Kernelized Temporal Cut for Online Temporal Segmentation and Recognition

Dian Gong, Gérard Medioni, Sikai Zhu, and Xuemei Zhao

Institute for Robotics and Intelligent Systems, University of Southern California
Los Angeles, CA, 90089
{diangong,medioni,sikaizhu,xuemeiz}@usc.edu

Abstract. We address the problem of unsupervised online segmenting human motion sequences into different actions. Kernelized Temporal Cut (KTC), is proposed to sequentially cut the structured sequential data into different regimes. KTC extends previous works on online change-point detection by incorporating Hilbert space embedding of distributions to handle the nonparametric and high dimensionality issues. Based on KTC, a realtime online algorithm and a hierarchical extension are proposed for detecting both action transitions and cyclic motions at the same time. We evaluate and compare the approach to state-of-the-art methods on motion capture data, depth sensor data and videos. Experimental results demonstrate the effectiveness of our approach, which yields realtime segmentation, and produces higher action segmentation accuracy. Furthermore, by combining with sequence matching algorithms, we can online recognize actions of an arbitrary person from an arbitrary viewpoint, given realtime depth sensor input.

1 Introduction

Temporal segmentation of human motion sequences (motion capture data, 2.5D depth sensor or 2D videos), i.e., temporally cut sequences into segments with different semantic meanings, is an important step for building an intelligent framework to analyze human motion. Temporal segmentation can be applied to motion animations [1], action recognition [2–4], video understanding and activity analysis [5, 6]. In particular, temporal segmentation is crucial for human action recognition. Most recent works in human activity recognition focus on simple primitive actions such as walking, running and jumping, in contrast to the fact that daily activity involves complex temporal patterns (walking then sit-down and stand-up). Thus, recognizing such complex activities relies on accurate temporal structure decomposition [7].

Previous work on temporal segmentation can be mainly divided into two categories. On one side, many works in statistics, i.e., either offline or online (quickest) change-point detections [8], are often restricted to univariate series (1D) and the distribution is assumed to be known in advance [9]. Because of the complex structure of motion dynamics, these works are not suitable for temporal segmentation of human actions. On the other side, temporal clustering has been proposed for unsupervised learning of human motions [10]. However, temporal clustering is usually performed offline, thus not suitable for applications such as realtime action segmentation and recognition.

A. Fitzgibbon et al. (Eds.): ECCV 2012, Part III, LNCS 7574, pp. 229–243, 2012.

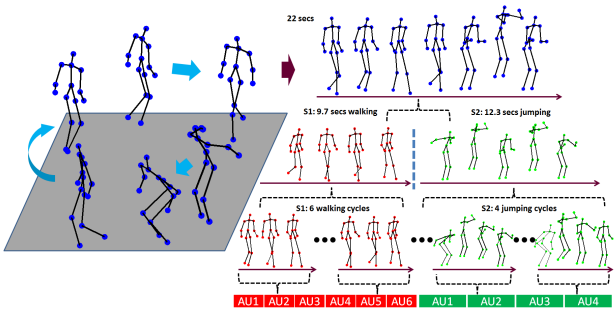

Fig. 1. Online Hierarchial Temporal Segmentation. A 22 secs input sequence is temporally cut into two segments; a walking segment (S1) which is further cut into 6 action units, and a jumping segment (S2) which is further cut into 4 action units.

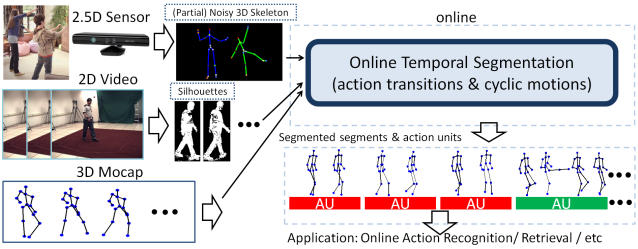

Fig. 2. System Flowchart. Input can be either Mocap, 2.5D depth sensor or 2D videos. Output are temporal cuts for both action transitions and cyclic action units.

Motivated by these difficulties, we propose an online temporal segmentation method with no parametric distribution assumptions, which can be directly applied to detect action transitions. The proposed method, Kernelized Temporal Cut (KTC), is a temporal application of Hilbert space embedding of distributions [11, 12] and kernelized two-sample test [13, 14] of online segmentation on structured sequential data. KTC can simultaneously detect action transitions and cyclic motion structures (action units) within a regime as shown in Fig. 1. Furthermore, a realtime implementation of KTC is proposed, incorporating an incremental sliding window strategy. Within a sliding window, segmentation is performed by the two-sample hypothesis test based on the proposed spatio-temporal kernel. The input and output of our system are shown in Fig. 2. In summary, our approach presents several advantages:

– **Online:** KTC can sequentially process the input and capture action transitions, which is extremely helpful for realtime applications such as continuous action recognition.
– **Hierarchical:** KTC can simultaneously capture transition points between different actions, and cyclic *action units* within an action regime, e.g., a walking segment contains several walking cycles. This is important for realtime activity recognition, i.e., action can be recognized after only one action unit.

– **Nonparametric:** nonparametric and high dimensionality issues are handled by the Hilbert space embedding based on the spatio-temporal kernel, which can be directly applied to complex sequential data such as human motion sequences.
– **Online Action Recognition and Transfer Learning:** KTC can be applied to a variety of input such as motion capture data, depth sensor data and 2D videos from an unknown viewpoint. More importantly, KTC can be applied to online action recognition from OpenNI and Kinect by combining methods such as [15]. The recognition is performed without any training data from depth sensor, which is truly *transfer learning*.

2 Related Work

Temporal segmentation is a multifaced area and several related topics in machine learning, statistics, computer vision and graphics are discussed in this section.

Change-Point Detection. Most of the work in statistics, i.e., offline or quickest (online) change-point detections (CD) [8], is often restricted to univariate series (1D) and parametric distribution assumption, which does not hold for human motions with complex structure. [16] uses the undirected sparse Gaussian graphical models and performs jointly structure estimation and segmentation. Recently, as a nonparametric extension of Bayesian online change-point detection (BOCD) [9], [17] is proposed to combine BOCD and Gaussian Process (GPs) to relax the i.i.d assumption in a regime. Although GPs improve the ability to model complex data, it also brings in high computational cost. More relevant to us, kernel methods have been applied to non-parametric change-point detection on multivariate time series [18, 19]. In particular, [18] (KCD) utilizes the one-class SVM as online training method and [19] (KCpA) performs sequentially segmentation based on the Kernel Fisher Discriminant Ratio. Unlike all the above works, KTC can not only detect action transitions but also cyclic motions.

Temporal Clustering. Clustering is a long standing topic in machine learning [20, 21]. Recently, as an extension of clustering, some works focus on how to correctly temporally segment time series into different clusters. As a elegant combination of Kernel K-means and spectral clustering, Aligned Cluster Analysis (ACA) is developed for temporal clustering of facial behavior with a multi-subject correspondence algorithm for matching facial expressions [6]. To estimate the unknown number of clusters, [22] use the hierarchical Dirichlet process as a prior to improve the switch linear dynamical system (SLDS). Most of these works offline segment time series and provide cluster labels as in clustering. As a complementary approach, KTC performs online temporal segmentation which is suitable for realtime applications.

Motion Analysis. In computer vision and graphics, some works focus on grouping human motions. Unusual human activity detection is addressed in [5] using the (bipartite) graph spectral clustering. [23] extracts spatio-temporal features to address event clustering on video sequences. [10] proposes a geometric-invariant temporal clustering algorithm to cluster facial expressions. More relevantly, [1] proposes an online algorithm to decompose motion sequences into distinct action segments. Their method is an elegant temporal extension of Probabilistic Principal Component Analysis for change-point detection (PPCA-CD), which is computationally efficient but restricted to (approximate) Gaussian assumptions.

Action Recognition. Although significant progress has been made in human activity recognition [3, 4, 7, 24], the problem remains inherently challenging due to viewpoint change, partial occlusion and spatio-temporal variations. By combining KTC and alignment approaches such as [25, 15], we can perform online action recognition for input from 2.5D depth sensor. Unlike other works on *supervised* joint segmentation and recognition [26], two significant features of our approach are, viewpoint independence and handling arbitrary person with a few labeled Mocap sequences, in the *transfer learning* module.

3 Online Temporal Segmentation of Human Motion

This section describes the Kernelized Temporal Cut (KTC), a temporal application of Hilbert space embedding of distributions [11] and kernelized two-sample test [14, 13], to sequentially estimate temporal cut points in human motion sequences.

3.1 Problem Formulation

Given a stream input $\boldsymbol{X}_{1:L_x} = \{\boldsymbol{x}_t\}_{t=1}^{t=L_x}$ ($\boldsymbol{x}_t \in \Re^{D_t}$, where D_t can be fixed or change over time), the goal of temporal segmentation is to predict temporal cut points c_i. For instance, if a person walks and then boxes, a temporal cut point must be detected. For depth sensor data, $\boldsymbol{x}_t$ is the vector representation of tracked joints. More details of $\boldsymbol{x}_t$ are given in sec. 6. From a machine learning perspective, the estimated $\{c_i\}_{i=1}^{N_c}$ can be modeled by minimizing the following objective function,

$$\mathcal{L}_{\boldsymbol{X}}(\{\boldsymbol{c}_i\}_{i=1}^{N_c}, N_c) = \sum_{i=1}^{N_c} I(\boldsymbol{X}_{c_{i-1}:c_i-1}, \boldsymbol{X}_{c_i:c_{i+1}-1}) \tag{1}$$

where $\boldsymbol{X}_{c_i:c_{i+1}-1} \in \Re^{D\times(c_{i+1}-c_i)}$ indicates the segment between two cut points c_i and c_{i+1} ($c_1 = 1,\ c_{N_c+1} = L_x + 1$). Here $I(\cdot)$ is the homogeneous function to measure the spatio-temporal consistency between two consecutive segments. It is worth noting that, both $\{\boldsymbol{c}_i\}_{i=1}^{N_c}$ and N_c need to be estimated from eq. 1. Next, the main task is to design $I(\cdot)$ and to *online* optimize eq. 1. As the counterpart, eq. 1 could be *offline* optimized by dynamic programming when N_c is given, which is out of the scope of this paper.

3.2 KTC-S

Instead of jointly optimizing eq. 1, the proposed Kernelized Temporal Cut (KTC) *sequentially* optimizes c_{i+1} based on c_i by minimizing the following loss function,

$$\mathcal{L}_{\{\boldsymbol{X}_{c_i:c_i+T-1}\}}(c_{i+1}) = I(\boldsymbol{X}_{c_i:c_{i+1}-1}, \boldsymbol{X}_{c_{i+1}:c_i+T-1}),\ i = 1, 2, ..., N_c - 1 \tag{2}$$

where c_i ($c_1 = 1,\ c_{N_c+1} = L_x + 1$) is provided by the previous step and T is a fixed length. We refer to this sequential optimization process for eq. 2 as KTC-S, where S stands for sequential. Sequentially optimizing $\mathcal{L}$ is actually a fixed-length sliding window process which is also used in [19]. However, setting T is a difficult task and how

to improve this process is described in sec. 3.3. Essentially, Eq. 2 is a two-class temporal clustering problem for $\boldsymbol{X}_{c_i:c_i+T-1} \in \Re^{D\times T}$. The crucial factor is constructing $I(\cdot)$, which is related to temporal version of (dis)similar functions in spectral clustering [20, 21, 10] and information theoretical clustering [27].

To handle the complex structure of human motion, unlike previous work, KTC utilizes Hilbert space embedding of distributions (HED) to map the distribution of $\boldsymbol{X}_{t_1:t_2}$ into the Reproducing Kernel Hilbert Space (RKHS). [11, 12] are seminal works on combining kernel methods and probability distribution analysis. Without going into details, the idea of using HED for temporal segmentation is straightforward. The change-point is detected by using a well behaved (smooth) kernel function, whose values are large on the samples belonging to the same spatio-temporal pattern and small on the samples from different patterns. By doing this, KTC does not only handle nonparametric and high dimensionality problems but also rests on a solid theoretical foundation [11].

HED. Inspired by [12], probabilistic distributions can be embedded in RKHS. At the center of the Hilbert space embedding of distributions are the mean mapping functions,

$$\mu(\boldsymbol{P}_{\boldsymbol{x}}) = \boldsymbol{E}_{\boldsymbol{x}}(k(\boldsymbol{x},\cdot)),\ \mu(\boldsymbol{X}) = \frac{1}{T}\sum_{t=1}^{T} k(\boldsymbol{x}_t,\cdot) \tag{3}$$

where $\boldsymbol{x}_{t=1}^{t=T}$ are assumed to be i.i.d sampled from the distribution $\boldsymbol{P}_{\boldsymbol{x}}$. Under mild conditions, $\mu(\boldsymbol{P}_{\boldsymbol{x}})$ (same for $\mu(\boldsymbol{X})$) is an element of the Hilbert space as follows,

$$< \mu(\boldsymbol{P}_{\boldsymbol{x}}), f >= \boldsymbol{E}_{\boldsymbol{x}}(f(\boldsymbol{x})),\ \ < \mu(\boldsymbol{X}), f >= \frac{1}{T}\sum_{t=1}^{T} f(\boldsymbol{x}_t)$$

Mappings $\mu(\boldsymbol{P}_{\boldsymbol{x}})$ and $\mu(\boldsymbol{X})$ are attractive because,

Theorem 1. *if the kernel k is universal, then the mean map μ: $\boldsymbol{P}_{\boldsymbol{x}} \rightarrow \mu(\boldsymbol{P}_{\boldsymbol{x}})$ is injective.* [12]

This theorem states that distributions of $\boldsymbol{x} \in \Re^D$ have a one-to-one correspondence with mappings $\mu(\boldsymbol{P}_{\boldsymbol{x}})$. Thus, for two distributions $\boldsymbol{P}_{\boldsymbol{x}}$ and $\boldsymbol{P}_{\boldsymbol{y}}$, we can use the function norm $||\mu(\boldsymbol{P}_{\boldsymbol{x}}) - \mu(\boldsymbol{P}_{\boldsymbol{y}})||$ to quantitatively measure the difference (denoted as $D(\boldsymbol{P}_{\boldsymbol{x}}, \boldsymbol{P}_{\boldsymbol{y}})$) between these two distributions. Moreover, we do not need to access the actual distributions but rather finite samples to calculate $D(\boldsymbol{P}_{\boldsymbol{x}}, \boldsymbol{P}_{\boldsymbol{y}})$ because:

Theorem 2. *Assume that $||f||_{\inf} \leq C$ for all $f \in \mathcal{H}$ with $|||f||_{\mathcal{H}} \leq 1$, then with probability at least $1-\theta$, $||\mu(\boldsymbol{P}_{\boldsymbol{x}}) - \mu(\boldsymbol{X})|| \leq 2R_T(\mathcal{H}, \boldsymbol{P}_{\boldsymbol{x}}) + C\sqrt{-T^{-1}log(\theta)}$).* [12]

As long as the Rademacher average is well behaved, finite samples yield error that converges to zero, thus they empirically approximate $\mu(\boldsymbol{P}_{\boldsymbol{x}})$. Therefore, $D(\boldsymbol{P}_{\boldsymbol{x}}, \boldsymbol{P}_{\boldsymbol{y}})$ can be precisely approximated by using finite sample estimation $||\mu(\boldsymbol{X}) - \mu(\boldsymbol{Y})||$.

Thanks to the above facts, we use HED to construct $I_{KTC}(\boldsymbol{X}_{1:T_1}, \boldsymbol{Y}_{1:T_2})$ to measure the consistency between distributions of two segments as follows,

$$\frac{2}{T_1T_2}\sum_{i,j} k(\boldsymbol{x}_i, \boldsymbol{y}_j) - \frac{1}{T_1^2}\sum_{i,j} k(\boldsymbol{x}_i, \boldsymbol{x}_j) - \frac{1}{T_2^2}\sum_{i,j} k(\boldsymbol{y}_i, \boldsymbol{y}_j) \tag{4}$$

Combining eq. 2 and eq. 4, c_{i+1} is estimated by minimizing the following function in matrix formulation as:

$$\mathcal{L}_{\{\boldsymbol{X}_{c_i:c_i+T-1}\}}(c_i') = -(\boldsymbol{E}_T^{c_i'})^H \boldsymbol{K}^{KTC}_{c_i:c_i+T-1} \boldsymbol{E}_T^{c_i'},\ \boldsymbol{E}_T^{c_i'} = \frac{1}{c_i'}\boldsymbol{e}_T^{1:c_i'} - \frac{1}{d_i}\boldsymbol{e}_T^{c_i':T} \quad (5)$$

where c_i' and d_i are short notations for $c_{i+1} - c_i$ and $c_i + T - c_{i+1}$. $\boldsymbol{e}_T^{t_1:t_2} \in \Re^{T\times 1}$ is a binary vector with 1 for positions from t_1 to t_2 and 0 for others. $\boldsymbol{K}^{KTC}_{c_i:c_i+T-1} \in \Re^{T\times T}$ is the kernel matrix based on the kernel function $k_{KTC}(\cdot)$.

Kernel. The success of kernel methods largely depends on the choice of the kernel function [11]. As mentioned before, the difficulty of human motion, is that both spatial and temporal structures are important. Thus, we propose a novel spatio-temporal kernel $k_{KTC}(\cdot)$ as follows,

$$k_{KTC}(\boldsymbol{x}_i, \boldsymbol{x}_j) = k_S(\boldsymbol{x}_i, \boldsymbol{x}_j) k_T(\boldsymbol{x}_i, \boldsymbol{x}_j) = k_S(\boldsymbol{x}_i, \boldsymbol{x}_j) k_T(\widetilde{\Delta}(\boldsymbol{x}_i), \widetilde{\Delta}(\boldsymbol{x}_j)) \quad (6)$$

where $k_S(\cdot)$ is the spatial kernel and $k_T(\cdot)$ is the temporal kernel. $\widetilde{\Delta}(\boldsymbol{x})$ is the estimated local tangent space at point $\boldsymbol{x}$. $k_S(\cdot)$ and $k_T(\cdot)$ can be chosen according to the domain knowledge or universal kernels such as Gaussian. For instance, the canonical component analysis (CCA) kernel [28] is used for joint-position features as,

$$k_S^{CCA}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\lambda_S d_{CCA}(\boldsymbol{x}_i, \boldsymbol{x}_j)^2) \quad (7)$$

where $d_{CCA}(\cdot)$ is the CCA metric based on $M \times 3$ matrix representation of $\boldsymbol{x} \in \Re^{3M\times 1}$[1]. Or in general, we set them as,

$$\begin{aligned} k_S(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \exp(-\lambda_S \parallel \boldsymbol{x}_i - \boldsymbol{x}_j \parallel^2) \\ k_T(\widetilde{\Delta}(\boldsymbol{x}_i), \widetilde{\Delta}(\boldsymbol{x}_j)) &= \exp(-\lambda_T \theta(\widetilde{\Delta}(\boldsymbol{x}_i), \widetilde{\Delta}(\boldsymbol{x}_j))^2) \end{aligned} \quad (8)$$

where λ_S is the kernel parameter for $k_S(\cdot)$ and λ_T is the kernel parameter for $k_S(\cdot)$. $\theta(\cdot)$ is the notation of principal angle between two subspace (range from 0 to $\frac{\pi}{2}$).

In short, the spatio-temporal kernel k_{KTC} captures both *spatial and temporal* distributions of data (a visual example in Fig. 3), which is suitable to model structured sequential data. As special cases, k_{ST} degenerates to spatial kernel if $\lambda_T \to 0$ and to temporal kernel if $\lambda_S \to 0$.

Optimization. Unlike the NP-Hard optimization in spectral clustering [21], eq. 5 can be efficiently solved because the feasible region of c_{i+1} is $[c_i + 1, c_i + T - 1]$, allowing to search the entire space to minimize $\mathcal{L}(c_{i+1})$. For each step, minimizing eq. 5 has complexity at most $O(T^2)$ to access $k_{KTC}(\cdot)$.

3.3 KTC-R

Sequentially optimizing eq. 1 is given in sec. 3.2. However, this process may not be suitable for realtime applications. A key feature of human motion is temporal variations,

[1] M is the number of 3D joints from Mocap or depth sensor.

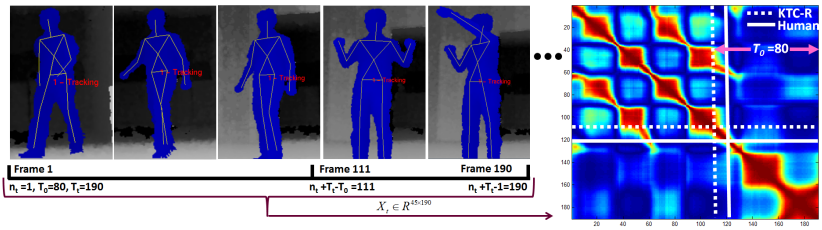

Fig. 3. An illustration of KTC-R. Left: tracked joints from depth sensor, right: $\boldsymbol{K}^{KTC}_{1:190} \in \Re^{190\times 190}$ for window $\boldsymbol{X}_{1:190}$. The decision to make no cut between frame 1 to 110 is made before the current window with a maximum delay of $T_0 = 80$ frames.

i.e., one action can last for a long time or only a few seconds. Thus, it is difficult to use a fixed-length-T sliding-window to capture transitions. Small values of T cause over-segmentation and large values of T cause large delays ($T = 300$ for depth sensor results in 10 secs delay). To overcome this problem, we combine *incremental* sliding-window strategy [1] and two-sample test [14, 13] to design a realtime algorithm for eq. 5, i.e., KTC-R (Fig. 3).

Given $\boldsymbol{X}_{1:L_x} = [\boldsymbol{x}_1, ..., \boldsymbol{x}_{L_x}] \in \Re^{D\times L_x}$, KTC-R sequentially processes the varying-length window $\boldsymbol{X}_t = [\boldsymbol{x}_{n_t}, ..., \boldsymbol{x}_{n_t+T_t}]$ at step t. This process starts from $n_1 = 1$ and $T_1 = 2T_0$, where T_0 is the pre-defined shorest possible action length. At step-t (assume the last cut is c_i), if there is no captured action transition point, the following updating process is performed,

$$n_{t+1} = n_t, T_{t+1} = T_t + \delta T \tag{9}$$

else if there is a transition point,

$$c_{i+1} = n_t + T_t - T_0, n_{t+1} = c_{i+1}, T_{t+1} = T_1 \tag{10}$$

where δT is the step length of increasing the window. This process ends when $n_t - L_x \leq T_0$. As shown in eq. 9 and eq. 10, $\boldsymbol{X}_{1:L_x}$ is sequentially processed and all cuts c_i are estimated when the algorithm requires the $(c_i + T_0 - 1)_{th}$ frame (same for non-cut frames). This fact indicates STC-R has a fixed-length time delay T_0, as shown in Fig. 3.

At each step, deciding on a cut (at frame $n_t + T_t - T_0$) is equivalent to the following hypothesis test,

$$H_0 : \{\boldsymbol{x}_i\}_{n_t}^{n'_t-1} \; and \; \{\boldsymbol{x}_i\}_{n'_t}^{n_t+T_t-1} \; are \; the \; same; \quad H_A : not \; H_0 \tag{11}$$

where n'_t is the short notation for $n_t + T_t - T_0$. eq. 11 is re-written by combining eq. 5 as follows,

$$\begin{aligned} &\mathcal{L}_t = -(\boldsymbol{E}_{T_t}^{n'_t-n_t})^H \boldsymbol{K}^{KTC}_{n_t:n_t+T_t-1} \boldsymbol{E}_{T_t}^{n'_t-n_t} \\ &- H_0 : \mathcal{L}_t \geq \delta_t : n'_t \; is \; not \; a \; cut; \quad - H_A : \mathcal{L}_t < \delta_t : n'_t \; is \; a \; cut \end{aligned} \tag{12}$$

where δ_t is the adaptive threshold for the hypothesis test 12. In fact, eq. 12 is directly inspired by [14] which proposes a kernelized two-sample test method. $\mathcal{L}_t$ is analogous

to the negative square of empirical estimation of Maximum Mean Discrepancy (MMD), which has the following formulation,

$$MMD[\mathcal{F}, \boldsymbol{X}_{1:T_1}, \boldsymbol{Y}_{1:T_2}] = (\frac{1}{T_1^2}\sum_{i,j=1}^{T_1} k(\boldsymbol{x}_i, \boldsymbol{x}_j) - \frac{2}{T_1 T_2}\sum_{i,j=1}^{T_1,T_2} k(\boldsymbol{x}_i, \boldsymbol{y}_j) + \frac{1}{T_2^2}\sum_{i,j=1}^{T_2} k(\boldsymbol{y}_i, \boldsymbol{y}_j))^{\frac{1}{2}} \tag{13}$$

where $\mathcal{F}$ is a unit ball in a universal RKHS $\mathcal{H}$, and $\{\boldsymbol{x}\}_{i=1}^{T_1}$ and $\{\boldsymbol{y}\}_{j=1}^{T_2}$ are i.i.d. samples from distributions $\boldsymbol{P}_{\boldsymbol{x}}$ and $\boldsymbol{P}_{\boldsymbol{y}}$. It can be shown that,

$$\lim_{\lambda_T \to 0} \mathcal{L}_t = -MMD[\mathcal{F}, \boldsymbol{X}_{n_t:n'_t-1}, \boldsymbol{X}_{n'_t:n_t+T_t-1}]^2 \tag{14}$$

if the same kernel in MMD is used as the spatial kernel in $k_{KTC}(\cdot)$ (eq. 6) and $k_T(\cdot)$ degenerates to 1 as $\lambda_T \to 0$. Based on eq. 14, δ_t is set as $B_R(t) + \delta$ where $B_R(t)$ is an adaptive threshold which is calculated from the Rademacher bound [14], and δ is a fixed global threshold which is the only non-trivial parameter in KTC-R (used to control the coarse-fine level of segmentation).

Analysis. In summary, both KTC-S and KTC-R are based on eq. 5. The main differences are, KTC-S performs segmentation by sequential optimization in a *two-class temporal clustering* way, and KTC-R performs segmentation by using incremental sliding-window in a *two-sample test* way. KTC-R requires more sliding-windows than KTC-S, but for each one, there is no optimization, and accessing $k_{KTC}(\cdot)$ $O(T_t \delta T)$ times is enough (linear to T_t). Only when a new cut is detected, $O(T_t^2)$ times accessing is required. Thus, KTC-R is extremely efficient and suitable for realtime applications.

It is notable that, even if the fixed-length sliding-window method (sec. 3.2) is improved to make the decision whether a cut happens or not in $\boldsymbol{X}_{c_i:c_i+T-1}$, a small T is still not reliable for realtime applications. The reason is that a clear temporal cut for human motion requires a large number of observations before and after the cut. Indeed, the required number of frames varies from action to action, even for manual annotation.

4 Online Hierarchical Temporal Segmentation

Besides estimating $\{c_i\}$, decomposing an action segment $\boldsymbol{X}_{c_i:c_{i+1}-1}$ into an unknown number of *action units* (e.g., three walking cycles) if cyclic motions exists, is also needed [29]. This is not only helpful for understanding motion sequences, but also for other applications such as recognition and indexing. Thus, an online cyclic structure segmentation algorithm, i.e., Kernelized Alignment Cut (KAC), is proposed as a generalization of kernel embedding of distributions and temporal alignment [15, 6]. By combining KAC and KTC-R, we get the *two-layer* segmentation algorithm KTC-H, where H stands for hierarchical. Action units segmentation is difficult for non-periodic motions (e.g., jumping), which are actions that are usually performed once locally. However, people can still perform two consecutive non-periodic motions, and these two

motions are not identical because of intra-person variations, which brings challenges for KAC.

KAC. As an online algorithm, KAC utilizes the sliding-window strategy. Each window $\boldsymbol{X}_{a_j+n_t-T_m:a_j+n_t-1}$ is sequentially processed, starting from $n_1 = 2T_m$, $a_1 = c_i$, where a_j is j_{th} action unit cut. T_m is a parameter which is the minimal length of one action units. We empirically find that results are insensitive to T_m.

For each window $\boldsymbol{X}_{a_j+n_t-T_m:a_j+n_t-1}$, this process has two branches. The last action unit continues: $n_{t+1} = n_t + \delta T_m$; or there is a new action unit: $a_{j+1} = a_j + n_t - T_m, n_{t+1} = 2T_m$. Here δT_m is the step length. This process ends when a new cut point c_{i+1} received. Deciding whether $\boldsymbol{X}_{a_j+n_t-T_m:a_j+n_t-1}$ is the start of a new unit or not can be formulated as,

$$\begin{aligned} &\mathcal{S}_t = \mathcal{S}_{Align}(\boldsymbol{X}_{a_j:a_j+T_m-1}, \boldsymbol{X}_{a_j+n_t-T_m:a_j+n_t-1}) \\ &- H_0 : \mathcal{S}_t \leq \epsilon_t : a_j + n_t - T_m \; is \; a \; new \; unit; \quad - H_A : \mathcal{S}_t > \epsilon_t : not \; H_0 \end{aligned} \tag{15}$$

where $\mathcal{S}_{Align}(\cdot)$ is the metric to measure the *structure similarity* between $\boldsymbol{X}_{a_j:a_j+T_m-1}$ and $\boldsymbol{X}_{a_j+n_t-T_m:a_j+n_t-1}$, to handle intra-person variations. ϵ_t is an adaptive threshold (empirically set by cross-validation) and ideally should be close to zero if alignment can perfectly leverage variations. Similar to KTC-R, KAC has delay T_m. In particular, KAC uses dynamic time warping (DTW) [15, 6] to design $\mathcal{S}_{KAC}(\cdot)$ by minimizing the following loss function based on the kernel from eq. 6,

$$\mathcal{S}_{KAC}(\boldsymbol{K}_{a_j:a_j+T_m-1}^{a_j+n_t-T_m:a_j+n_t-1} : \boldsymbol{W}_1, \boldsymbol{W}_2) \tag{16}$$

where $\boldsymbol{K}$ is the cross-kernel matrix for two segments, and $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ are binary temporal warping matrices encoding the temporal alignment path as shown in [15]. Interested readers are referred to [15, 6] for more details about $\mathcal{S}(\cdot)$. Eq. 16 can be optimized by using dynamic programming with complexity $O(T_m^2)$, and $\mathcal{S}_{KAC}(\cdot)$ measure the similarity between the current action unit (a part) and the current window. Importantly, alignment methods such as DTW are not suitable for eq. 12. This is because alignment requires two segments to have roughly the same starting and ending points, which does not hold in eq. 12.

KTC-H. By combining KTC-R and KAC, we can sequentially simultaneously capture action transitions (cuts) and action units, in the integrated algorithm KTC-H. Formally, KTC-H uses the two-layer sliding window strategy, i.e., the outer loop (sec. 3.3) to estimate c_i and the inner loop to estimate a_j between c_i and the current frame from the outer loop. Since KTC-R (eq. 12) and KAC (eq. 15) both have fixed delay (T_0 and T_m), KTC-H is suitable for realtime transition and action unit segmentation.

Discussion. We compare with several related algorithms: (1) Spectral clustering [21] can be extended to temporal clustering if only allowing temporal cuts (TSC) [10]. Similarly, minimizing eq. 5 can be viewed as an instance of TSC motivated by embedding distributions into RKHS. (2) PPCA-CD is proposed in [1] to model motion segments by Gaussian models, where CD stands for change-point detection. Compared to [1], KTC has higher computational cost but gains the ability to handle nonparametric distributions. (3) KTC is similar to KCpA [19], which uses the novel kernel Fisher discriminant ratio. Compared to [19], KTC performs change-point detection by using the

incremental sliding-window. More importantly, KTC detects both change-points and cyclic structures. This is crucial for online recognition, making action can be recognized after only one *unit* instead of the whole action. (4) As an elegant extension of Kernel K-means and spectral clustering, ACA is proposed in [6] for offline temporal clustering. KTC can be viewed as an online complementary approach to [6].

5 Online Action Recognition from OpenNI

We use KTC-H to recognize tracked OpenNI data online, based on labeled Mocap sequences only. To the best of our knowledge, this is the *first work* towards continuous action recognition from OpenNI in the *transfer learning* framework, without the need of training data from OpenNI.

Recognition. In our system, action recognition is done by combining KTC-H and sequences alignment. Assume there are N labeled Mocap segments (action units) $\{\boldsymbol{X}^i_{mocap} \in \Re^{3M\times L_i}\}_{i=1}^N$ associated with label $I^i \in \mathcal{I}$, where $\mathcal{I} = 1, 2, ..., C$ indicates C action classes. Given a segmented action unit $\boldsymbol{X}_{a_j:a_{j+1}-1} \in \Re^{3K\times(a_{j+1}-a_j)}$ from KTC-H, the estimated action label $\overline{I}^{Test}$ is given by using Dynamic Manifold Warping (DMW) [25],

$$\overline{I}^{Test}_{a_j:a_{j+1}-1} = \arg\min_{i\in\{1,2..,N\}} \mathcal{S}_{DMW}(\boldsymbol{X}_{a_j:a_{j+1}-1}, (\boldsymbol{X}^i_{mocap}, I^i)) \tag{17}$$

where K can be M or $\leq M$ when indicating noisy and possible occluded tracking trajectories from depth sensor data by using the OpenNI tracker. The spatial and temporal variations between $\boldsymbol{X}$ and $\boldsymbol{X}^i_{mocap}$ are handled by DMW.

6 Experimental Validation

We evaluate the performance of our system from two aspects: (1) temporal segmentation on depth sensor, Mocap and video, (2) online action recognition on depth sensor.

6.1 Online Temporal Segmentation of Human Action

In this section, quantitative comparison of online temporal segmentation methods is provided. KTC-R is compared with other state-of-the-art methods, i.e, PPCA-CD [1] and TSC-CD [23, 10], where TSC-CD is a change-point detection algorithm based on temporal spectral clustering by our implementation. PPCA-CD uses the same incremental sliding-window strategy as sec. 3.3 and TSC-CD uses the fixed-length sliding window as sec. 3.2. Thresholds (e.g., δ for KTC-R and thresholds for other methods) are set by cross-validation on one sequence. Methods like ACA [6] and [22] can not be directly compared since they are offline. Results are evaluated by three metrics, i.e., precision, recall and rand index. The first two are for cut points and the last one is for all frames. The ground-truth for rand index (RI) is labeled as consecutive numbers $1,\ 2,\ 3,\ ...$ for different segments. Importantly, T_0 is set as $80, 250, 60$ for depth sensor, Mocap and video respectively, making KTC-R have 2.3, 2.1 and 1 seconds delay.

Fig. 4. Examples of Online Temporal Segmentation. Top (Depth Sensor): a sequence includes 3 segments as walking, boxing and jumping. Noisy joint-trajectories are tracked by OpenNI. Middle (Mocap): a sequence has 4579 frames and 7 action segments. Bottom (Video): a clip includes walking and running. For all cases, KTC-R achieves the highest accuracy.

KTC-S achieves similar accuracy to KTC-R but with longer delay, details are omitted due to lack of space. Results are very robust to T_0 and δT. For instance, we got almost identical results when T_0 ranges from 60 to 120 in OpenNI data.

Depth Sensor. To validate online temporal segmentation on depth sensor, 10 human motion sequences are captured by the PrimeSense sensor. Each sequence is a combination of 3 to 5 actions (e.g., walking to boxing) with length around 700 frames (30Hz). For human pose tracking, we use the available OpenNI tracker to automatically track joints on human skeleton. $K \in [12, 15]$ key points are tracked, resulting in joint 3D position $\boldsymbol{x}_t$ in $\Re^{36}$ to $\Re^{45}$. Although human pose tracking results are often noisy (Fig. 4 and Fig. 5), we can correctly estimate action transitions from these noisy tracking

Table 1. Temporal Segmentation Results Comparison. Precision (P), recall (R) and rand index (RI) are reported

Methods	PPCA-CD (online)	TSC-CD (online)	**KTC-R** (online)
Depth	0.73(P)/0.78(R)/0.80(RI)	0.77(P)/0.81(R)/0.81(RI)	**0.87**(P)/**0.93**(R)/**0.88**(RI)
Mocap	0.85(P)/0.90(R)/0.90(RI)	0.83(P)/0.86(R)/0.88(RI)	**0.86**(P)/**0.91**(R)/**0.92**(RI)
Video	–	0.78(P)/0.85(R)/0.82(RI)	**0.85**(P)/**0.92**(R)/**0.88**(RI)

results. In particular, KTC-R ($\delta T = 30$) significantly improves the accuracy from other methods (Table 1). The main reason is the joint-positions of noisy tracked joints have *complex nonparametric* structures, which is handled by kernel embedding of distributions [14, 13, 12] in KTC.

−KTC-H. Besides action transitions, results on detecting both cyclic motions and transitions are reported by performing KTC-H ($T_m = 50, \delta T_m = 1$). Since other methods don't have the module, we report quantitative comparison on online hierarchial segmentation by using KTC-H or other methods plus our KAC algorithm in sec. 4. Results show (Table 2) that KTC-H gets higher accuracy than other combinations. It is notable that, because of the natural of RI, the RI metric will increase when the number of cuts increase, even for low P/R, which is the case for hierarchial segmentation (including two types of cuts).

Mocap. Similar to [1, 6], $M = 14$ joints are used to represent the human skeleton, resulting in joint-quaternions of joint angles in 42D. Online temporal segmentation methods are tested on 14 selected Mocap sequences from subject 86 in CMU Mocap database. Each sequence is a combination of roughly 10 action segments and in total there are around 10^5 frames (120Hz). Since the implementation of PPCA-CD differs from [1] (such as only forward pass is allowed in our experiments), results are not the same as in [1]. Table 1 shows that the gain of KTC-R ($\delta T = 50$) to other methods in Mocap is reduced, compared with depth sensor data. This is because the Gaussian property is more likely to hold for quaternions representation of *noiseless* Mocap data, which is not the case for real data in general.

Video. Furthermore, KTC-R is performed on a number of sequences from HumanEva-2, which is a benchmark for human motion analysis [30]. Silhouettes are extracted by background substraction, resulting in a sequence of binary masks (60Hz). $\boldsymbol{x}_t \in \Re^{D_t}$ is set as the vector representation of the mask at frame t. It is notable that, D_t (size of masks) in different frames may not be identical, so PPCA-CD can not be applied. This fact supports the advantage of KTC, which is applicable for complex sequential data as long as a (pseudo) kernel can be defined. In particular, we follow [6] to compute the matching distance of silhouettes to set the kernel. Results are shown in Fig. 4 and Table. 1. As a reference, state-of-the-art *offline* temporal clustering method ACA achieves higher accuracy than KTC-R on Mocap (96% precision). However, offline methods (1) are not suitable for real-time applications, and (2) require the number of clusters (segments) to be set in advance, which is not applicable in many cases.

6.2 Joint Online Segmentation and Recognition from OpenNI

We collect additional 5109 frames ($N = 30$) with 10 primitive actions from CMU Mocap as the labeled (training) data for recognition. In order to associate labeled Mocap sequences with data from other domains, joint-position trajectories ($M = 15$) are used in eq. 17 [25]. Testing data are previous collected sequences from depth sensor, and online segmentation and recognition are simultaneously performed by KTC-H and eq. 17. A significant feature of our approach is, there is *no extra-training* process for depth sensor, i.e., the knowledge from Mocap can be *transferred* to other motion

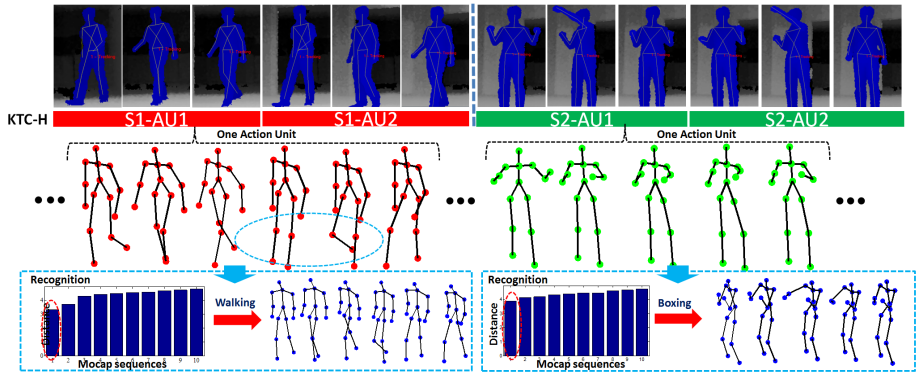

Fig. 5. Online action segmentation and recognition on 2.5**D depth sensor.** Top to bottom, depth image sequences, KTC-H results and action recognition results. For segmentation, blue line indicates the cut and different rectangles indicate different action units. The blue circle indicates noisy tracking results. For recognition: distance to labeled Mocap sequences, and inferred $3M$D motion sequences.

sequences, based on proper features. Tracked trajectories from OpenNI in an action unit (segmented by KTC-H) are associated with labeled Mocap sequences from 10 action categories.

Table 2. Online hierarchial segmentation and recognition on 2.5D depth sensor

Methods	PPCA-CD+KAC	**KTC-H**
Depth	0.72(P)/0.76(R)/0.89(RI)/0.71(Acc)	**0.85**(P)/**0.87**(R)/**0.94**(RI)/**0.85**(Acc)

Although OpenNI tracking results are often noisy (highlighted by blue circles in Fig. 5), we achieve 85% recognition accuracy (Acc) from these noisy tracking results (Table 2), without any additional training on depth sensor data. This result does not only benefit from DMW [25], but also from KTC-H. DMW requires the input only contain one action unit, while KTC-H performs a critical missing step, i.e., accurate online temporal segmentation, in order to perform recognition. As illustrated in Table 2, the accuracy on OpenNI is enhanced from 0.71 to 0.85, which strongly supports the effectiveness of KTC-H. Furthermore, the complete and accurate $3M$D human motion sequences can be inferred by associating the learned manifolds from Mocap.

7 Conclusion

In this paper, we propose an online temporal segmentation method KTC, as a temporal extension of Hilbert space embedding of distributions for change-point detection based on the novel spatio-temporal kernel. Furthermore, a realtime implementation of KTC

and a hierarchial extension are designed, which can detect both action transitions and action units. Based on KTC, we achieve realtime temporal segmentation on both Mocap, motion sequences from 2.5D depth sensor and 2D videos. Furthermore, temporal segmentation is combined with alignment, resulting in realtime action recognition on depth sensor input, without the need of training data from depth sensor.

Acknowledgments. This work was supported in part by NIH Grant EY016093 and DE-FG52-08NA28775 from the U.S. Department of Energy.

References

1. Barbic, J., Safonova, A., Pan, J.Y., Faloutsos, C., Hodgins, J.K., Pollard, N.S.: Segmenting motion capture data into distinct behaviors. In: Proc. Graphics Interface, pp. 185–194 (2004)
2. Lv, F., Nevatia, R.: Recognition and Segmentation of 3-D Human Action Using HMM and Multi-class AdaBoost. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006, Part IV. LNCS, vol. 3954, pp. 359–372. Springer, Heidelberg (2006)
3. Laptev, I., Marszalek, M., Schmid, C., Rozenfeld, B.: Learning realistic human actions from movies. In: Proc. CVPR (2008)
4. Weinland, D., Özuysal, M., Fua, P.: Making Action Recognition Robust to Occlusions and Viewpoint Changes. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part III. LNCS, vol. 6313, pp. 635–648. Springer, Heidelberg (2010)
5. Zhong, H., Shi, J., Visontai, M.: Detecting unusual activity in video. In: Proc. CVPR, pp. 816–8231 (2004)
6. Zhou, F., De la Torre, F., Hodgins, J.K.: Hierarchical aligned cluster analysis for temporal clustering of human motion. Under review at IEEE PAMI (2011)
7. Niebles, J.C., Chen, C.-W., Fei-Fei, L.: Modeling Temporal Structure of Decomposable Motion Segments for Activity Classification. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part II. LNCS, vol. 6312, pp. 392–405. Springer, Heidelberg (2010)
8. Chen, J., Gupta, A.: Parametric Statistical Change-point Analysis. Birkhäuser (2000)
9. Adams, R.P., MacKay, D.J.: Bayesian online changepoint detection. University of Cambridge Technical Report (2007)
10. De la Torre, F., Campoy, J., Ambadar, Z., Conn, J.F.: Temporal segmentation of facial behavior. In: Proc. ICCV (2007)
11. Hofmann, T., Scholkopf, B., Smola, A.: Kernel methods in machine learning. Annals of Statistics 36, 1171–1220 (2008)
12. Smola, A.J., Gretton, A., Song, L., Schölkopf, B.: A Hilbert Space Embedding for Distributions. In: Hutter, M., Servedio, R.A., Takimoto, E. (eds.) ALT 2007. LNCS (LNAI), vol. 4754, pp. 13–31. Springer, Heidelberg (2007)
13. Gretton, A., Fukumizu, K., Harchaoui, Z., Sriperumbudur, B.: A fast, consistent kernel two-sample test. In: NIPS, vol. 19, pp. 673–681 (2009)
14. Gretton, A., Borgwardt, K., Rasch, M., Scholkopf, B., Smola, A.: A kernel method for the two-sample-problem. In: NIPS, vol. 19, pp. 513–520 (2007)
15. Zhou, F., De la Torre, F.: Canonical time warping for alignment of human behavior. In: NIPS, vol. 22, pp. 2286–2294 (2009)
16. Xuan, X., Murphy, K.: Modeling changing dependency structure in multivariate time series. In: Proc. ICML (2007)
17. Saatci, Y., Turner, R., Rasmussen, C.: Gaussian process change point models. In: Proc. ICML (2010)
18. Desobry, F., Davy, M., Doncarli, C.: An online kernel change detection algorithm. IEEE Trans. on Signal Processing 53, 2961–2974 (2005)

19. Harchaoui, Z., Bach, F., Moulines, E.: Kernel change-point analysis. In: NIPS 21, pp. 609–616 (2009)
20. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: NIPS, vol. 14, pp. 849–856 (2002)
21. von Luxburg, U.: A tutorial on spectral clustering. Statistics and Computing 17, 395–416 (2007)
22. Fox, E., Sudderth, E., Jordan, M., Willsky, A.: Nonparametric bayesian learning of switching linear dynamical systems. In: NIPS 21, pp. 457–464 (2009)
23. Zelnik-Manor, L., Irani, M.: Statistical analysis of dynamic actions. IEEE PAMI 28, 1530–1535 (2006)
24. Satkin, S., Hebert, M.: Modeling the Temporal Extent of Actions. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part I. LNCS, vol. 6311, pp. 536–548. Springer, Heidelberg (2010)
25. Gong, D., Medioni, G.: Dynamic manifold warping for view invariant action recognition. In: Proc. ICCV, pp. 571–578 (2011)
26. Hoai, M., Lan, Z., De la Torre, F.: Joint segmentation and classification of human actions in video. In: Proc. CVPR (2011)
27. Faivishevsky, L., Goldberger, J.: A nonparametric information theoretic clustering algorithm. In: Proc. ICML, pp. 351–358 (2010)
28. Bach, F.R., Jordan, M.I.: Kernel independent component analysis. JMLR 3, 1–48 (2003)
29. Laptev, I., Belongie, S., Perez, P., Wills, J.: Periodic motion detection and segmentation via approximate sequence alignment. In: Proc. ICCV, pp. 816–8231 (2005)
30. Sigal, L., Balan, A.O., Black, M.J.: Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. IJCV 87, 4–27 (2010)