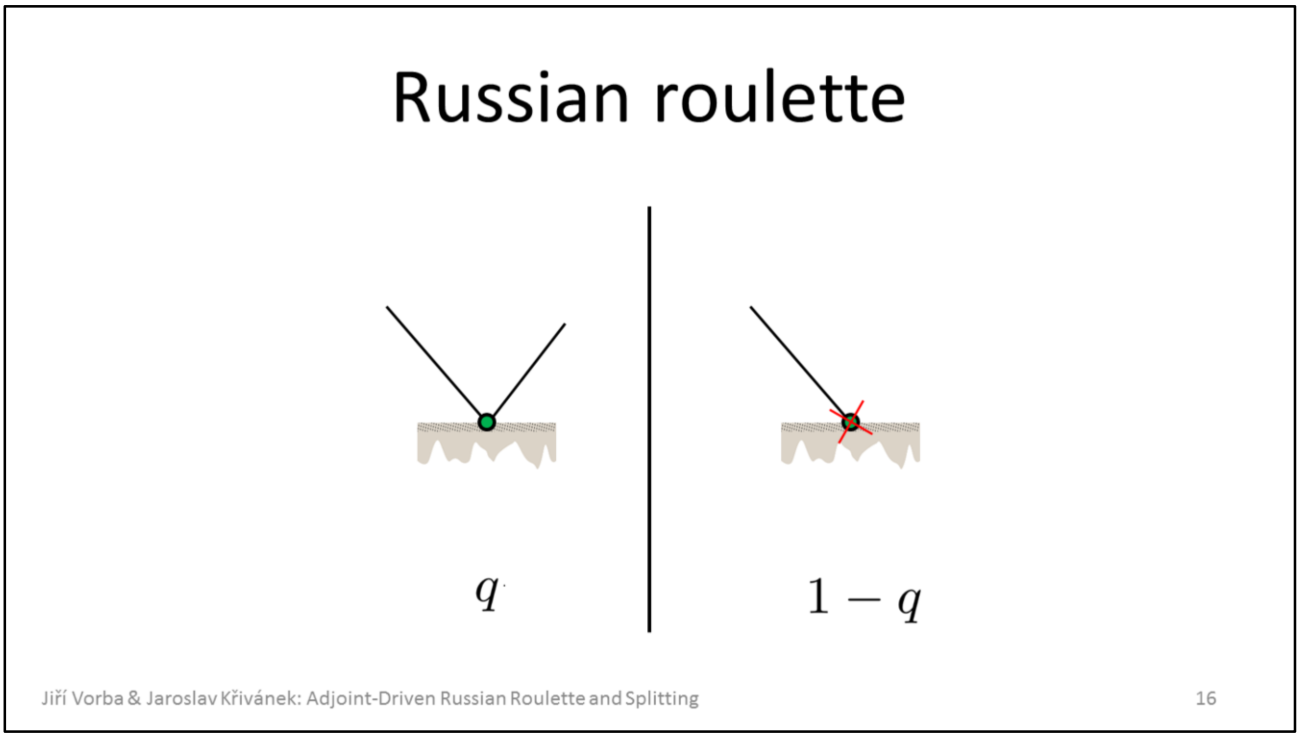
Russian roulette



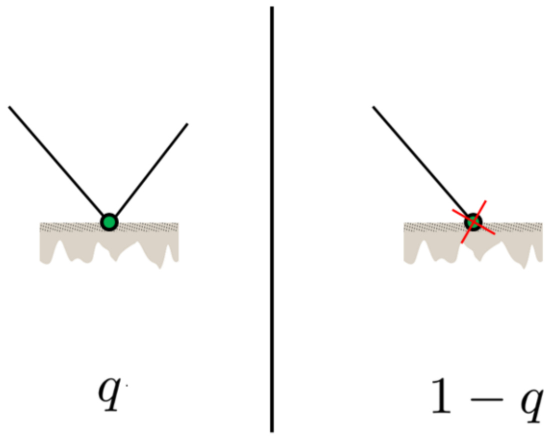
q

So lets have a look at the basics of Russian roulette and splitting.

Under Russian roulette at each vertex the path survives with some probability q …



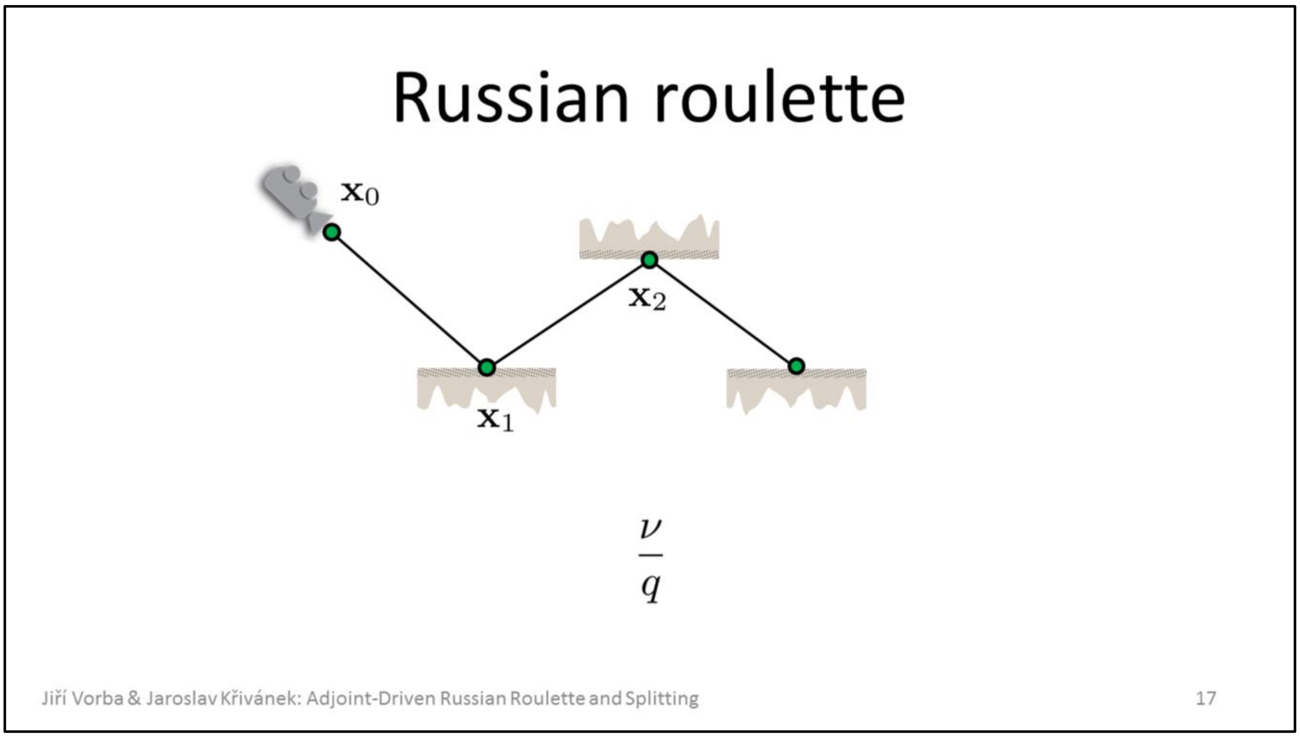
Russian roulette



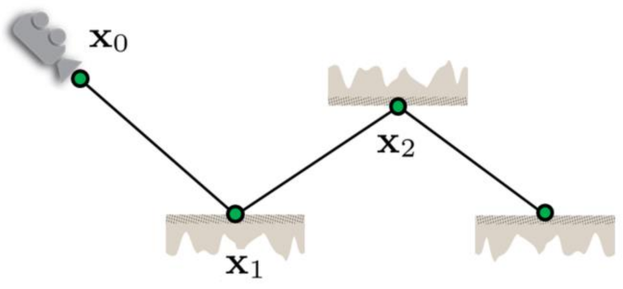
Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 16

… and is terminated with probability 1-q.

If path is terminated, it does not contribute to estimated pixel value.

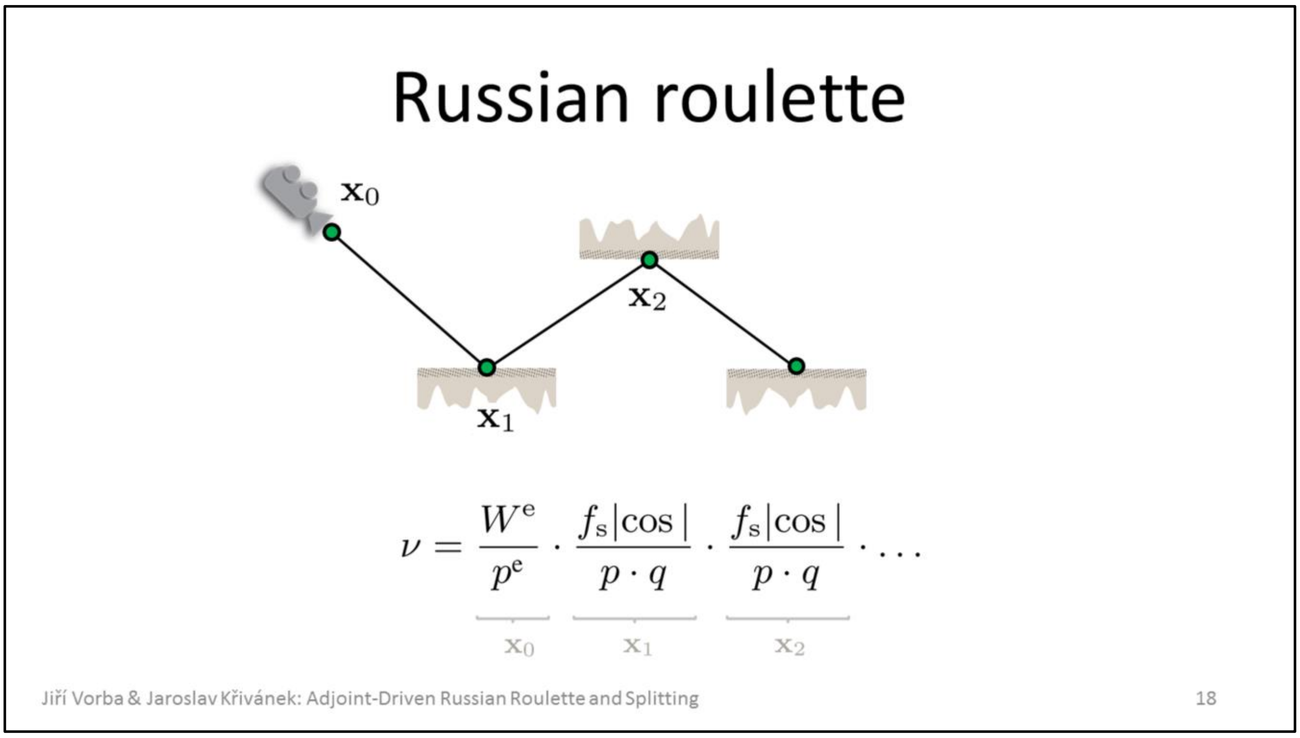


Russian roulette

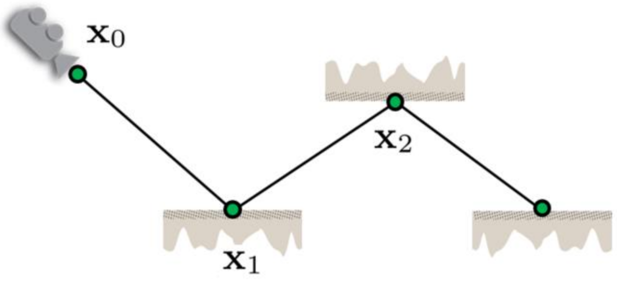


$$\frac{\nu}{q}$$

If the path survives its throughput is divided by the surviving probability to compensate for terminated paths.



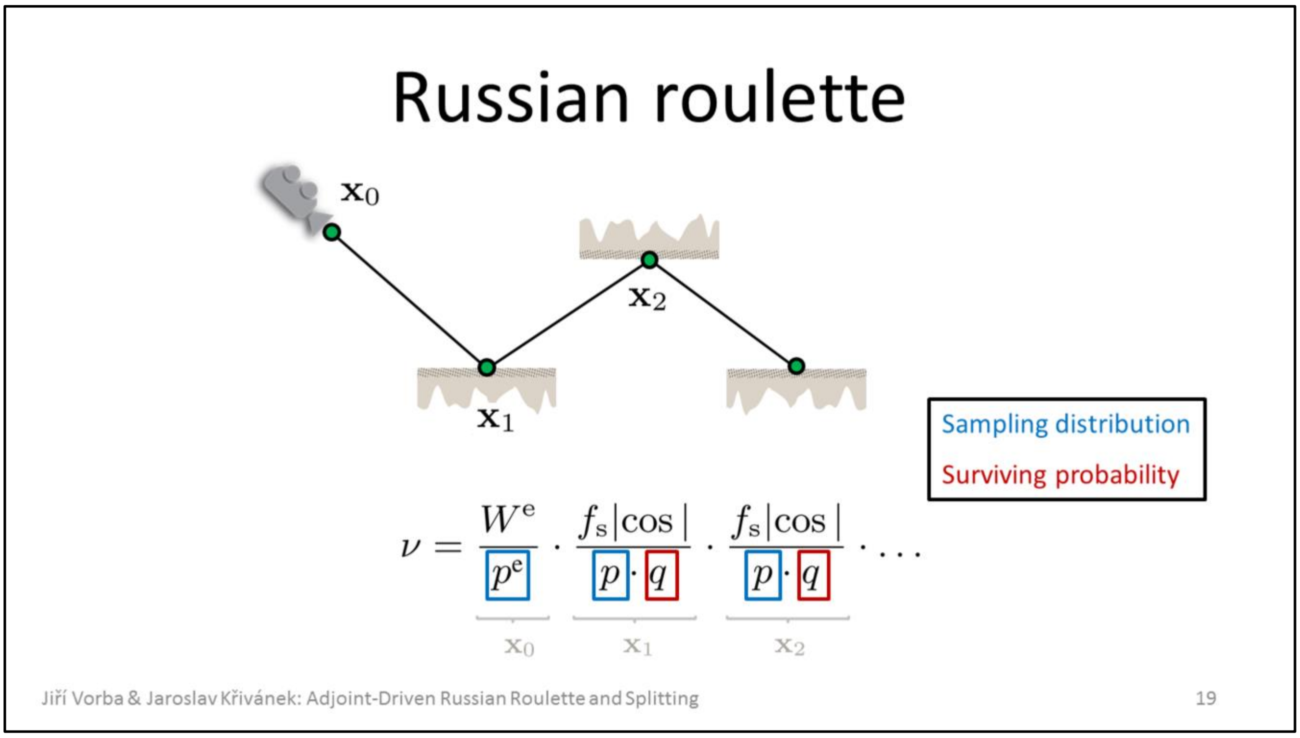
Russian roulette



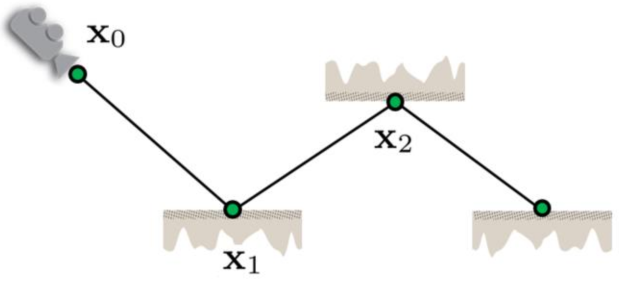
$$\nu = \underbrace{\frac{W^{\mathrm{e}}}{p^{\mathrm{e}}}}_{\mathbf{x}_0} \cdot \underbrace{\frac{f_{\mathrm{s}}|\cos|}{p \cdot q}}_{\mathbf{x}_1} \cdot \underbrace{\frac{f_{\mathrm{s}}|\cos|}{p \cdot q}}_{\mathbf{x}_2} \cdot \ldots$$

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 18

The throughput also known as path weight, is a product of emission, bsdf and geometric terms,



Russian roulette



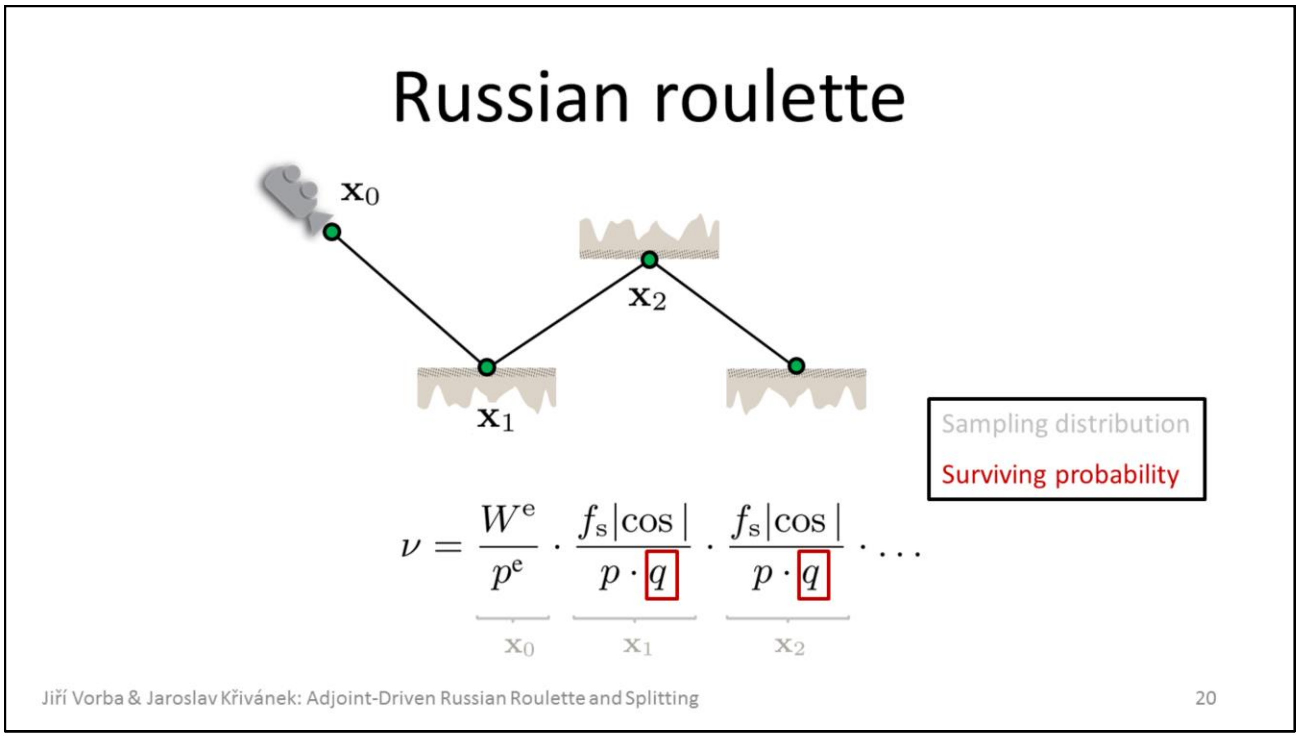
Sampling distribution

Surviving probability

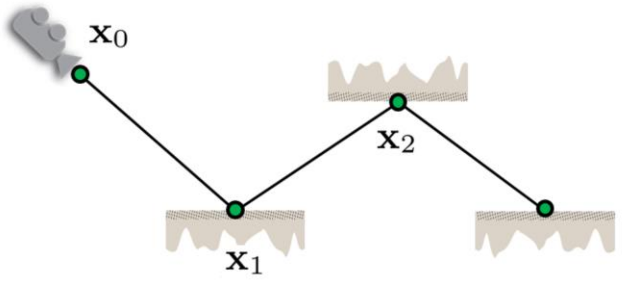
$$\nu = \underbrace{\frac{W^{\mathrm{e}}}{\boxed{p^{\mathrm{e}}}}}_{\mathbf{x}_0} \cdot \underbrace{\frac{f_{\mathrm{s}}|\cos|}{\boxed{p}\cdot\boxed{q}}}_{\mathbf{x}_1} \cdot \underbrace{\frac{f_{\mathrm{s}}|\cos|}{\boxed{p}\cdot\boxed{q}}}_{\mathbf{x}_2} \cdot \ldots$$

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 19

and these terms are also divided by the direction sampling probabilities p and surviving probabilities q at each vertex.



Russian roulette



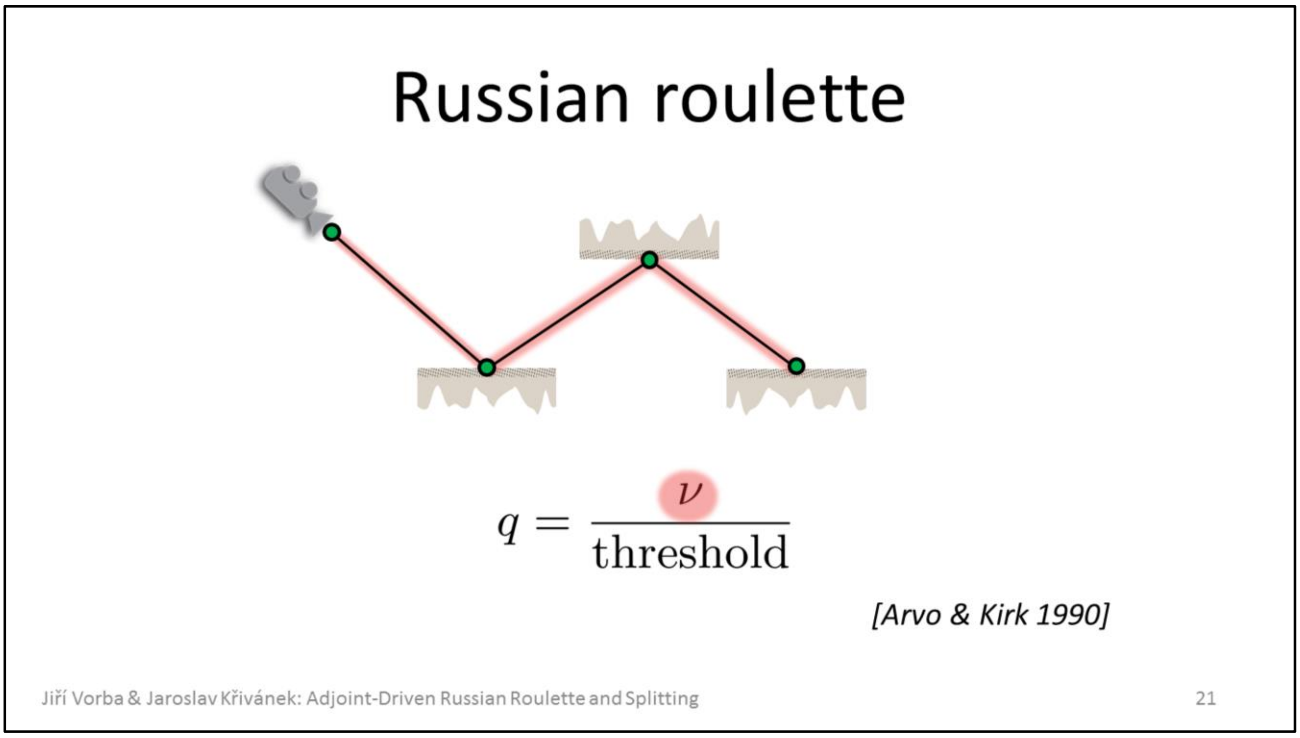
Sampling distribution

Surviving probability

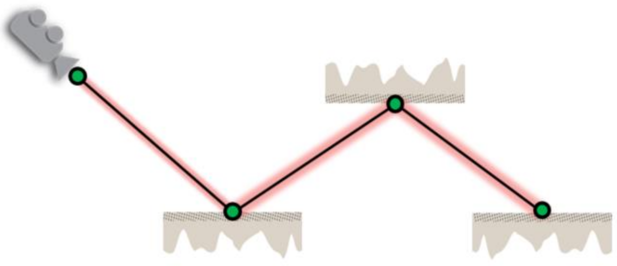
$$\nu = \underbrace{\frac{W^{\mathrm{e}}}{p^{\mathrm{e}}}}_{\mathbf{x}_0} \cdot \underbrace{\frac{f_{\mathrm{s}}|\cos|}{p \cdot \boxed{q}}}_{\mathbf{x}_1} \cdot \underbrace{\frac{f_{\mathrm{s}}|\cos|}{p \cdot \boxed{q}}}_{\mathbf{x}_2} \cdot \ldots$$

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 20

Russian roulette changes the path weight only through the surviving probability.



Russian roulette



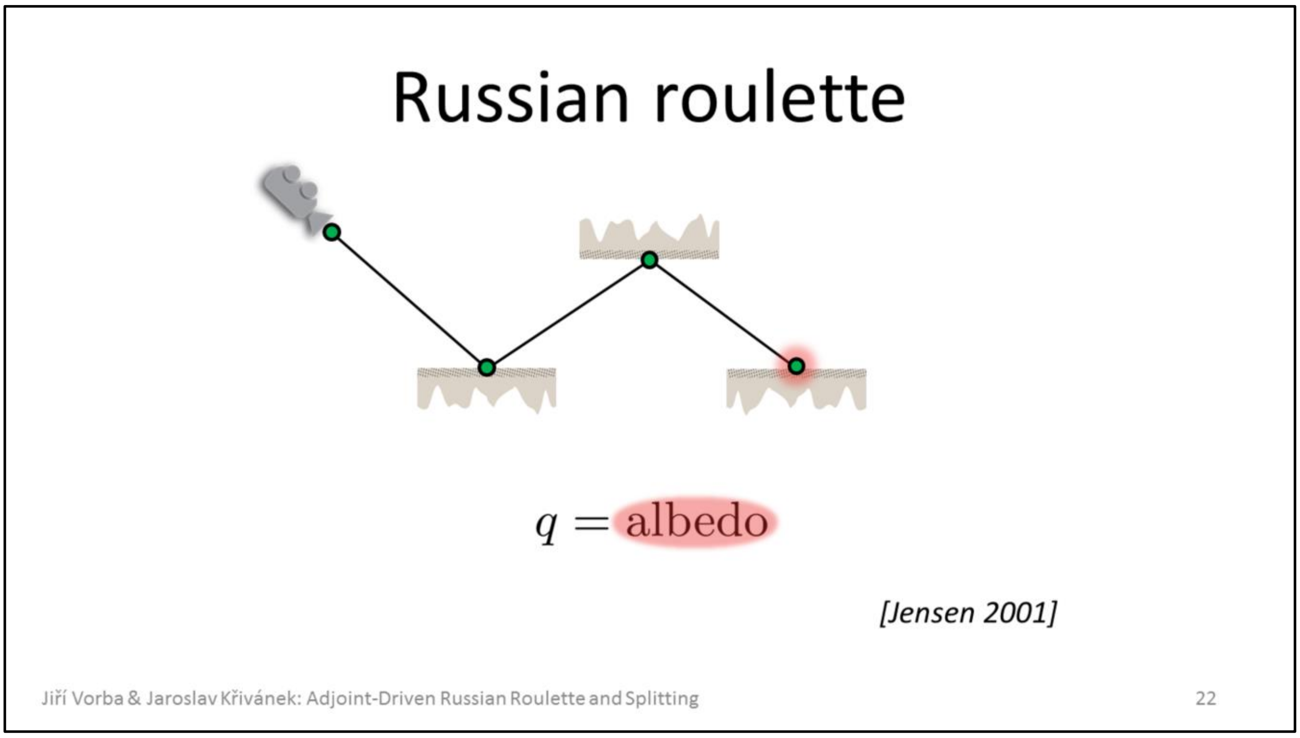
$$q = \frac{\nu}{\text{threshold}}$$

[Arvo & Kirk 1990]

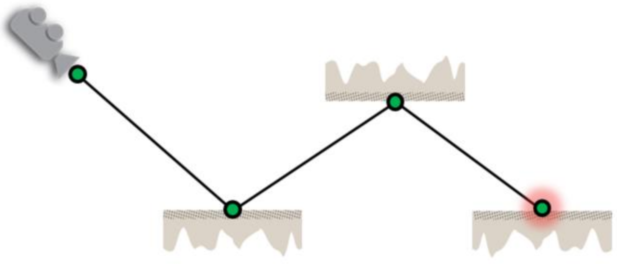
Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 21

Arvo & Kirk suggest to set the surviving probability to a ratio of the current path weight and a user defined threshold.

This allows unbiased rendering without spending too much time on tracing long paths with a small throughput.



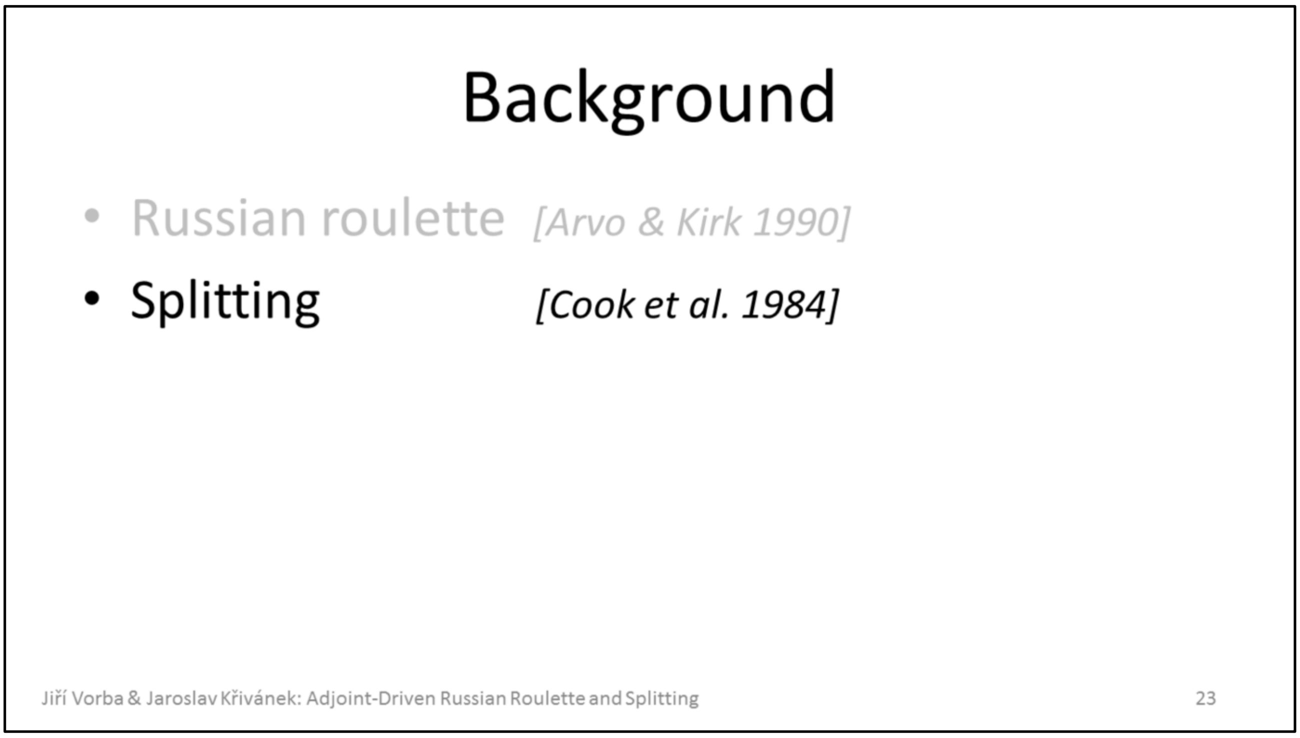
Russian roulette



$$q = \text{albedo}$$

[Jensen 2001]

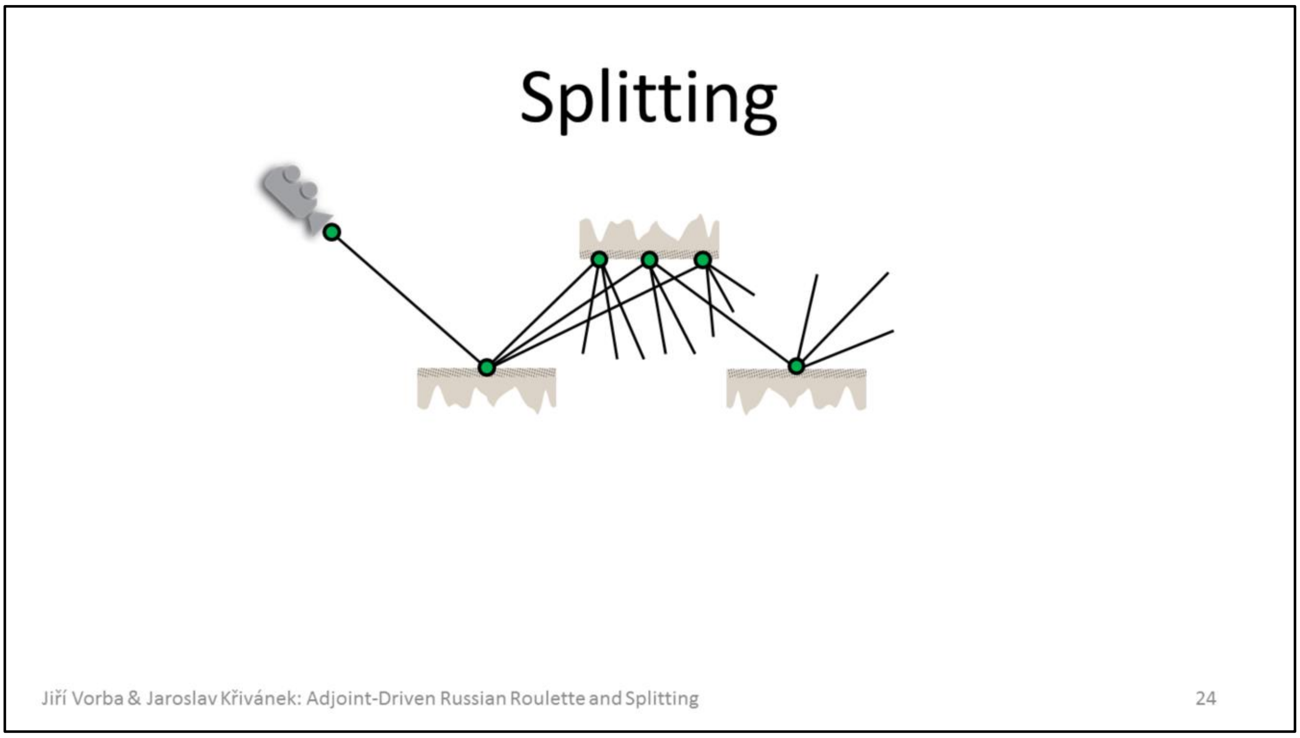
Another commonly used approach is to use local albedo of the material as a surviving probability.



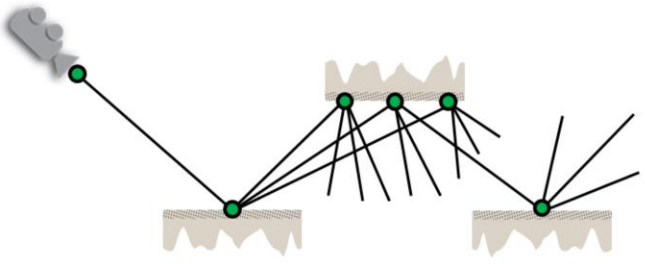
Background

- Russian roulette *[Arvo & Kirk 1990]*
- Splitting *[Cook et al. 1984]*

Splitting.

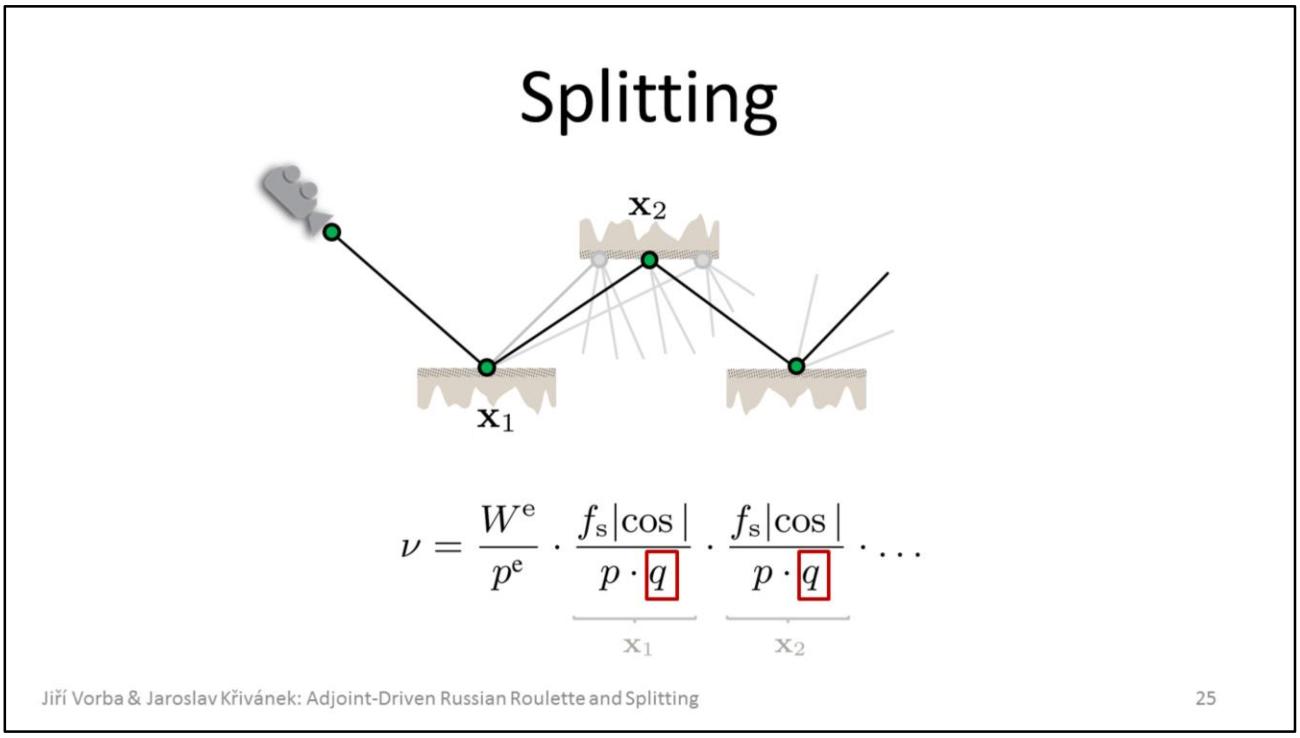


Splitting

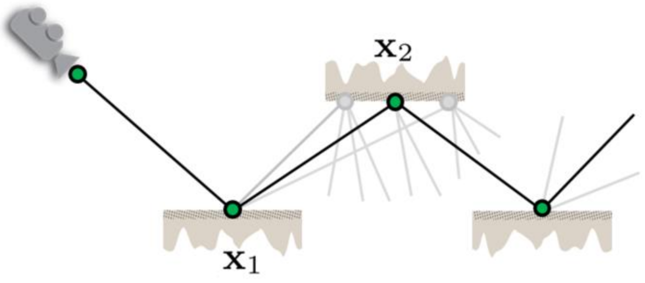


Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 24

It works by splitting a path into several diverging trajectories that are tracked independently.



Splitting

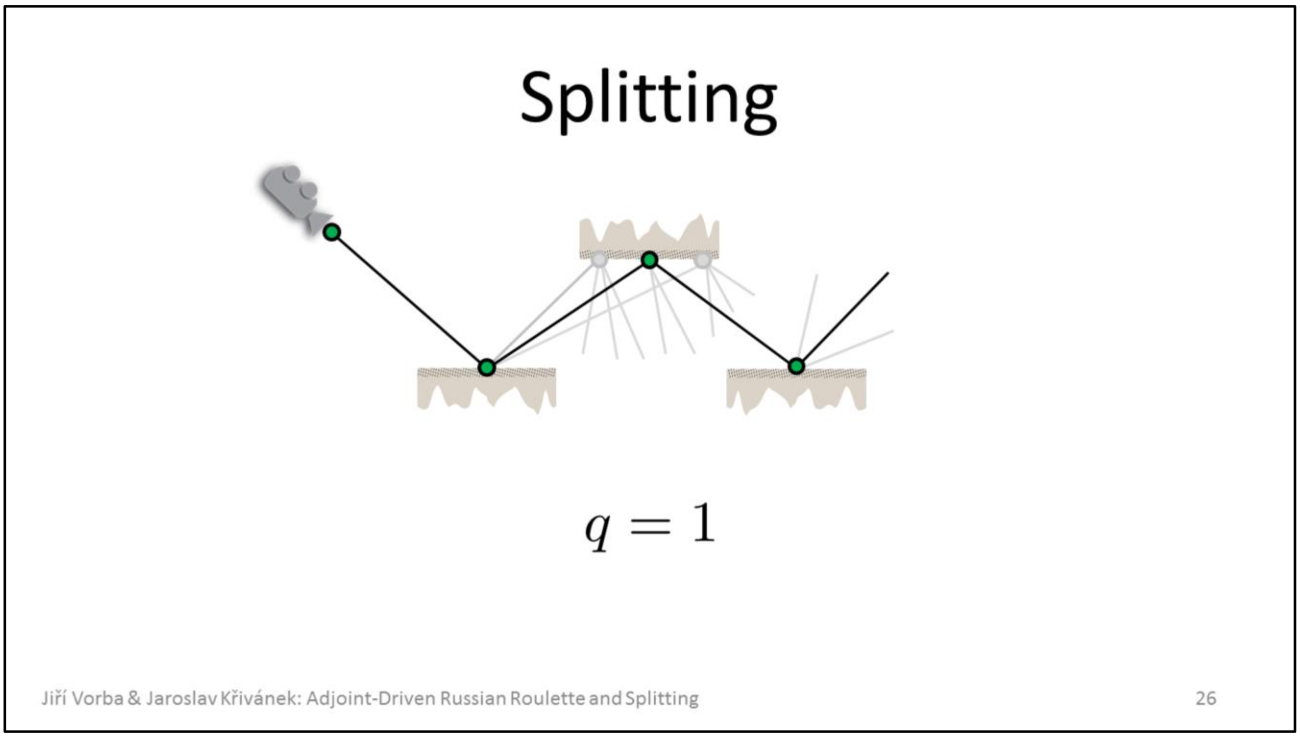


$$\nu = \frac{W^{\mathrm{e}}}{p^{\mathrm{e}}} \cdot \underbrace{\frac{f_{\mathrm{s}}|\cos|}{p \cdot \boxed{q}}}_{\mathbf{x}_1} \cdot \underbrace{\frac{f_{\mathrm{s}}|\cos|}{p \cdot \boxed{q}}}_{\mathbf{x}_2} \cdot \ldots$$

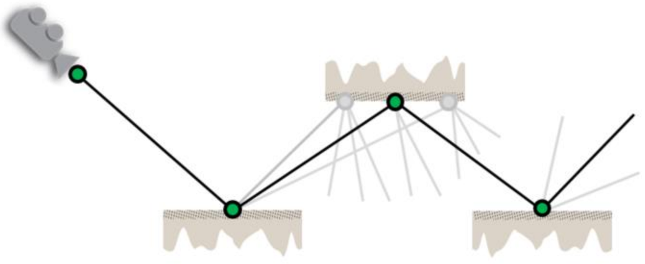
Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 25

Similarly to Russian roulette we need to correct the path weight after each split by the number of spawned paths.

The main problem is again to determine the splitting factor q.



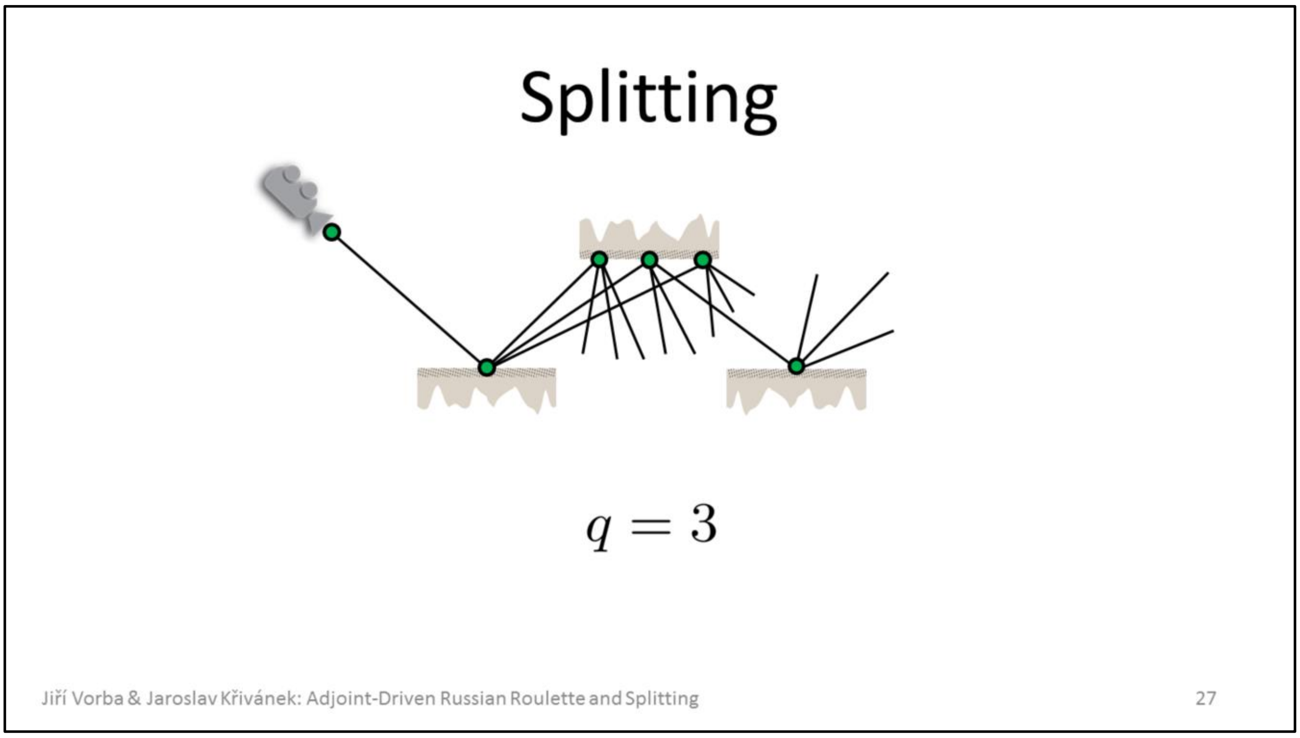
Splitting



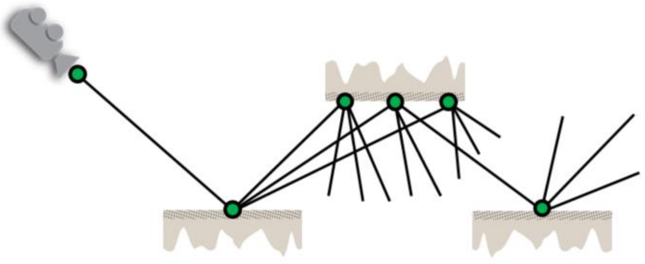
$$q = 1$$

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 26

If we set q to 1 we end up with path tracing.



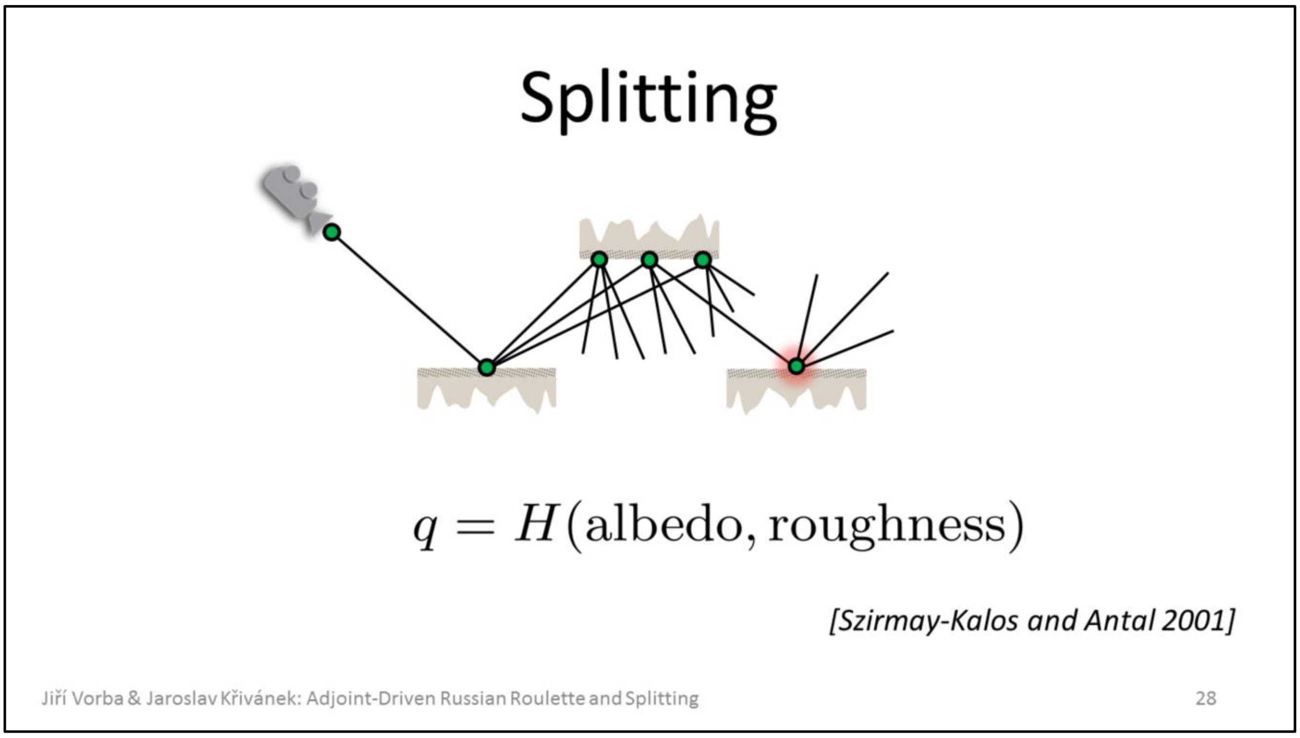
Splitting



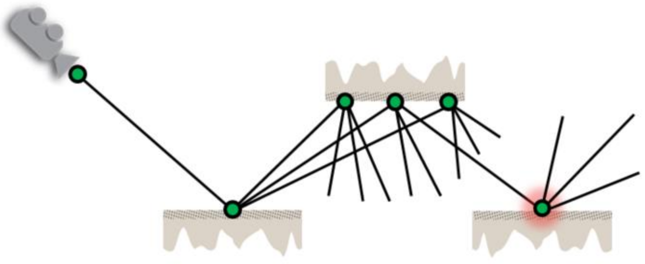
$$q = 3$$

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 27

If we use a fixed rate higher than 1 we get bushy trees due to exponential branching.



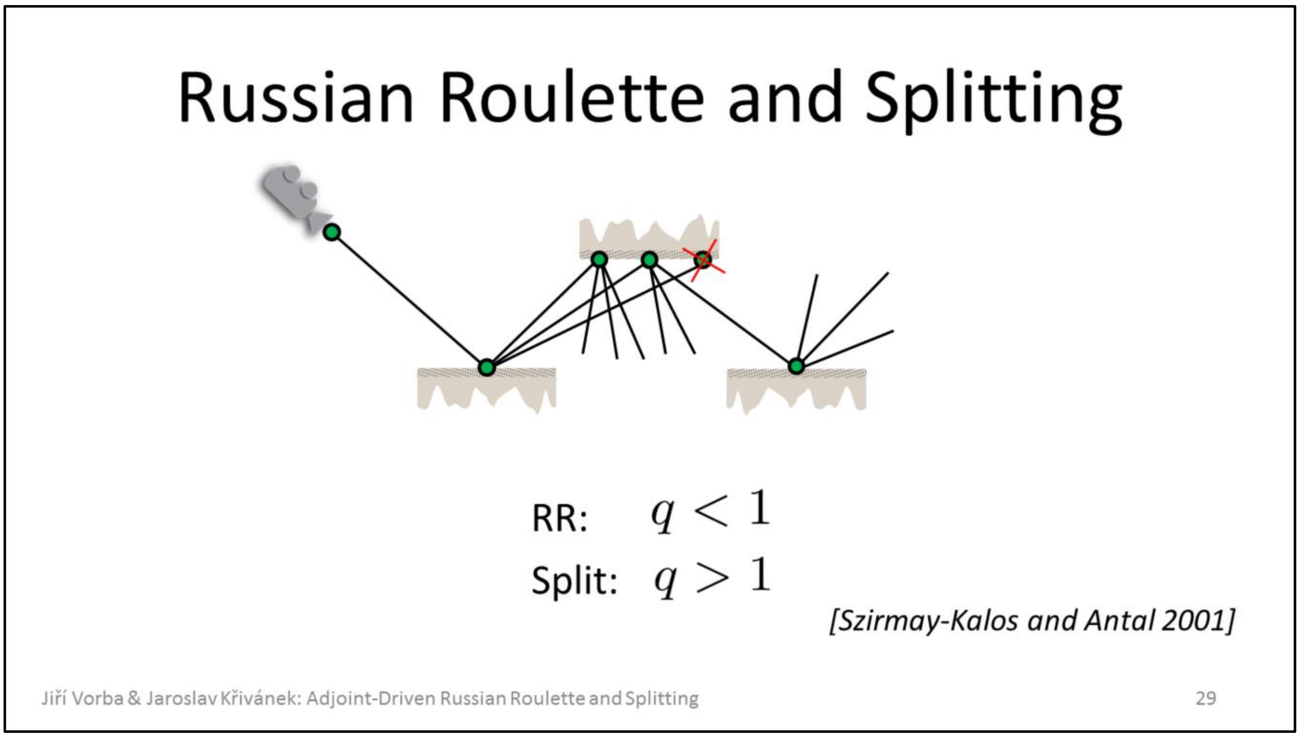
Splitting



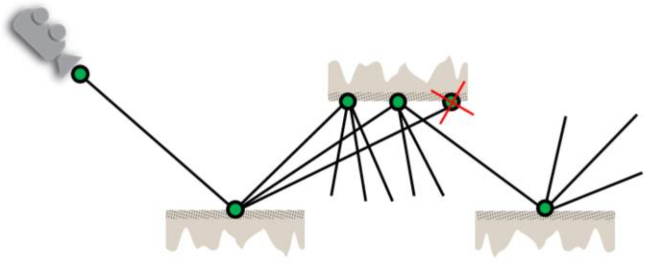
$$q = H(\text{albedo}, \text{roughness})$$

[Szirmay-Kalos and Antal 2001]

Instead of fixed rates, Szirmay-Kalos and Antal suggest to use a heuristic based on material properties like albedo and roughness.



Russian Roulette and Splitting



RR: $q < 1$

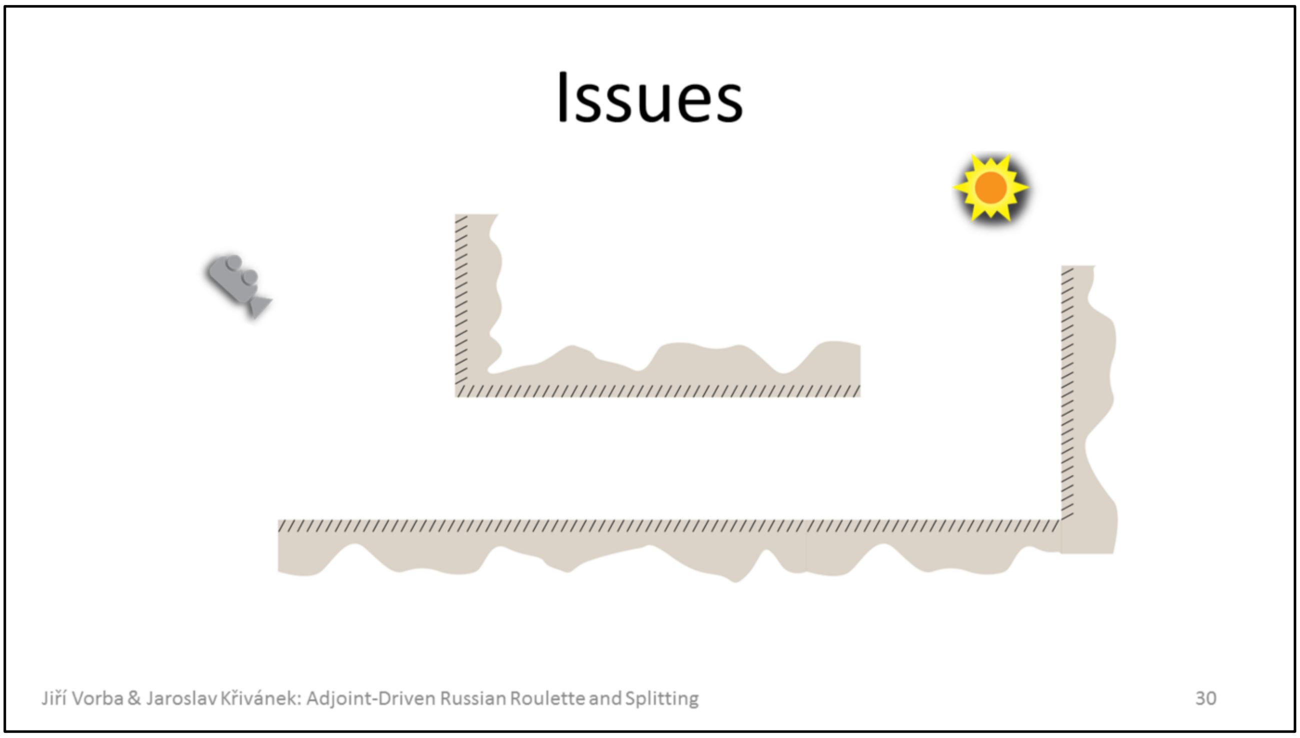
Split: $q > 1$

[Szirmay-Kalos and Antal 2001]

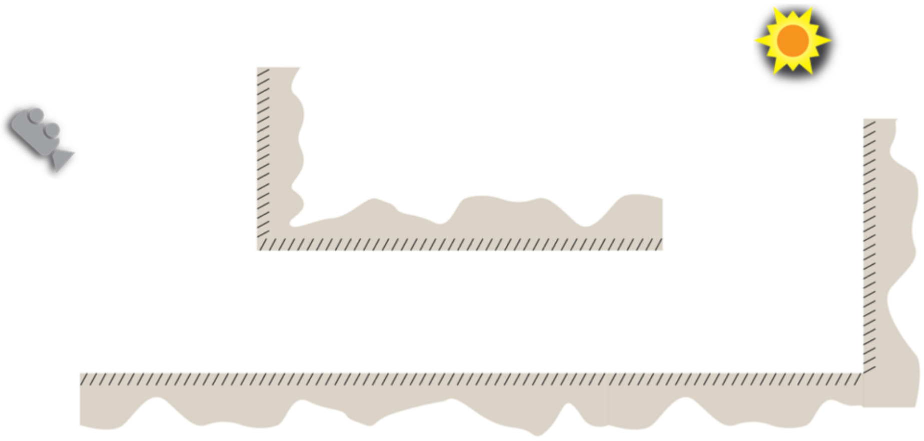
Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 29

Obviously, both Russian roulette and splitting can be naturally combined together through the factor q.

If q is less than one roulette is played if q is greater than one the path is split.



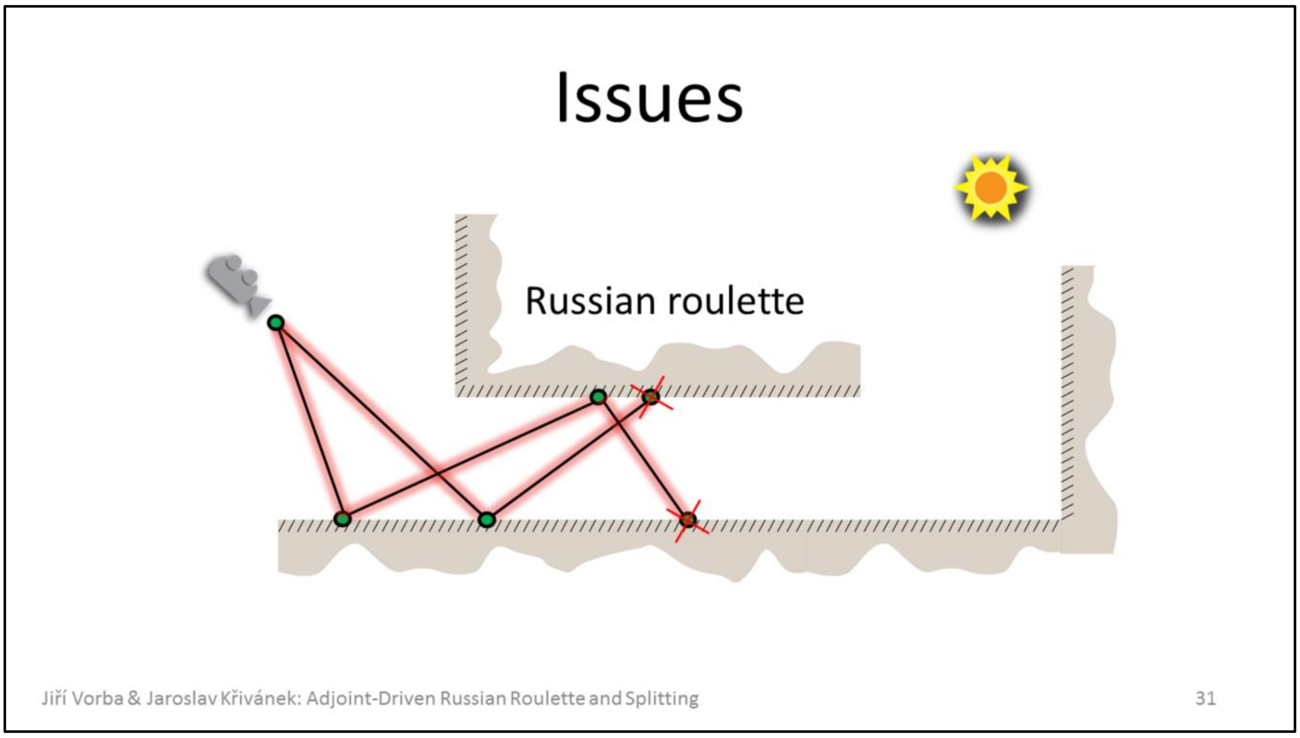
Issues



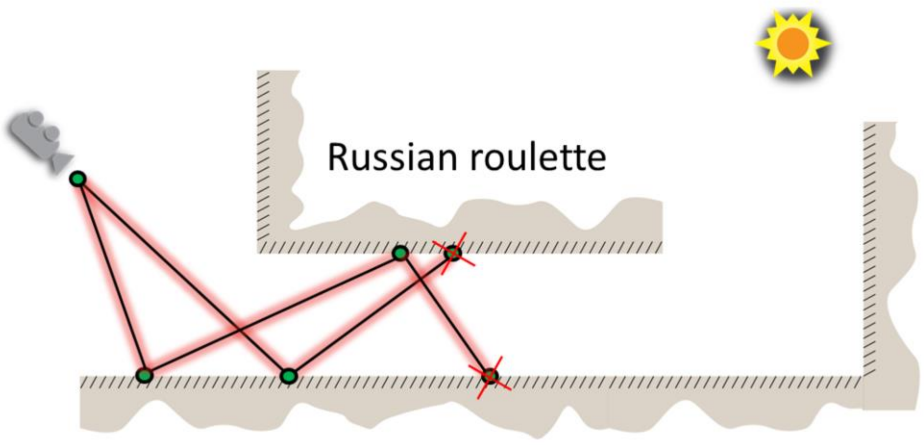
Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 30

Now I will present the main downsides of the current approaches.

It turns out that termination/splitting factor based on local reflectance properties is often sub-optimal.

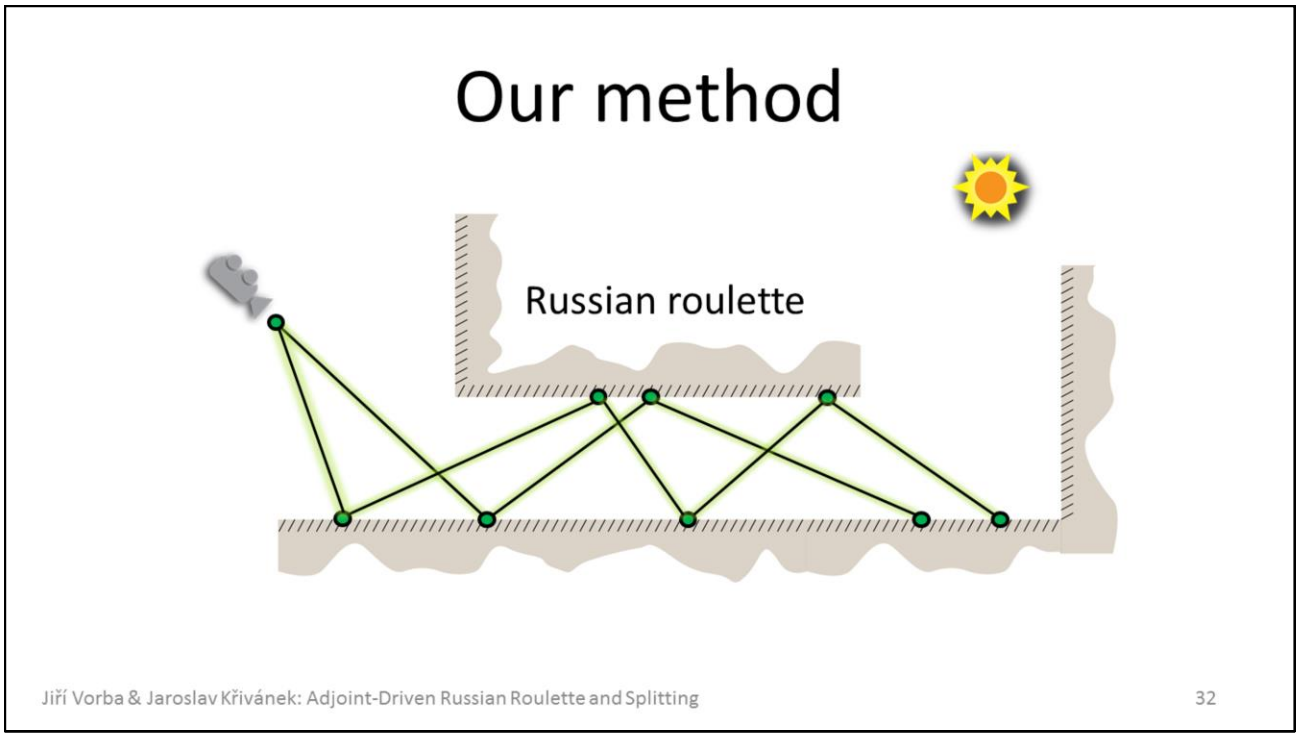


Issues



While RR saves time by terminating paths with small throughput it may tremendously increase variance per path if only a tiny fraction of them will have non-zero contribution.

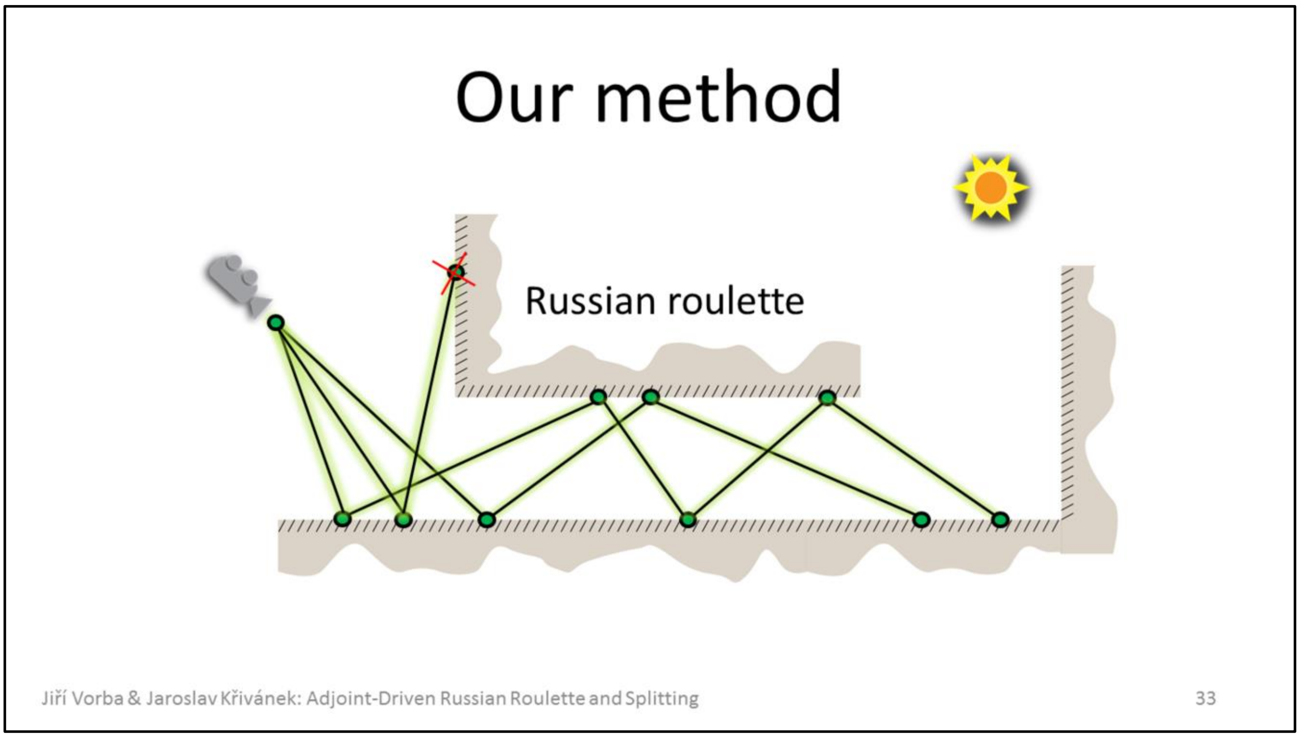
This is particularly pressing in scenes with strong indirect illumination that is reflected several times of materials with rather low albedo.



Our method

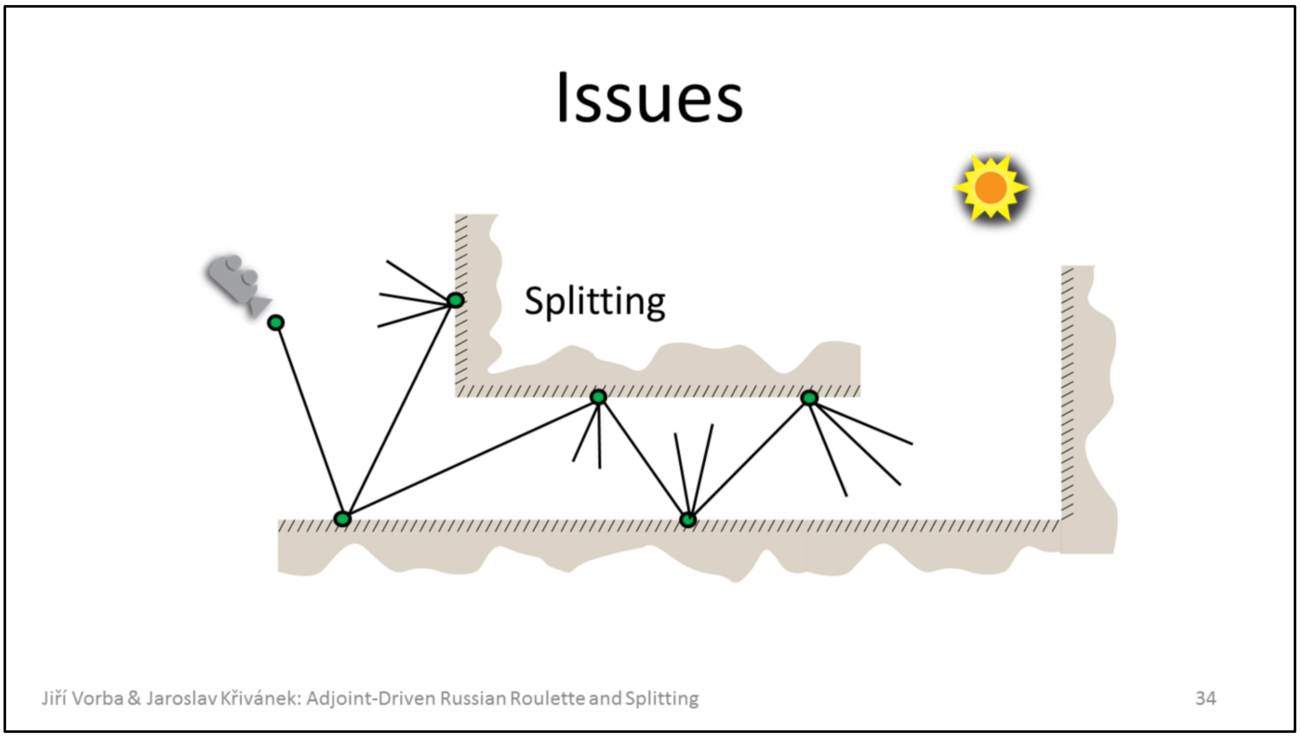
Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 32

In contrast, our method allows to survive paths that are heading into important regions close to light sources.



Our method

And its still capable of terminating paths that have very small expected contribution and their further tracing would be inefficient.



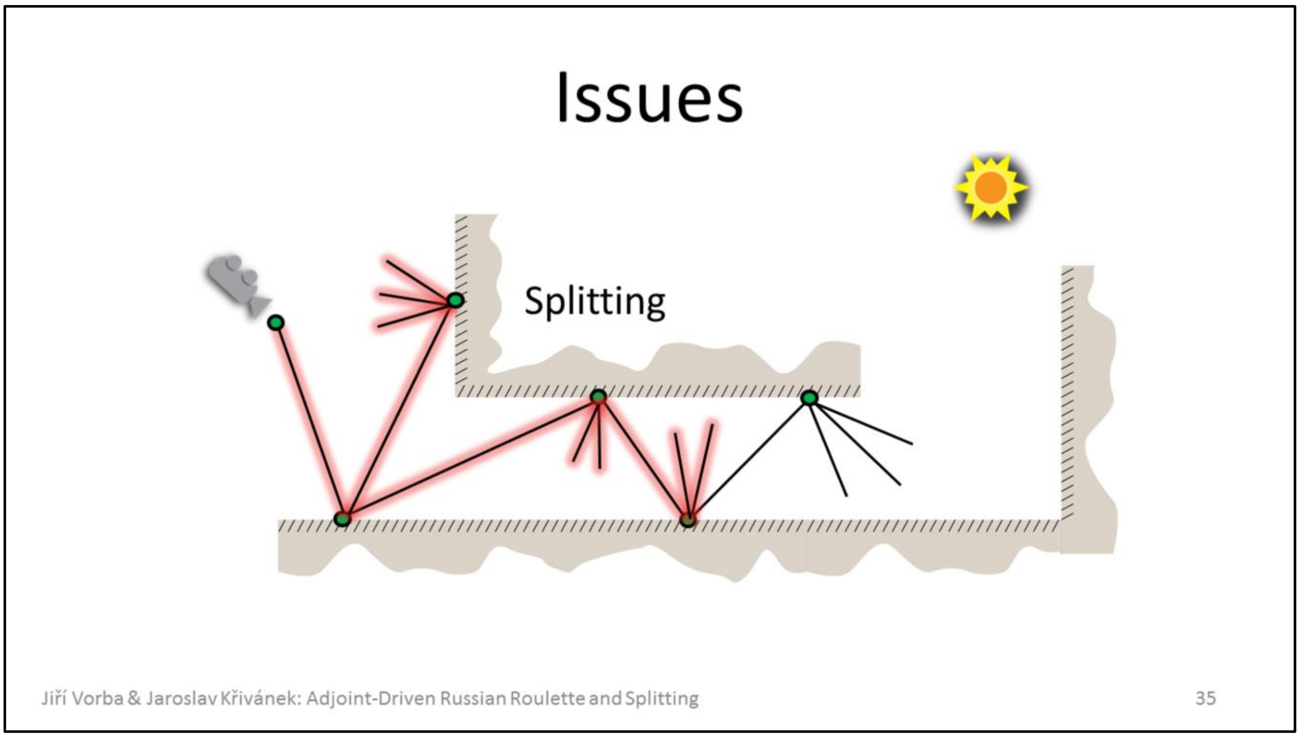
Issues

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 34

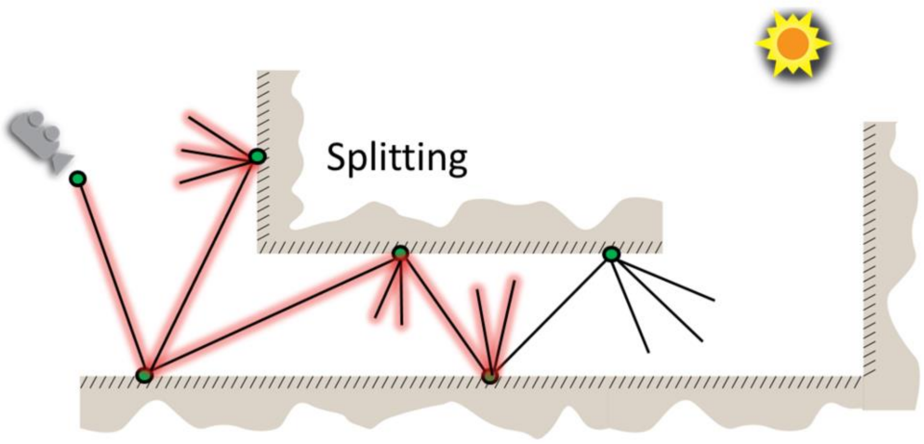
Splitting in general is a good thing.

It decreases variance per primary ray cast from the camera.

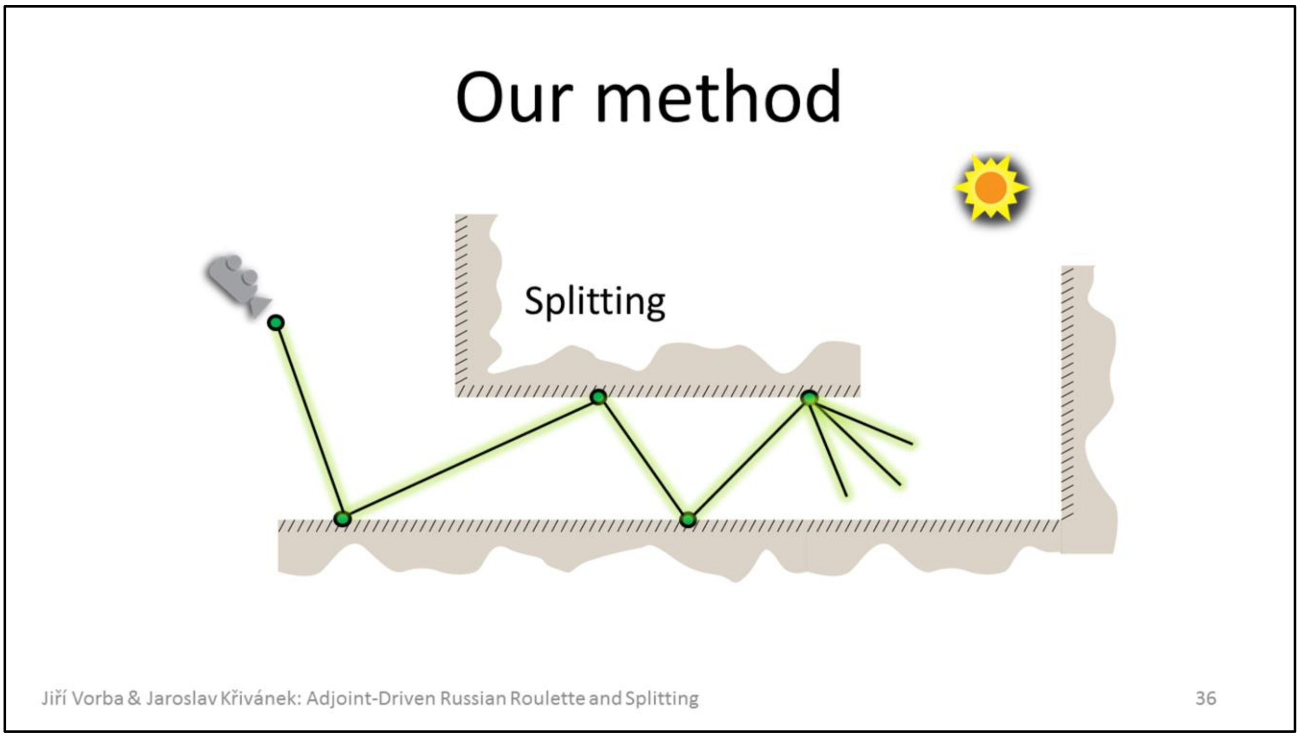
But this brings additional cost due to tracing more rays.



Issues



When splitting is based on local material properties, this cost becomes too high because paths are split also in regions where the expected path contribution is small.

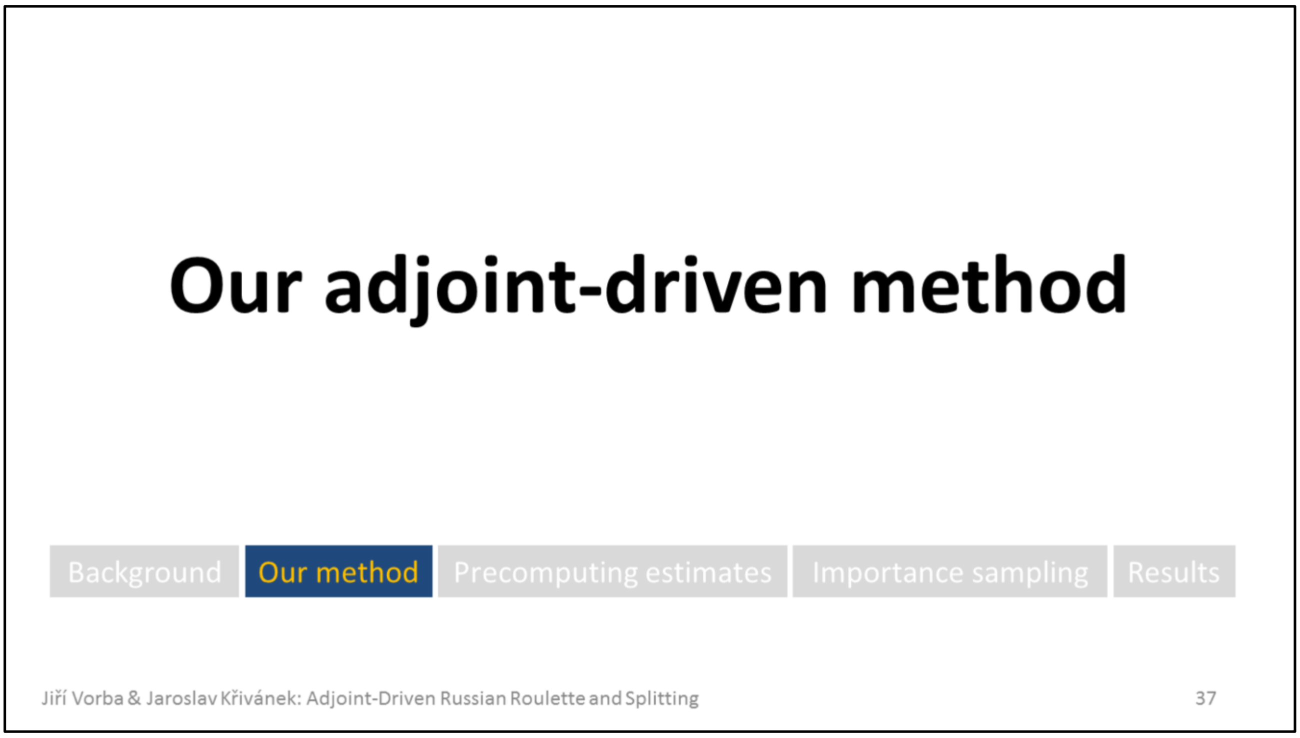


Our method

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 36

In contrast, our method starts splitting a path when it detects that its expected contribution is high.

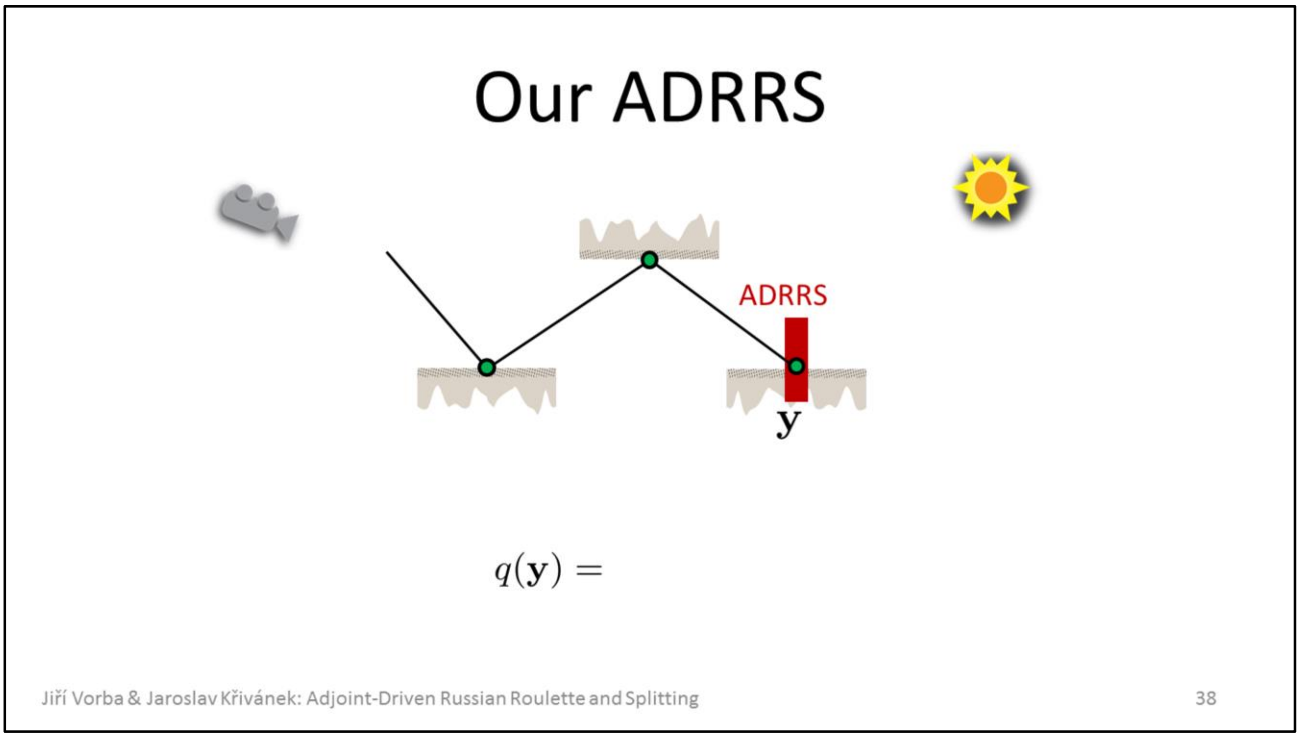
This simply means that path has made it into important region and we need to focus there the effort of our simulation.



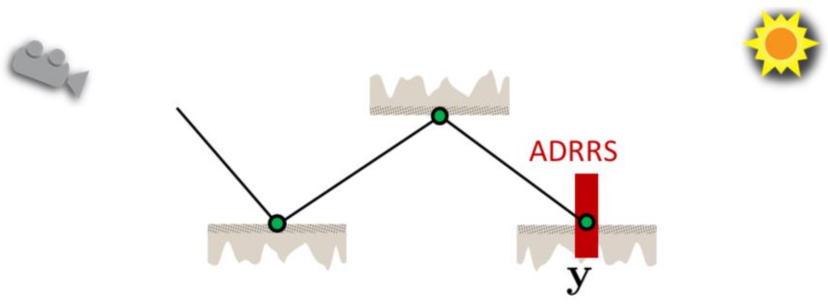
Our adjoint-driven method

Background | **Our method** | Precomputing estimates | Importance sampling | Results

I'm going to talk about our adjoint-driven Russian roulette and splitting in more detail.



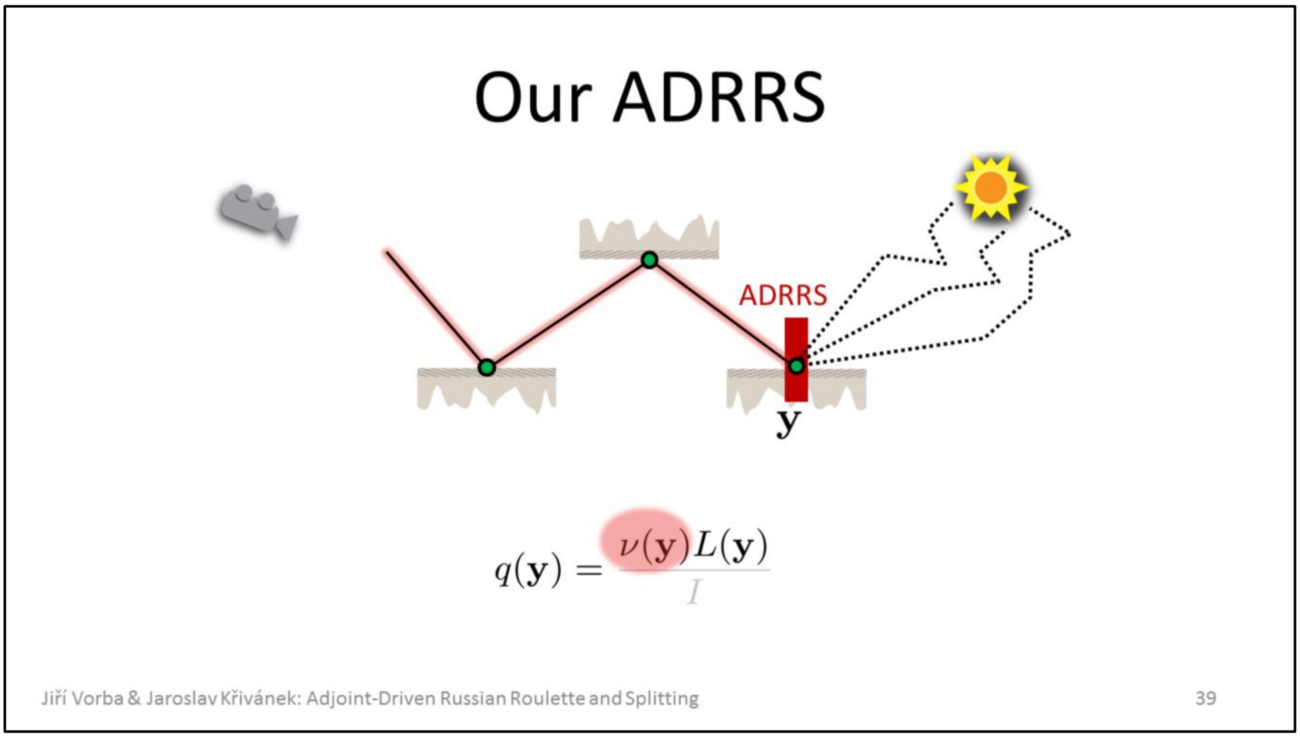
Our ADRRS



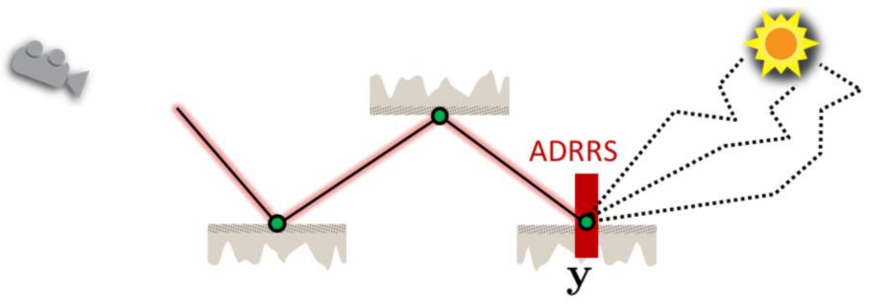
$q(\mathbf{y}) =$

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 38

The heart of our method is in computing the termination/splitting factor q at each vertex.



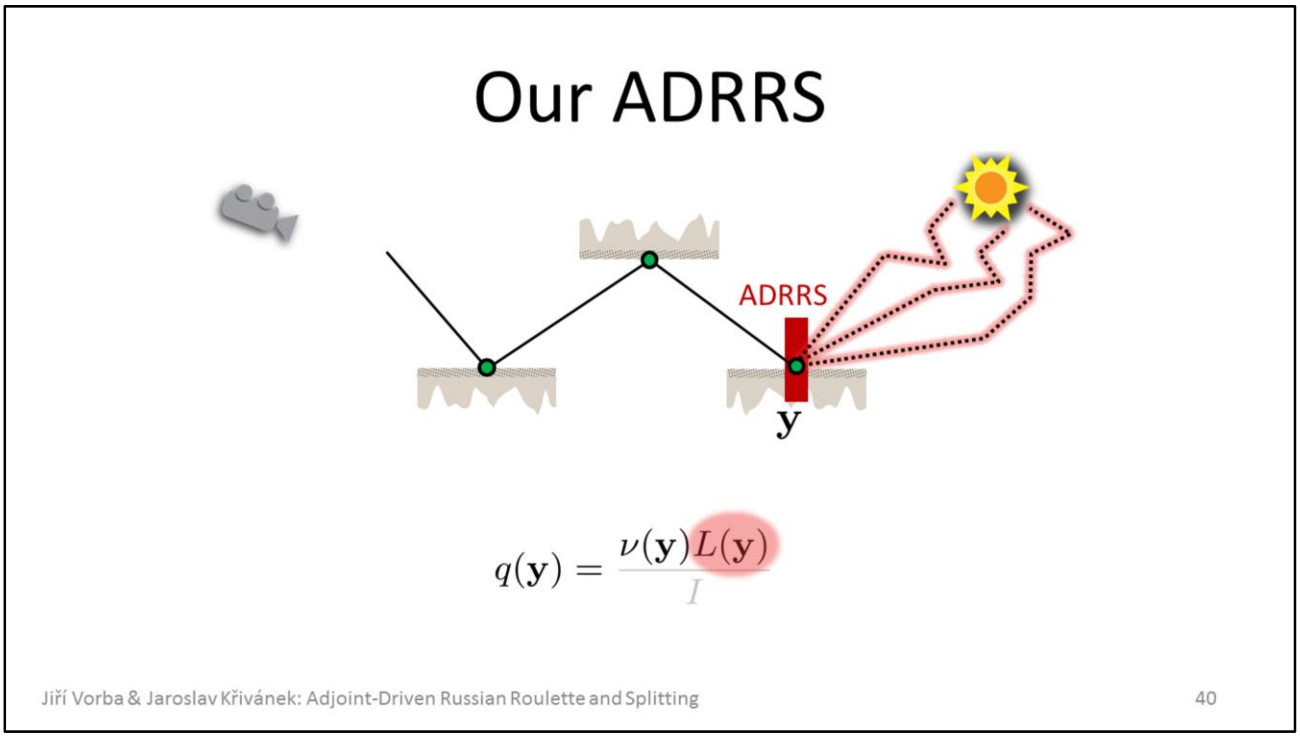
Our ADRRS



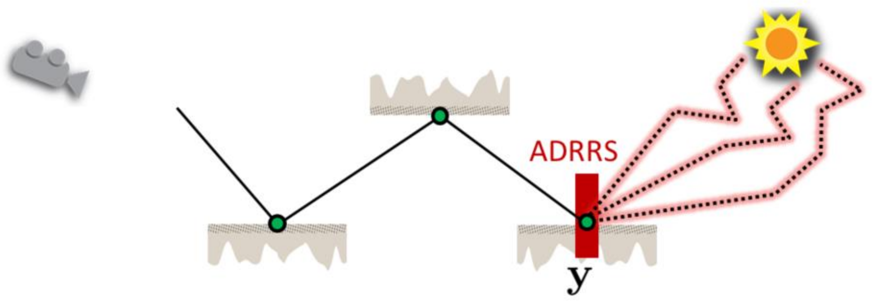
$$q(\mathbf{y}) = \frac{\nu(\mathbf{y})L(\mathbf{y})}{I}$$

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 39

Unlike previous methods we consider both history of the path in the form of its weight …



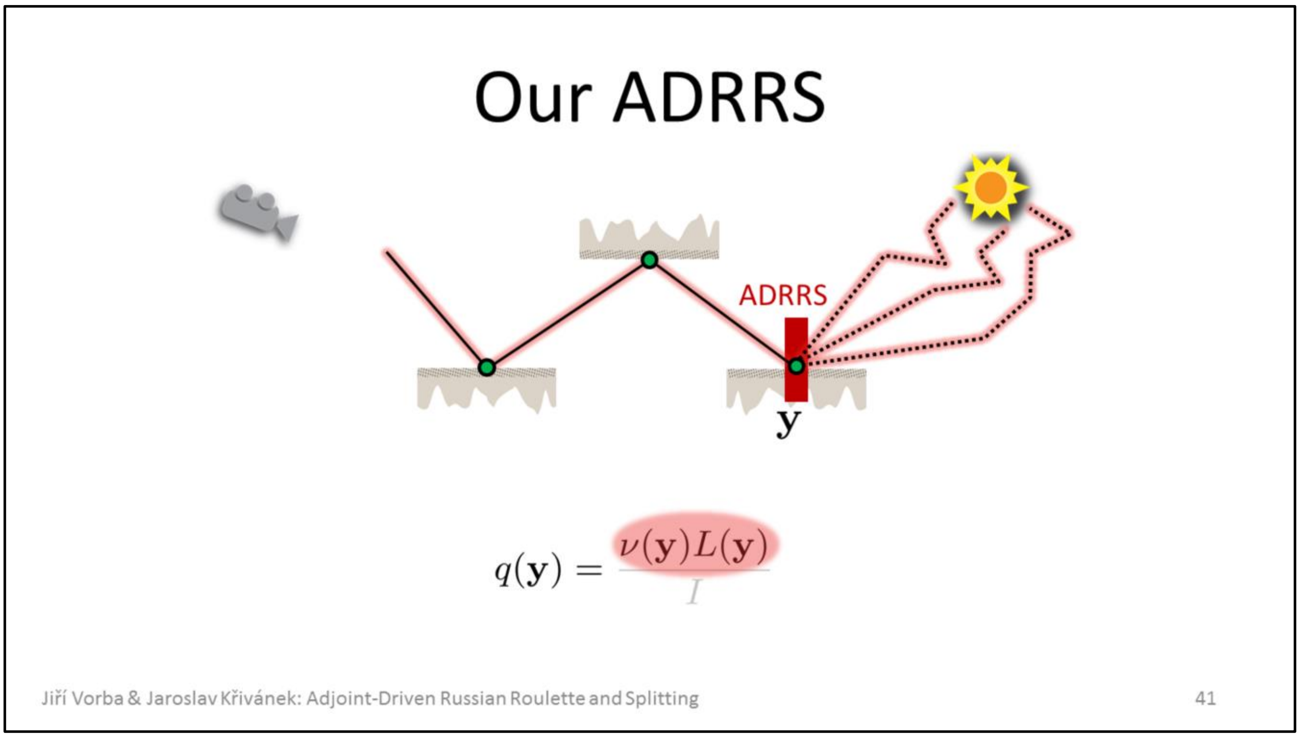
Our ADRRS



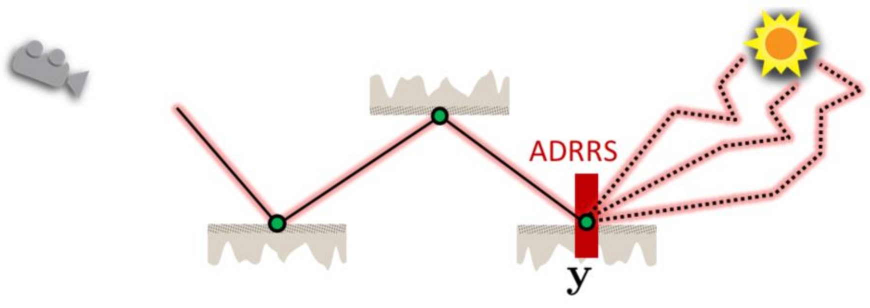
$$q(\mathbf{y}) = \frac{\nu(\mathbf{y})L(\mathbf{y})}{I}$$

… and also its future in the form of reflected radiance at y.

Note that the true value of radiance accounts for all possible path trajectories that can follow the current scattering event at y.



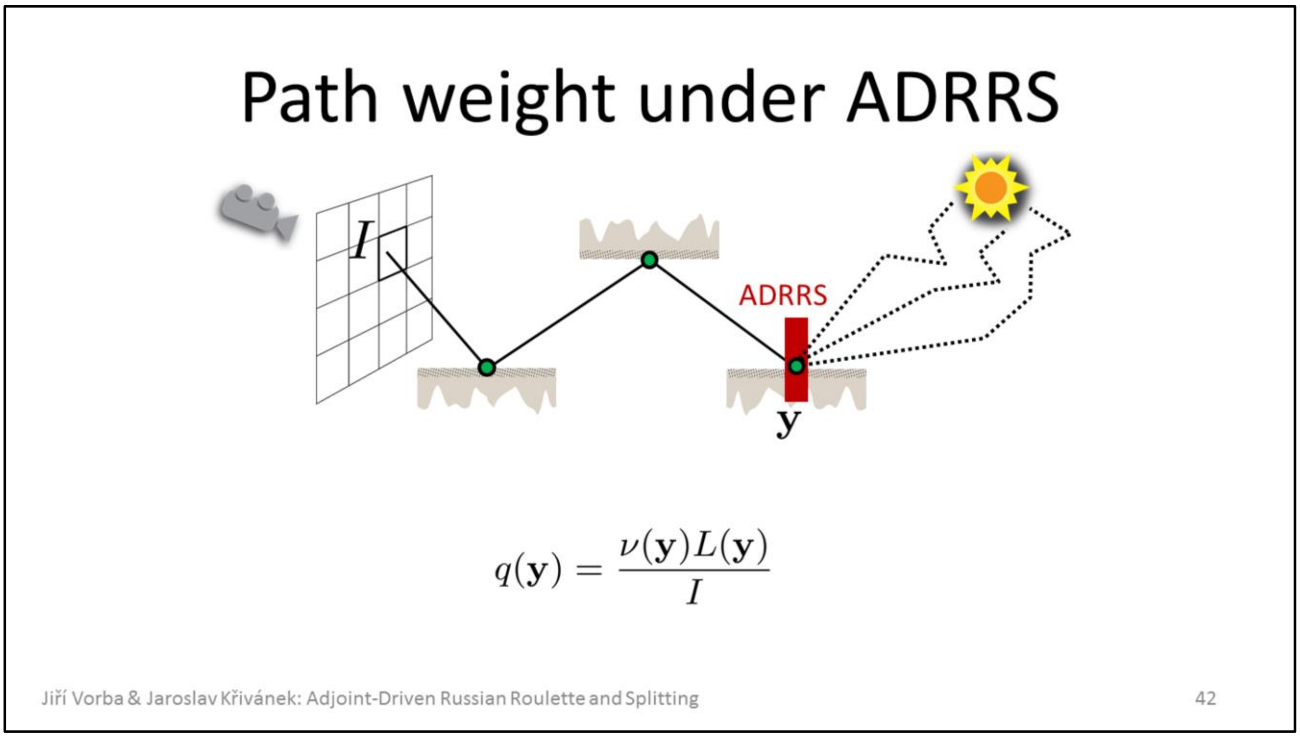
Our ADRRS



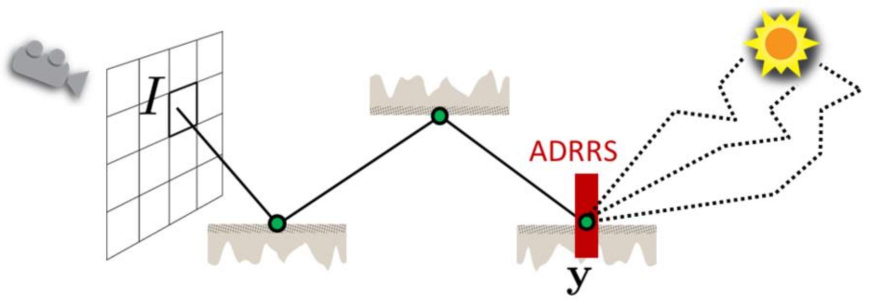
$$q(\mathbf{y}) = \frac{\nu(\mathbf{y})L(\mathbf{y})}{I}$$

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 41

Their product is the expected contribution that forms an unbiased estimate of pixel value I.



Path weight under ADRRS



$$q(\mathbf{y}) = \frac{\nu(\mathbf{y})L(\mathbf{y})}{I}$$

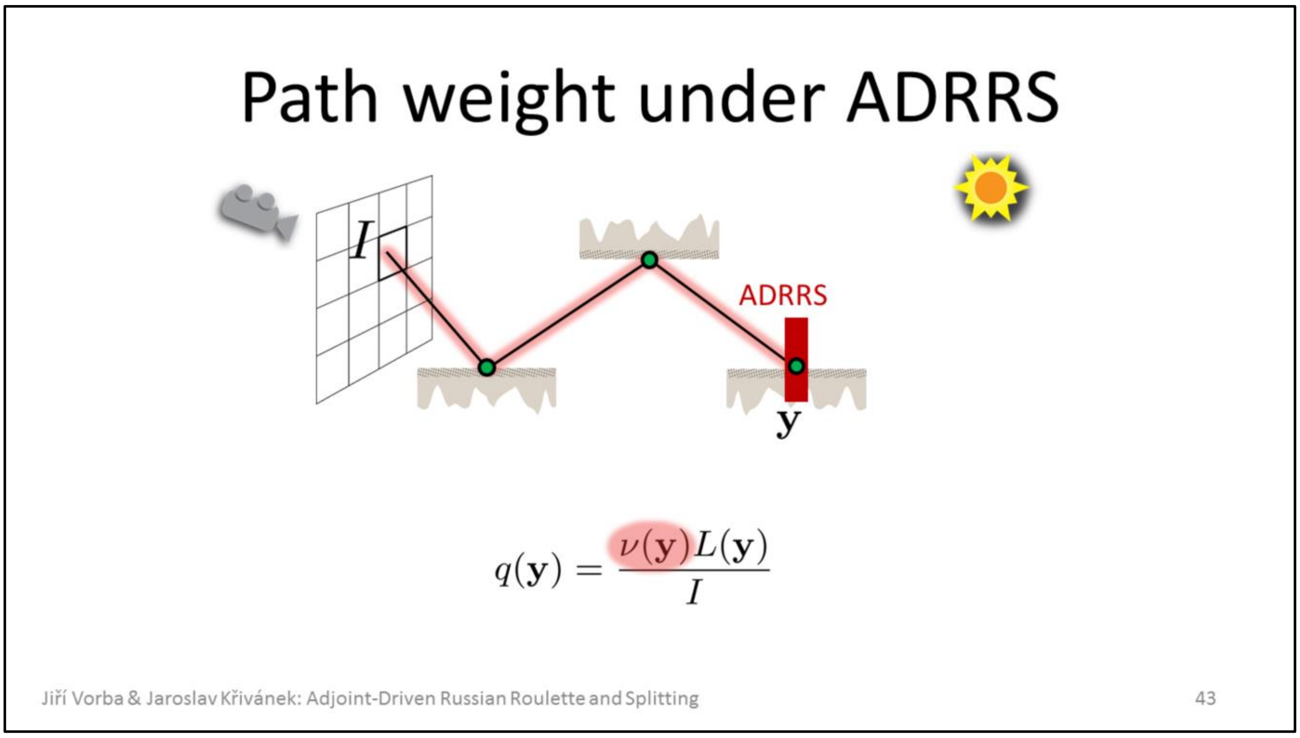
To obtain the termination/splitting factor we directly compare the expected contribution to the true pixel value associated with the currently traced path.

Note that the smaller the expected contribution the higher is the probability of terminating the path.

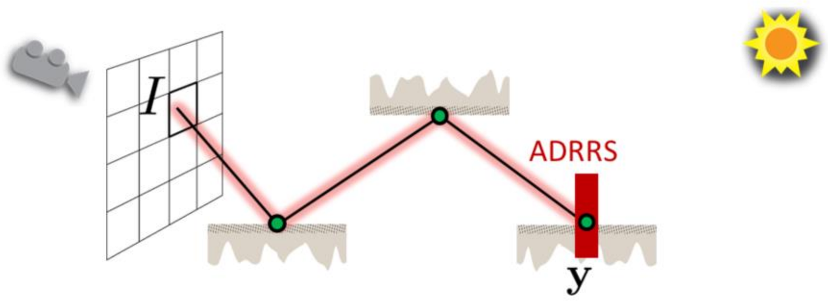
And vice versa: if the expected contribution is higher than the pixel value we will split the path.

To make it work we need to obtain rough estimates of the pixel value I and the radiance field in the scene.

I will get back to them later.

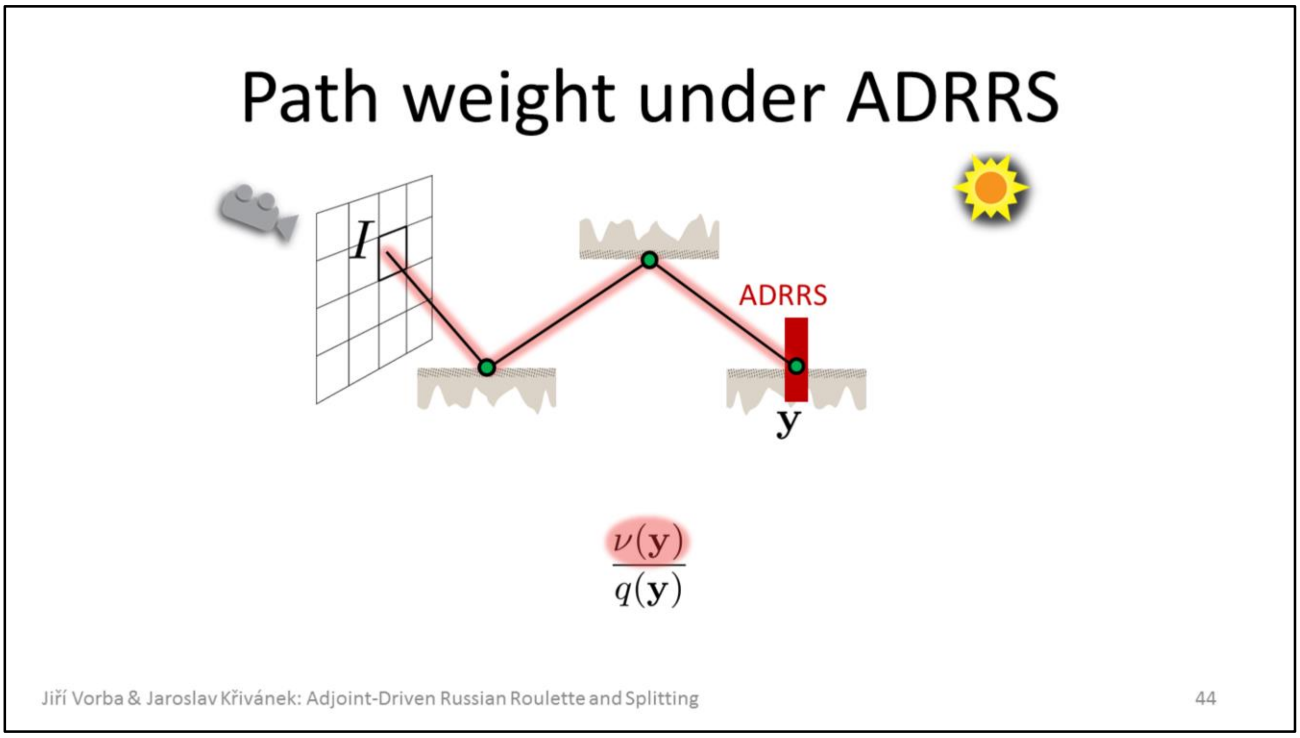


Path weight under ADRRS

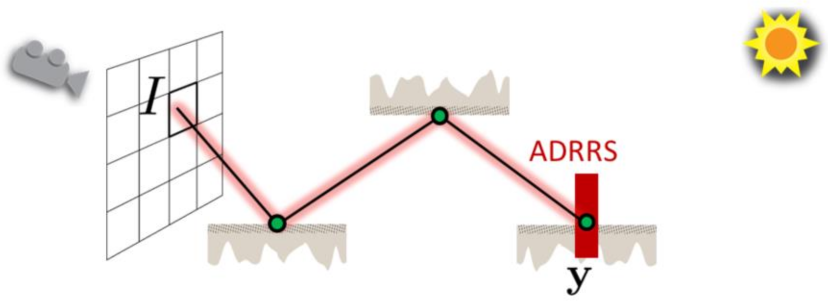


$$q(\mathbf{y}) = \frac{\nu(\mathbf{y})L(\mathbf{y})}{I}$$

Now I want to show what happens with the path weight under application of our ADRRS.

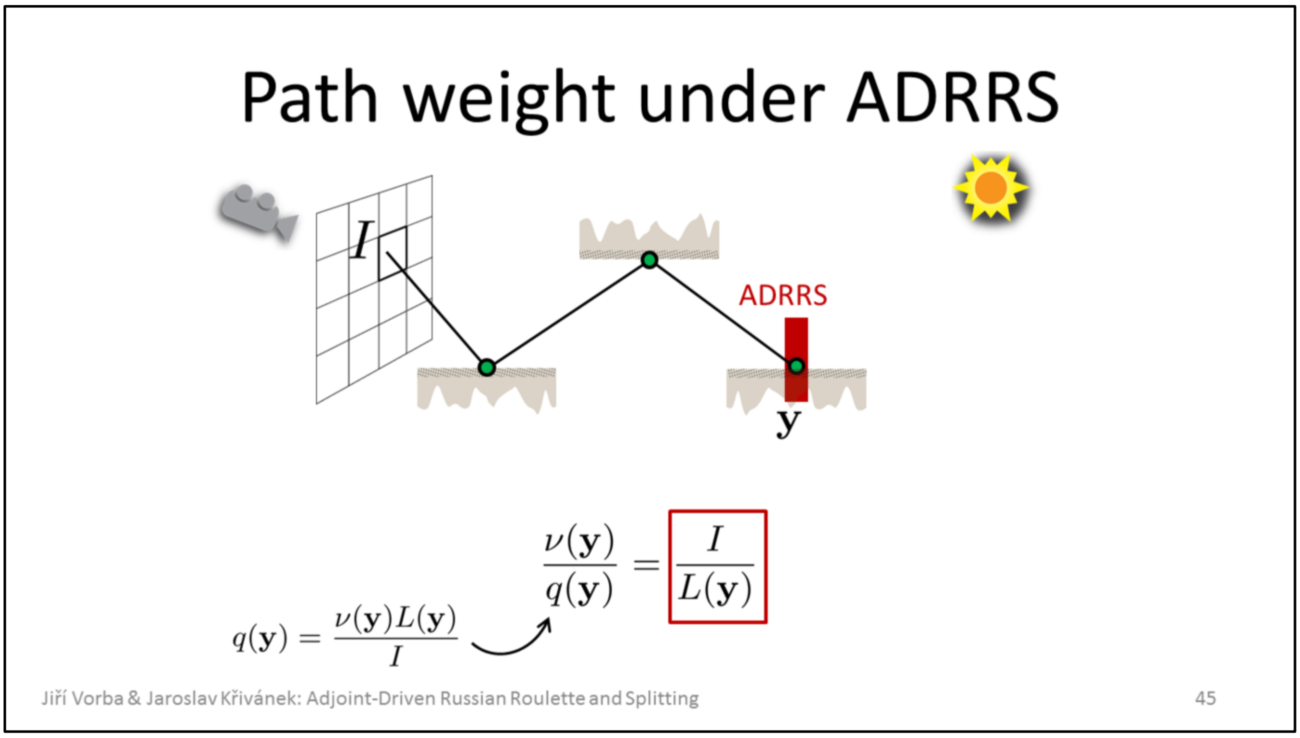


Path weight under ADRRS

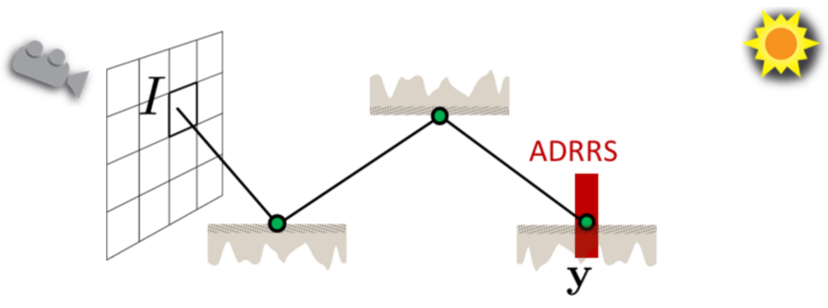


$$\frac{\nu(\mathbf{y})}{q(\mathbf{y})}$$

Recall that when a path survives or is split we need to divide the current path weight by the factor q.



Path weight under ADRRS



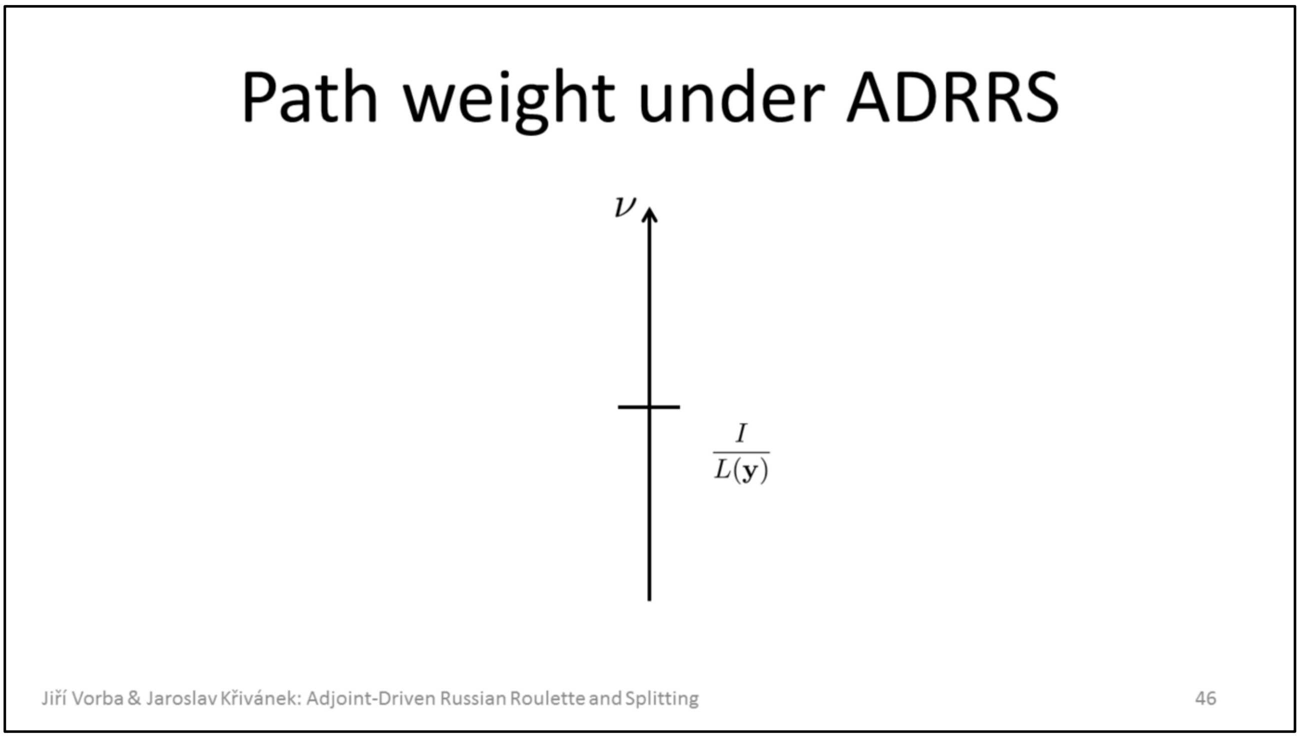
$$q(\mathbf{y}) = \frac{\nu(\mathbf{y})L(\mathbf{y})}{I} \curvearrowright \frac{\nu(\mathbf{y})}{q(\mathbf{y})} = \boxed{\frac{I}{L(\mathbf{y})}}$$

This results into the ratio of the pixel value I and the reflected radiance.

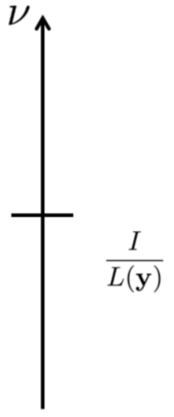
This is very important formula.

It shows that our ADRRS keeps this invariant in the whole simulation.

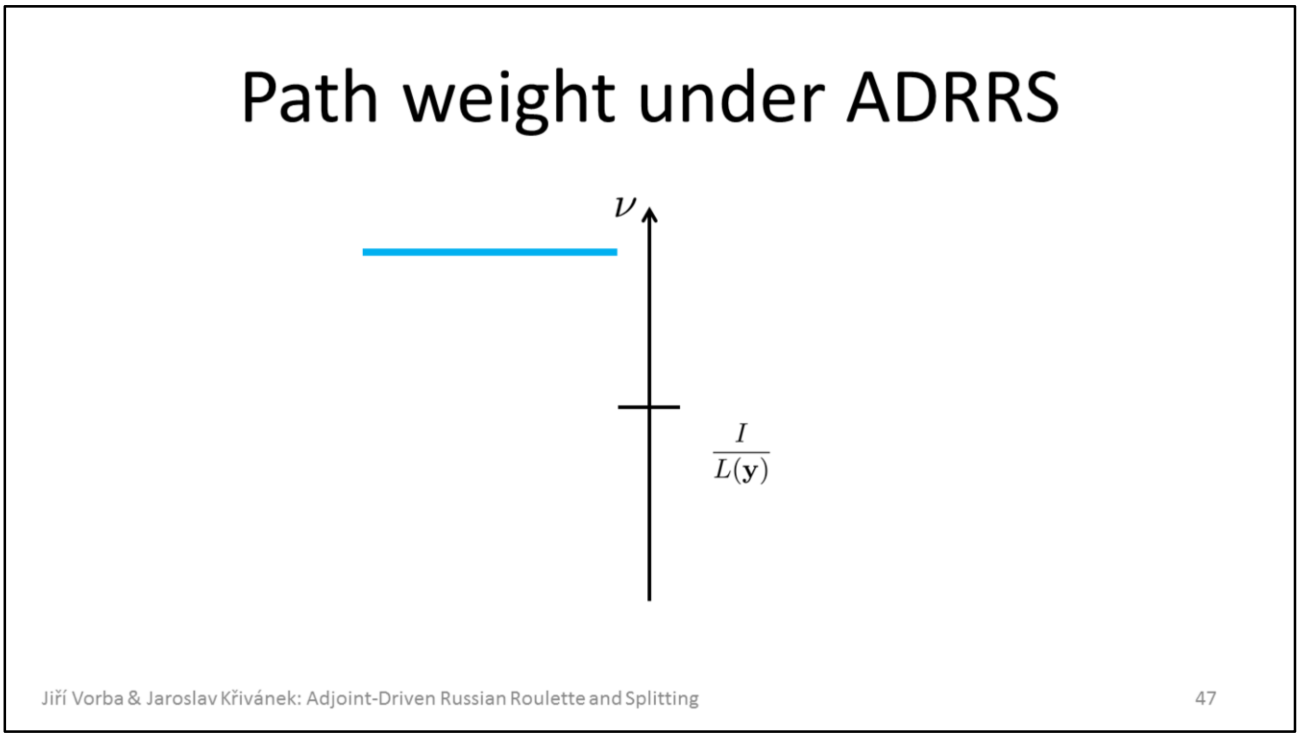
And if we hit a light source then the final contribution of each path will be equal to I.



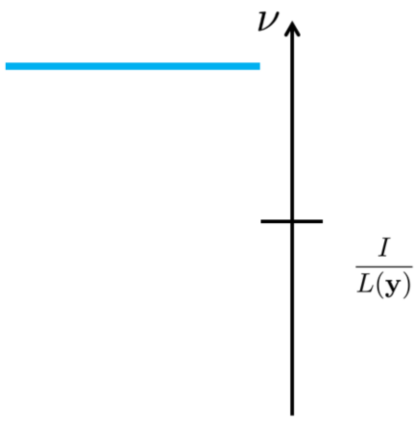
Path weight under ADRRS



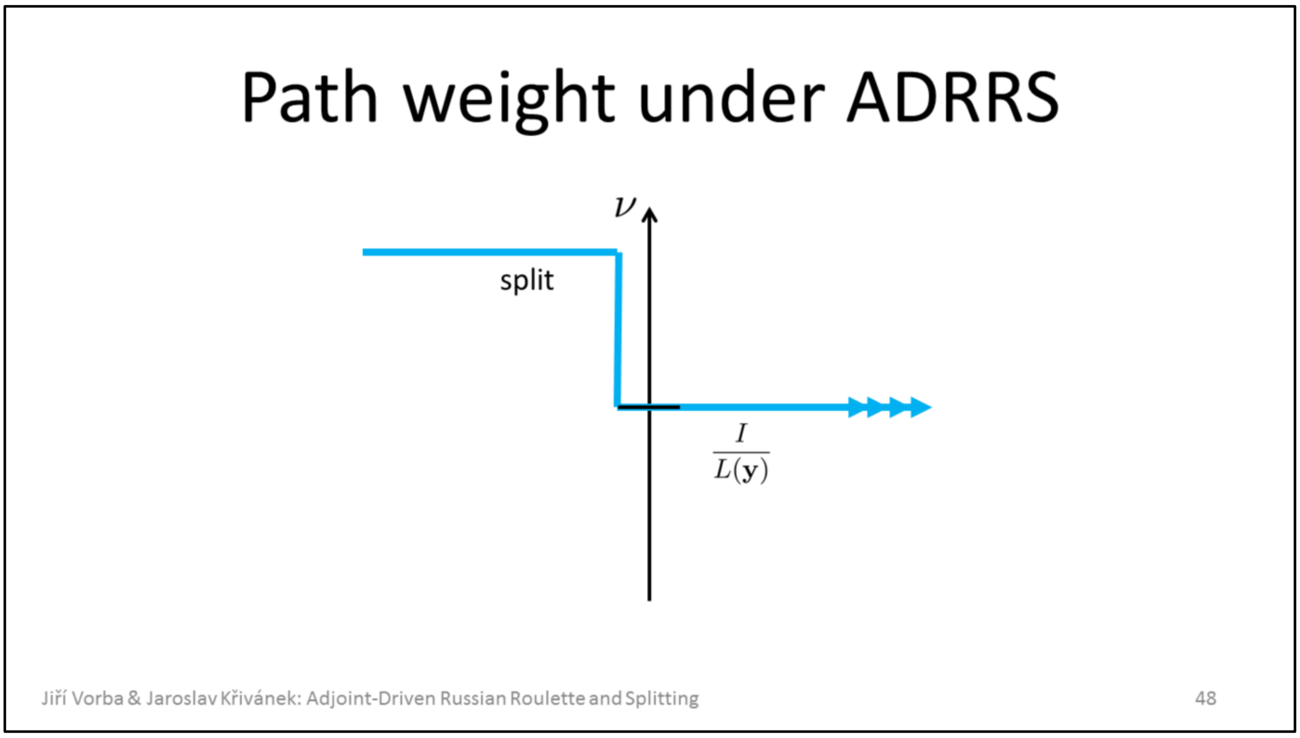
We can depict the behavior of our approach on this simple weight chart.



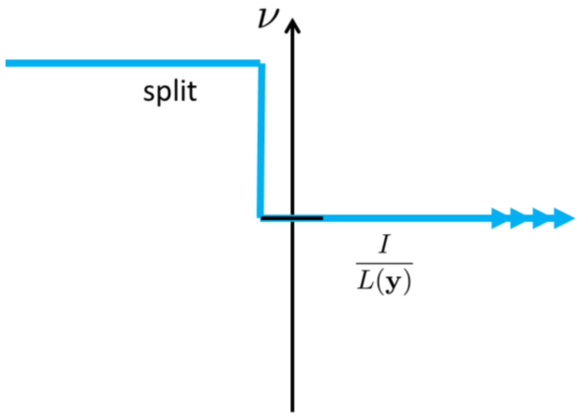
Path weight under ADRRS



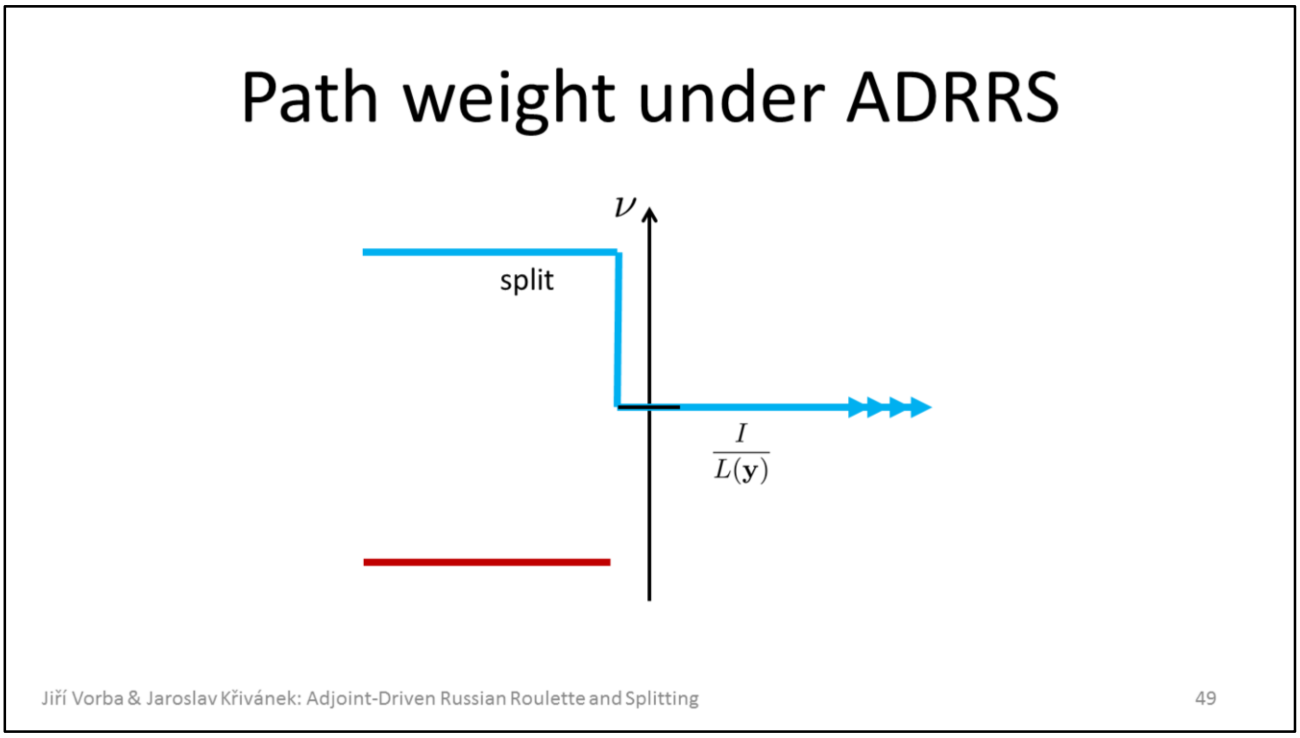
When the incoming weight is higher than the invariant ratio…



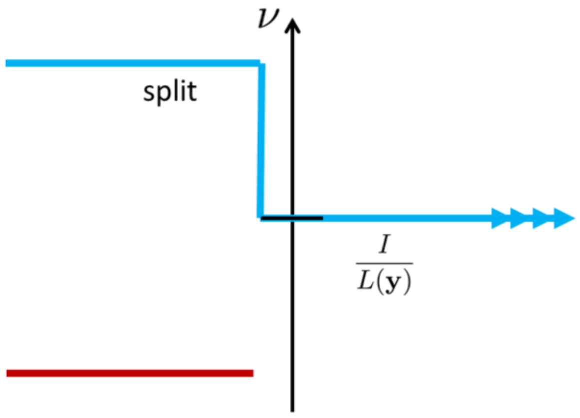
Path weight under ADRRS



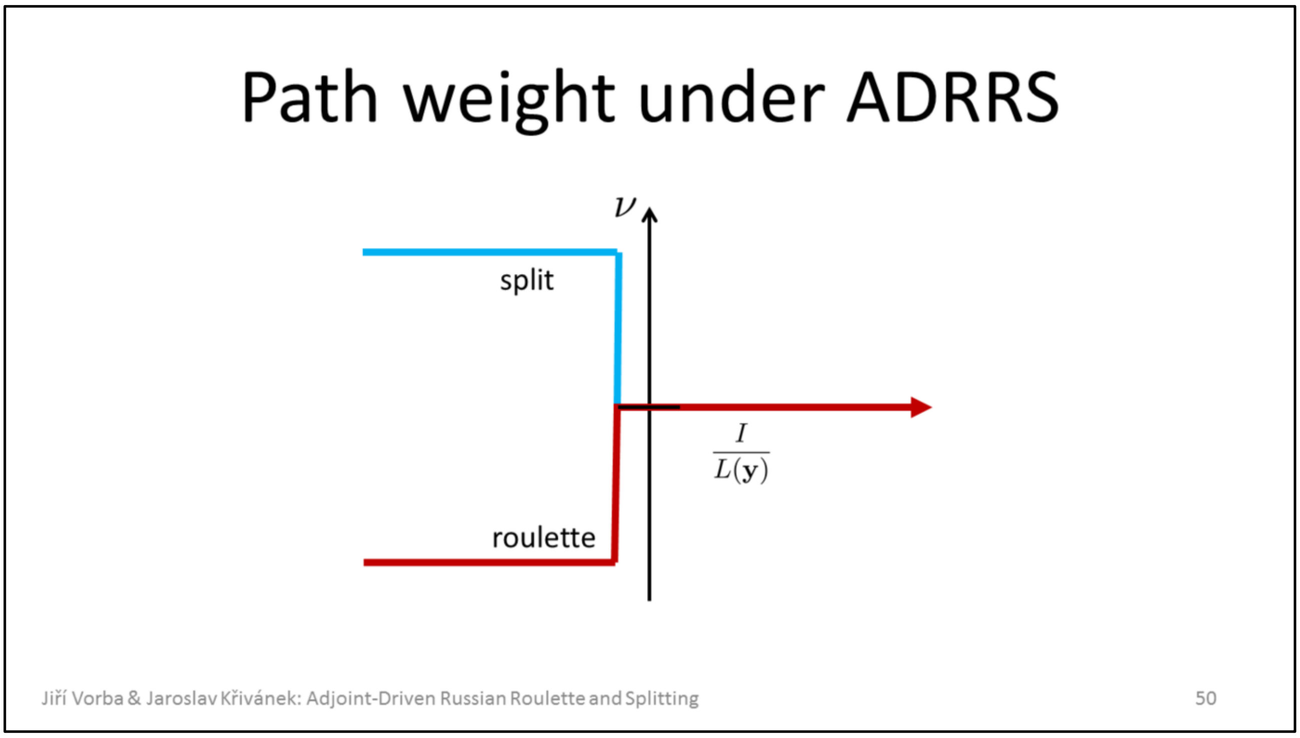
… we split the path and weight of split paths will achieve our invariant.



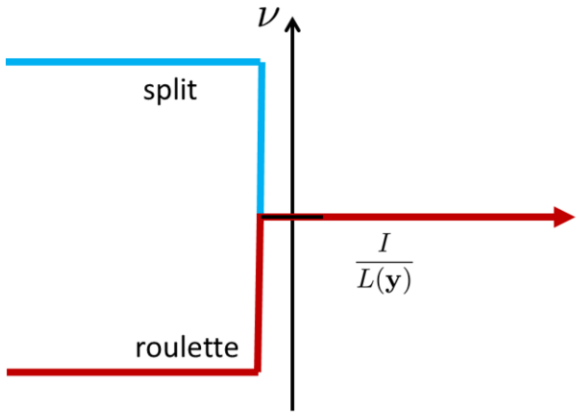
Path weight under ADRRS



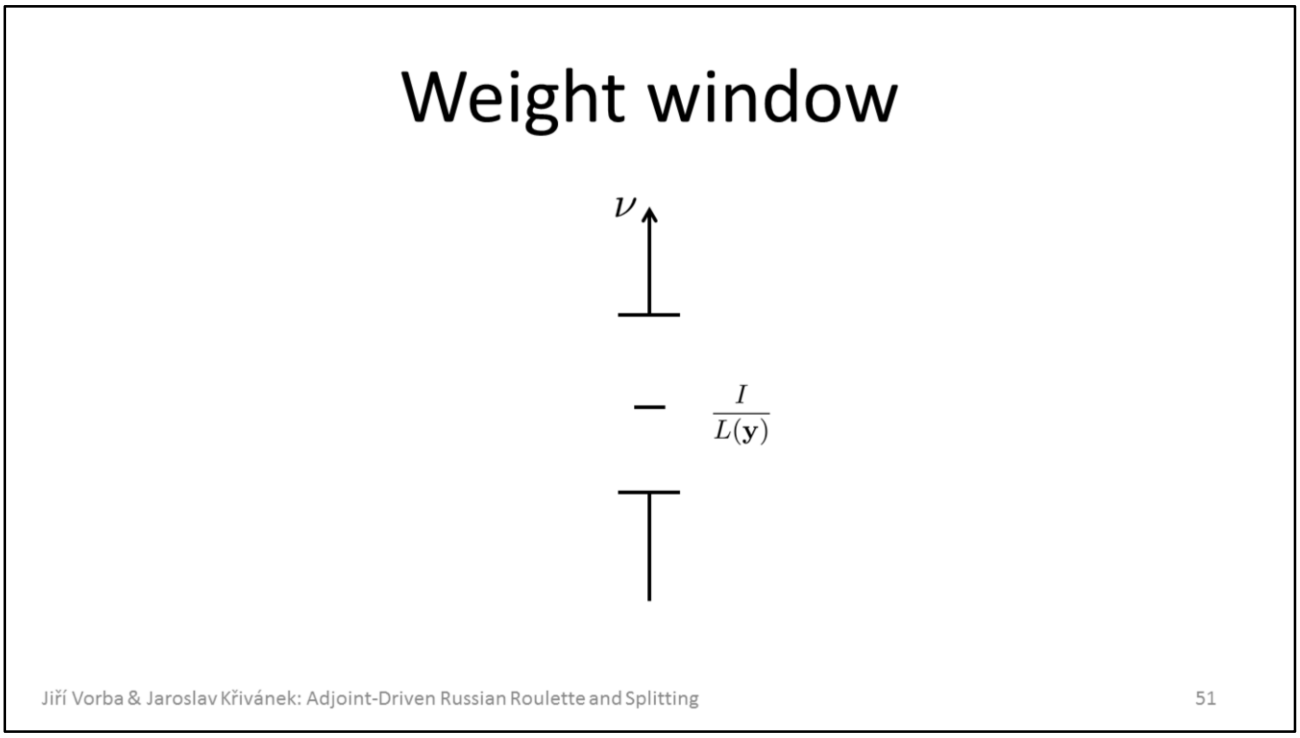
Similarly, when the weight is lower that the demanded invariant…



Path weight under ADRRS

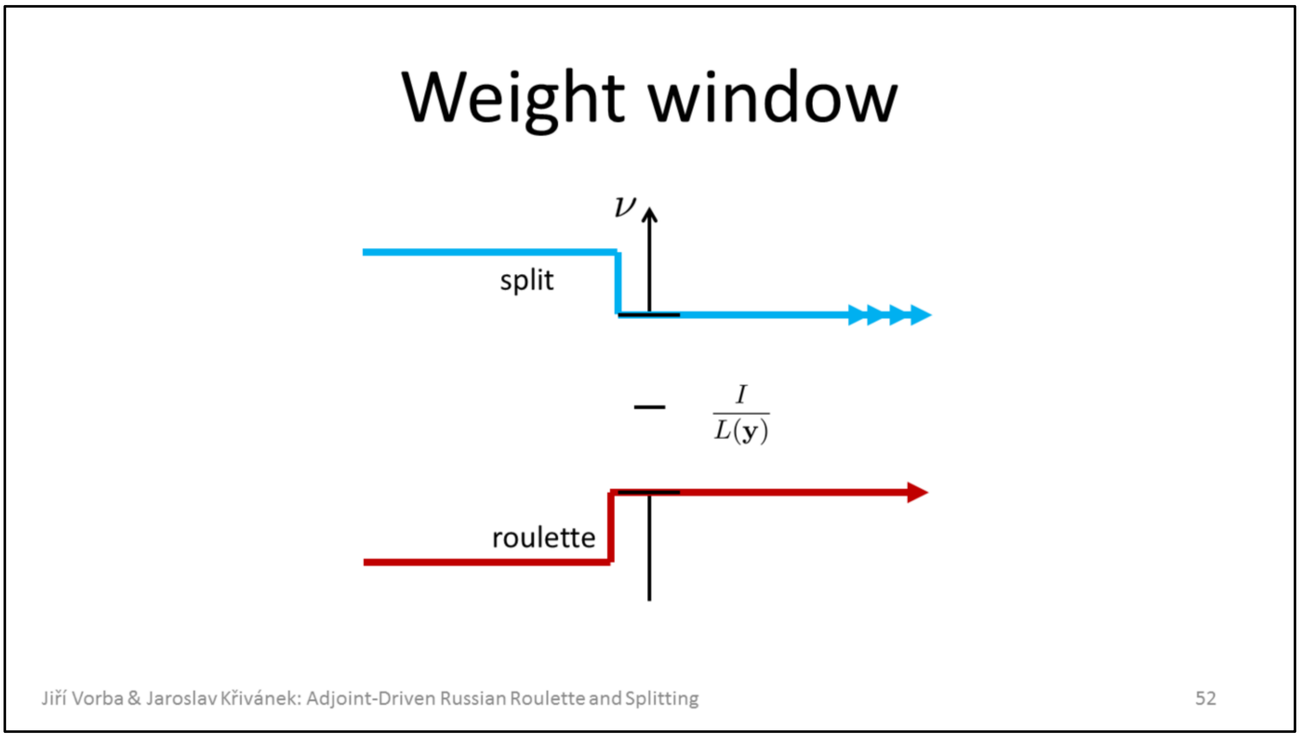


… we play Russian roulette.

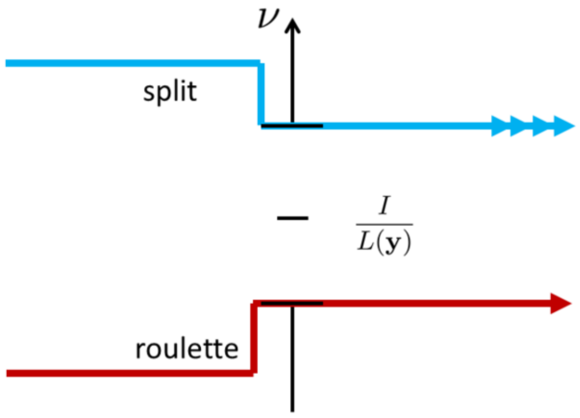


Weight window

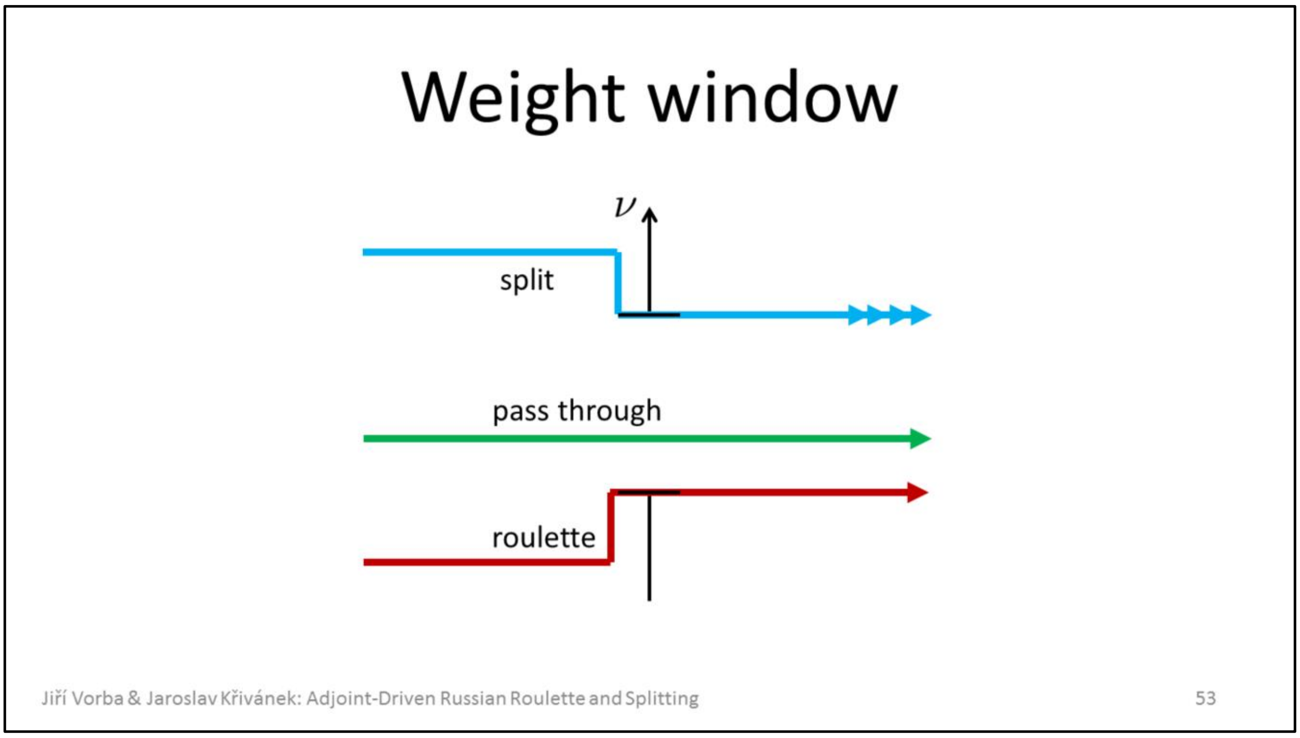
However, in our implementation we allow some leeway and we force the path weight to lie in an interval centered at the demanded weight ratio.



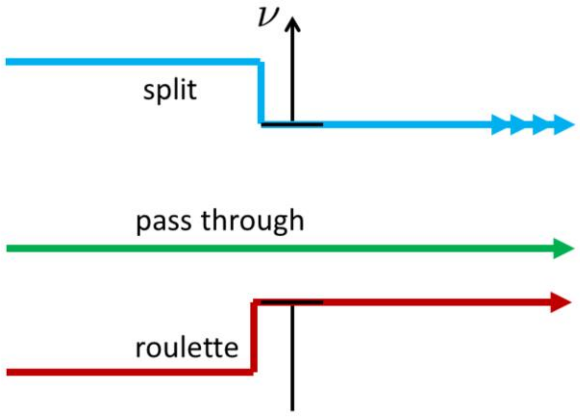
Weight window



so in fact we play the roulette so that surviving paths have weight at the lower boundary while we split paths so that the resulting weights are at the top boundary.

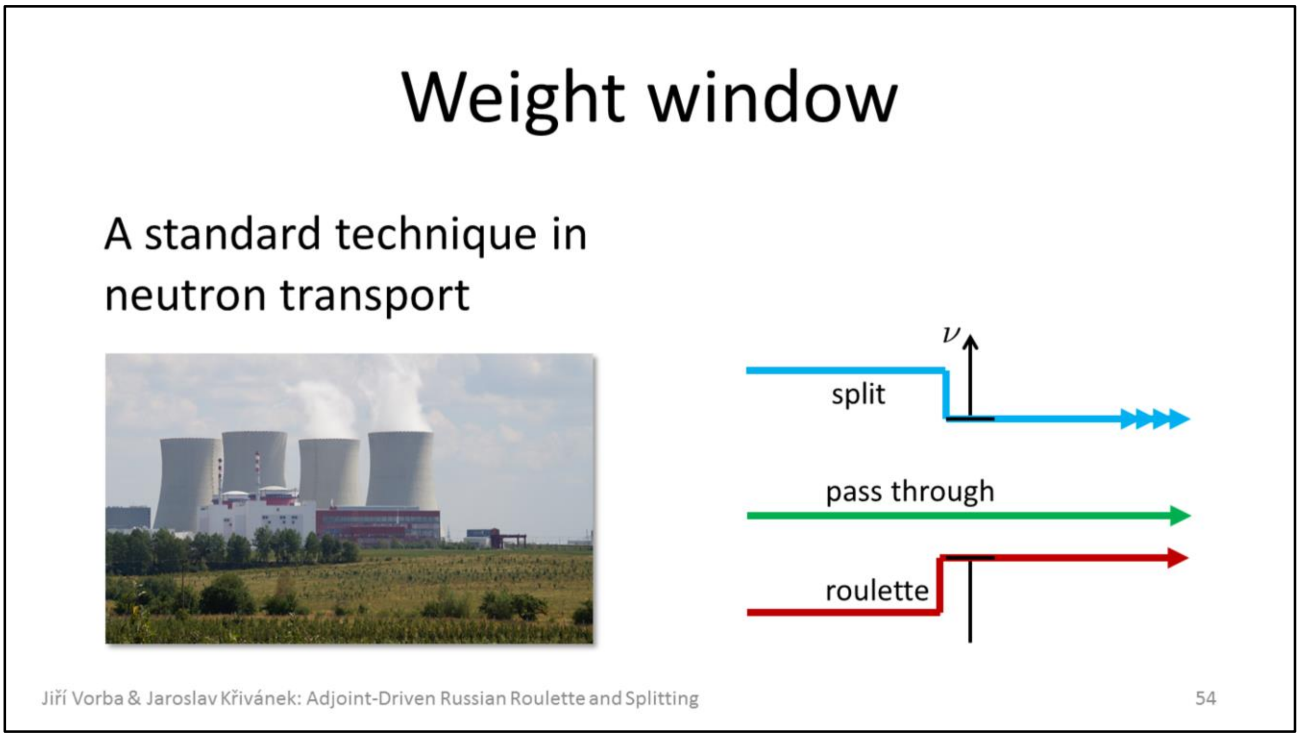


Weight window



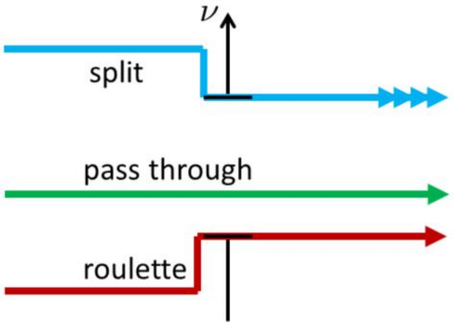
Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 53

If the path weight lies within the interval we let the path survive without playing roulette or splitting.



Weight window

A standard technique in neutron transport

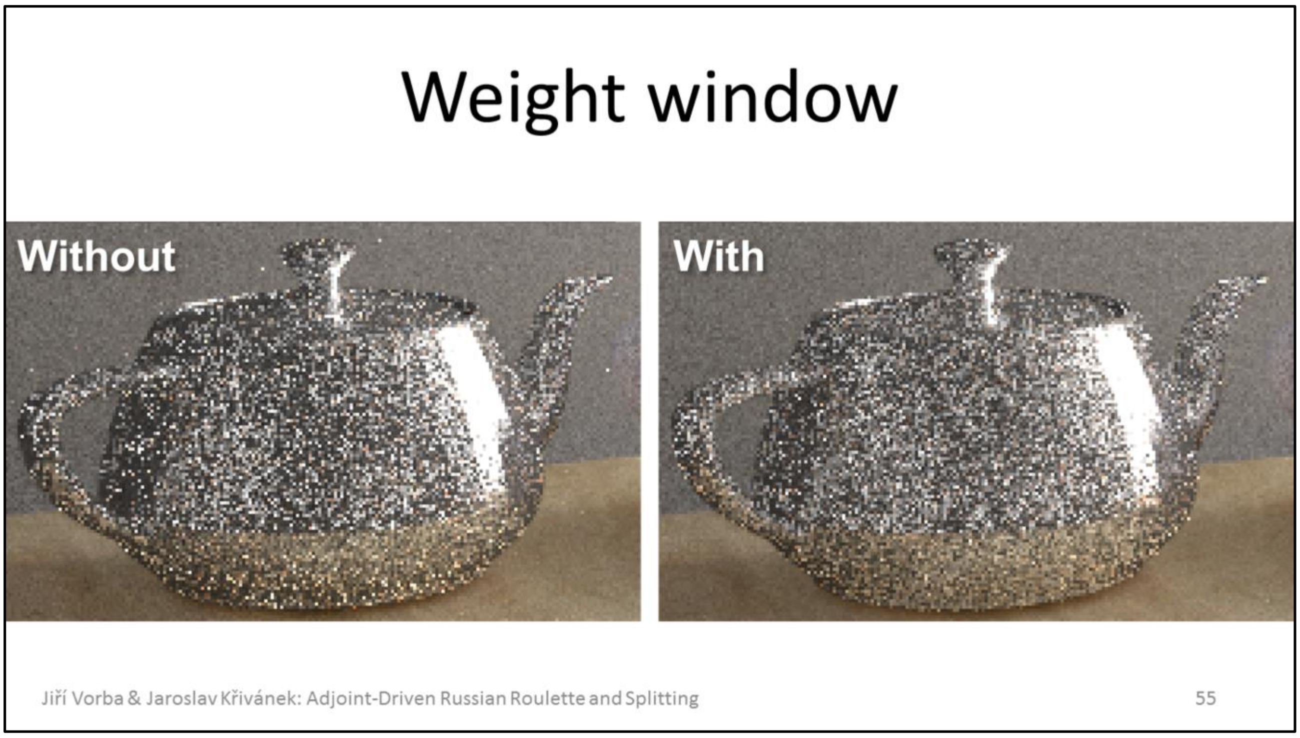


Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 54

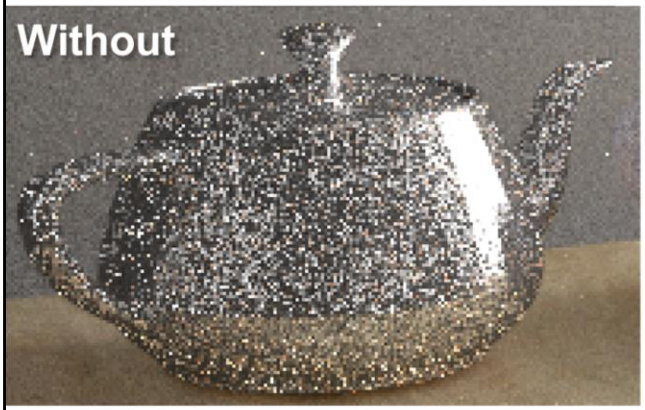
This technique called weight-window has been a standard technique in neutron transport simulations for about thirty years.

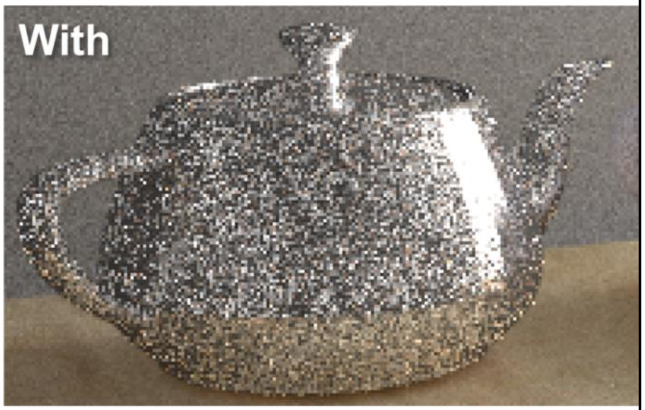
And it is a fundamental component of widely used simulation packages such as MCNP from Los Alamos national labs.

The purpose of weight window is to make our algorithm more robust to the inaccuracies of our measurement and radiance estimates.



Weight window

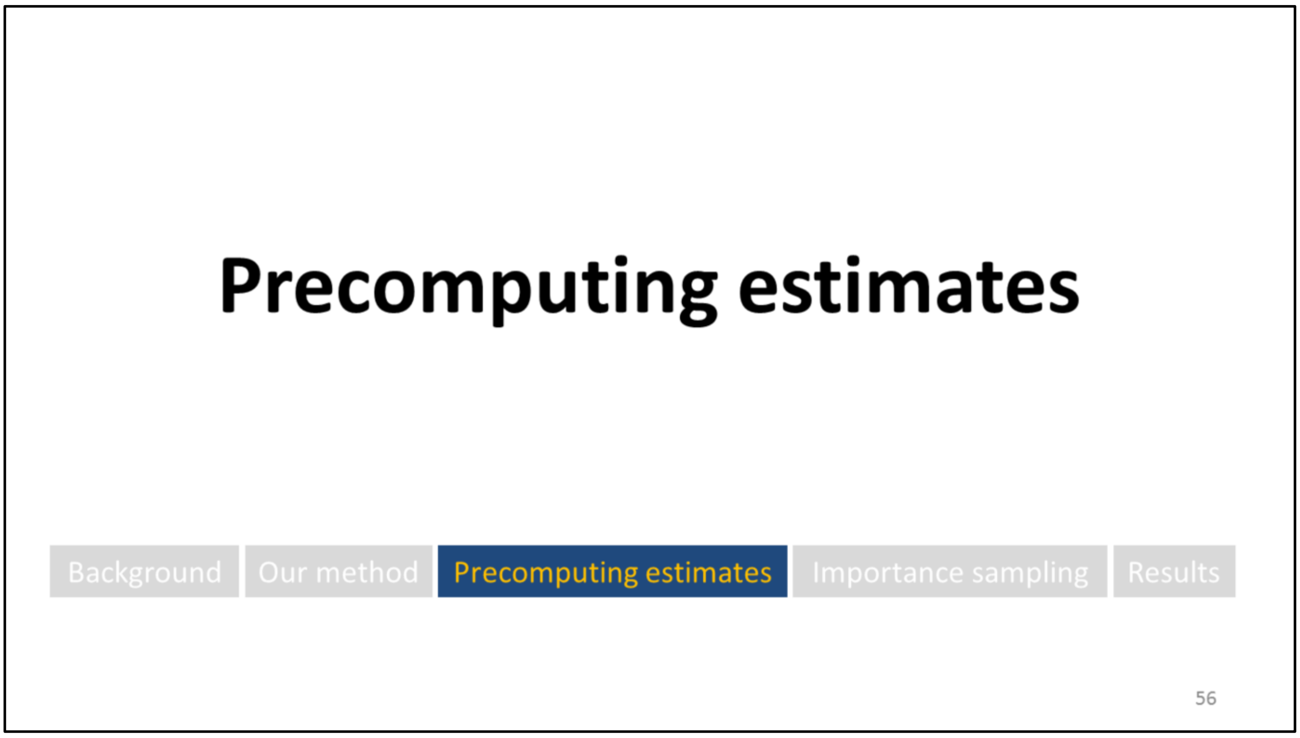




Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 55

There is a comparison of results without the weight window – the left image and with the weight window - right image.

We can see that the weight window prevented some high frequency noise around the tea pot and that for example the reflection of the table in the teapot is less noisy as well.

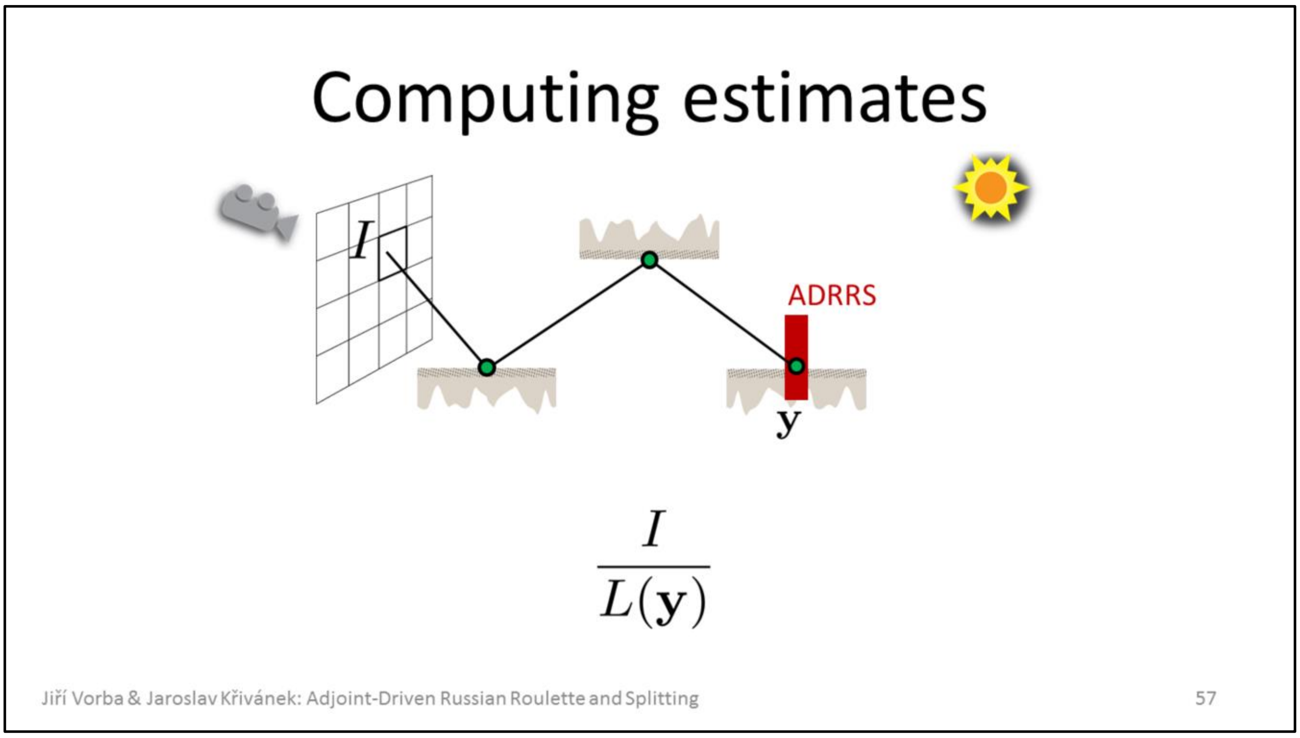


Precomputing estimates

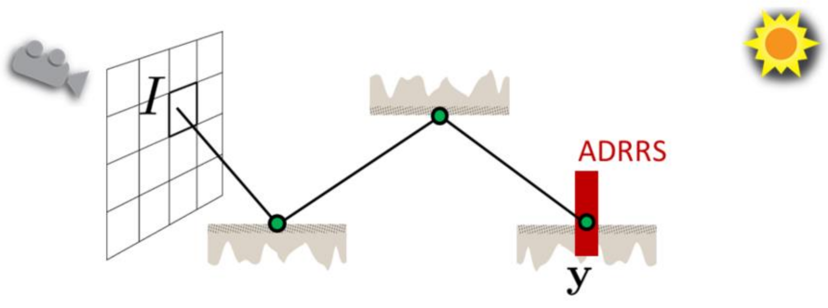
Background | Our method | Precomputing estimates | Importance sampling | Results

56

Now I am going to say more about how we compute the estimates necessary for setting-up a weight window.

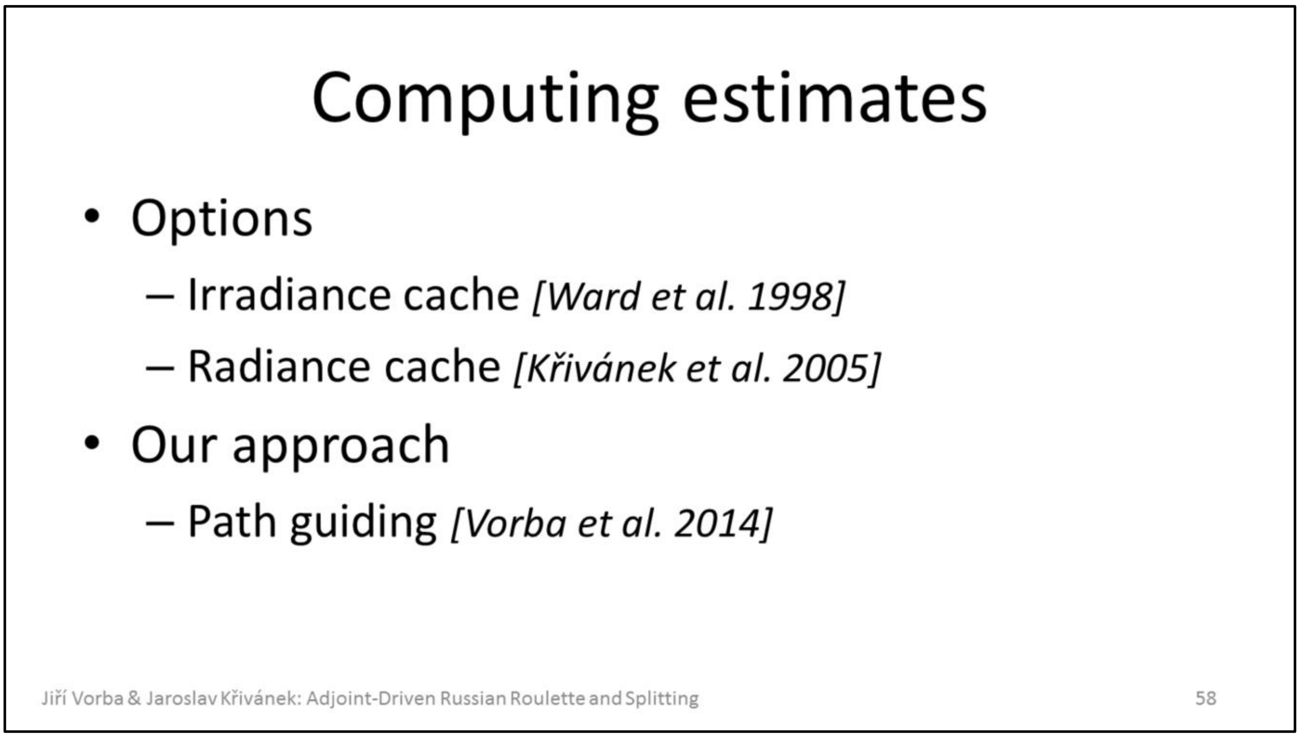


Computing estimates



$$\frac{I}{L(\mathbf{y})}$$

I will recall that to set up the weight window we need to estimate the pixel value I and the reflected radiance at y.



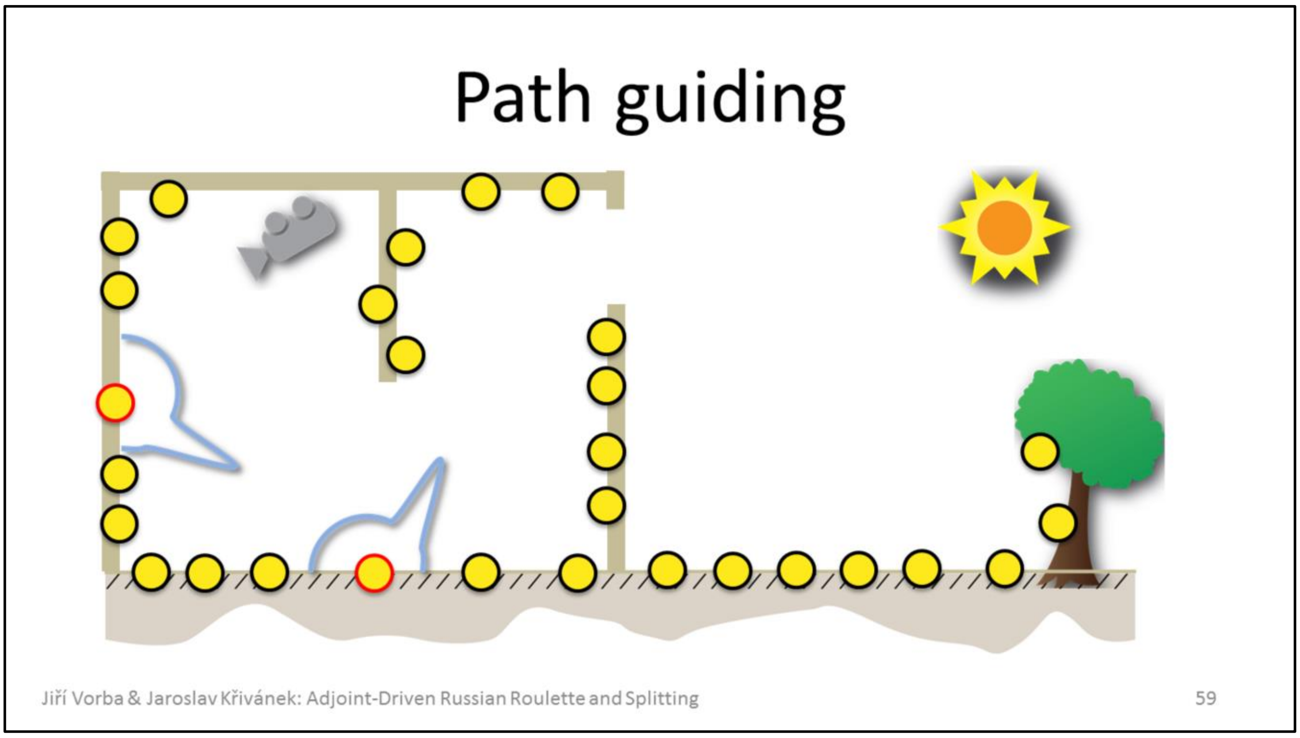
Computing estimates

- Options
  - Irradiance cache *[Ward et al. 1998]*
  - Radiance cache *[Křivánek et al. 2005]*
- Our approach
  - Path guiding *[Vorba et al. 2014]*

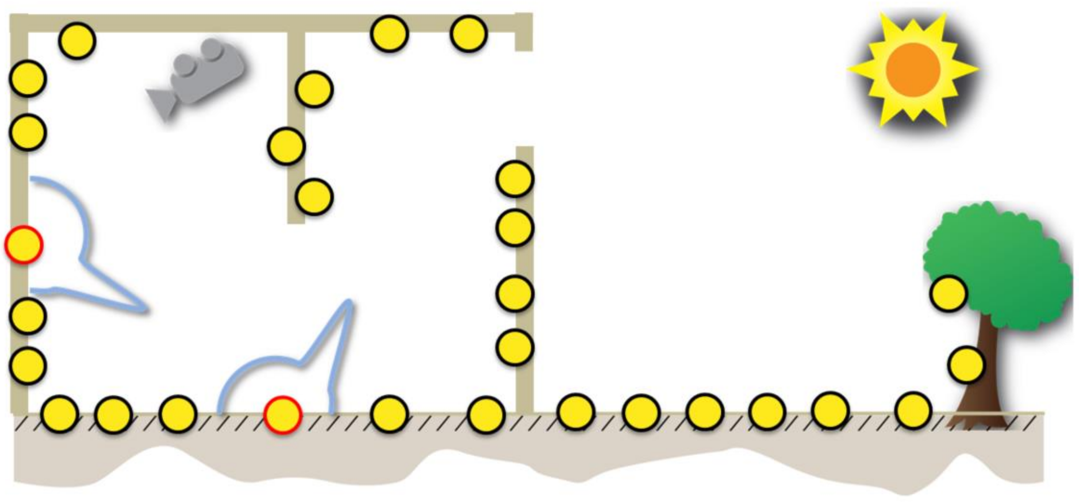
Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 58

To get the approximation of the radiance field we could for example precompute irradiance or radiance cache.

However, instead of implementing the cache according to works of Ward or Krivanek, we base our solution on our previous path guiding method.



Path guiding



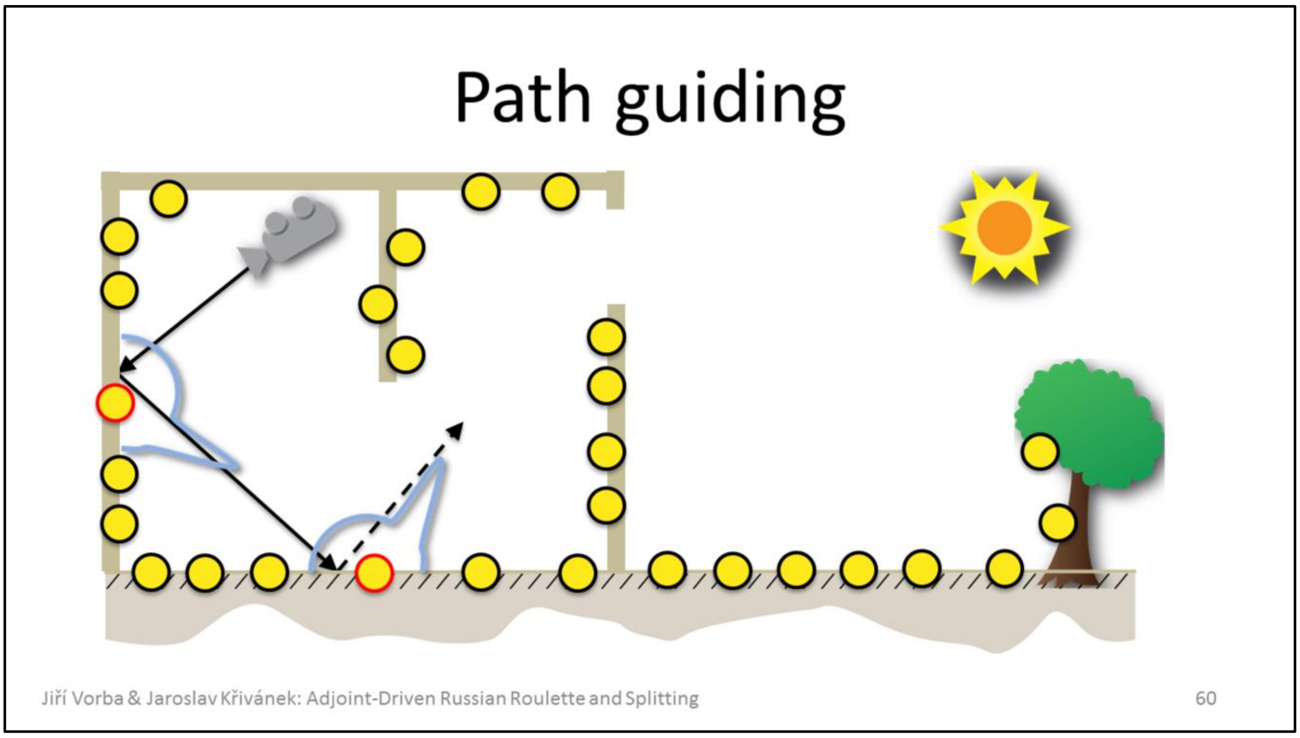
Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 59

This method precomputes directional distributions in the scene and stores them in a spatial cache.

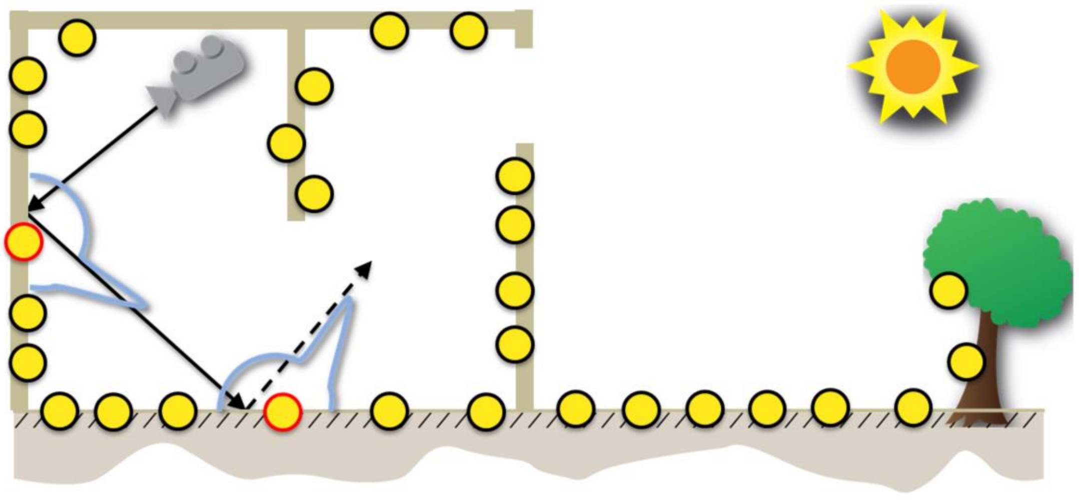
Each yellow dot corresponds to such a distribution.

Each distribution is fitted so that it is roughly proportional to incident radiance.

The primary goal of the method is to use the distributions for sampling scattering directions in order to guide paths towards the light sources.



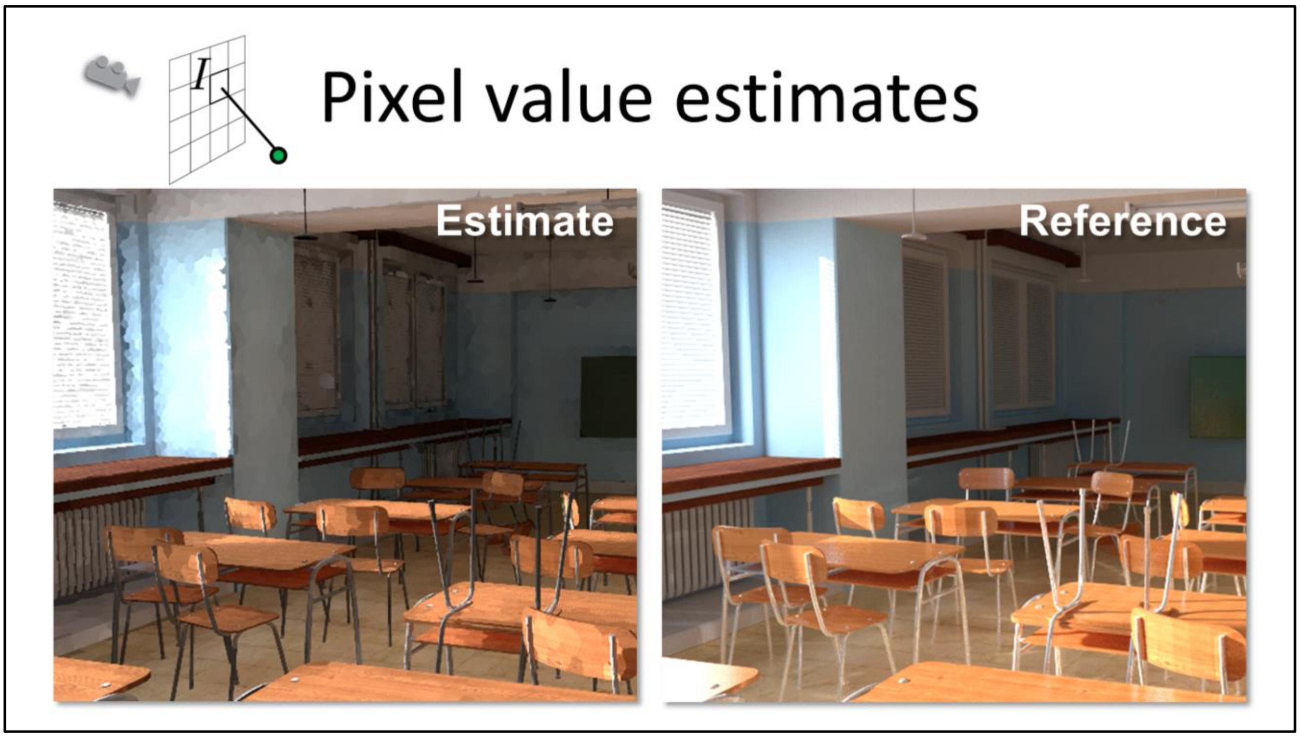
Path guiding



Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 60

However, in our ADRRS implementation we rather use the precomputed distributions to estimate the radiance field.

For this purpose we also store the irradiance with each cache record.

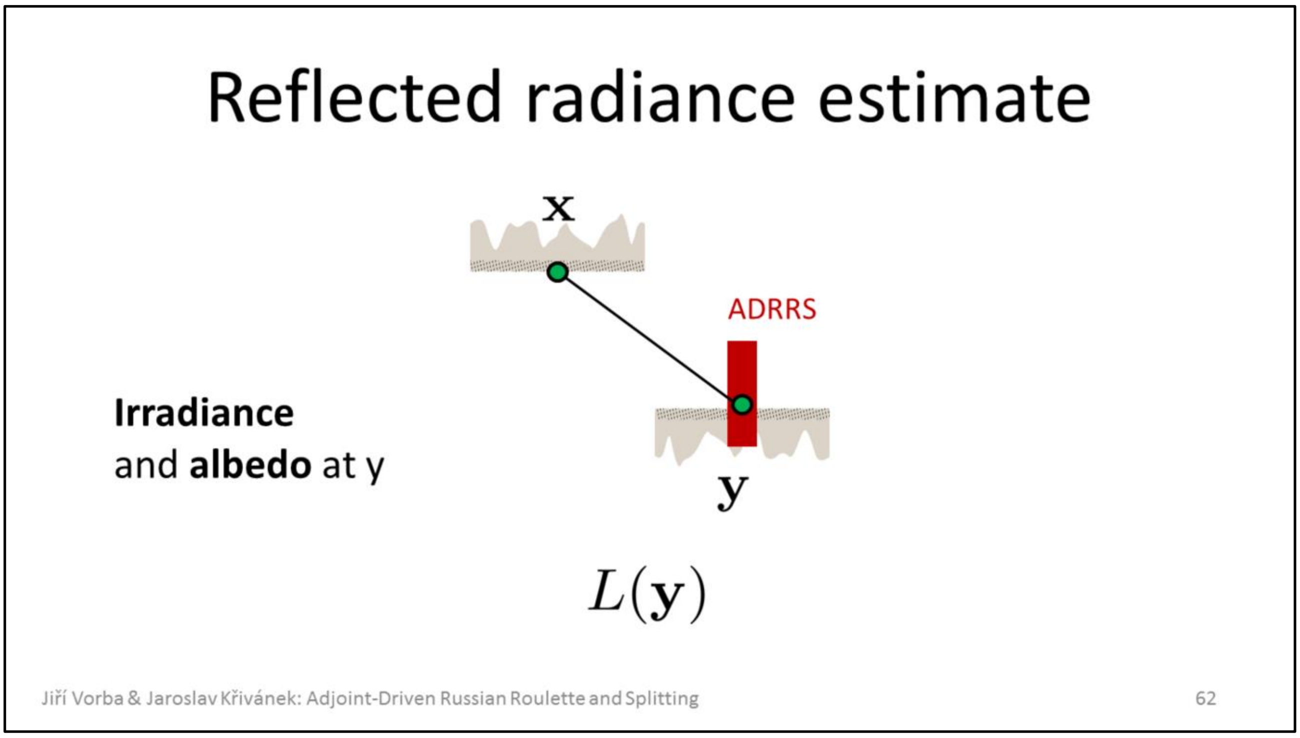


Pixel value estimates

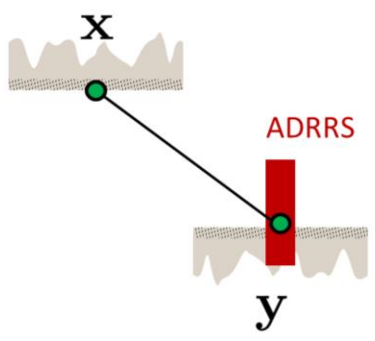
In our implementation we simply use irradiance multiplied by albedo of materials to estimate the pixel values I.

Note that the estimates visualized in the left image that we use in our implementation are rather poor.

Still even on such a coarse input ADRRS achieves very good results.



Reflected radiance estimate



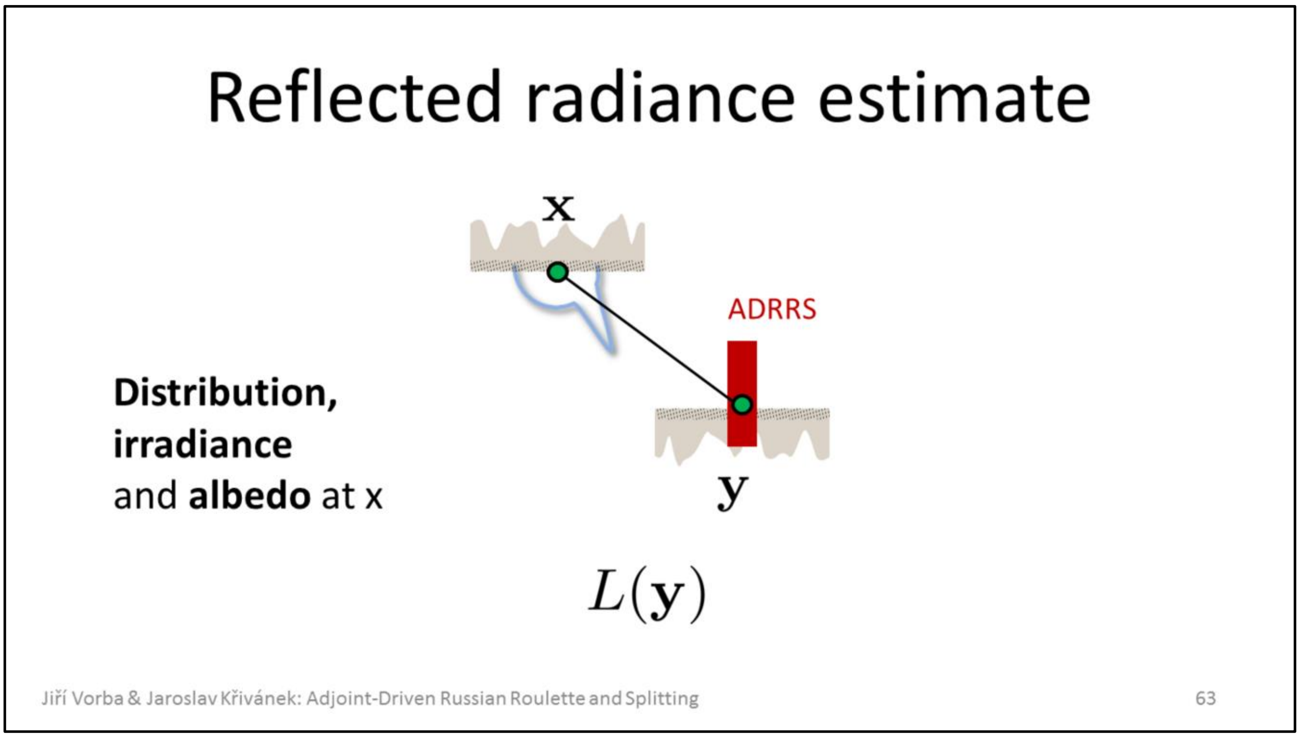
Irradiance
and **albedo** at y

$$L(\mathbf{y})$$

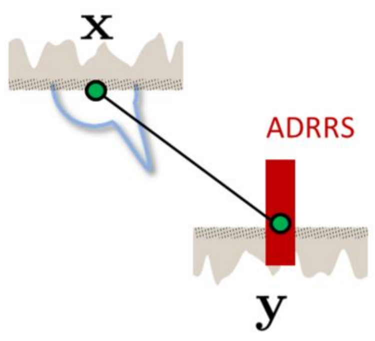
To approximate the reflected radiance from y on secondary hits we could use the same approach.

That is the irradiance multiplied by albedo at y.

It works well when y is a diffuse material, however, such an approximation underestimates glossy materials.



Reflected radiance estimate



Distribution,
irradiance
and **albedo** at x

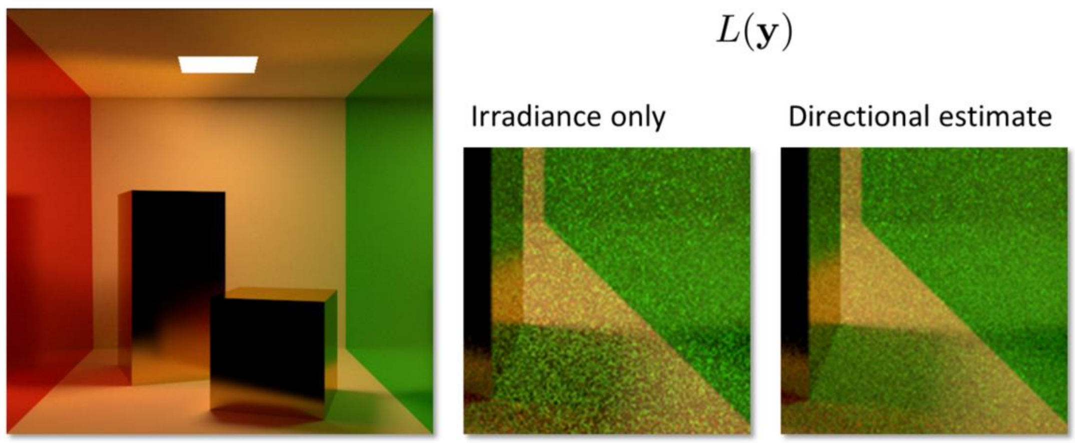
$L(\mathbf{y})$

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 63

To achieve better approximation we also use directional distribution and stored irradiance at x.

In this way we can achieve reasonable approximation of reflected radiance at y even if y is a glossy material.

ADRRS on glossy surfaces



Here on the right inset we can see the effect of better estimate of reflected radiance when we use also the directional information.

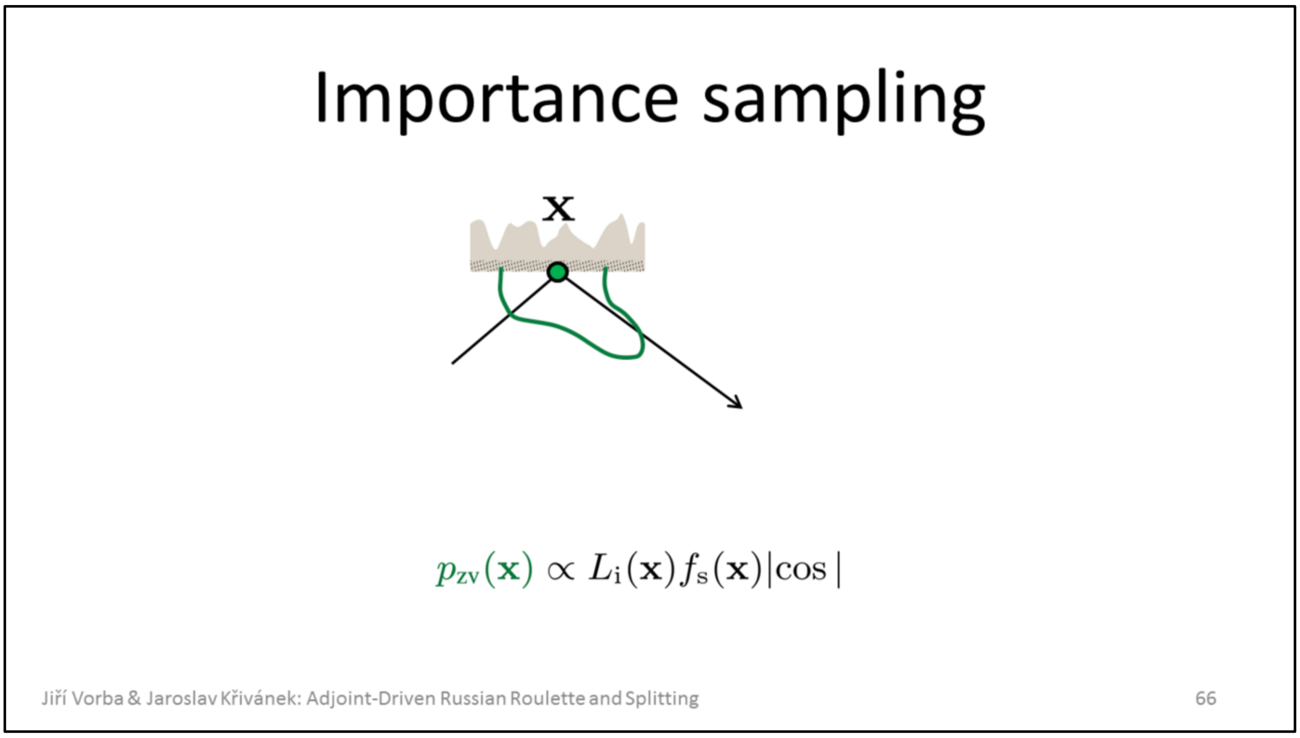
The left inset uses only irradiance multiplied by albedo.

Importance sampling

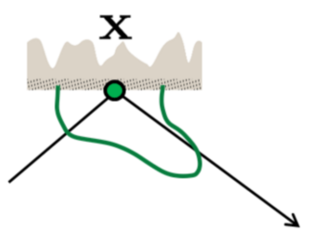
Background | Our method | Precomputing estimates | **Importance sampling** | Results

65

Now I would like to present our theoretical contribution that sheds light on importance sampling properties of our ADRRS.



Importance sampling



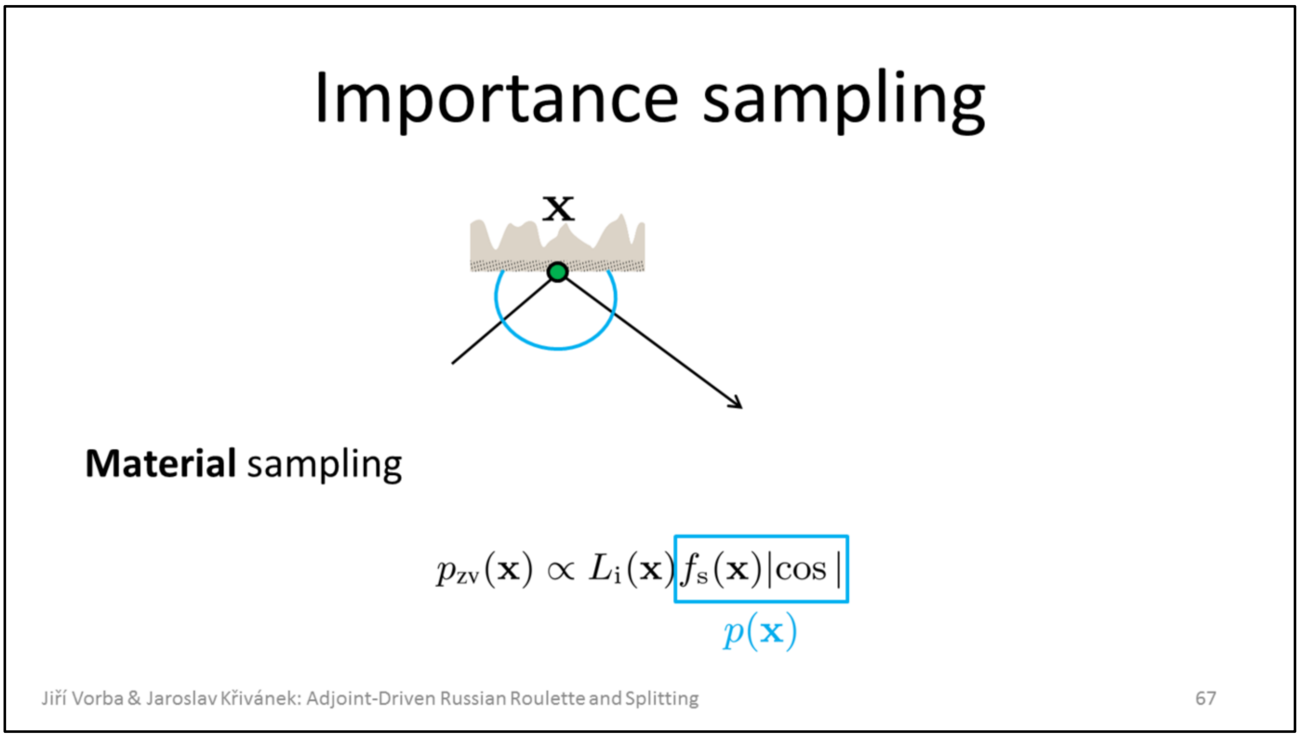
$$p_{\mathrm{zv}}(\mathbf{x}) \propto L_{\mathrm{i}}(\mathbf{x}) f_{\mathrm{s}}(\mathbf{x}) |\cos|$$

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 66

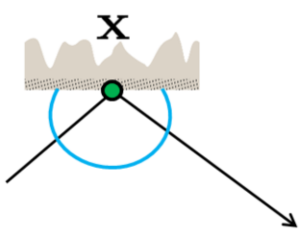
When we want to sample a direction at the position x, the rendering equation suggests that

the ideal zero-variance distribution is proportional to the product of incident radiance, bsdf and cosine term at x.

This is the holy grail of the importance sampling.



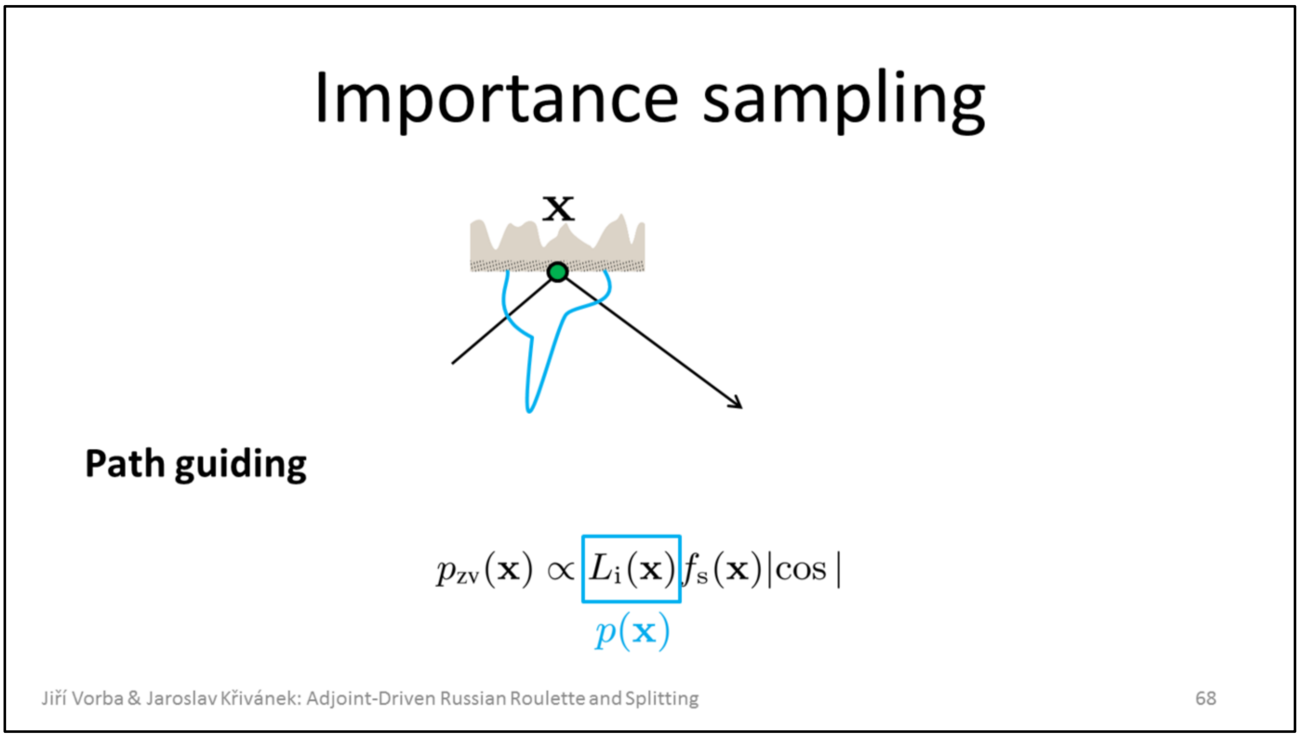
Importance sampling



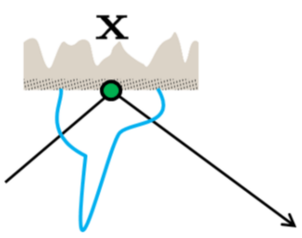
Material sampling

$$p_{\mathrm{zv}}(\mathbf{x}) \propto L_{\mathrm{i}}(\mathbf{x}) \underbrace{\boxed{f_{\mathrm{s}}(\mathbf{x})|\cos|}}_{p(\mathbf{x})}$$

Now the standard approach is to importance sample scattering directions according to materials because this is usually the only known term in the product.



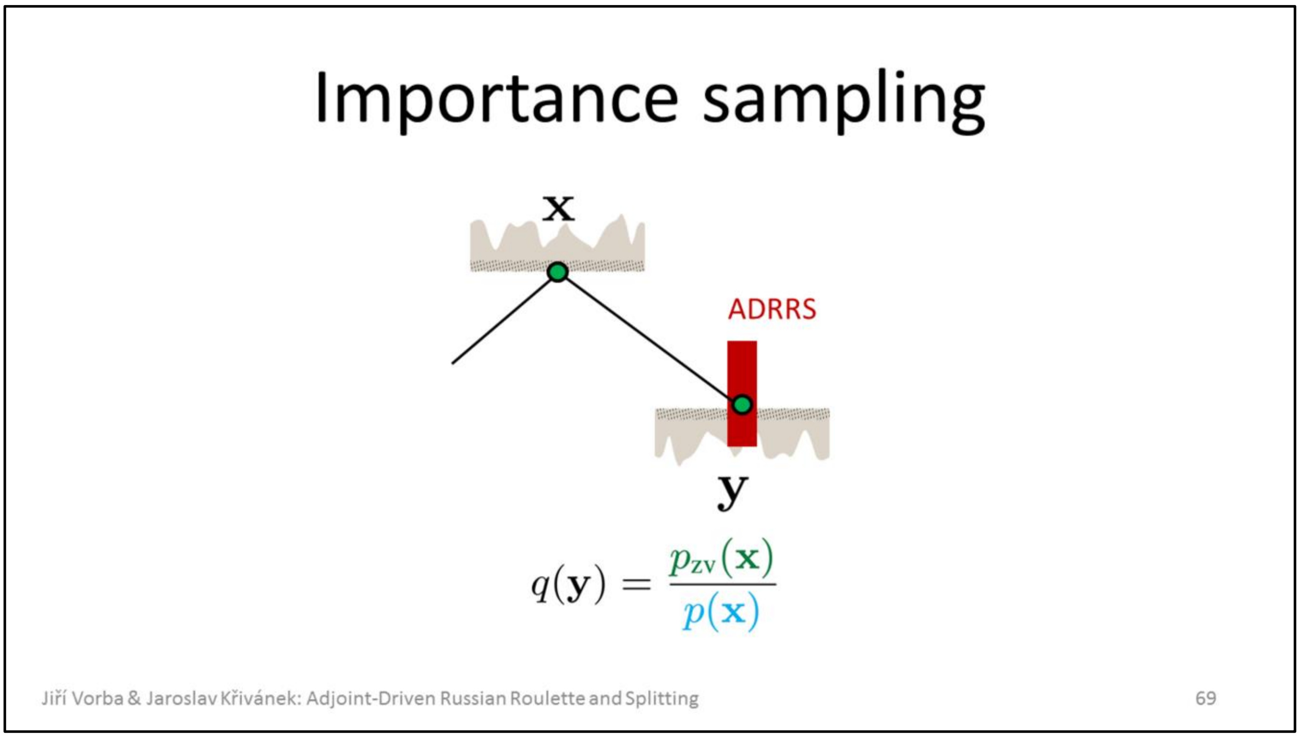
Importance sampling



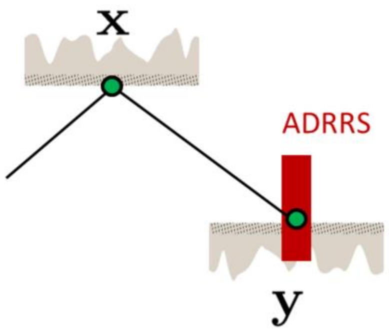
Path guiding

$$p_{\mathrm{zv}}(\mathbf{x}) \propto \underbrace{\boxed{L_{\mathrm{i}}(\mathbf{x})}}_{p(\mathbf{x})} f_{\mathrm{s}}(\mathbf{x})|\cos|$$

If we use path guiding we have also available the information about incident radiance so we can importance sample according to this term.



Importance sampling



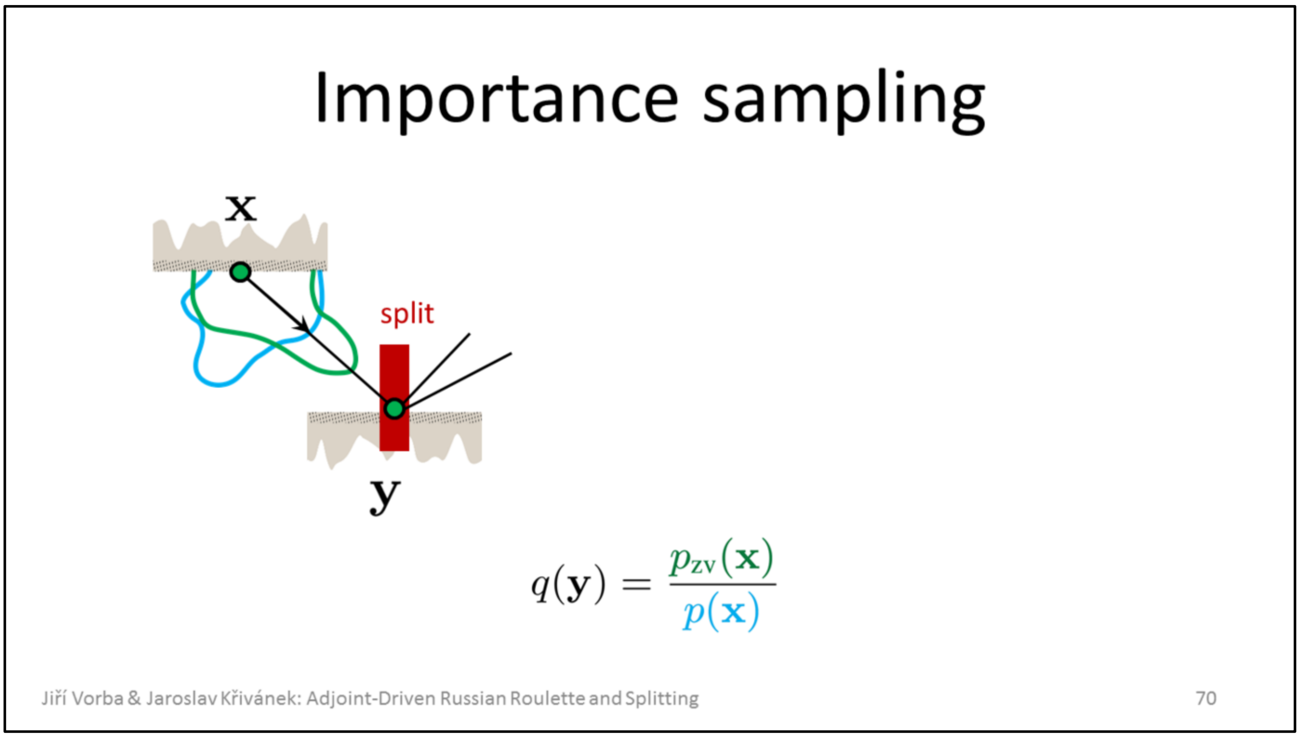
$$q(\mathbf{y}) = \frac{p_{\mathrm{ZV}}(\mathbf{x})}{p(\mathbf{x})}$$

Now this becomes very interesting.

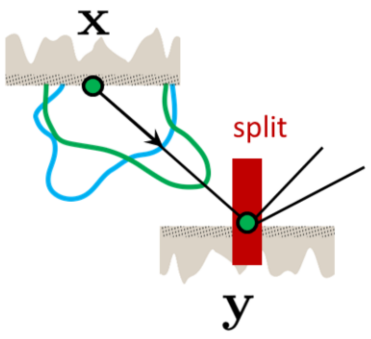
We show in the paper, that our termination/splitting factor based on the expected contribution
is actually equal to a ratio of the ideal zero-variance pdf and the actual pdf used for sampling the scattering directions.

This form of the termination/splitting factor also reveals very nice property of our method:

splitting and termination at y only depends on the event at x. Whatever has happened before doesn't matter.



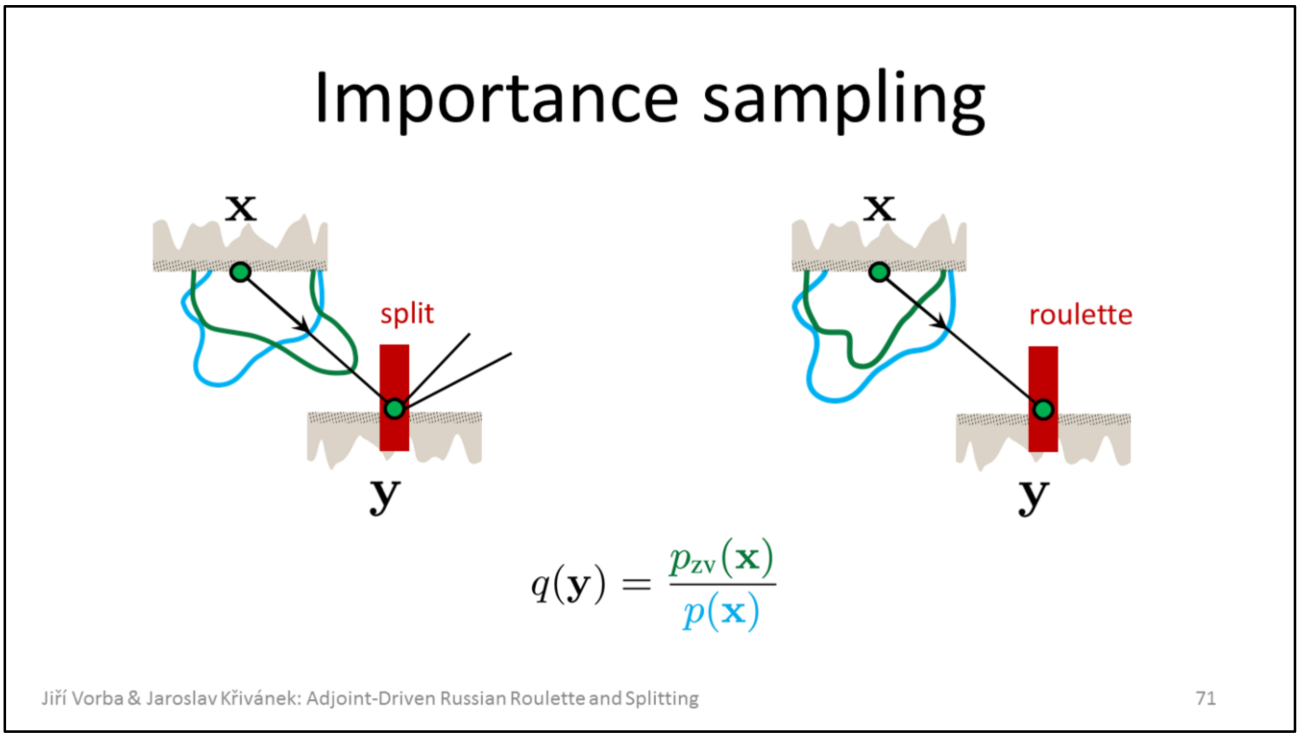
Importance sampling



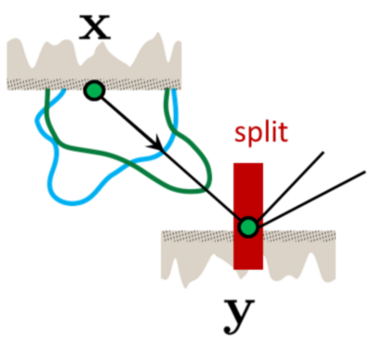
$$q(\mathbf{y}) = \frac{p_{\mathrm{ZV}}(\mathbf{x})}{p(\mathbf{x})}$$

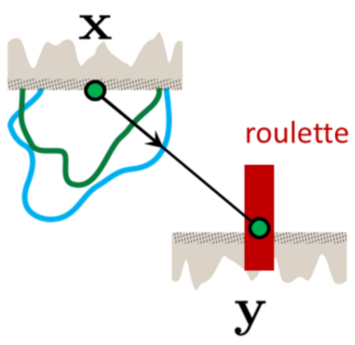
Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 70

The formula also suggests that whenever we undersample the ideal distribution in some direction that is when p is lower than the ideal distribution, we will split the path immediately at the next vertex to compensate for this undersampling.



Importance sampling

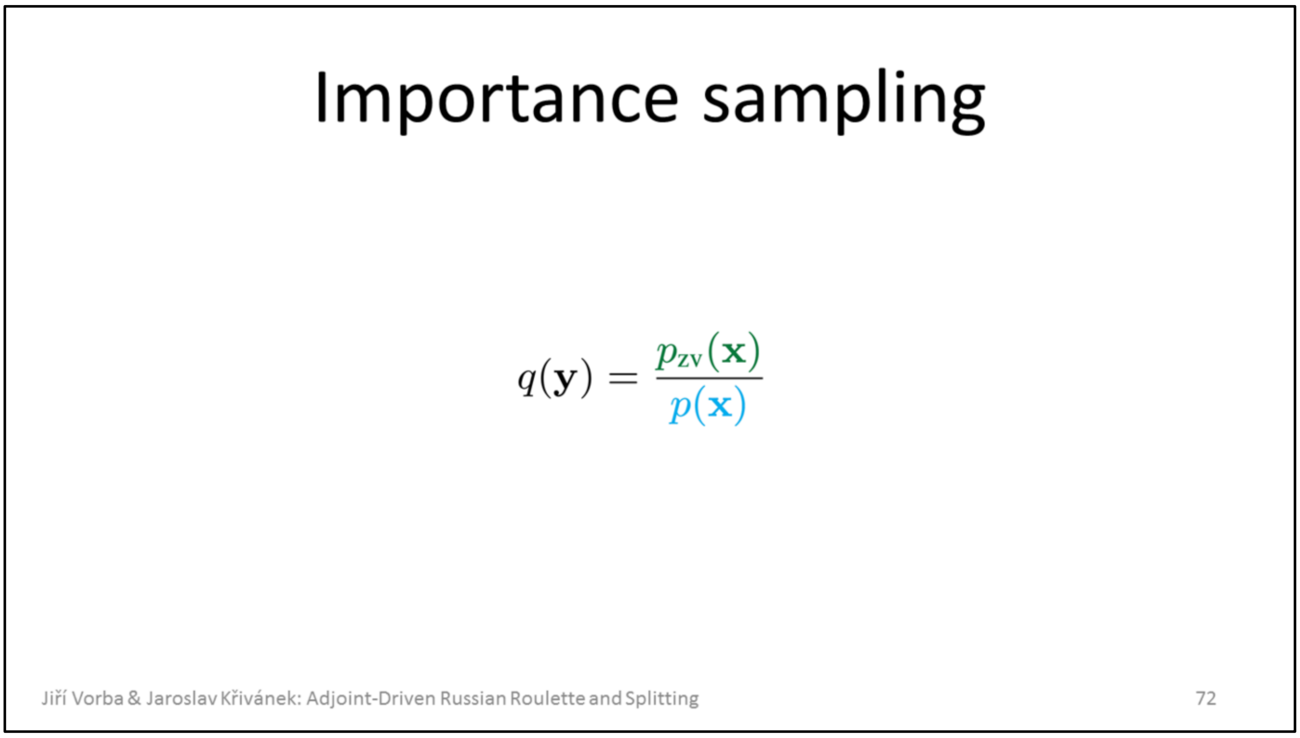




$$q(\mathbf{y}) = \frac{p_{\text{zv}}(\mathbf{x})}{p(\mathbf{x})}$$

Similarly, if we cast way too many rays in some direction, that is the actual sampling distribution is higher than the ideal distribution, Russian roulette will act as a rejection sampling to adjust the sample density.

So our method steps in immediately when it detects that sampling didn't go as it ideally should have. Then it tries to amend the wrong decision through termination and splitting and the effective distribution of our samples becomes equal to the ideal distribution.



Importance sampling

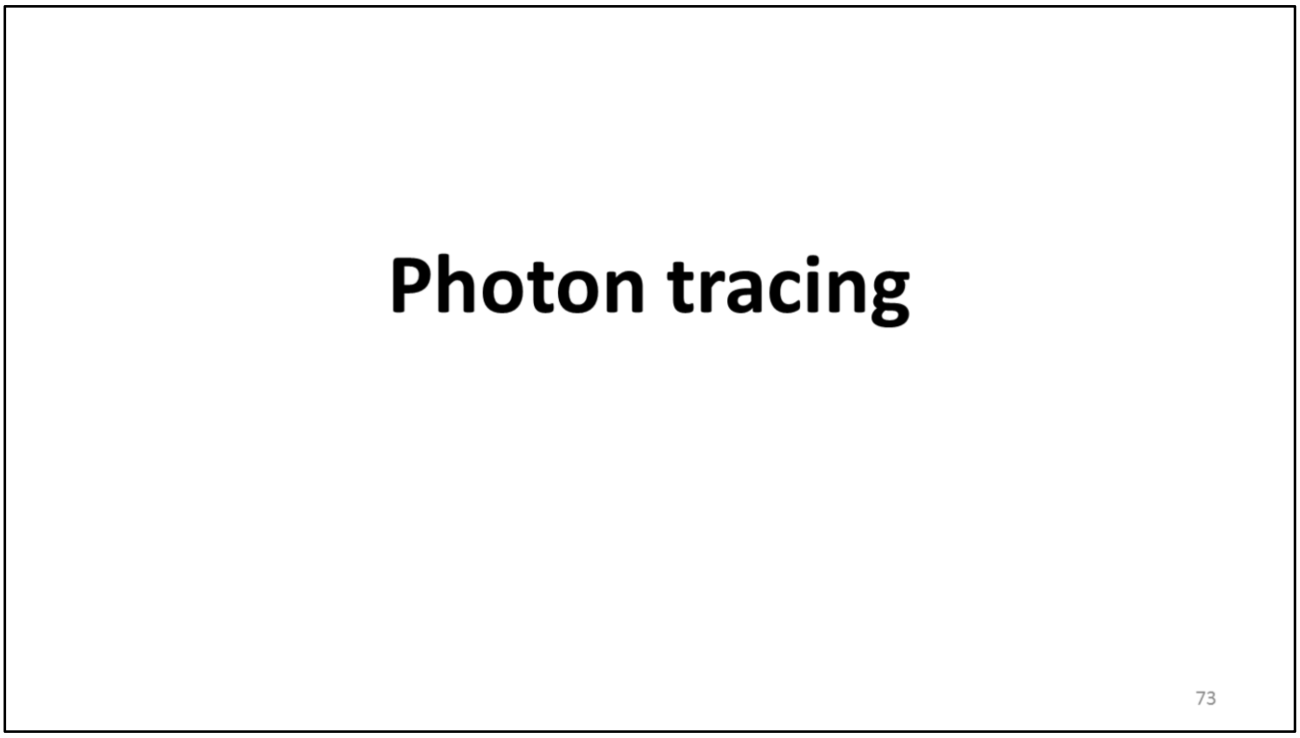
$$q(\mathbf{y}) = \frac{p_{\text{ZV}}(\mathbf{x})}{p(\mathbf{x})}$$

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 72

The formula has one more implication.

The better the sampling distribution matches the ideal distribution the less work will ADRRS do. If they would match our ADRRS would never increase the variance because it would simply do nothing.

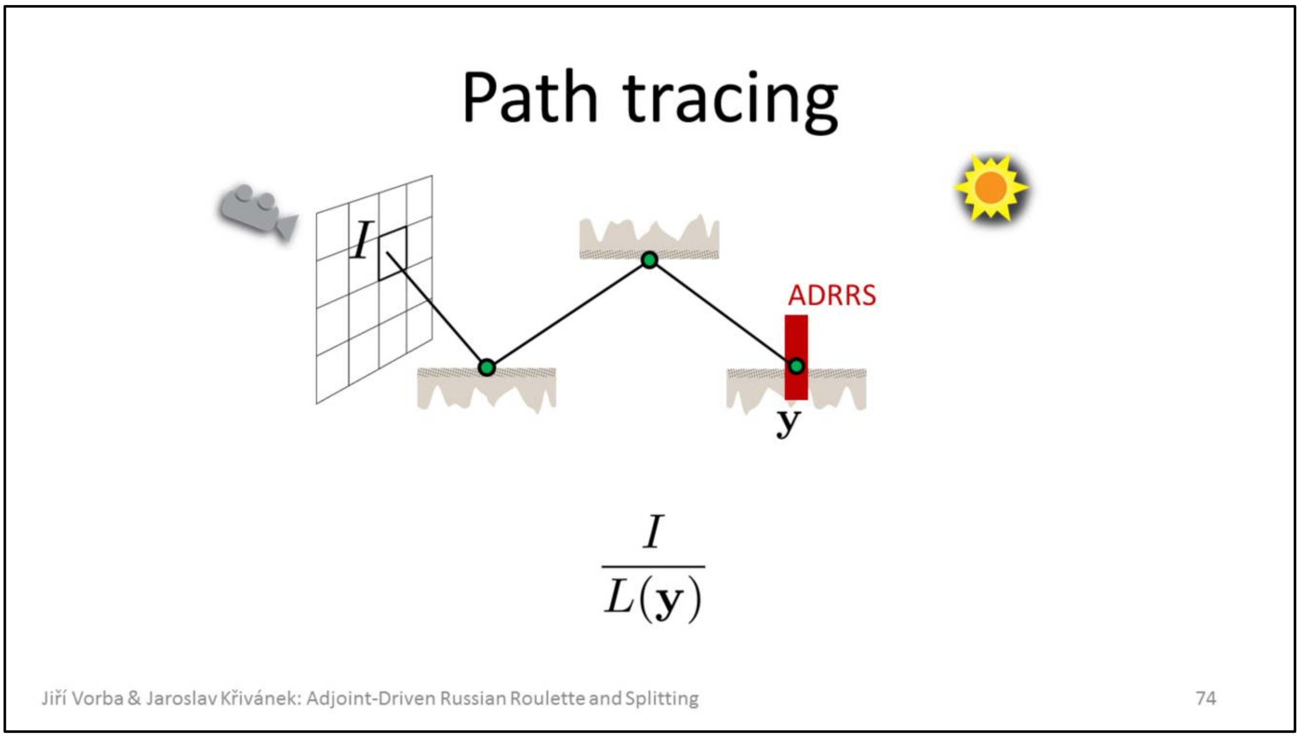
This explains why it works in synergy with path guiding.



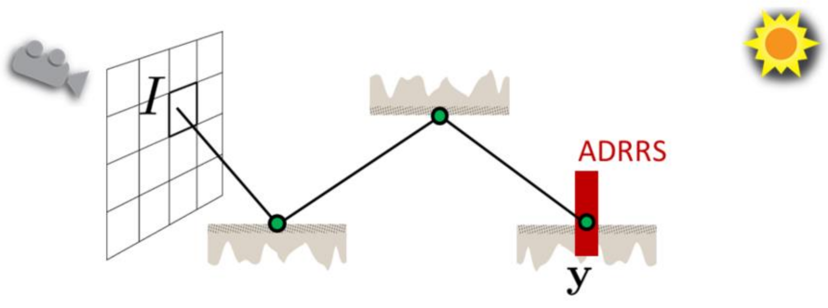
Photon tracing

73

Before I show you our results, I should briefly say what needs to be done for algorithms based on tracing photons from light sources.



Path tracing

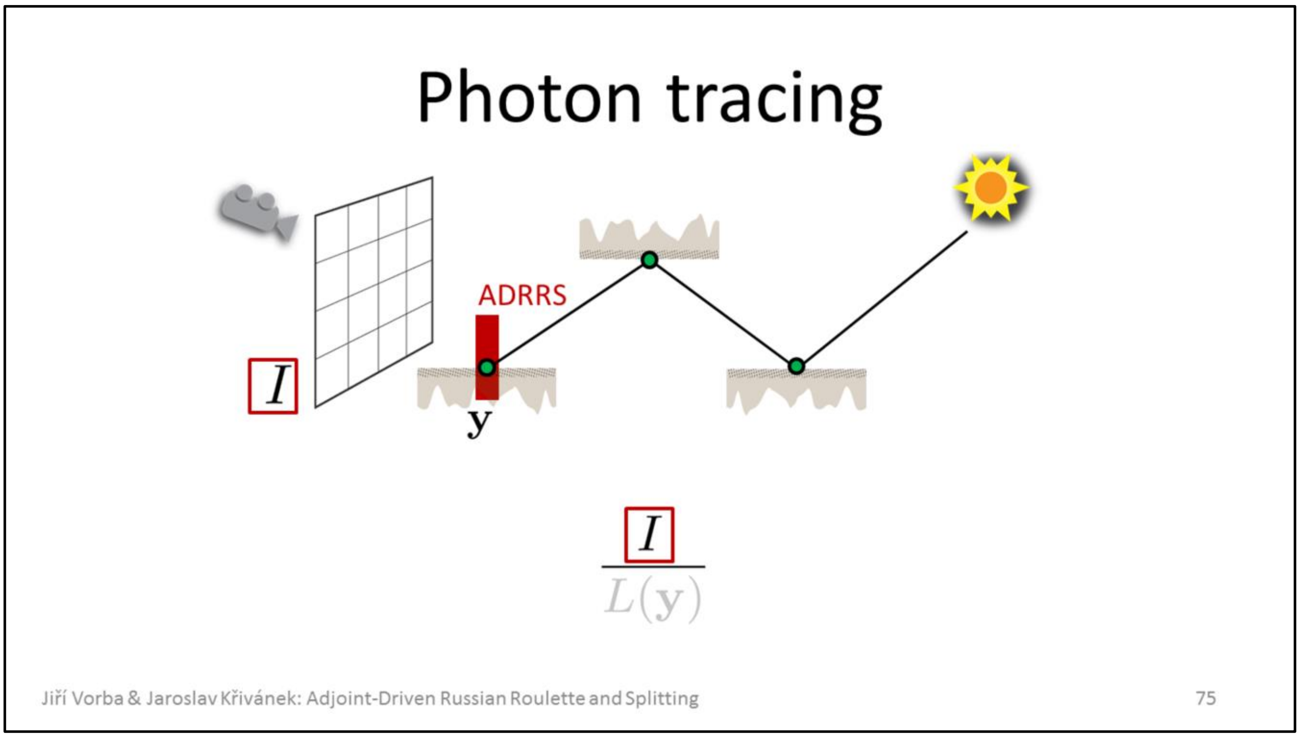


$$\frac{I}{L(\mathbf{y})}$$

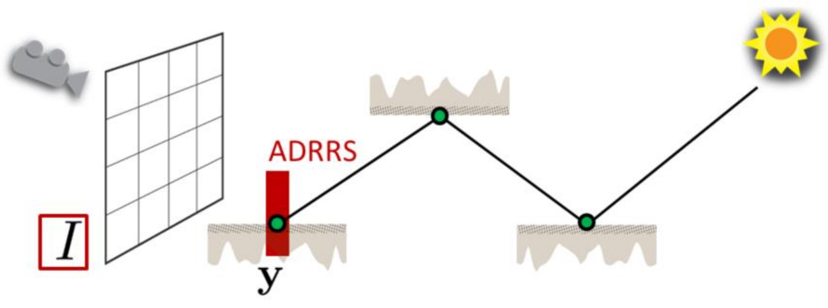
Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 74

So far I was explaining our ADRRS on a path tracing example and we needed to know pixel estimates I and radiance estimates L.

For light tracing or photon mapping, we only need to change these two estimates.

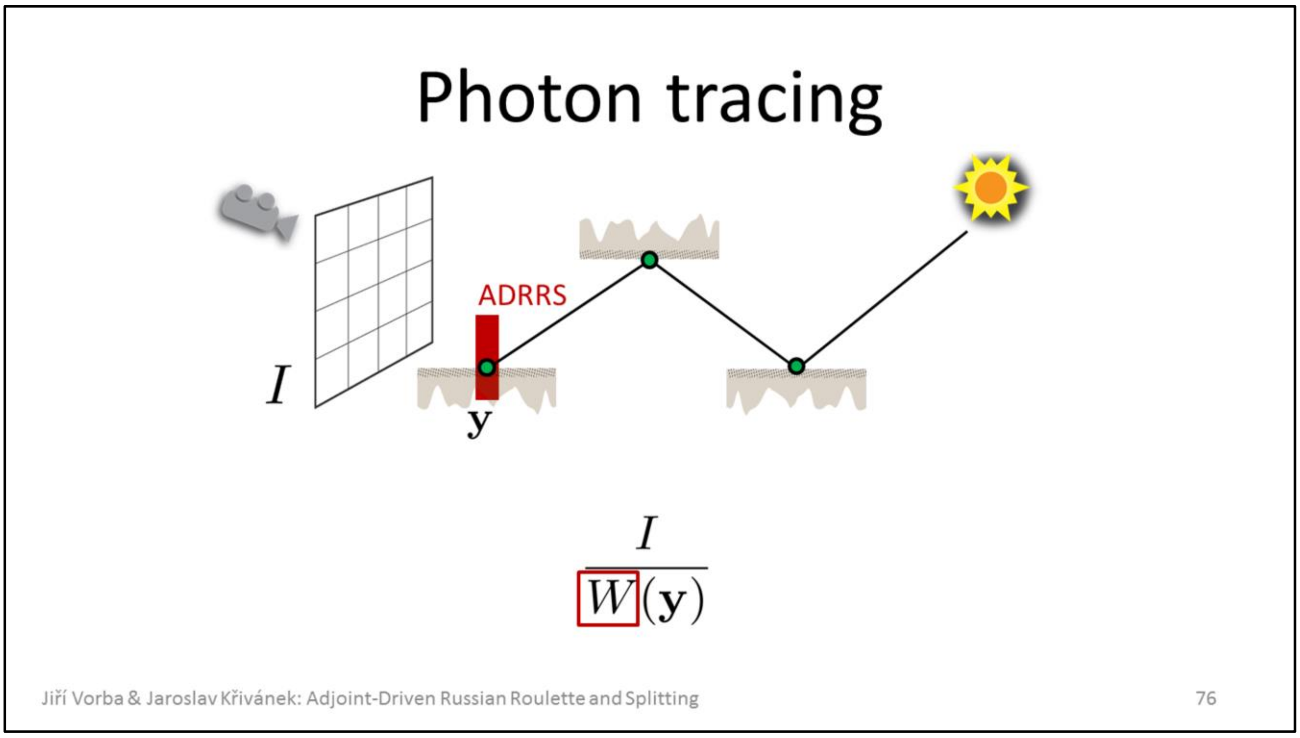


Photon tracing



$$\frac{\boxed{I}}{L(\mathbf{y})}$$

Because we do not know a priory what pixel will our path contribute to we take an average over all pixels to get the estimate I.



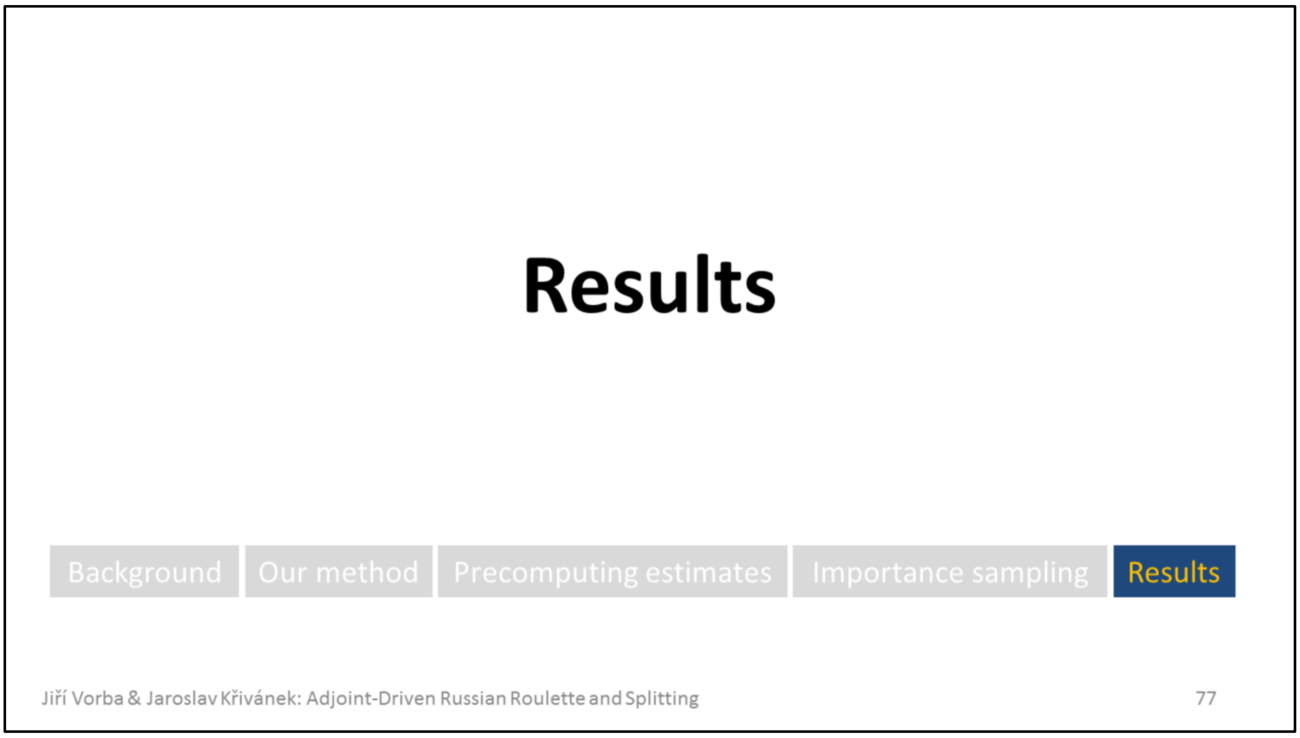
Photon tracing

$$\frac{I}{\boxed{W}(\mathbf{y})}$$

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 76

We also need to change the adjoint quantity that we use for setting up our weight window.

Now, instead of radiance, we use visual importance emitted from the camera.



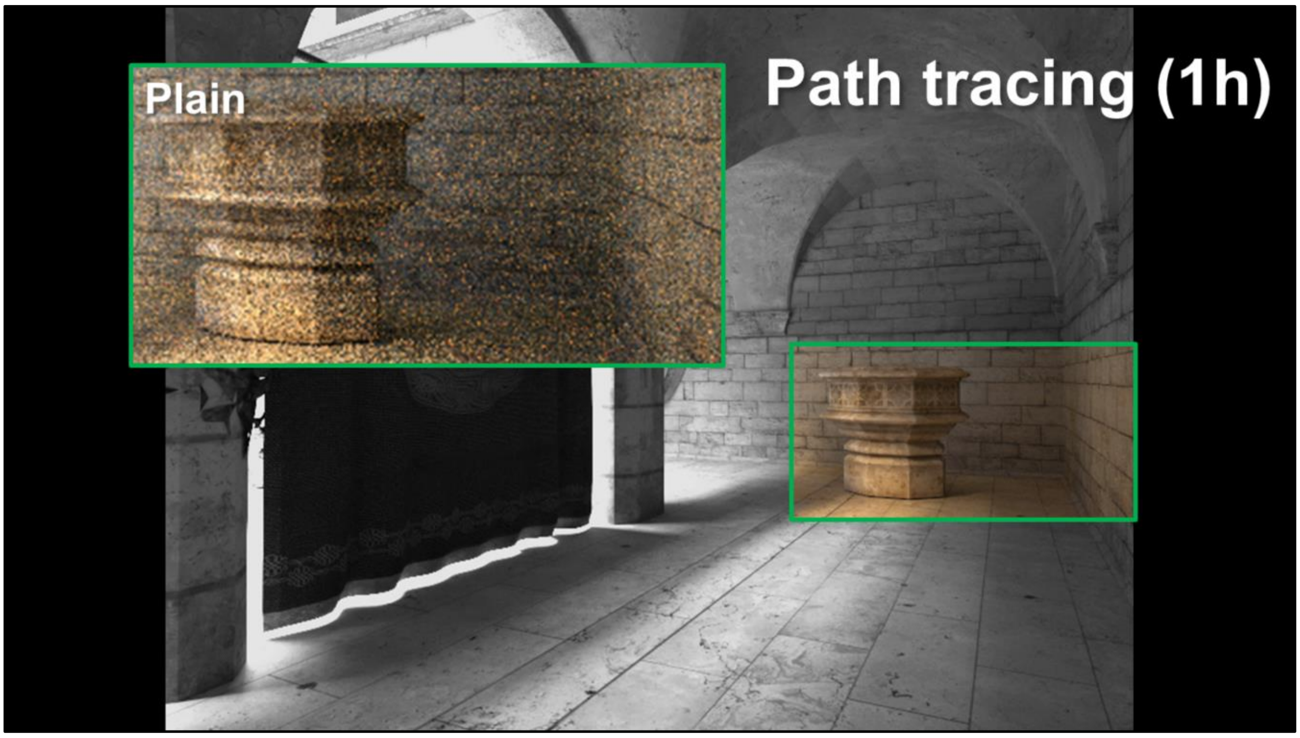
Results

Background | Our method | Precomputing estimates | Importance sampling | Results

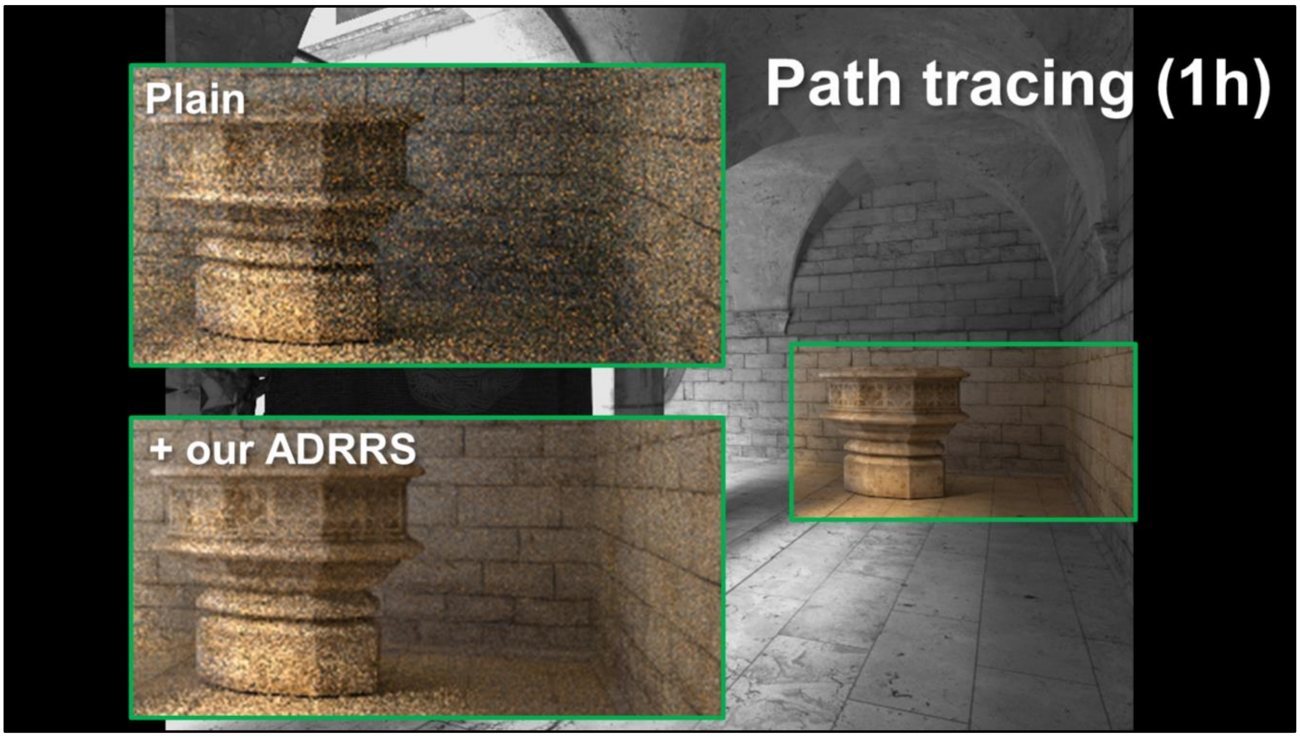
Now I will present some of our results.

We rendered all our results for one hour on Intel i7 with eight logical cores.

The preprocessing time is included in this one hour.



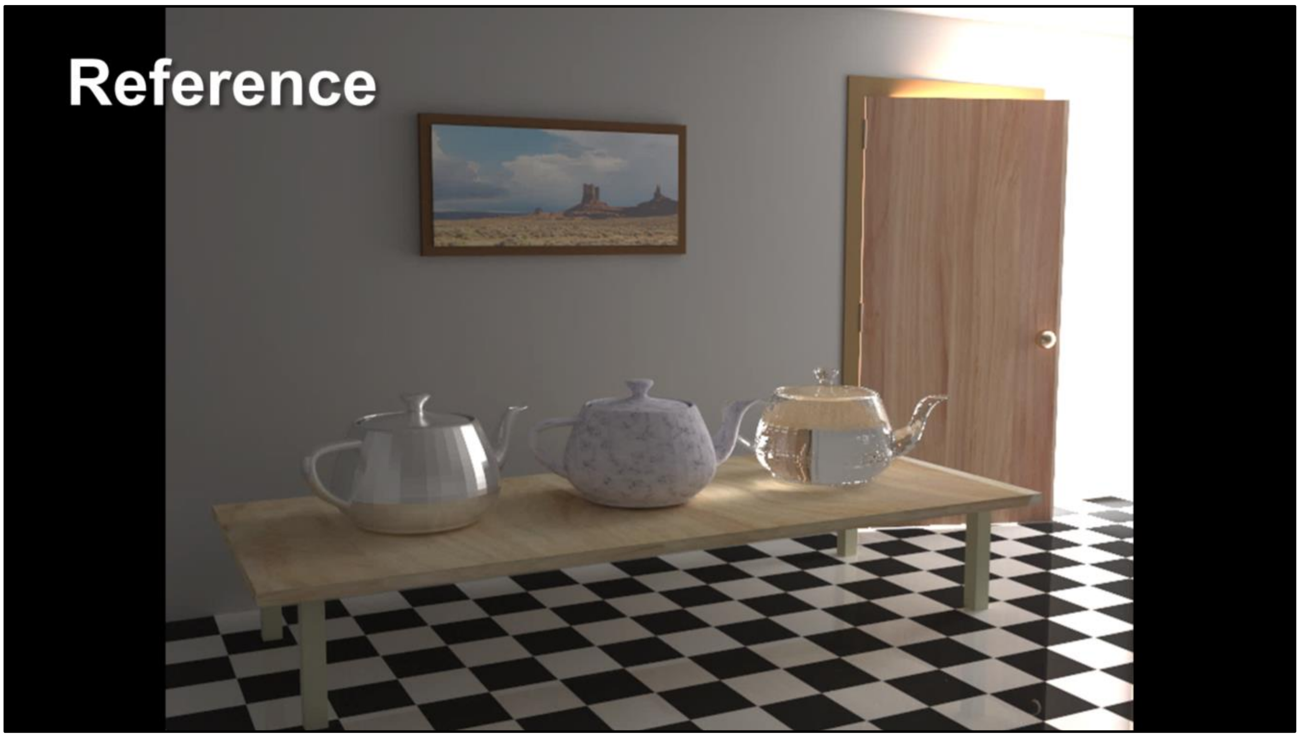
We can see that path tracing with RR roulette is very inefficient on the indirect illumination.



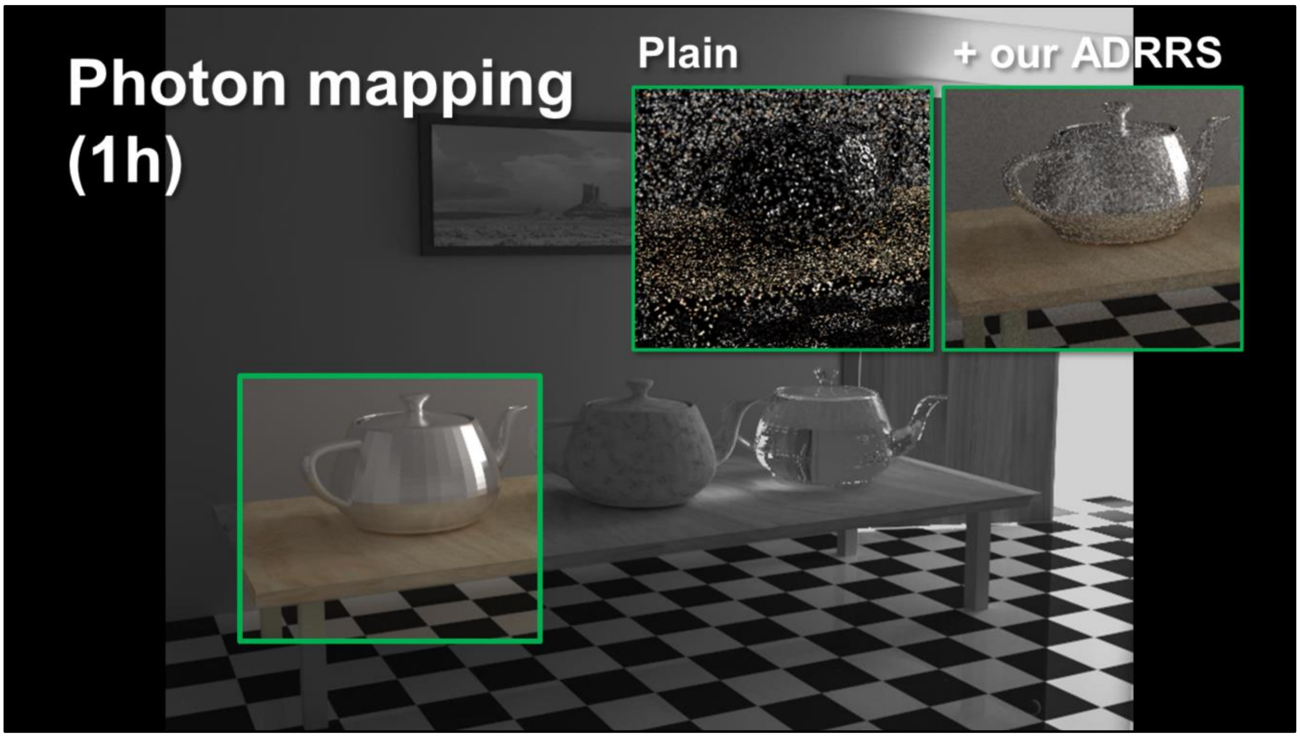
Our ADRRS improves mainly indirect illumination transported over long paths.

We can see that it didn’t improve much the noise at the bottom part of the well.

This illumination comes only from one indirect bounce.

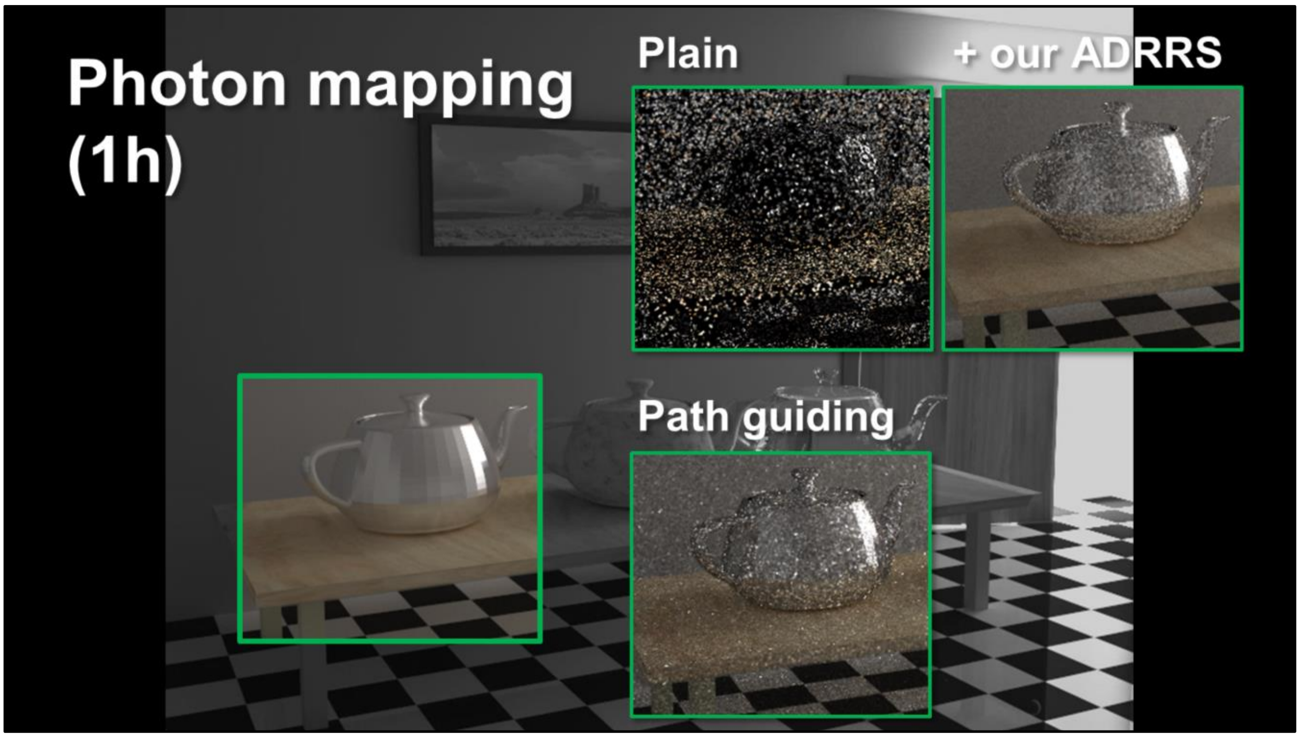


I will present a result of our ADRRS in photon mapping on this famous scene with teapots and light source behind the doors.

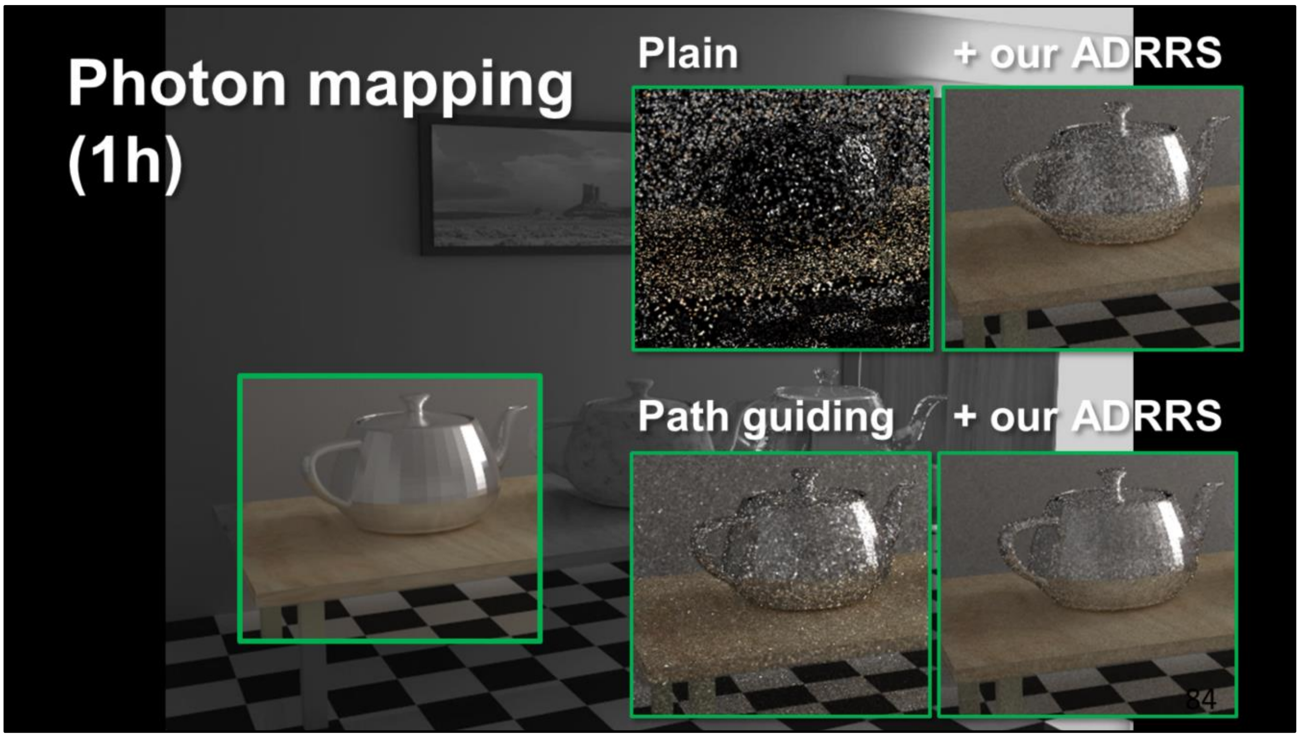


It turns out that ADRRS performs especially well with photon based approaches.

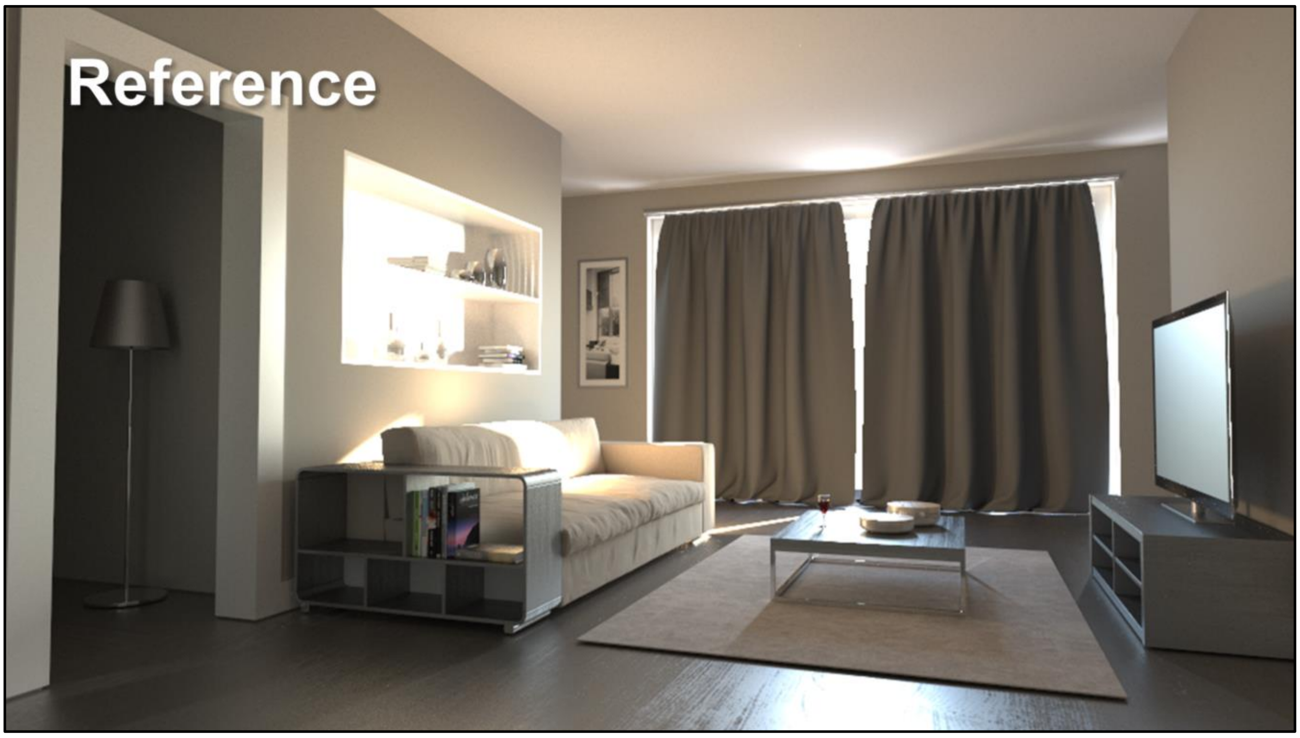
The upper left inset shows plain progressive photon mapping.



In this scene our ADRRS actually outperforms even the path guiding method.

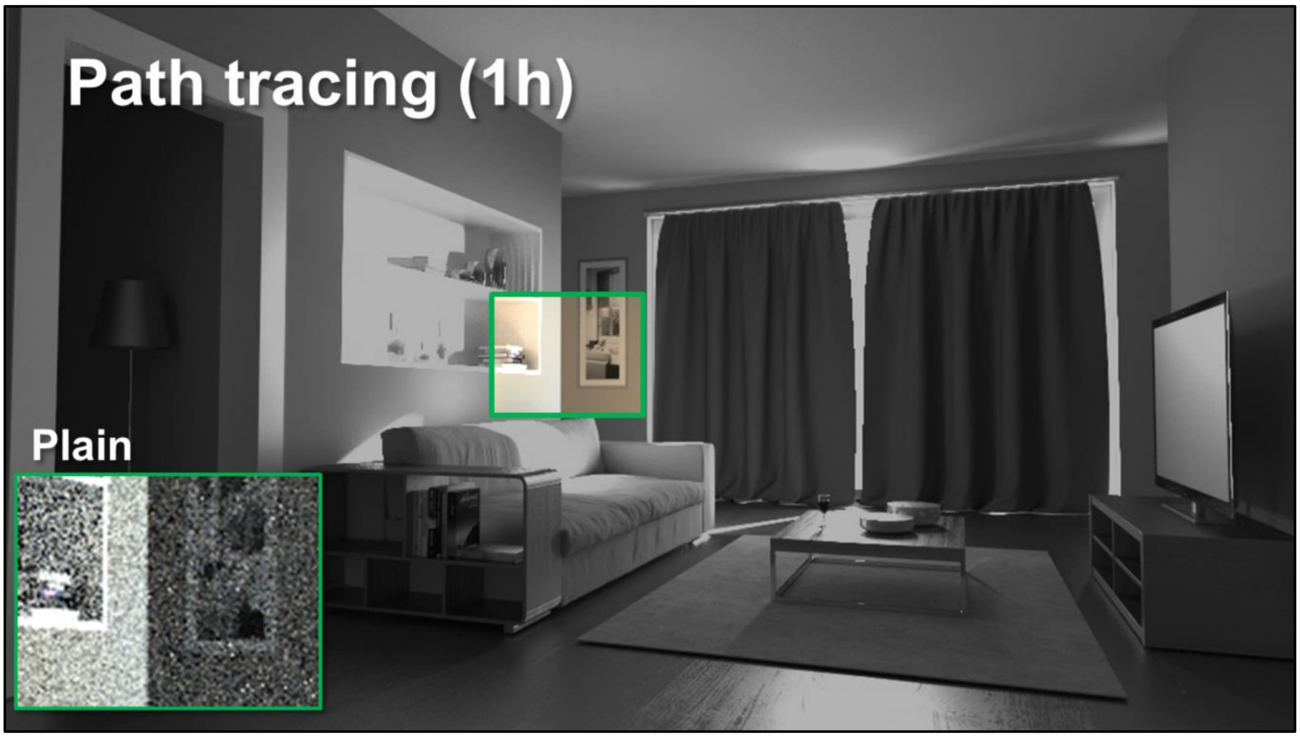


Together they achieve better result on the shiny teapot. This makes sense because guiding improves the direction dependent effects.

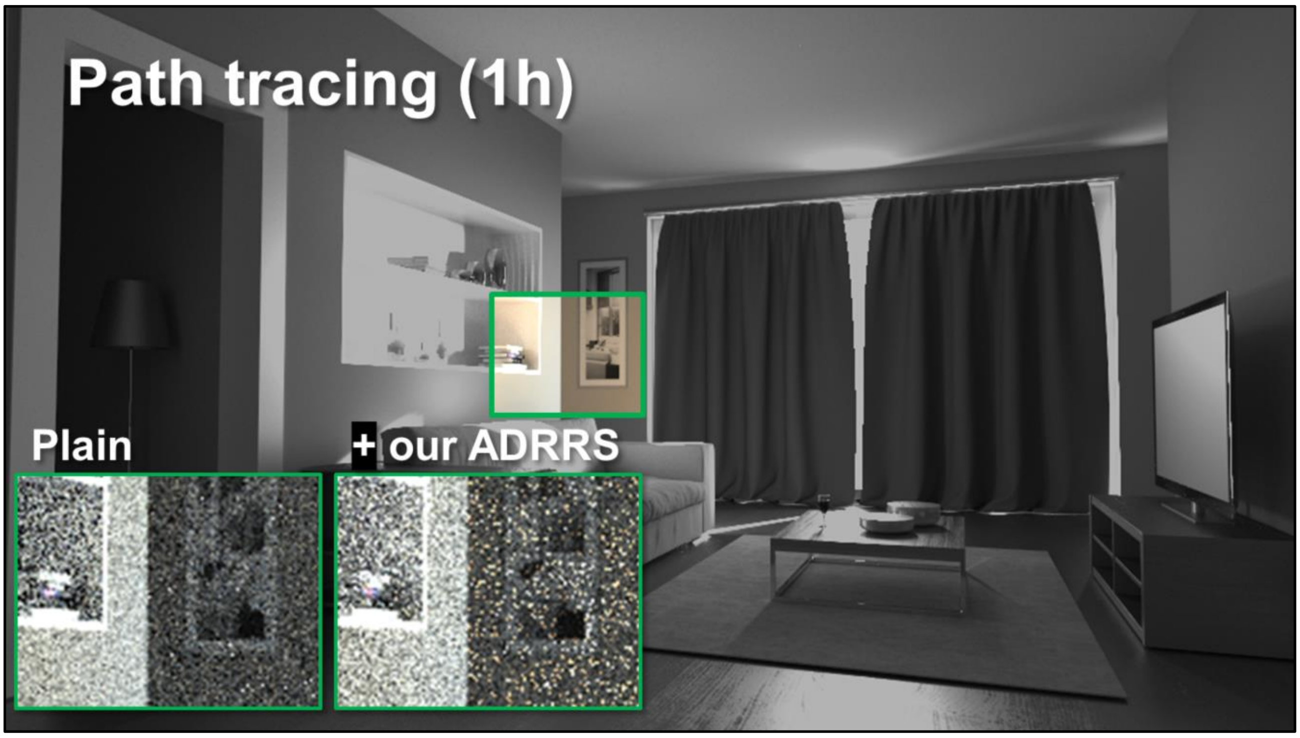


In this scene, the living room is illuminated by sun and sky.

Light comes only through the glass window and a small gap between the curtains.



These difficult conditions are a showstopper for plain path tracing.

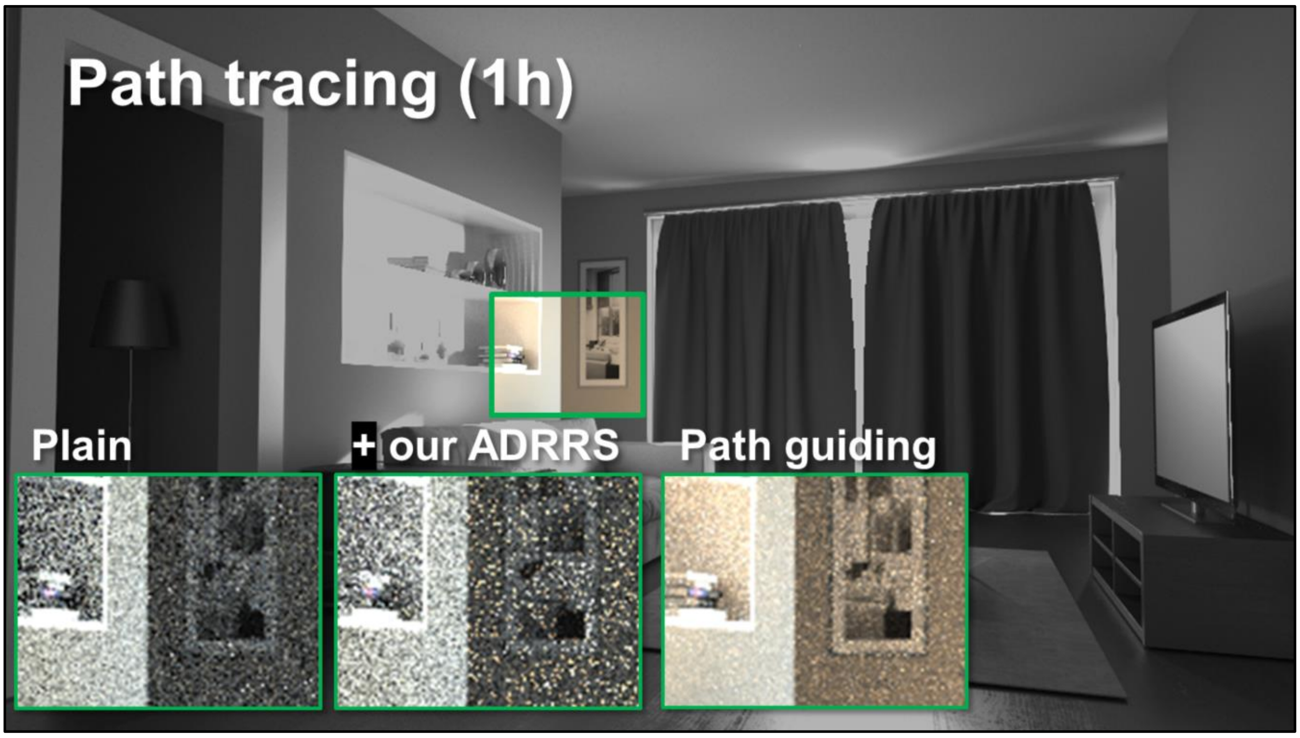


Our ADRRS doesn’t make any striking difference.

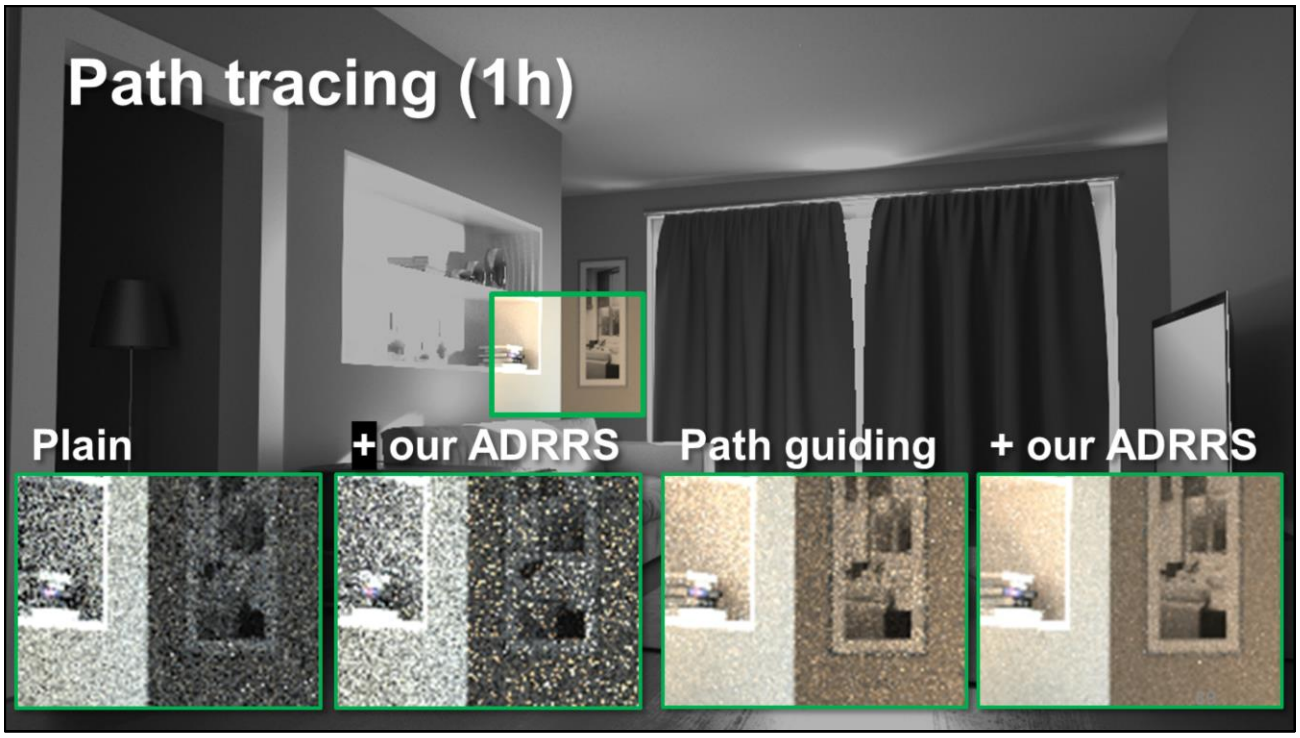
The main problem is that we can reach the light source only through a very narrow set of directions.

Our ADRRS detects that we highly undersample these directions only in the moment when path hits the window.

It is too late for splitting because all rays would be refracted in the same direction and end in the Sun.

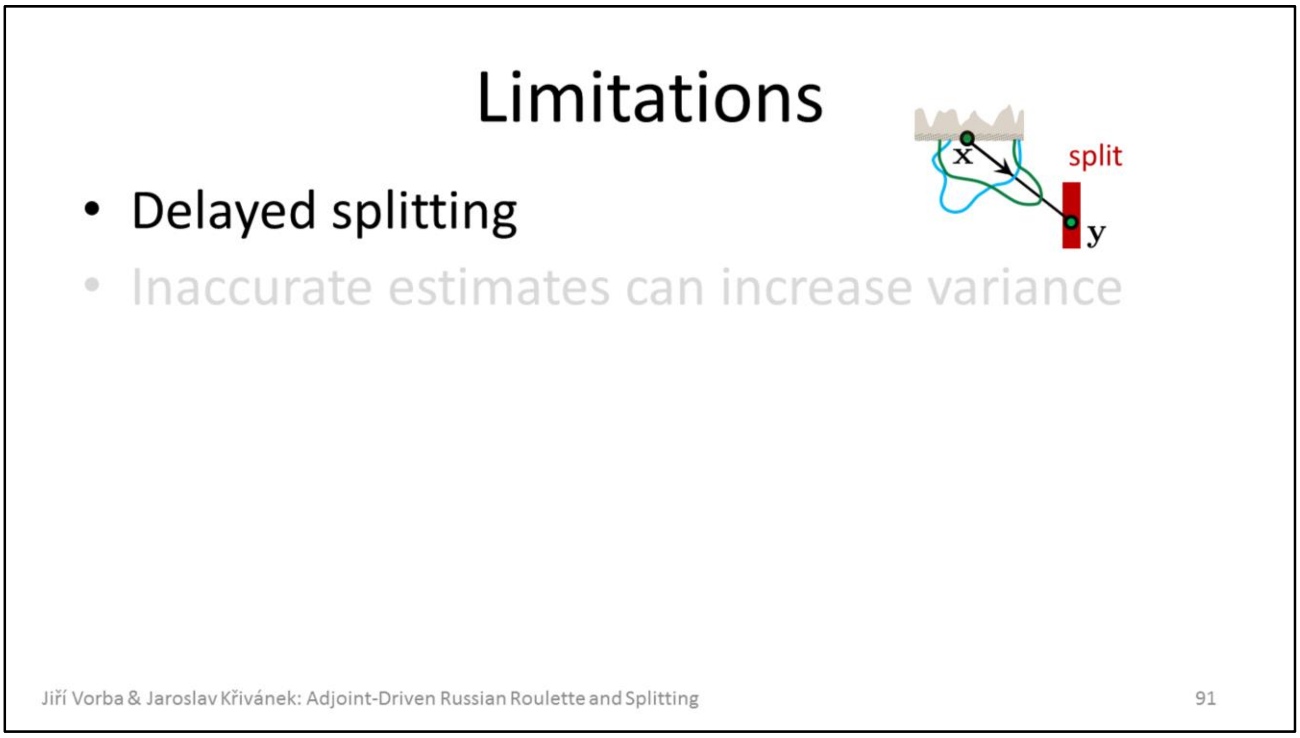


Nevertheless, this limitation can be complemented by path guiding.

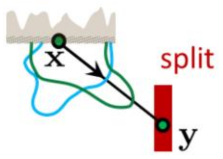


And as I already mentioned, path guiding works in synergy with our method and together achieve better results.

Limitations and future work



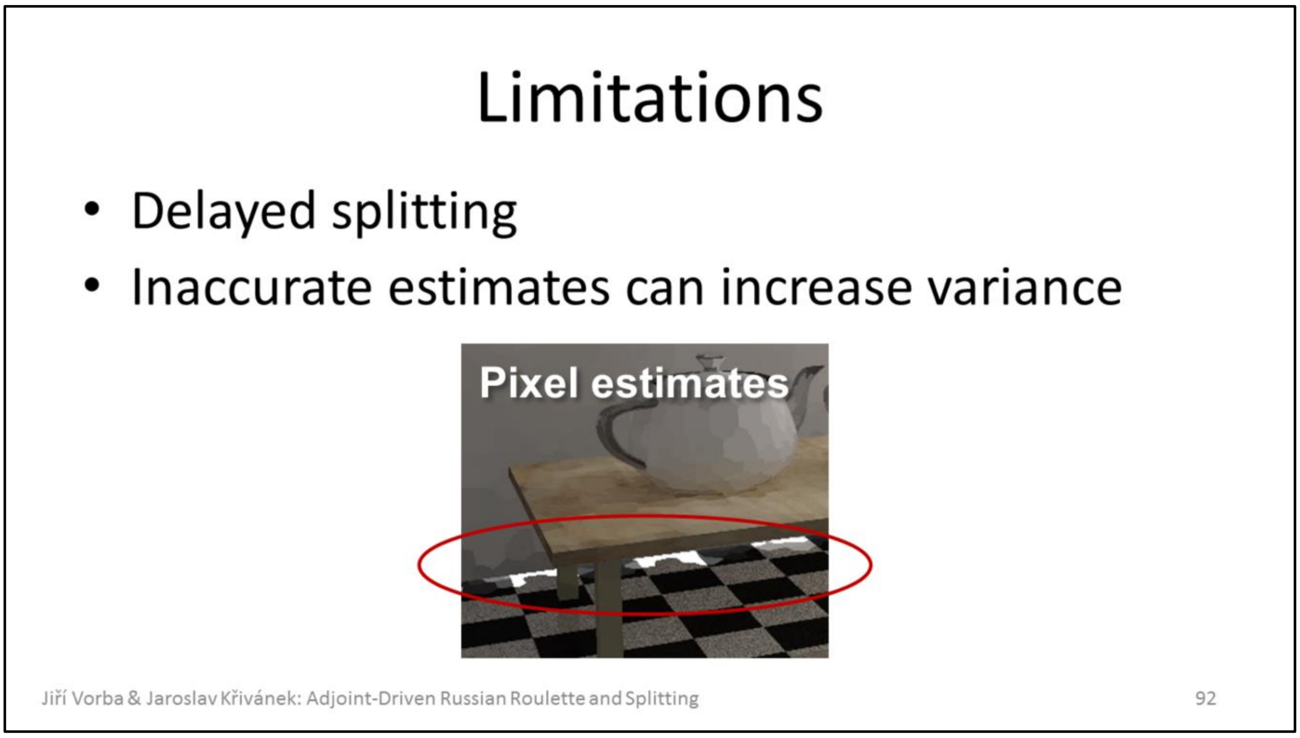
Limitations



- Delayed splitting
- Inaccurate estimates can increase variance

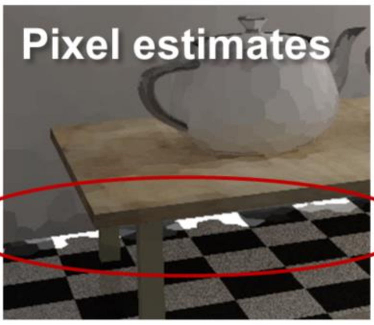
As a major limitation, we consider the fact that the splitting does not take place on the same vertex where the sub-optimal directional sampling introduce the variance.

However, this is nicely complemented by path guiding methods that improve the directional sampling.



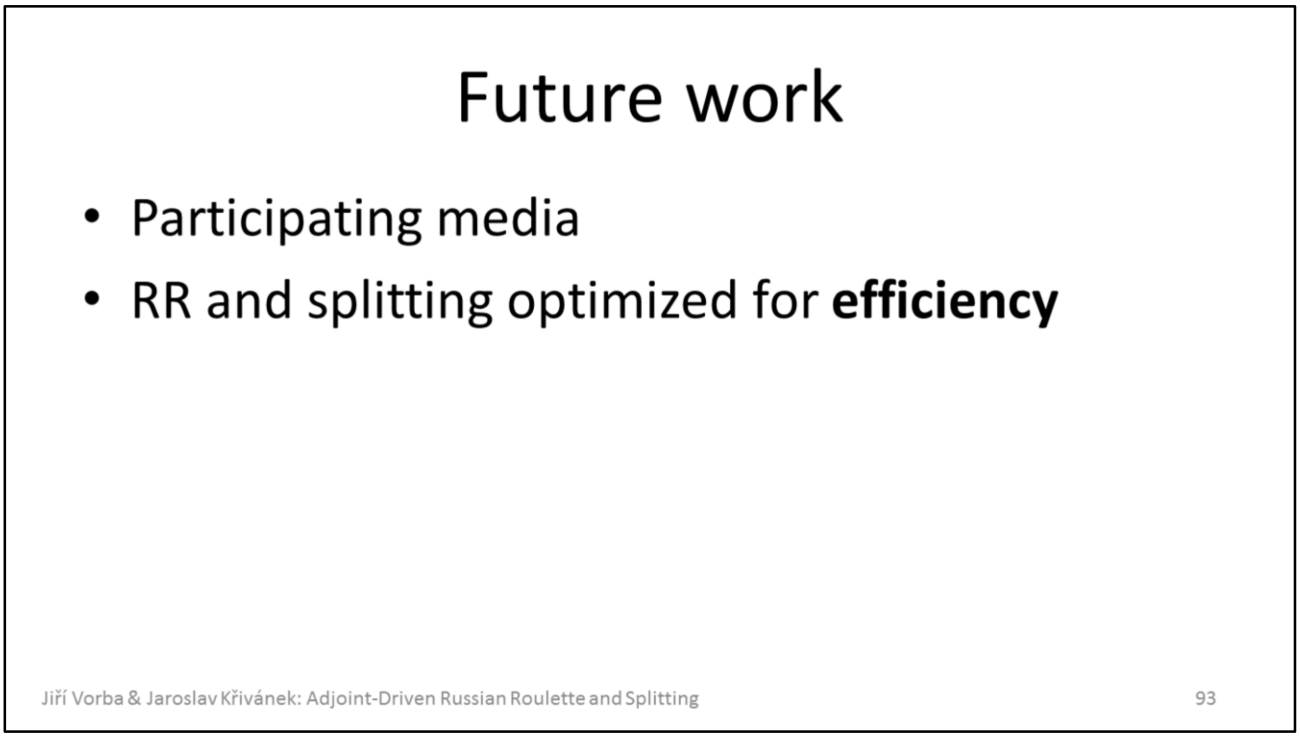
Limitations

- Delayed splitting
- Inaccurate estimates can increase variance



Highly inaccurate adjoint or pixel measurement estimates can increase variance.

Because in our implementation we based our pre-computation on the same principles as photon mapping we can get highly inaccurate estimates due to light leaks.



Future work

- Participating media
- RR and splitting optimized for **efficiency**

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 93

It would be interesting to further explore our approach with respect to efficiency.

At this moment we do not take the cost of sampled rays into account.

Conclusion

Revision of Russian roulette and splitting in light transport

We believe that the main contribution of our work is the revision of the Russian roulette and splitting that has been used in unchanged form in light transport simulation over the last 25 years.

We achieved significant improvement in rendering of indirect illumination and we believe that there is still space for further research.

Acknowledgements

Funding

- Charles University in Prague
- Czech Science Foundation

Scenes

- Ludvík Koutný
- Jaakko Lehtinen et al.
- Frank Meinl
- Toshiya Hachisuka

Proofreading

- Iliyan Georgiev
- Oskar Elek
- Peter Pearson

We would like to thank the following people and institutions for their help and funding.

Source code

Demo in (Coming soon) Mitsuba Physically Based Renderer

http://cgg.mff.cuni.cz/~jirka/papers/2016/adrrs/index.htm

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 96

We based our implementation on Mitsuba renderer and we will publish the source code.

Thank you!

Demo in

(Coming soon)

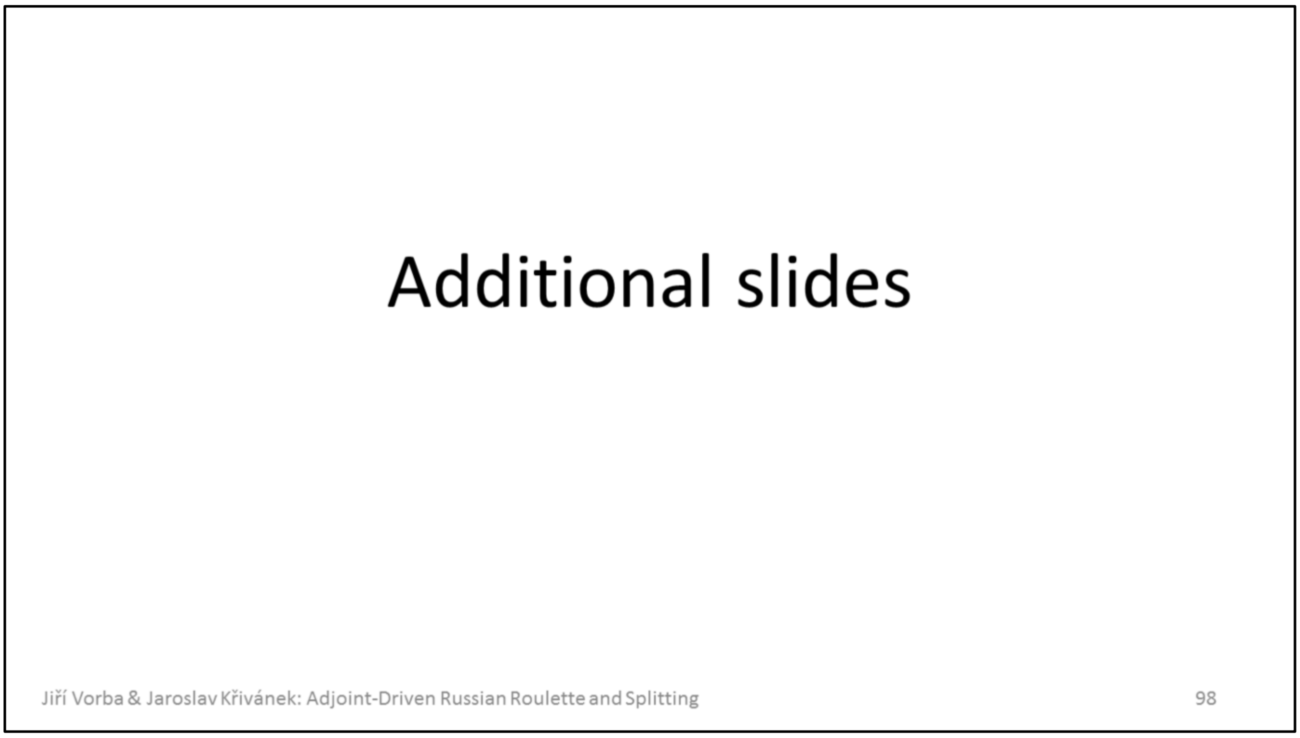
Mitsuba

Physically Based Renderer

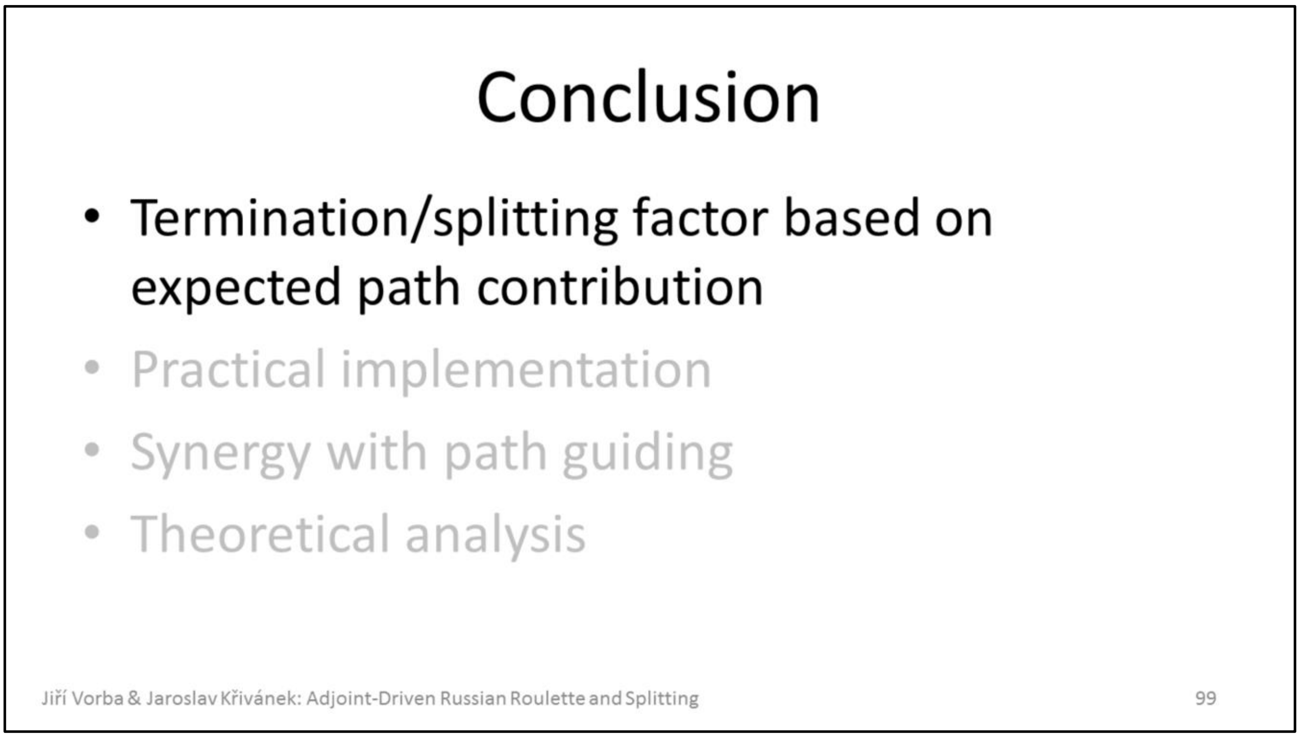
http://cgg.mff.cuni.cz/~jirka/papers/2016/adrrs/index.htm

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 97

Thank you.



Additional slides



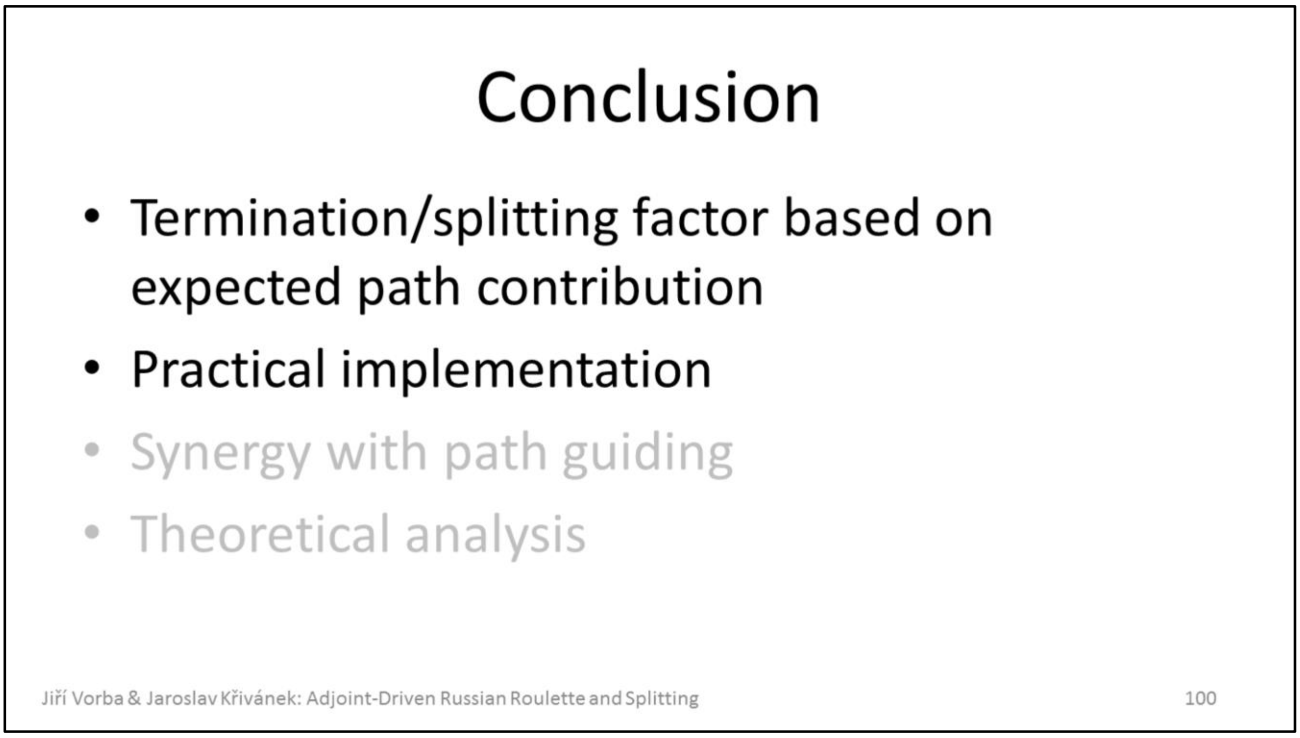
Conclusion

- Termination/splitting factor based on expected path contribution
- Practical implementation
- Synergy with path guiding
- Theoretical analysis

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 99

Specifically our technical contribution is that

we suggest termination/splitting factor based on approximation of adjoint solution and current path weight as opposed to local reflectance properties in the state-of-the-art methods.

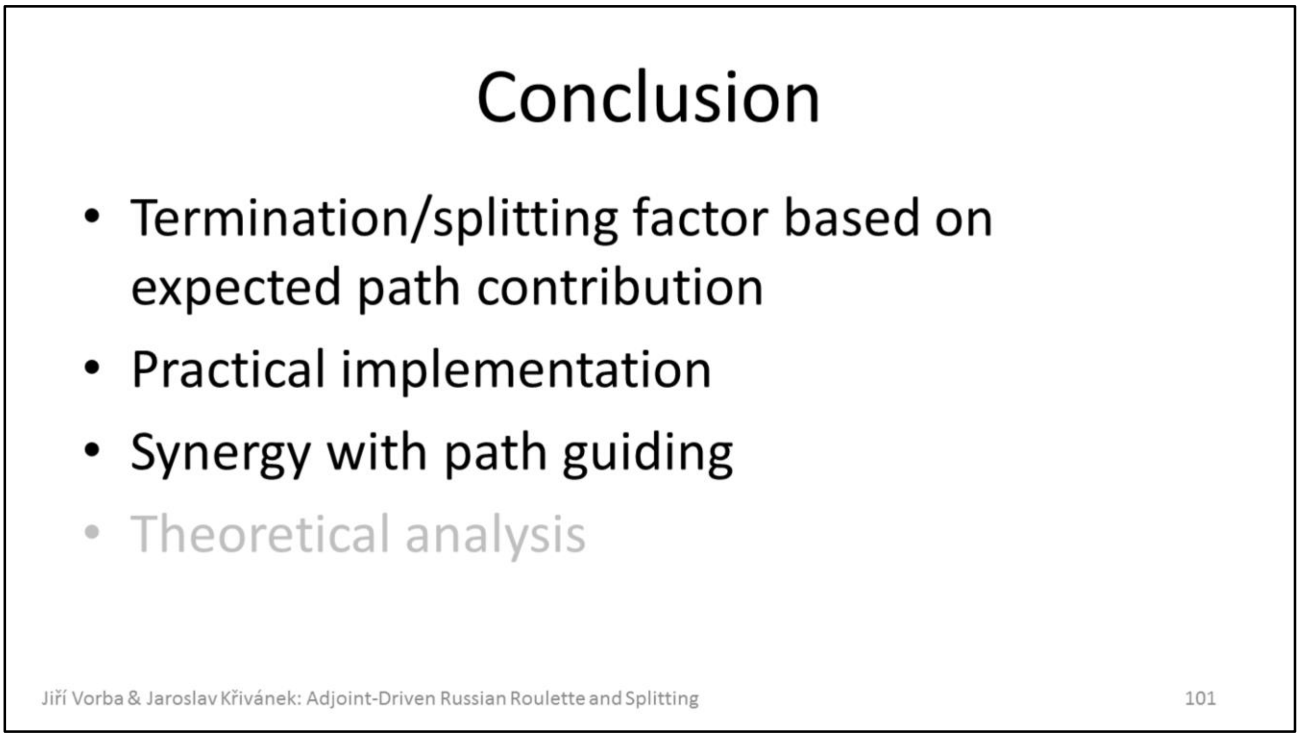


Conclusion

- Termination/splitting factor based on expected path contribution
- Practical implementation
- Synergy with path guiding
- Theoretical analysis

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 100

We achieved practical implementation of the suggested approach that is robust even in the presence of glossy and specular materials in the scene.

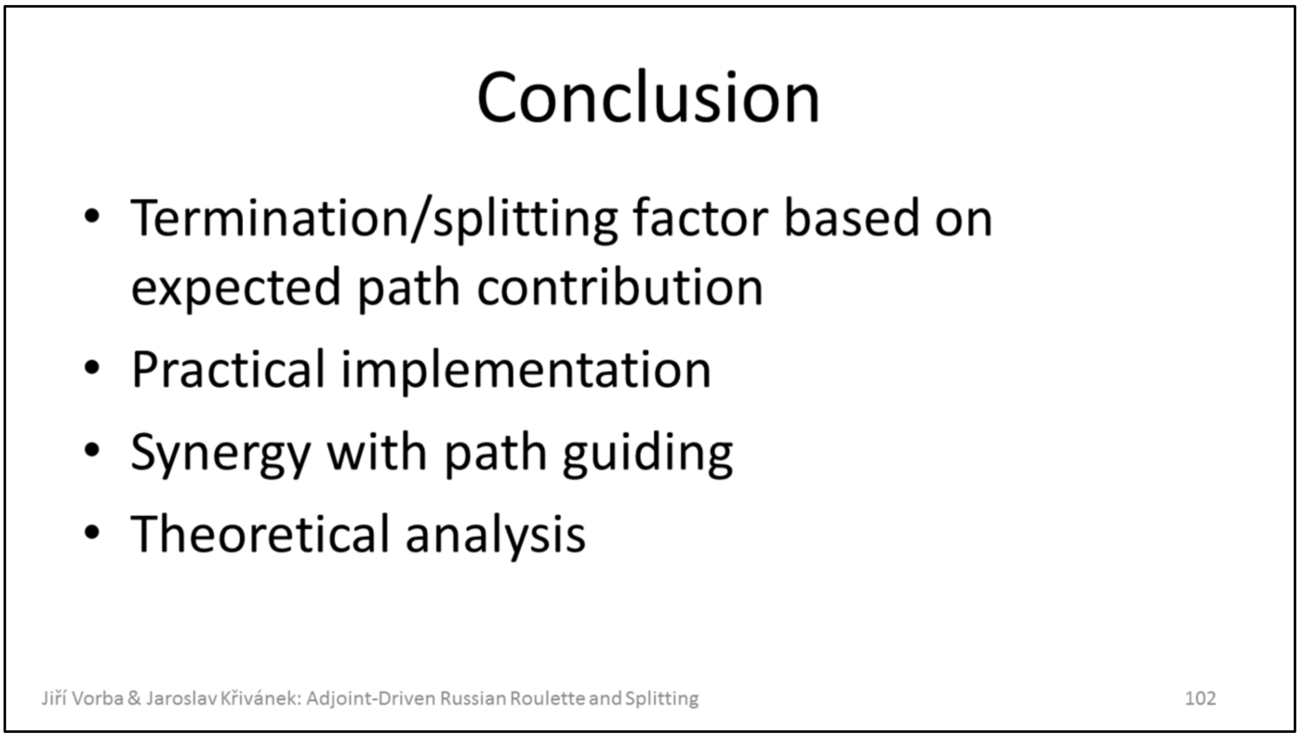


Conclusion

- Termination/splitting factor based on expected path contribution
- Practical implementation
- Synergy with path guiding
- Theoretical analysis

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 101

We showed that it works in synergy with path guiding methods.

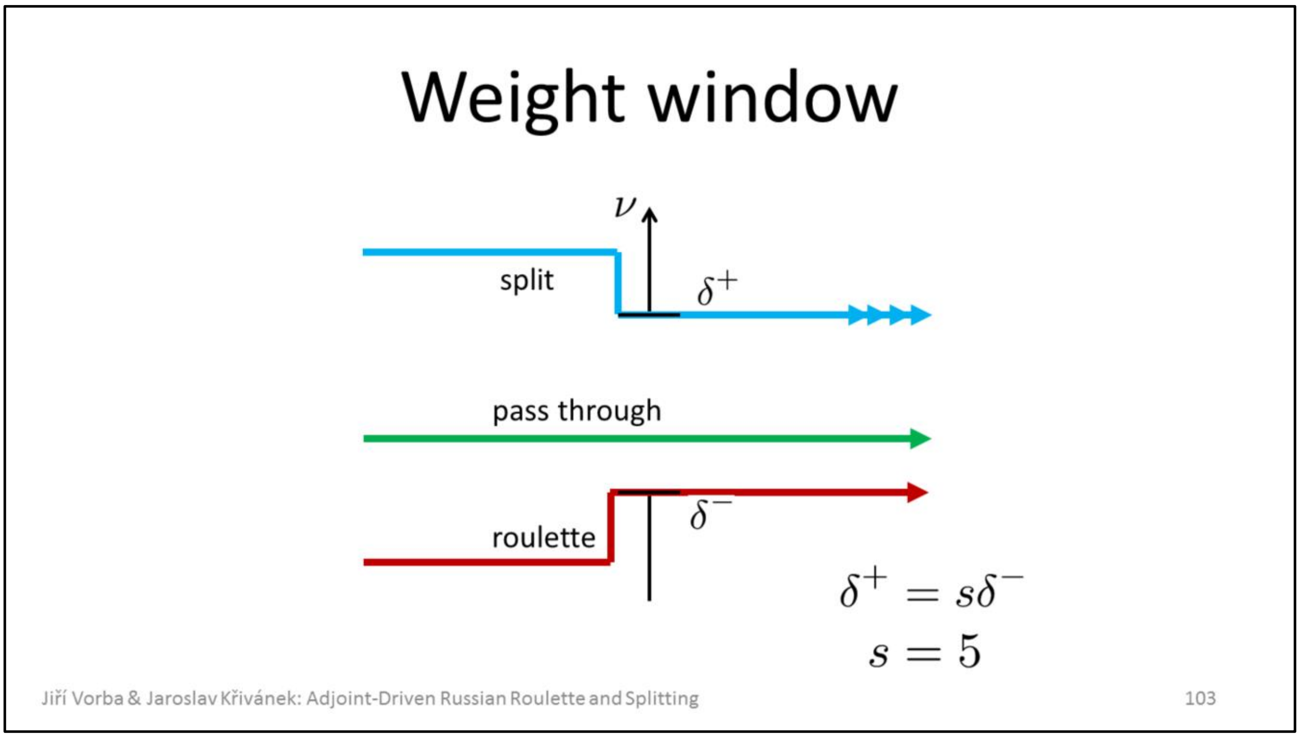


Conclusion

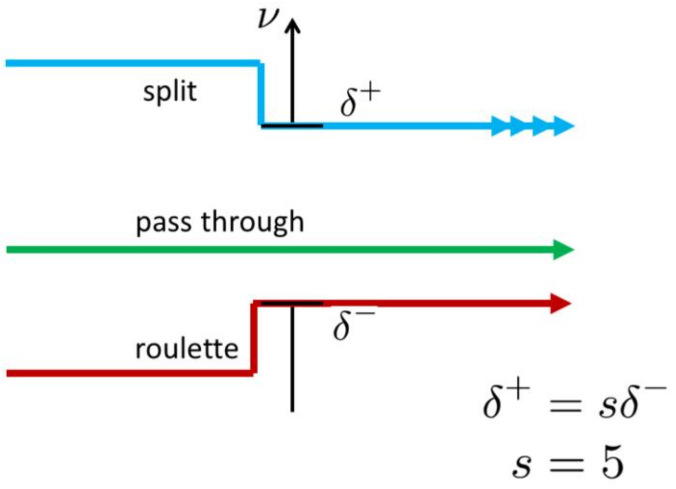
- Termination/splitting factor based on expected path contribution
- Practical implementation
- Synergy with path guiding
- Theoretical analysis

Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 102

And we provided a theoretical analysis of importance sampling properties of our approach which also explains its limitations.

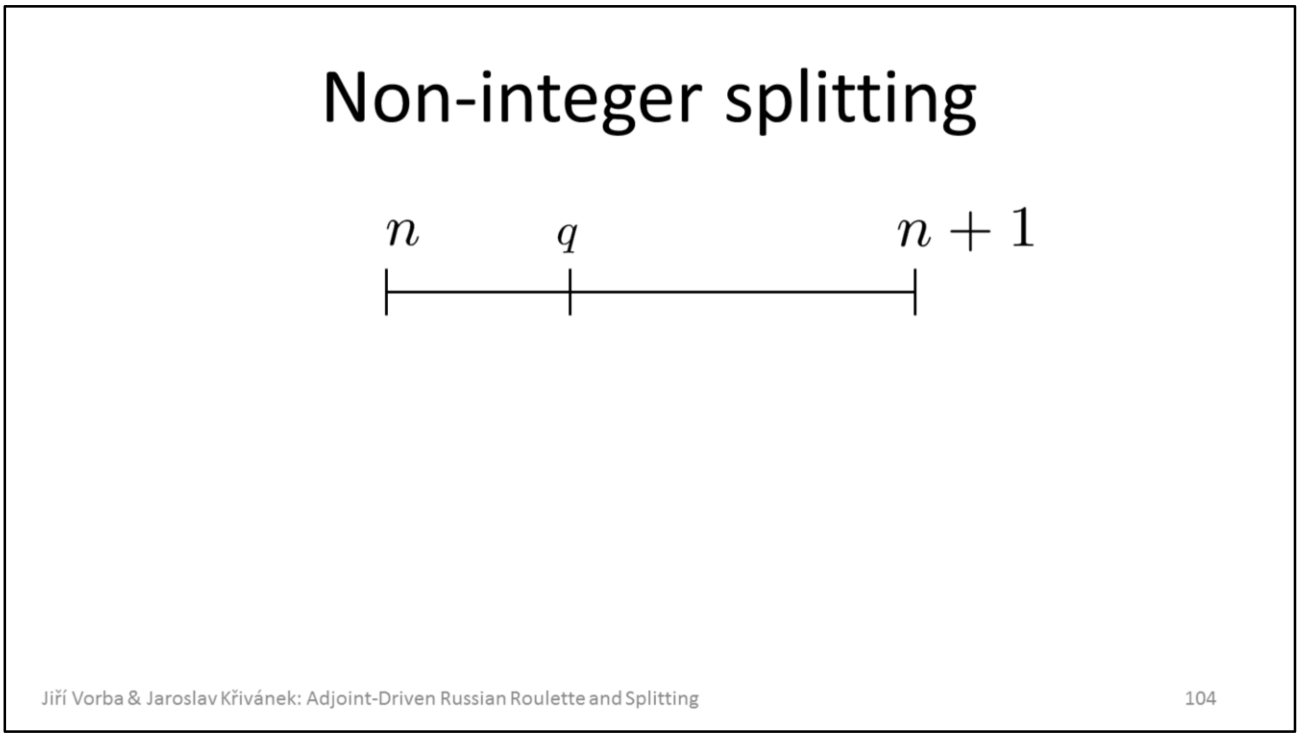


Weight window

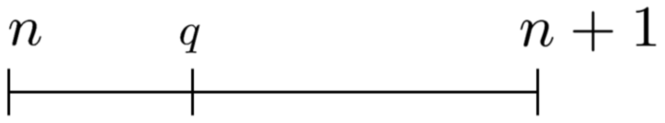


We set the weight-window size so that the upper boundary is five times the lower boundary.

We experimented with different window sizes and it turned out that the method is not particularly sensitive to that parameter.



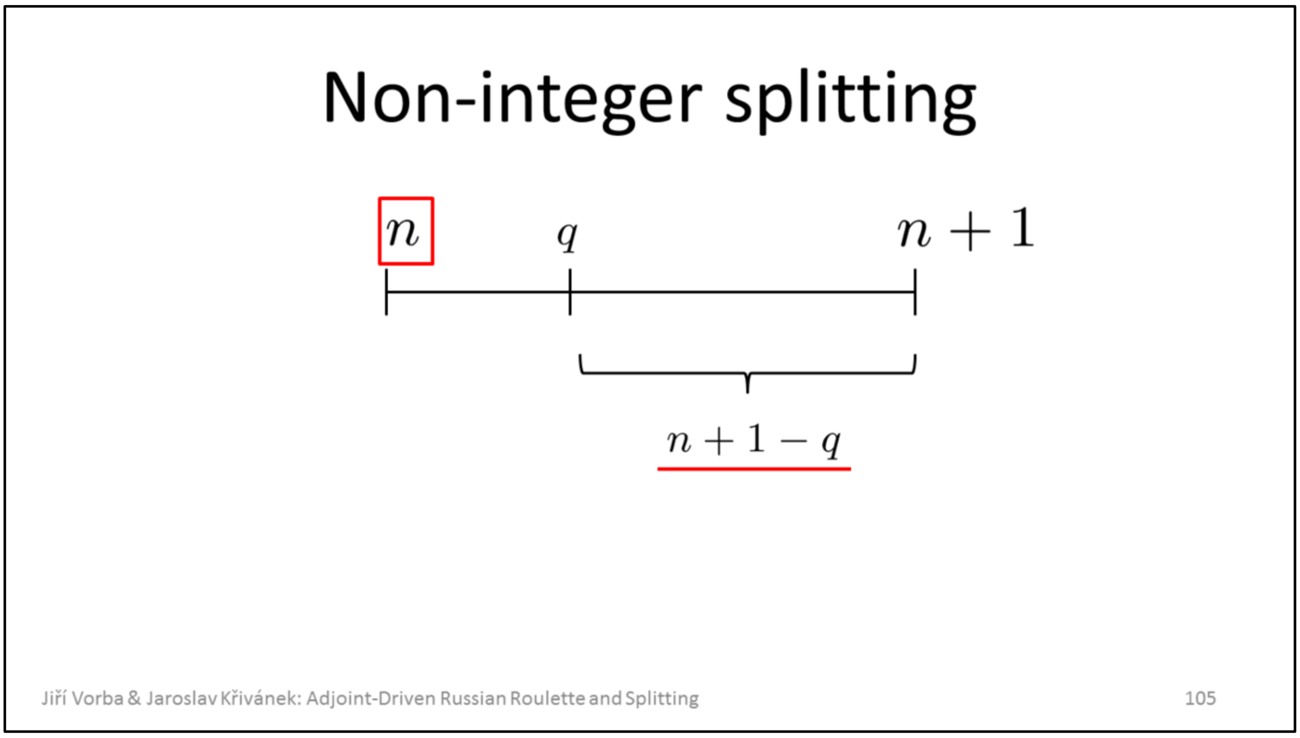
Non-integer splitting



Jiří Vorba & Jaroslav Křivánek: Adjoint-Driven Russian Roulette and Splitting 104

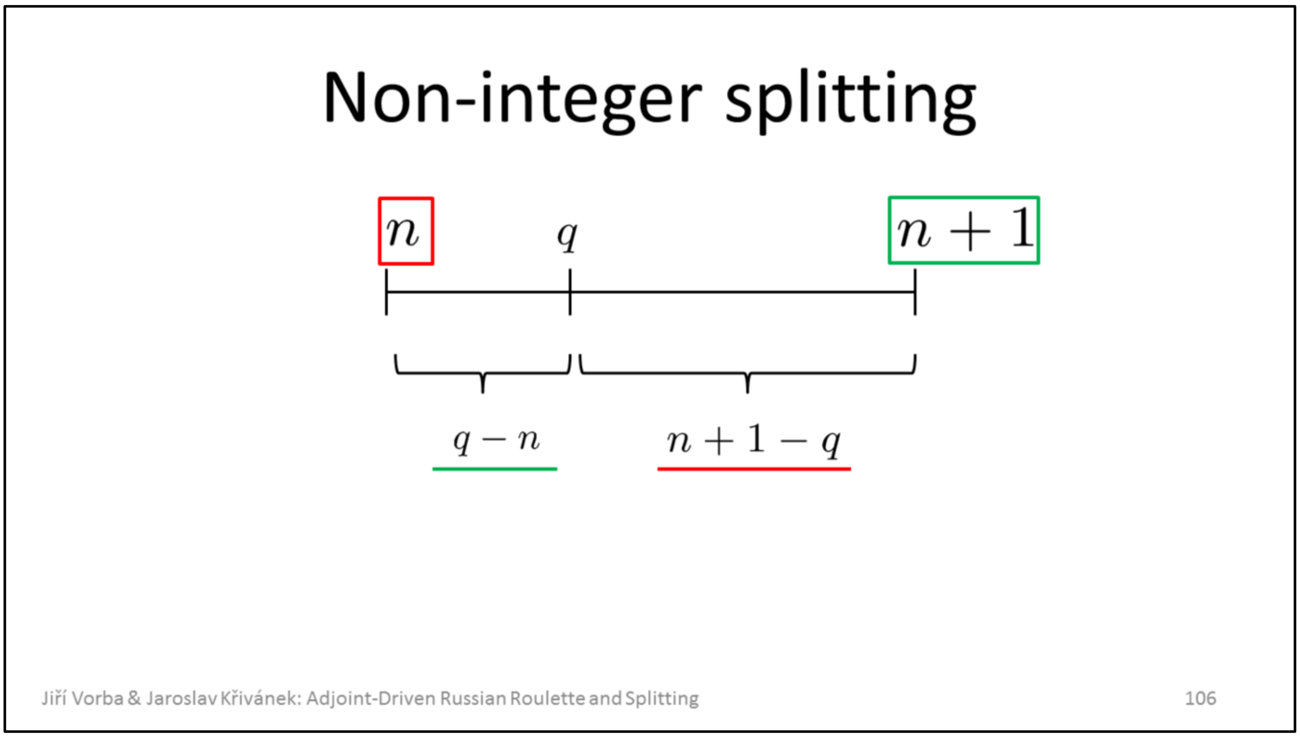
To cope with the non-integer values in the splitting factor q,

we probabilistically split into ...



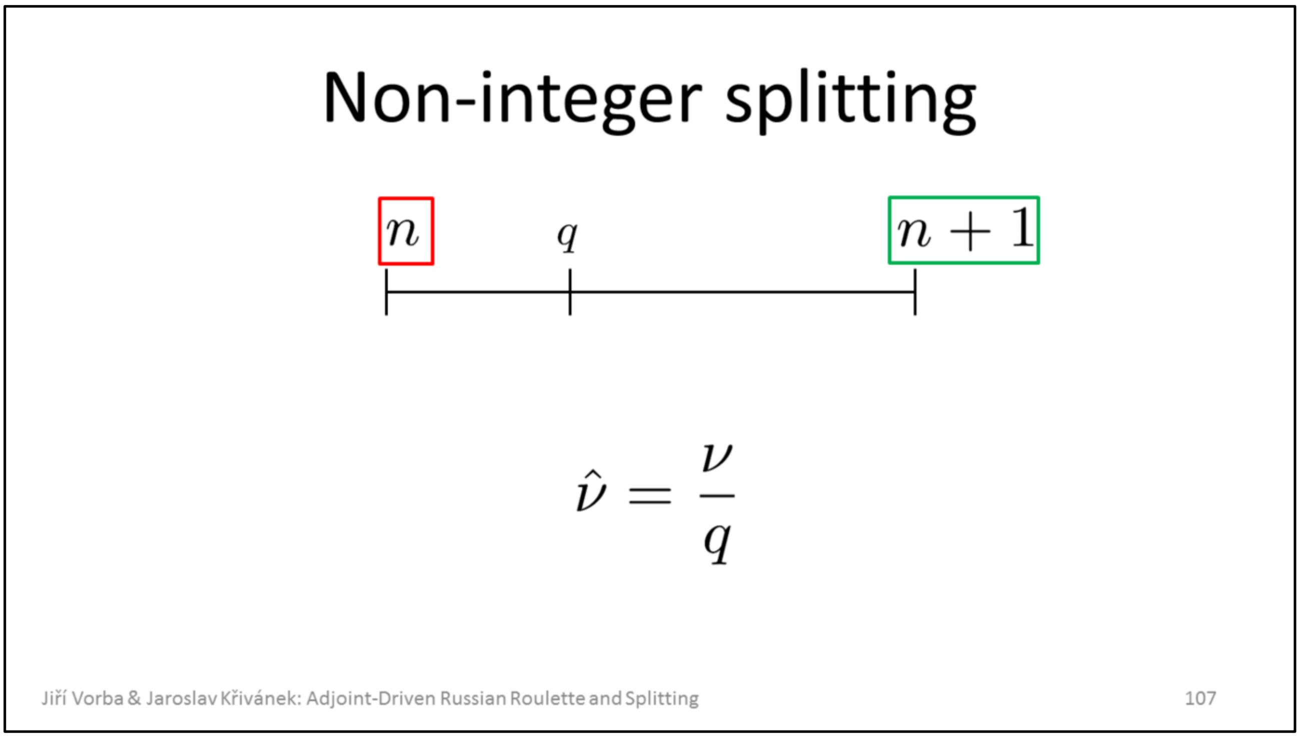
Non-integer splitting

… either closest smaller integer n with the probability n + 1 – q …

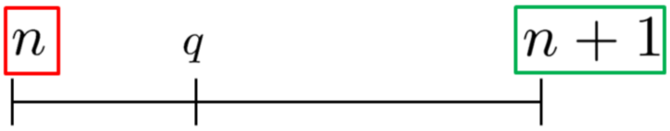


Non-integer splitting

… or into n + 1 paths with probability q – n.



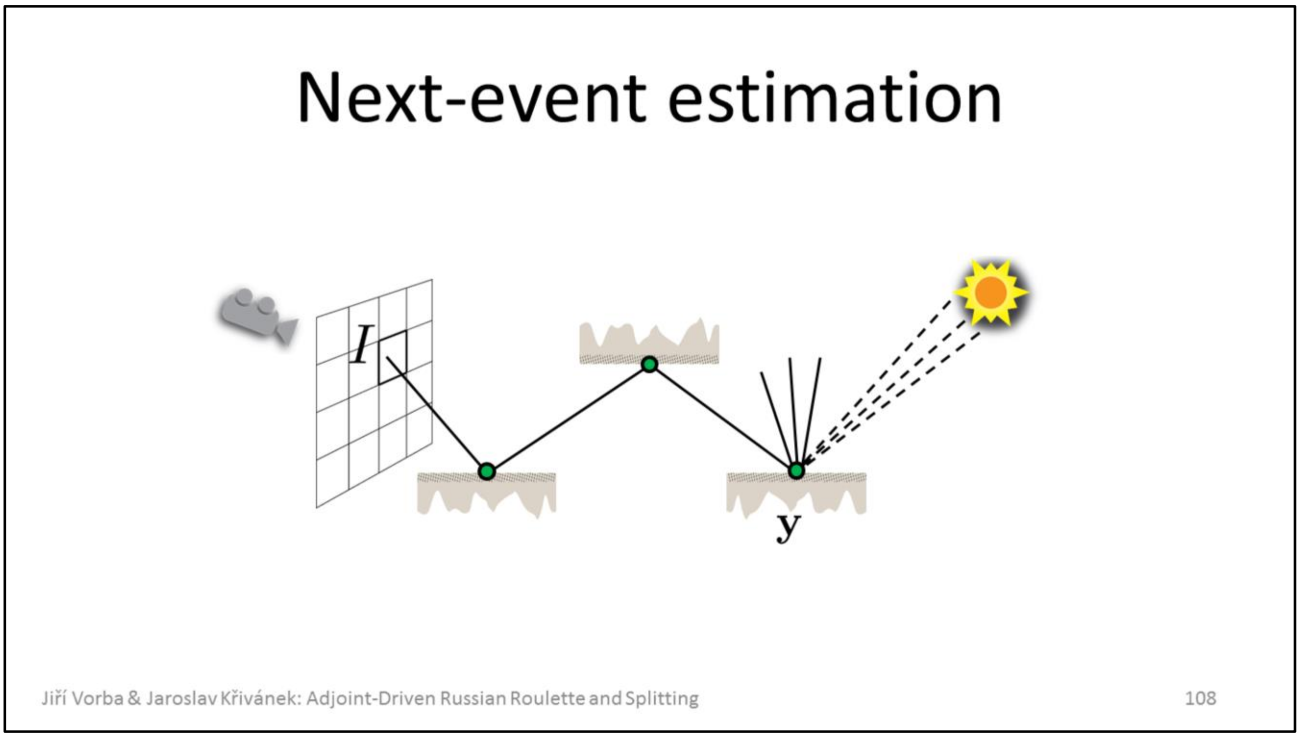
Non-integer splitting



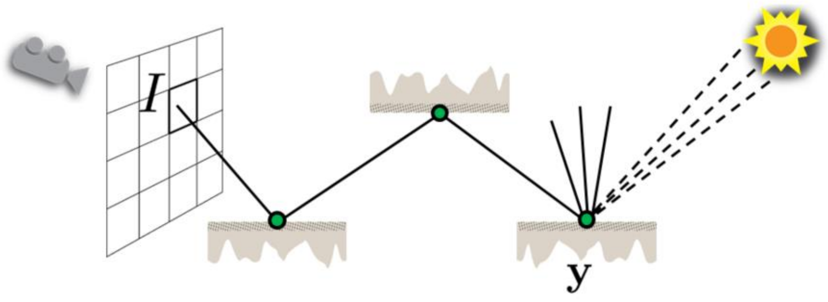
$$\hat{\nu} = \frac{\nu}{q}$$

The weight of each new path is computed using the real split factor q and is the same regardless of whether we happen to split into n or n+1 paths.

This approach is unbiased in the expected value sense.



Next-event estimation



In our implementation when path is split into n rays in a vertex y we use exactly the same number of next-event estimation samples in that vertex.

Reasoning behind this decision is motivated by the fact that splitting happens when we enter more important regions where expected contribution is high.

So the splitting is also very likely to happen when path enters a directly lit region.