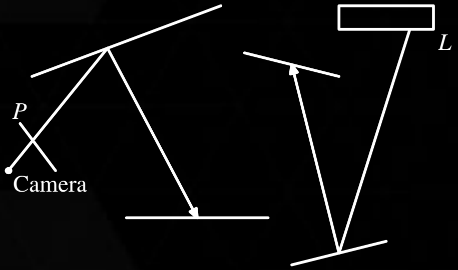
Path Space Filtering

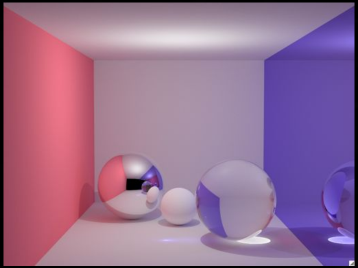
Alexander Keller, Ken Dahm, and Nikolaus Binder

Consistent Image Synthesis

Light transport simulation

- trace light transport path segments from the light and camera

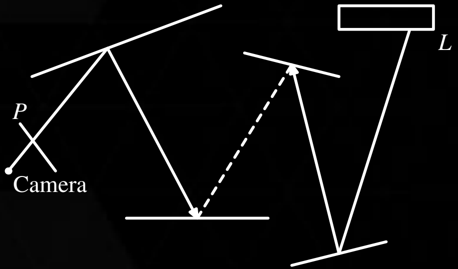


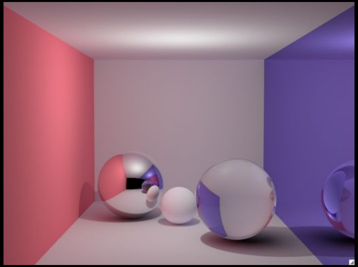


Consistent Image Synthesis

Light transport simulation

- trace light transport path segments from the light and camera,
  - connect them by shadow rays

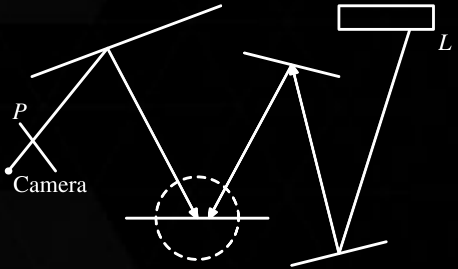


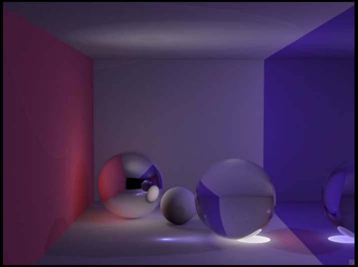


Consistent Image Synthesis

Light transport simulation

- trace light transport path segments from the light and camera,
  - connect them by shadow rays and/or proximity

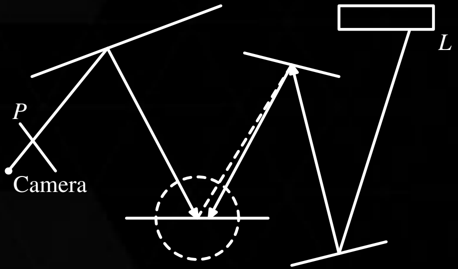


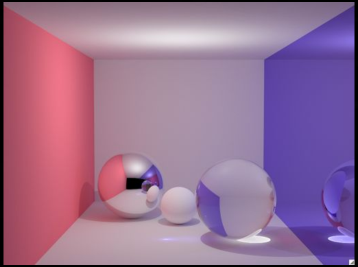


Consistent Image Synthesis

Light transport simulation

- trace light transport path segments from the light and camera,
  - connect them by shadow rays and/or proximity





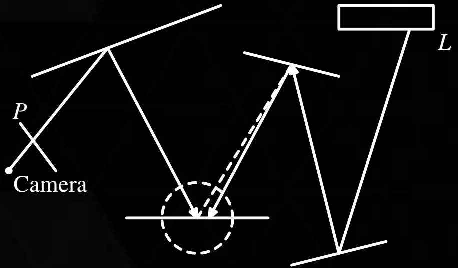
and sum up the weighted contributions of all light transport paths

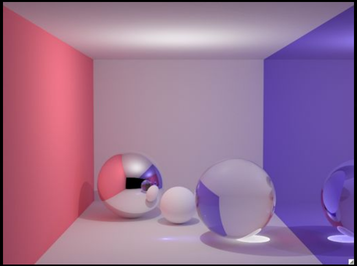
NVIDIA.

Consistent Image Synthesis

Light transport simulation

- trace light transport path segments from the light and camera,
  - connect them by shadow rays and/or proximity



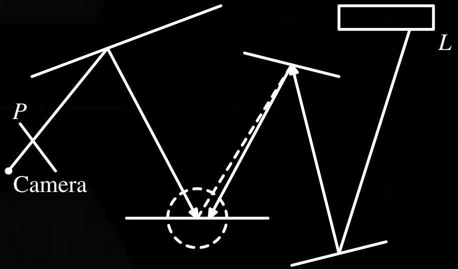


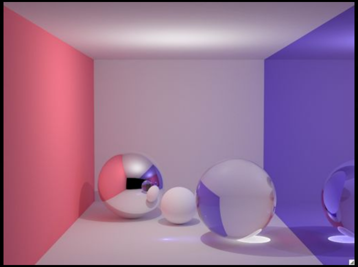
and sum up the weighted contributions of all light transport paths

Consistent Image Synthesis

Light transport simulation

- trace light transport path segments from the light and camera,
  - connect them by shadow rays and/or proximity



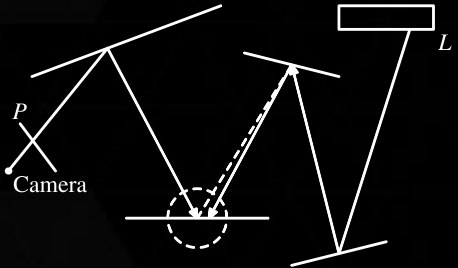


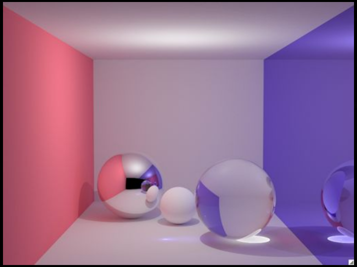
and sum up the weighted contributions of all light transport paths

Consistent Image Synthesis

Light transport simulation

- trace light transport path segments from the light and camera,
  - connect them by shadow rays and/or proximity





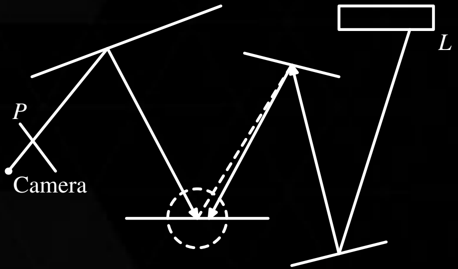
and sum up the weighted contributions of all light transport paths

  - check out the state of the art report "Quasi-Monte Carlo Image Synthesis in a Nutshell"

Consistent Image Synthesis

Light transport simulation

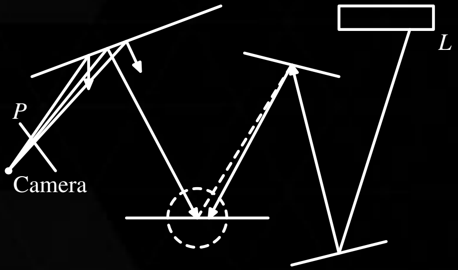
- sum up the weighted contributions of all light transport paths



Consistent Image Synthesis

Light transport simulation

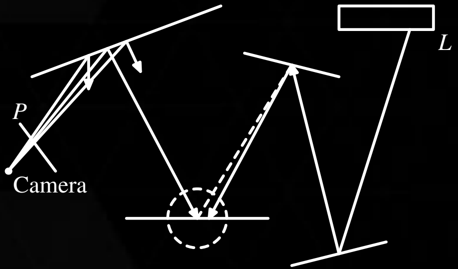
- sum up the weighted contributions of all light transport paths



Consistent Image Synthesis

Light transport simulation

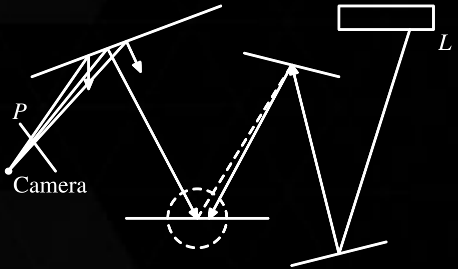
- sum up the weighted contributions of all light transport paths



Consistent Image Synthesis

Light transport simulation

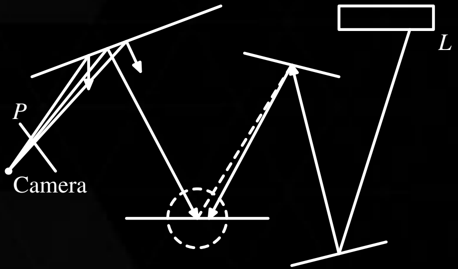
- sum up the weighted contributions of all light transport paths
  - noise vanishing only slowly with increasing number of samples



Consistent Image Synthesis

Light transport simulation

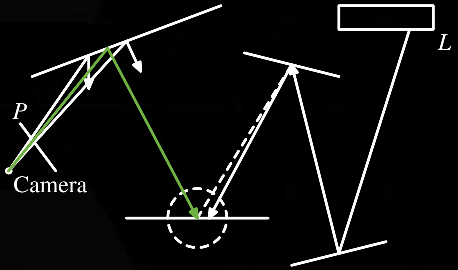
- sum up the weighted contributions of all light transport paths
  - noise vanishing only slowly with increasing number of samples

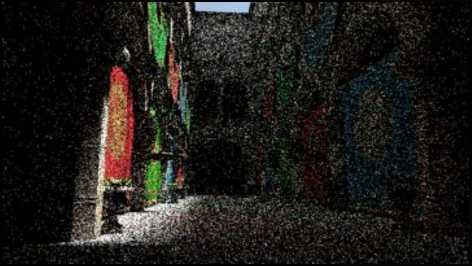


Path Space Filtering

Noise reduction by averaging contributions of close-by vertices

- range search



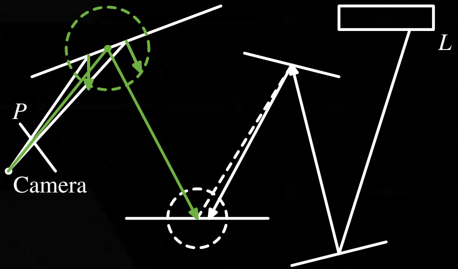


NVIDIA

Path Space Filtering

Noise reduction by averaging contributions of close-by vertices

- range search
  - initial radius r_0 proportional to distance of ray origin and location of averaging

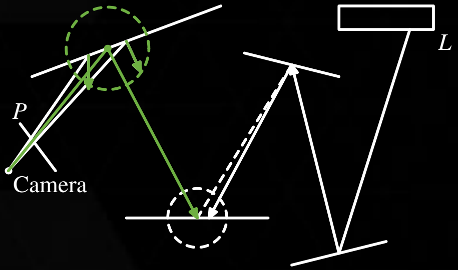


NVIDIA

Path Space Filtering

Noise reduction by averaging contributions of close-by vertices

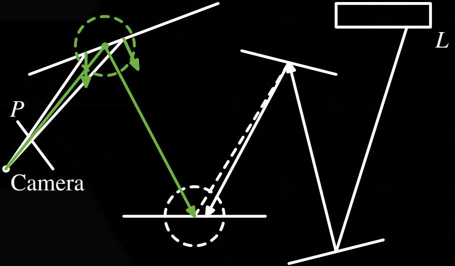
- range search
  - initial radius r_0 proportional to distance of ray origin and location of averaging



Path Space Filtering

Noise reduction by averaging contributions of close-by vertices

- range search with radius $r(n)$ decreasing with number n of samples
  - initial radius r_0 proportional to distance of ray origin and location of averaging

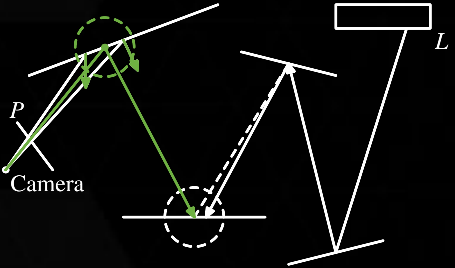


$$r(n) := \frac{r_0}{\sqrt{n^\delta}} \text{ for } \delta \in (0,1)$$

Path Space Filtering

Noise reduction by averaging contributions of close-by vertices

- range search with radius $r(n)$ decreasing with number n of samples
  - initial radius r_0 proportional to distance of ray origin and location of averaging

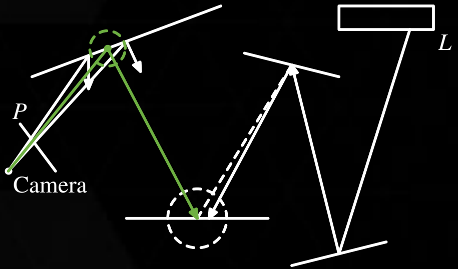


$$r(n) := \frac{r_0}{\sqrt{n^\delta}} \text{ for } \delta \in (0,1)$$

Path Space Filtering

Noise reduction by averaging contributions of close-by vertices

- range search with radius $r(n)$ decreasing with number n of samples
  - initial radius r_0 proportional to distance of ray origin and location of averaging

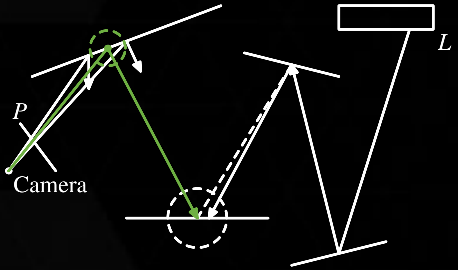


$$r(n) := \frac{r_0}{\sqrt{n^\delta}} \text{ for } \delta \in (0,1)$$

Path Space Filtering

Noise reduction by averaging contributions of close-by vertices

- range search with radius $r(n)$ decreasing with number n of samples
  - initial radius r_0 proportional to distance of ray origin and location of averaging



$$r(n) := \frac{r_0}{\sqrt{n^\delta}} \text{ for } \delta \in (0,1)$$

Path Space Filtering

Algorithm

- enumerate contiguous blocks of b^m light transport paths
  - for each light transport path store a selected vertex x_i with
    - its throughput α_i of the camera path segment and
    - its contribution c_i of the light path segment

NVIDIA

Path Space Filtering

Algorithm

- enumerate contiguous blocks of b^m light transport paths
  - for each light transport path store a selected vertex x_i with
    - its throughput α_i of the camera path segment and
    - its contribution c_i of the light path segment
  - for each vertex x_i determine the weighted average contribution

$$\bar{c}_i := \frac{\sum_{j=0}^{b^m-1} \chi_{\mathscr{B}(n)}\left(x_{s_i+j} - x_i\right) \cdot w_{i,j} \cdot c_{s_i+j}}{\sum_{j=0}^{b^m-1} \chi_{\mathscr{B}(n)}\left(x_{s_i+j} - x_i\right) \cdot w_{i,j}}$$

of all vertices x_{s_i+j} in the block starting at index $s_i := \left\lfloor \frac{i}{b^m} \right\rfloor b^m$

inside the ball $\mathscr{B}$ of radius $r(n)$ around x_i

Path Space Filtering

Algorithm

- enumerate contiguous blocks of b^m light transport paths
  - for each light transport path store a selected vertex x_i with
    - its throughput α_i of the camera path segment and
    - its contribution c_i of the light path segment
  - for each vertex x_i determine the weighted average contribution

$$\bar{c}_i := \frac{\sum_{j=0}^{b^m-1} \chi_{\mathscr{B}(n)}\left(x_{s_i+j} - x_i\right) \cdot w_{i,j} \cdot c_{s_i+j}}{\sum_{j=0}^{b^m-1} \chi_{\mathscr{B}(n)}\left(x_{s_i+j} - x_i\right) \cdot w_{i,j}}$$

    of all vertices x_{s_i+j} in the block starting at index $s_i := \left\lfloor \frac{i}{b^m} \right\rfloor b^m$

    inside the ball $\mathscr{B}$ of radius $r(n)$ around x_i
  - accumulate $\alpha_i \bar{c}_i$ instead of $\alpha_i c_i$

Path Space Filtering

Algorithm

- enumerate contiguous blocks of b^m light transport paths
  - for each light transport path store a selected vertex x_i with
    - its throughput α_i of the camera path segment and
    - its contribution c_i of the light path segment
  - for each vertex x_i determine the weighted average contribution

$$\bar{c}_i := \frac{\sum_{j=0}^{b^m-1} \chi_{\mathscr{B}(n)}\left(x_{s_i+j} - x_i\right) \cdot w_{i,j} \cdot c_{s_i+j}}{\sum_{j=0}^{b^m-1} \chi_{\mathscr{B}(n)}\left(x_{s_i+j} - x_i\right) \cdot w_{i,j}}$$

    of all vertices x_{s_i+j} in the block starting at index $s_i := \left\lfloor \frac{i}{b^m} \right\rfloor b^m$

    inside the ball $\mathscr{B}$ of radius $r(n)$ around x_i

  - accumulate $\alpha_i \bar{c}_i$ instead of $\alpha_i c_i$
- consistent due to $\lim_{n\to\infty} \bar{c}_i = c_i$, because $\lim_{n\to\infty} r(n) = 0$

Path Space Filtering

Mimicking trajectory splitting

- only include contributions from x_{s_i+j} that could have been generated in x_i
  - input of 16 path space samples per pixel

Path Space Filtering

Mimicking trajectory splitting

- only include contributions from x_{s_i+j} that could have been generated in x_i
  - input of 16 path space samples per pixel filtered without weighting

Path Space Filtering

Mimicking trajectory splitting

- only include contributions from x_{s_i+j} that could have been generated in x_i
  - input of 16 path space samples per pixel accounting for similar normals
    - weights $w_{i,j}$ in fact are similarity heuristics from irradiance interpolation

Path Space Filtering

Mimicking trajectory splitting

- only include contributions from x_{s_i+j} that could have been generated in x_i
  - input of 16 path space samples per pixel accounting for similar normals, reflectance
    - weights $w_{i,j}$ in fact are similarity heuristics from irradiance interpolation

Path Space Filtering

Mimicking trajectory splitting

- only include contributions from x_{s_i+j} that could have been generated in x_i
  - input of 16 path space samples per pixel accounting for similar normals, reflectance, and visibility
    - weights $w_{i,j}$ in fact are similarity heuristics from irradiance interpolation

NVIDIA.

Path Space Filtering

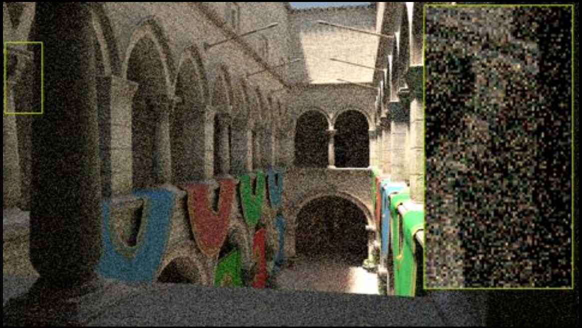
Mimicking trajectory splitting

- only include contributions from x_{s_i+j} that could have been generated in x_i
  - input of 16 path space samples per pixel accounting for similar normals, reflectance, and visibility
    - weights $w_{i,j}$ in fact are similarity heuristics from irradiance interpolation

Path Space Filtering

Mimicking trajectory splitting

- only include contributions from x_{s_i+j} that could have been generated in x_i
  - input of 16 path space samples per pixel

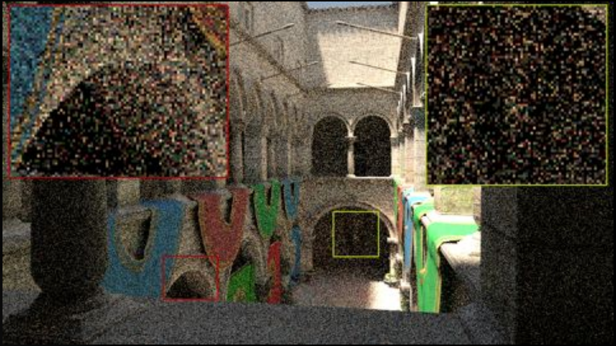


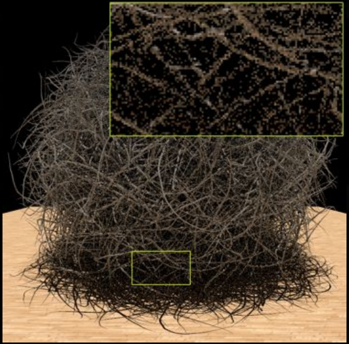
NVIDIA

Path Space Filtering

Select block size b^m as large as possible

- accumulating 16 passes
  - 1 path space sample per pixel

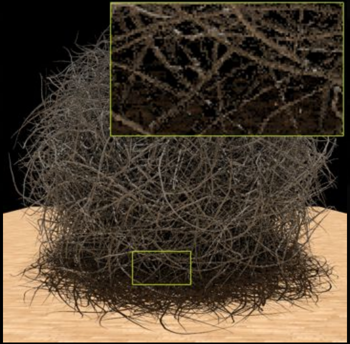




Path Space Filtering

Select block size b^m as large as possible

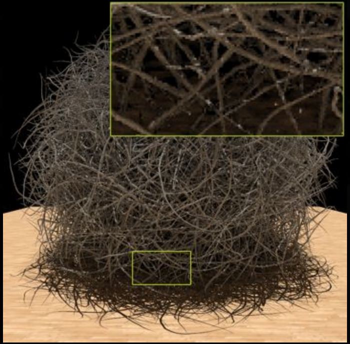
- accumulating 16 passes
  - filtered across 1 path space sample per pixel



Path Space Filtering

Select block size b^m as large as possible

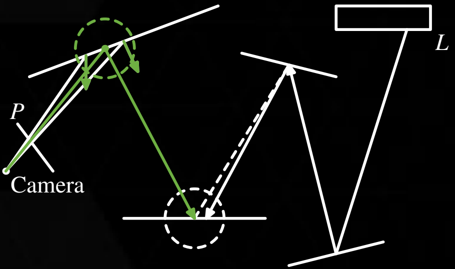
- accumulating 2 passes
  - filtered across 8 path space samples per pixel

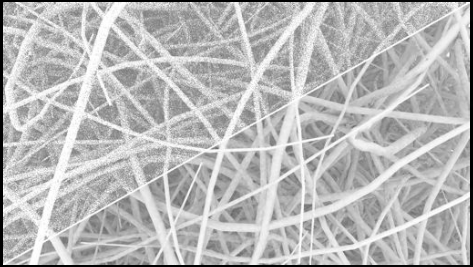


Path Space Filtering

Example applications

- results for
  - ambient occlusion

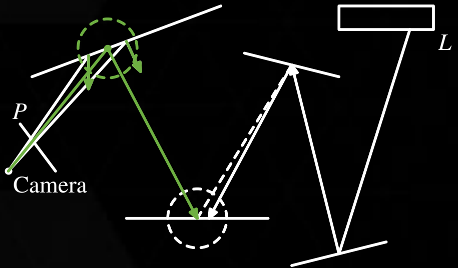


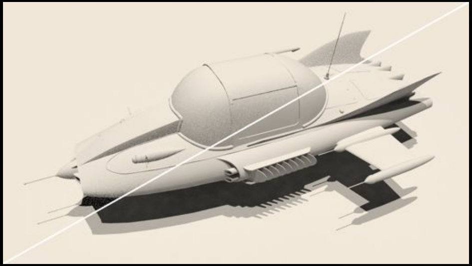


Path Space Filtering

Example applications

- results for
  - ambient occlusion, shadows

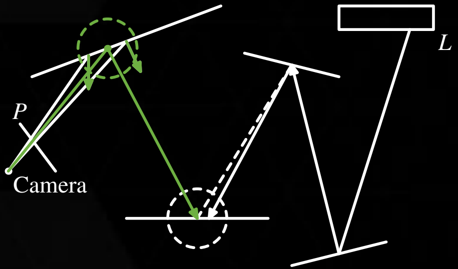




Path Space Filtering

Example applications

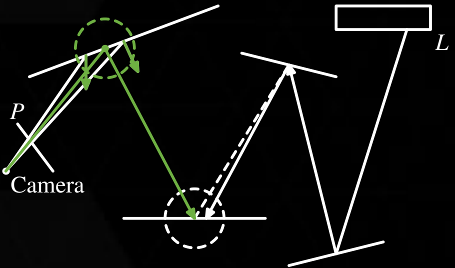
- results for
  - ambient occlusion, shadows, subsurface scattering



Path Space Filtering

Example applications

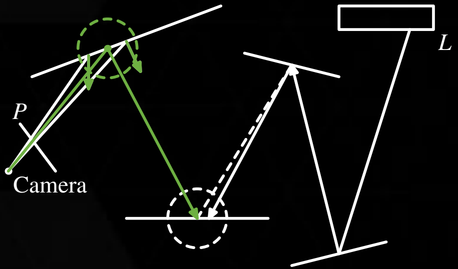
- results for
  - ambient occlusion, shadows, subsurface scattering, light transport simulation



Path Space Filtering

Example applications

- results for
  - ambient occlusion, shadows, subsurface scattering, light transport simulation across multiple cameras

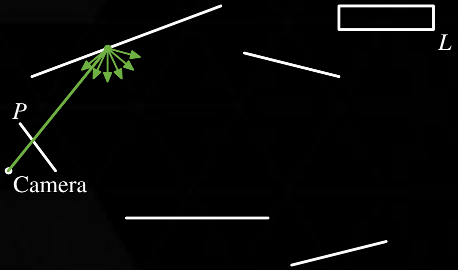


NVIDIA

Extensions of Path Space Filtering

Variable rate shading

- trajectory splitting is costly



Extensions of Path Space Filtering

Variable rate shading

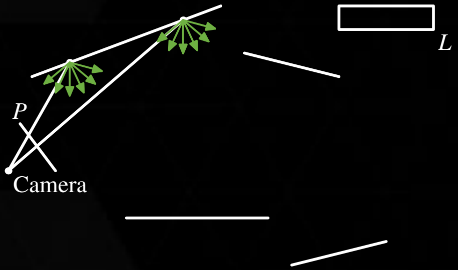
- trajectory splitting is costly

NVIDIA

Extensions of Path Space Filtering

Variable rate shading

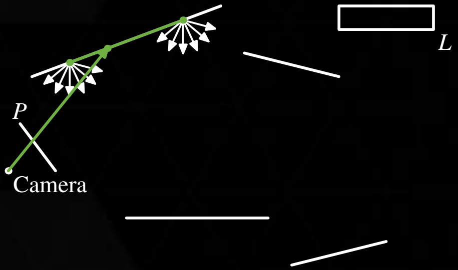
- trajectory splitting is costly, however, one may try to amortize cost by interpolation



Extensions of Path Space Filtering

Variable rate shading

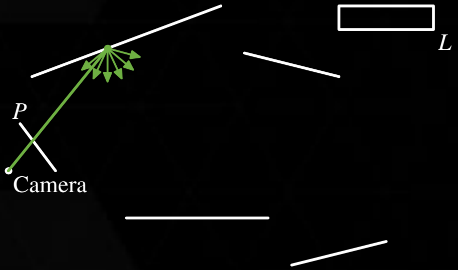
- trajectory splitting is costly, however, one may try to amortize cost by interpolation



Extensions of Path Space Filtering

Variable rate shading

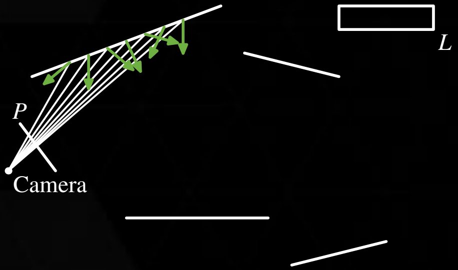
- mimicking trajectory splitting



Extensions of Path Space Filtering

Variable rate shading

- mimicking trajectory splitting by path space filtering is consistent

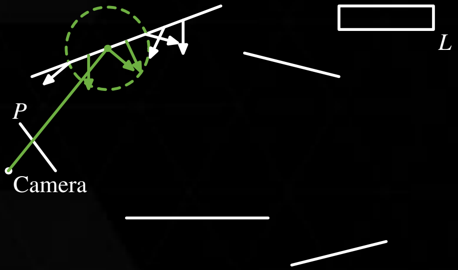


NVIDIA

Extensions of Path Space Filtering

Variable rate shading

- mimicking trajectory splitting by path space filtering is consistent

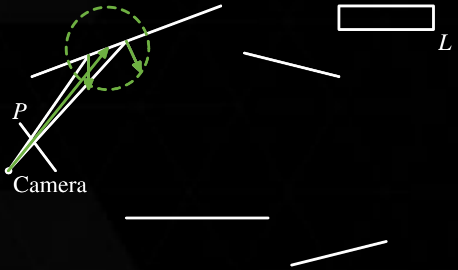


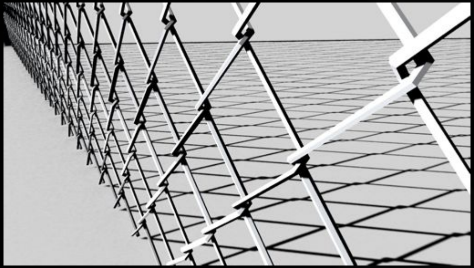
NVIDIA

Extensions of Path Space Filtering

Variable rate shading

- mimicking trajectory splitting by path space filtering is consistent
  - and allows for subsampling



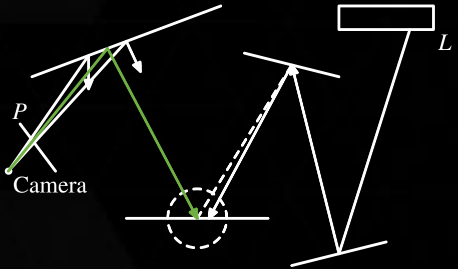


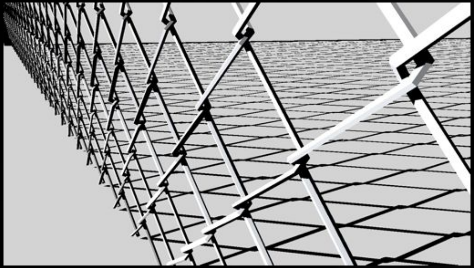
NVIDIA

Extensions of Path Space Filtering

Variable rate shading

- mimicking trajectory splitting by path space filtering is consistent
  - and allows for subsampling

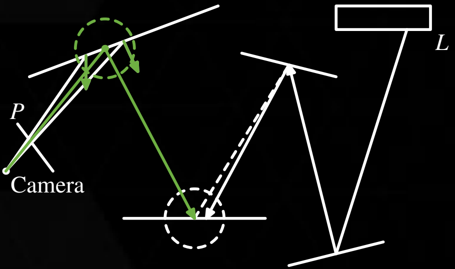




Extensions of Path Space Filtering

Iteration

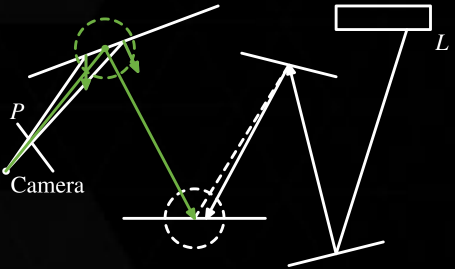
- input at one sample per pixel



Extensions of Path Space Filtering

Iteration

- input at one sample per pixel filtered once

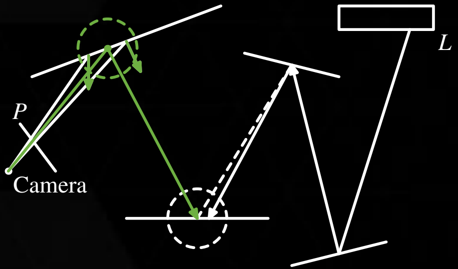


NVIDIA

Extensions of Path Space Filtering

Iteration

- input at one sample per pixel
  - after iterating the weighted average four times



Path Space Filtering

Consistent and efficient variance reduction method

- stop-and-go rendering paradigm
  - no persistent artifacts
  - no re-render, especially for animations

Path Space Filtering

Consistent and efficient variance reduction method

- stop-and-go rendering paradigm
  - no persistent artifacts
  - no re-render, especially for animations
- practical
  - orthogonal to any path space sampling based algorithm
  - two parameters

Path Space Filtering

Consistent and efficient variance reduction method

- stop-and-go rendering paradigm
  - no persistent artifacts
  - no re-render, especially for animations
- practical
  - orthogonal to any path space sampling based algorithm
  - two parameters
- dual to photon mapping
  - weighted average vs. division by disk area
  - similar particle inclusion heuristics and transient artifacts

Courtesies

Models

- Cornell Box by Kevin Beason in smallpt
- Sponza by Marko Dabrovic, modified by Crytek
- Hairball by Samuli Laine
- Trees by `http://www.laubwerk.com`
- Lucy Statue from the Stanford 3D Scanning Repository
- San Miguel by Guillermo M. Leal Llaguano
- Fence by Chris Wyman

Note that image quality may have suffered from PDF compression.