コンピュータビジョンと
イメージメディア 121－9
（2000．3．23）

ＰＣクラスタを用いた身体動作の実時間３次元映像化

ウ 小軍　東海 彰吾　和田 俊和　松山 隆司
京都大学 大学院 情報学研究科

異なる視点に配置した複数のカメラの映像に基づいて、対象となる人間の身体動作を３次元形状として連続的に獲得し映像化するためには、多視点での同時撮影、高速な３次元形状算出、および、身体動作に伴う入力の動的変動への対応が必要である。本研究では、複数のＰＣが高速なネットワークで相互に接続されたＰＣクラスタを用い、各ＰＣに接続したカメラでの多視点映像の撮影、ＰＣ群での並列処理による映像群からの高速な３次元形状算出、さらに、撮影失敗を含む動的な状況変動に対してＰＣ間でのデータ授受や並列計算の適応的な制御の方法について、作成したＰＣクラスタシステム（９カメラ、１０ＰＣ）による実験結果と併せて述べる。

Realtime 3D visualization of human gestures using PC cluster system

Wu Xiaojun, Shogo Tokai, Toshikazu Wada, Takashi Matsuyama
Graduate School of Informatics, Kyoto University

In this paper, we propose a method for capture and visualize a human gesture as 3D image sequence in real time. To realize these functions, we use a PC cluster system. PC cluster consists of multiple PCs which are connected by high speed network. Each PC captures video image of its own video camera simultaneously. We develop a method to obtain 3D shape of the gesture by a parallel volume intersection algorithm on a PC cluster. We also consider dynamical load balancing to adapt variation of the dynamical scene situation. We show experimental results with our PC cluster system that consists of 9 pan-tilt-zoom cameras and 10 PCs.

1　はじめに

シーン内の状況を取得・認識することは、視覚センサを用いたシステムの大きな目的の一つである。特に、複数の視覚センサを用いてシーン内の人間の３次元的な身体動作の取得・認識を行うことは、単なる動作の解析だけでなく、ジェスチャによるマンマシンインタフェースやＶＲ、さらには、能・狂言や演劇など身体動作を伴う芸術を３次元情報としてアーカイブするなど、幅広い分野への適用が考えられる。

これまで、人間の３次元的動作を獲得する方法として、主に関節や、人体の各部位（頭、胴体、腕、足など）の３次元位置を実時間で取得する様々なモーションキャプチャシステムが製作されている。さらに、身体動作を形状として獲得する方法として、シーンに配置した多数のカメラの映像を統合する方法が提案されてきた [1, 2]。しかし、例えば、カーネギーメロン大の [1] では、５０台あまりのカメラで多視点撮影を行い高精度な形状の算出を行えるものの、基本的には撮影後のバッチ処理によるものであり、実時間での形状取得や映像化については実現されていない。また、メリーランド大の [2] では、非常に高速に形状を取得できるものの、計測領域はたかだか数メートル四方程度の非常に狭い範囲に限定されてしまうという問題点がある。

本研究では、より一般的なシーンでの人物の身体動作を３次元的な形状として連続的に取得し映像化することを考える。この時、計測可能な人物の行動範囲を広く保ち、かつ、広範囲に動き回る人物を的確に捉え、その身体動作を連続的に取得し映像化するためことが求められる。このためには、視線やズームを変化できる首振りカメラを、シーンの複数の視点に設置し、それぞれのカメラをシーンの状況に適応的に制御と、複数カメラの協調動作が必要となる。

このような、身体動作のアクティブな実時間３次元映像化を目指し、本文では複数のＰＣがネットワーク結合されたＰＣクラスタを用い、３次元形状復元の一手法である視体積交差法に基づいて、シーン内の人物の３次元形状を実時間で連続的に取得する方法について述べる。

本文は以下の構成で議論する。まず、**2** で、３次元形状復元の方法として用いる視体積交差法について説明する。次に、**3** で平面から平面への透視投影計算が線形演算で高速化できる注目した、改良視体積交差法について述べ、さらに、**4** では、ＰＣクラスタシステムへの実装のための並列化アルゴリズムについて述べる。**5** で実際に作成したシステムで行った実験について説明し、**6** で本文全体をまとめる。

2 視体積交差法

視体積交差法は、シーン内に複数設置したカメラで観測された物体のシルエットに基づいて物体の形状を求める手法である [3, 4, 5, 6]。この方法では、まず、シーン内の複数の位置に設置したカメラによって、対象の様々な方向からの画像を撮影する。次に、得られた画像の中から対象を表すシルエットを抽出して、あらかじめ較正で得られているカメラの内部パラメータ (焦点距離、画像中心など) や外部パラメータ (視点位置や視線方向) に基づいて、シルエットを空間に逆に投影する。複数カメラから逆投影されたシルエット像の交わり部分を計算することによって、最終的な物体形状を得るものである (図 1)。

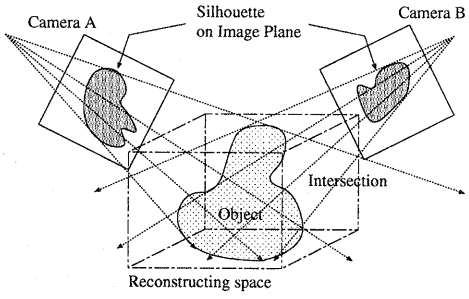

図 1: 視体積交差法

この方法は、ステレオ視 [7] など３次元形状獲得の他の手法と比較すると、

欠点１：カメラの台数や配置と物体との位置関係によっては、虚像が生じる場合がある。

欠点２：凹み部分を扱えない場合がある。

などの欠点はあるものの、

利点１：シルエット抽出が比較的単純な画像処理によって可能なため、照明などに影響されにくい。

利点２：複数カメラを分散して配置することによって、精度の高い形状を獲得することができ、広域シーンで適用しやすい。

などの利点がある。また、欠点１については、設置するカメラの台数、つまり、観測視点を増やすことにより虚像の発生を抑制して、精度の高い形状の獲得が期待できる。

さらに、２台以上のカメラグループを考える必要があるステレオ視に比べて、視体積交差法では、精度の差は起こるものの、カメラ台数によって計算手法が大幅な影響を受けないため、すべてのカメラが対等に扱われるという点で、前述のアクティブな形状取得に適していると考えられる。

2.1 基本的アルゴリズム

まず、視体積交差法の計算アルゴリズムとして、文献 [8] で紹介されているものについて考察する。

この方法では、まず、形状を復元する３次元空間を立方体格子 (ボクセル) に分け、各ボクセル毎に実対象が存在する (occupied) か、存在しない (empty) かを判定し、最終的に occupied なボクセルの集合を実際の３次元形状と考える。シルエットは背景差分などの画像処理によって得られた２値画像である。画像の取得、および、シルエット領域の算出の後の具体的な処理の流れは図２のようになる。

```
1  全てのボクセルを occupied に設定
2  for 全てのボクセル v do
3      for 全てのカメラ c do
4          ボクセル v をカメラ c の画像平面に投影
5          if 投影点が物体シルエットに含まれない
6              then
7                  ボクセル c を empty に設定
8                  goto ENDINNERLOOP
9          end_if
10     end_for
11     ENDINNERLOOP:
12 end_for
```

図 2: 視体積交差法の基本アルゴリズム

この場合の計算量は、３次元形状再構築のボクセルが、$n_x \times n_y \times n_z$ 個、カメラが m 台ある場合、図２のライン４の投影計算を最大 $n_x n_y n_z m$ 回行う必要があり、投影計算１回に必要な計算時間を T_α とすると、全体の計算時間は、最大で以下の計算時間が必要

となる。

$$n_x n_y n_z m T_\alpha \tag{1}$$

ここで、より高い解像度での3次元形状再構成を考えると、n_x, n_y, n_z の増加に伴って繰り返しの回数が増加するため、計算時間の増加も顕著である。しかし、手法の性質上、基本的な繰り返し回数を減らすことは困難であるため、繰り返し内部の計算時間の短縮による、全体の処理時間の削減が必要である。

3 平面を利用した改良視体積交差法

前述の基本アルゴリズムでは、ボクセルを単位とした処理が繰り返される。しかし、平行平面群の各平面を処理単位と考え、この平面へ各カメラのシルエットを投影し、シルエット群(2値画像)のAND演算を行うことによっても、視体積交差法の計算が可能である。この時、以下に述べるように、平面から平面への投影計算が線形演算を利用して高速化できるため、全体の処理時間を削減できると考えられる。平行平面群を利用することによって、平面間の投影計算が高速に行えることは、Collinsによって示されている[9]。本研究も基本的にこの考え方に基づく。

以下では、まず、平面から平面への透視投影計算を高速に行う方法について述べ、それを利用した視体積交差法の改良アルゴリズムについて説明する。

3.1 2平面間の透視投影の高速化

今、ある視点 o を投影中心として、パラメータが既知の平面 P_{src} から別の平面 P_{dst} への透視投影を想定し、P_{dst} の矩形領域 $n_x \times n_y$ への投影計算を行う場合を考える。各点毎に行う透視投影の一回あたりの計算時間を T_α とすると、P_{dst} 全体への投影のための計算時間 T_1 は、

$$T_1 = n_x n_y T_\alpha \tag{2}$$

となる。しかし、2平面の交線(共役線)に平行な直線上での透視投影を考えると、この2直線と視点は一つの平面上に存在し、2直線上では相似関係が成り立つ(図3)。つまり、2平面の共役線に平行な直線毎に透視投影計算を行う場合、投影すべき線分の2端点の透視投影と、P_{src}、P_{dst} 上での対応するステップを定めれば、単純な等差数列計算によって、残りの透視投影を実現できる。

この時、走査する直線数と、ある直線上で投影すべき点数は、両者ともおよそ $\sqrt{n_x n_y}$ であると考えると、2端点の投影計算には、$2\sqrt{n_x n_y} T_\alpha$ が、また、等差数列計算による1点当たりの投影計算時間を T_β とした場合、1直線当たりの投影には、$\sqrt{n_x n_y} T_\beta$ だけ必要で、P_{dst} 全体の計算時間 T_2 は、以下のようになる。

$$T_2 = n_x n_y T_\beta + 2\sqrt{n_x n_y} T_\alpha \tag{3}$$

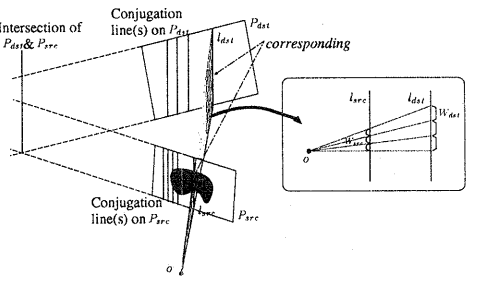

図3: 平面から平面への投影(2平面が交わる場合)

2平面が平行な場合は、両平面同士が相似の関係を持つので、透視投影計算は、画像の拡大縮小と平行移動によって実現できる(図4)。この時は、P_{dst} 上の一点についてだけ投影計算を行えば拡大率や平行移動量を決定でき、残りの部分の投影計算は線形演算に置き換えられる。この時、P_{dst} 全体の投影に必要な時間 T_3 は、以下ようになる。

$$T_3 = n_x n_y T_\beta + T_\alpha \tag{4}$$

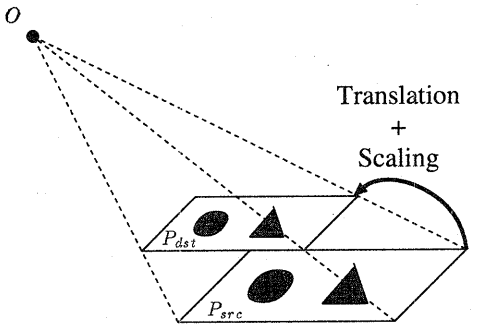

図4: 平面から平面への投影(2平面が平行な場合)

ここで、行列演算の時間である T_α と ベクトルの和演算の時間である T_β を比べれば、$T_\alpha > T_\beta$ と考えて良く、$T_1 > T_2 > T_3$ の関係が成り立ち、(n_x, n_y) がある程度以上大きい場合は、その差は顕著になる。

3.2 平面投影を利用した改良アルゴリズム

このように、平面から平面への投影計算を用いれば、点単位で投影計算するより高速に処理できる。こ

のことを利用して、視体積交差法を高速化する。基本的な考え方は、従来のボクセル空間を平行な平面群(スライス)として捉え、各スライスを単位として、スライス面へのシルエットの投影と2値画像のANDの演算の処理を行い、最終的な3次元形状を得るものである(図5)。

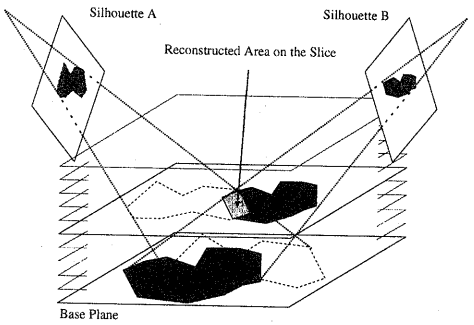

図 5: 平面投影を利用した視体積交差法

今、xyz 空間を、n_x, n_y, n_z の解像度で再構成を行う場合を考える。なお、スライスは、xy 平面に平行であるとする。まず、$z = 0$ へのシルエット像の投影を行う。この投影処理には、平行でない2平面間の投影計算法を用いる。これを基準平面シルエットとする。次に、$z \neq 0$ のスライスに対しては、この基準平面からの投影によってスライス上のシルエットを生成する。基準平面と各スライス面は平行であるため、平行平面間の投影計算法が利用でき、カメラ画像平面から直接スライス上へ投影するより計算時間は少なくなる。

各スライス上には、複数のカメラに対応したシルエットが投影されるので、それらのシルエット画像(2値画像)間のAND演算を行うことによって、スライス上の対象存在領域が決まる。この処理を n_z 個のスライス全てで行うことによって、最終的な3次元形状が得られる。処理の流れを図6にまとめる。

この方法での計算時間について考察する。図6のライン 1〜3 の処理では、画像平面から基準平面への投影が行われる。この時、n_x, n_y に対応した部分だけでなく、後の処理に必要な領域もあわせて投影する必要がある。このため、カメラと再構成すべき領域との位置関係にもよるが、投影すべき点数は、n_x, n_y それぞれ数倍程度必要となる。この時の倍率を q、カメラの台数を m とすると、ライン 1〜3 の部分の計算時間は、

$$(q^2 n_x n_y T_\beta + 2q\sqrt{n_x n_y} T_\alpha)m \qquad (5)$$

となる。次に、あるスライスを処理する場合は、基準平面からスライス面への平行平面間の投影計算で実

```
1  for 全てのシルエット do
2      シルエットを基準平面 (z = 0) に投影
3  end
4  for 全てのスライス do
5      for 全ての基準面シルエット do
6          基準面からスライスへシルエットを投影
7      end_for
8      for スライス上の全ての点 do
9          シルエットの AND 計算
10     end_for
11 end_for
```

図 6: 平面投影を利用したアルゴリズム

行できるので、これを全てのスライスで計算すると、

$$n_z \times (n_x n_y T_\beta + T_\alpha)m \qquad (6)$$

が必要になる。このため、全体の処理では、これらの和となるので、以下の計算時間が必要である。

$$m((n_z + 2q\sqrt{n_x n_y})T_\alpha + n_x n_y (n_z + q^2)T_\beta) \qquad (7)$$

今、$n_x = n_y = n_z = n$ として、式1の基本アルゴリズムの計算時間と比較してみると、

- 基本アルゴリズム：$mn^3 T_\alpha$
- 改良アルゴリズム：
  $m((2q+1)nT_\alpha + (n^3 + q^2 n^2)T_\beta$

であり、n が大きい場合には、両者の間には、およそ、$n^3(T_\alpha - T_\beta)$ の差が生じることになる。つまり、解像度を上げた場合に、平面を利用した計算方法が非常に効果的であることが分かる。

4 並列アルゴリズムとPCクラスタへの実装

アクティブな3次元形状取得を考えた場合、視体積交差法が手法として適していることは前に述べた。逆に、この手法を、**3** で述べた改良アルゴリズムを用いて実装する場合を考えると、カメラと計算機の組を構成単位としたクラスタ型の計算機アーキテクチャは、

- シーンの状況に適応的に各カメラを独立に制御したり、各カメラで撮影された画像からのシルエット生成処理などを独立に処理できる。(分散処理)
- 対象の部分を詳細に捉えたり、広域なシーンで複数の対象を扱う場合、各カメラの役割を動的

に制御し、システム全体でシーンの状況に適応的に機能できる。(協調処理)

- シルエットからの3次元形状生成処理を複数の計算機で並列に実行できる。(並列処理)
- カメラと計算機を同時に追加すれば、システムの大幅な変更なく、より詳細な形状が生成できる。(拡張性)

などの点で適していると考えられる。

以下では、実際に作成したPCクラスタの概要について述べ、そのアーキテクチャを踏まえた改良視体積交差法の並列アルゴリズムについて説明する。

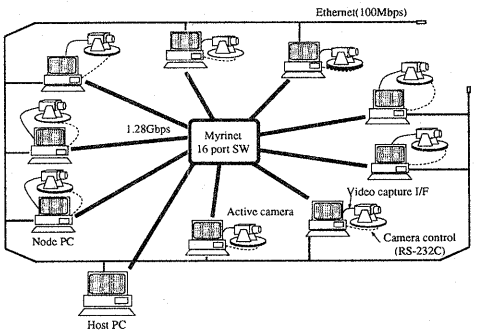

図 7: PCクラスタシステムの構成

4.1 システムの概要

まず、我々の作成したPCクラスタについて説明する(図7)。このPCクラスタは、10台のPC (CPU: Dual-PentiumIII 600MHz, Mem:256MB) をネットワークで相互に接続したものであり、10台中9台に計算機から制御可能な首振りカメラが接続され、残り1台で結果の表示とシステム全体の統括を行う。

首振りカメラとしては、視点固定型のパン・チルト・ズームカメラ (EVI-G20 by Sony) を用いる。このカメラは、仮想的な超広角画像を生成でき、視点固定の性質を利用した背景差分に基づく対象の追跡などの手法が開発されている [10]。

ネットワークとしては、100Mbps の EtherNet の他、全二重 1.2Gbps の高速ネットワーク (Myrinet by myricom) を使用する。このネットワークは、クロスバースイッチによる経路制御を行い、PC対PC間の通信高速に実行できる。提供される通信ライブラリを用いれば、通常転送 (25MB/s) の他、DMA 転送を利用して相手側のPCのメモリに直接データを書き込む転送 (最大 100MB/s) が可能である。特に後者は、640x480 のフルカラーの画像をビデオレートで転送できる能力を持ち、相互に画像データを交換する際に有用である。

4.2 並列アルゴリズム

前述の改良視体積交差法の処理の流れを再掲すると、以下のようになる。

SIP (silhouette image production) : カメラで撮影した画像を取り込み背景差分によってシルエットを作成する処理。

BPS (base-plane silhouette production) : 基準平面にシルエットを投影する処理。

LMC (local model creation) : 基準平面からスライス面にシルエットを投影する処理。

INT (intersection) : スライス上でシルエット同士の AND 演算を行う処理。

この中で、**SIP**～**LMC** は個別のカメラで撮影されたシルエットに対する処理である。この中で、**SIP** と **BPS** は、撮影に際して1回実行されれば良いので、基本的にはカメラを持つ個々のPCで処理すれば良い。**LMC** と **INT** はスライスの枚数回実行する必要があり、特に **INT** は、PC間で必要なデータを授受した後にはじめて処理が可能になるという性質を持つ。これらを考慮すると、PC間でのシルエット情報を授受を、**LMC** の後で行う場合と、前で行う場合の2種類のアルゴリズムが考えられる。そこで、以下の2種類の並列化手法について検討する。

スライス内分割法 : 処理の基準であるスライス内を分割し、各PCに割り当てて処理する方法。シルエットの授受は **LMC** の後で行う。

基準平面複製法 : 基準平面シルエットをすべてのPCに複製し、個々のPCが異なるスライスの再構成を行う方法。シルエットの授受は **LMC** の前に行う。

4.2.1 スライス内分割法

スライス内分割法は、**LMC** までをそれぞれのPCで実行し、一つのスライスをPCの数に分割して、それぞれのPCが担当する部分の再構成を担当する方法である (図8)。この手法では、**LMC** と **INT** の間で、各PCが担当する部分について、他のPCが算出したスライス上の部分シルエットを交換する。

この方法の利点としては、各スライスが処理単位であるため、各PCは基準平面シルエットと、今処理しているスライス上のシルエットを保持するメモリのみを持てば良く、必要なリソースは小さくて済む。しかし、システム全体でスライスを順番に処理する

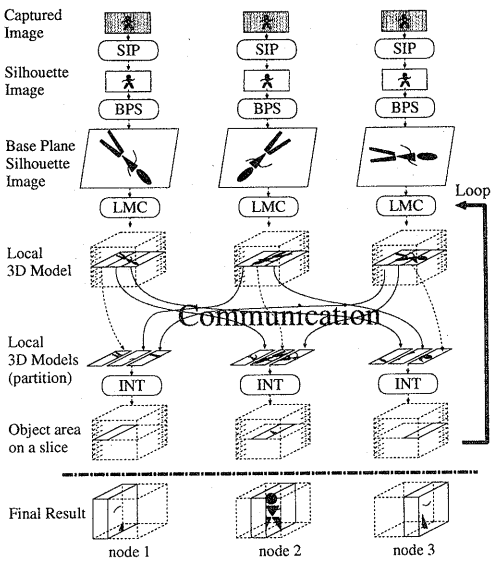

図 8: スライス内分割法

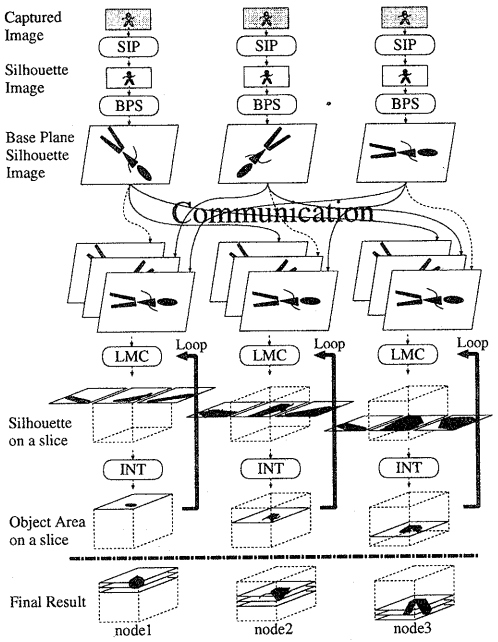

図 9: 基準平面複製法

ため、スライス毎に処理開始の同期が必要になる。この時、実際には物体が存在しない部分を担当するＰＣと、物体が存在する部分を担当するＰＣで処理時間に差が生じ、同期までの待機時間が無駄となる。さらに、各スライスの処理毎に画像の転送が発生するため、再構築の解像度を高く設定した場合に通信のオーバヘッドが大きくなるという問題が生じる。

4.2.2 基準平面複製法

基準平面複製法では、各ＰＣで算出した基準平面シルエットを全てのＰＣ上に複製する。これにより、全てのＰＣが全ての３次元形状再構成を行う能力を持つことになる。各ＰＣは統括ＰＣ上で動くスケジューラによって指示されたスライスを再構成する処理を全体の形状が求まるまで繰り返す方法である (図 9)。この時、データの転送は、**BPS** と **LMC** の間で行われ、全ての基準平面シルエットが全てのＰＣに転送される時間が必要となる。スケジューラは、形状の再構成を行うスライス番号リストを保持し、各ＰＣからのジョブリクエストに対して、算出が終っていないスライス番号を返す。各ＰＣは指示されたスライスについてすべてのシルエットに対する **LMC** とそれらの **INT** を行い、そのスライス上の形状を獲得する。１つのスライスの処理が終れば、再びスケジューラにジョブリクエストを出すという処理を繰り返す。

この方法では、並列処理の最初に基準平面シルエットの複製を配る通信時間が必要となる。また、各ＰＣは全ての基準平面シルエットの保持だけでなく、あるスライスについて３次元形状復元処理の全てを行う必要があるため、ある程度大きいメモリを必要とする。しかし、全てのＰＣがどのスライスの処理でも行えるため、ＰＣ間での同期の必要がなく、早く処理が終ったものから次のスライスの処理に移ることが出来るため、ＰＣの待機時間が必要なく、全体として、高速な処理が可能となる。このことから、以下の実験では、この基準平面複製法を用いることにする。

4.3 動的負荷分散

身体動作を対象とする場合、撮影の入力映像の状況も刻一刻と変化する。この時、例えば、視線制御や撮影のタイミングによって撮影に失敗する装置が発生する可能性が高くなる。この時、視体積交差法として全ての観測画像を使うと、撮影失敗によって生じた入力変動の影響によって、必要な部分の形状が欠てしまうなどの不都合が発生する。そこで、各ＰＣで判断した撮影の成功／失敗の情報を基準平面の複製の際に同時に授受し、撮影に成功した画像情報のみを使用して、３次元形状の生成を行う。

これにより、例えば、ある１台のカメラが撮影に失敗した場合でも、多少の精度は落ちるものの連続的な形状復元を継続的に実行することが可能となる。

さらに、カメラの接続されていない計算機をシステムに増設する場合、撮影失敗と同様の扱いをすることによって、複製後の各スライスの計算処理を並列実行させることができ、ソフトウェアの変更なしにシステムを拡張できるという利点を併せ持つことになる。

5 実験結果

ここでは、9台のカメラを図10のように設置して行った実験の結果について述べる。この実験では、基準平面複製法を実装して利用している。

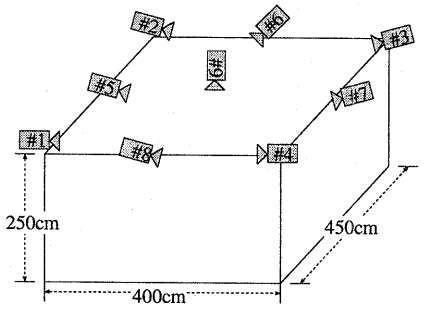

図 10: 実験に用いたカメラ配置

まず、部屋の中央に等身大のマネキンを置き、9台のカメラで同時に撮影て得られた基準平面シルエット像を図11に示す。カメラとの位置関係によって、異なるサイズのシルエット像が生成される。これらから再構築された形状の結果を図12に示す。解像度は1cm立方のボクセルサイズである。この時の計算時間は約1.5秒であった。マネキンの手足の形状など、詳細な形状が得られていることが分かる。

次に、処理を連続的に実行して、実際の人間の身体動作を映像化する実験を行った。最終的な表示例を図13に示す。ここでのボクセルサイズは2cmとした。身体動作の連続的な動きが映像化されていることが分かる。この時の各ＰＣでの各処理の計算時間を表 1にまとめる。なお、装置の都合で8台計算機（カメラ）を使用している。複製処理時間がＰＣによって異なっているのは、**SIP-BPS** の処理が早く終了したものから複製の処理に移るものの、**LMC-INT** の処理に移るためには、**SIP-BPS** の処理が遅く終了したＰＣからの基準平面の複製を待つ必要があるためである。結果として、1フレームの処理については、全てのＰＣでほぼ同様の計算時間が必要となっており、このシステムでは、現時点で秒3回の程度の形状取得が実行できる。

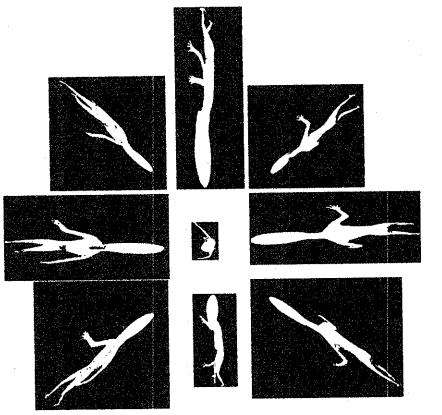

図 11: 基準平面シルエット像

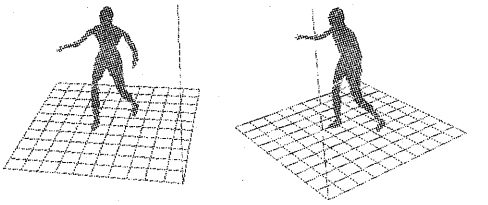

図 12: 3次元形状算出例 (1cm ボクセル)

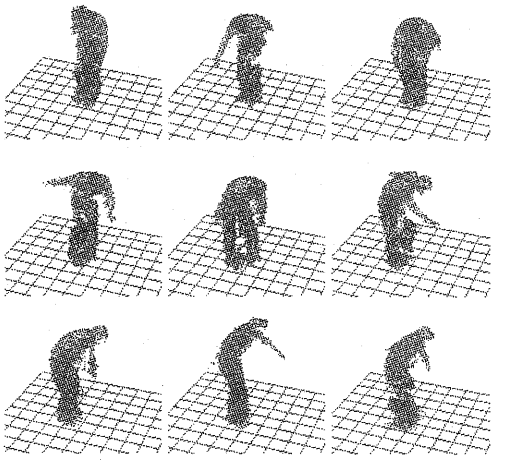

図 13: 身体動作の3次元映像例 (2cm ボクセル)

表 1: 各ＰＣでの各処理の実行時間（2cm ボクセル）

	SIP -BPS	Dupli- cation	LMC -INT	total
No.1	89ms	76ms	82ms	297ms
No.2	85ms	88ms	105ms	300ms
No.3	89ms	91ms	108ms	306ms
No.4	107ms	47ms	107ms	302ms
No.5	81ms	89ms	112ms	303ms
No.7	78ms	94ms	102ms	299ms
No.8	85ms	85ms	101ms	298ms
No.9	72ms	105ms	112ms	299ms

6 まとめ

本文では、アクティブな人物の３次元形状再構築を目指して、視体積交差法に基づいた形状復元において、平面から平面への投影が線形演算で実行できることに注目した視体積交差法の高速な算出方法と、その手法をＰＣクラスタに実装するための並列アルゴリズムの開発を行った。結果として、９台のカメラからの入力画像に基づいて、形状復元を行うシステムを構築し、人間程度の大きさの対象を 1cm 立方の解像度で 1.5 秒に１枚程度、2cm 立方の解像度で秒３枚程度の形状復元が可能となっている。

今後は、アクティブな身体動作の獲得と映像化を考えると、各カメラに人物追跡の機能を付加したシステムへの拡張を予定している。この中で、各カメラの制御、それらの協調動作、形状算出のための同期の問題を扱う必要がある。これらに加えて、より高精度で高速な形状復元を合わせて考える必要があり、以下の事項が挙げられる。

- 形状復元精度の向上：視体積交差法の性質上、表面が凹な部分の復元が困難である。そこで、表面の陰影やテクスチャの情報を利用した方法、例えば、space carving [11] などと組み合わせるなどして、復元形状の高精度化を図る。
- 提示方法の改良：表面テクスチャを併せて計算し、最終形状にマッピングして表示するなど、結果のリアリティの向上を図る。この時、撮影カメラと異なる視点での映像提示のためには、光学的にも妥当なテクスチャを合成する必要がある。

謝辞

本研究を行うにあたり、日本学術振興会未来開拓学術研究推進事業 (JSPS-RFTF 96P00501) の補助を受けた.

参考文献

[1] H. Saito, S. Baba, M. Kimura, S. Vedula, T. Kanade, "Appearance-Based Virtual View Generation of Temporally-Varying Events from Multi-Camera Images in the 3D Room", Computer Science Technical Report, CMU-CS-99-127 (1999).

[2] E. Borovikov, "Toward real-time volume reconstruction", http://www.umiacs.umd.edu/users/yab/3DSceneMonitor/rtvr.html

[3] W.N. Martin and J.K. Aggarwal, "Volumetric descriptions of objects from multiple views", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.5, No.2, pp.150–158 (1983).

[4] P. Srinivasan, P. Liang, and S. Hackwood, "Computational geometric methods in volumetric intersection for 3d reconstruction", *Pattern Recognition*, Vol.23, No.8, pp.843–857 (1990).

[5] M. Potmesil, "Generating octree models of 3D objects from their silhouettes in a sequence of images", *Computer Vision, Graphics, and Image Processing*, Vol.40, pp.1–29 (1987).

[6] R. Szeliski, "Rapid octree construction from image sequences", *CVGIP: Image Understanding*, Vol.58, No.1, pp.23–32 (1993).

[7] M. Okutomi and T. Kanade, "A multiple-baseline stereo", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.15, No.44, pp.353–363 (1993).

[8] S. Moezzi, Li-Cheng Tai, and P. Gerard, "Virtual View Generation for 3D Digital Video", *IEEE MultiMedia*, Vol.4, No.1, pp.18–26 (1997).

[9] R.T. Collins, "A space-sweep approach to true multi-image matching", *IEEE Computer Vision and Pattern Recognition*, pp.358–363 (1996).

[10] 和田俊和, 浮田宗伯, 松山隆司, "視点固定型パン・チルト・ズームカメラ とその応用", 信学論, Vol.J81-D-II, No.6, pp.1182–1193 (1998).

[11] K.N. Kutulakos and S.M. Seitz, "A theory of shape by space carving", *IEEE International Conference on Computer Vision*, pp.307–314 (1999).