Vol.2013-CG-153 No.11
Vol.2013-CVIM-189 No.11
2013/11/28

圧縮テクスチャとビルボードによる広域３次元空間情報のレンダリングシステム

若元　友輔[1,a)]　赤木　康宏[1]　子安 大士[2]　小野　晋太郎[3]　川崎　洋[1]

概要：広域空間を計算機上でレンダリングする手法として，複数の実画像から任意視点の画像を生成する Image Based Rendering（IBR）に基づく手法が提案されている．IBR に基づく手法は，写実的な表現が容易である一方で，実際の形状と計算機内の形状が異なる部分でテクスチャが歪むかあるいは，膨大なデータを用意しなくてはならないという問題がある．本研究では，実画像に対応する 3 次元形状情報を同時に取得し，視点に依存して回転する微小な平面（ビルボード）で構成されるモデルを用いることで，この問題を解決する．さらに，IBR に用いるテクスチャを GPU でのデコードに適した形式に圧縮することで容量を削減する手法を提案する．提案手法を用いた都市空間のレンダリングシステムを実装し，性能評価を行った．

1. はじめに

都市などの広域空間を，実際に計測した 3 次元情報をもとに計算機上で再現する研究や，技術開発は古くから行われている．特に，昨今のコンピュータビジョン・グラフィックス分野の発展に伴い，より本物らしく街並みを表現することが可能となり，それらの技術を応用したカーナビゲーションシステムやドライビングシミュレータ，ストリートビューなどのアプリケーションが増えつつある．

これらのシステムおよびアプリケーションで都市景観をレンダリングしようとした場合，主に二つの手法がある．一つは，建物等の幾何形状モデルを定義し，描画を行う Model Based Rendering（MBR）と呼ばれる手法である．この方法では，モデルの構築は手作業になるため，複雑な形状を描画することが難しく，現実感に乏しい描画結果となってしまうことがある．もう一つの手法は，車載カメラ等による複数視点の画像を用いて，任意の視点の描画を行う Image Based Rendering（IBR）と呼ばれる手法である．この手法は，データ量が十分にあれば MBR より現実感豊かな街並みを容易に描画することが可能になる．しかし，IBR のみで都市のような広域環境を描画する場合，データ量が膨大になりすぎるという問題点がある．

本研究では，車載カメラから撮影した大規模な都市景観映像を入力とし，実際の形状に近い概形モデルに IBR を適用することで，より写実的な表現が可能となるレンダリングシステムを提案する．具体的な手法としては，計測した 3 次元情報を用いて平面（ビルボード）で構成されたモデルを作成し，これに圧縮したテクスチャをマッピングすることで描画を行う．提案手法では，実際の 3 次元情報をもとに配置した概形モデルを用いているため，形状の違うモデルに IBR を行った際に発生するテクスチャの歪みを軽減できる．また，微小面を視点に依存して回転 (ビルボーディング) させることで，任意視点から見たときモデルに穴が空くことを防ぐ．加えて，IBR に用いるテクスチャを GPU でのデーコードに適した形式に圧縮することで，データ量を削減するとともに，リアルタイムでのレンダリングを行うことができる．本研究で提案するレンダリングシステムの有効性を確認するため，各手法を実装し評価実験を行う．

2. 関連研究

都市のような広域空間をレンダリングする研究は広く行われており，それらの研究に必要な手法として，シーンの「取得」と「描画」の二つがある．シーンの取得方法は，おもに航空写真 [2] や車載カメラ [5]，飛行船?などを利用しているものがある．航空写真や飛行船など，遠方から撮影する方式であれば，広範囲を容易に計測できるが，取得するテクスチャの解像度が粗くなってしまう．逆に車載カメラを用いれば，前者のシステムで取得したテクスチャより解像度の良いテクスチャを取得することができるが，広範囲のデータ取得は難しくなる．本研究では，IBR に用いる

[1] 鹿児島大学大学院理工学研究科
[2] 埼玉大学大学院理工学研究科
[3] 東京大学生産技術研究所
[a)] sc108062@ibe.kagoshima-u.ac.jp

テクスチャとして，より解像度の良いテクスチャを用いるため，車載カメラを利用してシーンの取得を行う．

また，計算機による都市景観の描画手法には主に MBR と IBR の二つがある．MBR はハードウェアで効率よく描画されるため頻繁に利用される手法だが，正確な地形データが必要であったり，データサイズが巨大になるといった問題点がある．この問題点を解決するため，IBR によるアプローチが提案されている．たとえば，左右両目用の画像を合成することで自由に広域空間を歩き回ることを実現した研究 [3] や，短冊のスリットを用いた手法 [8] などがある．しかし，IBR は膨大なデータが必要なことから，一般に利用することは難しい．

一方で，実画像をもとにレンダリングを行う技術として，作成したモデルに取得した画像をテクスチャとしてマッピングする Image Based Modeling and Rendering（IBMR）という手法がある．主に，特定のモデル（建物など）を対象としたもの [1] や一般的な投影幾何的性質を利用したもの [4] がある．永塚らは，MBR と IBR を組み合わせた広域空間を効率的にモデリングする手法 [9] を提案した．この手法では，類似する任意視点の画像を圧縮テクスチャにし、GPU 上でデコードすることで、IBR に必要な膨大なデータを縮小し、リアルタイムでのレンダリングを行うことができる．しかし，この手法では IBR を行う平面が固定されているため，実際の形状とは違う部分でテクスチャが歪んでしまい、描画結果に違和感がでてしまう．また，画像を取得した経路以外の場所を描画できないといった問題点がある．

本研究では，山崎らによる微小平面を用いた複雑形状の表示手法 [7] を用いることで，永塚らの問題点であった実際の形状との違いによるテクスチャの歪みや，視点位置の制限を解消する．

3. レンダリングシステムの概要

提案するレンダリングシステムの概要を図 1 に示す．レンダリングシステムは大きく「事前処理」と「レンダリング」の二つに分かれる．

事前処理ではまず，図 1 上部左に示すような 3 台のカメラを車上に直列に並べ，同期撮影を行いながら走行を行うことで，動画像の取得を行う．次に，撮影した動画からステレオ復元することで，3 次元情報（色付の 3 次元点群）を取得する（図 1(a)）．この 3 次元情報を用いてビルボードデータ，およびテクスチャの作成をおこなう（図 1(b)）．このとき，各ビルボード毎に視線方向に応じたテクスチャを全て用意すると，容量が膨大となるので，ビルボード毎にテクスチャを圧縮する工夫をおこなう．本ビルボードの詳細については，4.1 で述べる．

次に、レンダリングの際には視点から見えるビルボードを，事前処理で作成したビルボードデータから選択する

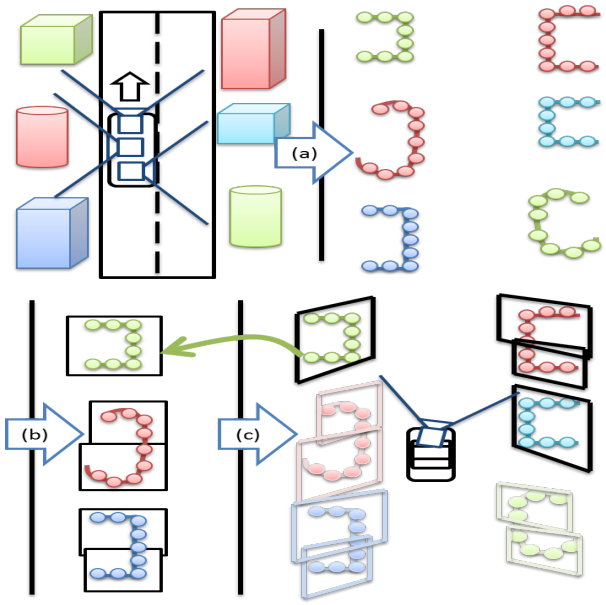


図 1　レンダリングシステムの概要 (a) ステレオ復元による色付 3 次元点群の取得，(b) 点群のグループ化および圧縮テクスチャの生成，(c) ビルボードを用いたリアルタイムレンダリング

（図 1(c)）．そして，ビルボードに対応する圧縮済みのテクスチャを GPU を用いて高速に展開することで，処理全体の高速化を行う．テクスチャの圧縮および展開については 4.2 で詳しく述べる．

4. 提案手法の詳細

4.1 ビルボードデータの作成

ステレオ復元によって取得した 3 次元情報は点群データとして与えられるため，点群に対して 1:1 対応の微小板を描画することでも 3 次元空間を表現できるが，広域空間ではデータサイズが巨大になり，現実的な方法ではない．そこで，本研究では空間全体をボクセルという正規グリッドで分割し，ボクセル内に復元点が十分に含まれると判定されたものは 3 次元形状があるとして，レンダリングの対象とする．ボクセルとビルボードの関連性を図 2 に示す．ビルボード位置およびそのビルボードに適用するためのテクスチャ生成のために，3 次元点群の含まれるボクセルを決定する．その方法は，Octree による空間分割により一定の大きさ（深さ）のノードに達するまで空間を分割し，その葉ノードのみを描画するボクセルとして定義する（図 2(b)）．この時，各ボクセルの中心座標を保存し，各ビルボードの描画に利用する．ビルボードの描画を行う際には，各ボクセルの中心座標対して，ボクセルの 1 面と同じ大きさの平面を発生させ，視線方向に回転させる方法を用いる．本手法は，山崎らによって提案された手法により行う [7]．これらの処理により，形状の違うモデルに IBR を行った時のテクスチャの歪みを軽減することができる．

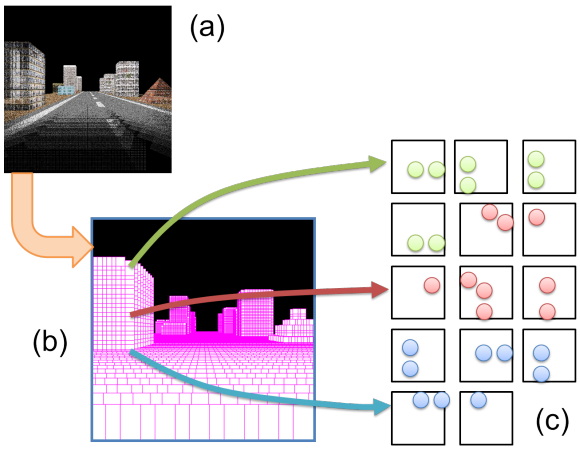


図 2　ボクセルによる空間分割とビルボード．(a) 計測結果の点群データ，(b) 点群を内包するボクセル，(c) 点群の描画されたテクスチャ

4.2　テクスチャの作成

前節で述べたように，本研究では 1 つのボクセルに含まれる点群を 1 枚のテクスチャとして表現する．ここで，あるボクセルの見た目は視点により異なる可能性があるので，本研究では視線方向に応じて複数枚のテクスチャを，1 つのボクセルを表現するビルボードに割り当てる．しかし，ビルボード毎に複数のテクスチャを用いた場合，空間全体の描画に要するテクスチャデータ量が膨大となり，表現できる空間が限られてしまうという問題がある．そこで本研究では，視線方向が多少異なるテクスチャ同士は同様の色領域を含むという性質を利用し，佐藤らの提案する固有空間法を用いたテクスチャ圧縮手法 [5] を適用することで，テクスチャデータの量を削減する．また，この圧縮テクスチャを階層化し，GPU にまとめて圧縮テクスチャを転送することで，個々のテクスチャ描画を行う際のテクスチャ切り替えを削減し，リアルタイムでのレンダリングを実現する．

以下の項では，ビルボードにマッピングするテクスチャのサンプリング手法に関しては 4.2.1 で，固有空間法によるテクスチャ圧縮に関しては 4.2.2 で，圧縮テクスチャの階層化とマルチテクスチャによる復元手法に関しては 4.3 で説明する．

4.2.1　テクスチャのサンプリング

図 3 に示すように，視点からある程度の距離にある 1 つのビルボードは，視点移動し，見える角度が変化した場合でも類似の見た目になる．

そして，このような類似する複数画像列は固有空間法による圧縮が効率よく行えるという性質がある．そこで，本研究では 1 つのビルボードに対して用いる，複数方向の画像をまとめて圧縮することで，テクスチャのデータサイズを縮小する．具体的には，1 枚のビルボードに対して d 枚の類似テクスチャを取得し，この類似テクスチャの集合に

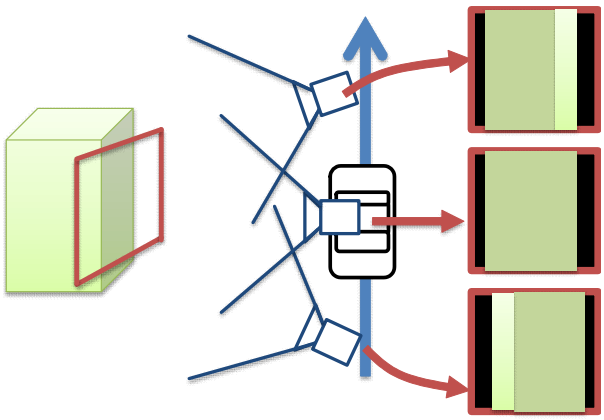


図 3　視線方向による同テクスチャの見え方の違い

固有空間法を適用することで r 枚の固有画像に圧縮することができる（$r < d$）（図 4）．本手法の詳細について以下に詳しく述べる．

4.2.2　固有空間法によるテクスチャ圧縮

固有空間法を用いることで，ある画像列 X_i $(i \in d)$ は式 (1) に示す線形和により復元することができる．ここで，V_k は k 番目の固有画像，w_{ik} は画像 X_i を復元する際の固有画像 V_k の重み係数，ave は画像 X_i の平均色である．また r は，ある累積寄与率を達成するために必要な主成分の枚数である．

$$X_i = \sum_{k=0}^{r}(w_{ik} * V_k) + ave \qquad (1)$$

本研究ではこの画像列 X_i として，1 つのビルボードの複数視点からの見た目を与えることで，ビルボードに用いるテクスチャの圧縮を実現する．テクスチャの復元に必要なパラメータを図 4 に示すような固有画像，係数画像，平均画像の 3 種類の圧縮テクスチャにそれぞれ変換し，保持しておく．

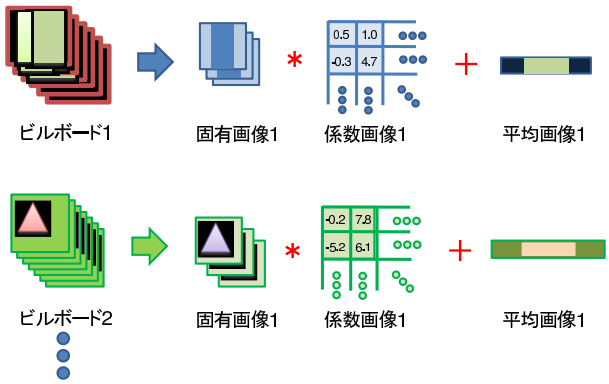


図 4　圧縮テクスチャの構成

4.3　圧縮テクスチャの階層化とマルチテクスチャによる復元手法

レンダリング時は，4.2.2 で作成した 3 種類の圧縮テクス

チャをあらかじめ GPU に転送しておき，積和演算はピクセルシェーダで行う．ピクセルシェーダは高速な並列プロセッサであり，ビデオフレームレートでの処理が実現できる．したがって，一度に複数枚の圧縮テクスチャを GPU に転送できれば，新たにテクスチャを復元する際に改めてテクスチャを作成しなおす必要がなくなり，リアルタイムでのレンダリングが可能になる．

また，テクスチャそのもののサイズに加え，ビルボード毎に使用するテクスチャを切り替える（バインド操作をしなおす）ことは実行速を遅くする原因となる．そこで本研究では，複数枚の圧縮テクスチャをまとめて 1 枚の巨大なテクスチャに梱包し，GPU に転送することで効率化を図る．具体的な手法としては，まず 3 種類のテクスチャの内もっともサイズの大きい固有画像テクスチャを，レンダリングシステムが扱えるテクスチャの最大サイズに合わせてタイリングする．具体的には，実験環境のシステムで扱えるテクスチャの最大サイズは 8192 × 8192（pixel）であったので，タイリングできるテクスチャの枚数は 4096 枚になる．係数画像，平均画像テクスチャに関しては，シェーダ内で効率よく復元を行うため，固有画像に合わせてタイリングを行う．各画像のタイリングの模式図を図 5 に示す．GPU 上のピクセルシェーダでは，固有画像，係数画像，平均画像テクスチャの積和演算を，マルチテクスチャ処理で実現する．

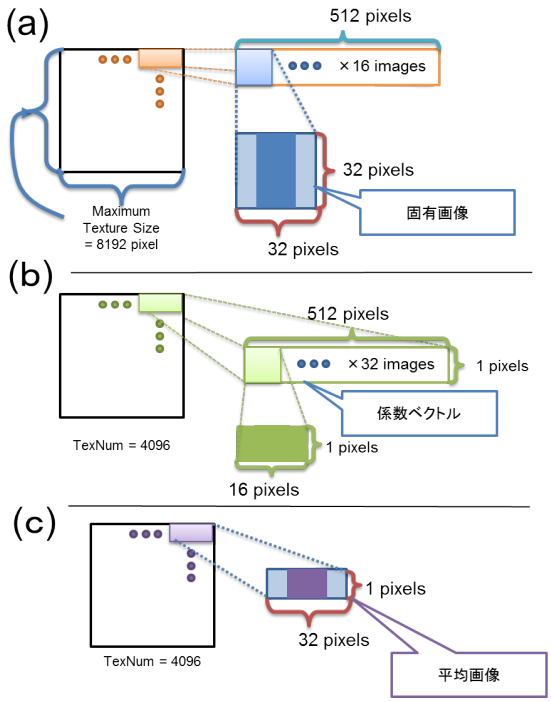


図 5　圧縮テクスチャ画像のタイリング (a) 固有画像，(b) 係数画像，(c) 平均画像

5. 実験

3 節で説明した処理を用いて，シミュレーション環境のレンダリングを行った．実行環境は，Intel Core i7(3.40GHz)，GPU は AMD Radeon HD 7450(4GB) である．またシミュレーション環境は，300m の 2 車線道路を想定し，入力画像として 100 フレーム分の画像を用いた．

シミュレーションに用いた都市モデルの表示例を図 6 に示す．5.1 節では描画速度の向上に関して，5.3 節ではテクスチャ圧縮を用いることに起因する，描画時の画像品質の劣化に関する考察を行う．

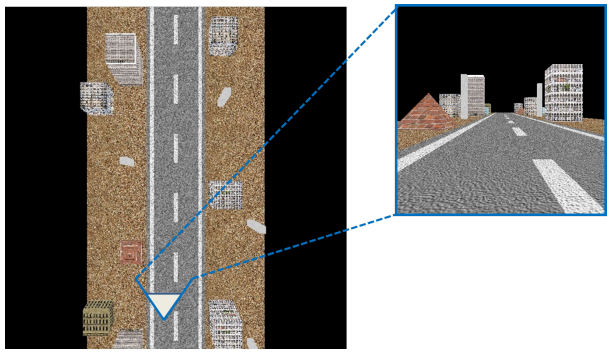


図 6　シミュレーションに用いた都市モデル

5.1 描画速度の向上

本研究では，複数枚のビルボードにそれぞれマッピングされるテクスチャの計算を，階層化した圧縮テクスチャを GPU に転送することで並列処理し，リアルタイムでのレンダリングを実現している．そこで，CPU 上でビルボード毎にテクスチャの計算を行った場合と，提案手法とのフレームレートの差を評価する．表 1 に示すように，CPU で 1 枚ずつテクスチャの計算を行った場合は 11.42fps 程度しかフレームレートが出ていないが，GPU で並列に計算を行った場合は 31.90fps フレームレートが出ており，本研究の提案手法がリアルタイムでのレンダリングに適していることがわかる．

表 1　描画速度

	CPU	GPU
FPS	11.42 (fps)	31.90 (fps)

5.2 データサイズの削減

シミュレーション環境のレンダリングを行うにあたりマッピングされるテクスチャの固有空間法による圧縮をおこなうことで，データサイズの削減を行った．入力テクスチャとして，6 度ずつの角度でサンプリングしたテクスチャ 32 枚（視野：180 度）を用いた．サンプリングしたテクスチャはそれぞれ 32 × 32pixel，3Channel の画像で

あるため、データサイズは合計で 98kbyte になる．一方、圧縮テクスチャは固有画像（32 × 32pixel，3Channel，32 枚），係数画像（16 × 1pixel，3Channel，32 枚），平均画像 (1 × 1pixel，3Channel，32 枚) にそれぞれ圧縮され，データサイズは合計で約 50kbyte となる．したがって，データサイズは約 1/2 に削減できたことになる．

5.3 描画品質の評価

図 8 に提案手法による都市の描画結果を示す．結果では，都市の道路や建物等が入力シーン（図 6）と同様の位置に描画されていることが分かる．また，図 9 に示すように，テクスチャを取得しなかった経路を走行した場合でも，本研究で提案したビルボードモデルにより，テクスチャの歪みが少ない状態でレンダリングできている．

一方で，サンプリングの段階で誤ったテクスチャを取得してしまい，レンダリング結果がおかしくなっている個所がある．この原因は，図 7 に示すような、テクスチャのサンプリング時に別の板の陰になって隠れてしまっている個所を取得していることが考えられる．図 7 の例では，本来建物の壁面が正しいテクスチャとしてサンプリングされなければならない個所で柱の陰となってしまい，柱のテクスチャを誤って取得している．この問題の解決策として，テクスチャを取得する際，板同士のオクルージョンを考慮してサンプリングすることが考えられる．

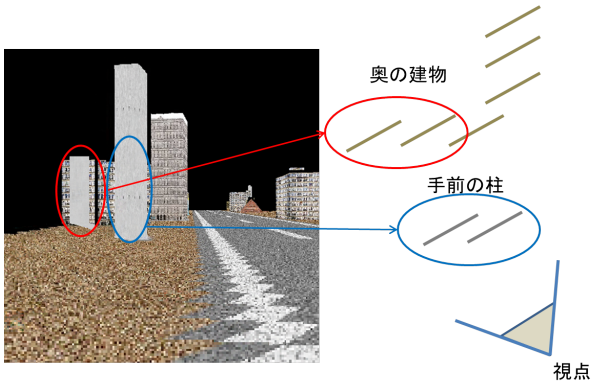


図 7 板のオクルージョンによる誤ったサンプリング

6. まとめ

本研究では，車載カメラで取得したシーンから広域環境の 3 次元情報を獲得し，それをもとに微小なビルボードで構成し，さらに個別のビルボードに IBR を適用することで，少ないデータ量で写実的レンダリングを行う手法を提案した．また，GPU で圧縮テクスチャの復元，描画を行うことで，リアルタイムレンダリングを実現することに成功した．実験では，シミュレーション用の都市モデルを用いて，描画速度および描画品質の評価を行った．

今後，実環境での撮影結果を用いたものに適用することで，より現実感豊かなドライビングシミュレータやストリートビューワーなどを開発できる可能性がある．

謝辞 本研究の一部は，文部科学省科研費（25870570）および内閣府 NEXT プログラム (LR030) の助成を受けて実施されたものである．

参考文献

[1] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, pp. 11–20, New York, NY, USA, 1996. ACM.

[2] Christian Fruh and Avideh Zakhor. An automated method for large-scale, ground-based city model acquisition. *International Journal of Computer Vision*, Vol. 60, pp. 5–24, 2004.

[3] Maiya Hori, Masayuki Kanbara, and Naokazu Yokoya. Arbitrary stereoscopic view generation using multiple omnidirectional image sequences. In *Proceedings of the 2010 20th International Conference on Pattern Recognition*, ICPR '10, pp. 286–289, Washington, DC, USA, 2010. IEEE Computer Society.

[4] P. J. Narayanan, Peter W. Rander, and Takeo Kanade. Constructing virtual worlds using dense stereo. In *Proceedings of the Sixth International Conference on Computer Vision*, ICCV '98, pp. 3–, Washington, DC, USA, 1998. IEEE Computer Society.

[5] Sato Ryo, Shintaro Ono, Hiroshi Kawasaki, and Katsushi Ikeuchi. Real-time image-based rendering system for virtual city based on image compression technique and eigen texture method. In *IAPR International Conference on Pattern Recognition (ICPR08)*, pp. 1–4, 2008.

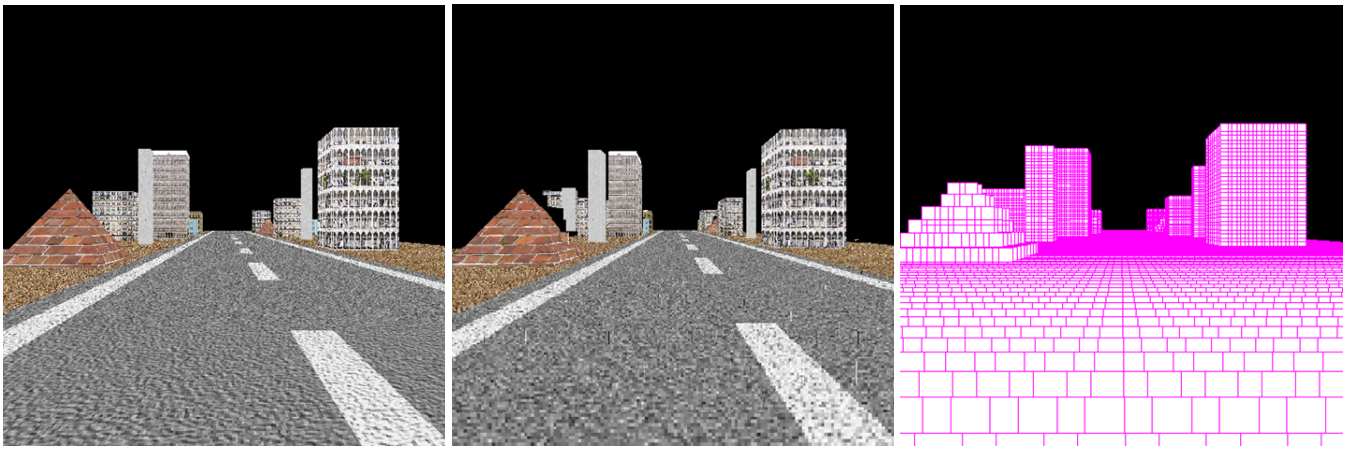
[6] 大倉史生, 神原誠之, 横矢直和. 無人飛行船からの空撮全方位動画像を用いた蓄積再生型拡張テレプレゼンス (¡特集¿複合現実感 5). 日本バーチャルリアリティ学会論文誌, Vol. 16, No. 2, pp. 127–138, 2011-06-30.

[7] 山崎俊太郎, 池内克史, 坂内正夫, 佐川立昌, 川崎洋. 視点依存の微小面を用いた複雑形状の表示法. 第 66 回 計測自動制御学会, パターン計測部会研究会, 産業技術総合研究所臨海副都心センター 4F（お台場）, 2004.

[8] 高橋拓二, 川崎洋, 池内克史, 坂内正夫. 全方位画像を用いた広域環境の自由視点レンダリング. 情報処理学会論文誌, Vol. 42, No. SIG13(CVIM 3), pp. 99–109, 2001.

[9] 川崎洋, 永塚遼, 小野晋太郎, 栗林宏輔, 子安大士, 前川仁, 池内克史. 都市など広域空間の効率的モデリングおよびレンダリング手法について. 情報処理学会 CVIM 研究会, 第 176 巻, pp. 1–7, 2011.

Vol.2013-CG-153 No.11
Vol.2013-CVIM-189 No.11
2013/11/28



[a] Ground Truth　[b] レンダリング結果　[c] ビルボードのみの描画

図 8　テクスチャ取得経路でのレンダリング結果

[a] Ground Truth　[b] レンダリング結果　[c] ビルボードのみの描画

図 9　非テクスチャ取得経路でのレンダリング結果