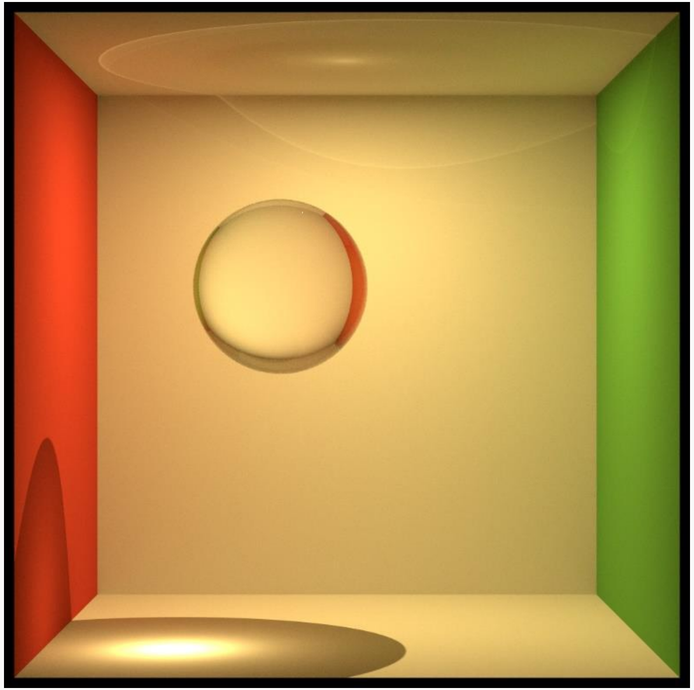
Gradient-Domain Volume Rendering

GMT Utrecht University

G.Lu

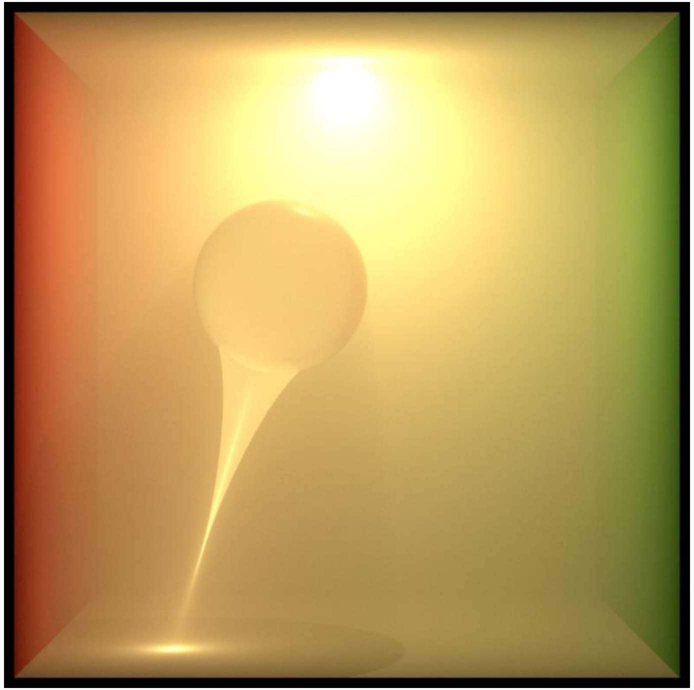
Supervisors
Jerry Guo, MSc
Prof. dr. Alexandru C. Telea
Prof. Dr. Elmar Eisemann
Dr. ing. Jacco Bikker

Motivation



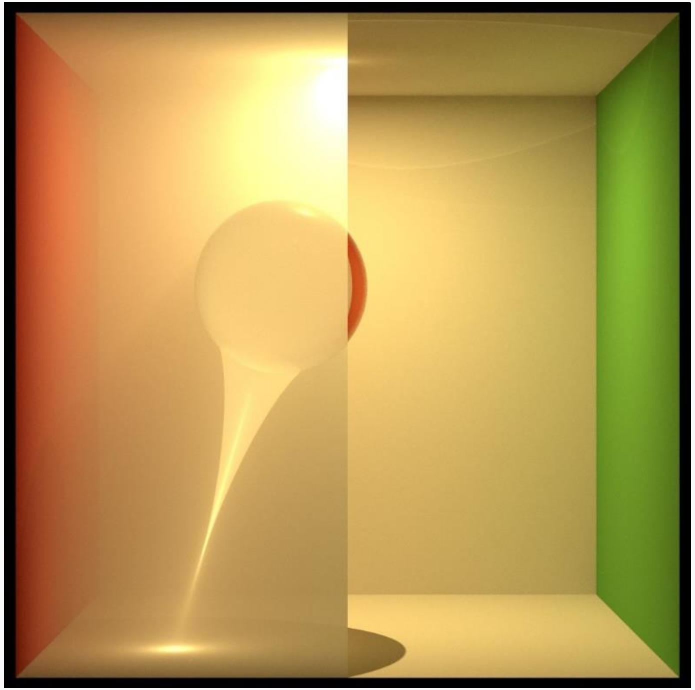
- A surface-only scene
  - Area light
  - Walls and ceiling
  - A glass ball

Motivation

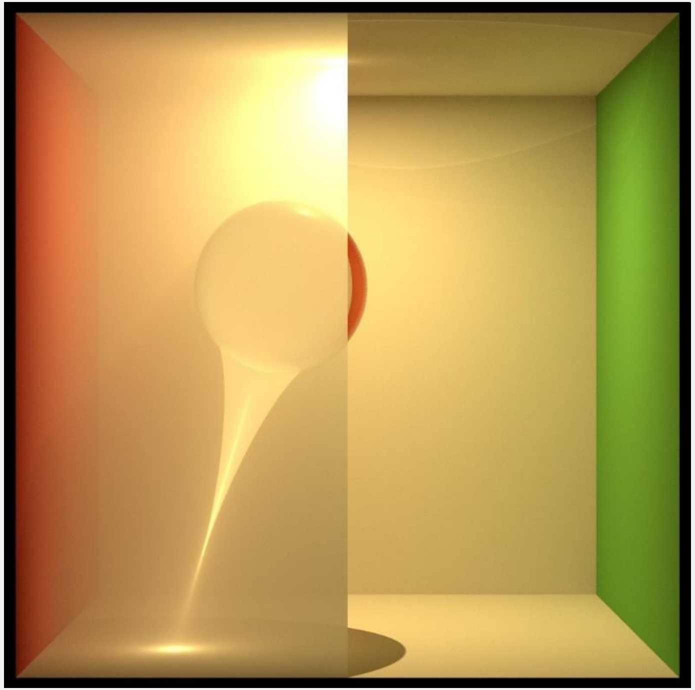


- A ~~surface-only~~ scene filled with gas
    - Area light
    - Walls and ceiling
    - A glass ball

Motivation

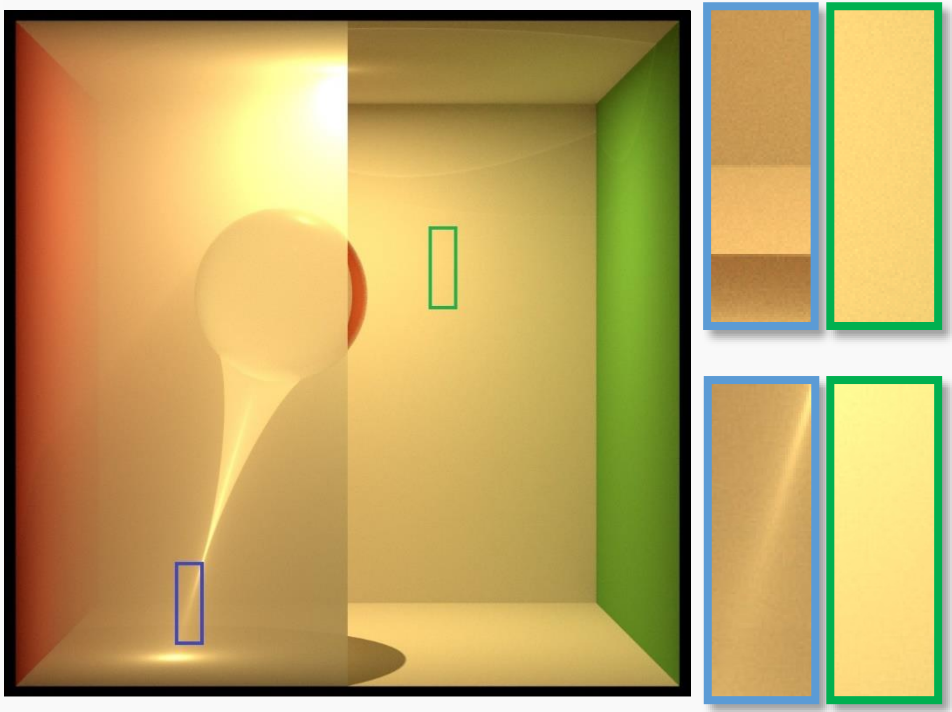


Motivation



- Sampling Challenge
  - Surface: O(N^2)
  - Volume: O(N^3)

Motivation



- Sampling Challenge
  - Surface: O(N^2)
  - Volume: O(N^3)
- Strong Correlation
  - Important light path
  - Estimation
  - Gradient

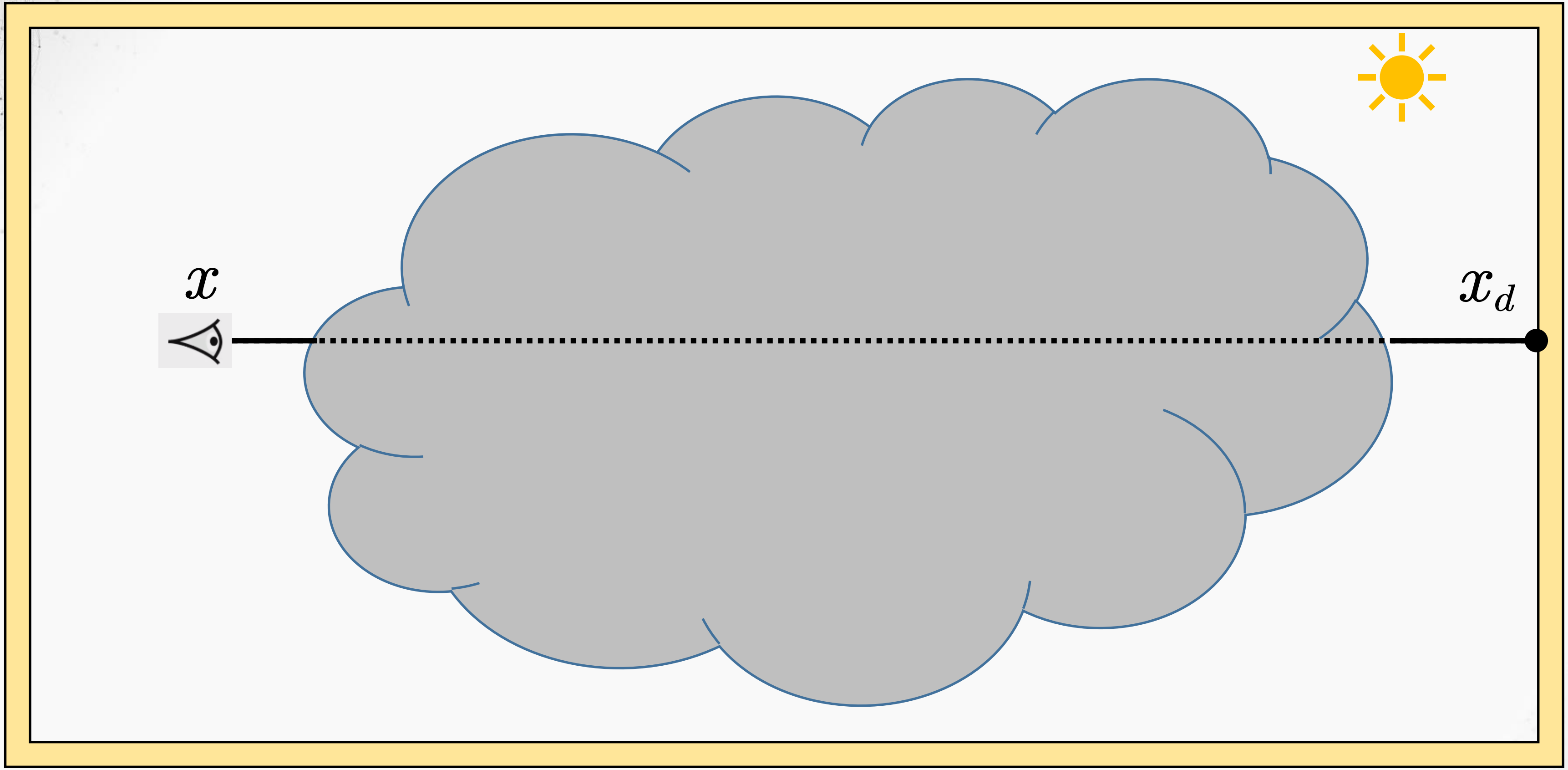
01 Motivation

02 Background

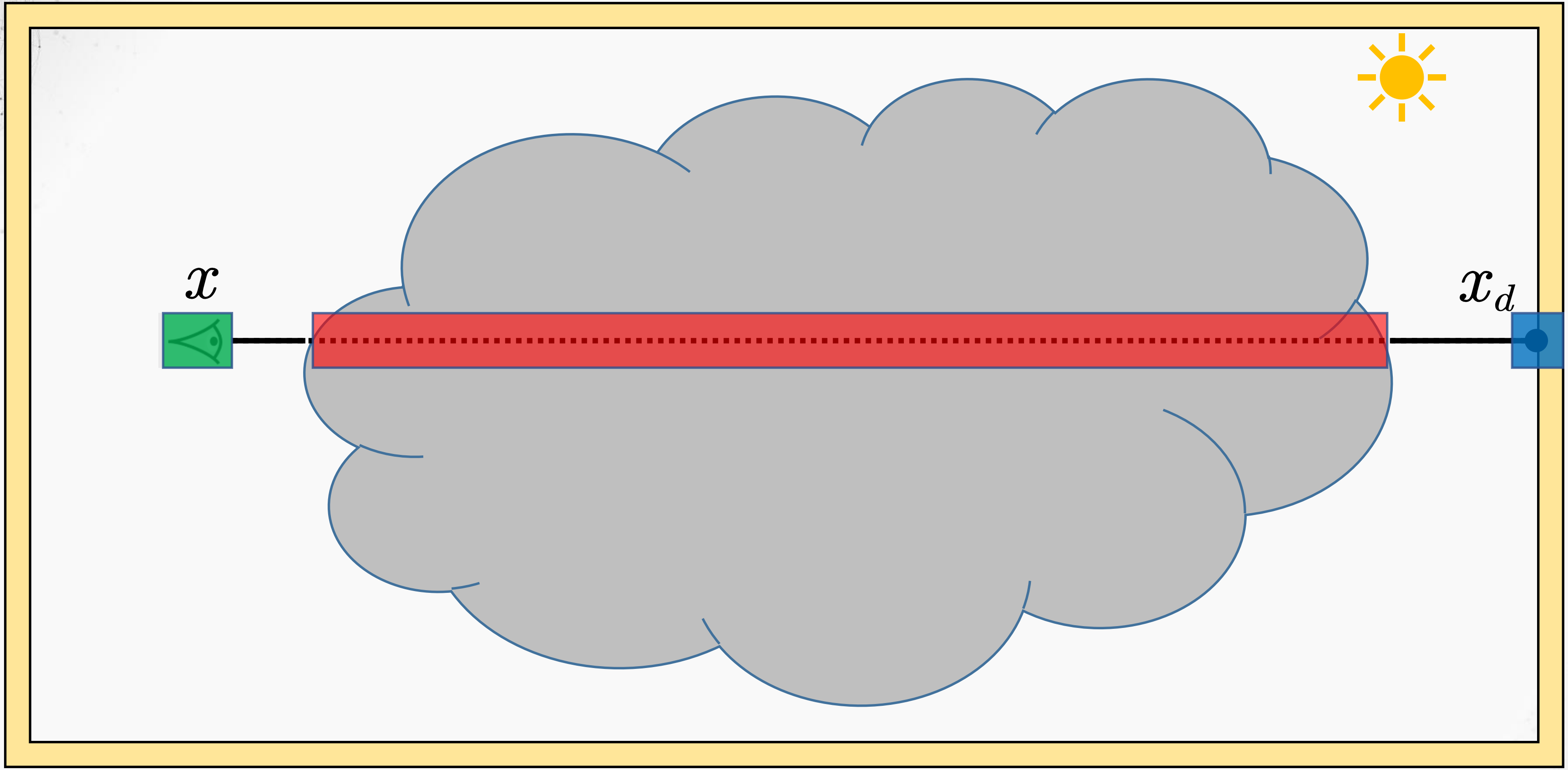
03 Gradient Algorithm

04 Result

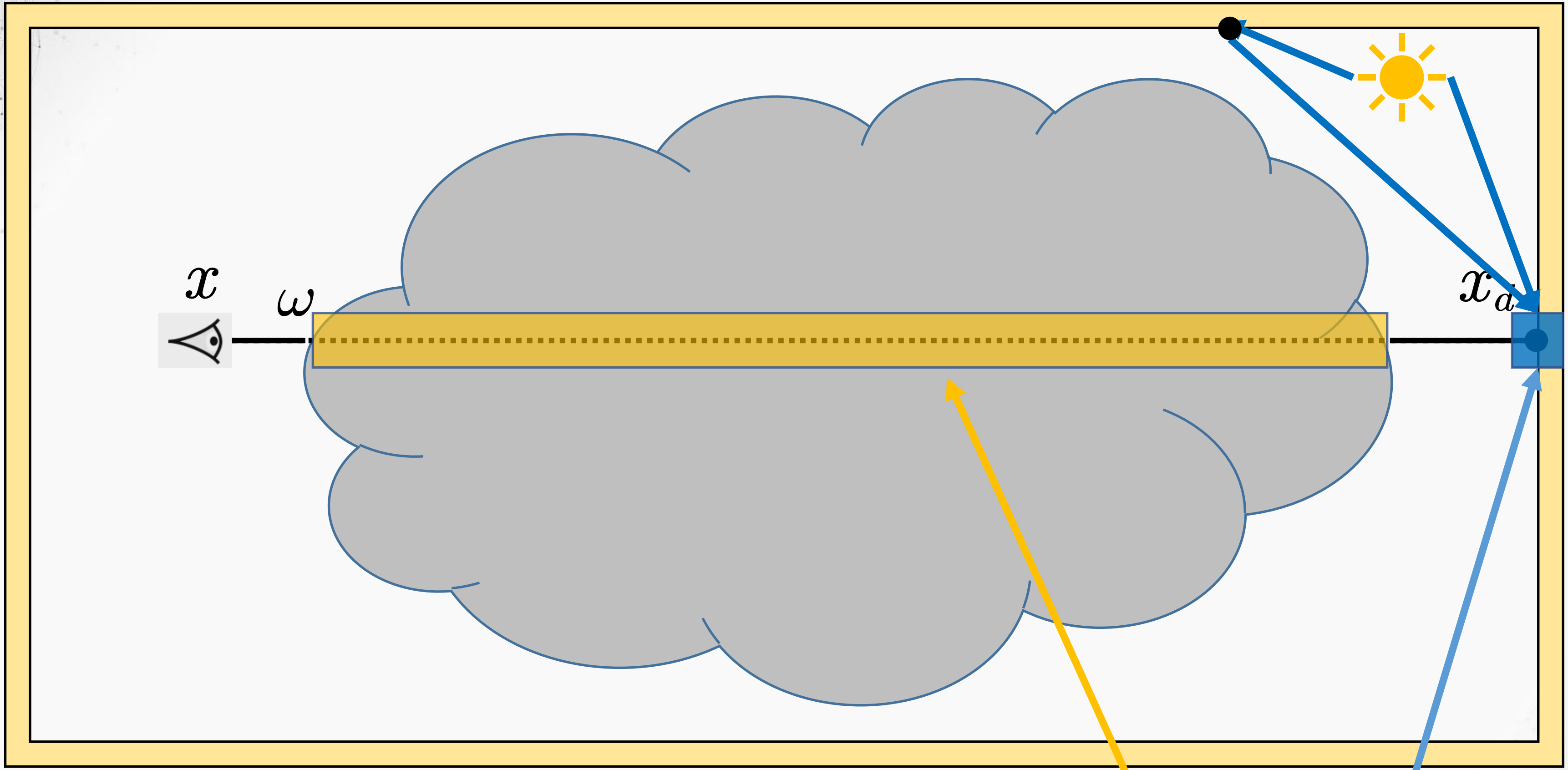
05 Conclusion



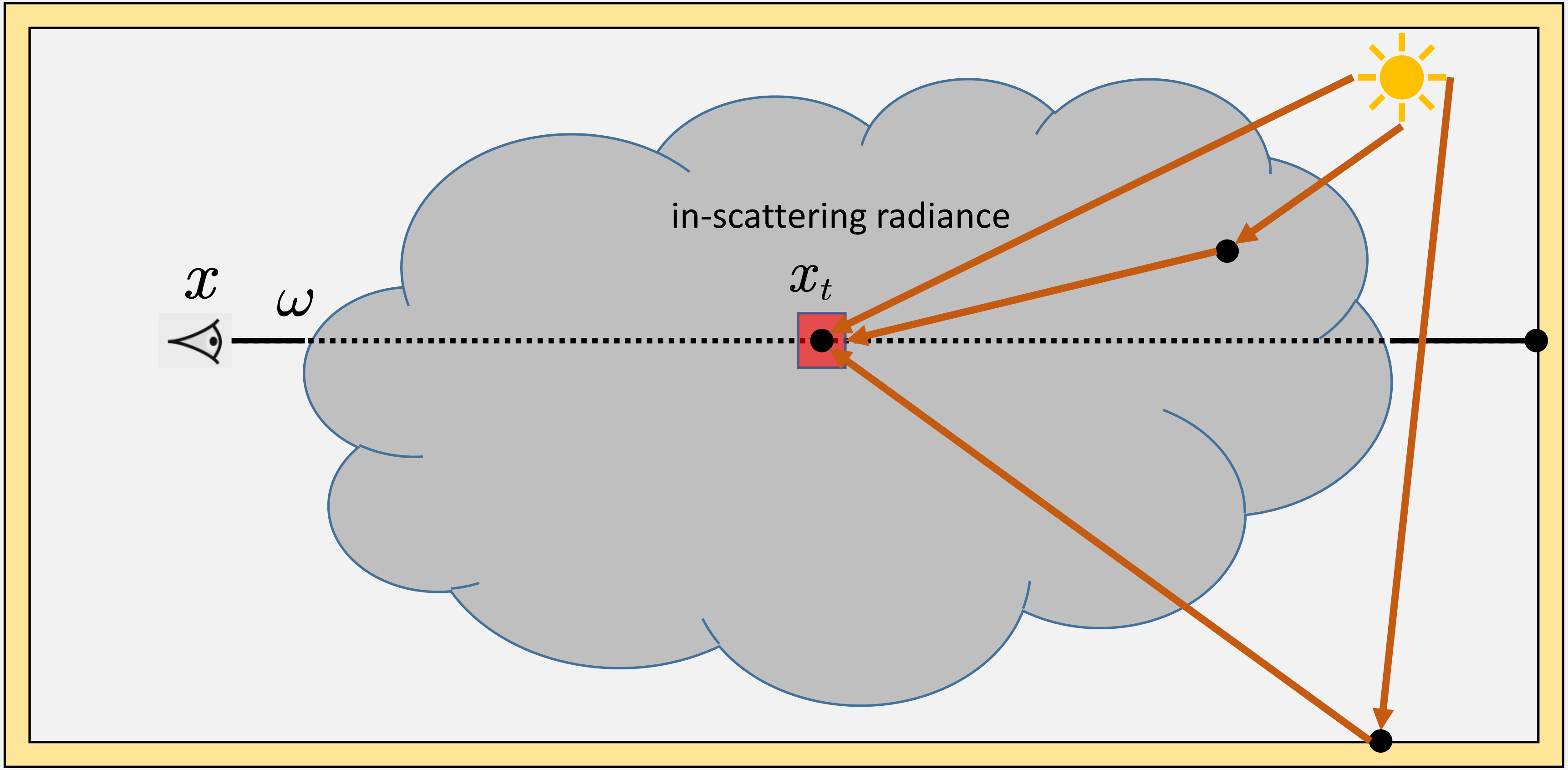
$$L(x,\omega) = \int_{x_d}^{x} T_r\,(x_t \leftrightarrow x)\,\sigma_s\,(x_t)\,L_s\,(x_t,\omega)\,dx_t + T_r\,(x_d \leftrightarrow x)\,L_d\,(x_d,\omega)$$



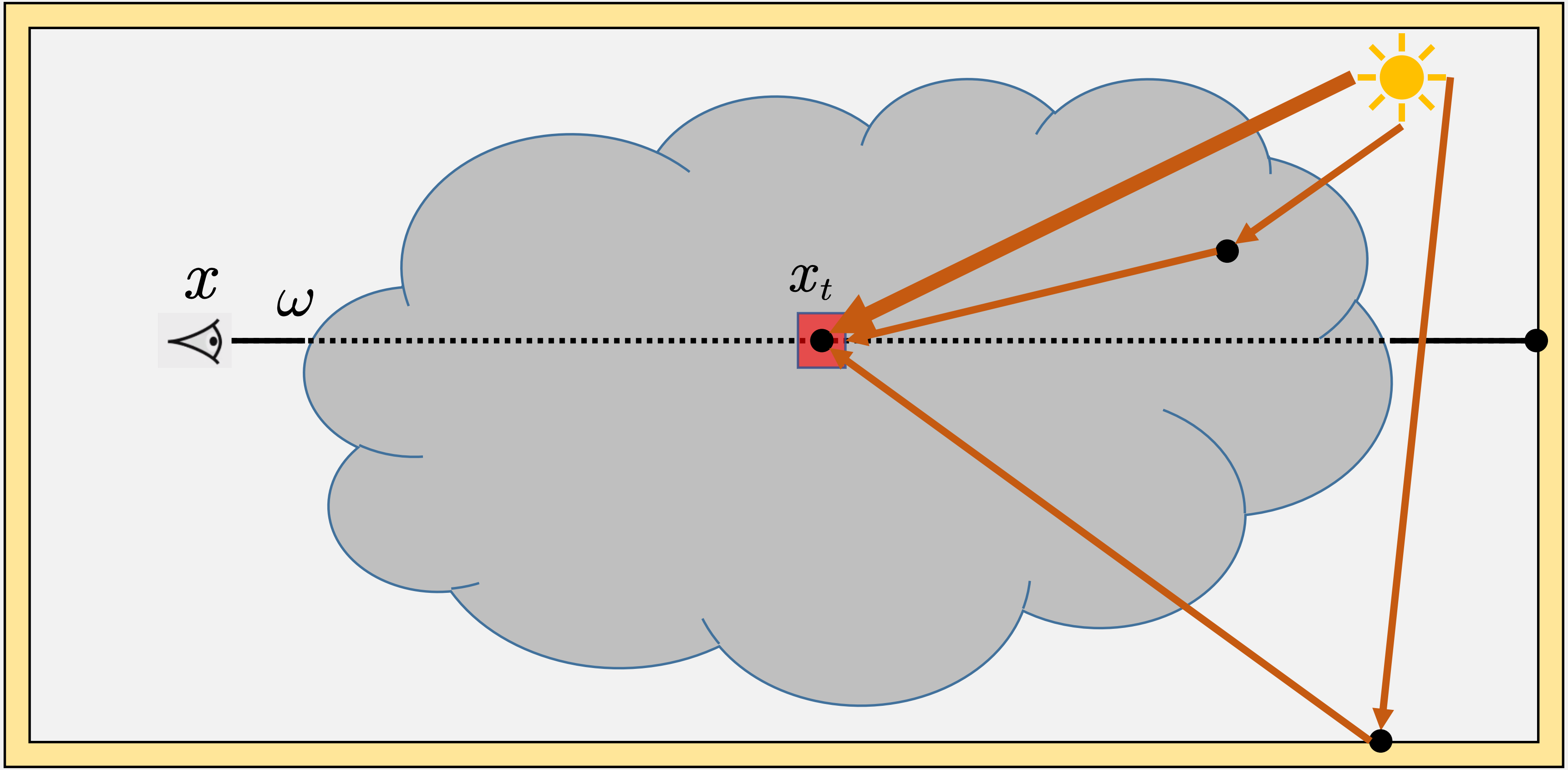
$$L(x,\omega) = \int_{x_d}^{x} T_r(x_t \leftrightarrow x)\sigma_s(x_t)L_s(x_t,\omega)\,dx_t + T_r(x_d \leftrightarrow x)L_d(x_d,\omega)$$



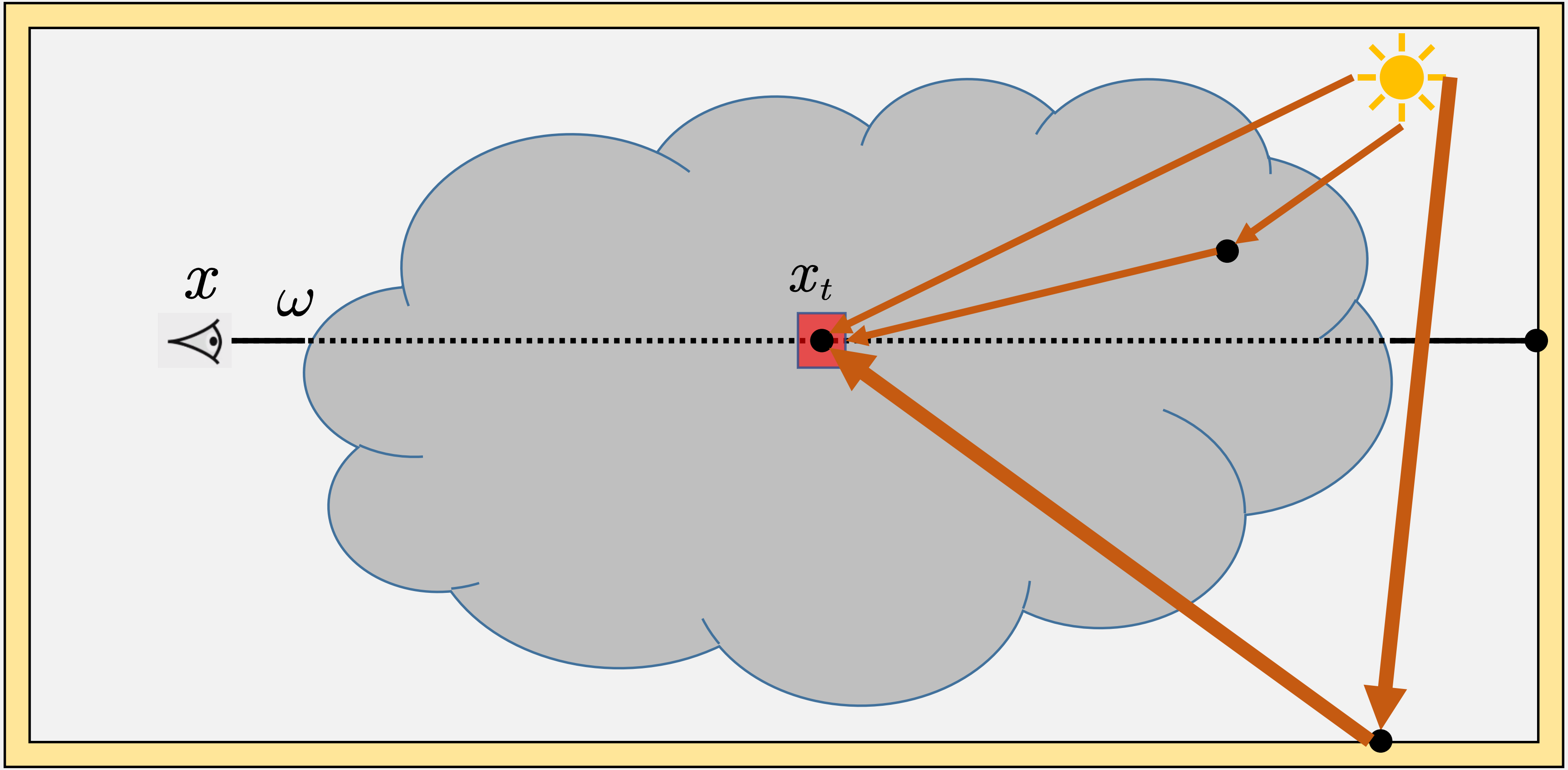
$$L(x,\omega) = \int_{x_d}^{x} T_r(x_t \leftrightarrow x)\,\sigma_s(x_t)\,L_s(x_t,\omega)\,dx_t + T_r(x_d \leftrightarrow x)\,L_d(x_d,\omega)$$



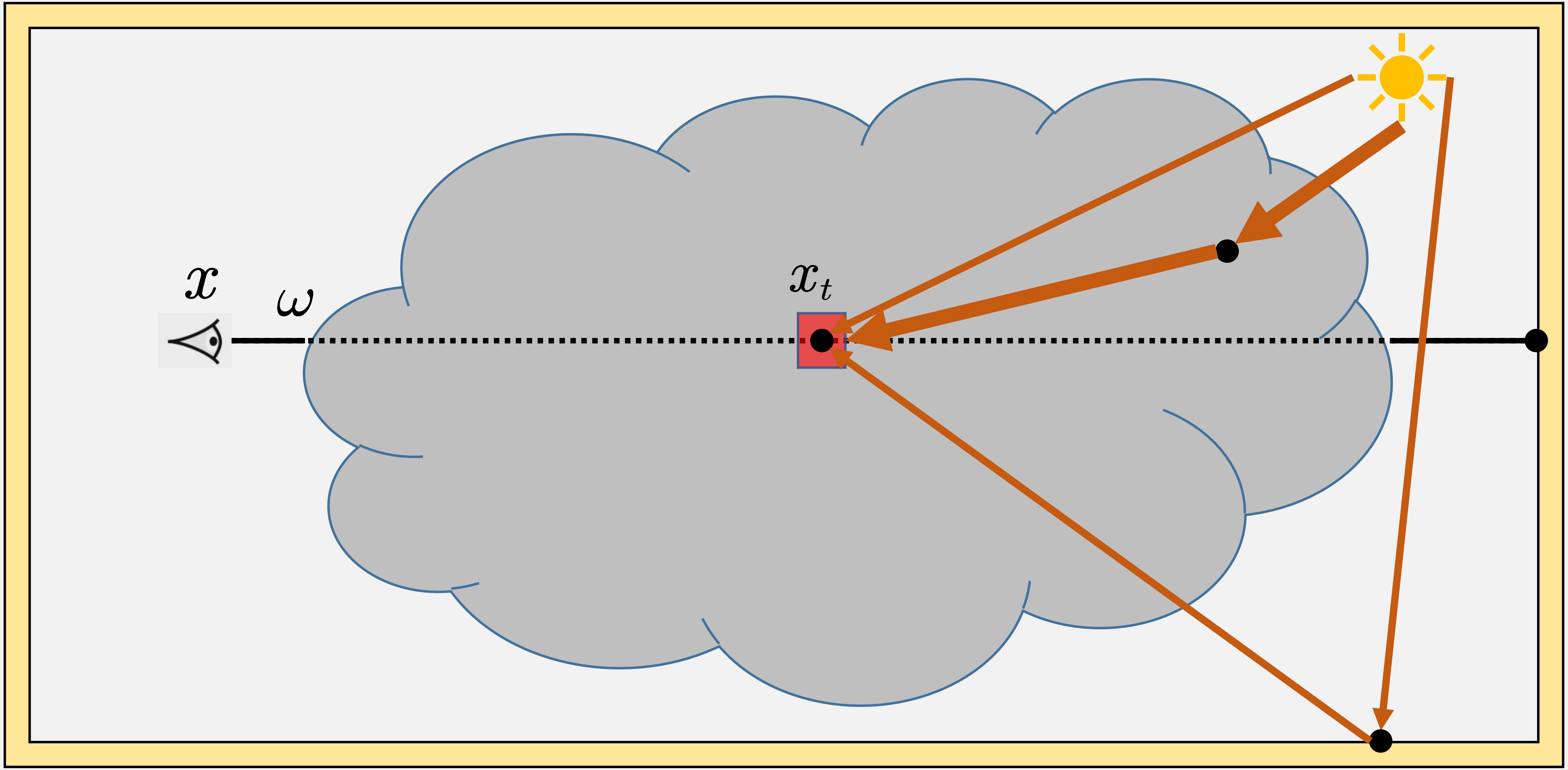
$$L(x,\omega) = \int_{x_d}^{x} T_r(x_t \leftrightarrow x)\sigma_s(x_t) L_s(x_t,\omega)\, dx_t + T_r(x_d \leftrightarrow x) L_d(x_d,\omega)$$



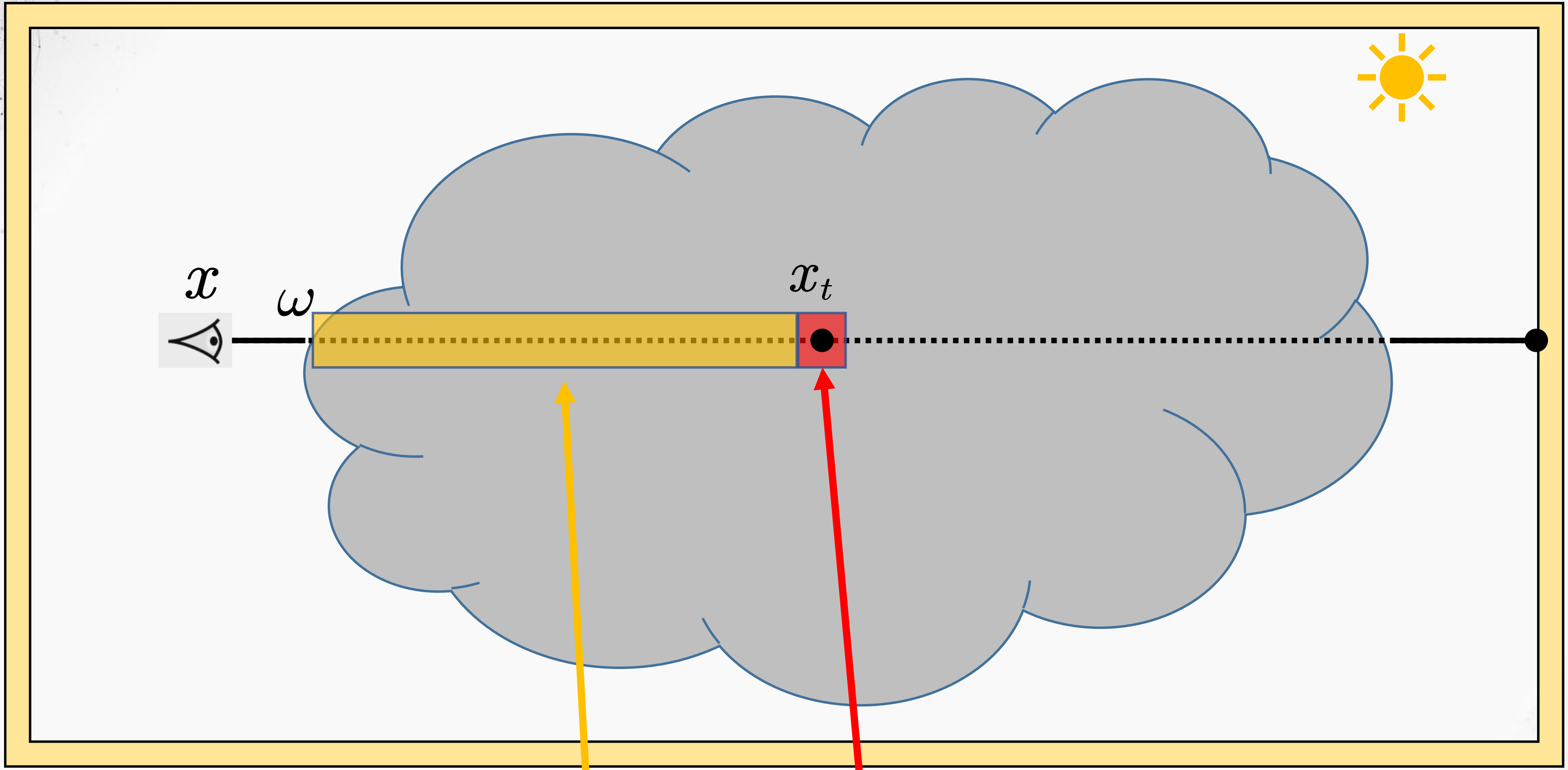
$$L(x,\omega) = \int_{x_d}^{x} T_r(x_t \leftrightarrow x)\sigma_s(x_t) L_s(x_t,\omega)\, dx_t + T_r(x_d \leftrightarrow x) L_d(x_d,\omega)$$



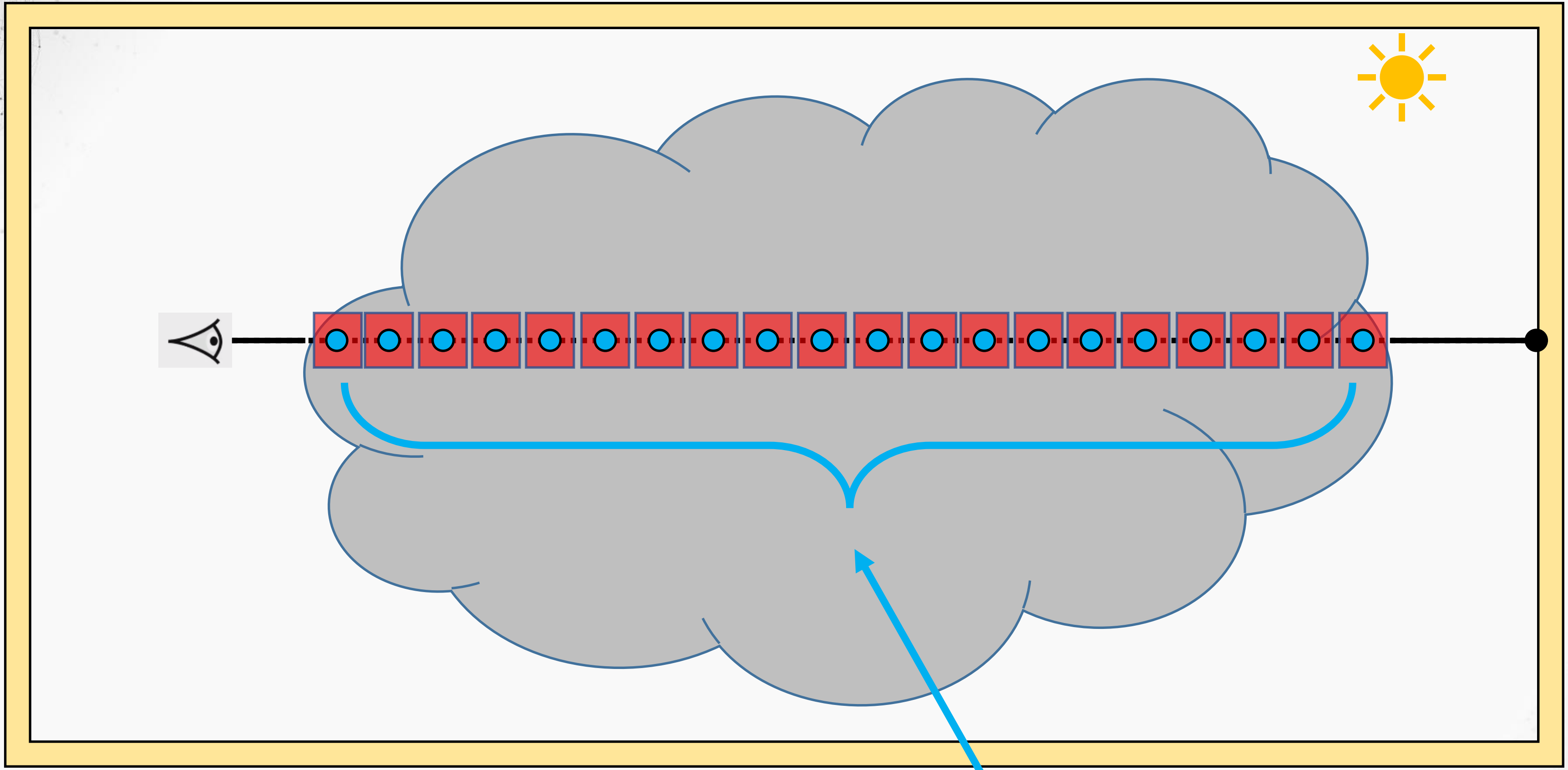
$$L(x,\omega) = \int_{x_d}^{x} T_r(x_t \leftrightarrow x)\,\sigma_s(x_t)\,\color{red}{L_s(x_t,\omega)}\,dx_t + T_r(x_d \leftrightarrow x)\,L_d(x_d,\omega)$$



$$L(x,\omega) = \int_{x_d}^{x} T_r(x_t \leftrightarrow x)\,\sigma_s(x_t)\,\color{red}{L_s(x_t,\omega)}\,dx_t + T_r(x_d \leftrightarrow x)\,L_d(x_d,\omega)$$



$$L(x,\omega) = \int_{x_d}^{x} T_r(x_t \leftrightarrow x)\,\sigma_s(x_t)\,L_s(x_t,\omega)\,dx_t + T_r(x_d \leftrightarrow x)\,L_d(x_d,\omega)$$



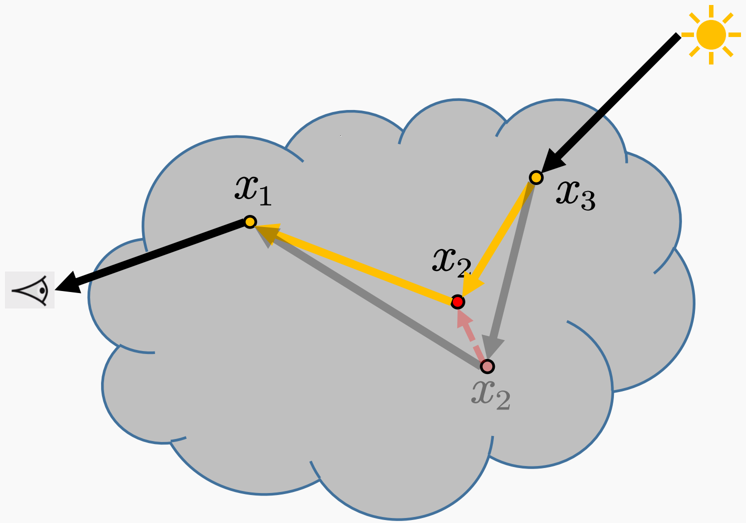
$$L(x,\omega) = \int_{x_d}^{x} T_r(x_t \leftrightarrow x)\,\sigma_s(x_t)\,L_s(x_t,\omega)\,dx_t + T_r(x_d \leftrightarrow x)\,L_d(x_d,\omega)$$

Previous work

- Rendering Equation
- Monte Carlo
  - Path Tracing
  - BDPT
- Markov Chain Monte Carlo
  - MLT
  - MLT for participating media

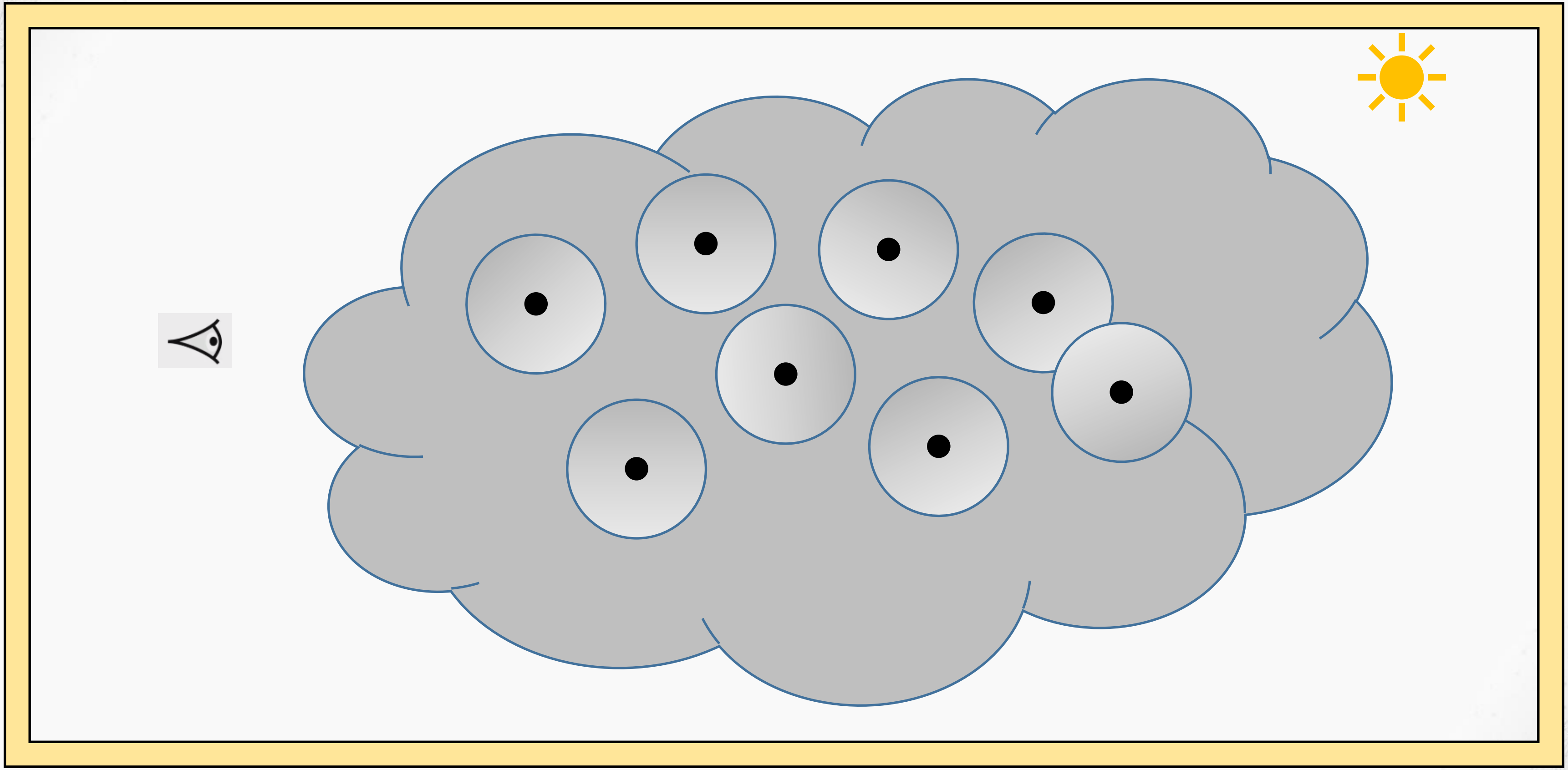
Goals

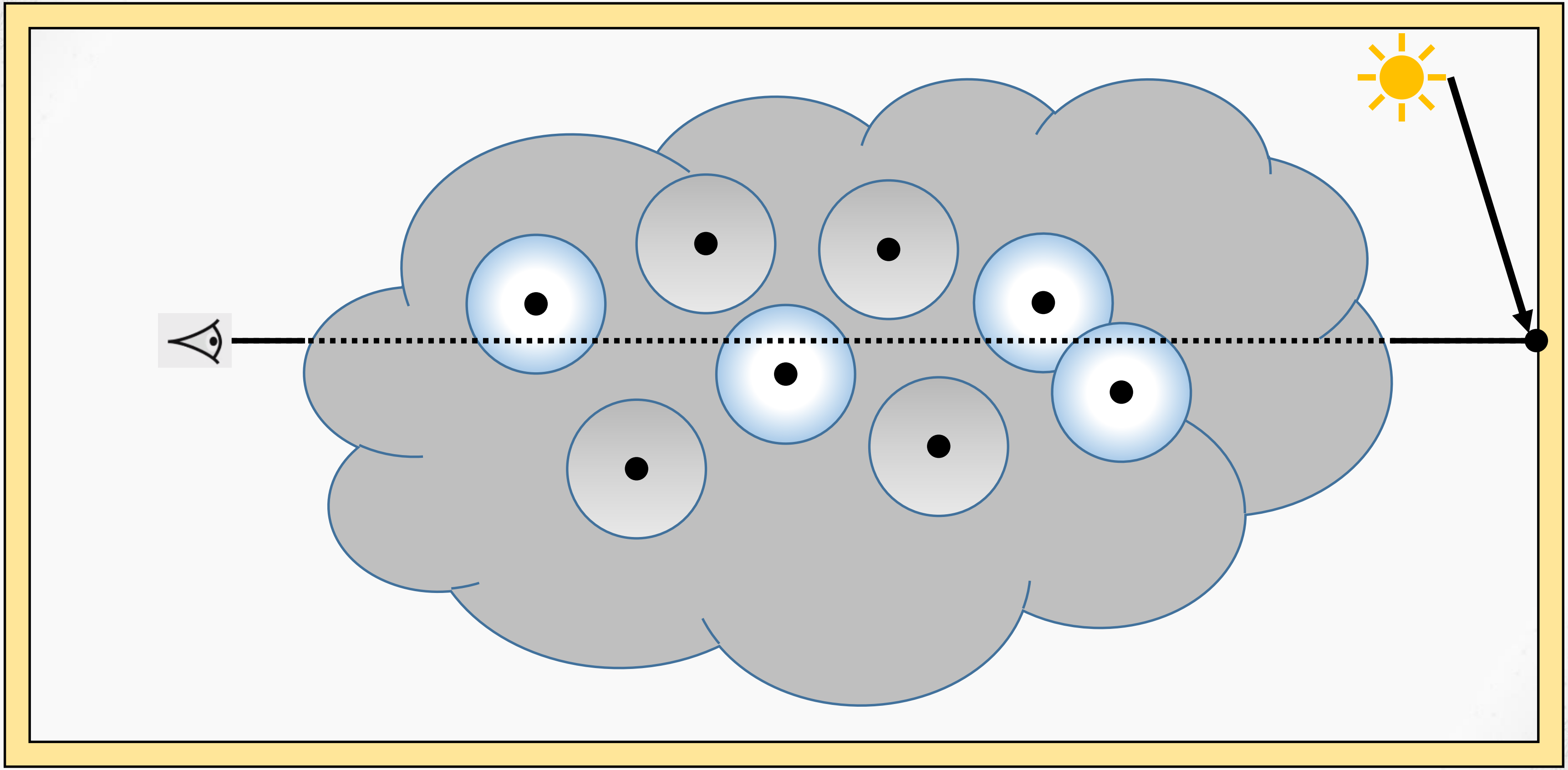
- Gradient Algorithm
- Gradient mutation

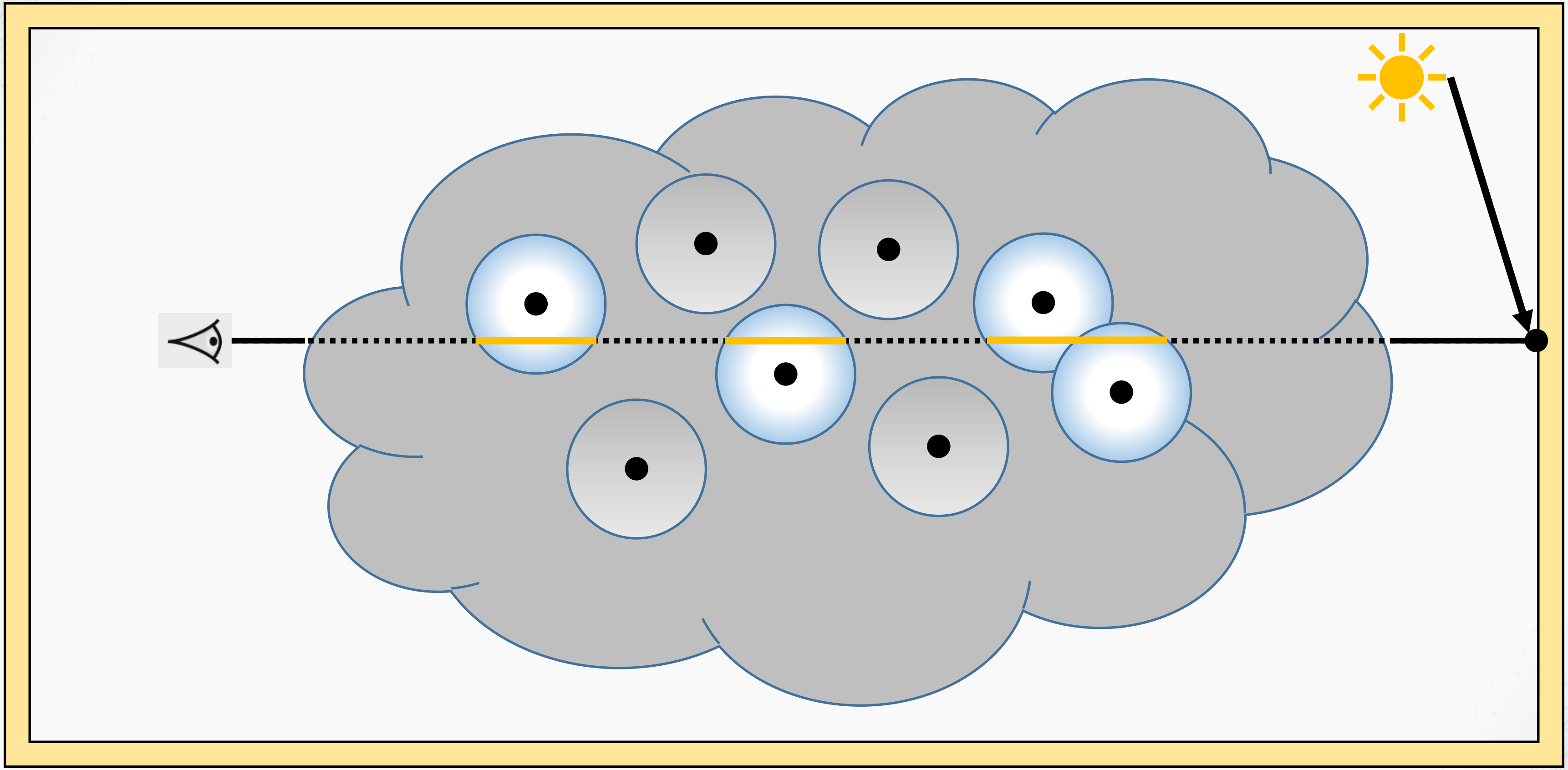


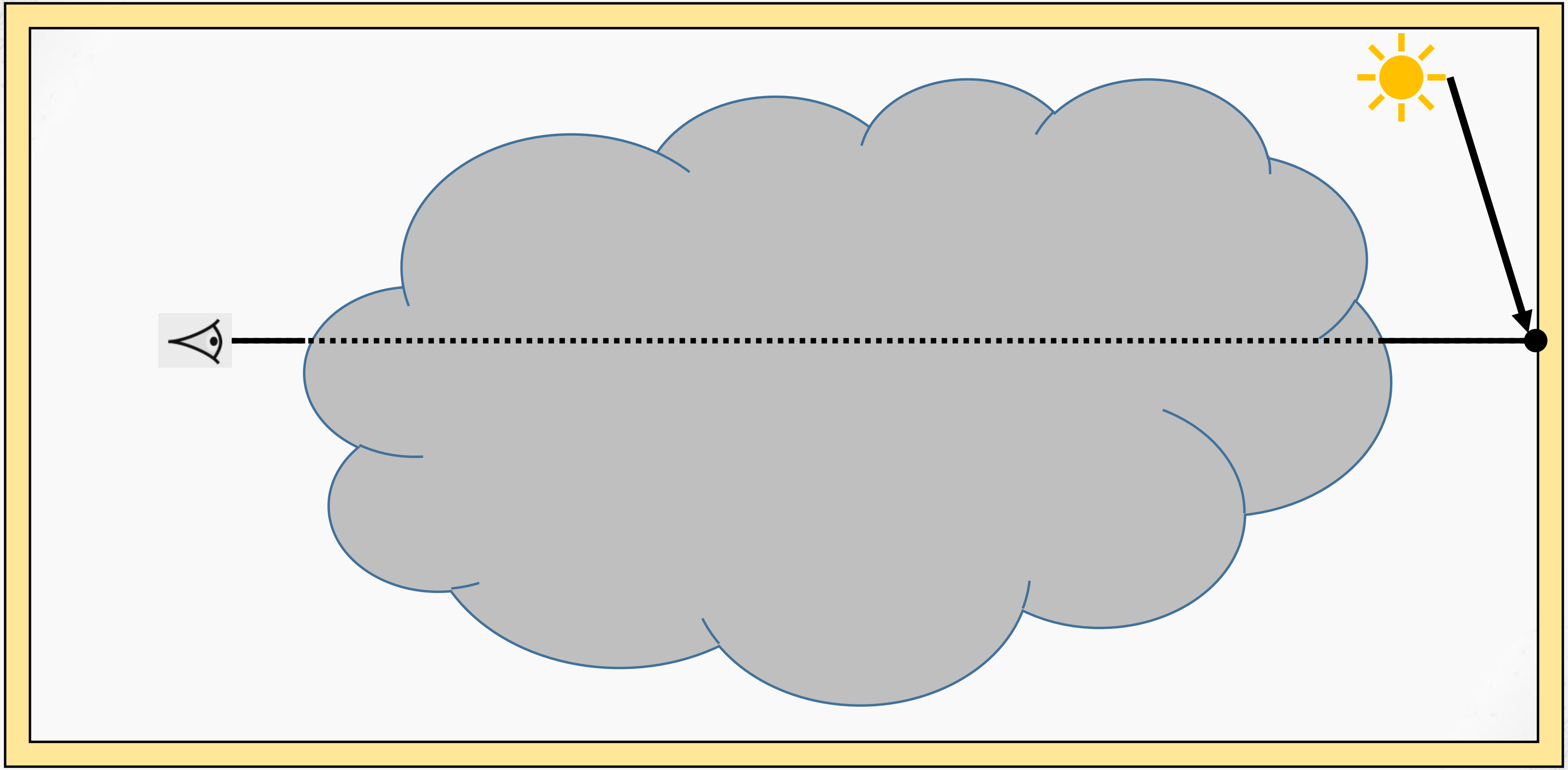
Related Work

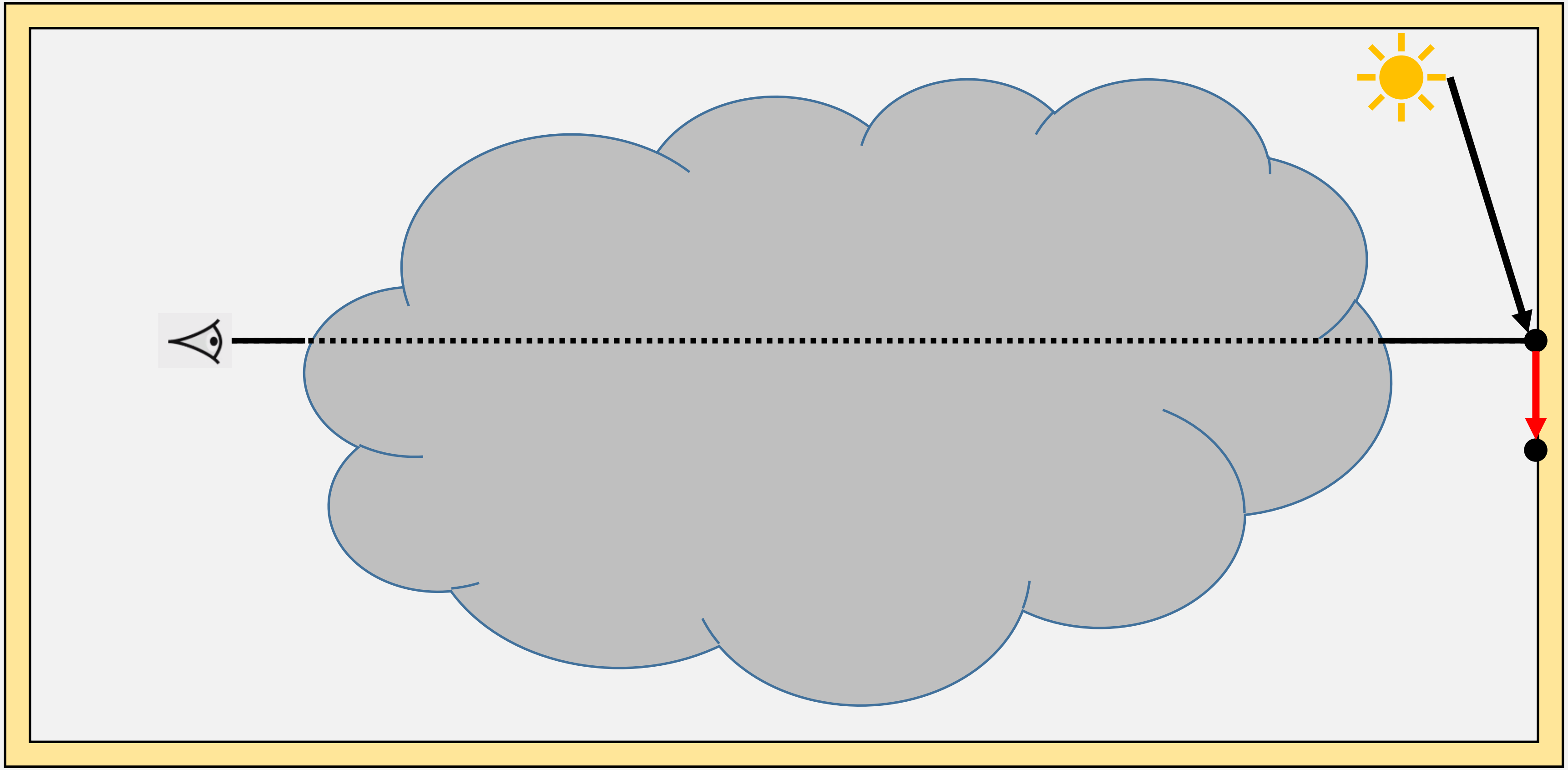
- Radiance cache for participating media

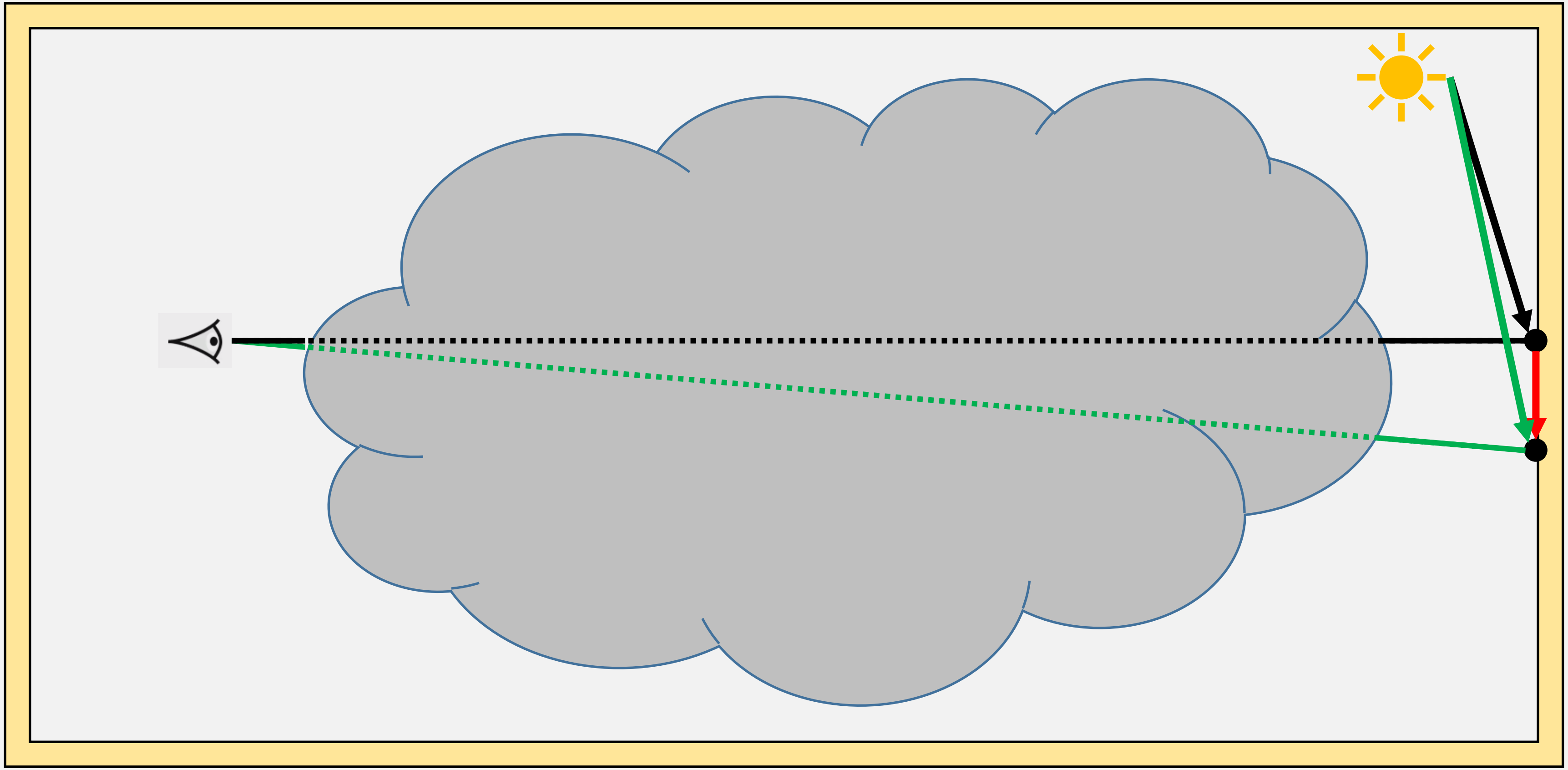


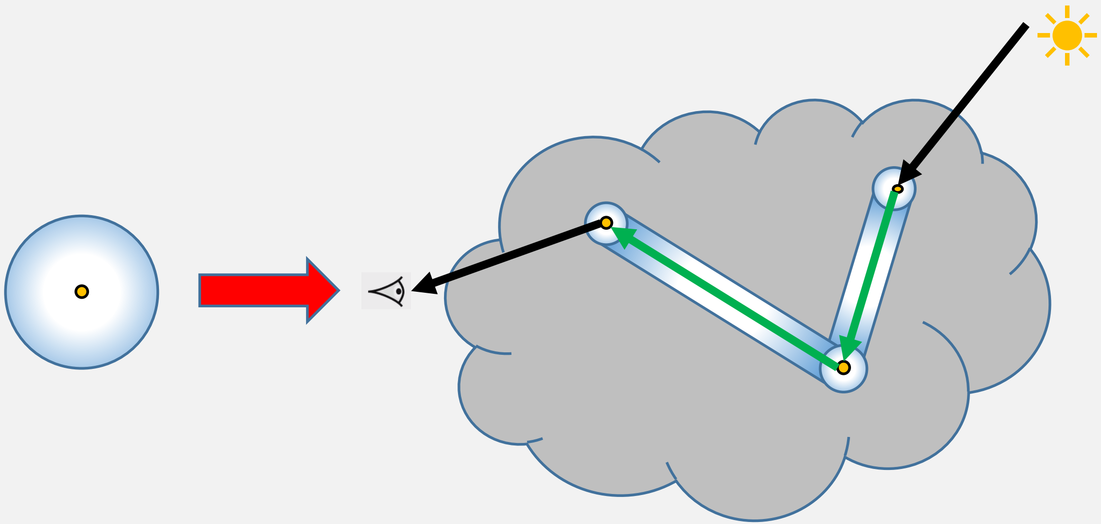


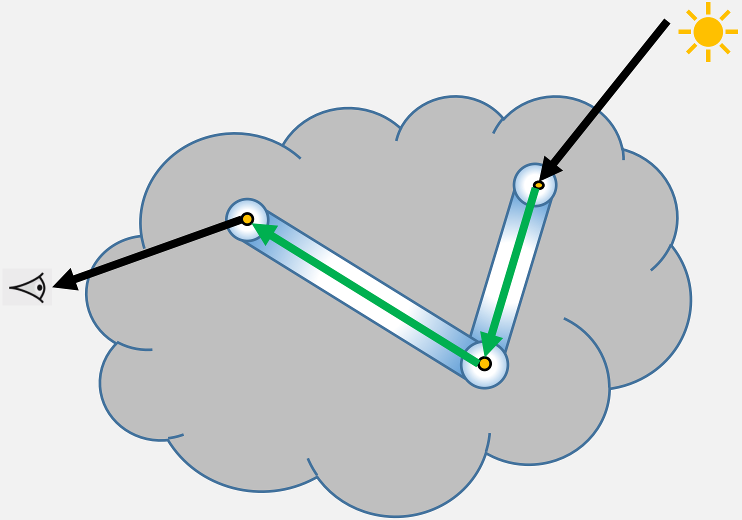






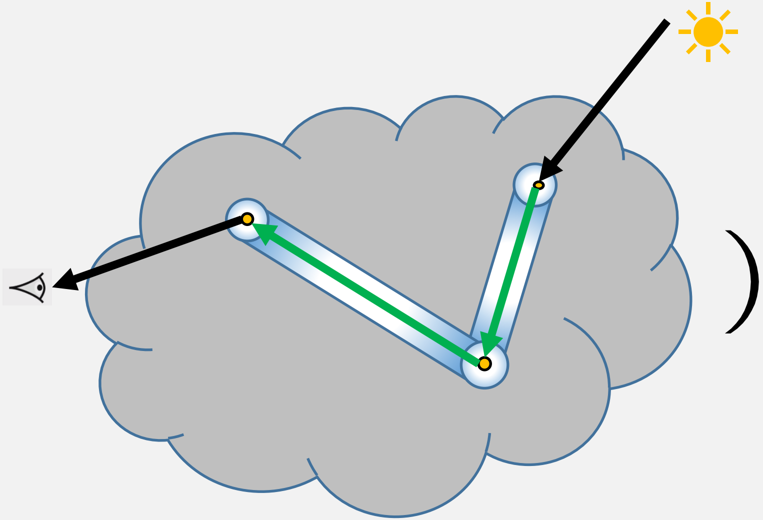






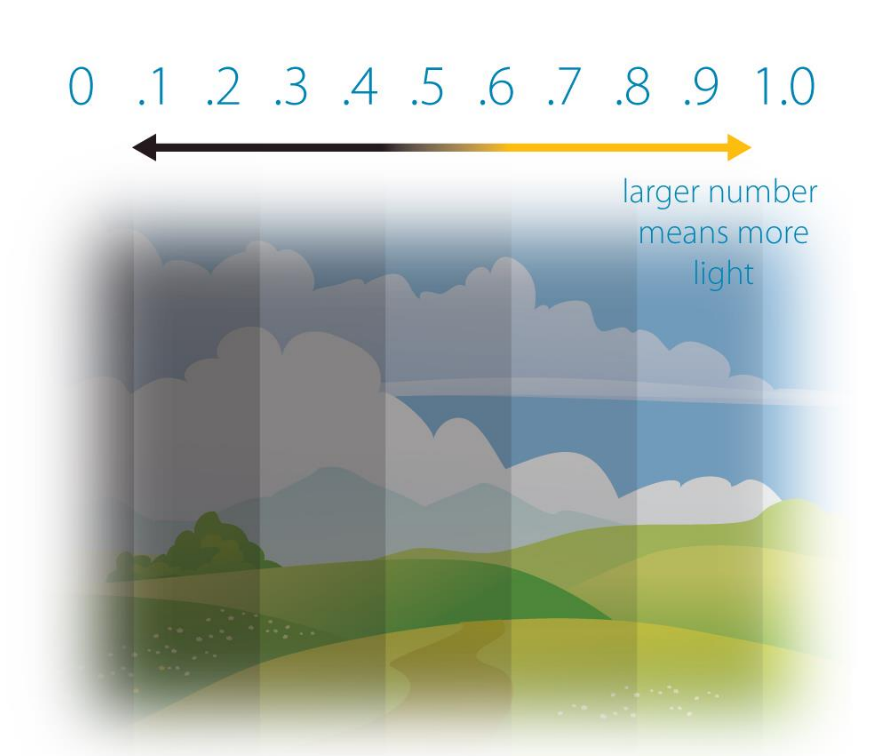
$$= \begin{bmatrix} Tr \\ f_p \\ G \end{bmatrix}$$

$$\nabla L\Big($$



$$\Big) = \begin{bmatrix} \nabla Tr \\ \nabla f_p \\ \nabla G \end{bmatrix}$$

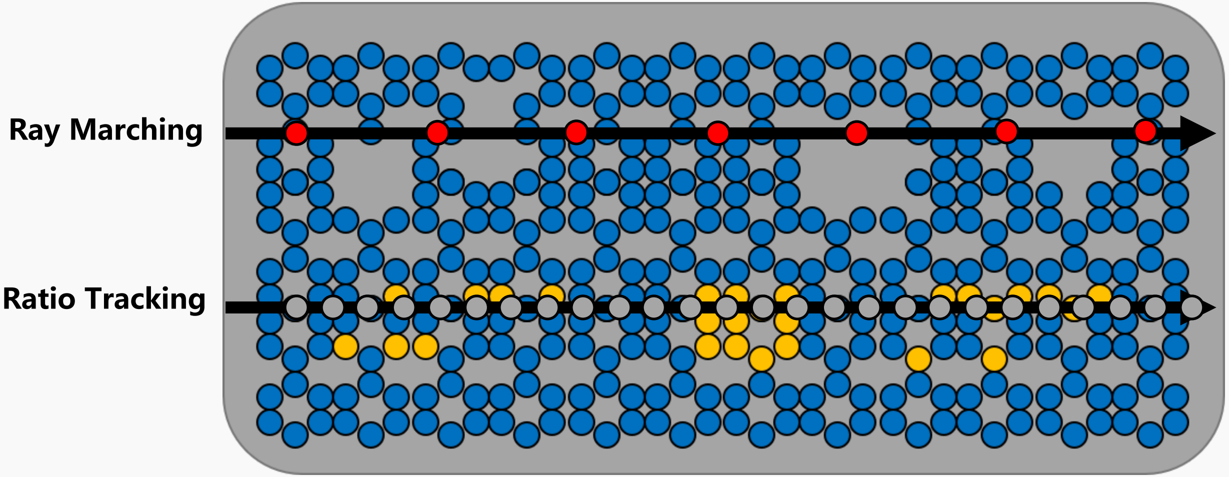
Transmittance



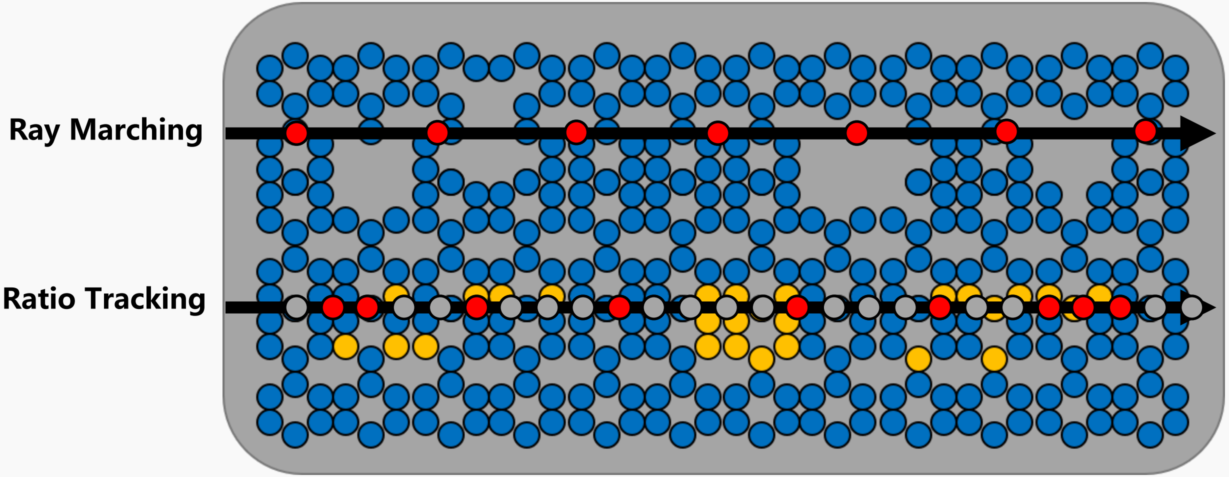
https://bayviewwindows.ca

Transmittance gives the fraction of radiance that is transmitted between two points

Ratio Tracking

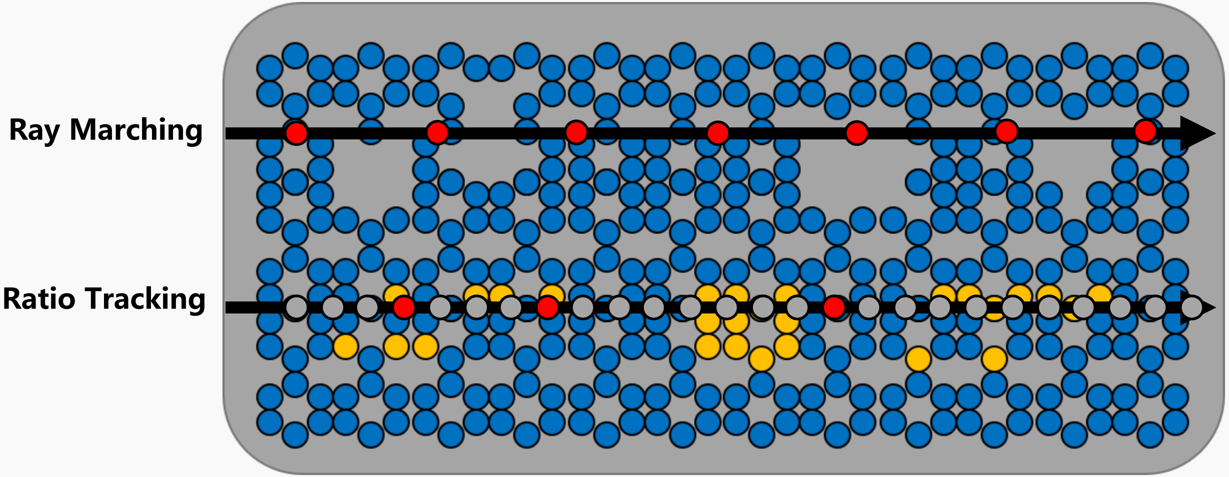


Ratio Tracking



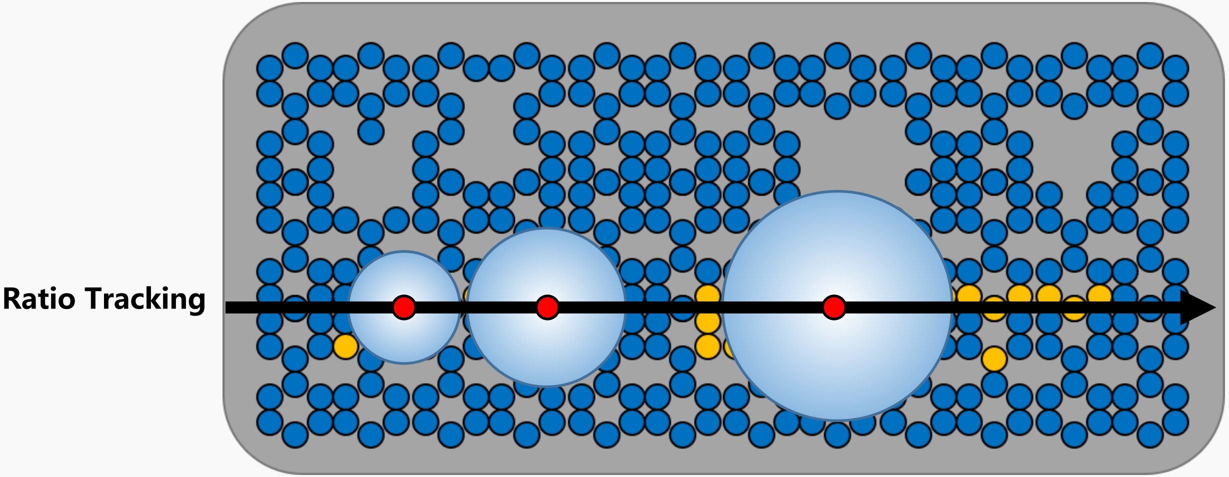
$$T_r(x_i \leftrightarrow x_{i+1}) = \prod_{j=0}^{k}\left(1 - \frac{\sigma_t(x_j)}{\overline{\sigma}}\right)$$

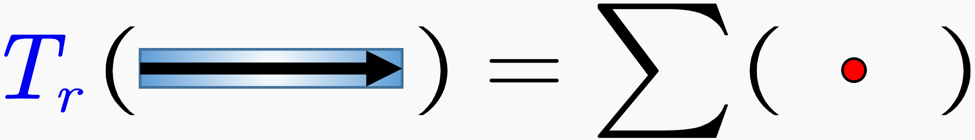
Ratio Tracking



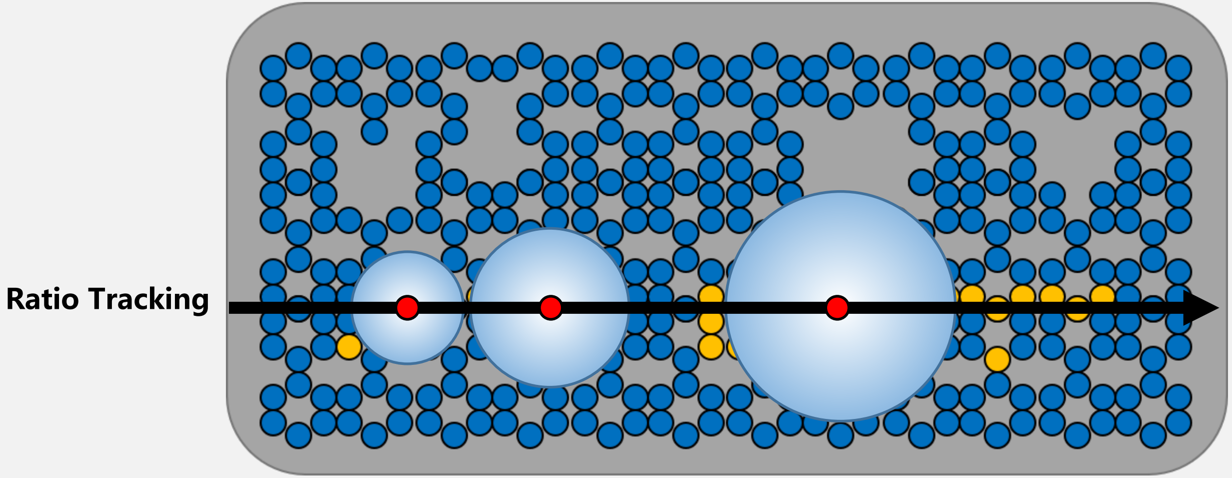
$$T_r(x_i \leftrightarrow x_{i+1}) = \prod_{j=0}^{k}\left(1 - \frac{\sigma_t(x_j)}{\bar{\sigma}}\right)$$

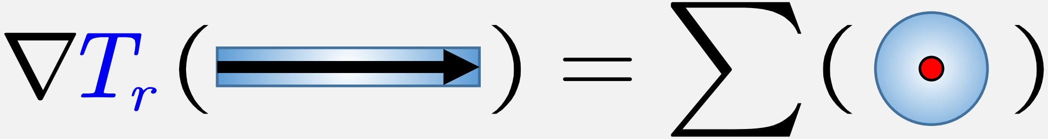
Gradient of Transmittance



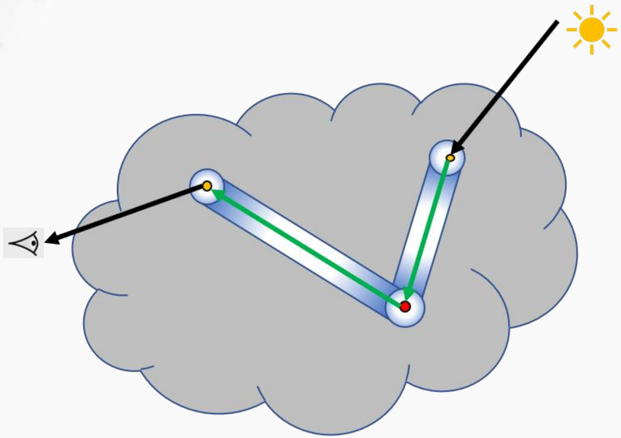


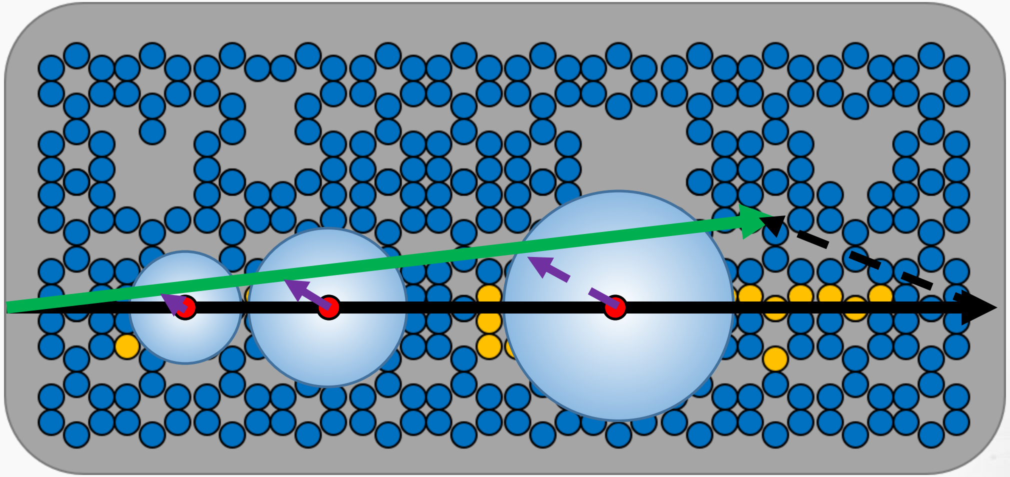
Gradient of Transmittance





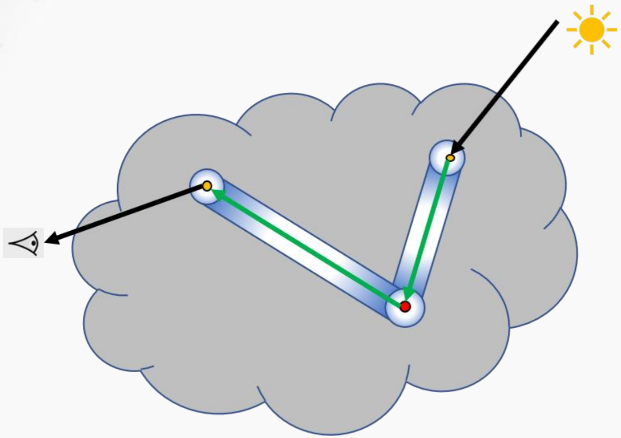
Gradient of Transmittance

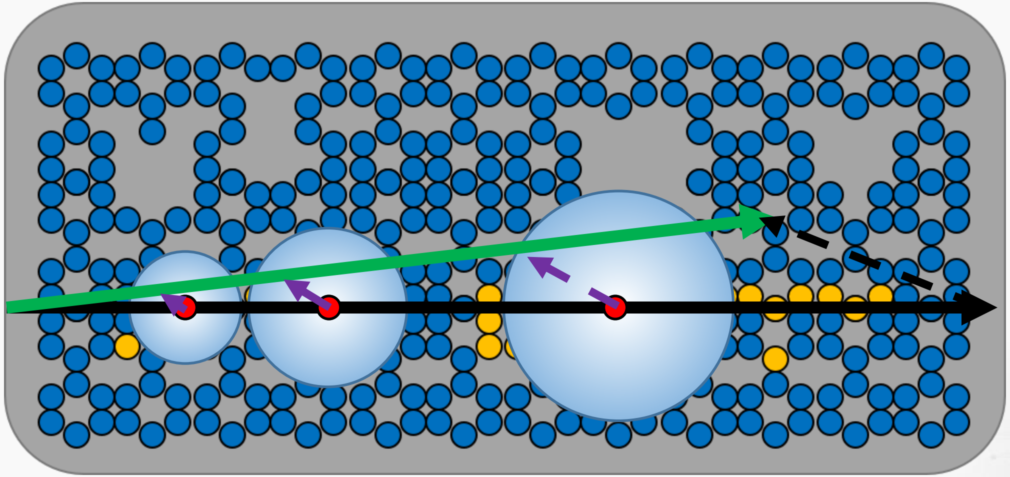




$$\nabla\left(T_r\left(x_i \leftrightarrow x_{i+1}\right)\right) = -T_r\left(x_i \leftrightarrow x_{i+1}\right)\left(\sum_{j=0}^{n} \frac{\frac{c_j}{c_n}\nabla\sigma_t\left(x_j\right)}{\bar{\sigma}-\sigma_t\left(x_j\right)} + \sigma_{avg}\frac{\overrightarrow{x_i x_{i+1}}}{\left\|\overrightarrow{x_i x_{i+1}}\right\|}\right)$$

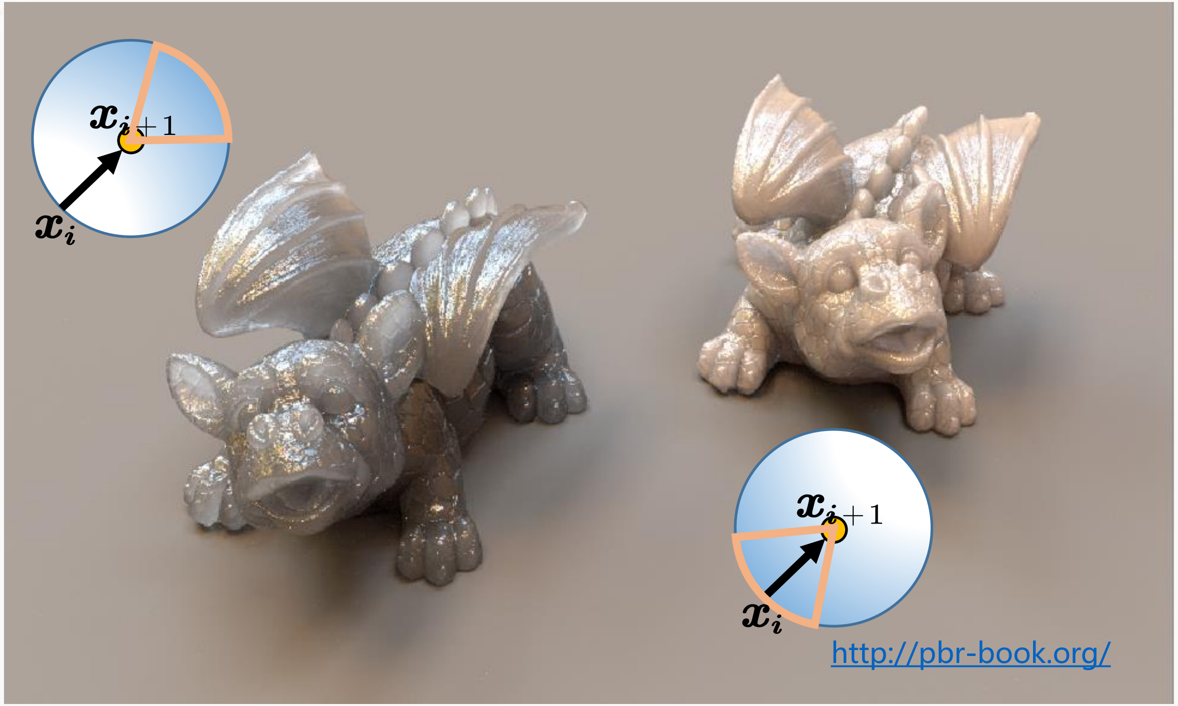
Gradient of Transmittance





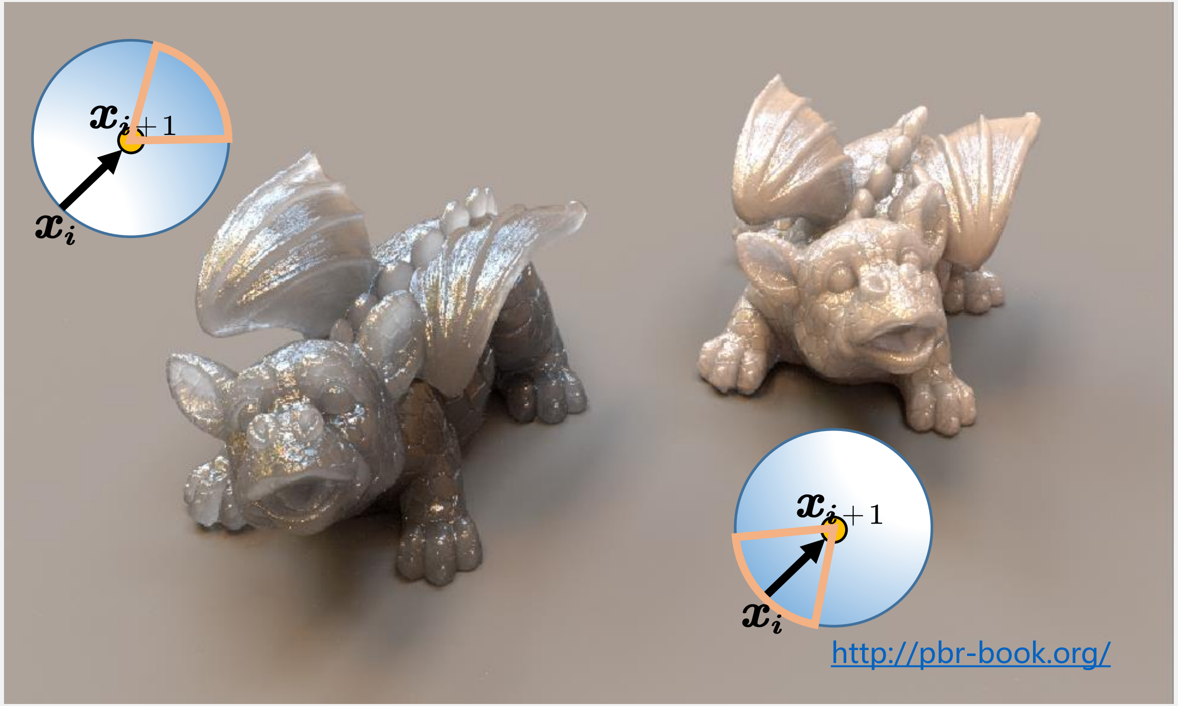
$$T_r(\nearrow) \approx T_r(\longrightarrow) + \langle \nabla T_r(\Longrightarrow), V(\searrow) \rangle$$

Phase function



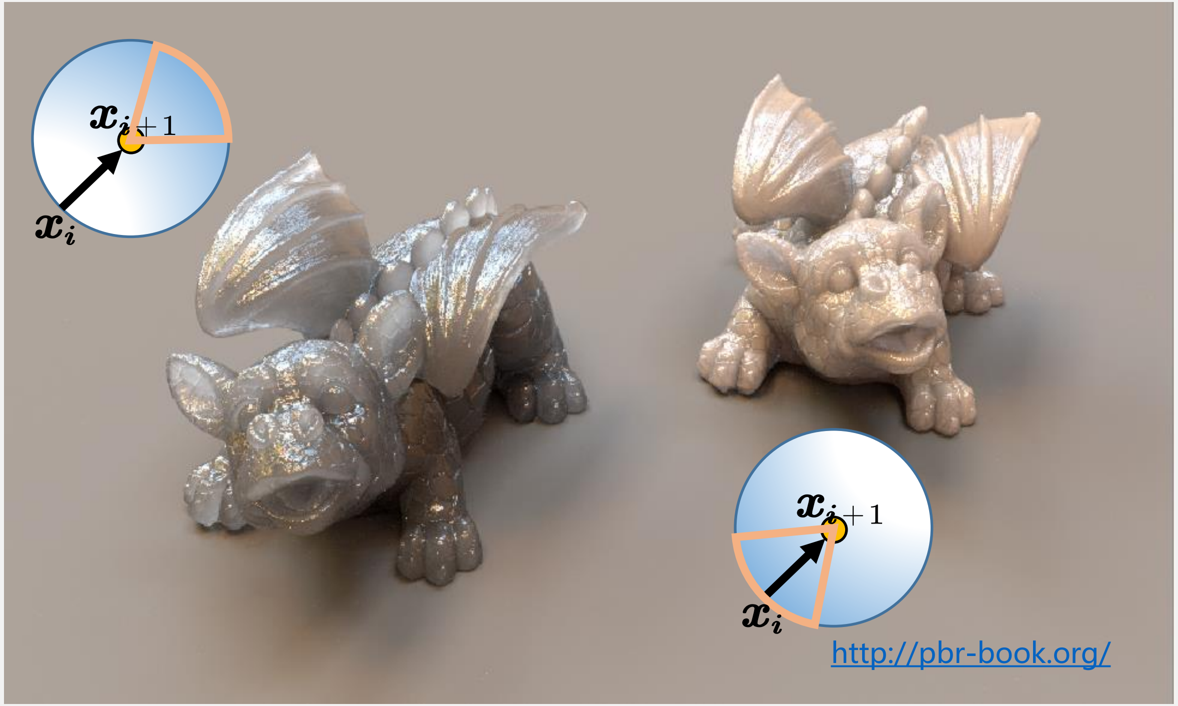
http://pbr-book.org/

Phase function



A Phase function describes this angular distribution of scattered radiance at a given point.

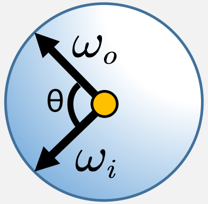
Phase function



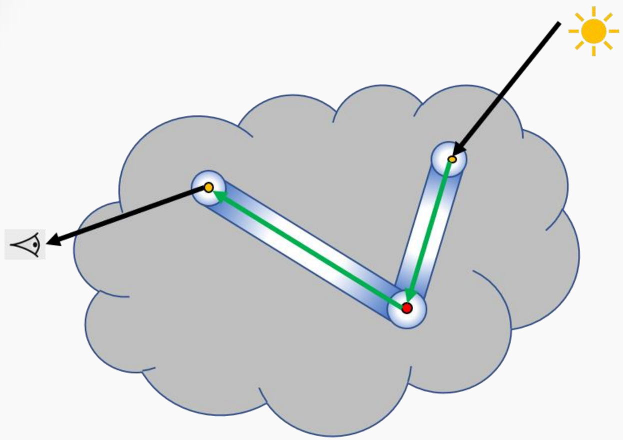
A Phase function describes this angular distribution of scattered radiance at a given point.

Henyey-Greenstein phase function:

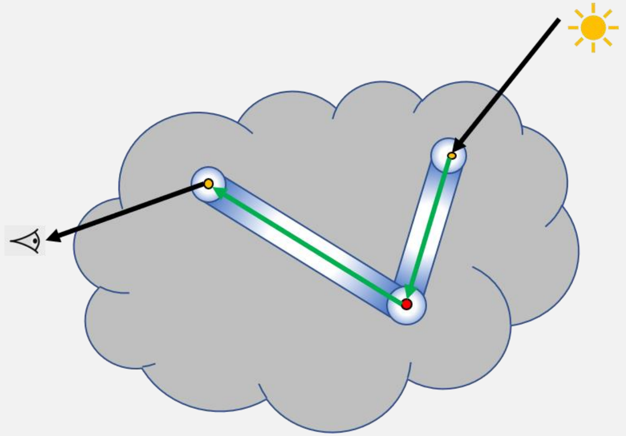
$$f_p = \frac{1}{4\pi}\frac{1-g^2}{(1+g^2-2g(\cos\theta))^{3/2}}$$

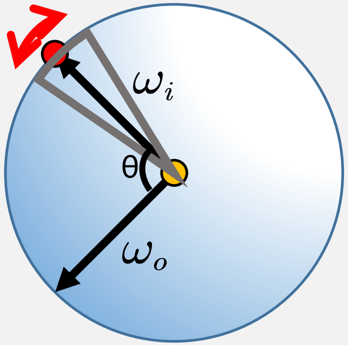


Gradient of Phase function



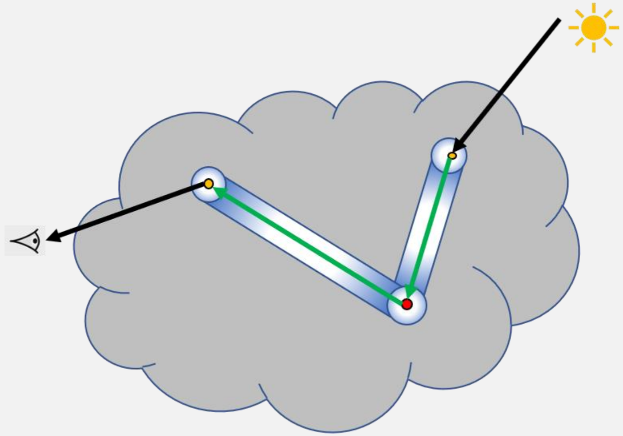
Gradient of Phase function

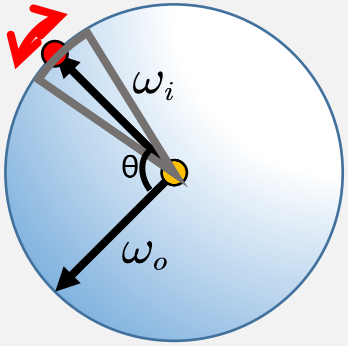




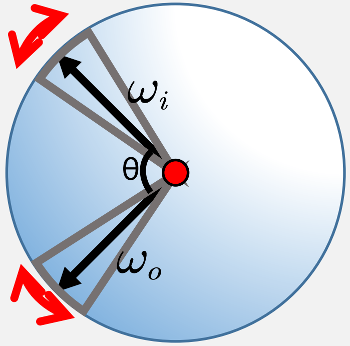
$$\nabla \cos\theta(\omega_i) = \frac{\hat{\omega}_o}{|\omega_i|} - \cos\theta \frac{\omega_i}{|\omega_i|^2}$$

Gradient of Phase function





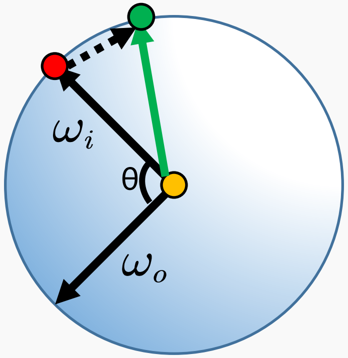
$$\nabla \cos\theta(\omega_i) = \frac{\hat{\omega}_o}{|\omega_i|} - \cos\theta \frac{\omega_i}{|\omega_i|^2}$$

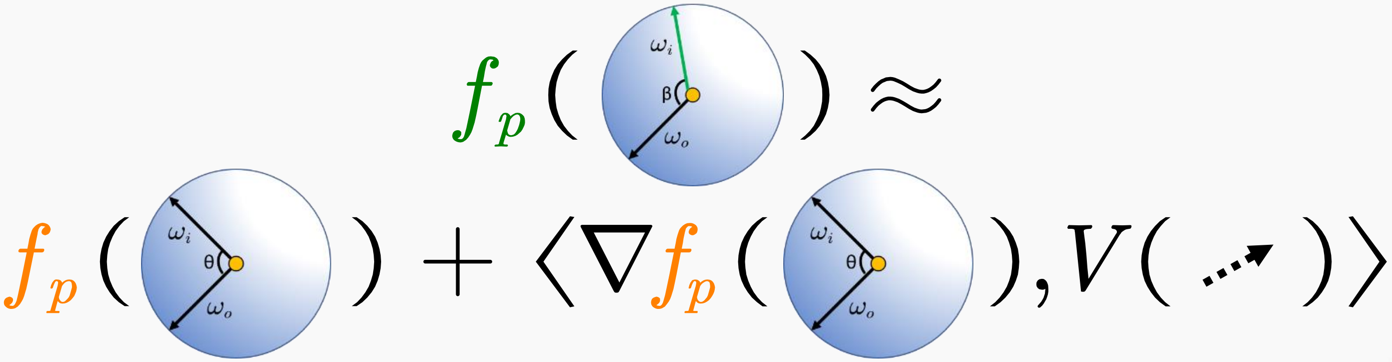


$$\nabla \cos\theta(\omega_i, \omega_o) =$$

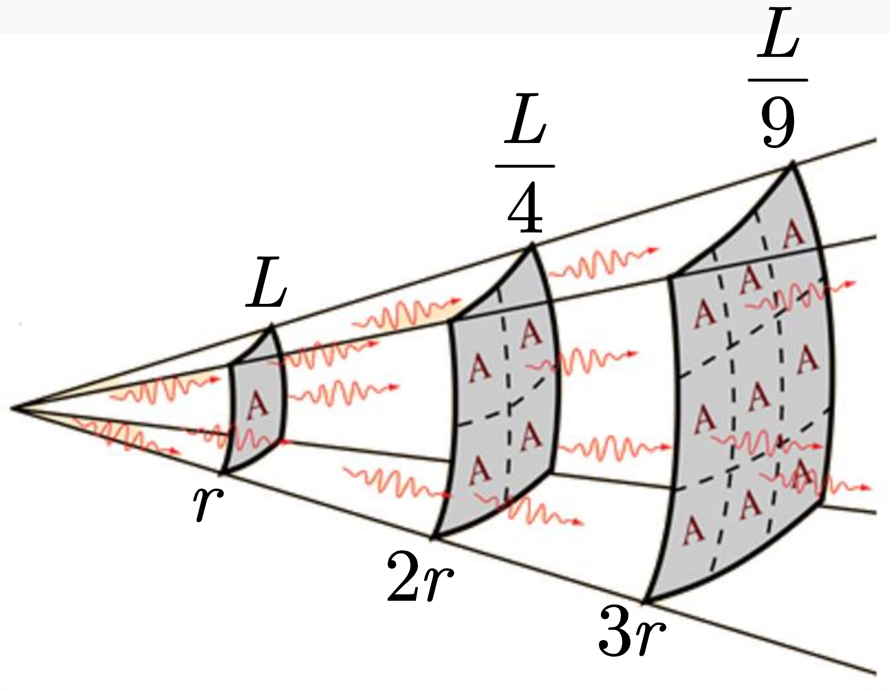
$$\frac{\hat{\omega}_o}{|\omega_i|} - \cos\theta \frac{\omega_i}{|\omega_i|^2} + \frac{\hat{\omega}_i}{|\omega_o|} - \cos\theta \frac{\omega_o}{|\omega_o|^2}$$

Gradient of Phase function



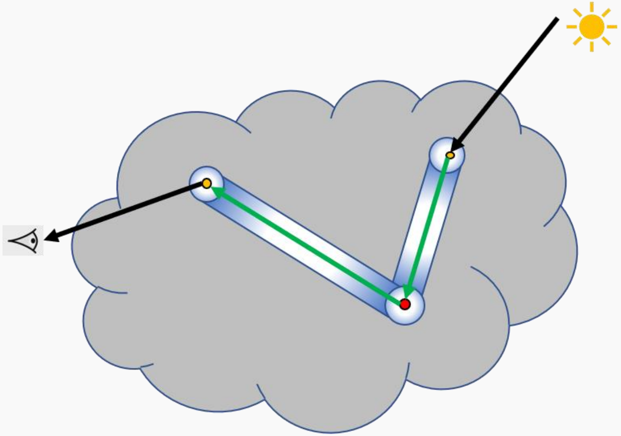


Geometry factor



The geometry factor is the derivative of projected solid angle with respect to the area

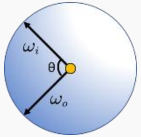
Gradient of Geometry factor

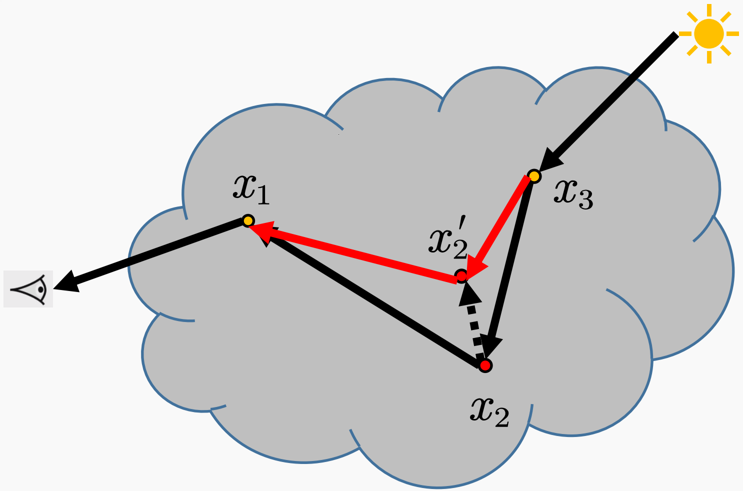


$$G(\rightarrow) = \frac{1}{\|\rightarrow\|^2}$$

$$\nabla G(\rightarrow) = \frac{2}{\|\rightarrow\|^4}\overrightarrow{(\rightarrow)}$$

Gradient of light path

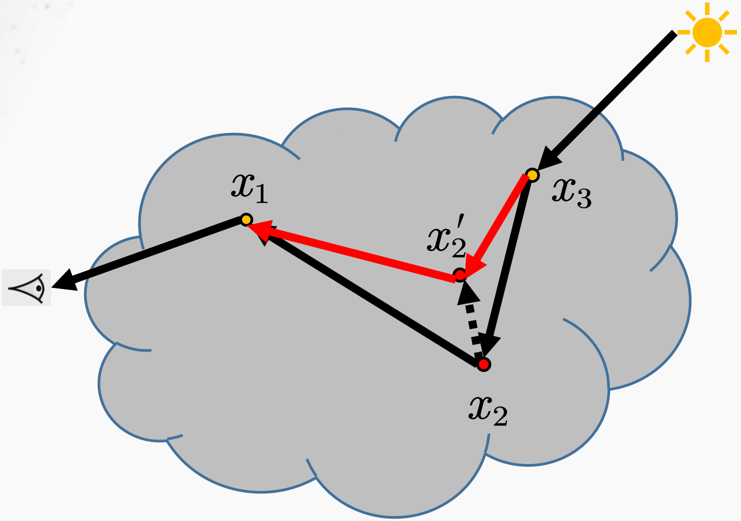
$\nabla T_r(\;)$ $\quad \nabla f_p($  $)$ $\quad \nabla G(\longrightarrow)$



$$\nabla L = L^* \left(\frac{\nabla T_r(x_1 \leftrightarrow x_2)}{T_r(x_1 \leftrightarrow x_2)} + \frac{\nabla T_r(x_2 \leftrightarrow x_3)}{T_r(x_2 \leftrightarrow x_3)} + \frac{\nabla f_p(x_1)}{f_p(x_1)} + \frac{\nabla f_p(x_2)}{f_p(x_2)} + \frac{\nabla f_p(x_3)}{f_p(x_3)} + \frac{\nabla G(x_1 \leftrightarrow x_2)}{G(x_1 \leftrightarrow x_2)} + \frac{\nabla G(x_2 \leftrightarrow x_3)}{G(x_2 \leftrightarrow x_3)} \right)$$

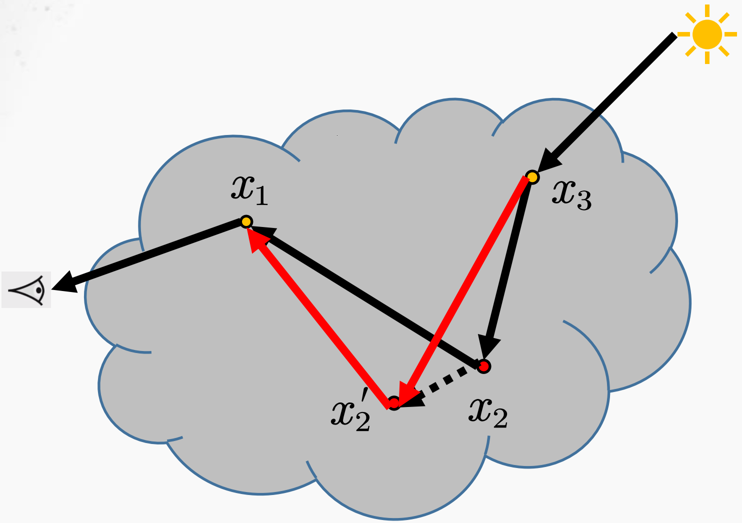
$$L(\rightarrow) \approx L(\rightarrow) + \langle \nabla L(\rightarrow), V(\uparrow) \rangle$$

Gradient of light path



$$L(\rightarrow) \approx \boxed{L(\rightarrow)} + \langle \boxed{\nabla L(\rightarrow)}, V(\uparrow) \rangle$$

Gradient of light path

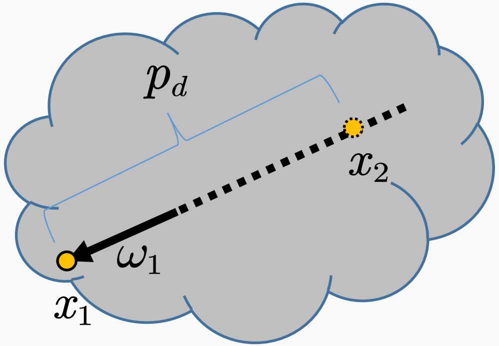


$$L(\longrightarrow) \approx \boxed{L(\longrightarrow)} + \langle \boxed{\nabla L(\longrightarrow)}, V(\swarrow) \rangle$$

Implementation

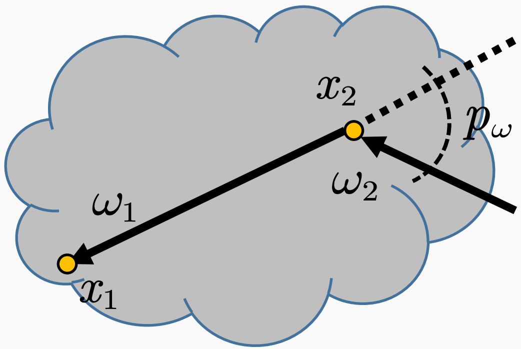
- MLT
  - BDPT
  - M-H algorithm

Sampling in the medium



(1) distance

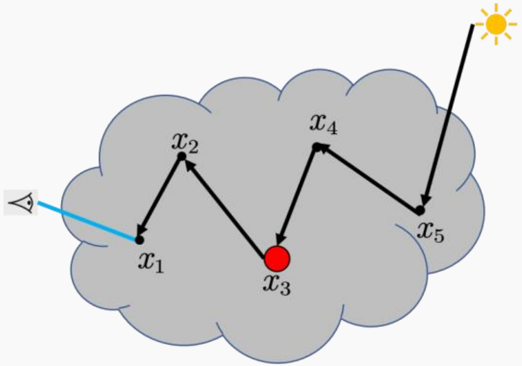
Sampling in the medium



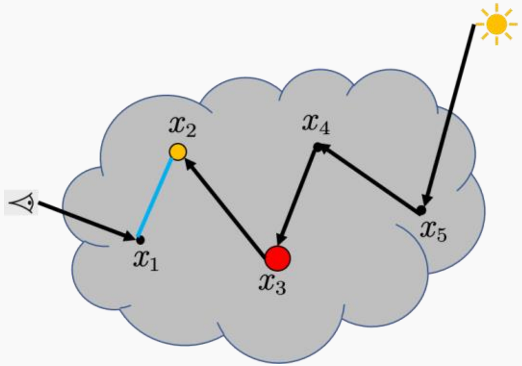
(2) direction

$$pdf_{medium} = pdf_{distance} \cdot pdf_{direction}$$

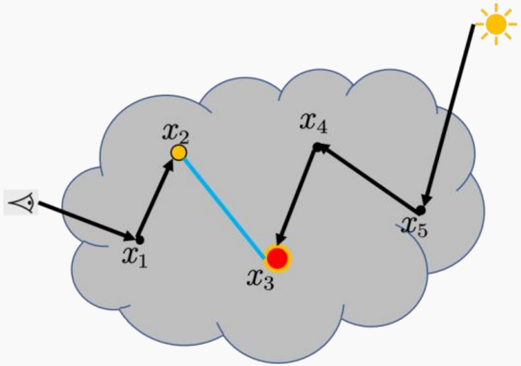
BDPT Connection



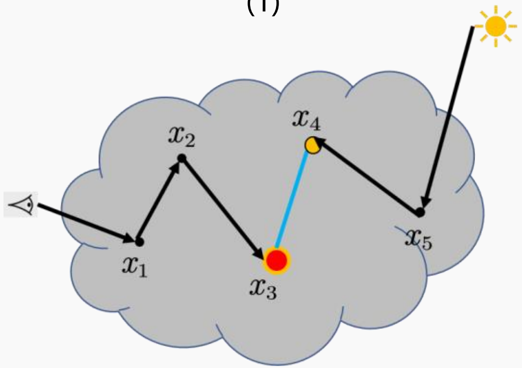
(1)



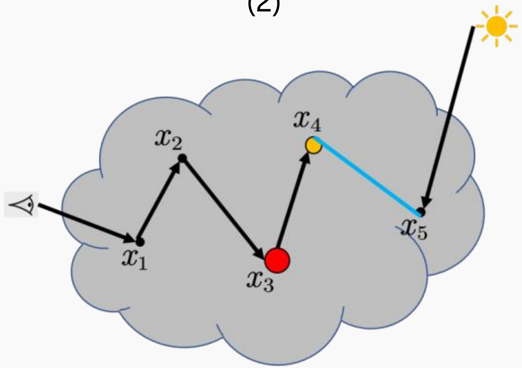
(2)



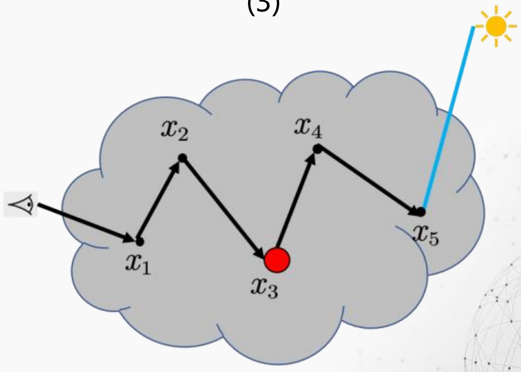
(3)



(4)



(5)



(6)

Vertex Type

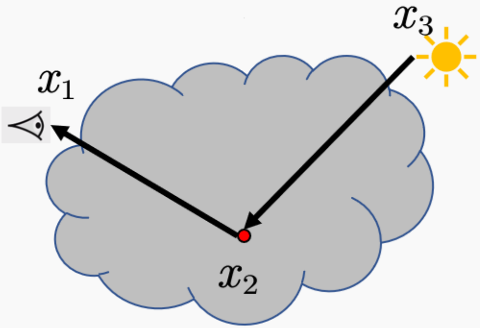
Diffuse

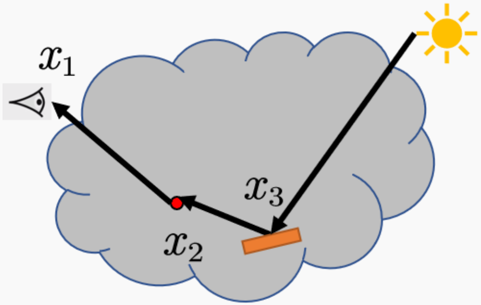
Specular

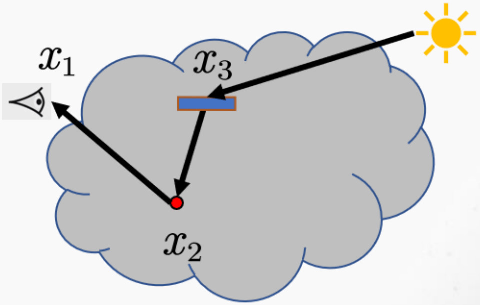
(1) Camera

(2) Surface

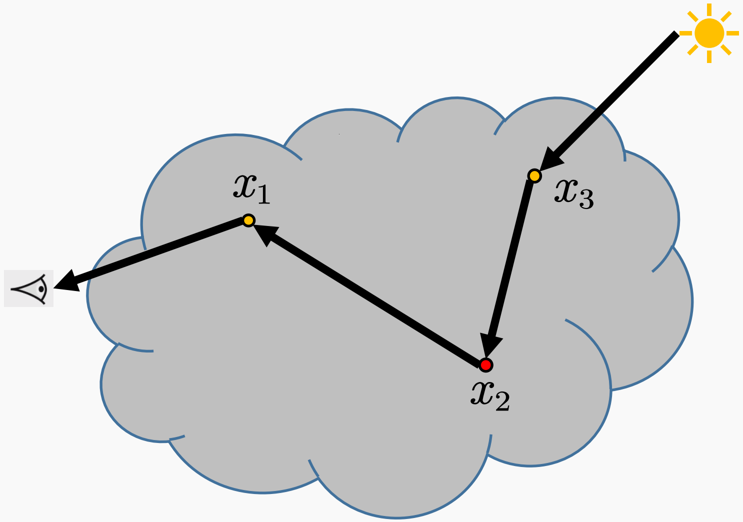
(3) Light





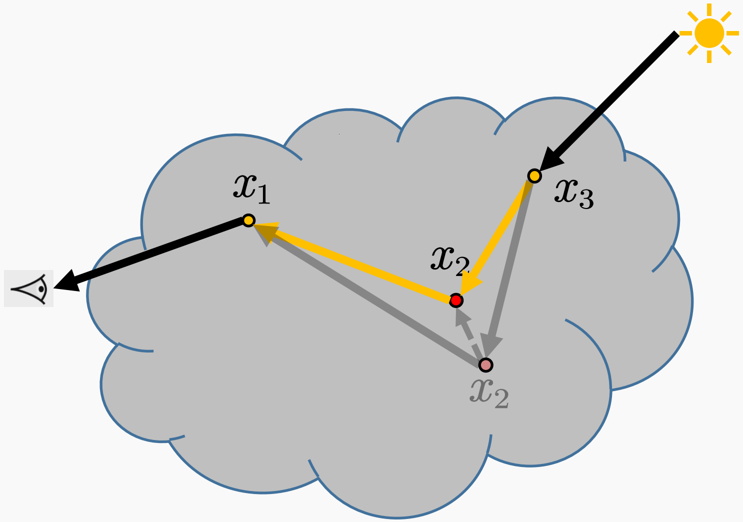


Gradient Mutation Strategy

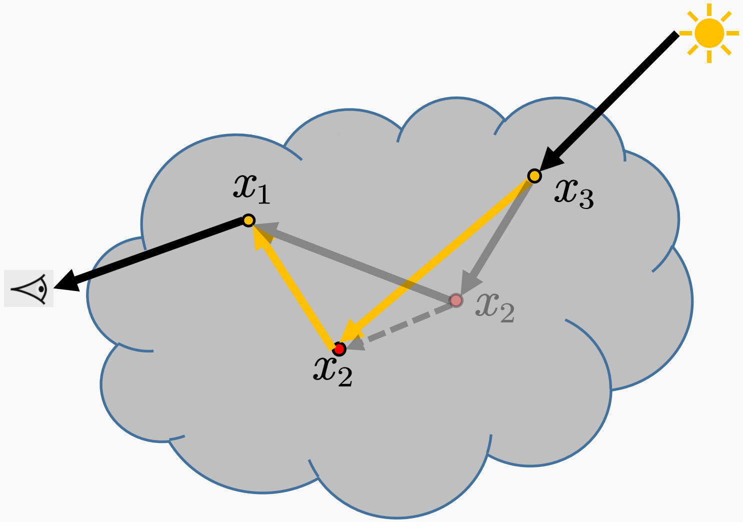


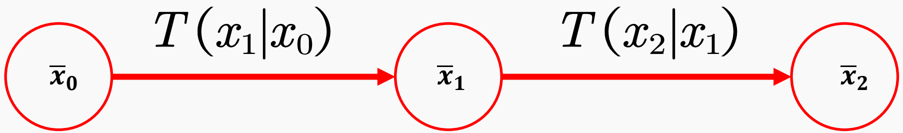
$\bar{x}_0$

Gradient Mutation Strategy

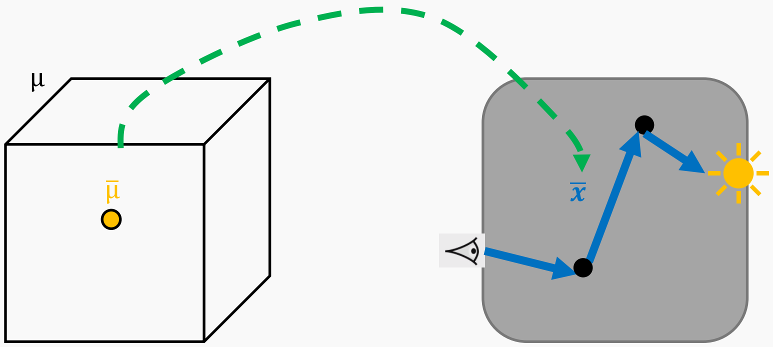


Gradient Mutation Strategy

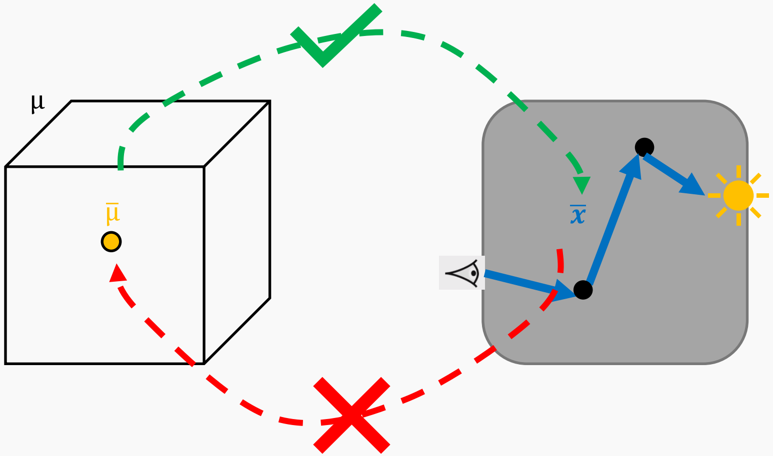




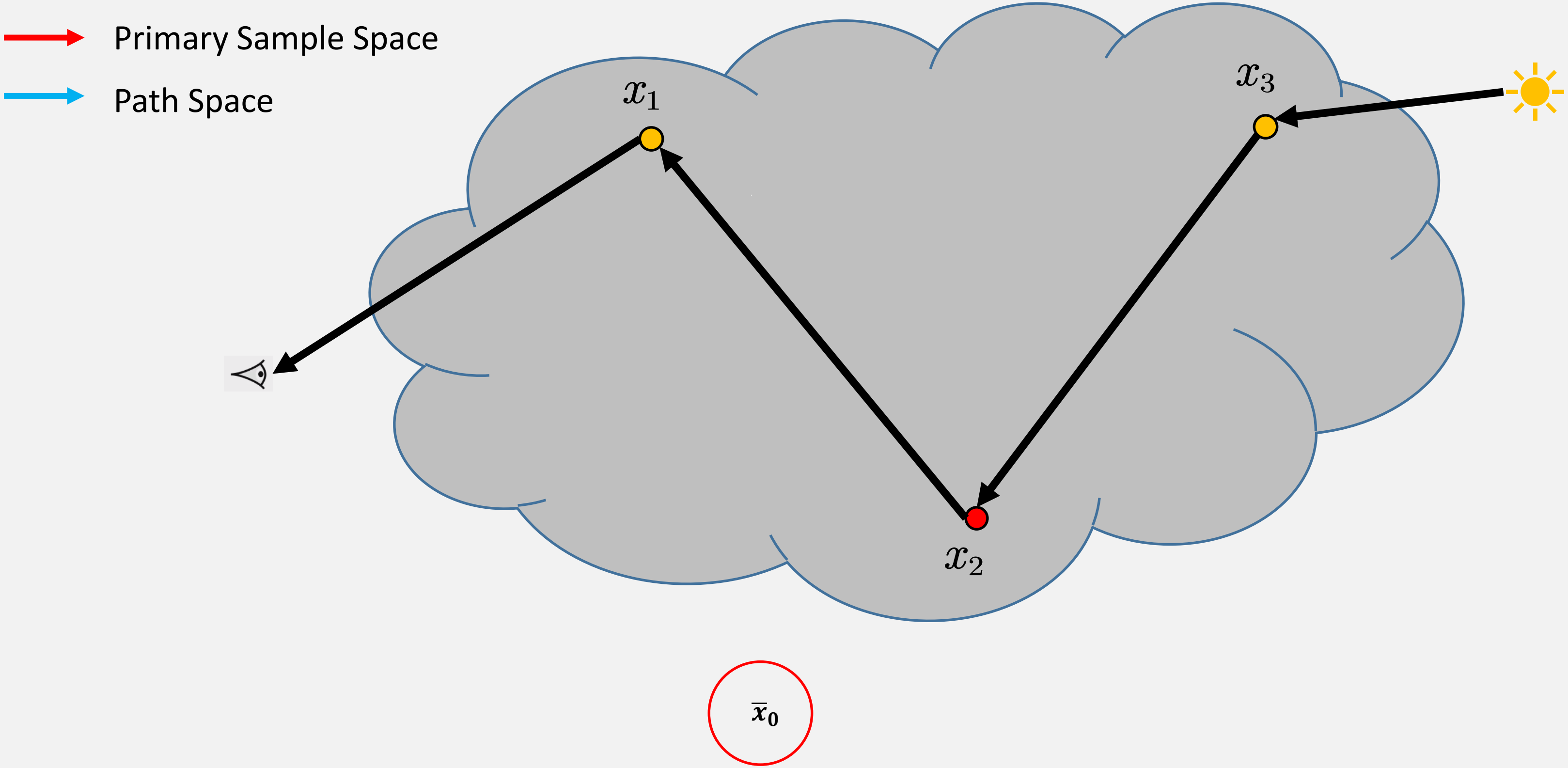
Primary Sample Space



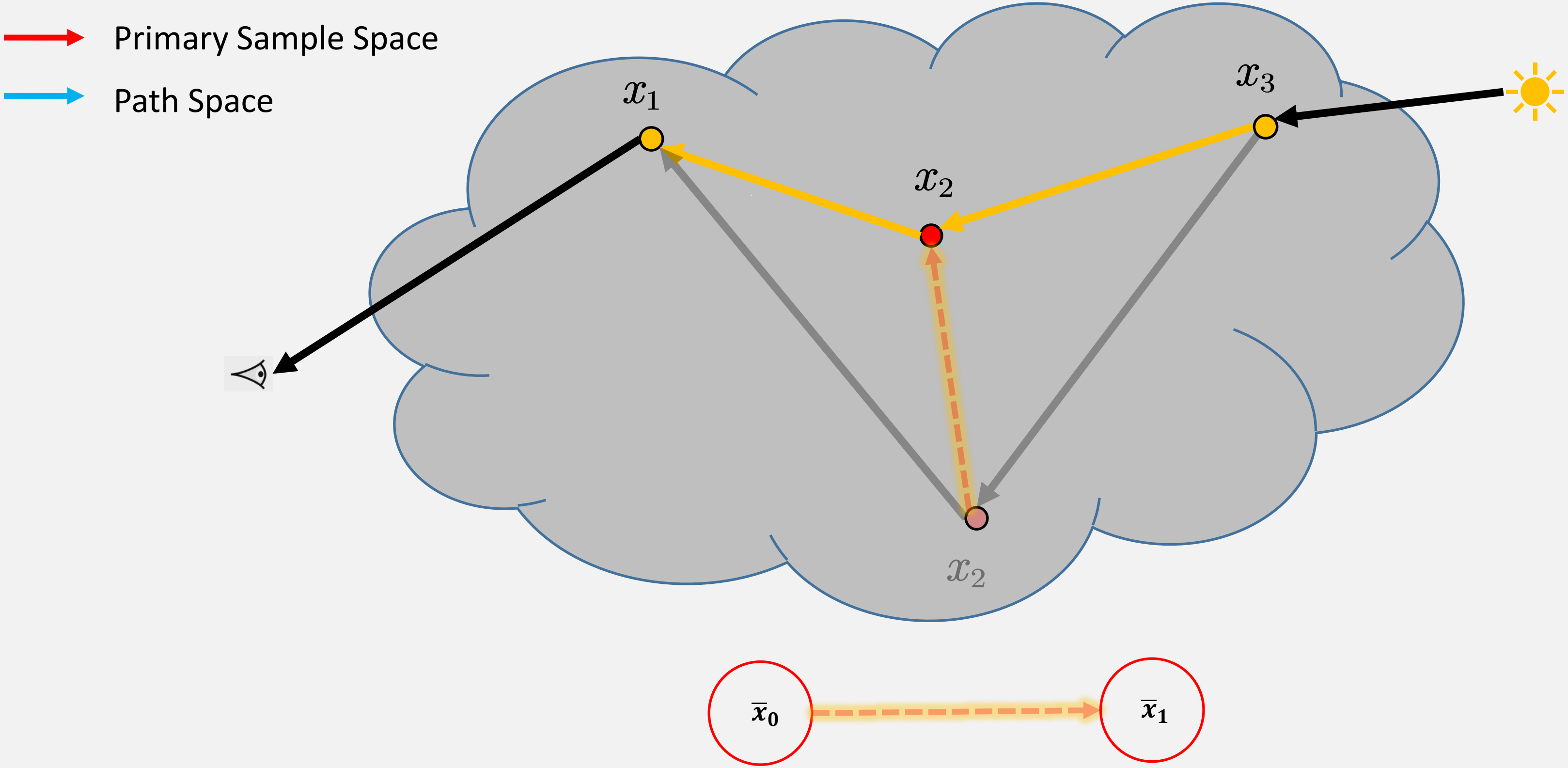
Primary Sample Space



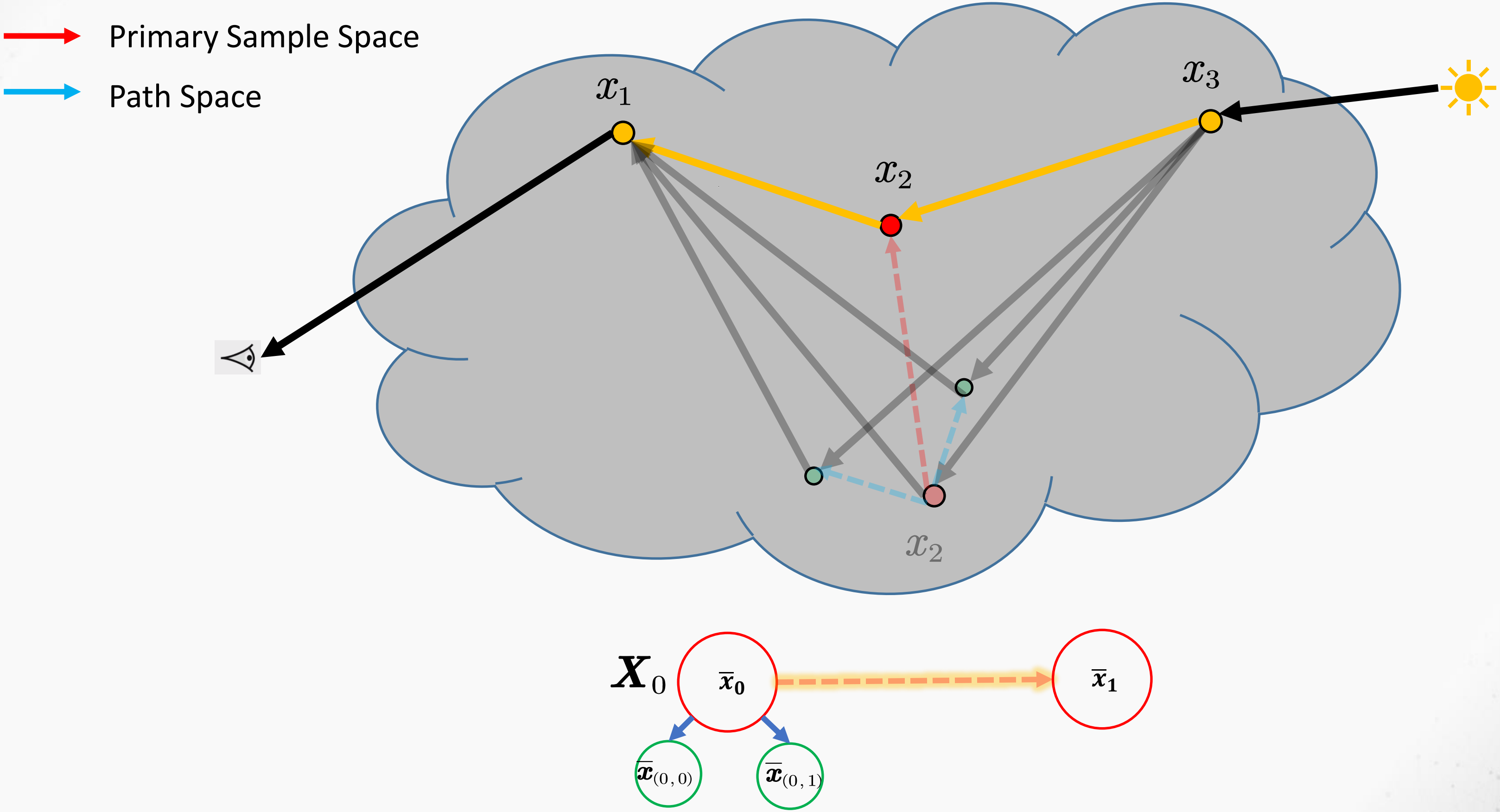
Gradient Mutation Strategy



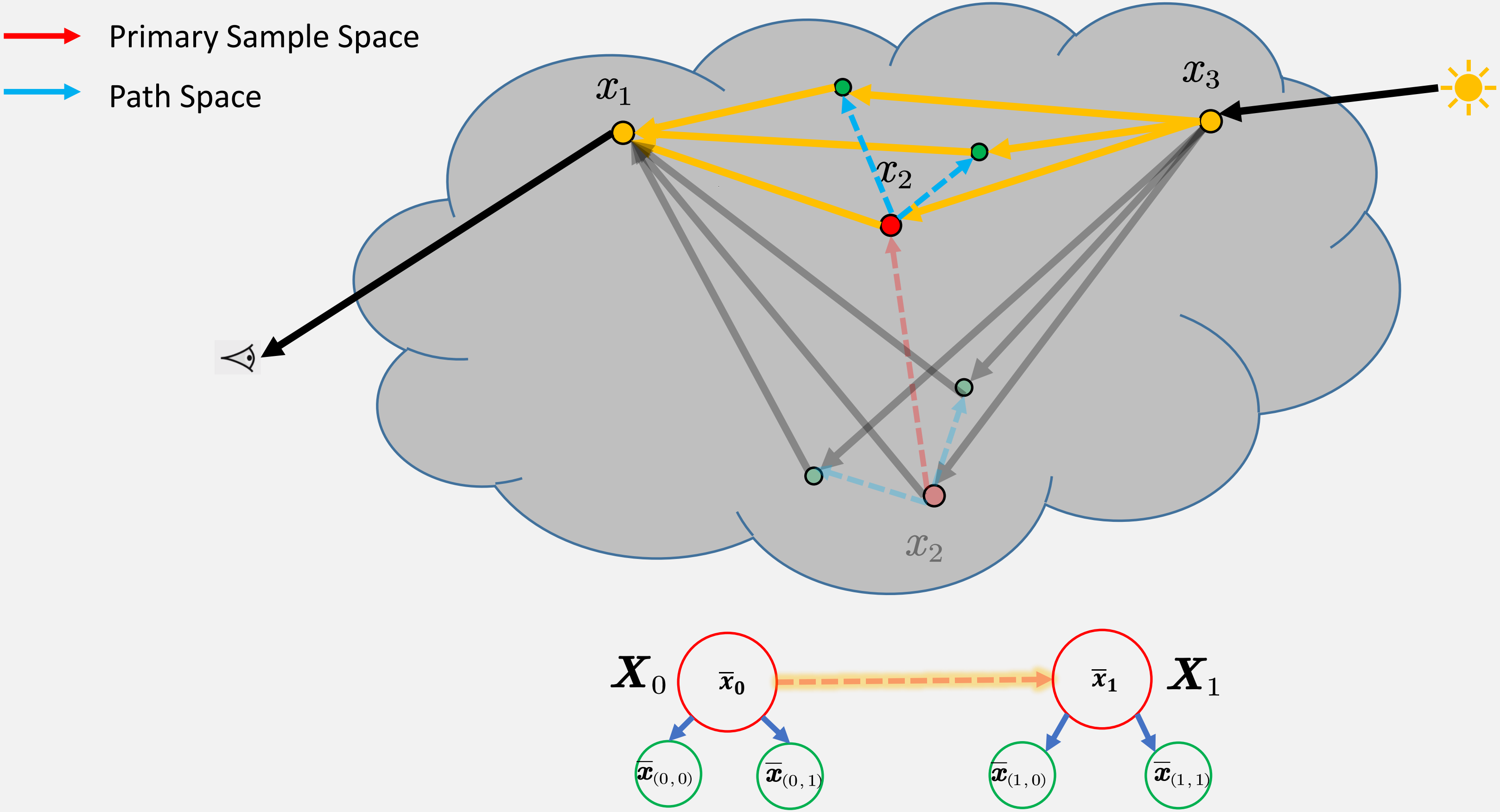
Gradient Mutation Strategy



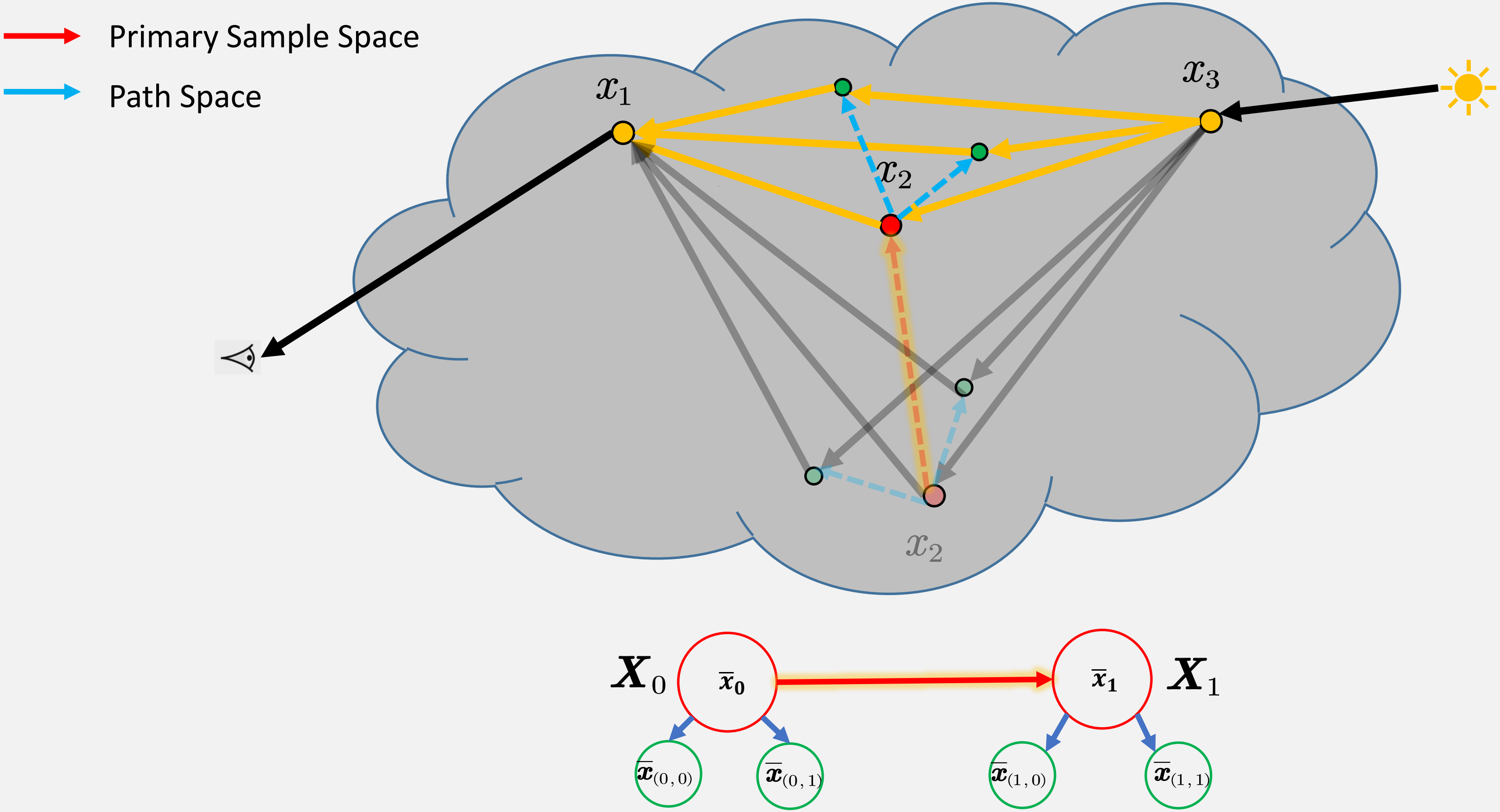
Gradient Mutation Strategy



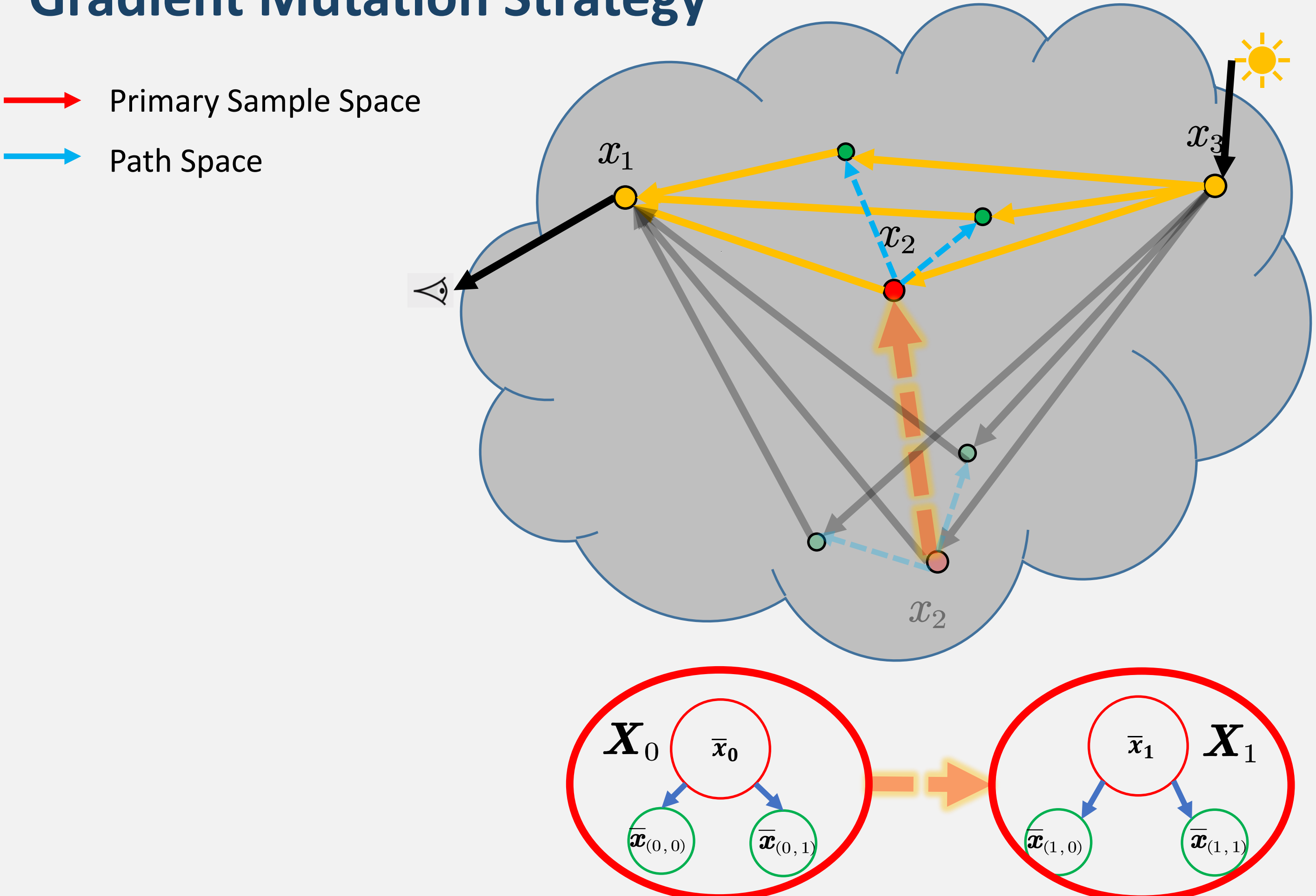
Gradient Mutation Strategy



Gradient Mutation Strategy



Gradient Mutation Strategy



01 Motivation

02 Background

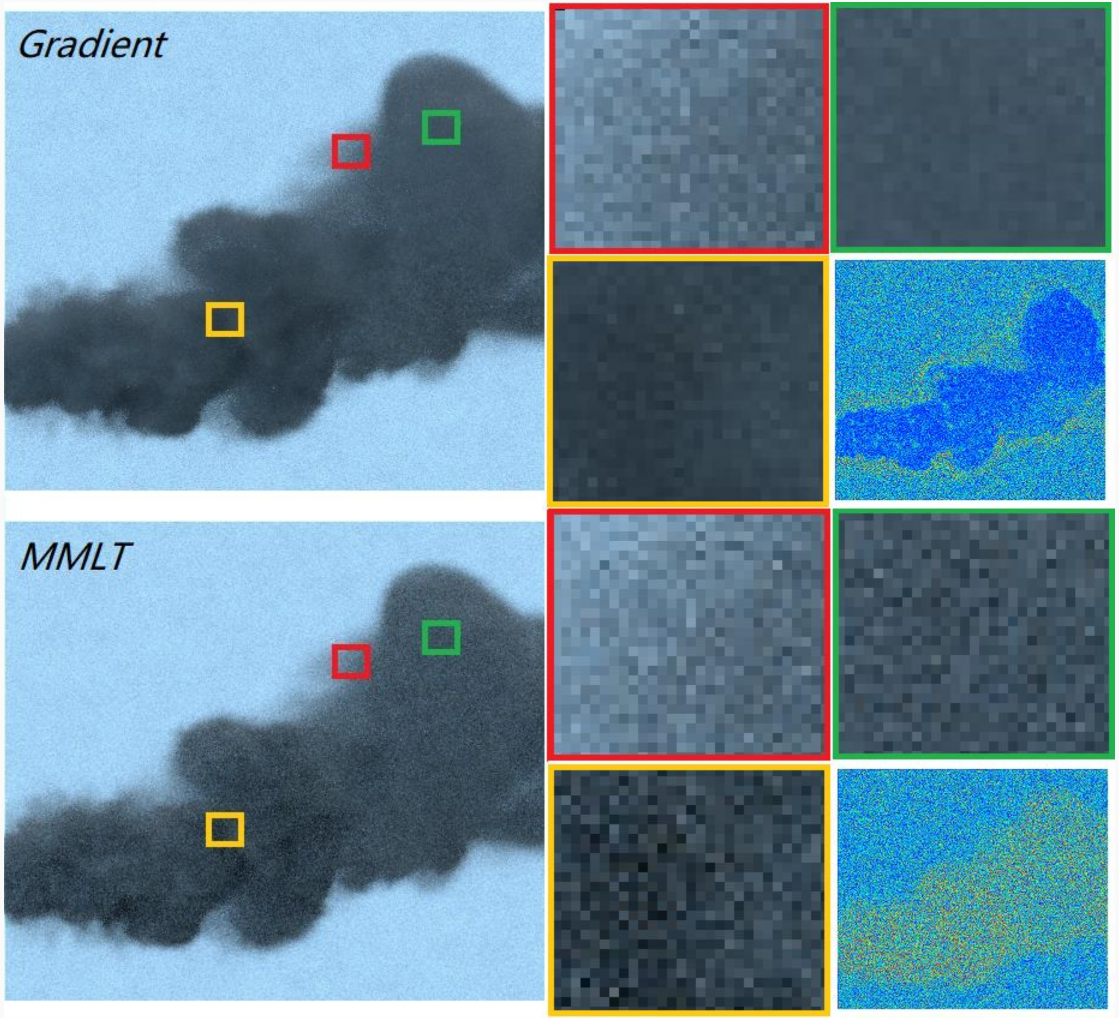
03 Gradient Algorithm

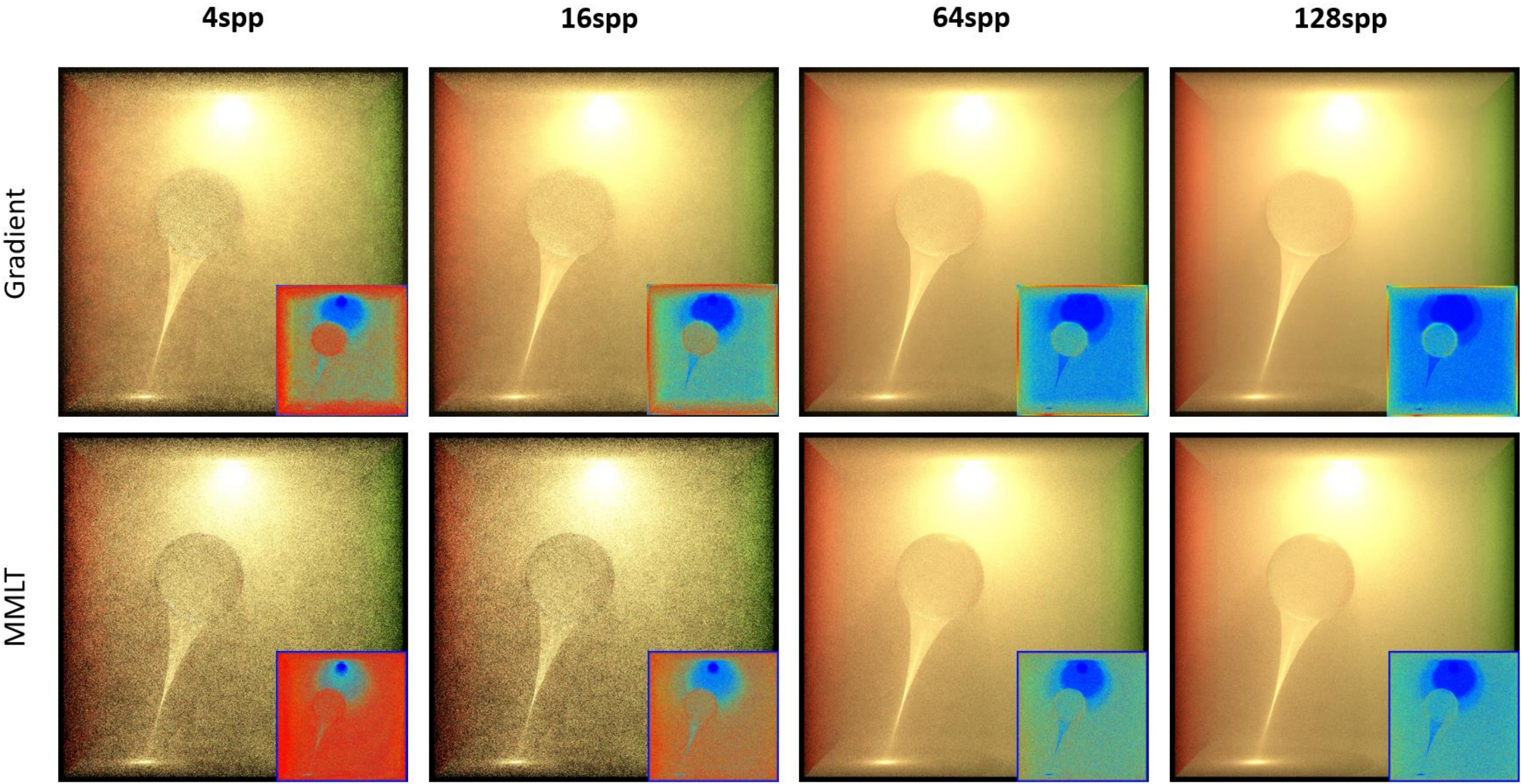
04 Result

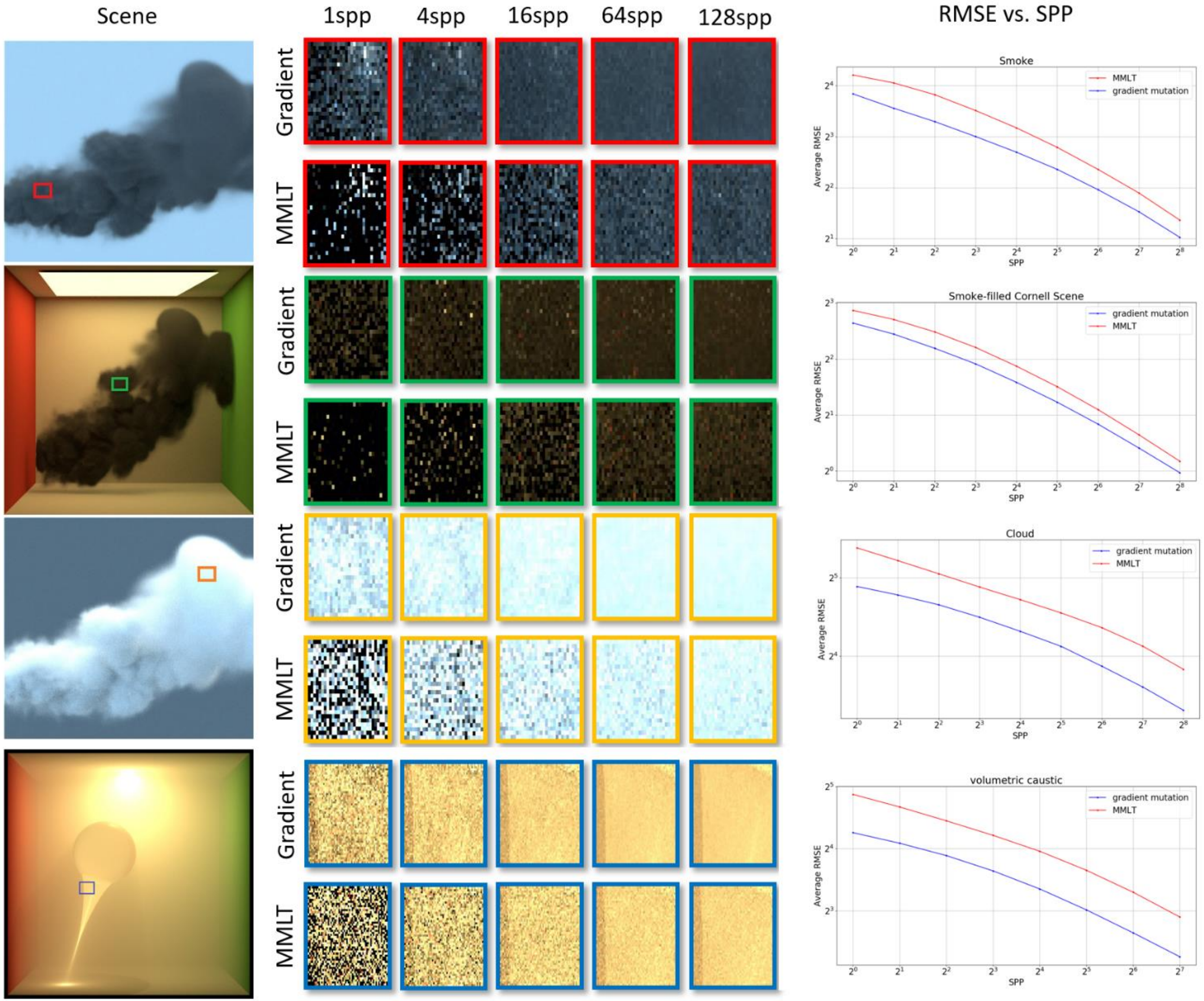
05 Conclusion

Result

- All result rendered:
  - With up to 128 samples per pixel
  - an Intel Core i7-8550U at 1.8GHz using 8 cores
  - Pbrt-v3
  - 4 scenes


Gradient
MMLT


4spp
16spp
64spp
128spp
Gradient
MMLT



Scene
1spp
4spp
16spp
64spp
128spp
Gradient
MMLT
RMSE vs. SPP
Smoke
Smoke-filled Cornell Scene
Cloud
volumetric caustic
MMLT
gradient mutation
Average RMSE
SPP

01 Motivation

02 Background

03 Gradient Algorithm

04 Result

05 Conclusion

Contributions

- Gradient Algorithm
  - Computation
  - Availability
- Gradient Strategy
  - Path Space Strategy

Limitation

- Gradient algorithm is biased
- Edge Detection

Future Work

- Availability
  - GPU
  - Complicated BSDF function
- Accuracy
  - Precision Loss analysis
  - A well-designed radius-reduction scheme

THANK YOU