

# 固有画像を用いたイメージベースによる仮想広域空間の 実時間レンダリング手法

尾池 治郎<sup>†</sup> 川崎 洋<sup>†</sup> 大澤 裕<sup>†</sup> 小野 晋太郎<sup>††</sup> 池内 克史<sup>††</sup>

<sup>†</sup> 埼玉大学 〒338-8570 埼玉県さいたま市桜区下大久保 255

<sup>††</sup> 東京大学 〒153-8505 東京都目黒区駒場 4-6-1

E-mail: <sup>†</sup>{oike,kawasaki,ohsawa}@mm.ics.saitama-u.ac.jp, <sup>††</sup>{onoshin,ki}@cvl.iis.u-tokyo.ac.jp

あらまし 近年、都市のような広域な空間をコンピュータに取り込み仮想空間を生成することが、コンピュータビジョンやコンピュータグラフィックスの分野で注目されている。これを実現する手法として、これまでモデルベースによる手法が主流であったが、大規模で複雑な環境を現実感高く再現するには、これまでのようなモデルベース手法を適用するだけでは難しいことが認識されつつある。一方、大量の画像を蓄積し、これらの実画像を様々な加工・処理し、再構築することで撮影点以外での視点映像を作りだすイメージベースと呼ばれる手法が注目されている。この手法を用いればモデルベースでは困難であった現実感豊かな画像を再現出来る可能性がある。ところがイメージベース手法は、蓄積した画像ファイル群から必要なデータを選択し合成する手法であるため、広域空間のレンダリングにはデータサイズの観点から大きな問題がある。過去に広域空間に適用した例も複数あるが、再現できる仮想空間は限られた範囲内であった。IBR で用いる画像データは隣接データにおける変化が少なく、一般に冗長である。そこで本論文では、この冗長性を考慮し、効率よく全方位画像を圧縮し、また圧縮されたデータを用いて高速にレンダリングする手法を提案する。本手法の有効性を示すために、実際に車載カメラで撮影した都市の映像を用いて実験を行なったところ、良好な結果を得ることが出来た。

キーワード 仮想広域空間, IBR, KL 展開, リアルタイムシステム

## Real-time Image Based Rendering System for Virtual Large-Scale Scene using Eigen Image

Jiro OIKE<sup>†</sup>, Hiroshi KAWASAKI<sup>†</sup>, Yutaka OHSAWA<sup>†</sup>, Shintaro ONO<sup>††</sup>, and Katsushi  
IKEUCHI<sup>††</sup>

<sup>†</sup> Saitama University 255 Shimo-Okubo, Sakura-ku, Saitama City, Saitama 338-8570, JAPAN

<sup>††</sup> Tokyo University 4-6-1 Komaba Meguro-Ku, Tokyo 153-8505, JAPAN

E-mail: <sup>†</sup>{oike,kawasaki,ohsawa}@mm.ics.saitama-u.ac.jp, <sup>††</sup>{onoshin,ki}@cvl.iis.u-tokyo.ac.jp

**Abstract** In this paper, we present an efficient method to synthesize large-scale scenes, such as broad city landscapes. To date, model based approaches have mainly been adopted for this purpose, and some fairly convincing polygon cities have been successfully generated. However, the shapes of real world objects are usually very complicated and it is infeasible to model an entire city realistically. On the other hand, image based approaches have been attempted only recently. Image based methods are effective for realistic rendering, but their huge data sets and restrictions on interactivity pose serious problems for an actual application. Thus, we propose an efficient compression technique for image based method, which uses geometry. The technique can drastically reduce data size without degrading quality of synthesized image. Another strength of the compression technique is that it can be applied for real-time rendering with ordinary PCs.

**Key words** virtual large-scale scene, IBR, KL expansion, real-time system

## 1. はじめに

近年、都市空間を計算機内に仮想的に再現する研究が盛んに行われている。これを実現する方法として、大きく Model Based Rendering(MBR) と Image Based Rendering(IBR) の二つが知られている。

MBR は、ポリゴン等の 3 次元データとその表面のテクスチャ情報を用いることで、あらゆる見えを容易に生成することができる。しかし、都市空間において、街路樹や電柱など細かい形状やテクスチャ等の情報を全てモデル化することは極めて難しい。

一方 IBR では、撮影した画像を直接利用するため、モデル化の難しい複雑な物体であっても容易に描画することが可能であるが、再現する空間すべての画像データを保持せねばならず、データの容量的な問題がある。また、IBR で用いるデータは同じ物体をあらゆる方向から撮影した画像群であるため、一般に冗長である。そこで、この冗長性を考慮し、効率よく画像を圧縮することが出来れば、データ容量を押さえて現実感の高い広域空間の描画が可能となる。

本論文では、車載カメラで撮影した全方位画像を対象とし、EPI(Epipolar Plane Image) を利用することで効率よく、KL 展開により画像を圧縮する手法を提案する。提案する手法は、線形計算で画像復元が可能であるため、高速に IBR 手法により画像合成が実現でき、リアルタイムシステムに応用することができる。

## 2. 街づくりのシステム

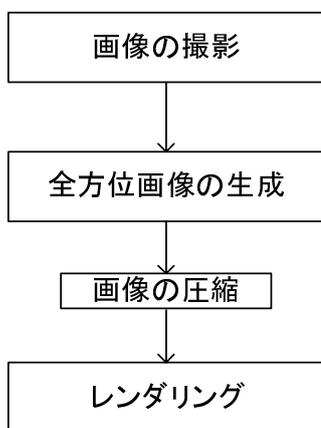


図 1 街づくりのシステムの流れ  
Fig.1 Overview of city modeling.

本論文における街づくりのシステムは、画像の撮影、撮影した画像から全方位画像の生成、全方位画像データの圧縮、圧縮データからのレンダリング、の 4 つの処理に分けることが出来る(図 1)。圧縮とレンダリング部分に関しては 3, 4 章で説明するため、ここでは画像の撮影と撮影した画像からの全方位画像の生成について述べる。

### 2.1 画像の取得

都市空間のデータの撮影には、鏡面型全方位カメラを用いる方法と、複数カメラを用いる方法がある。

鏡面型全方位カメラは、一度に水平方向 360° の画像を撮影することができる利点がある反面、解像度が低く、現在の CCD 密度では現実感の高い都市のモデリングに十分な解像度を得ることが難しい。

一方複数カメラを用いれば、解像度の問題は解決できるが、画像を統合することによる歪みが問題となる。しかし、この歪みについては多くの解決方法が提案されている。また最近では市販されているものもある [1]。特にカメラを図 2 のように一直線上に配置すれば通常のカメラでもカメラの光学中心を一致させることができ、画像を統合する際に都合がよい [2]。そこで、本論文においてもデータの撮影にはこの方式を用いた。

撮影の環境として、車載カメラは等速直線運動を仮定した。これにより、3.1 で説明する EPI を生成することができ、また、EPI を用いることで画像の統合、パターンへのトラッキングなどを効率良く行うことができる。ところが、実際の都市空間において道路は必ずしも直線ではなく、緩やかに歪曲していたり交差点では 90° に曲がっている。そこで、このような場合には車に搭載した GPS やジャイロセンサなどを用いて、道路を直線セグメントに分けることで、対処するものとする。ただし、本実験では、手動によりセグメント処理を行った。セグメント処理の自動化は今後の課題である。



図 2 車載カメラ  
Fig.2 Data capturing car.

### 2.2 全方位画像

撮影された複数画像から図 3 のような全方位画像を生成する。全方位画像の特徴としては、360 度撮影されているために効率的に画像を管理出来ることが挙げられる。また、IBR によるレンダリング手法も提案されている [3] ~ [5]。

ところが、全方位画像は 360 度撮影されているために 1 枚の画像サイズ自体が大きい。さらに、IBR によるレンダリングの場合、わずかな空間を再現するだけでも何千から何万枚と必要であるため、全体の容量が非常に大きくなる。また従来のレ



図 3 全方位画像

Fig.3 Omni-directional image.

ングリング方式では、膨大なデータベースの中からランダムに画像を選ぶ必要があるためメモリ上に全てのデータを保持できず、読み込み時間がかかり描画速度自体が低下してしまう問題もあった。

そこで、全方位画像の隣接するフレームでは変化が少なく、冗長性が高いという性質を利用して、画像を圧縮することを考える。同時にリアルタイムレンダリングのためには、高速に復元できることが望ましい。以下では、圧縮およびレンダリング手法について述べる。

### 3. 画像圧縮手法

#### 3.1 EPI

等速直線運動する移動体に、カメラを進行方向に対して鉛直横向きに設置して撮影した連続画像を、時間軸方向に積層することで、時空間ポリゴンを作ることが出来る。この、時空間ポリゴンをエピポーラ線に平行に切断したものがEPIである。

図4(a)はEPIの例であり、 $u$ 軸は画像の水平方向(epipolar方向)を、 $z$ 軸は時間方向を表している。EPI上での特徴点の動きは直線状の軌跡を描く。また、カメラに対するオブジェクトの奥行き値が大きく変化せず、表面のテクスチャが一様な場合には、オブジェクトはEPI上で『帯』状の軌跡を描く。さらに、複数のオブジェクトが、カメラの進行方向に対して平行に存在しているとすると、オブジェクトの奥行きに大きな変化がないため、図4(a)のように、それぞれのオブジェクトが作る帯は重なること無くほぼ平行な軌跡を描くことになる。一方、奥行きに大きな差がある複数のオブジェクトが存在する時は、図4(b)のように、帯に重なりが現れる。都市などにおいては、通常道路に面して建物が立っているため、図4(b)のようになることが多い。

#### 3.2 ハフ変換を用いたトラッキング

EPIにおいて直線の傾きはオブジェクトの推移を表しているため、このパターンを抽出できればオブジェクトのトラッキングが可能となる。そこでEPIの直線の傾きをハフ変換を用いて検出する。 $\rho$ を座標原点から直線へおろした垂線の長さ、 $\theta$ を垂線と $u$ 軸とのなす角とすると、EPI上の直線は

$$\rho = u \cos \theta + z \sin \theta \quad (1)$$

の式で表される。EPIのエッジ画像にハフ変換を適用し $\theta, \rho$ のパラメータを求める。EPIにおいて $z$ 軸は時間方向を表すため、 $z$ 軸方向に1ピクセル動くことは1フレーム変化することを意味する。よって、式(1)より

$$\frac{\delta u}{\delta z} = -\tan \theta \quad (2)$$

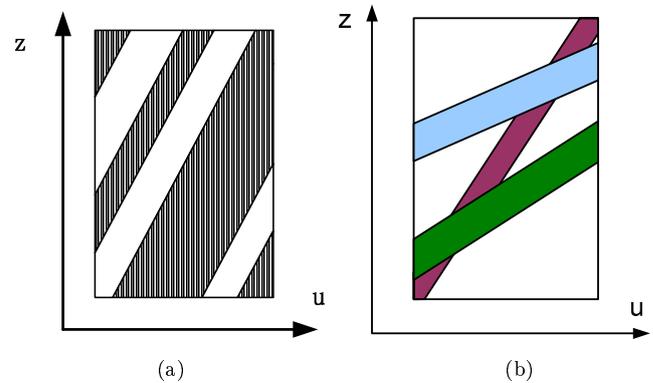


図 4 (a) デプスが等しい時の EPI, (b) デプスの異なる時の EPI  
Fig.4 (a)EPI with different depth and (b)EPI with constant depth.

となり、1フレームでのオブジェクトの $u$ 方向の移動量は $-\tan \theta$ となる。これを用いてオブジェクトのトラッキングが可能となる。

#### 3.3 画像のサンプリング

EPIを作成するためにはカメラを進行方向に対して鉛直横向きになっていることが条件となるが、全方位画像からEPIを作成する際に進行方向の部分を平行化しようとする、平行化画像は無限に広がってしまい(図5)正確なEPIを作成することが出来ない(図6)。また平行化した画像は、進行方向から鉛直横向き付近はカメラの位置によって見えの変化が大きいが、進行方向付近の画像は、平行化すると1ピクセルが拡大されてしまうため、データとしての意味は小さい。

そこで、見えの変化の大きい鉛直横向き付近は密にサンプリングし、データの重要度の低い進行方向付近は疎にサンプリングするものとした。



図 5 鉛直に投影した画像

Fig.5 Warped image.

#### 3.4 KL展開による圧縮

まず画像を図7のように $m \times n$ の小さなブロック単位に分割する。このそれぞれのブロックに関して、 $d$ フレームまで3.2



図 6 長いスパンの EPI

Fig. 6 EPI with wide viewing angle.

のトラッキングを適用し、類似ブロックの集合を得る．図 7 においては、赤の破線で示してある．

こうして生成したブロックの集合を KL 展開し、 $k$  次元の固有ベクトルに圧縮する．これを  $m \times n$  全てのブロックに対して行うことで画像列を圧縮することができる．この時、ブロックの集合は建物の同じ場所を視点を変えながら撮影したものであるため類似度が高く、高い圧縮率が期待出来る．この処理により固有画像が得られる．

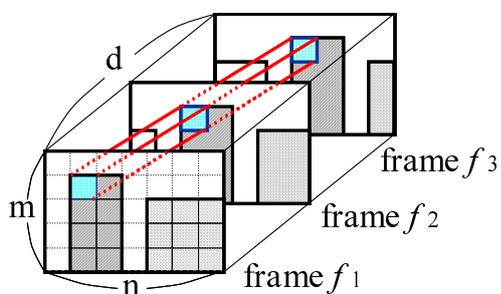


図 7 トラッキング

Fig. 7 Tracking.

#### 4. 視点依存を考慮したレンダリング

##### 4.1 画像の復元

KL 展開によって得られた固有画像を用いて、オリジナルの  $i$  フレーム目の画像は以下に示す線形和により復元することが出来る．ここで、 $V_k$  は、 $k$  番目の固有画像、 $w_{ik}$  は、フレーム  $i$  の重み係数である．また、 $r$  は、ある累積寄与率達成するために必要な主成分の枚数である．本論文では、この累積寄与率は 90% とした．

$$X_i = \sum_{k=1}^r (w_{ik} * V_k) \quad (3)$$

##### 4.2 角度によるサンプリング

3.3 で述べたように、本手法では、画像解像度に応じたサンプリングを行っている．実際には、撮影した画像のフレームではなく、視点方向の角度が均一となるようにサンプリングを行っている．図 8(a) が、フレームから均等にサンプリングした時であり、図 8(b) が、視点方向の角度が均一となるようにサンプリングをした場合である．図から明らかなように、視線が

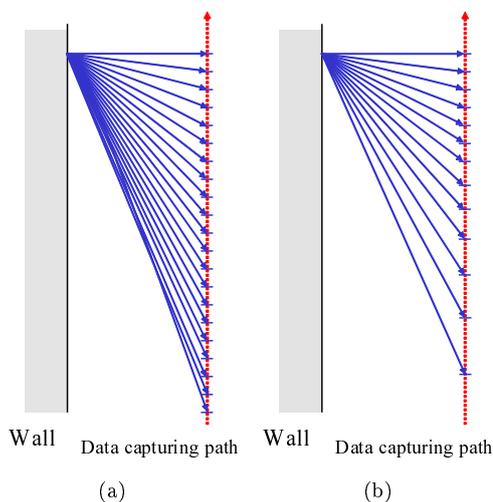


図 8 (a) フレーム均一サンプリング, (b) 角度均一サンプリング  
Fig. 8 (a) Frame uniform and (b) angle uniform sampling.

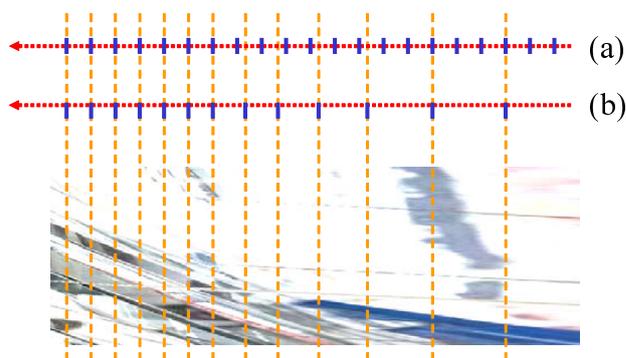


図 9 EPI 上でのサンプリング密度

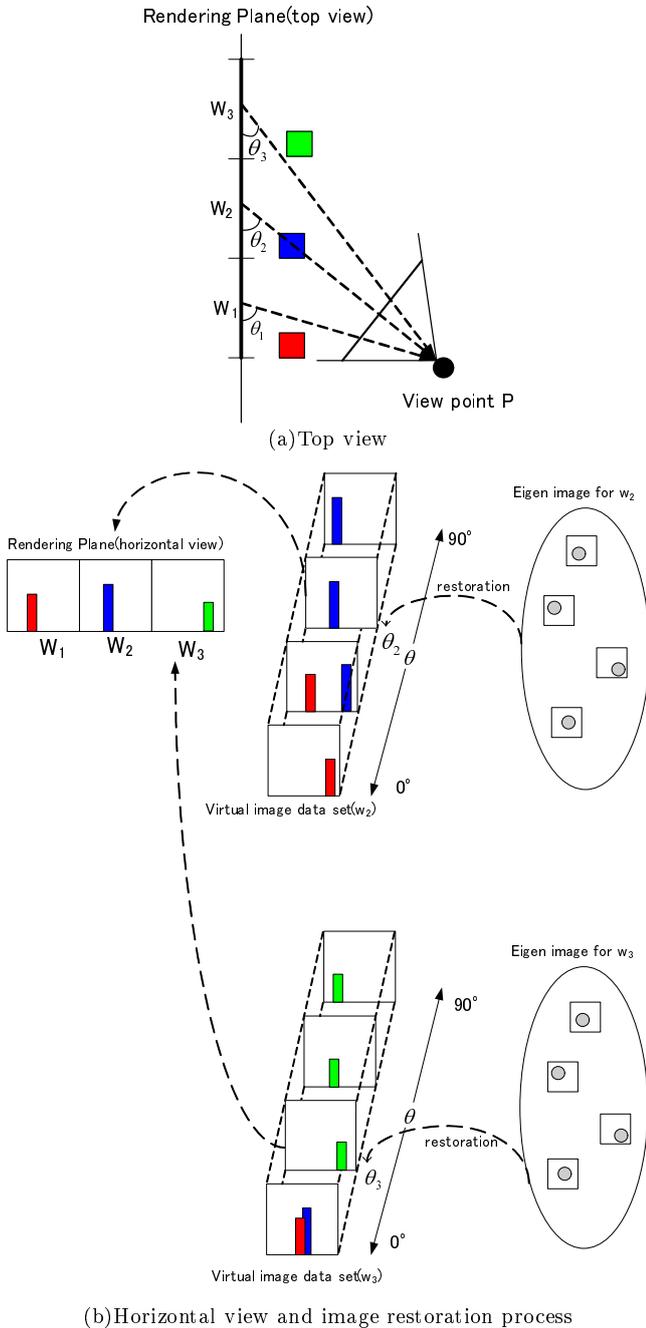
Fig. 9 Sampling on EPI.

進行方向に近づくに従い、フレームから均等にサンプリングした場合には、急激にサンプリング数が増加するのに対し、角度均一サンプリングの場合は一定である．

本論文では、EPI を用いてトラッキングを行い、オブジェクトのテクスチャを得るため、サンプリングも EPI 上で考えることが出来る．図 9 に、前述のそれぞれのサンプリング手法により、EPI 上でサンプリングした例を示す．EPI の横軸はフレーム均一サンプリングと同比率になるため、本論文で提案する視点方向の角度均一サンプリングにおいては、サンプリングする間隔が一定ではなくなっていることが分かる．

##### 4.3 レンダリングアルゴリズム

ここでは、圧縮したデータを用いたレンダリング手法について



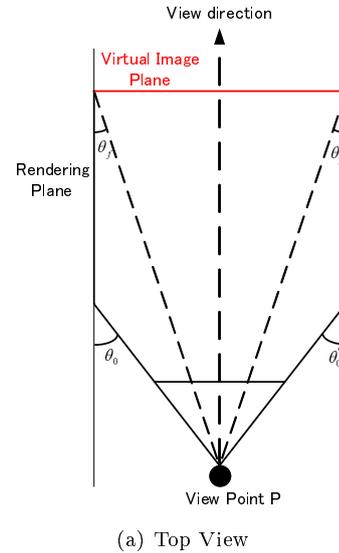
(b) Horizontal view and image restoration process

図 10 レンダリングアルゴリズム

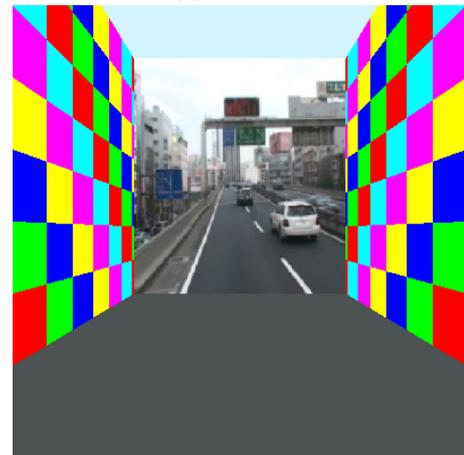
Fig. 10 Rendering algorithm.

て述べる。まず図 10(a) で、ある視点 P から壁面 W を見ることを考える。このとき、視点 P と壁面 W2 中心部分との成す角は  $\theta_2$  となる。そこで、W2 の画像データセットから、角度  $\theta_2$  にあたるテクスチャを獲得して、壁面 W2 に貼れば良い(図 10(b) 上段左側処理)。ところが、本論文では、画像は全て圧縮されているため、W2 の画像データセットは実際には存在しておらず、必要な角度に応じたフレームの画像が、固有画像から復元される(図 10(b) 上段右側処理)。視点の動きに応じて復元が必要なフレームはランダムに変化するが、固有画像からの復元は線形演算であるため、実時間描画が可能である。

また、壁面に張られるテクスチャは視点に応じて様々に変化するため、現実感の高いレンダリングが可能となる。



(a) Top View



(b) Rendering image of forward direction

図 11 進行方向のレンダリングアルゴリズム

Fig. 11 Rendering algorithm of forward direction.

#### 4.4 進行方向のレンダリング

進行方向については視線方向とレンダリングする壁面とが平行に近くなり(図 11(a))、視点と壁面の成す角が非常に小さくなるため 4.3 節のアルゴリズムによるレンダリングが困難となる。そこで、進行方向については以下のアルゴリズムでレンダリングを行う。

図 11(a) において、視点 P と壁面の成す角がある一定の角度  $\theta_f$  までは、4.3 節で述べた方法でレンダリングを行う。それ以降に関しては、3.3 節で述べたようにテクスチャとしての重要度が下がることから、線形演算で復元することは行わず、 $\theta_f$  の地点に仮想壁面を想定し、そこに進行方向にあたる画像をはりつけることで対応する。したがって、実際のレンダリングしたイメージは、図 11(b) のように、左右の壁面は 4.3 節のアルゴリズムで復元されたテクスチャが貼り付けられ、進行方向に関しては一枚の画像が貼り付けられた形となる。

本論文では  $\theta_f$  は 15 度とし、進行方向に貼り付ける画像は視点の位置に最も近い地点で撮影されたものを使用している。

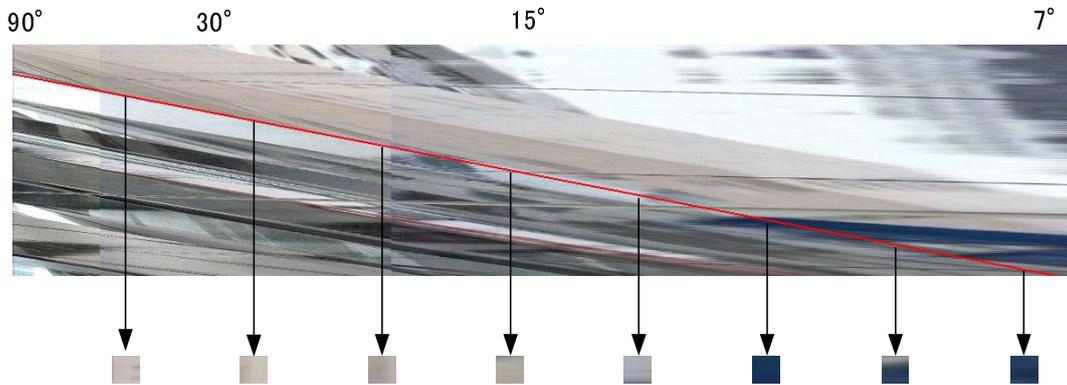


図 12 トラッキングの結果

Fig.12 Tracking result.

## 5. 実 験

### 5.1 トラッキング結果

図 12 は左下のブロックをトラッキングした結果で、赤い直線がそれを示しており、上の角度は EPI 上での角度を示している。

3.3 節で述べたように、進行方向に近づくにつれて正確な EPI を作成することができなくなるため、進行方向近くではトラッキングを行うことができない。したがって、正確な EPI が作られている図 12 の左側はトラッキングがうまくできているが、右下部分はまったく違う色となっている。しかし、角度均一サンプリングと 4.4 節のレンダリング手法により、この影響は限定的となる。

### 5.2 圧縮結果

図 13(b) は、全 238 枚の入力画像のうち、累積寄与率 90% を達成するのに必要な固有画像の枚数を表している。図 13(a) は、入力画像 238 枚から 3 枚選択し、固有画像を作成するためのブロックに区切ったものを示している。この時、例えば図 13(a) の赤の四角形のブロックを復元するために必要な主成分の枚数は図 13(b) の  $(x = S1, y = 1)$  にあたり、必要枚数は 8 枚となる。

また、図 13(b) から見てとれるようにそれぞれのブロックを復元するための固有画像の枚数は異なる。これは元画像のジオメトリの凹凸に起因している。凹凸の大きなブロックは、視点方向が変化するに従い、見え方が大きく変わるため、必要な固有画像の枚数は多くなる。

具体的には、すべての画像を用いる場合、合計のデータサイズは 1.56Mbyte となった。また、累積寄与率 90% を達成するために必要な固有画像の枚数は約 13.9 枚となり、その場合のサイズは 63.8Kbyte となった。このように、全体として 25 分の 1 に圧縮することができた。この数値は常に一定であるものではなく、対象により変化する。高精度のトラッキングや、より正確なジオメトリを用いることでさらに高い圧縮が可能となる。

### 5.3 復元結果

図 14(a), (b) は、それぞれ累積寄与率 90% と 100% で復元した結果である。ほぼ画像の劣化がなく復元できていることが

分かる。また、建物の前にある木も復元することが可能である。

### 5.4 レンダリング結果

図 15(a), (b), (c) は図 15(d) において、それぞれ View point (a), (b), (c) からの見えを生成した図である。

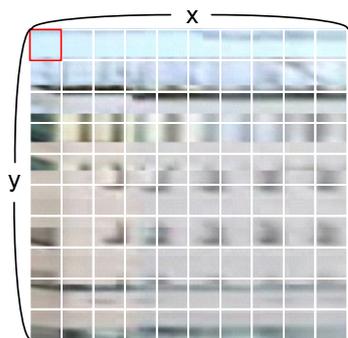
また (a), (b) は進行方向に関しては 1 枚画像を単純に貼り付けているため、そのつなぎ目に違和感がある。ただし、進行方向はもともと物体が圧縮されてテクスチャとなっているため、この違和感は、それほど気になるものとなっていない。しかし、自然な進行方向の画像合成は今後の重要な研究課題である。

## 6. ま と め

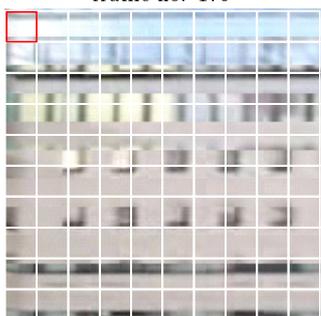
本稿では、全方位画像を KL 展開を用いて、少ない枚数の固有画像に圧縮し、その固有画像を用いてイメージベースレンダリングにより広域空間を再現する手法について述べた。EPI 画像にハフ変換を用いたトラッキングをおこなうことで、長く連続したフレームで安定したトラッキングを実現でき、全方位画像を効率よく圧縮することができた。また実際にプログラムを実装し、正しく広域空間がレンダリング出来ることを確認した。今後の課題として、圧縮効率をあげるためのトラッキング精度の向上や、さらなる広域な都市空間のレンダリング等があげられる。

## 文 献

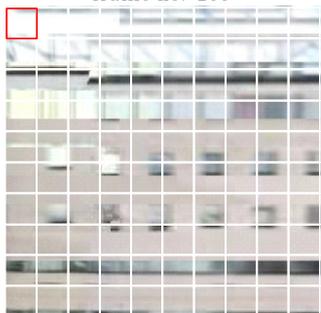
- [1] P. G. R. Inc.:Ladybug: “<http://www.ptgrey.com/>”.
- [2] 小野, 川崎, 池内, 坂内: “EPI 解析による複数ビデオカメラの画像統合”, Computer Vision and Image Media 2003 (2003).
- [3] H. Kawasaki, K. Ikeuchi and M. Sakauchi: “Light field rendering for large-scale scenes”, CVPR, Vol. 2, Kauai, Hawaii, USA, pp. 64-71 (2001).
- [4] C. J. Taylor: “Video plus”, IEEE Workshop on Omnidirectional Vision, pp. 3-11 (2000).
- [5] 山口, 山澤, 竹村, 横矢: “全方位動画画像を用いた任意方向両眼ステレオ画像の実時間生成と提示”, 信学技報, PRMU99-159 (1999).
- [6] 除, 辻: “3次元ビジョン”, 共立出版 (1998).
- [7] 石井, 上田, 前田, 村瀬: “わかりやすいパターン認識”, オーム社出版 (1998).



frame no. 170

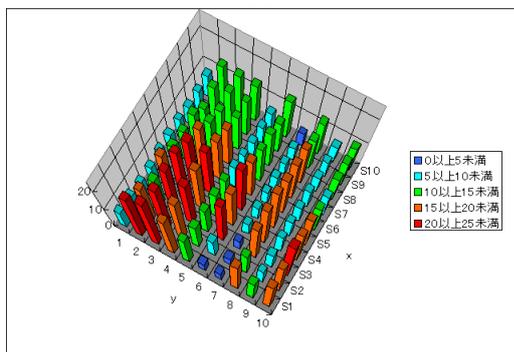


frame no. 200



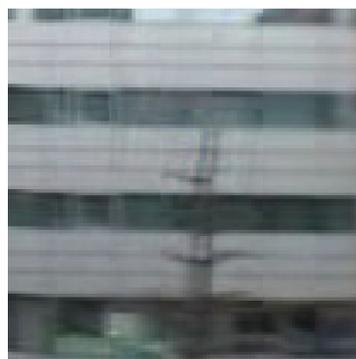
frame no. 230

(a)

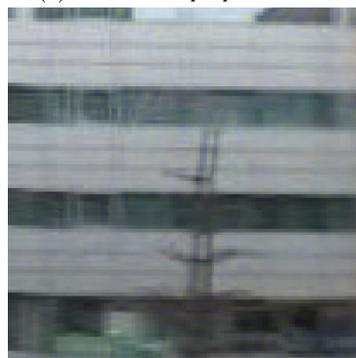


(b)

図 13 寄与率 90% に必要な固有画像の枚数  
Fig. 13 Compression result of KL expansion.



(a)Cumulative propotion 90%



(b)Cumulative propotion 100%

図 14 画像の復元結果

Fig. 14 Restore Image .



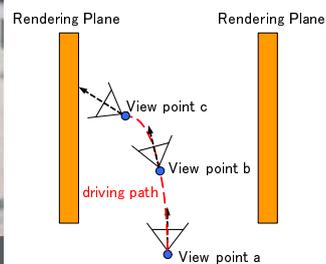
(a)



(b)



(c)



(d)

図 15 レンダリング結果  
Fig. 15 Rendering results.