グラフィクスとＣＡＤ 106－2
（２００２．２．２１）

境界エッジアンチエイリアシング回路

比留川　香平[†]　青谷知幸[††]　池戸恒雄[†,††]

本稿は境界エッジアンチエイリアシングハードウェアについて述べる．背景色バッファ，ポリゴンエッジバッファ，そして輝度変更器をハードウェアで実装することで，シングルパスでのポリゴンエッジアンチエイリアシングを可能とした．ポリゴンアウトラインに沿ってエッジバッファに direction, slope, fraction そしてポリゴン ID を記録し複数のポリゴンエッジが 1 つのピクセルグリッドを通りエッジが衝突する場合を解決した．今回の実装ではハードウェアコストは 9byte/pixel となり，レンダリング性能を落とすことなくアンチエイリアシングしたピクセルを毎秒 12 億ピクセルの描画速度で出力する．

Boundary Edge Antialiasing Circuit

Kohei Hirukawa,[†] Tomoyuki Aotani[††] and Tsuneo Ikedo[†,††]

This paper describes the hardware algorithm of boundary edge antialiasing. Our scheme achieves polygon edge antialiasing with a single pass rendering, comprising front and background color buffers, polygon edge buffer and intensity modulator. The edge buffer stores direction, slope, fraction and polygon identifier along polygon outlines in order to solve aliasing at the edges where multiple outlines are contacted or passed through a pixel grid. PEB hardware cost is 9 bytes per pixel. The performance does not change corresponding to the existence of antialiasing process and produces an antialiased pixel at the rate of 1.2 billions per second.

1. はじめに

コンピュータ・グラフィクスの描画はアプリケーション，ジオメトリ，ラスタライズの 3 つのプロセスに分けられる[1]．このうちラスタライズは，単純かつ高速処理が要求されるため一般にハードウェア化されている．

ジオメトリでは多角形の頂点データを座標変換し，ラスタライズではその内部を 1 画素毎に内挿補間 (サンプリング) する．この 1 画素 1 サンプルの結果エイリアシングが生じる．エイリアシングを和らげる技術であるアンチエイリアシング (以下 AA) には大きく分けて全画面 AA とエッジ AA があり，ラスタ化段階での処理が必要でリアルタイム性を得るにはハードウェア化が必須となる．

全画面 AA はアキュムレーションバッファ[2] などマルチパスレンダリングを用いるものがある．これはジオメトリ演算を複数回行う必要があり計算コストが高い．またスーパーサンプリング[3] はコストパフォーマンスに問題があり，特に高解像度の表示システムでのハードウェア化は現実的でない．これらの手法は，サンプリングレートを増すことでエイリアシングを緩和する，ポイントサンプリングによるアプローチである．

一方，エッジ AA の手法は画素に占めるポリゴンの面積比を基本とするエリアサンプリングであり，既存のハードウェアで行うにはデプスソート，2 パスレンダリングが必要である．

エッジ AA ハードウェアとして，A-buffer[4] を改善した提案[5)~8)] があるが，A-buffer は描画画像の複雑度によって要求されるメモリ容量が異なるため，ハードウェア化には適していない．

以上のことをふまえ，AA ハードウェアとしてエッジ AA を下記の理由で選択した．

- 全画面 AA はコストの面から現実的でない．
- テクスチャをマッピングする場合は適宜フィルタリングを行うため，ポリゴン内部ではエイリアシングが際だって目立つことはない．
- オブジェクト境界部にエイリアシングであるジャギーおよびポッピング[9] が見られるが、ポリゴン内部では上述の理由から目立たないため、オブ

† 法政大学工学研究科
Dept. of Engineering, Hosei University
†† 法政大学情報科学部
Dept. Information Sciences, Hosei University

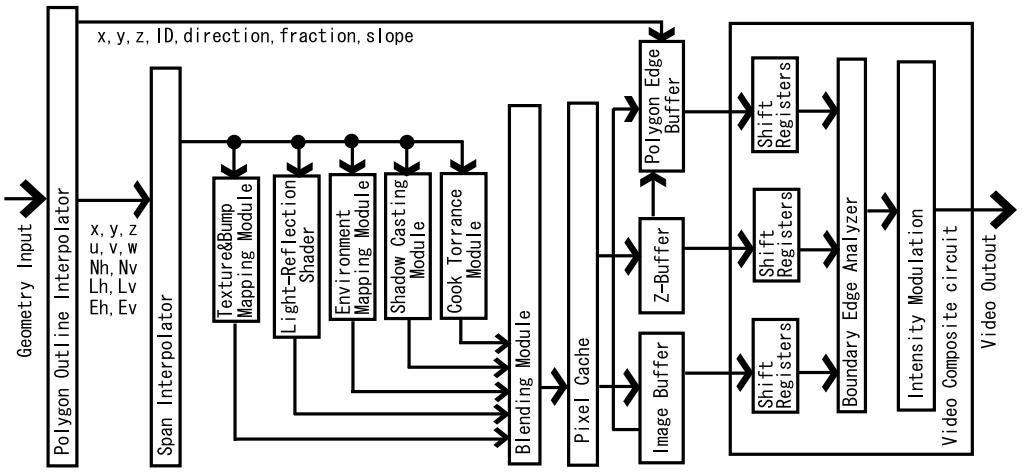

図 1 レンダリングハードウェア

ジェクト境界部のジャギーおよびポッピングが際だつ.

本稿で提案する手法は，Z バッファを用いたポリゴンベースのアーキテクチャへの実装を前提としている．既存のハードウェアの多くではマルチパスレンダリングを必要とする機能を，我々のアーキテクチャではシングルパスで実現可能としたことを特徴とする[10]．よって，AA についても同様にシングルパスによる手法を確立する．また，AA 機能の有無によるレンダリング速度の低下を生じさせないため，ハードウェアのパス上でボトルネックとなりうる複雑な算術計算は避けメモリテーブルと判定回路を基本に構成した．

以降では，まず，レンダリングハードウェアの概要を，次に AA ハードウェアの詳細とそのソフトウェアシミュレーション結果を述べる．最後に考察と比較を行い結論とする．

2. レンダリングハードウェア

図 1 に示すのが AA ハードウェアを含めたレンダリングハードウェアの全体ブロック図である．本レンダリングハードウェアは毎秒 12 億ピクセルの描画速度 (動作周波数 300MHz) である．

2.1 座標変換前処理

ジオメトリにおいては，描画するシーンの視点情報に基づくオブジェクト座標系における頂点データのスクリーン座標系への変換や，クリッピングを行い，頂点データをポリゴン単位でレンダリングハードウェアへ転送する．

2.2 アウトライン補間回路

ジオメトリステージから受け取った 1 ポリゴンの頂点データのうち，y 軸の最小値を始点に，最大値を終点にして，DDA(直線補間回路) が左側および右側を並列に同期を取りつつ，ポリゴンのアウトラインを補間する．2 頂点の x の差分絶対値と y の差分絶対値の大きいほうを長軸と呼び，一方を短軸と呼ぶ．短軸および Z など全てのパラメータの 2 頂点間の差分値を長軸で除算し，各パラメータ毎の傾き (以下 slope) を求める．始点を初期値として補間を開始し，クロックサイクル毎に，長軸には 1 を加算，短軸など他の各パラメータには対応する slope を加算する．短軸小数部を fraction とする．短軸座標値は，fraction の逐次加算に伴うオーバーフローをもって ±1 する．左 DDA と，右 DDA による補間とで，両 DDA が終点に辿りついた時点で，アウトライン補間回路での 1 ポリゴンの処理を終了する．アウトライン補間時，左右 2 つの DDA から出力される各 y 座標値で，もっとも外側となる左右の 2 点をスパン補間回路へ送る．スパン補間回路はこの 2 点を始終点として各水平ラインを補間する．

2.3 スパン補間回路

アウトライン補間回路から得られた座標点を元に DDA を用いて水平方向の補間を行う．補間された値は，各種シェーディングプロセッサに出力される．アウトライン補間回路およびスパン補間回路での処理を図 2 に図示する．

2.4 各種シェーディングプロセッサ

図 1 に示すようにスパン補間回路より補間された座標点，多角形法線，光入射角および視点角などのパラメータは，画素毎にバンプ，ソフトシャドウなどのシェーディングプロセッサへ渡される．それぞれのプロセッサでは並列に計算が実行され，最終的にピクセル 1 点毎の色として合成した後，ピクセルキャッシュ

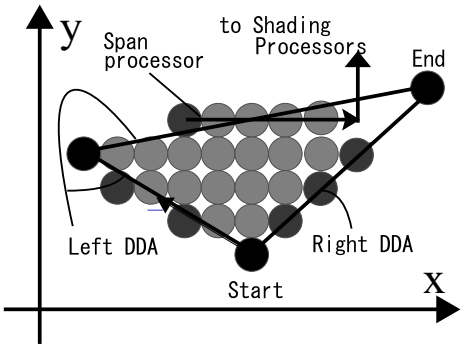

図 2 アウトラインとスパンインタポレータでの処理

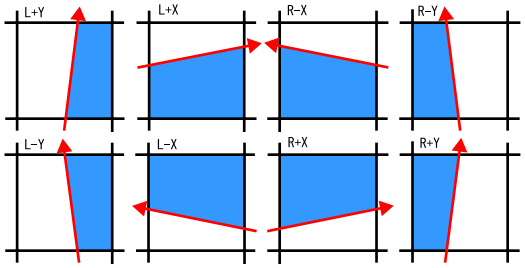

図 3 direction によるエッジパターンの判別．左右 DDA(L|R), X 軸進行方向 (+|−), 長軸 (X|Y) から判別.

に記録される．ピクセルキャッシュは，64 ピクセル分を 1 ブロックとして一括して画像メモリに出力し，Z バッファと比較，可視点をイメージバッファに記録する．以上のようにピクセルキャッシュはレンダリング速度を保証するために実装される．

2.5 ビデオコンポジット回路

全ポリゴンの描画を終了したイメージバッファをモニター (最終出力) へ転送する際，γ 補正など必要となる処理を加える．

以上のアーキテクチャを前提に AA ハードウェアの詳細を述べる．

3. AA ハードウェア

エッジ AA は，アウトライン補間回路，スパン補間回路，ビデオコンポジット回路で構成する．エッジが通るピクセルグリッド内でのポリゴンが占める面積を記

表 1 PEB およびバッファ内容

内容	ビット数	組数	説明
エッジフラグ	1	1	エッジであるか否か
頂点フラグ	1	1	頂点であるか否か
direction	3	2	DDA 進行方向
slope	3	2	1bit 符号付き
fraction	3	2	短軸 小数絶対値
ポリゴン ID	12	1	オブジェクト識別用
背景色バッファ	8	2	エッジ背景用
第 2 Z バッファ	24	1	エッジ背景用
合計容量	9byte/pixel		

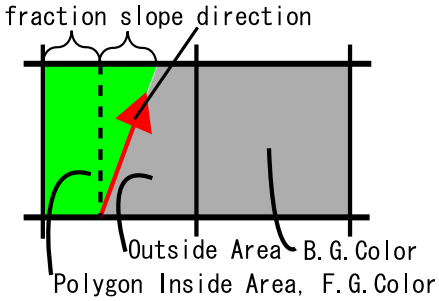

図 4 基本補正方法

録するため，イメージバッファと同解像度の Polygon Edge Buffer(以下 PEB) を定義し，ここに必要なエッジデータを記録・更新する．表 1 に PEB に記録するデータおよび追加バッファを示す．PEB のほかに追加するバッファは背景色バッファx2 と第 2Z バッファである．表において direction は，エッジが Parallel DDA の左右どちらであるか，X 軸における進行方向の正負，および長軸が X であるか Y であるかの合わせて 3 つの情報を用いて，図 3 の 8 パターンに区別した 3bit で定義した値である．面積計算には，これと前節で述べた短軸の fraction と slope を用いる．同一ピクセルに対する複数エッジの描画時のため，direction, fraction, slope は 1 ピクセルにつき最大 2 組まで保持させる．以降の記述の簡略のため，fraction, slope, direction を 1 組のエッジデータと定義，2 組のデータをそれぞれ，AD0, AD1 と定義する．AD0 は最新のエッジデータ，AD1 は AD0 からシフトされた古いエッジデータとする．

AA 処理は、アウトライン補間回路およびスパン補間回路のデータを元にした PEB の更新と，ビデオコンポジット回路による PEB を元にしたイメージバッファのピクセル補正の 2 つのプロセスに分けられる．

基本的には図 4 のように，AD0(direction, slope, fraction) から面積をもとめ，ポリゴン内部の色と，背景色として direction で位置を定める外側の色とを合成し補正する．

3.1 アウトライン補間回路による PEB の更新

アウトライン補間時，1 画素毎の DDA 出力から，該当する座標の PEB を必要に応じて更新する．

複数のポリゴン描画において同一画素を複数のエッジが通るとき，2 エッジデータの場合と 3 エッジデータの場合を，direction から図 5 のようにSAME, DUAL, SMDL, CROSS に領域判別し処理する．2 エッジデータによる領域判別が起こるのは，PEB 更新時に，PEB に AD0 だけを記録している場合である．3 エッジデー

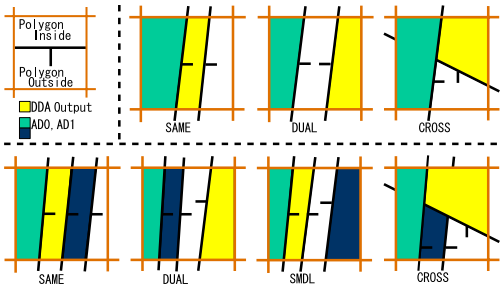

図 5 複数エッジの種別．上段：2 エッジデータの場合，下段：3 エッジデータの場合

タによる領域判別が起こるのは，PEB に AD0,AD1 を記録している場合である．

また，2 エッジデータによる領域判別はイメージバッファ補正時にも用いる．

PEB 更新アルゴリズムは，この判別結果と，DDA 出力 Z 値と Z バッファの比較結果に依る．なお，視点より遠いほど Z 値は大きな値とする．以降，'Z 値小' を，その時点での DDA により補間された z 値が，Z バッファの該当する座標の値より小さい (視点に近い) とし，'Z 値大' を，その時点での DDA により補間された Z 値が Z バッファの該当する座標の値より大きい (遠方) とする．

以下，アウトライン補間時の PEB 更新アルゴリズムの詳細を示す．

エッジフラグが有効か、Z 値小のとき: PEB 更新を検討する．PEB の該当する座標点に記録しているエッジデータの数から以下に示す場合分けをする．

0 の場合: DDA 出力エッジデータを AD0 に記録．

1 の場合: 2 エッジデータの領域判別から以下に示す場合分けをする．

SAME: DDA 出力による面積が小さく，かつ，Z 値大の場合は不可視と判定し更新しない．DDA 出力エッジとピクセルグリッドで分割されたポリゴン内部面積が残りの面積より大きく，かつ，Z 値小の場合は DDA 出力によるポリゴンのみ可視と判定し AD0 に DDA 出力エッジデータを上書き．AD0 とピクセルグリッド間の面積と DDA 出力による面積が等しい場合，Z 値小ならば，AD0 を DDA 出力エッジデータで上書き．他の全ての場合は可視とし，AD0 を AD1 にシフトして DDA 出力エッジデータを AD0 に記録．

DUAL: ともに可視と判定し AD0 を AD1 にシフトして，DDA のエッジデータを AD0 に記録．

CROSS: Z 値小ならば，AD0 に DDA のエッジデータを上書き．Z 値大の場合，更新無し．

2 の場合: 3 エッジデータの領域判別から以下に示す場合分けをする．

SAME: 面積の大きい 2 データを残す．DDA 出力による面積が AD0, AD1 によるどちらかの面積より大きければ，AD0 を AD1 にシフトし，DDA 出力エッジデータを AD0 に記録．

DUAL: AD0, AD1 のうち，面積の大きいほう (以下，主面積という) を残し，(AD0>AD1 なら AD0 を AD1 にシフトして) DDA 出力エッジデータを AD0 に記録．

SAME_DUAL: DDA 出力と同じ direction の AD のうち，主面積のエッジデータを残す．

CROSS: Z 値小ならば，AD1 をクリアして AD0 にエッジデータを上書き (DDA のエッジデータのみ記録)．

3.2 スパン補間回路による PEB の更新

該当ピクセル座標点にエッジフラグがなく，かつ，Z 値小である場合，PEB をクリアする．機構上，スパン補間回路とアウトライン補間回路が同一 pixel 座標点を更新することはない．

3.3 ビデオコンポジット回路での補正

イメージバッファのデータを並列からを直列に変換する．複数エッジを記録している，外側の点もエッジであるなどの条件により補正方法は異なる．我々はシミュレーションにより，視覚的に最善と判断した補正方法を定めた．以下，各場合毎の補正方法について示す．

エッジデータがない場合: そのままイメージバッファのピクセルを出力する．

1 エッジデータがある場合: 外側の点を背景色とし，面積比配分する．ただし，外側がエッジである場合，そのピクセルグリッドに対応する点の第 1 背景色バッファを背景色とする．

2 エッジデータがある場合: 2 エッジデータの領域判別から以下に示す場合わけをする．

SAME: イメージバッファの輝度と AD0 による面積の乗算結果，第 1 背景色バッファの輝度と AD0 と AD1 による面積の差分値の乗算結果，および外側の点のイメージバッファの輝度と 1 から主面積を引いた値の乗算結果を加算して補正値とする．外側がエッジである場合，上述の，外側の点のイメージバッファの輝度と 1 から主面積を引いた値の乗算を，第 2 背景色バッファの輝度と 1 から主面積を引いた値の乗算とする。

DUAL: AD0 と AD1 による面積の合計値から以下に示す場合分けをする．

1 未満の場合: イメージバッファのピクセルの輝度と AD0 による面積の乗算結果，第 1 背景色バッファのピクセルの輝度と AD1 による面積の乗算結果，第 2 背景色バッファのピクセルの輝度と 1 から AD0 と AD1 による面積の合計値を引いた値のの乗算結果を加算して補正値とする．

1 より大きい場合: イメージバッファのピクセルの輝度と AD0 と AD1 による面積の合計値から 1 を引いた値の乗算結果，AD0 が示す外側の点のイメージバッファのピクセルの輝度と AD0 による面積の乗算結果と，AD1 が示す外側の点のイメージバッファのピクセルの輝度と AD1 による面積の乗算結果を加算して補正値とする．AD0(AD1) の外側の点がエッジである場合は，第 1 背景色バッファの輝度を用いる．

1 の場合: ポリゴンの ID を，AD0 が示す外側の点, AD1 が示す外側の点の ID と比較し以下に示す場合分けをする．

全て等しい場合: イメージバッファのピクセルの輝度と AD0 による面積の乗算結果，第 1 背景色イメージバッファのピクセルの輝度と AD1 による面積の乗算結果を加算して補正値とする．

あるいは AD0 が異なる場合: イメージバッファのピクセ

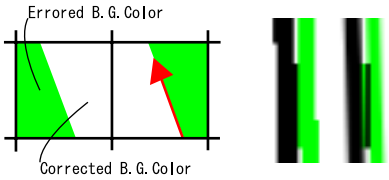

図 6 背景バッファがない場合，外側点がエッジであるときに適正な背景色を得られず補正を誤る，右： 誤補正のサンプル出力画像 (AA 無と AA 有)

ルの輝度と AD0 による面積の乗算結果，AD0 の外側の点のイメージバッファのピクセルの輝度と AD0 による面積の乗算結果を加算して補正値とする．

あるいは AD1 が異なる場合: 第 1 背景色イメージバッファのピクセルの輝度と AD1 による面積の乗算結果，AD1 の外側の点のイメージバッファの輝度と AD0 による面積の乗算結果を加算して補正値とする．

CROSS: イメージバッファの輝度と AD0 による面積の乗算結果，第 1 背景色バッファの輝度と AD1 による面積の乗算結果を加算して補正値とする．

3.4 問題点と解決法

シミュレーションの際の問題点と我々がとった解決法を述べる．

頂点: 頂点には 2 本以上の複数のエッジが入り込んでくる．頂点フラグを立て，周辺ピクセル輝度との平均をとることとした．

1 ピクセル未満の大きさのオブジェクト: 頂点と同じ扱いとなる．

1 ピクセル未満の幅を持つオブジェクト: 前節に示したように PEB に 2 エッジデータを保持することで解決した．

背景誤補正: 図 6 に示す外側がエッジである場合の誤った補正は，8bit x 2 バッファを用いて背景色を保持することで解決した．また，適正な背景色を記録するため 2 つの Z バッファを用意した．

4. シミュレーション結果および考察

前節までに述べた内容を Java でシミュレーションした．オブジェクトには ViewPoint 社等の DXF データを使用した．結果得られた画像を図 7, 8 に示す．なお誌面では鮮明に結果を写せないことをふまえ，`http://www.parims.org/` (R&D project) にも映像を記載した．エッジについて 2x2 スーパーサンプリング以上の成果を出せていることが視認できる．

ハードウェアによる全画面 AA はポイントサンプリングであるため，ポリゴンエッジのピクセルグリッドに対する角度によって AA にバラつきが生じる．しかし本方式はエリアサンプリングであるため，バラつきはない．

他のエッジ AA ハードウェアは 64MByte の外付 DRAM を要するもの[6] など，単一チップを想定しない方式があるが，本方式では単一チップへの実装を想定している．1000x1000 の解像度で 9Mbyte であるが，この程度のサイズであれば数年内に実装可能であると考えられる．

また，オブジェクトを構成するポリゴンが辺を共有する場合の処理は，例えばフラットシェーディングにおいて，辺を共有するポリゴンの面法線の角度差および光源位置によっては AA が必要な場合がある．本方式では背景色バッファを用いたことで角度差などを考慮せずに適正に AA できる．しかし，背景色バッファが 8bit であるため誤差が生じる．イメージバッファと同解像度であれば理想的な処理を行えるが，ハードウェアコスト上 8bit とした．

5. おわりに

本稿ではエッジ AA ハードウェアについて報告した．ハードウェアでのポリゴン描画時に得られるデータを記録し利用することで，レンダリング性能を落とすことなくアンチエイリアシングしたピクセルを出力する．エッジ AA を用いれば全画面 AA より低コストかつ効果的にエイリアシングを和らげることが可能であると考えられる．今後の課題としては，ビット数の精度検証，FPGA への実装・シミュレーションなどが挙げられる．

参 考 文 献

1) Moller, T. and Haines, E.: *Real-Time Rendering*, A K Peters (1999).
2) Haeberli, P. E. and Akeley, K.: The Accumulation Buffer: Hardware Support for High-Quality Rendering, *Computer Graphics (Proc. of ACM SIGGRAPH 90)*, Vol. 24, No. 4, pp. 309–318 (1990). ISBN 0-201-50933-4.
3) Mammen, A.: Transparency and Antialiasing Algorithms Implemented with the Virtual Pixel Maps Technique, *IEEE Computer Graphics & Applications*, Vol. 9, No. 4, pp. 43–55 (1989).
4) Carpenter, L.: The A-buffer, an Antialiased Hidden Surface Method, *Computer Graphics (Proc. of ACM SIGGRAPH 84)*, Vol. 18, No. 3, pp. 103–108 (1984).
5) Lee, J.-A. and Kim, L.-S.: SPARP: a single pass antialiased rasterization processor, *Computers & Graphics*, Vol. 24, No. 2, pp. 233–243 (2000). ISSN 0097-8493.
6) Jouppi, N. P. and Chang, C.-F.: Z3: an eco-

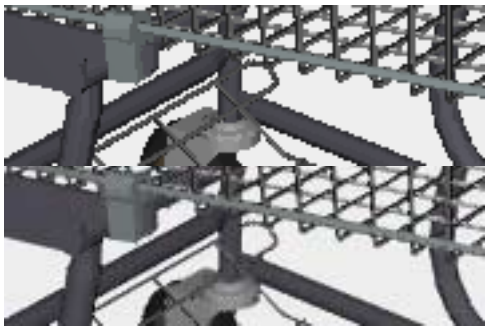

図 7 複雑なオブジェクトのサンプル画像．上: AA 無し， 下: AA 有り

nomical hardware technique for high-quality antialiasing and transparency, *1999 SIGGRAPH / Eurographics Workshop on Graphics Hardware*, pp. 85–93 (1999).

7) Winner, S., Kelley, M., Pease, B., Rivard, B. and Yen, A.: Hardware Accelerated Rendering of Antialiasing Using a Modified A-Buffer Algorithm, *Proc. of ACM SIGGRAPH 97*, Computer Graphics Proc., Annual Conference Series, Los Angeles, California, ACM SIGGRAPH / Addison Wesley, pp.307–316 (1997).
8) Schilling, A. and Straßer, W.: EXACT: Algorithm and Hardware Architecture for an Improved A-buffer, *Proceedings of ACM SIGGRAPH 93*, Computer Graphics Proceedings, Annual Conference Series, Anaheim, California, pp. 85–92 (1993). ISBN 0-201-58889-7.
9) Catmull, E.: A hidden-surface algorithm with anti-aliasing, *Computer Graphics (Proc. of SIGGRAPH 78)*, Vol.12, No.3, pp.6–11 (1978).
10) Ikedo, T. and Ma, J.: The Truga001: A Scalable Rendering Processor, *IEEE Computer Graphics & Applications*, Vol. 18, No. 2, pp. 59–79 (1998).

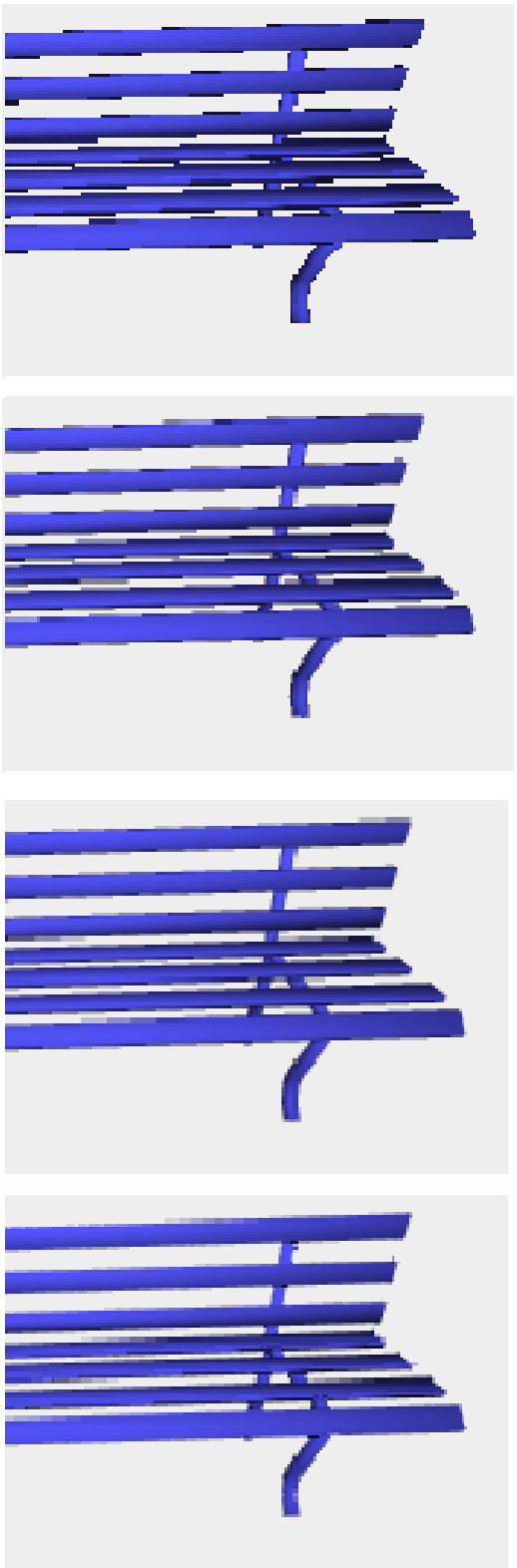

図 8 スーパーサンプリングとの比較画像．上: AA 無し， 中上: スーパーサンプリング (2x2)，中下: スーパーサンプリング (4x4) 下: 我々の方法