2009－CG－134　（5）
2009／2／16

フロントガラスを流れる雨滴の CG 表現

後藤 優*　田中 敏光　佐川 雄二（名城大学）

抄録

本研究の目的は，フロントガラスを流れる雨滴を実時間でリアルに表示することである．水滴の不規則な運動や尾を引いて流れ落ちる様子は，ガラス面をメッシュに分けて親和係数を使って運動を制御することで表現する．ワイパーで雨滴がかき集められる様子は，水を粒子で定義し，その反発を計算することで表現する．この 2 つの手法を状況により切り替えることで，ガラス面上の水滴運動をシミュレーションする．表示には 2 次元メタボールを用いるが，高さを非線形圧縮することで，滑らかな水滴表面と，自然な融合・分離を表示する．また，これらの処理を GPU に実装することで，実時間表示を実現する．

CG simulation of flow of raindrops on a windshield

Gotou Yuu*，Toshimitsu Tanaka，Yuji Sagawa，(Meijo University)

Extract

This research presents a method to display realistic flow of raindrops on windshield in real time. The grid system well simulates irregular movements of the water drops. The particle system is suitable to handle water pushed by windscreen wipers. Thus, by the distance from the wipers, our method selects one from those two methods properly. Shape of the water drops are defined with 2D metaballs. By compressing their height non-linearly, their shape becomes natural and smooth in comparable to 3D metaballs. The 2D metaballs can be displayed in realtime by using GPU.

１．はじめに

CG（Computer Graphics）アニメーションで物体をリアルに表現するには，物体の運動を忠実に再現する必要がある．運動中に形が変わる物体では，形の変化の再現も必要となる．そこで，物理法則に従って，物体の運動や形状変化を導き出す手法が研究されている．ここで言う物体には，キャラクタや乗り物だけでなく，風や雨，炎，煙，水など，映像を作る上で必要なもの全てが含まれる．

窓ガラスの上を水が流れていく様子も，リアルに表現すべき自然現象の一つである．窓ガラスは，我々が利用する建物や乗り物のほとんどに使われているので，それを表示する必然性は高い．特に，ドライブシミュレータやトレインシミュレータ，カーレーシングゲーム，CG アニメーションなどで雨のシーンを表示するときには，窓ガラスを流れる水滴のリアルな表現が不可欠となる．

ガラス面上の水は，その状態により様々な姿を見せる．少量の水が付着している場合には，半球状水滴としてガラスに留まる．しかし，やがて，他の水滴と融合したり，重力や風力によって尾を引いて流れ出したりする．窓ガラスならこれまでだが，車のフロントガラスでは，水滴はワイパーで集められ，ガラス面から除去される．その際，水滴が押し合って形を変えたり，ワイパーの端からこぼれたりする．

したがって，車や電車のフロントガラスをリアルに表現しようとすると，このような水の状態の変化を表示する必要がある．さらに，インタラクティブなシミュレータ等に利用するには，実時間でアニメーションを生成する必要がある．

２．従来の研究

ガラス面上を流れる水滴のシミュレーションでは，水滴に働く重力，風力，摩擦力を考慮した運動方程式を解くことで，水滴の動きを決めている．さらに金田らの手法[1]では，ガラス面をメッシュ状に分割し，傷や汚れなどのガラス面の不均一性を示す情報を親和係数として各区画に設定している．水滴の移動も区画単位で行うが，親和係数を使って移動先を確率的に求めることで，水滴が蛇行して流れる様子を再現している．これに続く研究[2][3][4]では，対象を曲面に広げ，風の影響を考慮するようにモデルが拡張されている．さらに海野ら[5]は，親和係数と摩擦係数を別に設定したり，親和係数を動的に変えることで，水滴が一度通った場所は他の水滴が通りやすくなる様子を表示できるように，親和係数モデルを発展させている．ただし，これらの手法では，水滴を 1 つの粒子として扱っているため，水滴の融合や分離，衝突時の形状変形などは表現できていない．

益村の研究[6]では，それまで半球形状で表示していた水滴を 2 次元メタボールで表現することで，水滴が尾を引いて流れる様子を表現している．さらに静止している水滴同士の融合条件に制限を加え，水滴同士が触れていても一つに融合しない場合も想定したことで，ある程度複雑な形状の水滴を残すことにも成功している．しかし，静止状態の水滴の複雑な形状を表示することはできない．

五十住らの研究[7]では，1 個の水滴を複数のメタボールで表現することで，ガラスに付着した水滴の複雑な形状や，融合時の形状の変化，水滴の尾の生成と消滅などを表現している．さらに，水滴形状を 3 次元メタボールを使って定義し，リフレクション／リフラクションマッピングで周辺景色の映りこみを表現することで，リアルな表示が可能になった[8]．この手法は，従来手法に比べるとはるかに自然な水滴を表示できるのだが，3 次元メタボールの処理に時間がかかるため，リアルタイム表示はできない．

また，従来の手法では水滴間の相互作用を考慮していないため，水滴が密集した状態では表示が不自然になる．融合処理を行う手法では，メッシュの 1 区画に入った水滴は全て融合するため，水滴の密集した場所に巨大な水滴ができてしまう．

３．提案手法の概要

本研究では，五十住らの手法[7]を拡張し，ワイパーの作用を含めて，フロントガラスを流れる水滴を表現する．雨滴がガラス面に接触した瞬間に，その場所にサブメタボールを持つ水滴を発生する．この水滴はメッシュモデルに従ってガラス面を運動するが，ワイパーの周辺ではワイパーで集められた水滴が密集するため，それらが融合して巨大な水滴ができてしまう．このため，この方法では，水滴が集まって水が溜まるような状況を表現することはできない．そこで，ワイパーの近傍に入った水滴を，一時的に均一の大きさの粒子に変換する．そしてワイパーや近隣粒子との相互作用を計算することで，水滴が他の水滴に押される様子や，水滴が集まって塊になる様子を表現する．

水滴の表示には 2 次元メタボールを利用するが，メタボールを高さ方向に圧縮することで，3 次元メタボールを使った手法[8]とほぼ同等の品質で，複雑な水滴形状を表示する．さらに，グラフィックハードウェアに適したアルゴリズムを実装することで，水滴をリアルタイム表示する．

４．水滴の運動処理

この章では，従来研究で使われていたメッシュモデルと本研究で導入する粒子モデルを説明し，その切り替え方法を示す．

4.1 メッシュモデル

ガラス面を等間隔のメッシュに区切り，メッシュのセルのそれぞれに親和係数と呼ばれるガラス面の揺らぎを表す値を記録する．水滴はメッシュを単位として移動する．水滴に働く重力，風力，摩擦力などの力を考慮した運動方程式を解いて，水滴の移動ベクトルを求める．この方向にあるセルとその両脇のセルの３つから，親和係数を考慮して水滴の移動先を確率的に選ぶことで，水滴の蛇行を表現する．

水滴が移動して同じセルに 2 つの水滴が入ったら，それらを 1 つに融合する．これにより，水滴が細かい水滴を集めながら流れる様子が表現できる．また，水滴が移動した跡に小さな水滴を残しておくことで，尾を引いて流れる様子を表現できる．尾の水滴は一定時間後に縮小し，消滅する．

4.2 粒子モデル

水を均一の大きさの粒子の集合として表現する手法で，滝や川など動きのある水の表示に使われている．本研究ではガラス面上の水滴を対象とするので，2 次元のモデルを使う．この場合，水滴は質量と半径が一定の円として表現される．

粒子と粒子の衝突では，粒子の中心を結ぶ距離が半径の２倍以下であれば，衝突したと判定する．衝突した場合には，それぞれの粒子の中心を，中心を結ぶ線上を互いに遠ざかる方向に，重なった長さの半分だけ移動する．これにより，半径の重なりを解消する．加えて，水滴の反発を表現するため，この移動距離に一定の係数をかけた距離だけ，さらに水滴を遠ざける．

粒子と物体との衝突処理も必要になる．本研究で対象となる移動物体はワイパーだけである．ワイパーはブレードでガラス面と接しているが，その形状は細長い長方形で近似できる．そこで，長方形の移動方向を向いた長辺と粒子の中心との距離が，半径より小さくなったとき，衝突したと判定する．衝突した場合には，球の中心を，辺の鉛直方向に距離が半径となる位置，つまり，水滴とワイパーが接する位置まで移動する．

4.3 モデルの適用領域

雨の日のフロントガラスを見ると，ワイパーから離れた場所では，水滴が離散的にガラス面に付着しているが，ワイパー周辺では，ワイパーに集められた雨水が塊となっている．先に述べたモデルの特徴を考慮すると，ワイパー付近では粒子モデルが，その他の部分ではメッシュモデルがふさわしい．そこで，図 1 に示すように，それぞれのモデルを適用する領域を決める．図中の白い領域がワイパーの領域，その周辺の黒い領域が粒子モデルの適応領域，その他の灰色の領域がメッシュモデルの適用領域である．

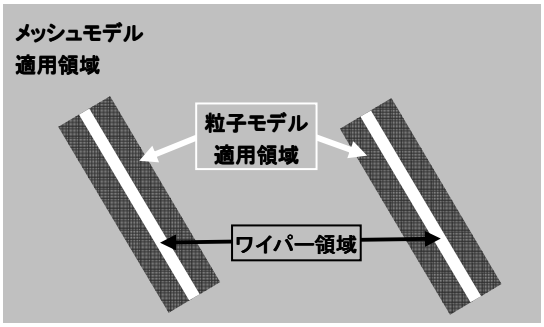


図 1　モデルの適用範囲

4.4 粒子モデルへの切り替え

メッシュモデルの水滴がワイパーと衝突したときに，その水滴を粒子モデルに変換する．図２では，白い矩形領域がワイパーで，グレーの円がメッシュモデルの水滴，黒の円が粒子モデルの水滴を示している．

図２の左側はワイパーが水滴に接触する直前の様子を，右側は接触直後の様子を表している．水滴がワイパーに触れたとき，メッシュモデルの水滴が粒子モデルの水滴に変換される．ただし，メッシュモデルの水滴は質量が可変なので，その質量を粒子モデルの水滴の質量で割った数だけ粒子を生成する．この図では，3 個生成されている．生成される粒子が 1 個の場合には，メッシュモデルの水滴と同じ位置に置くが，複数生成される場合には，それぞれの粒子を，基準の位置から，ランダムに少しずらした位置に配置する．

4.5 メッシュモデルへの切り替え

粒子がワイパー周辺の矩形領域を越えたとき，粒子モデルの水滴をメッシュモデルに変換する．図３はワイパーが折り返した後の様子を示している．この図の白い矩形領域はワイパーで，その周辺のグレーの斜線の範囲が粒子モデル領域，黒い円が粒子モデルの水滴，グレーの円がメッシュモデルの水滴である．

図３の左側が変換前の状態で，右側が変換後の状態である．範囲から外れた粒子モデルの水滴が，同じ大きさのメッシュモデルの水滴に変換されている．ワイパーが水滴を集めると，ワイパーの進行方向に水滴の層ができる．このため，ワイパーからある程度の距離が離れた場合に変換を行う．

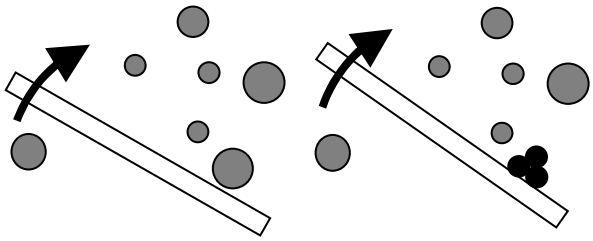


図２　メッシュモデルから粒子モデルへの変換

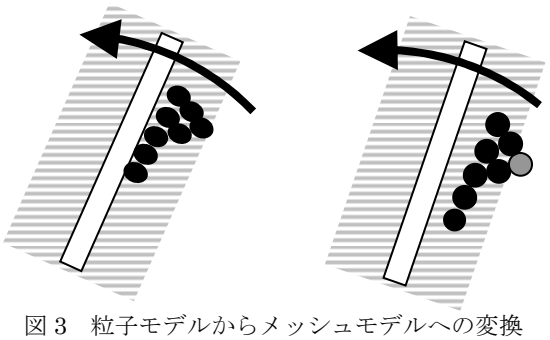


図３　粒子モデルからメッシュモデルへの変換

4.6 変換の保留

粒子モデルからメッシュモデルに変換される際，ワイパーの折り返し地点では，一度に多くの水滴が粒子モデルの領域から外れる．このとき，これらの水滴を全てメッシュモデルに変換すると，同一のセルに多数の水滴が入るため，融合して巨大な水滴ができてしまう．これを防ぐために，条件を設定して変換を保留する．

まず変換先のセルを中心とした正方領域内のセルを探索し，閾値以上の大きさの水滴が存在するなら変換を保留する．閾値を水滴が流れ始める大きさより大きく選ぶことで，変換が保留されても，その原因がすぐに解消されるようにしている．また，探索範囲を融合が起きる限界より広くすることで，変換直後に大きな水滴ができてしまうことを防いでいる．この処理により，不自然に大きな水滴が発生することはない．

5. 水滴のレンダリング

この章では，前章で求めた水滴の位置と大きさから，リアルな水滴の画像を実時間で表示する手法を述べる．

5.1 2 次元メタボール

本研究では，リアルタイム表示を実現するため，水滴の形状定義に処理コストの低い 2 次元メタボールを使っている．水滴の輪郭は 2 次元メタボールでそれらしく求まるが，凹凸はポテンシャル関数で決まってしまうので，できる限り半球形状に近い形状の関数を使う必要がある．そこで，本研究では式(1)で１つの水滴を定義する．

$$N = \frac{1}{1+\left(\frac{h}{R}\right)^4} \qquad \text{..... 式(1)}$$

この式の R は水滴の半径（定数），h は水滴の中心位置からの距離，N が求める濃度となる．図 5 左は，式(1)の半径を 20，中心座標を(64,64)として求めた濃度を 128*128pixel のハイトマップテクスチャとして出力したもので，図 5 右はそれを 3 次元表示したものである．

このハイトマップから，値が閾値を越える部分だけを取り出すと，水滴 1 つの形状となる．図 6 は閾値を 0.5 以上の部分を取り出して，0.5 を引いて表示した形状である．

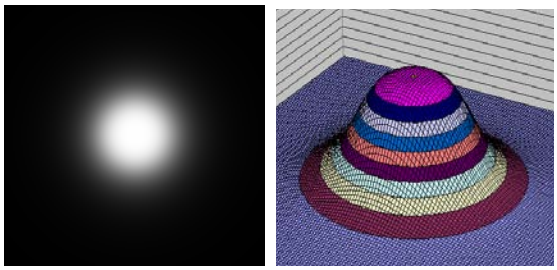


図 5　濃度関数より作成したハイトマップ

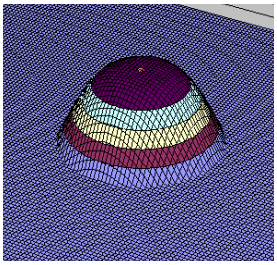


図 6　単一の水滴の半球形状

5.2 ガラス面の上に分布する水滴の表現

ガラス面の大きさのハイトマップを用意し，前述した運動モデルで定まった水滴の位置に，単一水滴のハイトマップを加算合成する(図 7). その後，ハイトマップより値が閾値以上の部分を取り出して，ガラス面上に分布する水滴の形状とする．

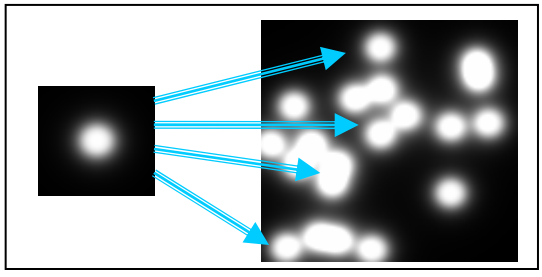


図 7　ハイトマップの加算合成

5.3 ハイトマップの補正

前節で作成したハイトマップを立体表示すると，図 8 となる．一部で水滴が不自然に高くなっているが，これは，水滴が集まった場所に水滴のハイトマップが何回も加算されたためである．しかし，これでは水滴が尖って表示されてしまう．また，法線の傾きも大きすぎるため，実際には起こりえない反射や屈折が表示される．

そこで本研究では，ハイトマップの値が大きい部位分を圧縮することで，高さの変化を滑らかに補正する．具体的には，基準の高さ V_o を決め，1 画素ごとにハイトマップの値を調べる．そして，高さ V を次の式で高さ V_mに置き換える．

$$V_m = \begin{cases} k(V - V_0) + V_0 & V > V_0 \\ V & V \le V_0 \end{cases} \quad \text{… 式(2)}$$

補正係数 k は 0～1 の間で決める．この処理 1 回だけでは，高さを十分に圧縮できないので，同じ処理を高さの基準を変えて何回か行う．図 9 は，図 8 のハイトマップを V_o=0.5，k=0.3 から V_o を 0.1 ずつ増加しながら 5 回補正した結果である．図 8 と 9 を比較すると，尖ったところがなくなり，全体に滑らかな形状になっている．このような繰り返し法を用いるのは，後述するように GPU を活用するためである．

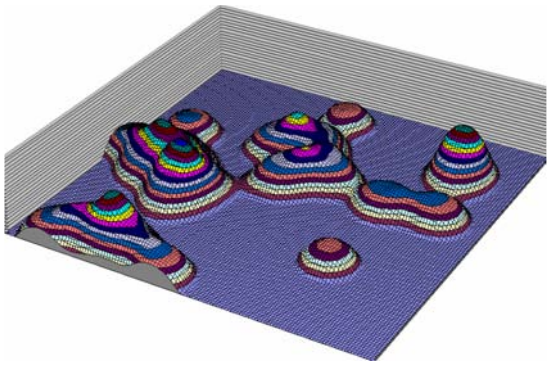


図 8　補正前のハイトマップ

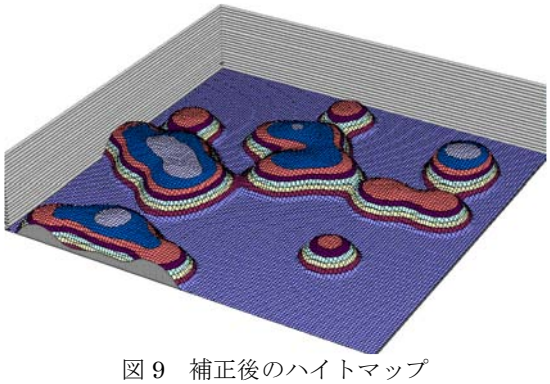


図 9　補正後のハイトマップ

5.4 環境マッピングによる反射と屈折の表現

ガラス面上に付着した水滴には，周囲の風景が歪んで映り込む．これはガラス面に入る光線が水滴によって反射・屈折して起こる現象である．そこで，反射と屈折をリフレクションマッピングとリフラクションマッピングで表示する．このマッピングには水滴表面の各点の法線が必要になるが，

ハイトマップ上の任意の位置の法線は，その位置における高さと，上と右に隣接する高さの値の差分より，上下／左右方向の接ベクトルを求め，その外積をとることで計算できる．

フロントガラスの水滴に映りこむのは，ほぼ正面の景色に限られるので，背景画像として正面画像 1 枚だけを用いる．もしはみ出した場合には，折り返して参照する．屈折は空気とガラスの境界でも生じるが，影響が小さいので無視する．

5.5 GPU を用いた提案手法の実装

これまで述べてきた手法を，リアルタイムで処理する為に，GPU を利用する．GPU では複雑な処理は行えないが，並列演算により CPU の数十倍のパフォーマンスを出すことが出来る．最近では Microsoft が提供する DirectX に含まれる HLSL を使うことで，GPU でも，ループ処理やフロー制御など，C 言語ライクなプログラミングを行うことが出来るようになった．本研究ではこの HLSL と DirectX の基本的な機能を使い，提案手法をリアルタイム処理する．以下に具体的な手法を述べる．各処理の結果は，処理ごとにバッファに保存し，マルチパスレンダリングを行う．

(1) ハイトマップの加算合成

ハイトマップは予め図 5 左のような一枚のテクスチャとして作成した物を読み込んで使うことで，濃度マップの計算時間を省略する．ガラス面への加算合成にはポイントスプライトを用いる．メッシュモデルの水滴の大きさは固定されていないので，基本の大きさの水滴のテクスチャを，水滴の大きさに合わせてバイリニアフィルタで拡大／縮小してから，加算合成する．

(2) ハイトマップの補正

HLSL によりフロー制御を行い，基準の高さ以上のピクセルに対して 5.3 節に述べた式(2)を適用する．この処理を，基準の高さを上げならが，指定回数だけ繰り返す．

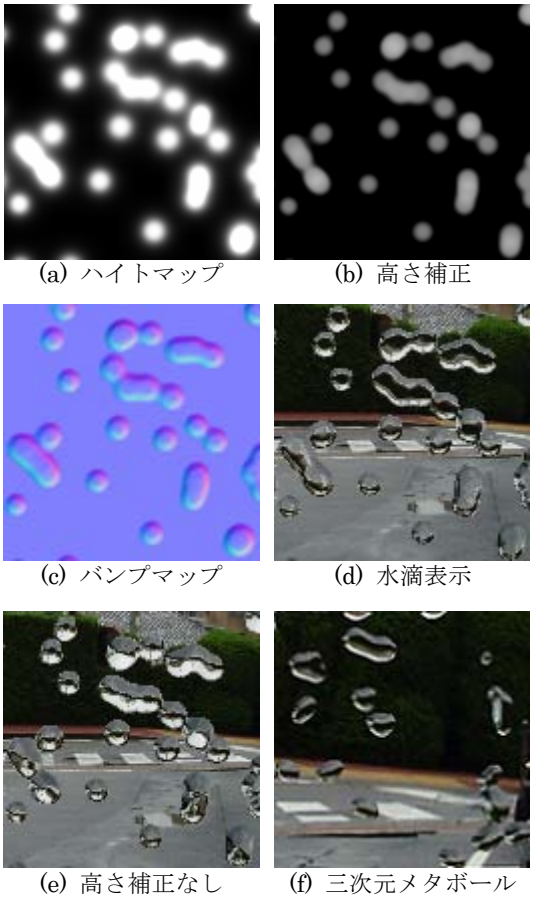
(3) 法線の計算

法線の計算ではベクトルの外積を計算する必要があるが，HLSL には外積関数が用意されているので，そのまま利用する．計算された法線は法線マップとしてバッファに保存する．

(4) 環境マッピング

背景の画像をテクスチャバッファに保存する．前項の処理で求めた法線マップをハードウェアでリフレクション／リフラクションマッピングして，画素毎に背景画像の参照位置を求める．このとき，視線はほぼガラス面に垂直に入射するので，法線ベクトルのガラス面と平行な成分を求め，その分だけ背景画像の参照位置をずらすことで，反射屈折を簡易表現する．

一連の処理を GPU で実行した結果を図 10 に示す．図 10(a)は合成したハイトマップで，そこから閾値以上の部分を取り出して高さを補正し(b)，バンプマップを作成した(c)．そこから作成した画像が(d)である．比較のため，5.3 節に示した補正処理を行わない場合の出力結果を図 10(e)に示す．図 10(d)のほうが自然な写りこみになっているのが分かる．また，図 10(f)は五十住の手法[8]で作成した，3 次元メタボールで表現された水滴である．これと比べても，図 10(d)は遜色のない品質といえる．



(a) ハイトマップ　(b) 高さ補正
(c) バンプマップ　(d) 水滴表示
(e) 高さ補正なし　(f) 三次元メタボール

図 10　GPU によるレンダリング結果

6. パフォーマンスの評価

これまでに提案した運動モデルと表示手法を統合し，パフォーマンスを評価した．アニメーションの数コマを図 11 に示す．わかりやすいように，ワイパーは矩形ポリゴンで半透明表示している．

図 11(a)はワイパーが折り返した直後の画像である．ワイパーに集められた水滴がその右側に集まっている．(b)はそれから少したった後で，集められた水滴の一部が流れている．ワイパーから離れるとメッシュモデルに代わるため，水滴が尾を引いて流れるようになる．しかし，一部は変換を抑制され，粒子モデルのままとどまっている．さらに時間がたつと，多くの水滴は流れ落ちるが，尾の一部は再び水滴となってガラス面上に残る．

この画像には，ガラス面に付着する水滴，尾を引いて流れる様子，水滴同士の滑らかな融合，背

景の写り込み，ワイパー周辺の水の集まり，などがリアルに再現されている．

Windows XP，Pentium4 3.0GHz，Geforce7600GS の環境で映像を表示したが，フレームレートは 30~40fps の間を推移しており，旧世代の実行環境でも良好な結果が残せた．ためしに運動モデルを統合せず表示したところ，60fps 以上出た．運動モデルはすべて CPU で処理され，サブメタボールの制御などの処理が複雑であるため，時間がかかるものと推察される．

7. まとめ

本研究では，車のフロントガラスを流れる雨滴をリアルに表現するための手法を提案した．この手法は，メッシュモデルと粒子モデルをワイパーからの距離で切り替えることで，蛇行しながら尾を引いて流れ落ちる水滴と，ワイパーで集められて塊になった水滴の両方をリアルに表示することができる．

水滴の表示では，2 次元メタボールを高さ方向に圧縮補正する手法で，3 次元メタボール並みの品質で表示できるようにした．メタボールの補正や，環境マッピングによる表示には GPU を用いるため，実時間表示が可能になった．

今後は，ガラス面を曲面に拡張する．また，運動モデルに車の加減速や走行に伴う風力を加えて，より現実に忠実なモデルにする．さらに，ドライブシュミュレータ等に組み込み，映像を評価することも，今後の課題である．

参考文献

[1] K.Kaneda, T.Kagawa, H.Yamashita: Animation of Water Droplets on a Glass Plate, Proc. Computer Animation'93 Models and Techniques in Computer Animation, Springer-Verlag, Tokyo, pp.77-189(1993)

[2] 寿山康彦, 金田和文, 山下英生: 風の影響を考慮したガラス表面を流れる水滴のアニメーション, グラフィクスとCADシンポジウム(1995)

[3] Kazufumi Kaneda, Yasuhiko Zuyama, Hideo Yamashita, Tomoyuki Nishita: Animation of Water Droplet Flow on Curved Surfaces, Proceedings of the Fourth Pacific Conference on Graphics and Applications, pp.50-65(1996)

[4] Kazufumi Kaneda, Shinya Ikeda, ideo Yamashita: Animation of Water Droplets Moving Down a Surface, J.Visual.Comput.Animat. 10, 15-26(1999)

[5] 海津祐也: ガラス面を流れる水滴のシミュレーション ～可変粘性係数を用いた動きの制御～, 電気学会東海支部若手セミナー (002 年度第二回) 予稿集(2003)

[6] 益井明人: 透明物体の上を流れる雨粒のシミュレーション, 名古屋大学工学部情報工学科卒業論文 (2000)

[7] 五十住拓哉, 田中敏光, 佐川雄二, 杉江昇: ガラス面上を流れる水滴の形状変化の表現, 像情報メディア学会論文誌, Vol.160, No.4, pp.538-544(2006)

[8] 五十住拓哉・田中敏光・佐川雄二：ガラス面上を流れる水滴のCG表現 ―形状表現の改良―, 信学全大D-12-47 (2007-3)

(a)ワイパーが右端で折り返した直後

(b)折り返して少し経過

(c)さらに時間が経過

(d)ワイパーが左端に到達したとき

図 11　フロントガラスの表示