J. Vis. Commun. Image R. 38 (2016) 627–640

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

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

Parsing 3D motion trajectory for gesture recognition ☆

Jianyu Yang [a,b,*], Junsong Yuan [b], Youfu Li [c]

[a] School of Urban Rail Transportation, Soochow University, 8 Jixue Road, Xiangcheng District, Suzhou, Jiangsu, China
[b] School of Electrical and Electronic Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore
[c] Department of Mechanical and Biomedical Engineering, City University of Hong Kong, 83 Tat Chee Avenue, Kowloon, Hong Kong

ARTICLE INFO

Article history:
Received 7 June 2015
Revised 23 March 2016
Accepted 8 April 2016
Available online 9 April 2016

Keywords:
3D trajectory representation
Motion recognition
Trajectory primitive

ABSTRACT

Motion trajectories have been widely used for gesture recognition. An effective representation of 3D motion trajectory is important for capturing and recognizing complex motion patterns. In this paper, we propose a view invariant hierarchical parsing method for free form 3D motion trajectory representation. The raw motion trajectory is first parsed into four types of trajectory primitives based on their 3D shapes. These primitives are further segmented into sub-primitives by the proposed shape descriptors. Based on the clustered sub-primitives, trajectory recognition is achieved by using Hidden Markov Model. The proposed parsing approach is view-invariant in 3D space and is robust to variations of scale, temporary speed and partial occlusion. It well represents long motion trajectories can also support online gesture recognition. The proposed approach is evaluated on multiple benchmark datasets. The competitive experimental results and comparisons with the state-of-the-art methods verify the effectiveness of our approach.

1. Introduction

Motion trajectory analysis has extensive applications in video surveillance, human-robot interaction and gesture recognition [1,2]. By tracking the target object, e.g., the hand, in the real environment, a 3-dimensional (3D) motion trajectory is a sequence of 3D points that capture the position information over time. With the advent of cheaper depth sensors [3], it is clear that recognizing semantic gestures from 3d data will be a widely applicable technology in the coming years. Compared with conventional 2D trajectory analysis, e.g., via object tracking in videos, 3D motion trajectory usually contains more complex information and is composed of a series of basic motion elements that may convey meanings, e.g., sign language. Thus understanding 3D motion trajectories is different from 2D analysis. Fig. 1(a) and (b) illustrate two cases of recognizing complex motion trajectories.

To capture the 3D shape of the motion trajectory, traditional approaches process motion trajectory as a 3D curve, where local shapes are captured by the point feature for analysis, e.g., raw trajectory data [4,5], or simple shape descriptors [6–9]. However, the point features usually ignore the context information, and the local shape descriptor is sensitive to local noise. Such a detailed level description by using point features has difficulty of handling the intra-class variations of the motion patterns too. For example, the same sign language operated by two persons could be significantly different in the detailed trajectory, but similar in terms of global 3D shapes. Thus a more robust representation of global structural information of 3D motion trajectory is required.

In recent work, hierarchical trajectory segmentation approaches [10–12] show promising results in motion recognition. Unlike point-level descriptors, these approaches segment trajectories into primitives where the global structure and distribution of motion primitives are used to represent the 3D motion trajectory. However, these hierarchical approaches usually require the primitives to be specially designed for different motion recognition tasks, which is not flexible as the primitives and parsing rules may vary with different applications. Also, it is a time consuming process to parse trajectories into pre-defined primitives.

To address the above problems, we propose a novel parsing method for gesture recognition. In our method, we first segment the long motion trajectory into four types of motion trajectory primitives based on their local 3D shapes, i.e., A. straight line, B. plane arc, C. right-hand helix and D. left-hand helix. After the qualitative segmentation, we further quantize each trajectory primitive into temporal segments, i.e., sub-primitive, and describe each sub-primitive using the proposed shape descriptor. These

☆ This paper has been recommended for acceptance by M.T. Sun.

* Corresponding author at: School of Urban Rail Transportation, Soochow University, 8 Jixue Road, Xiangcheng District, Suzhou, Jiangsu, China.
E-mail addresses: jyyang@suda.edu.cn (J. Yang), JSYUAN@ntu.edu.sg (J. Yuan), meyfli@cityu.edu.hk (Y. Li).

http://dx.doi.org/10.1016/j.jvcir.2016.04.010
1047-3203/

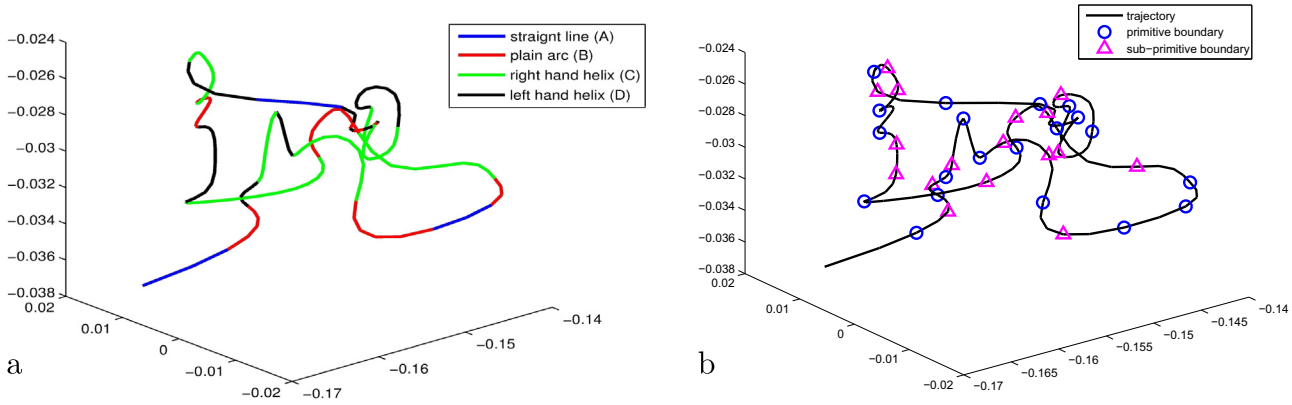

Fig. 1. A case of 3D motion trajectory parsing and segmentation results. (a) present the parsing result of a motion trajectory in 3D space, where the four colors represent the primitives of straight line, plane arc, left hand helix and right hand helix. (b) is the segmentation results of (a), where the blue circles and red triangles represent the boundaries of primitives and sub-primitives, respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

sub-primitives serve as the middle level presentation between the point level and primitive level presentation. Finally, the trajectory is represented by a sequence of sub-primitives. Hidden Markov Model (HMM) can be used to model and recognize the trajectory, i.e., a sequence of sub-primitives, where each sub-primitive is an observation of HMM. Our trajectory representation method is sketched in Fig. 2.

Our hierarchical representation of 3D motion trajectories has the following advantages:

- Our representation of 3D trajectories is view-invariant, robust to scale variations, temporal speed variations, as well as local noise and partial occlusion.
- Our segmentation and parsing method can represent arbitrary 3D motion trajectories, which is flexible for general applications. The proposed trajectory features are able to tolerate the intra-class variations of the motion trajectories well, and are also compatible to existing trajectory recognition approaches.
- The proposed parsing method can be implemented in real time, which is important for gesture recognition and human machine interaction.

The proposed approach is evaluated on three benchmark datasets: HDM05 dataset [13], ASL dataset [14] and Berkeley MHAD dataset [15], including clean and detailed 3D daily motion trajectories of hands and body joints captured by a multi-camera motion capturing system. The experiments are conducted on these datasets to validate the invariant properties, effectiveness and robustness of the proposed approach and the experimental results are compared with the state-of-the-art methods.

The organization of the remainder of this paper is as follows. The next section summarizes relevant work on motion representation and recognition. In Section 3, the parsing of motion trajectory primitives is presented. Quantizing the primitives into sub-primitives is detailed in Section 4. Section 5 discusses the cluster model and motion recognition process with the proposed

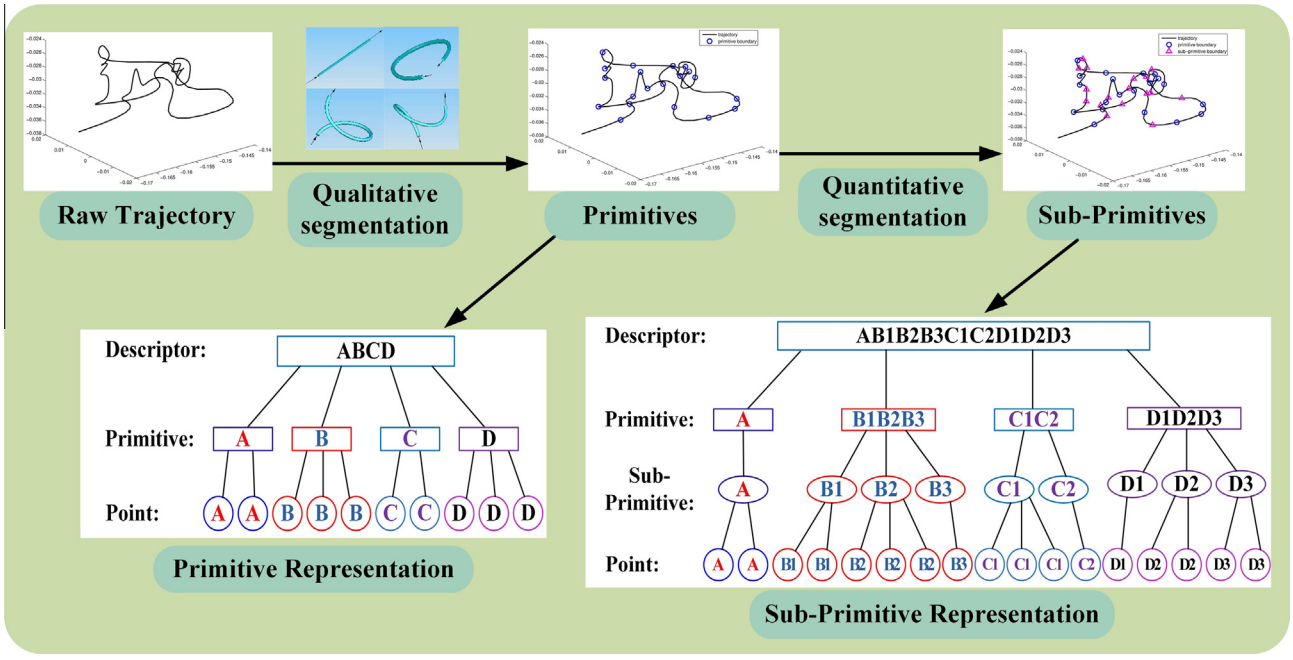

Fig. 2. The flowchart of the proposed representation method. The trajectory data are segmented into primitives first, and then quantized to sub-primitives to get the sub-primitive sequence.

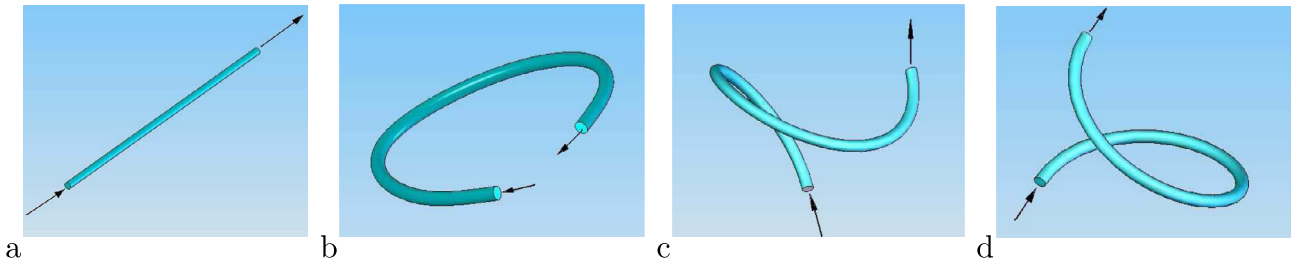

Fig. 3. Four types of trajectory primitives: (a) Primitive A, (b) Primitive B, (c) Primitive C and (d) Primitive D. Primitive A is a straight line with zero curvature and zero torsion, Primitive B is planar arcs with arbitrary curvatures and zero torsion, Primitive C and D are right and left hand helix trajectories respectively with arbitrary curvatures and torsions.

descriptors, and experiments follow up in Section 6. Finally, this work is concluded in Section 7.

2. Related work

Many efforts have been devoted to describe motion trajectories for motion analysis. An intuitive scheme [4,16] is to adopt the raw motion trajectory to represent the salient features of motion, which is not robust or flexible, e.g., sensitive to local noise. Trajectory contour functions have been proposed to capture motion features from shape, e.g., the chain code [6,17], the centroid-contour distance (CCD) [18] and the curvature scale space (CSS) [19]. Transformation functions have also been employed to model motion by transforming trajectories into frequency domains, including Fourier descriptor [20] and Wavelet transform [21]. Based on algebraic curves, Cohen et al. [22], Chen and Guan [23] and Hsieh et al. [24] use B-spline, NURBS and Bezier curve respectively to represent motion trajectories. However, all these shape descriptors are not capable of representing shape features well. Contour functions are more suitable for modeling simple shapes. Transformation functions ignore most of the local features and less-important information in transforming to the frequency domain, but these features and information are also important in motion analysis. Spline models use a group of control points to represent trajectories where a fitting process is needed, and such models cannot be compared directly by their parameters and control points for recognition.

Probabilistic approaches, i.e., Dynamic Bayesian Network (DBN) [25] and its special cases, Kalman filtering and Hidden Markov Model (HMM) [26], have been employed to model motion trajectories as sequences of signal frames. Oliver et al. [27] propose a coupled HMM model to represent the motion interaction between two persons. Suk et al. [25] propose a DBN architecture with three chains of HMM code to model the motion trajectories of two hands and the relative position between them. Hongeng et al. [28] use a Semi-Markov Chain (SMC) to represent the temporal information of motion trajectories. These probabilistic models requires considerable training data and computational cost, especially as the length of the motion trajectory increases.

A number of existing motion analysis approaches treat motion trajectories as pairwise similarity models between two trajectories. Some of them require the alignment of the corresponding points between two trajectories for similarity (distance) calculation. For example, Keogh and Pazzani use the Dynamic Time Warping (DTW) algorithm [29] for pairwise matching. For partial alignment of motion trajectories, Vlachos et al. present the Longest Common Subsequence (LCSS) distance [30] to match the longest similar part of trajectories. Latecki et al. propose another partial alignment algorithm, the Minimal Variance Matching (MVM) [31] approach. The drawback of these similarity models is that the trajectory points are used separately, so that features composed of point sequences cannot be recognized.

The invariant descriptors have been widely used for motion trajectory representation. In recent years, several methods [8,9,32–35] focus on using spatial shape invariants to represent 3D motion features and parameters. Rao et al. [8] use curvature to calculate the instants and intervals of motion which are too simple to fully represent complex motions. Wu and Li [32] and Yang [33] use the differential invariants to represent the 3D motion trajectory directly without the 3D-2D projection, where most of the 3D information is preserved. After that, Shao and Li [35] extends the integral invariants [36–38] to the 3D space to represent 3D motion trajectories, which is more robust to high frequency noise than the differential invariants. Based on these point-level invariants, the proposed parsing method is to represent the trajectory in the primitive level with invariant properties.

To represent the global structure of motion, many parsing based approaches [7,8,12,10,11] have been proposed to capture the semantic features or global structures of motion trajectories. The curvature of a trajectory is an invariant parameter which is used to segment motion trajectories in the work of Rao et al. [8]. However, complex motion trajectories cannot be segmented based on curvature only, which can sacrifice robustness. Berretti et al. [7] segment curves into simple tokens, but simple shapes are ineffective when representing curves with manifold trajectories. The grammar based approaches [10,11] have also been proposed, which segment motion trajectories into primitives (basic events) and model their temporal structures with parsing rules. However, the primitives need to be specially defined for specific applications beforehand, which is inefficient. As the primitives are captured from 2D scenes, the 3D information is lost. Therefore, these parsing approaches are not appropriate for freeform 3D motion trajectories. Wang et al. [39] propose a method to classify the 3D surface first and then performing analysis on top of the integrated classification result. Li et al. [40] propose a 3D extension of shape context. Both of these methods give a good inspiration of 3D curve analysis.

3. Parsing 3D motion trajectory

In this section, we first define four types of trajectory primitives in Section 3.1, and then we propose algorithms to segment and parse arbitrary 3D trajectories into a symbolic sequence composed by the four types of primitives in Section 3.2. Finally we explain the invariant properties of the proposed parsing method in Section 3.3.

3.1. Trajectory primitives

Given a trajectory Γ, the 3D coordinates of the trajectory points are $\Gamma = \{p(t)|t \in [1, N]\}$, where t is the temporal index and N is the total length of a trajectory. A trajectory point

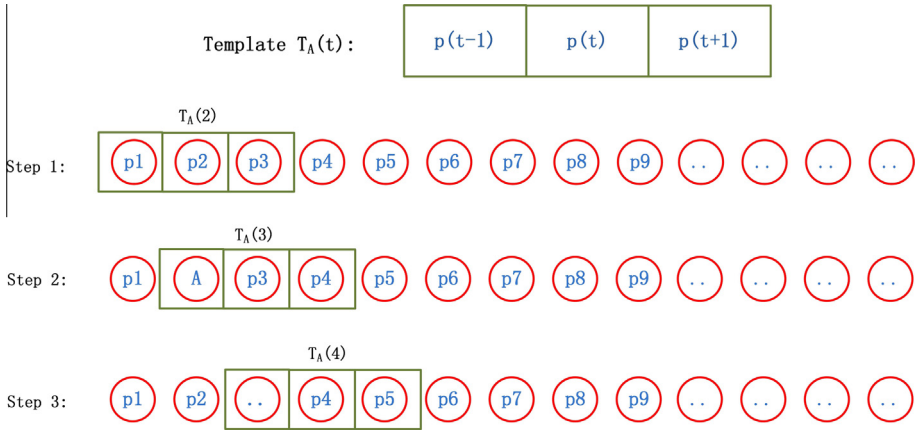

Fig. 4. Template T_A and the first 3 steps of testing with T_A. If $T_A(2)$ is 1 in Step 1, then p(2) is labeled with A and goes to Step 2. If $T_A(3)$ is 0 in Step 2, then p(3) is not labeled and goes to the next step.

$p(t) = \{x(t), y(t), z(t)\} \in \mathcal{R}^3$ is described by its location in the 3D space.

In this work, we propose to decompose an arbitrary 3D trajectory into four types of trajectory primitives, i.e., A-straight line, B-plane arc, C-right hand helix and D-left hand helix, as shown in Fig. 3. For a segment of trajectory points $\Lambda = \{p(t), p(t+1), \ldots, p(t+k)\}$, the four primitive types are defined as follows:

Primitive Type A (straight line): Λ is a straight line if any three sequential trajectory points are collinear: $\forall a \in [1, k-1]$, $\exists : \langle \vec{v}(t+a), \vec{v}(t+a+1) \rangle = ||\vec{v}(t+a)|| \times ||\vec{v}(t+a+1)||$, where $\vec{v}(t+a)$ is the vector from $p(t+a-1)$ to $p(t+a)$, $||\vec{v}||$ is the l_2 norm of vector $\vec{v}$, and $\langle \vec{u}, \vec{v} \rangle$ is the inner product of vectors $\vec{u}$ and $\vec{v}$. $\vec{v}(t) = \overrightarrow{p(t) - p(t-1)}$ is the moving direction of the tracked object from frame $t-1$ to frame t.

Primitive Type B (plane arc): Λ is an arc on a plane if (1) for any three sequential points, they are not collinear and (2) for any four sequential points, they are coplane: (1) $\forall a \in [1, k-1]$, $\exists : \langle \vec{v}(t+a), \vec{v}(t+a+1) \rangle < ||\vec{v}(t+a)|| \times ||\vec{v}(t+a+1)||$ and (2) $\forall a \in [1, k-2], \exists : V_{p(t+a-1),p(t+a),p(t+a+1),p(t+a+2)} = 0$, where $V_{p(t-1),p(t),p(t+1),p(t+2)}$ is the volume of the tetrahedron with vertices $p(t-1), p(t), p(t+1)$ and $p(t+2)$:

$$V_{p(t-1),p(t),p(t+1),p(t+2)} = \frac{1}{3!} \times \begin{vmatrix} x(t) & y(t) & z(t) & 1 \\ x(t-1) & y(t-1) & z(t-1) & 1 \\ x(t+1) & y(t+1) & z(t+1) & 1 \\ x(t+2) & y(t+2) & z(t+2) & 1 \end{vmatrix}. \quad (1)$$

Primitive Type C (right hand helix): Λ is a right hand helix if for every four sequential points, they are not coplane and the helix direction of every four sequential points is positive: $\forall a \in [1, k-2], \exists$: (1) $V_{p(t+a-1),p(t+a),p(t+a+1),p(t+a+2)} > 0$ and (2) $\langle \vec{v}(t+a) \otimes \vec{v}(t+a+1), \vec{v}(t+a+2) \rangle > 0$, where $\vec{u} \otimes \vec{v}$ is the outer product of vectors $\vec{u}$ and $\vec{v}$.

Primitive Type D (left hand helix): Λ is a left hand helix if for every four sequential points, they are not coplane and the helix direction of every four sequential points is negative: $\forall a \in [1, k-2], \exists$: (1) $V_{p(t+a-1),p(t+a),p(t+a+1),p(t+a+2)} > 0$ and (2) $\langle \vec{v}(t+a) \otimes \vec{v}(t+a+1), \vec{v}(t+a+2) \rangle < 0$.

We only select these four types of primitives due to three reasons:

1. The geometrical property of 3D motion. For 1D motion, the object can only move along a straight line (arbitrary length), and the primitives are all of Type A. For 2D motion, the object can move along a piece of plane arc (with arbitrary curvature or non-uniform curvature) or a straight line (straight line can be seen as a special arc with zero curvature), and the primitives include Type A and B. The plane arc is the characteristic primitive of 2D space which the 1D space does not have. For 3D motion, besides straight line and plane arc, the object can move along a helix trajectory (left or right hand helix with arbitrary or non-uniform torsion and curvature, as characteristic primitive of 3D space). An object can never move along a trajectory out of these four types of primitives, e.g., a 4D primitive.
2. Invariant to further segmentation. For a primitive belonging to type X (X can be A, B, C or D), if it is further segmented into smaller pieces, each piece must belongs to type X. That is, if a trajectory has been segmented into the four types of primitives, the further segmentation of the primitives does not introduce new primitive and the type of the child primitive is the same as its father primitive. The child primitive is only shorter than the father primitive, but has the same representation and property. We segment a complex trajectory, because it consists of primitives with different properties. If the fragments have no difference in representation and property after segmentation, the father primitive is enough to represent the trajectory.
3. The four types of primitives are complete to represent a trajectory. Types A, B, C and D include 1D, 2D and 3D curves separately. If a segment of trajectory cannot be classified into one of the four types, it includes at least two dimensional of curves, e.g., a trajectory passes from a 2D curve to a 3D curve. Thus we segment it at the boundary point between curves of two dimensions until each segment belongs to one of the four types. In this way, any motion trajectory can be segmented to the four types of primitives.

Therefore, we use these four types of primitives as basic atoms of 3D trajectory. Any complex motion is a combination of these primitives that can be segmented into fragments and each fragment belongs to one of the four types. Based on this 4-primitive descriptor, a "rich" descriptor which is much more representative can be defined in the following two manners: (1) define new primitives as combinations of the 4 basic primitives or (2) divide each type of primitives into more sub-types according to special defined features. That is, the representative primitives can be learned based on the 4-primitive descriptor for specific tasks (as a pattern). As both the further segmented pieces and the complex patterns can be represented by the four types of primitives, there cannot be other alternative to be used as basic atom of 3D trajectory.

In the following section, we show that any 3D trajectory can be decomposed into the above four types of primitives by parsing all the points of the trajectory.

3.2. Trajectory parsing and segmentation

In this section, we parse and segment motion trajectory into primitives according to the primitive definitions. In order to reduce the effect of noise in trajectory data while preserving the shape and kinematic properties, the Kalman-based linear smoother [41] is employed to reduce noise and outliers. The trajectory is then parsed by Algorithm 1 to label points of type A, and parsed by Algorithm 2 to label points of type B, C and D. At last, the boundary points are localized by Algorithm 3 for the trajectory segmentation.

3.2.1. Parsing Primitive A

Following the definition of Primitive Type A, we use a template T_A, to parse collinear points of the trajectory:

$$T_A(t) = \mathbb{F}\{\langle \vec{v}(t), \vec{v}(t+1)\rangle = \|\vec{v}(t)\| \times \|\vec{v}(t+1)\|\}, t \in [2, n-1], \quad (2)$$

where $\mathbb{F}\{*\}$ is a Boolean Function, i.e., $\mathbb{F} = 1$ if '$*$' is true and $\mathbb{F} = 0$ otherwise. Fig. 4 shows template T_A and the parsing processes. If $T_A(t) = 1$, the point $p(t)$ with its neighbor points $p(t-1)$ and $p(t+1)$ satisfy the definition of Primitive Type A. Therefore, $p(t)$ is classified into the Type A, which means the object moves within a one-dimensional (straight line) trajectory, regardless of the direction and position of this line. If $T_A(t) = 0$, the point is not labeled, which means that the object moves out of the original direction into a two-dimensional (plane) or three-dimensional (helix) trajectory. The parsing and labeling of trajectory points A are described in Algorithm 1. To make it robust to noise, the threshold e_A is set based on the mean square distance of the original trajectory with respect to the Kalman smoothed trajectory.

Algorithm 1. Labeling **A** points by T_A

Input: Raw trajectory data $\Gamma(t)$, $t \in [1, N]$; threshold e_A
Output: The labeling result $L(t)$, $t \in [2, N-1]$
for $t = 2 \rightarrow N-1$ **do**
  $\delta = 1 - (p(t) - p(t-1)) \cdot (p(t+1) - p(t))/(|p(t) - p(t-1)| \cdot |p(t+1) - p(t)|)$
  if $\delta < e_A$ **then**
    $L(t) = A$
  else
    $L(t) = 0$
  end if
end for

3.2.2. Parsing Primitive B, C and D

Following the definition of Primitive Type B, the template T_B shown in Fig. 5 is defined as follows:

$$T_B(t) = \mathbb{F}\{V_{p(t-1),p(t),p(t+1),p(t+2)} = 0\}, \quad t \in [2, n-2] \cap T_A(t) = T_A(t+1) = 0. \quad (3)$$

If $T_B(t) = 1$, the volume of the tetrahedron is zero, that is, the four points $p(t-1), p(t), p(t+1)$ and $p(t+2)$ are coplane, which means the object moves within a spatial plane. In this case, the points $p(t)$ and $p(t+1)$ are labeled with B, and then move the template T_B to the next two unlabeled points. If $T_B(t) = 0$, which means the four points $p(t-1), p(t), p(t+1)$ and $p(t+2)$ are not coplane, such that the object moves out of the previous plane into a helix trajectory. In this case, the points $p(t)$ and $p(t+1)$ are labeled with C for the right hand helix trajectory, or with D for the left hand helix trajectory. To determine the helix direction of four sequential points $p(t-1), p(t), p(t+1)$ and $p(t+2)$, we employ the concept of the oriented plane [42]. The oriented plane spanned by the first three points is denoted as $\langle p(t-1), p(t), p(t+1)\rangle$, and its direction is determined by the order of points. The helix direction α of these four points is defined as follows:

$$\alpha(p(t-1), p(t), p(t+1); p(t+2)) = \begin{cases} \text{C}, & \text{if } \langle \vec{v}(t+a) \otimes \vec{v}(t+a+1), \vec{v}(t+a+2)\rangle > 0, \\ \text{D}, & \text{if } \langle \vec{v}(t+a) \otimes \vec{v}(t+a+1), \vec{v}(t+a+2)\rangle < 0, \end{cases} \quad (4)$$

where $\alpha = C$ means $p(t+2)$ lies in front of $\langle p(t-1), p(t), p(t+1)\rangle$ and the points both $p(t)$ and $p(t+1)$ are labeled with C, while $\alpha = D$ means $p(t+2)$ lies behind $\langle p(t-1), p(t), p(t+1)\rangle$ and the points both $p(t)$ and $p(t+1)$ are labeled with D. The parsing and labeling of trajectory points B, C and D are described in Algorithm 2. As the definitions of Primitives A, B, C and D are invariant due to the well known invariance of collinearity and coplanarity, the parsing of trajectory by Algorithms 1 and 2 is both viewpoint and scale invariant.

Algorithm 2. Labeling **B**, **C**, **D** points by T_B

Input: $\Gamma(t)$, $t \in [1, N]$; $L(t)$, $t \in [2, N-1]$ obtained by Algorithm 1; threshold e_B
Output: The updated labeling result $L(t)$, $t \in [2, N-1]$
for $t = 2 \rightarrow N-2$ **do**
  if $L(t) = 0 \cap L(t+1) = 0$ **then**
    $\delta = V_{p(t-1),p(t),p(t+1),p(t+2)}$ (from Eq. (1))
    if $\delta < e_B$ **then**
      $L(t) = L(t+1) = B$
    else if $\alpha = C$ (from Eq. (4)) **then**
      $L(t) = L(t+1) = C$
    else
      $L(t) = L(t+1) = D$
    end if
  end if
end for

3.2.3. Invariant boundary points

After parsing and labeling the trajectory with Algorithms 1 and 2, the trajectory points are labeled with A, B, C and D, as well as the unlabeled isolated points. Here we obtain the boundary points in two situations: (1) the position of label transition and (2) isolated points. The label transition is a natural boundary where the labels of sequential trajectory points transit from one type to another in time series, e.g., the transition from B to C in "...*BBCCC*...". In this case, the last B or the first C should be labeled with *O* as the boundary point. For the isolated points, there are two types: one is an unlabeled point between two A points, in terms of "...*AOA*...", where a straight line trajectory turns its motion direction to another straight line at the boundary point *O*. The other type of isolated points exist because the length of the sequential unlabeled points is odd after the parsing by T_A. After parsing the odd sequence by T_B, an unlabeled point is left at the end of this sequence, in terms of "...*XXOA*...". This point *O* is the boundary between Primitive A and Primitive X, where X could be B, C or D. The localization of the boundary points is described in Algorithm 3. Given the trajectory points labeled with A, B, C, D and O, a label sequence $\{L(t)|t \in [1, N]\}$ in time series is generated, where $L \in \{A, B, C, D, O\}$. Then, the trajectory is able to be segmented into primitives at the boundary points.

From Algorithm 3, we can find that the positions of boundary points are determined after the parsing by Algorithms 1 and 2.

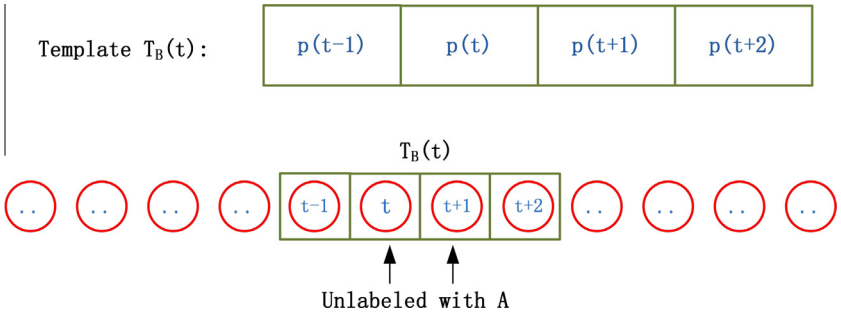

Fig. 5. Template T_B.

As the parsing of trajectory by Algorithms 1 and 2 is viewpoint and scale invariant, the labeling of trajectory points with A, B, C and D is invariant, and the number of boundary points is invariant as well. Therefore, boundary points of label transitions are invariant to viewpoint and scale variation.

Algorithm 3. Localizing boundary points O

Input: $L(t)$, $t \in [2, N-1]$ obtained by Algorithm 2;
Output: The updated labeling result $L(t)$, $t \in [2, N-1]$ with boundary points O
1 **for** $t = 2 \rightarrow N-2$ **do**
2 **if** $L(t) = 0$ **then**
3 $L(t) = O$ (*for isolated points*)
4 **else if** $L(t+1) \neq 0 \cap L(t) \neq L(t+1)$ **then**
5 $L(t) = O$ (*for label transition*)
6 **end if**
7 **end for**

After the trajectory label sequence is segmented into a sequence of primitives, the trajectory Γ is represented by this primitive sequence as follows:

$$\Gamma = \{\mathcal{V}(1), \mathcal{V}(2), \ldots, \mathcal{V}(i), \ldots, \mathcal{V}(m)\}, \tag{5}$$

where $\mathcal{V}(i) \in \{A, B, C, D\}$ is the i-th trajectory primitive and the description of $\mathcal{V}(i)$ is explained in the next section. The structure of the segmentation result is shown in Fig. 2. As the primitive sequences are simple, it is efficient to use them for motion analysis. Moreover, a dataset of motion trajectories can be easily indexed by ordering the sequences similar to the gene sequence. Trajectories belonging to the same motion class have similar primitive sequences. The trajectory segmentation result of a 3D motion trajectory in the ASL dataset is shown in Fig. 1(b), where the blue circles represent the boundary points.

3.3. Properties of the proposed trajectory parsing and segmentation

The proposed primitive parsing and segmentation have the following properties:

- Our method is *viewpoint and scale invariant*. The viewpoint variation is common in trajectory acquisition. It is well known that the collinearity and coplanarity of sequential points are invariant if the trajectory is rotated or scaled globally. In the experiments, we find that the proposed method is also robust to scaling inconsistently in different dimensions.
- The primitive segmentation can handle *partial occlusion*. The primitive representation of trajectory is calculated locally. If the trajectory is partially occluded, the non-occluded portion of trajectory is still correctly segmented and labeled. Therefore, the parsing result of the partially occluded trajectory can be matched with that of a full or occluded trajectory.
- The proposed trajectory representation method is *consistent*. For a given trajectory $\Gamma(t) = \{p(1), p(2), \ldots, p(N)\}$, the labeling and segmentation results $\{L(1), L(2), \ldots, L(N)\}$ and $\{\mathcal{V}(1), \mathcal{V}(2), \ldots, \mathcal{V}(m)\}$ are independent on the motion direction along the trajectory. That is, if the time series is reversed to $\Gamma\prime(t) = \{q(1), q(2), \ldots, q(N)\} = \{p(N), p(N-1), \ldots, p(1)\}$, the labeling and segmentation results are $\{L(N), L(N-1), \ldots, L(1)\}$ and $\{\mathcal{V}(m), \mathcal{V}(m-1), \ldots, \mathcal{V}(1)\}$, respectively. For a given primitive, if the point sequence is reversed, this primitive satisfies the primitive definition as well.

4. Description of trajectory primitives

By translating a 3D trajectory into a sequence of trajectory primitives, we can easily index and search the motion trajectories. However, detailed local shape information may get lost in such a representation. For example, two different trajectory pieces may be represented by the same label sequence while their detailed shapes are significantly different, as shown in Fig. 6. For discriminative representation of motion trajectories, we need to further describe the primitives, instead of simply using the primitive type representation.

For a segmented trajectory, we first delete the *too short* primitives which is likely to be caused by noises. To describe the detailed trajectory features of the four primitive types, we define a new shape feature descriptor f. The four primitive types have different features and we use the most representative features for each type, e.g., the torsion of trajectory is representative for Type C and D, while useless for Type A and B, etc. We define the new descriptor as follows:

$$f(p(t)) = \begin{cases} 1, \text{if } p(t) \in A \\ \sqrt{k^2(t) + \lambda \cdot k_s^2(t)}, \text{ if } p(t) \in B \\ \sqrt{\tau^2(t) + \lambda \cdot \tau_s^2(t)}, \text{ if } p(t) \in C \\ \sqrt{\tau^2(t) + \lambda \cdot \tau_s^2(t)}, \text{ if } p(t) \in D \end{cases} \tag{6}$$

where λ is a complexity factor. The four parameters $\{k, k_s, \tau, \tau_s\}$ are employed to evaluate the detailed shape features in the point level. According to [32], the four parameters are differential invariants: curvature $k = ||\dot{\Gamma}(t) \otimes \ddot{\Gamma}(t)||/||\dot{\Gamma}(t)||^3$, torsion $\tau = \langle \dot{\Gamma}(t) \otimes \ddot{\Gamma}(t), \dddot{\Gamma}(t) \rangle / ||\dot{\Gamma}(t) \otimes \ddot{\Gamma}(t)||^2$ and their first order derivatives $k_s = \dot{k}(t)/||\dot{\Gamma}(t)||$ and $\tau_s = \dot{\tau}(t)/||\dot{\Gamma}(t)||$ with respect to the Euclidean arc-length $s = \int_0^t ||\dot{\Gamma}(t)|| dt$.

The four parameters are invariant and representative for the local shape features and we use them to evaluate trajectory primitives in our descriptor f. The curvature k indicates the variation of moving direction, and the torsion τ indicates the degree that the object moves out of the original plane. An approx-

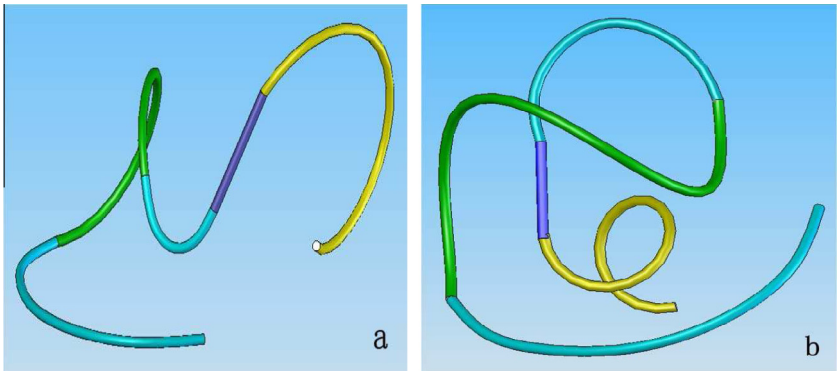

Fig. 6. Different trajectories with the same primitive sequence: BDBAC.

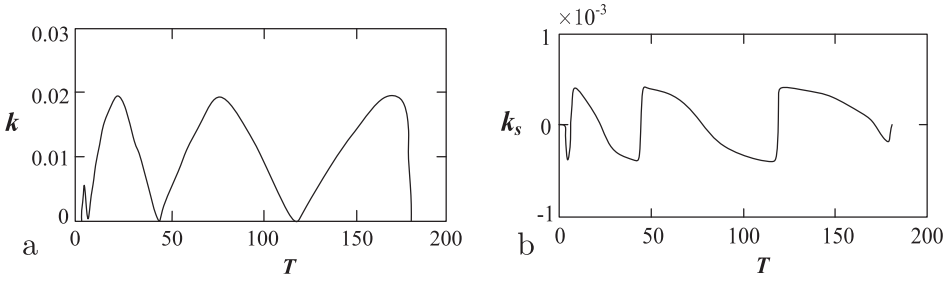

Fig. 7. The signature values k in (a) and k_s in (b) of a trajectory. The zero points of k_s value correspond to the extreme k values.

imate calculation of these parameters [32] is employed to avoid the high order derivatives and reduce the sensitivity to noise. The invariant properties of these parameters have been proved in [32], and the descriptor f calculated from them inherits the invariant properties.

In our feature extraction, the factor λ is set according to the complexity of motion. The parameter k and τ represent the dynamic of motion variation in the trajectory, while k_s and τ_s represent the dynamic of the variations of k and τ, respectively. That is, the salient motion feature includes not only the trajectory curve but also kinematics of motion. As shown in Fig. 7, k_s represents the variation of k, and each zero point in k_s corresponds to an extreme point (top or bottom) in k. In simple motion trajectory, there is only one zero point in k_s within a B primitive (exclude the boundary points). In complex motion, there are more zero points within a primitive. In our approach, we use the average number of zero points in the primitives as the λ value of a given dataset.

After calculating the f values of trajectory points, we uniformly quantize them into β quantization intervals, where β is the total interval number. We set the number β first, and then the f values are partitioned into β intervals, ranging from the minimal to the maximum value, e.g., the intervals of Type B are represented by $\{I_{B1}, I_{B2}, \ldots, I_{B\beta}\}$. With the β intervals, the trajectory points are labeled with respective interval numbers corresponding to their f values. The points are labeled with $\{B1, B2, \ldots, B\beta\}$, $\{C1, \ldots, C\beta\}$, and $\{D1, \ldots, D\beta\}$, for Type B, C and D respectively. Since a trajectory piece within a primitive is continuous, the f values of the trajectory points in this primitive are also continuous in time series. Hence, adjacent points with similar f value have the same label and grouped into a sub-primitive. In this way, a primitive $\mathcal{V}(i)$ is further segmented into a sequence of sub-primitives:

$$\mathcal{V}(i) = \{\mathcal{U}(i,1), \mathcal{U}(i,2), \mathcal{U}(i,3), \ldots, \mathcal{U}(i,j), \ldots, \mathcal{U}(i,m_i)\}, \quad (7)$$

where $\mathcal{U}(i,j)$ is the j-th sub-primitive of the i-th primitive, and m_i is the number of sub-primitives in the i-th primitive. The sub-primitive $\mathcal{U}(i,j)$ with s_{ij} trajectory points $\{p(n_{ij}+k)\}_{k=0}^{s_{ij}-1}$ is defined as:

$$\mathcal{U}(i,j) = \{p(n_{ij}), p(n_{ij}+1), \ldots, p(n_{ij}+k), \ldots, p(n_{ij}+s_{ij}-1)\}, \quad (8)$$

where the point $p(n_{ij})$ is the first point of the sub-primitive $\mathcal{U}(i,j)$, n_{ij} is the order of this point in the whole trajectory, and s_{ij} is the number of points in this sub-primitive.

After quantizing and segmenting the primitives into sub-primitives, motion trajectory is represented hierarchically by a sequence of sub-primitives. For primitive A, there is only one interval, and it cannot be segmented into sub-primitives. In contrast to the primitive representation with two levels including primitive level and point level, the sub-primitive representation is composed of three levels: primitive level, sub-primitive level and point level (Fig. 8). The sub-primitive sequence is longer, since each primitive is represented by a string of sub-primitives. Therefore, more motion information is included in this representation, which is necessary for complex motion characterization. The point-level descriptor is unhandy for large amount of point calculation, while the primitive-level descriptor loses much detailed information. The sub-primitive description represents motion trajectories more flexibly by adjusting the interval number β. The sub-primitive segmentation results for 3D motion trajectory in the ASL dataset is shown in Fig. 1b, where the red triangles represent the boundaries of sub-primitives.

The interval number β is application dependent and should be set adaptively to the complexity of motion trajectories. It should be bigger for complex motions with great variation, and smaller otherwise. In our approach, β is set to give the best accuracy in the dataset. Sub-primitives of different types are shown in Figs. 9–11.

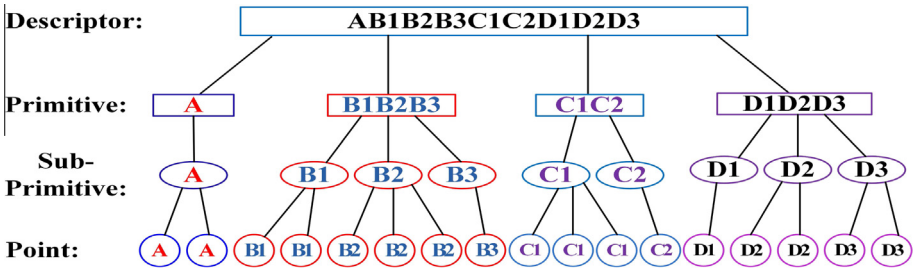

Fig. 8. The hierarchical structure of the F-IGS descriptor.

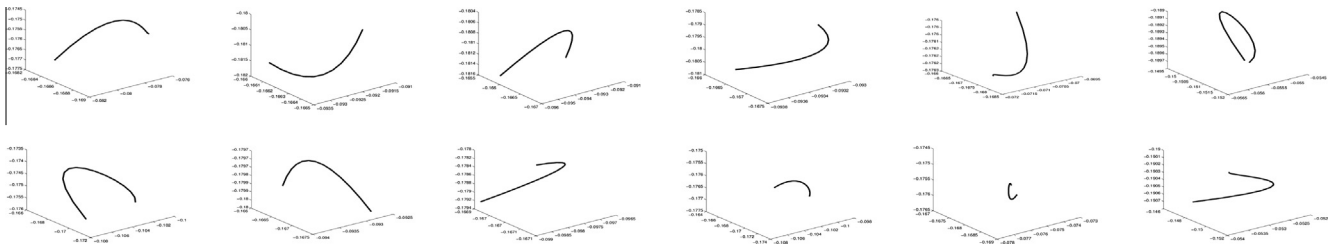

Fig. 9. Sub-primitives of Type B.

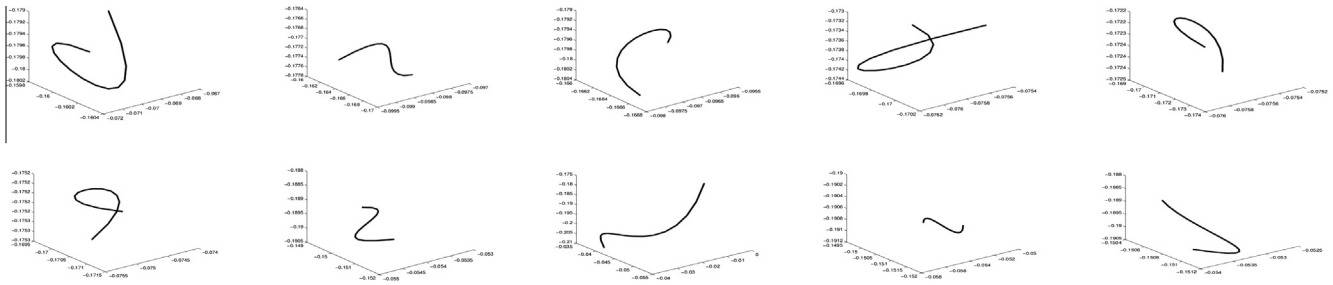

Fig. 10. Sub-primitives of Type C.

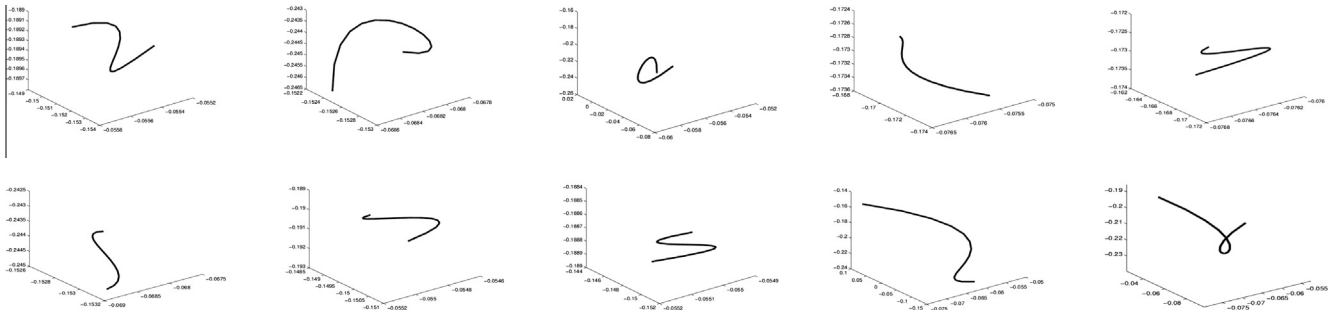

Fig. 11. Sub-primitives of Type D.

5. Motion trajectory recognition

As each motion trajectory is represented as a time series of sub-primitives, the Hidden Markov Model (HMM) is employed to recognize the sub-primitive sequences, which has been widely used in speech recognition [43]. HMM is a statistical model used to describe the characteristics of stochastic process. There are a finite number of states in the HMM, and there is a transition probability from each state to another state. The HMM stays at a certain state each time, and the Markov assumption is that the state of the present time t relies on only the previous one time $t-1$, which is called an order 1 model. At each time arriving a new state, there is an observed output generated according to the observation probability from the new state to the observation. We can only observe the output while the state variables are hidden and unobservable.

In this work, each motion trajectory is represented by a sub-primitive sequence, which are output observations of the hidden states in time series. We model each motion by an HMM, where the moving motion over time is considered as a time series of hidden states. The observed sub-primitive sequences are used to train the HMM model, and then this model is used to predict the unknown sequences. For an HMM model with N states

$S = \{s_1, s_2, \ldots, s_N\}$, there are N^2 transitions as it is possible for any state to transfer to another state (or the original state itself). Denote the hidden state at time t by $q_t \in S$ and the corresponding observation by o_t. The N^2 transition probabilities are described by an $N \times N$ matrix A, called *state transition matrix*:

$$A = \{a_{ij} | a_{ij} = P(q_t = s_j | q_{t-1} = s_i), 1 \leqslant i, j \leqslant N\}, \quad (9)$$

where $0 \leqslant a_{ij} \leqslant 1$ and $\Sigma_{j=1}^{N} a_{ij} = 1$ that the sum probability of transitions from one state to all states (including itself) is 1. The output probabilities from each hidden state to all the possible observations are described by an $N \times M$ matrix B, called *output probability matrix*:

$$B = \{b_j(k) | b_j(k) = P(o_t = v_k | q_t = s_j), 1 \leqslant j \leqslant N, 1 \leqslant k \leqslant M\}, \quad (10)$$

where $V = \{v_1, v_2, \ldots, v_M\}$ is the set of all possible observation symbols. M is the number of different possible observation symbols. The initial state probability distribution π is:

$$\pi = \{\pi_i | \pi_i = P(q_1 = s_i), 1 \leqslant i \leqslant N\}. \quad (11)$$

Thus the HMM model is fully described by a triplet $\lambda = \{A, B, \pi\}$.

To estimate the parameters of HMM, the number of states is set first. Each motion class is modeled by a separate HMM model, and the number of states is set equally to the length of the longest sub-primitive sequence in the training trajectories for this motion class. Once the state number is fixed, the parameters of the triplet λ are initialized to random values. The estimation of the HMM parameters is also called the "decoding process" and the Baum–Welch algorithm is employed using the forward–backward procedure [43]. The Baum–Welch algorithm is a generalized expectation–maximization (EM) algorithm, which iteratively update the parameters in order to better expect the training sequences.

After the model parameters of λ are uniformed, the recognition of a query trajectory is performed by Maximum a Posterior (MAP) inference. From all the HMM models $\lambda_1, \lambda_2, \ldots, \lambda_C$ describing C motion classes, we find the one which best describe the query trajectory by calculation the posterior likelihood. The query trajectory is firstly processed by the proposed parsing method to get a primitive representation, and then quantified to a sub-primitive sequence. The sub-primitive sequence is used as an observation sequence to each HMM. The query trajectory is assigned with a class label c as the HMM which maximizes the likelihood given the query sequence:

$$c = arg \max_{i \in [1,C]} \Sigma_j P(O_{t+1:m} | q_t^i = j, O_{1:t}) P(q_t^i = j, O_{1:m}). \quad (12)$$

This process is calculated using the forward recursion procedure of the Baum–Welch algorithm [43].

6. Experiments

In the experiments, we test the invariant properties of the proposed method to rotation, variation of scales, and reversion of time series. The recognition accuracy of the proposed approach is evaluated with the HDM05 dataset [13], ASL dataset [14] and Berkeley MHAD dataset [15]. The online parsing and recognition of motion trajectory are implemented in the experiments as well.

6.1. Invariant properties

The trajectory usually suffers different kinds of variations in visual tracking, including: view point variation (rendering rotation and wholly scale variation), inconsistent scale variation, reversion of time series. In the practical motion tracking, it is more likely that these variations occur simultaneously. Hence, we combine two or more variations together in experiments to test the invariance of our 3D motion representation. As the invariant properties of the primitive and sub-primitive descriptors to rotation, translation, and wholly scale variation are illustrated in Sections 3 and 4 respectively, we only test the invariant properties to inconsistent scale variation with rotation and the reversion of time series in this experiment. The robustness for partial occlusion is validated in Section 6.2.

This experiment is carried out on the UCI KDD Australian Sign Language (ASL) [14], to test the invariant performance of the proposed representation. We select this dataset because it is the most challenging one, which makes it more difficult for a trajectory descriptor to preserve the invariant properties (as shown in Fig. 15). In the literature, the ASL is widely used to compare the performance. The ASL dataset consists of 95 motion classes with 27 samples in each class. Each trajectory is processed purposely with variations and then we calculate the distance between the sub-primitive representations of the processed trajectory and the original trajectory. All the trajectories in the ASL dataset are tested to get the average result. The distance is calculated via the Dynamic Time Warping (DTW) [29] algorithm which provides reliable measure of the inter-trajectory similarity. A small DTW distance reflects a good invariance of the representation method.

6.1.1. Robustness of inconsistent scale variation and rotation

Inconsistent scale variation means that the scale of trajectories in the three spatial dimensions varies in different ratios. It is easy to perceive that the consistent scale variation in the three dimensions does not affect the shape feature and spatial distribution of the trajectory. However, in general motions, the trajectory scale is probably varied inconsistently while performing the same motion, which makes it more challenging for a representation method to maintain invariance under this complex scale variation. The signature descriptor [32] fails to maintain the invariance in this form of scale variation, which affects the calculation of the trajectory parameters in that descriptor. This problem is also indicated in [33] and is solved by adding global parameters as additional information into the descriptor for better accuracy, but that method is still not invariant to this variation.

In this experiment, we test the proposed descriptor on all the ASL trajectories under this inconsistent scale variation, where the scale factors of different dimensions are randomly and separately selected from 0.5 to 2. Their average Dynamic Time Warping (DTW) [29] distances from the original trajectories are calculated, which reflect the invariant performance of our approach. Moreover, we also combine 3D rotation on the scale varied trajectories to test the invariance of our approach for a complex integrated variation. In the DTW calculation, we define the distance between different labels as 1 and that of the same label as 0, and the distance is divided by the length of the label sequence of trajectory.

The average distances of the tests are listed in Table 1. From the table we can find that the distances of both two descriptors for different variations are very small, which indicates that our method shows good invariance for inconsistent scale variation combined with a rotation. The distances of the primitive representation in the table are caused by the scale variation which modifies the label distinguishing nearby the threshold of noise, which renders incorrect labels of primitives. However, these distances are too small to affect the result significantly. On the other hand, the sub-primitive distances are relatively larger due to two reasons. One is that the

Table 1
Average distance of inconsistent (NC) scale variation with rotation.

Descriptor	NC-Scale	NC-Scale + Rotation
Primitive	0.014	0.014
Sub-primitive	0.033	0.033

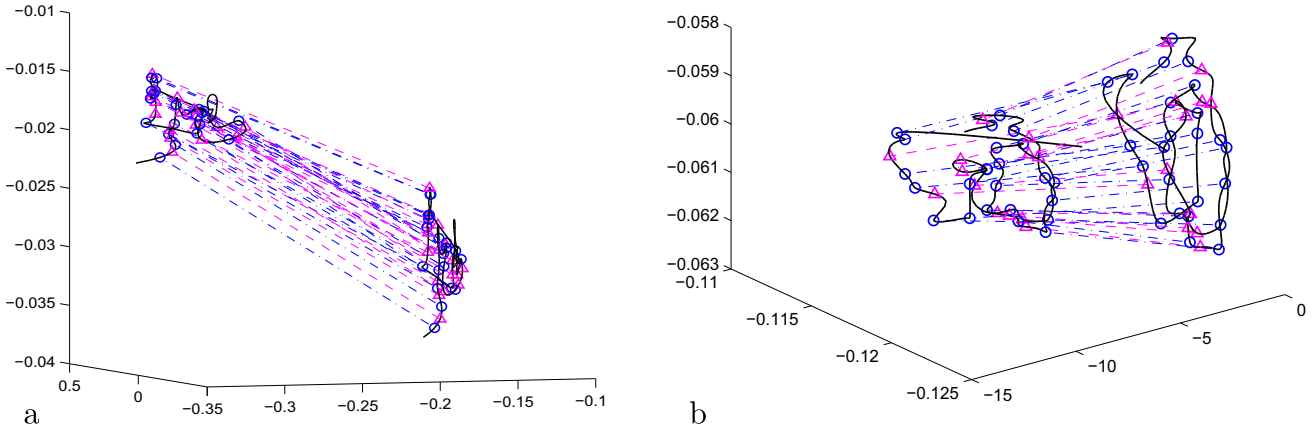

Fig. 12. The segmentation results of the original trajectory and the 2D scaled trajectory. (a) is 'right' and (b) is 'girl'.

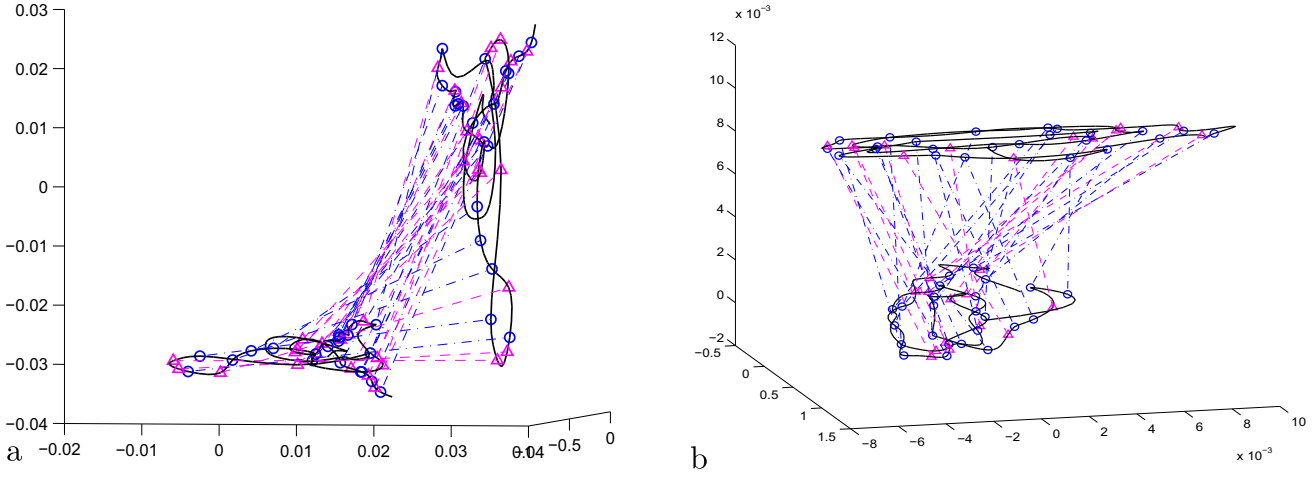

Fig. 13. The segmentation results of the original trajectory and the 2D scaled and rotated trajectory. (a) is 'right' and (b) is 'girl'.

incorrect primitive labels make the sub-primitive labels incorrect as well. The other reason is the incorrect quantization of sub-primitives due to the distortion of primitives. Although most primitives are correctly labeled under the distortion, the f values are modified, which make the sub-primitives to have incorrect scale numbers in their labels. From the table we also find that the integration of rotation with the scale variation does not increase the distances, and our method is invariant to the complex integrated variations. Figs. 12 and 13 show the segmentation results of the original trajectories and the four varied ones. Despite the variations, the segmentation results are still the same.

For the trajectories under inconsistent scale variation and rotation, we further test the intra-class distances of the ASL dataset. In each motion class, the 27 trajectory instances $\{I_j\}_{j=1}^{27}$ are totally inconsistent varied in scale and rotated. And then, we calculate the average distance between each pair of them as the average intra-class distance $D_C = \Sigma_{ij=1}^{27} d_{ij}/27^2$. The average distances $D_1 - D_{95}$ of the 95 classes are summarized in the histograms in Fig. 14. Each column in the histograms represents the number of average distances whose values are between the boundaries aside the column. The distribution of the 95 average distances is represented in the histograms for the four different cases. The histograms (a) and (b) represent the average intra-class distance distributions of the original trajectories represented by the primitive descriptor and sub-primitive descriptor respectively. The distance distributions of the integrated varied trajectories are shown in histograms (c) and (d). For the original histograms (a) and (b), the mean value of primitive and sub-primitive based method are different because of the DTW based distance calculation. In the same trajectory, the number of sub-primitives is much more than that of primitives, and thus the mean distance of sub-primitive based method is larger. This effect also exists in the integrated varied histograms (b) and (d). From the histograms we find that the means of distance distributions of the varied trajectories are bigger than those of the original trajectories for both descriptors. However, the change of the mean values are very small, which indicates that the proposed descriptors are robust to the complex integrated variation.

6.1.2. Invariance of the reversion of time series

This experiment is implemented to test the consistence of the proposed representation method as mentioned in Section 3.2. All the trajectories in the ASL dataset are reversed in time series and represented with the proposed descriptor. The obtained primitive and sub-primitive sequences are then reversed again and matched with the description of the original trajectories. The matching dis-

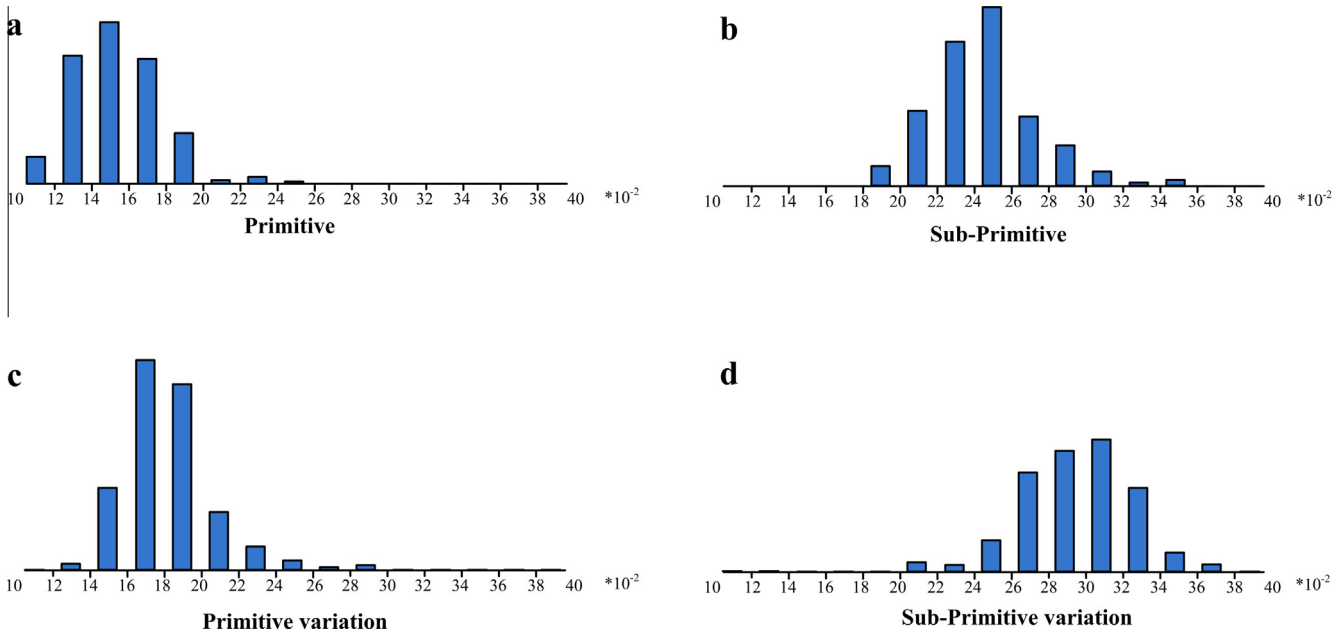

Fig. 14. The histograms of average intra-class distances of primitive descriptor (a), (c) and sub-primitive descriptor (b), (d) for both original trajectories and complex varied trajectories under inconsistent scale variation and rotation.

Table 2
Average distance of reversion.

Descriptor	Reversion distance
Primitive	0.003
Sub-primitive	0.003

tances are calculated by the DTW algorithm, and the matching results are listed in Table 2. The result distances are normalized by respective sequence length, and the results in the table are average distances of the whole dataset. From the table we can see that the average distances of both descriptors are near to zero, which indicate that the corresponding primitives/sub-primitives are rarely wrongly represented in the reversed trajectory. The results of sub-primitive representation are the same as those of the primitive representation, which indicate that the error only occurs in the primitive segmentation, and the sub-primitive segmentation is not affected by the reversion of trajectory.

6.2. Gesture recognition

6.2.1. HDM05 dataset

This experiment is performed on the human motion capture dataset HDM05 [13] which contains 3D motion sequences captured by a motion capturing system at the frame rate 240 Hz. The HDM05 dataset consists of around 100 different motion classes performed by five actors. Most of these classes contain 10–50 different samples for each class, amounting to around 1500 motion clips where each clip contains only one type of motion. The intra-class variation makes the motion recognition a challenging task. However, the sub-primitive sequences of the same class are similar, which is an important clue to distinguish them.

For the test, we use half of the samples for training and the rest for testing. We test our method following the same manner of other methods, and 16 classes of samples are randomly picked and recognized in each time testing. This test is repeated over 50 times. We compare our method with the state-of-the-art methods and the recognition accuracies of different methods are shown in Table 3. The Fourier descriptor (FD) [20], Dynamic Instants and Intervals (DII) 3D-2D projected descriptor [8], CSS + HMM [18] and CDF + HMM [18] have the accuracies no more than 90%, because much motion information is lost in the transformation or 3D–2D projection. The Signature descriptor [32] represents motion trajectory in point level only. The proposed approach achieves a comparable accuracy, which is a good performance on the HDM05 dataset. The IGS [44] approach also has a high accuracy because this descriptor includes the global motion information of trajectories. However, the accuracy of the IGS approach will be much lower if only the primitive descriptor is used without the global information, and the global information is not robust for occlusion. In other words, the description of the sub-primitives significantly improves the discrimination of the descriptor.

Table 3
Recognition accuracy comparison for HDM05 dataset.

Methods	Accuracy (%)
CSS + HMM [18]	43.9
CDF + HMM [18]	52.4
Fourier [20]	59.3
Signature [32]	77.1
IGS [44]	88.7
Integral invariants [35]	90.3
Our method	**91.7**

6.2.2. ASL dataset

In this experiment, motion recognition is carried out on the ASL dataset [14] to test the recognition performance of our method. 3D trajectory samples from different classes are plotted in Fig. 15. The trajectories in this figure are complex and hard to distinguish. In the testing, each time 16 classes of samples are randomly selected, where half of the samples are used for training, and the other half for testing. The experiment is repeated more than 50 times. Other state-of-the-art methods are also compared with our method.

The average recognition performances are summarized in Table 4. We obtain a comparable recognition accuracy with other methods on the ASL dataset. Especially, our method outperforms the most recent state-of-the-art method with integral invatiants

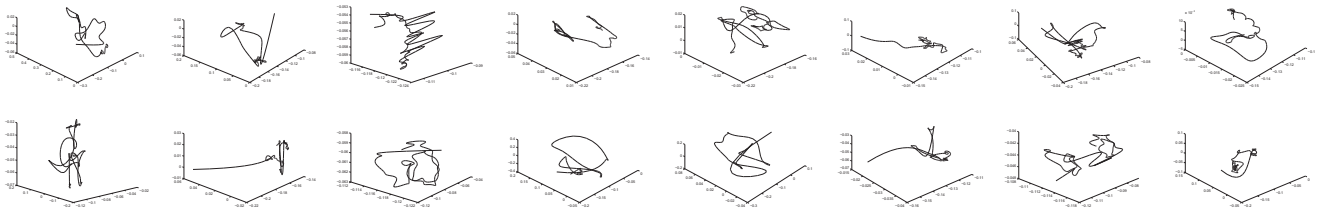

Fig. 15. Sample trajectories from different motion classes of ASL dataset.

Table 4
Recognition accuracy comparison for ASL dataset.

Methods	Accuracy (%)
CSS + HMM [18]	50.4
CDF + HMM [18]	53.5
Fourier [20]	57.9
Signature [32]	79.0
IGS [44]	93.4
Integral Invariants [35]	94.2
Our Method	**94.8**

Table 5
Recognition accuracy with different quantization interval number β.

β	1	2	3	4
Accuracy (%)	86.3	91.9	94.8	94.2

[35] which have the best performance in the literature. It is fair to say that the integral invariants is more robust to noise than the differential invariants [32] which are used in our descriptor. Therefore, our method is more effective to represent the salient features of motion trajectories.

In our experiments, we set the interval number β with different choices and select the one which makes the best accuracy. Here, we test our method using different β on the ASL dataset, and the accuracy of recognition is illustrated in Table 5. In the table, the sub-primitive representation ($\beta > 1$) performs better than the primitive representation ($\beta = 1$) in recognition. As β increases, the recognition accuracy becomes better. However, if β is bigger than a certain number (3 in this case), the accuracy will not increase but drop. Hence, a bigger number β is not always better than a smaller number. In fact, $\beta = 3$ is appropriate for this dataset. Moreover, a too large β increases the computational costs of both trajectory segmentation and model learning.

6.2.3. All classes recognition on MHAD dataset

In this experiment, we test the recognition accuracy of the proposed method with all the classes in Berkeley MHAD dataset [15] compared with other methods. The Berkeley MHAD dataset contains 656 motion sequences, including 11 motion categories performed by 12 subjects with 5 repetitions for each subject. The motion trajectories were captured by an active optical motion capture system with 480HZ sampling rate. This experiment is implemented in the same manner as the HDM05 experiment. Other methods are also tested in the same way to compare the performances with our method. In this experiment, we compare our method with only the representative methods of different kind of descriptors which have good performances in the previous experiments.

The average recognition results are listed in Table 6, which indicates that the best recognition accuracy with 93.4% is obtained by the proposed method. The accuracy is reduced to 91.3% using the IGS descriptor. The Signature descriptor and Fourier descriptor have much worse performances compared with the proposed approach. In summary, the proposed approach has a comparable performance on motion recognition in all classes of the MHAD dataset to state-of-the-art methods.

Table 6
Recognition accuracy comparison for MHAD dataset.

Method	Accuracy (%)
CSS + HMM [18]	59.6
CDF + HMM [18]	64.1
Fourier [20]	71.8
Signature [32]	86.4
IGS [44]	91.3
Integral Invariants [35]	91.9
Our Method	**93.4**

Table 7
Average processing time of recognition.

Methods	Time (ms)		
	Ratio of query trajectory		
	0.6	0.8	1.0
Fourier [20]	N/A	N/A	145
Signature [32]	N/A	N/A	730
Our method	**6.47**	**7.65**	**8.13**

6.2.4. Online trajectory parsing and gesture recognition

In this experiment, we implement the online trajectory parsing and recognize motion trajectories to evaluate our method. From each new frame, a new trajectory point is received and the unfinished motion trajectory is updated in realtime. Since the parsing templates T_A and T_B are used locally for several sequential trajectory points, we update the sub-primitive representation of the online trajectory sequentially. When a new trajectory point is received from a new frame, only the last 4 points of the updated trajectory need to be parsed rather than the whole trajectory, rendering low-latency processing for each frame. After parsing and labeling a new point, the sub-primitive sequence is updated and recognized by the proposed recognition engine online. To evaluate the performance, we fix the ratio of received trajectory in the recognition process. The machine used in our experiment is with a memory of 8 GB 1600 MHz DDR3 and 3.4 GHz Intel quad-core Core i7 processor. The average computing time of parsing and quantizing for a new frame is 0.07 ms. The average processing time of recognition for different ratios of query trajectories is listed in Table 7. From the table we find that the average processing time is very short, and the parsing time is much less with respect to the recognizing time. The total processing time of each frame is less than 10 ms, which is able to support an online motion recognition with a frame rate of 100 fps.

The online recognition accuracies for different ratios of query trajectories are shown in Fig. 16(a) and (b), for the HMD05 and

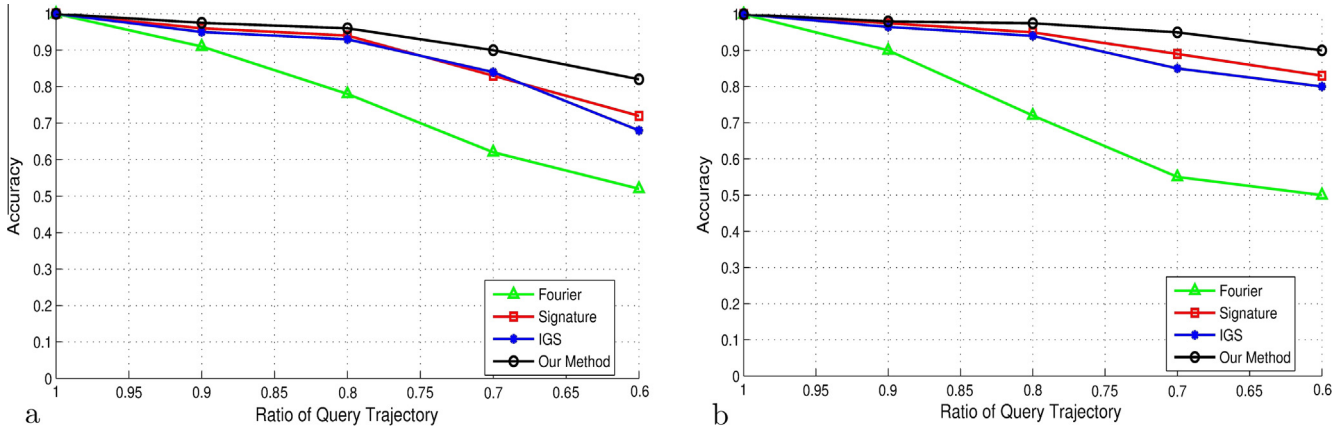

Fig. 16. The recognition accuracy if different ratios. (a) is result on the HDM05 dataset and (b) is result on the MHAD dataset.

Table 8
Recognition accuracy of multiple separate tracks.

Number of tracks	2			4			8		
Ratio of deletion	0.1	0.2	0.3	0.1	0.2	0.3	0.1	0.2	0.3
Fourier [20]	0.886	0.752	0.594	0.827	0.689	0.513	0.741	0.625	0.430
Signature [32]	0.948	0.926	0.877	0.892	0.874	0.835	0.828	0.793	0.738
IGS [44]	0.943	0.905	0.852	0.933	0.897	0.841	0.918	0.882	0.823
Our Method	0.968	0.941	0.906	0.957	0.934	0.892	0.949	0.921	0.874

MHAD datasets respectively. The ratios mean that only part of the whole trajectory is received up to the present frame and recognized online. In this experiment, we also test the accuracy of other state-of-the-art methods in the same form for comparison. The comparison tests of other methods are off-line so that we only evaluate their accuracies as shown in Fig. 16. The figure shows that as the ratio of trajectory decreases, the accuracy of Signature [32], IGS [44] and our method drop relatively small, due to the locality of our representation. The accuracy of the IGS descriptor is a little lower than that of the Signature descriptor because the global information used in IGS is not accurate. The recognition accuracy of the Fourier descriptor drops drastically as the occlusion ratio increases, due to the inaccurate global information in computing this descriptor. In summary, our method performs the best. This result also reflects the robustness of these methods for partial occlusion, which is very important for the real environments of application. The query trajectory is considered as a partial occluded trajectory where the unfinished portion is occluded. Hence, the recognition accuracy of an 80% received trajectory corresponds to that of a 20% occluded trajectory, and our method is more robust for occlusion compared with other methods.

6.2.5. Recognizing trajectory with multiple separate tracks

In real applications, the tracking algorithms may fail when the target moves fast or changes its speed dramatically. This makes the tracking lost or result in several separate tracks. The former condition can be treated as the occlusion, i.e., the lost part of motion is occluded, and we perform the recognition with only partial of gesture trajectory. This is the same as the previous experiment (Section 6.2.4) where the recognition is processed online that only partial trajectory is obtained. The latter condition is a challenge in gesture recognition as well, and we implement the recognition of trajectories with multiple separate tracks to test the robustness of our method to this problem.

In this experiment, each trajectory is randomly segmented into 2, 4 and 8 separate tracks. Then the tracks are processed separately and the trajectory is represented by a few sub-primitive sequences. Different ratios of trajectory points are removed as the gap between tracks and the remained sub-primitive sequences are recognized by the proposed method. The comparison results are shown in Table 8. From the results we can find that the accuracy of the Fourier Descriptor drops significantly as the ratio of deletion and track number increases. The Signature descriptor maintains a better accuracy as the deletion increases, but it drops as the track number increases. The IGS descriptor is not so robust as the signature descriptor for deletion that its accuracy drops significantly as the ratio increases, although its accuracy drops slightly as the track number increases. From the results of our method we can find that the accuracy of our method maintains a relative high score as both the track number and deletion increase, which validates the robustness of our method.

7. Conclusion

In this paper, we have proposed a motion trajectory parsing and segmenting method for gesture representation and recognition. Motion trajectory is first parsed and segmented into primitives and then quantized by shape descriptors into sub-primitives. We select different scale number β in the quantification flexibly for various applications. The HMM model is employed to model motion classes, which makes use of the prior knowledge of the motion dataset. We perform the motion recognition using our motion primitive and sub-primitive representations, based on the Maximum A Posterior criterion. The extensive experimental results validate that the proposed approach is view invariant and robust to scale variation, temporal speed variation, local noise and partial occlusion. The online gesture recognition is implemented as well. Experimental results on the standard datasets show that our algorithm compares favorably with state-of-the-art methods.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (Grant No. 61305020), the Natural

Science Foundation of Jiangsu province, China (Grant No. BK20130316), and the Singapore MoE Tier-2 project MOE2015-T2-2-114.

References

[1] T.B. Moeslund, A. Hilton, V. Kruger, A survey of advances in vision-based human motion capture and analysis, Comput. Vis. Image Underst. 104 (2–3) (2006) 90–126.

[2] B.T. Morris, M.M. Trivedi, A survey of vision-based trajectory learning and analysis for surveillance, IEEE Trans. Circ. Syst. Video Technol. 18 (8) (2008) 1114–1127.

[3] W.B. Chen, G.D. Guo, TriViews: a general framework to use 3D depth data effectively for action recognition, J. Vis. Commun. Image Represent. 26 (2015) 182–191.

[4] A. Psarrou, S. Gong, M. Walter, Recognition of human gestures and behavior based on motion trajectories, Image Vis. Comput. 20 (5–6) (2002) 349–358.

[5] N.C. Kılıboz, U. Güdükbay, A hand gesture recognition technique for human computer interaction, J. Vis. Commun. Image Represent. 28 (2015) 97–104.

[6] E. Bribiesca, A chain code for representing 3D curves, Pattern Recogn. 33 (2000) 755–765.

[7] S. Berretti, A.D. Bimbo, P. Pala, Retrieval by shape similarity with perceptual distance and effective indexing, IEEE Trans. Multimedia 2 (4) (2000) 225–239.

[8] C. Rao, A. Yilmaz, M. Shah, View-invariant representation and recognition of actions, Int. J. Comput. Vision 50 (2) (2002) 203–226.

[9] A. Hervieu, P. Bouthemy, J.P. Le Cadre, A statistical video content recognition method using invariant features on object trajectories, IEEE Trans. CSVT 18 (2008) 1533–1543.

[10] Z. Zhang, T. Tan, K. Huang, An extended grammar system for learning and recognizing complex visual events, IEEE Trans. Pattern Anal. Mach. Intell. 32 (2) (2011) 240–255.

[11] M.S. Ryoo, J.K. Aggarwal, Recognition of composite human activities through context-free grammar based representation, Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition, vol. 2, 2006, pp. 1709–1718.

[12] X.G. Wang, K.T. Ma, G.W. Ng, W.E.L. Grimson, Trajectory analysis and semantic region modeling using nonparametric hierarchical Bayesian models, Int. J. Comput. Vis. 95 (2011) 287–312.

[13] M. Muller, T. Roder, M. Clausen, B. Eberhardt, B. Kruger, A. Weber, Documentation Mocap Database HDM05, Technical Report, No. CG-2007-2, ISSN 1610-8892, Universitat Bonn, June 2007.

[14] UCI KDD ASL Archive, [Online], Available: <http://kdd.ics.uci.edu/databases/auslan2/auslan.html>.

[15] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, R. Bajcsy, Berkeley MHAD: a comprehensive multimodal human action database, in: Proceedings of the IEEE Workshop on Applications on Computer Vision, Clearwater Beach, USA, 2013.

[16] J. Min, R. Kasturi, Activity recognition based on multiple motion trajectories, Proc. International Conference on Pattern Recognition, Cambridge, U.K., vol. 4, 2004, pp. 199–202.

[17] B. Zalik, D. Mongus, N. lukac, A universal chain code compression method, J. Vis. Commun. Image Represent. 29 (2015) 8–15.

[18] F. Bashir, A. Khokhar, D. Schonfeld, View-invariant motion trajectory-based activity classification and recognition, Multimedia Syst. 12 (2006) 45–54.

[19] F. Bashir, A. Khokhar, Curvature scale space based affine invariant trajectory retrieval, in: IEEE International Multi topic Conference, INMIC 2004, Lahore, Pakistan, 2004, pp. 20–25.

[20] P.R.G. Harding, T.J. Ellis, Recognizing hand gesture using Fourier descriptors, in: International Conference on Pattern Recognition, UK, 2004, pp. 286–289.

[21] C. Cattani, Shannon Wavelets for the Solution of Integrodifferential Equations, Math. Probl. Eng. 200 (2010) 1.

[22] F.S. Cohen, Z. Huang, Z. Yang, Invariant matching and identification of curves using B-splines curve representation, IEEE Trans. Image Process. 4 (1) (1995) 1C10.

[23] S.Y. Chen, Q. Guan, Parametric shape representation by a deformable NURBS model for cardiac functional measurements, IEEE Trans. Biomed. Eng. 58 (3) (2011) 480C487.

[24] J.W. Hsieh, S.L. Yu, Y.S. Chen, Motion-based video retrieval by trajectory matching, IEEE Trans. Circ. Syst. Video Technol. 16 (3) (2006) 396–409.

[25] H. Suk, B.K. Sin, S.W. Lee, Hand gesture recognition based on dynamic Bayesian network framework, Pattern Recogn. 43 (2010) 3059–3072.

[26] M.F. Abdelkadera, W. Abd-Almageeda, A. Srivastavab, R. Chellappaa, Silhouette-based gesture and action recognition via modeling trajectories on Riemannian shape manifolds, Comput. Vis. Image Underst. 115 (3) (2011) 439–455.

[27] N. Oliver, B. Rosario, A. Pentland, A Bayesian computer vision system for modeling human interactions, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 831–843.

[28] S. Hongeng, R. Nevatia, F. Bremond, Video-based event recognition: activity representation and probabilistic recognition methods, Comput. Vis. Image Underst. 96 (2) (2003) 129–162.

[29] E. Keogh, M. Pazzani, Scaling up dynamic time warping for datamining applications, in: Proc. of ACM SIGKDD, 2000.

[30] M. Vlachos, G. Kollios, D. Gunopulos, Elastic translation invariant matching of trajectories, Mach. Learn. 58 (2005) 301–334.

[31] L.J. Latecki, V. Megalooikonomou, Q. Wang, R. Lakaemper, C.A. Ratanamahatana, E. Keogh, Elastic partial matching of time series, in: Proc. Principles and Practice of Knowledge Discovery in Databases, Porto, Portugal, 2005.

[32] S.D. Wu, Y.F. Li, Flexible signature descriptions for adaptive motion trajectory presentation, perception and recognition, Pattern Recogn. 42 (1) (2009) 194–214.

[33] J.Y. Yang et al., Mixed signature: an invariant descriptor for 3D motion trajectory perception and recognition, in: Mathematical Problems in Engineering, Special Issue: Information and Modeling in Complexity, 2012.

[34] J.Y. Yang, J.S. Yuan, Y.F. Li, Flexible trajectory indexing for 3D motion recognition, in: Proceedings of the IEEE Winter Conference on Applications of Computer Vision, Hawaii, USA, 2015.

[35] Z.P. Shao, Y.F. Li, Integral invariants for space motion trajectory matching and recognition, Pattern Recogn. 48 (2015) 2418–2432.

[36] S. Manay, D. Cremers, B.W. Hong, A.J. Yezzi, S. Scatto, Integral invariants for shape matching, IEEE Trans. Pattern Anal. Mach. Intell. 28 (10) (2006) 1602–1618.

[37] Jianyu Yang, Junsong Yuan, Hongxing Wang, Youfu Li, Jianyang Liu, Invariant multi-scale descriptor for shape representation, matching and retrieval, Comput. Vis. Image Underst. 145 (2016) 43–58.

[38] Jianyu Yang, Haoran Xu, Metric learning based object recognition and retrieval, Neurocomputing 190 (2016) 70–81.

[39] J. Wang, L. Yin, X. Wei, Y. Sun, 3D facial expression recognition based on primitive surface feature distribution, Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition, vol. 2, 2006.

[40] Y. Li, Y. Wang, M. Case, S.F. Chang, P.K. Allen, Real-time pose estimation of deformable objects using a volumetric approach, in: Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, 2014, pp. 1046–1052. September.

[41] G. Terejanu, Crib Sheet, Linear Kalman Smoothing, Web-tutorial, Department of Computer Science and Engineering, University at Buffalo, Buffalo, New York, 2008. Available: <http://www.cse.sc.edu/~terejanu/files/tutorialKS.pdf>.

[42] M. Müller, T. Röder, M. Clausen, Efficient content-based retrieval of motion capture data, ACM Trans. on Graphics, vol. 24(3), 2005, pp. 677–685.

[43] L.R. Rabiner, A tutorial on hidden markov models and selected applications in speech recognition, Proc. of the IEEE, 77(2), 1989, pp. 257–285. February.

[44] J.Y. Yang, Y.F. Li, K.Y. Wang, Invariant trajectory indexing for real time 3D motion recognition, in: Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, CA, USA, 2011, pp. 3440–3445.