3次元ビデオ映像のオンライン生成

上田, 恵
九州大学システム情報科学研究院知能システム学部門

有田, 大作
九州大学システム情報科学研究院知能システム学部門

谷口, 倫一郎
九州大学システム情報科学研究院知能システム学部門

出版情報：情報処理学会研究報告. CVIM, [コンピュータビジョンとイメージメディア]. 2004 (40), pp.133-140, 2004-05. 情報処理学会
バージョン：

3 次元ビデオ映像のオンライン生成

上　田　恵† 　有田　大作‡ 　谷口　倫一郎‡

† 九州大学大学院システム情報科学府

‡ 九州大学大学院システム情報科学研究院

複数のカメラ映像から 3 次元情報を復元することで，任意の視点からの映像，つまり 3 次元ビデオ映像を生成する研究が近年盛んに行われている．3 次元ビデオ映像生成は処理に時間がかかり，オンライン処理が難しいため，その研究の多くは，できる限り正確な 3 次元モデルの復元をオフラインで行う，あるいは，陽には 3 次元形状を復元しないがオンラインで 3 次元ビデオ映像を生成するというアプローチをとっている．本稿では，ＰＣクラスタを利用した 3 次元形状復元による自然な 3 次元ビデオ映像生成をオンラインで行う手法について述べる．視体積交差法によって 3 次元形状を復元し，三角パッチで表現された 3 次元表面に仮想視点位置とカメラ画像を考慮した色を付けることで，写実性の高い 3 次元ビデオ映像を生成する．また，生成された 3 次元ビデオ映像を示し，提案手法を評価する．

On-line 3D video generation

Megumu Ueda†, Daisaku Arita‡ and Rin-ichiro Taniguchi‡

†Department of Intelligent Systems, Kyushu University

‡Department of Intelligent Systems, Kyushu University

Recently, there are a lot of researches for generating 3D video by reconstructing 3D models from multiple camera images. Since it is difficult to generate 3D video on-line for the large amount of computation, most of these researches aim to generate 3D video off-line, or generate 3D video without 3D model reconstruction. In this paper, we will propose a method that generates natural 3D video by reconstructing 3D models on-line. The method first reconstructs 3D models by visual cone intersection method using multiple cameras, second colors the surfaces of 3D models in terms of triangular patch representation, and displays the colored models on a screen on-line. And we show generated free-viewpoint images to estimate the method.

1. はじめに

近年では DVD，ディジタルテレビ放送，広帯域インターネットなどといった大容量メディアの普及や，機器の高性能化により容易に大量の情報を扱うことが可能になった．そこで情報メディアのさらなる発展，特に 3 次元情報を持った新たな情報メディアの開発を目的とする研究が盛んに行なわれている．3 次元の情報メディアの応用例として，3 次元ビデオ映像が考えられる[1)2)3)4)5)6)]．3 次元の情報メディアは，例えばスポーツのＴＶ中継や，伝統芸能の記録，保存などの応用が考えられ，将来は娯楽から学術利用まで幅広く利用されることが期待できる．

3 次元ビデオ映像生成を実現するには，対象物体の形状情報と色情報の獲得と，生成した映像の表示が必要である．本稿ではこれらをオンラインで実現し，3 次元ビデオ映像を生成する手法について述べる．本研究は PC クラスタを用いて，視体積交差法によって獲得された 3 次元形状に，仮想視点位置に応じた重み付き色を着色することでオンラインで 3 次元ビデオ映像を生成するというアプローチをとった．3 次元ビデオ映像に関する多くの研究は，できる限り正確な形状復元を行うが処理に時間がかかり過ぎるためオフラインで行っていたり，正確な形状復元を行わないが処理の高速化によりオンライン処理が可能というアプローチをとっている．しかし，本研究ではできる限り正確な形状を復元し，オンラインで自然な 3 次元ビデオ映像生成を実現した．さらに本研究では実験を行い，あら

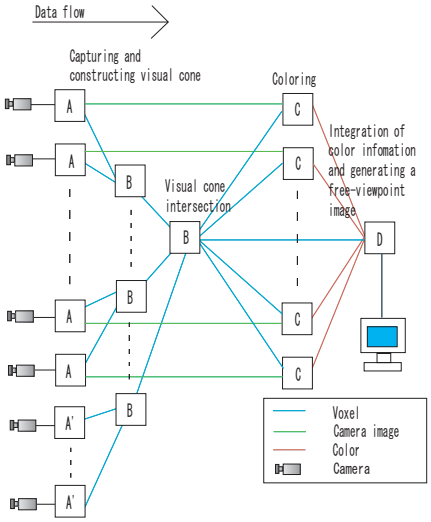


図 1 本研究のシステム構成

かじめ決められた空間内に存在する対象物体を，固定されたカメラで撮影し，オンラインでの 3 次元ビデオ映像生成が可能なことを確認した．

2. オンライン 3 次元ビデオ映像生成

2.1 システム構成

実時間での 3 次元ビデオ映像生成には多くの処理時間を必要とする．そこで分散並列計算機の一種である PC クラスタ，および PC クラスタをプラットフォームとする実時間並列画像処理システムのためのプログラミングツール RPV (Real-time Parallel Vision)[7] を用いて，オンラインで並列画像処理を行なう．

図 1 に本研究のシステム構成を示す．以下に各 PC での処理を述べる．

ノード A: カメラ画像を取得し，その中から対象物体を抽出し，対象物体を抽出した画像から視体積を構築する．視体積をノード **B** に，対象物体を抽出した画像をノード **C** に送る．対象物体の抽出は，カメラ画像に対し背景差分を施した後に，クロージング処理を施すことで行う．ノード **A'** は形状復元だけに使用しており，ノード **B** に視体積を送り，カメラ画像は送信しない．色付けに多くのカメラを使用しても，精度向上にあまり寄与しないと考え，処理時間の短縮のため色付けには使用しない．

ノード B: ノード **A** およびノード **A'** から送られてきた各視体積の共通領域を求める．得られた対象形状のボクセル表現に対し，離散マーチング・キューブ法[8)9)] を施し三角パッチ表現へ変換する．その結果をノード **C** と **D** に送るが，三角パッチ表現を送信するとデータ量が膨大になってしまい，送受信に時間がかかり過ぎてしまうので，三角パッチに変換されるボクセルの座標と，対応する三角パッチの生成パタンを送信し，ノード **C** と **D** のそれぞれで三角パッチ表現を再構築させる．1 台の PC で全てのカメラ画像に対して視体積交差を行なうと，視体積の送受信に時間がかかりすぎるため多段に分けて視体積交差を行なう．

ノード C: ノード **B** から送られてきた 3 次元情報から三角パッチ表現を再構築する．得られた三角パッチに，ノード **A** から送られてきた画像を基に Z バッファ法を用いて色を付ける．この詳細については 2.2 節で述べる．得られた色情報をノード **D** に送る．

ノード D: ノード **B** から送られてきた 3 次元情報から三角パッチ表現を再構築する．ノード **C** からの色情報とユーザーから入力された仮想視点位置から重み付き色付き対象形状を生成，すなわち 3 次元ビデオ映像を生成する．

ノード **A** からノード **B** へのボクセルの流れは RPV が提供するストリーミング処理を利用して遅延の削減を図っている．ノード **B** 以降の流れはノード **C** における Z バッファ法のため，全ての 3 次元情報が揃わないと処理ができないので，RPV が提供するフレーム同期処理を利用している．

2.2 色 付 け

本節ではノード **C**，ノード **D** における，三角パッチデータへの色付けについて説明する．

2.2.1 色付け手法

対象物体形状への色付けの従来手法は，

(1) カメラ位置と対象物体表面方向の関係を利用する手法

(2) カメラ位置と仮想視点位置の関係を利用する手法

の二つに分類できる．本研究では，できるだけ自然な 3 次元ビデオ映像を実時間で生成することを目指しており，文献[1)2)] で後者の手法の有効性が示されていることから後者の手法を採用した．

色は三角パッチの頂点に各カメラ視点の重みを基に付ける．三角パッチの各頂点の色をスムージングにより補間した色を三角パッチの面に塗ることで自然な色

付けを実現する．また，どのカメラからも見えないと判断された頂点については，隣接する三角パッチの頂点全ての平均の色を付ける．このようにすることで，どのカメラからも見えない頂点にもある程度自然な色を付けることが可能となる．

2.2.2 色情報の取得手法

カメラ画像を基に三角パッチ頂点の色情報の取得手法について述べる．

色情報は Z バッファ法に基づいて取得する．しかし，単純に Z バッファ法で色情報を取得すると，まばらな色付けになってしまう．なぜなら，一つの画素に複数の頂点が投影される可能性があり，その場合，それらの頂点のうちの一つの頂点にしか色が付かないためである (図 2(a))．そこで本研究では Z バッファ法を発展させた以下の処理をすることによって頂点に色を付けることを提案する．

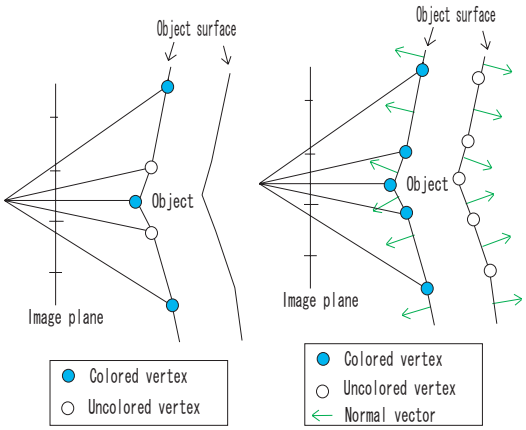
- **処理 1:** カメラの視線方向のベクトルと面の法線方向のベクトルとの内積が正となる面 (つまり，カメラの方向を向いていない面) はカメラから見えないと判断し，頂点には色を付けずに，カメラからの距離値を Z バッファに保存する．
- **処理 2:** カメラの視線方向のベクトルと面の法線方向のベクトルとの内積が負となる面 (つまり，カメラの方向を向いている面) はカメラから見える面と判断し，Z バッファ法で色を付ける．ただし，Z バッファの距離値を置き換えることはしない．

こうすることで隣接する頂点の距離値が Z バッファに保存されることがないので，まばらな色付けにならず，カメラ視点から可視な頂点のみに色を付けることができる (図 2(b))．

また，通常の三角パッチ (図 3(a)) を六個の三角パッチ (図 3(b)) に分割して色を付けることで色付けの精度向上を図った[1)]．ボクセルの量子化間隔を小さくすることでも，形状および色付けの精度を向上させることができるが処理時間が劇的に増加してしまう．そこで，三角パッチを分割することで 3 次元形状の精度は変わらないが，処理時間を劇的に増加させることなく，より細かい模様の表現が可能となる．

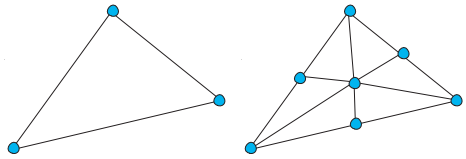
2.2.3 色情報の重み

すべての頂点について，可視と判定されたカメラ画像から得られるその頂点の色の重み付き平均をとり，その頂点の色とする．ある頂点が N 個のカメラから可視である場合，その中のカメラ $n(1 \leq n \leq N)$ から得られる色の重み W_n は，仮想視点から頂点へのベクトルと，カメラ n から頂点へのベクトルとがなす角度を θ_n として (図 4)，



(a) 単純な Z バッファ法による色情報獲得手法　(b) 本研究の提案する色情報獲得手法

図 2 色情報獲得手法



(a) 通常の三角パッチ　(b) 高精度な色付けのため分割された三角パッチ

図 3 三角パッチの分割

$$W_n = \frac{(cos\theta + 1)^\alpha}{\sum_{n=0}^{N}(cos\theta_n + 1)^\alpha}$$

として求めている．括弧内を $cos\theta$ のみにすると色の重みが負の値をとるので $cos\theta + 1$ としている．また，経験的に $\alpha = 5$ としている．このようにすることで仮想視点と方向が近いカメラからの色が優先され，仮想視点と方向が遠いカメラほど色の優先度が下がることになる．

3. 実験と考察

3.1 実　　験

本手法を用いてオンラインで 3 次元ビデオ映像を生成し，その処理時間，遅延時間，通信量，色付け誤差を計測した．

本実験では合計 17 台の PC を利用した．各 PC はスイッチ型ギガビット LAN の一つである Myrinet に

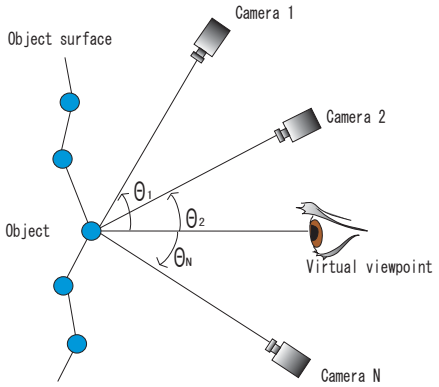


図 4　色情報の重みの付け方

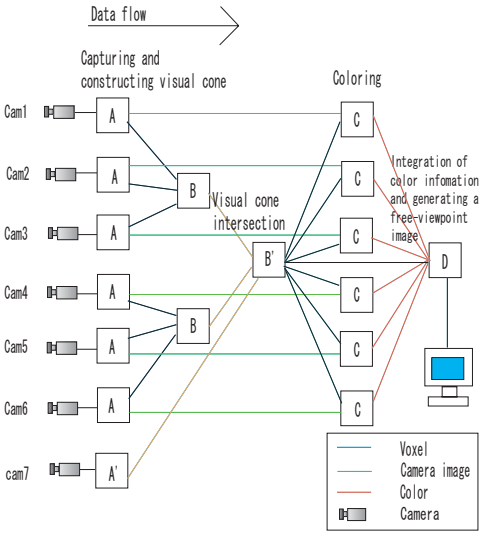


図 5　本実験で実装したシステム構成

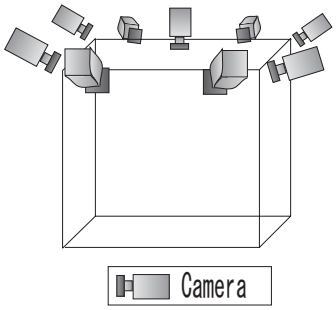
よって相互に結合されており，1Gb/sec の通信が可能である．さらに 7 台の IEEE1394 ディジタルカメラ[10] が接続されており，全てのカメラは同期信号発生装置により同期がとられている．図 5 に本実験で実装したシステム構成を示し，図 6 にカメラ配置を示す．

ノード **A'** には天井カメラの cam7 を用いた．全てのカメラが上方から見下ろしているので，天井カメラは色付けにあまり寄与しないと考えたからである．また，カメラは予めキャリブレーションしておいたものを使用した．カメラキャリブレーション手法としては，レンズ歪みを考慮した Tsai の手法[11] を利用した．カメラ画像の解像度は 320×240 で，空間解像度は $128 \times 128 \times 128$，ボクセルの一辺を $2cm$ として実験を行った．実験で使用した PC の概要を表 1 に示す．

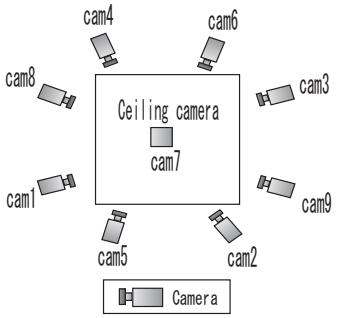
実験には縞模様のある服装を使用し，縞模様の間隔

表 1　PC の性能

OS	Red Hat Linux9
CPU	Intel Pentium4 3GHz
メモリ	1GB
コンパイラ	gcc-3.2.2



(a) 斜め上から見たカメラ配置



(b) 上から見たカメラ配置

図 6　本実験のカメラ配置

はボクセル表現の量子化間隔よりも広い．オンラインで処理を行うことは可能であるが，評価実験においては先に述べた測定を同じカメラ画像に対して行うために，保存しておいたカメラ画像を入力としてオフラインで実験を行った．このとき保存しておいた画像に対するオフライン処理は全く施していない．したがって，計算処理および画像データは実際にオンラインで処理したものと同等と考えてよい．また，3 次元ビデオ映像の表示には OpenGL を用いた．OpenGL は，ハードウェアや OS には依存しない 3 次元グラフィックスのためのプログラミングインタフェースである．

3.2　3 次元ビデオ映像についての考察

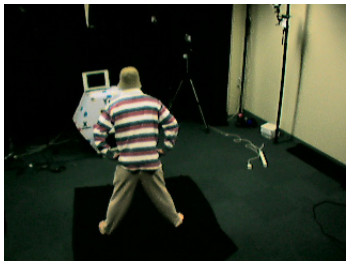
図 7 に原画像と生成された 3 次元ビデオ映像を，図 8 に実際にはカメラのない視点からの 3 次元ビデオ映

(a) 原画像 1

(b) 生成画像 1

(c) 原画像 2

(d) 原画像 3



(e) 原画像 4

(f) 生成画像 2

(g) 生成画像 3

(h) 生成画像 4

図 7 原画像と生成画像

像を示す．図 8 の周りの小さな立方体はカメラを表している．

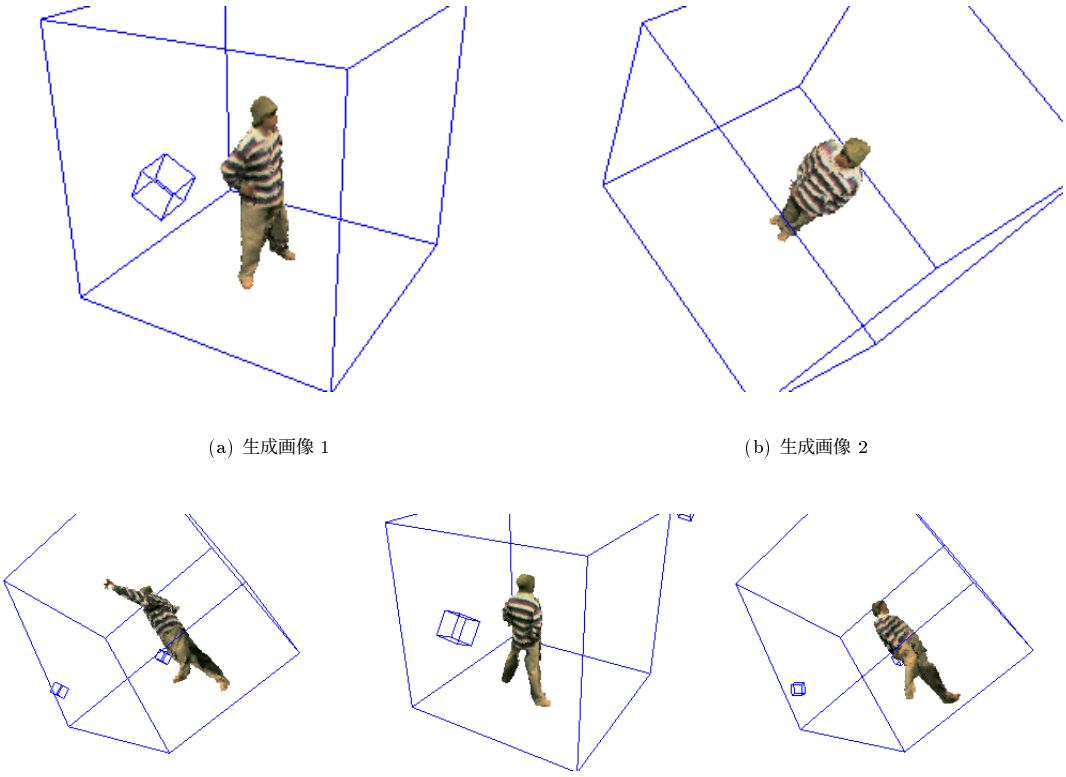
図 7 を見ると，服の模様が再現されていることがわかる．また，体の一部が欠けている生成画像があるが，これは周りに暗幕があり，床にも黒い布を敷いているため，服の影の部分が背景と誤認識され背景差分に失敗し，対象領域を抽出できなかったためである．

実際の動画像を主観的に評価すると

- ほぼ忠実に動きが再現されている
- 形状が多少不正確である
- 320 × 240 の解像度のカメラ画像とほぼ遜色ないように見える

という評価ができた．

以上より，3 次元ビデオ映像の生成はほぼ実現されているが，より自然な 3 次元ビデオ映像を生成するためには形状復元の高精度化，背景差分の強化が必要で



(a) 生成画像 1　(b) 生成画像 2

(c) 生成画像 3　(d) 生成画像 4　(e) 生成画像 5

図 8　仮想視点からの画像

あるということがわかった．

3.3　色付け誤差についての考察

測定した色付け誤差を，図 9(a) に示す．

色付け誤差の測定は，得られた色付きの三角パッチ頂点をカメラ画像に投影し，RGB 値の 2 乗平均平方根誤差を求めた．その際，投影された三角パッチ頂点間には線形補間した RGB 値を用い，その誤差を求めた．

色付け誤差の原因は，以下の四つが考えられる．

(1)　視体積交差法により復元した形状の誤差のため
(2)　キャリブレーションの誤差のため
(3)　細かい模様をスムージングによって表現しようとするため
(4)　重み付き色情報の統合による誤差のため

また，色付けに使用したカメラとの誤差と比べて色付けに使用していないカメラとの誤差が 2 倍程度になっていることがわかる．このことから，提案手法の効果によりカメラ視点と仮想視点の方向が近いときには生成画像の精度が高くなることが確かめられた．

3.4　処理時間についての考察

処理時間の測定には，送信完了待ち，受信待ちの時間を除いた，各ノード特有の処理にかかった時間を計測した．

図 9(b)(c) に処理時間，図 9(d) にスループットを示す．

平均では 20fps 程度の速度での処理が可能であり，実用十分な速度が得られている．しかし図 9(b)(d) から，処理速度は安定しないことがあることがわかる．これは対象物体の大きさや形によって三角パッチの数が変わるからである．

3.5　遅延時間についての考察

遅延時間は，全 PC の内部時計は一致しているという前提で，カメラ画像が入力された時刻と 3 次元ビデオ映像を生成した時刻との差を計算することによって求めた．

図 9(e) に遅延時間を示す．

パイプラインの各段における処理時間の合計よりも短く，ストリーミング処理の効果がでている．しかし，

主観的にはまだ遅延を感じるので，本システムを人同士のインタラクションなどに使う場合には，さらなる改善が必要である．ノード **B'** 以降ではストリーミング処理を行なっていないので，ストリーミング処理化ができれば遅延時間の削減が見込めると思われる．

3.6 通信量についての考察

図 9(f) に各ノードからのデータの送信量を示す．図 9(f) にはノード **A** からのボクセル情報送信量を載せていないが，ボクセル情報の量は常に一定であり，1 ビット 1 ボクセルで処理をしているので $128 \times 128 \times 128$ ビットである．現時点ではそれを 8 回にわけてストリーミング処理をしているので一回の送信量は 32.768 キロバイトとなる．また，ノード **A** からのカメラ画像は，前景領域のみを送信しているので送信量は可変となっている．

ノード **D** に送信されるノード **B'**，**C** からのデータ量の合計，つまり 3 次元ビデオ映像生成に必要なデータ量は，図 9(f) から 1 フレーム平均で 600 キロバイト程度必要であることがわかる．3.1 節で述べた Myrinet の性能から通信時間を計算すると，各ノード間の通信時間は数ミリ秒でありスループットへの影響はほとんどなく，また，遅延時間に及ぼす影響は十数ミリ秒程度であると予測される．したがって，通信時間がシステムに及ぼす影響はほとんどないと思われる．しかし，ノード **C** からの色情報の送信量がノード **B** からの形状情報に比べて格段に多く，3 次元ビデオ映像のライブ配信などといったシステムの応用を考えるとデータを圧縮する必要があると考えられる．

4. おわりに

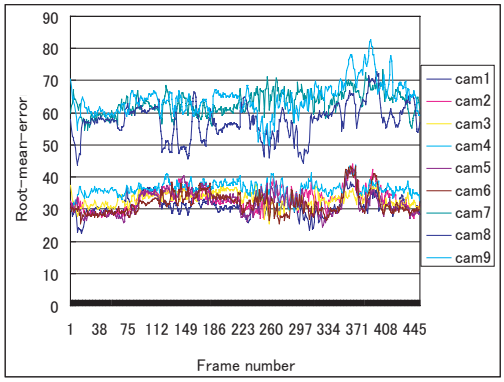
本論文では，PC クラスタを利用した多視点画像からのオンライン 3 次元ビデオ映像生成の手法を提案した．実験によりオンラインでの 3 次元ビデオ映像生成が可能なことが確かめられた．したがって，今後の課題としては，

- **ノード B 以降の処理のストリーミング処理化:**
  全てのノードでストリーミング処理をすることにより，遅延時間を削減する
- **処理速度が対象物体の大きさに影響を受けにくいアルゴリズムの開発:**
  対象の大きさや形状に影響を受けにくいアルゴリズムを開発することにより，安定したスループットで高速に 3 次元ビデオ映像を生成する
- **形状復元の高精度化:**
  形状復元を高精度化することにより，より自然な映像を生成する

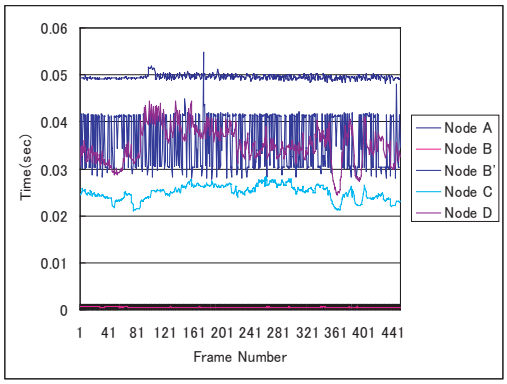
などが挙げられ，高速でかつより自然な 3 次元ビデオ映像の生成を目指す．

参 考 文 献

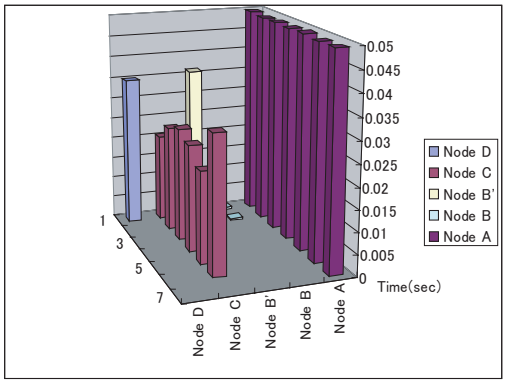
1) 高井 勇志, 松山 隆司: “ 3 次元ビデオ映像の高精細表示アルゴリズムと編集システム”, 映像情報メディア学会誌, Vol. 56, No. 4, pp. 593–602, 2002.
2) 松山 隆司, 高井 勇志, ウ 小軍, 延原 章平: “ 3 次元ビデオ映像の撮影・編集・表示”, 日本バーチャルリアリティ学会論文誌, Vol. 7, No. 4, pp. 521–532, 2002.
3) 延原 章平, 和田 俊和, 松山 隆司: “ 弾性メッシュモデルを用いた多視点画像からの高精度 3 次元形状復元”, CVIM 研究会論文誌, Vol. 43, No. SIG 11(CVIM 5), pp. 53–63, 2002.
4) 斉藤 英雄, 木村 誠, 矢口 悟志, 稲木 奈穂 : “ 射影幾何に基づく多視点カメラの中間視点映像生成”, 情報処理学会論文誌, Vol. 43, No. SIG 11(CVIM 5), pp. 21–32, 2002.
5) 北原 格, 大田 友一 : “ 大規模空間に適した 3 次元形状表現手法による自由視点映像の実時間生成”, 信学技法, PRMU2003, pp. 61–66, 2003.
6) 古山 孝好, 北原 格, 大田 友一 : “ スタジアムの自由視点ライブ中継が可能な 3 次元映像システム”, 3 次元画像コンファレンス 2003, pp. 225–228.
7) 有田 大作, 花田 武彦, 谷口 倫一郎 : “分散並列計算機による実時間ビジョン”, 情報処理学会論文誌, Vo. l43, No. SIG 11(CVIM 5), pp. 1–10, 2002.
8) William E. Lorensen, Harvey E. Cline: “Marching Cubes: A High Resolution 3D Surface Construction Algorithm” , Computer Graphics, Vol. 21, No. 4, pp. 163–169, 1987.
9) 剣持 雪子, 小谷 一孔, 井宮 淳: “ 点の連結性を考慮したマーチング・キューブ法”, 信学技報, PRMU98-218, pp. 197–204, 1999.
10) 吉本廣雅, 有田大作, 谷口倫一郎: “1394 カメラを利用した多視点動画像獲得環境”, 第 6 回 画像センシングシンポジウム講演論文集, pp. 285-290, 2000.
11) Roger Y. Tsai: “A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses”, IEEE Trans, on Robotics and Automation, Vol. 3, No. 4, pp. 323–344, 1987.



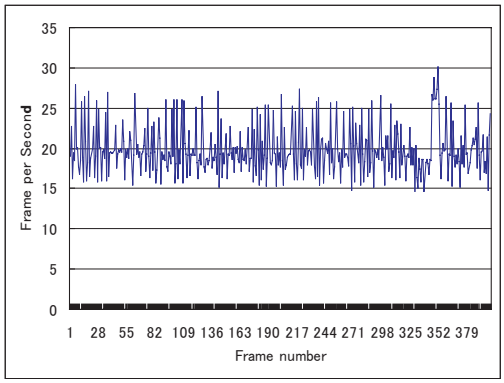
(a) カメラ画像との色の誤差



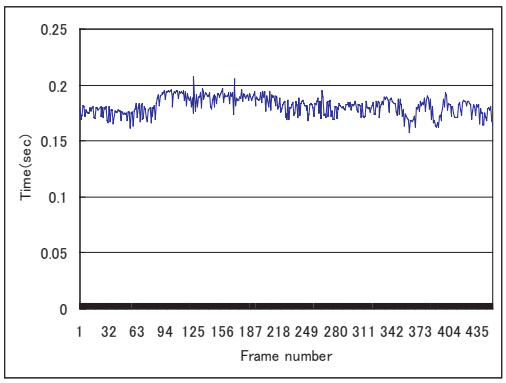
(b) 各パイプライン毎の平均処理時間



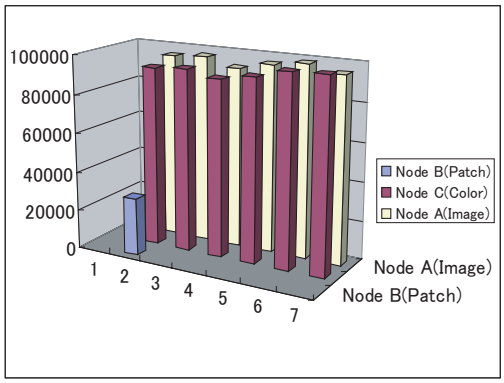
(c) 各ノードの平均処理時間



(d) スループット



(e) 遅延時間



(f) 各ノードの平均送信量

図 9 計測結果