Human Motion Recognition with a Convolution Kernel

Dongwei Cao, Osama T Masoud, Daniel Boley
Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455
Email: {dcao, masoud, boley}@cs.umn.edu

***Abstract*— We address the problem of human motion recognition in this paper. The goal of human motion recognition is to recognize the type of motion recorded in a video clip, which consists of a set of temporarily ordered frames. By defining a Mercer kernel between two video clips directly, we propose in this paper a recognition strategy that can incorporate both the information of each individual frame and the temporal ordering between frames. Combining the proposed kernel with the support vector machine, which is one of the most effective classification paradigms, the resulting recognition strategy exhibits excellent performance over real data sets.**

***Indexed Terms*— human motion recognition, convolution kernels, support vector machines.**

I. Introduction

The purpose of human motion recognition is to assign a specific label to a human motion, which is usually recorded on a video clip. Depending on the requirements of the specific application, recognition needs to be performed either offline or online. In offline recognition, a video clip recording a single type of motion is available and one wants to identify the type of motion recorded in the video clip. Online recognition is usually needed in real time surveillance, and one wants to identify the most probable motion type at each instance. In either case, the motion type should be inferred from a sequence of temporarily ordered frames, and a motion recognition strategy should exploit both the content of each frame and the temporal ordering between consecutive frames. Thus, the strategies for offline and online recognition are essentially the same, and we focus on offline recognition in this paper.

A common strategy to build a motion recognition system consists of two steps. The first step is to extract a set of features that characterizes the motion type, and the second step is to construct a classifier that takes these features as input and outputs the motion type. Most work in motion recognition differs from each other on the feature extraction method used, including extracting features from 2-D tracking data [1]–[7] or 3-D tracking information [8], [9], or extracting motion information directly from images [10]–[13]. Given a set of extracted features, most recognition algorithms are based on either template matching [12], [13] or state-space matching which usually uses Hidden Markov Model (HMM) [11]. Neural networks have also been used for this purpose [2]. The performance of these recognition algorithms, especially those based on template matching, is highly dependent on the quality of the extracted motion features. A comprehensive review on human motion analysis can be found, for example, in [14].

Rooted in statistical learning theory, the Support Vector Machines (SVMs) [15], [16] have shown to be one of the most effective paradigm for classification. There are two key ingredients in a support vector machine. The first is the idea of margin maximization, which was proved to provide good generalization performance of the resulting classifier. The second ingredient is the use of kernel, which measures the similarity between objects. Since the idea of margin maximization is hard-wired into the SVM formulation, in order to apply the SVM in practice, the main task is to choose a kernel appropriate for the given problem. More specifically, to use SVM for motion classification, we need to choose a kernel that incorporates the fact that each video clip consists of a sequence of temporarily ordered frames.

The main contribution of this paper is to propose a kernel that is defined directly over video clips, which incorporates not only the information of each individual frame but also the temporal ordering between consecutive frames. Using support vector machines with the proposed kernel, the resulting motion recognition strategy exhibits excellent recognition performance over real data sets.

In the rest of this paper, Section II briefly introduces support vector machines, Sections III through VI describe the proposed kernel between video clips, Section VII presents the experimental results, and Section VIII concludes the paper with future research directions.

II. Support Vector Machines

In the typical setting of a binary classification problem, we are given a training data set $\mathcal{D}$ of size n

$$\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathcal{X},\ y_i \in \{1, -1\}, i = 1, \ldots, n\}, \quad (1)$$

where $\mathbf{x}_i$ represents the i-th object and y_i is the label of $\mathbf{x}_i$.

For a test datum $\mathbf{x}_0 \in \mathcal{X}$, the predicted label $h(\mathbf{x}_0)$ given by the support vector machine (SVM) classifier h is [15]

$$h(\mathbf{x}_0) = \text{sign}\left(f(\mathbf{x}_0)\right) = \begin{cases} 1 : f(\mathbf{x}_0) \geq 0 \\ -1 : f(\mathbf{x}_0) < 0 \end{cases}, \quad (2)$$

where $f(\mathbf{x}_0)$ is called the *functional margin* and is defined as

$$f(\mathbf{x}_0) = \sum_{i=1}^{n} \alpha_i^* y_i K(\mathbf{x}_i, \mathbf{x}_0) + \theta^*. \quad (3)$$

The vector $\boldsymbol{\alpha}^* = [\alpha_1^* \ \dots \ \alpha_n^*]^T$ is the solution of the following quadratic optimization problem

$$\text{Maximize}: W(\alpha) = \boldsymbol{\alpha}^T \mathbf{1} - \frac{1}{2}\boldsymbol{\alpha}^T \mathbf{Y}^T \mathbf{G} \mathbf{Y} \boldsymbol{\alpha} \tag{4a}$$

$$\text{Subject to}: 0 \leq \boldsymbol{\alpha} \leq C \text{ and } \boldsymbol{\alpha}^T \mathbf{y} = 0, \tag{4b}$$

where $\mathbf{1}$ is a vector of ones, C is a regularization coefficient that controls the trade-off between accuracy and smoothness of the classifier and needs to be specified through model selection, $\mathbf{Y}$ is a $n \times n$ diagonal matrix with $Y_{ii} = y_i$ for $i = 1, \dots, n$, and $\mathbf{G}$ is the so-called *Gram matrix* with $G_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$ for $i, j = 1, \dots, n$. Detailed explanations on how to solve problem (4) efficiently and compute θ^* based on $\boldsymbol{\alpha}^*$ can be found, for example, in [17].

In the above formulation, $K : \mathcal{X} \times \mathcal{X} \longmapsto \mathbb{R}$ is called the *Mercer kernel* [15], which is usually abbreviated as *kernel*, and $\mathbb{R}$ is the set of real number. By choosing an appropriate kernel K, we implicitly specify a (usually nonlinear) mapping ϕ from $\mathcal{X}$ to some Hilbert space $\mathcal{H}$ such that the following equation holds for all $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle, \tag{5}$$

where $\langle \cdot, \cdot \rangle$ is the dot product in $\mathcal{H}$. With reference to (3), this means that the SVM classifier is a linear classifier in $\mathcal{H}$, which could result in a (possibly highly) non-linear classifier in $\mathcal{X}$ if the mapping ϕ is non-linear.

Thus, in order to build a SVM classifier, all we need is to choose a value for the penalizing coefficient C and specify a kernel K suitable for the problem interested. There are many kernels that have been developed. For example, when $\mathcal{X}$ is the Euclidean space, popular kernels are:

$$\text{Linear kernel:} \quad K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j, \tag{6a}$$

$$\text{Polynomial kernel:} \quad K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j)^d, \tag{6b}$$

$$\text{Gaussian kernel:} \quad K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\sigma \|\mathbf{x}_i - \mathbf{x}_j\|^2\right), \tag{6c}$$

where $\sigma > 0$ in the Gaussian kernel. The linear kernel has been successfully used in, for example, text classification problem [18]. The polynomial kernel and Gaussian kernel have been proven to be very effective in a handwritten digits recognition problem [19]–[21].

In general, a symmetric function $K : \mathcal{X} \times \mathcal{X} \longmapsto \mathbb{R}$ is a kernel if and only if it satisfies one of the following two conditions (see, e.g. [22], [23]).

- **Condition I:** There exists a mapping $\phi : \mathcal{X} \longmapsto \mathcal{H}$ such that, for all $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$, we have

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle. \tag{7}$$

- **Condition II:** For any $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$, where n is any positive number, the $n \times n$ Gram matrix $\mathbf{G}$ with $G_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$ is positive semi-definite.

Strictly speaking, the necessity of condition I, i.e., the existence of mapping ϕ, requires additional technical assumptions on the space $\mathcal{X}$ and the function K [23]. Since these assumptions are satisfied for most practical problems, we drop them in this paper to make the presentation less complex.

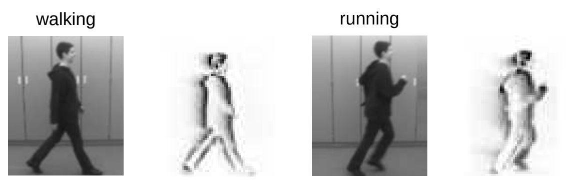


Fig. 1. Raw images and filtered images of "walking" and "running".

In the following Sections III through VI, we will propose a kernel between two video clips for motion recognition problems, and prove its validity using the above two conditions.

III. Motivation for a Kernel Between Two Video Clips

We assume that each video clip consists of several temporally ordered frames, which are called *raw images* here. Figure 1 shows snapshots, i.e., raw images, of two common type of human motions, "walking" and "running."

Also shown in Figure 1 is the *filtered image* for each raw image, which was introduced in [24]. For each raw image in a video clip, its filtered image is defined as the weighted sum of the current raw image and all earlier raw images, and the weight is larger for the more recent raw image. As shown in Figure 1, the filtered image encodes a short period of motion history prior to the current raw image.

Assuming every raw image is represented by its filtered images, reference [25] proposed a motion recognition strategy based on SVMs. In [25], one first trains a SVM classifier to classify filtered images, where the training data are the filtered images of a set of labeled video clips and the label of each filtered image is the same as the label of the video clip to which it belongs, and a kernel defined between filtered images is used. The label of a test video clip is obtained by applying majority voting over the labels of its filtered images, which are predicted by the SVM classifier (c.f. Equation (2)).

Although promising results were obtained, one limitation of the strategy in [25] is that it ignores the temporal ordering of the filtered images in a video clip and treats each video clip as a bag of filtered images. Figures 2 and 3 show a sequence of filtered images corresponding to "walking" and "running", respectively, each of which records approximately half circle of the motion. The most distinctive part between the filtered image of walking and that of running lies in the bottom half of the image, i.e., the layout of two legs. Viewing each filtered image individually, there is a close resemblance between the 15-th filtered image in "walking" and the 5-th filtered image in "running" in terms of the layout of two legs. In other words, there is some overlap between the set of filtered images of "walking" and the set of filtered images of "running". The majority voting scheme in [25] was designed to address this overlapping problem, with the assumption that most of the filtered images of a video clip do not lie in the overlapped region. However, this assumption may not always be valid, for example, when there is a substantial amount of noise in the recorded video clip.

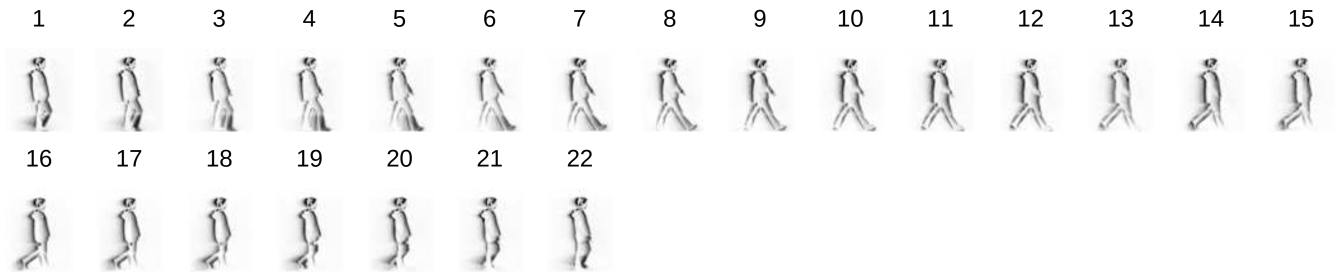


Fig. 2. Filtered images of a video clip of "walking."

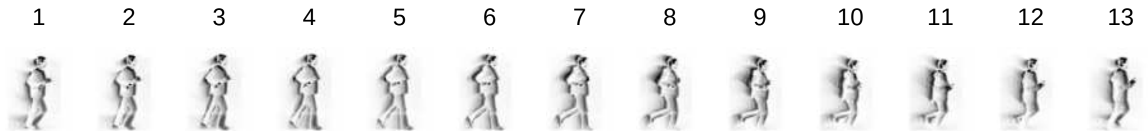


Fig. 3. Filtered images of a video clip of "running."

In this paper, while still using SVM as the underlying classification paradigm, we address the above issues by proposing a kernel defined *directly over video clips*. The main idea is to take the relative position of a filtered image in a video clip into consideration. More specifically, taking Figures 2 and 3 as an example, we will incorporate the fact that the 15-th filtered image in Figure 2 is close to the end of the video clip, while the 5-th filtered image in Figure 3 is close to the start of the video clip.

IV. A Kernel Between Two Video clips

We assume that, for $i = 1, \dots, n$, there are n_i temporarily ordered frames in the i-th video clip $\mathbf{v}_i$ and the p-th frame of $\mathbf{v}_i$ is represented by its filtered image $\mathbf{v}_i^p$.

To quantify the relative position of filtered image $\mathbf{v}_i^p$ in $\mathbf{v}_i$, we define the *ordering index* t_i^p for $\mathbf{v}_i^p$ as

$$t_i^p = \frac{p-1}{n_i - 1}. \tag{8}$$

Here, $t_i^p = 0$ means that $\mathbf{v}_i^p$ corresponds to the first frame in $\mathbf{v}_i$ and $t_i^p = 1$ means that $\mathbf{v}_i^p$ corresponds to the last frame in $\mathbf{v}_i$. For the p-th filtered image $\mathbf{v}_i^p$ in video clip $\mathbf{v}_i$ and the q-th filtered image $\mathbf{v}_j^q$ in video clip $\mathbf{v}_j$, we define the *temporal distance* $d(\mathbf{v}_i^p, \mathbf{v}_j^q)$ between them as

$$d(\mathbf{v}_i^p, \mathbf{v}_j^q) = \left| t_i^p - t_j^q \right|. \tag{9}$$

The value of the temporal distance $d(\mathbf{v}_i^p, \mathbf{v}_j^q)$ indicates how well two frames $\mathbf{v}_i^p$ and $\mathbf{v}_j^q$ are temporarily aligned. More specifically, $d(\mathbf{v}_i^p, \mathbf{v}_j^q) = 0$ means that frames $\mathbf{v}_i^p$ and $\mathbf{v}_j^q$ have the same relative position, for example, being the first frame of $\mathbf{v}_i$ and $\mathbf{v}_j$, respectively. At another extreme, $d(\mathbf{v}_i^p, \mathbf{v}_j^q) = 1$ means, for example, that one frame is the first frame in a video clip and the other frame is the last frame in another video clip.

Based on the ordering indices of all filtered images in the video clip $\mathbf{v}_i$, we represent $\mathbf{v}_i$ as a set $\mathbf{x}_i$ of size n_i, i.e.,

$$\mathbf{x}_i = \{(\mathbf{v}_i^p, t_i^p) \mid p = 1, 2, \dots, n_i\}$$

$$= \left\{ \left(\mathbf{v}_i^1, t_i^1\right), \left(\mathbf{v}_i^2, t_i^2\right), \dots, \left(\mathbf{v}_i^{n_i}, t_i^{n_i}\right) \right\}. \tag{10}$$

Definition 1 (Kernel Between Video Clips): Let $K_{\mathrm{F}} : \mathcal{F} \times \mathcal{F} \longmapsto \mathbb{R}$ be a kernel defined over filtered images, where $\mathcal{F}$ denotes the space in which a filtered image lies and the subscript "F" means "filtered image", and $K_{\mathrm{O}} : \mathbb{R} \times \mathbb{R} \longmapsto \mathbb{R}$ be a kernel defined over real numbers, where the subscript "O" means "ordering index." We define a "video" kernel $K_{\mathrm{V}} : \mathcal{X} \times \mathcal{X} \longmapsto \mathbb{R}$ between two video clips $\mathbf{x}_i$ and $\mathbf{x}_j$ as

$$K_{\mathrm{V}}(\mathbf{x}_i, \mathbf{x}_j) = \sum_{p=1}^{n_i} \sum_{q=1}^{n_j} K_{\mathrm{O}}(t_i^p, t_j^q) K_{\mathrm{F}}(\mathbf{v}_i^p, \mathbf{v}_j^q), \tag{11}$$

where n_i is the number of filtered images in $\mathbf{x}_i$, n_j is the number of filtered images in $\mathbf{x}_j$, t_i^p is the ordering index corresponding to the filtered image $\mathbf{v}_i^p$, and t_j^q is the ordering index corresponding to the filtered image $\mathbf{v}_j^q$. ■

We will discuss the choice of kernels K_{F} and K_{O} in the next section. The following Proposition 2 shows that the symmetric function K_{V} defined in Equation (11) is indeed a kernel.

Proposition 2: Let $K_{\mathrm{F}} : \mathcal{F} \times \mathcal{F} \longmapsto \mathbb{R}$ and $K_{\mathrm{O}} : \mathbb{R} \times \mathbb{R} \longmapsto \mathbb{R}$ be kernels defined over spaces $\mathcal{F}$ and $\mathbb{R}$, respectively. The symmetric function $K_{\mathrm{V}} : \mathcal{X} \times \mathcal{X} \longmapsto \mathbb{R}$ defined in Equation (11) is a kernel.

Proof: The statement is proved by combining the following Lemmas 3 and 4. ■

Lemma 3: Let $K_1 : \mathcal{M} \times \mathcal{M} \longmapsto \mathbb{R}$ and $K_2 : \mathcal{N} \times \mathcal{N} \longmapsto \mathbb{R}$ be two kernels defined over the space $\mathcal{M}$ and $\mathcal{N}$, respectively. The following symmetric function $K : (\mathcal{M}, \mathcal{N}) \times (\mathcal{M}, \mathcal{N}) \longmapsto \mathbb{R}$ is a kernel

$$K((\mathbf{m}_p, \mathbf{n}_p), (\mathbf{m}_q, \mathbf{n}_q)) = K_1(\mathbf{m}_p, \mathbf{m}_q) K_2(\mathbf{n}_p, \mathbf{n}_q), \tag{12}$$

where $\mathbf{m}_p, \mathbf{m}_q \in \mathcal{M}$ and $\mathbf{n}_p, \mathbf{n}_q \in \mathcal{N}$.

Proof: The statement is proved by showing that the symmetric function K defined in Equation (12) satisfies Condition II described in Section II, where the fact that the tensor product of two positive semi-definite matrices is a positive semi-definite matrix is used. We refer readers to Theorem 2.20 in reference [22] for details of the proof. ■

Lemma 4: Let $K_{\mathrm{U}} : \mathcal{U} \times \mathcal{U} \longmapsto \mathbb{R}$ be a kernel defined over the space $\mathcal{U}$, and $\mathbf{u}_1$ and $\mathbf{u}_2$ be two arbitrary finite subsets of $\mathcal{U}$. Then, the following symmetric function K defined over subsets of $\mathcal{U}$ is a kernel

$$K(\mathbf{u}_1, \mathbf{u}_2) = \sum_{u_1' \in \mathbf{u}_1} \sum_{u_2'' \in \mathbf{u}_2} K_{\mathrm{U}}(u_1', u_2''). \tag{13}$$

Proof: A rigorous proof of this Lemma can be found in, for example, Section 7 of reference [23]. We provide here an intuitive proof by giving a mapping ϕ such that the Condition I described in Section II is satisfied, i.e., the symmetric function K defined in Equation (13) corresponds to an inner product.

Since K_{U} is a kernel, using Condition I described in Section II, there exists a mapping ϕ_{U} such that, for all $u_1', u_2'' \in \mathcal{U}$,

$$K_{\mathrm{U}}(u_1', u_2'') = \left\langle \phi_{\mathrm{U}}(u_1'), \phi_{\mathrm{U}}(u_2'') \right\rangle .$$

We define the mapping ϕ as

$$\phi(\mathbf{u}) = \sum_{u' \in \mathbf{u}} \phi_{\mathrm{U}}(u'),$$

where $\mathbf{u}$ is a finite subset of $\mathcal{U}$. Then, the inner product between $\phi(\mathbf{u}_1)$ and $\phi(\mathbf{u}_2)$ is

$$\begin{aligned}
\langle \phi(\mathbf{u}_1), \phi(\mathbf{u}_2) \rangle &= \left\langle \sum_{u_1' \in \mathbf{u}_1} \phi_{\mathrm{U}}(u_1'), \sum_{u_2'' \in \mathbf{u}_2} \phi_{\mathrm{U}}(u_2'') \right\rangle \\
&= \sum_{u_1' \in \mathbf{u}_1} \sum_{u_2'' \in \mathbf{u}_2} \langle \phi_{\mathrm{U}}(u_1'), \phi_{\mathrm{U}}(u_2'') \rangle \\
&= \sum_{u_1' \in \mathbf{u}_1} \sum_{u_2'' \in \mathbf{u}_2} K_{\mathrm{U}}(u_1', u_2'') \\
&= K(\mathbf{u}_1, \mathbf{u}_2).
\end{aligned}$$

Thus, the symmetric function K defined in Equation (13) corresponds to an inner product and, according to Condition I described in Section II, is a kernel. ■

V. Choice of Kernel K_{F} between Filtered Images and Kernel K_{O} between Ordering Indices

We discuss in this section how to choose the kernel K_{V}, which measures the similarity between filtered images, and the kernel K_{O}, which is defined over ordering indices.

Without losing generality, we assume that each filtered image has width a and height b and corresponds to a matrix of size $a \times b$. By concatenating the columns of the matrix, each filtered image can be represented by a vector of length $a \times b$, i.e., an element in the Euclidean space $\mathbb{R}^{a \times b}$. Thus, all kernels defined over the Euclidean space, such as those in Equations (6), can be used as kernel K_{F}. For example, assuming the Gaussian kernel shown in Equation (6c) is used, the term $K_{\mathrm{F}}(\mathbf{v}_i^p, \mathbf{v}_j^q)$ in Definition 1 can be written as

$$K_{\mathrm{F}}(\mathbf{v}_i^p, \mathbf{v}_j^q) = \exp\left(-\sigma \|\mathbf{v}_i^p - \mathbf{v}_j^q\|^2\right) \tag{14}$$

where $\sigma > 0$ needs to be fixed through model selection.

In Definition 1, the term $K_{\mathrm{O}}(t_i^p, t_j^q)$ quantifies the temporal similarity between two filtered images, and its value should be large for two filtered images that are well temporarily aligned, i.e., having similar ordering indices. One choice is to make $K_{\mathrm{O}}(t_i^p, t_j^q)$ vary inversely to the temporal distance $d(\mathbf{v}_i^p, \mathbf{v}_j^p)$ defined in Equation (9). There are many kernels satisfying this requirement and one of them is the familiar Gaussian kernel, based on which the term $K_{\mathrm{O}}(t_i^p, t_j^q)$ in Definition 1 (c.f. Equation (11)) can be written as

$$K_{\mathrm{O}}(t_i^p, t_j^q) = \exp\left(-\gamma \left|t_i^p - t_j^q\right|^2\right), \tag{15}$$

where $\gamma > 0$ needs to be specified through model selection.

Putting things together, we arrive at the following kernel to measure the similarity between video clips

$$\begin{aligned}
K_{\mathrm{V}}(\mathbf{x}_i, \mathbf{x}_j) &= \sum_{p=1}^{n_i} \sum_{q=1}^{n_j} K_{\mathrm{O}}(t_i^p, t_j^p) K_{\mathrm{F}}(\mathbf{v}_i^p, \mathbf{v}_j^q) \\
&= \sum_{p=1}^{n_i} \sum_{q=1}^{n_j} \exp\left(-\sigma \|\mathbf{v}_i^p - \mathbf{v}_j^q\|^2 - \gamma \left|t_i^p - t_j^q\right|^2\right). \quad (16)
\end{aligned}$$

VI. Related Works

As we mentioned in Section I, there have been many works on human motion recognition. We review here some related works on kernel design.

There is a general type of kernel defined over discrete structures called *convolution kernel* [23], [26], and the proposed kernel K_{V} can be seen as a kind of convolution kernel.

In one of the simplest formulations of the convolution kernel, it is assumed that each object consists of several components, and there is a (usually simple) kernel associated with each component, which is called base kernel. The convolution kernel between two objects is the sum of the base kernels, each of which is evaluated over a pair of components coming from two objects. As shown in Equation (10), we represent each video clip as a set whose members are the pairs of filtered image and ordering index. In other words, we assume that the video clip $\mathbf{v}_i$ is represented by a set $\mathbf{x}_i$ consisting of n_i components, and the p-th component is the pair $(\mathbf{v}_i^p, t_i^p)$, where $\mathbf{v}_i^p$ is the filtered image and t_i^p is the ordering index. Viewing the product $K_{\mathrm{F}}(\cdot, \cdot) K_{\mathrm{O}}(\cdot, \cdot)$ as a base kernel, the kernel K_{V} defined in Definition 1 can also be seen as the sum of the base kernels evaluated between pairs of components.

The proposed video kernel K_{V} is also related to the string kernel, which is a special type of convolution kernel and has been applied to, for example, text classification [27] and protein sequence classification [28] problems.

For strings defined over a finite alphabet and of finite length, the string kernel $K_{\mathrm{s}}(\mathbf{s}_1, \mathbf{s}_2)$ between two strings $\mathbf{s}_1$ and $\mathbf{s}_2$ is defined as the *weighted sum* of the similarities between sub-strings of $\mathbf{s}_1$ and $\mathbf{s}_2$ [27]. This definition is equivalent to the familiar recursive definition based on which one can evaluate $K_{\mathrm{s}}(\mathbf{s}_1, \mathbf{s}_2)$ efficiently using dynamic programming. For sub-string $\mathbf{s}_1'$ of $\mathbf{s}_1$ and sub-string $\mathbf{s}_2'$ of $\mathbf{s}_2$, the similarity between $\mathbf{s}_1'$ and $\mathbf{s}_2'$ is 1 if they are identical and 0 otherwise. The weight associated with the similarity between $\mathbf{s}_1'$ and $\mathbf{s}_2'$ is a decreasing function of the sum of the *number of gaps* of $\mathbf{s}_1'$ and $\mathbf{s}_2'$ [29]. Here, the sub-string $\mathbf{s}_1'$ has a non-zero number of gaps only if it is non-contiguous in $\mathbf{s}_1$. Viewing the number of gaps as a measure of the *quality of spatial alignment* between two sub-strings, the string kernel weights the similarity between two sub-strings based on how well they are aligned spatially.

The idea of weighting based on the quality of alignment in the string kernel motivated the proposed kernel K_{V}. More specifically, we can view the kernel $K_{\mathrm{V}}(\mathbf{x}_i, \mathbf{x}_j)$ between two video clips $\mathbf{x}_i$ and $\mathbf{x}_j$ as a *weighted sum* of the similarities between filtered images of $\mathbf{x}_i$ and $\mathbf{x}_j$, where the similarity is measured by the kernel K_{F} and the weight is controlled by the kernel K_{O}. As shown in Equation (15), the kernel K_{O} is a decreasing function of the temporal distance $d(\mathbf{v}_i^p, \mathbf{v}_j^q)$ between $\mathbf{v}_i^p$ and $\mathbf{v}_j^q$. Viewing the temporal distance $d(\mathbf{v}_i^p, \mathbf{v}_j^q)$ as a measure of *quality of temporal alignment* between two filtered images, the proposed video kernel K_{V} weights the similarity between two filtered images based on how well they are aligned temporarily.

VII. Experimental Results

In this section, we will demonstrate the effectiveness of the proposed kernel K_{V} on motion recognition using experiments on real data sets. Let us denote the motion recognition strategy based on SVM with kernel K_{V} as "SVM-VideoKernel". We will compare SVM-VideoKernel against the strategy studied in reference [25], which is denoted as "SVM-FrameKernel". As we mentioned in Section III, the strategy SVM-FrameKernel trains a SVM classifier using a kernel between filtered images, and classifies a test video clip by applying majority voting over the predicted labels of the filtered images.

Our goal here is to differentiate two types of motion, i.e., "walking" and "running." There are 58 video clips recording the walking and running of 29 persons, where every person performed each type of motion exactly once. The number of frames, which is also the number of filtered images, of a video clip varies from 21 to 53.

With reference to Equation (16) and after some experiments, we use the following kernel K_{V} to measure the similarity between video clips $\mathbf{x}_i$ and $\mathbf{x}_j$ and the following regularization coefficient C in the experiment

$$K_{\mathrm{V}}(\mathbf{x}_i, \mathbf{x}_j) = \sum_{p=1}^{n_i}\sum_{q=1}^{n_j} \exp\left(-0.02 \times \|\mathbf{v}_i^p - \mathbf{v}_j^q\|^2 - \left|t_i^p - t_j^q\right|^2\right). \quad (17a)$$

$$C = 10. \quad (17b)$$

After extensive experiments, the strategy "SVM-FrameKernel" gives the lowest error rate when the following kernel and regularization coefficient are used to train the SVM

$$K(\mathbf{v}_i^p, \mathbf{v}_j^q) = \exp\left(-0.1 \times \|\mathbf{v}_i^p - \mathbf{v}_j^q\|^2\right) \text{ and } C = 5. \quad (18)$$

The two recognition strategies are evaluated using a modified cross validation called "Leave One Person Out Cross Validation" (LOOCV-Person). In each fold of the LOOCV-Person, two video clips performed by one person are used as test data and the SVM classifier is trained using the other 56 video clips performed by the other 28 persons. Table I compares "SVM-VideoKernel" and "SVM-FrameKernel" in terms of average error rate on test data over 29-fold LOOCV-Person. Here, the error rate is defined as the fraction of misclassified video clips.

TABLE I
Experimental Results of 29-fold LOOCV-Person

Recognition Strategy	Error Rate (%)
SVM-FrameKernel [25]	1.72
SVM-VideoKernel	0

The advantage of strategy SVM-VideoKernel over strategy SVM-FrameKernel is evident, since the former achieves a zero error rate. As mentioned in Section III, in order for strategy SVM-FrameKernel [25] to correctly classify a video clip, the majority of the filtered images of this video clip must be correctly classified. However, when the filtered images are very noisy, a video clip can still be misclassified. Figure 4 shows the raw frames and filtered images of the video clip that is misclassified by the SVM-FrameKernel strategy, and correctly classified by the SVM-VideoKernel strategy where the proposed kernel K_{V} is used. We can see that, due to the appearance of the subject, especially the gray level of pant, the filtered images are quite noisy compared to the filtered images shown in Figures 2 and 3. The zero error rate of SVM-VideoKernel shown in Table I thus further substantiates the robustness of the proposed recognition strategy, where a kernel between two video clips is used.

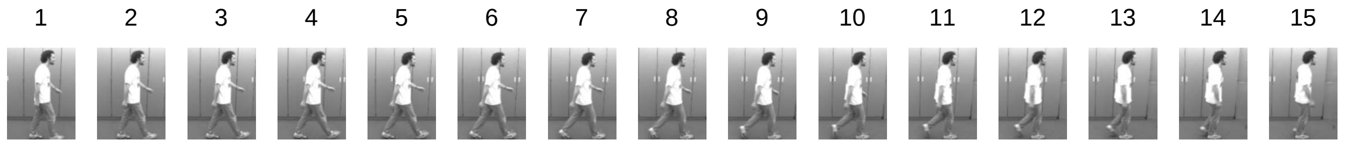
VIII. Conclusions and Future Work

We proposed in this paper a novel human motion recognition strategy based on support vector machines by proposing a kernel that is defined directly over video clips. We proved the validity of the proposed kernel and demonstrated the effectiveness of the proposed strategy through experiments on real data sets.

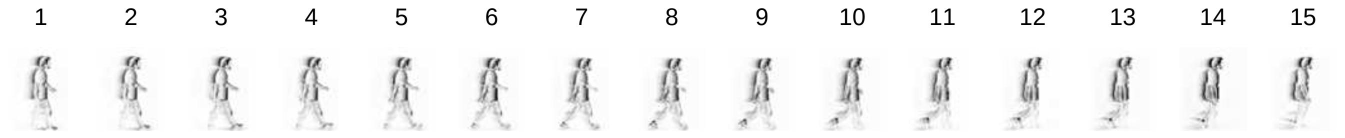
There are several directions along which the current work can follow. In this paper, we assume that a video clip is temporarily properly aligned, i.e., the first frame roughly corresponds to the beginning of the motion and the last frame roughly corresponds to the end of the motion. The first direction is thus to remove this assumption and one possible solution is to use the following kernel K'_{V}

$$K'_{\mathrm{V}}(\mathbf{x}_i, \mathbf{x}_j) = \min_{\mathbf{x}'_i \in W(\mathbf{x}_i), \mathbf{x}'_j \in W(\mathbf{x}_j)} K_{\mathrm{V}}(\mathbf{x}'_i, \mathbf{x}'_j), \quad (19)$$

where $W(\mathbf{x}_i)$ and $W(\mathbf{x}_j)$ are all possible temporal wrappings of $\mathbf{x}_i$ and $\mathbf{x}_j$, respectively, and K_{V} is, for example, the kernel defined in Definition 1. Second, we can use the kernel between video clips to exploit other tasks in video data mining, such as video segmentation through support vector clustering [30], suspicious activity detection through one class SVM [31], [32]. Finally, as shown in Section VI, the proposed kernel K_{V} can be seen as a specialization of a convolution kernel. It is thus natural to exploit other specializations by using



(a) Raw images



(b) Filtered images

Fig. 4. A sub-sequence of the raw images and filtered images of a video clip of "walking" that is misclassified by strategy SVM-FrameKernel [25], but correctly classified by strategy SVM-VideoKernel where the proposed kernel K_{V} is used.

different decompositions of a video clip [23]. One example is the following kernel K''_{V}

$$K''_{\mathrm{V}}(\mathbf{x}_i, \mathbf{x}_j) = \prod_{p=1}^{n_i} \prod_{q=1}^{n_j} K_{\mathrm{O}}(t_i^p, t_j^p) K_{\mathrm{F}}(\mathbf{v}_i^p, \mathbf{v}_j^q), \qquad (20)$$

where K_{O} and K_{F} are, for example, the kernels described in Section V.

ACKNOWLEDGMENT

This work is supported by NSF IIS grants 0208621, 0219863, 0534286.

REFERENCES

[1] N. H. Goddard, "The perception of articulated motion: Recognizing moving light displays," Ph.D. dissertation, University of Rochester, 1992.
[2] Y. Guo, G. Xue, and S. Tsuji, "Understanding human motion patterns," in *ICPR*, 1994, pp. 325–329.
[3] W. H. Dittrich, "Action categories and the perception of biological motion," *Perception*, vol. 22, pp. 15–22, 1993.
[4] Y. Yacoob and M. J. Black, "Parameterized modeling and recognition of activities," *Computer Vision and Image Understanding*, vol. 73, no. 2, pp. 232–247, 1999.
[5] C. Bregler, "Learning and recognizing human dynamics in video sequences," in *IEEE CVPR*, 1997, pp. 568–574.
[6] K. Rangarajan, W. Allen, and M. Shah, "Matching motion trajectories using scale space," *Pattern Recognition*, vol. 26, no. 4, pp. 595–610, 1993.
[7] V. Pavlovic and J. Rehg, "Impact of dynamic model learning on classification of human motion," in *IEEE CVPR*, 2000, pp. 788–795.
[8] L. W. Campbell and A. F. Bobick, "Recognition of human body motion using phase space constraints," in *ICCV*, 1995, pp. 624–630.
[9] D. M. Gavrila and L. S. Davis, "3-D model-based tracking of humans in action: A multiview approach," in *IEEE CVPR*, 1996, pp. 73–80.
[10] N. Krahnstöver, M. Yeasin, and R. Sharma, "Towards a unified framework for tracking and analysis of human motion," in *Proceedings of the IEEE Workshop on Detection and Recognition of Events in Video*, 2001, pp. 47–54.
[11] J. Yamato, J. Ohya, and K. Ishii, "Recognizing human action in time-sequential images using a Hidden Markov Model," in *IEEE CVPR*, 1992, pp. 379–385.
[12] J. W. Davis and A. F. Bobick, "The representation and recognition of human movement using temporal templates," in *IEEE CVPR*, 1997, pp. 928–934.
[13] R. Polana and R. C. Nelson, "Low level recognition of human motion," in *Proceedings of the IEEE Workshop on Non-rigid Motion*, 1994, pp. 77–82.
[14] J. K. Aggarwal and Q. Cai, "Human motion analysis: A review," *Computer Vision and Image Understanding*, vol. 73, no. 3, pp. 428–440, March 1999.
[15] V. N. Vapnik, *Statistical Learning Theory*. NY: Wiley, 1998.
[16] C. Cortes and V. N. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.
[17] J. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in Kernel Methods: Support Vector Learning*, B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds. MIT Press, 1999, pp. 185–208.
[18] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *ECML*, 1998, pp. 137–142.
[19] Y. LeCun, L. D. Jackel, L. Bottou, A. Brunot, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, U. A. Muller, E. Sackinger, P. Simard, and V. Vapnik, "Comparison of learning algorithms for handwritten digit recognition," in *ICANN*, vol. 2, 1995, pp. 53–60.
[20] B. Schölkopf, K. Sung, C. J. C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik, "Comparing support vector machines with Gaussian kernels to radial basis function classifiers," MIT, AI Memo 1599, 1996.
[21] B. Schölkopf, C. J. C. Burges, and V. Vapnik, "Incorporating invariances in support vector learning machines," in *ICANN*, 1996, pp. 47–52.
[22] R. Herbrich, *Learning Kernel Classifiers: Theory and Algorithms*, ser. Adaptive Computation and Machine Learning. Cambridge, Massachusetts; London, England: MIT Press, 2002.
[23] D. Haussler, "Convolution kernels on discrete structures," Department Computer Science, University of California in Santa Cruz, Tech. Rep. UCSC-CRL-99-10, 1999.
[24] O. Masoud and N. Papanikolopoulos, "A method for human action recognition," *Image and Vision Computing*, vol. 21, pp. 729–743, 2003.
[25] D. Cao, O. T. Masoud, D. L. Boley, and N. Papanikolopoulos, "Online motion classification using support vector machines," in *IEEE ICRA*, 2004, pp. 2291–2296.
[26] C. Watkins, "Dynamic alignment kernels," in *Advances in Large Margin Classifiers*, A. J. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, Eds. MIT Press, 2000, pp. 39–50.
[27] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, "Text classification using string kernels," *Journal of Machine Learning Research*, vol. 2, pp. 419–444, Feb. 2002.
[28] C. Leslie and R. Kuang, "Fast string kernels using inexact matching for protein sequences," *Journal of Machine Learning Research*, vol. 5, pp. 1435–1455, 2004.
[29] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.
[30] A. Ben-Hur, D. Horn, H. T. Siegelmann, and V. N. Vapnik, "Support vector clustering," *Journal of Machine Learning Research*, vol. 2, pp. 125–137, 2001.
[31] B. Schölkopf, J. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.
[32] D. Tax and R. Duin, "Support vector domain description," *Pattern Recognition Letters*, vol. 20, pp. 1191–1199, 1999.