Signal Processing: *Image Communication* 42 (2016) 109–119

Contents lists available at ScienceDirect

Signal Processing: *Image Communication*

journal homepage: www.elsevier.com/locate/image

Profile HMMs for skeleton-based human action recognition

Wenwen Ding [a,b], Kai Liu [a,*], Xujia Fu [b], Fei Cheng [a]

[a] School of Computer Science and Technology, Xidian University, Xi'an, China
[b] School of Mathematical Sciences, Huaibei Normal University, Anhui, China

ARTICLE INFO

Article history:
Received 11 April 2015
Received in revised form
17 January 2016
Accepted 21 January 2016
Available online 29 January 2016

Keywords:
View-invariant representation
Skeleton joints
Human activity recognition
Profile HMMs
Self-organizing map

ABSTRACT

In this paper, a novel approach based Hidden Markov Models (HMMs) approach is proposed for human action recognition using 3D positions of body joints. Unlike existing works, this paper addresses the challenging problem of spatio-temporal alignment of human actions which come from intra-class variability and inter-class similarity of actions. The first and foremost actions are segmented into meaningful action-units called dynamic instants and intervals by using motion velocities, the direction of motion, and the curvatures of 3D trajectories. Then action-units with its spatio-temporal feature sets are clustered using unsupervised learning, like Self-Organizing Mapping (SOM), to generate a sequence of discrete symbols. To overcome an abrupt change or an abnormal in its gesticulation between different appearances of the same kind of action, profile HMMs are applied with these symbol sequences using Viterbi and Baum–Welch algorithms for human activity recognition. The effectiveness of the proposed method is evaluated on three challenging 3D action datasets captured by commodity depth cameras. The experimental evaluations show that the proposed approach achieves promising results compared to other state-of-the-art algorithms.

1. Introduction

Recognizing human activity is a key component in many applications, such as Video Surveillance, Ambient Intelligence, Human–Computer Interaction systems, and even Health-Care. There are several major difficulties to vision based human action recognition, such as low level challenges: occlusions, shadows, illumination conditions, view changes, and scale variances [1]. With the development of the commodity depth sensors like Microsoft Kinect [2], we can easily access to the 3D data as a complement to the traditional RGB imagery. Shotton et al. [3] proposed a method to extract 3D body joint locations from a single depth image from Kinect.

The introduction of 3D data largely alleviates these difficulties by providing the structure information of the scene, slightly rotated view and true 3D dimension of the subject, etc. But the challenge of intra-class variability and inter-class similarities of actions is still a hard problem for algorithms using various types of data. Intra-class variability is that same action completed with different subjects is variant according to people's habit and comprehension to this action. For example, as like *waved-goodbye* action, some person wave his hand over the head and others only wave in front of the chest. Some person wave his hand only once and others wave his hand twice and even more. The inter-class similarity is that only very subtle spatio-temporal features can be acquired to distinguish different actions. For example, the actions of *drink water* and *have a phone* will be regarded as same action. These variabilities

* Corresponding author. Tel.: +86 13892810532.
E-mail addresses: dww2048@163.com (W. Ding), kailiu@mail.xidian.edu.cn (K. Liu), fxjmsy@aliyun.com (X. Fu), chengfei8582@163.com (F. Cheng).

http://dx.doi.org/10.1016/j.image.2016.01.010
0923-5965/

and similarities have seriously affected the accuracy of action recognition.

The objective of this paper is to build a model that can extract the principal characters of each action which solve the problem of intra-class variability and inter-class similarity of actions. We take advantage of profile analysis [4] based on HMMs [5] theory and hope to find the structure information of action sequences. Profile analysis is a sequence comparison method for aligning action sequences and identifying new sequences with known action. Basically, a profile is a description of the consensus of a multiple sequence alignment. HMMs are probabilistic models that are generally applicable to time series of linear sequences. Different from general statistical models, HMMs need to construct a suitable finite state automaton as the topology of its hidden states according to the prior knowledge. Due to the uncertainty of the topology of its hidden states, the structure information of action is not obvious. Profile HMMs [6] are strong linear state machines consisting of a series of nodes, each of which corresponds roughly to a position (column) in the alignment. The linear state machines are statistical models which match given sequences to aligned sequence families. In this paper, we show that profile HMMs can have the ability to align multiple human action sequences. A profile can be obtained by the series of nodes that the structure information of action is obvious. After the structure information of action sequences are obtained, the variability and similarity of action sequences to an existing profile can be easily tested.

Our overall approach is sketched in Fig. 1. First, trajectories of action, also referred to as discrete curves, can be drawn by several 3D joint points. The segmentation points S, splitting actions into meaningful action-units, can be captured by the direction of motion and curvature of the trajectory with maximum velocity. These newly obtained segmentation points are also able to determine the start-frame and end-frame of an action, and eliminate noise to a certain degree. Then, the features of action units, consisting of dynamic instants (postures) $\boldsymbol{\xi}_p$ and intervals (actionlets[1]) $\boldsymbol{\xi}_a$, are extracted from these segmented trajectories and then are mapped into two Self-Organizing Mappings (SOMs) [7] recorded as T_{ξ_p} and T_{ξ_a}, respectively. Thus, numerically similar adjacent features can be mapped to a single representative vector (model vector) $\boldsymbol{m}$ on a SOM, which itself is a form of clustering process with unsupervised learning. Unlike actions have been labeled such as *High Wave*, *Draw X* etc, postures and actionlets will not be labeled so easily. Therefore, T_{ξ_p} and T_{ξ_a} can be divided into chunks according to the Davies–Bouldin Index (DBI) value [8] which decide the clustering boundaries in T_{ξ_p} and T_{ξ_a}. These chunks in SOM can be named with upper-case and lower-case letters respectively referred as the labels of postures and actionlets. Finally, capturing the spatio-temporal relationships between action-units of given actions, profile HMMs are generated by sequences of discrete symbols of each action. With these profile HMMs, actions are trained and aligned.

The use of profile HMMs is especially appealing for human action recognition for a few reasons.

First, unlike most other sequence alignment techniques, a profile HMM can generate an alignment for a large number of sequences of same action without first calculating all pairwise alignments. For our application, this is particularly important as it means that we can train a profile HMM on some trajectories of the same action and then generate a profile for same action without human assistance.

Second, a profile HMM can be trained so that it is most likely to generate a symbol pattern for its action category and also possesses traits involving insertions and deletions to align many abrupt, abnormal, or period action-units.

Third, standard dynamic programming algorithms, called Forward (for scoring) and Viterbi (for alignment), are applied on classical HMMs. Using the Forward algorithm, we can calculate the all probabilities of a sequence being generated by profile HMMs, i.e., can be used to classify unknown sequences for which model. Using the Viterbi algorithm, we can reckon the most likely path through profile HMMs that generates a sequence, i.e. the most likely alignment of the sequence against the model. But these algorithms have a worst-case algorithmic complexity of $O(NM^2)$ in time and $O(NM)$ in space for a sequence of length N and an HMM of M states. For profile HMMs that have a constant number of state transitions per state rather than the vector of M transitions per state in fully connected HMMs, both algorithms run in $O(NM)$ in time and $O(NM)$ space [6].

The last attractive characteristic is that a profile HMM can be trained so that it is most likely to generate a symbol pattern for its action category. As shown in Fig. 8b, the structure of the action sequences can be represented as symbol sequence *DaEkEdEtD* because the consensus sequence character corresponds to the highest value in the row. Therefore the structure of the action sequences can be easily recognized within longer activity sequences.

The rest of the paper is organized as follows: Section 2 presents the related work; Section 3 elaborates our method of features extraction, clustering of action-units and aligning multiple actions; Section 4 discusses the parameters setting and presents our experimental results; and Section 5 concludes this paper.

2. Related work

Action recognition: In the past decades, video-based action recognition has a great number of literatures [9–11]. Spatial Temporal Interest Points (STIP) [12] are proposed to represent an action by extracting dense local features of Histogram of Oriented Gradients (HOG) [13] and Histogram of Optical Flows (HOF) [14]. Bag-of-words (BoW) [15] representation has been popularly applied with spatio-temporal local feature based approaches.

Recently, with the development of the commodity depth sensors like Microsoft Kinect [2], there has been a lot of interests in human action recognition from depth

[1] We use actionlet to refer to meaningful atomic actions obtained from spatio-temporal decomposition.

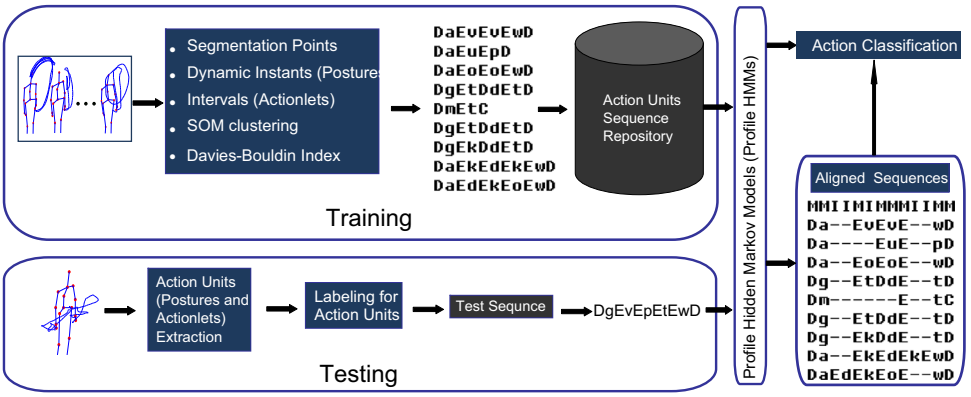

Fig. 1. The general framework of the proposed approach.

data. By using the Kinect sensor, the information with RGB, depth, and skeleton joint positions can be obtained. Several researches utilize depth maps as features for action recognition. Wang et al. [16] extracted semi-local features in the 4D space of a depth sequence. Xia and Aggarwal [17] extracted the features of STIP from depth videos, and developed a novel depth cuboid similarity feature to describe the local 3D depth cuboid and to repress the noises. Instead of relying on depth maps, many researchers utilize skeleton joint positions as features for action recognition. Li et al. [18] employed a bag-of-3D-points graph approach to encode actions based on 3D projection of body silhouette points. Xia et al. [19] mapped 3D skeletal joints to a spherical coordinate system and used a histogram of 3D Joint Locations to achieve view-invariant posture representation. The joints were then translated to a spherical coordinate system to achieve view-invariance. A Hidden Markov Model is used for action classification. Similarly, Miranda et al. [20] described each pose in a spherical angular representation and the SVM classifier to identify key poses. The action is represented as a sequence of key poses and recognized by a decision forest. Yang and Tian [21] developed the EigenJoints features from RGBD sequences, which combine multiple aspects of action information including human postures in each frame, motion information. But these current algorithm only works well when the human subject is in an upright position facing the camera. The human body viewed partly will lead to that the results of experiment are not very reliable. Wang et al. [22] utilize skeleton joints to define action as the interactions that occur between subsets of these joints. An actionlet mining technique is used to represent an action as an actionlet ensemble.

Spatio-temporal alignment: Given two human action sequences, an important question is to consider whether those two sequences represent the same action or different actions. This can be viewed as a spatio-temporal alignment problem. Spatio-temporal alignment of human actions has been a hot topic of particular interest. HMM and Dynamic Time Warping (DTW) are two main methods for this problem based on temporal sequential representation. HMM is usually used to resolve the problem from the point of view of probabilistic. In [23], by using the Viterbi algorithm, each action was modeled as a series of 2D human poses. Mapping poses or frames into symbols is the main challenge of HMM approaches. But these frame by frame representations suffer from redundancy. Furthermore, HMM structure must be designed for domain specific application. DTW is an algorithm for measuring similarity between two temporal sequences which may vary in time or speed. Ukrainitz and Irani [24] designed a consolidated method to the problem of temporal sequence alignment for a wide range of scenarios, while the temporal warping is restricted by 1D affine transformation. However, DTW is likely to be affected by the noise and periodic motions, thus degrading classification performance. Unlike most other sequence alignment techniques, profile HMMs can generate an alignment for a large number of sequences of the same action without first calculating all pairwise alignments and also possess traits involving insertions and deletions to align many abrupt, abnormal, or period action-units. Though received little attention in the computer vision field, profile HMMs have been extensively studied in many other fields. Fischer et al. [25] employed profile HMMs to learn amino acid sequences and then used to search on the six-frame translation of nucleotide sequences. Wright et al. [26] used profile HMMs to identify unknown TCP connection in wide-area Internet traffic. Bhargava and Kondrak [27] adapted profile HMMs to the task of aligning multiple words and sets of multilingual cognates and show that they produce good alignments.

3. Proposed method

3.1. Representation of meaningful action units

We address the problem of modeling and analyzing human actions in the joint trajectories space. Action is represented as a sequence of dynamic instants and intervals, which are computed using the direction of motion and the spatio-temporal curvature of 3D trajectories. It uses depth cameras to track 3D trajectories that each trajectory represents the evolution of one coordinate x, y, or z over time for indicating the position of a specific joint of

human. Motion trajectories provide rich spatio-temporal information about an object's behavior.

To obtain meaningful action-units, we must learn superior segmentation points $\mathcal{S}=\{s_1,\ldots s_i,\ldots,s_j,\ldots,s_m\}$ $(1<i<j<m)$ to segment 3D trajectory of an action, as shown in Fig. 2a. The problems of under-segmented and over-segmented trajectories will always lead to insignificant action units. Based on the previous studies [28], superior segmentation points S for trajectories of an action can be obtained by using the direction of motion (like up, down, left, right, further, closer) and curvature of the trajectory. The direction of motion contains some zero crossing points, which easily yield over-segmented trajectories. In order to avoid the problem of over-segmented, a three times spline function is introduced to smooth curve in 3D space. Thus, the possible segmentation points are detected according to the value of curvature, as this point is closer to the hilltop, whereas the value of the direction of motion at this point is zero crossing. Segmentation points can also determine the start-frame s_1 and end-frame s_m of an action and eliminate noise to a certain degree in an action.

For dynamic instants of action, we can utilize human postures to represent in this moment. Human postures can be represented by relative distances d and angles θ from 3D star skeleton, as shown in Fig. 2b. For intervals of action, we can utilize actionlets to represent these intervals. An outline of the construction process to obtain human postures and actionlets from trajectories is shown in Algorithms 1 and 2, respectively.

Algorithm 1. Extracting feature of postures $\mathcal{F}_p$ using segmentation points.

Input: Joint point trajectories R and segmentation points $\mathcal{S}=\{s_1,\ldots s_i,\ldots,s_j,\ldots,s_m\}(1<i<j<m)$
Output: features set of postures $\mathcal{F}_p$
1: set $\mathcal{F}_p=[\,]$;
2: **for** frame t=s_1 to s_m **do**
3: Compute and store distances between *joint* and the body hip centroid joint $(d^t_{head}, d^t_{leftHand}, d^t_{rightHand}, d^t_{leftFeet}, d^t_{rightFeet})$;
4: Compute and store angles between two adjacent body extremities $(\theta^t_{head}, \theta^t_{leftHand}, \theta^t_{rightHand}, \theta^t_{leftFeet}, \theta^t_{rightFeet})$;
5: Obtain feature of posture $\boldsymbol{\xi}^t_p$ concatenating d and θ into a one-dimensional vector;
6: **end for**
7: the normalization is applied to $\mathcal{F}_p=[\boldsymbol{\xi}^{s_1}_p\ \boldsymbol{\xi}^{s_2}_p\ \ldots\ \boldsymbol{\xi}^{s_m}_p]$;

Algorithm 2. Extracting feature of actionlets $\mathcal{F}_a$ using segmentation points.

Input: Joint point trajectories R and segmentation points $\mathcal{S}=\{s_1,\ldots s_i,\ldots,s_j,\ldots,s_m\}(1<i<j<m)$
Output: features set of actionlets $\mathcal{F}_a$
1: set $\mathcal{A}=\{a_1,a_2,\ldots,a_i,\ldots,a_{m-1}\}=\{\langle s_1,s_2\rangle,\langle s_2,s_3\rangle,\ldots,\langle s_i,s_i+1\rangle,\ldots,\langle s_{m-1},s_m\rangle\}$, where $a_i=\langle s_i,s_{i+1}\rangle$ indicates that an actionlet a starts from frame s_i and will finish at frame s_{i+1};
2: set $\mathcal{F}_a=[\,]$;
3: **for** i=1 to $m-1$ **do**
4: **for** t=s_i to s_{i-1} **do**
5: Compute and store invariant value v_t, Δh_t, $\Delta\phi_t$, $\Delta\theta_t$, k_t of joint point at frame t in one actionlet, where v_t is the motion velocity; Δh_t, $\Delta\phi_t$, $\Delta\theta_t$ are the directions of up-down, left-right and further-closer information; k_t is the curvature;
6: Obtain feature of actionlet $\boldsymbol{\xi}^i_a$ concatenating $v_t, \Delta h_t, \Delta\phi_t, \Delta\theta_t, k_t, d_t$ and d_t into a one-dimensional vector;
7: **end for**
8: **end for**
9: the normalization is applied to $\mathcal{F}_a=[\boldsymbol{\xi}^1_a\ \boldsymbol{\xi}^2_a\ \ldots\ \boldsymbol{\xi}^{m-1}_a]$;

3.2. Clustering feature using unsupervised learning

Unlike action labels are easily labeled in real life, such as walk, sit down, stand up, and throw, an actionlet or a posture is hardly labeled or highly generalized using our human language. Therefore, SOM and DBI are used to cluster postures and actionlets.

The SOM is an unsupervised neural network learning algorithm and project complex nonlinear high-dimensional data to two-dimensional space. The sizes of SOM can be determined according to the number of training samples. It should be noted that the larger the size of SOM, the more the over-fitting phenomenon occurs, and conversely, the smaller the size of SOM, the more the under-fitting phenomenon occurs. The features of postures ξ_p and actionlet ξ_a map to SOM forming similar neural units in T_{ξ_p} and T_{ξ_a} need to be clustered and labeled later. But it is difficult to find clustering boundaries from the mapping result of SOM. According to this limitation, we can use the DBI value to find the clustering boundaries. Therefore, we use SOM to produce the prototypes, and then cluster these prototypes.

The DBI is defined as the ratio of S_c and d_{ce} and the best clustering minimizes

$$DBI=\frac{1}{C}\sum_{k=1}^{c}\max_{l\neq k}\left\{\frac{S_c(Q_k)+S_c(Q_l)}{d_{ce}(Q_k,Q_l)}\right\}, \tag{1}$$

where C is the number of clusters, $\{Q_i|i=1,\ldots,C\}$ is a set of clusters, $S_c=\frac{\sum_i\|x_i-c_k\|}{N_k}$ is the within-cluster distance, $d_{ce}=\|c_k-c_l\|$ is the between-clusters distance, $x_i\in Q_i$, N_k is the number of samples in cluster Q and $c_k=\frac{1}{N_k}\sum_{x_i\in Q_k}x_i$. By definition, the lower the *DBI*, the better the separation of the clusters and the tightness inside the clusters.

The number of chunks of the T_{ξ_p} and T_{ξ_a} can be decided by the lowest *DBI* value. As mentioned previously, the features of postures and actionlet are mapped onto T_{ξ_p} and T_{ξ_a}, respectively. Then, the T_{ξ_p} and T_{ξ_a} are divided by DBI value. Each chunk in T_{ξ_p} is symbolized by upper-case letters

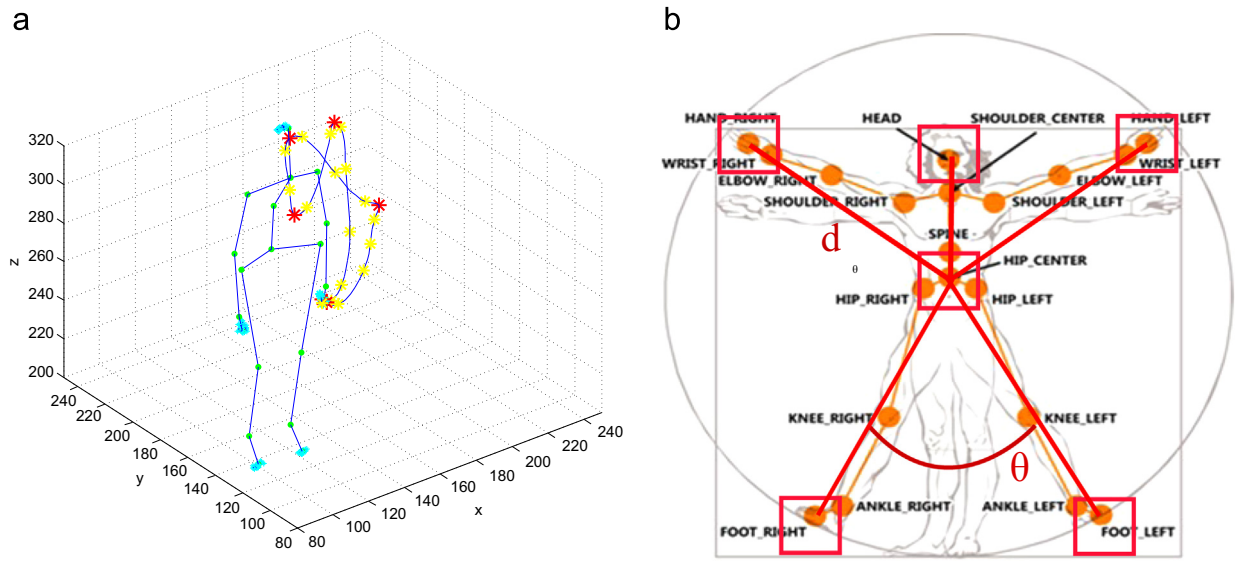

Fig. 2. (a) The trajectory of an action of *high hand wave* is segmented by *red stars*. (b) Illustration of human posture representation based on relative distance d and angles of star skeleton θ.

according to posture and each chunk in T_{ξ_a} is symbolized lower-case letters according to actionlet, as shown in Fig. 3. Therefore, the feature of postures ξ_p and actionlets ξ_a were transformed into symbols from a discrete alphabet so that an action can be represented by upper-case and lower-case letters generated alternately in a individual sequence, for example, *DaEvEvEwD*. Sequences of the same kind of action will correspond a sequence family to generate a profile HMM. It should be noted that the 26 upper-case and 26 low-case letters are not enough to nominate the actionlets and postures with a complex action.

3.3. Profile HMMs for spatio-temporal alignment of human action

3.3.1. Hidden Markov models

A first-order discrete HMM $\Lambda = \{A, B, \pi\}$ is described by the following notations:

Q = the set of states = $\{q_1, q_2, \ldots, q_n\}$.
V = the output symbols = $\{v_1, v_2, \ldots, v_m\}$.
$\mathbf{A}_{n\times n}$ = state transition probabilities = $\{a_{ij} | a_{ij} = Pr(s_{t+1} = q_j | s_t = q_i)\}$, where state s_t is the tth state.
$\mathbf{B}_{n\times m}$ = symbol output probabilities = $\{b_j(k) | b_j(k) = Pr(v_k | s_t = q_j)\}$.
π = initial state probability = $\{\pi_i | \pi_i = Pr(s_1 = q_i)\}$.
$O = (O_1, O_2, \ldots, O_T)$: Observed symbol sequence (length = T)

Fig. 4a shows a simple HMM. The probability of the observation symbol sequence can be calculated as the product of the state transitions and the symbol emissions. But the HMM transition states cannot be observed: it is hidden. Only the symbol sequence that these hidden states emit can be observed. Therefore, given a class of action training sequences, a HMM must be trained to determine the model parameters $\{A, B, \pi\}$. The probability of the observation symbol sequence can be acquired from the HMM by using the probability $Pr(O|\Lambda)$.

3.3.2. Profile hidden Markov models

Symbol sequences of same kind of action are large-scale similar sequences. We want to discover the structure of action in these data to carry out pattern recognition. For example, the *high arm wave* action of a subject may has six action-units *abcbcd*: lifting hand above head *a*, waving hand towards left *b*, waving hand towards right *c*, waving hand towards left and right again *bc*, and final laying down hand *d*. Different subjects do this action may have different action-units: *abcd*, *abcbcbcbcd* or others. So what is this structure in the end for most subjects? Suppose that six position-specific column as shown in Fig. 4b is selected as the structure of "high arm wave" action which model the distribution of action-units allowed in the column. Symbol sequences of action will align this structure with match, insertion and deletion states. Thus profile HMMs for detecting position-specific information

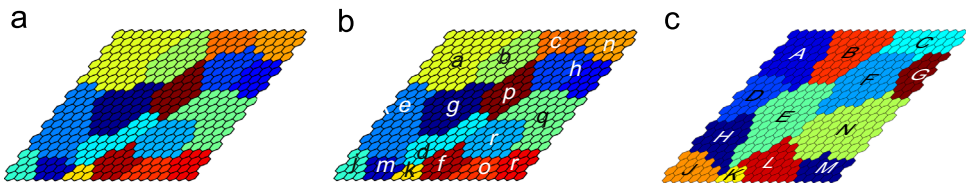

Fig. 3. (a) a SOM is clustered to organize the model vectors into "natural groups". (b) Different clusters in T_{ξ_a} are represented by different lower-case letters. (c) Different clusters in T_{ξ_p} are represented by different upper-case letters.

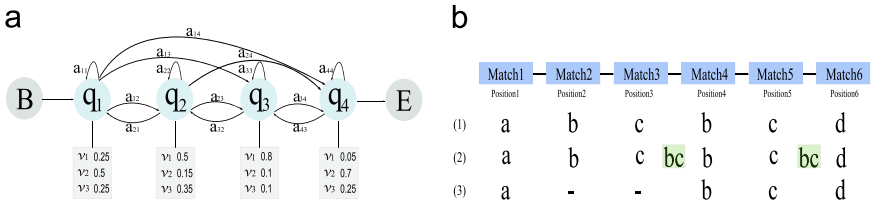

Fig. 4. (a) A HMM has four-state q_1, q_2, q_3, q_4 and three-symbol v_1, v_2, v_3 to describe action-units sequence. There are 3×4 symbol output probabilities and 4×4 directed lines which are transitions from one state to another. (b) The structure of "high arm wave" action can be depicted as a six position-specific column. The first case, *abcbcd*, perfectly matches with this structure. The second case, *abcbcbcbcd*, matches the structure by using insertion *bc* after columns 3 and 5, respectively. The third case, *abcd*, matches the structure by using deletion columns 2 and 3. That is to say, columns 2 and 3 are omitted from the sequence and do not match anything in the above model.

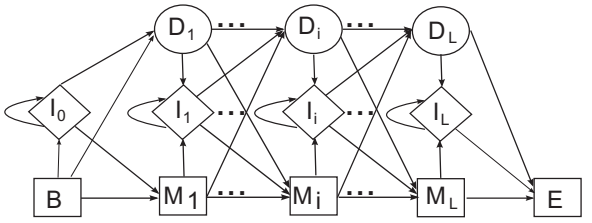

Fig. 5. A general profile HMM of length L. M_i is the ith match state, I_i is the ith insert state, Di is the ith delete state. B is the begin state, and E is the end state.

from multiple sequence comparison are introduced for this purpose.

Profile HMMs can be imagined as a series of consensus columns as shown in Fig. 5 which consist of three types of states: match states M_i, insert states I_i, and delete states D_i in each column i. Any arbitrary sequence can be described as a states traversal from column i to column $i+1$. Each state can probabilistically transit to the next states. Transition probabilities a_{ij} are the probability from one state i to next state j, such as $a_{M_iI_i}$, $a_{I_iI_i}$, and $a_{D_iD_{i+1}}$.

The bottom line of square shaped states is main states referred to as the match states M_i, which form the kernel of the model. Each match state M_i is represented by a set of emission probabilities $e_i(a)$ indicating that the distribution of values for a given position i emits symbol a in the output alphabets $\sum$.

The second row of diamond shaped states is called the insert states I_i, which are portions of sequences that do not match anything in the profile HMMs. These states are applied to construct variable regions that can be inserted at a given position in a sequence. Insert states I_i are used to account for symbols that can been inserted after ith column in our alignment.

The top line of circular states are called delete D_i or silent states, which represent symbols that have been removed from a given position as well as gaps in a sequence. For a sequence to use a delete state for a given position indicates that a given character position in the model has no emitting any symbols in the given sequence.

Given a profile HMM, how to align multiple sequences based on the model is the first problem to solve. Viterbi algorithm is used for seeking the most likely path of each sequence generated by the model. Multiple sequence alignment is to find Viterbi path of each sequence. Given a small example of a set of human posture sequences as shown in Fig. 6a, the output alphabets $\sum = \{F, K, L, N, Q, S, T, W, Y\}$ have different probability distributions in each column as shown in Fig. 6c. These distributions also match the structure of the sequences family as shown in Fig. 6b. Both of them reflect the normal structure of the sequences family.

3.3.3. *Adapting profile HMMs for human action recognition*

In this section we describe the structure of our profile HMMs as shown in Fig. 7. The main difference between our profile HMMs and others is that the profile HMMs used in biology have only a single chain of Match states. In our case, the addition of a second match state per position is intended to allow the model to represent the correlation between action units in videos. In the context of human action recognition, actions are segmented into meaningful action-units: postures and actionlets. The labels of postures and actionlets are upper-case and lower-case letters respectively. Therefore, an action can be represented by a string, for example, *DaEvEvEwD*. Pay attention to the first and the end symbol is upper-case letters meaning that an action begin or end with a posture in our observation. This is necessary as postures and actionlets obviously alternated in an action. To allow for variations between the observed action-units in the same action sequences, the model has two additional states for each position in the chain. One is insert states I_i representing one or more extra abrupt or abnormal action-units inserted in a sequence

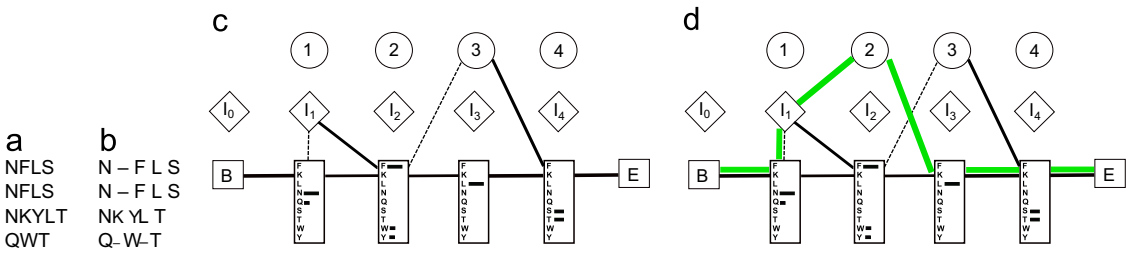

Fig. 6. (a) A sequences family of human posture. (b) The structure of the sequences family of human posture. (c) The numbers in the circular delete states are position numbers. Solid lines with no arrow head are transitions from left to right. The thickness of solid line indicates that the value of transition probabilities a_{ij} are big or small. Transitions with probability zero are not appeared. Dashed lines are very small probabilities. Transitions from an insert state to itself are not shown. The length of line in square match states indicates the value of emission probabilities $e_i(a)$. (d) The green path is the very likely Viterbi path corresponding to sequence *NSLS*.

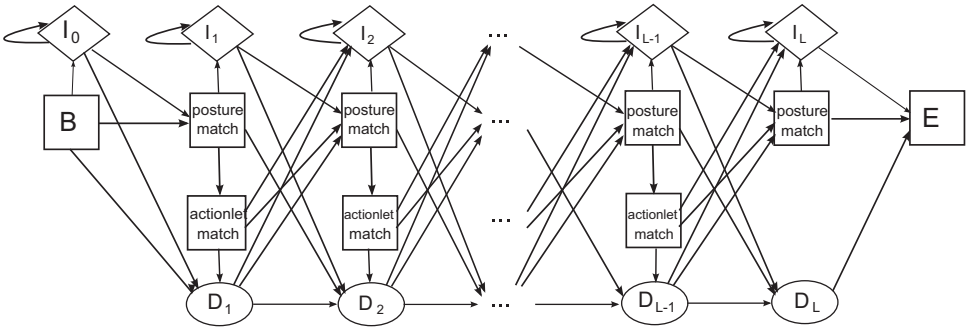

Fig. 7. The profile HMMs for human action recognition.

between two normal parts of the chain. The other is Delete states D_i allowing period action-units to be omitted from the action sequences.

The classifiers we build for human action recognition are based on our profile HMMs. Using the Forward–Backward algorithm [29], we can calculate the all probabilities of a sequence being generated by profile HMMs, i.e., can be used to classify unknown sequences as which model. Using the Viterbi algorithm [30], we can reckon the most likely path through profile HMMs that generates a sequence as shown in Fig. 6d, i.e., the most likely alignment of the sequence against the model. Using initial parameters that assign uniform probabilities over all action units in each time step, we employ the Baum–Welch algorithm [31] to iteratively find new parameters which maximize the likelihood of the model for the sequences of action units in the training videos.

We now explain the design and use of profile HMMs Λ of k classes with models $\Lambda_1, \Lambda_2, \ldots, \Lambda_k$ which employ to capture characteristics exhibited by each kind of actions. If we already have a set of action-unit sequences (Fig. 8a) belonging to a family, a profile HMM $\Lambda_c (1 < c < k)$ can be constructed from the set of unaligned sequences after using the Baum–Welch algorithm. The length L of the $\Lambda_c (1 < c < k)$ must be chosen, and is usually equal to the average length of the unaligned action unit sequences in the training set. The transition and emission probabilities are initialized from Dirichlet distributions.

Once profile HMMs Λ have been obtained, a classifier $\mathcal{C}_1$ can be constructed for the task of choosing the best

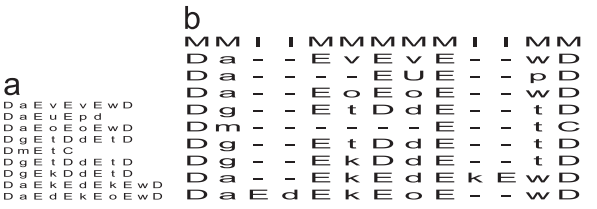

Fig. 8. (a) A set of action-unit sequences of action *high arm wave*. (b) The alignment generated via the profile HMM method for the set of action-unit sequences of action *high arm wave*. The match and insert columns are marked with the letters *M* and *I* respectively in the first line.

model $\Lambda_c (1 < c < k)$ for new test sequence q

$$c = C_1(q) = \arg\max_c P(q|\Lambda_c). \tag{2}$$

This is done via a straightforward application of the Forward–Backward algorithm to get the full probability of the given sequence q.

The second classifier $\mathcal{C}_2$ uses of the well-known Viterbi algorithm for finding the most likely alignment of the sequence to the family, i.e., Viterbi path V. For a given output sequence q and the associated probability of the most likely Viterbi path V_c to each profile HMM, the Viterbi classifier $\mathcal{C}_2$ finds Viterbi paths for the sequence in each profile HMM $\Lambda_1, \Lambda_2, \ldots, \Lambda_k$ and chooses the class c whose model produces the best Viterbi path V_c:

$$c = C_2(q) = \arg\max_c P_{viterbi}(q, \Lambda) = \max_{V_c} P(q, V|\Lambda). \tag{3}$$

In practical terms, the Viterbi classifier $\mathcal{C}_2$ finds each model's best explanation of the action-units generation in

the sequence. We choose the Viterbi classifier $\mathcal{C}_2$ that provides the best explanation for the observed action-units for experimental evaluation.

4. Experimental evaluation

The performance of the activity recognition is primarily evaluated based on its accuracy. In this section, we performed this evaluation on three different datasets: MSR-Action3D [18], UTKinect-Action [19], and UCF Kinect Dataset [32]. In all experiments, we used the Viterbi classifier $\mathcal{C}_2$.

4.1. RGB-D action dataset

The MSR Action3D Dataset from Li et al. [18] was captured using a depth camera similar to Kinect and 3D joint positions extracted from the depth sequence. Due to severely corrupted skeleton data in some of the action sequences available, we selected a subset of 557 sequences out of the original 567 sequences in this dataset. The duration of the sequences range from 14 to 76 frames, which is approximately equivalent to 1–5 actionlets. This dataset contains 20 actions performed by 10 different subjects with 3 repetitions of each action. The 3D locations of 20 joints were stored in the screen coordinates.

The UTKinect-Action Dataset from Xia et al. [19] was collected as part of research on action recognition using depth sequences. In this dataset, the actions include *walk*, *sit down*, *stand up*, *pick up*, *carry*, *throw*, *push*, *pull*, *wave hands*, and *clap* hands. UTKinect-action contains 10 actions and each action was performed by 10 subjects.

The UCF Kinect Dataset was gathered form 16 actions and a total of 1280 sequence samples suitable for a gaming environment. These sequences sustain ranging from 29 to 269 frames. In each frame, the 3D coordinates, orientation, and binary confidence values of each of 15 joints are available and RGB images and depth maps are not stored. In this dataset, the actions contain *balance*, *climbladder*, *climbup*, *duck*, *hop*, *kick*, *leap*, *punch*, *run*, *stepback*, *stepfront*, *stepleft*, *stepright*, *twistleft*, *twistright*, and *vault*.

4.2. Evaluation settings

For MSR Action3D Dataset, in order to allow a fair comparison with the state-of-the-art methods, we followed the test setting of [18], dividing the 20 actions into three subsets *AS*1, *AS*2 and *AS*3 as shows in Table 1, each having 8 actions. The *AS*1 and *AS*2 group actions with similar movements, while the *AS*3 is relatively complex with more joints engaged. For cross-subject test setting, half of the subjects is used for training and the other half is used for testing.

For UTKinect-Action Dataset, to allow for comparison with [19], we follow the same experimental set up using Leave One Sequence Out Cross Validation (LOOCV) on the 200 sequences. For UCF Kinect Dataset, we follow the experimental setting used in Ellis et al. [32]. All of our experiments on this dataset are implemented using 4-fold cross-validation.

Table 1
The three subsets of actions used in the experiments.

Action set 1 (AS1)	Action set 2 (AS2)	Action set 3 (AS3)
Horizontal Wave (HoW)	High Wave (HiW)	High Throw (HT)
Hammer (H)	Hand Catch (HC)	Forward Kick (FK)
Forward Punch (FP)	Draw X (DX)	Side Kick (SK)
High Throw (HT)	Draw Tick (DT)	Jogging (J)
Hand Clap (HC)	Draw Circle (DC)	Tennis Swing (TSw)
Bend (B)	Hands Wave (HW)	Tennis Serve (TSr)
Tennis Serve (TSr)	Forward Kick (FK)	Golf Swing (GS)
Pickup Throw (PT)	Side Boxing (SB)	Pickup Throw (PT)

4.3. Discussion on parameters setting

Similar to other action recognition methods, our solution depends on the following parameters:

The sizes of the SOM: The first type of parameter refers to the sizes of the grid arrays of T_{ξ_a} and T_{ξ_p}. The larger the size of grid array, the more the over-fitting phenomenon occurs, and conversely, the smaller the size of grid array, the more the under-fitting phenomenon occurs. In the SOM Toolbox [33], the default number of neurons is $5\sqrt{n}$ where n is the number of training samples. Hence, this parameter could be set in terms of the rule given in our paper.

The number of clusters on SOM : This parameter refers to the number of chunks on SOM divided by DBI. The number $D(1 \le D \le 26)$ of chunks is limited by the number of upper-case and lower-case English letters. The activity recognition rate is directly affected by this parameter.

The number of match states: To evaluate the effectiveness of our profile HMMs in practice, we use the aforementioned heuristic of setting the initial model length to the average length of the action-unit sequences.

The initial probabilities and pseudocount weight: The probability $e_j(a)$ of state j emitting symbol a is estimated by counting the number of times $c_j(a)$ that represents the observed counts of state j emitting symbol a

$$e_j(a) = \frac{c_j(a)+1}{\sum_{a'} c_j(a')+W}. \tag{4}$$

where W is the weight given to the pseudo-counts. The probability a_{kl} of state k transitioning to state l can be sampled from a uniform-parameter Dirichlet distribution.

4.4. Experimental results

We first evaluate the performance of the proposed approach on the three challenging 3D action datasets. The proposed method's primary advantage is robustness temporal misalignment. The experiment results on MSR-Action3D datasets are shown in Table 2. In our experiments, the cross-subjects action recognition is conducted, which is more difficult than using the same subjects for both training and testing. From the results of MSR Action3D dataset on cross-subjects test, the recognition accuracy of our method is 86.4% which significantly outperforms the other joint-based action recognition methods. We observe that the proposed approach outperforms the methods using classical HMM [19]. The performance

on subset *AS*3 indicates that the proposed representation is better than [19] in modeling complex actions. But the performance with [19] on subsets *AS*1 and *AS*2 indicates that [19] is better than ours in differentiating similar actions. We conjecture the reason to be that a profile HMM requires a large number of training sequences (>100) for good similar action recognition [6]. Each action in MSR-Action3D dataset is performed by 10 different subjects with 3 repetitions of each action. Therefore, training sequences for a profile HMM are only 15 sequences (far less than 100) with cross-subject setting.

On a 3.30 GHz Intel Core i5 CPU machine, vector quantization to features of actionlets and postures use Matlab implementation. Profile HMMs are implemented by using C++. The average testing time of one sequence is 7.3 ms using Matlab. In paper [19] adopted classical HMMs, the average testing time of one sequence is 12.5 ms using Matlab on a 2.93 GHz Intel Core i7 CPU machine. Thus it is illustrated that the time complexity of profile HMMs is superior to classical HMMs.

Table 2
Comparison: recognition rate (%) on the MSR-Action3D dataset in cross-subject setting based on *AS*1, *AS*2, and *AS*3.

Method	AS1	AS2	AS3	Overall
Dynamic temporal warping [34]	–	–	–	54.0
Hidden Markov model [35]	–	–	–	63.0
Bag of 3D points [18]	72.9	71.9	79.2	74.7
Histogram of 3D Joints+HMM [19]	**88.0**	**85.5**	63.3	78.9
Eigenjoints [21]	74.5	76.1	96.4	83.3
Spatio-temporal feature chain [28]	82.2	85.4	85.6	84.4
Proposed method	84.7	79.2	**95.2**	**86.4**

Table 3
Human recognition accuracies on UTKinect-Action and UCF Kinect datasets.

UTKinect-Action	**Accuracy**
HO3DJ [19]	90.9
Spatio-temporal feature chain [28]	91.5
Proposed method	91.7
UCF Kinect	**Accuracy**
LAL [32]	95.9
Eigenjoint [21]	97.1
Spatio-temporal feature chain [28]	98.04
Proposed method	97.6

Following [18], the AS1 and AS2 were intended to group actions with similar movement, while AS3 was intended to group complex actions together. The parameters of D_a and D_p are the number of clusters of actionlets and postures, which directly affects the activity recognition rate. From Fig. 10a, we find that the accuracy hits their highest level (86.67%, 81.67%, and 97.5%) while the value of D_a is 18, 22, and 26 respectively in *AS*1, *AS*2, and *AS*3 as shown in Fig. 10b. These data show that the smaller D_a are, the lower the action recognition accuracy would be. But this situation is not suitable for the value of D_p because these accuracies hit their highest level while the value of D_p is 10, 10, and 18 respectively in *AS*1, *AS*2, and *AS*3. These phenomena indicated that D_a and D_p in complex actions (*AS*3) are larger than the ones in simple actions (*AS1* and *AS2*) and D_p is less than D_a in the same action. Therefore, we choice the number of actionlets and postures with 18 and 10 for *AS*1, 22 and 10 for *AS*2, and 26 and 18 for *AS*3 to calculate the mean accuracy of action recognition. The mean accuracy is 84.75%, 79.17%, and 95.25%, respectively. The total mean accuracy is 86.4%, the best accuracy is 88.6% and the standard deviation is 2.2%.

Specifically, it outperforms the state-of-the-art on UTKinect-Action dataset and UCT Kinect dataset as shown in Table 3. On the UTKinect-Action dataset, our approach has an accuracy of 91.7% which outperforms the HOJ3D feature in [19] (90.9%). Finally, we compare our result with all others on the UCF Kinect dataset. Our approach obtained a much better accuracy compared to the state-of-the-art works on this dataset.

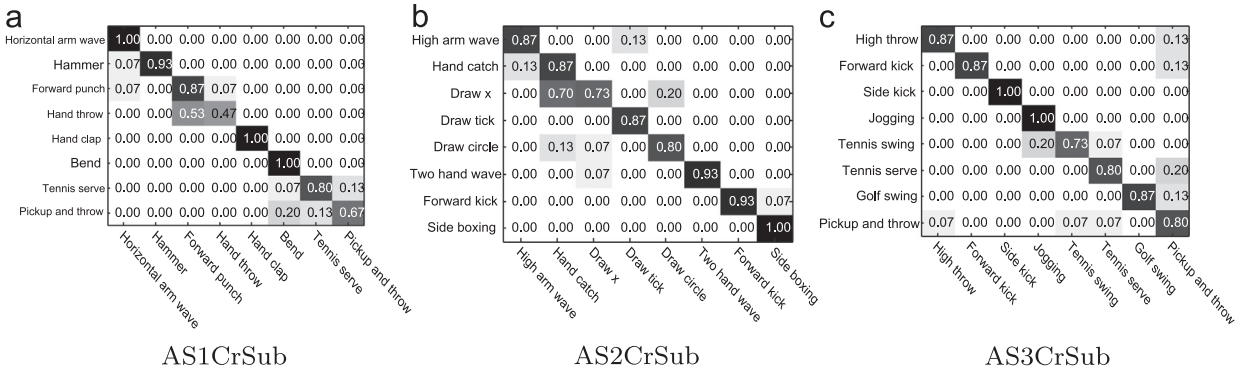

Fig. 9. Confusion matrix in AS1, AS2 and AS3 under cross subject test using profile HMMs.

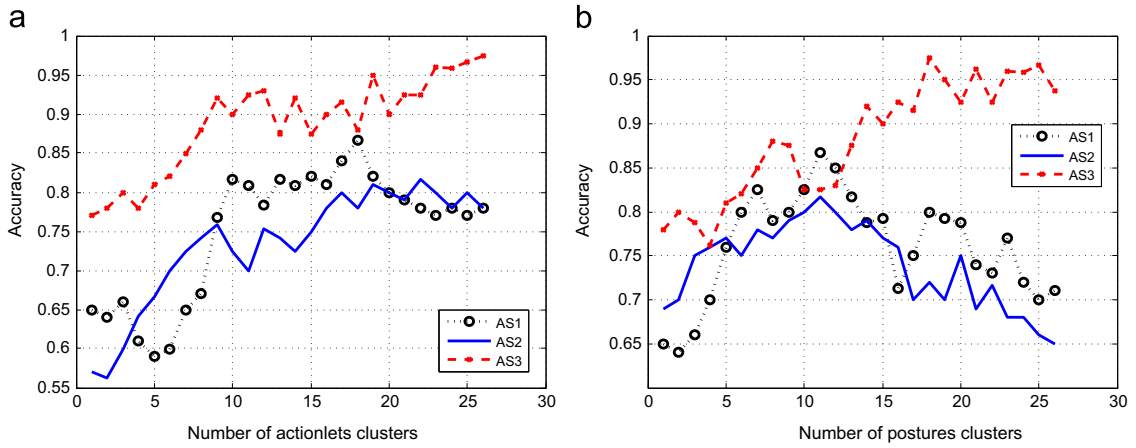

Fig. 10. (a) The accuracy of action recognition with the number of actionlets clusters D_a from 1 to 26. (a) The accuracy of action recognition with the number of postures clusters D_p from 1 to 26.

5. Conclusions and future work

In this paper, we obtain meaningful action-units through take advantage of segmentation points. With labeling these action-units, an action can be represented by discrete symbol sequences. To overcome an abrupt change or an abnormal in its gesticulation between different performances of the same action, profile HMMs are applied with these symbol sequences using Viterbi and Baum–Welch algorithms for human activity recognition. These methods eliminate the noise and the periodic motion problems experienced by methodologies that either solve it only by hand setup or ignore it. Applying action sequences to profile HMMs resulted in our approach to significantly outperform other state-of-the-art methods. The trends in this domain may devote more attention to applications on group activity recognition. With the development of technologies of the commodity sensors, depth data with more subjects or even groups of people may become available. In addition, obtaining robustness to occlusion may be considered by future algorithms which are essential to work in real scenarios. Therefore the next step is to understand and predict human activities, and more importantly, human interactions with the associated object affordances.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grant no. 61571345, the National Natural Science Foundation of China under Grant no. 9153801, the National Natural Science Foundation of China under Grant no. 61550110247, the Natural Science Foundation of the AnHui Higher Education Institutions of China under Grant no. KJ2014B14, and the Natural Science Foundation of Anhui Provice under Grant no. 1608085MF127.

References

[1] J. Aggarwal, L. Xia, Human activity recognition from 3d data: a review, Pattern Recognit. Lett. 48 (2014) 70–80.
[2] Z. Zhang, Microsoft kinect sensor and its effect, IEEE Multimed. 19 (2) (2012) 4–10.
[3] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, R. Moore, Real-time human pose recognition in parts from single depth images, Commun. ACM 56 (1) (2013) 116–124.
[4] D.M. Green, Profile Analysis: Auditory Intensity Discrimination, Oxford University Press, New York, NY, USA, 1988.
[5] L.R. Rabiner, B.-H. Juang, An introduction to hidden Markov models, IEEE ASSP Mag. 3 (1) (1986) 4–16.
[6] S.R. Eddy, Profile hidden Markov models, Bioinformatics 14 (9) (1998) 755–763.
[7] T. Kohonen, The self-organizing map, Proc. IEEE 78 (9) (1990) 1464–1480.
[8] D.L. Davies, D.W. Bouldin, A cluster separation measure, IEEE Trans. Pattern Anal. Mach. Intell. 2 (1979) 224–227.
[9] R. Poppe, A survey on vision-based human action recognition, Image Vis. Comput. 28 (6) (2010) 976–990.
[10] D. Weinland, R. Ronfard, E. Boyer, A survey of vision-based methods for action representation, segmentation and recognition, Comput. Vis. Image Underst. 115 (2) (2011) 224–241.
[11] M. Ye, Q. Zhang, L. Wang, J. Zhu, R. Yang, J. Gall, A survey on human motion analysis from depth data, in: Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications, Springer, Berlin, Heidelberg, 2013, pp. 149–187.
[12] I. Laptev, On space-time interest points, Int. J. Comput. Vis. 64 (2–3) (2005) 107–123.
[13] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005, San Diego, CA, USA. CVPR 2005, vol. 1, IEEE, 2005, pp. 886–893.
[14] N. Dalal, B. Triggs, C. Schmid, Human detection using oriented histograms of flow and appearance, in: Computer Vision—ECCV , Graz, Austria, 2006, Springer, Berlin, Heidelberg, 2006, pp. 428–441.
[15] Y. Ke, R. Sukthankar, M. Hebert, Efficient visual event detection using volumetric features, in: 2005 IEEE International Conference on Computer Vision (ICCV), Beijing, China, vol. 1, IEEE, 2005, pp. 166–173.
[16] J. Wang, Z. Liu, J. Chorowski, Z. Chen, Y. Wu, Robust 3d action recognition with random occupancy patterns, in: Computer Vision—ECCV 2012, Springer, 2012, pp. 872–885.
[17] L. Xia, J. Aggarwal, Spatio-temporal depth cuboid similarity feature for activity recognition using depth camera, in: 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Portland, OR, USA, IEEE, 2013, pp. 2834–2841.
[18] W. Li, Z. Zhang, Z. Liu, Action recognition based on a bag of 3D points, in: 2010 IEEE Computer Society Conference on Computer

Vision and Pattern Recognition Workshops (CVPRW), San Francisco, CA, USA, IEEE, 2010, pp. 9–14.
[19] L. Xia, C.-C. Chen, J. Aggarwal, View invariant human action recognition using histograms of 3D joints, in: 2012. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Providence, RI, USA, IEEE, 2012, pp. 20–27.
[20] L. Miranda, T. Vieira, D. Martinez, T. Lewiner, A.W. Vieira, M.F.M. Campos, Real-time gesture recognition from depth data through key poses learning and decision forests, in: 2012 25th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), IEEE, 2012, pp. 268–275.
[21] X. Yang, Y. Tian, Effective 3d action recognition using eigenjoints, J. Vis. Commun. Image Represent. 25 (1) (2014) 2–11.
[22] J. Wang, Z. Liu, Y. Wu, Learning Actionlet Ensemble for 3D Human Action Recognition, in: Human Action Recognition with Depth Cameras, Springer International Publishing, New York, NY, USA, 2014, pp. 11–40.
[23] F. Lv, R. Nevatia, Single view human action recognition using key pose matching and Viterbi path searching, in: IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, MN, USA, 2007, CVPR 07, IEEE, 2007, pp. 1–8.
[24] Y. Ukrainitz, M. Irani, Aligning sequences and actions by maximizing space–time correlations, in: Computer Vision—ECCV 2006, IEEE, Graz, Austria, Springer, Berlin, Heidelberg, 2006, pp. 538–550.
[25] C.N. Fischer, C.M. Cararetto, R.A. dos Santos, R. Cerri, E. Costa, L. Schietgat, C. Vens, Learning HMMs for nucleotide sequences from amino acid alignments, Bioinformatics 31 (11) (2015) 1836–1838.
[26] C.V. Wright, F. Monrose, G.M. Masson, On inferring application protocol behaviors in encrypted network traffic, J. Mach. Learn. Res. 7 (2006) 2745–2769.
[27] A. Bhargava, G. Kondrak, Multiple word alignment with profile hidden Markov models, in: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Student Research Workshop and Doctoral Consortium, Association for Computational Linguistics, 2009, pp. 43–48.
[28] W. Ding, K. Liu, F. Cheng, J. Zhang, STFC: spatio-temporal feature chain for skeleton-based human action recognition, J. Vis. Commun. Image Represent. 26 (2015) 329–337.
[29] J.D. Ferguson, Variable duration models for speech, in: Proceedings of the Symposium on the Application of HMMs to Text and Speech, 1980, pp. 143–179.
[30] A.J. Viterbi, Error bounds for convolutional codes and an asymptotically optimum decoding algorithm, IEEE Trans. Inf. Theory 13 (2) (1967) 260–269.
[31] L.E. Baum, T. Petrie, G. Soules, N. Weiss, A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains, Ann. Math. Stat. (1970) 164–171.
[32] C. Ellis, S.Z. Masood, M.F. Tappen, J.J. Laviola Jr, R. Sukthankar, Exploring the trade-off between accuracy and observational latency in action recognition, Int. J. Comput. Vis. 101 (3) (2013) 420–436.
[33] J. Vesanto, J. Himberg, E. Alhoniemi, J. Parhankangas, Self-organizing map in Matlab: the SOM Toolbox, in: Proceedings of the Matlab DSP Conference, vol. 99, 1999, pp. 16–17.
[34] M. Müller, T. Röder, Motion templates for automatic classification and retrieval of motion capture data, in: Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, Eurographics Association, Switzerland, 2006, pp. 137–146.
[35] F. Lv, R. Nevatia, Recognition and segmentation of 3-d human action using hmm and multi-class adaboost, in: Computer Vision–ECCV 2006, 2006, Springer, pp. 359–372.