スクリーン空間での 2 回屈折を考慮した液体の実時間描画

今井 拓也† 金森 由博‡ 福井 幸男‡ 三谷 純‡

†‡筑波大学 〒305-8573 茨城県つくば市天王台 1-1-1

E-mail: †imai-t@npal.cs.tsukuba.ac.jp, ‡{kanamori, fukui, mitani}@cs.tsukuba.ac.jp

あらまし ゲームなどの対話的なアプリケーションにおいて，粒子で表現された液体がシミュレーションに使われている．この液体の描画では高速化のために液体前面でのみ屈折が計算されてきたが，液体の厚みを表現できず写実性を損なう問題がある．本論文では，粒子の集合として表現された液体に対して，液体の前面および背面を抽出し，液体の前面と背面の 2 回の屈折を考慮した実時間描画手法を提案する．液体を表現した粒子の集合から液面を実時間で抽出する既存手法と，ポリゴンメッシュに対し 2 回屈折を考慮して実時間描画を行う既存手法を用いる．さらに液体背面と視線レイとの交点計算に割線法を用いることで高速化を図る．

キーワード 実時間描画, 液体, 2 回屈折

Real-Time Screen-Space Liquid Rendering with Two-Sided Refractions

Takuya IMAI† Yoshihiro KANAMORI‡ Yukio FUKUI‡ and Jun MITANI‡

†‡University of Tsukuba 1-1-1 Tenno-dai, Tsukuba, Ibaraki 305-8573, Japan

E-mail: †imai-t@npal.cs.tsukuba.ac.jp, ‡{kanamori, fukui, mitani}@cs.tsukuba.ac.jp

Abstract In interactive applications such as games, particle-based liquids are often used for simulation. Such liquids are rendered with only a single refraction at the front-facing surface, which damages photorealism because it cannot express the thickness of liquids. In this paper, we propose an approach to render particle-based liquid with twice refraction at front- and back-facing surfaces in real time. We use two existing approaches, i.e., one is to generate the liquid surface from particles in real time, and the other is to render a polygon mesh considering two-sided refraction in real time. Additionally, we accelerate the intersection calculation of the back-facing surface and viewing rays by secant method.

Keyword real-time rendering, particle-based liquid, two-sided refraction

1. 研究背景

ゲームなど対話的なアプリケーションでは，液体を有限個の粒子として離散的に表現し，各粒子の動きを計算して液体のシミュレーションを行う．しかし対話的なアプリケーションにおける液体の描画結果は十分に写実的であるとはいえない．高速化のため，液体に対する光の反射や屈折の計算を省略し，最初に光が液体に入射した際にのみ反射や屈折を計算しているからである．そのため液体の厚みを表現できず，写実性を損ねてしまう問題がある．

そこで本研究では粒子の集合として表現された液体に対し 2 回屈折，つまり前面と背面の 2 面での屈折を考慮した実時間描画を実現する手法を提案する．液体ではなくポリゴンメッシュに対し 2 回屈折を考慮した実時間描画手法は既に存在する[1, 2]が，これらの手法を液体に適用するためには視点からの液体の前面および背面のデプス値と法線ベクトルが必要になる．液体を表現した粒子の集合から液体前面のデプス値と法線ベクトルを取得する手法[3]を利用し，液体の前面および背面のデプス値と法線ベクトルを取得する．本研究ではさらに，Dual Depth Peeling[4]を用いて粒子の前面と背面のデプスを同時に取得することで描画回数を減らし，また背面と視線レイとの交点計算において，ポリゴンメッシュに対する 2 回屈折を考慮した実時間描画[2]で用いられていた二分探索法ではなく，割線法を用いて計算を高速化する．

提案手法での描画結果を液体前面でのみ屈折を考慮した場合および，液体背面と視線レイとの交点計算に既存手法で利用された二分探索法を用いた場合と比較し，写実的な画像が高速に生成できることを示す．

2. 関連研究

各粒子を球として描画した後，得られたデプスマップに平滑化を施すことでスクリーン空間において実時間で液面を抽出する手法は既に存在している[3, 6]．これらの手法は液体前面のみを対象とした手法であり，2 回屈折を考慮していない．本研究では液体の前面および背面での 2 回屈折を考慮した描画を実現すること

で写実性を高める．

Wyman[1]はポリゴンメッシュモデルに対し，前面および背面での 2 回屈折を計算すれば十分に写実的な結果が得られることを示した．しかし Wyman の手法では事前にジオメトリの各頂点での法線方向の奥行を計算する必要があり，ジオメトリが変形するような物体には適用できない．また，視線レイとジオメトリの背面との交点をスクリーン空間において二分探索法を用いて反復的に計算して求め，変形するポリゴンメッシュモデルに対して 2 回屈折を考慮した効率的な実時間描画を実現する手法も存在する[2, 7]．これらの研究は 2 回または複数回の屈折を考慮して実時間描画を実現しているが，ポリゴンメッシュモデルを対象としている．このため，粒子の集合として表現された液体に対して直接適用するためには別途液体のデプスを計算する必要がある．本研究では液面を抽出する手法と組み合わせることで，粒子の集合として表現された液体に対し 2 回屈折を考慮した実時間描画を実現した．

3. 提案手法

3.1. 概要

本研究で提案する手法の計算の流れについて説明する．提案手法は粒子の位置と半径を入力とする．Cords と Staadt[3]の手法を基に粒子の集合として表現された液体から液面を抽出し，抽出した液面に対し Oliveira と Brauwers[2]の手法を基に描画を行う．提案手法では光の反射や屈折の計算を省略し，前面での反射光と 2 回屈折した屈折光を用いて最終的な色を決定する．提案手法における計算の流れを図 1 に示す．

提案手法では以下の改善を行う．液体前面および背面のデプスマップを作成するためには粒子を球として 2 度描画しなければならず，処理時間が増えてしまう．そこで粒子の描画回数を減らすために，Dual Depth Peeling[4]を用いて粒子の前面と背面のデプス値を同時に取得し，液体前面および背面のデプスマップを一度に作成する．また，デプスマップ中のデプス値に大きな差のあるエッジ部分を保ちつつ平滑化を行うために，デプスマップの平滑化において，既存手法で利用されていたバイノミアルフィルタではなくバイラテラルフィルタを用いる．平滑化を高速に実行するためにカーネル半径の小さなフィルタを繰り返し適用する．また，計算回数を減らすため前面および背面デプスマップを同時に平滑化する．液体背面と視線レイとの交点計算において計算の反復回数を減らすために，既存手法で利用されていた二分探索法ではなく割線法を用いる．

3.2. 液面の抽出

Cords と Staadt の手法を基に液面を抽出する．最初に粒子を球として描画し，デプスマップを作成する．提案手法では最適化された球の描画手法である Kanamori ら[5]の手法を用いる．

次にデプスマップを平滑化する．Cords と Staadt はバイノミアルフィルタを用いていたが，提案手法ではバイラテラルフィルタを利用した．入力デプスマップを D，バイラテラルフィルタを適用したデプスマップを D' とすると座標 (i, j) の計算結果は次のようになる．

$$D'(i,j) = \frac{\sum_{n=-w}^{w}\sum_{m=-w}^{w} D(i+m, j+n)\, W_{m,n,i,j}}{\sum_{n=-w}^{w}\sum_{m=-w}^{w} W_{m,n,i,j}} \tag{1}$$

$$W_{m,n,i,j} = \exp\left(-\frac{m^2+n^2}{2\sigma_1^2}\right)\exp\left(-\frac{\left(D(i,j)-D(i+m,j+n)\right)^2}{2\sigma_2^2}\right) \tag{2}$$

ここで w はカーネル半径，σ_1 と σ_2 はバイラテラルフィルタのパラメータである．バイラテラルフィルタを複数回適用することでデプスマップを十分平滑化する．提案手法では r を粒子の半径とし，$\sigma_1 = w/2\sqrt{2}$, $\sigma_2 = 2\sqrt{2}r$ として計算した．平滑化は一般的に，カーネル半径が大きいほど，平滑化回数が多いほどより平滑化されるが，平滑化にかかる時間が増えるトレードオフの関係がある．実験で求めた最適値として，$w = 4$ を用い，平滑化回数 20 回で描画結果を作成した．そして平滑化されたデプスマップから法線マップを作成する[6]．

3.3. 割線法での視点レイと液体背面との交点計算

液体背面での屈折を計算するためには液体前面で屈折された視線レイと液体背面との交点を計算する必要がある．Oliveira と Brauwers の手法では液体前面で屈折された視線レイとポリゴンメッシュの背面との交点を計算するために二分探索法を用いた．二分探索法を利用するためには初期点として背面デプスマップよりも手前にある点と奥にある点がそれぞれ必要になる．そこで Oliveira と Brauwers の手法では屈折された視線レイ上にある，デプス値が背面デプスマップの最小値と一致する点および最大値と一致する点を用いた．このとき，背面デプスマップの最小値と最大値を計算するための処理を別途行う必要がある．そこで提案手法では背面デプスマップの最小値と最大値を用いずに液体背面と視線レイとの交点を計算するために割線法を用いる．

液体前面と視線レイとの交点を $\mathbf{P}_f$，液体前面で屈折された視線レイの単位ベクトルを $\mathbf{T}_f$ とすると，パラメータ $t\ (\geq 0)$ を用いて液体前面で屈折された視線レイ上の点 $\mathbf{P}(t)$ は次式で表現できる．

$$\mathbf{P}(t) = \mathbf{P}_f + t\,\mathbf{T}_f \tag{3}$$

点 $\mathbf{P}(t) = (x(t), y(t), z(t))$ をスクリーンに投影したときの点を $\mathbf{P}'(t)$ とする．視点座標系での背面のデプス値を記録したデプスマップを D_{back}^{view} とし，点 $\mathbf{P}'(t) = (x'(t), y'(t), z'(t))$ に対し点 $(x'(t), y'(t))$ におけるデプス値を $D_{back}^{view}(x'(t), y'(t))$ と

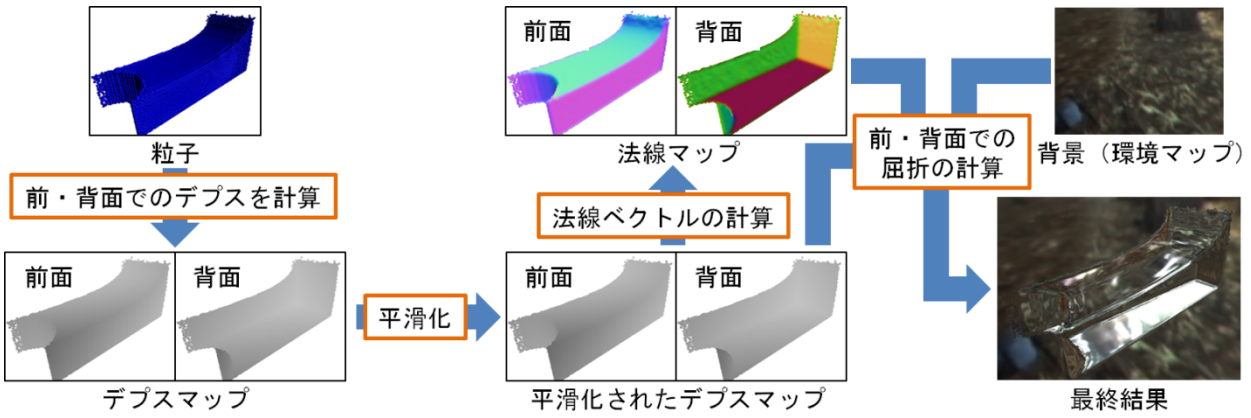

図 1: 提案手法における計算の流れ．まず粒子を球として描画し，前面および背面マップを作成する．次に作成したデプスマップを平滑化する．その後，平滑化されたデプスマップから法線マップを作成する．そして，平滑化されたデプスマップ，法線マップ，背景を用いて前背面での屈折の計算を行い，最終的な色を決定する．

記す．点$\mathbf{P}(t)$のz成分$z(t)$と$D_{back}^{view}(x'(t), y'(t))$との差（以下，$\mathbf{P}(t)$と背面デプスマップとの差と表現する）$\Delta z(t)$を次のように定義する（図 2）．

$$\Delta z(t) = z(t) - D_{back}^{view}(x'(t), y'(t)) \tag{4}$$

屈折された視線レイ上の 2 点$\mathbf{P}(t_s), \mathbf{P}(t_e)$ $(t_s < t_e)$と背面デプスマップとの差$\Delta z(t_s), \Delta z(t_e)$に対し，2 点$(t_s, \Delta z(t_s))$，$(t_e, \Delta z(t_e))$を通るような変数$t$の一次関数$f(t)$を考える．

$$f(t) = \frac{\Delta z(t_e) - \Delta z(t_s)}{t_e - t_s}(t - t_s) + \Delta z(t_s) \tag{5}$$

$f(t) = 0$となるようなtの値t^*を求めると，

$$t^* = t_s - \frac{t_e - t_s}{\Delta z(t_e) - \Delta z(t_s)}\Delta z(t_s) \tag{6}$$

となる．求めたt^*における液体の前面で屈折された視線レイ上の点$\mathbf{P}(t^*)$と背面デプスマップとの差$\Delta z(t^*)$を求め，$\Delta z(t^*)$が一定の範囲に収まっていれば点$\mathbf{P}(t^*)$を液体前面で屈折された視線レイと液体背面での交点とする．それ以外の場合，$\mathbf{P}(t_s)$か$\mathbf{P}(t_e)$のどちらかを$\mathbf{P}(t^*)$で置き換え，さらに$(t_s < t_e)$となるように$\mathbf{P}(t_s)$と$\mathbf{P}(t_e)$を入れ替えて次のt^*を計算していく．

スクリーン座標系の点$\mathbf{P}'(t)$に粒子の領域が存在しない場合，背面デプスマップに粒子のデプス値が書き込まれず，設定していた初期値が書き込まれ無効な値になる．無効な値が格納された点$\mathbf{P}'(t)$を参照した場合，デプス値が有効な値になるまでtの値を半分にする操作を繰り返し行った．

3.4. 出力する色の計算

出力する色$\mathbf{C}$は次式を用いて計算される．

$$\mathbf{C} = F(\mathbf{v}, \mathbf{n}_f)\,\mathbf{C}_f + (1 - F(\mathbf{v}, \mathbf{n}_f))\,\mathbf{C}_b \tag{7}$$

ここで$\mathbf{v}$は視線ベクトル，$\mathbf{n}_f$は液体前面での法線ベクトル，$\mathbf{C}_f$は液体前面での反射ベクトルを基に環境マップから取得した色，$\mathbf{C}_b$は液体背面での屈折ベクトルを基に環境マップから取得した色である．またFはフレネル係数を表している．ここではフレネル係数としてSchlick[8]の近似式を用いた．

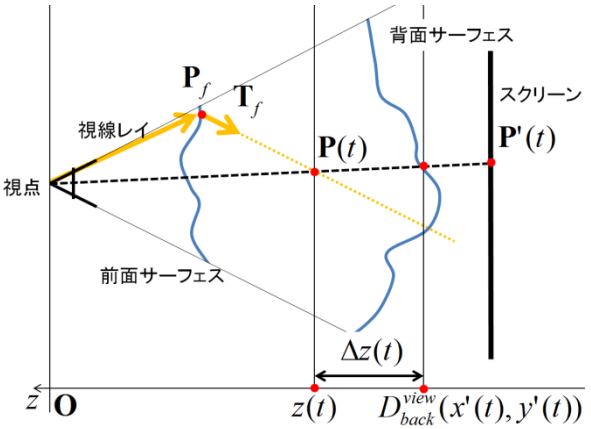

図 2: 割線法の模式図．

4. 結果

提案手法は C++を用いて実装した．ライブラリは OpenGL, GLUI, GLUT, GLEW, NVIDIA SDK を使用した．シェーダには GLSL を用いた．実行環境として Intel Core i7-4770 3.40GHz の CPU, 8GB のメモリ，NVIDIA GeForce GTX TITAN の GPU を搭載した PC を使用した．約 10 万個の粒子を事前にシミュレートしたデータを用いて実行結果を計算し，640 × 480と1024 × 768の解像度で出力した．結果にはシミュレーションの計算時間は含まれていない．提案手法を用いて描画した場合，視線レイと背面交点の計算に既存手法[2]で利用された二分探索法を用いた場合，および液体前面においてのみ屈折を計算した場合の描画結果を図 3 に示す．また，提案手法を用いて描画した際の描画にかかる時間を表 1 に，背面交点の計算に二分探索法を用いて描画した際の描画にかかる時間を表 2 に，液体前面のみ屈折を考慮して描画した際に描画にかかる時間を表 3 に示す．表 1 と表 2 を比較すると，二分探索法を用いた場合の背面デプスマップにおける最大最小の計算と描

表 1: 提案手法を用いて描画する際にかかるフレームあたりの計算時間．単位はミリ秒である．

	640 × 480	1024 × 768
総描画時間	12.64	30.78
デプスマップ作成	2.13	5.14
平滑化	10.24	25.02
法線マップ作成	0.10	0.25
屈折の計算	0.17	0.37

表 2: 二分探索法を用いて描画する際にかかるフレームあたりの計算時間．単位はミリ秒である．

	640 × 480	1024 × 768
総描画時間	12.72	30.88
デプスマップ作成	2.13	5.14
平滑化	10.24	25.02
法線マップ作成	0.10	0.25
最大最小計算	0.12	0.20
屈折の計算	0.13	0.27

表 3: 前面屈折のみを考慮して描画する際にかかるフレームあたりの計算時間．単位はミリ秒である．

	640 × 480	1024 × 768
総描画時間	8.78	19.83
デプスマップ作成	0.71	1.91
平滑化	8.00	17.77
法線マップ作成	0.03	0.06
屈折の計算	0.04	0.09

画にかかる合計時間よりも，提案手法で利用した割線法を用いた場合の描画時間の方が短く，提案手法の方が速く計算できる．表 1 と表 3 を比較すると，提案手法描画した場合よりも液体前面のみ屈折を考慮した場合の方が 1.5 倍速く計算できるが，図 3(a)と図 3(c)を比較すると 2 回屈折を考慮した結果の方が液体の厚みを感じることができる．

5. 結論と今後の課題

本論文では粒子の集合として表現された液体に対し 2 回屈折を考慮した実時間描画手法を提案した．液面を抽出する既存手法とポリゴンメッシュに対し 2 回屈折を考慮して実時間描画を行う既存手法を組み合わせる．さらに Dual Depth Peeling[4]を用いて描画回数を減らし，液体前面で屈折された視線レイと液体背面との交点の計算に割線法を用いることでより高速な描画を可能にした.一方で飛沫が飛散する場合，不自然な結果が生成されてしまう．提案手法では液体前面と背面との間は液体（粒子）で満たされていると想定しているため，液体前面と背面との間に粒子で満たされていない空間があると不自然な結果が生成されてしまう．この問題を解決するためには，例えば 3 枚以上のデプスマップを用いた複数回の屈折を考慮した液体の

(a) 提案手法を用いた場合の描画結果

(b)二分探索法を用いた場合の描画結果

(c) 液体前面のみ屈折を考慮した場合の描画結果

図3: 各手法によってシーンを描画した結果．

実時間描画の実現が挙げられる．Dual Depth Peeling を繰り返し適用すればデプスマップを複数枚取得し，複数レイヤの液面を作成することができる．しかし計算コストが高いため，より効果的な液体の描画手法を考えていく必要がある．

文　献

[1] C. Wyman: “An Approximate Image-Space Approach for Interactive Refraction”, ACM Trans. Graph., 24, 3, pp.1050-1053 (2005)

[2] M. M. Oliveira and M. Brauwers: “Realtime Refraction through Deformable Objects”, In Proc. of the 2007 Symposium on Interactive 3D Graphics and Games, pp.89-96 (2007)

[3] H. Cords and O. Staadt: “Instant liquids”, In Poster proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation (2008)

[4] L. Bavoil and K. Myers: “Order Independent Transparency with Dual Depth Peeling”, Technical report, NVIDIA (2008)

[5] Y. Kanamori, Z. Szego, and T. Nishita: “GPU-based Fast Ray Casting for a Large Number of Metaballs”, Computer Graphics Forum, 27, 2, pp.351-360 (2008)

[6] W. J. van der Laan, S. Green, and M. Sainz: “Screen Space Fluid Rendering with Curvature Flow”, In Proc. of the 2009 Symposium on Interactive 3D Graphics and Games, pp.91–98 (2009)

[7] S. T. Davis and C. Wyman: “Interactive Refractions with Total Internal Reflection”, In Proceedings of Graphics Interface 2007, pp.185–190 (2007)

[8] C. Schlick: “An inexpensive brdf model for physically-based rendering”, Computer Graphics Forum, 13, 3, pp.233–246 (1994)