

即時生成型アニメーションのキー姿勢の実時間抽出

山口 将司 栗山 繁

豊橋技術科学大学大学院 工学研究科 情報・知能工学専攻

1 はじめに

CG 映像作品において、仮想人物に自然な動きをつけるにはモーションキャプチャデータが頻繁に用いられている。これを手描きアニメーション制作に利用する場合、毎フレームでの微小な動きや振動まで正確に記録しているため、そのままの状態を用いると生成結果に違和感のある動きが生じる問題が指摘されている。そこで、筆者らの研究室では過去に、手描きアニメーション独特の動作のメリハリなどを再現するため、動作による姿勢変化の特徴を求め、キー姿勢を自動判定してフレームをダウンサンプリングするシステムを提案した^[1]。実際のアニメーション制作に利用した結果、システムの利便性や生成結果、作業効率の向上などの点で一定の評価がアニメータから得られた。本研究ではさらに、オフラインでのキー姿勢抽出処理を、対象アニメーションの生成と同時に実行できるように拡張し、ビデオゲーム等の実時間で動作が生成される対話的な映像環境へ適用させることを目的とする。

2 時間局所的動作特徴を用いたキー姿勢抽出

2.1 姿勢突出度の定義

本手法で扱う人体動作データは、ルート関節を基準とする各関節の空間座標で構成された姿勢ベクトルを、時系列順に並べたもので表現される。2つのフレーム i, j 間の姿勢間距離 $D_{i,j}$ は、姿勢ベクトル同士の差分のノルムにより表される。

姿勢間距離の値を利用したキー姿勢抽出のための特徴量(以下、姿勢突出度)は以下の式で表される。

$$S_{f_i} = D_{f_{i-1}, f_i} + D_{f_i, f_{i+1}} - \alpha(D_{f_{i-1}, f_{i+1}}) \quad (1)$$

ここで、 f_i は i 番目のキー姿勢、 α は重み係数を表し、本稿後述の評価実験においては、 $\alpha = 0.5$ とした。図 1 中の曲線は姿勢遷移の様子を表しており、各線分が姿勢移動の速度、 θ が姿勢の変化方向とみなせる。 θ が π に近い場合、姿勢が一様かつ滑らかに変化している動作部分である。一方、 $(\theta - \pi)$ が小さくなると $D_{f_{i-1}, f_{i+1}}$ の値が小さくなり、姿勢突出度の値が大きくなる。この時は、運動の方向が急激に変化する姿勢箇所であるので、キー姿勢として抽出すべき重要な箇所であると判断される。

2.2 抽出アルゴリズム

本手法におけるキー姿勢抽出アルゴリズムを次に示す。

- 1) 抽出するキーフレームの総数を決定する
- 2) 全てのフレームでの姿勢に対し、姿勢突出度を求める
- 3) 最小の姿勢突出度を持つ姿勢のフレームを除去する
- 4) 除去されたフレームの1つ前と1つ後のキーフレームにおける姿勢突出度を、新たな隣接関係で再計算する
- 5) 1)で決定した数となるまで3),4)を反復する

“On-the-fly Key-pose Extractions for Real-time Animations”, Masashi Yamaguchi and Shigeru Kuriyama, Department of Information Engineering, Graduate School of Engineering, Toyohashi University of Technology

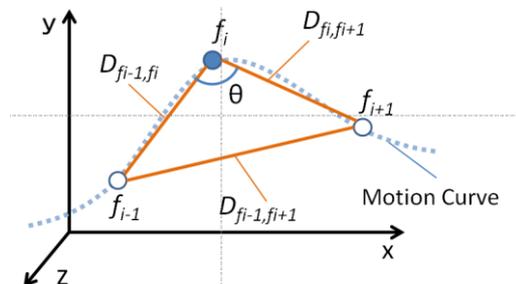
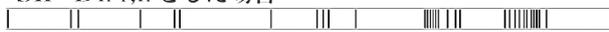
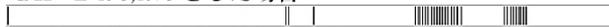


図 1 姿勢間距離と姿勢変化方向の概念図

$S_{f_i} = D_{f_{i-1}, f_i}$ とした場合



$S_{f_i} = D_{f_{i-1}, f_{i+1}}$ とした場合

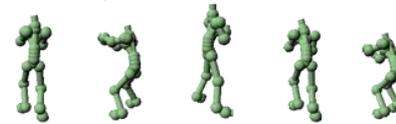


(1) 式を姿勢突出度とした場合



図 2 キーフレーム抽出結果の一覧

$S_{f_i} = D_{f_{i-1}, f_i}$ とした場合



(1) 式を姿勢突出度とした場合



図 3 46~149 フレーム間の各手法で抽出されたキー姿勢

4)での隣接関係の更新処理は、単に 2)で求めた初期値によるダウンサンプリングとなることを避けるものである。フレーム除去により前後キーフレーム間の時間的距離が増大することで、姿勢間距離が大きくなる場合があり、4)の処理により各キーフレームの姿勢突出度の順位が変化する可能性がある。これにより、初期時に突出度が高かった箇所にみにキーフレームが集中することが避けられる。

2.3 抽出結果の評価

2.2 節のアルゴリズムでの抽出結果の一例を図 2 に示す。姿勢突出度の有用性を検証するため、単に姿勢間距離を突出度 $S_{f_i} = D_{f_{i-1}, f_i}$ 、 $S_{f_i} = D_{f_{i-1}, f_{i+1}}$ として特徴量に使用して同様のアルゴリズムで抽出したものと比較する。使用したモーションデータは、静止した姿勢が続く箇所と急激な動作箇所が混在する動作内容となっている。図 2 は、総フレーム数 433 から、キーフレーム 31 を抜き出した結果を示しており、横軸が時間軸を表し、キーフレーム判定されたフレームには縦線が記録されている。図 3 は 46~149 フレーム目までのキー姿勢をピックアップし、時系列順に左から並べたものである。単に姿勢間距離を特徴量としたものは、データ後半の姿勢変化が激しい箇所にキー姿勢が

集中してしまい、前半の箇所でのキー姿勢の時間分布が疎になってしまっている。一方、姿勢突出度を用いた結果では、前後フレームでの速度・方向の変化が特徴量に含まれているため、前半の箇所でもキー姿勢が欠損することなく抽出できていることが確認できる。

3 生成とキーフレーム抽出の同時処理

前節での抽出処理は、動作データに対するオフラインでの処理であった。本節では、アニメーション生成と同時に、生成される毎フレームについて、キー姿勢となるか否かを判定し、前節のアルゴリズムと同等の処理を実時間で実現する機構について述べる。

3.1 実時間抽出のアルゴリズム

同時処理を実現するには、ある時点までに生成されたフレームまでの姿勢情報を用いて、未来のキー姿勢を予測する必要がある。そこで、モデルの様々な動作についてデータベースを作成しておき、生成された姿勢の時系列変化と類似するものを検索する方法をとる。姿勢の時系列変化を表す特徴ベクトル V_i を次に示す。

$$V_i = [X_i \ X_{i-1} \ X_{i-2} \ \dots \ X_{i-M}] \quad (2)$$

V_i はフレーム X_i と、それに至る直前の M 個のフレーム $X_{i-1} \sim X_{i-M}$ での各姿勢位置情報で構成される。これをデータベースの要素とする場合は、 $i=k$ 番目のキーフレームとし、 X_k と次のキーフレーム X_{k+1} までの毎フレームでの姿勢変化量の積算値 $I_{k,k+1}$ を求め、 V_i とセットの値とする。

V_i のデータベースは、モデルの様々な動作を生成しておき、それらに対し3節の抽出処理をオフラインで実行することによってキーフレームを取得して作成する。

次のキー姿勢の判定は、姿勢変化量 D_{f_{i-1},f_i} を、直前に選択されたキー姿勢のフレームから現在のフレームまでの間、毎フレームで求めて積算した値と、検索結果から得られた $I_{i,j}$ との大小関係で判定する。

このアルゴリズムの制約として、 V_i のデータベース作成時に、抽出アルゴリズムで指定したフレーム削減率(総フレームに対するキーフレーム数の比)が同一である必要がある。よって、実時間処理時に所望するフレーム削減率に合わせて複数のデータベースを用意する必要がある。

3.2 近似近傍探索手法を用いた計算時間の削減

V_k は大きな次元数を持つため、検索を単位フレーム時間(24[fps]の場合で41.667[msec])以内に完了させるために、近似近傍探索手法の一種である Locality-Sensitive Hashing(LSH)^[2]を用いる。LSHにより類似するデータを大まかに分類・取得した後に、それらを全探索するという方法をとる。360次元の特徴ベクトル5006個を持つデータベースからあるベクトルをクエリとし、完全一致するものを探索したところ、表1に示すとおり、平均で約400倍の検索速度が確認できた。ただし、マシンにはWindowsXP32bit, intel core-i7 2.67GHz, 3.23GBRAMを使用し、開発言語にはjavaを用いた。

3.3 評価

実時間抽出アルゴリズムの有用性を検証するために、2節の手法を用いたオフライン抽出結果をベースラインとの性能を比較した。3.2節で使用したデータベースと同様のものを使用し、幾つかのデータに対して各アルゴリズムで抽出処理を行った。このうち、走る→立ち止まるの

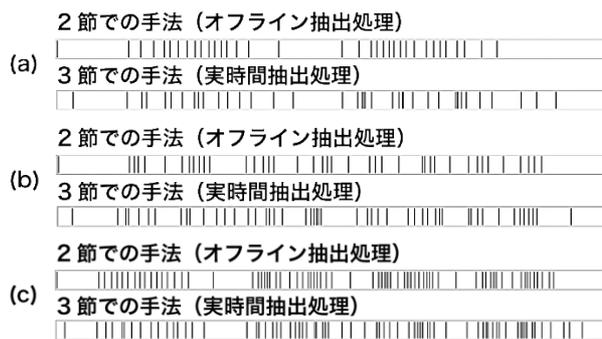


図4 抽出結果・ベースラインとの比較

表1 検索の所要時間[msec]

	全探索	LSH
最遅	11.244	0.029
10回の試行平均	10.748	0.027

表2 キーフレームベクトル間のコサイン類似度

モーションデータ	フレーム数	キーフレーム数	
		2節手法	3節手法
a) 歩く→しゃがむの繰り返し	360	36	40
b) 様々な方向へのジャンプ	420	36	43
c) 走る→立ち止まるの繰り返し	730	73	74

繰り返し動作についての抽出結果を図4に示す。図4は図2と同様に、時間軸上のキーフレーム位置を示している。図4より、両手法ともに動作の急激な部分と微少な部分が判別できており、ベースラインの結果におおむね近い抽出結果が実時間抽出アルゴリズムで得られていることが確認された。視覚的にも動作のメリハリを保有したキーフレームアニメーションがおおむね生成できていることを確認した。

4 今後の課題

4.3節で示した結果は、特徴ベクトルの次元数、学習に用いるデータベースの量、キー姿勢判定方法の一貫性といった様々な点で更なる考察が必要である。また、定量的評価のため独自の評価基準を検討していく必要が考えられる。本稿ではアルゴリズムの有用性を検証するに留まったが、今後は簡易な動作計測装置で実時間生成される動作データにも適用して有用性を検証する予定である。

参考文献

[1] S. Morishima, S. Kuriyama, S. Kawamoto, T. Suzuki, M. Taira, T. Yotsukura, and S. Nakamura, "Data-driven efficient production of cartoon character animation," in ACM SIGGRAPH 2007 sketches. ACM, 2007.
 [2] M. Datar, N. Immorlica, P. Indyk and V. Mirrokni S., "Locality-sensitive hashing scheme based on p-stable distributions", SCG 2004 Proceedings of the Symposium on Computational Geometry