

研究報告

舞踊符自動生成のための
自己組織化マップによる動作の分割化

柴田 傑**・湯川 崇†**・海賀孝明‡**・門脇さくら***・横山洋之***・玉本英夫***

A Motion Data Segmentation Method Using Self-Organizing Map
for Making a Buyo-fu AutomaticallyTakeshi Shibata**, Takashi Yukawa †**, Takaaki Kaiga ‡**,
Sakura Kadowaki***, Hiroshi Yokoyama*** and Hideo Tamamoto***

Abstract

We have proposed a human motion description method using a Buyo-fu(dancing note) in order to reuse motion data captured by a motion capturing system. The Buyo-fu can give us an efficient support for archiving and learning traditional dances. The motion of a dance can be segmented into some motion primitives, and we define a Buyo-fu as a code that is assigned to each motion primitive. Since we have been making a Buyo-fu manually so far, there exists a problem that making a Buyo-fu requires not only much cost but also an expert knowledge about the dance.

In this report, we propose a method to segment human motion data into some motion primitives in order to make a Buyo-fu automatically. We consider that human motion data consist of a series of human postures and that a motion primitive is a segment which consists of a series of human postures having similar characteristics to each other. According to these considerations, we discussed a method to segment human motion data into motion primitives by clustering postures. As a result, we have developed a clustering method for segmentation by using a modified self-organizing map(SOM) that can treat a time-series of data.

We made an experiment on a Japanese folk dance whose name is “Nishimonai Ganke”, and we confirmed that our proposed method could be used to make a Buyo-fu automatically.

1. はじめに

近年, モーションキャプチャリングシステム (MoCap) により, 3次元空間中での人の動きを精細にセンシングできるようになってきている. MoCap は, センサやマーカと呼ばれる目印の位置,

回転角を, 磁気や光によって取得することができるシステムである. 身体の部位毎に目印を付与することで, 人の動きを姿勢の時系列データとして取得することができる. これにより, 複雑な動きも精細に記録することが可能となっている. MoCap による動きの精細な情報は, 映画, テレビ, ビデオゲームなどの分野におけるアプリケーションだけでなく, 福祉や人間工学, ロボット工学, デジタルアーカイブなどのさまざまな分野で注目されている.

筆者らも舞踊の保存, 継承という場面において, MoCap による人の動きの情報に着目しており, これまでに, MoCap による動きの情報を有効に再利用する方法として, 舞踊符による動作記述法^{(1), (2)}を提案してきた.

2007年7月17日受理

**秋田大学大学院工学資源学研究科電気電子情報システム工学専攻, Department of Electrical Electronic and Computer System Engineering, Akita University.

***秋田大学工学資源学部情報工学科, Department of Computer Science and Engineering, Faculty of Engineering and Resource Science, Akita University.

† ノースアジア大学教養部, Division of General Education, North Asia University.

‡ 株式会社わらび座デジタルアートファクトリー, Digital Art Factory, Warabi-za.

舞踊の動作は個々の基本動作に分割することが可能であると考えられる。この基本動作に割り振った符号を舞踊符と呼ぶ。動作を舞踊符の表現で保存することにより、取得したデータに符号を用いてアクセスすることができ、有効に再利用することができる。

現在、舞踊符は人手によって作成されている。取得した動作を目視により基本動作に分割し、符号を割り振る作業には、多大な労力と専門知識が必要となる。そのため、自動で舞踊符を生成することが望まれる。

動作を基本動作に分割する手法としていくつかの研究例がある。分割と認識を同時に行う方法として HMM を改良した手法⁽³⁾や DP マッチングを改良した手法⁽⁴⁾などがある。これらは認識を目的としてパターンマッチングを行っている。このため、標準データや教師データを必要とし、舞踊符自動生成には適さない。教師データを用いない分割手法として、繰り返しが行われる区間を基本動作とする手法⁽⁵⁾がある。この手法では対象動作が繰り返し動作と限定されているため、舞踊には利用できない。筆者らは、基本動作の分割点付近において、注目部位の速度に関する特徴に基づいて分割する手法^{(6)・(7)}を提案してきた。しかし、全身の動きに関する扱いが課題であり、この手法を有効に利用するには、分割点付近での特徴についてのさらに詳細な知識が必要となってくる。

そこで、本報告では、現在手動で行っている舞踊符生成の自動化を目指し、専門的な知識を利用せずに基本動作の自動分割を行う手法を検討する。基本動作は類似の姿勢が連続する区間であり、姿勢をクラスタリングすることによって、基本動作に分割できると考えられる。クラスタリング手法として自己組織化マップ(SOM: Self-Organizing Map)⁽⁸⁾に着目する。SOM は教師なしでクラスタリングを行える手法で、対象に関する知識を必要としない利点がある。一方で、入力データの時間順は考慮されていないため、時系列データのクラスタリングには向かない。そこで、時間順を考慮するように SOM を改良することで、姿勢のクラスタリングを実現する。

2. クラスタリングによる舞踊符の生成

筆者らは動作データの記述方法として舞踊符を定義し、舞踊符を用いた動作データの再利用方法について検討してきた。

舞踊符とは対象となる動作を個々の基本動作に

分割し、各区間に割り振った符号である。動作を舞踊符の表現で保存することにより、動作データに符号を用いて容易にアクセスすることができる。さらに、Motion Composer^{(1)・(2)}を用いて舞踊符を並べて表現することにより新たな舞踊を生成することができる。これにより、舞踊などの伝統芸能の記録、保存、継承の場面で有効な支援が可能となると考えられる。

現在、舞踊符の作成は人手で行っている。これは、取得した動作データをビューアで再生しながら、目視により基本動作に分割し、符号を割り振る作業である。この作業は目視で行っているため、多大な時間と労力が必要となる。また、分割の基準が明確でなく、分割作業には対象動作に関する専門的な知識が必要となる。したがって、分割を行う際に、対象動作に関する知識や見解が統一されていない場合には、分割を行う人によって分割点が異なるという問題が発生する。このため、動作データを自動で分割し、舞踊符を生成することでこれらの問題を解決することが必要である。

そこで、動作データを姿勢の時系列データであると捉え、姿勢をクラスタリングすることによって分割を行う。図 1 に示すように、同一のクラスに属する姿勢が連続する区間を基本動作とし、そのクラス名が舞踊符であるという立場で舞踊符を捉えることで、舞踊符生成の自動化を目指す。

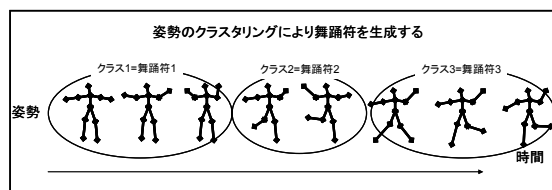


図 1 姿勢クラスタリングによる舞踊符生成

3. 主成分分析によるデータ圧縮

3.1 データの取得

分割する入力データとして、磁気式モーションキャプチャシステムである POLHEMUS 社の STAR TRAK モーションキャプチャリングシステムによって取得された全身のデータを用いる。

図 2- (a) に示す各部位に装着した 15 個のセンサーによってデータを取得する。身体を表現するモデルとして図 2- (b) に示すモデル^{(1)・(2)}を利用し、得られたデータからモデルの特徴点となる 15 箇所の部分について、位置と回転を計算し姿勢とする。特

徴点毎に 3 次元空間中での位置 (3 次元) および、回転 (3 次元) の計 6 次元のデータとなり、全身の動作データは 90 次元の時系列データとなる。

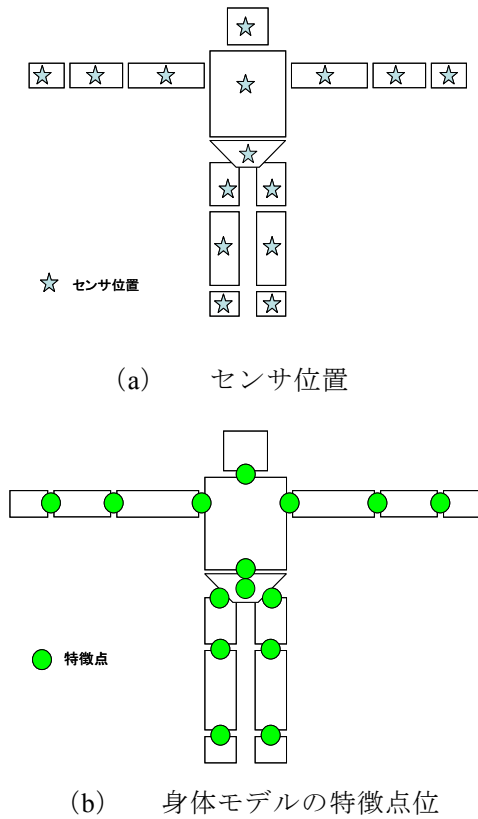


図 2 センサ位置と身体モデル

データの取得は 1 秒間に 30 回行う。データ取得の 1 単位をフレームと呼ぶ (以後、データの時刻はフレームを単位とする)。ある時刻 t で取得されるデータ \mathbf{f}_t は 90 次元のベクトルであらわされる。 \mathbf{f}_t は時刻 t での姿勢をあらわしており、MoCap により取得されるデータは、 \mathbf{f}_t の系列として得られる。長さ T フレームの動作は行列 \mathbf{M} を用いて式 (1) のように表現する。

$$\mathbf{M} = [\mathbf{f}_1, \mathbf{f}_2, \dots, \mathbf{f}_n]^t \quad (1)$$

このような姿勢の時系列データを動作データと呼ぶこととする。

3.2 基準化

MoCap で全身の姿勢を取得した場合、高次元のデータとなり、クラスタリングの計算に時間がかかると考えられる。そこで、計算の高速化のため、主成分分析⁽⁹⁾により次元の圧縮を行う。

主成分分析は入力となるデータ行列の列ベクト

ル間の分散が最も大きくなるような軸を成分とすることでデータの次元圧縮を行う方法である。そのため、ベクトル間で比較を行い共分散を求める必要がある。動作データは、位置と回転が混在するデータであり、列毎にデータの単位や分布は異なっているため、適切な比較が行えない。そこで、基準化を行うことにより、入力データの分布を整形し、各列ベクトルが平均 0、分散 1 の正規分布に従うようにする。これにより、列ベクトル間の比較が可能となり、主成分分析が可能となる。入力データ行列の要素を f_{ij} 、 j 列のベクトルの平均を m_j 、 j 列のベクトルの標準偏差を s_j とする。基準化されたデータ行列の要素 z_{ij} は式 (2) によって求めることができる。

$$z_{ij} = \frac{f_{ij} - m_j}{s_j} \quad (2)$$

3.3 主成分分析による次元の圧縮

基準化された動作データを用いて次の手順により主成分分析を行う。

まず、共分散行列 \mathbf{C} を求める。共分散行列は式 (3) で示す v_{jk} を各要素に持つ行列として求められる。

$$v_{jk} = \frac{1}{N-1} \sum_{i=1}^N (z_{ij} - m_j)(z_{ik} - m_k) \quad (3)$$

ここで、 N は動作データのフレーム長となる。

次に、 \mathbf{C} の固有値と対応する固有ベクトルを求めることで各成分の軸を決定する。 \mathbf{C} の固有値でもっとも大きいものを λ_1 とし、以後、大きさの順序を添え字で表し、 l 番目の大きさの固有値を λ_l と表現する。このとき、 λ_l に対応する固有ベクトル \mathbf{A}_l が第 l 主成分の軸となる。フレーム t での姿勢 \mathbf{z}_t の第 l 主成分 p_{tl} は、式 (4) で示すように \mathbf{A}_l と \mathbf{z}_t の内積で求めることができる。

$$p_{tl} = \mathbf{z}_t \cdot \mathbf{A}_l \quad (4)$$

ここで、主成分の上位 n 個のみを用いてデータを表現することで、次元の圧縮を行うことができる。固有値の総和 S と圧縮に用いる成分の固有値の和を用いて、累積寄与率 r は式 (5) を用いて求める。

$$r = \frac{\sum_{i=1}^n \lambda_i}{S} \quad (5)$$

式 (5) であらわされる累積寄与率は、圧縮後のデータが元のデータの情報をどの程度保存しているかを示す基準となる。ある閾値を設け、累積寄与率が閾値以上となるように圧縮後の次元 n を決定する。

時刻 t での MoCap からの入力を圧縮してできた n 次元のベクトル \mathbf{x}_t と、 \mathbf{x}_t の系列で表現される行列 \mathbf{X} は式 (6) (7) で表される。以後、 \mathbf{x}_t を入力データ、 \mathbf{X} を入力データ行列と呼ぶこととする。

$$\mathbf{x}_t = (p_{t1}, p_{t2}, \dots, p_{tm}) \quad (6)$$

$$\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T]^t \quad (7)$$

入力データは姿勢を表し、入力データ行列は動作を表しているともみなすことができる。

4. 自己組織化マップによる分割

4.1 時間軸を考慮した自己組織化マップ

舞踊における基本動作は類似の姿勢が連続する区間であるとみなして、動作の分割を行う。分割は、得られた姿勢をクラスタリングすることで、実現する。クラスタリング手法として自己組織化マップ (SOM : Self-Organizing Map) に着目する。SOM は教師なし学習を行うニューラルネットワーク (NN) の一種で、クラスタリングやベクトル量子化の手法として知られている。教師なし学習であるため、対象となるデータに関する事前の知識を利用せずにクラスタリングを行うことができる。SOM は入力層と出力層を接続するネットワークの重みが入力データに近づくように学習を行う。これにより、出力層のユニットが各クラスの代表データとなるようなクラスタリングが行われる。学習された重みを見ることにより、クラスタリングの後処理に必要な情報も同時に取得できるという特徴がある。

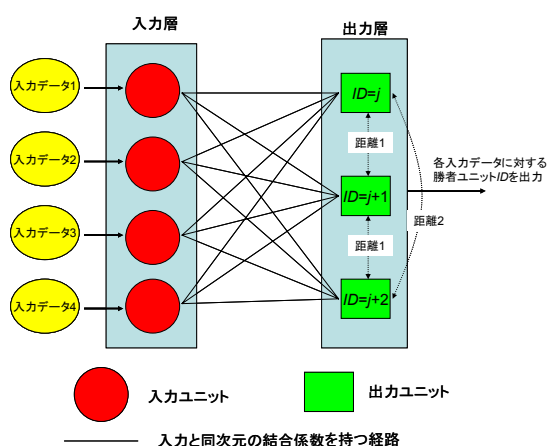


図3 SOM の例

SOM は図 3 に示すような、入力層と出力層の 2 つの層から構成される NN として表現することができる。図 3 では、出力層に存在するニューロン・ユ

ニットは 1 次元上に配置されているが、一般に出力層では n 次元空間にニューロン・ユニットを配置することができる。

入力層に存在するニューロン・ユニット (以後、単に入力ユニットと呼ぶ) に入力データを入力し、出力層に存在するニューロン・ユニット (以後、単に出力ユニットと呼ぶ) と重み付きの経路で接続した構造となっている。経路に付与された重みを結合係数と呼ぶ。出力ユニットは出力層内部での自身の位置を知っており、他のユニットとの距離、近傍領域が定義されている。また、ユニット名である ID が与えられている。入力層に入力を与えられると、入力データともっとも近い結合係数を持つ出力ユニットが発火する。発火した出力ユニットを勝者ユニットと呼び、勝者ユニットの近傍に位置するユニットを近傍ユニットと呼ぶ。

クラスタリングは適当な終了条件の下で、次の手続き 1 で結合係数を学習することで行われる。

[手続き 1] (結合係数の学習)

- step1: 結合係数を初期化する
- step2: 学習に用いる入力ユニットを決定する
- step3: 入力ユニットに対する勝者ユニットと近傍ユニットを計算する
- step4: 勝者ユニットと近傍ユニットの結合係数を更新する
- step5: 終了条件を満たしていなければ step2 へ戻る

step4 の結合係数の更新において、出力ユニットの位置関係が入力データ間の位置関係に近づくような学習規則を設定することにより、類似の入力データが共通の出力ユニットに対応付けされるような学習を行うことができる。学習された NN にデータを入力し、各データに対する勝者ユニット名をそのデータのクラス名とすることでクラスタリングを行うことができる。

SOM によるクラスタリングでは、入力データの入力順に対する考慮はなされておらず、そのままでは時系列データの分割に有効な手法とはいえない。

時間順を考慮せず、入力データと結合係数のユークリッド距離のみにより勝者ユニットを決定する SOM の例を図 4 に示す。

$t=f$ となる入力に対する勝者ユニットの ID を ID_f と表すと、図 4 に示すような入力データと結合係数の位置関係であった場合、 $ID_4 < ID_3$ となり、勝者ユニットの ID の大小が時間順を保存していない。時間順を考慮せず、勝者ユニットをユークリッド距離によってのみ決定する場合、このような対応付けが起こりうる。

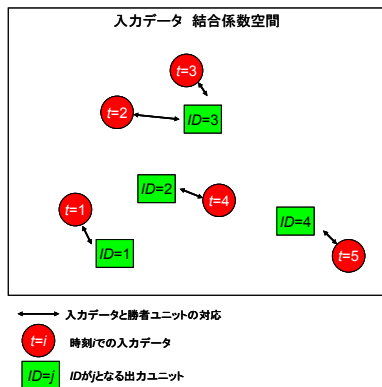


図 4 SOM によるクラスタリング例

動作の分割には入力データを時間順にクラスタリングすることが必要であるため、このような対応付けは好ましくない。

そこで、SOM の学習において、入力データの時間順を考慮して結合係数を更新するように改良を行う。これにより、時間的に連続し、同一のクラスに分類される区間を求める。

入力データの入力順を制御するカウンタと、以前の勝者ユニットを記録するメモリを用意し、これにより SOM のネットワークを制御することで、時間順を保存するように改良を加える。

時間順を考える場合、動作の開始から終了に向かう順方向と、終了から開始へ向かう逆方向の 2 種類の方向が考えられ、学習はそれぞれの方向に対して行うことができる。順方向学習はカウンタの値を開始のフレームから終了のフレームまで増加させる学習であり、逆方向学習はカウンタの値を終了フレームから開始フレームまで値を減少させる学習である。順方向と逆方向の 2 種類の学習を同時に行うこととし、順方向用と逆方向用にカウンタとメモリはそれぞれ 2 個を用意する。

4.2 時間順を考慮した SOM の構築

図 5 に提案する SOM の順方向学習の様子を示す。入力データの時間順を出力ユニットの ID が保存するような学習が行われるように改良したものである。従来の SOM に加え、入力を制御するカウンタと経

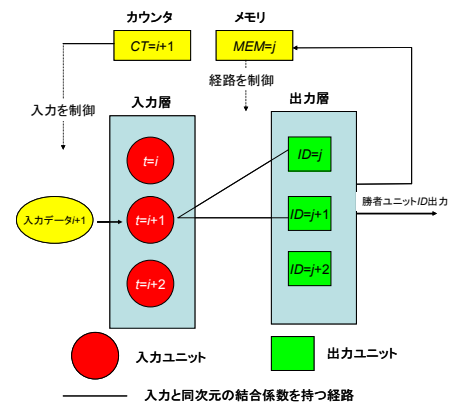


図 5 順方向学習中の提案 SOM

路を制御するメモリが追加されている。

入力層では、入力データのフレーム長 T と同数の入力ユニットを用意する。入力ユニットは、入力データの時刻を ID として持つ。

出力層では、 N 個の出力ユニットを用意し、1 次元上に配置する。出力ユニットは 1 から N までの値をとる ID を持ち、出力ユニット間には距離と近傍領域が定義される。出力ユニットの ID の差分をユニット間の距離とし、距離 1 の範囲を近傍領域と定義する。

カウンタ値 CT は 1 から T までの値をとり、 $CT = ID$ となる入力ユニットが入力を行なう。メモリ値 MEM には 1 つ前の入力における勝者ユニットの ID を代入する。

経路は入力データと同次元の結合係数を重みとして持ち、入力ユニットと出力ユニットを接続する。ある入力ユニットと接続される出力ユニットは、 MEM とその近傍に位置する出力ユニットを候補とし、学習順に従って選択される。順方向学習の場合は ID が MEM と $MEM+1$ である出力ユニットに接続され、逆方向の場合には ID が MEM と $MEM-1$ である出力ユニットに接続される。

図 5 に示した例では、 CT が $i+1$ であるため、 $ID=i+1$ である入力ユニットが入力を行う。ここで、順方向学習で $MEM=j$ であるため、入力ユニットは $ID=j$ および $ID=j+1$ である出力ユニットの 2 つのユニットにのみ接続される。勝者ユニットは接続された出力ユニットから選択される。

このようなネットワークを構築することにより、図 6 の例に示すような場合においても、 $t=4$ では $ID=2$ である出力ユニットは勝者ユニットの候補から外れるため、出力層の ID の順が入力データの時刻の順序を保存する学習を行うことができる。

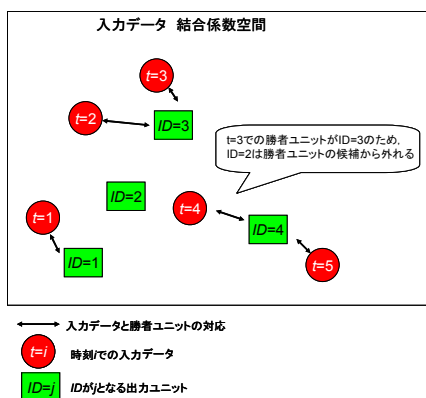


図6 時間順を考慮した勝者ユニットの決定

4.3 結合係数の学習

4.1節で示した手続き1の各stepを提案するSOMに適応し、結合係数の学習を行う。

学習順に従いすべての入力ユニットを入力し終えるまでを1世代と呼ぶこととし、学習の終了条件として世代数を与える。

step1の結合係数の初期化は図7に示すように、入力データそのものを用いて行う。入力データの系列から等間隔で入力データを取得し、結合係数の初期値として設定する。

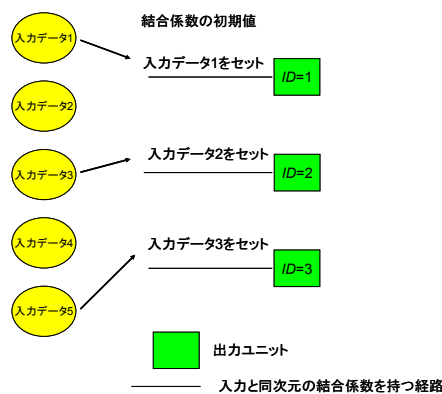


図7 結合係数の初期化

step2の入力ユニットの決定はCTに従って行う。

step3の勝者ユニットと近傍ユニットの決定では、勝者ユニットを決定し、その近傍を求めることで実現する。勝者ユニットは、入力ユニットと接続されている経路のうち、入力データと結合係数のユークリッド距離が最も近い経路で接続される出力ユニットである。近傍ユニットは、勝者ユニットの近傍領域に位置する出力ユニットとする。

step4の結合係数の更新は、結合係数を入力データ

に近づけることによって行う。どの程度結合係数を更新するのかを与える学習比を定義し、更新は学習比に従って行う。学習比 α_E は世代Eによって決定する値で、1世代毎に減少し、学習終了世代で0となるよう設計する。

学習比の初期値を α_0 、学習終了世代を E_{max} としたとき、世代 E_{now} での学習比 $\alpha_{E_{now}}$ は式(8)によって求める。

$$\alpha_{E_{now}} = \alpha_0 - \left(\frac{\alpha_0}{E_{max}} \right) \times E_{now} \quad (8)$$

結合係数の更新は、現在の結合係数に、更新ベクトルを加えることで行う。更新ベクトルは勝者ユニットの結合係数を更新する場合と、近傍ユニットの結合係数を更新する場合で異なるものを用いる。勝者ユニットがより大きく入力データに近づくように、勝者ユニットと近傍ユニットの学習量の比 β ($0 < \beta < 1$)を設定する。学習に用いる入力データのベクトルを \mathbf{x} 、更新する結合係数のベクトルを \mathbf{w} とすると、勝者ユニットの更新ベクトル \mathbf{D}_w 、近傍ユニットの更新ベクトル \mathbf{D}_n を式(9)(10)で表す。

$$\mathbf{D}_w = \alpha_{E_{now}} \times (\mathbf{x} - \mathbf{w}) \quad (9)$$

$$\mathbf{D}_n = \beta \times \alpha_{E_{now}} \times (\mathbf{x} - \mathbf{w}) \quad (10)$$

5. 実験と考察

本手法の有効性を確認するため、「西馬内がんけ」のデータを用いて分割実験を行った。データは1365フレームからなり、600フレーム前後で繰り返し同じような動作を行う構造になっている。

本実験では、累積寄与率が0.8以上となるように主成分分析による圧縮を行った。SOMでは出力層のニューロン数を10とし、10世代の学習を行った。学習比の初期値 α_0 は0.02、勝者ユニットと近傍ユニットの学習量の比 β は0.01とした。

学習されたネットワークに対し、入力データを入力し、順方向学習と同様の手順で勝者ユニットを決定した。同一の出力ユニットが発火した入力データを同一の舞踏符として考え、分割点を求めた。

実験は、MacPro (CPU: Dual-Core Intel Xeon 2.66 GHz, メモリ: 1GB, OS: Mac OSX10.4.10) を用い、この上で動作するMATLAB R2007aのNeural Network Toolbox環境によって行った。

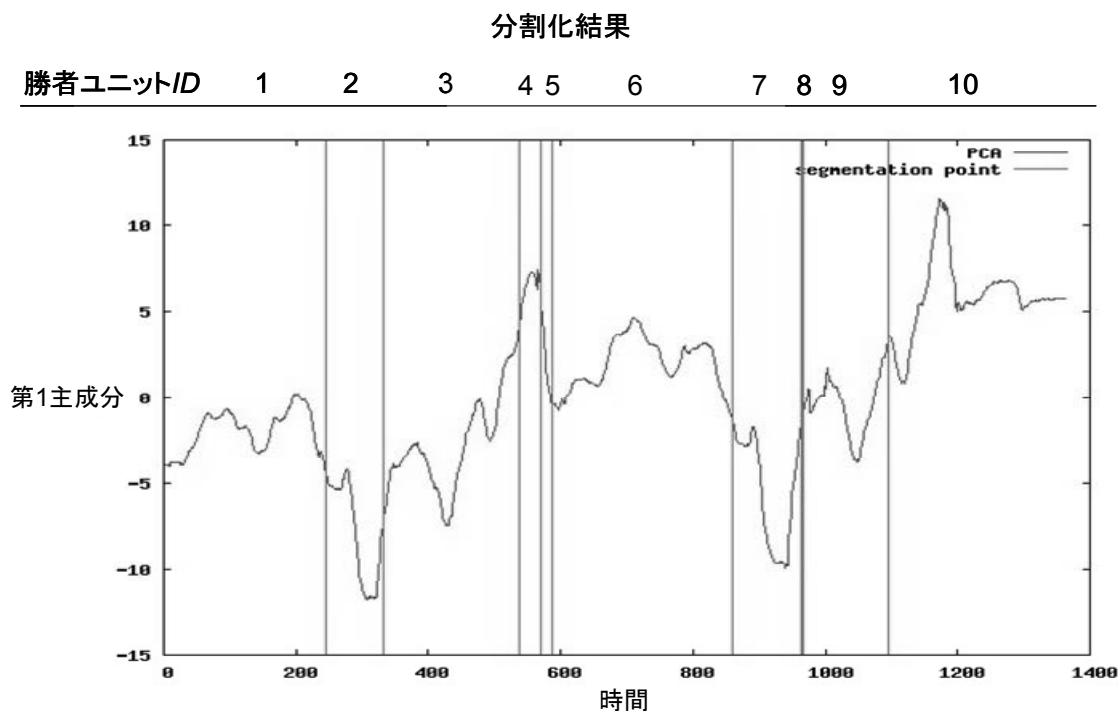


図8 第1主成分と分割化結果

主成分分析による圧縮の結果，次元数を 10 次元に抑えることができた．図 8 に提案手法による分割結果と動作の第 1 主成分を示す．図 8 は，横軸に時間，縦軸に第 1 主成分の値をとっており，図中の垂線が分割点となる．また，図の上部には分割された各区間に対応するよう，SOM の勝者ユニットの ID を表示する．

勝者ユニットの ID は昇順に並んでおり，入力データの時間順を保存していることがわかる．また，分割された区間ごとの第 1 主成分を目視により比較すると，ID が 1 と 6，2 と 7 のように，繰り返しの同一のパターンが正しく切り出されていることがわかる．これは，基本動作の出現時刻が異なった場合においても，類似の区間として正しく分割ができることを示している．従って，基本動作の切り出しにおいて，提案手法が有効であると考えられる．

一方で，ID=4 の区間は 31 フレーム，ID=5 の区間は 20 フレーム，ID=8 の区間は 1 フレームと短い区間で区切れてしまっている区間も存在する．このような区間は，動作データの再利用を考えた場合に適切な基本動作とは言えないため，隣接するクラスに統合するなどの後処理が必要と考えられる．

今後の検討すべき点として，まず，パラメータの決定方法があげられる．本実験では，すべてのパラメータについて手動で設定を行っている．各パラメ

ータに関する詳細な実験と検討が必要である．

次に，データの圧縮手法に関する詳細な検討も重要と考える．SOM は入力データに従って結合係数を更新するため，主成分分析に変わり，動作の特徴をよく保存した圧縮方法が設計できれば，よりよい分割化結果が得られると考えられる．

6. おわりに

本報告では，動作データの再利用に有効な舞踊符を自動で生成するために，動作を自動で基本動作に分割する手法について述べた．

基本動作は類似の姿勢が連続する区間であり，姿勢をクラスタリングすることによって，基本動作に分割できると考え，クラスタリング手法として SOM に着目した．SOM では入力データの時間順は考慮されていないため，時系列データのクラスタリングには向かない．そこで，時間順を考慮するように SOM を改良することで，姿勢のクラスタリングを実現した．この手法により，姿勢の時系列データである MoCap によって取得した動作データを姿勢によってクラスタリングし，基本動作に分割できる可能性について確認した．

今後の課題として，パラメータの決定方法の検討，データの圧縮手法の改良，SOM の出力層の構造の拡張に関する検討などがあげられる．

今後の展開として、基本動作に分割された区間を適切に統合し、舞踊符に名前を割り当て、実際に舞踊符を自動で生成する手法の開発が必要である。さらに、舞踊符の定量的な評価方法の決定、実験などが必要となる。

謝辞

本研究の一部は、平成 18、19 年度科学研究費補助金基盤研究 (C) (課題番号 18500071) の支援によって実施した。

参考文献

- (1) 海賀孝明, 湯川崇, 長瀬一男, 佐々木信也, 玉本英夫 (1999): 舞踊符による動作の記述法の提案, NICOGRAPH / MULTIMEDIA 論文コンテスト 論文集 (CD-ROM 版) .
- (2) 湯川崇, 海賀孝明, 長瀬一男, 玉本英夫 (2000): 舞踊符による身体動作記述システム, 情報処理学会論文誌, 41 巻, 10 号, 2873-2880 頁.
- (3) Daiki Kawanaka, Shun Ushida, Takayuki Okatani, Koichiro Deguchi (2005): HHMM Based Recognition of Human Activity from Motion Trajectories in Image Sequences, *proceedings MVA2005 IAPR Conference on Machine Vision Applications (Tsukuba, Japan)* 14-3, pp. 578-581.
- (4) 呉海元, 木戸隆太郎, 塩山忠義 (2001): ジェスチャ認識のための連続 DP の改良, 「インタラクションによる感性的コミュニケーション総合特集号-I」論文, システム制御情報学会論文誌, Vol.14 No.6, 283-290 頁
- (5) 浜田玲子, 坂井修一, 田中英彦, 佐藤真一 (2001): 料理映像における繰り返し動作のスポッティング手法, 信学技報, PRMU, Vol.2001, No. 46, 77-82 頁.
- (6) 湯川崇, 小原直子, 玉本英夫 (2002): 動作プリミティブ抽出と舞踊符割当ての自動化. 秋田大学工学資源学部研究報告, 23 号, 33-40 頁.
- (7) 湯川崇, 小原直子, 玉本英夫 (2004): モーションキャプチャからの基本動作分割点の自動検出. 情報処理学会論文誌, 45 巻, 4 号, 1198-1201 頁.
- (8) Teuvo Kohonen (1990): The Self-Organizing Map, *proceedings of the IEEE*, Vol. 78, No. 9, pp. 1464-1480.
- (9) 芝祐順 (初版 1975 年): 行動科学における相関分析法 (第 2 版), 東京大学出版会, ISBN 4-13-042056-9