Vol.2013-CG-150 No.20
2013/2/19

動きを考慮した水彩画風リアルタイムレンダリングにおけるGPUの応用

向 友香[1]　床井 浩平[2,a)]

概要：非写実的レンダリング手法の一つである水彩画風レンダリングにおいて，オブジェクトの動きに伴う絵の具の引き延ばしやかすれにもとづく動的なボケを生成する手法を提案する．絵の具が紙の上で動くということは通常考えられないが，本研究ではカトゥーンブラーに着想を得て，もし水彩画として描いたものが紙の上を動いた場合に，それに対して水彩画風のボケを付与することを試みた．提案手法はGPUを用いたスクリーン空間法にもとづくモーションブラー手法を拡張して実装し，オブジェクトの軌跡に沿ってリアルタイムにボケの発生と消滅を再現する．

Mukai Yuka[1]　Tokoi Kohe[2,a)]

1. はじめに

1.1 Real-Time Non-Photorealistic Rendering

三次元コンピュータグラフィックス (3D CG) による映像制作では，シーンを写実的に再現するだけでなく，セルアニメーション [1] や鉛筆画 [2,3]，水彩画 [4–9]，水墨画 [10–12] などの非写実的な表現も用いられている．このような映像生成手法を Non-Photorealistic Rendering (NPR) と呼ぶ．

三次元の NPR は，写実的なレンダリングと同様に，形状モデルや光源，視点などのシーン情報に基づき，投影像や陰影などを計算により求めて画像を生成する．その際，鉛筆や筆，絵の具，および紙などの画材特有の質感の再現も行うことによって，手描きより少ない手間で，手描きでは困難な複雑な動きの再現などを行うことができる．

このため，NPR はアニメーション映画の制作やビデオゲームなどに用いられている．特にビデオゲームでは，以前はフォトリアリズムの向上を目指してハードウェアの性能向上やソフトウェアの技術開発が行われてきたが，最近ではコンテントの作風に合わせた映像表現や，新しい視覚表現を導入するために，様々な NPR 手法が用いられるようになってきている．しかし，毎秒数十回の描画処理が求められるリアルタイム 3D CG では計算量などの点で実現が困難な表現も存在する．

中でも水墨画や水彩画風の表現では，絵の具と紙との相互作用を考慮する必要があり [4,11,13]，この精密な再現をリアルタイムに行うことは難しい．水彩画に使われる水彩絵の具は，主に顔料と展色材であるアカシア樹脂 (アラビアガム) から成り，その比率によって透明水彩か不透明水彩かが決まることから，紙色の影響も考慮する必要がある．また水彩画は水が顔料を運び，それが乾くことによって，描かれた領域の端に顔料が溜まることや，重ね塗によって色に深みが出る．顔料のこれらの粒子としての性質が，水彩画的な特徴をもたらしている．

このように，水彩画風の表現を精密に行おうとすれば，絵の具の紙への着色後，最終的な発色が決まるまでの複雑なプロセスを再現する必要があるため，水彩画の持つ多くの特徴を完全に再現できるような手法は，現時点では確立されていない．

1.2 水彩ブラー

この NPR による動画生成では，シャワードア効果 [14] など写実的な映像生成にはない課題があるほか，モーションライン [1] やカトゥーンブラー [15] のような非写実的な表現独特の動画表現も存在する．

すでに水彩画風の表現による動画の作成も行われているが [6–8]，既存のものはフレーム間の一貫性 [14] を考慮しない静止画の動画化であるか，水彩画の特徴であるブラシ

[1] 和歌山大学システム工学研究科
Graduate School of Systems Engineering，Wakayama University
[2] 和歌山大学システム工学部
Faculty of Systems Engineering，Wakayama University
[a)] tokoi@sys.wakayama-u.ac.jp

ストロークの動画化にとどまっており，セル画風表現におけるカトゥーンブラーのような，動きそのものを表現に取り入れるアプローチはなされていない．

本研究はカトゥーンブラーに着想を得て，NPR のうちの水彩画風の表現に対して，その重要な要素である絵の具の引き延ばしやかすれを動画表現に取り入れようとするものである．ここではそれを水彩ブラーと呼ぶことにする．本稿ではこの手法をプログラム可能なグラフィックスハードウェア (Graphics Processing Unit, 以降 GPU と呼ぶ) 上に実装し，3D CG のアニメーションに対してリアルタイムに水彩ブラーを生成する手法を提案する．

2. 関連研究

Curtis らは水彩画風の表現を行うために，絵の具と絵の具の支持体である紙の相互作用を “絵の具の流れ”，“紙に対する色素の着脱”，“絵の具の紙への染み込み” の三つの要素に分類し，紙の高さ（粒状度）の影響を考慮しながら，流体方程式にもとづいて物理的にシミュレーションによって求め，それぞれを Kubelka-Munk の理論 [16] をもとに合成する手法 [4] を提案している．この手法は非常にリアルな水彩画風イメージが得られるが，計算量が多く，リアルタイムにレンダリングすることは困難である．

顔料や紙をボリュームとしてモデル化する手法も提案されている．Takagi らは色鉛筆画風のレンダリングにおいて，色鉛筆の顔料とその支持体である紙の繊維をボリュームとしてモデル化することによって，色鉛筆画の特徴である鉛筆ストロークのかすれの再現に成功している [2]．しかし，画像の生成にはボリュームレンダリング手法を用いており，これも全体が動的に変化するシーンをリアルタイムにレンダリングすることは難しい．

これらの物理シミュレーションベースの手法に対し，GPU のテクスチャマッピング機能を活用してリアルタイムにレンダリングする手法が考案されている．これはトゥーンシェーディングを利用して陰影付の簡易化を行い，紙テクスチャを用いて顔料や紙による歪みを表現し，模擬的に水彩画風表現を得る [5, 9]．また Sloan らは特徴的なタッチを再現する手法として，Lit-Sphere と呼ばれる手法を提案している [17]．この手法では，まず手書き又は絵画から陰影付けの元となる球状画像を抜き出し，その球状画像を形状モデル面の法線に対応させてマッピングすることで，特徴的なタッチを再現した陰影を得ることができる．ただし，この手法は三次元空間上の光源の位置を考慮した計算ができないため，動的なアニメーションの生成には適していない．

Luft らは GPU を用いてレンダリングした画像に対してフィルタを適用し，作成された複数のテクスチャを合成して水彩画風の映像をリアルタイムに生成する手法 [6–8] を提案している．樹木のような細かな構造を持つ 3D モデルの可視判定において，デプスバッファを平滑化してデプステストを行う [8] ことにより，オブジェクトの滑らかな移動やフレーム間の一貫性の維持に加えて，樹木の詳細形状の水彩画風の表現やソフトシャドウの生成に成功している．

一方，セルアニメーション風のレンダリングにおいては，Lake らにより GPU の機能を活用したリアルタイムレンダリング手法 [1] が提案されている．セルアニメーション風のレンダリングでは細かなテクスチャがあまり用いられないため，この手法ではフレーム間の一貫性は配慮していないが，移動するオブジェクトの後にモーションラインを生成することを試みている．

川岸らはこのモーションラインを拡張し，動きを強調するモーションブラー効果のひとつであるカトゥーンブラー [15] を提案している．これは 3D CG のセルアニメ風レンダリングにおいて，“線の効果”，“残像の効果”，“ゆがみの効果” の三種類の効果を生成する．

本研究はこのカトゥーンブラーに着想を得て，水彩画風のリアルタイムレンダリングにおいて，ブラシストロークではなく，オブジェクト自体の移動に伴う絵の具の引き延ばしやかすれの効果である水彩ブラーを動的に生成する手法を提案する．

図 1　実現した水彩画の特徴
Fig. 1　Implemented characteristics of watercolor paint

3. 提案手法

提案手法について，まず水彩画風の陰影を求める手法について述べ，その後，水彩ブラーを生成する手法について述べる．

3.1 水彩画の特徴

Curtis [4] らは水彩画の特徴を，(a) かすれ効果，(b) 塗りの周辺部の色が濃くなる効果，(c) 水のにじみ効果，(d) 粒状効果，(e) ぼかし効果，(f) 重ね塗り効果，の 6 種類に分類した．

本研究では，このうちオブジェクトの塗りつぶし部分では (b) 塗りの周辺部の色が濃くなる効果と (d) 粒状効果の実現を行い，水彩ブラーの部分については，絵の具の引き延ばす効果に加えて (a) かすれの効果の実現を行った (図 **1**)．

3.2 水彩画風の陰影

水彩画風の陰影の生成には，テクスチャを用いる手法を

(a) intensity

(b) color map

図 2　反射光強度とカラーマップ
Fig. 2　Intensity and color map

採用した．これは，まず陰影計算の結果を使って色のテクスチャを参照して，オブジェクトの色を決定する．そして，それに “塗りの周辺部の色が濃くなる効果” を得るためのテクスチャと “粒状効果” を得るためのテクスチャを合成して，水彩画風の陰影を得る．

3.2.1　オブジェクトの色

オブジェクトの色は，Lei らの手法 [5] に準じて，図 **2** (a) に示す反射光強度の変化を，同図 (b) に示す色の一次元テクスチャに (色マップ) より色に置き換えて決定する．このこのテクスチャのサンプリングに用いるテクスチャ座標値 s は，反射光強度をもとに式 (1) により求める．ここで $\mathbf{N}$ はオブジェクト表面の法線単位ベクトル，$\mathbf{L}$ は光源方向単位ベクトル，$\mathbf{H}$ は視線方向単位ベクトルと光源方向単位ベクトルの中間ベクトルを正規化したものである．

$$s = 0.5(max(\mathbf{N}\cdot\mathbf{L}, 0) + max(\mathbf{N}\cdot\mathbf{H}, 0)) \tag{1}$$

これを用いて式 (2) により色マップ T_{color} をサンプリングし，図 **3** (a) に示すオブジェクトの色 C_{object} を得る．ここで $texture(T, s)$ は，テクスチャ T のテクスチャ座標 s の位置の値を取り出す関数である．

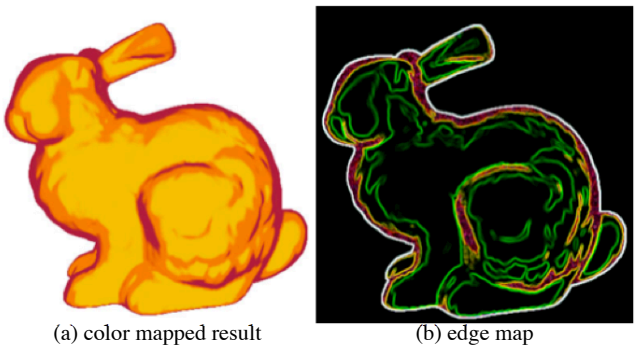
$$C_{object} = texture(T_{color}, s) \tag{2}$$

3.2.2　塗りの周辺部の色が濃くなる効果

塗りの周辺部は絵の具の顔料の集中により色が濃くなる．これを再現するために，まず 3.2.1 の処理によって得られた画像の各画素に対して式 (3) の変換を実行する．T_{gran} は紙表面の粒状性を表す一次元テクスチャである．これに対して Sobel フィルタを適用することにより，輪郭線を強調して塗りの周辺部の色を強調し，その部分や特に陰影の暗い部分に粒状感を与えている．

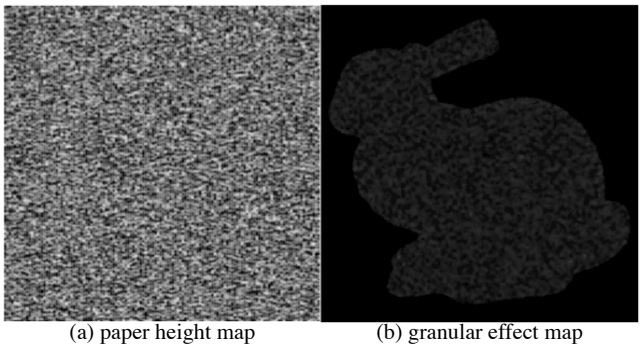
$$C_{gran} = texture(T_{gran}, max(\mathbf{N}\cdot\mathbf{L}, 0), max(\mathbf{N}\cdot\mathbf{H}, 0)) \tag{3}$$

図 3 (b) に Sobel フィルタの結果を示す．



(a) color mapped result　(b) edge map

図 3　オブジェクトの色とエッジマップ
Fig. 3　Mapping results of color map and edge map



(a) paper height map　(b) granular effect map

図 4　紙の高さマップを使った粒状効果
Fig. 4　Granular effect using paper height map

3.2.3　粒状効果

絵の具が紙の凹凸の低い場所に粒状になって溜まる効果を再現する．入力データとして図 **4** (a) に示すスクリーン上の紙の凹凸の高さを設定した二次元の高さマップ T_{height} を用意する．これを式 (4) によりオブジェクトのスクリーン上の位置から決定される二次元のテクスチャ座標値 $\mathbf{t}$ によりサンプリングして，同図 (b) に示すオブジェクト表面上の紙の高さ h を得る．h は 0 から 1 の値を持つ．

$$h = texture(T_{height}, \mathbf{t}) \tag{4}$$

3.2.4　陰影の生成

塗りつぶしによる最終的な陰影は，Lei らの手法 [5] に準じて，式 (5) により求める．C_{paper} は紙の色である．

$$\begin{aligned} C_{output} &= (1-c) + C_{object}\, c \\ &- (C_{sobel}\, s + C_{paper}(1-h)\, p) \end{aligned} \tag{5}$$

ここで c はオブジェクトの色の濃さ，s は塗りの端が暗くなる効果，p は粒状効果を制御する変数である．それぞれ 0 から 1 の値を設定する．c，s，p の重みをそれぞれ 0.6, 0.1，0.3 としたときの出力画像を図 **5** に示す．

3.3　水彩ブラーの生成

水彩ブラーは図 **6** のようにオブジェクトの移動方向の背後に生成する．この部分には水彩画の特徴である “かすれ

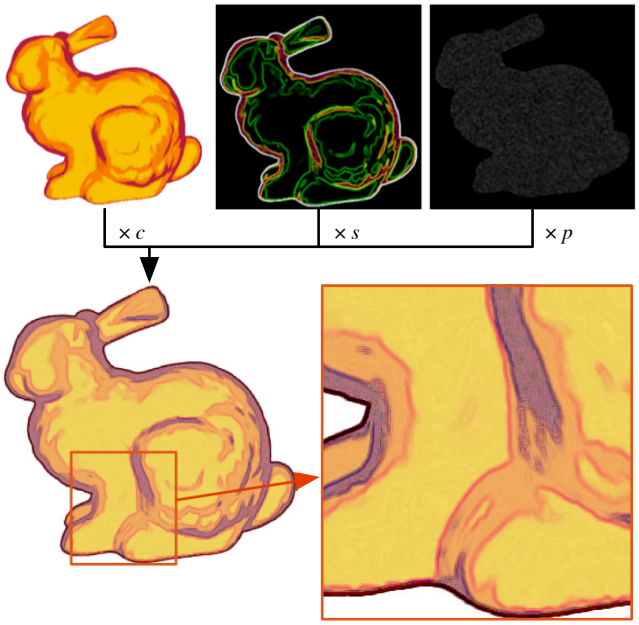
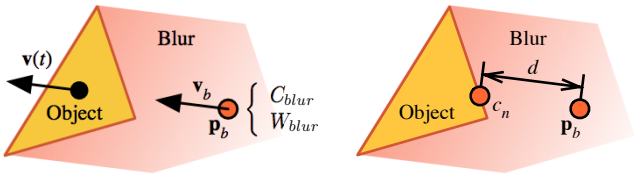


図 5　オブジェクトの陰影
Fig. 5　Shaded image of a object



(a) The amount of paint and water　(b) Base blur color and the distance

図 6　水彩ブラーの色
Fig. 6　The color of watercolor blur

効果”，“粒状効果” を動的に再現する．そのために，スクリーン上の一点 $\mathbf{p}_b$ における，移動するオブジェクトによって残された絵の具の量 C_{blur} と水の量 W_{blur} を決定する．

3.3.1　絵の具の量

画面上のある画素における水彩ブラーの色 C_{blur} は，オブジェクトがその画素のスクリーン上の位置 $\mathbf{p}_b$ 上を通過したときの速度 $\mathbf{v}_b$ から，式 (6) により求める．ここで C_{paper} は紙色，C_n は $\mathbf{p}_b$ から $\mathbf{v}_b$ 方向にある最も近いオブジェクトの境界部分の色，d はそこまでの距離である．また a はブラーの絵の具の量を制御する 0 から 1 の定数，k はブラーの伸びの量を制御する任意の定数である．

$$C_{blur} = \begin{cases} C_n + (C_{paper} - C_n)\left(\frac{2d(1-a)}{k|\mathbf{v}_b|}\right), & a \geq 0.5 \\ C_n + (C_{paper} - C_n)\left(\frac{2da}{k|\mathbf{v}_b|} + (1-2a)\right), & a < 0.5 \end{cases} \quad (6)$$

絵の具の量 C_{blur} は，$a \geq 0.5$ ならオブジェクトの色 c_n を基準として，d にしたがって紙色 c_{paper} に近づけるが，$a < 0.5$ なら基準の色を紙色に近づける (図 7).

3.3.2　水の量

絵の具のかすれを再現するために，水彩ブラーの色 C_{blur} 対して，水彩ブラーの水の量 W_{blur} と紙の凹凸をもとにマスキングを行う．W_{blur} は式 (7) により求める．w はブラーの水の量を制御する 0 から 1 の定数である．最終的な水彩ブラーの色は式 (8) により求める．

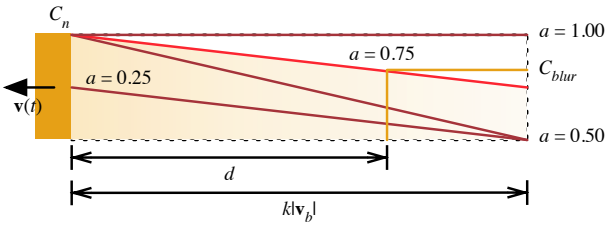


図 7　絵の具の量の変化
Fig. 7　Variation in the amount of paint

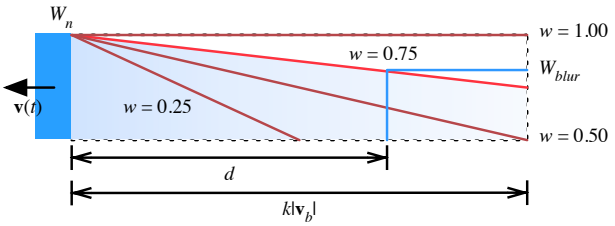


図 8　水の量の変化
Fig. 8　Variation in the amount of water

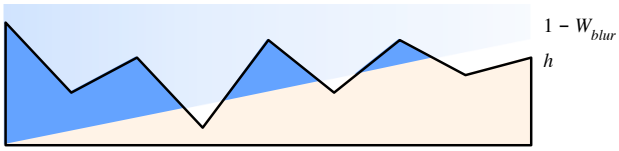


図 9　紙の高さによるマスキング
Fig. 9　Masking by paper height

$$W_{blur} = \begin{cases} 1 - \frac{2d(1-w)}{k|\mathbf{v}_b|}, & w \geq 0.5 \\ 1 - \frac{d}{2wk|\mathbf{v}_b|}, & w < 0.5 \end{cases} \quad (7)$$

水の量 W_{blur} は，$w \geq 0.5$ なら水彩ブラーの長さ $k|\mathbf{v}_b|$ まで保持するが，$a < 0.5$ なら途中で尽きるものとする (図 8).

3.3.3　マスキング

最終的な水彩ブラー部分の色 C_{out} は，この水の量 W_{blur} と紙の高さマップ T_{height} の $\mathbf{p}_b$ の位置の高さ h との比較により，式 (8) により決定する．

$$C_{out} = \begin{cases} C_{paper}, & 1 - W_{blur} \geq h \\ C_{blur} - C_{paper} W_{blur}(1-h), & 1 - W_{blur} < h \end{cases} \quad (8)$$

絵の具は水によって運ばれるので，水の量が少なくなれば紙は着色されず，紙色が現れる．

3.4　速度の算出

次に，オブジェクトが $\mathbf{p}_b$ 上を通過したときの速度 $\mathbf{v}_b$ を求める方法について解説する．$\mathbf{p}_b$ はオブジェクトの内部にないため，この上を通過したオブジェクトを過去のフレームにさかのぼって判定する必要がある．そこで提案手法では，速度バッファを使ったスクリーン空間法にもとづくモーションブラー手法 [18–20] を採用した．

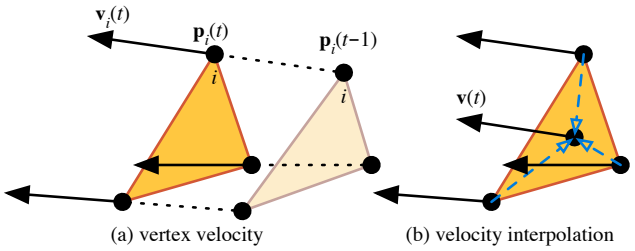


図 **10** 速度バッファの作成
Fig. 10 Generating velocity buffer

3.4.1 速度バッファの作成

速度バッファはオブジェクトをレンダリングする際，オブジェクト表面の色の代わりに，その速度ベクトルを用いたものである．これは表示画面上ではなく，テクスチャメモリ上にレンダリングする (Render to texture).

まず，オブジェクトのレンダリング時に，スクリーンへの投影像を求めるために算出した頂点 i の位置 $\mathbf{p}_i(t)$ を保存しておく．そして，次のフレームのレンダリング時に，保存しておいた前のフレームの頂点の位置との差から，式 (9) により現在のフレームにおける頂点 i の速度 $\mathbf{v}_i(t)$ を求める (図 **10** (a))．ここで t はそのフレームの時刻であり，フレームの時間間隔は 1 とする．

$$\mathbf{v}_i(t) = \mathbf{p}_i(t) - \mathbf{p}_i(t-1) \tag{9}$$

この処理は GPU の Transform Feedback 機能を使って頂点の位置を Feedback Buffer に保存しておけば，CPU を介すこと無く GPU 上で完結できる．

そして，こうして求めた頂点の速度を頂点の色の代わりに使って，フレームバッファへの描き込みを行う．この用途のフレームバッファを，色を描き込むフレームバッファ（色バッファ）と区別して，速度バッファと呼ぶ．

ポリゴンをフレームバッファに描き込む際，ポリゴンの頂点のデータはラスタライザによって線形補間され，ポリゴン内部の画素の値に設定される (図 10 (b))．したがって，このポリゴンの各画素には頂点の速度を線形補間したものが格納される．この処理では色と速度の二つの画像を生成する必要があるが，GPU の Multiple Render Target (MRT) 機能を使えば，これらは同時に作成できる．

3.4.2 速度の取得

フレームとフレームの間はオブジェクトの移動速度が一定であると見なせば，$\mathbf{v}_b$ は現在のフレームのオブジェクトのスクリーン上の速度 $\mathbf{v}(t)$ に等しい．そこで，作成した速度バッファを参照して，オブジェクトが水彩ブラー部分の画素上を通過したときの速度 $\mathbf{v}_b$ を求める．

Rosado の手法 [18] はデプスバッファを参照することによって，スクリーン空間法でありながら，視点からの距離に応じたブラーの量を生成できるが，水彩ブラー部分のように速度バッファのオブジェクトの存在しない領域には，速度が書き込まれない．

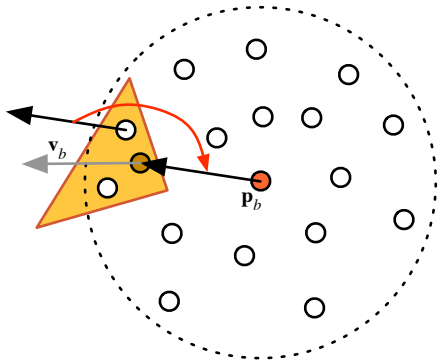


図 **11** 速度バッファのサンプリング
Fig. 11 Sampling velocity buffer

そこで Green は GPU のジオメトリシェーダを用いて，オブジェクトを速度方向に伸ばした掃引図形を描くことを提案した [19]．しかし，この方法はオブジェクトの形状が色と速度で異なるため，これらを同時に描き込むことができず，描画のパスが一つ増えてしまう．

これらに対し，再構成フィルタを用いる McGuire の手法 [20] は，描画のパスを増加させずに見かけ上高品位なブラーを生成できる．しかし本研究では，オブジェクトそのもののぶれを再現することが目的ではないため，実装が簡単な以下の手法を用いた．

(1) あらかじめ色バッファと速度バッファをクリアしておく．速度バッファとして使用するフレームバッファをクリアする際は，アルファ値を 0 にしておく．
(2) まず，Rosado の手法 [18] と同様に，速度バッファのオブジェクトの存在する部分に速度を描き込む．速度バッファに速度を描き込む際は，アルファ値を非 0 にしておく．また，同時に色バッファにも色を描き込んでおく．
(3) 表示領域全体を覆う一枚の矩形ポリゴンを描画する．その際，各画素において以下の処理を行うよう GPU のシェーダプログラムを設定しておく．
  (a) 描画する画素の位置の速度バッファのアルファ値が非 0 なら，色バッファの値を出力する．
  (b) 描画する画素の位置の速度バッファのアルファ値が 0 なら，その周りの速度バッファをランダムにサンプリングする (図 **11**).
  (c) その結果，アルファ値が非 0 のものが見つかったら，その速度を描画する画素の位置に加えて，前フレームにおいてその画素の位置にあった画素の位置を推定する．
  (d) その位置のアルファ値が 0 なら，(a) から繰り返す．
  (e) その位置のアルファ値が非 0 なら，そこに描き込まれている速度を，描画する画素の上を通過したオブジェクトの速度 $\mathbf{v}_b$ として水彩ブラーの色を求め，それを出力する．

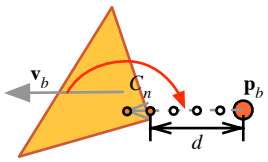
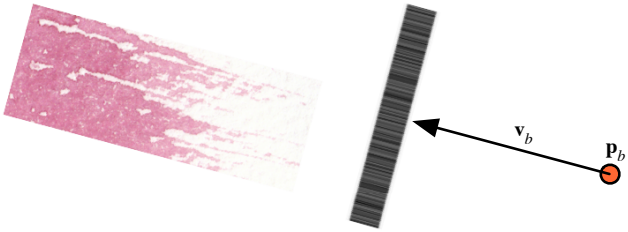


図 12　オブジェクト境界までの距離の算出
Fig. 12　Calculation of the distance to the object border



(a) Blur by brush　　(b) One dimensional brush pattern

図 13　筆の毛によるかすれ
Fig. 13　Blur by brash

3.4.3　オブジェクト境界までの距離の算出

描画する画素から $\mathbf{v}_b$ 方向にある最も近いオブジェクトの境界までの距離 d と，その部分の色 C_n は，以下の手順により求める．

(1) 速度バッファ上で描画しようとする画素のスクリーン上の位置 $\mathbf{p}_b$ とその $\mathbf{v}_b$ 方向の先の位置 $\mathbf{p}_b + \mathbf{v}_b$ の間を等間隔にサンプリングする (図 **12**).

(2) 速度バッファのアルファ値が非 0 であれば，$\mathbf{p}_b$ からその位置までの距離を d とする．またオブジェクトの境界部分の色 C_n には，その位置からオブジェクトの内部のサンプリング点の色を平均したものを用いる．

3.5　筆の毛によるかすれ

以上の手法により生成された水彩ブラーは，紙の凹凸によるかすれは表現されているものの，筆の毛に付着した絵の具の量の不均一さ (図 **13** (a)) は反映されていない．そこで，速度 $\mathbf{v}_b$ に対して垂直に一次元テクスチャを配置し (同図 (b))，オブジェクト境界の絵の具の量を制御した．

3.6　ゆがみの効果

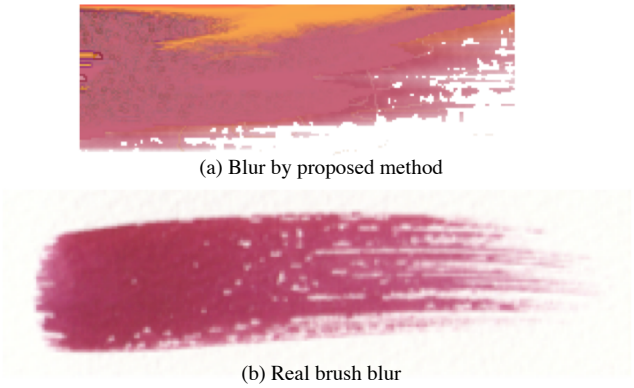
オブジェクトの速い動きを強調するために，「ゆがみの効果」[15] の実装を行った．これは頂点の法線ベクトルと頂点の速度ベクトルの内積が負である頂点に対して，その位置に速度ベクトルに前述の内積値とオブジェクトの中心からの距離に比例した係数をかけたものを加えて実現した．

4.　実験結果

図 **14** に提案手法による生成画像を示す．また図 **15** に水彩ブラー部の詳細と現実の筆による描画との比較を示す．

実験には CPU Intel Core i5 2.67GHz，メインメモリ 4GB，OS Windows 7 Home Premium 32bit，GPU NVIDIA GeForce GT 240，ビデオメモリ 1GB のパソコンを使用した．またモデルには三角形数 69451 の Stanford Bunny を使用した．これを表示解像度 1920 × 1080 画素の画面にリアルタイム (フレームレート 60fps) で描画できた．

図 14　提案手法による水彩ブラー
Fig. 14　The watercolor blur by proposed method



(a) Blur by proposed method

(b) Real brush blur

図 15　提案手法と現実の筆との比較
Fig. 15　Comparison of the proposed method and the real brush

5.　課題

提案手法は，現時点ではオブジェクトを速く動かしたときに，アーティファクト (図 **16**) が発生することがある．これは採用したスクリーン空間法によるモーションブラー手法の問題に起因すると考えられる．品質と効率の向上を図るには，手法の変更も検討する必要がある．

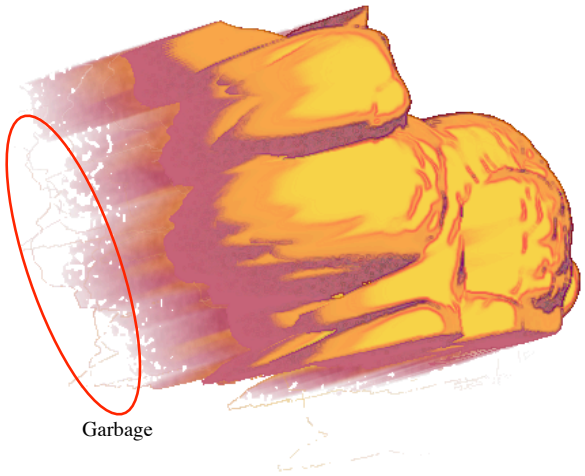


図 **16** ゴミの発生
Fig. 16 Problem of generating garbage

6. おわりに

水彩画風レンダリングにおいて，オブジェクトの動きに伴う絵の具の引き延ばしを再現し，その部分にかすれを伴う動的なボケを生成する手法を開発した．提案手法はスクリーン空間法に基づき，GPU の機能を使用してリアルタイムに処理可能である．

参考文献

[1] Lake, A., Marshall, C., Harris, M. and Blackstein, M.: Stylized rendering techniques for scalable real-time 3D animation, *NPAR '00 Proceedings of the 1st international symposium on Non-photorealistic animation and rendering*, Association for Computing Machinery, pp. 13–20 (2000).
[2] Takagi, S. and Fujishiro, I.: Volumetric modeling of colored pencil drawing, *Proceedings of Seventh Pacific Conference on Computer Graphics and Applications*, Springer Berlin Heidelberg, pp. 250–258, 329 (1999).
[3] Sousa, M. C. and Buchanan, J. W.: Computer-Generated Graphite Pencil Rendering of 3D Polygonal Models, *Computer Graphics Forum*, Vol. 18, pp. 195–208 (1999).
[4] Curtis, C. J., Anderson, S. E., Seims, J. E., Fleischer, K. W. and Salesin, D. H.: Computer-Generated Watercolor, *SIGGRAPH '97 Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, Association for Computing Machinery, pp. 421–430 (1997).
[5] Lei, S. I. E. and Chang, C. F.: Real-Time Rendering of Watercolor Effects for Virtual Environments, *Advances in Multimedia Information Processing*, Springer Berlin Heidelberg, pp. 474–481 (2004).
[6] Luft, T. and Deussen, O.: Interactive watercolor animations, *Proceedings of Pacific Graphics '05*, Springer Berlin Heidelberg, pp. 7–9 (2005).
[7] Luft, T. and Deussen, O.: Real-Time Watercolor for Animation, *Journal of Computer Science and Technology*, Vol. 21, pp. 159–165 (2006).
[8] Luft, T. and Deussen, O.: Real-time watercolor illustrations of plants using a blurred depth test, *NPAR '06 Proceedings of the 4th international symposium on Non-photorealistic animation and rendering*, Association for Computing Machinery, pp. 11–20 (2006).
[9] Bousseau, A., Kaplan, M., Thollot, J. and Sillion, F. X.: Interactive watercolor rendering with temporal coherence and abstraction, *NPAR '06 Proceedings of the 4th international symposium on Non-photorealistic animation and rendering*, Association for Computing Machinery, pp. 141–149 (2006).
[10] Lee, J.: Simulating oriental black-ink painting, *Computer Graphics and Applications*, Vol. 19, pp. 74–81 (1999).
[11] Lee, J.: Diffusion rendering of black ink paintings using new paper and ink models, *Computer & Graphics*, Vol. 25, pp. 295–308 (2001).
[12] 川嵜敬二，中丸幸治，大野義夫：NPR におけるストローク方向の決定と水墨画調レンダリングへの適用，芸術科学会論文誌，Vol. 3, p. 235?243 (2004).
[13] 森田有紀，田中正幸，鶴野玲治，富松　潔：拡散, 及び, 吸着理論に基づいた染色のビジュアルシミュレーション，情報処理学会研究報告，Vol. 2008, pp. 13–18 (2008).
[14] Meier, B. J.: Painterly rendering for animation, *SIGGRAPH '96 Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, Association for Computing Machinery, pp. 477–484 (1996).
[15] 川岸祐也，初山和秀，近藤邦雄：カトゥーンブラー: セルアニメーションのための非写実的モーションブラー，情報処理学会グラフィクスと CAD 研究会報告，Vol. 2002, pp. 37–42 (2002).
[16] Kubelka, P.: New Contributions to the Optics of Intensely Light-Scattering Materials. Part II: Nonhomogeneous Layers, *Journal of the Optical Society of America*, Vol. 44, pp. 330–334 (1954).
[17] Sloan, P.-P. J., Martin, W., Gooch, A. and Gooch, B.: The Lit Sphere: A Model for Capturing NPR Shading from art, *GRIN'01 Proceedings of Graphics interface 2001*, Canadian Information Processing Society, pp. 143–150 (2001).
[18] Rosado, G.: Motion blur as a post-processing effect, *GPU Gems 3*, Addison-Wesley Professional, pp. 575–581 (2007).
[19] Green, S. and Overview, N.: Stupid OpenGL Shader Tricks, *Advanced OpenGL Game Programming Course, Game Developers Conference* (2003).
[20] McGuire, M., Hennessy, P., Bukowski, M. and Osman, B.: A reconstruction filter for plausible motion blur, *I3D '12 Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, Association for Computing Machinery, pp. 135–142 (2012).