Global Illumination I

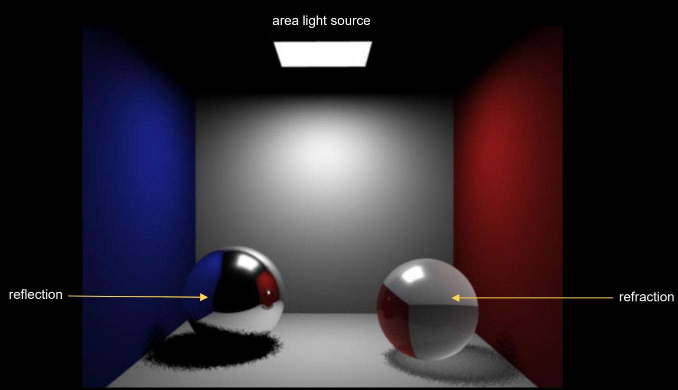
70001 – Advanced Computer Graphics: Photographic Image Synthesis

Abhijeet Ghosh

Lecture 14, Feb. 20th 2024

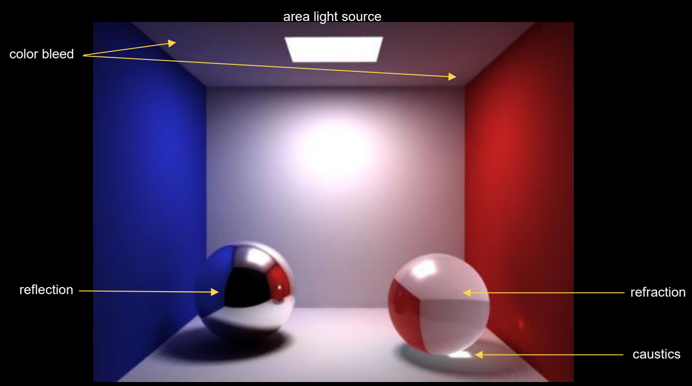
1

Cornell Box – direct illumination



2

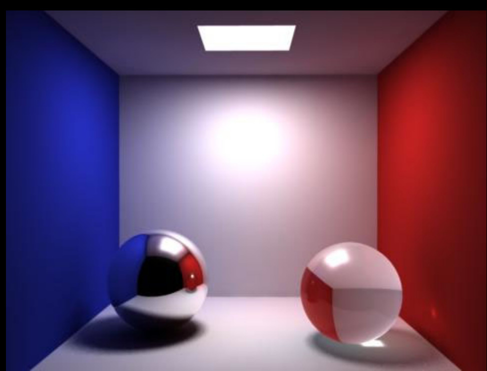
Cornell Box – global illumination



3

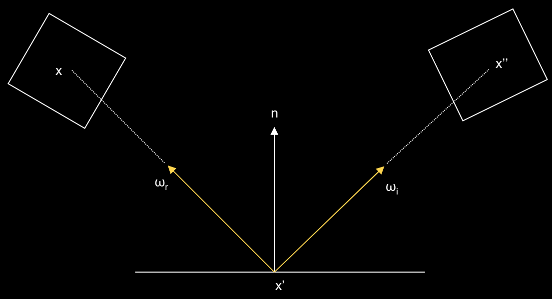
Rendering Equation [Kajiya 86]

- $L_r(x, \omega_r) = L_e(x, \omega_r) + \int_\Omega f_r(x, \omega_r, \omega_i)\, L_i(x, \omega_i) \cos\theta_i \, d\omega_i$.
  - L_e is the emitted radiance
  - No explicit visibility term!
- Energy balance!
  - $\Phi_r - \Phi_i = \Phi_e - \Phi_a$
- Path tracing for computing LTE
  - Monte Carlo integration



4

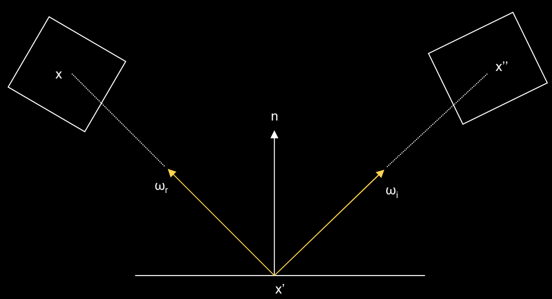
Surface Form of LTE



- Integral over area instead of directions
  - $L(x' \to x) = L(x', \omega_r)$
  - $f_r(x'' \to x' \to x) = f_r(x', \omega_r, \omega_i)$

5

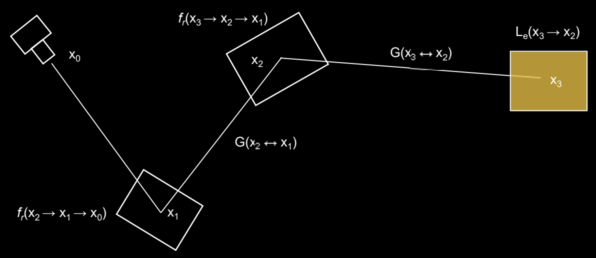
Surface Form of LTE



- Geometric coupling term
  - $G(x' \leftrightarrow x) = V(x' \leftrightarrow x)\, |\cos\theta|\, |\cos\theta'| \,/\, \| x - x' \|^2$
- LTE as area integral
  - $L(x' \to x) = L_e(x' \to x) + \int_A f_r(x'' \to x' \to x)\, L(x'' \to x')\, G(x'' \leftrightarrow x')\, dA(x'')$.

6

Integrals over paths



- $L(x_1 \to x_0) = L_e(x_1 \to x_0) +$

$$\int_A L_e(x_2 \to x_1)\, f_r(x_2 \to x_1 \to x_0)\, G(x_2 \leftrightarrow x_1)\, dA(x_2) +$$

$$\int_A \int_A L_e(x_3 \to x_2)\, f_r(x_3 \to x_2 \to x_1)\, G(x_3 \leftrightarrow x_2)$$

$$f_r(x_2 \to x_1 \to x_0)\, G(x_2 \leftrightarrow x_1)\, dA(x_3)\, dA(x_2) + \ldots$$

7

Path tracing

- $L(x_1 \to x_0) = \sum_{i=0}^{\infty} P(x_i)$

$$= P(x_1) + P(x_2) + \sum_{i=3}^{\infty} P(x_i)$$

  – need to terminate path after finite terms!
  – need to sample paths for MC estimate of integral

8

Path termination – Russian roulette

- Integrand not evaluated with probability q
  - approx. with constant c
- Evaluate integrand with probability $1 - q$, weighted by $1/(1 - q)$:

$$F' = \begin{cases} (F - qc)/(1 - q) & u > q \\ c & \textit{otherwise} \end{cases}$$

- $E(F') = (1 - q)\,((E(F) - qc)/(1 - q)) + qc$

  $= E(F)$

- Same expected value but you pay the price of higher variance!

9

Path termination

- Russian roulette allows us to terminate a path after finite terms
- $L(x_1 \rightarrow x_0) = P(x_1) + P(x_2) + 1/(1 - q) \sum_{i=3}^{\infty} P(x_i)$
- Alternatively:

  $L(x_1 \rightarrow x_0) = 1/(1 - q_1)\,(P(x_1) + 1/(1 - q_2)\,(P(x_2) + 1/(1 - q_3)\,(P(x_3) + \ldots$

- Path usually terminated from 3rd bounce onwards due to lower contribution
  - higher variance of low contribution is tolerable.

10

Path sampling

- Sample light sources by area, and vertices according to BRDF
- Need to convert solid angle probability p_ω to area p_A

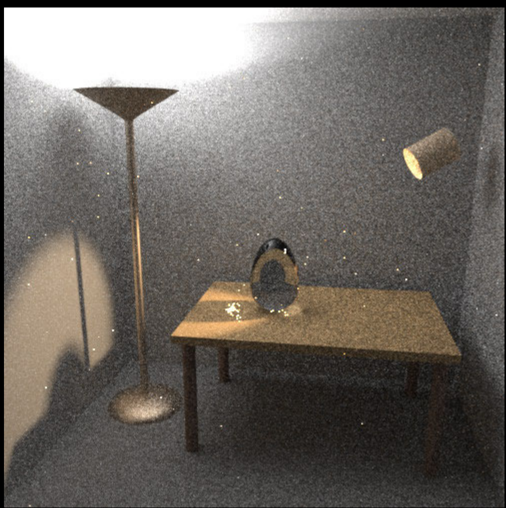
$$p_A = p_\omega * (|\cos\theta_i| / ||x_i - x_{i-1}||^2)$$

- MC estimate for path is then given as:

$$\frac{L_e(x_i \rightarrow x_{i-1})}{p_A(x_i)} \prod_{j=1}^{i-1} \frac{f_r(x_{j+1} \rightarrow x_j \rightarrow x_{j-1})\,|\cos\theta_j|}{p_\omega(x_{j+1} - x_j)}$$

11

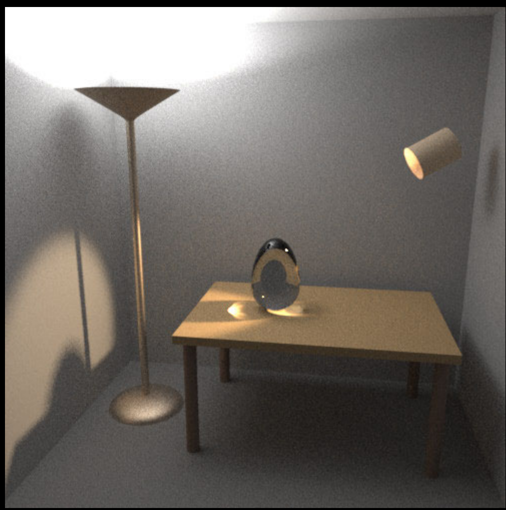
Path tracing from camera



56 samples per pixel

12

Bidirectional path tracing



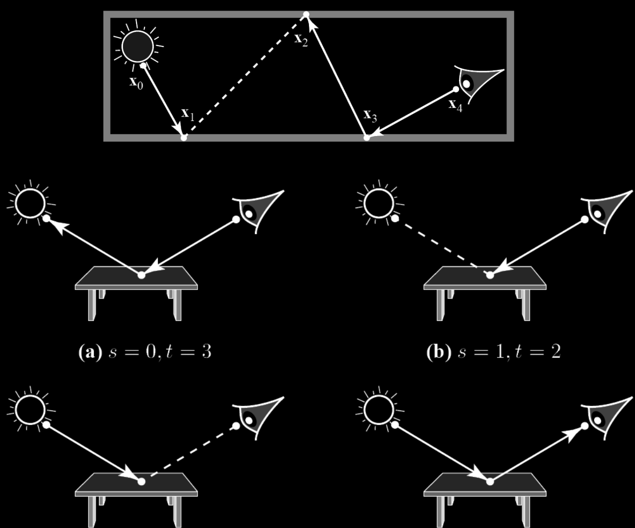
25 samples per pixel

Veach&Guibas 94

also see
Lafortune&Willems93

13

Bidirectional path tracing



(a) $s = 0, t = 3$

(b) $s = 1, t = 2$

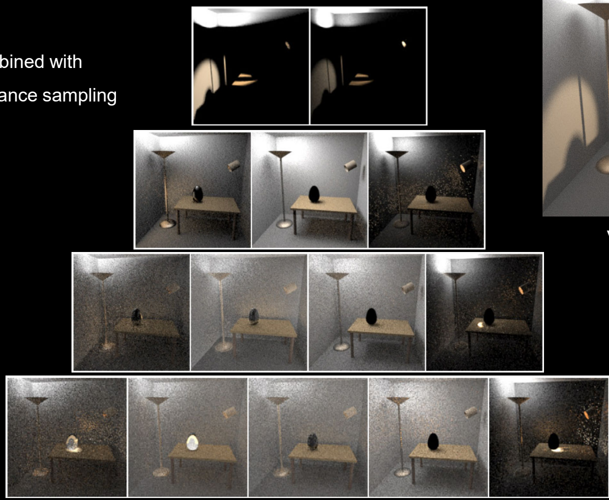
(c) $s = 2, t = 1$

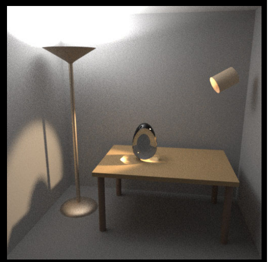
(d) $s = 3, t = 0$

14

Bidirectional path tracing

Paths combined with
multiple importance sampling





Veach&Guibas 94

15

Bidirectional path tracing

40 samples per pixel

16

Metropolis Light Transport – Veach&Guibas 97

250 mutations per pixel

17

MLT

- Mutate an existing path to obtain a new path!
  - accept mutation based on energy balance
- Advantage over path tracing
  - path re-use
  - local exploration

18

Metropolis-Hastings algorithm

```
x = x0
for i = 1 to n
    x' = mutate(x)
    a = accept(x, x')
    record(x, (1-a) * weight)
    record(x', a * weight)
    if (random() < a)
        x = x'
```

19

Detailed balance

- Mutations proposed with transition probability $T(x \rightarrow x')$
- Mutations accepted with acceptance probability $a(x \rightarrow x')$
- Random walk in equilibrium requires:

  $f(x)\ T(x \rightarrow x')\ a(x \rightarrow x') = f(x')\ T(x' \rightarrow x)\ a(x' \rightarrow x)$
- $a(x \rightarrow x') = \min\left(1, \dfrac{f(x')\ T(x' \rightarrow x)}{f(x)\ T(x \rightarrow x')}\right)$

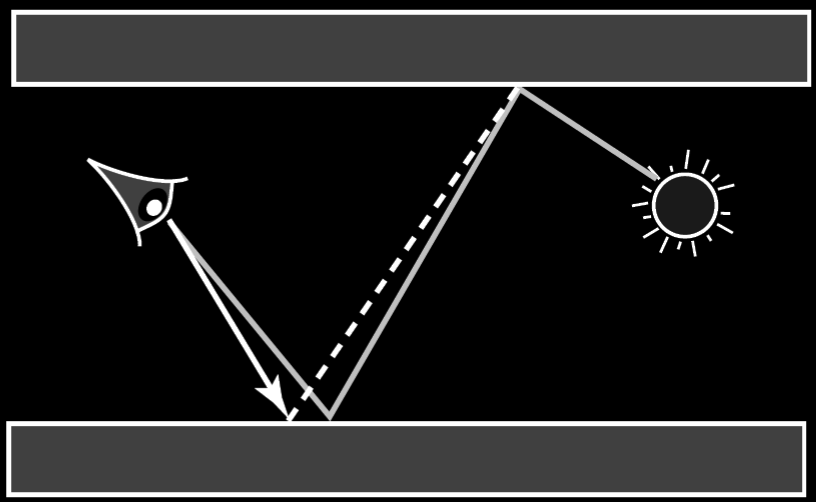
20

Mutation strategies

- Local perturbations
  - Good for explorations of high importance paths
- Proposals from a distribution
  - Good for maintaining ergodicity
  - purturbations can get stuck in a local minima!

21

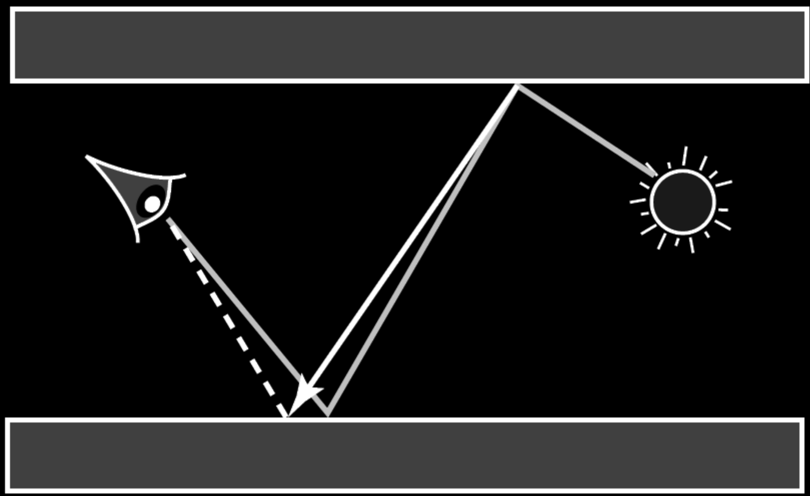
Lens perturbation



- Ideal for sub-paths of the form (L|D)DS*E (context free grammar)

22

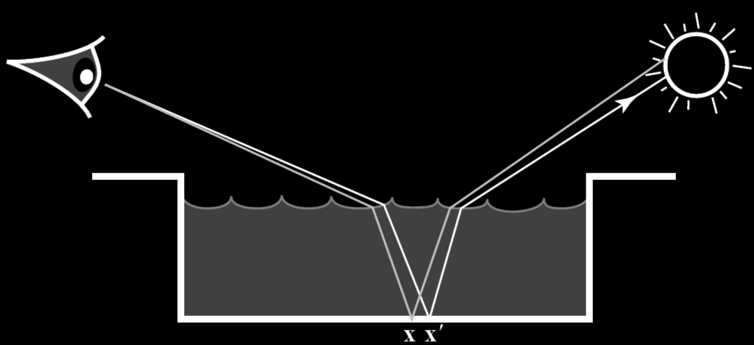
Caustic perturbation



- Ideal for sub-paths of the form (D|L)S*DE (context free grammar)

23

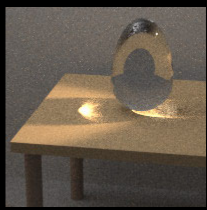
Multi-chain perturbation



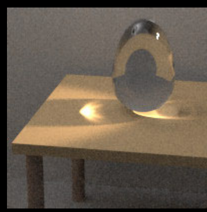
- Lens perturbation followed by caustic perturbation
  - Sampling caustics in a pool of water!

24

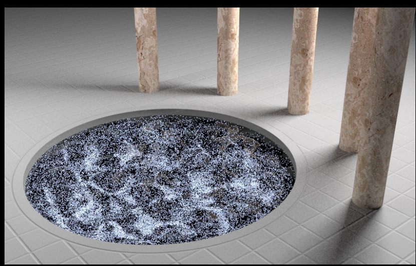
MLT vs path tracing



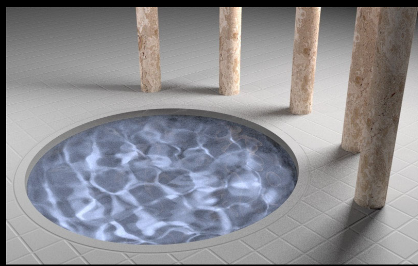
Bidirectional PT



MLT



Path tracing



MLT

25

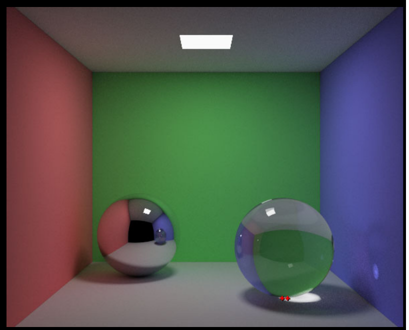
Start-up bias

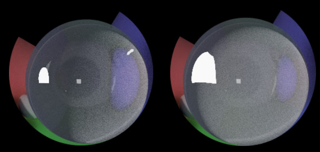
- MLT assumes sample *x* to already be drawn from the target distribution *p*
- Need to draw many samples before achieving the target distribution
  - initial samples have to discarded
  - wastage of samples ☹
- Draw from a proposal distribution *q* and weight the samples with *f*/*q*
  - sub-optimal

26

Energy Re-distribution Path Tracing – Cline et al. 05

- Combining MC path tracing and MLT mutations!
- Redistribution of energy from MC samples over image plane
  - lens perturbation!
- MLT has better convergence than MC PT for high dimensional problems
  - but worse for low dimensional problems!





27

ERPT

EnergyRedistributionSampling ()
 for each integral domain, Ω_i
  for $j = 1$ to m
   create an MC sample, $\bar{x}$, in Ω_i according to $\mathbf{S}_p$
   evaluate $\mathbf{X}_f(\bar{x}) = f(\bar{x})/p(\bar{x})$
   if $\mathbf{X}_f(\bar{x}) > 0$
    redistribute the energy of $\mathbf{X}_f(\bar{x})$ using
     a balanced energy flow filter.

28

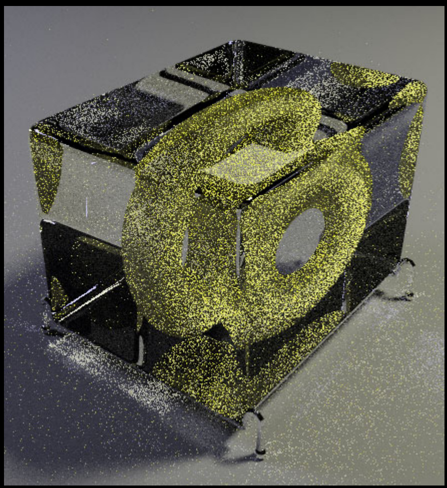
Equal deposition flow

EqualDepositionFlow $(\bar{x},\ e,\ m,\ e_d)$
    $numChains = \lfloor \mathrm{random}(0,1) + e/(m \times e_d) \rfloor$
    for $i = 1$ to $numChains$
        $\bar{y} = \bar{x}$
        for $j = 1$ to m
            $\bar{z} = \mathrm{mutate}(\bar{y})$
            if $q(\bar{y} \to \bar{z}) \ \geq\ \mathrm{random}(0,1)$
                $\bar{y} = \bar{z}$
            deposit e_d energy at $\bar{y}$

- Here mutation implies selecting another pixel in a local neighborhood and depositing some of the sample's energy to its neighbor
  - Image pixels in a neighborhood will all converge to similar energy

29

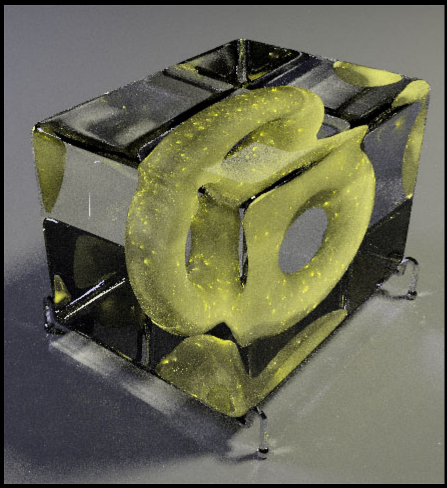
Path tracing



100 samples per pixel

30

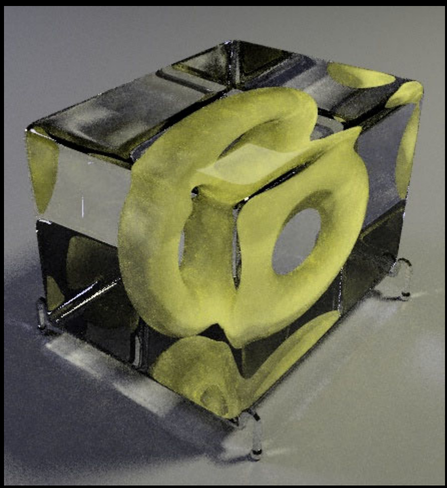
MLT



100 mutations per pixel

31

ERPT

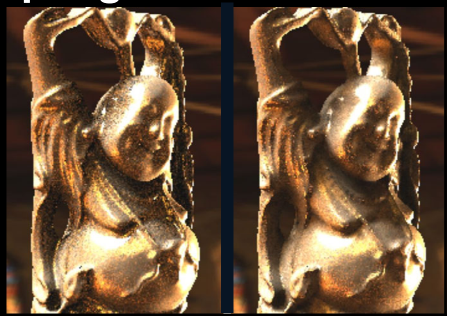


36 MC samples +
50 mutations per pixel

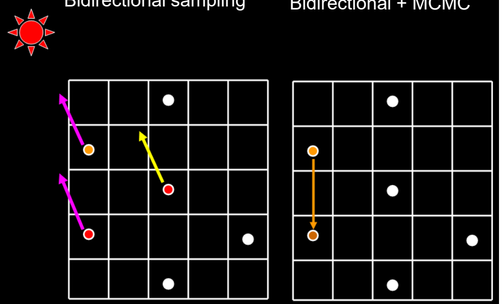
32

Correlated visibility sampling – [Ghosh & Heidrich 06]

- Similar in spirit to ERPT
  – but applied to visibility sampling for direct illumination
- Bidirectional sampling for un-occluded regions – 1st pass
  – Mark partially occluded pixels
- Metropolis energy exchange on the image plane for partially occluded regions – 2nd pass



Bidirectional sampling　　Bidirectional + MCMC



33

Correlated visibility sampling

Bidirectional sampling
20 SIR samples

Visibility sampling
16 SIR + 16 MCMC samples

34

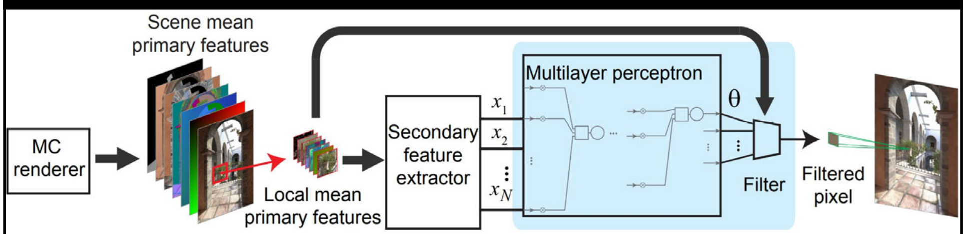
Filtering MC Noise in Rendering [Kalantari et al. 2015]

Our result with a cross-bilateral filter (4 spp) Our result with a non-local means filter (4 spp)

- Neural network is trained to predict the appropriate parameters of a filter kernel which is then applied to de-noise a noisy MC rendering output.
- Training uses additional features of the image besides pixel colors such as surface normal, surface positions, texture as features for predicting the kernel

35

Filtering MC Noise in Rendering [Kalantari et al. 2015]

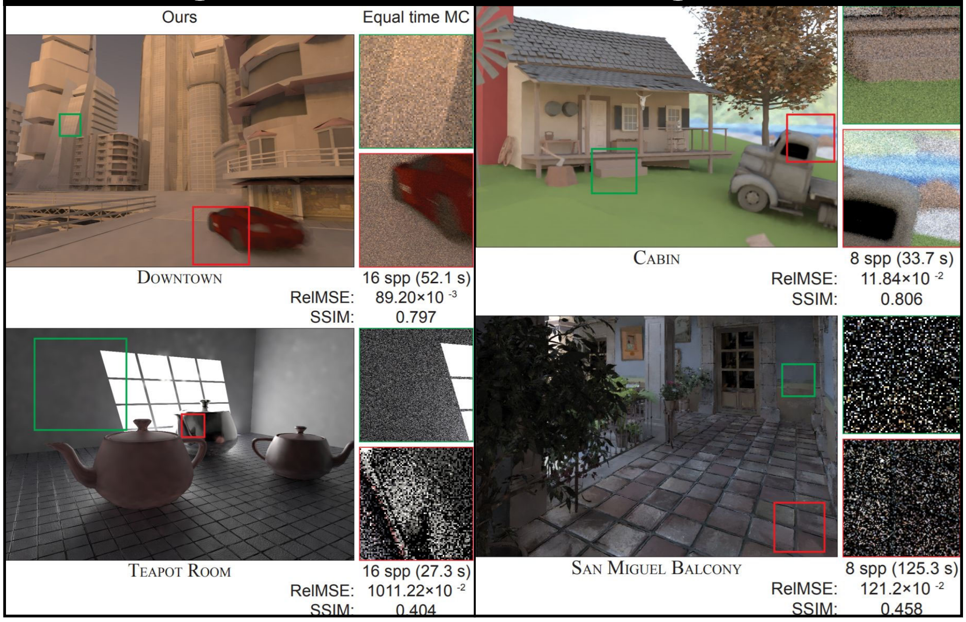


- Neural network is trained to predict the appropriate parameters of a filter kernel which is then applied to de-noise a noisy MC rendering output.
- Training uses additional features of the image besides pixel colors such as surface normal, surface positions, texture as features for predicting the kernel

36

Filtering MC Noise in Rendering

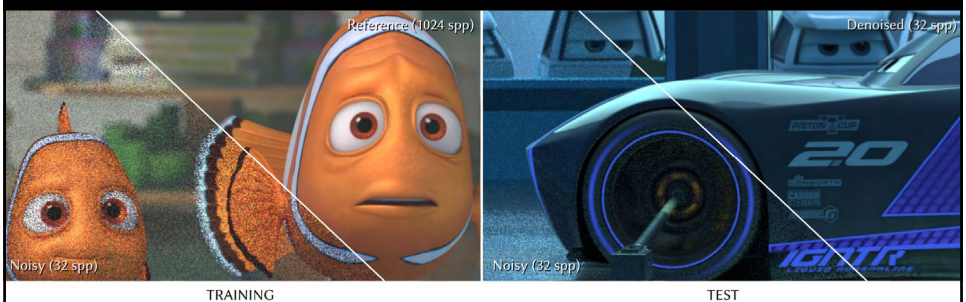
[Kalantari et al. 2015]



37

Filtering MC Noise with CNNs

[Bako et al. 2017]

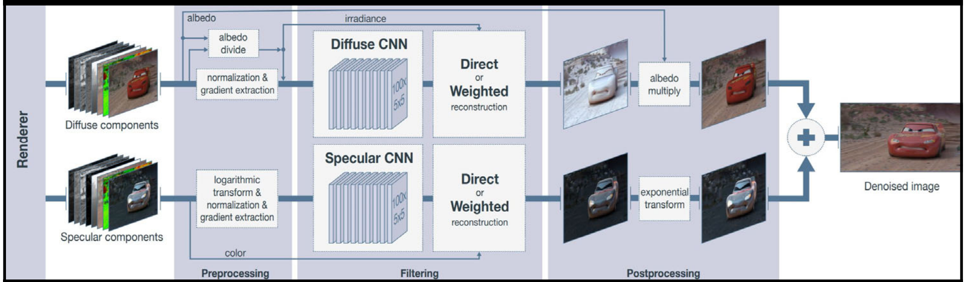


- CNN trained on noisy and converged frame pairs from Finding Dory
- Trained CNN can be applied to other films with different style of lighting and appearance such as Cars 3.
- Denoising stable enough to be used in production rendering pipelines to create final quality animation frames.

38

Filtering MC Noise with CNNs

[Bako et al. 2017]

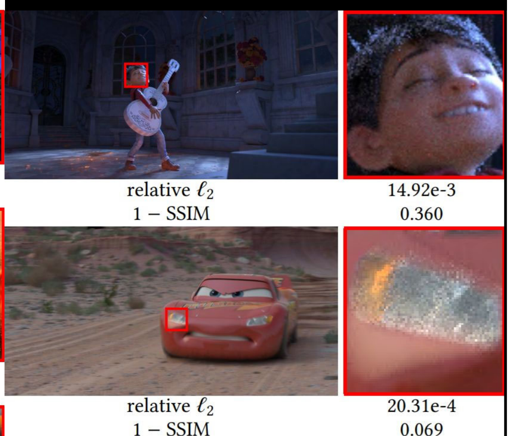


- Separate CNN networks trained for denoising diffuse and specular renderings
- Diffuse shading is denoised after factoring out the albedo, which is post-multiplied.

39

Filtering MC Noise with CNNs

[Bako et al. 2017]

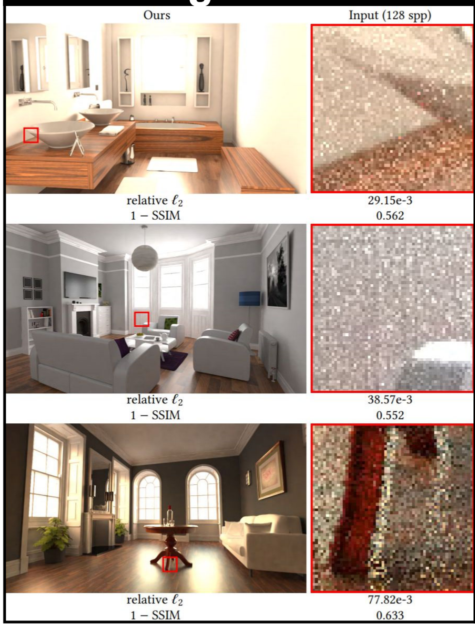


The CNN trained on Finding Dory generalizing to other types of scenes in animation film production

40

Filtering MC Noise with CNNs

[Bako et al. 2017]



CNN training is a bit content specific. Hence re-training is required for different type of renderd input, e.g., indoor scenes as opposed to Pixar animations. After re-training on the appropriate type of content, the method generalized well.

41