

## 研究速報

自由視点映像生成のための自然特徴点を用いた  
多眼カメラの自己位置推定法高橋 桂太<sup>†a)</sup> (正員) 金子 正秀<sup>††</sup> (正員)

Self-Localization of a Multi-View Camera Using Natural Feature Points for Free-Viewpoint Image Synthesis

Keita TAKAHASHI<sup>†a)</sup> and Masahide KANEKO<sup>††</sup>, Members

<sup>†</sup> 名古屋大学大学院工学研究科電子情報システム専攻, 名古屋市  
Department of Electrical Engineering and Computer Science,  
Graduate School of Engineering, Nagoya University, Nagoya-  
shi, 464-8603 Japan

<sup>††</sup> 電気通信大学大学院情報理工学研究科, 調布市  
Graduate School of Informatics and Engineering, The Uni-  
versity of Electro-Communications, Chofu-shi, 182-8585  
Japan

a) E-mail: keita.takahashi@ieee.org

あらまし 筆者らは, 自由視点映像生成において観察視点の移動範囲を拡張するため, 多眼カメラを移動機構に搭載する撮影系を検討している. そのような撮像系を想定して, 自然特徴点の追跡と対応付けにより, 多眼カメラの位置と姿勢をリアルタイムに推定する手法を開発したので報告する.

キーワード 自由視点映像生成, 多眼カメラ, 自己位置推定

## 1. ま え が き

本研究では, 多眼カメラによる映像入力を用いて, リアルタイムに自由視点映像を生成する技術[1]を対象とする. この技術を用いれば, 視点位置を連続的かつ滑らかに移動しつつ, 被写体を三次元的に観察することが可能である. しかしながら, 視点移動できる範囲は多眼カメラの物理的なサイズに制約される. なぜなら, 映像生成は多眼カメラによって取得された光線情報の補間処理に基づくため, 取得範囲外の光線情報は生成できないからである.

この欠点を補う方法として, 多眼カメラを移動機構(ロボットアーム等)に搭載し, 観察視点の移動に応じて逐次的に多眼カメラを適切な位置に移動することにより, 視点移動の自由度の拡張を図ることが考えられる. 多眼カメラが固定されている場合, 観察視点が例えば右方向に移動していくと, やがては多眼カメラの光線取得範囲を逸脱するため, 画像が合成できなくなる. しかし, 視点移動に伴って同時に多眼カメラを右方向に移動すれば, 上記の逸脱を防止し, より広範囲な視点での画像合成が可能になる. 言い換えれば, これは, 観察視点の移動を, 自由視点映像生成技術と物

理的な移動機構の両方でサポートするという考え方である. この枠組みにおいて, 多眼カメラの移動中にも矛盾のない自由視点映像生成を継続するためには, 各時刻において多眼カメラの自己位置を推定する必要がある. そこで, 本研究速報では, 多眼カメラのうちの二視点の映像から抽出される自然特徴点を追跡することにより, 多眼カメラの自己位置(位置と姿勢)をリアルタイムに推定する手法を提案する.

本研究速報で述べる自己位置推定法は, 原理的には既知の技術の組合せである. しかし, 本研究で重要なのは, リアルタイム自由視点映像生成技術に物理的な移動機構の逐次制御を導入する独自のフレームワークの実現であり, その目的に適した自己位置推定手法を構築する点が本研究速報の主眼である.

## 2. 多眼カメラの自己位置推定手法

提案手法では, 対象シーンに特別なマーカ物体を置くことを前提とせず, 画像から自動的に抽出される自然特徴点のみを用いて, 各撮影時刻における多眼カメラの自己位置を推定する. この自己位置とは, 被写体空間に設定した基準座標系に対する多眼カメラの相対的な位置と姿勢の情報を併せたものである. 各時刻において多眼カメラの相対的な運動を求めることで, 多眼カメラの自己位置を最新の状態に更新する.

特徴点の追跡・対応付けにおいては, 同一時刻の異なる視点間での対応, 及び, 同一視点の連続する時刻間での対応の両者を維持・更新する. 特徴点追跡の過程においては, 特徴点の見失いや誤対応が頻繁に生じること虑しなければならぬ. また, 全ての被写体が完全に静止している場合は当然のこと, 一部の被写体が動いている状況も扱いたい. 運動物体上の特徴点と静止物体上の特徴点は, 共通の剛体運動の仮定を満たさない. そのため, 多眼カメラの運動を求めるときには, 運動物体上にある特徴点を外れ値として除外し, 静止物体上の特徴点だけを用いる.

特徴点の対応から多眼カメラの自己位置を推定するための幾何学的な手法については付録に詳述する. これらの手法は, いずれも線形方程式を解く問題に帰着され, 高速に計算できる. なお, 解の精度を更に向上するためには, 線形解法による解を初期値として, 再投影誤差を最小化する非線形最適化処理(バンドル調整)を行うことが推奨される[2]が, 本研究ではリアルタイム処理を目指すため, バンドル調整を省略し, 最低限の処理にとどめている. また, 単一のカメラの時系列情報のみから, カメラの自己位置と各特徴点の座

標を同時に推定する技術 (structure-from-motion) [3] もあるが、本研究では、問題設定上、多眼カメラの使用が前提となるため、視点間の対応も同時に用いることで問題を単純にしている。

各フレームごとの処理手順を以下のとおりである。ここで、 $A, B$  は多眼カメラの二つの異なる視点を表し、視点  $A$  における特徴点セットを  $\mathcal{P}_A = \{x_{A,n}\}$  のように表す。 $n$  は特徴点の番号である。

(1) 一時刻前の特徴点セットを  $\mathcal{P}'_A := \mathcal{P}_A$ ,  $\mathcal{P}'_B := \mathcal{P}_B$  として保持する。

(2) 新たに取得された画像  $I_A, I_B$  において特徴点を追跡し、 $\mathcal{P}_A, \mathcal{P}_B$  を得る。追跡に失敗した特徴点は削除し、全ての  $n$  について  $x'_{A,n} \Leftrightarrow x'_{B,n} \Leftrightarrow x_{A,n} \Leftrightarrow x_{B,n}$  の対応関係が満たされるようにする。特徴点の追跡には、高速に計算できて安定性も高い、Lucas-Kanade 法のピラミッド型の実装 [4] を用いる。最初のフレーム及び特徴点数がしきい値を下回った場合には、上記の実装の標準的なアルゴリズムで特徴点の検出を行うが、十分な数の特徴点が継続的に追跡できている場合には新たな検出は行わない。

(3)  $\{x'_{A,n}\} \Leftrightarrow \{x'_{B,n}\}$  の対応と付録の式 (A.1) から、現時刻の三次元座標点の集合  $\mathcal{X}' = \{\mathbf{X}'_n\}$  を得る。同時に、各三次元座標点の再投影誤差 [2] も計算する。同様に、 $\{x_{A,n}\} \Leftrightarrow \{x_{B,n}\}$  の対応関係から、一時刻前の三次元座標点の集合  $\mathcal{X} = \{\mathbf{X}_n\}$  を得る。再投影誤差が大きい場合には、誤対応である可能性が高い。そこで、再投影誤差にしきい値を設定し、 $\mathbf{X}_n, \mathbf{X}'_n$  のどちらか一方が基準を満たさない場合には、両者を  $\mathcal{X}$  及び  $\mathcal{X}'$  から取り除く。基準をパスした三次元座標点の集合の時刻間の対応を  $\mathcal{X} \Leftrightarrow \mathcal{X}'$  と表す。

(4) 剛体運動を表す付録の式 (A.3) のモデルに、RANSAC の枠組み [5] を適用することで、集合の対応  $\mathcal{X} \Leftrightarrow \mathcal{X}'$  から外れ値を除去する。特徴点の大部分が静止物体上にある場合には、運動物体上にある特徴点は自動的に外れ値とみなされる。外れ値を除いた集合の対応  $\mathcal{X} \Leftrightarrow \mathcal{X}'$  と付録の式 (A.4) を用いて、運動パラメータ  $\mathbf{m}$  を推定する。 $\mathbf{m}$  を用いて現在の多眼カメラの自己位置を更新する。

### 3. 実験及び今後の課題

実験には、ViewPLUS 社製の 25 眼カメラ Profusion 25 と、Xeon 2.66 GHz の CPU、メインメモリ 3.0GB を備えた PC を用いた。25 眼カメラの 5 行 5 列の視点配列のうち、中央行の左右両端のカメラを  $A, B$  とした。多眼カメラと PC を接続した状態で、多眼

画像の撮影、多眼カメラの自己位置推定、及び自由視点映像生成を繰り返し実行するソフトウェアを開発した。自己位置推定は C++ と OpenCV によって実装した。自由視点映像生成は、原理的には文献 [1] と同等の手法であり、GPU を用いて実装した。自由視点映像の観察視点は、マウス操作で自在に制御できる。

処理のリアルタイム性は、本研究で提案する、物理的な移動機構と自由視点映像生成を組み合わせたフレームワークにおいて必須の要件である。そこで、実装したソフトウェアの処理時間を、100 フレーム分の処理時間の平均として計測したところ、1 フレーム当たり約 95 ms (約 10 fps) であり、そのうち約 34 ms が自己位置推定 (約 430 点の特徴点を使用) に相当していた。現段階では、自己位置推定の部分では並列化等の工夫をしていないが、GPGPU を用いた実装等により、今後、いっそうの高速化が見込まれる。

更に、多眼カメラの自己位置が正しく推定されていることを確認するため、擬似的な stabilization を実装した。これは、多眼カメラの自己位置の変動を補償するように自由視点映像の観察視点を動かすことで実現される。多眼カメラを人手で動かしたときの結果の一例を図 1 に示す。動画 [6] も併せて参照されたい。キャプチャ画面の内部において、左は自由視点映像、右は視点  $A, B$  の映像である。 $A, B$  の映像には特徴点とその軌跡を重ね書きした。上下のキャプチャ画面を比較すると、視点  $A, B$  の映像は多眼カメラの水平方向



図 1 擬似 stabilization の結果  
Fig. 1 Result of pseudo-stabilization.

の動きに伴って大きく変化しているが、自由視点映像では物理的な移動が補償されているため、被写体の位置がほとんど変化していないことが分かる。なお、今回の実験において、被写体の大部分は静止しているが、画面中央やや左の植物の形をした玩具では、常に葉の部分が動いている。特徴点の一部はこの葉の上にあるが、RANSACの枠組みのおかげで、自己位置推定に破綻は生じない。

今後は、移動機構に搭載した多眼カメラに対して、本研究速報の提案手法を適用して有効性を確かめ、移動機構の制御と自由視点映像生成をシームレスに組み合わせた三次元可視化システムを実現したい。

謝辞 本研究はJSPS科研費24700162の助成を受けて行われた。

## 文 献

- [1] Y. Taguchi, K. Takahashi, and T. Naemura, "Real-time all-in-focus video-based rendering using a network camera array," 3DTV-Conference, pp.241-244, 2008.
- [2] R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University Press, 2004.
- [3] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: A factorization approach," Int. J. Comput. Vis., vol.9, no.2, pp.137-154, 1992.
- [4] J.-Y. Bouguet, "Pyramidal implementation of the Lucas Kanade feature tracker," Description of the algorithm, OpenCV Documents, 1999.
- [5] M.-A. Fischler and R.-C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," Commun. ACM, vol.24, no.6, pp.381-395, 1981.
- [6] <http://www.youtube.com/watch?v=FE2QIzoAaTY>

## 付 録

本付録では、提案手法の背景となる対応点の幾何学的な取り扱いについて述べる。より体系的な解説については、文献[2]等が参考になる。

### 1. 視点間の対応関係の扱い

2台のカメラの画像上で得られた点の対応から、その点の三次元座標を復元する方法を述べる。被写体空間の座標  $\mathbf{X} = (X, Y, Z, 1)^T$  にある点が、ピンホールカメラAによって撮影され、画面上の点  $\mathbf{x}_A = (x_A, y_A, 1)^T$  に投影されると仮定する。カメラAの3行4列の投影行列を  $\mathbf{P}_A$  とすると、 $\mathbf{x}_A \sim \mathbf{P}_A \mathbf{X}$  が成立する。 $\sim$  は定数倍の不定性を除いて両辺が等しいことを表す。別

のカメラBについても同様に、 $\mathbf{x}_B \sim \mathbf{P}_B \mathbf{X}$  が成立すると仮定する。これらの関係を用いれば、カメラA, Bの間で画面上の点対応  $\mathbf{x}_A \leftrightarrow \mathbf{x}_B$  が得られた場合、以下の線形方程式が成立し、三次元座標  $\mathbf{X}$  はその最小二乗解として求まる。

$$\begin{pmatrix} x_A \mathbf{p}_{A,3} - \mathbf{p}_{A,1} \\ y_A \mathbf{p}_{A,3} - \mathbf{p}_{A,2} \\ x_B \mathbf{p}_{B,3} - \mathbf{p}_{B,1} \\ y_B \mathbf{p}_{B,3} - \mathbf{p}_{B,2} \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = 0 \quad (\text{A.1})$$

ここで、 $\mathbf{p}_{-,i}$  は、行列  $\mathbf{P}_-$  の第  $i$  行ベクトルを表す。

### 2. 時刻間の対応関係の扱い

三次元空間の中で  $N$  個の点が剛体運動（相互の位置関係を保って並進と回転）する場合を仮定し、その運動を求める問題を考える。 $n$  番目の点が、座標  $\mathbf{X}_n = (X_n, Y_n, Z_n, 1)^T$  から座標  $\mathbf{X}'_n = (X'_n, Y'_n, Z'_n, 1)^T$  に移動した場合、 $\mathbf{X}'_n = \mathbf{M} \mathbf{X}_n$  が成立する。 $\mathbf{M}$  は、

$$\mathbf{M} = \left( \begin{array}{ccc|c} \mathbf{R} & \mathbf{t} \\ \hline 0 & 0 & 0 & 1 \end{array} \right) \quad (\text{A.2})$$

と表せる。 $\mathbf{R}$  は3行3列の回転行列、 $\mathbf{t}$  は3行1列の並進ベクトルである。行列  $\mathbf{M}$  のうち、上の3行4列の部分を行優先順で12行1列のベクトルに展開し、 $\mathbf{m} = (m_1, m_2, \dots, m_{12})^T$  と表すと、 $\mathbf{X}'_n = \mathbf{M} \mathbf{X}_n$  は以下のように書き直せる。

$$\bar{\mathbf{X}}'_n = \begin{pmatrix} \mathbf{X}_n^T & 0 \cdots 0 & 0 \cdots 0 \\ 0 \cdots 0 & \mathbf{X}_n^T & 0 \cdots 0 \\ 0 \cdots 0 & 0 \cdots 0 & \mathbf{X}_n^T \end{pmatrix} \begin{pmatrix} m_1 \\ \vdots \\ m_{12} \end{pmatrix} \quad (\text{A.3})$$

ただし、 $\bar{\mathbf{X}}'_n$  は、 $\mathbf{X}'_n$  の非斉次表現 ( $\mathbf{X}'_n = (\bar{\mathbf{X}}'_n, 1)^T$ ) とする。右辺の3行12列の行列を  $\mathbf{A}(\mathbf{X}_n)$  と表記すると、剛体運動の仮定を満たす  $N$  点の対応  $\mathbf{X}'_n \leftrightarrow \mathbf{X}_n$  ( $n = 1, \dots, N$ ) に対して、以下の線形方程式が成り立ち、運動のパラメータ  $\mathbf{m}$  はその最小二乗解として求められる。

$$\begin{pmatrix} \bar{\mathbf{X}}'_1 \\ \vdots \\ \bar{\mathbf{X}}'_N \end{pmatrix} = \begin{pmatrix} \mathbf{A}(\mathbf{X}_1) \\ \vdots \\ \mathbf{A}(\mathbf{X}_N) \end{pmatrix} \begin{pmatrix} m_1 \\ \vdots \\ m_{12} \end{pmatrix} \quad (\text{A.4})$$

更に、行列  $\mathbf{M}$  の左上の  $3 \times 3$  要素は回転行列であるため、 $\mathbf{R} \mathbf{R}^T = \mathbf{I}$  が要請されるが、本論文ではこの制約を明示的には加えていない。

(平成24年12月10日受付)