



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



01

Motivation

02

Background

03

Gradient Algorithm

04

Result

05

Conclusion



01

Motivation

02

Background

03

Gradient Algorithm

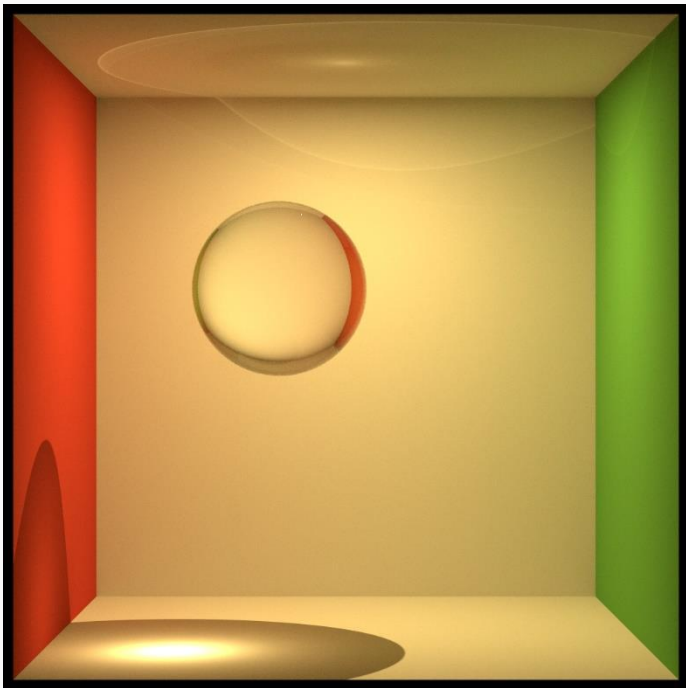
04

Result

05

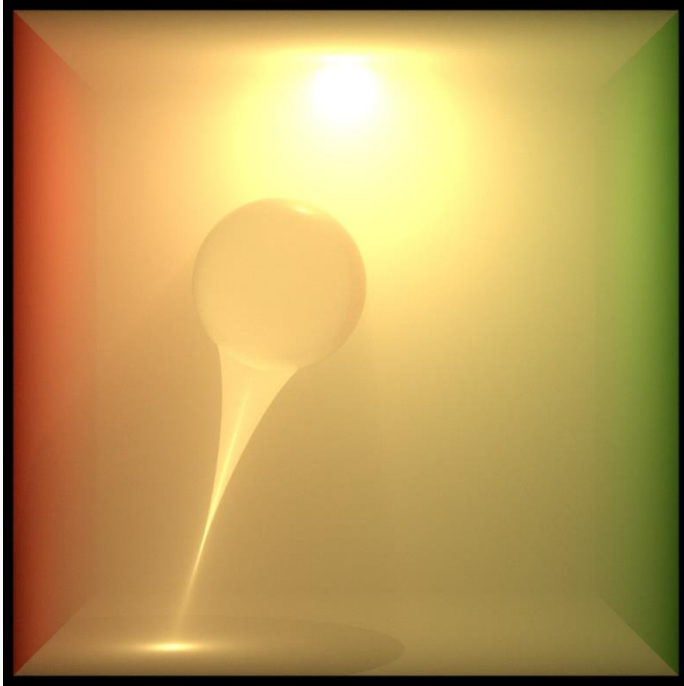
Conclusion

Motivation



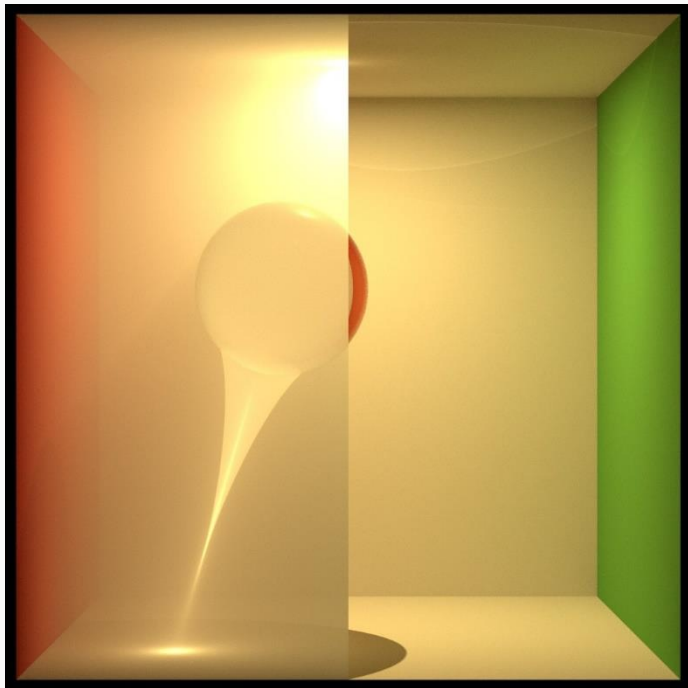
- A surface-only scene
 - Area light
 - Walls and ceiling
 - A glass ball

Motivation

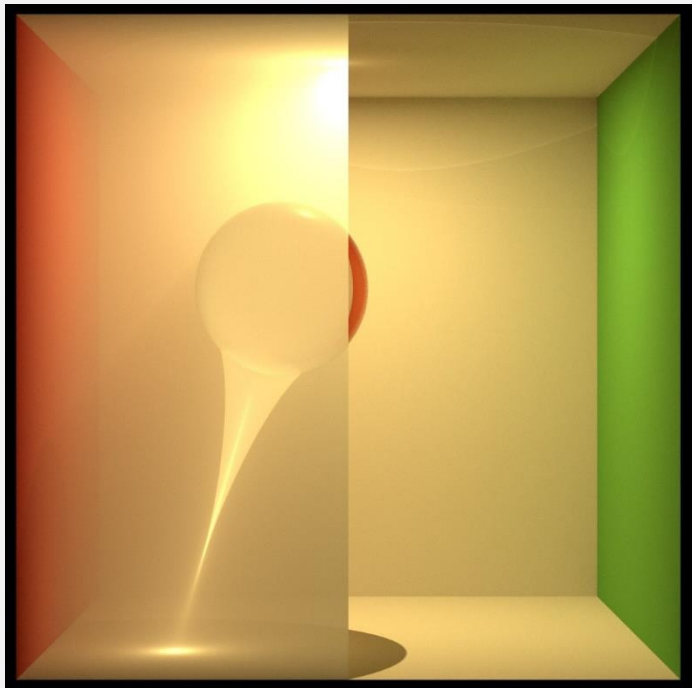


- A ~~surface-only~~ scene filled with gas
 - Area light
 - Walls and ceiling
 - A glass ball

Motivation

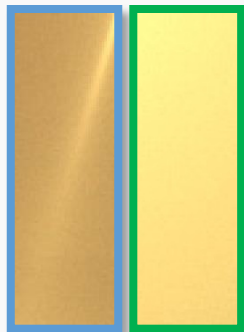
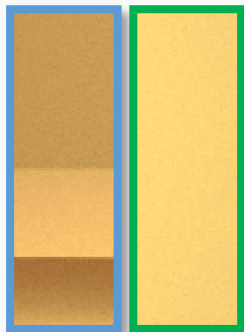
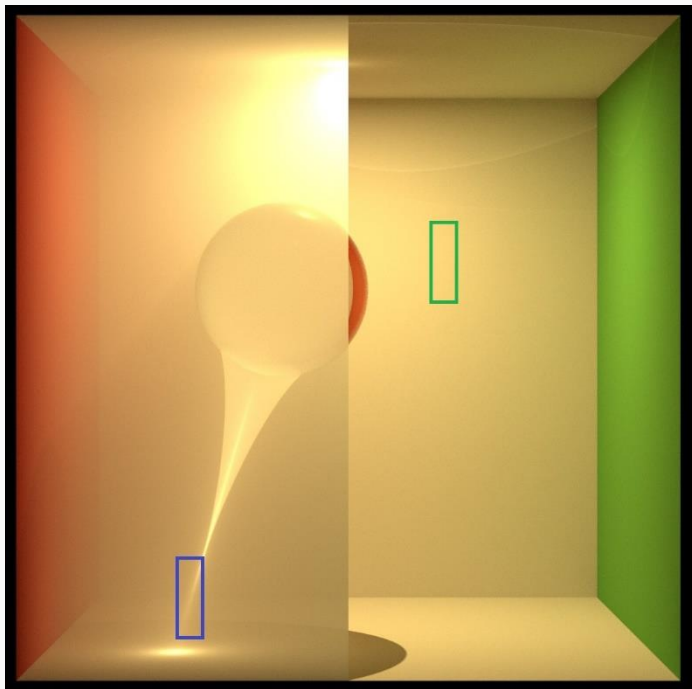


Motivation



- 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

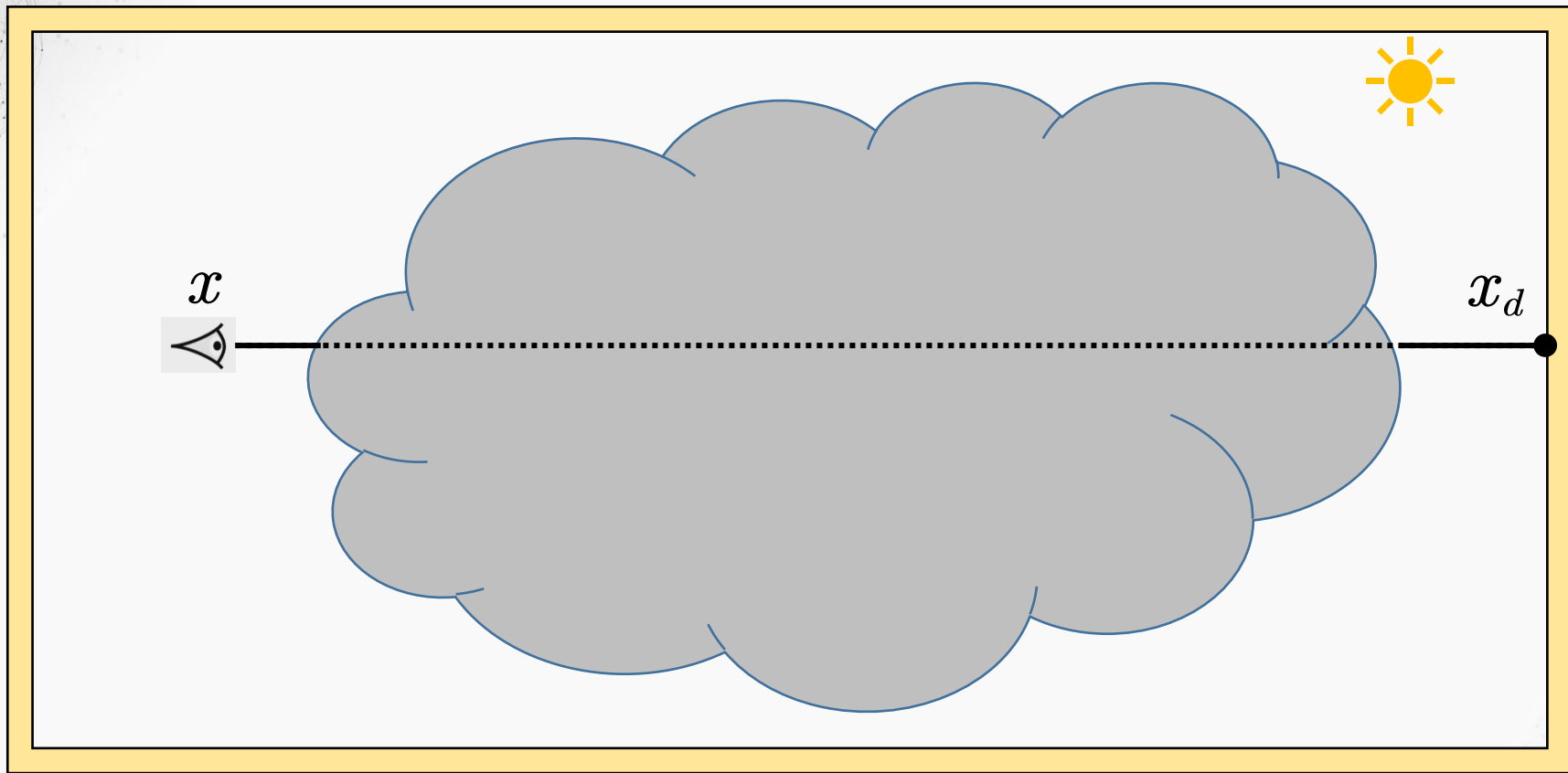
Gradient Algorithm

04

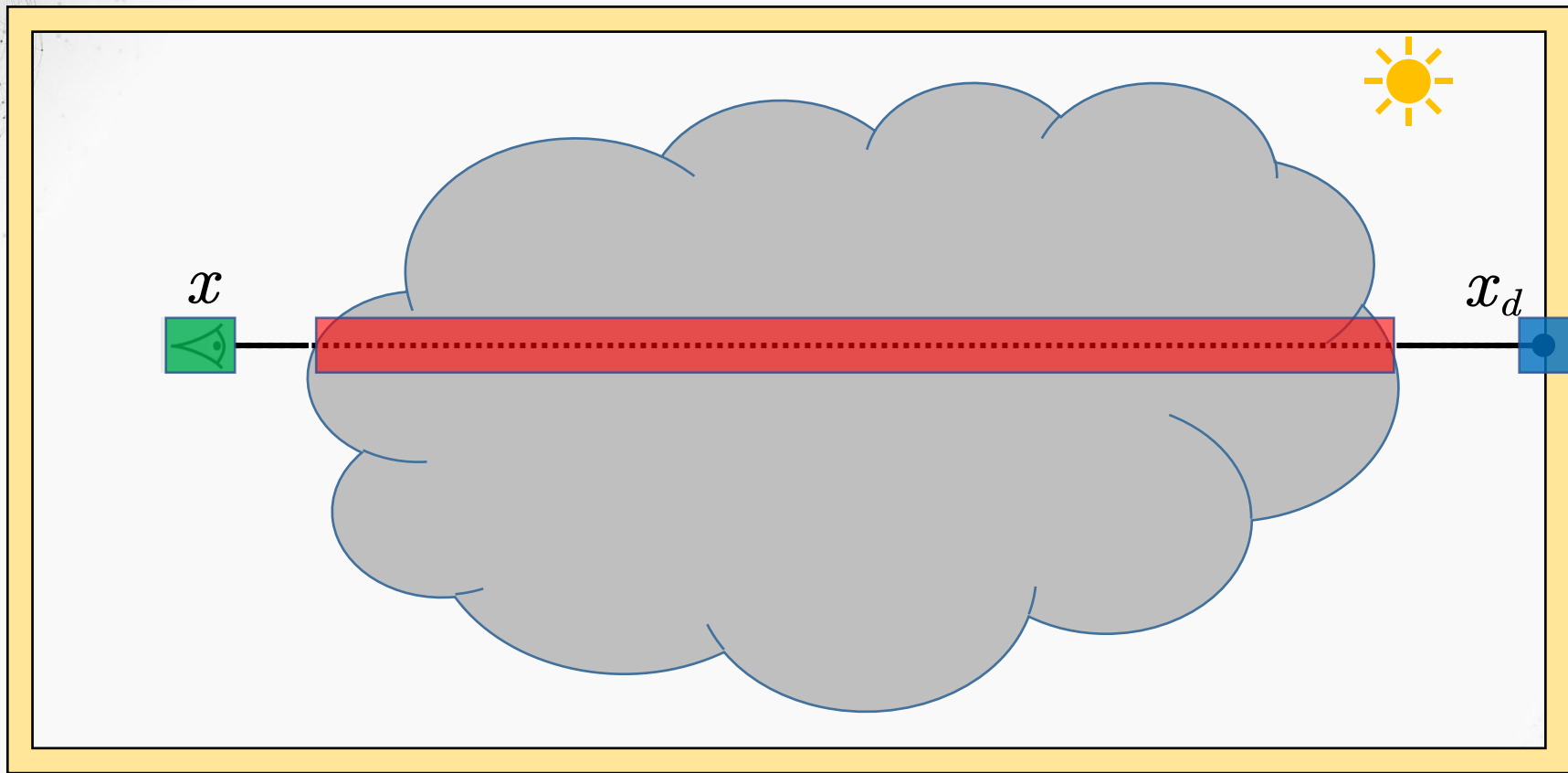
Result

05

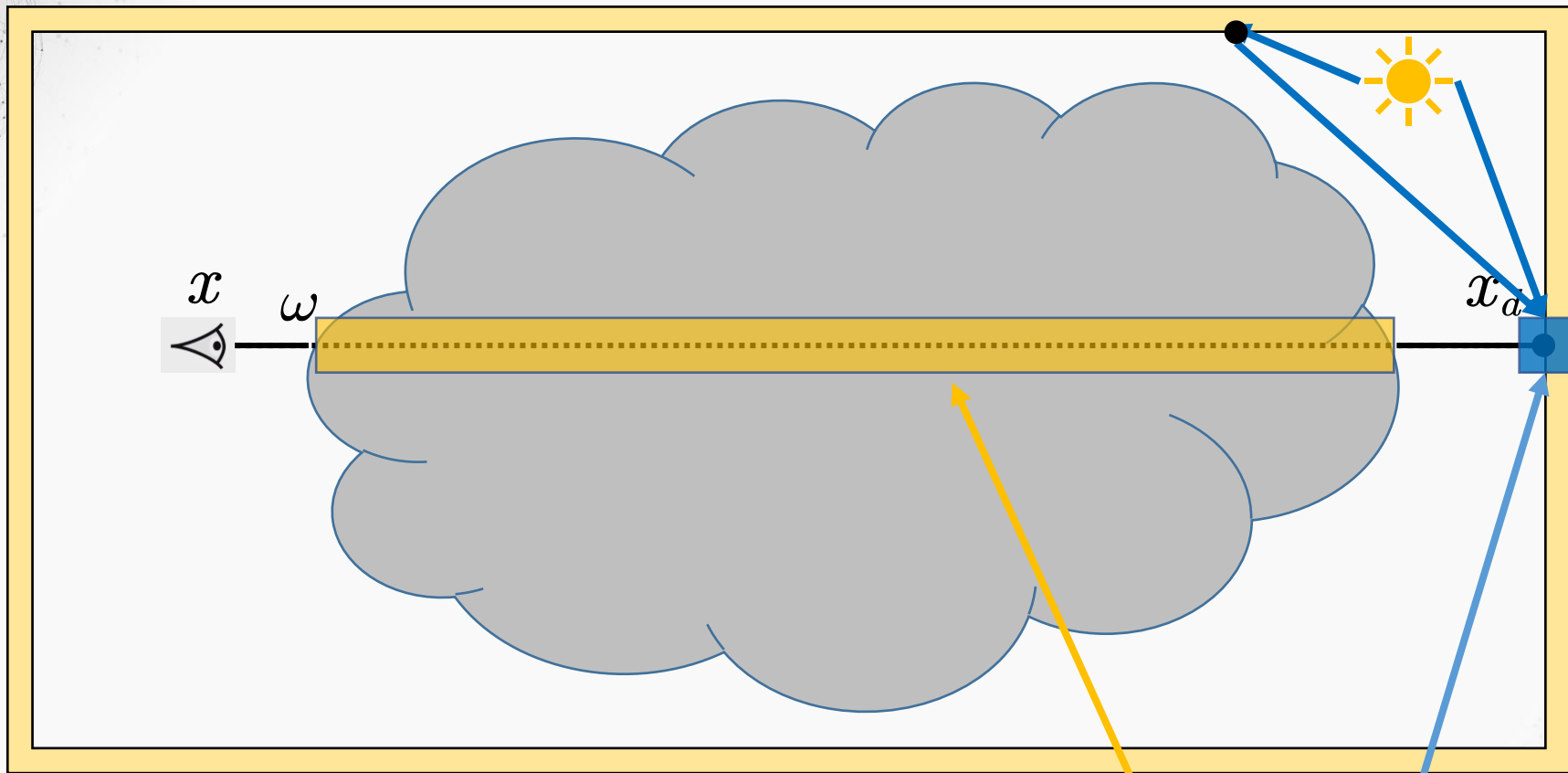
Conclusion



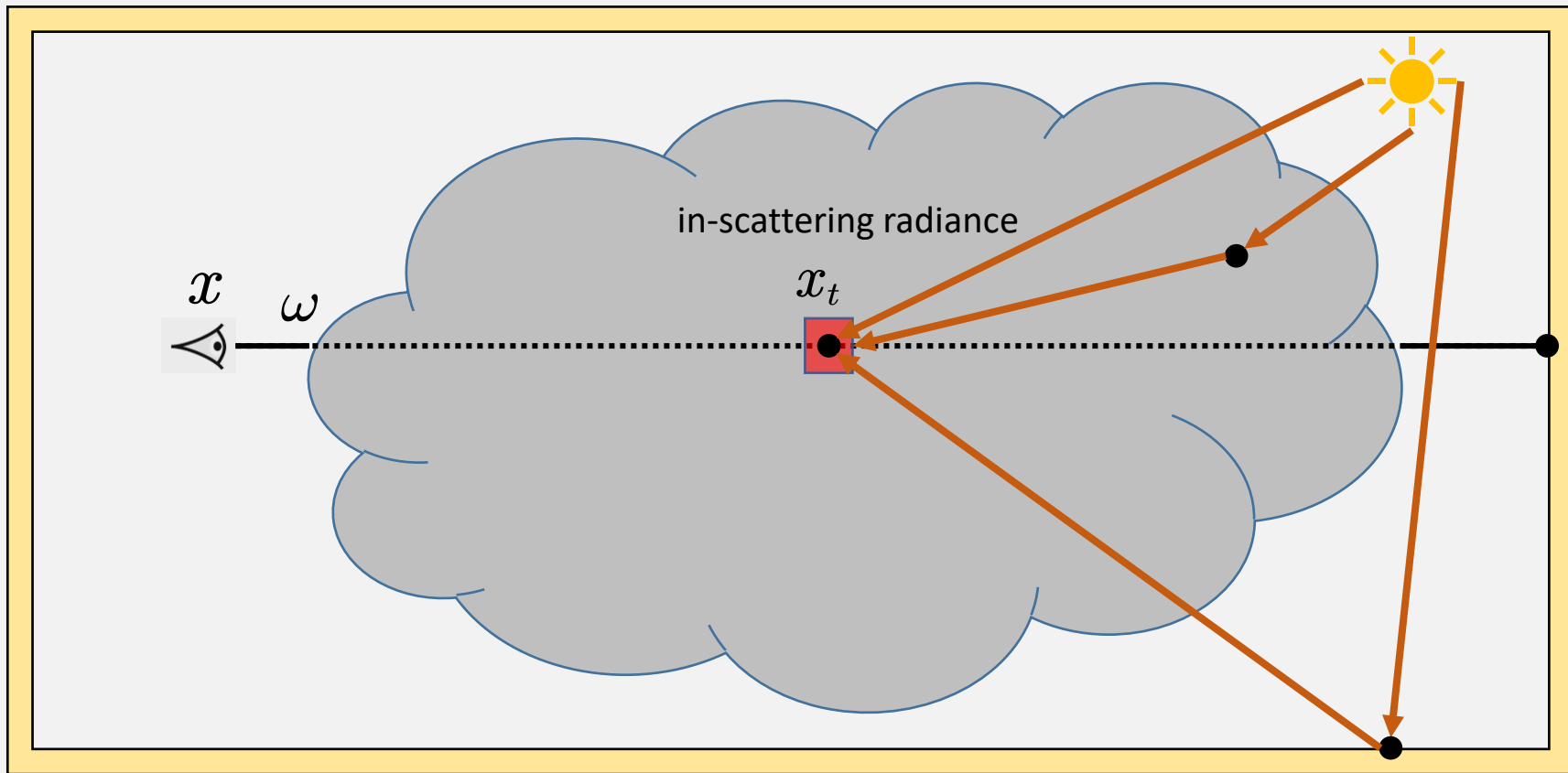
$$L(x, \omega) = \int_{x_d}^x T_r(x_t \leftrightarrow x) \sigma_s(x_t) L_s(x_t, \omega) dx_t + T_r(x_d \leftrightarrow x) L_d(x_d, \omega)$$



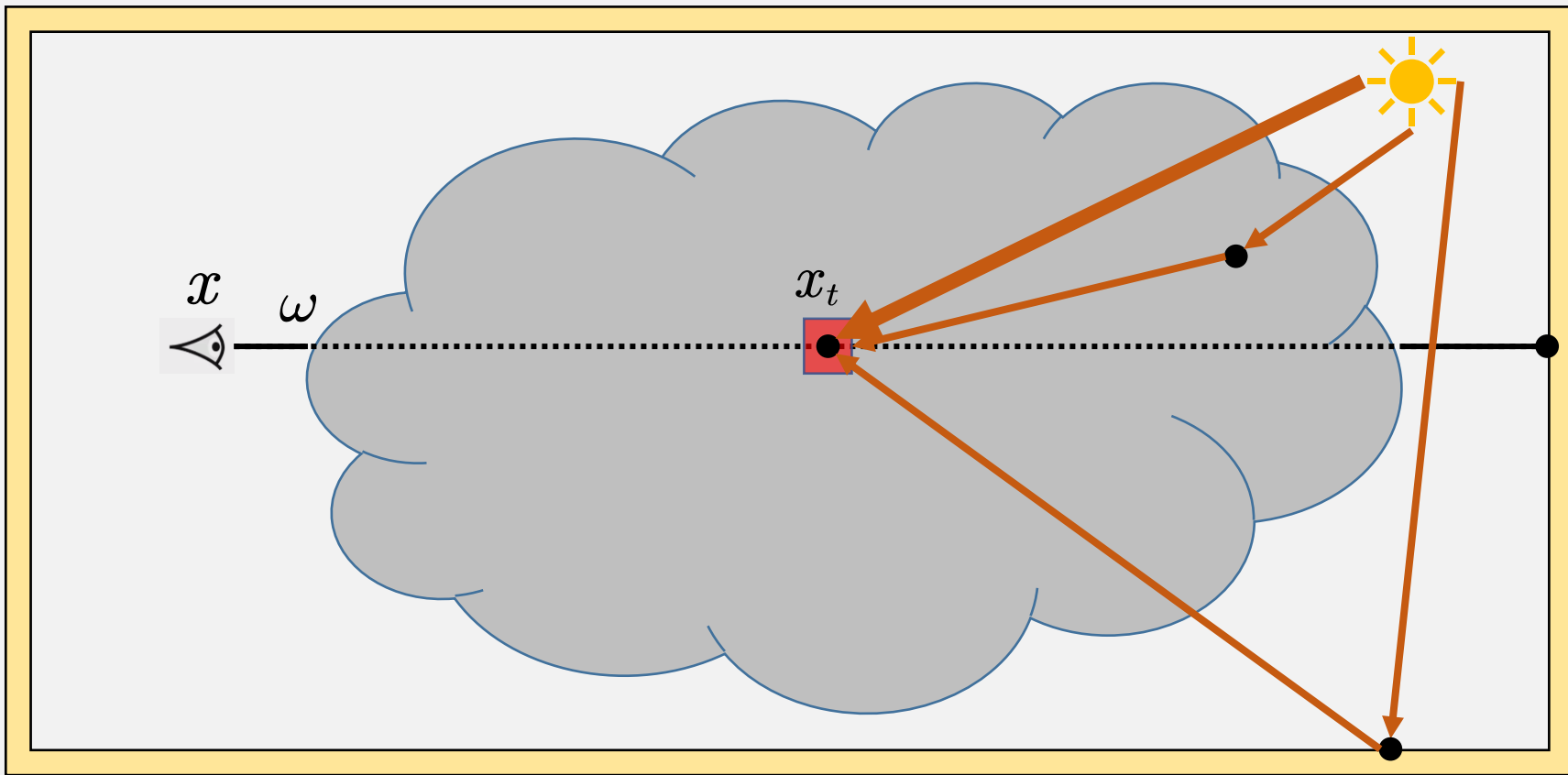
$$L(x, \omega) = \int_{x_d}^x T_r(x_t \leftrightarrow x) \sigma_s(x_t) L_s(x_t, \omega) dx_t + T_r(x_d \leftrightarrow x) L_d(x_d, \omega)$$



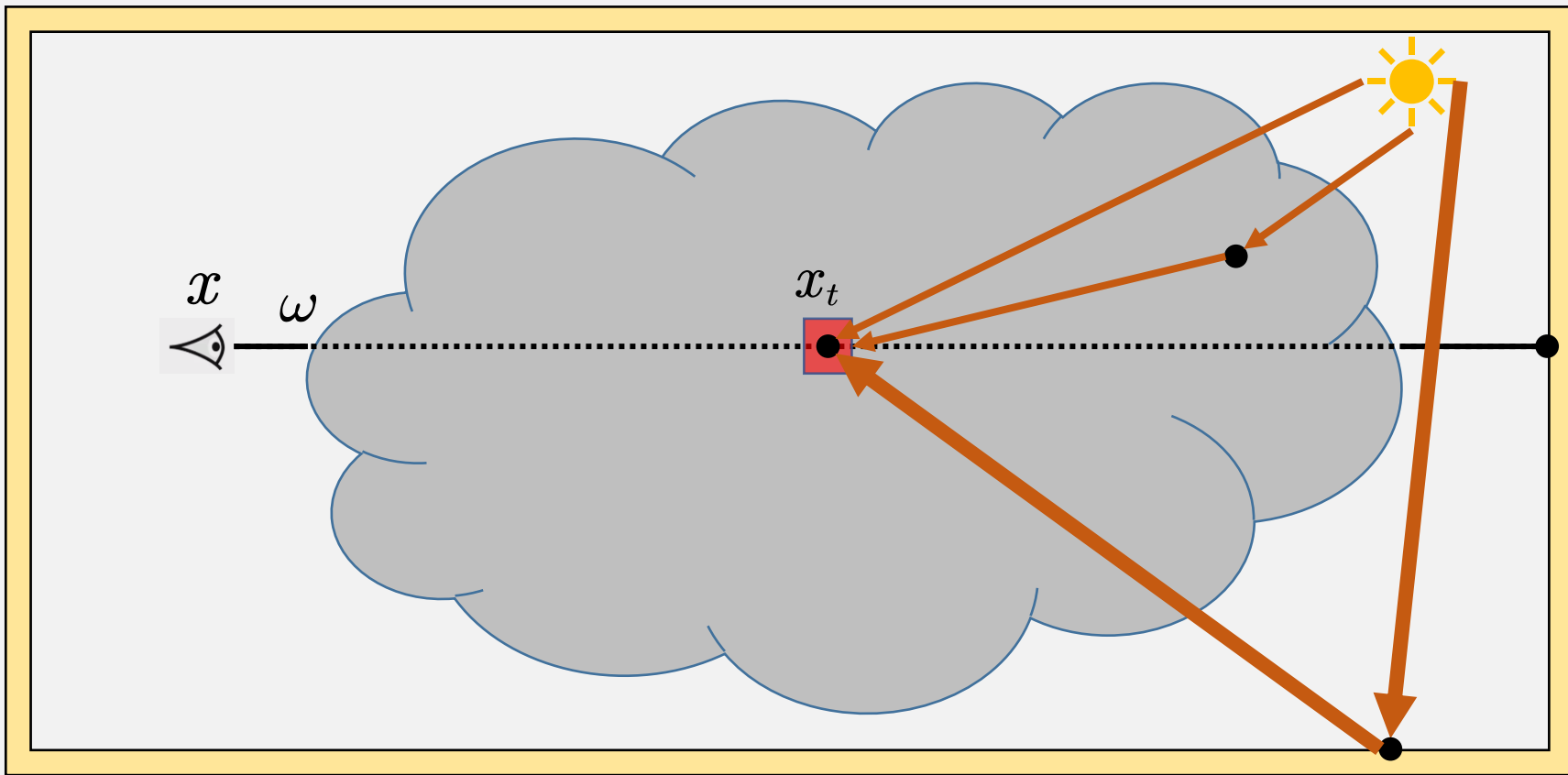
$$L(x, \omega) = \int_{x_d}^x T_r(x_t \leftrightarrow x) \sigma_s(x_t) L_s(x_t, \omega) dx_t + T_r(x_d \leftrightarrow x) L_d(x_d, \omega)$$



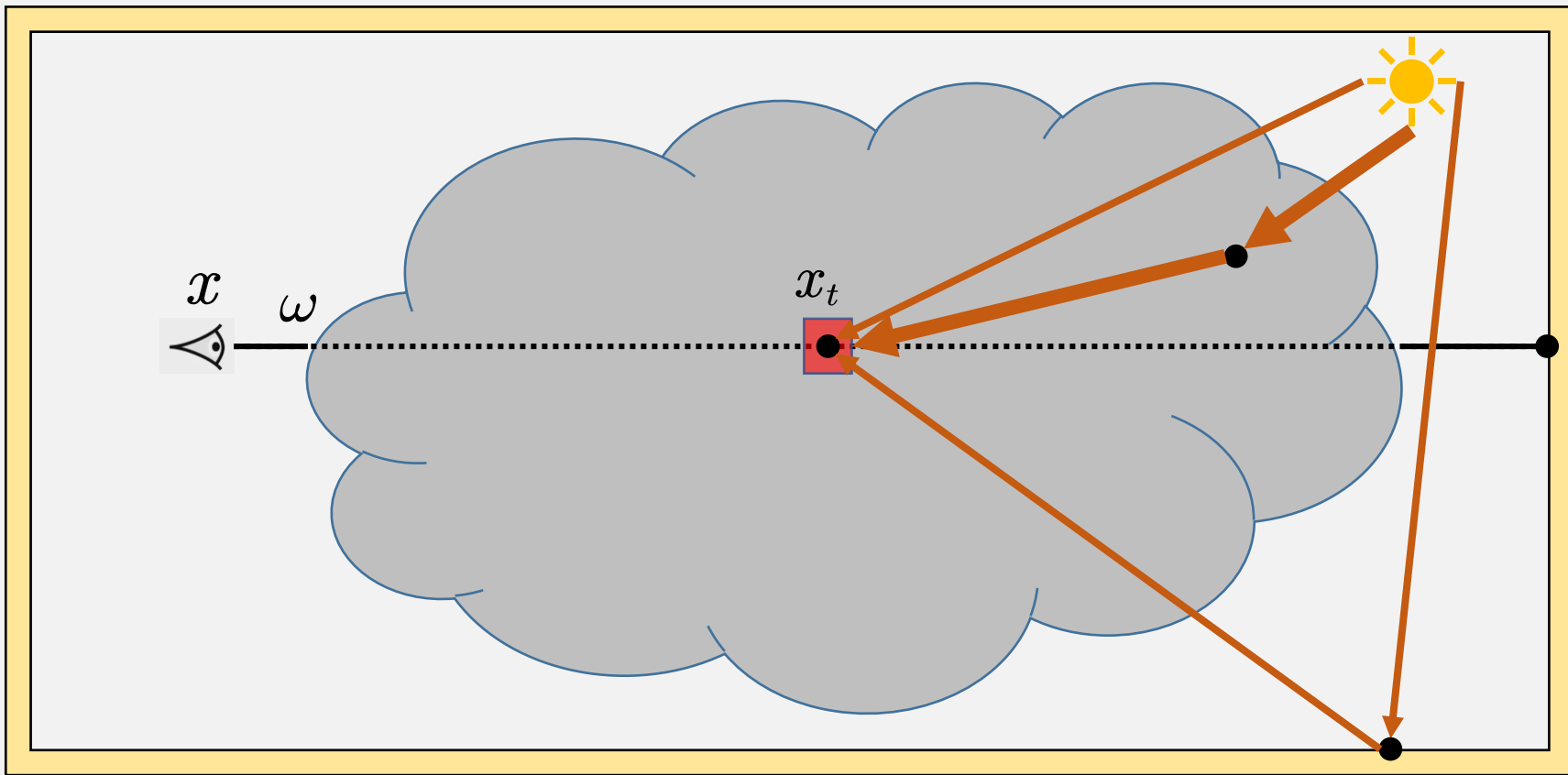
$$L(x, \omega) = \int_{x_d}^x T_r(x_t \leftrightarrow x) \sigma_s(x_t) \mathbf{L}_s(x_t, \omega) dx_t + T_r(x_d \leftrightarrow x) L_d(x_d, \omega)$$



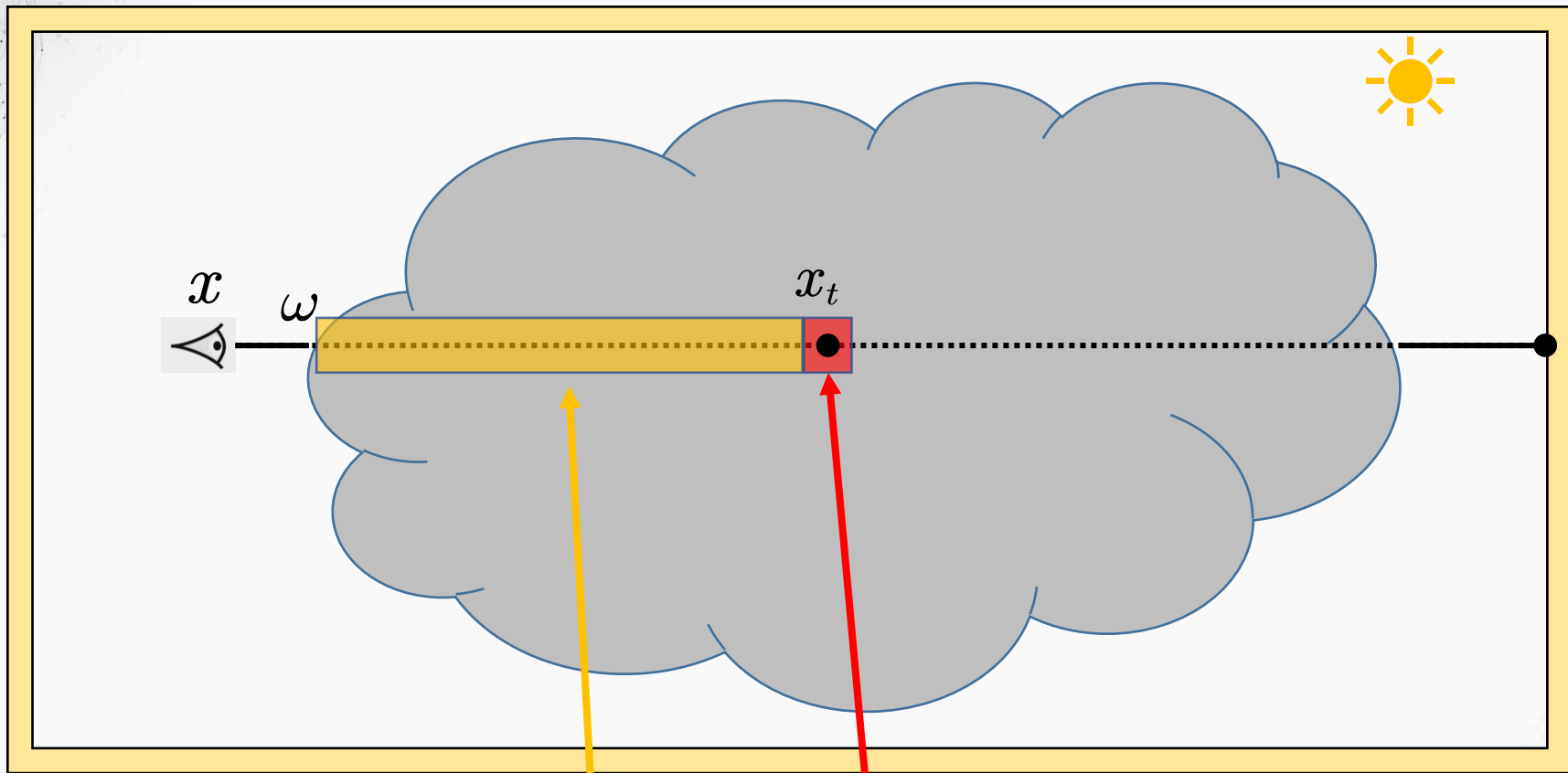
$$L(x, \omega) = \int_{x_d}^x T_r(x_t \leftrightarrow x) \sigma_s(x_t) \mathbf{L}_s(x_t, \omega) dx_t + T_r(x_d \leftrightarrow x) L_d(x_d, \omega)$$



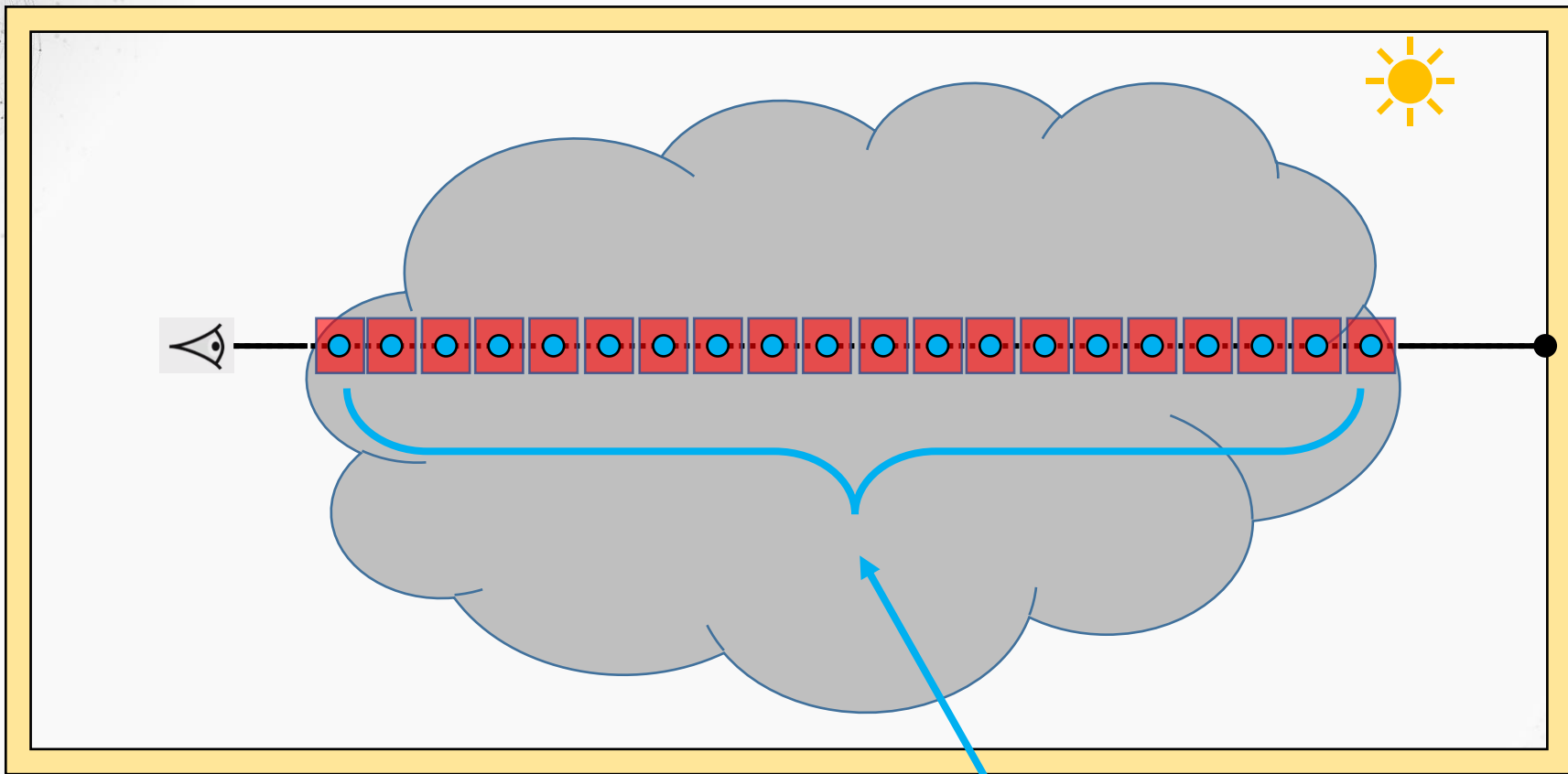
$$L(x, \omega) = \int_{x_d}^x T_r(x_t \leftrightarrow x) \sigma_s(x_t) \textcolor{red}{L_s(x_t, \omega)} dx_t + T_r(x_d \leftrightarrow x) L_d(x_d, \omega)$$



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




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



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

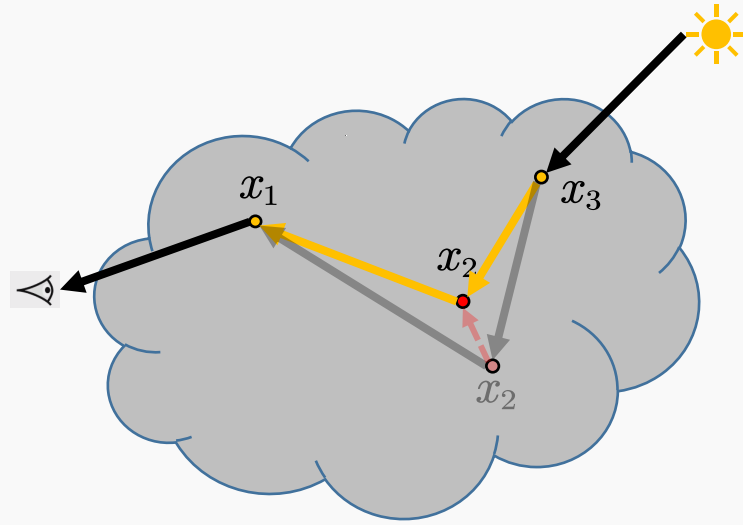


Previous work

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

Goals

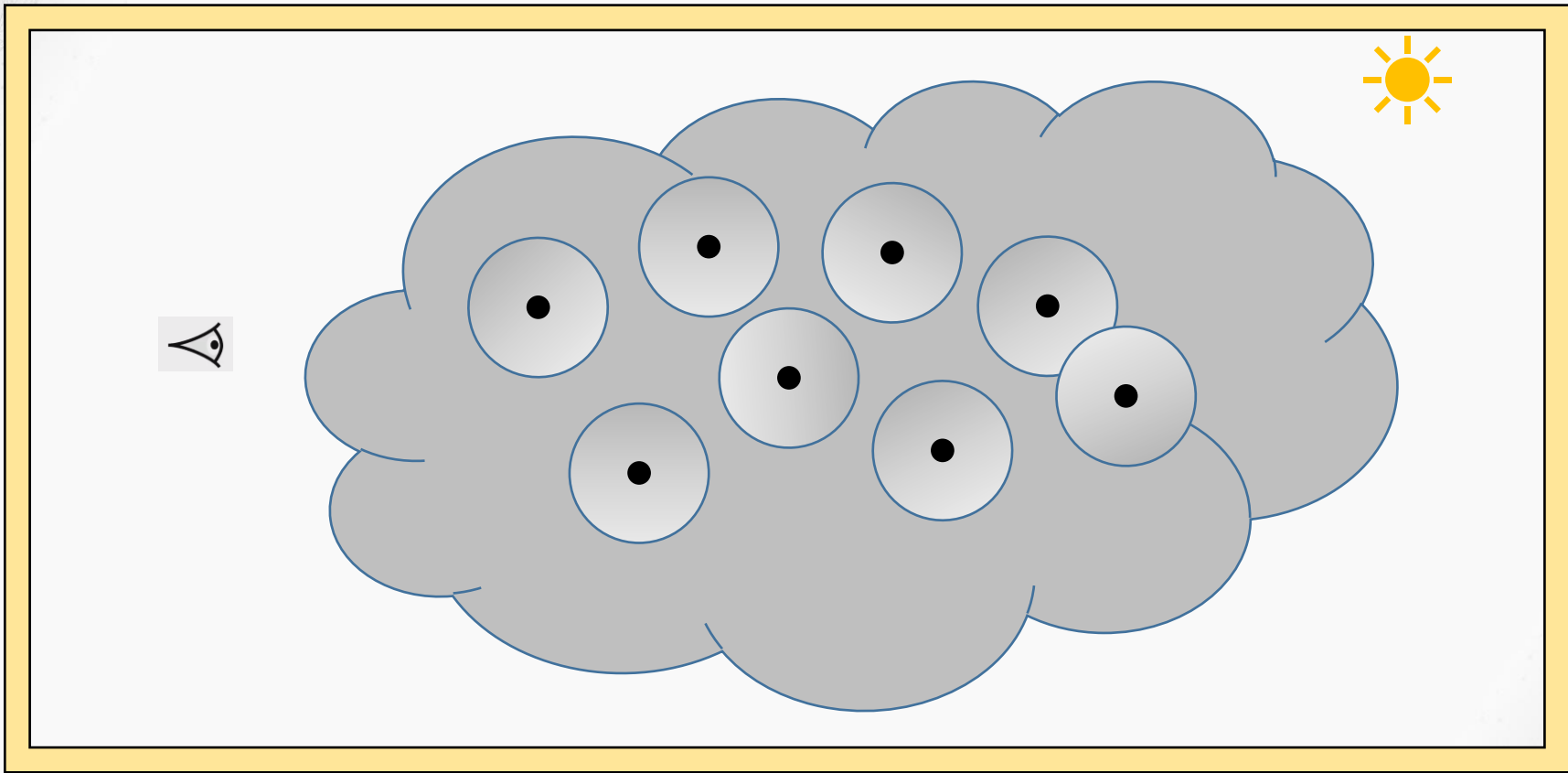
- Gradient Algorithm
- Gradient mutation

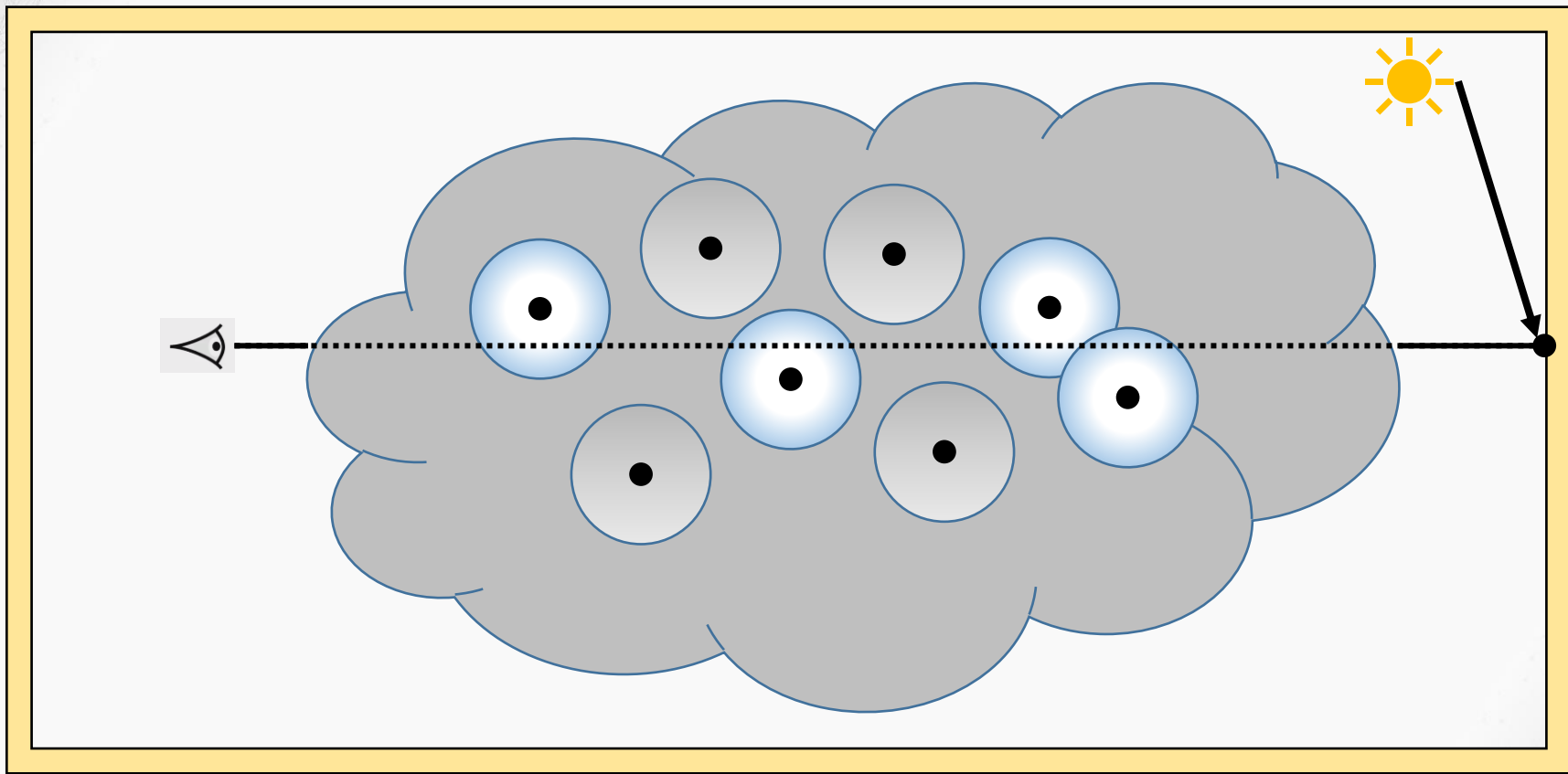


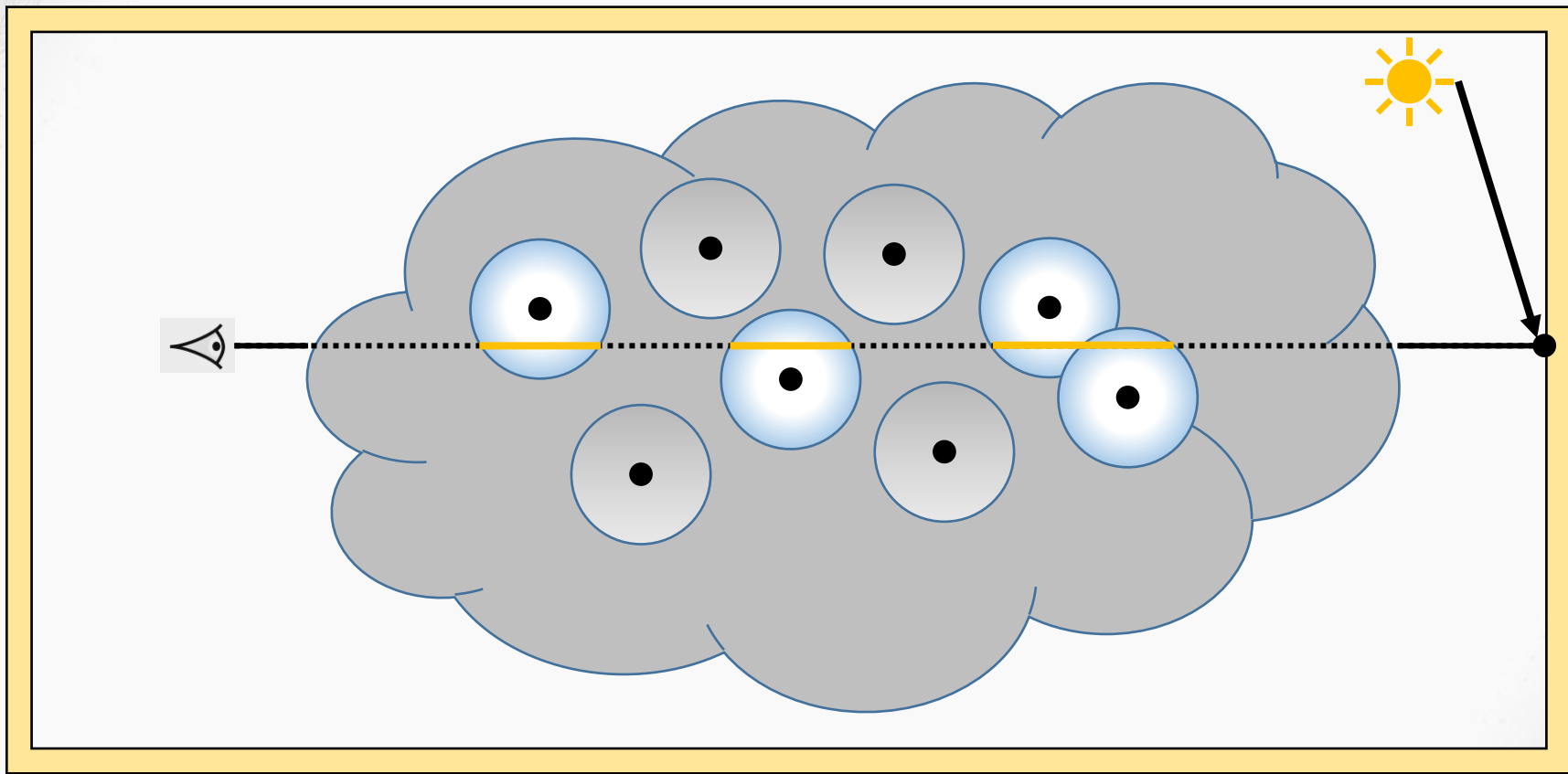
A decorative geometric pattern in the top-left corner, consisting of a network of interconnected nodes and lines forming a spherical shape.

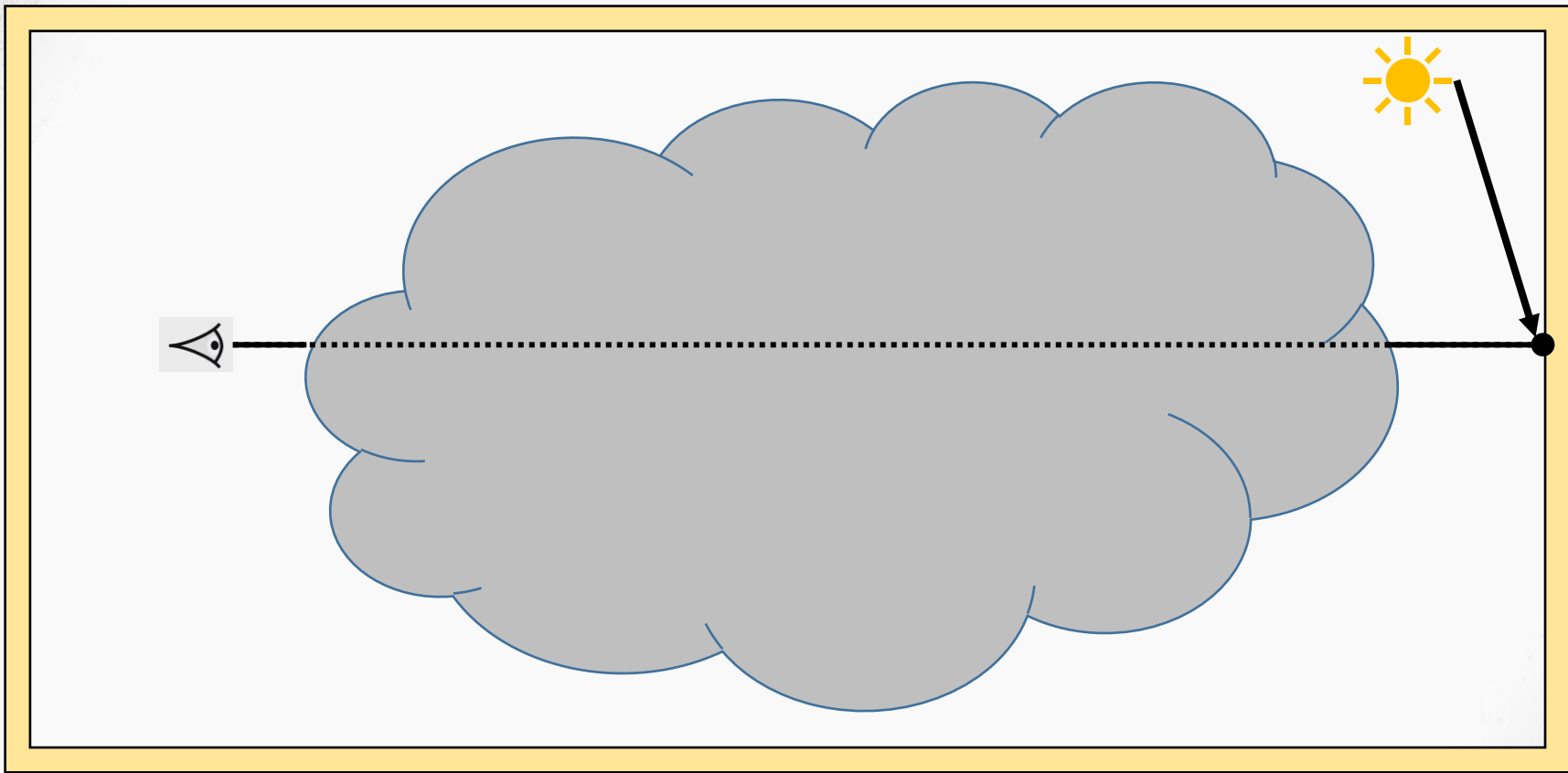
Related Work

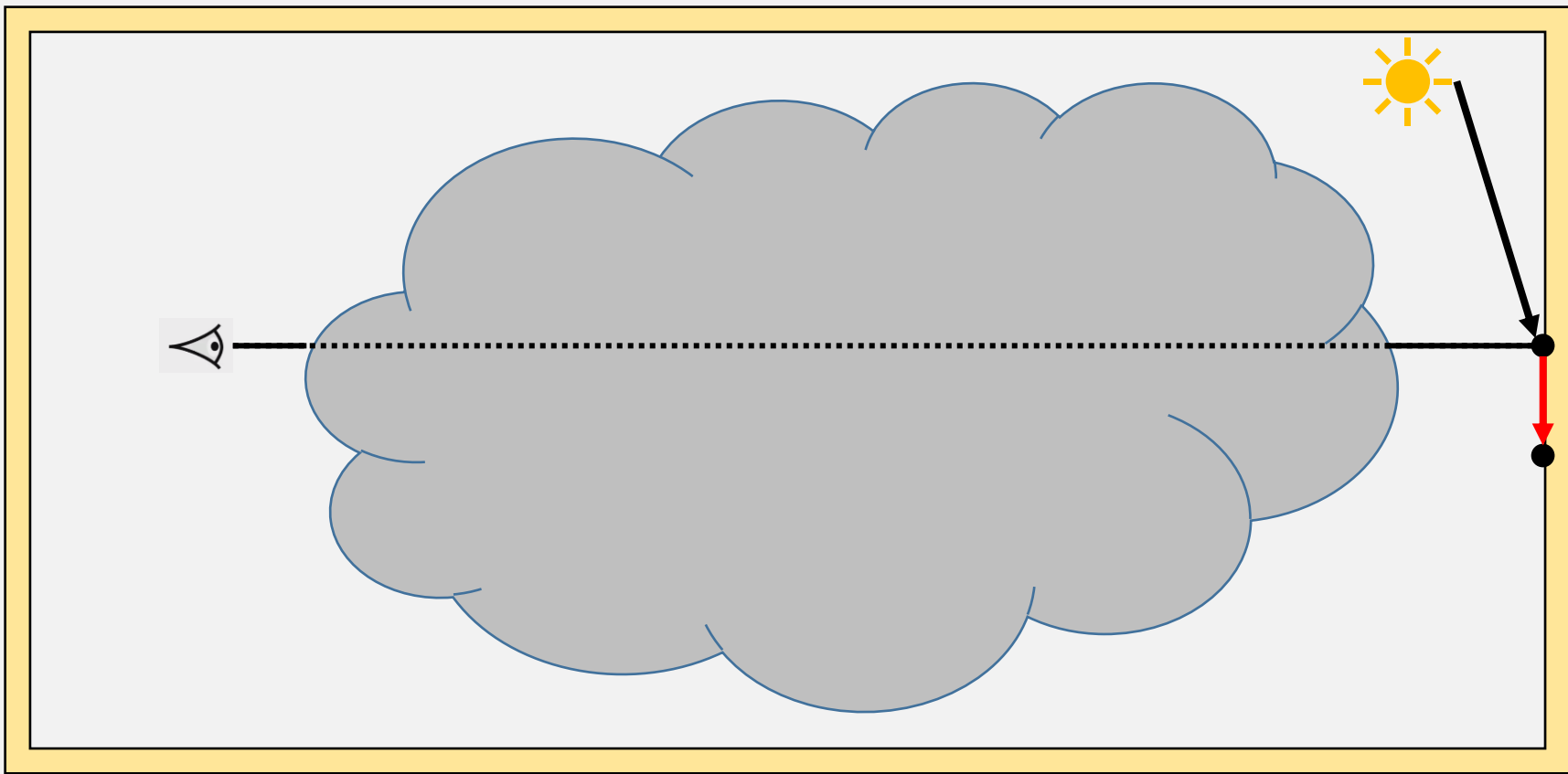
- Radiance cache for participating media
- 
- A decorative geometric pattern in the bottom-right corner, consisting of a network of interconnected nodes and lines forming a spherical shape.

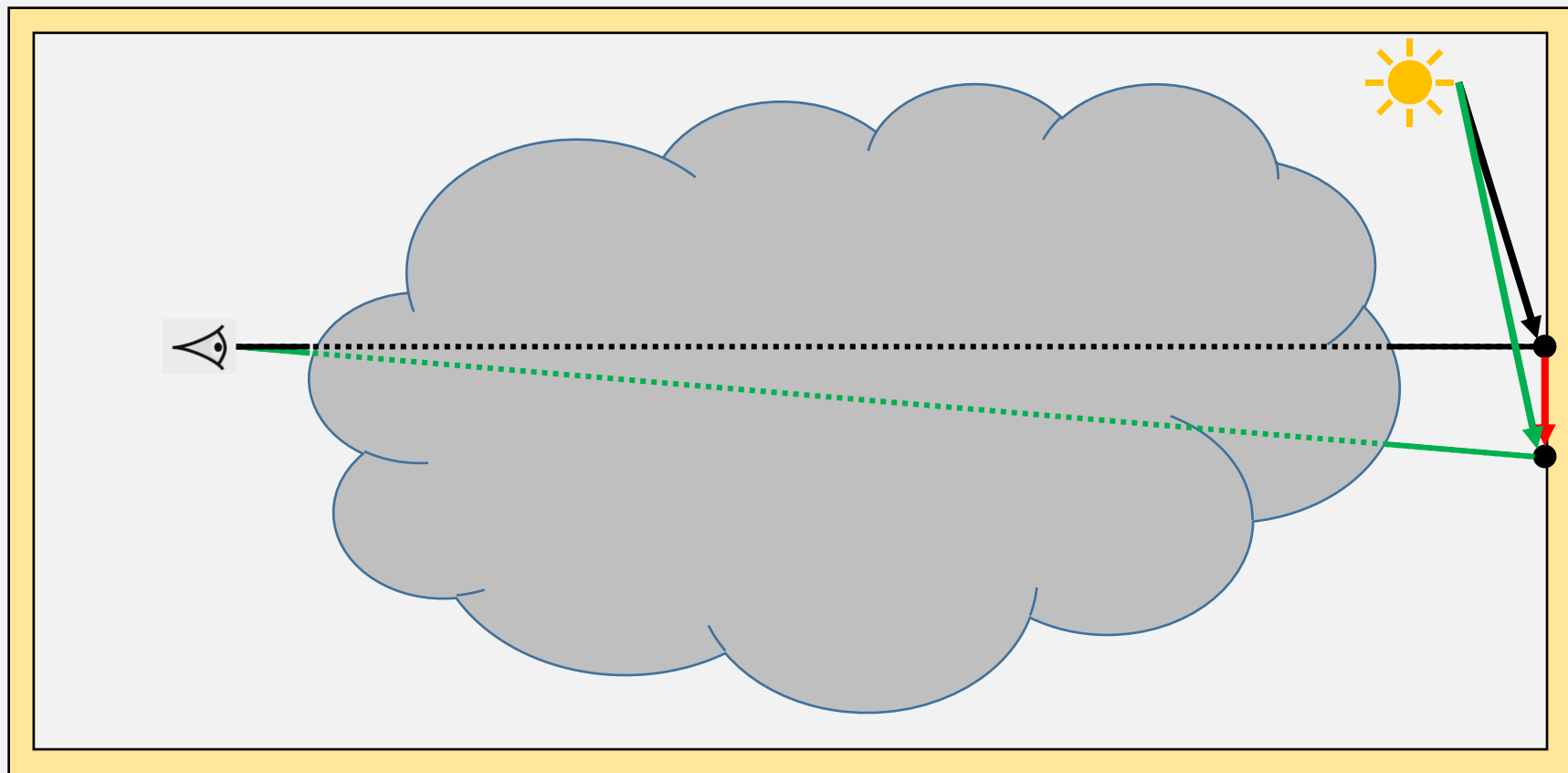


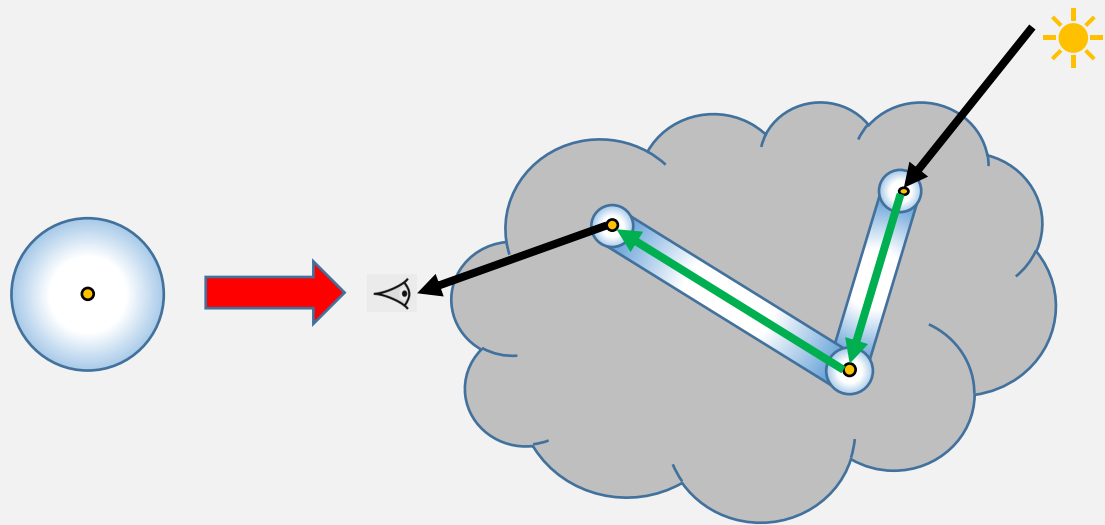


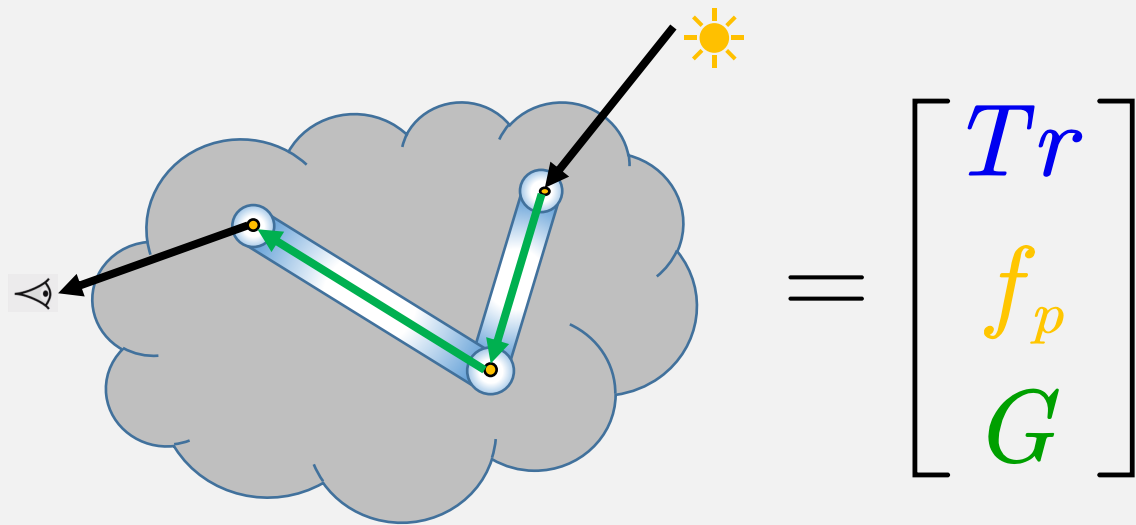


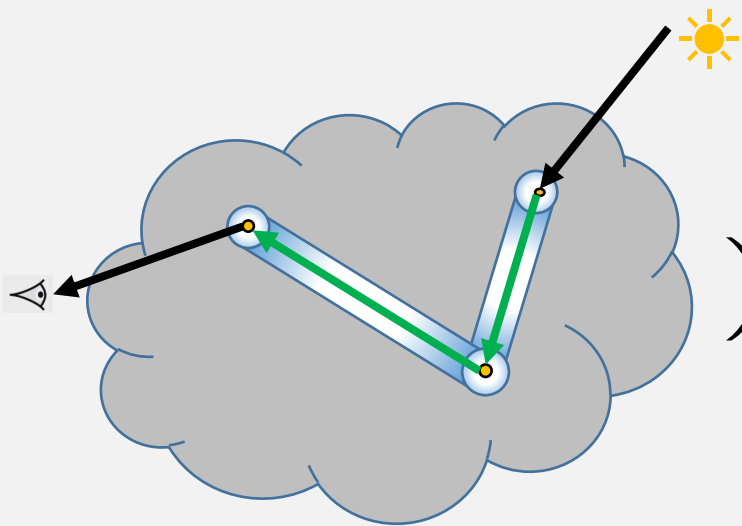










$$\nabla L(\text{telescope}, \text{robot arm}, \text{sun}) = \begin{bmatrix} \nabla T_r \\ \nabla f_p \\ \nabla G \end{bmatrix}$$


The diagram illustrates a robotic system within a cloud environment. A grey, fluffy cloud contains a blue robot arm with two joints and a green link. A black arrow points from the left joint of the arm to a telescope icon. Another black arrow points from the right joint of the arm to a sun icon. The equation shows the gradient of the loss function L with respect to these three elements, resulting in a vector of three gradients: ∇T_r (blue), ∇f_p (orange), and ∇G (green).



01

Motivation

02

Background

03

Gradient Algorithm

04

Result

05

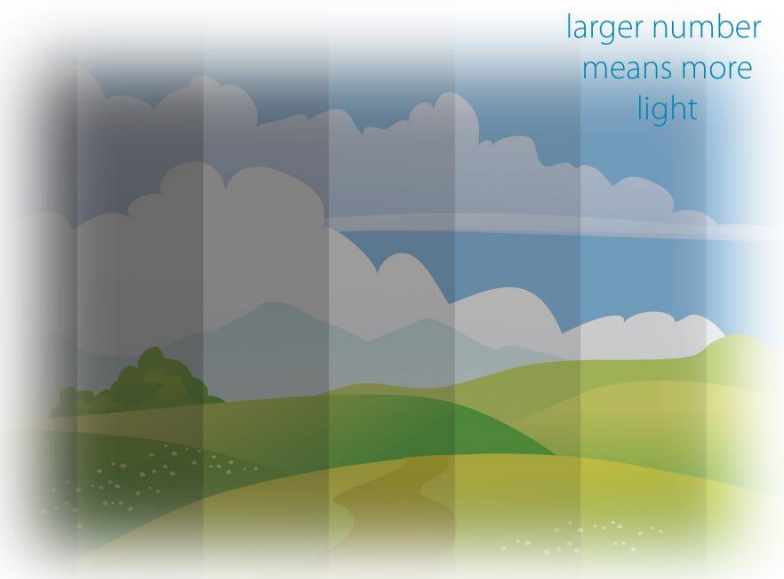
Conclusion

Transmittance

0 .1 .2 .3 .4 .5 .6 .7 .8 .9 1.0



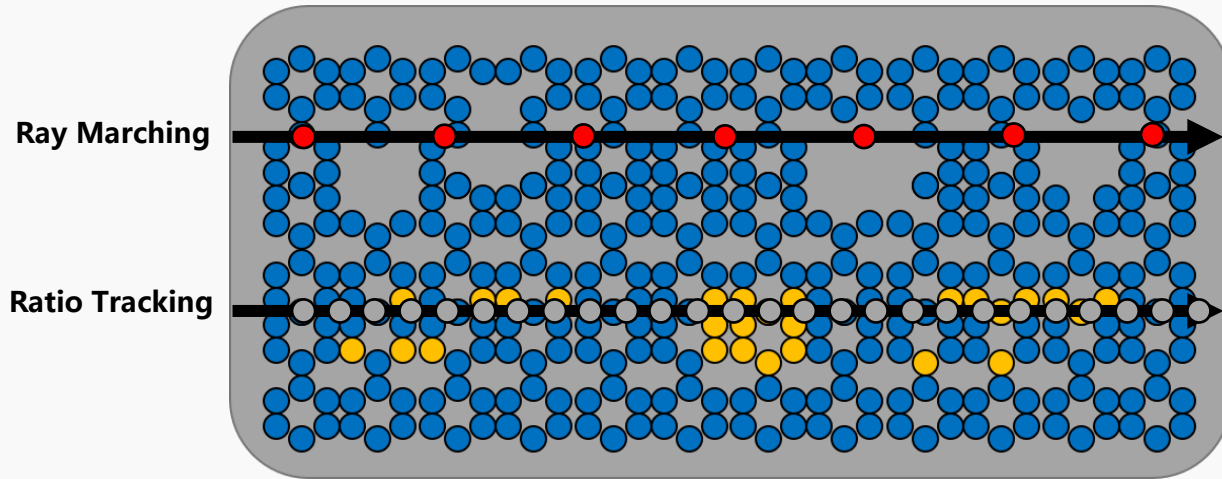
larger number
means more
light



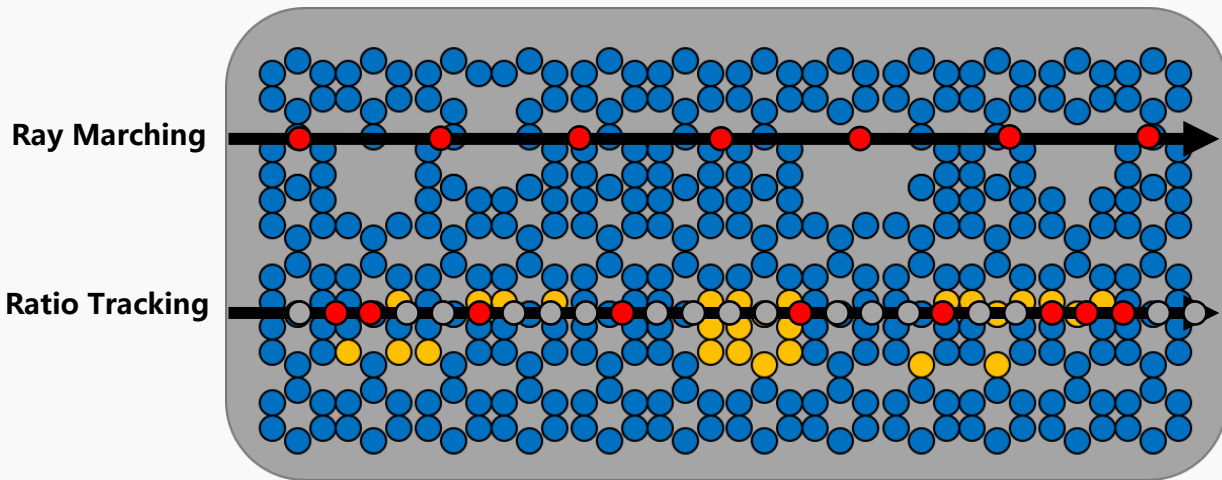
<https://bayviewwindows.ca>

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

Ratio Tracking

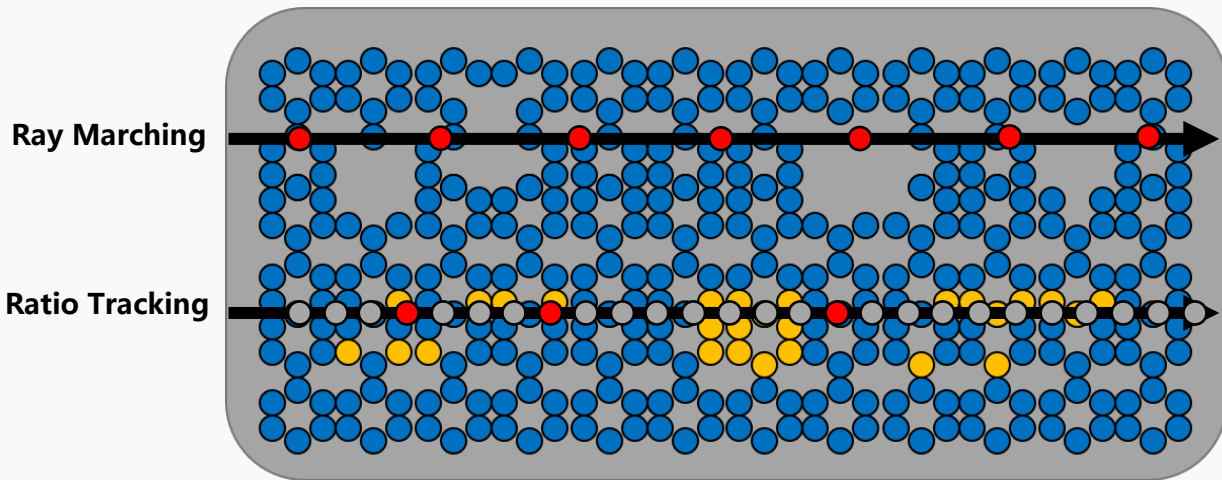


Ratio Tracking



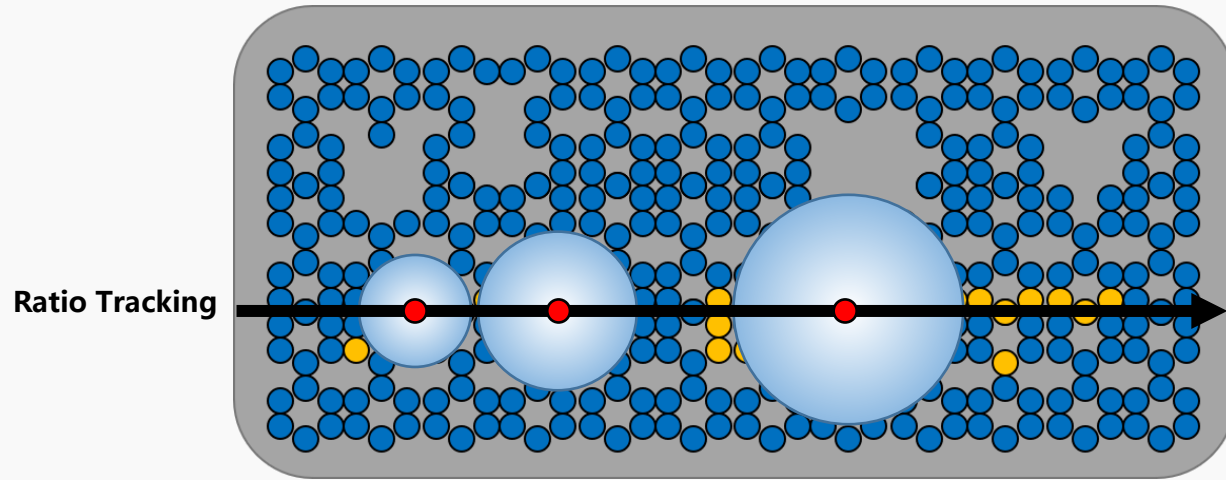
$$T_r(x_i \leftrightarrow x_{i+1}) = \prod_{j=0}^k \left(1 - \frac{\sigma_t(\mathbf{x}_j)}{\bar{\sigma}}\right)$$

Ratio Tracking



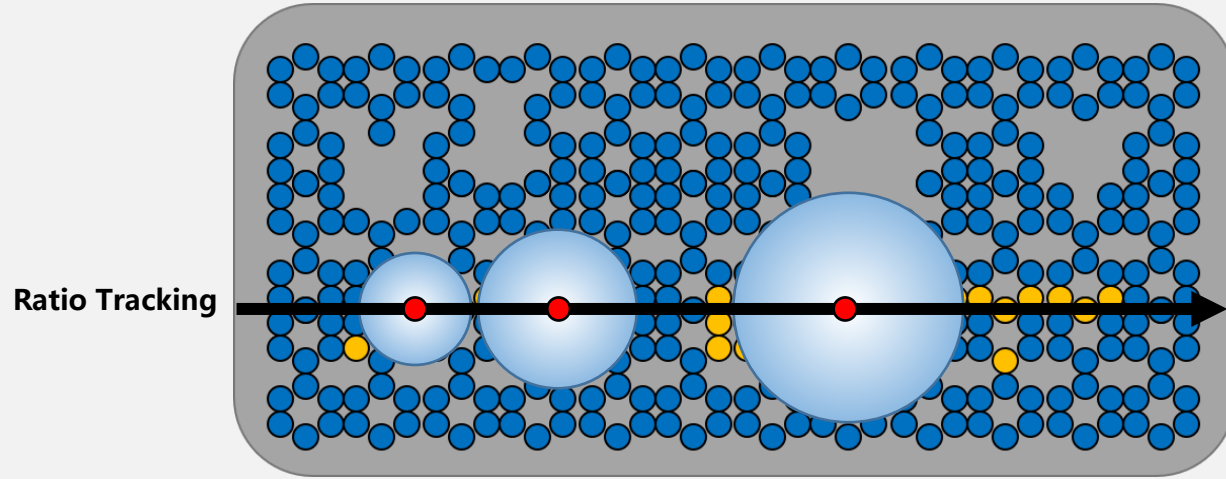
$$T_r(x_i \leftrightarrow x_{i+1}) = \prod_{j=0}^k \left(1 - \frac{\sigma_t(\mathbf{x}_j)}{\bar{\sigma}}\right)$$

Gradient of Transmittance



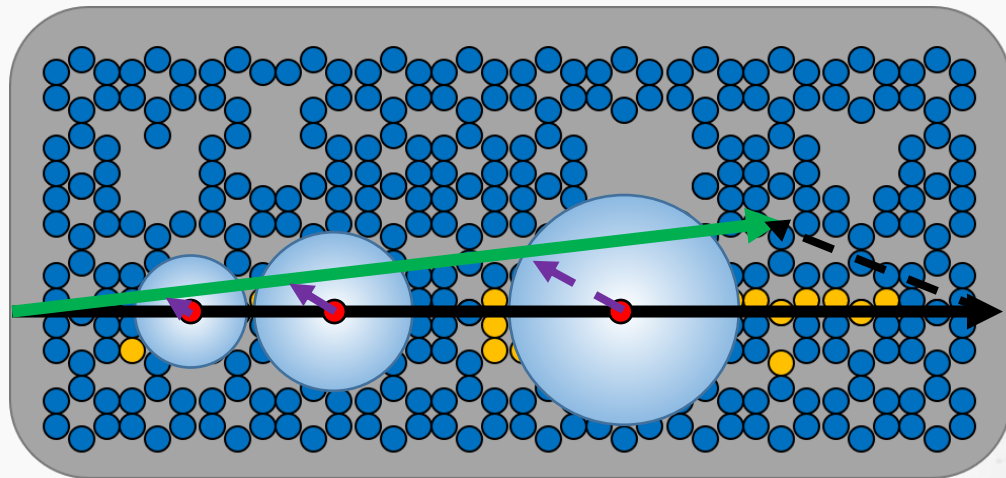
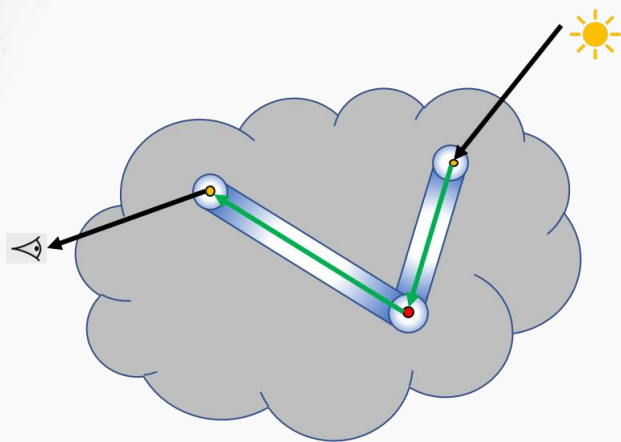
$$T_r \left(\text{blue arrow} \right) = \sum \left(\text{red dot} \right)$$

Gradient of Transmittance



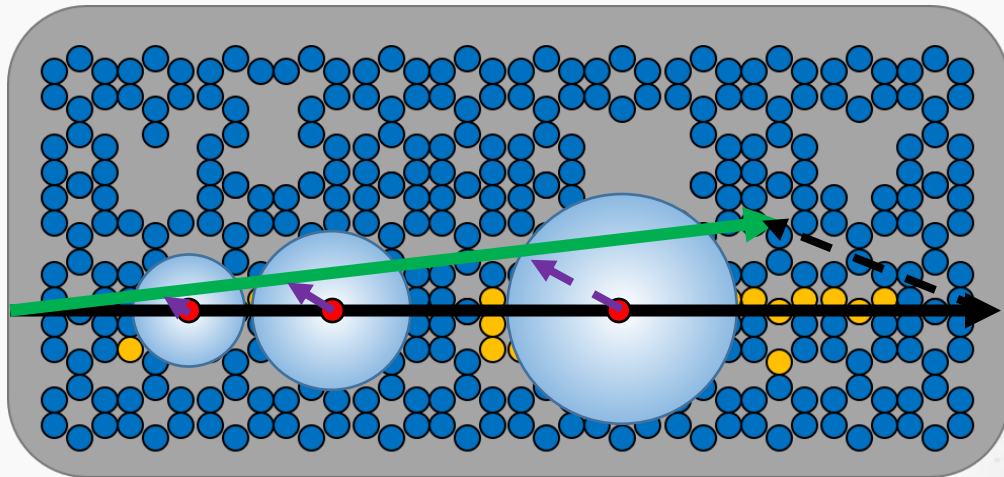
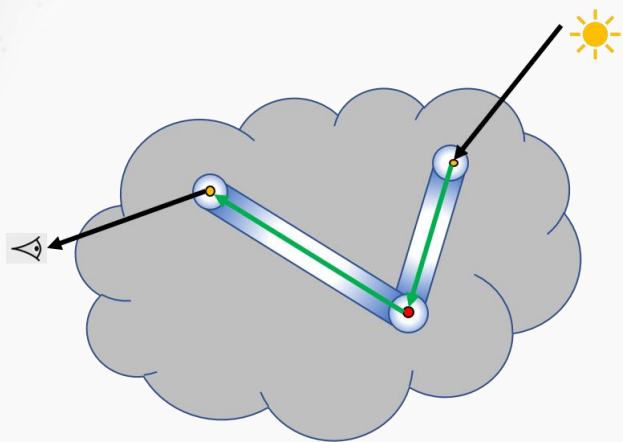
$$\nabla T_r \left(\text{blue arrow} \right) = \sum \left(\text{blue circle with red dot} \right)$$

Gradient of Transmittance



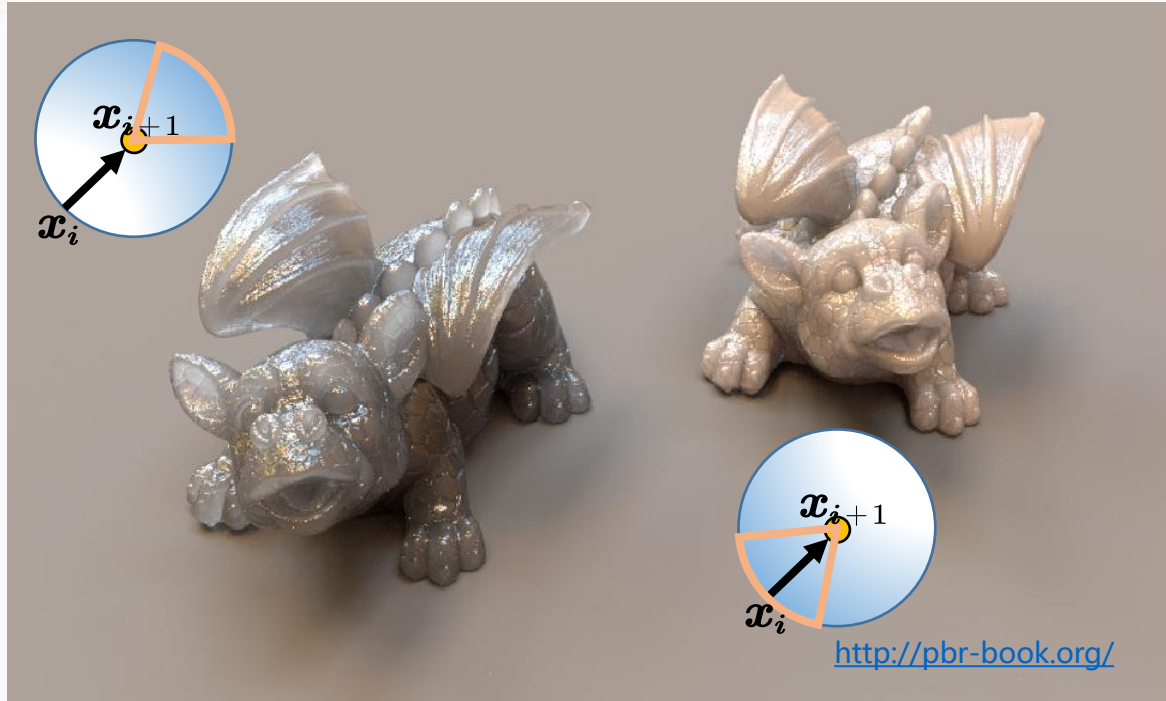
$$\nabla(T_r(x_i \leftrightarrow x_{i+1})) = -T_r(x_i \leftrightarrow x_{i+1}) \left(\sum_{j=0}^n \frac{\frac{c_j}{c_n} \nabla \sigma_t(\mathbf{x}_j)}{\bar{\sigma} - \sigma_t(\mathbf{x}_j)} + \sigma_{avg} \frac{\overrightarrow{x_i x_{i+1}}}{\left\| \overrightarrow{x_i x_{i+1}} \right\|} \right)$$

Gradient of Transmittance



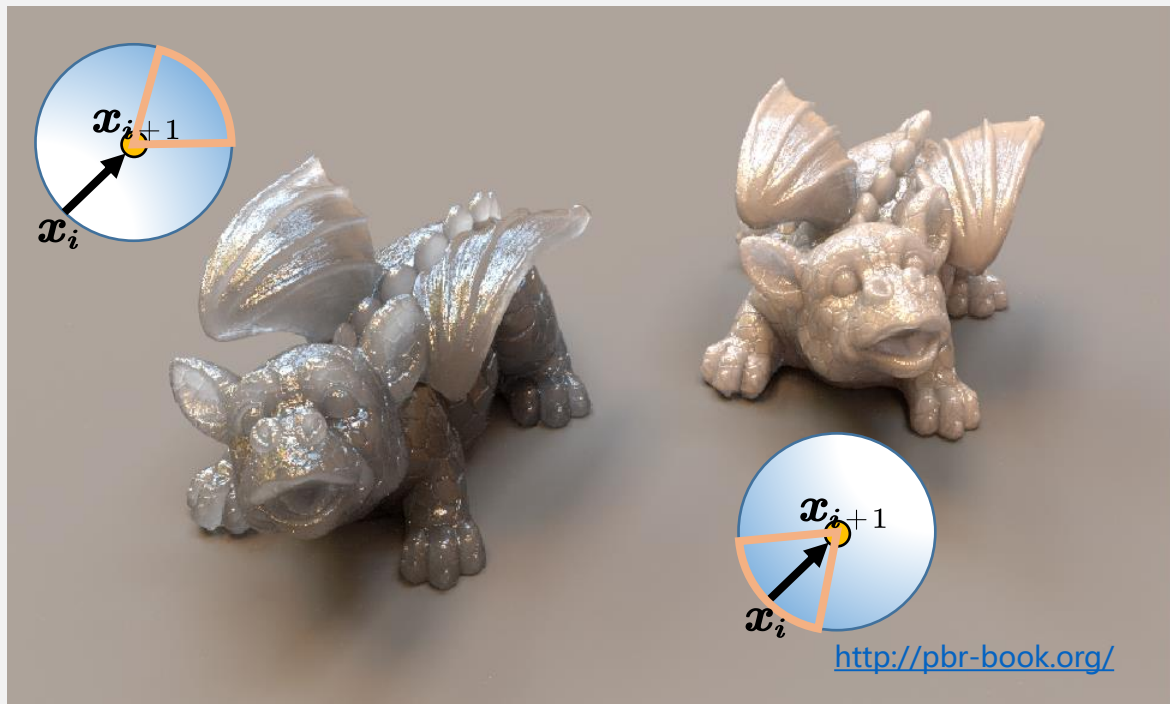
$$T_r(\text{green arrow}) \approx T_r(\text{black arrow}) + \langle \nabla T_r(\text{blue arrow}), V(\text{dashed arrow}) \rangle$$

Phase function

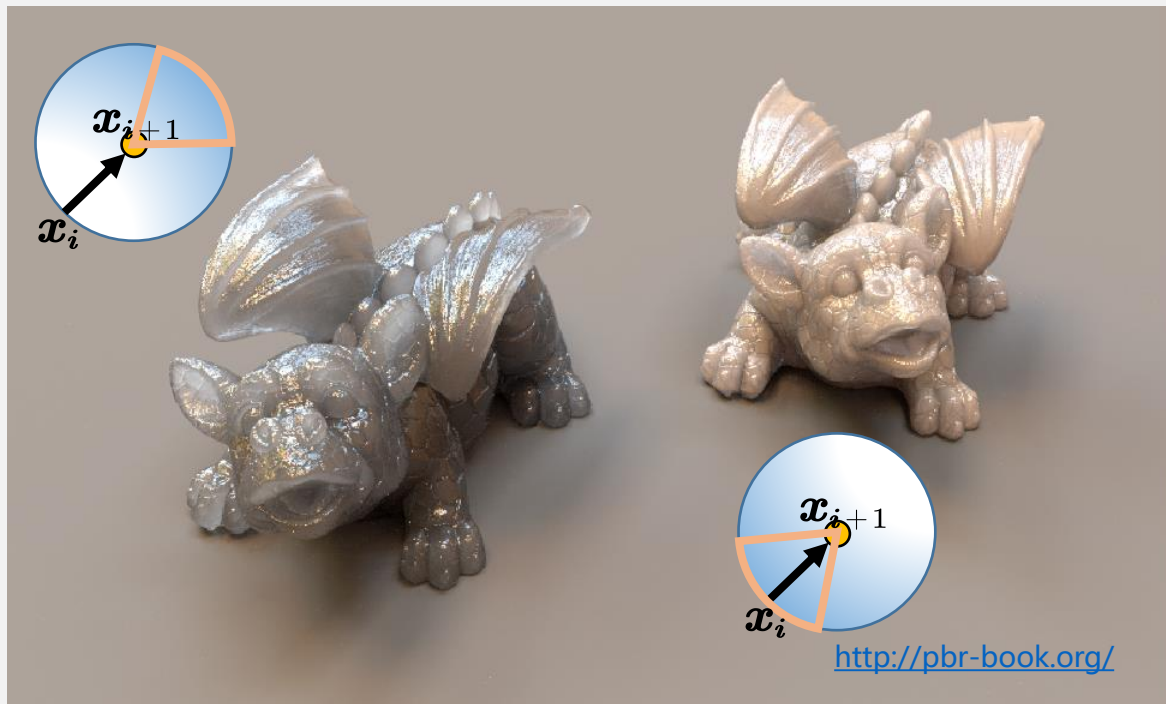


Phase function

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



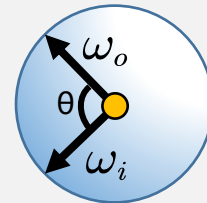
Phase function



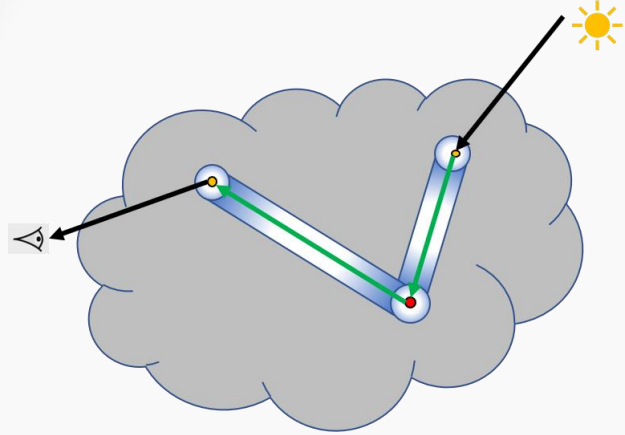
A Phase function describes this angular distribution of scattered radiance at a given point.

Henye-Greenstein phase function:

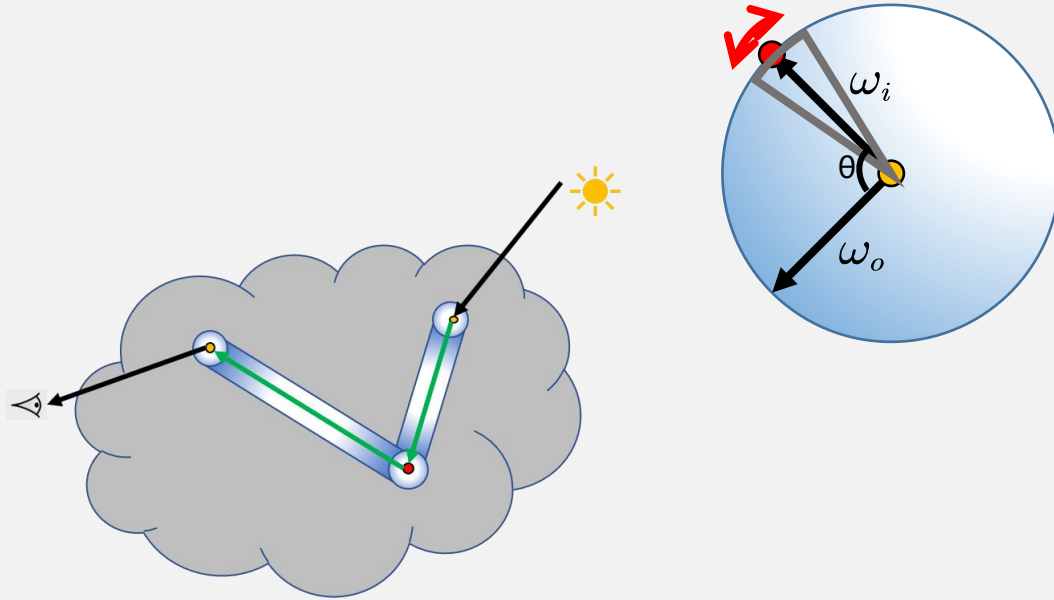
$$f_p = \frac{1}{4\pi} \frac{1 - g^2}{(1 + g^2 - 2g(\cos \theta))^{3/2}}$$



Gradient of Phase function

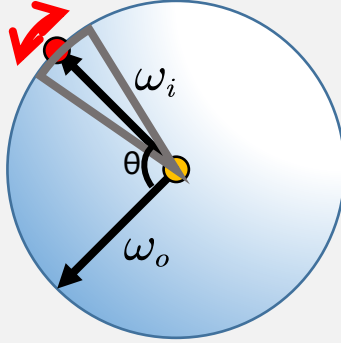
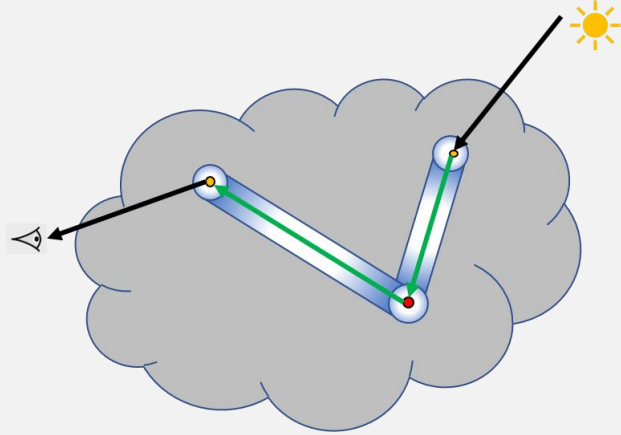


Gradient of Phase function

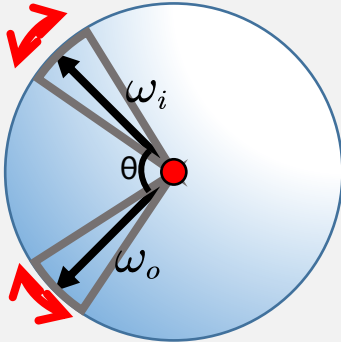


$$\nabla \cos \theta(\omega_i) = \frac{\hat{\omega}_o}{|\omega_i|} - \cos \theta \frac{\omega_i}{|\omega_i|^2}$$

Gradient of Phase function

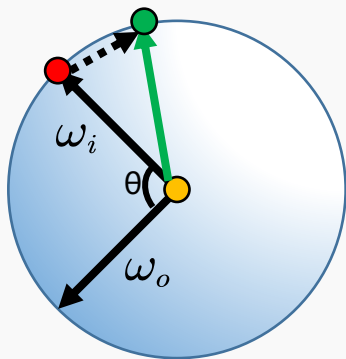


$$\nabla \cos \theta(\omega_i) = \frac{\hat{\omega}_o}{|\omega_i|} - \cos \theta \frac{\omega_i}{|\omega_i|^2}$$



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

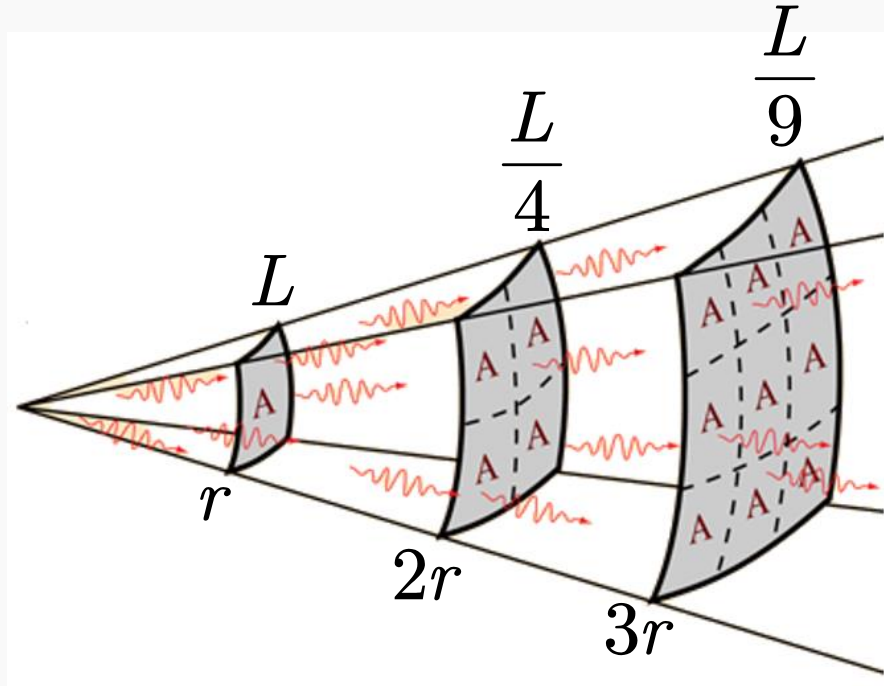
Gradient of Phase function



$$f_p \left(\text{Bloch Sphere} \right) \approx f_p \left(\text{Bloch Sphere} \right) + \left\langle \nabla f_p \left(\text{Bloch Sphere} \right), V \left(\text{Dashed Arrow} \right) \right\rangle$$

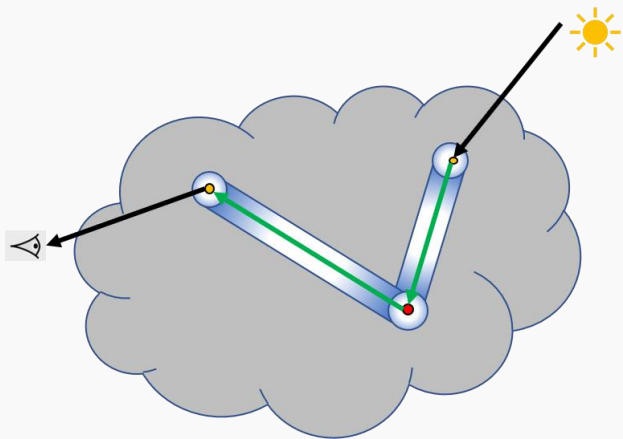
The equation shows the approximation of the phase function f_p at a new state. The first term is the phase function at the current state, represented by a Bloch sphere with vectors ω_i and ω_o and an angle θ . The second term is the inner product of the gradient of f_p at this state and a vector V pointing in the direction of the dashed arrow from the red dot in the top diagram. The new state is shown in a Bloch sphere with vectors ω_i and ω_o and an angle β .

Geometry factor



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

Gradient of Geometry factor



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

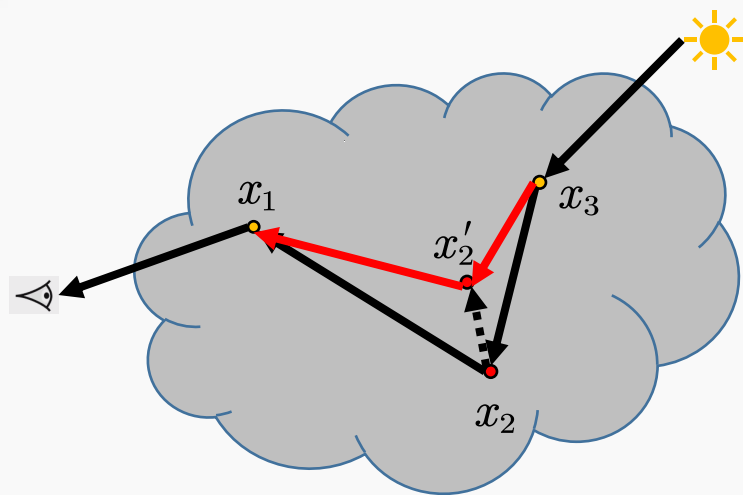
$$\nabla G(\overrightarrow{}) = \frac{2}{\|\overrightarrow{}\|^4} \overrightarrow{()}$$

Gradient of light path

$$\nabla T_r(\text{---})$$

$$\nabla f_p(\text{---})$$

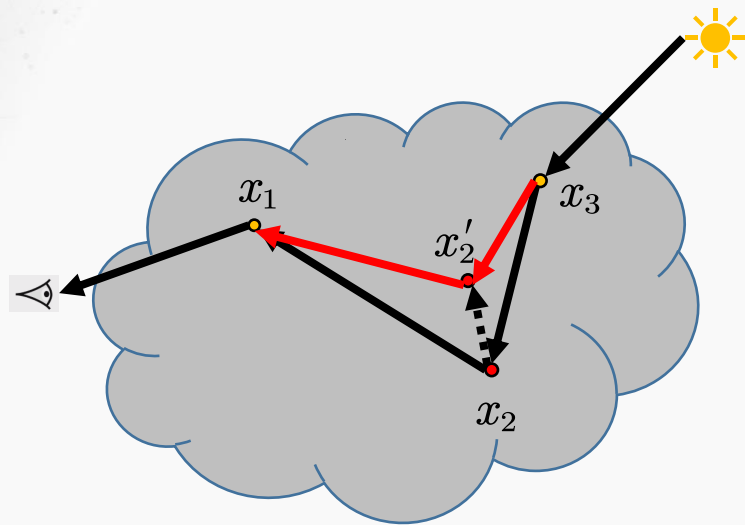
$$\nabla G(\text{---})$$



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

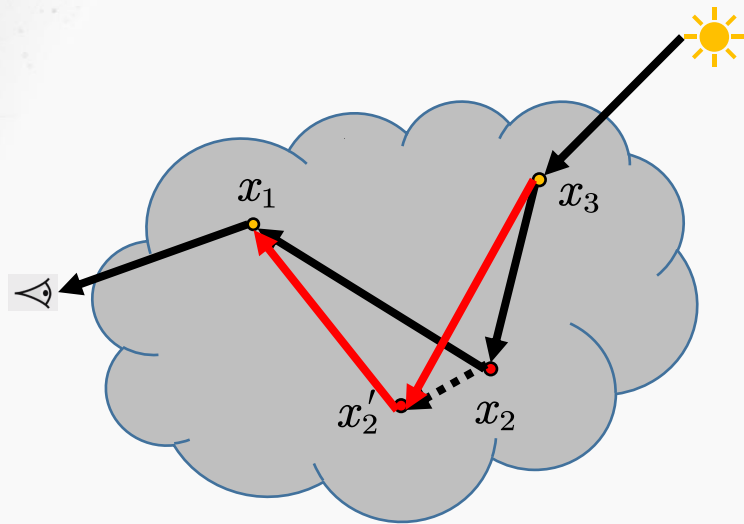
$$L(\text{---}) \approx L(\text{---}) + \langle \nabla L(\text{---}), V(\text{---}) \rangle$$

Gradient of light path



$$L(\text{red arrow}) \approx L(\text{black arrow}) + \langle \nabla L(\text{black arrow}), V(\text{dashed arrow}) \rangle$$

Gradient of light path



$$L(\text{red arrow}) \approx L(\text{black arrow}) + \langle \nabla L(\text{black arrow}), V(\text{dashed arrow}) \rangle$$

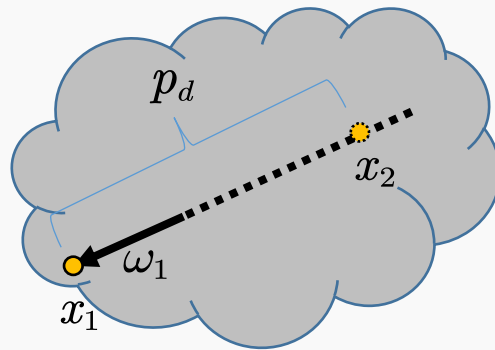


Implementation

- MLT
 - BDPT
 - M-H algorithm

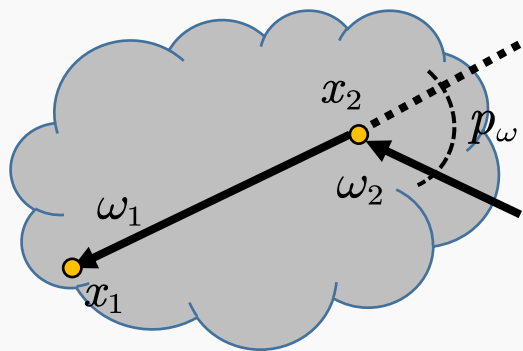


Sampling in the medium



(1) distance

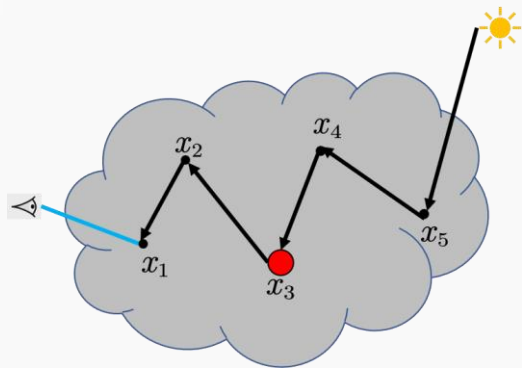
Sampling in the medium



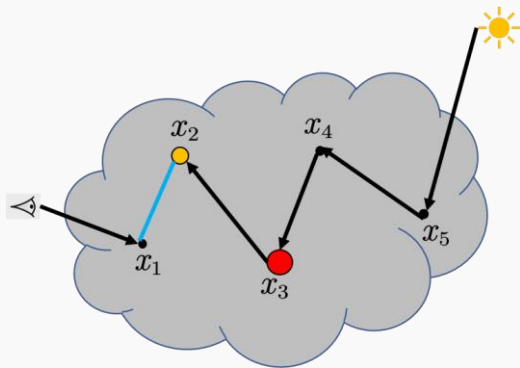
(2) direction

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

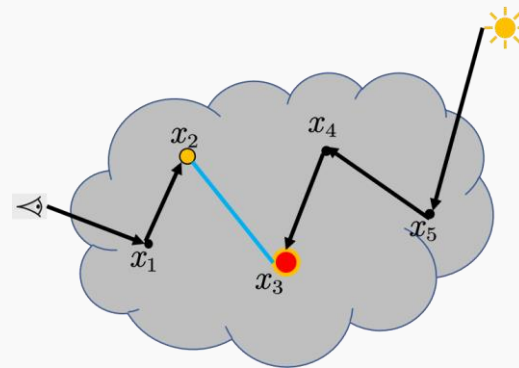
BDPT Connection



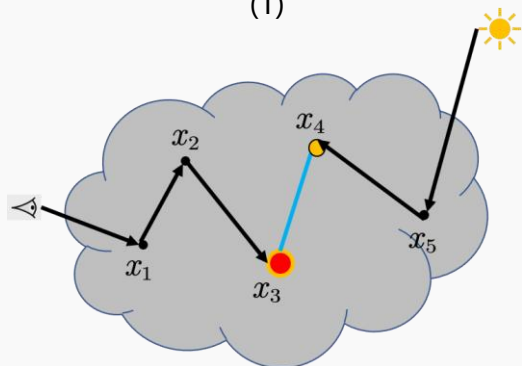
(1)



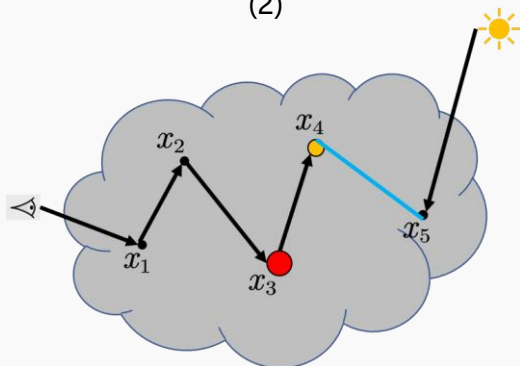
(2)



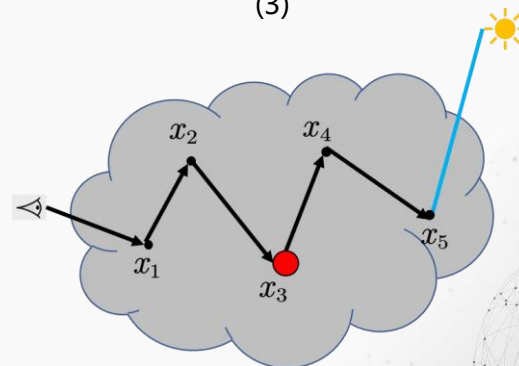
(3)



(4)



(5)



(6)

Vertex Type



(1) Camera

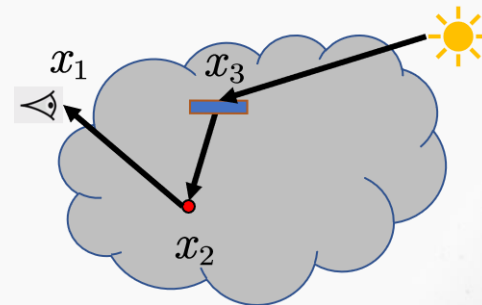
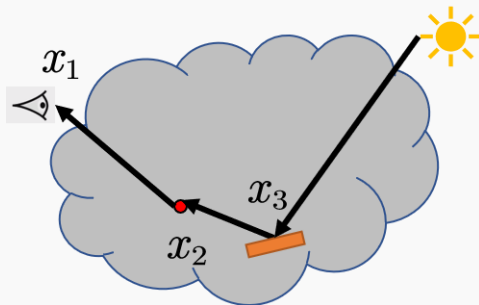
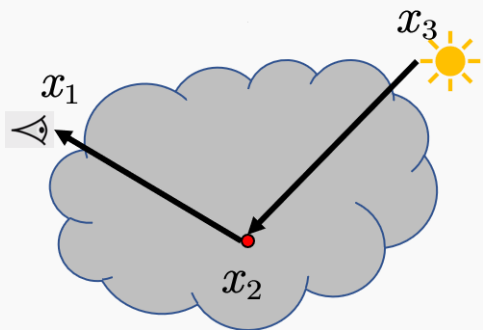
Diffuse

Specular

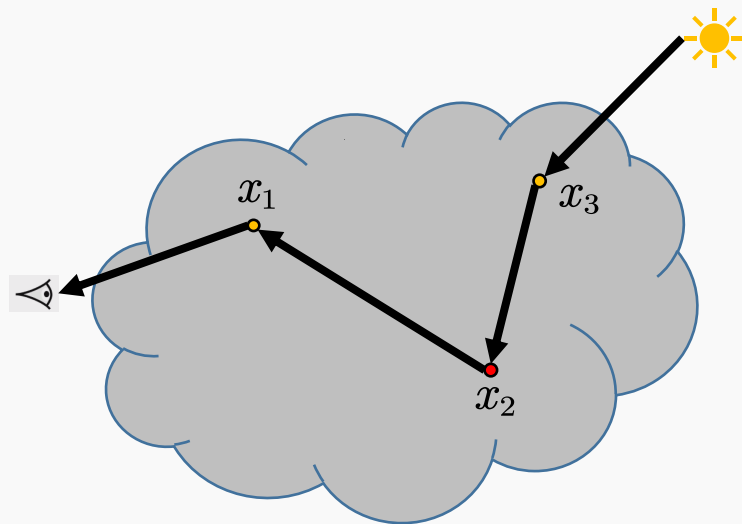
(2) Surface



(3) Light

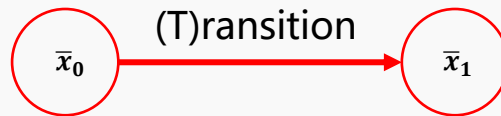
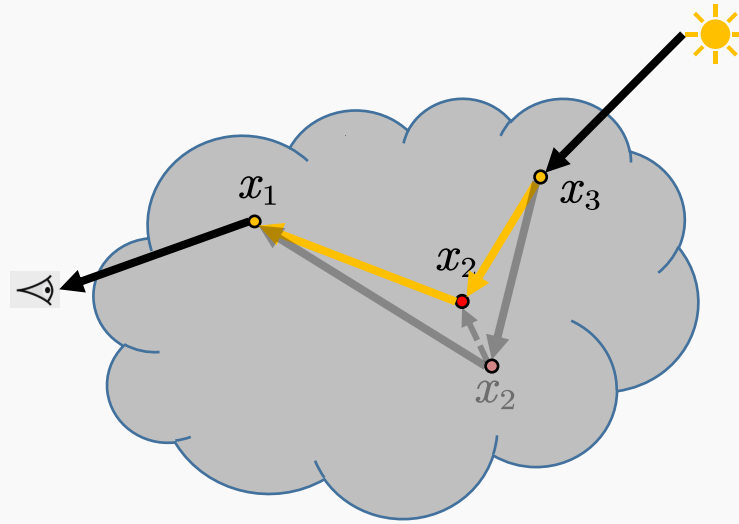


Gradient Mutation Strategy

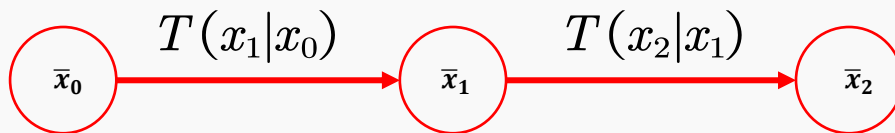
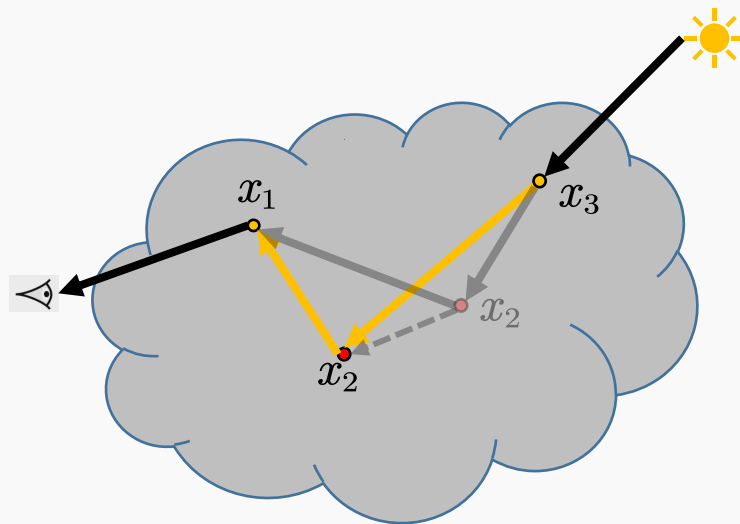


$$\bar{x}_0$$

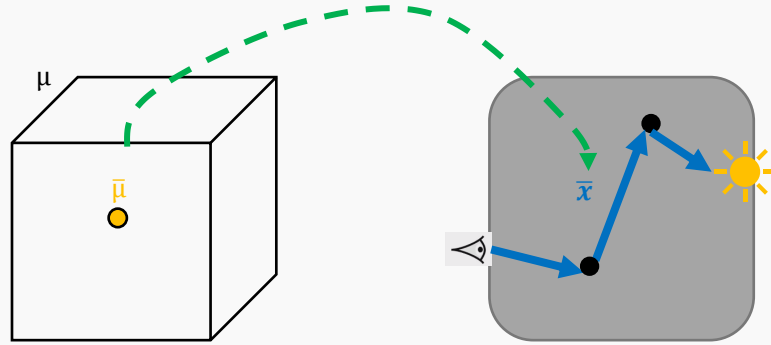
Gradient Mutation Strategy



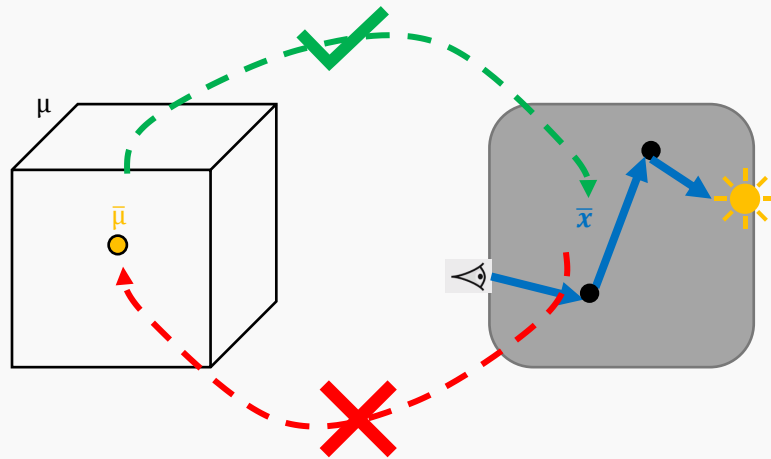
Gradient Mutation Strategy



Primary Sample Space



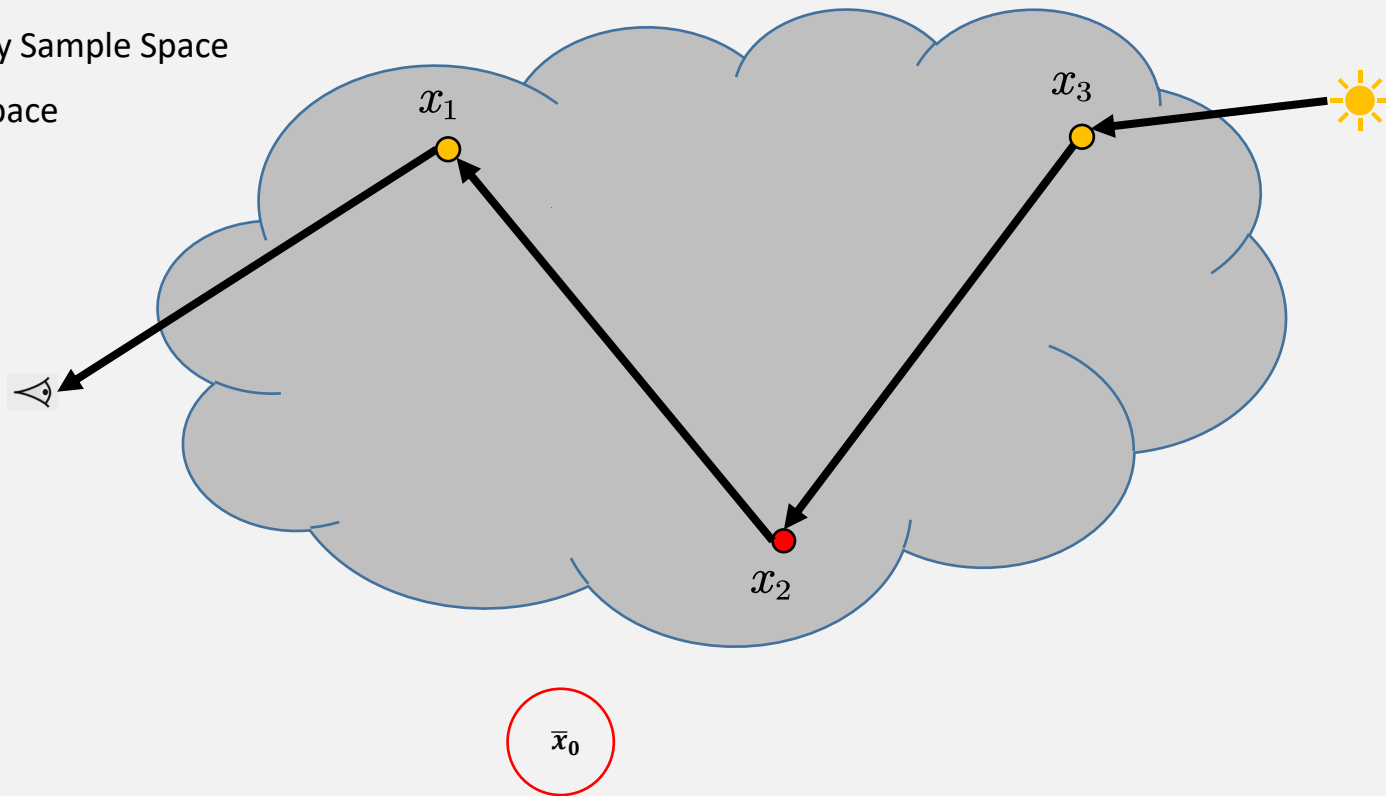
Primary Sample Space



Gradient Mutation Strategy

→ Primary Sample Space

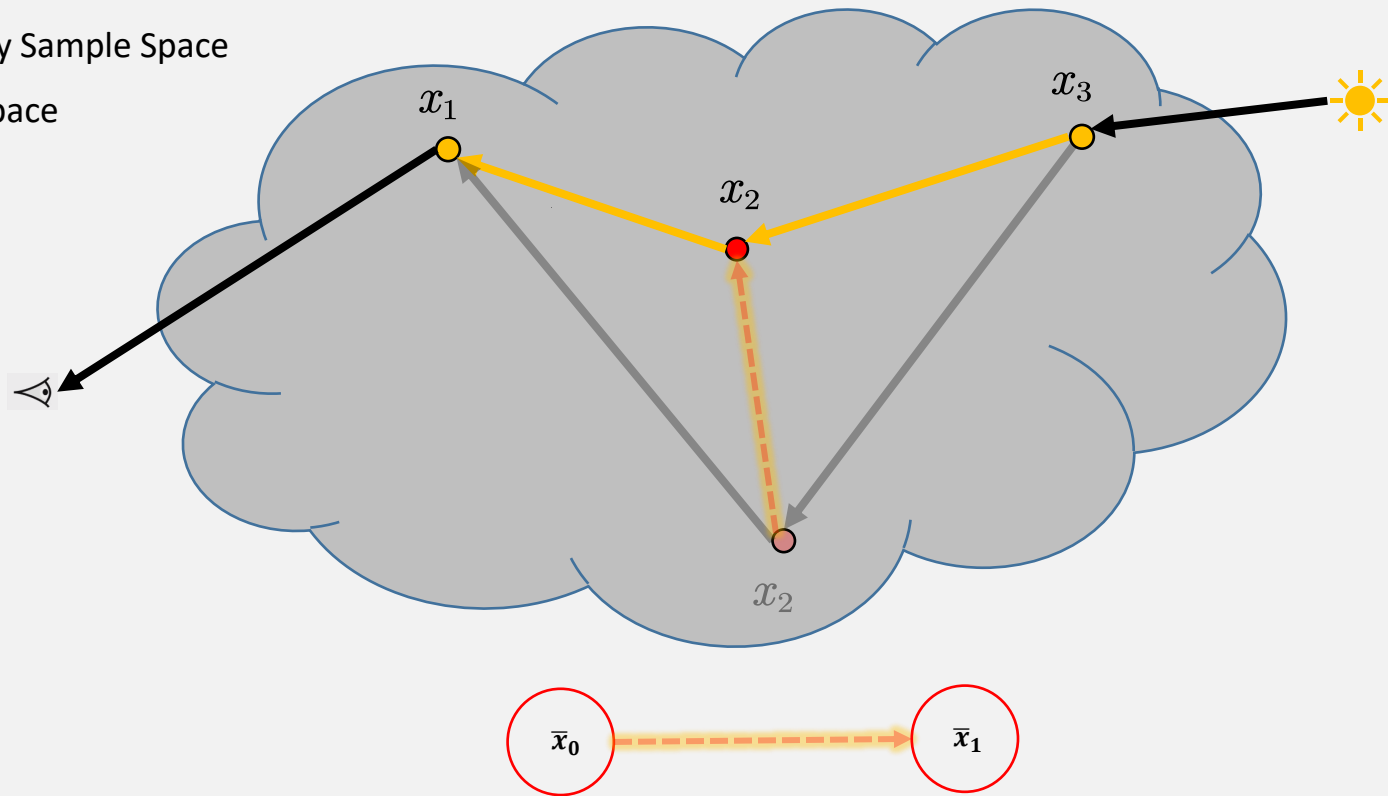
→ Path Space



Gradient Mutation Strategy

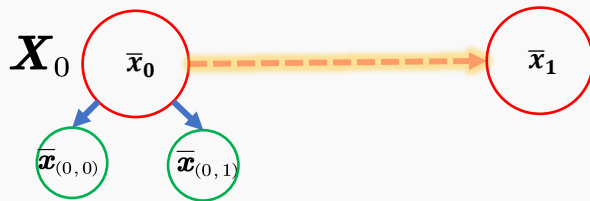
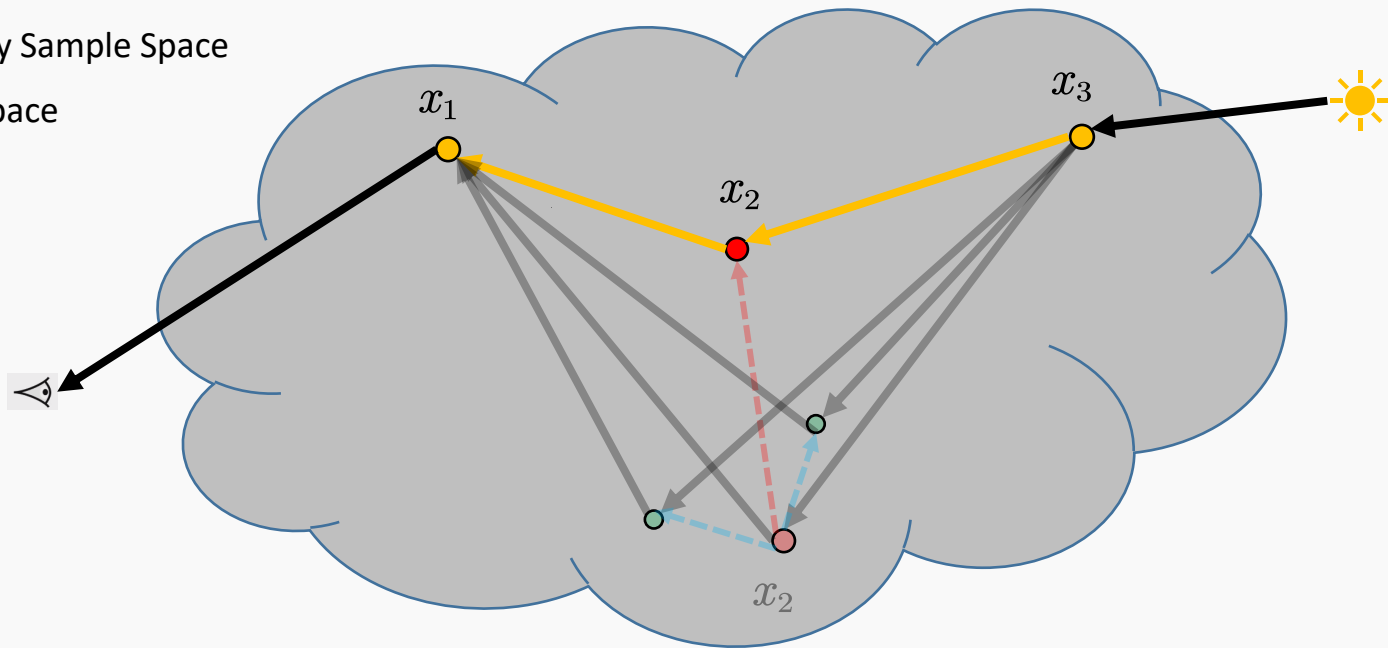
→ Primary Sample Space

→ Path Space

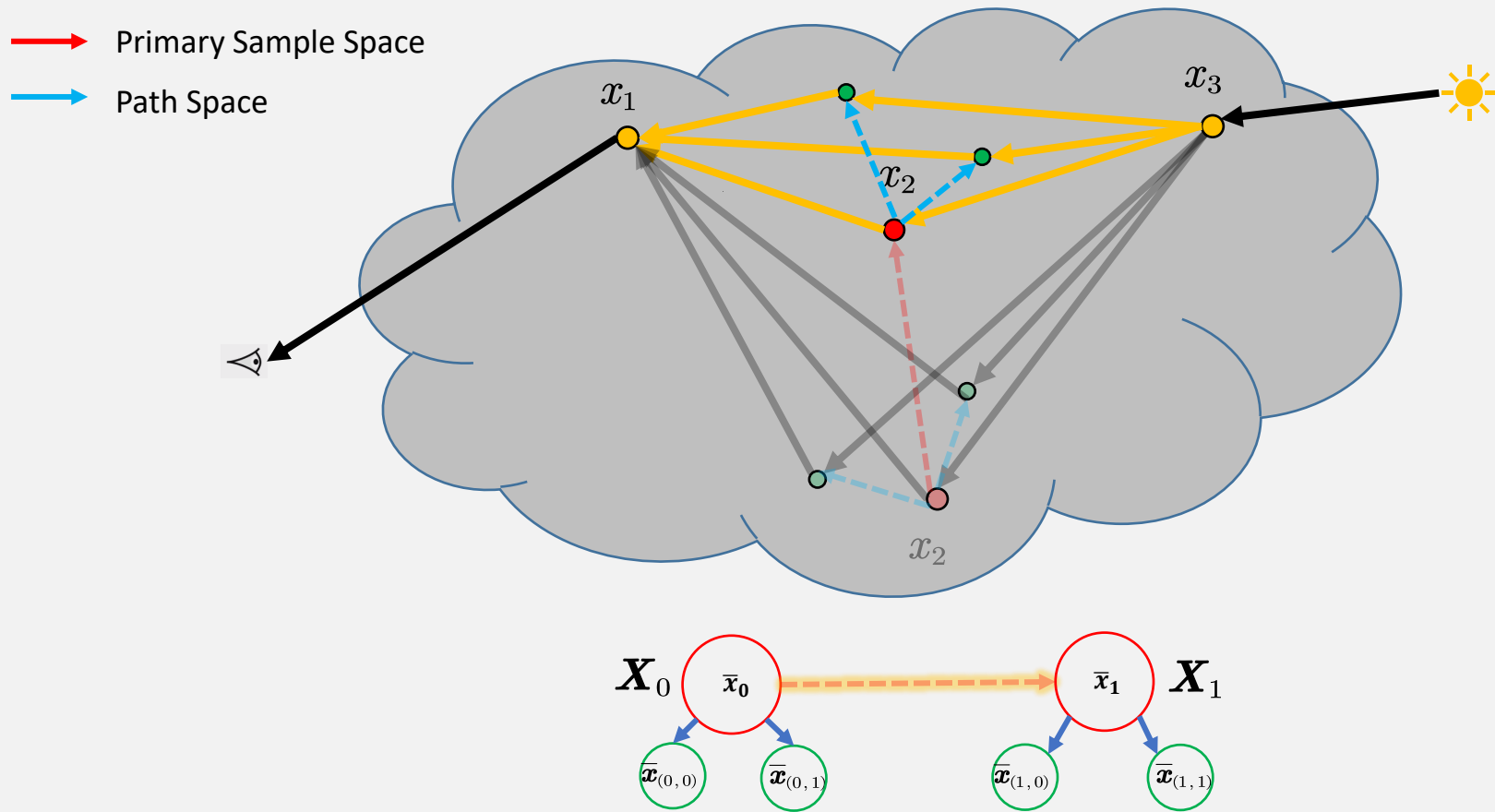


Gradient Mutation Strategy

- Primary Sample Space
- Path Space



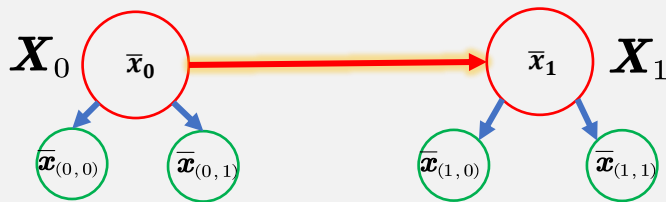
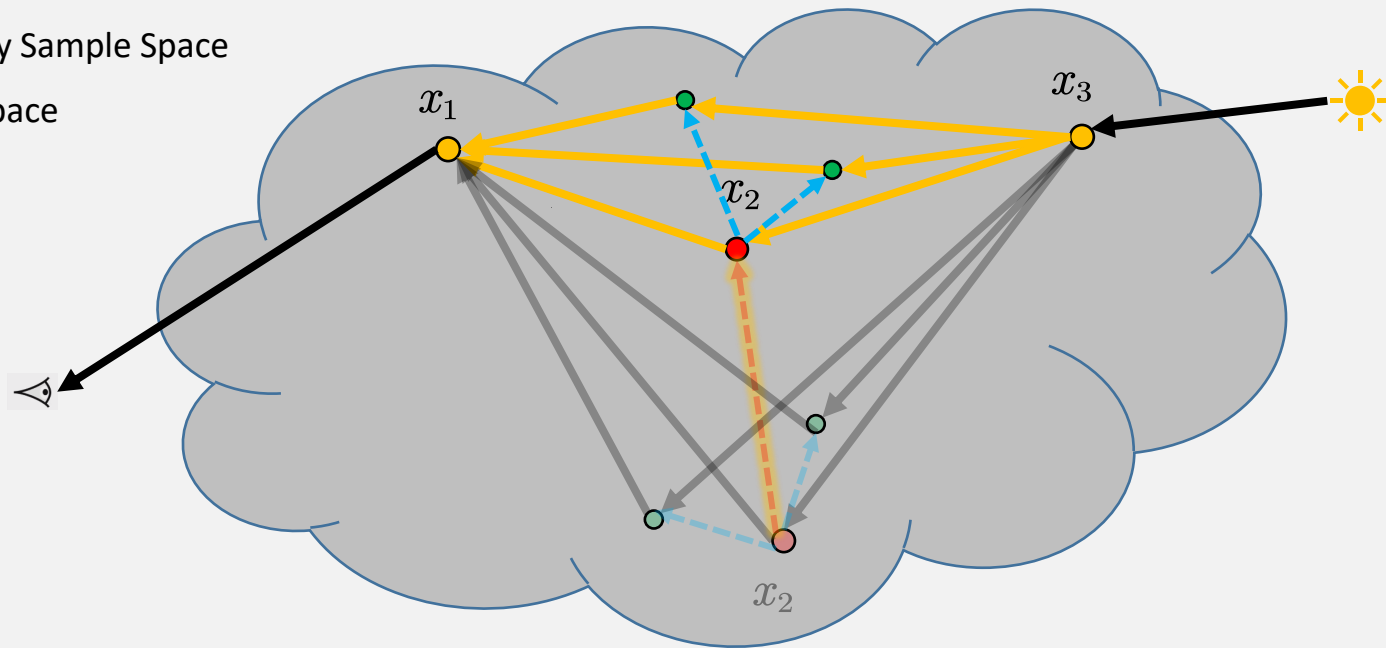
Gradient Mutation Strategy



Gradient Mutation Strategy

→ Primary Sample Space

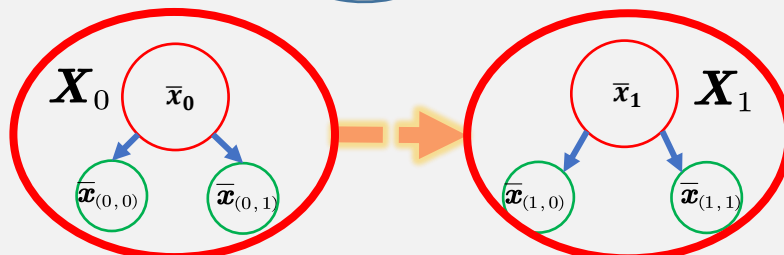
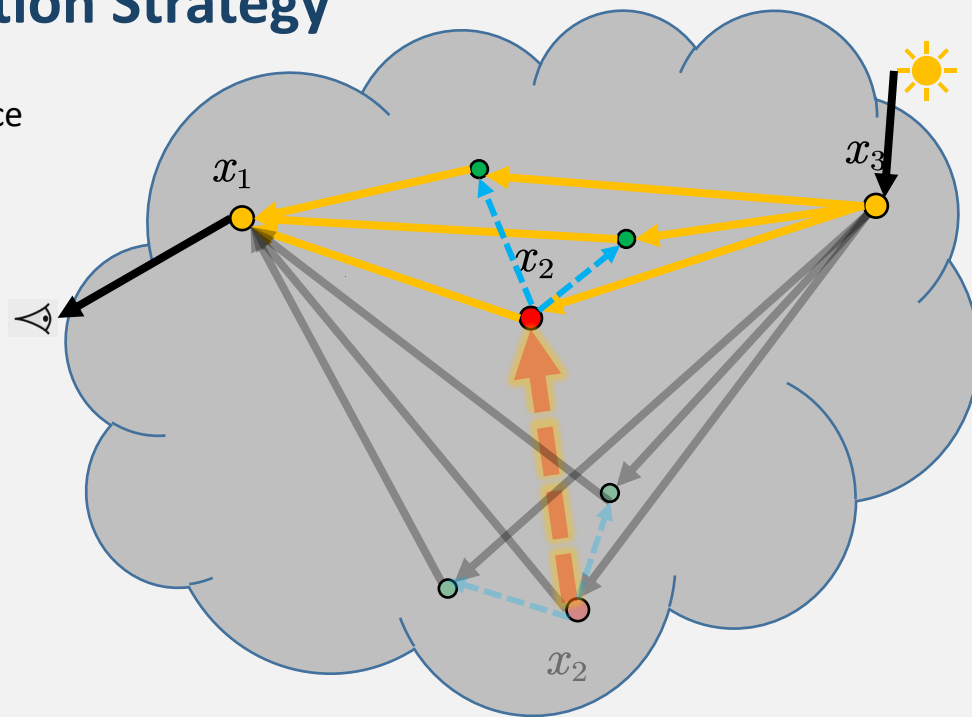
→ Path Space



Gradient Mutation Strategy

→ Primary Sample Space

→ Path Space





01

Motivation

02

Background

03

Gradient Algorithm

04

Result

05

Conclusion

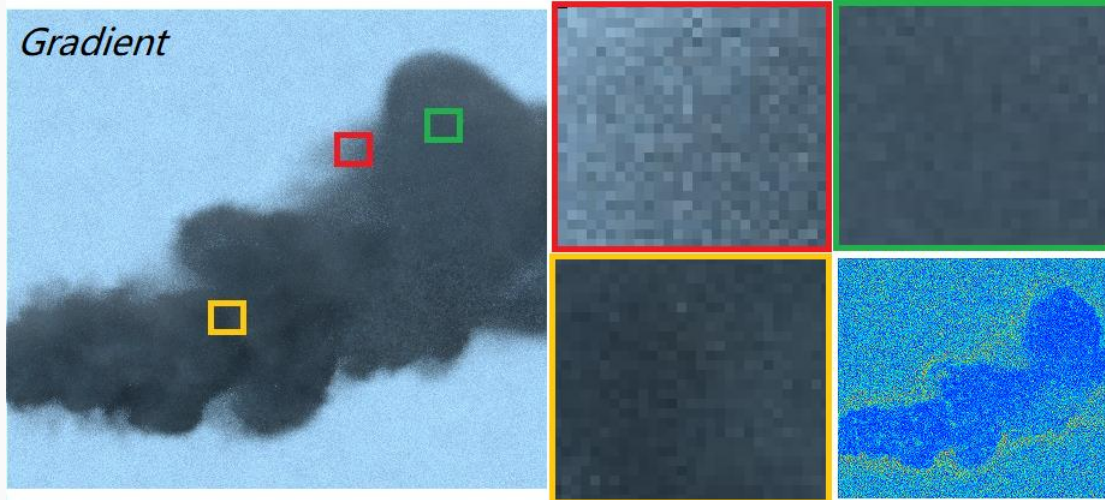


Result

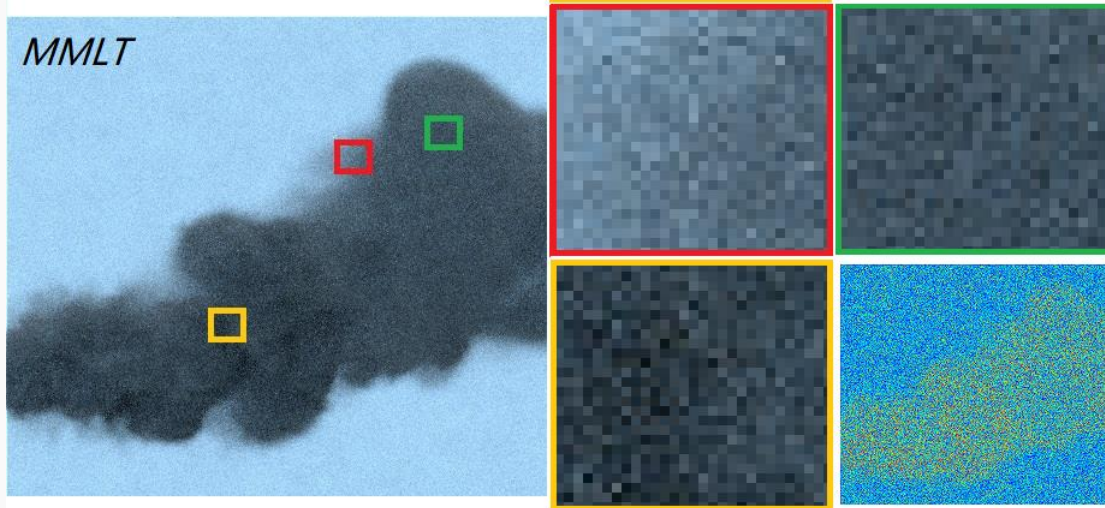
- All result rendered:
 - With up to 128 samples per pixel
 - an Intel Core i7-8550U at 1.8GHz using 8 cores
 - Pbrt-v3
 - 4 scenes



Gradient



MMLT



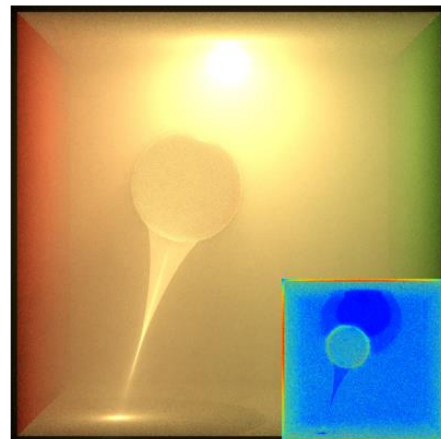
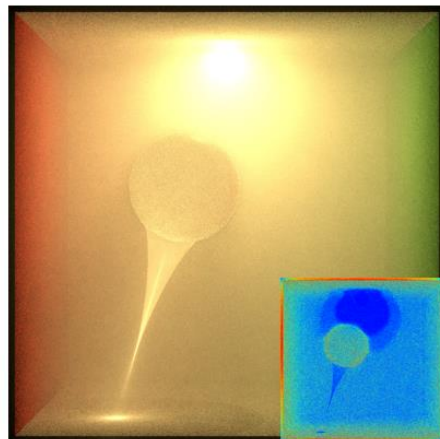
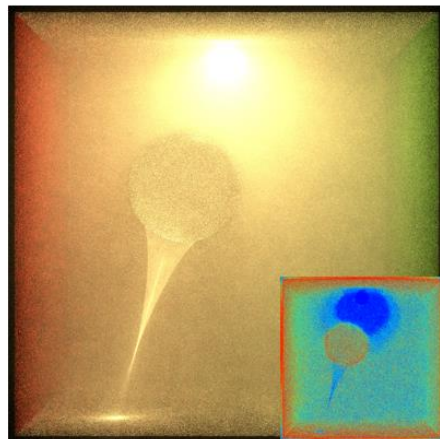
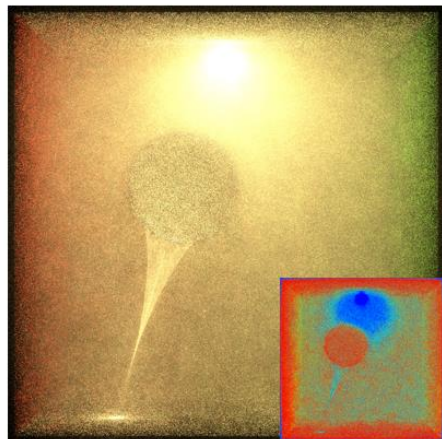
Gradient

4spp

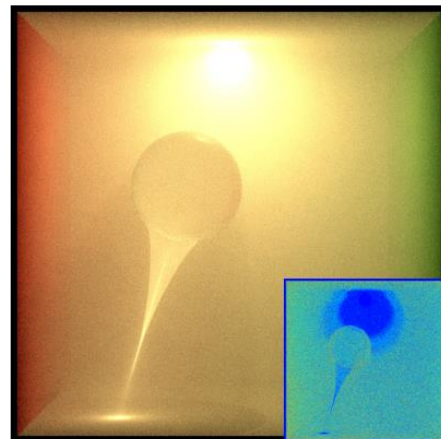
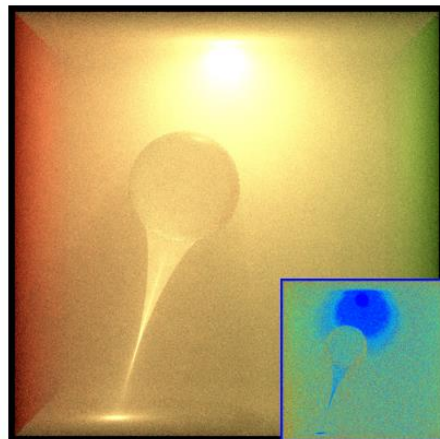
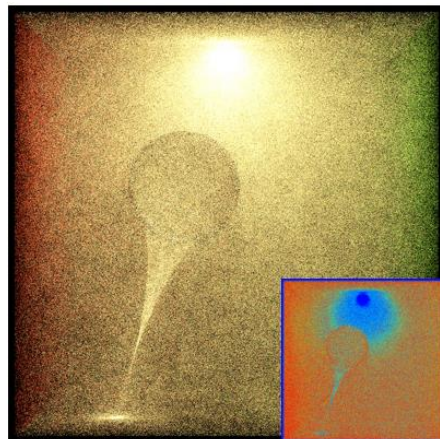
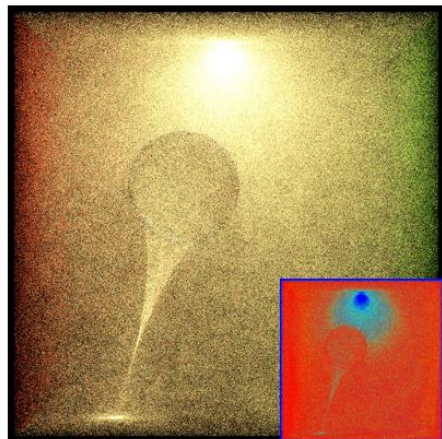
16spp

64spp

128spp



MMLT

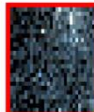




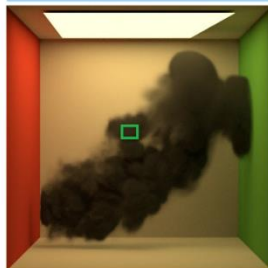
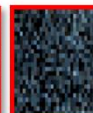
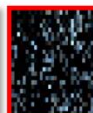
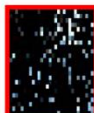
Scene



Gradient



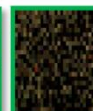
MMLT



Gradient



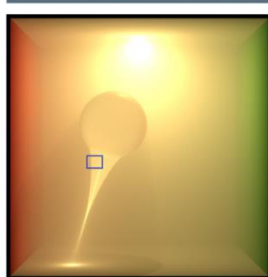
MMLT



Gradient



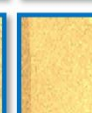
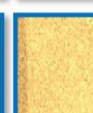
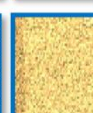
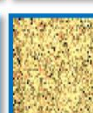
MMLT



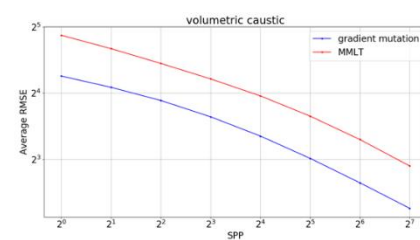
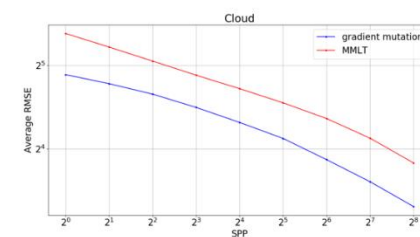
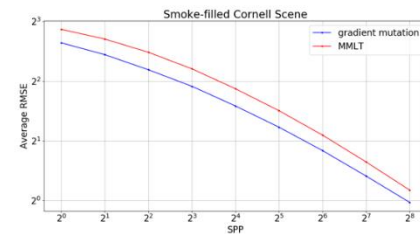
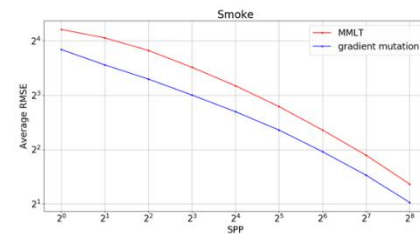
Gradient



MMLT



RMSE vs. 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
- 

The background features faint, light gray geometric patterns in the corners. These patterns consist of interconnected dots and lines, forming a mesh-like structure that resembles a stylized globe or a complex network diagram. The patterns are most prominent in the top-left and bottom-right corners, with some lines extending towards the center.

THANK YOU

