レーザポインタを用いた静止物体の三次元復元システムの開発

田房　友典＊・長谷川　将司＊＊

Three-Dimensional Modeling of an Object by Using a Laser Pointer

Tomonori Tabusa＊ and Masashi Hasegawa＊＊

Abstract

This paper proposes a technique for recovering 3-D object by using a laser pointer. Our technique enables us to the 3-D object without markers on the object. A point irradiated by a laser pointer is detected as a feature point by image processing. The feature point is tracked on the image streams and stored in a measurement matrix. The matrix is decomposed into a camera orientation and a shape matrix by the singular value decomposition method. The shape matrix give the coordinate of recovered of 3-D object. In the practical experiments, we show some examples recovered the 3-D objects such as a cube, a columns and plate.

1．はじめに

三次元モデリングにおけるデータの取得方法には，カメラを利用する光学式と，センサや磁気などを利用する非光学式がある[1)]。非光学式法は，磁気などを用いるため測定環境の制約が多く，汎用性がないが，高精度の三次元モデリングが可能である。それに対して光学式は，カメラを用いて三次元計測を行う。近年，カメラが小型かつ高精度化され，ネットワーク接続が容易になったため様々な分野への応用が期待できる。

光学式による三次元モデリングは，計測対象に特徴点としてマーカーを取り付け，カメラで撮影されたその特徴点の座標を抽出する。手動による特徴点抽出作業は，人の労力となり即時性を妨げや復元精度の低下にもつながるため，自動抽出に関する研究も行われている[2，3)]。比較的多く行われている手法は，反射型のマーカーを取り付けて，そこに光源を与えることで画像処理を容易にし，自動抽出を行っているが，室内を暗くしたりするため一般的な環境での応用は困難である。また，相関法を応用して半自動で座標抽出を行い，測定環境に柔軟に対応する座標抽出法も報告されている。しかし，大型構造物や微小物体，人の立ち入りが困難な場所や空間においては，マーカーを付着させることはできない。

本研究では，静止している計測対象にマーカーを取り付けず，複数台のカメラで撮影するだけで，三次元モデリングを行うシステムを開発する。実現には，計測対象をレーザポインタによって走査し，その映像をDirect Xを用いてカメラ複数台で同時にキャプチャする。画像上に写る計測対象に照射されたレーザポインタの座標を画像処理によって自動抽出する。三次元座標の算出は，任意の大きさや距離の物体に柔軟に対応するために，カメラキャリブレーション不要の因子分解法[4)]を用いる。

2．Direct Xを用いたリアルタイムキャプチャ

2．1　カメラ画像キャプチャの概要

DirectXとは，Windowsのマルチメディア機能を強化するためのライブラリ（API群）のことである。画像のキャプチャには，ライブラリの一つであるDirect Showを用い，その原理を次に示す。

Direct Showライブラリは，動画や音声等のデータのストリームをフィルタグラフで扱い，フィルタグラフマネージャに再生すべきメディアが与えられると，その処理に必要な圧縮解凍のためのフィルタなどを自動的に選択し，フィルタグラフを構築する。カメラから入力された画像は，順にデバイスフィルタ，グラバフィルタ，ディスプレイへと出力される。グラバフィルタからデータを取得したサンプルグラバは，BYTE型の画像データをコールバックとして外部出力する。この「コールバック」を参照することで，フレーム画像を取得することができる。

＊情報工学科
＊＊情報工学科卒業生

平成17年 9 月30日受理

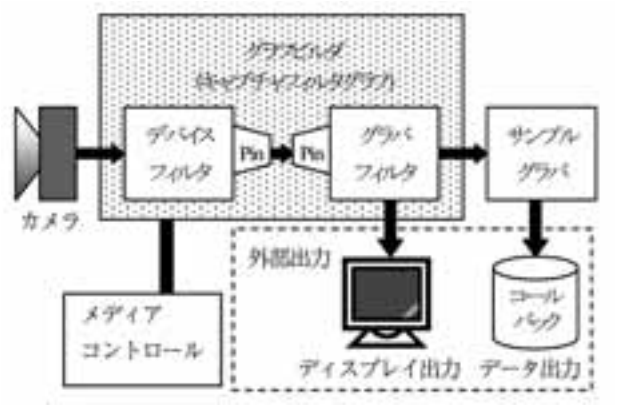

図1　カメラ画像をディスプレイに表示する仕組み

デバイスフィルタとグラバフィルタおよびディスプレイは，「グラフビルダ」という図1に示すキャプチャ・フィルタグラフによって接続され，メディアコントロールによって全体を制御できるため，カメラ画像をキャプチャすることができる[5)]。

2．2　複数台のカメラ画像キャプチャ

多視点画像により三次元データを取得するためには，1台のコンピュータ上で複数台のカメラを用いてキャプチャしなければならない。カメラ*N*台からそれぞれのフレーム画像を得るには，図2に示すようにそれぞれのカメラに対して，デバイスフィルタ，グラバフィルタが必要である。この*N*個のフィルタを管理するフィルタグラフ・マネージャが*N*台のデバイスを識別するために，これらのフィルタはそれぞれ個別にフィルタグラフに追加しなくてはならない。

デバイスの識別には，デバイスフィルタ生成後，デバイス列挙子を生成し，コンピュータに接続されたビデオキャプチャデバイス（カメラ）を識別する。識別されたビデオキャプチャデバイスをそれぞれ別にキャプチャ・フィルタグラフに追加し，グラバフィルタも同じく，それぞれのデバイスフィルタに対応するような場所に挿入することで，カメラ*N*台それぞれのフレーム画像を取得することができる。

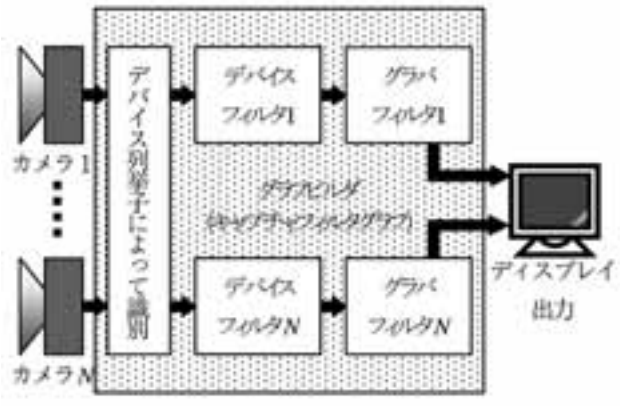

図2　複数台のカメラ接続によるグラフビルダの構成

2．3　複数台のカメラ接続

デバイスコントローラが複数のポートを持つ場合，1つのコントローラだけで複数のカメラデバイスを接続することは可能である。しかし，実際にはコントローラが持つ周波数帯域幅を超過してしまいキャプチャできない，もしくはフレーム伝送に遅延が生じるという現象がしばしば起こる。従って，カメラ1台に対して1つのデバイスコントローラを用いることが好ましい。この割り当てによって，前項のデバイス識別も容易に行なえ，スムーズなフレーム伝送を行う事ができる。

3．レーザーポインタの走査

3．1　座標値の算出

被写体に写像されたレーザポインタの光を特徴点として抽出する。特徴点の抽出は，連続してキャプチャする映像を処理するために高速処理でなければならない。

抽出処理は，フレームに投影されたレーザポインタのRGB（24bit）輝度成分を閾値によって識別する。実験的にレーザポインタのRGB輝度成分の値は，表1に示す範囲の値でフレームに投影されており，この値を閾値として，フレーム画像をサンプル時刻毎に走査して，抽出された画素の重心値(x_g, y_g)を求める。

1フレーム中において，画像処理によって抽出された画素の個数をP，抽出された画素のx座標値の総和をx_m，y座標値の総和をy_mとすると，重心値(x_g, y_g)は，

$$x_g = x_m / P \qquad (1)$$

$$y_g = y_m / P \qquad (2)$$

となり，式(1)，(2)で求めた重心値(x_g, y_g)を特徴点の座標値とする。但し，表1は，本実験環境での値であり，カメラの種類や光量，計測対象などの環境による変化に対応するため，プログラム上ではユーザが設定を変更できる。

表1　レーザポインタのRGB輝度成分

RGB 成分	輝度範囲
R	255
G	245-120
B	0-170

3．2　特徴点座標の保存

前項の式(1)，(2)で求めた重心値は，ファイルへ書き込む必要があるが，サンプル時刻毎にレーザポインタを走査した座標値をファイルへ直接書き込むとエラーが発生する。これは，プログラミングによってファイルの入出力を行う場合，通常は直接にファイルを読み書きすることなく，C/C++では，ストリームと呼ばれるデータ構造を使用するためである。具体的に，入出力時にバッファリングを行う必要があり，出力はストリームに一旦格納され，データが適当な大きさを超えたとき，はじめて物理的媒体であるファイルへ書き出される。バッファリングの途中，または書き込み処理中に，プログラムが異常終了したり，他の処理が行われたりした場合などには，ファイルへの書き込みが途中で終了したり，正確に書き込まれないなどの現象が起こる。このため，特徴点の座標値は一旦メモリに格納しておき，一定時間毎にファイルへ書き込む必要がある。

実際には，複数台のカメラを使用するため，それぞれのカメラ画像から算出されたx座標，y座標をメモリからファイルへそれぞれ書き込む。処理を行なうCPUやメモリの大きさによって数値は異なるが，本実験では，書き込みは，バッファリング処理などを考慮して，100点ずつ行う。このとき，ファイルへの書き込み速度がバッファリング作業に対して遅いため，プログラム上ではsleep関数などを用いて時間を遅らせている。また，ある時刻において１つ以上のカメラで特徴点が抽出されなかった場合，全てのカメラから得られた特徴点をファイルへは書き込まないように同期処理を行なう。

4．三次元復元システム

4．1　システムの概要

パソコン１台にUSBカメラ２台を接続し，静止物体をレーザーポインタで走査すると静止物体の三次元形状を復元するシステムを開発する（図３）。復元のアルゴリズムには，拡大計測行列に基づく復元法[6]を用いる。

図３　システム概観

システムの開発環境を表２に示す。

Direct X9.0 SDKはMicrosoft社のホームページ[7]から，Borland C++ Compiler5.5（以下BCC）はBorland社のホームページ[8]からそれぞれダウンロードする。Direct X SDKにはBorland社製のコンパイラで扱うことのできる形式のインポート・ライブラリが含まれていないため，Borland用のインポート・ライブラリで変換する[9]。

表２　システム開発環境

項目	仕様
PC	PentiumⅢ 1GHz 128MB RAM
OS	Windows2000
言語	C++
ライブラリ	Microsoft 社 DirectX9.0 SDK (Direct Show)
コンパイラ	Borland 社 Borland C++ Compiler5.5
カメラ	サンワサプライ社 USB PC カメラ CMS-V13
USBコントローラ	IO-DATA 社 USB2-PCI2 USB 2.0 interface
レーザポインタ	PLUS 社 Corp. LP320 635nm 1mW

4．2　アプリケーションソフト

カメラキャプチャとレーザポインタの走査制御を行なうアプリケーションソフトの概要を示す。

まず，プログラムを起動すると図４のメインウィンドウが表示されDirect Showの初期化が行なわれる。初期化処理中に，ビデオキャプチャデバイスであるカメラの識別が行われ，カメラは複数のデバイスフィルタによってそれぞれを識別することができる。

次に，図５に示すストリーム形式設定ウィンドウが表示される。このサブウィンドウではカメラストリームの解像度，映像フォーマット，フレームレート等を設定する。設定確認後，図６のRGB閾値設定ウィンドウが表示され，レーザポインタを走査するためのRGB成分それぞれの下限値と上限値を設定する。デフォルト値は，Rが255，Gが245～120，Bが170～0である。設定終了後，アプリケーションは，カメラストリームをグラバフィルタより取得し，ディスプレイに表示させる。このとき，グラバフィルタを通過するストリームはすべて，レーザポインタの検出および座標値の算出が行なわれている。メインウィンドウの「開始」ボタンを押すと，走査して得られた特徴点の座標値を出力するためのファイルが開かれ，メモリへの書き込みが行なわれる。走査終了後，MENUから「特徴点抽出」メニューを選択すると，ファイルに特徴点の座標値が書き込まれる。

図 4　メインウィンドウ

図 5　ストリーム形式設定ウィンドウ

図 6　RGB閾値設定ウィンドウ

表 3　計測環境

計測対象	計測時間(秒)	処理したフレーム数	同期した特徴点数(個)	書き込み時間
直方体	185.0	2045	1901	3分49秒
円筒	36.3	386	180	22秒

図 7　実験環境

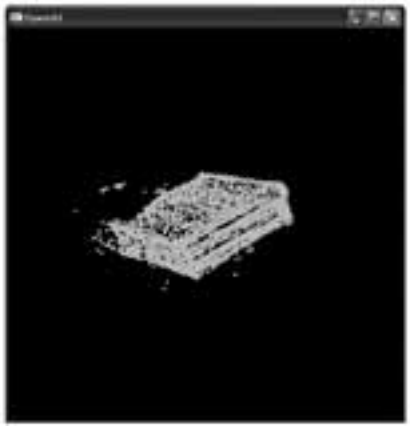

図 8　直方体のモデリング結果

5．復元実験

5．1　物体のモデリング

計測対象として直方体の箱，円筒，丼器の 3 点を選択し，三次元モデリングを行う。提案システムによって得られた特徴点の二次元座標から，因子分解法によって三次元座標を算出し三次元モデリングを行う[10]。計測データを表 3 に，計測環境を図 7，図 8 に特徴点を三次元空間上にグラフィカルに表示させたモデリング結果を示す。

5．2　システムの評価実験

前項での計測対象を，計測条件を変化させながら三次元モデリングし提案システムの評価を図る。計測対象をモデリングした結果，実際に処理したフレーム数はフレーム落ちが発生しているため，取得した理論フレーム数（フレーム率×計測時間）とは大きく異なる。また，処理したフレーム数と抽出した特徴点数の差から，両カメラ間で同期していない特徴点が存在することも明らかである。

システムの評価を行なうために，欠落フレーム率，欠落フレーム数，両カメラ間の同期率，ファイルへの書き込み速度を定義し，各実験におけるデータを表4に示す。但し，欠落フレーム数と欠落フレーム率は1秒当たりに何フレーム取得できたかを計測することができないため理論値を用いる。

$$\text{欠落フレーム率} = 1 - \frac{F}{\text{フレーム率} \times M} \tag{3}$$

$$\text{欠落フレーム数} = \frac{S \times M - F}{M} \tag{4}$$

$$\text{同期率} = \frac{Y}{F} \tag{5}$$

$$\text{同期率} = \frac{Y}{K} \tag{6}$$

ここで，F：処理したフレーム数，M：計測時間（秒），S：サンプリング時間（秒），P：処理したフレーム数，Y：同期した特長点数，K：書き込み時間（秒）とする。

表4より，本システムは取得フレームのうち平均6割以上を処理できず，フレーム落ちが発生している。フレーム率を30に設定した場合，1秒間に約19枚のフレームが失われている。このフレーム欠落原因は，画像処理にかかる処理時間の影響である。提案システムでは取得したフレーム画像を一旦一次元配列に展開し，閾値処理と重心値計算処理，特徴点追跡処理を行う。この処理を行うことで生じるタイムラグによるフレーム落ちが発生している。しかし，フレーム落ちが発生しても，両カメラ間で同期がとれる特徴点の数が増えれば，三次元計測としては大きな問題はない。両カメラ間の同期率はレーザポインタの反射やカメラの視差に等の影響を受け，計測対象によって変化するため一定ではない。以上の結果をまとめると本システムは，以下の特徴を持つ。

① 1秒間に特徴点をファイルへ約7.95×2（カメラ2台分）点書き込む。

② 1秒間に全取得フレームの約64.32%を失う。

表4　測定官許うによる復元結果

計測対象	計測した環境	書き込み速度（点・カメラ数／秒）	欠落フレーム数（枚／秒）	欠落フレーム率(%)	同期率(%)
直方体	通常の環境	8.30	18.95	63.15	92.96
	速く動かした場合	8.29	19.10	63.67	87.81
	遅く動かした場合	8.03	19.74	65.81	92.61
	明るい環境(1)	8.26	19.01	63.38	94.63
	明るい環境(2)	7.81	19.13	63.76	53.14
	暗い環境	8.31	19.63	65.44	94.62
	平均	8.17	19.26	64.20	85.96
円筒	通常の環境	8.18	19.37	64.55	46.63
	速く動かした場合	8.21	19.38	64.58	57.94
	遅く動かした場合	8.20	19.01	63.37	64.70
	明るい環境(1)	8.14	19.40	64.67	67.38
	明るい環境(2)	8.05	19.05	63.50	49.40
	暗い環境	7.48	19.95	66.52	67.19
	平均	8.04	19.36	64.53	58.87
井器	視差の大きい環境	7.14	19.19	63.96	40.76
	視差の小さい環境	6.89	19.24	64.12	56.96
	平均	7.02	19.21	64.04	48.86
平均		7.95	19.30	64.32	69.05

5．おわりに

本研究では，被写体に照射されたレーザポインタを特徴点として走査し，三次元計測を行うシステムを開発した。従来は，あらかじめ撮影されたカメラ画像から特徴点を手動で抽出していたが，提案システムは複数台のカメラ映像を同時キャプチャしながら，特徴点となるマーカーを取り付けず，特徴点座標を自動でファイルへ出力することができる。従来の手作業での特徴点抽出作業には多大な時間を要するが，提案システムでは大幅に時間を短縮できる。例えば，1600個×カメラ2台分の特徴点抽出作業に，手作業では約4時間を要するが，開発システムでは3分程度でファイルへ書き込みができる。また，特徴点の重心値の計算では，計測環境にも影響を受けるが，閾値を的確に設定できれば，撮影環境に応じて柔軟に重心位置が求めることができ，復元精度の高いモデリング結果を得ることができる。

提案システムによって得られた二次元座標を元に拡大計測行列に基づく因子分解法によって三次元モデリングを行った結果，レーザポインタを早く動かした場合と，ゆっくり動かした場合では大きな差が生まれる。直方体をモデリングした例では，人間の視覚的な感覚として，前者は直方体の形状を確認しにくいが，後者は直方体の形状を三次元モデリングしていることが確認できる。レーザポインタで計測対象をゆっくり走査すると，特徴点数は増加し，特徴点数の増加によって静止物体の三次元モデリング結果の形状の復元性を高めることができる。今後の課題として，特徴点走査アルゴリズム関係とシステムウェア関係の2項目が挙げられる。まず，前者では特徴点となるレーザポインタが計測環境によって反射し，その反射光も閾値処理によって抽出される問題がある。これは，前フレームの特徴点との相関によって反射光を抽出しないというアルゴリズムによって対応可能である。将来的には複数の特徴点自動抽出システムへの拡張が必要である。フレーム画像処理の部分にラベリング処理等を用いて改良を行い，特徴点数に合わせて抽出しなければならない。また，精度の高い特徴点抽出を行うには，特徴点となるマーカーの開発も必要となる。1フレーム中に複数個の特徴点を自動抽出することができれば，変形体の三次元計測が即時に行えるため，本研究の目的でもある疲労軽減，時間短縮，精度向上といった効果がさらに高まる。

後者では，Direct Showの初期化処理中のキャプチャデバイスの選択に課題がある。本研究では，この選択をプログラムコードで決定しているため，デバイスの変更があると再コンパイルが必要となる。ユーザが任意で選択することができるようになれば，より汎用的なシステムへと改善される。また，提案システムは特徴点をファイルに書き込んだあと，別のプログラムによってモデリング処理を行っている。特徴点がファイルに展開されてすぐにモデリング処理ができれば，全体のモデリング速度の向上につながり，またリアルタイム処理への拡張が実現できる。

参考文献

[1] ミニ特集　人間の運動計測とその応用，計測と制御，Vol.36, No.9（1997）.

[2] タンジュークイ，石川聖二，加藤清史：因子分解を利用したモーションキャプチャ法，計測自動制御学会論文集，Vol.36, No.11, pp.980-984（2000）.

[3] 香川仁美，田房友典：多視点画像処理による3次元モデルの生成～自動特徴点抽出に関する研究～，弓削商船高等専門学校平成13年度卒業論文（2002）.

[4] 金出武雄，コンラッドポールマン，森田俊彦：因子分解法による物体形状とカメラ運動の復元．電子情報通信学会論文誌，Vol.76，D-II，No.8，pp.1497-1505（1993）.

[5] 土井淳，大澤文孝，成田拓郎：DirectX8実践プログラミング，工学社（2002）.

[6] Tan. J. K, and Ishikawa. S, Human motion recovery by the factorization based on a spatio-temporal measurement matrix, Computer Vision and Image Understanding, Vol.82，No.2，pp.101-109（2001）.

[7] Microsoft DirectX Home Page，http://www.microsoft.com/windows/directx/default.aspx

[8] Borland Home Page :http://www.borland.co.jp/

[9] 横山大明，関口正浩，USB接続のパソコン用カメラで物体認識と動き検出，トランジスタ技術2月，pp.197-204（2003）.

[10] 田房　友典，安藤美紀：人の運動解析のための3次元表示に関する研究，弓削商船高等専門学校紀要第25号，pp.29-34（2003）.