Action Recognition Using Rate-Invariant Analysis of Skeletal Shape Trajectories

Boulbaba Ben Amor, Jingyong Su, and Anuj Srivastava

Abstract—We study the problem of classifying actions of human subjects using depth movies generated by Kinect or other depth sensors. Representing human body as dynamical skeletons, we study the evolution of their (skeletons') shapes as trajectories on Kendall's shape manifold. The action data is typically corrupted by large variability in execution rates within and across subjects and, thus, causing major problems in statistical analyses. To address that issue, we adopt a recently-developed framework of Su et al. [1], [2] to this problem domain. Here, the variable execution rates correspond to re-parameterizations of trajectories, and one uses a parameterization-invariant metric for aligning, comparing, averaging, and modeling trajectories. This is based on a combination of transported square-root vector fields (TSRVFs) of trajectories and the standard Euclidean norm, that allows computational efficiency. We develop a comprehensive suite of computational tools for this application domain: smoothing and denoising skeleton trajectories using median filtering, up- and down-sampling actions in time domain, simultaneous temporal-registration of multiple actions, and extracting invertible Euclidean representations of actions. Due to invertibility these Euclidean representations allow both discriminative and generative models for statistical analysis. For instance, they can be used in a SVM-based classification of original actions, as demonstrated here using MSR Action-3D, MSR Daily Activity and 3D Action Pairs datasets. Using only the skeletal information, we achieve state-of-the-art classification results on these datasets.

Index Terms—Action recognition, Riemannian geometry, manifold trajectories, depth sensors, skeletal data

1 Introduction

Automated analysis of human behavior through visual data has attracted a tremendous interest in the computer vision community. This is due to its huge potential in a wide spectrum of areas, such as human-machine interaction, psychology [3], surveillance security, healthcare and social assistance, video games, and so on. Beyond verbal or vocal communication data, visual data forms one of the most important cues in developing systems for understanding human behavior [4]. It applications range from tracking daily activities to classifying emotional states, as well as detecting abnormal and suspicious activities. Earlier studies in such behavior analysis used videos produced by conventional cameras [5], [6], [7], [8], [9], [10]. These papers analyzed videos in order to exploit visual cues relating to the human 'form' (or silhouette) in temporal evolutions of images and to capture its dynamics (e.g., using optical flows). However, real-life applications of this approach encounter several difficulties, coming from self-occlusions, pose variations, and the complexity of both imaging environments and activities themselves. Additionally, there are two major challenges that also negatively affect video-based inferences, (i) the *temporal variability* caused by differences in execution rates and periods of actions [8], including arbitrary starting/ending observation times, and (ii) the *spatial variability* due to interpersonal differences when performing similar gestures (e.g., waving hands but at different elevations and different sweeps).

The recent growth in cheap and mobile range-imaging sensors [11] has pushed forth a new research direction where one explores an additional source of information called *depth*. An important advantage of this data is that it is relatively easy to remove background, and to isolate and track human body in this situation. These were difficult issues when dealing with videos from conventional cameras. Moreover, the depth maps are independent of human appearance (textures) and provide a more complete human 'form' relative to the silhouette information used in the past. They can also help in mitigating the issue of pose variability due to availability of 3D registration methods and/or existence of pose-independent features.

The relevant question in using depth maps is: How to exploit the depth information in an efficient and robust way? The different possibilities include: (1) directly processing depth data as images using tools from image processing, (2) forming *point clouds* and extracting geometric features (normal, curvature, etc.) from these clouds for classifications, or (3) forming *skeletons* [12] that summarize human body using a set of connected, articulate joints. A temporal sequence of skeletons, one for each image frame, forms an action, i.e., a spatiotemporal signature of a moving body changing its shape in time. The choice of the right representation—images, point sets or skeletons—and the right space/metric to analyze actions is an important issue. Another important conceptual issue is the choice of statistical models to capture variability of body parts and their dynamics, within and across action classes.

- *B. Ben Amor is with the Institut Mines-Télécom/Télécom Lille, CRIStAL (UMR 9189), Lille, France. E-mail: benamor@telecom-lille.fr.*
- *A. Srivastava is with the Department of Statistics, Florida State University, Tallahassee, FL. E-mail: anuj@stat.fsu.edu.*
- *J. Su is with the Department of Mathematics & Statistics, Texas Tech University, Lubbock, TX 79409. E-mail: jingyong.su@ttu.edu.*

Manuscript received 26 Mar. 2014; revised 10 Nov. 2014; accepted 21 May 2015. Date of publication 31 May 2015; date of current version 9 Dec. 2015. Recommended for acceptance by I. Laptev.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPAMI.2015.2439257

In this work we choose the skeletal representation of human body and use Kendall's shape framework to characterize shapes of individual skeletons. The specification of a shape manifold and the corresponding metric enables us to compare arbitrary skeletons in terms of their shapes, using geodesic lengths. Additional shape tools, such as computation of sample mean and sample covariance statistics, and the transfer of deformations using parallel transports also becomes straightforward. Then, we define an action as a sequence of skeletal shapes or, more precisely, a parameterized trajectory on this manifold of skeleton shapes. Since actions are represented by trajectories, the process of comparing, summarizing, classifying, and modeling actions requires a specification of the space of such trajectories and the choice of metric on this space. Before we present details of our approach, we briefly discuss some related works and their strengths and weaknesses.

1.1 Related Work

The recent focus on analysis of human behavior has resulted in a variety of approaches. Rather than covering all ideas exhaustively, we direct interested readers to some recent surveys [5], [6], [7], [9] that together overview this research area. Instead, we focus our attention on papers that use depth data, especially those that explicitly study temporal modeling and time-warping solutions, and use motion features from dynamic flows for action classification. They are most closely related to our approach. We also point out that this subarea of depth-based action recognition has benefitted greatly from recent availability of public RGB-Depth datasets, like the MSR Action 3D [13], MSR Daily Activity 3D [14], 3D action pairs dataset [15], Cornell Activity Dataset [16] and others. Some of these datasets are also used in the experimental studies described later.

1.1.1 Action Recognition Using Depth Sensors

Using 3D point clouds, Li et al. [13] proposed action graphs to model the dynamics, where the nodes of a graph are salient postures detected using a bag-of-3D points (BOPs). They use a Gaussian mixture model (GMM) to capture the statistical distribution of the postures and demonstrate the superiority of their approach over the 2D silhouette-based approach. Yang et al. [17] introduced a depth motion map (DMM) obtained by projecting a sequence of depth maps to three orthogonal planes. Then, they computed histograms of oriented gradients (HoG) from DMMs to describe the shape and the motion information. Taking a different approach, Wang et al. [14], [18] extract skeletal data directly from a kinect sensor and used an actionlet ensemble model trained to represent actions. More recently, Oreifej and Lu [15] presented a new descriptor to capture both the shape and the motion features jointly from a sequence of depth images. This descriptor is based on histograms of (orientations of) surface normals. In [19], Yang and Tian employ the differences of skeleton joints, called EigenJoints, to describe the spatial posture, motion, and overall dynamics across an action. They use a notion of key frame selection to limit the impact of noise and to reduce the computational cost. In [20], Ellis et al. train a logistic-regression-based classifier to determine distinctive canonical poses and to robustly recognize the actions. The key part of their work, termed a trade-off between accuracy and latency, is how to overcome the ambiguity in initial poses of an action. Devanne et al. [21] mapped sequences of skeletons to square-root velocity function, introduced in [22] for shape analysis of curves, and performed optimal registration of curves. Xia et al. [23] used histograms of 3D joint locations (HOJ3D) to describe postures along a 3D sequence. Furthermore, they obtained a set of visual words (prototypical poses of actions) using LDA and modeled the temporal evolution of those visual words by discrete hidden Markov models (HMMs). These methods differ in terms of the nature of data used: depth-maps, 3D point clouds or skeletal data. While the depth-map-based approaches [17] take advantage of image analysis tools, the approaches based on point clouds [15] rely on more geometric tools. Similarly, the approaches using skeletal data adapt existing shape analysis tools [9] to 3D skeletons [21].

1.1.2 Temporal Modeling and Warping of Actions

An important outstanding issue is how to compare actions while being invariant to rates of executions? This issue was first highlighted by Veeraraghavan et al. [8], [24] who showed that different rates of executions of the same activity can greatly decrease recognition performance, if ignored. For instance, similar actions performed by the same or different actors can have different execution rates and different starting points in a periodic process (cycle). Veeraraghan et al. [8] and Abdelkader et al. [10] used the dynamic time warping (DTW) for temporal alignment before comparing trajectories of shapes of planar curves that represent silhouettes in videos. The same solution (DTW) has been also used by Gritai et al. [25] to compensate for unsynchronized actions. In these works, action realizations have been viewed as trajectories of anatomical landmarks tracked in 2D videos. The DTW is then used to find an optimal alignment between the two time-series, shape trajectories α_1 and α_2 in this case, according to $\operatorname{argmin}_\gamma \int_0^1 d_s(\alpha_1(t), \alpha_2(\gamma(t)))dt$, where d_s is a metric defined on the space of shape representations. However, this cost function used for alignment is not a proper metric; it is not even symmetric. That is, the optimal alignment of sequence A to sequence B may not be the same as the alignment of sequence B to sequence A. This makes it difficult to interpret the results and generate consistent inferences.

Another solution to temporal misalignment is *exemplar-based modeling*, where each action is represented by a sequence of prototype postures (or features, more generally) and the dynamics of the action are trained as a set of transitions between these prototypes. An example of this idea is *hidden Markov model*, used by Abdelkader et al. [10] for activity recognition using shapes of silhouettes extracted from 2D videos. They used an unsupervised clustering of the training shapes and the centers of these clusters to build prototype shapes. Then, each action is modeled by a Markov process that transitions across these prototypes. Lv and Nevatia [26] also employed an HMM to capture transitions between prototype skeletons. In [27], Turaga et al. modeled time-series data using the auto-regressive and moving average (ARMA) model. Then, representing the observability matrix of this system

by an element of Grassmann manifold, they used an intrinsic metric on this manifold to compare and analyze different activities. Similar framework has been adapted by Slama et al. [28] for action recognition from Kinect.

Most existing approaches are limited to previously-segmented actions and activities. The processing of high-level unsegmented (complex) activities is a challenging task and still a distant goal [29]. One solution is to take a two-step approach: first locate (or detect) discriminative key-frames or action sub-sequences and then perform recognition. The main idea is to restrict the action modeling to a sparse local representations (called key-frames) and compute local features instead of using whole videos. For example, [29] proposed to learn and select a sparse collection of key-frames. In [30] the authors used both pose (3D joints) and kinematic features (velocity and acceleration) computed on discriminative key-frames for action recognition from 3D skeletal sequences.

1.2 Contributions and Paper Organization

The main contributions of this paper are:

1) It develops a suite of geometrical tools for processing human skeleton data provided by a depth camera, such as the MS Kinect, as both static (individual skeletons) and dynamic (skeletal trajectories) using the geometry of Kendall's shape manifold [31].
2) It adapts the framework presented in Su et al. [1], [2] to this application area, and designs a fully automated and efficient pipeline for going from depth camera output to action recognition.
3) It provides an extensive set of experiments and discussions on three publicly available datasets: MSR Action 3D, Pairs Actions 3D and MSR Daily Activity 3D, as well as a comparisons with previous studies on the same datasets.

The rest of the paper is organized as follows: In Section 2, we describe the mathematical framework for static and dynamic 3D shape analysis of skeletal data. Section 3 presents a collection of geometric tools for treating both individual skeletons and their temporal evolutions. Two strategies for action classification are presented and discussed in Section 4, and experimental results are reported in Section 5. Finally, the paper ends with some conclusions in Section 6.

2 Mathematical Framework

We start by outlining a mathematical framework for analyzing shapes of human skeletons, and their temporal evolutions. Since human skeletons are characterized by sets of registered points (or landmarks), it is natural to use Kendall's approach to shape analysis here. Let $X \in \mathbb{R}^{n\times 3}$ represents a skeleton or a configuration of n landmarks in $\mathbb{R}^3$. Note that skeleton data comes with both joint locations and orientations [32], but we will discard the orientation information and focus only on the joint locations in this paper. We are interested in analyzing shapes of skeletons, i.e analysis should be invariant to rotations, translations, and global scaling of configurations. This setup is described next.

2.1 Shape Space of Skeletons

To remove translation, we force X to satisfy: $\sum_{i=1}^{n} X_{i,j} = 0$, for $j = 1, 2, 3$. Similarly, to remove the scale we assume that $\|X\|_F = \sqrt{\sum_{i,j} X_{i,j}^2} = 1$. Following [31], we introduce the notion of Helmert sub-matrix, a $(n-1)\times n$ sub-matrix of a Helmert matrix, to perform centering of configurations. The first row of the Helmert matrix has all elements equal to $1/\sqrt{n}$ and the remaining rows are orthogonal to this first row (and to each other). These remaining rows form the desired sub-matrix H. The jth row of $H \in \mathbb{R}^{(n-1)\times n}$ is given by: $(h_j, \ldots, h_j, -jh_j, 0, \ldots, 0)$, with $h_j = -\{j(j+1)\}^{-1/2}$, $j = 1, 2, \ldots, n-1$. For example, for $n = 3$,

$$H = \begin{bmatrix} -1/\sqrt{2} & 1/\sqrt{2} & 0 \\ -1/\sqrt{6} & -1/\sqrt{6} & 2/\sqrt{6} \end{bmatrix}.$$

For any $X \in \mathbb{R}^{n\times 3}$, the product $HX \in \mathbb{R}^{(n-1)\times 3}$ represents the Euclidean coordinates of the centered configuration.

Let $\mathcal{C}_0$ be the set of all such centered configurations of n landmarks in $\mathbb{R}^3$, i.e., $\mathcal{C}_0 = \{HX \in \mathbb{R}^{(n-1)\times 3} | X \in \mathbb{R}^{n\times 3}\}$. $\mathcal{C}_0$ is a $3(n-1)$ dimensional vector space and can be identified with $\mathbb{R}^{3(n-1)}$. Since we are also interested in removing the scale variability, we define the pre-shape space to be: $\mathcal{C} = \{X \in \mathcal{C}_0 | \|X\|_F = 1\}$. $\mathcal{C}$ is a unit sphere in $\mathbb{R}^{3(n-1)}$ and, thus, is $(3n-4)$ dimensional. The tangent space at any pre-shape X is: $T_X(\mathcal{C}) = \{V \in \mathcal{C}_0 | \text{trace}(V^T X) = 0\}$.

While we have removed translation and scaling from our representation, we still need to account for the rotation variability. For any $X \in \mathcal{C}$, we define an equivalence class: $[X] = \{XO | O \in SO(3)\}$, the set of all rotated versions of X. The set of all such equivalence classes, $\mathcal{S} = \{[X] | X \in \mathcal{C}\} = \mathcal{C}/SO(3)$, is called the *shape space* of skeletons having n landmarks. We are interested in analyzing shape trajectories as curves in the tangent spaces of $\mathcal{S}$. For this purpose, we need to define tangent spaces and to specify parallel translations of vectors across tangent spaces. The tangent space at any shape $[X]$ is:

$$T_{[X]}(\mathcal{S}) = \{V \in \mathcal{C}_0 | \text{trace}(V^T X) = 0, \quad \text{trace}(V^T XA) = 0\}, \tag{1}$$

where A is any 3×3 skew-symmetric matrix. The first condition makes V tangent to $\mathcal{C}$ and the second makes V perpendicular to the rotation orbit and, together they force V to be tangent to the shape space $\mathcal{S}$ at $[X]$. Fig. 1 shows the spherical structure of $\mathcal{C}$ and introduces some notation: α_1 and α_2 denote curves (or trajectories) on the manifold; the tangent space attached to $\mathcal{C}$ at R termed $T_R(\mathcal{C})$; a geodesic path connecting arbitrary points on α_1 and α_2; and a shooting vector V on R towards an arbitrary point $\alpha_2(t)$. In view of the spherical structure of $\mathcal{C}$, the expressions for the exponential map and its inverse are well known, and can be easily adapted to $\mathcal{S}$.

1. *Exponential map.* The exponential map is given by: for any $V \in T_{[X]}(\mathcal{S})$,

$$\exp_{[X]}(V) = \left[\cos(\theta)X + \frac{\sin(\theta)}{\theta}V\right], \tag{2}$$

with $\theta = \sqrt{\langle V, V\rangle} = \sqrt{\text{trace}(VV^T)}$.

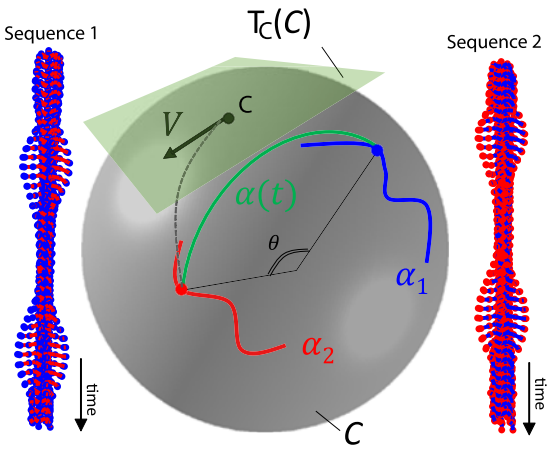

Fig. 1. Pre-shape space $\mathcal{C}$, the trajectories α_1 and α_2, the geodesic $\alpha(t)$ connecting arbitrary points on α_1 and α_2, the tangent space at a shape R, and the shooting vector V from R towards a point on $\mathcal{C}$.

2. *Inverse exponential*. The inverse exponential map is given by:

$$\exp^{-1}_{[X]}([Y]) = \frac{\theta}{\sin(\theta)}(YO^* - \cos(\theta)X), \tag{3}$$

with $\theta = \cos^{-1}(\mathrm{trace}(X(YO^*)^T))$. Here O^* is the optimal rotation of Y that aligns it with X: $O^* = \mathrm{argmin}_{O\in SO(3)} \|X - YO\|_F^2$.

3. *Geodesic path*. Also, assuming standard Riemannian metric on $\mathcal{S}$, the geodesic between any two configurations $[X], [Y] \in \mathcal{S}$ is given by $[\alpha(t)]$ where

$$\alpha(t) = \frac{1}{\sin(\theta)}(\sin((1-t)\theta)X + \sin(t\theta)YO^*), \tag{4}$$

where $\theta = \cos^{-1}(\langle X, YO^* \rangle)$ and O^* is the optimal rotation as stated in the previous item. This θ is also the geodesic distance between $[X]$ and $[Y]$ in the shape space $\mathcal{S}$, i.e.,

$$d_s([X],[Y]) = \cos^{-1}(\langle X, YO^* \rangle)\ . \tag{5}$$

4. *Parallel translation*. For shapes $[X]$ and $[Y]$, and a tangent vector $V \in T_{[X]}(\mathcal{S})$, We need the parallel transport of V to $[Y]$, along a geodesic connecting $[X]$ and $[Y]$. It is given by solving a differential equation ([31]), usually by Euler's method over multiple time steps. However, a computationally simpler approximation results when using a single time step and is given by:

$$V_{[X]\to[Y]} = V - \frac{2\langle V, YO^* \rangle}{\|X + YO^*\|_F^2}(X + YO^*)\ . \tag{6}$$

The methodology used in this paper is based on transporting tangent vectors from arbitrary points in $\mathcal{S}$ to a reference shape termed $[R] \in \mathcal{S}$. Consequently, it is useful to express any transported vector $V_{[X]\to[R]}$ using a fixed orthonormal basis of $T_{[R]}(\mathcal{S})$, so that the ensuing analysis will only involve the coefficients with respect to this basis. This creates a need for an orthonormal basis of the tangent space $T_{[R]}(\mathcal{S})$.

Let $\{E_i \in \mathbb{R}^{n\times 3}, i = 1, 2, \ldots, 3n\}$ denote the canonical Euclidean basis of $\mathbb{R}^{n\times 3}$. Then, the set $\{\tilde{E}_i = HE_i \in \mathbb{R}^{(n-1)\times 3}, i = 1, 2, \ldots, 3n\}$ denotes an overcomplete basis of $\mathcal{C}_0$. Let $\{A_1, A_2, A_3\}$ form an orthonormal basis for all 3×3 skew-symmetric matrices. Define new basis elements for $i = 1, 2, \ldots, 3n$ using:

$$F_i = \tilde{E}_i - \langle \tilde{E}_i, R\rangle R - \sum_{j=1}^{3} \frac{\langle \tilde{E}_i, RA_j\rangle RA_j}{\langle RA_j, RA_j\rangle}\ .$$

Perform Gram-Schmidt on $\{F_i\}$ to form the set $\mathcal{F}$. $\mathcal{F}$ has $3n-7$ basis elements and forms an orthonormal basis for the tangent space $T_{[R]}(\mathcal{S})$. Now we can represent any element $V \in T_{[R]}(\mathcal{S})$ with respect to this basis as a vector in $\mathbb{R}^{3n-7}$.

In the rest of the paper, the reference point R will be simply be a pre-selected skeleton representing the neutral pose.

2.2 Space of Skeleton Trajectories

The next item is to represent actions as sequences of skeletal shapes in a precise mathematical form. Let α denote the observation of an action over the time interval $[0, 1]$. For each time $t \in [0, 1]$, the skeleton at time t has a shape denoted by $\alpha(t) \in \mathcal{S}$. In other words, we are focusing only on the shape of the skeleton at any time, ignoring its scale, position, and pose. The space of such trajectories can be generally written as $\mathcal{S}^{[0,1]}$ (although we will restrict to absolutely continuous trajectories later on). Note that the parametrization of a trajectory denotes the rate at which the corresponding action was performed. To explain that idea further, let $\tilde{\Gamma}$ denote the set of all monotonically-increasing functions on $[0, 1]$ such that they take value 0 on the left corner and 1 on the right corner. Further, let Γ be the set of all diffeomorphisms from $[0, 1]$ to itself with the same boundary conditions. It can be shown that Γ is dense in $\tilde{\Gamma}$, a technical condition that we will exploit later. The elements of $\tilde{\Gamma}$ play the role of execution rates of different actions. For instance, for any action α and $\gamma \in \tilde{\Gamma}$, the composition $\alpha \circ \gamma$ denotes the same action as α but performed at a new temporal rate dictated by γ.

In order to perform action classification, we can take one of two approaches—metric-based or model-based. In a model-based approach one defines a statistical model, using generative or discriminative ideas, and uses it to classify future actions. In the metric-based approach, the main ingredient is the choice of a distance, with appropriate properties, that can be combined with a classifier, such as the nearest-neighbor (NN) classifier or SVM, for performing classification. Taking a metric approach, we seek a metric on the space $\mathcal{S}^{[0,1]}$ that can be used to distinguish between different activities. A natural choice will be to use the shape metric d_s to induce a metric on $\mathcal{S}^{[0,1]}$ according to:

$$d_{ne}(\alpha_1, \alpha_2) = \int_0^1 d_s(\alpha_1(t), \alpha_2(t))dt\ . \tag{7}$$

While this quantity, d_{ne} *(ne: non-elastic distance)*, is a proper distance on $\mathcal{S}^{[0,1]}$, a major limitation here is that is dependent on the parameterizations of trajectories α_1 and α_2. For instance, this value will change if we compare $\alpha_1 \circ \gamma$ and

$\alpha_2 \circ \gamma$ for some $\gamma \in \tilde{\Gamma}$. From the perspective of classification, an action is invariant to its execution rate and, therefore, we need a distance between trajectories that is invariant to their arbitrary re-parameterizations. We will use the distance proposed in [1], as follows. For a trajectory α, define a transported square-root vector field (TSRVF) $h_\alpha : [0,1] \to T_{[R]}(\mathcal{S})$ according to the expression:

$$h_\alpha(t) = \frac{\dot{\alpha}(t)_{\alpha(t)\to[R]}}{\sqrt{|\dot{\alpha}(t)|}} \ , \tag{8}$$

where $\dot{\alpha}(t) \in T_{\alpha(t)}(\mathcal{S})$ is the velocity vector along the trajectory at time t, $\dot{\alpha}(t)_{\alpha(t)\to[R]}$ is its transport from $\alpha(t)$ to $[R]$ along a geodesic, and $|\cdot|$ is the norm according to the Riemannian metric on $\mathcal{S}$. Given a TSRVF h_α and an initial point $\alpha(0)$, one can reconstruct the original trajectory α by solving the ODE: $\frac{d\alpha(t)}{dt} = |h_\alpha(t)|(h_\alpha(t))_{[R]\to\alpha(t)}$. Also, we can represent $h_\alpha(t)$, for each t, with respect to the basis $\mathcal{F}$ and obtain a time-varying coefficient vector $c_\alpha(t) \in \mathbb{R}^{3n-7}$. Any comparisons of trajectories involving TSRVFs can thus be performed in terms of the coefficients. Algorithm 1 outlines the steps to compute the T-SRVF h_α of a given discrete trajectory α.

Algorithm 1. Computing h_α for a discrete trajectory α

Given a trajectory $\alpha(t)_{t=1,..,T}$ and $[R] \in \mathcal{S}$,
$t \leftarrow 1$,
while $(t < T)$ **do**
  Compute $\dot{\alpha}(t) = T \exp^{-1}_{\alpha(t)}(\alpha(t+1))$,
  Parallel translate $\dot{\alpha}(t)_{\alpha(t)\to[R]}$ (Eqn. (6)),
  Compute $h_\alpha(t) = \dot{\alpha}(t)_{\alpha(t)\to[R]}/\sqrt{|\dot{\alpha}(t)|}$,
  $t \leftarrow t+1$,
end while
Set $\dot{\alpha}(T)$ to be $\dot{\alpha}(T-1)_{\alpha(T-1)\to\alpha(T)}$ (Eqn. (6))
Compute $h_\alpha(T) = \dot{\alpha}(T)_{\alpha(T)\to[R]}/\sqrt{|\dot{\alpha}(T)|}$
return h_α.

Let $\|h_\alpha\| = \sqrt{\int_0^1 |h_\alpha(t)|^2 dt}$ denote the $\mathbb{L}^2$ norm of TSRVFs viewed as curves in the tangent space $T_{[R]}(\mathcal{S})$. As described in [1], an important property of TSRVF is that: $\|h_{\alpha_1} - h_{\alpha_2}\| = \|h_{\alpha_1\circ\gamma} - h_{\alpha_2\circ\gamma}\|$ for all $\gamma \in \tilde{\Gamma}$. This sets up a parameterization-invariant distance between trajectories on $\mathcal{S}$ according to:

$$\begin{aligned} d_\alpha(\alpha_1,\alpha_2) &= \inf_{\gamma_1,\gamma_2\in\tilde{\Gamma}} \|h_{\alpha_1\circ\gamma_1} - h_{\alpha_2\circ\gamma_2}\| = \inf_{\gamma_1,\gamma_2\in\tilde{\Gamma}} \|c_{\alpha_1\circ\gamma_1} - c_{\alpha_2\circ\gamma_2}\| \\ &\approx \inf_{\gamma\in\Gamma} \|c_{\alpha_1} - c_{\alpha_2\circ\gamma}\| \ . \end{aligned} \tag{9}$$

The last equation comes from the fact that Γ is dense in $\tilde{\Gamma}$. Note that since c_{α_1} and c_{α_2} are paths in $\mathbb{R}^{3n-7}$ we can safely use the Euclidean distance between corresponding points to evaluate d_α. This quantity d_α has several interesting properties: (1) it is invariant to arbitrary execution rates of actions, i.e., $d_\alpha(\alpha_1,\alpha_2) = d_\alpha(\alpha_1\circ\gamma_1, \alpha_2\circ\gamma_2)$, for arbitrary actions rates $\gamma_1,\gamma_2 \in \Gamma$, (2) it provides an objective function for temporal registration of trajectories and removal of execution rate variability in actions; this objective function is symmetric and inverse consistent (registration of α_1 to α_2 is same as that of α_2 to α_1), and (3) it is a proper distance that can be used for further action classification. Hence, the registration and comparison of trajectories is performed simultaneously under the same criterion. The actual optimization over Γ in Eqn. (9) is performed using the Dynamic Programming algorithm as described in [1].

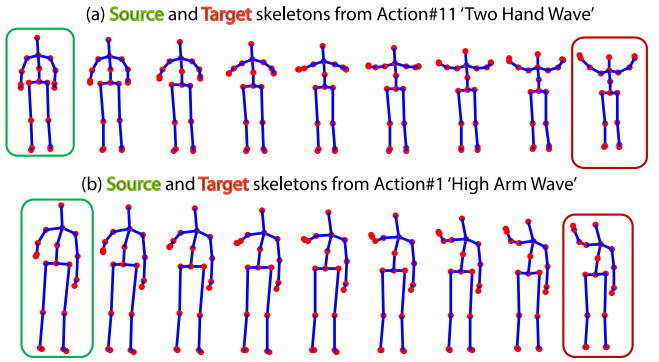

Fig. 2. Two examples of geodesic interpolation between arbitrary shapes using Eqn. 4.

3 Relevant Geometric Tools

Using the mathematical framework described in the previous section, we now develop a set of computational tools for help in analyzing actions when viewed as trajectories on shape space of skeletons. These tools respect the underlying geometry of the shape space $\mathcal{S}$ and help maintain desired invariances. We will start by tools for analyzing individual shapes and then proceed to tools for shape trajectories, all in the context of action classification using skeletons.

3.1 Geometric Tools For Skeletons

1. *Shape interpolation using geodesics*. We have characterized actions as continuous paths on shape spaces of skeletons but, in practice, one observes skeletons at discrete times only. This creates a need for interpolating between observed skeletons, or rather their shapes, at arbitrary points. Given any two skeletons $X, Y \in \mathcal{C}$, we can interpolate between their shapes $[X]$ and $[Y]$ in $\mathcal{S}$, using the geodesic connecting them. For any $w_1, w_2 > 0$ such that $w_1 + w_2 = 1$, the weighted interpolation between $[X]$ and $[Y]$ is given by $\alpha(w_1)$ where α is as specified in Eqn. (4). Fig. 2 shows two examples of dense interpolation between shapes of two pairs of skeletons taken from different action sequences. Notice the smooth variation of limbs and other parts as the shape changes from one skeleton to the other.

2. *Mean and median shape estimation*. Another important tool in shape analysis of skeletons is computation of their statistical summaries. Given a set of skeletons, taken from the corresponding time points in different observations of the same action, or even different actions, one would like to compute their statistical mean as a template for that shape. In case of noisy observations and in the presence of outliers, one often uses a median instead of the mean to minimize the influence of outliers. Therefore, we need tools to compute mean and median of sets of skeletal shapes under the chosen shape metric. For a set of given shapes $[X_1], \ldots, [X_k]$, the two quantities are:

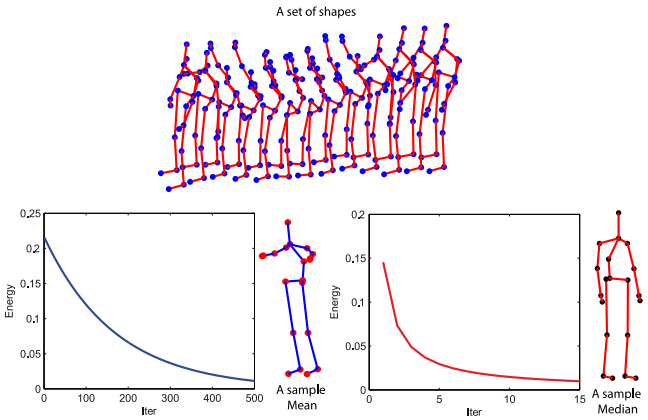

Fig. 3. Examples of mean and median shapes, and the decreases in energies during their computations.

$$\text{Mean} \quad \hat{\mu}_k = \underset{[X]\in\mathcal{S}}{\operatorname{argmin}} \sum_{i=1}^{k} d_s([X],[X_i])^2,$$

$$\text{Median} \quad \hat{m}_k = \underset{[X]\in\mathcal{S}}{\operatorname{argmin}} \sum_{i=1}^{k} d_s([X],[X_i]) \ .$$

Here d_s is as defined in Eqn. (5). Algorithm 2 describes the steps to compute a sample mean of a set of skeletons. The formulas for computing the inverse exponential map ($\exp^{-1}$) and the exponential map ($\exp$) are given by Eqns. (3) and (2), respectively.

Algorithm 2. Computing a Sample Mean on $\mathcal{S}$

Given a set of shapes $\{[X_j]\}_{(1\le j\le k)} \in \mathcal{S}$ and $\epsilon_1, \epsilon_2 > 0$ small.
Initialize $\hat{\mu}_0 \leftarrow [X_1]$, $i \leftarrow 0$
repeat
  Compute $v_j \leftarrow \exp^{-1}_{\hat{\mu}_i}([X_j])$, for $j = 1, \ldots, k$
  Compute average tangent vector $\bar{v} \leftarrow \frac{1}{k}\sum v_j$
  Update $\hat{\mu}_i$ according to $\hat{\mu}_{i+1} \leftarrow \exp_{\hat{\mu}_i}(\epsilon_2 \bar{v})$
  $i \leftarrow i + 1$
until $(|\bar{v}| < \epsilon_1)$
return $\hat{\mu}$, a sample Mean of $\{[X_j]\}_{(1\le j\le k)}$.

Similarly, Algorithm 3 provides a median shape estimator $\hat{m}_k$ for a given set of shapes $[X_1], \ldots, [X_k]$. The main difference here from the mean estimation (Algorithm 2) is in the weights applied to the shooting vectors v_i to obtain the weighted average $\bar{v}$. The weights are the inverse of the geodesic distance between the current mean and the individual shapes.

Algorithm 3. Median Estimation

Given $\{[X_j]\}_{(1\le j\le k)} \in \mathcal{S}$ and $\epsilon_1, \epsilon_2 > 0$
Initialize $\hat{m}_0 \leftarrow [X_1]$, $i \leftarrow 0$,
repeat
  Compute $v_j \leftarrow \exp^{-1}_{\hat{m}_i}([X_j])$, for $j = 1, \ldots, k$
  Compute $d_j = \sqrt{\langle v_j, v_j \rangle}$, for $j = 1, \ldots, k$
  Compute average vector $\bar{v} \leftarrow (\sum \frac{v_j}{d_j})(\sum \frac{1}{d_j})^{-1}$
  Move $\hat{m}_i$ according to $\hat{m}_{i+1} \leftarrow \exp_{\hat{m}_i}(\epsilon_2 \bar{v})$
  $i \leftarrow i + 1$
until $(|\bar{v}| < \epsilon_1)$
return $\hat{m}$ a sample Median of $\{[X_j]\}_{(1\le j\le k)}$.

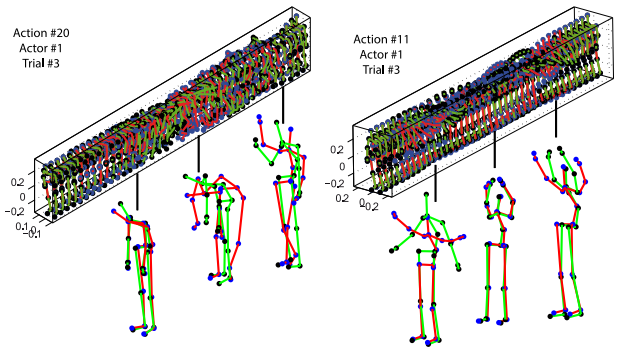

Fig. 4. Denoising trajectories using median filtering.

Fig. 3 shows the summary shapes—mean and median—for a set of skeletons. The accompanying plots show evolutions of respective energy functions during the computations of these representatives using iterative algorithms. It is interesting to note that not only the two shapes are perceptibly different but also the computational cost for estimating the median is much smaller than that of the mean.

3.2 Geometric Tools for Actions

Now we present some tools for pre-processing actions as trajectories on $\mathcal{S}$.

1. *Denoising action trajectories using median filtering*. As described in [12], one problem in extracting skeletons from depth images is the lack of temporal consistency. That is, some adjacent skeletons can have very different shapes despite being temporal neighbors. The reason can be partial occlusion, noisy range image, or something else. To limit the effect of this noise, it is desirable to be able to perform some kind of temporal smoothing (or filtering) and improve the quality of shape sequences in trajectories before their analysis. We propose to use the notion of median filtering, adapted to the shape space $\mathcal{S}$, for this purpose. Our approach is to take a moving window and replace the current central shape by a median of all the shapes in that time window, as outlined in Algorithm 4. Fig. 4 shows some examples of median filtering of trajectories containing noisy skeletons. To highlight the effects of this filtering, we have enlarged a few skeletons, displaying their before (red) and after (green) shapes.

Algorithm 4. Denoising Trajectory by Median Filtering

Given a trajectory $\alpha(s) \in \mathcal{S}$, $s = s_0, .., s_m$ and ω where $2w + 1$ is the size of the sliding window
$k \leftarrow w$,
repeat
  Apply Algorithm 3 to $\{\alpha(s)\}_{s=k-\omega\le s\le k+\omega}$
  Set $\alpha_F(k)$ to be the median and set $k \leftarrow k + 1$,
until $(k = m - \omega)$
return α_F a filtered trajectory of α.

2. *Up or down temporal resampling*. Another important need is to increase or decrease the number of skeleton shapes in a fixed time interval. The up-sampling helps achieve a higher temporal resolution needed to perform registration of trajectories (described later) and down-sampling helps improve computational efficiency in some

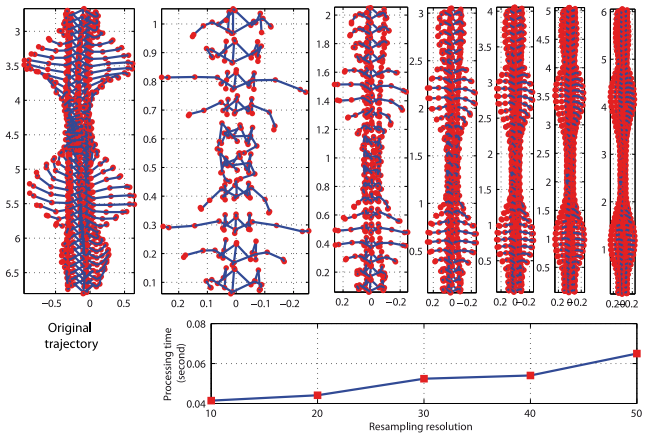

Fig. 5. The shape sequence shown on the left is resampled using different sample sizes on the right and the bottom plot shows computational costs.

situations, especially when the actions involved are very different and do not need a dense sampling for classification. We use the notion of piecewise geodesic to interpolate between shapes along a trajectory. Algorithm 5 provides a summary of steps needed for arbitrary resampling of a given trajectory. Fig. 5 shows examples of this idea where an original trajectory with 25 shapes is resampled using 10, 20, 30, 40, and 50 number of shapes. The computational cost of resampling at different sample sizes (based on a matlab code) is plotted below the resampled trajectories.

Algorithm 5. Re-Sampling Trajectory

Given a discrete trajectory $\alpha(t)_{t=t_1,..,t_n}$ we want $\alpha(s)_{s=s_1,..,s_m}$, where $n < m$ for up-sampling and $m < n$ for down-sampling).
for $i = 1 \dots m$ **do**
Find t_{i_1}, t_{i_2} such that $t_{i_1} \leq s_i \leq t_{i_2}$
Compute $w_1 = \frac{s_i - t_{i_1}}{t_{i_2} - t_{i_1}}$ and $w_2 = \frac{t_{i_2} - s_i}{t_{i_2} - t_{i_1}}$
$x = \alpha(i_1), y = \alpha(i_2), \theta = d_{\mathcal{S}}(x, y)$, then
$\alpha(s_i) = \frac{1}{\sin(\theta)}(\sin(w_2\theta)x + \sin(w_1\theta)y)$
end for
return Re-sampled trajectory $\alpha(s)$, $s = s_1, .., s_m$.

3. *Pairwise temporal-alignment of trajectories*. The next tool of interest is pairwise and group wise temporal alignment of several trajectories. This is an important tool since the original actions are often observed at arbitrary execution rate and, if ignored, this artificially inflates distance between two observations of the same action. The temporal alignment is performed using the optimization in Eqn. (9). For any two trajectories α_1 and α_2, we compute their TSRVFs and then solve for γ^* that minimizes the $\mathbb{L}^2$ norm according to Eqn. (9). Algorithm 6 summarizes steps needed in pairwise alignment of trajectories and Fig. 6 shows three illustrations of this tool on real data. It shows, in each case, α_1, α_2 (first column) , α_2 and $\alpha_2 \circ \gamma^*$ (second column), and α_1 and $\alpha_2 \circ \gamma^*$ (third column). The temporal misalignment of the red and the blue skeleton sequences is clear in the left column. In contrast, they are well matched in the third column after temporal registration. The plots on the right show the optimal warping γ^* obtained using the DP algorithm in Eqn. (9). Also stated in the plot are the

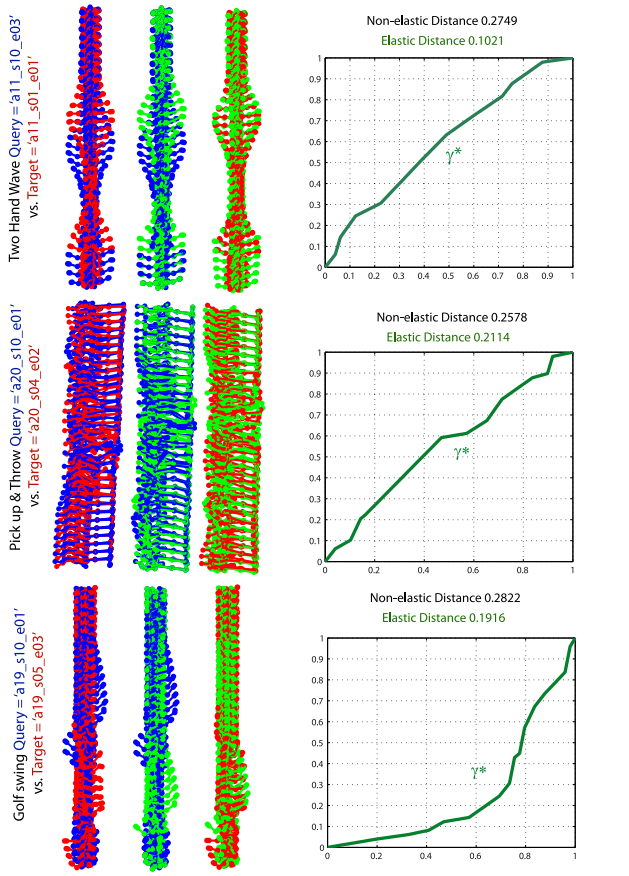

Fig. 6. Examples of pairwise temporal alignment of trajectories using Eqn. (9).

values of $\|h_{\alpha_1} - h_{\alpha_2}\|$ and $\|h_{\alpha_1} - h_{\alpha_2 \circ \gamma^*}\|$ to demonstrate the decrease in distance between the two trajectories due to alignment. For instance, the distance in the first example decreases from 0.2749 to 0.1021 due to an improved registration of shapes across actions. The registration performance naturally improves if the trajectories are up-sampled, using the resampling tool mentioned earlier. (Illustrations using animated gifs are provided at *http://www.telecom-lille.fr/people/benamor/Action3D.html*).

Algorithm 6. Temporal Alignment of Two Trajectories

Given two trajectories $\alpha_1(t), \alpha_2(t), t = t_1, .., t_n$
Compute TSRVFs $h_{\alpha_1}(t)$ and $h_{\alpha_2}(t)$ using Eqn. (8),
Project h_{α_1} and h_{α_2} onto an orthonormal basis $\mathcal{F}$ of $T_{[R]}(\mathcal{S})$ which results in c_{α_1} and c_{α_2}
Apply Dynamic Programming to find an optimal path γ^* to match c_{α_1} and c_{α_2}
return γ^*.

4. *Mean computation and multiple alignment of trajectories*. So far we have compared trajectories two at a time but our real need is to register and summarize multiple trajectories. For this we define and compute a mean trajectory μ_α under the distance d_α. In the process of computing the mean μ_α we recursively align individual trajectories to the mean, resulting in the temporal registration of all trajectories. This calculations is performed using the TSRVF representation. Since TSRVF is actually a curve in the vector space $T_{[R]}(\mathcal{S})$, expressed as coefficients with respect to the basis $\mathcal{F}$, we can

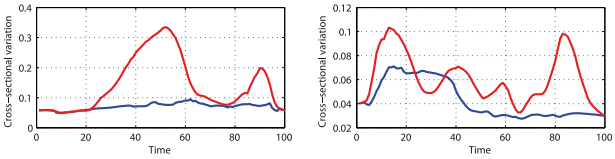

Fig. 7. Cross-sectional variance $\rho(t)$ versus t before (red) and after (blue) the temporal registration, for examples taken from MSR Action3D dataset.

use the Euclidean structure to define the covariance matrix associated with each time along the curve. Given a set of action trajectories: $\alpha_1, \alpha_2, \ldots, \alpha_N$, we define their mean according to: $\hat{\mu}_\alpha = \text{argmin}_\beta \sum_{i=1}^N d_\alpha(\beta, \alpha_i)^2$, with d_α as defined in Eqn. (9). Algorithm 7 outlines the steps for computing a mean action $\hat{\mu}_\alpha$.

Algorithm 7. Compute Sample Mean of Trajectories

Given a set of N trajectories $\{\alpha_j(t)_{t=1,..,n}\}_{1\leq j\leq N}$ and $\epsilon > 0$
Initialize $\hat{\mu}_1 \leftarrow \alpha_1$, $i \leftarrow 1$
repeat
  Find γ_j^* that aligns α_j to $\hat{\mu}_i$ using Algorithm 6, for each $1 \leq j \leq N$ and compute $\tilde{\alpha}_j = \alpha_j o \gamma_j^*$
  For $t = 1, .., n$, compute the Karcher Mean $\hat{\mu}_i(t) \in \mathcal{S}$ of $\{\tilde{\alpha}_j(t)\}$ using Algorithm 2
  Compute $E_i = \sum_{t=1}^n d_{\mathcal{S}}(\hat{\mu}_i(t), \tilde{\alpha}_i(t))^2$
  $i \leftarrow i + 1$,
until $(|E_i - E_{i-1}| < \epsilon)$
return $\hat{\mu}_i(t)_{t=1,..,n}$ a sampler mean trajectory.

Given a mean action, it is now possible to define covariance at each time t as follows. At each time t, let v_i be the shooting vectors from the mean shape orbit $\mu_\alpha(t)$ to the shape orbits $\{\alpha_i(t)\}_{i=1..N}$. Assume that $\{B_j(t)|i = 1..k\}$, $k = (3n - 7)$ is an orthonormal basis for each tangent space $T_{\mu_\alpha(t)}(\mathcal{S})$ (refer to item (4) of Section 2.1 for details on generating a basis). Next, we project each of the shooting vectors v_i onto the corresponding basis $v_i = \sum_{i=1}^k c_{i,j} B_j(t)$. Each original shape $\alpha_i(t)$ has now been represented using the coefficient $c_i = \{c_{i,j}\}$. Thus, the covariance matrix is computed in the coefficient space by $K(t) = (1/(N-1))$ $\sum_{i=1}^N c_i c_i^t$, $\in \mathbb{R}^{k\times k}$. The cross-sectional variance function is defined by taking the trace $\rho(t) = trace(K(t))$ at each time t. Fig. 7 displays the decreases in cross-sectional variances due to alignment of multiple trajectories. We plot $\rho(t)$ versus t before alignment (red) and after alignment (blue). These trajectories represent the same action performed by different subjects, taken from the MSR Action3D dataset. As these results demonstrate, the cross-sectional variance, in general, decreases considerably as a result of the temporal registration.

4 Action Classification

Now that we have geometrical tools for processing static and dynamic skeletal data, we illustrate their use to designing algorithms for action classification. As described previously, we explore two different classifiers: (1) *nearest-neighbor* or (2) *support vector machine* (SVM). While NN is based on a straightforward comparison of query actions to target actions using d_α, as defined in Section 3.2 item **(3)**, the SVM approach is based on an Euclidean representation derived from aligned trajectories. The gallery trajectories may either consist of several samples per action class or just a representative template sample per class (a mean trajectory, for example).

4.1 Direct Nearest-Neighbor Classifier

Here we choose to compute a mean (template) trajectory for every action category and use these means as target trajectories for future classification. To this end, all the skeletal trajectories in the same action class (in a training subset) are temporally registered and averaged to compute a sample mean using Algorithm 7. Given a query Q, it is simultaneously aligned and compared to the template trajectory and is classified in an action class using the NN classifier.

4.2 SVM Classifier After Smoothing

In this case we use d_α implicitly under an SVM framework, as follows. Recall that d_α is same as $\mathbb{L}^2$ norm between TSRVFs $h_\alpha(t)$ of registered trajectories, or their coefficients $c_\alpha(t)$ with respect to the basis $\mathcal{F}$. As mentioned in Algorithm 1, the TSRVFs are time derivatives of α computed using shooting vectors between successive points along trajectories, i.e., $h_\alpha(t) = v(t)_{\alpha\to[R]}/\sqrt{|v(t)|}$, where $v(t) = \exp^{-1}_{\alpha(t)}(\alpha(t+1))$. However, we have discovered that if we increase the time separation in computation of $v(t)$, i.e., use $\alpha(t+\delta)$ instead of $\alpha(t+1)$ where $\delta = 2, 3, 4 \ldots$, it has the potential to increase eventual classification performance. Note that finite difference with a larger time step is akin to approximating the derivative after smoothing the trajectory α. The resulting smoothed TSRVF, denoted by $h_{\alpha,\delta}(t)$, is still a curve in the tangent space $T_{[R]}(\mathcal{S})$ and can be treated as a Euclidean representation of a trajectory α. Furthermore, one can determine its coefficients with respect to the basis $\mathcal{F}$ as earlier; call them $c_{\alpha,\delta}(t)$. We can use these Euclidean representation in an SVM classification framework to improve action classification performance over the NN classifier.

There is one more step needed before a full Euclidean signature of each trajectory is specified. Let $j = 1, 2, \ldots, J$ be the index associated with different action classes and $\hat{\mu}_j$ be the mean trajectory of the j^{th} class. For any action sequence α, let $\alpha_j = \alpha \circ \gamma_j^*$ denote the optimal alignment of α to μ_j. Then, let $h^{(j)}_{\alpha,\delta}(t)$ be the smoothed TSRVF of the α_j and $c^{(j)}_{\alpha,\delta}(t)$ be its coefficients. A concatenation of these J individual Euclidean representations results in a large Euclidean vector that becomes the final Euclidean signature of any trajectory—training or test—and is fed in the classical SVM classifier.

5 Experiments and Results

Next we provide some experimental illustrations, validations, relative evaluations of the proposed method on real datasets. We use the following three challenging datasets—MSR Action-3D [13], MSR Daily Activity 3D [14], and 3D action pairs dataset [15].

5.1 MSR Action-3D Dataset Description

The *MSR Action-3D* dataset [13] is an interesting dataset from the perspective of developing and testing approaches

TABLE 1
NN-Classification Rate for Different Metrics

NN-to-Targets Distance	No Up-Sampling	Up-Sampling
Non-elastic d_{ne} (Eqn. (7))	0.51	0.42
Elastic d_α (Eqn. (9))	0.63	**0.69**

on elementary action recognition. It consists of a total of twenty types of segmented actions: *high arm wave*, *horizontal arm wave*, *hammer*, *hand catch*, *forward punch*, *high throw*, *draw x*, *draw tick*, *draw circle*, *hand clap*, *two hand wave*, *sideboxing*, *bend*, *forward kick*, *side kick*, *jogging*, *tennis swing*, *tennis serve*, *golf swing*, *pick up & throw*. Each action starts and ends with a neutral pose and is performed two or three times by each of the ten subjects. In total, there are 563 sequences in this dataset, 266 for the first five actors and 297 for the last five. Following the cross-subject experimental setting described in [13] and used later by [14], [15], [17], [23], we use the five first subjects for training and the last five for testing. The two classifiers mentioned previously—(1) NN-classifier, and (2) SVM-classifier—are used for classification purposes. We also study the impact of resampling and temporal registration of trajectories on performance.

5.2 Impact of Up-Sampling and Alignment

While it makes intuitive sense that a denser time sampling of actions will improve their classification, we have performed experiments to test this hypothesis. In these experiments we used the up-sampling tool presented earlier to increase the sampling rate of each realization to 100 time steps (or skeletons). Table 1 presents classification results obtained using the NN classifier under two metrics: the non-elastic distance d_{ne} presented in Eqn. (7) and the elastic distance d_α, in both cases—before and after the up-sampling. Each query sequence of the 297 test samples is compared to the target set (of 266 samples) associated with the 20 elementary actions. Thus, target and test sample sets are from different actors.

Table 1 provides two conclusions: (1) the temporal alignment incorporated in d_α helps improve classification rate in all cases, and (2) up-sampling of skeleton trajectories provides mixed results. The performance actually decreases for the non-elastic distance while it increases for the elastic-distance d_α. This stands to reason since a denser sampling of trajectories without proper temporal alignment can lead to increased misalignment and, thus, an increase in distances between trajectories within the same class. The denser sampling, when used in conjunction with proper temporal alignment, pays off with an increase in performance by 6 percent. Conversely, the idea of temporal alignment is more effective when there is a denser sampling of trajectories.

5.3 Impact of Median Filtering and Alignment

In the previous section we reported results based on NN classification. While this setting has the advantage that it does not need any training, it is expensive for real-time classification. An alternative idea is to compute a template action for each action class, using the statistical mean of observed actions in that class, and use this template for future classification. Algorithm 7 describes the computation

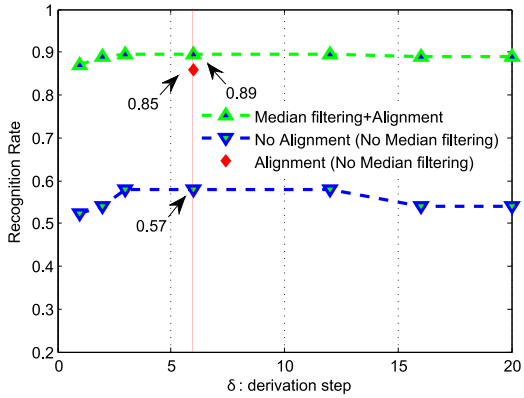

Fig. 8. Impact of the temporal alignment and the changes in δ on SVM-based classification accuracy.

of a mean trajectory, and we use it to compute mean trajectory $\hat{\mu}_j$ for each of the 20 action classes, $j = 1, 2, \ldots, 20$.

This time we use the SVM classifier using Euclidean coefficients $c_{\alpha,\delta}(t)$ described in Section 4.2. Recall that the basic idea is to approximate TSRVF with a larger time step δ, resulting in TSRVF $h_{\alpha,\delta}(t)$ for an action trajectory α, and then form large Euclidean representation by concatenating the coefficients $c^{(j)}_{\alpha,\delta}(t)$ for $j = 1, 2, \ldots, 20$. If each action is represented by T time samples, then the eventual Euclidean representation is in the space $\mathbb{R}^{20T(3n-7)}$. Then, we use an SVM classifier for classifying test trajectories and study the classification performance by changing the time step δ. The results obtained are reported in Fig. 8.

Using the SVM classifier and $\delta = 1$, our approach achieved 86.9 percent average recognition rate, 18 percent higher accuracy than the NN-classifier using d_α (69 percent in Table 1). This can be attributed to differences between corresponding shapes across trajectories in the same class despite having temporal synchronization. Different subjects, for example, raise their hands to different heights, when performing the waving action. It seems better in this context to study the deformations of shapes rather than shapes themselves. Although the use of TSRVF h_α does this by using a time derivative on the shape space, the increase of time step δ in computing $h_{\alpha,\delta}$ stabilizes this computation and provides an improved deformation descriptor. This descriptor is found to be less susceptible to limb variability across subjects when performing the same action, while preserving the distances across activities. As shown in Fig. 8, the performance increases slightly for $\delta > 2$ and reaches 89.04 percent. We further studied the case for δ=6, without the temporal median filtering step and with w=2 (see Algorithm 4). The average recognition rate is found to be lower by about 4 percent (85 percent compared to 89 percent with median filtering, see Fig. 8). This demonstrates the importance of the denoising step at the skeletal level in addition to the feature level (when increasing the derivation step δ). Similar to [15], we have conducted a cross-validation experiment by considering all the possible combinations of choosing half of the actors for training and remaining actors for testing (in total 252 folds for Action3D dataset). Using the format—mean (std. dev) percent—to report performance, we obtained a classification rate of $82.03(3.62)$ percent, as compared to $82.15(4.18)$ percent reported in [15].

TABLE 2
Comparison to Previous Studies on MSR Action-3D Dataset
(Metric-Based Shape Analysis)

Method	Accuracy
Recurrent Neural Network [33] (from [18])	0.42
Hidden Markov Model [26] (from [18])	0.63
Non-Elastic Distance d_{ne} (Eqn. (7))	0.51
DTW Distance d_{DTW} [34] (from [18])	0.54
This work – Elastic Distance d_α	**0.63**

TABLE 3
Comparison of Our Approach to Previous Methods on MSR Action-3D Dataset

Method	Accuracy
HON4D [15]	0.85
HON4D + D_{dis}[15]	0.88
Action Graph on Bag-of-3D Points [13]	0.74
Actionlet Ensemble [14], [18]	0.88
Motion Maps + HOG [17]	0.85
Joints distances + key poses [20]	0.65
Histograms of 3D joint locations (HOJ3D) [23]	0.78
Moving Pose (MP) [30]	0.91
This work – SVM classifier on $h_{\alpha,\delta}(t)$	**0.89**

5.4 Evaluation Relative to Previous Methods

Now we compare our results to some previous results on identical databases and experimental settings. For the challenging MSR Action-3D dataset, several papers have reported performances using the evaluation protocol of [13]. Our results are compared to mainly two types of methods—(1) Time-invariant approaches that first align the time-series using time warping solutions or exemplar-modeling (e.g., using HMM), as described in Section 1.1.2, and (2) Approaches that capture and learn motion-based descriptions to build classifiers.

5.4.1 NN Classifier with Time-Invariant Metric

First we compare with methods that utilize either an explicit temporal alignment (continuous-time model) or use other rate-invariant methods, such as HMM (discrete-time model), and the actions are analyzed using the shapes of skeletons. (We point out that in the current literature temporal alignment is generally regarded as a pre-processing step before actual classification, as opposed to our solution where alignment and classification are joint.) The results are reported in Table 2. To facilitate valid comparisons, we have used trajectories at their original time samples without any up-sampling. It is interesting to note the improvement of d_α over a standard DTW procedure. Recall that DTW is generally applied on the original space of representation and has theoretical issues (lack of symmetry, lack of triangle inequality, etc). This result shows that using a proper distance, in the form of d_α, provides not just theoretical elegance but also a substantial improvement in empirical performance.

Another point is that temporal alignment helps improve performance in all cases. The proposed distance d_α, the DTW and the HMM-based models all outperform the non-elastic distance. The HMM-based approach uses a discrete time model, and represents an action via a set of transition probabilities of visiting prototype shapes. Thus, it is naturally invariant to the changes in execution rates of actions. Although the HMM-based and the d_α-based NN classifier achieved similar results, the former requires a training step while the latter does not.

5.4.2 SVM Classifier After Smoothing

We now present results using SVM classifier on the $c_{\alpha,\delta}(t)$-representation.

5.4.3 Evaluation on MSR Action-3D dataset

Table 3 summarizes results of our method as well as some recently published methods on the MSR Action-3D dataset. Our approach outperforms the highest published classification rates in [14], [15], [18]. It is important to note that these methods are pose- and camera view-dependent as they compute features based on the depth images returned by the sensor. Our experience is that skeletal data, after application of our pre-processing tools (median filtering, etc), is more reliable than a direct use of histograms of normals (HON4D) or local occupancy pattern (LOP), in classification of actions. On the other hand, the skeleton data is more susceptible to self-occlusion, leading to missing parts in depth images. The approach published in [14], [18], uses both the skeletal data and the depth-maps around the joints.

To better understand this performance, we report classification rates per action, taken individually, for three different methods in Fig. 9. From this bar plot, one can observe that results are quite similar for the first two and the last thirteen classes, but differ significantly for actions performed using one hand (Hammer, Hand catch, High Throw and Draw X). A few remarks can be made about these actions: First, the recognition rate achieved by all three approaches are lower for these actions relative to other actions implying that these actions are indeed difficult to separate. Second, except for "Hammer" action, the approaches based on global shapes (our and [15]) outperform local approach ([14]). Finally, we point out that for actions involving moving hand(s), our method generally outperforms [15].

5.4.4 Evaluation on 3D Action Pairs Dataset

Oreifej and Liu [15] collected an interesting dataset with actions that present similar motion cues. The dataset consist of six pairs of actions: *Pick up a box/Put down a box, Lift a box/Place a box, Push a chair/Pull a chair, Wear a hat/Take off a hat, Put on a backpack/Take off a backpack, Stick a poster/Remove a poster*, played by ten actors three times each. This database is important in the sense that methods that aggregate

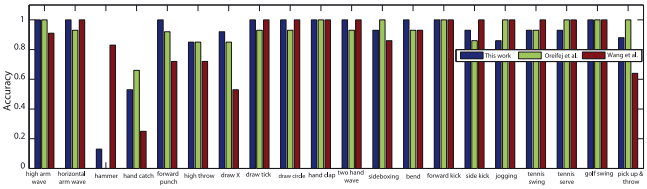

Fig. 9. Classification rates of actions, taken individually, of our approach and Oreifej and Liu [15] (reported results are generated using the implementation available online), and Wang et al. [14], [18].

TABLE 4
Comparison of Our Approach to Previous Methods on 3D Action Pairs Dataset

Method	Accuracy
HON4D + D_{dis} [15]	0.96
HON4D [15]	0.93
Skeleton+LOP [14] (from [15])	0.63
Skeleton+LOP+Pyramid [14] (from [15])	0.82
Motion Map [17] (from [15])	0.66
This work – SVM classifier on $h_{\alpha,\delta}(t)$	**0.93**

features temporally, and lose the time stamps associated with these features, such as those based on histograms, will not be able to distinguish between forward and backward actions. The results of several methods, including ours, are presented in Table 4.

Following the experimental protocol described in [15], our approach achieved a classification rate of 93 percent when using only the skeletal data. This result confirms the fact that our approach is able to distinguish between forward and backward actions. This is due to the fact that our approach captures the motion while respecting its natural time evolution.

5.4.5 Evaluation on Daily Activity 3D Dataset

This daily activity dataset is captured by a Kinect device and it consists of 16 activity types : *drink, eat, read book, call cellphone, write on a paper, use laptop, use vacuum cleaner, cheer up, sit still, toss paper, play game, lay down on sofa, walk, play guitar, stand up, sit down*. The challenging part here is that each subject performs an activity in two different poses: sitting and standing. The total number of the activity sequences is 320 = 16×10×2 (10 actors and 2 trials/actor).

Following the same experimental cross-subject setting, where the first five actors are taken as training and the last five for testing, our approach achieved a classification rate of 70 percent. While *stand-up*, *sit-down*, *cheer-up*, *lay down* are well recognized (around 100 percent), the others, involving human-object interaction, are not recognized as well. These include: *read book*, *call cell phone*, *write*, *use laptop* and *use vacuum cleaner*. In this situation our method is at a disadvantage since it uses only the skeleton data. On the other hand, methods such as [18] (Actionlet Ensemble) and [15] (a local adaptation of the holistic description HON4D, on each joint, called Local HON4D) add the depth data around the joints to better describe the human-to-object interaction, which make the classification of *write* or *use laptop* more accurate. A complete comparison with previous studies is reported in Table 5. We remark that in [18], an experiment using only the joint positions achieved 68 percent accuracy, 2 percent less accurate than our approach. Note also that a lower performance of our approach, compared to [15] and [18], can also be explained due to the noisy skeletal data. In fact, the actors are often sitting on the sofa, which makes it less accurate to track skeletons and corrupts the skeleton data.

TABLE 5
Comparison of Our Approach to Previous Methods on MSR Daily Activity 3D Dataset

Method	Accuracy
Actionlet Ensemble [18]	0.85
Local HON4D [15]	0.80
LOP+Pyramid features (from [18])	0.78
Random Occupancy Pattern [35] (from [18])	0.64
LOP features (from [18])	0.64
Moving Pose [30]	0.73
This work – SVM classifier on $h_{\alpha,\delta}(t)$	**0.70**

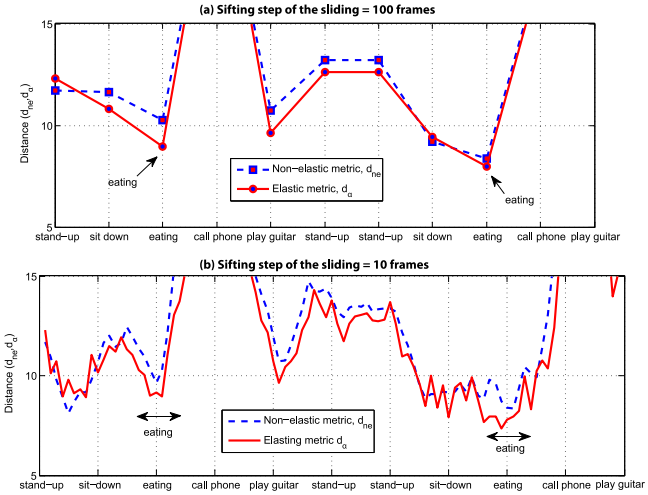

Fig. 10. Action sub-sequence detection in a long video of actor#1 of Daily Activity 3D dataset. The test sequence is from a different actor.

So far we have assumed that the temporal locations of actions (in long sequences) is provided. However, this framework is potentially useful in temporal location also, as demonstrated in Fig. 10. In this experiment, we consider a long video of several daily activities performed by actor#1, taken from MSR Daily Activity 3D dataset. The action "eating" is present at two different times in this sequence. Then, we take a test sequence of this class (performed by a different actor), and compare this sequence to a moving window over long sequence. Fig. 10 provides distances d_{ne}, the non-elastic metric (i.e without temporal alignment) and d_α, the elastic metric. In case (a), the long sequence is analyzed with a window step of 100 frames, while in (b) this parameter is taken to be 10. The valleys in the distance functions correctly denote the location of eating activity.

5.5 Computational Efficiency

In Table 6 we provide the average time costs of each step performed in the proposed pipeline. We note that only the

TABLE 6
Computational Efficiency of the Proposed Pipeline

Pipeline Step	Time (sec)
Trajectory Denoising, window size = 5 (in Matlab)	0.5
Trajectory Up-Samling step = 0.01 (in Matlab)	0.1
Temporal Alignment of 2 trajectories (in C++)	0.25
Trajectory Motion extraction (in Matlab)	2
Overall (in C++ and Matlab)	**2.85**

temporal alignment step, stated in Eqn. (9), uses a C++ implementation of the Dynamic Programming algorithm. All the remaining pieces are implemented in Matlab. The experiments where carried out using a 3.1 GHz single CPU machine, with 8 GB of RAM, running Windows. These times underline the efficiency of our framework. In relative terms, it is deemed computationally more efficient than the representations based on depth images or 2D videos, that need more processing time to compute geometric features, as in presented in [15], or compute depth-maps as proposed in [17]. The processing times (less than 3 seconds) associated with our method point to a potential for a real-time system.

6 Conclusions & Discussion

We have presented a comprehensive framework for analyzing human actions using shape evolutions of skeletons. This framework is based on representations and metrics that are independent of the execution rates of activities being classified. It comes with a suite of tools derived from Riemannian geometry to process static skeletal data and their temporal evolutions, and accomplishes several fundamental tasks, such as interpolating shapes, denoising trajectories and computing summary statistics of these trajectories. This paper studies Euclidean representations of shape trajectories via smoothed TSRVFs and uses them in conjunction with an SVM classifier to obtain state-of-the-art results. Several experimental results on publicly available datasets, each denoting a different challenge, are presented. We achieve high accuracy on two action datasets (MSR Action 3D and 3D Action Pairs) despite using only the skeleton information.

The main advantage of this approach, relative to those based on keyframes, is that it is based on generative models. One can use it to perform PCA and similar ideas to extract dominant modes of variations in observed sequences, and to develop a synthetic model for full action trajectories. This synthesis can be useful in applications in graphics, domain adaptations, etc. The limitations of the proposed approach are, *(i) Prone to severe noise in skeletal data*—despite denoising via median filtering and implicit smoothing in computing TSRVF, this method is prone to the obscuration-related noise that is often present in skeletons. A method that explicitly detects and fixes large deformations in shapes of skeletons in successive frames is needed; *(ii) Restriction to short video sequences*—while most of the tools proposed here are applicable to sequences independent of their size, the notion of classification here assumes that the relevant action subsequences have already been extracted and isolated; *(iii) Confusion between similar actions*—This method will fail wherever skeleton data is not sufficient to characterize actions.

Future work includes extension to complex (high-level) activities such as those in the CAD-120 dataset [36]. These long videos can be considered as concatenations of atomic (low-level) actions and one can perform key sub-actions detection using a scanning window and the rate-invariant metric proposed here. Another item for future development is the sequential hypothesis test for early detection of activities.

Acknowledgments

This research was performed during B. Ben Amor's visit to the Florida State University during 2013-2014. It was supported in part by Institut Mines-Télécom, the MAGNUM project (BPI and Région Nord-Pas de Calais) to BB, and NSF grants DMS 1208959 and 1217515, and a Fulbright scholar grant to AS.

References

[1] J. Su, S. Kurtek, E. Klassen, and A. Srivastava,, "Statistical analysis of trajectories on Riemannian manifolds: Bird migration, hurricane tracking, and video surveillance," *Ann. Appl. Statist.*, vol. 8, no. 1, pp. 530–552, 2013.

[2] J. Su, A. Srivastava, F. Souza, and S. Sarkar, "Rate-invariant analysis of trajectories on Riemannian manifolds with application in visual speech recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 620–627.

[3] A. Kleinsmith and N. Bianchi-Berthouze, "Affective body expression perception and recognition: A survey," *IEEE Trans. Affective Comput.*, vol. 4, no. 1, pp. 15–33, Jan.–Mar. 2013.

[4] A. Mehrabian and M. Wiener, "Decoding of inconsistent communications," *J. Personality Soc. Psychol.*, vol. 6, no. 1, pp. 109–114, 1967.

[5] J. Aggarwal and M. Ryoo, "Human activity analysis: A review," *ACM Comput. Surv.*, vol. 43, no. 3, pp. 16:1–16:43, 2011.

[6] J. M. Chaquet, E. J. Carmona, and A. Fernández-Caballero, "A survey of video datasets for human action and activity recognition," *Comput. Vis. Image Understanding*, vol. 117, no. 6, pp. 633–659, 2013.

[7] S.-R. Ke, L. U. T. Hoang, Y.-J. Lee, J.-N. Hwang, J.-H. Yoo, and K.-H. Choi, "A review on video-based human activity recognition," *Computers*, vol. 2, no. 2, pp. 88–131, 2013.

[8] A. Veeraraghavan, A. Srivastava, A. K. R. Chowdhury, and R. Chellappa, "Rate-invariant recognition of humans and their activities," *IEEE Trans. Image Process.*, vol. 18, no. 6, pp. 1326–1339, Jun. 2009.

[9] A. Srivastava, P. K. Turaga, and S. Kurtek, "On advances in differential-geometric approaches for 2D and 3D shape analyses and activity recognition," *Image Vis. Comput.*, vol. 30, nos. 6/7, pp. 398–416, 2012.

[10] M. F. Abdelkader, W. Abd-Almageed, A. Srivastava, and R. Chellappa, "Silhouette-based gesture and action recognition via modeling trajectories on Riemannian shape manifolds," *Comput. Vis. Image Understanding*, vol. 115, no. 3, pp. 439–455, 2011.

[11] Z. Zhang, "Microsoft kinect sensor and its effect," *IEEE Multimedia*, vol. 19, no. 2, pp. 4–10, Apr.-Jun. 2012.

[12] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2011, pp. 1297–1304.

[13] W. Li, Z. Zhang, and Z. Liu, "Action recognition based on a bag of 3D points," in *Proc. IEEE Inter. Workshop CVPR Human Commun. Behavior Anal.*, 2010, pp. 914.

[14] J. Wang, Z. Liu, Y. Wu, and J. Yuan, "Mining actionlet ensemble for action recognition with depth cameras," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2012, pp. 1290–1297.

[15] O. Oreifej and Z. Liu, "Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 716–723.

[16] J. Sung, C. Ponce, B. Selman, and A. Saxena, "Unstructured human activity detection from RGBD images," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2012, pp. 842–849.

[17] X. Yang, C. Zhang, and Y. Tian, "Recognizing actions using depth motion maps-based histograms of oriented gradients," in *Proc. 20th ACM Int. Conf. Multimedia*, 2012, pp. 1057–1060.

[18] J. Wang, Z. Liu, Y. Wu, and J. Yuan, "Learning actionlet ensemble for 3D human action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 5, pp. 914–927, May 2014.

[19] X. Yang and Y. Tian, "Effective 3D action recognition using eigenjoints," *J. Vis. Commun. Image Representation*, vol. 25, no. 1, pp. 2–11, 2014.

[20] C. Ellis, S. Z. Masood, M. F. Tappen, J. J. Laviola Jr., and R. Sukthankar, "Exploring the trade-off between accuracy and observational latency in action recognition," *Int. J. Comput. Vis.*, vol. 101, no. 3, pp. 420–436, 2013.

[21] M. Devanne, H. Wannous, S. Berretti, P. Pala, M. Daoudi, and A. D. Bimbo, "Space-time pose representation for 3D human action recognition," in *Proc. ICIAP Workshops*, 2013, pp. 456–464.
[22] A. Srivastava, E. Klassen, S. H. Joshi, and I. H. Jermyn, "Shape analysis of elastic curves in Euclidean spaces," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 7, pp. 1415–1428, Jul. 2011.
[23] L. Xia, C.-C. Chen, and J. K. Aggarwal, "View invariant human action recognition using histograms of 3D joints," in *Proc. CVPR Workshops*, 2012, pp. 20–27.
[24] A. Veeraraghavan, R. Chellappa, and A. Roy-Chowdhury, "The function space of an activity," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2006, vol. 1, pp. 959–968.
[25] A. Gritai, Y. Sheikh, C. Rao, and M. Shah, "Matching trajectories of anatomical landmarks under viewpoint, anthropometric and temporal transforms," *Int. J. Comput. Vis.*, vol. 84, no. 3, pp. 325–343, 2009.
[26] F. Lv and R. Nevatia, "Recognition and segmentation of 3-d human action using hmm and multi-class adaboost," in *Proc. 9th Eur. Conf. Comput. Vis.-Volume Part IV*, 2006, pp. 359–372.
[27] P. K. Turaga, A. Veeraraghavan, A. Srivastava, and R. Chellappa, "Statistical computations on Grassmann and Stiefel manifolds for image and video-based recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 11, pp. 2273–2286, Nov. 2011.
[28] R. Slama, H. Wannous, and M. Daoudi, "Grassmannian representation of motion depth for 3D human gesture and action recognition," in *Proc. 22nd Int. Conf. Pattern Recog.*, Stockholm, Sweden, 2014, pp. 3499–3504.
[29] M. Raptis and L. Sigal, "Poselet key-framing: A model for human activity recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 2650–2657.
[30] M. Zanfir, M. Leordeanu, and C. Sminchisescu, "The moving pose: An efficient 3D kinematics descriptor for low-latency action recognition and detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 2752–2759.
[31] I. Dryden and K. Mardia, *Statistical Shape Analysis*. New York, NY, USA: Wiley, 1998.
[32] R. Vemulapalli, F. Arrate, and R. Chellappa, "Human action recognition by representing 3D skeletons as points in a lie group," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 588–595.
[33] J. Martens and I. Sutskever, "Learning recurrent neural networks with Hessian-free optimization," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 1033–1040.
[34] M. Müller and T. Röder, "Motion templates for automatic classification and retrieval of motion capture data," in *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animation*, 2006, pp. 137–146.
[35] J. Wang, Z. Liu, J. Chorowski, Z. Chen, and Y. Wu, "Robust 3D action recognition with random occupancy patterns," in *Proc. 12th Eur. Conf. Comput. Vis.-Volume II*, 2012, pp. 872–885.
[36] H. Koppula and A. Saxena, "Learning spatio-temporal structure from RGB-D videos for human activity detection and anticipation," in *Proc. 30th Int. Conf. Mach. Learn.*, 2013, vol. 28, no. 3, pp. 792–800.

Boulbaba Ben Amor received the MS and PhD degrees from Ecole Centrale de Lyon in 2003 and 2006, respectively. He received the Habilitation (HDR) degree in computer science from LILLE 1 University, in 2014. He is an associate professor of computer science with Institut Mines-Télécom Lille/Télécom Lille and a member of the CRIStAL Research Center (UMR CNRS 9189), since 2007. He was a visiting researcher in FSU (US) during 2014. His areas of research include 3D computer vision, 3D/4D shape analysis and pattern recognition.

Jingyong Su received the BE and MS degrees in electrical engineering from the Harbin Institute of Technology, China, in 2006 and 2008, respectively, and the PhD degree from the Department of Statistics, Florida State University, in May 2013. He is currently an assistant professor in the Department of Mathematics and Statistics, Texas Tech University in Lubbock. His main research interests include statistical image analysis, computer vision and computational statistics.

Anuj Srivastava received the PhD degree from Washington University, St. Louis, in 1996. He is a professor of statistics in Florida State University. He was a visiting research associate in Brown University during 1996-1997. He joined the Department of Statistics at the Florida State University in 1997 as an assistant professor, and became a full professor in 2007. His areas of research include statistics on nonlinear manifolds, functional data analysis, and statistical shape theory. He has published more than 200 papers in journals and conference proceedings. He is/has been the associate editor for the *IEEE TSP*, *TPAMI*, and *TIP*.