Marginalized Bags of Vectors Kernels on Switching Linear Dynamics for Online Action Recognition

Masamichi Shimosaka, Taketoshi Mori, Tatsuya Harada and Tomomasa Sato
Graduate School of Information Science and Technology
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan
{*simosaka, tmori, harada, tomo*}*@ics.t.u-tokyo.ac.jp*

***Abstract*— In this paper, we propose a novel kernel computation algorithm between time-series human motion data for online action recognition. The proposed kernel is based on probabilistic models called switching linear dynamics (SLDs). SLD is one of the powerful tools for tracking, analyzing and classifying human complex time-series motion. The proposed kernel incorporates information about the latent variables in SLDs with simplified designing approach called marginalized kernels. The empirical evaluation using real motion data shows that a classifier using SVM with our proposed kernel has much better performance than the classifier with some conventional kernel techniques. Another experiment using walking around motion shows that a classifier with the proposed kernel can properly segment the start and the end of the target action.**

***Index Terms*— Mixed-State Dynamics, Probabilistic Product Kernel, Complex Motion, Motion Capture Data**

I. Introduction

Recognizing human action is one of essential foundations to achieve smooth communication between intelligent systems, especially robots, and human. It is also a key technical element in achieving analysis and surveillance of human activity by intelligent systems. We have built a recognition algorithm and system for human daily life action [1]. This is based on a statistical learning algorithm using kernels [2] that compute similarity between the motions. The approach using kernels has several advantages in the following aspects. The systems with kernels can use robust learning algorithms such as those that support vector machines and Gaussian processes. The recognition processes in the system can be unified because the kernels can absorb the difference between several type of data structures.

In general, it is well known that the performance of a classifier using kernel is very dependent on kernel itself. If a kernel cannot reflect the property of target input data, the machine fails to attain desirable performance. In the area of action recognition, the property of motion must be studied because action is a symbol of time-series motion. In example of action walking, the feet motion pattern must be addressed. Thus, we must incorporate the following property to model the motion data. The first property to be incorporated is symbolization scheme because action should be handled with symbols so as to manipulate or interpret of motion easily. The second property is variation of time-series motion in time and space, because time-series motion such as walking and running has wide variety of motions.

Although hidden Markov model (HMM) is one of the good probabilistic models to satisfy the above properties and is also used by many action recognition researchers [3], [4], HMM has shortage in the following aspect. The main weak point of HMM is that it is hard for HMM itself to handle both dynamic property of human motion and measurement error at the same time. This is because HMM is originally designed for discrete dynamics and their observation is independent at each time. Modeling dynamical actions, such as raising hand and walking, requires properties of both dynamics and measurement error.

Recently, a flexible probabilistic model as an alternative for HMM, called switching linear dynamics (SLDs) has been studied [5], SLDs incorporate both intuitive symbolic representation, Markov property and dynamics property of motion, linear dynamics. At this time this approach looks to have the potential to solve the problem of HMM for recognizing dynamical actions, because dynamical action like walking has strong non-linearity and the property of the dynamics changes drastically with times. It is natural to incorporate SLDs with kernel methods for recognizing dynamical action such as walking and running. To the best our knowledge, there are two types of kernel computation algorithm with SLDs. The first technique is based on Monte Carlo methods [6] and the second is based on the Fisher score derived from the Markov chain in SLDs [7]. Both methods have deficiencies in case of online use. The first technique requires a very large amount of computation to optimize the parameters of SLDs and Monte Carlo integration. The second lacks the capability of handling dynamics property.

In order to realize a novel kernel computation that is better than the conventional techniques, we need to focus on the following properties. Our first concern is that the computational cost per kernel must be very small. Our second concern is that kernel be able to incorporate the property of dynamics of motion. Based on these considerations, we propose a novel kernel that can use and adopt techniques of general design policy with latent probabilistic models called marginalized kernels [8]. This is because SLDs can be categorized into latent probabilistic models. And the marginalized kernels have the following good properties. The marginalized kernels have very high tolerance for noisy

data. The marginalized kernels allow a designer of kernels to make new kernel between complex structured data with combination of simple and robust kernel instead of making a new complex kernel.

There are several kernel computation methods based on probabilistic model without restriction for specific models. But all of them are difficult to apply for SLDs. The Fisher kernel [9] is a simple and natural framework for any probabilistic models. The Fisher kernel has an advantage because of automatic derivation once the probability model is assigned, however, "curse of dimensionality" may occur in SLDs case. This is because the number of the parameters of SLDs is very large with dimension equal to the Fisher score. There are some smart kernel methods computed with integral operation [10], [11], however, the difficulties on integrating some parameters in SLDs will arise. Smola et al. [12] derives an elegant closed-formed kernel for dynamics, especially linear dynamics (LDs), however, this kernel cannot run in online action recognition. This is because the start or end point of the two time-series motion cannot be clearly given a priori in online action recognition.

II. Switching Linear Dynamics and Marginalized Kernels

In this section, SLDs and marginalized kernel, the basis of the proposed kernel, are introduced briefly. Details of these basic components are in Ref. [5], [8], [13].

A. Switching Linear Dynamics: SLDs

Formulation as Stochastic Process: SLDs are stochastic processes and can be interpreted as combination of HMM and LDs. The system can be described using the following set of state-space equations for the physical system and symbolic transition with Markov chain.

$$\begin{aligned} p(\mathcal{X},\mathcal{Y},\mathcal{S}) &= p(\boldsymbol{y}_1|\boldsymbol{x}_1)p(\boldsymbol{x}_1|\boldsymbol{s}_1)p(\boldsymbol{s}_1) \\ &\quad \prod_{t=2}^{T} p(\boldsymbol{y}_t|\boldsymbol{x}_t)p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1},\boldsymbol{s}_t)p(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}), \\ p(\boldsymbol{y}_t|\boldsymbol{x}_t) &= \mathcal{N}\left(C\boldsymbol{x}_t, V\right), \quad p(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}) = \boldsymbol{s}_t^{\mathsf{T}}\Pi\boldsymbol{s}_{t-1}, \\ p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1},\boldsymbol{s}_t) &= \mathcal{N}\left(A\boldsymbol{x}_{t-1} + D\boldsymbol{s}_t, W\right), \\ p(\boldsymbol{x}_1|\boldsymbol{s}_1) &= \mathcal{N}\left(D_1\boldsymbol{s}_1, W_1\right), \quad p(\boldsymbol{s}_1) = \boldsymbol{s}_1^{\mathsf{T}}\boldsymbol{\pi}. \end{aligned}$$

The meaning of the variables is as follows: $\boldsymbol{y}_t \in \mathbb{R}^m$ denotes measured vector in time t, $\boldsymbol{s}_t \in \mathbb{R}^L$ represents symbolic hidden state, $\boldsymbol{x}_t \in \mathbb{R}^d$ is continuous state space vector in dynamics. For example, position of head or foot can be a candidate for $\boldsymbol{x}$. Although $\boldsymbol{s}_t$ is a vector, this serves as a symbol. If discrete state is at symbol i, $\{\boldsymbol{s}_t\}_j = \delta_{ij}$. Parameters A, W, W_1, C, V are the typical parameters in LDs. The state initialization vector and its transition matrix in Markov chain are written with $\boldsymbol{\pi}, \Pi$. Parameters D, D_1 serves as a driving force converter from symbols to LDs. Variables $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$, $\mathcal{Y} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T\}$, $\mathcal{S} = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_T\}$ represent the sequences with length T. We denote $\boldsymbol{\theta}$ as a whole set of parameters in SLDs, i.e. $\boldsymbol{\theta} = \{A, C, D, D_1, W, W_1, V, \boldsymbol{\pi}, \Pi\}$. The graphical model of SLDs is shown in Fig. 1.

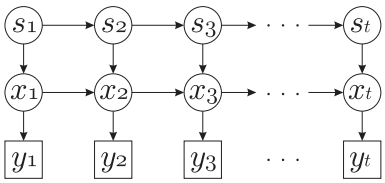

Fig. 1. Graphical model of switching linear dynamics

Estimating Hidden State Space and Parameters: When we use hidden variables $\mathcal{X}, \mathcal{S}$ as cues to compute similarity, the conditional posterior probability $p(\mathcal{X}, \mathcal{S}|\mathcal{Y})$ must be estimated. There are two types of the approximation estimation method for the posterior probability. One is based on sequential Monte Carlo methods, sometimes called CONDENSATION [14]. The other is based on the factorized method with variational parameters [5]. The last technique estimate the probability by factorizing $p(\mathcal{S}, \mathcal{X}|\mathcal{Y}) \approx Q(\mathcal{X})Q(\mathcal{S})$. The parameters in SLDs are optimized with an expectation maximization (EM) algorithm [15]. This algorithm can be formulated as an iteration of the following parameter updating until parameters $\boldsymbol{\theta}^{\mathsf{old}}$ converge as

$$\boldsymbol{\theta}^{\mathsf{old}} \leftarrow \arg\max_{\boldsymbol{\theta}} \mathcal{E}_{\mathcal{X},\mathcal{S}}\left[\ln p(\mathcal{X}, \mathcal{S}, \mathcal{Y}; \boldsymbol{\theta})|\mathcal{Y}; \boldsymbol{\theta}^{\mathsf{old}}\right], \qquad (1)$$

where operation $\mathcal{E}$ denotes expectation as $\mathcal{E}_x\left[f(x)|z\right] = \int p(x|z)f(x)dx$. In case of SLDs, the expectation in (1) is computed with $Q(\mathcal{X})Q(\mathcal{S})$ because the conditional posterior probability cannot be acquired analytically. The parameters updating is iteratively executed in the same way as in LDs and HMM.

B. Marginalized Kernels

The marginalized kernel proposed by Tsuda et al. [8] is a general design framework of a kernel for data modeled with a latent (hidden) variable probabilistic model. In the marginalized kernel, the similarity between $\boldsymbol{x}$, $\tilde{\boldsymbol{x}}$ can be formulated as follows.

$$K(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = \int p(\boldsymbol{h}|\boldsymbol{x})p(\tilde{\boldsymbol{h}}|\tilde{\boldsymbol{x}})K_{\boldsymbol{z}}(\boldsymbol{z}, \tilde{\boldsymbol{z}})d\boldsymbol{h}d\tilde{\boldsymbol{h}} \qquad (2)$$

where $\boldsymbol{h}$ denotes a hidden variable of the model and $\boldsymbol{z} = \{\boldsymbol{x}, \boldsymbol{h}\}$ is a joint variable of the model which can be called complete data in EM algorithm. Function $K_{\boldsymbol{z}}(\boldsymbol{z}, \tilde{\boldsymbol{z}})$ is called a joint kernel. The role of joint kernel is similar to the complete data probabilistic function of latent probabilistic model. Thus, a joint kernel is often designed by combining a simple kernel function.

As Tsuda et al. proves, a Fisher kernel in a latent variable probabilistic model is a special case of a marginalized kernel with the same probabilistic model. Automatic derivation for the Fisher kernel is a desirable property but the dimension of the Fisher score of SLDs is too large to obtain good performance from the data. Instead, there is a room for the designer of a kernel to make a simple and efficient kernel with a joint kernel $K_{\boldsymbol{z}}$.

III. Marginalized Bags of Vectors Kernels

This section describes the details of our proposed kernel computation algorithm. First, we define the marginalized kernel on SLDs. Next, a simple kernel computation using a bags of vectors representation is introduced as a core component in the joint kernel. Finally, we derive the formulation of our proposed kernel.

A. Definition of Marginalized Kernel with SLDs

When two time-series motion $\mathcal{Y} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_T\}$, $\tilde{\mathcal{Y}} = \{\tilde{\boldsymbol{y}}_1, \tilde{\boldsymbol{y}}_2, \ldots, \tilde{\boldsymbol{y}}_{\tilde{T}}\}$ can be modeled with SLDs, the marginalized kernel can be formulated as

$$K(\mathcal{Y}, \tilde{\mathcal{Y}}) = \int \sum_{\mathcal{S}, \tilde{\mathcal{S}}} Q(\mathcal{X}, \mathcal{S}) Q(\tilde{\mathcal{X}}, \tilde{\mathcal{S}}) K_{\mathcal{Z}}(\mathcal{Z}, \tilde{\mathcal{Z}}) d\mathcal{X} d\tilde{\mathcal{X}}. \quad (3)$$

In this paper, we take notice of the symbolic state in SLDs and design joint kernel $K_{\mathcal{Z}}(\mathcal{Z}, \tilde{\mathcal{Z}})$ as a combination of LDs in the following formulation as

$$K_{\mathcal{Z}}(\mathcal{Z}, \tilde{\mathcal{Z}}) = \sum_{l=1}^{k} n_l \tilde{n}_l K_{\mathcal{Z}}^{(l)}(\mathcal{Z}, \tilde{\mathcal{Z}}), \quad (4)$$

where n_l denote the ratio of time when symbol l occurs in T frame: $n_l = \sum_{t=1}^{T} \delta(s_t = l)/T$. $K_{\mathcal{Z}}^{(l)}(\mathcal{Z}, \tilde{\mathcal{Z}})$ denotes a similarity value between data in symbol state l, $[\mathcal{Y}^{(l)}, \mathcal{X}^{(l)}, \mathcal{S}^{(l)}]$ via unimodal LDs of symbol l. This design policy enables us to concentrate on designing kernels on unimodal LDs instead of designing kernels of SLDs.

B. Bags of Vectors Kernels via LDs

Design Policy of the Kernels: why Bags of Vectors?: For online recognition, it is difficult to use an alignment technique such as dynamic time warping [16], because the alignment technique requires the start, end, or a corresponding point between two time-series data and these point in input motion cannot be clearly given a priori in online action recognition case. In case of simple dynamic time warping, the role of reference motion and input motion is clearly different, while the role of reference and input is handled equally in kernel computation. In addition to this, the time series motion may be input intermittently in the online tasks. The features of the time-series motion should not depend on its length, because the length of two time-series motions is usually not the same. Fisher score can resolve this problem but in doing so produces a hard problem; the curse of dimensionality.

To avoid these problems we adopt a bags of vectors (BoV) representation proposed by Jebara [17]. BoV is a natural extension of the bags of words (BoW) representation, a well-known classical feature representation in text domain. Similarity between data denoted by BoV is derived from its frequency and probability density function, similar to BoW. The reason why BoV can avoid the above problems is that BoV neglects the order and the length of the data.

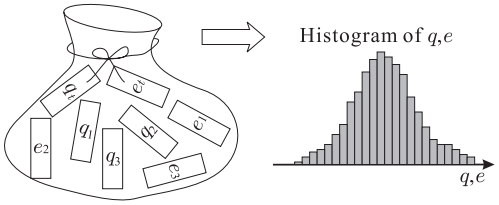

Fig. 2. The bag of vectors represent the data of the linear dynamics. The tuples $\boldsymbol{q}_\cdot$, $\boldsymbol{e}_\cdot$ represent differential information such that, $\boldsymbol{q}_t = \boldsymbol{x}_t - A\boldsymbol{x}_{t-1} - \boldsymbol{d}$, $\boldsymbol{e}_t = \boldsymbol{y}_t - C\boldsymbol{x}_t$. The distribution of $\boldsymbol{q}$, $\boldsymbol{e}$ has information of the data.

Derivation of Bags of Vectors Kernels: When $\mathcal{Y}, \mathcal{X}$ and $\tilde{\mathcal{Y}}, \tilde{\mathcal{X}}$ can be acquired in a unimodal linear dynamics, we derive the kernel function with BoV representation as follows. At first, we assume that the time-series data are approximately generated from the following systems,

$$\begin{aligned} p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) &= \mathcal{N}(A\boldsymbol{x}_{t-1} + \boldsymbol{d}, W) \\ p(\boldsymbol{y}_t|\boldsymbol{x}_t) &= \mathcal{N}(C\boldsymbol{x}_t, V). \end{aligned}$$

In the above settings, $\boldsymbol{q}_t = \boldsymbol{x}_t - A\boldsymbol{x}_{t-1} - \boldsymbol{d}$ and $\boldsymbol{e}_t = \boldsymbol{y}_t - C\boldsymbol{x}_t$ are approximately generated from $\mathcal{N}(\boldsymbol{0}, W)$, $\mathcal{N}(\boldsymbol{0}, V)$ at each time. In reality, the set of $\boldsymbol{q}$ and $\boldsymbol{e}$ differ slightly from the ideal zero mean Gaussian. This difference serves as a cue to computing similarity between two time-series data. Thus we derive the similarity between two time-series motion via unimodal LD using the set of $\boldsymbol{q}$ and $\boldsymbol{e}$, the bags of vectors. The image of BoV is shown in Fig. 2.

We can compute the similarity between data represented with BoV by using information from its probability density function. We adopt probability product kernel (PPK) [6], the kernel technique with probability density function, because this kernel computation was originally designed for a natural settings for BoV representation. Especially, PPK can be formulated as

$$K([\mathcal{X}, \mathcal{Y}], [\tilde{\mathcal{X}}, \tilde{\mathcal{Y}}]) = \int \Big(p_D(\boldsymbol{q}) p_O(\boldsymbol{e}) \tilde{p}_D(\boldsymbol{q}) \tilde{p}_O(\boldsymbol{e}) \Big)^{\rho} d\boldsymbol{q} d\boldsymbol{e}.$$

where the probability density function of $\boldsymbol{q}$ and $\boldsymbol{e}$ can be written as $p_D(\boldsymbol{q})$, $p_O(\boldsymbol{e})$ in time-series data $\mathcal{X}, \mathcal{Y}$, also $\tilde{p}_D(\boldsymbol{q})$, $\tilde{p}_o(\boldsymbol{e})$ in $\tilde{\mathcal{X}}$, $\tilde{\mathcal{Y}}$, and the parameter $\rho > 0$ serves as an adjustable coefficient.

The most important things when we use PPK is selecting the model for probability density function p_D, p_O. In this paper, we adopted the Gaussian distribution $p_D(\boldsymbol{q}) = \mathcal{N}(\boldsymbol{\zeta}, W)$, $\tilde{p}_D(\boldsymbol{q}) = \mathcal{N}(\tilde{\boldsymbol{\zeta}}, W)$, $p_O(\boldsymbol{e}) = \mathcal{N}(\boldsymbol{\eta}, V)$, $\tilde{p}_O(\boldsymbol{e}) = \mathcal{N}(\tilde{\boldsymbol{\eta}}, V)$ because of its simplicity. In general, we can write the PPK between Gaussian distributions as

$$\begin{aligned} K(p, \tilde{p}) &= \left(\frac{(2\pi)^{1-2\rho}}{\rho} \right)^{\frac{\gamma}{2}} |\Sigma^{\dagger}|^{\frac{1}{2}} |\Sigma|^{-\frac{\rho}{2}} |\tilde{\Sigma}|^{-\frac{\rho}{2}} \\ &\quad \exp\left(-\frac{\rho}{2} \left[\boldsymbol{\mu}^{\mathsf{T}} \Sigma^{-1} \boldsymbol{\mu} + \tilde{\boldsymbol{\mu}}^{\mathsf{T}} \tilde{\Sigma}^{-1} \tilde{\boldsymbol{\mu}} - \boldsymbol{\mu}^{\dagger \mathsf{T}} \Sigma^{\dagger} \boldsymbol{\mu}^{\dagger} \right] \right), \end{aligned}$$

where p and $\tilde{p}$ denotes the Gaussian probability density function $p = \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, $\tilde{p} = \mathcal{N}(\tilde{\boldsymbol{\mu}}, \tilde{\Sigma})$ and γ represents the dimensionality of $\boldsymbol{\mu}$, i.e. $\boldsymbol{\mu}, \tilde{\boldsymbol{\mu}} \in \mathbb{R}^{\gamma}, \Sigma, \tilde{\Sigma} \in \mathbb{R}^{\gamma \times \gamma}$. The other parameters $\Sigma^{\dagger}, \boldsymbol{\mu}^{\dagger}$ can be written as $\Sigma^{\dagger -1} = \Sigma^{-1} + \tilde{\Sigma}^{-1}$, $\boldsymbol{\mu}^{\dagger} = \Sigma^{-1}\boldsymbol{\mu} + \tilde{\Sigma}^{-1}\tilde{\boldsymbol{\mu}}$. When we set $\Sigma = \tilde{\Sigma}$,

the kernel value can be written as

$$K(p,\tilde{p}) = (2\rho)^{-\frac{\gamma}{2}}|2\pi\Sigma|^{\frac{1-2\rho}{2}}\exp\left(-\frac{\rho}{4}\mathrm{MD}_\Sigma(\boldsymbol{\mu},\tilde{\boldsymbol{\mu}})\right)$$

where operator $\mathrm{MD}_\Sigma(\cdot,\cdot)$ denotes the Mahalanobis distance as $\mathrm{MD}_\Sigma(\boldsymbol{\mu},\tilde{\boldsymbol{\mu}}) = (\boldsymbol{\mu}-\tilde{\boldsymbol{\mu}})^\mathsf{T}\Sigma^{-1}(\boldsymbol{\mu}-\tilde{\boldsymbol{\mu}})$. Thus the kernel between two time-series motion via a unimodal LDs with BoV representation can be written as

$$\begin{aligned} K_{\mathcal{Z}}^{(l)}(\mathcal{Z},\tilde{\mathcal{Z}}) &= (2\rho)^{-\frac{d+m}{2}}|2\pi W|^{\frac{1-2\rho}{2}}|2\pi V|^{\frac{1-2\rho}{2}} \\ &\quad \exp\left(-\frac{\rho}{4}\mathrm{Df}(\boldsymbol{\zeta}_l,\tilde{\boldsymbol{\zeta}}_l,\boldsymbol{\eta}_l,\tilde{\boldsymbol{\eta}}_l)\right), \end{aligned} \tag{5}$$

where $\mathrm{Df}(\boldsymbol{\zeta}_l,\tilde{\boldsymbol{\zeta}}_l,\boldsymbol{\eta}_l,\tilde{\boldsymbol{\eta}}_l) \equiv \mathrm{MD}_W(\boldsymbol{\zeta}_l,\tilde{\boldsymbol{\zeta}}_l) + \mathrm{MD}_V(\boldsymbol{\eta}_l,\tilde{\boldsymbol{\eta}}_l)$, the mean parameters $\boldsymbol{\zeta}_l$, $\boldsymbol{\eta}_l$ can be defined as

$$\begin{aligned} \boldsymbol{\zeta}_l &= \frac{1}{n_l}\sum_{t=2}^{T}\delta(s_t=l)\left(\boldsymbol{x}_t - A\boldsymbol{x}_{t-1} - \boldsymbol{d}_l\right), \\ \boldsymbol{\eta}_l &= \frac{1}{n_l}\sum_{t=2}^{T}\delta(s_t=l)\left(\boldsymbol{y}_t - C\boldsymbol{x}_t\right), \end{aligned} \tag{6}$$

and $\boldsymbol{d}_l$ represents l-th column of D.

C. Marginalizing Bags of Vectors Kernels

Following from (3), (4), (5), (6), our proposed marginalized kernel can be written as $K(\mathcal{Y},\tilde{\mathcal{Y}}) = \sum_{l=1}^{L}\nu_l\tilde{\nu}_l K^{(l)}(\mathcal{Y},\tilde{\mathcal{Y}})$, where $\nu_l = \sum_{t=1}^{T}p(s_t=l|\mathcal{Y})/T$ and the component kernel $K^{(l)}(\mathcal{Y},\tilde{\mathcal{Y}})$ can be derived from $K_{\mathcal{Z}}^{(l)}(\mathcal{Z},\tilde{\mathcal{Z}})$ by marginalizing with $Q(\mathcal{X}), Q(\mathcal{S})$. Because of the difficulty in marginalizing with $Q(\mathcal{X})$, we derive the marginalized value $K^{(l)}(\mathcal{Y},\tilde{\mathcal{Y}})$ by approximating the distribution of $\langle\boldsymbol{q}\rangle$, $\langle\boldsymbol{e}\rangle$, the marginalized value with $Q(\mathcal{S})$ and $Q(\mathcal{X})$, as Gaussian distribution. Then the value is computed as

$$\begin{aligned} K^{(l)}(\mathcal{Y},\tilde{\mathcal{Y}}) &= (2\rho)^{-\frac{d+m}{2}}|2\pi W|^{\frac{1-2\rho}{2}}|2\pi V|^{\frac{1-2\rho}{2}} \\ &\quad \exp\left(-\frac{\rho}{4}\left(\mathrm{Df}(\boldsymbol{\xi}_l,\tilde{\boldsymbol{\xi}}_l,\boldsymbol{\varphi}_l,\tilde{\boldsymbol{\varphi}}_l)\right)\right), \end{aligned} \tag{7}$$

where the mean parameters $\boldsymbol{\xi}_l$, $\boldsymbol{\varphi}_l$ can be acquired with knowledge of $Q(\mathcal{X}) = \prod_{t=1}^{T}Q(\boldsymbol{x}_t)$ and $Q(\boldsymbol{x}_t) = \prod_{t=1}^{T}\mathcal{N}(\hat{\boldsymbol{x}}_t,\hat{\Sigma}_t)$ as

$$\begin{aligned} \boldsymbol{\xi}_l &= \frac{1}{\nu_l}\sum_{t=2}^{T}p(s_t=l|\mathcal{Y})(\hat{\boldsymbol{x}}_t - A\hat{\boldsymbol{x}}_{t-1} - \boldsymbol{d}_l), \\ \boldsymbol{\varphi}_l &= \frac{1}{\nu_l}\sum_{t=2}^{T}p(s_t=l|\mathcal{Y})(\boldsymbol{y}_t - C\hat{\boldsymbol{x}}_t). \end{aligned} \tag{8}$$

D. Practical Consideration

Generally, online recognizers estimate current status of action from the history of input motion in certain intervals. In this section, some practical considerations for online recognition tasks with the proposed kernel are described. Specifically, we explain how to segment the (endless) time-series motion for our proposed kernel, because the length of time-series motions; T and $\tilde{T}$, are given explicitly in (3). In other words, (3) requires the input motions to be segmented a priori. Simplest way to realize the online recognition is that we set $T = \tilde{T} = \text{constant}$ at each frame. This approach seems to be very simple and not to consider anything about alignments of the two motions. However, the inner state information about the SLDs; $p(\mathcal{S},\mathcal{X}|\mathcal{Y})$, $p(\tilde{\mathcal{S}},\tilde{\mathcal{X}}|\tilde{\mathcal{Y}})$ provides a certain level of the alignment of input motion. In this paper, we modify subtly the formulation of our proposed kernel in order to realize not only preserving context of global motion pattern but also improving the response for the drastic change of input motion. Specifically, we calculate $p(\mathcal{X},\mathcal{S}|\mathcal{Y})$ not from $\mathcal{Y}$ but from $\mathcal{Y}_L$ whose length is much longer than $\mathcal{Y}$. This means the estimating the inner state of the SLDs performs before segmenting the input motion with T. This modification provides the kernel the context for global motion pattern even if T is very short.

IV. Experimental Results

In this section, we illustrate the performance of marginalized BoV kernel in recognition experiments using real human time-series motion data. The recognition task for this experiment was to classify motion whether walking or not in frame wise with support vector machines [2]. In a real recognition system, a combination scheme such as *one vs. one* and *one vs. all* of SVMs to handle multi-class classification can be used. But we don't ask for such a recognition task. This is because there is a large difference between designing classification tasks and designing kernels itself. In order to concentrate on evaluating the kernel itself, we give only simple tasks.

Motion Data: In the following sentences, we illustrate the training and testing of motion data used in this experiment. The motion data contains human skeletal configuration and its time-series of joints angles acquired by a magnetic motion capture system with 30 Hertz. The skeletal configuration in the experiments has 36 degrees of freedom. Specifically, the format of the motion data is BVH. The number of the BVH files is 60. The total time is 183.6 sec. (avg. 3.1 sec.).

The number the motion capture file used in this experiment is 60. They include 19 files with walking only, 20 files with running only, 5 files with lying, 5 files with standing still, 5 files with sitting, 5 files with transitional motion from standing to sitting, a file with transitional motion where walking motion is observed in part. The tempo of walking in the experiment ranges from slow-moving walking to brisk walking. Fig. 3 shows the thumbnail of motion used in this experiment. Lying, sitting, standing still motion is very stillness. The time length of the transitional motion that contains walking motion is about 15 seconds. The frame-by-frame labels to be given for classifier are tagged as follows. The value +1 is tagged when walking motion occurs and -1 is tagged when walking motion does not occur. When it was ambiguous to classify motion, we tagged it as non-walking.

Evaluation Method: Next, we describe the method for evaluating the performance. The criterion of the performance we used in this experiment is F-measure. F-measure with adjustable positive parameter β is defined as

$$F_\beta = \frac{(\beta+1)RP}{\beta R + P},\ \beta > 0,$$

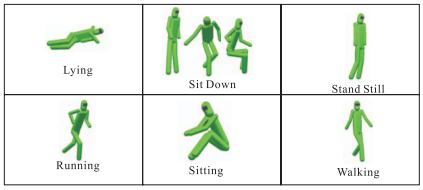

Fig. 3. Action categories used in this experiment

where R denotes recall and P denotes precision performance. Because F-measure can be interpreted as a harmonic mean of the recall and the precision, a higher F-measure indicates the higher performance of the classifier. In order to make a fair evaluation of the performance from a statistical view point, we used a cross validation type method. Specifically, the training and testing data were randomly divided from the motion data noted above. The amount of the training data was set as 30% compared to whole dataset. The training and testing phase was iteratively done in 20 times for every condition. The adjustable parameter in F-measure, β, was set at 1.0.

In order to clarify the quality of the proposed kernel, we compare two kinds of conventional kernel techniques. One kernel, that was proposed by Shimosaka et al. [1], uses spectrum information of gazed input motion in order to capture repetitive motion. Specifically, the similarity can be written as $K(\boldsymbol{y}_{t-W_F+1:t}, \tilde{\boldsymbol{y}}_{\tilde{t}-W_F+1:\tilde{t}}) = K(\boldsymbol{f}_t, \tilde{\boldsymbol{f}}_{\tilde{t}})$ from two motion $\boldsymbol{y}_{t-W_F+1:t}$, $\tilde{\boldsymbol{y}}_{\tilde{t}-W_F+1:\tilde{t}}$ spanning W_F frames, where $\boldsymbol{f}_t$ can be computed by Fourier analyzer from $\boldsymbol{y}_{t-W_F+1:t}$.

The second method to be compared uses history of input motion. The classifier with this kernel can be interpreted as an auto regressive model. If we use a non-linear kernel, the classifier can be interpreted as a nonlinear regression models. Specifically, the input feature for kernel, $\boldsymbol{g}_t = \left[\boldsymbol{y}_t^{\mathsf{T}}, \boldsymbol{y}_{t-1}^{\mathsf{T}}, \ldots, \boldsymbol{y}_{t-W_R+1}^{\mathsf{T}}\right]^{\mathsf{T}}$, can be transformed from W_R frames of motion. Then the similarity value is written as $K(\boldsymbol{g}_t, \tilde{\boldsymbol{g}}_{\tilde{t}})$.

Parameters and Conditions: In the following, we describe the specific parameters used in this experiment. We designed the topology of symbol transition for walking and select the continuous state vectors. Specifically, the symbol state architecture of the SLD is set as a cyclic state transition architecture because walking can be viewed as a repetitive motion. We also manually set the position and the velocity of the both feet as the hidden continuous state vector $\boldsymbol{x}_t \in \mathbb{R}^4$. In response to the setting for $\boldsymbol{x}$, the observed time-series data $\boldsymbol{y}_t \in \mathbb{R}^2$ represents the frontal (back) position of left and right feet relative to the hip. Parameters of the SLD was optimized from the walking motion by the EM algorithm. Parameter ρ in the marginalized BoV kernel is set to 4 and 8. The time window size is set to 16 frames. The SLDs used in this experiment represents walking motion and are optimized from the walking motion.

The parameters of the compared kernel are set as in [1]: i.e. $W_F = 64$. This is the minimal number to capture sufficient resolution of frequency for walking because the frequency of stable human walking ranges from 0.5 to 1.5 Hertz. The observed value of the regression like kernel is the same as the proposed kernel, both feet position. The time span to represent time-series motion, W_R, is 16. The kernel used in the two types of kernels is linear and RBF kernel. RBF kernel can be written as $K_{\mathrm{RBF}}(\boldsymbol{a}, \boldsymbol{b}) = \exp(-\sigma^{-2}||\boldsymbol{a} - \boldsymbol{b}||^2)$, where $\sigma > 0$ is a adjustable parameters. In this paper, we took the parameter σ from the dimensionality of the input features to the kernel d_f from [1]. In each kernel, we select some positive constant values for the regularization parameter of SVM. Specifically, we gave $C = 100, 1000$ for SVMs.

TABLE I
CLASSIFICATION PERFORMANCE IN EACH KERNEL

Type of kernel	Parameter of the kernel	C in SVM	F_1 measure	Rate of SV
M.BoV Kernel	$\rho = 4$	$C = 100$	95.7	11.6
M.BoV Kernel	$\rho = 4$	$C = 1000$	**95.8**	10.0
M.BoV Kernel	$\rho = 8$	$C = 100$	95.6	11.1
M.BoV Kernel	$\rho = 8$	$C = 1000$	95.6	**9.6**
Freq+RBF	$\sigma = 1.5\sqrt{d_f}$	$C = 100$	92.4	29.9
Freq+RBF	$\sigma = 1.5\sqrt{d_f}$	$C = 1000$	95.1	17.4
Freq+Linear	$\emptyset$	$C = 100$	90.2	31.7
Freq+Linear	$\emptyset$	$C = 1000$	90.3	30.5
Regr+RBF	$\sigma = 1.5\sqrt{d_f}$	$C = 100$	84.3	63.1
Regr+RBF	$\sigma = 1.5\sqrt{d_f}$	$C = 1000$	91.2	36.2
Regr+Linear	$\emptyset$	$C = 100$	NaN	NaN
Regr+Linear	$\emptyset$	$C = 1000$	NaN	NaN

Result: The performance for each condition of each kernel is shown in Table I. In Table I, the M.BoV Kernel represents the proposed marginalized BoV kernel, Freq+· denotes a spectrum based method and Regr+· denotes regression like technique. Parameter C means the regularization variables of SVM. The marginalized BoV kernels achieve high F-measure in every condition. Although Freq+RBF achieves high performance to a certain degree, the score is worse than the worst value in the marginalized BoV kernel. In addition, the rate of the ♯ of the support vectors (♯ of support vectors / ♯ of training samples) in the proposed method is much smaller than for the other methods. The rate of number of the support vectors indicates the generalization error of the SVM. For example, the rate in the proposed method is about 10% of support vectors in contrast with 17 % of support vectors in Freq+RBF technique. This empirical result shows that much higher performance of our proposed method for capturing walking motion data than the other methods. Table I also clarifies that classifiers using linear kernel fail to achieve high classification performance. Especially, we cannot calculate F-measure in case of Regr+Linear because the precision rate cannot be computed. This means the classifiers in that case never detect walking motion.

Classification Performance for Walking around Motion: Finally, we demonstrated the performance of the classifiers obtained from the previous experiment in reaction to the novel walking around motion. Specifically, the actor walks in 4 $\sim$ 5 steps and turns. The thumbnail of the input motion is shown in Fig. 4. The mapping between input and binarized output of SVM with marginalized BoV kernel and Freq+RBF kernel is shown in Fig. 5. This result

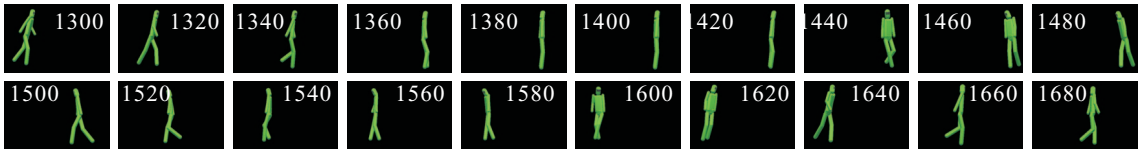

Fig. 4. Thumbnail of walking around motion is shown. The figures, for example, 1415, indicate frame from the start.

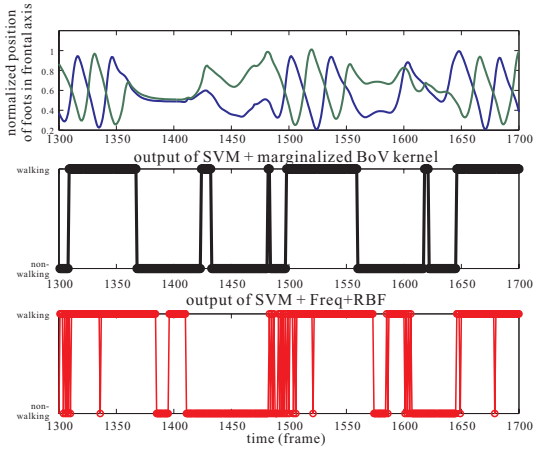

Fig. 5. Walking around motion, the corresponded output of SVM with the proposed kernel and the conventional kernel.

shows that the proposed kernel can almost detect walking motion. There are some "mistaken" result around frame 1420, however, we do not think this is particularly bad. This is because the actor around this time takes one step to turn and the next for walking and the classifier judges them both as walking. We think it is hard even for a human to judge this kind of motion as walking or not. When you think this motion should not be categorized as walking, the question can be resolved by using static information such as forward movement of hip into another kernel. On the other hand, the result of Freq+RBF oscillates very high. This does not fit into human intuition. What is worse, detection of the end point of walking in Freq+RBF method is delayed.

V. Conclusion

In this paper, we propose a new kernel computation for online action recognition. The proposed kernel incorporates switching linear dynamics with the technique of marginalized kernels. Specifically, our kernel is a combination of kernels using unimodal linear dynamics with bags of vectors representation. We call the proposed kernel as marginalized Bags of Vectors (BoV) kernel.

In order to evaluate the performance of our marginalized BoV kernel, we gave it the task of classifying whether walking or not per frame. Using various types of real motion capture data, the experimental works show that the proposed kernel has excellent power to classify the target time-series motion.

Our suggestion for future work is as follows. At first, we have plans to evaluate versatility of our marginalized BoV kernels through applying for several types of dynamical actions, such as raising hand and getting up. Second suggestion is about interdependency of output from the recognizers. Because simple support vector machine (SVM) does not incorporate label interdependencies in conceptual aspect, the output of the SVM is sometimes shaky and oscillatory even if using the our proposed kernel easily enables us to make a great performing classifier.

References

[1] M. Shimosaka, T. Mori, T. Harada, and T.Sato. Recognition of human daily life action and its performance adjustment based on support vector learning. In *CD-ROM of the Third IEEE/Robotics and Automation Society Conference on Humanoids Robots*, 2003.

[2] B. Schölkopf and A. Smola. *Learning with kernels*. MIT Press, 2002.

[3] J. Ohya, J. Yamato, and K.Ishii. Recognizing human action in time-sequential images using hidden Markov model. In *Proceedings of the 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 379–385, 1992.

[4] T. Inamura, Y. Nakamura, H. Ezaki, and I. Toshima. Imitation and primitives symbol acquisition of humanoids by integrated mimesis loop. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, pages 4208–4213, 2001.

[5] V. Pavlović, B. Frey, and T. Huang. Time-series classification using mixed-state dynamic Bayesian networks. In *Proceedings of the 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 609–615, 1999.

[6] T. Jebara, R. Kondor, and A. Howard. Probability product kernels. *Journal of Machine Learning Research*, 5:819–844, 2004.

[7] M. Shimosaka, T. Mori, T. Harada, and T. Sato. Kernel design using mixed-state dynamics for time-series action recognition (in Japanese). In *CD-ROM Proceedings of the Twenty-Second Annual Conerence on Robotics Society of Japan*, 2004.

[8] K. Tsuda, T. Kin, and K. Aasai. Marginalized kernels for biological sequences. *Bioinformatics*, 18(1):S268–S275, 2002.

[9] T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems 11*, pages 487–493. MIT Press, 1999.

[10] M. Seeger. Covariance kernels from Bayesian generative models. In *Advances in Neural Information Processing Systems 14*, pages 905–912. MIT Press, 2002.

[11] C. Watkins. Dynamic alignment kernels. In *Advances in Large Margin Classifiers*, pages 39–50. MIT Press, 2000.

[12] A. Smola and S. Vishwanathan. Hilbert space embeddings in dynamical systems. In *Proceedings of the 13th IFAC symposium on system identification*, 2003.

[13] B. North, A. Blake, M. Isard, and J. Rittscher. Learning and classification of complex dynamics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(9):1016–1034, 2000.

[14] M. Isard and A. Blake. CONDENSATION – Conditional Density Propagation for Visual Tracking. *International Journal of Computer Vision*, 29(1):5–28, Aug 1998.

[15] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B(Methodological)*, 39(1):1–38, 1977.

[16] T. Darrell and A. Pentland. Space-time gestures. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 335–340, 1993.

[17] T. Jebara. Images as bags of pixels. In *Proceedings of the Ninth IEEE International Conference on Computer Vision*, volume 1, pages 265–273, 2003.