Vol.2014-CG-154 No.2
2014/2/20

キャプチャにもとづいた照明 リアルタイムゲームのための実践的な大域照明

森重伸也[†1]

リソースが厳しいリアルタイムゲームにおいてアーティストが意図した大域照明を行うための実践的な方法を提案する．提案手法はシーンの間接照明をアーティストが配置したライトプローブの集合で構成した四面体の空間で表現する．ライトプローブはアーティストがシーン内で間接照明をキャプチャしたい任意の位置に配置する．四面体空間は，ドロネー三角形分割を使い，照明の補間は対象オブジェクトのワールド位置を含む四面体の重心座標系を使う．

従来の格子配置と直方体補間は，オーバーサンプリングとアンダーサンプリングの問題があり，それに比べて本手法は最小のライトプローブ数で最大の照明効果が得られ，シーンの静的なオブジェクトに対して動くオブジェクトの照明と陰影を馴染ませられる．さらにゲームで要求が多いアーティストが意図した照明と色をシーンから作り出せる．実験は，リソースが限られる WebGL と Web ブラウザで行った．提案手法が，従来手法の格子配置と直方体補間に比べて，最小で 20%のライトプローブ数で最大の間接照明効果が得られることを示す．ユーザ支援として Google Map Street View[a)]を使った現実空間からのキャプチャを紹介する．

Capture Based Lighting Practical Global Illumination for Real-Time Games

SHINYA MORISHIGE[†1]

1. はじめに

近年，リアルタイムゲームのグラフィックスにおいて，GPU の性能が向上し，BRDF や BSSRDF など物理ベースのシェーディングや光の伝播をシミュレーションする手法が提案されている[1]．それらの手法は，現在主流のスマートフォン，タブレット，携帯ゲーム機器，今後の Web ブラウザベースのゲームで描画 60fps/30fps を達成しようとした場合，GPU コストが割高で処理落ちやティアリングが発生してしまい，ユーザ体験が損なわれてしまいがちである．一方で, 実際の制作では, 制作期間と開発資源が限定された条件下でアーティストが意図した照明と陰影づけが求められる場合が多い．さらに大規模プロジェクトでかつ多種多様なシーンを制作する場合, 参加者全員が照明と陰影について経験豊富ではない状況が発生する．具体的には， Final Gathering や Photon Mapping などを使って意図した照明と陰影づけを行う状況である．リアルタイムゲームにおけるこれらの手法の問題点は，動くオブジェクトへのリアルタイム照明と高品質で自然な照明を行うための調整コストが挙げられる．特に DCC ツールで調整した後に再度，実際の動作環境上でも調整が必要となる問題がある．このような事情から，リアルタイムゲームでの照明と陰影は下記の要件を満たすことが必要である．

- 空気感をもたらす滑らかな照明と陰影の変化
- 静的と動的オブジェクトの照明と陰影が自然に馴染む
- 実機の GPU が出力する色調を再現できる
- アーティスト・フレンドリで照明と陰影が調整可能
- 描画 60fps/30fps を実現できる GPU コスト

提案手法は，これらの要件を満たすために，Web ブラウザ上で動作する in-game のシーンからアーティストが意図した照明をキャプチャして，オブジェクトに適用する．

2. アルゴリズム

間接照明をキャプチャして、オブジェクトに適用するまでの流れは，次の通りである．

2.1 ライトプローブを任意配置する
2.2 間接照明をキャプチャする
2.3 シーンを空間分割する
2.4 間接照明の補間を行う
2.5 間接照明をオブジェクトに適用する

2.1 ライトプローブを任意配置する

アーティストが間接照明をキャプチャしたい場所にライトプローブを配置する．

図 1　ライトプローブの配置

†1 Independent 個人

a) Google Map Street View は，米国 Google Inc.の米国およびその他の国における商標または登録商標です．

ⓒ2014 Information Processing Society of Japan

ライトプローブとは，キャプチャした間接照明を球面調和関数で展開してコンパクトにしたものである．6 枚のキャプチャ画像を 9 個の三次元実数ベクトルにして，SH 定数として保持する．

配置は，シーンの背景ポリゴンモデルの頂点位置から自動生成するか，アーティストに DCC ツール上でロケーターを配置してもらう．その他に Google Map Street View[2]を使って、現実世界の写真から直観的に生成する．

図 2　ライトプローブの配置（Google Map Street View）

2.1.1 トポロジに意味を持たせる

ライトプローブの配置は、そのトポロジに意味を持たせることで，アーティストは直観的な配置が行える．

具体的には図 3 に示すように，ステージの上空に sky light，地面に bounce light，背景オブジェクトからの color bleeding といった，ライトプローブに意味を持たせて配置できる．

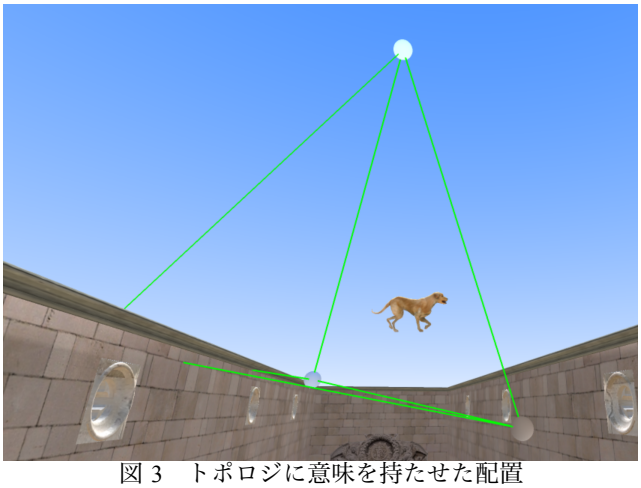

図 3　トポロジに意味を持たせた配置

2.2 間接照明をキャプチャする

2.1 で配置した位置で環境マップを作成する．

環境マップは，キューブマップで 6 枚のテクスチャで構成される．リアルタイムで照明するために，間接照明の低周波成分をテクスチャから球面調和関数で展開して，SH 定数として，9 個の三次元ベクトルとして保持する．図 4 の画像内で右端にキャプチャ画像，その画像を SH 復元した結果を示す．

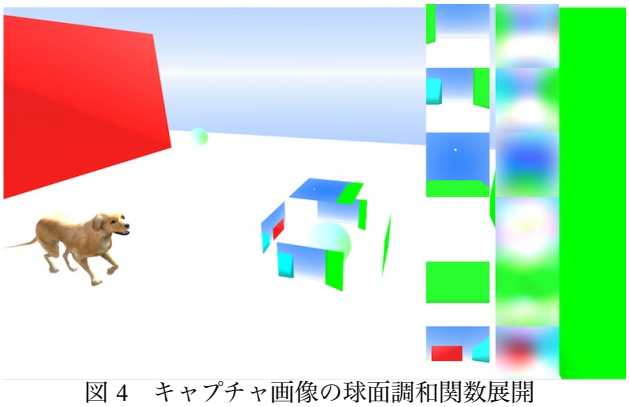

図 4　キャプチャ画像の球面調和関数展開

キャプチャした環境マップからの球面調和関数展開は下記の手順で行う．

Step 1. サンプリング方向の計算
Step 2. サンプリング方向の SH 基底の計算
Step 3. SH 定数の数値積分
for each(キューブマップの全ての面 face) {
 for each(面の全てのテクセル $L_{face}(u,v)$) {
  Step 3.1. サンプリング方向の取得
  Step 3.2. SH 基底の取得
  Step 3.3. テクセルのサンプリング
  Step 3.4. SH 定数の積分
 }
}
Step 4. SH 定数の正規化

2.2.1 SH 定数の数値積分

参考文献[4]にもとづいて下記の近似式で計算する．

$$L_l^m \approx \sum_{face=1}^{6}\sum_{v=1}^{h}\sum_{u=1}^{w} L_{face}(u,v)\,Y_l^m(\omega)\,A(\omega)$$

$$A(\omega) \approx \frac{\vec{r}\cdot\vec{n}}{\|\vec{r}\|^3}\,dS$$

$$dS \approx \left(\frac{2}{h}\right)\cdot\left(\frac{2}{w}\right)$$

L_l^m　SH 定数
$face \in \{1,2,3,4,5,6\}$　キューブマップ全ての面
$L_{face}(u,v) \in [1,w]\times[1,h]$　キューブマップテクセル
$Y_l^m(\omega)$　球面調和関数

立体角 A(ω)はテクセルを正方形とみなした近似式を使う．
dS は立体角の微小面積でここではテクセルの面積である．
w と h はキューブマップ画像の解像度で 8 から 256 程度までとする．
r はキャプチャ位置からキューブマップ面へのサンプリング方向である．
n はキューブマップの面法線とする．

ⓒ2014 Information Processing Society of Japan

2.2.2 現実世界の写真画像から間接照明をキャプチャする

in-game のシーンだけでなく，Google Map Street View やスマートフォンのデジタルカメラを使って，現実世界の写真画像からキャプチャして SH 定数を計算することもできる．

図 5　間接照明のキャプチャ

2.3　シーンを空間分割する

step 1.で配置したライトプローブを頂点集合とみなして，ドロネー三角形分割を行う．従来手法は、図 1 に示すようにライトプローブを格子状に配置していた．しかしながら，この方法は，下記の問題がある．

- オーバーサンプリングあるいはアンダーサンプリング
  間接照明の情報源となる地形や建物等のモデルデータはシーン内に任意配置されることが多く，格子配置ではアーティストが意図しないサンプリングが行われてしまう．四面体で空間分割すれば，アーティストが意図した場所のみを重点的にサンプリングできる．

- ライトプローブの数
  リアルタイム性の高いゲームでは出来る限りメモリ消費量を抑えることが求められる．例えば，幅方向に 4 個，高さ方向に 4 個，奥行き方向に 4 個，それぞれ配置しただけでも 64 個必要である．

四面体分割を行うことによって，最小のライトプローブ数で最大の照明効果が得られる．四面体分割は，参考文献[12]の外接球を使った逐次外挿のアルゴリズムを使う．

2.4　間接照明の補間を行う

間接照明を適用したいオブジェクトのワールド位置に応じて，照明と陰影の補間を行う．照明の補間は，参考文献[11]にもとづいて行う．図 6 のようにオブジェクトのワールド位置を含む四面体の重心座標系を計算して，その重心座標を補間の重みとみなして補間する．四面体を構成する頂点は，ライトプローブである．

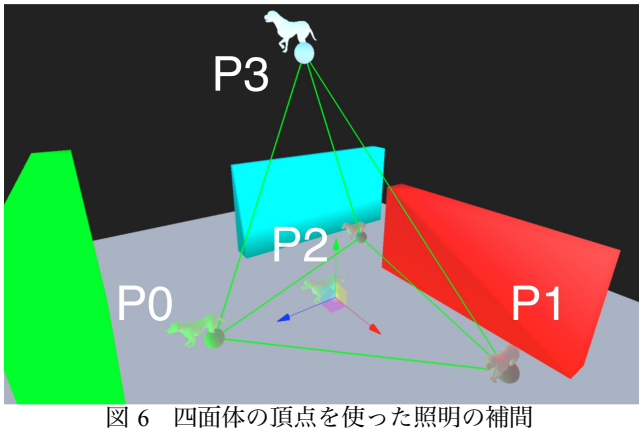

図 6　四面体の頂点を使った照明の補間

重心座標系の計算は，

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \left[\vec{p}_0 - \vec{p}_3 \quad \vec{p}_1 - \vec{p}_3 \quad \vec{p}_2 - \vec{p}_3 \right]^{-1} \left[\vec{p} - \vec{p}_3 \right]$$

$\begin{bmatrix} a & b & c & d \end{bmatrix}$　：重心座標系

ただし，$d = 1 - (a + b + c)$　とする．

$\vec{p}$　：オブジェクトのワールド位置

$\vec{p}_0, \vec{p}_1, \vec{p}_2, \vec{p}_3$　：四面体を構成するライトプローブのワールド位置（図 6）

とする．

重心座標計算式の右辺で P を含まない行列は，事前に 1 度だけ計算すればよい．in-game で四面体は変化しないからである．格子状配置の直方体はライトプローブを 8 個を使うのに対して，四面体なら，任意のトポロジで，かつより少ない 4 個のライトプローブで照明の補間が行える．

2.5　間接照明をオブジェクトに適用する

2.4 で計算した補間の重みと SH 定数を使って，下記の手順で，間接照明をオブジェクトに適用する．

Step 1. SH 定数を線形補間する
Step 2. SH 定数から低周波の色を復元する
Step 3. オブジェクトの拡散反射に復元した色をブレンドする

Step 2.復元の計算は，参考文献[7]にもとづいた式を使う．
リアルタイム性の高いゲームでは処理負荷の観点から低周波のみを復元する．そうではない場合は，中周波と高周波も復元することができる．今回は，低周波のみを復元する．

$$E(\vec{n}) = \vec{n}^t M \vec{n}$$

$\vec{n}$　は，オブジェクト表面のワールド法線である．

計算の便宜上，$\vec{n}^t = \begin{bmatrix} x & y & z & 1 \end{bmatrix}$　とする．

per-vertex，per-pixel のどちらでも計算可能である．

M　は 9 個の SH 定数で構成した 4x4 行列である．

ⓒ2014 Information Processing Society of Japan

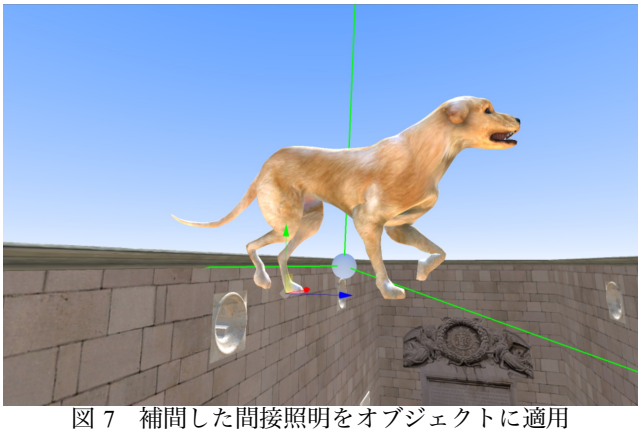

図 7　補間した間接照明をオブジェクトに適用

図 7 は，オブジェクトの真上にスカイライト用のライトプローブ，オブジェクトの足もとにバウンスライト用のライトプローブを配置して，それらを四面体補間して，オブジェクトに適用した例である．ライトプローブの配置は，図 3 に示す通りである．

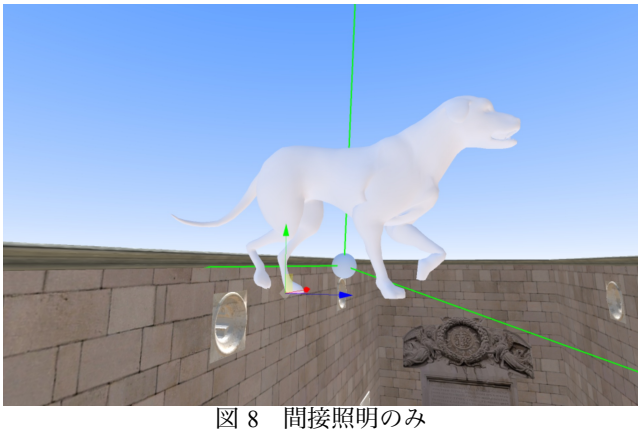

図 8　間接照明のみ

図 8 は，間接照明のみの効果を示している．オブジェクトは,スカイライトの青みがかった効果を少しだけ受けながら,地面のバウンスライトの効果によって空気感が表現できている．

図 9　平行光源のみ

図 9 は，平行光源と半球ライティングのみの照明効果を示している. オブジェクト全体が明るく白とびしており，シーンと馴染んでいないのが確認できる．

3. 実験

屋内と屋外のシーンで，従来手法と提案手法の比較実験を行った結果を表に示す．
従来手法は，格子配置と直方体補間である．
提案手法は，任意配置と四面体補間である．

3.1 シーン

表 1 のシーンとキャラクタ dog trot[15]で従来手法と提案手法の比較実験を行った．

表 1　シーン一覧

シーン名	内容
Basic	赤、緑、青の壁を使った四面体補間の確認
Sponza	屋外と屋内の間接照明の確認 [14]
Sibenik	屋内の間接照明の確認 [14]
Kyoto	写真からのキャプチャ

3.2 実験環境

表 2 に示す動作環境で実験を行った.ライトプローブのキャプチャ，シーンの四面体分割，Real-Time Shading は WebGL を使ったフレームワーク Three.js に追加拡張として実装した．実験は Web Browser 上で行った．キャプチャ解像度は R8G8B8 の 128x128 pixel とした．

表 2　実験環境

項目	内容
PC	MacBook Pro Retina 13inch Late 2012
CPU	Intel Core i5 2.5 GHz
GPU	Intel HD Graphics 4000 1024MB
Memory	8GB 1600 MHz DDR3
OS	Mac OSX 10.9.1
Web Browser	Safari 7.0.1
Google Map	version 3.0
WebGL	version 2.0 [3]
Framework	Three.js [13]

3.3 実験結果

表 3 にキャプチャで使用したライトプローブの数をまとめる．表 4 に生成した直方体と四面体の数をまとめる．
Basic では，従来の格子配置（図 10）に比べて，提案手法の配置（図 11）では，ライトプローブの数を約 80%減らすことができた．見た目の品質は，格子配置，提案手法ともに Color Bleeding が確認できた．

屋外と屋内が含まれる広域の Sponza では，従来の格子配置（図 15）に比べて，提案手法の配置（図 16）では，ライトプローブの数を約 80%減らすことができた．見た目の品質は，格子配置は均等間隔のために陰影が変化する角のキャプチャがうまく出来なかった．（図 17）任意配置では，こちらが意図した場所で陰影をキャプチャしたライトプローブを追加して，角のキャプチャをうまく実現できた．（図 18 から図 23）屋上では SkyLight 用のライトプローブ 1 つをキャプチャするだけで青みを実現できた．（図 24）

ⓒ2014 Information Processing Society of Japan

屋内の Sibenik は，格子配置（図 25）に比べて，提案手法の配置（図 26）では，ライトプローブの数を約 90%減らすことができた．Sponza と同じく，角のキャプチャで格子配置（図 27）に比べてうまく実現できた．（ 図 28，図 29）

Google Street Map View[2)]を使ったパノラマ写真からのキャプチャのみとした．平面地図のため空間分割は行っていない. 格子配置（図 30）に比べて，提案手法（図 31）ではキャプチャしたい場所を重点的にキャプチャできた．（図 32）

表 3　ライトプローブの配置数

シーン名	格子配置	任意配置
Basic	64 個 (4x4x4)	13 個
Sponza	432 個 (12x6x6)	85 個
Sibenik	432 個 (12x6x6)	39 個
Kyoto	20 個(5x4)	12 個

表 4　直方体と四面体の数

シーン名	直方体の数	四面体の数
Basic	8 個	14 個
Sponza	54 個	401 個
Sibenik	54 個	124 個
Kyoto	空間分割なし	空間分割なし

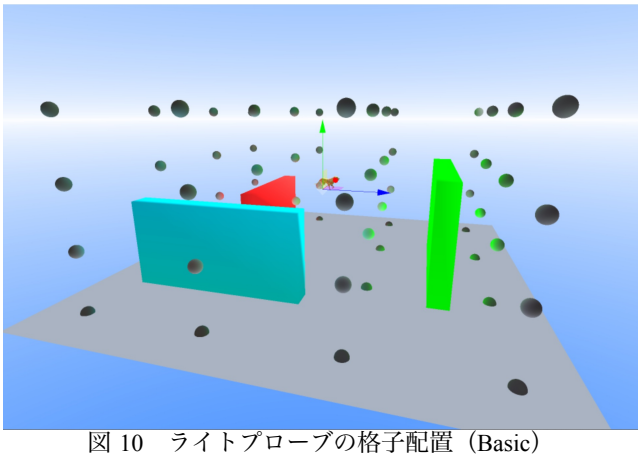
図 10　ライトプローブの格子配置（Basic）

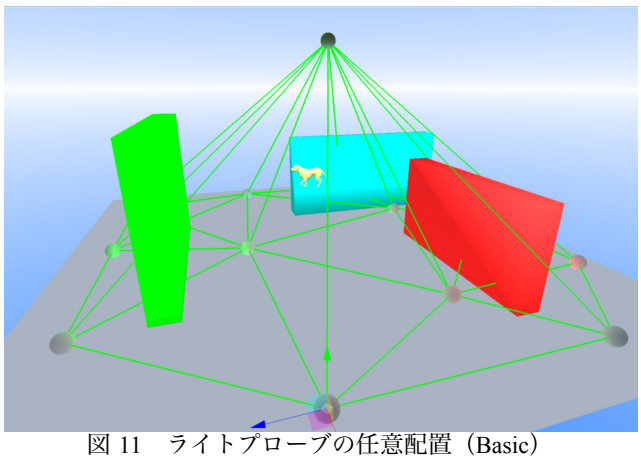
図 11　ライトプローブの任意配置（Basic）

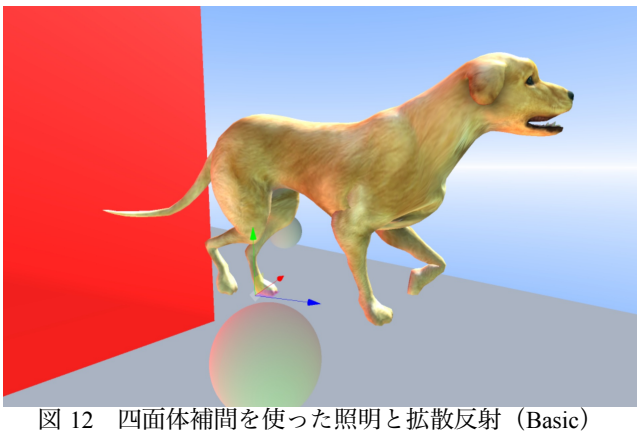
図 12　四面体補間を使った照明と拡散反射（Basic）

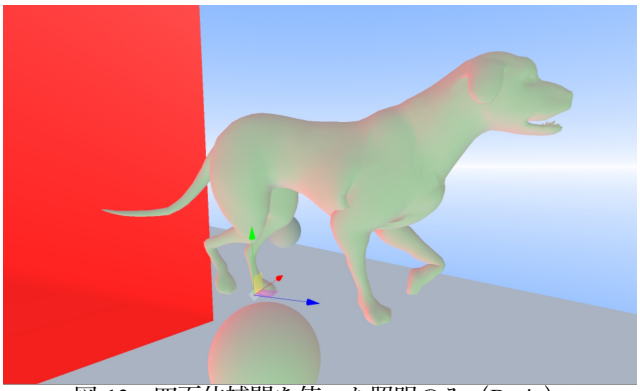
図 13　四面体補間を使った照明のみ（Basic）

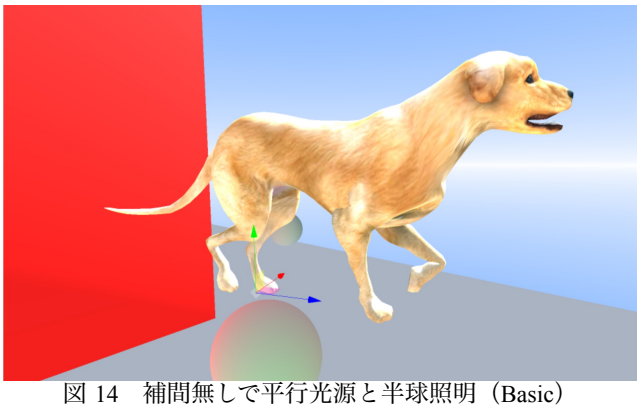
図 14　補間無しで平行光源と半球照明（Basic）

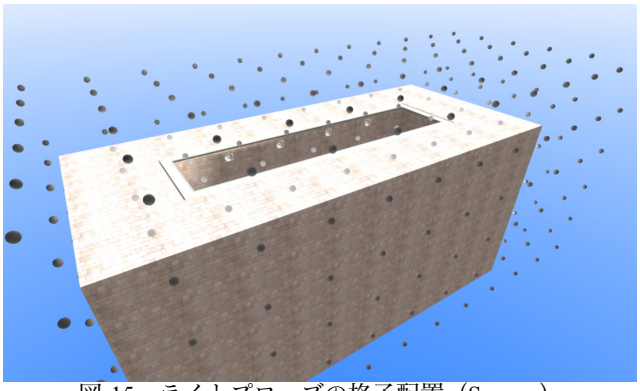
図 15　ライトプローブの格子配置（Sponza）

ⓒ2014 Information Processing Society of Japan

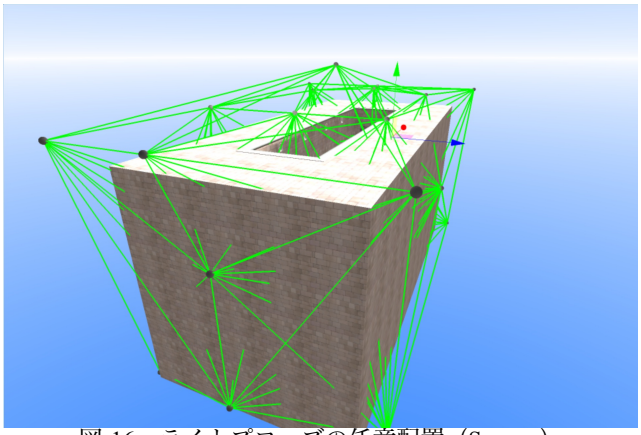
図 16　ライトプローブの任意配置（Sponza）


図 17　格子配置の角キャプチャ（Sponza）

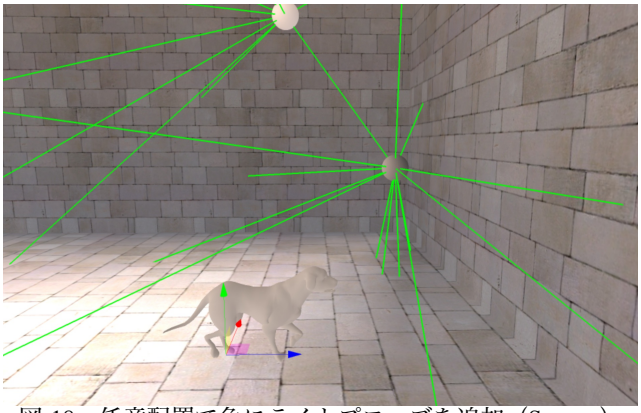
図 18　任意配置で角にライトプローブを追加（Sponza）

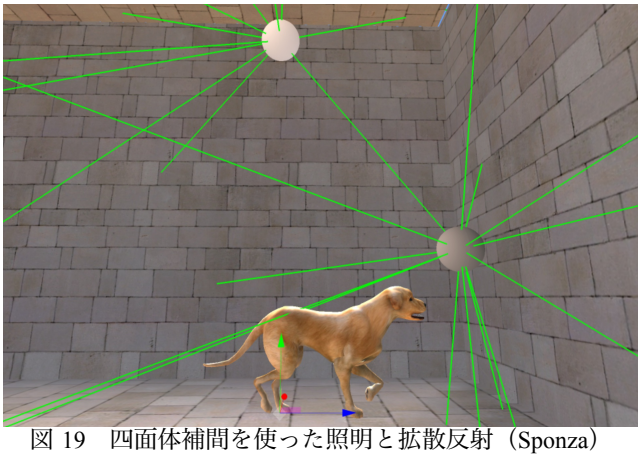
図 19　四面体補間を使った照明と拡散反射（Sponza）

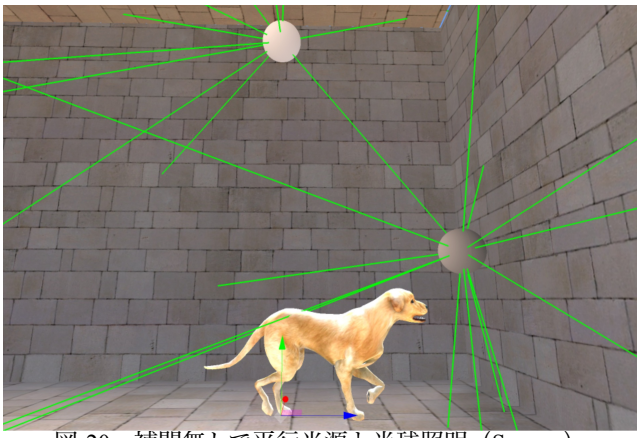
図 20　補間無しで平行光源と半球照明（Sponza）

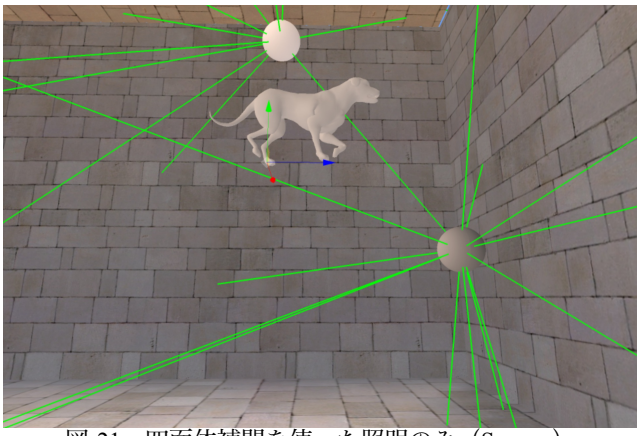
図 21　四面体補間を使った照明のみ（Sponza）

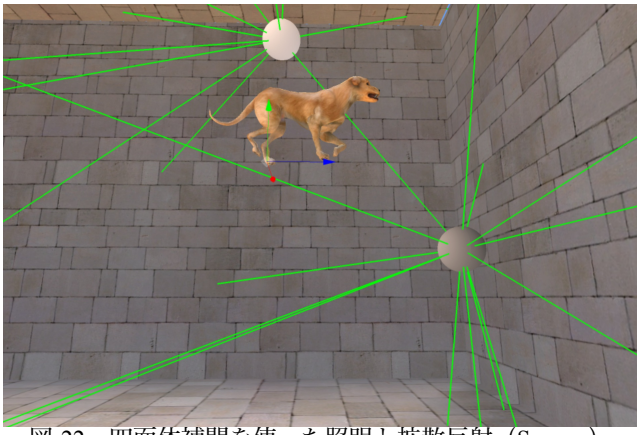
図 22　四面体補間を使った照明と拡散反射（Sponza）

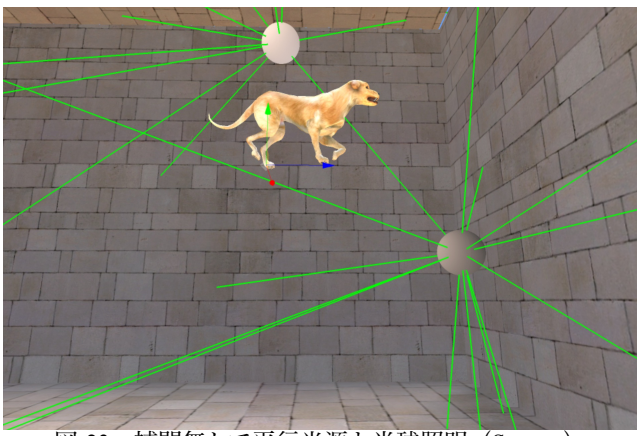
図 23　補間無しで平行光源と半球照明（Sponza）

ⓒ2014 Information Processing Society of Japan

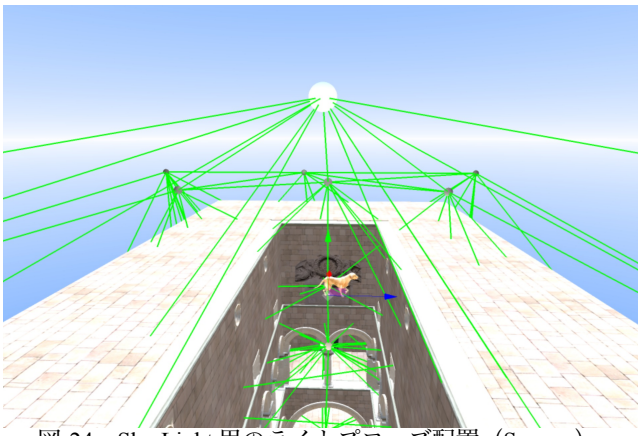

図 24　Sky Light 用のライトプローブ配置（Sponza）

図 27　格子配置での照明のみ（Sibenik）

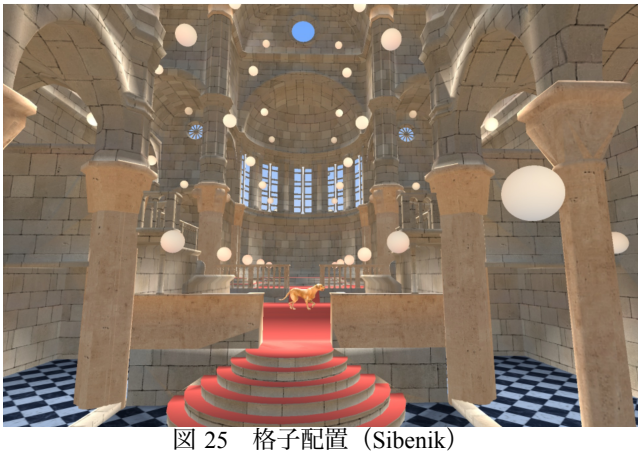

図 25　格子配置（Sibenik）

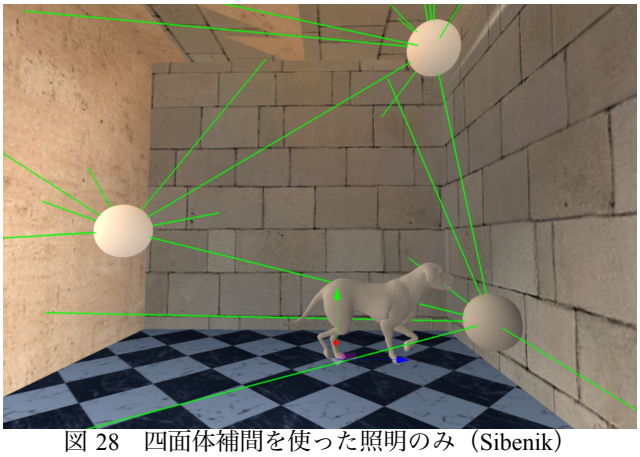

図 28　四面体補間を使った照明のみ（Sibenik）

図 26　任意配置（Sibenik）

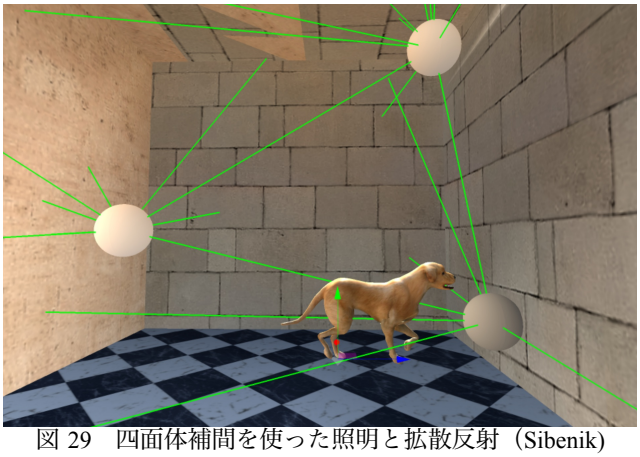

図 29　四面体補間を使った照明と拡散反射（Sibenik）

図 30 格子配置（Kyoto）

ⓒ2014 Information Processing Society of Japan

図 31 任意配置（Kyoto）


図 32 四面体補間（Kyoto）

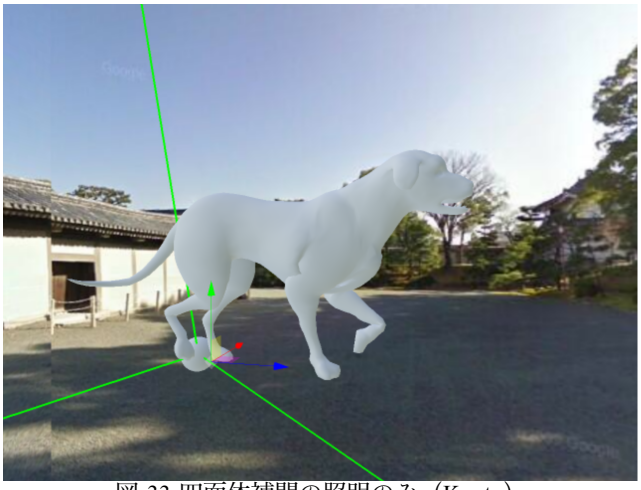
図 33 四面体補間の照明のみ（Kyoto）

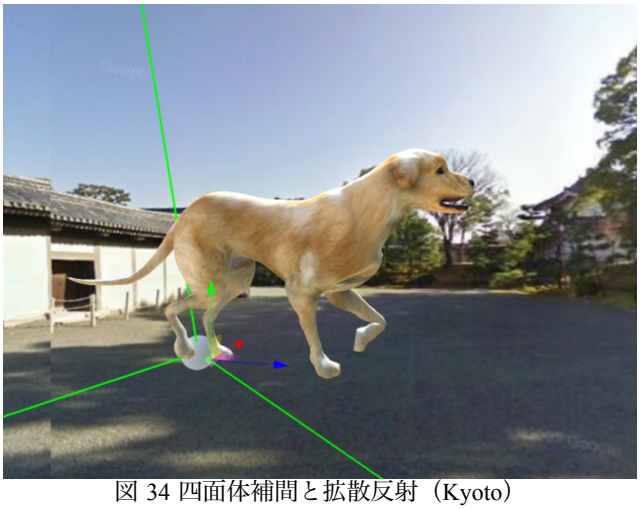
図 34 四面体補間と拡散反射（Kyoto）

3.4 考察

格子配置は，自動配置に適したやり方である．あるライトプローブの近傍に配置されたライトプローブのキャプチャ結果は，類似した内容が多いのが確認できる．（図 15）実験結果の表 4 から直方体の数に比べて四面体の数は多くなる傾向がわかる．

任意配置と四面体分割は，ユーザが重点的サンプリングを手動で行っているとみなせる．格子配置で類似したライトプローブをまとめている．実際の運用では格子配置あるいは無作為抽出を行って，そこからユーザが任意配置で最適化していく手順になる．ライトプローブのキャプチャ結果に対する類似度を定義してクラスタリングを行い，格子配置から任意配置に変換する方法も考えられる．

四面体による空間分割は凸形状になってしまう．非凸形状に対応するためには照明の範囲でグループに分割することが必要になる．四面体分割で構築した情報は，間接照明だけでなく，AI の経路探索にも使うことができる．Real-Time Game では望ましい性質である．キャプチャの範囲設定は少なくとも三種類必要である．Sky Light と Bounce Light とオブジェクトからの反射である．

Google Map Street View など写真からのキャプチャでは，撮影時の天候や時刻，カメラなど照明環境が多様なので何らかの前処理が必要になる．Computer Vision における光学的な Camera Calibration であり，照明環境を仮定して抽出する一種の逆問題だとみなせる．今回は低周波のみのキャプチャだったのでそこまで顕著な差は観られなかった．

4. おわりに

今回は低周波のみで照明環境のキャプチャとその適用を行った．今後の課題として
・環境遮蔽、陰影のキャプチャ
・光沢反射のキャプチャ（中周波と高周波）
・多重反射、相互反射のための照明の計算モデル構築

ⓒ2014 Information Processing Society of Japan

・写真からのキャプチャ
・Camera Calibration を使った照明情報の自動抽出
　例：Google Map Street View から照明環境の自動抽出
・非凸形状の空間分割、リアルタイム化
・空間分割における四面体数の最小化
が挙げられる.

参考文献

1) SIGGRAPH 2012 Course: Practical Physically Based Shading in Film and Game Production
http://blog.selfshadow.com/publications/s2012-shading-course/
2) Google Map Street View
https://www.google.com/maps/views/?hl=ja&gl=jp
3) WebGL 2.0
http://www.khronos.org/registry/webgl/specs/latest/2.0/
4) Real-Time Global Illumination on GPU
http://www.gtmacdonald.com/wp-content/uploads/2012/05/GI-Paper.pdf
5) Hao Chen. “Lighting And Material Of Halo 3”
http://developer.amd.com/wordpress/media/2012/10/S2008-Chen-Lighting_and_Material_of_Halo3.pdf
6) Spherical Harmonic Lighting: The Gritty Details
http://www.research.scea.com/gdc2003/spherical-harmonic-lighting.pdf
7) An Efficient Representation for Irradiance Enviroment Maps
http://www.cs.berkeley.edu/~ravir/papers/envmap/envmap.pdf
8) Stupid Spherical Harmonics (SH) Tricks
http://www.ppsloan.org/publications/StupidSH36.pdf
9) Efficient Spherical Harmonics Lighting with the Preetham Skylight Model
http://www.cg.tuwien.ac.at/research/publications/2008/Habel_08_SSH/
10) Emmanuel Briney, Victor Ceitelis and David Crémoux, Fast Fake Global Illumination
http://www.shaderx7.com/TOC.html
11) Light probe interpolation using tetrahedral tessellations
http://robert.cupisz.eu/stuff/Light_Probe_Interpolation-RobertCupisz-GDC2012.pdf
12) 3 次元 FEM のための自動要素分割法
http://www.morikita.co.jp/books/book/2380
13) Three.js
http://threejs.org
14) Dabrovic Sponza Model, Sibenik Model
http://graphics.cs.williams.edu/data/meshes.xml
15) Dog Trot Model
http://www.mixamo.com

ⓒ2014 Information Processing Society of Japan