

Gradient-Domain Path Tracing

Markus Kettunen	Aalto University
Marco Manzi	University of Bern
Miika Aittala	Aalto University
Jaakko Lehtinen	Aalto University and NVIDIA Research
Fredo Durand	MIT CSAIL
Matthias Zwicker	University of Bern

To appear in ACM Transactions on Graphics (Proc. SIGGRAPH 2015)

CG技術の実装と数理 2015

2015/07/26

Introduction

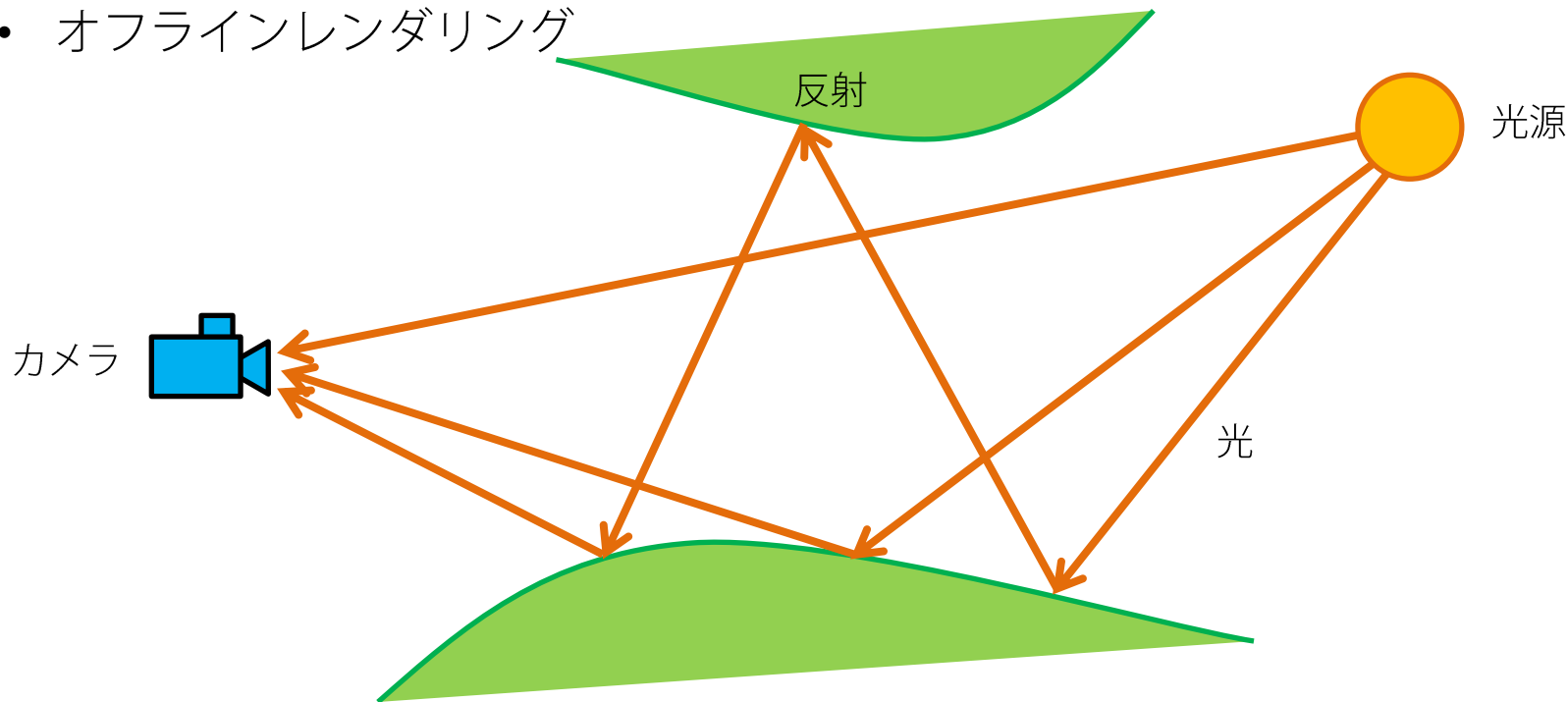
Introduction

- レンダリング方程式 [Kajiya 1986]
 - レンダリングにおける支配方程式。

$$L(\vec{x}, \vec{\omega}) = L_o(\vec{x}, \vec{\omega}) + \int_{\Omega} f_r(\vec{x}, \vec{\omega}, \vec{\omega}') \cdot L_i(\vec{x}, \vec{\omega}') \cdot \cos \theta d\mu(\vec{\omega}')$$

Introduction

- オフラインレンダリング



$$L(\vec{x}, \vec{\omega}) = L_o(\vec{x}, \vec{\omega}) + \int_{\Omega} f_r(\vec{x}, \vec{\omega}, \vec{\omega}') \cdot L_i(\vec{x}, \vec{\omega}') \cdot \cos \theta d\mu(\vec{\omega}')$$

Introduction

- オフラインレイトレーシング手法は数多く存在する
 - Unbiased系

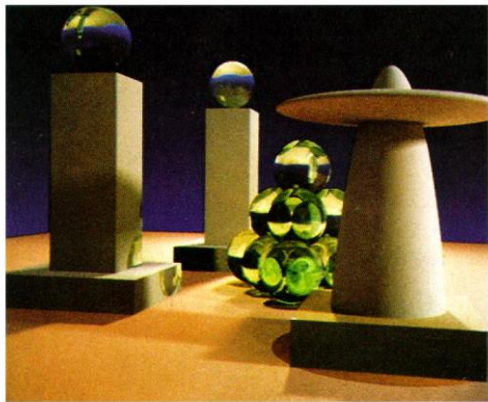


Figure 6. A sample image. All objects are neutral grey. Color on the objects is due to caustics from the green glass balls and color bleeding from the base polygon.

Path Tracing [Kajiya 1986]

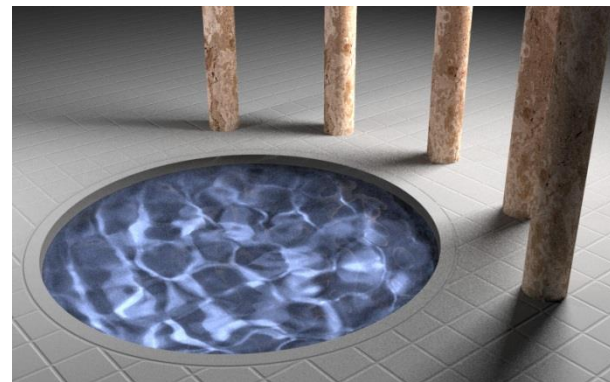


(a) Bidirectional path tracing with 25 samples per pixel



(b) Standard path tracing with 56 samples per pixel (the same computation time as (a))

Bi-directional Path Tracing
[Veach and Guibas 1994]



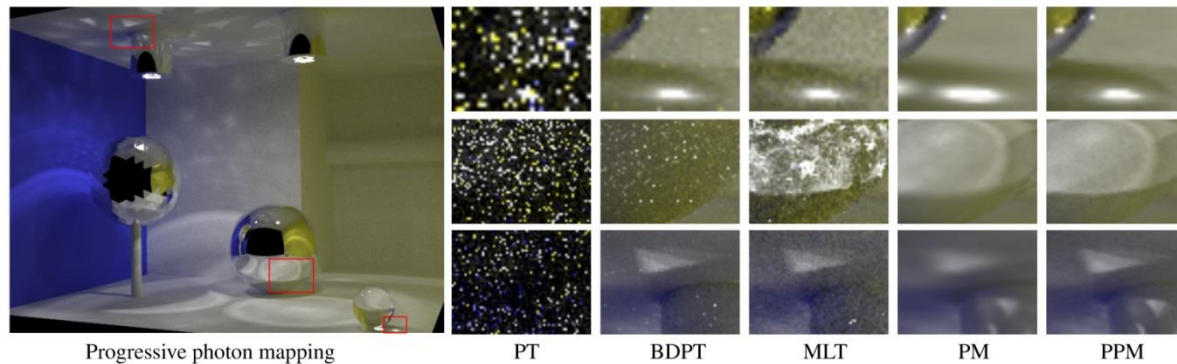
Metropolis Light Transport
[Veach and Guibas 1997]

Introduction

- オフラインレイトレーシング手法は数多く存在する
 - フォトンマップ系



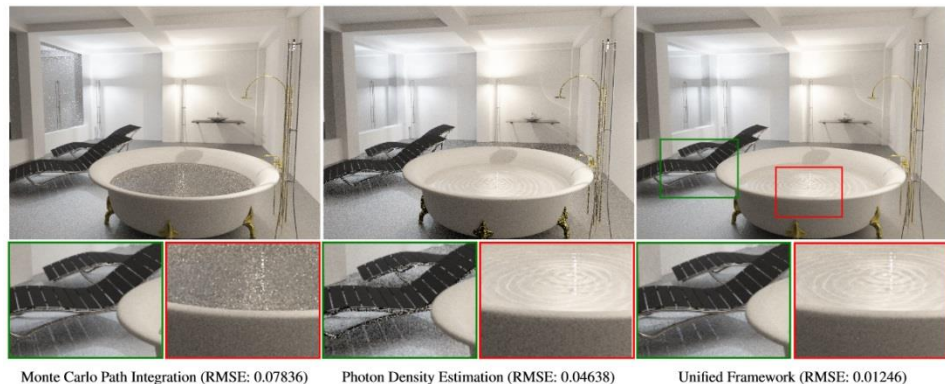
Photon Mapping [Jensen 1996]



Progressive Photon Mapping [Hachisuka 2008]

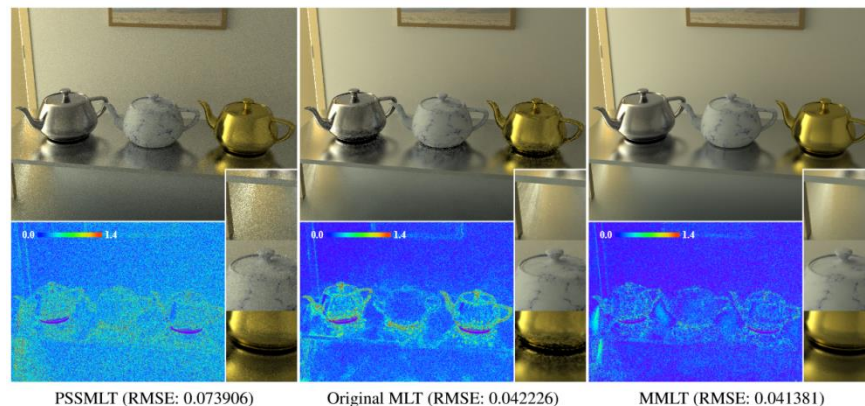
Introduction

- オフラインレイトレーシング手法は数多く存在する
 - ハイブリッド系



Vertex Connection and Merging [Georgiev 2012]
Unified Path Space sampling [Hachisuka 2012]

Photon Mapping
+
Bi-directional Path Tracing



Multiplexed Metropolis Light Transport
[Hachisuka 2014]

Metropolis Light Transport
+
Bi-directional Path Tracing

Introduction

- Gradient-Domain Rendering



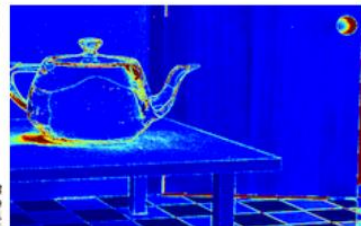
Horiz. differences I^{dx}



Vertical differences I^{dy}



Coarse image I^g



Sample density



Result

Gradient-Domain Metropolis Light Transport [Lehtine 2013]



Coarse image



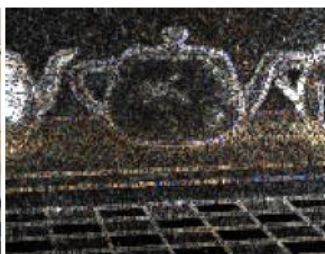
L_2 (OURS)



Gradients (OURS)



L_2 (GDMLT)



Gradients (GDMLT)

Improved Sampling for Gradient-Domain Metropolis Light Transport [Manzi 2014]

Introduction

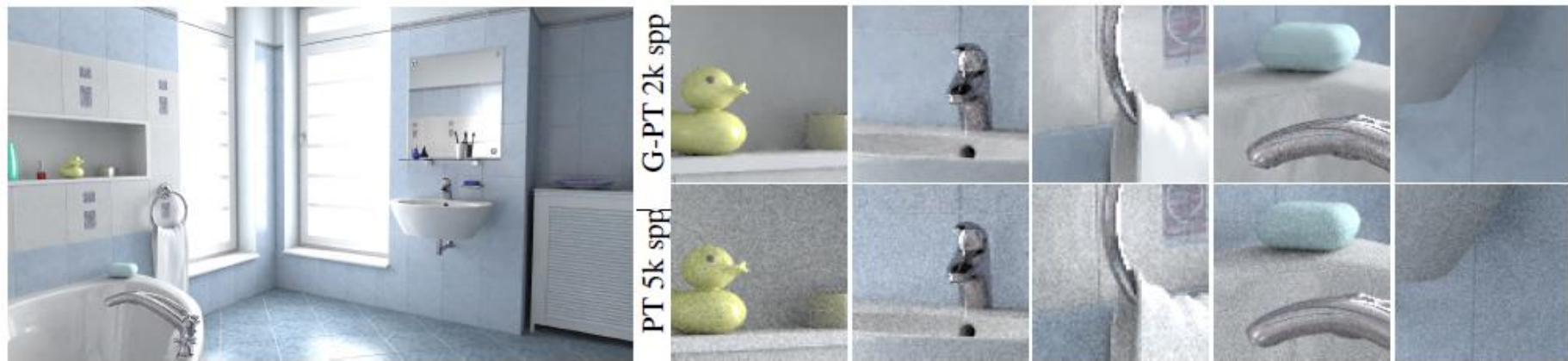
- 高度なレンダリング手法は実装が大変

“Metropolis light transport is *notoriously hard to implement*, however, and to the best of our knowledge, *Mitsuba [Jakob 2012]* is the *only publicly-available implementation* of Veach’s original algorithm [Veach and Guibas 1997].”

“メトロポリス光輸送法は実装が超難しいし、自分の知る限り、Veachのオリジナルの手法を実装してあるオープンなレンダラはMitsubaだけだよ!”

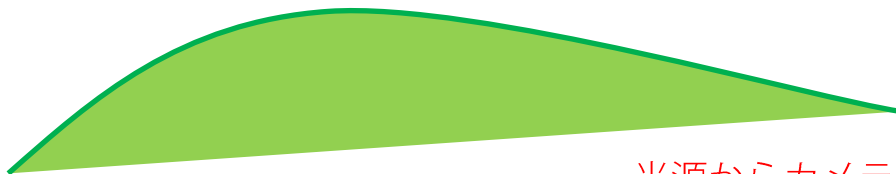
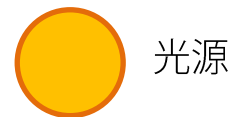
Introduction

- 提案手法
 - Gradient-Domain Rendering → 高効率！
 - 普通のPath Tracingを利用 → 実装が簡単！



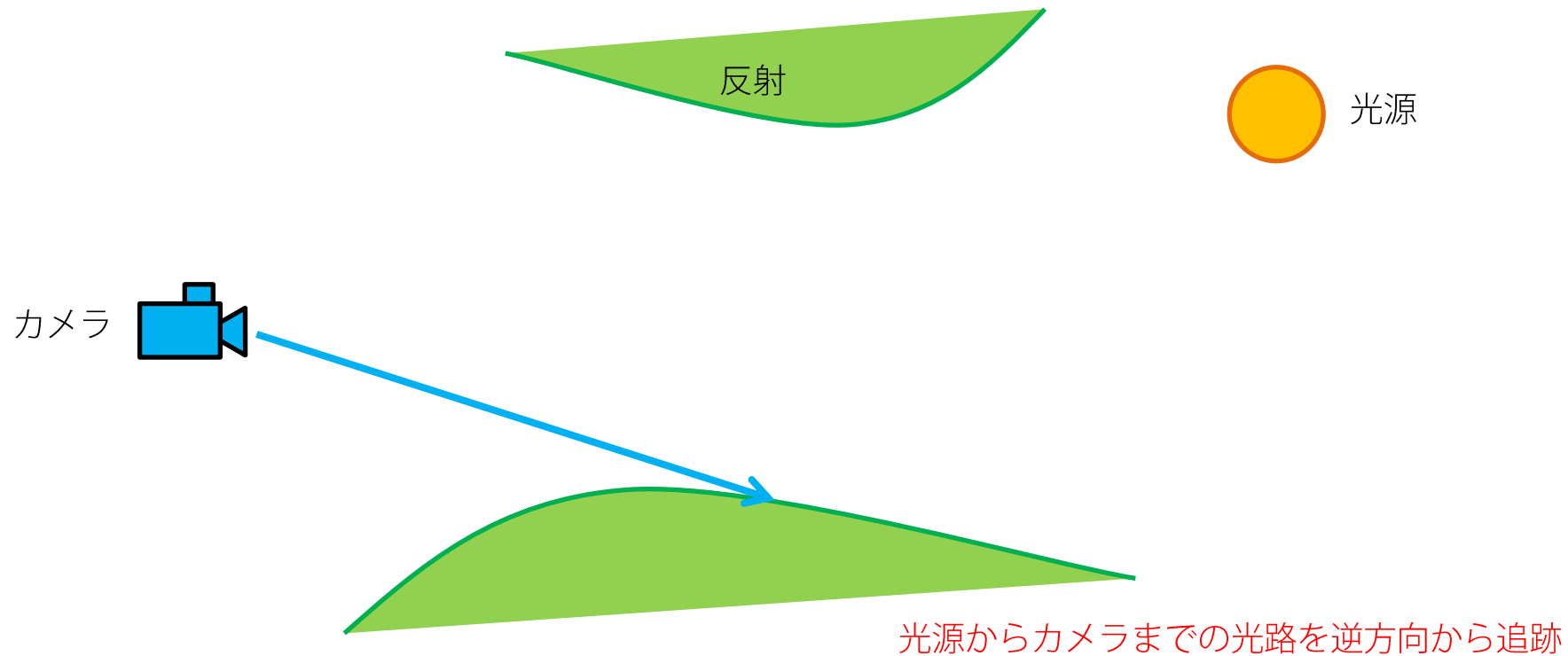
Path Tracing

Path Tracing

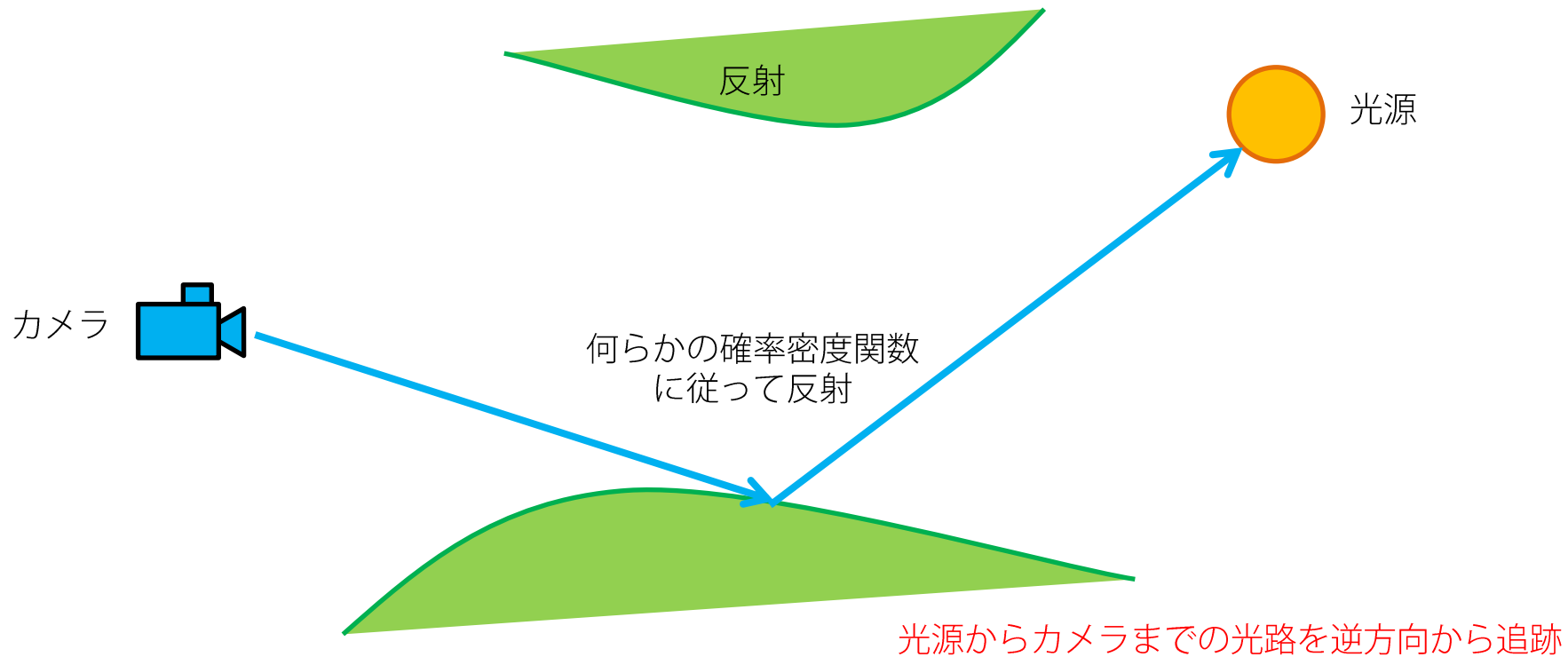


光源からカメラまでの光路を逆方向から追跡

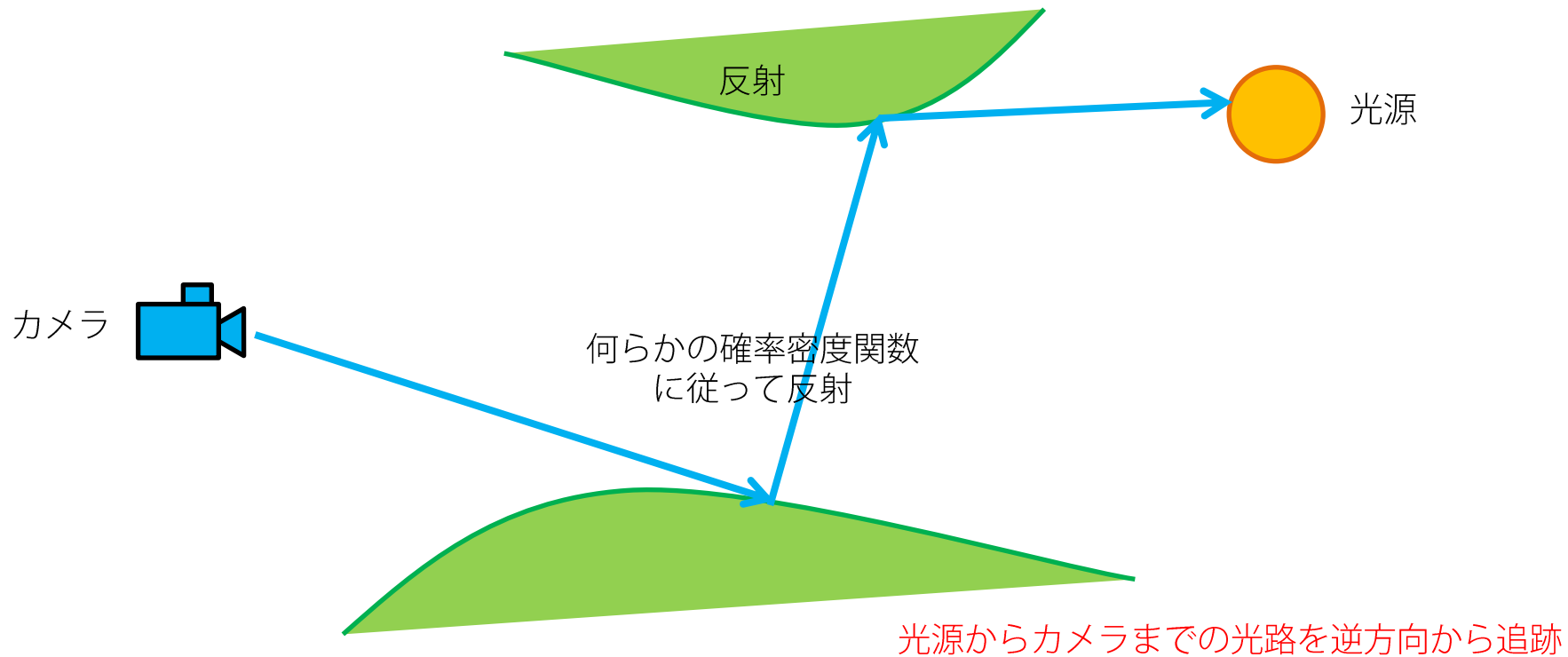
Path Tracing



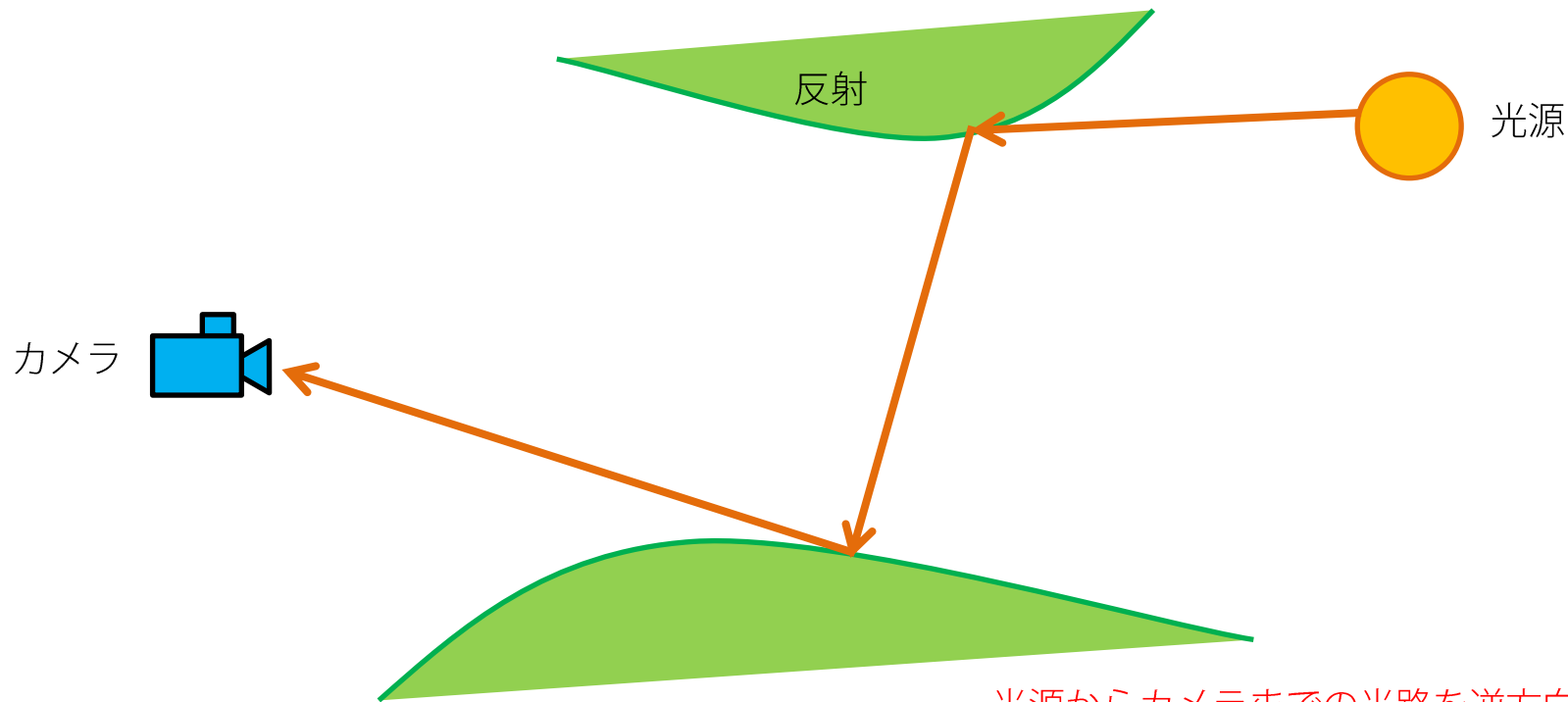
Path Tracing



Path Tracing



Path Tracing



光源からカメラまでの光路を逆方向から追跡

Overview

Overview

1. 勾配画像を直接Path Tracingでレンダリングして得る。(モンテカルロ積分)



Horiz. differences I^{dx}



Vertical differences I^{dy}

Overview

1. 勾配画像を直接Path Tracingでレンダリングして得る。(モンテカルロ積分)
2. 普通にPath Tracingでレンダリング。(モンテカルロ積分)



Coarse image I^g

Overview

1. 勾配画像を直接Path Tracingでレンダリングして得る。(モンテカルロ積分)
2. 普通にPath Tracingでレンダリング。(モンテカルロ積分)
3. 2の結果をヒントにしつつ、1の結果に対するPoisson Problemを解いて元の画像を復元。

$$\operatorname{argmin}_I \left(\left\| \begin{pmatrix} H^{dx} I \\ H^{dy} I \end{pmatrix} - \begin{pmatrix} I^{dx} \\ I^{dy} \end{pmatrix} \right\|_2^2 + \|\alpha (I - I^g)\|_2^2 \right).$$

直接得られた
勾配画像

復元画像の勾配

復元画像と
Path Tracing画像の差

Overview

- 勾配画像のレンダリング

$$\Delta_{i,j} = \left(h(x) * \int_{\Omega} f(x, \bar{p}) d\mu(\bar{p}) \right) (x_i) \\ - \left(h(x) * \int_{\Omega} f(x, \bar{p}) d\mu(\bar{p}) \right) (x_j),$$

隣接ピクセル間で有限差分法
(Ω path space)

Overview

- 式を変形

決定的な変換関数：Shift Map T_{ij}
+ 変数変換

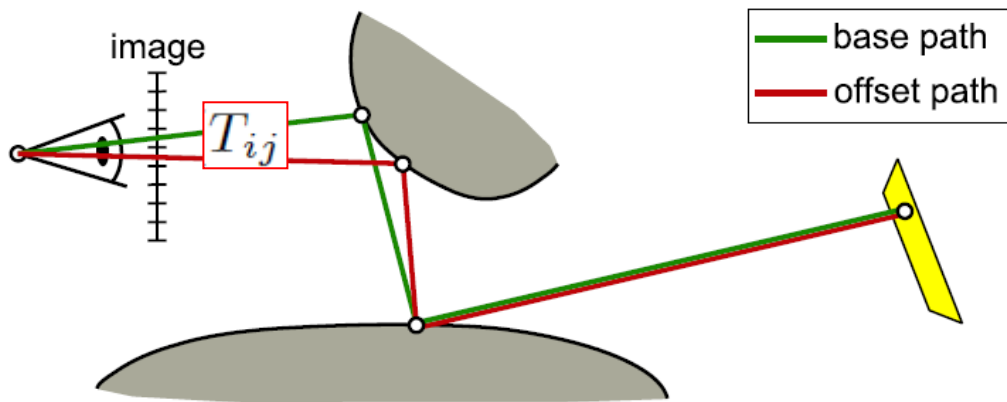
$$\begin{aligned} \Delta_{i,j} &= \left(h(x) * \int_{\Omega} f(x, \bar{p}) d\mu(\bar{p}) \right) (x_i) \\ &\quad - \left(h(x) * \int_{\Omega} f(x, \bar{p}) d\mu(\bar{p}) \right) (x_j), \end{aligned} \quad \Rightarrow \quad \begin{aligned} \Delta_{i,j} &= \left(h(x) * \int_{\Omega} f(x, \bar{p}) - f(T_{ij}(x, \bar{p})) |T'_{ij}| d\mu(\bar{p}) \right) (x_i) \\ &= \left(h(x) * \int_{\Omega} g_{ij}(x, \bar{p}) d\mu(\bar{p}) \right) (x_i), \end{aligned} \quad (2)$$

ピクセルごとに独立に計算してナイーブに勾配を求めるより、決定的な変換を行って勾配を計算した方が推定誤差が小さくなる。

Base Path \rightarrow Offset Path

Overview

1. 勾配画像のレンダリング



Theoretical Analysis

周波数空間における振る舞い

- サンプルング過程をWide-Sense Stationaryと仮定。
 - サンプルング範囲が無限大になる。
- この時、サンプルング過程はPoisson過程になる。

周波数空間における振る舞い

勾配画像の周波数
空間でのエネル
ギー

勾配画像の周波数
空間での誤差

$$\frac{|\epsilon_G(\omega_x)|^2}{|\epsilon_F(\omega_x)|^2} = \frac{\|G\|^2}{\|F\|^2},$$

通常のレンダリング
結果の周波数空間で
の誤差

GはFに対して低周波成分は小さく、高周波成分は大きくなる。

通常のレンダリング
結果の周波数空間で
のエネルギー

周波数空間における振る舞い

復元結果の周波数
空間での誤差

$$|\epsilon_{R_\alpha}(\omega_x)|^2 = \frac{1}{n} \frac{\alpha^4 \|F\|^2 + |D(\omega_x)|^2 \|G\|^2}{(\alpha^2 + |D(\omega_x)|^2)^2}. \quad (7)$$

最適なパラメータ α

$$\alpha_*^2 = \|G\|^2 / \|F\|^2$$

周波数空間における振る舞い

通常のレンダリング結果と
復元結果の誤差の比
(最適なパラメータ α 下)

$$\frac{|\epsilon_F(\omega_x)|^2}{|\epsilon_{R_{\alpha_*}}(\omega_x)|^2} = \frac{\|G\|^2 / \|F\|^2 + 2 - 2 \cos(2\pi\omega_x)}{\|G\|^2 / \|F\|^2},$$

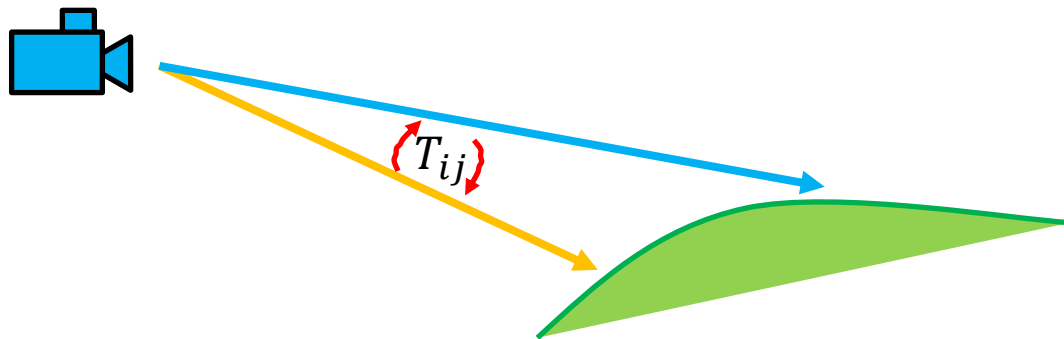
高周波成分ほど、誤差の比
が小さくなる！

Shift Mapと詳細な定式化

Shift Map

- T_{ij} をどう設計するか
- 従来のShift Mapは実装が大変だった！
 - 全ての空間で全単射なMapを作る必要があった。

そのような制限を緩和した
定式化を行う。



新しい定式化

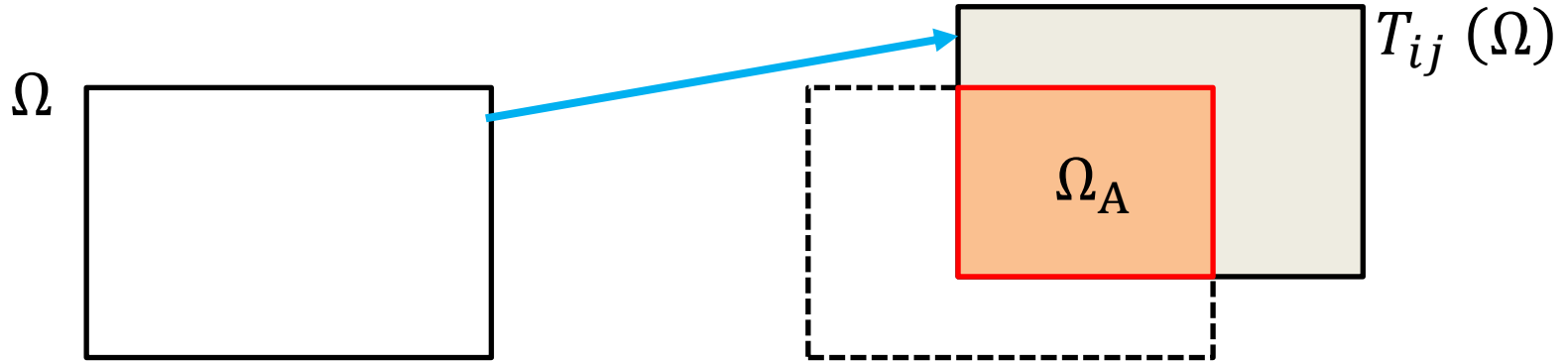
- Symmetric Gradients [Manji 2014]

$$\begin{aligned}\Delta_{i,j} &= \left(h(x) * \int_{\Omega} f(x, \bar{p}) - f(T_{ij}(x, \bar{p})) |T'_{ij}| d\mu(\bar{p}) \right) (x_i) \\ &= \left(h(x) * \int_{\Omega} g_{ij}(x, \bar{p}) d\mu(\bar{p}) \right) (x_i),\end{aligned}\tag{2}$$

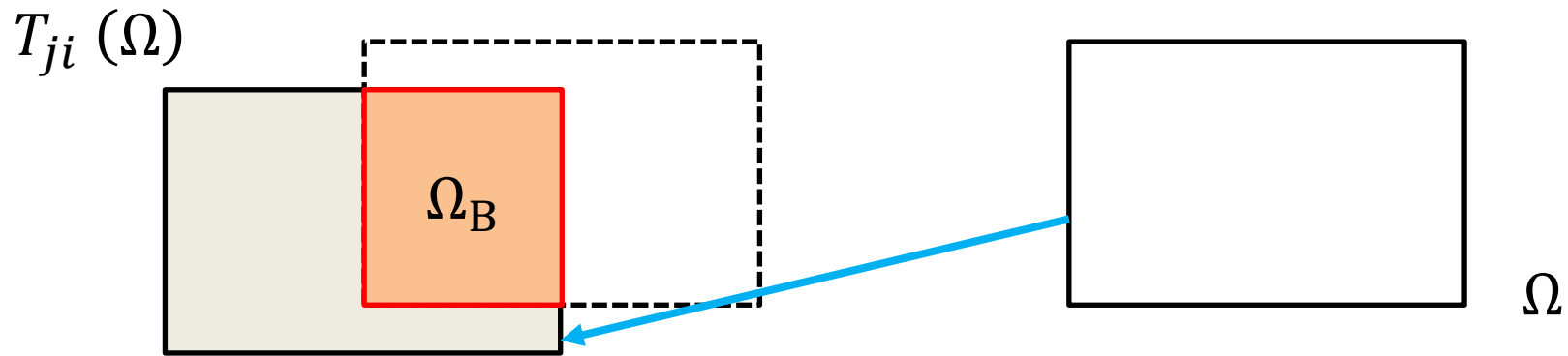
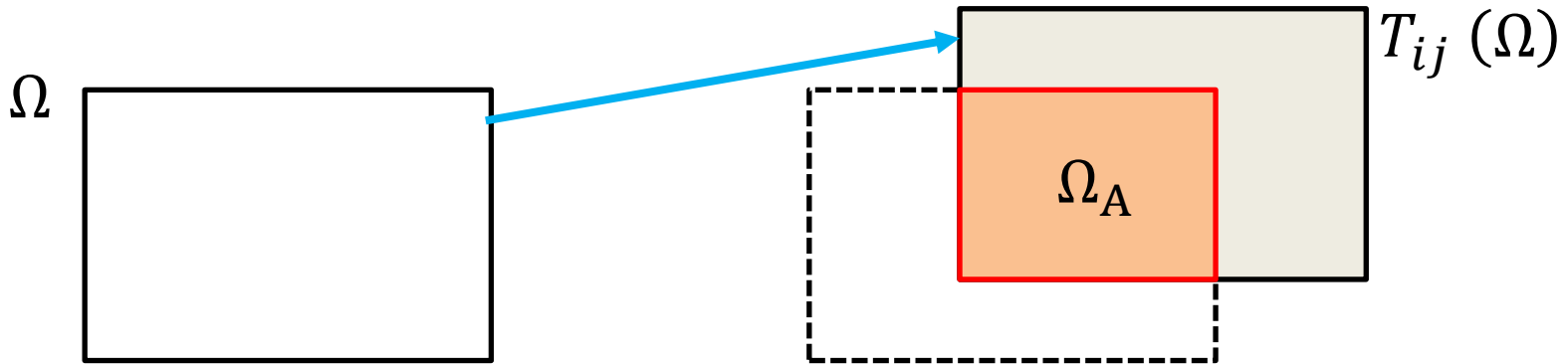


$$\begin{aligned}\Delta_{i,j} &= \left(h(x) * \int_{\Omega} w_{ij}(x, \bar{p}) g_{ij}(x, \bar{p}) d\mu(\bar{p}) \right) (x_i) + \\ &\quad \left(h(x) * \int_{\Omega} w_{ji}(x, \bar{p}) g_{ji}(x, \bar{p}) d\mu(\bar{p}) \right) (x_j).\end{aligned}$$

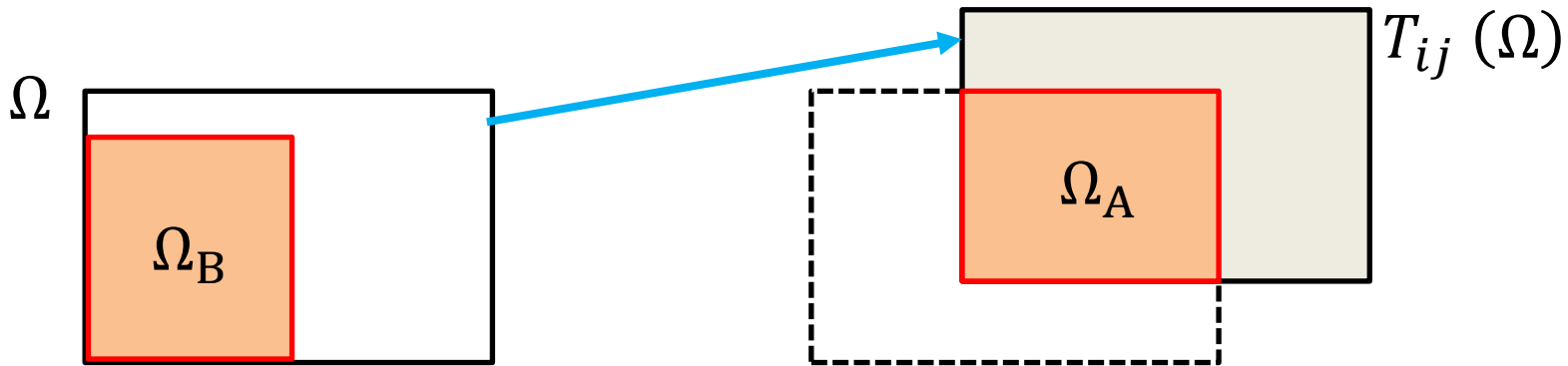
積分範囲をShift Map



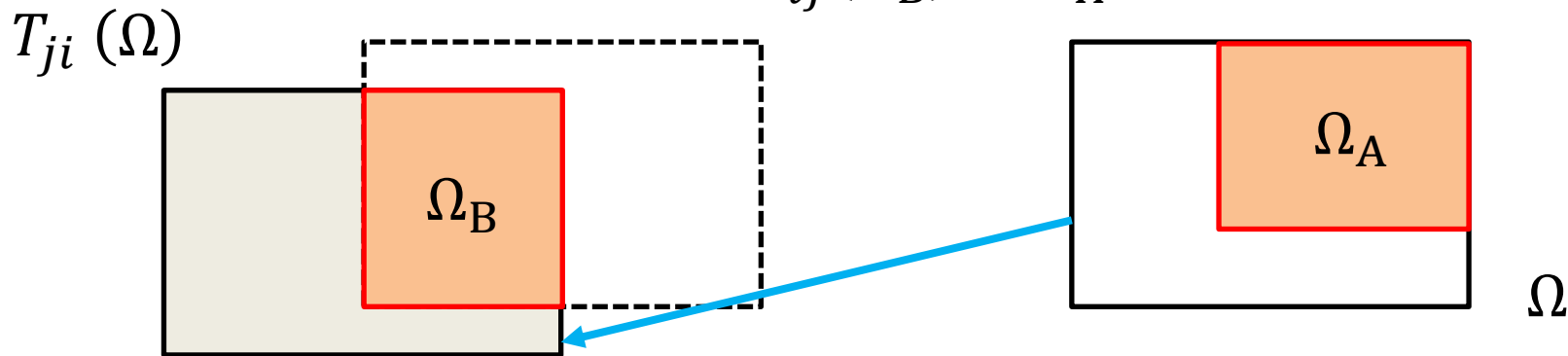
積分範囲をShift Map



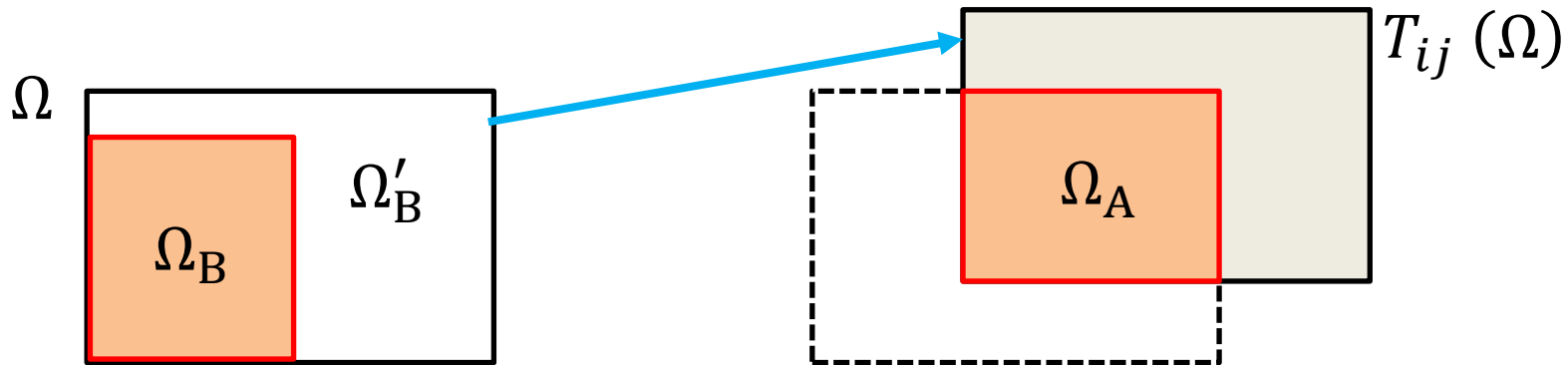
積分範囲をShift Map



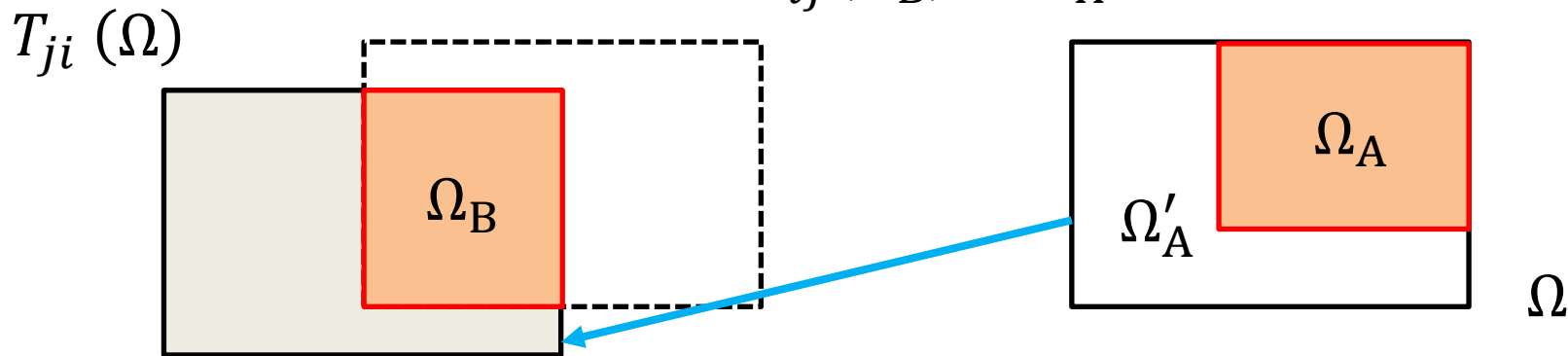
$$T_{ij}(\Omega_B) = \Omega_A$$



積分範囲をShift Map



$$T_{ij}(\Omega_B) = \Omega_A$$

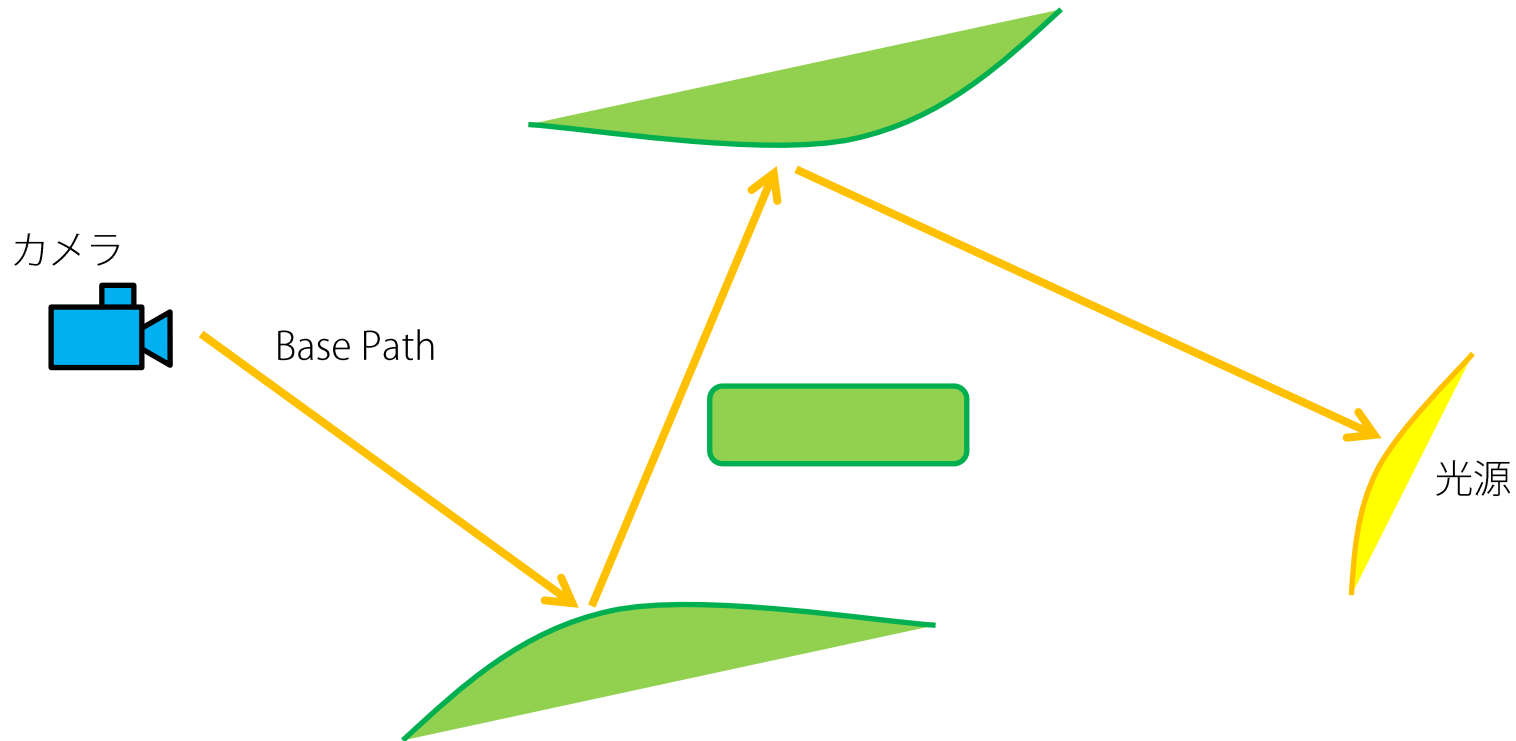


式の意味

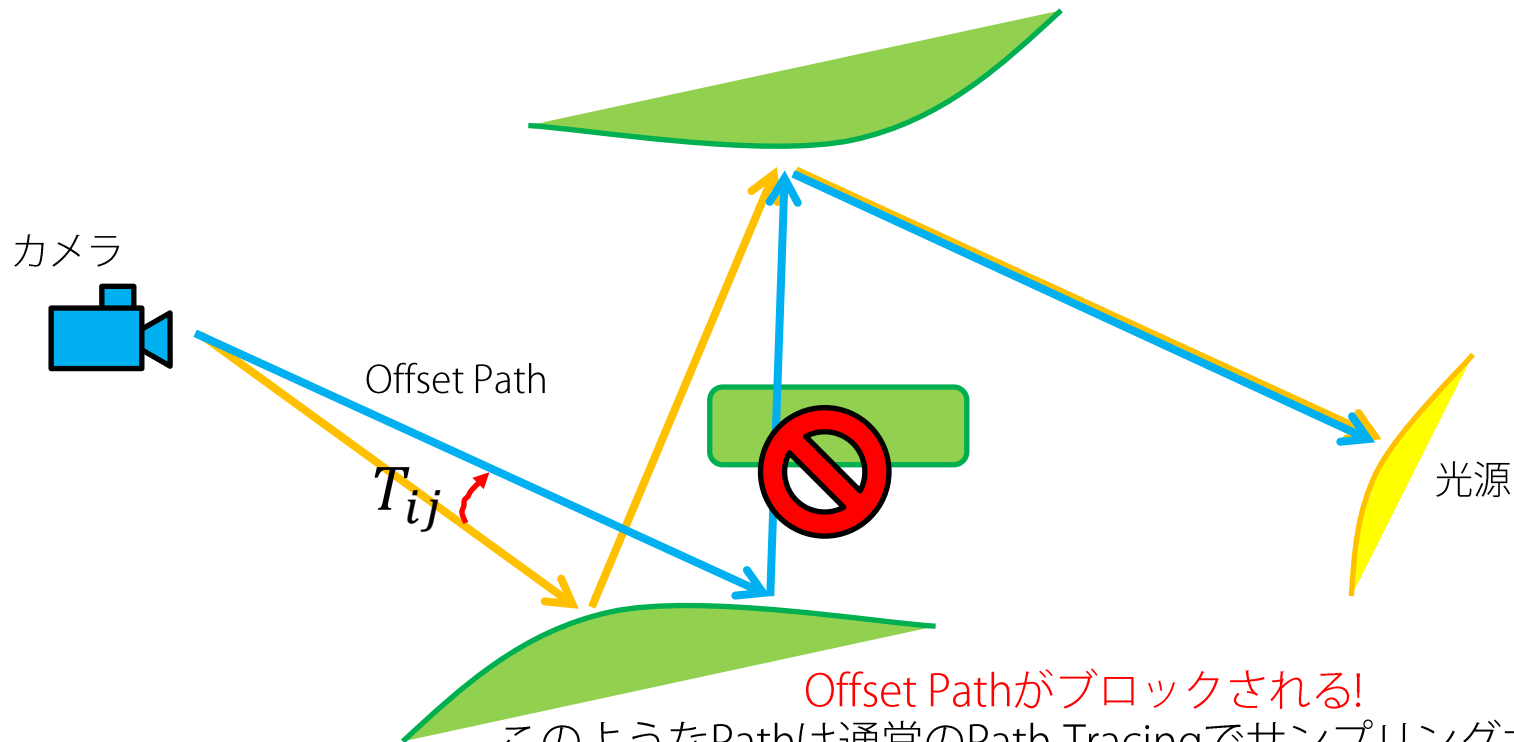
- 部分空間 $\Omega_A = \Omega_B$
 - T_{ij} が基本的には全単射になる。
 - 普通に解ける。（ダブルカウントしてしまうので w_{ij} と w_{ji} で重みづけ）
- それ以外の空間 Ω'_B
 - 射影先が Ω の外側になる。
 - Non symmetric
- 部分空間 Ω_0
 - 射影先が Ω の内側になるが、 T_{ij} が全単射にならない部分空間。
 - Non invertible

$$\Delta_{i,j} = \left(h(x) * \int_{\Omega} w_{ij}(x, \bar{p}) g_{ij}(x, \bar{p}) d\mu(\bar{p}) \right) (x_i) + \left(h(x) * \int_{\Omega} w_{ji}(x, \bar{p}) g_{ji}(x, \bar{p}) d\mu(\bar{p}) \right) (x_j).$$

Non symmetricな射影の例



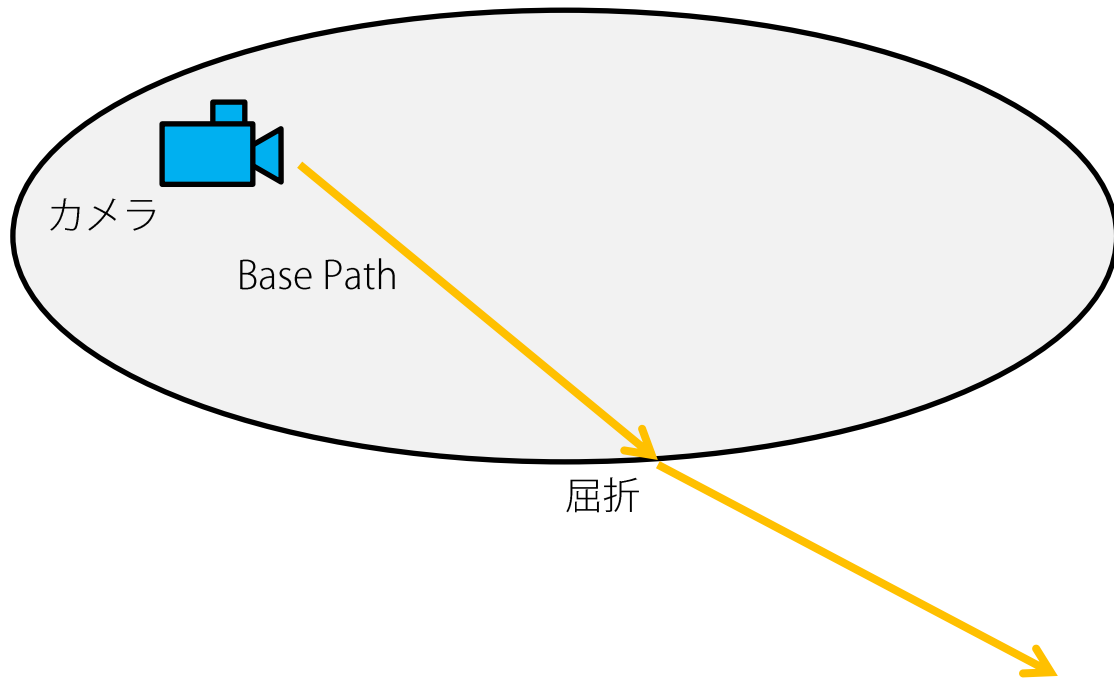
Non symmetricな射影の例



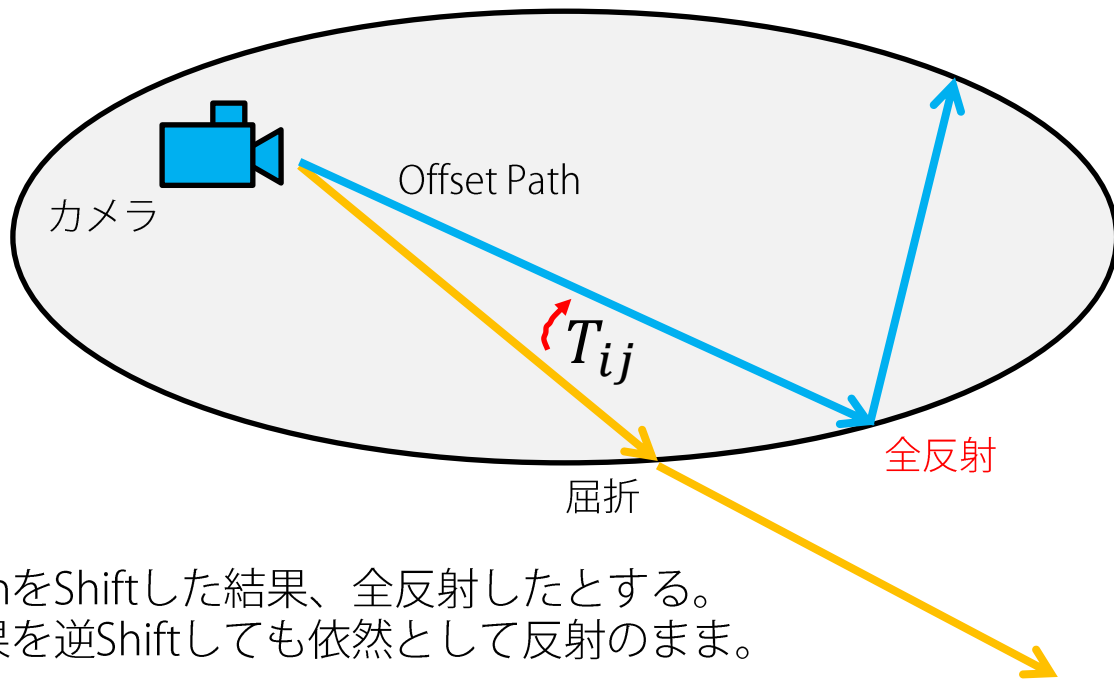
Offset Pathがブロックされる!

このようなPathは通常のPath Tracingでサンプリングされない
= Ω に含まれない

Non invertibleな射影の例



Non invertibleな射影の例



屈折するPathをShiftした結果、全反射したとする。
全反射した結果を逆Shiftしても依然として反射のまま。

式の意味

- 基本的には普通に g を評価する
 - Ω_A 、 Ω_B それぞれの上で積分を解くことになる（ので二つの積分項）
 - Non symmetricだったりNon invertibleだったりする場合は $w_{ij} = 1$ としてナイーブに勾配を計算する

$$\Delta_{i,j} = \left(h(x) * \int_{\Omega} w_{ij}(x, \bar{p}) g_{ij}(x, \bar{p}) d\mu(\bar{p}) \right) (x_i) + \left(h(x) * \int_{\Omega} w_{ji}(x, \bar{p}) g_{ji}(x, \bar{p}) d\mu(\bar{p}) \right) (x_j).$$

重みづけ

- Multiple Importance Samplingで重みづけ

$$w_{ij}(\bar{x}) = \frac{p(\bar{x})}{p(\bar{x}) + p(T_{ij}(\bar{x}))|T'_{ij}(\bar{x})|}$$



Pixels, 1spp



L_2 rec., with MIS



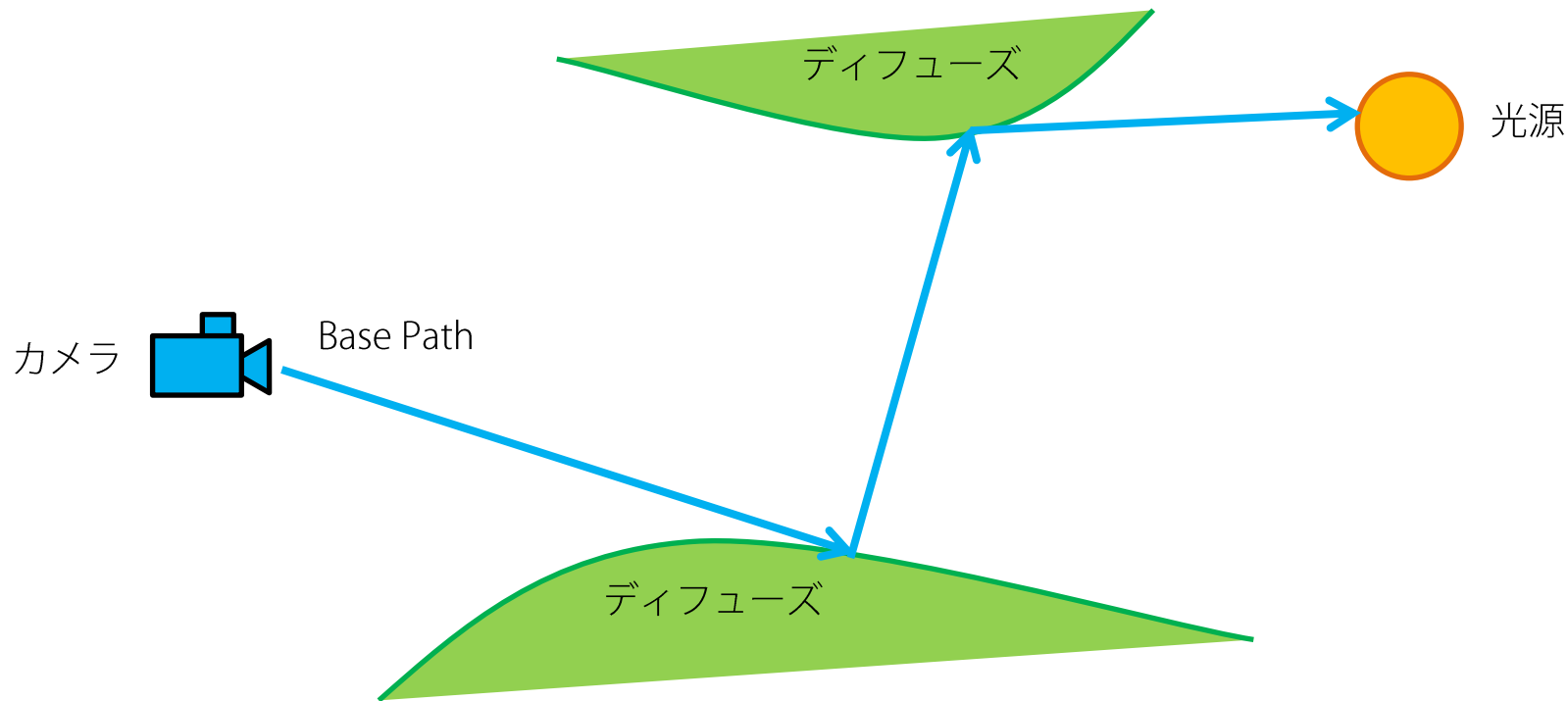
L_2 rec., without MIS

Gradient-Domain Path Tracing

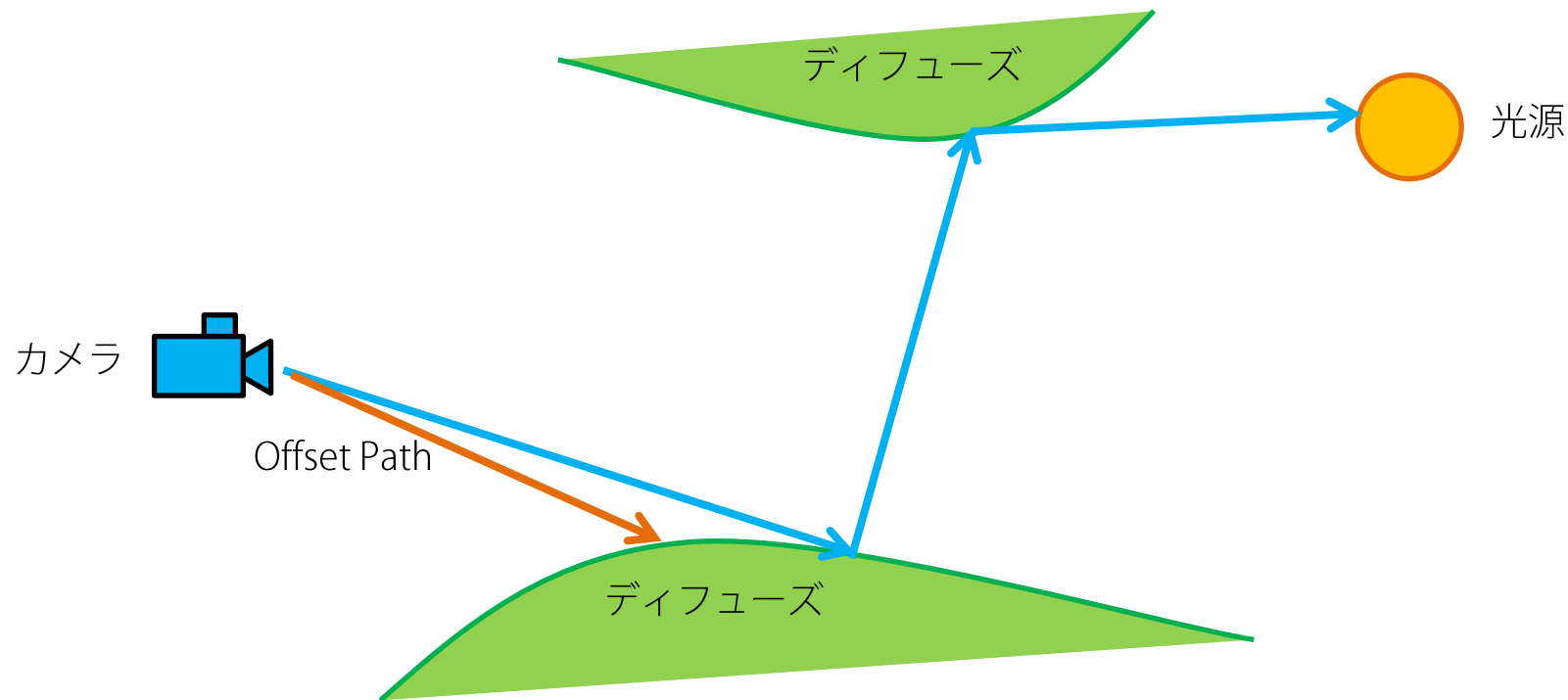
GD-PTのShift Map

- A Novel Shift Mapping for GD-PT
 - なるべくBase Pathに近い方が良い。
 - 1ピクセル分レイの方向をずらし、あとはなるべくBase Pathと同じ光路を辿るようにする。
 - スペキュラ（ガラスとか鏡）が絡む場合は、反射・屈折についてはBase Pathと同じになるようにしつつ、適切に追跡。

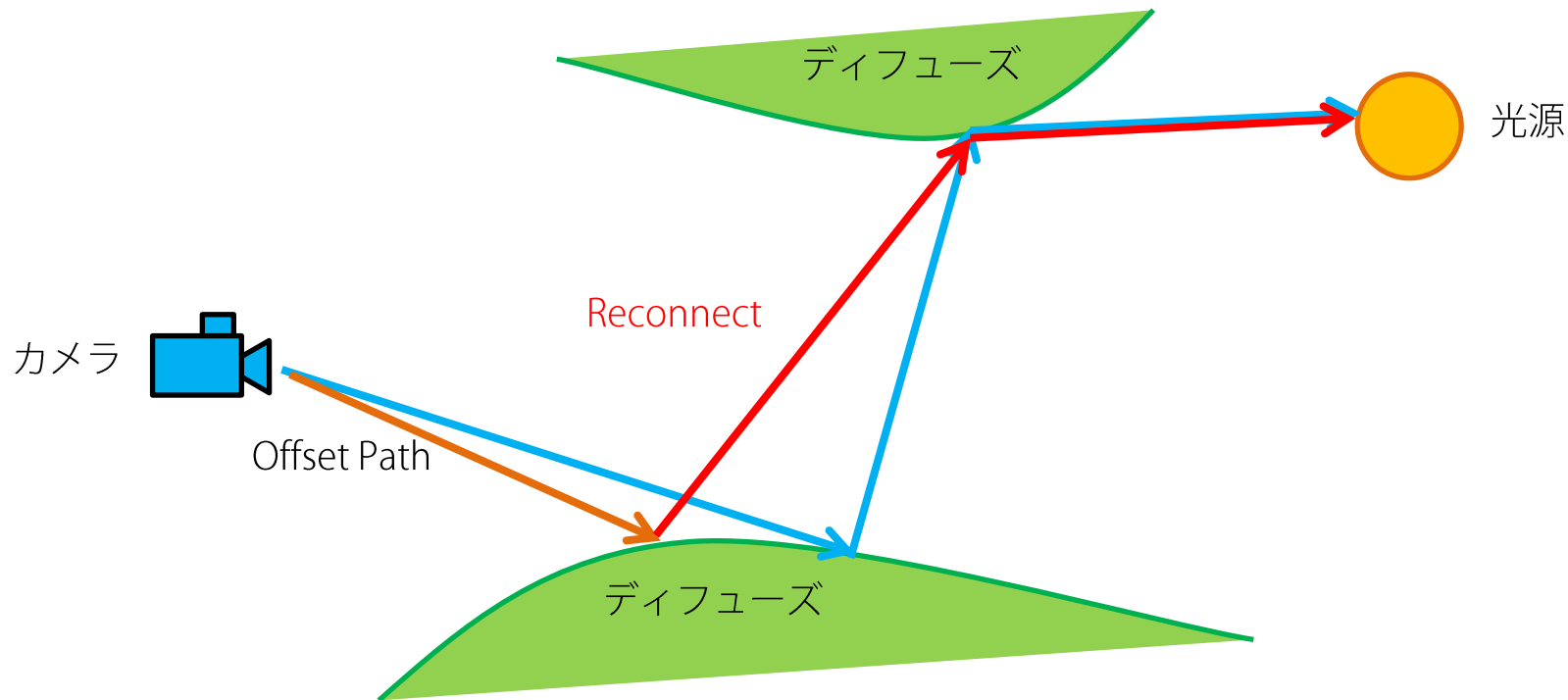
GD-PTのShift Map



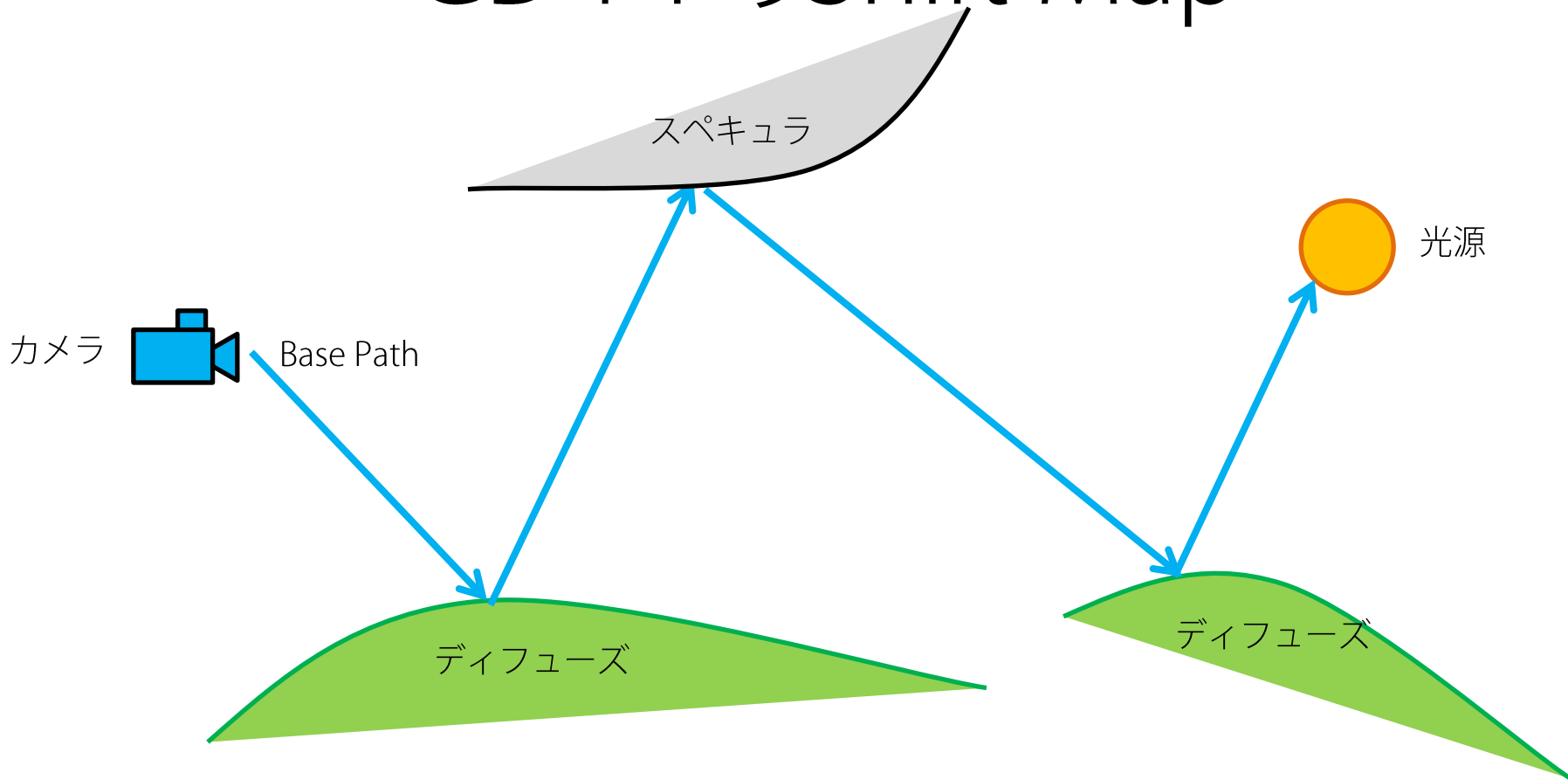
GD-PTのShift Map



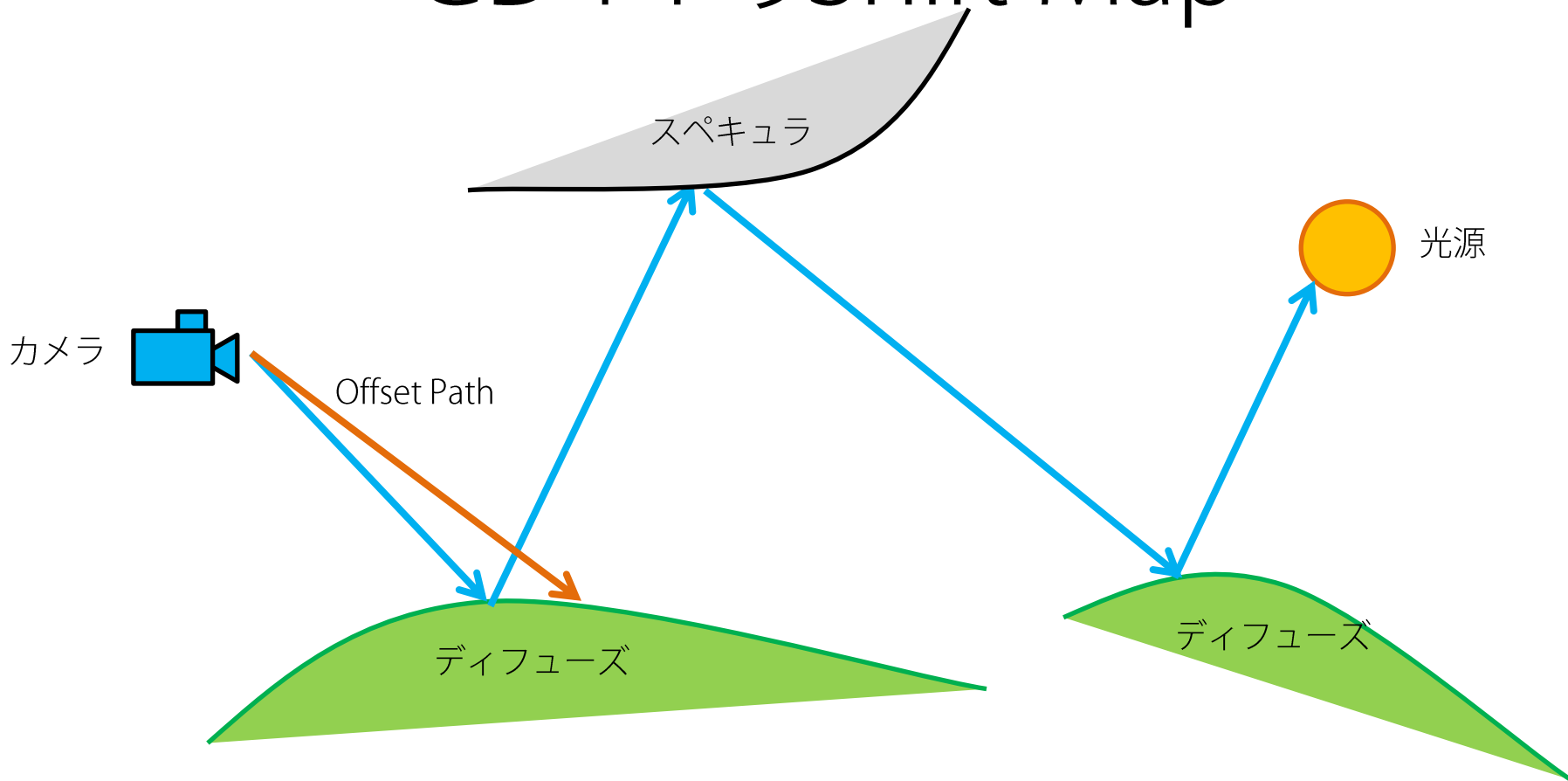
GD-PTのShift Map



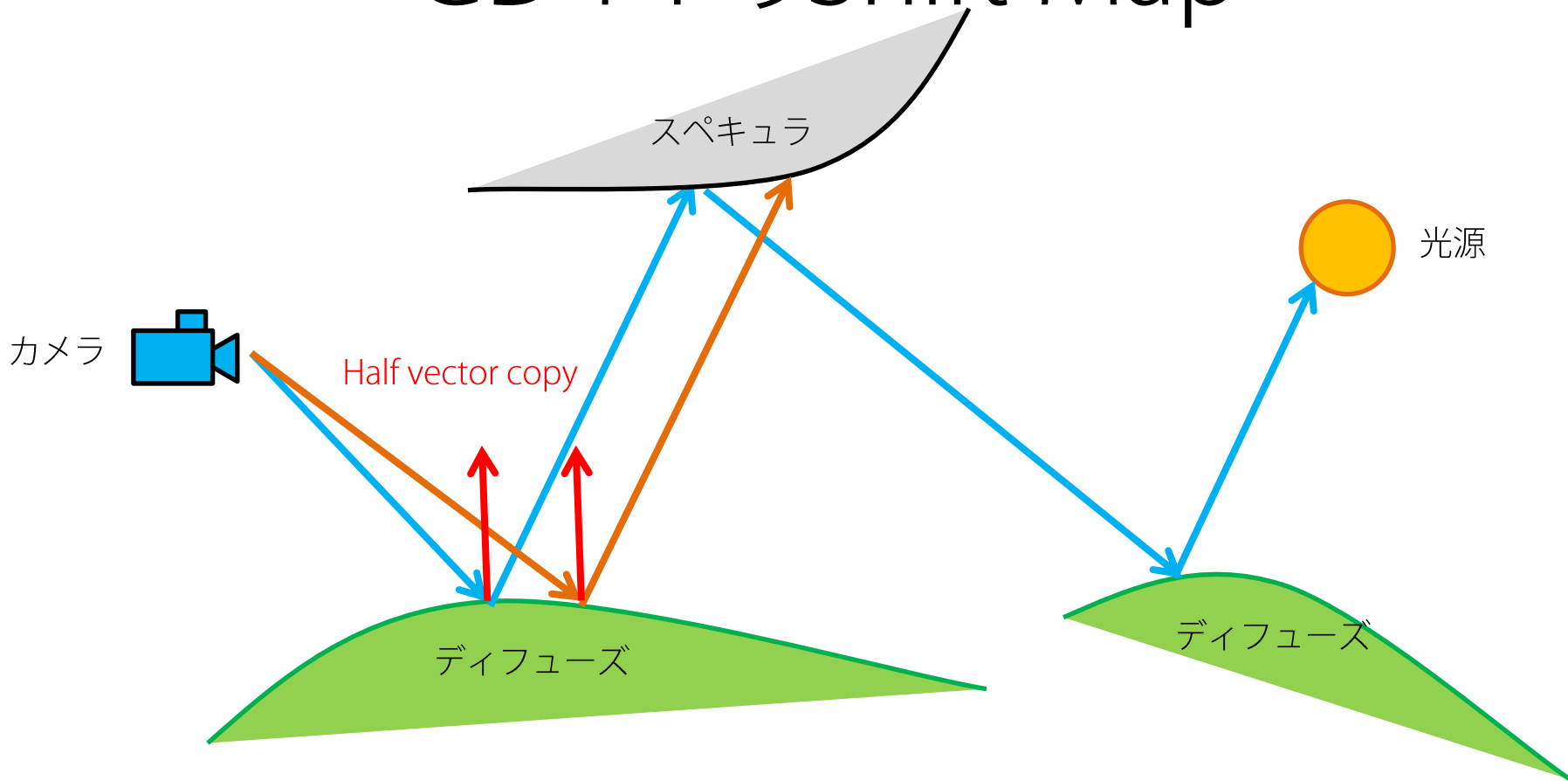
GD-PTのShift Map



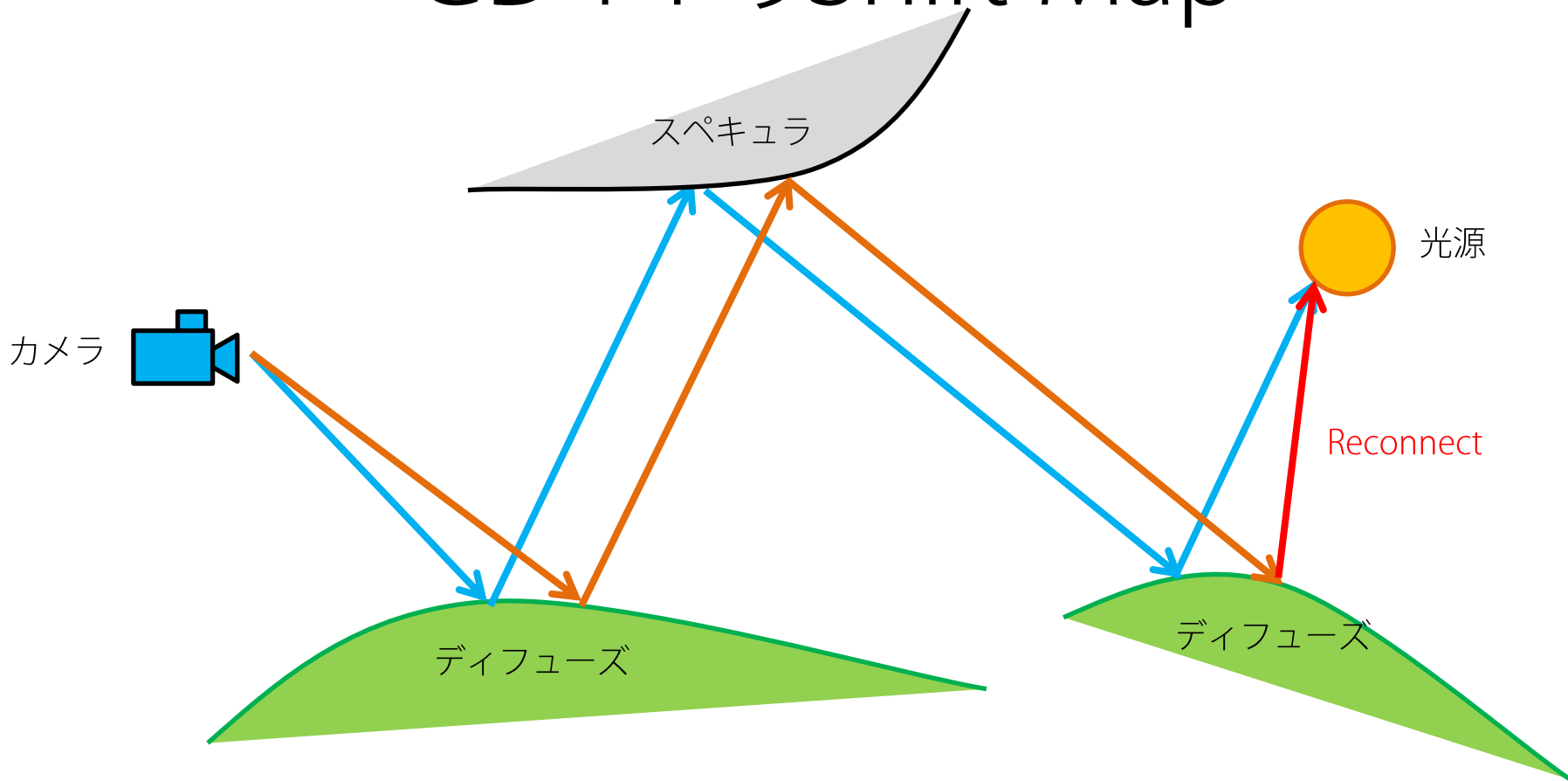
GD-PTのShift Map



GD-PTのShift Map



GD-PTのShift Map



Jacobians

$$\begin{aligned}\Delta_{i,j} &= \left(h(x) * \int_{\Omega} f(x, \bar{p}) - f(T_{ij}(x, \bar{p})) \boxed{|T'_{ij}|} d\mu(\bar{p}) \right) (x_i) \\ &= \left(h(x) * \int_{\Omega} g_{ij}(x, \bar{p}) d\mu(\bar{p}) \right) (x_i),\end{aligned}\tag{2}$$

Jacobians

計算するだけ！

$$\left| \frac{\partial \omega_i^y}{\partial \omega_i^x} \right| = \left| \frac{\partial \omega_i^y}{\partial \mathbf{h}^y} \right| \left| \frac{\partial \mathbf{h}^x}{\partial \omega_i^x} \right| = \frac{\omega_o^y \cdot \mathbf{h}^y}{\omega_o^x \cdot \mathbf{h}^x},$$

$$\left| \frac{\partial \omega_i^y}{\partial \omega_i^x} \right| = \left| \frac{\partial \omega_i^y}{\partial \mathbf{x}_2^y} \right| \left| \frac{\partial \mathbf{x}_2^x}{\partial \omega_i^x} \right| = \frac{\cos \theta_2^y}{\cos \theta_2^x} \frac{|\mathbf{x}_1^x - \mathbf{x}_2^x|^2}{|\mathbf{x}_1^y - \mathbf{x}_2^y|^2}.$$

$$\left| \frac{\partial \omega_i^y}{\partial \omega_i^x} \right| = \left| \frac{\partial \omega_i^y}{\partial \mathbf{h}^y} \right| \left| \frac{\partial \mathbf{h}^x}{\partial \omega_i^x} \right| = \frac{\left| \omega_i^y + \frac{n_2^y}{n_1^y} \omega_o^y \right|^2}{\left| \omega_i^x + \frac{n_2^x}{n_1^x} \omega_o^x \right|^2} \frac{\omega_i^x \cdot \mathbf{h}^x}{\omega_i^y \cdot \mathbf{h}^y},$$

Reconstruction

Reconstruction

- L1 or L2 Reconstruction
 - L2: 線形最適化になり、Unbiased
 - L1: 非線形最適化になり、Biased

$$\operatorname{argmin}_I \left(\left\| \begin{pmatrix} H^{dx} I \\ H^{dy} I \end{pmatrix} - \begin{pmatrix} I^{dx} \\ I^{dy} \end{pmatrix} \right\|_2^2 + \|\alpha (I - I^g)\|_2^2 \right).$$

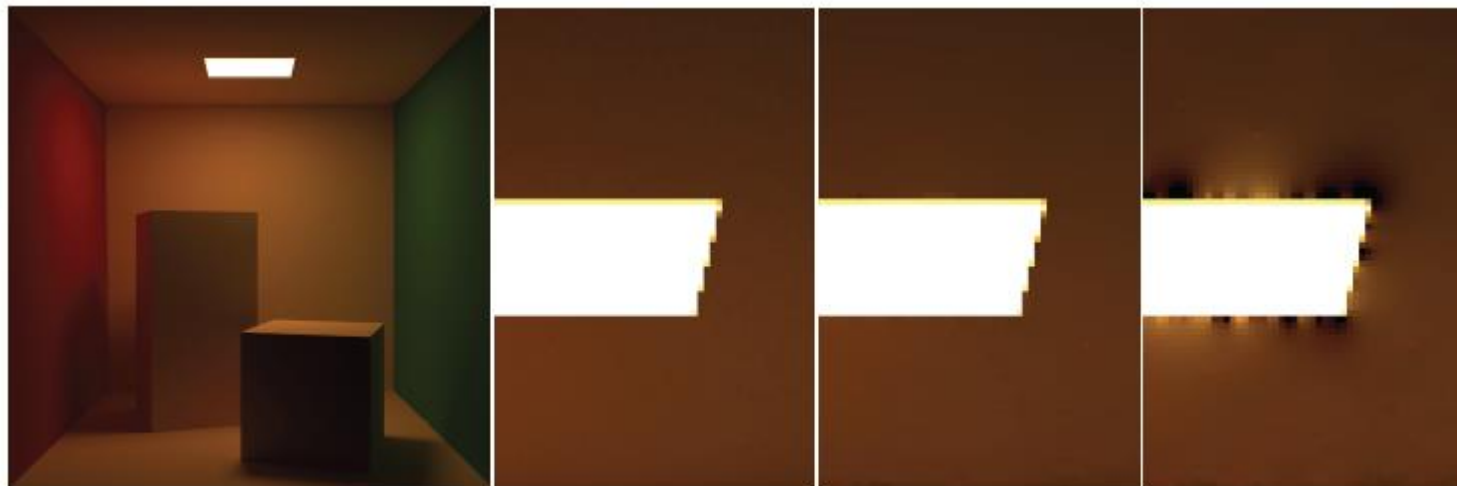
直接得られた
勾配画像

復元画像の勾配

復元画像と
Path Tracing画像の差

Reconstruction

- L1 or L2 Reconstruction
 - L2: 線形最適化でUnbiased
 - L1: 非線形最適化でBiased



Image

Ground truth

L^1

L^2

Improved Sampling for Gradient-Domain Metropolis Light Transport [Manzi 2014]

Results and Discussion

Scene

1spp

4spp

16spp

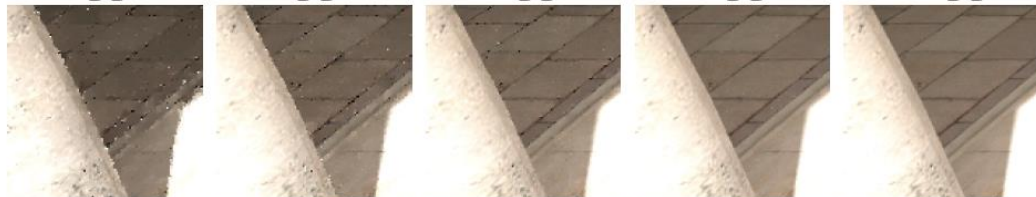
64spp

256spp

Sponza



G-PT



PT



Kitchen



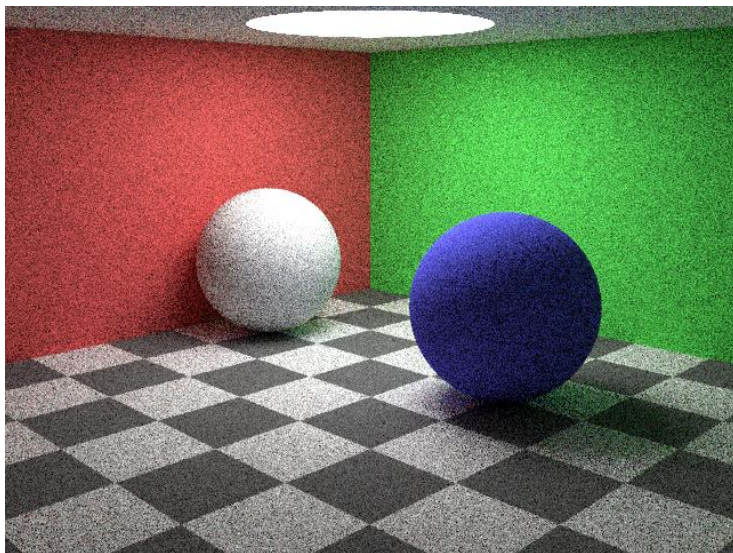
G-PT



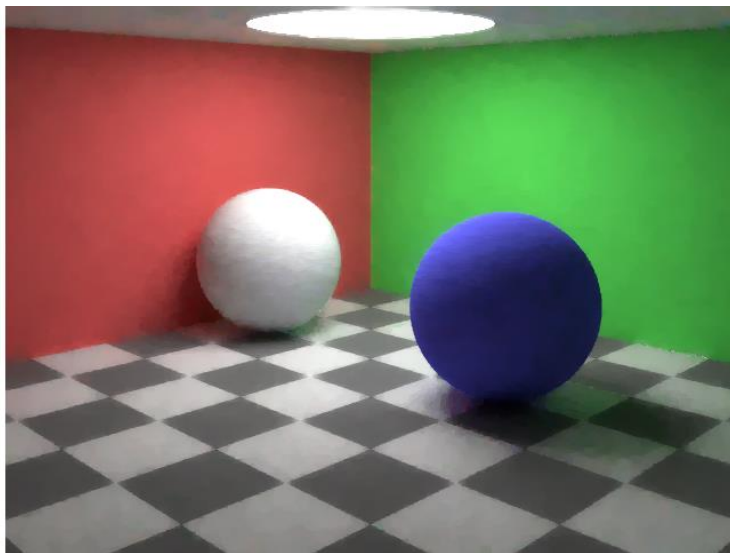
PT



Implementation



100 spp Path Tracing



GD-PT (L1)