GPU を用いた水中の光跡の高速レンダリング

一木 誠史† 岩崎 慶‡ 高木 佐恵子‡ 吉本 富士市‡
和歌山大学大学院システム工学研究科† 和歌山大学システム工学部‡

1 はじめに

自然物の写実的な画像の生成は，コンピュータグラフィックスにおいて重要な研究課題の一つである．水に関する自然物は，海や湖，プールなどに代表されるように，景観を表現する際の重要な要素の一つである．本稿では，水中景観のレンダリングに着目する．水中景観の表現では，水中の散乱光によってできる光跡の表現が重要であり，それは，水中での散乱光を視線に沿って積分することでレンダリングされる．しかしながら水中の輝度分布は非一様であり，計算コストが高い．

そこで本稿では GPU を用いた水中の光跡の高速レンダリングを提案する．本手法は従来法[1]の高速化を行う．従来法の課題として計算処理に時間がかかることがあげられる．この理由として，輝度計算処理の一部にしか GPU を使用していないこと，積分区間を細かくとる必要があることがあげられる．そこで，提案法では輝度計算処理をすべて GPU で行い，また分割数を削減することで高速化を行った．

2 水中の光跡

視点が水中にある場合の水の色の計算法は[2]を基にしている．水面上の点 Q から視点 P_v に入射する光の輝度は以下の式を計算することによって求められる(図 1).

$$I_v(\lambda) = I_Q(\lambda)\exp(-c(\lambda)L) + \int_0^L I_P(\lambda)\exp(-c(\lambda)l)dl \quad (1)$$

ここで，I_Q は水面の輝度，$c(\lambda)$ は水の減衰係数，λ は光の波長，L は P_vQ 間の距離，l は PP_v 間の距離を表す．I_P は点 P での散乱光で，以下の式で求められる．

$$I_P(\lambda) = \left(I_i T(\theta_i,\theta_t) F_P \beta(\lambda,\varphi)\exp(-c(\lambda)l_s) + I_a\right)\rho \quad (2)$$

Fast Rendering of Underwater Shafts of Light using GPU
† Masafumi Ichiki, Graduate School of Systems Engineering, Wakayama University
‡ Kei Iwasaki, Saeko Takagi and Fujiichi Yoshimoto, Faculty of Systems Engineering, Wakayama University

ここで I_i は水面での太陽光の輝度，$T(\theta_i,\theta_t)$ は点 P_s での透過率，F_P は集光率，$\beta(\lambda,\varphi)$ は体積散乱関数，I_a は環境光，ρ は密度を表す(図 1).

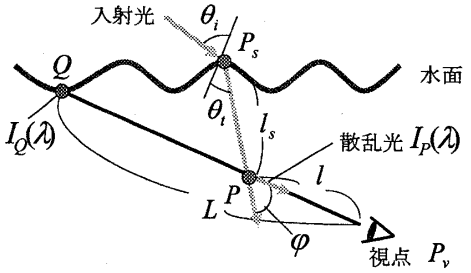

図 1: 視点に到達する光の輝度

3 水中の光跡のレンダリング

3.1 従来法

水中の光跡の輝度計算は，前節の散乱光の積分項(式(1)の第 2 項)を計算することによって求められる．水面は格子分割し，細分化した三角形メッシュと考える．各格子点における法線ベクトルと太陽光の入射ベクトルによって決まる屈折ベクトルを掃引することによってできる volume を illumination volume とする(図 2).

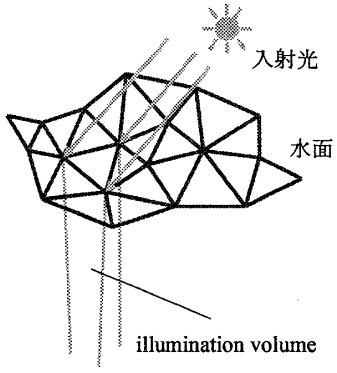

図 2: illumination volume

この illumination volume をいくつかの volume に分割し，これを sub-volume と呼ぶ．この sub-volume を通過する散乱光の輝度を計算する．sub-volume は，さらに 3 つの四面体に分割さる．次に各四面体はスクリーンへ三角形ポリゴンの集合として描画される．従来法[1]は illumination volume を sub-volume に分割しさらに四面体に分割しているので，分割数が多く計算時間がかかるという問題点がある．

3.2 提案法

提案法は従来法[1]を基にしている．提案法においても illumination volume を用いて光跡のレンダリングを行う．illumination volume を通過する散乱光の輝度を求めるとき，illumination volume と視線との交差部分の距離が最も長い所を求める必要がある．illumination volume をスクリーンに投影したとき中心に来る屈折ベクトルを通る視線が，最も交差部分が長い(図 3 参照)．交差部分の最も長いところが illumination volume を通過する散乱光の輝度が最も大きいところとなる．これを求めることによりフラグメントプログラムを用いて illumination volume の輝度を補間できる．

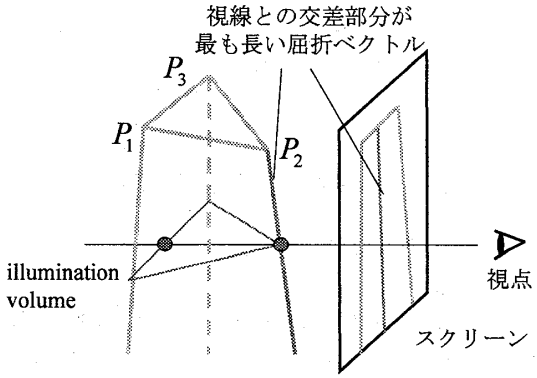

図 3: 交差部分が最も長くなる屈折ベクトル

illumination volume をスクリーン上に投影すると各屈折ベクトルがスクリーン上で交わる場合がある．このとき交差点ごとに視線との交差部分が最も長い部分が入れ替わる．そこで，各屈折ベクトルのスクリーン上での交差点ごとに illumination volume を分割し，分割された各 volume を提案法での sub-volume とする．この分割された sub-volume を投影した各三角形，四角形を描画することにより光跡を描画する．

3.2.1 sub-volume の輝度計算

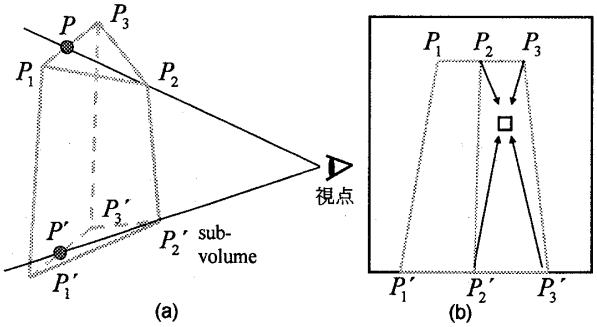

図 4: 視線と sub-volume の交点と投影図
(a)交点　(b)投影図

sub-volume の散乱光を計算するため視点と点 P_2 を通るベクトルと平面 $P_1P_1'P_3'P_3$ との交点 P を求める．同様に視点と点 P_2' を通るベクトルとの交点 P' を求める(図 4 参照)．フラグメントプログラムを用いてこの交点を補間することで各フラグメントでの視線ベクトルと sub-volume との交差部分の長さを求めることができる．同様のフラグメントプログラムを用いて sub-volume の各点での輝度と水面からの距離を補間し散乱光の輝度を計算する．全ての illumination volume の散乱光を足し合わせることにより水中の光跡をレンダリングできる．

4 結果とまとめ

図 5 に水中の光跡のレンダリング結果を示す．計算環境は CPU Pentium4 3.40GHz, GPU GeForce8800GTX である．三角形メッシュの総数は 131,072(256 × 256 × 2)とした．実行時間は 0.501 秒要した．従来法[1]では同じ環境で 4.078 秒要した．

輝度計算処理をすべて GPU で行い，また sub-volume の新しい分割法を提案し分割数を減少させた．その結果，水中の光跡を従来手法[1]より約 8 倍高速にレンダリングすることができた．

今後の課題は水中の物体によってできる光跡の影を GPU を用いてレンダリングすることである．

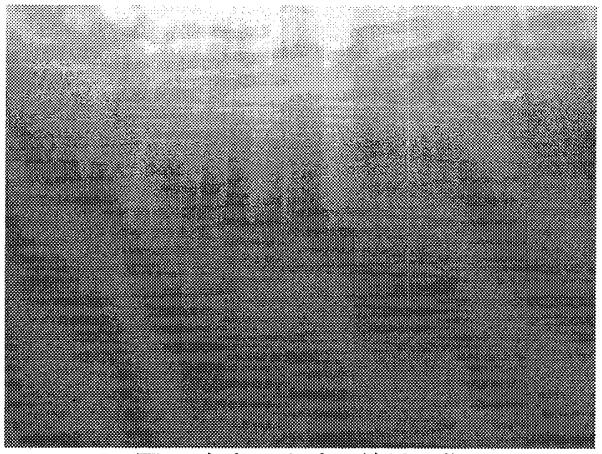

図 5: 水中の光跡の結果画像

参考文献

[1]K. Iwasaki, Y. Dobashi, T. Nisita, An Efficient Method for Rendering Underwater Optical Effects Graphics Hardware, Computer Graphics Forum, Vol.21, No.4, pp.701-712, 2002.

[2]T. Nisita, E. Nakamae, Method of Displaying Optical Effects within Water using Accumulation-Buffer, Proc. of SIGGRAPH'94, pp.373-380,1994.