Unsupervised View and Rate Invariant clustering of Video Sequences

Pavan Turaga, Ashok Veeraraghavan and Rama Chellappa

Department of Electrical and Computer Engineering and
Center for Automation Research, UMIACS
University of Maryland, College Park, MD 20742
{pturaga,vashok,rama}@umiacs.umd.edu

Abstract

Videos play an ever increasing role in our everyday lives with applications ranging from news, entertainment, scientific research, security and surveillance. Coupled with the fact that cameras and storage media are becoming less expensive, it has resulted in people producing more video content than ever before. This necessitates the development of efficient indexing and retrieval algorithms for video data. Most state-of-the-art techniques index videos according to the global content in the scene such as color, texture, brightness etc. In this paper, we discuss the problem of activity-based indexing of videos. To address the problem, firstly we describe activities as a cascade of dynamical systems which significantly enhances the expressive power of the model while retaining many of the computational advantages of using dynamical models. Secondly, we also derive methods to incorporate view and rate-invariance into these models so that similar actions are clustered together irrespective of the viewpoint or the rate of execution of the activity. We also derive algorithms to learn the model parameters from a video stream and demonstrate how a single video sequence may be clustered into different clusters where each cluster represents an activity. Experimental results for five different databases show that the clusters found by the algorithm correspond to semantically meaningful activities.

Key words: Video Clustering, Summarization, Surveillance, Cascade of Linear Dynamical Systems, View Invariance, Affine Invariance, Rate Invariance

[1] Some parts of this work were presented at CVPR 2007 [1]. This research was funded (in part) by the U.S. Government VACE program.

1 Introduction

Recent years have seen a tremendous explosion of multimedia content fueled by inexpensive video cameras and the growth of the internet. The amount of video data being recorded and accessed by the general populace has been increasing with the rising popularity of several video sharing websites. Several technical challenges need to solved to enable efficient search and retrieval in such large and often unstructured datasets. In the context of videos, most video-sharing websites feature user-supplied tags. It is assumed that the tags describe what is going on in a video. This has several drawbacks, since tags are subjective and the same meaning may be conveyed by a multitude of tags. Further, it is not uncommon to encounter tags whose meaning is not known to a user and are sometimes irrelevant to the video. The idea of user generated tags would also not scale with the increasing size of the dataset. Instead of providing such textual descriptions, we propose to analyze the patterns of motion in a video and automatically extract 'clusters' which when visually presented to a user can convey maximum information about the contents of the video. For example, given a tennis video, short segments depicting elements such as fore-hand, back-hand, smash etc when visually presented to a user would convey more information than a set of textual descriptions.

Unsupervised activity based indexing goes far beyond the traditional problems of activity analysis and recognition, where one knows what one is looking for. Unsupervised indexing requires that activity patterns be discovered without deciding a priori what to look for. As a motivating example, consider the problem of understanding a foreign language. If one hears only a continuous stream of words, how does one know where a word begins and where it ends. If one knew the words, the boundaries between them can be easily perceived. And if one knew the boundaries, then the words can be learnt as well. Similarly, given a continuous video stream, if we knew what activities occur in it, we can discover the boundaries between them – and if we were given the boundaries, the individual activities could be learnt as well. In the context of activities each action primitive is composed of a coherent set of features, and an activity is defined by the way the primitives are put together. Activity-based indexing can benefit by gaining insight into how humans perceive and recognize activities. First, we discuss a general framework of activity perception. Then, we discuss how the cascade of linear dynamical systems model (CLDS) can be derived from the proposed framework.

Applications for automatic discovery of activity patterns are numerous. For example, security and surveillance videos typically have very repetitive activities. If the typical activities can be clustered, then several problems such as unusual activity detection, efficient indexing and retrieval can be addressed. Forensic analysis of surveillance videos is another fast growing and important application area. Current approaches to video forensics involve linear searches over the entire video feed by a human analyst and hence are not scalable when there are a large number of cam-

eras deployed at various locations. Instead of expecting an analyst to sift through the voluminous data, we ask - can 'clusters' of activities be presented that embody the essential characteristics of the videos ? The need for such activity based indexing stands to increase in the near future as more security installations are deployed in a wider variety of locations.

Traditionally, the problem of recognizing human activities from video has received more attention than automatic discovery. One of the earliest experiments demonstrating the richness of spatio-temporal features was done by Johansson [2] who showed that it was possible to recognize humans based on their motion. Since then there has been an explosion of research in computer vision into automatic analysis and recognition of human activities from video. Parallel to the development of accurate and efficient recognition techniques, there has also been a lot of interest in automatic discovery of patterns from raw data. Pattern recognition vs pattern discovery is a fundamental choice that is faced in almost all areas of machine learning. Specific to the activity analysis area, existing literature focuses on the recognition problem to a large extent. In a largely unrelated setting, there has been significant research into indexing of multimedia data such as news clips, sports videos etc according to their content such as in [3]. The pattern discovery approach has also been pursued for this problem domain such as in [4]. In practical applications it is often a combination of supervised and unsupervised approaches that yields the best results [4]. In such cases, an 'unsupervised' approach would involve a very small amount of supervision. The approach we present is largely unsupervised where we assume very little about the data and the supervision is maintained at very low-levels - for example in the choice of features, temporal segmentation criterion etc.

Organization of the paper: The rest of the paper is organized as follows. First, we present a general framework of activity perception in section 2 and draw connections to computational and mathematical models in section 3. Then, we propose the CLDS model for the specific purpose of activity based mining in section 4. The problem of learning the model parameters is addressed in section 5. Further, in section 6 we show how invariances to view, affine transforms and execution rate can be built into the mining algorithm. Finally, in section 7 we describe several experiments to demonstrate the effectiveness of our algorithms.

2 Perception of Activities

In this section, we propose a general framework for activity perception and recognition, from which specific algorithms can be derived. The perception of activities can be seen as proceeding from a sequence of 2-D images to a semantic description of the activity. Activity perception can be naturally decomposed into the following three stages:

(1) Dynamic Sketches
(2) Action sketch
(3) Semantic sketch

(1) **Dynamic Sketches:** The purpose of early stages of vision is to construct primitive descriptions of the action contents in the frame. These primitive descriptions must be rich enough to allow for inference and recognition of activities [5]. Most of the sensory information that is available in videos is actually uninteresting for the purpose of activity-based video indexing and only serves to confound the latter stages of the algorithms. One very important characteristic of this stage is to weed out all the unnecessary sensory information and retain just those elements that are relevant for activity-based video indexing. Visual encoding mechanisms present in the human brain mimic this phenomenon and this is called predictive coding. Barlow [6] and Srinivasan et.al. [7] contend that predictive coding is not just a mechanism for compression but actually goes much further and enables animals to process information in a timely manner. We refer the interested reader to early works of Barlow, Srinivasan and Marr ([6], [7], [5]) on the importance of this stage of visual processing in order to enable vision systems to react and process information in a timely manner.
(2) **Action Sketch:** Studies into human behavior show that human actions can be temporally segmented into elementary units, where each unit consists of functionally related movement [8]. For example, a car parking activity may be considered to be formed of the following primitives - 'Car enters a parking lot', 'Car stops in the parking slot', 'Person walks away from the car'. Such a description requires the ability to segment an activity into its constituents and then develop a description for each of the constituent actions. Each constituent action is like a word describing a short, consistent motion fragment. Hence, this stage can be interpreted as providing a 'vocabulary' with which to create sentences (activities). *In the remainder of the paper, by 'action' we refer to a short segment of consistent motion, whereas, by 'activity' we refer to a composition of such actions that leads to an activity.*

    Representing activities using such linguistic models has been in existence in various other fields and disciplines. Several dance notation schemes are used in practice to interpret complex dance moves. Though not extremely detailed, they are easy to interpret and reproduce in actual steps. It has also been found that the most commonly observed human activities in surveillance settings such as reaching, striking etc are characterized by distinctive velocity profiles of the limbs that can be conveniently modeled as a specific sequence of individual segments – constant acceleration followed by constant velocity followed by constant deceleration [9]. This lends credence to the fact that human actions can be modeled as a sequence of primitive actions, where each action is governed by a simple model.
(3) **Semantic descriptions:** Semantic descriptions perform the same function as grammatical rules for a language. They detail how several constituent action

primitives may be combined together in order to construct or recover complex activities. The most common rules for creating complex activities from constituent actions are sequencing, co-occurrence and synchronization. For example, a single-threaded activity can be said to consist of a linear sequence of a few primitives. An example of a single-threaded activity is 'Person approaches a door' → 'Person swipes the access card' → 'Person enters a building'. Similarly, a complex multi-threaded activity can be seen as a collection of several single-threaded activities with some constraints such as concurrence and synchronization among them. Thus, this stage can be seen as providing the rules for combining the primitives - similar to a set of grammatical rules needed to construct meaningful sentences from individual words.

In the next section, we draw connections with computational approaches and show how several well-known mathematical tools can be used at each of these stages.

3 Computational Models

There exists a huge wealth of literature on building computational models for each of the stages outlined above. In this section, we review some of the important and well-known techniques that can be used at each of the stages.

3.1 Dynamic Sketches

The search for suitable low-level features that can compactly represent the specific information that we seek from images has been at the heart of computer vision research for many years [10]. Low-level features that can compactly represent the information we seek from very short segments of videos (typically 1 or 2 frames) form the dynamic sketch or the frame sketch. The appropriateness of a specific feature is dependent on the specific application and the nature of the video sequences being analyzed. In this paper, we are interested in clustering video sequences according to the type of activity present in the video sequences. Therefore, these low-level features must be able to compactly capture the instantaneous motion of the various scene and actor elements in a manner that enables the next levels (action sketch and the semantic sketch) to efficiently represent the activity occurring in these videos. We summarize in Table 1 some widely used low-level features and their respective characteristics.

Feature	Type of Video	Type of Activity	Illumination Invariance	View Robustness	Examples
Background Subtracted Silhouette	Near Field, Medium Field	Single agent or small number of agents	Moderate	Not Robust	Gait Recognition ([11,12])
Shape	Near Field	Single agent	Moderate	Can be incorporated by affine invariance on shapes	Gait Recognition ([11,13]), Far Field Activity Recognition ([14])
Optical Flow or Texture Flow	Near Field, Medium Field, Restricted Far-Field	Single agent (Near Field), Small number of agents (Medium Field) and Large number of agents (Crowds in far field)	Moderate	Affine invariance can be incorporated	Traffic Monitoring ([15,16], Crowd Monitoring ([17])
Point Trajectories	Far Field or Constrained Medium Field	Single agent (Constrained) or small number of agents (far-field)	Strongly illumination insensitive	Easy to incorporate	View Invariant action recognition([18], Traffic monitoring ([19]), Far-field surveillance [20])
Circular Fourier Features	Medium and Near field	Single Agent	Moderate	View Invariant	Action recognition [21]

Table 1
Various Features for the low-level representation (Dynamic Sketch) and their properties and applicability in various scenarios

3.2 Action-sketches

A significant body of work in activity recognition builds upon extracting action-primitives and modeling the interactions between them. Computationally, automatic primitive extraction may be achieved by mapping the low-level sketches to specific model spaces. There are several choices for the model space such as is reviewed below. Most of the popular approaches can be divided into two broad classes - Spatio-temporal models and Dynamical models.

Spatiotemporal models: These approaches typically encode configurations of spatio-temporal patterns as a model for a video segment, for example, as representative human poses or bags of spatio-temporal features etc. [22] represent human actions using a series of codewords called ‘movelets’ where each movelet encodes a particular configuration of the human body - head, torso, upper and lower limbs. A similar approach was used in [23] to learn human actions performed in the profile view from a long sequence. Temporal templates called motion-history and motion energy which encode both the shape and temporal motion characteristics of the action were proposed as features in [24]. Describing an activity by a collection of space-time interest points which represent points of high gradient in the three-dimensional space-time was proposed by [25]. In a similar approach, [26] represent video segments as histograms of spatio-temporal gradients at multiple

temporal scales. Each segment of video was modeled as a document with words drawn from a corpus of quantized spatial motion histograms in [27].

Dynamical Models: Dynamical approaches explicitly encode the temporal evolution of features for each action. A method to segment human actions into elementary building blocks called movemes - each moveme assumed to belong to a known alphabet of dynamical systems was presented in [28]. Modeling of complex activities using a switching linear dynamic system, where each system corresponding to an action-primitive was proposed in [29] and [30]. Similarly, human gait patterns have been modeled as linear dynamical systems in [11,31] and by HMM's in [32].

We summarize in Table 2 some of the well-known tools and their respective characteristics.

3.3 Semantic Sketches

In the activity recognition context, semantic sketches for activities essentially model the spatio-temporal constraints between the primitives. The major approaches to model such constraints fall into two classes - statistical and rule-based.

Statistical Approaches: HMM's provide an elegant mathematical tool to model the temporal relationships between action primitives [22], [33]. Dynamic belief networks allow complex conditional dependencies between several primitives to be expressed using directed acyclic graphs and have been used for traffic scene analysis in [19]. Complex activities can be modeled as being generated by a switching linear dynamic system as in [29], [30], [34] where each system corresponds to a particular primitive. Textural video sequences have been modeled as a finite collection of visual processes, each of which is a dynamic texture in [35].

Rule based approaches: Syntactic approaches such as stochastic context free grammars allow expressing the relationships as a set of production rules and have been used for action recognition in [36,37]. Temporal logic networks which encode logical relationships between primitives were used for recognizing events involving multiple objects in [38]. A bag of primitives approach is used in [39] to represent activities. Petri-nets provide rich descriptive capabilities to express complex interactions such as synchronization, co-occurrence and concurrence, and have been used in [40].

3.4 Discovery of Action Patterns

The approaches discussed so far are mainly concerned with modeling and recognition of activities and not discovery of action classes. The problem of discovering

Property	**CLDS**	SLDS [29] [30]	Grammars [36]	DBNs [19]	Sliding window approaches [27,26]
View Invariance	Yes	No	Yes	Yes	No
Rate Invariance	Yes	No	Maybe	Maybe	No
Activity based Clustering	Yes	No	No	No	Yes
Action Recognition	Yes	Yes	Yes	Yes	Yes
Frame Sketch	Any appropriate low level feature	Any appropriate feature	Any Appropriate	Any Appropriate	Any appropriate
Action Sketch	Linear Dynamical System (ARMA)	Linear Dynamic System	Vocabulary of Primitives	Vocabulary of primitives	Action prototypes
Semantic Sketch	Cascade Structure	Switching	Grammatical Rules	Directed Acyclic Graph	Bag of features
Sports Video	Yes	Yes	Yes	Yes	Yes
Surveillance Video	Yes	Yes	Yes	Yes	Yes

Table 2
Various approaches for activity based mining from video and their characteristics

action patterns has received less attention compared to the problem of recognition of actions and activities. However, the modeling methods are often similar in both settings. Some of the relevant methods and approaches for discovering human action patterns from video are as follows.

Stauffer and Grimson [41] presented a system to learn patterns of motion in a far-field surveillance setting. They rely on tracking information such as position, size and velocity to describe individual entities. A sequence of observations is considered to be a set of independent features. They use simple co-occurrence statistics of feature prototypes to characterize the motion patterns of vehicles and humans in a far field setting. This works well in such a setting since the motion patterns are simple trajectories which are highly constrained in nature.

Zelnik-Manor and Irani [26] presented a method to cluster videos into event consistent sub-sequences when the events of interest are not known. They model videos as temporal objects and define a distance metric between videos based on histograms of local space-time gradients. Thus, this approach can be viewed as approximating a stochastic process using a first order model i.e. the histogram of features. From this point of view, our approach which is based on dynamic models, is a second order model of the underlying stochastic process where temporal correlations in the stochastic process are also explicitly encoded. Further, the CLDS model is more descriptive and can be viewed as a generative model.

Another related approach is that of Zhong et al [27] who represent each segment of video as a document with words drawn from a corpus of quantized spatial motion histograms. They model a long video as a collection of documents and perform co-clustering of documents and prototypes. Their approach is essentially a

bag-of-words approach which ignores the dynamics of complex actions. A similar approach was presented by Niebles et al [25], where the authors model a video as a collection of words. The words are assumed to be derived from a latent topic, where the latent topic corresponds to a motion-class. They present a graphical model to describe the conditional dependence between a video, the latent topic and the observable features and present a method to learn the latent topics/motion-class using unlabeled data. However, temporally extended complex actions are characterized by complex dynamics. This dynamics information is not explicitly modeled in such a bag-of-words approach. They also note that due to the local nature of the space-time interest point detectors, sufficiently strong responses are not induced for purely translational motion and for regions that do not have sufficient temporal changes.

Unsupervised learning of human action classes from still images has also been proposed by several researchers (c. f. [42]). These approaches rely on information extracted from static images such as the pose of the human to infer the action being performed. Since there is no temporal information, these approaches are best suited when sufficiently informative key-poses are available for each action. We refer the interested reader to [42] and references therein, as they are beyond the scope of the the current discussion.

In our approach, we explicitly model the underlying dynamics of actions using linear dynamic models. A further higher-level of sophistication is achieved by the cascade structure of dynamic systems. This makes the model suitable for representing complex actions which are temporally extended. Further, none of the above approaches deal with view-invariance and execution rate invariance in the clustering/discovery scheme. We deal with the problems of view and rate invariance in a systematic manner which is consistent with the linear dynamical system framework.

4 Cascade of Dynamical Systems

Most activities involving a single human in surveillance settings consist of the human executing a series of action elements in order to achieve a certain goal. For example, a man driving a car into a parking lot, parking the car, alighting from it, walking out of the parking lot (series of action elements) contributes to a typical activity. Moreover, several multi-human activities may also be adequately represented by a sequence of actions. Thus, the CLDS model is an appropriate model for representing a wide variety of common activities.

The model for an activity must be able to represent each of the action elements separately while simultaneously being able to detect the boundaries between them. As we mentioned earlier, we use the consistency of features within each action-element as a cue to discover the boundaries between them. The specific way the

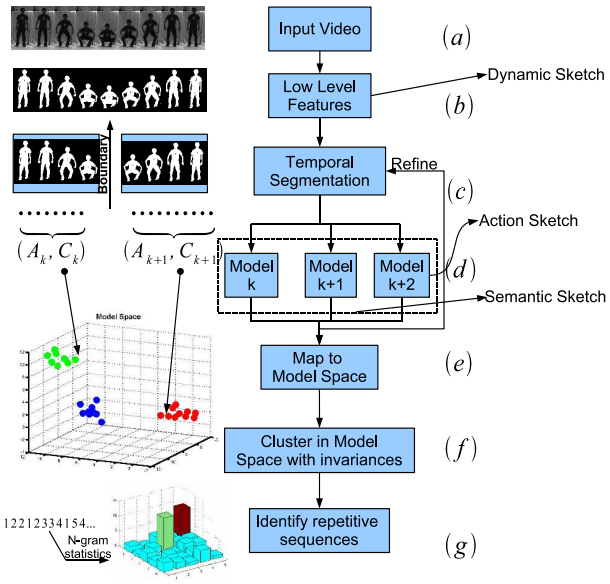


Fig. 1. System Overview: (a) Input video, (b) Feature extraction (Dynamic Sketch), (c) Temporal segmentation, (d) Build and learn dynamical models, (e,f) Cluster in model space taking into account invariances on the data, (g) Identify repetitive activities

action-elements interact with each other is used to discover the activities themselves. The overall system overview is shown in figure 1. Each of the components will be described in detail in the ensuing discussion.

4.1 Modeling Action Elements

Following the discussion presented in section 2 on 'Action-Sketches' and the intuition provided above, we assume that a complex activity can be broken down into its constituent action elements. During each action element, the motion of the actor remains consistent. In fact, it is this consistency of motion that segments an activity into action elements. Therefore, each action element is modeled using a time-invariant dynamical system and the activity is modeled as a cascade of dynamical systems.

Linear Dynamical System for Action Elements: As already discussed, the dynamics of each action element can be modeled using a time-invariant dynamical system. In several scenarios (such as far-field surveillance, objects moving on a plane etc), it is reasonable to model constant motion in the real world using a linear dynamic system (LDS) on the image plane. Given the boundaries between action elements, we model each of these segments using a linear time-invariant (LTI) model. Let us assume that the $P+1$ consecutive frames $s_k, ..., s_{k+P}$ belong to the k^{th} segment and let $f(i)$ denote the observations (flow/silhouette etc) from that frame. Then, the dynamics during this segment can be represented as

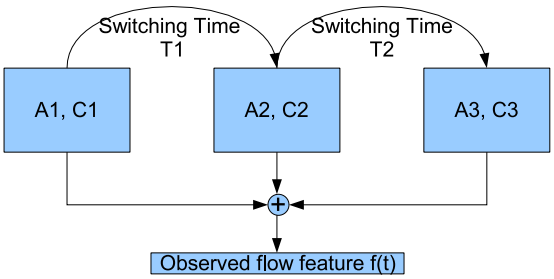


Fig. 2. Illustration of a cascade of three linear dynamical systems. The temporal order of the execution of these dynamical models and their switching times are shown with arrows.

$$f(t) = Cz(t) + w(t) \quad w(t) \sim N(0, R) \tag{1}$$
$$z(t+1) = Az(t) + v(t) \quad v(t) \sim N(0, Q) \tag{2}$$

z is the hidden state vector, A the transition matrix and C the measurement matrix. w and v are noise components modeled as normal with 0 mean and covariance matrices R and Q respectively. When flow is used as the feature, we can write similar equations for the x and y components independently. We assume independence of flow components for simplicity and to reduce the dimensionality of the estimation problem. Similar models have been successfully applied in several tasks such as dynamic texture synthesis and analysis [43], comparing silhouette sequences [11], [31] etc. But we differ from these as we do not assume that we know the temporal span of the segments. We explicitly deal with the temporal segmentation problem in section 5.1. In summary, the parametric model for each segment consists of the measurement matrix C and the transition matrix A.

4.2 Sequence of Dynamical Systems

An activity is composed of a series of action elements. We have modeled each action element using an LTI system. The activity model is now composed of a cascade or a sequence of such dynamical systems. In reality, most activities have a very specific temporal order for the execution of action elements. For example, if our goal is to get to the office, then the sequence of actions executed might be - drive into parking lot, park car, alight from car, walk away from the parking lot. Therefore, we model an activity as a cascade of action elements with each action element modeled as an LTI system. Figure 2 illustrates the complete model for such an activity.

4.3 Relation with Switching Linear Dynamical Systems

Learning the switching instants between LTI models is also encountered in the area of Switching Linear Dynamical Systems (SLDS). The proposed CLDS framework is a conceptually simpler version of the more general SLDS. As will be discussed in this section there are several differences that make learning and inference easier and more efficient than the SLDS framework. But, first we discuss why the CLDS framework is well suited for modeling human actions.

A large class of everyday human actions such as sitting, bending, reaching etc is naturally decomposed into a sequence of predetermined segments of simpler subactions. This observation leads to modeling of complex actions as a sequence of simpler action-elements, with the transitions or switching between action elements specifying the structure of the action. Moreover, more complex activities such as ballet dancing or cooking can usually be decomposed into a simpler sequence of steps. In computational terms this means that the transitions between actions elements can be well approximated as a nearly-deterministic sequence. Thus, in this context CLDS is a special case of the more general SLDS framework, corresponding to cases where the transition matrix between switch states is strongly diagonal in nature. For example, consider the the transition matrices T for two actions – 'Cross Arms' and 'Sit Down' – which were estimated by training a 3-state HMM on the IXMAS dataset [21]. The transition matrices are

$$T_{cross} = \begin{bmatrix} 0.9910 & 0 & 0.0090 \\ 0 & 0.8796 & 0.1231 \\ 0.0469 & 0.0306 & 0.9225 \end{bmatrix} \tag{3}$$

$$T_{sit} = \begin{bmatrix} 0.9271 & 0.0488 & 0.0241 \\ 0.0094 & 0.9529 & 0.0377 \\ 0 & 0.0639 & 0.9361 \end{bmatrix} \tag{4}$$

We see that the transition matrix has a strong diagonal structure. Moreover, the transition probabilities from one state to the remaining two are usually biased more toward one specific state.

The second major difference between the proposed CLDS framework and the SLDS framework is in the notion of 'dwell times'. In the SLDS framework, the switching between models is governed by a Markov chain. This assumption induces an

exponential density on the dwell-times in each state whose mode is at 0. But, the amount of time spent executing an action requires a finite non-zero amount of time. In the CLDS framework, we do not impose this restrictive and counter-intuitive exponential density form. Instead, we will discuss how more general and meaningful parametric models can be learnt in this framework which is similar in principle to the methods presented in [44,34].

The third major difference is in the notion of switch states. In SLDS, usually an extra hidden state is used to model switches. Any change in this hidden state corresponds to a switch between the LTI models such as in [45] and [46]. Usually, the number of states to switch amongst is assumed to be known (equal to the number of distinct actions). These methods are well suited for modeling moderate length sequences since in such cases it is reasonable to assume a small finite set of switch states. Since we are interested in indexing very long videos, it is difficult to estimate the number of switch states a priori. Thus, the CLDS framework does not rely on estimating the number of switch states. Instead, it simply detects switching instants in a recursive manner and fits linear models to each segment. From this point of view, CLDS may be viewed as a recursive approximation to SLDS for long videos.

The fourth difference is in the computational efficiency obtained by restricting the model to a cascade structure as opposed to a general switching structure. Estimating the number of switch states, the model parameters in each state and the switching instants simultaneously is computationally expensive. Recently, an approach was presented in [47] for a special class of systems to estimate the number of states, the switching instants as well as the model parameters for each state. The presented method dealt with single-input single-output auto-regressive (SISO-ARX) processes and its multi-dimensional extension is computationally very expensive. In comparison, since the CLDS framework recursively splits a complex video sequence into smaller segments according to a predefined segmentation criterion, learning is computationally less expensive than the general SLDS model. The structure of the activity is then discovered in the post-clustering stage where we look for repetitive and quasi-repetitive action-labels.

5 Learning Model Parameters

We have modeled an activity as a cascade of dynamical systems. But given a video sequence, we first need to segment the video into action elements and discover the relationship among them. The challenge is to accomplish all of this in a completely unsupervised manner while being invariant to variabilities in an activity like execution rate, resolution of video, rotation and translation etc. We will now describe an algorithm to automatically segment the video and learn the model parameters in an unsupervised manner.

5.1 Discovering Action Boundaries

As mentioned earlier, we use 'consistency' of features within each action-element as a cue to discover boundaries between them. Naturally, the exact measure of 'consistency' is tied to the specific feature at hand. For example, space-time curvature [18] is a widely-used metric to discover boundaries for point trajectories. Measures for shape deformation such as [48] are suited for discovering segment boundaries in shape sequences. In this section, we describe a simple method for discovering action boundaries that works well for background subtracted silhouettes and optical-flow.

For each time-instant t, we predict the current observation $\hat{f}_t$ using a set of K past observations $\{f_{t-1}, \ldots f_{t-K}\}$. If the observation f_t deviates significantly from the predicted value by a threshold i.e. if $\left\| f_t - \hat{f}_t \right\| > thresh$, then a boundary is detected at the time instant t. In our case, the prediction $\hat{f}_t$ is derived from the past observations as follows. For the first few (about 5) set of frames after the beginning of a new segment, we cumulatively learn a single set of affine parameters for the change in the feature. For every incoming new frame, we predict the new feature using the estimated set of affine parameters. Learning the affine parameters for each segment can be achieved in closed-form using the properties of the fourier-transform [49].

This segmentation scheme is suboptimal due to the assumption of affine motion. To overcome this we iterate back and forth between learning the LTI model for each segment and tweaking the segment boundaries till convergence is reached. Taking the output of this initial segmentation as an starting point, we learn the LTI model for each segment. Without loss of generality, let $S_1 = (A_1, C_1)$ and $S_2 = (A_2, C_2)$ be two adjacent segments and their corresponding LTI models. Suppose the temporal span of S_1 is $[t_1, t_b)$ and that of S_2 is $[t_b, t_2]$. Here t_b denotes the boundary between the segments. As will be described in section 5.2, columns of C_k correspond to the top d principal components (PCs) of the observations in segment k. To *evaluate* the boundary according to the learnt models, we compute the reconstruction error of all the observations according to the PCs in the corresponding segments. We move the boundary by an amount τ in forward and backward directions and choose the one that minimizes this error. Thus, we search for the minima of the following cost functional:

$$\Delta(\tau) = \sum_{t=t_1}^{t_b+\tau} \left\| C_1(C_1^T f_t) - f_t \right\|^2 + \sum_{t=t_b+\tau}^{t_2} \left\| C_2(C_2^T f_t) - f_t \right\|^2 \qquad (5)$$

f_t is the observation at time t and $\tau \in [-T, T]$. In our experiments, we typically chose T to be 10. The new boundary is found as $t_b^{new} = t_b^{old} + \arg\min_\tau \Delta(\tau)$.

With the new boundary the models are learnt again, and the process is repeated till convergence, i.e. the boundary does not change anymore $\arg\min_\tau \Delta(\tau) = 0$. For cases when the linear models do not provide a good fit to the observations in each segment due to possible non-linear dynamics, convergence can be an issue. However, in practice we observed that convergence is always achieved. This further confirms the hypothesis that most human activities can be segmented into simpler actions where each action can be modeled by a simple linear model. We show some segmentation results on a near-field video sequence of a human performing 5 different activities. Each activity is repeated several times at random. Note that the segmentation algorithm is independent of the rate of execution of the activity. The video sequence was consistently segmented at the same pose in several instances of the same activity. Some segmentation results obtained on actual video sequences of a person performing 5 different activities are shown in Figure 3 from two different views.

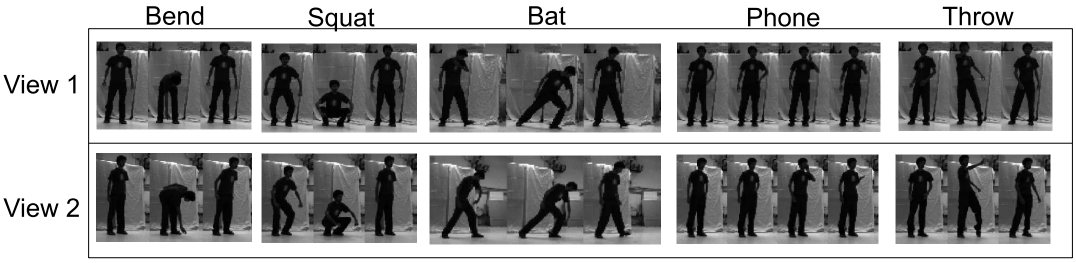


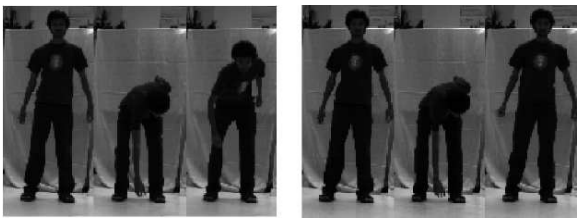
Fig. 3. Sample segment boundaries for 5 activities. Note that the temporal segmentation algorithm finds a boundary whenever there is a change in the direction of motion. Notice that the segmentation results are consistent across view changes.

We see that the videos are segmented at the same pose consistently in both views. This indicates that our algorithm indeed finds semantically meaningful segment boundaries consistently and in a view-invariant manner.

Effect of Boundary Improvement: In most cases, temporal segmentation based on affine parameters gave consistent results for segmenting a sequence into its constituent action elements. Nevertheless, there were some sequences where the segmentation was inadequate and we found that refinement of these boundaries using feedback improved the results. We show one such example in figure 4. We notice that the last segment boundary is incorrect, and it is corrected by refinement using feedback. Note that the boundary improvement algorithm itself is independent of what feature is used.

5.2 Learning the LTI Models for each segment

As described earlier, each segment is modeled as an LTI system. We use tools from system identification to estimate the model parameters for each segment. The most popular model estimation algorithms are N4SID [50], PCA-ID [43] and EM [51].



(a) (b)

Fig. 4. Bending boundaries (a) Before refinement, (b) After refinement

N4SID is a subspace identification algorithm and is an asymptotically optimal solution. However, for large dimensions the computational requirements make this method prohibitive. PCA-ID [43] is a sub-optimal solution to the learning problem. It makes the assumption that filtering in space and time are separable, which makes it possible to estimate the parameters of the model very efficiently via principal component analysis (PCA). The learning problem can also be posed as a maximum likelihood estimation of the model parameters that maximize the likelihood of the observations which can be solved by expectation-maximization (EM) [51]. For computational simplicity we have chosen the PCA based solution of [43] in this paper. Further, as will be discussed in section 6, the PCA based method allows us to derive view-invariant metrics by exploiting the fact that the columns of the observation matrix are principal components of the features of the corresponding segment.

We briefly describe the PCA based method to learn the model parameters here. Let observations $f(1), f(2), \ldots f(\tau)$, represent the features for the frames $1, 2, ...\tau$. The goal is to learn the parameters of the model given in equation (2). The parameters of interest are the transition matrix A and the observation matrix C. Let $[f(1), f(2), \ldots f(\tau)]$ = $U\Sigma V^T$ be the singular value decomposition of the data. Then, the estimates of the model parameters (A, C) are given by $\hat{C} = U, \hat{A} = \Sigma V^T D_1 V (V^T D_2 V)^{-1} \Sigma^{-1}$, where D_1 = [0 0;$I_{\tau-1}$ 0] and D_2 = [$I_{\tau-1}$ 0;0 0]. These estimates of C and A constitute the model parameters for each action segment. For the case of flow, the same estimation procedure is repeated for the x and y-components of the flow separately. Thus, each segment now is represented by the matrix pair (A, C) as shown in figure 1 (d) in order to estimate the corresponding system and transition matrices. The data matrix is a tall thin matrix (size $MN \times \tau$). Computing the singular vectors of the data matrix can be reduced to finding the singular vectors for a $\tau \times \tau$ matrix and taking appropriate linear combinations of those singular vectors. The details of these matrix operations are fairly standard and one may refer to [52] for details of the approach. This makes the algorithm to learn the system and transition matrices, efficient, robust, simple and closed-form.

5.3 Switching between Dynamical Systems:

In order to completely specify the model we also need to specify the switching times between these dynamical systems or equivalently, the amount of time (or frames) spent executing an action element i.e. the *dwell* time. We considered modeling the activity as a Markov model, in which case the probability distribution of the dwell time turns out to be an exponential distribution whose mode is at 0. But, physically the amount of time spent doing one particular action takes a finite amount of time. Thus, to model the dwell time, we need a continuous distribution over time that satisfies the following requirements - a) Support set which is the entire non-negative real line, b) Non-zero mode. The Gamma distribution satisfies both the above requirements. Simpler choices such as Gaussian, exponential, double exponential violate one or the other requirement. Thus, we model the dwell time for each action element as a Gamma distribution with parameters α_k and β_k with $\alpha_k > 1$ (this constraint ensures a non-zero mode). The Poisson distribution also shares the above properties except that it is a discrete distribution.

The parametric Gamma distribution is given by

$$g(x;\alpha,\beta) = x^{\alpha-1}\frac{\beta^{\alpha}e^{-\beta x}}{\Gamma(\alpha)} \quad for\ x > 0 \tag{6}$$

where $\Gamma(\alpha)$ is the gamma function. The mean μ and variance σ^2 of the gamma distribution are given by

$$\mu = \frac{\alpha}{\beta}, \quad \sigma^2 = \frac{\alpha}{\beta^2} \tag{7}$$

Given samples drawn from the above distribution, we can estimate the parameters α and β as follows. Denoting the the sample mean by $\hat{\mu}$ and the sample variance by $\hat{\sigma}^2$, we obtain

$$\hat{\alpha} = \frac{\hat{\mu}^2}{\hat{\sigma}^2}, \quad \hat{\beta} = \frac{\hat{\mu}}{\hat{\sigma}^2} \tag{8}$$

5.4 Generative Power of Learnt Model

A useful test for a representational model is how well it synthesize patterns, and see if the synthesized samples resemble real-world phenomenon. In this section,

we show a few synthesis results obtained using the learnt models. In the first experiment, we used one walk sequence from the USF gait gallery data [53] to learn one walk pattern. We use background subtracted images as the features. We modeled the entire walk sequence using just one LTI model. Then, we used the learnt model to generate the sequence. A few frames from the generated sequence are shown in figure 5.

Fig. 5. A model for the silhouette dynamics for gait was learnt using 1 segment. Shown above is the generated gait sequence from the learnt model.

In the next experiment, we generated a bending sequence. During the learning stage, the sequence was segmented automatically into 3 segments by the proposed segmentation technique. A model was learnt for each segment. To synthesize the activity, we generated sequences from each of the models, and switched from one model to the other according to the discovered cascade. The dwell time in each segment was sampled from the learnt distributions. The generated sequence is shown in figure 6.

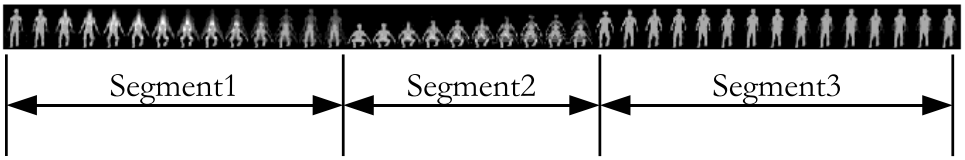


Fig. 6. A model for silhouette dynamics during 'bending' was learnt using 3 segments. Shown above is the generated bending sequence from the learnt cascade of LTI models.

5.5 Clustering Action Element Prototypes

We have now segmented a long video sequence into several distinct segments and learnt the model parameters $(\hat{A}, \hat{C})$ for each of these segments. Even though a long video might consist of several segments, not all of them will be distinct. We need to cluster these segments (figures 1 (e), (f)) to discover the distinct action elements (words). In order to perform this clustering, we need a distance measure on the space of LTI models. Several distance metrics exist to measure the distance between linear dynamic models. The simplest method to measure distance is the L-2 norm between model parameters. Martin [54] proposed a more principled method to measure the distance between ARMA models based on cepstral coefficients. A unifying framework based on subspace angles of observability matrices was presented in [55] to measure the distance between ARMA models. Specific metrics such as the Frobenius norm and the Martin metric [54] can be derived as special cases based on the subspace angles. Recently, [56] presented a framework to extend the Cauchy-Binet kernels to the space of dynamical systems and incorporated the dependence on initial conditions of the dynamical systems as well. However, for

ease of computation we use the subspace angles based kernel in our experiments. Subspace angles $(\theta_i, i = 1, 2, ...n)$ between two ARMA models are defined in [55] as the principal angles $(\theta_i, i = 1, 2, ...n)$ between the column spaces generated by the observability spaces of the two models extended with the observability matrices of the inverse models [55]. The subspace angles $(\theta_1, \theta_2, ...)$ between the range spaces of two matrices A and B is recursively defined as follows [55],

$$cos\theta_1 = \max_{x,y} \frac{\left|x^T A^T B y\right|}{\|Ax\|_2 \|By\|_2} = \frac{\left|x_1^T A^T B y_1\right|}{\|Ax_1\|_2 \|By_1\|_2} \tag{9}$$

$$cos\theta_k = \max_{x,y} \frac{\left|x^T A^T B y\right|}{\|Ax\|_2 \|By\|_2} = \frac{\left|x_k^T A^T B y_k\right|}{\|Ax_k\|_2 \|By_k\|_2} for \quad k = 2, 3, ... \tag{10}$$

subject to the constraints $x_i^T A^T A x_k = 0$ and $y_i^T B^T B y_k = 0$ for $i = 1, 2 \ldots, k-1$. The subspace angles between two ARMA models $[A_1, C_1, K_1]$ and $[A_2, C_2, K_2]$ can be computed by the method described in [55]. Using these subspace angles $\theta_i, i = 1, 2, ...n$, three distances, Martin distance (d_M), gap distance (d_g) and Frobenius distance (d_F) between the ARMA models are defined as follows:

$$d_M^2 = \ln \prod_{i=1}^{n} \frac{1}{\cos^2(\theta_i)}, \quad d_g = \sin \theta_{max}, \quad d_F^2 = 2 \sum_{i=1}^{n} \sin^2 \theta_i \tag{11}$$

We use the Frobenius distance in all the results shown in this paper. Suppose we have N segments in the video sequence, then we create an $N \times N$ matrix W whose $(i, j)^{th}$ element contains the distance between the models of segment i and segment j.

Clustering the Segments In the current setting, we only have a notion of a 'distance' between two points (segments), but we do not have a Euclidean representation of the points. Thus, this precludes the use of clustering techniques that rely on Euclidean representation. The other popular alternative for clustering rely on graph-theoretic methods such as Normalized cuts ([57]) which does not rely on Euclidean representations. The only requirement is that a distance metric be defined between any two points. Hence, graph clustering algorithms are a natural choice in the current setting. But, a practical problem in using these algorithms is choosing the number of clusters. Results in spectral graph theory provide principled means for estimating the number of clusters. A well known result regarding the Laplacian of a graph is briefly summarized as follows.

Result: If G is a graph and L its Laplacian, then the multiplicity of 0 as an eigenvalue of L is equal to the number of connected components of G ([58]).

This result is true for the normalized graph-Laplacian as well. While this result holds for unweighted graphs, in our case the pairwise distance/similarity matrix

represents a weighted graph with the similarities as the edge weights. Connected components in our case represent the clusters that we are looking for. Thus for the weighted case, the smallest eigenvalues will be close to 0 but not exactly 0. We have used this result to estimate the number of clusters given the similarity matrix by analyzing the eigenvalues of the Laplacian and searching for an 'elbow' that represents a sudden change in the eigenvalues. The index at which the elbow is located is the estimated number of clusters. Practically, it is easier to use the normalized Laplacian to search for the elbow, since its non-zero eigenvalues are all 1 by a result similar to the one above.

Once we have estimated the number of clusters, we can compute the clustering using any standard graph clustering algorithm. We have used normalized cuts in our experiments [57]. Let the K cluster labels thus obtained be given by $L_1, L_2, L_3, \ldots L_K$. The segmented video is then given by a sequence of these labels.

5.6 Discovering the Cascade Structure

After clustering the action elements each segment is assigned a label. Suppose we have the following sequence of labels $(L_1, L_3, L_2, L_6, L_7, L_8, L_1, L_3, L_5, L_2, L_6, L_1, L_7, L_8)$. Persistent activities in the video would appear as a repetitive sequence of these labels. From this sequence, we need to find the *approximately* repeating patterns. We say *approximate* because oversegmentation may cause the patterns to be not exactly repetitive. We can say that (L_1, L_3, L_2) and (L_6, L_7, L_8) are the repeating patterns, up to one insertion error. To discover the repeating patterns, we build the n-gram statistics of the segment labels as shown in figure 1 (g). We start by building a bi-gram, tri-gram and four-gram models. In our experience, oversegmentation of the video is more common than undersegmentation. Thus, we allow for up to one insertion error while building the n-gram statistics. We prune the bi-grams which appear as a subsequence of a tri-gram. We prune the tri-grams in a similar fashion. Finally, we declare the n-grams with a count above a threshold (depending on the length of the video) as the repeating patterns in the video. The cascade structure of individual activities is the exact sequence of the prototypes in the n-grams. Once we have the cascade structure, we can go one step further and build a generative model by learning the statistics of the duration of each action prototype. We model the duration of each action prototype as a Gamma distribution with parameters $\alpha_k > 1$ and β_k. The parameters of the distribution can be learnt from training data as described in section 4.2.

6 Building Invariances into CLDS Model

The distance metrics defined in the previous section in equation (11) will break down when there is a change in viewpoint or there is an affine transformation of the low-level features. Some features such as shape are invariant to affine transformations by definition. Features such as point trajectories can be easily made invariant to view and affine transforms. But, in general, it is not guaranteed that a given feature is invariant under these transformations (optical flow, background subtracted masks, motion-history ([24]) and other 'image-like' features). Reliance on the feature to provide invariance to these factors will tie the rest of the processing to that particular feature, which is not desirable as different features are appropriate for different domains and video characteristics. Thus, instead of relying on the feature, we propose a technique to build these invariances into the distance metrics defined above. This allows the algorithm flexibility in the choice of features.

6.1 Affine and View Invariance

In our model, under feature level affine transforms or view-point changes, the only change occurs in the measurement equation and not the state equation. As described in section 5.2 the columns of the measurement matrix (C) are the principal components (PCs) of the observations of that segment. Thus, we need to discover the transformation between the corresponding C matrices under an affine/view change. We start by proving a theorem that relates low level feature transforms to transformation of the principal components.

Theorem 6.1: Let $\{X(\overline{p})\}$ be a zero-mean random field where $\overline{p} \in D_1 \subseteq R^2$. Let $\{\lambda_n^X\}$ and $\{\phi_n^X\}$ be the eigenvalues and corresponding eigenfunctions in the K-L expansion of the covariance function of X. Let $T : D_2 \longrightarrow D_1$, where $D_2 \subseteq R^2$ be a continuous, differentiable one-to-one mapping. Let $\{G(\overline{q})\}$, $\overline{q} \in D_2$ be a random field derived from X as $G(\overline{q}) = X(T(\overline{q}))$. If the Jacobian of T, denoted by $J_T(\overline{r})$, is such that $det(J_T(\overline{r}))$ is independent of $\overline{r}$, then the eigenvalues and eigenfunctions of G are given by $\lambda_n^G = \frac{\lambda_n^X}{|J_T|^{1/2}}$ and $\phi_n^G(\overline{q}) = \frac{\phi_n^X(T(\overline{q}))}{|J_T|^{1/2}}$.

Proof: Let $K_X(\overline{p}, \overline{s})$ be the covariance function of X. Then by the definition of the K-L expansion the following equations hold.

$$\int_{D_1} K_X(\overline{p}, \overline{s})\phi_n^X(\overline{s})d\overline{s} = \lambda_n^X \phi_n^X(\overline{p}), \quad \int_{D_1} \phi_m^X(\overline{s})\phi_n^X(\overline{s}) = \delta(m, n) \tag{12}$$

where both $\overline{p}, \overline{s} \in D_1$ and $\delta(m, n) = \{1$ if m = n, 0 otherwise$\}$. Now, $\{G(\overline{q})\}$ is related to X as $G(\overline{q}) = X(T(\overline{q}))$. For $\overline{q}, \overline{r} \in D_2$, the covariance function of G is

given by $K_G(\overline{q},\overline{r}) = E[G(\overline{q})G(\overline{r})] = E[X(T(\overline{q}))X(T(\overline{r}))] = K_X(T(\overline{q}),T(\overline{r}))$. Now consider the following equation.

$$\int_{D_2} K_G(\overline{q},\overline{r})\phi_n^X(T(\overline{r}))d\overline{r} = \int_{D_2} K_X(T(\overline{q}),T(\overline{r}))\phi_n^X(T(\overline{r}))d\overline{r} \tag{13}$$

$$= \int_{D_1} K_X(\overline{p},\overline{s})\phi_n^X(\overline{s})\frac{1}{|J_T(\overline{r})|}d\overline{s} \tag{14}$$

where (14) is obtained by a change of variables given by $\overline{p} = T(\overline{q}), \overline{s} = T(\overline{r})$, and $|J_T(\overline{r})|$ is the determinant of the Jacobian of T with respect to $\overline{r}$ evaluated at $\overline{r} = T^{-1}(\overline{s})$. Now, if $|J_T(\overline{r})| = |J_T| = constant$, then it comes out of the integral in (14), and using (12) we obtain

$$\int_{D_2} K_G(\overline{q},\overline{r})\phi_n^X(T(\overline{r}))d\overline{r} = \frac{\lambda_n^X}{|J_T|}\phi_n^X(T(\overline{q})) \tag{15}$$

It can further be shown that the set of functions $\{\frac{\phi_n^X(T(\overline{q}))}{|J_T|^{1/2}}\}$ form an orthonormal set. Thus, we have shown that the eigenvalues and eigenfunctions of G are given by $\{\frac{\lambda_n^X}{|J_T|^{1/2}}\}$ and $\{\frac{\phi_n^X(T(\overline{q}))}{|J_T|^{1/2}}\}$ respectively. The utility of this theorem is that if the low-level features like flow/silhouettes undergo a spatial transformation which satisfies the conditions stated in the theorem, then the corresponding PCs also undergo the same transformation.

Note: It is important to note that we are not considering transformations of the pixel intensities, but we are interested in transformations of the 'image-grid'.

6.2 Application to Invariances

Two images that are related by a general spatial transform (affine, homography etc), can be mathematically expressed as $I_2(x,y) = I_1(T(x,y))$.

Affine Transforms: Let $\overline{p} = [x,y]'$ denote a point on the image lattice. Consider the set of 2-D affine-transforms given by $T(\overline{p}) = A\overline{p} + \overline{t}$. Expressing this in inhomogeneous coordinates

$$T(\overline{p}) = \begin{bmatrix} a_{11}x + a_{12}y + t_1 \\ a_{21}x + a_{22}y + t_2 \end{bmatrix} \tag{16}$$

The Jacobian for the transformation is given by $J_T = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ whose determinant is a constant. Thus, by the above theorem, if a set of observations are affine

transformed then their principal components also get transformed by the same affine parameters.

Homography: Consider now a 2-D plane homography given by $H = [h_{ij}]$. In the inhomogeneous coordinates the transformation is given by

$$T(\overline{p}) = \begin{bmatrix} (h_{11}x + h_{12}y + h_{13})/(h_{31}x + h_{32}y + h_{33}) \\ (h_{21}x + h_{22}y + h_{23})/(h_{31}x + h_{32}y + h_{33}) \end{bmatrix} \tag{17}$$

As is apparent, the theorem does not hold for a general homography. We discuss approximations under which the theorem may be applied to homographies.

Let, the transformation between the coordinate frame of the first camera and that of the second camera be given by a rotation and translation. Then, the homography induced by a plane π, between the two views is given by [59]

$$H = M'(R + \frac{Tn^T}{d_\pi})M^{-1} \tag{18}$$

where R and T are the rotation matrix and translation vector respectively, n is the normal to the plane π and d_π is the distance of the plane π from the origin, M and M' are the transformation from the image plane to the camera coordinate system for the two cameras. In the simplest case, we can take $M = M' = \begin{bmatrix} f & 0 & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{bmatrix}$, where f denotes the focal length of the camera, and x_0, y_0 is the origin of the image plane. When the two views are close to each other, we can approximate $T = [\epsilon_x, \epsilon_y, \epsilon_z]'$ and R using small rotations as [60]

$$R \approx \begin{bmatrix} 1 & -n_3\theta & n_2\theta \\ n_3\theta & 1 & -n_1\theta \\ -n_2\theta & n_1\theta & 1 \end{bmatrix} \tag{19}$$

where, θ is the rotation angle, n_1, n_2, n_3 are the directional cosines of the axis of rotation, hence, related by $n_1^2 + n_2^2 + n_3^2 = 1$. On substituting these quantities and

the plane normal $n = [n_x, n_y, n_z]$, in (18) and simplifying, we obtain the following relations between the required elements of $H - h_{31}, h_{32}, h_{33}$,

$$\frac{h_{31}}{h_{33}} = \frac{a/f}{-ax_0/f - by_0/f + c} \tag{20}$$

$$\frac{h_{32}}{h_{33}} = \frac{b/f}{-ax_0/f - by_0/f + c} \tag{21}$$

where $a = -n_2\theta + \frac{\epsilon_z n_x}{d_\pi}, b = n_1\theta + \frac{\epsilon_z n_y}{d_\pi}, c = 1 + \frac{\epsilon_z n_z}{d_\pi}$. In the limit, when $\theta \to 0$ and $\epsilon_x, \epsilon_y, \epsilon_z \to 0$, we obtain $a \to 0, b \to 0, c \to 1$.

$$\lim_{\theta, \epsilon_x, \epsilon_y, \epsilon_z \to 0} \frac{h_{31}}{h_{33}} = 0 \tag{22}$$

$$\lim_{\theta, \epsilon_x, \epsilon_y, \epsilon_z \to 0} \frac{h_{32}}{h_{33}} = 0 \tag{23}$$

Thus, for small view changes $h_{31}, h_{32} << h_{33}$. Under these conditions, the Jacobian of the above transformation can be approximated by $J_T = \frac{1}{h_{33}} \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix}$ whose determinant is also a constant. Thus, the above theorem can be used even in the case where observations are transformed by a homography under the above approximation.

Note: The invariance theorem was proved for continuous random fields. In real images, spatial transforms are not one-to-one maps due to the discrete nature of the underlying lattice. But, our experiments suggest that this theorem can be used to get very good approximations even in the discrete case.

Modified Distance Metric: Proceeding from the above, to match two ARMA models of the same activity related by a spatial transformation, all we need to do is to transform the C matrices (the observation equation). Given two systems $S_1 = (A_1, C_1)$ and $S_2 = (A_2, C_2)$ we modify the distance metric as

$$d_{compensated}(S_1, S_2) = \min_T d(T(S_1), S_2) \tag{24}$$

where $d(.,.)$ is any of the distance metrics in equation (11), T is the transformation. $T(S_1) = (A_1, T(C_1))$. Columns of $T(C_1)$ are the transformed columns of C_1. The optimal transformation parameters are those that achieve the minimization in (24).

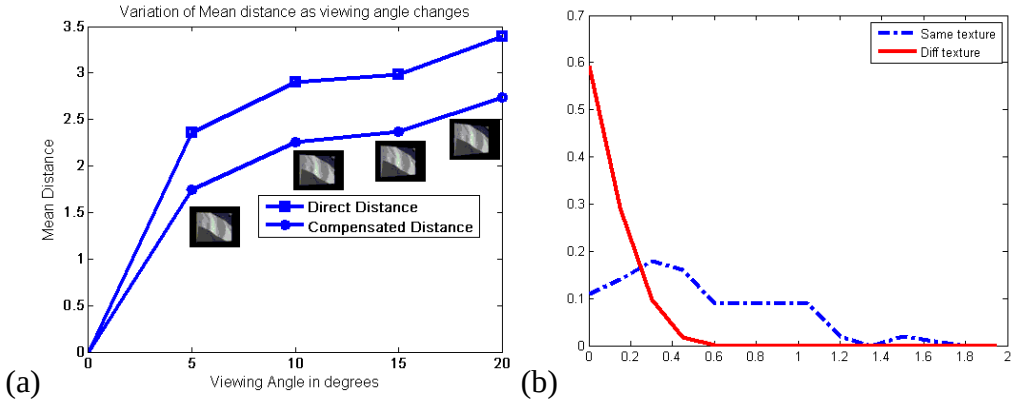


Fig. 7. (a)Variation of Mean Distance as viewing angle changes. Sample views shown, (b)Histogram of difference between Frobenius and $d_{compensated}$ as seen from different views

Depending on the complexity of the transformation model, one can use featureless image registration techniques such as [49], [61] to arrive at a good initial estimate of T. Computing the gradient of the proposed distance metric is extremely difficult due to the recursive way the subspace angles are defined (section 5.5). We could not arrive at closed form expressions for the gradients. Instead, we use Nelder-Mead's (NM) simplex method to perform the optimization. The NM method is a direct search algorithm that is used when gradients cannot be easily computed or accessed. Even though only limited convergence results for the NM method are known, it is known to work well in practice [62].

To illustrate the effectiveness of our proposed technique, we conducted the following experiment. We took a set of 10 dynamic textures from *http://www.cwi.nl/projects/dyntex/index.html* [63]. The textures were modeled to be lying on a plane in front of the camera perpendicular to the optical axis, and a change in viewing angle from 0° to 20° in increments of 5° was simulated by means of a homography (0° corresponds to the frontal view). The images were taken as observations. Figure 7(a) shows how the Frobenius distance breaks-down as the viewing angle is changed. The plot also shows $d_{compensated}$. It can be seen that the proposed technique indeed works better. In figure 7(b), we plot normalized histograms of $(d_F - d_{compensated})$ for same textures as seen from different views and different textures as seen from different views. When comparing different textures, $d_{compensated}$ is not significantly lower than d_F, hence the peak at 0. But, for the same texture as seen from different views, we see that $d_{compensated}$ is significantly lower than d_F.

6.3 Invariance to Execution Rate of Activity

While building models for activities, one also needs to consider the effect of different execution rates of the activity [64]. In the general case, one needs to consider

warping functions of the form $g(t) = f(w(t))$ such as in [65] where Dynamic time warping (DTW) is used to estimate $w(t)$. We consider linear warping functions of the form $w(t) = qt$ for each action segment. Linear functions for each segment give rise to a piece-wise linear warping function for the entire activity, which accounts for variabilities in the execution rate well. It can be shown that, under linear warps the stationary distribution of the Markov process in (2) does not change. Hence, a linear warp will affect only the state equation and not the measurement equation i.e. the A matrices and not the C matrices. Consider the state equation of a segment: $X_1(k) = A_1 X_1(k-1) + v(k)$. Ignoring the noise term for now, we can write $X_1(k) = A_1^k X(0)$. Now, consider another sequence that is related to X_1 by $X_2(k) = X_1(w(k)) = X_1(qk)$. In the discrete case, for non-integer q this is to be interpreted as a fractional sampling rate conversion as encountered in several areas of DSP. Then, $X_2(k) = X_1(qk) = A_1^{qk} X(0)$. i.e. the transition matrix for the second system is related to the first by $A_2 = A_1^q$.

Estimating q: Given two transition matrices of the same activity but with different execution rates, we need a technique to estimate the warp factor q. Consider the eigendecomposition of $A_1 = V_1 D_1 V_1^{-1}$, and $A_2 = V_2 D_2 V_2^{-1}$. Then, for rational q, $A_2 = A_1^q = V_1 D_1^q V_1^{-1}$. Thus, $D_2 = D_1^q$, i.e. if λ is an eigenvalue of A_1, then λ^q is an eigenvalue of A_2 and so forth. Thus, we can get an estimate of q from the eigenvalues of A_1 and A_2 as

$$\hat{q} = \frac{\sum_i log \left| \lambda_2^{(i)} \right|}{\sum_i log \left| \lambda_1^{(i)} \right|} \tag{25}$$

where $\lambda_2^{(i)}$ and $\lambda_1^{(i)}$ are the complex eigenvalues of A_2 and A_1 respectively. Thus, we compensate for different execution rates by computing $\hat{q}$. In the presence of noise, the above estimate of q may not be accurate, and can be taken as an initial guess in an optimization framework similar to the one proposed in section 6.1. Note that compensation for execution rate is done only for segments which have very similar $\hat{C}$ matrices.

7 Discussion and Experiments

In order the validate and show the efficacy of the CLDS model for activity based unsupervised clustering of videos, we perform experiments on 5 databases.

(1) **UMD Dataset:** This dataset contains 10 activities and 10 sequences per activity performed by one actor and captured in 2 views.
(2) **USF Database:** This is a publicly available human gait database of 124 individuals and 10 different sequences per individual.
(3) **UMD Far field data:** This dataset consists of unconstrained far field activities occurring in front of a building entrance such as pedestrians walking, vehicles

parking and exiting etc.

(4) **INRIA database:** This database consists of 10 actors performing 11 activities in a near field setting and contains 3 executions per actor. Actors freely change their orientation.

(5) **Simon Fraser University (SFU) figure skating data [42]:** We have used figure skating videos from [42]. This is completely unconstrained data and involves real world conditions – pan, tilt and zoom of camera and rapid motion of the actor.

Note: Since most of the results are best viewed as videos, we refer the reader to http://www.umiacs.umd.edu/∼pturaga/VideoClustering.html for video results.

7.1 Experiments on UMD Dataset [65]

In the experiment described in section 5.1, five different complex activities – throw, bend, squat, bat and pick phone were discovered automatically. We were also able to learn the cascade of dynamical systems model in a completely unsupervised manner. We manually validated the segment boundaries and the corresponding discovered activities. We call each discovered repetitive pattern a *motif*. To counter over segmentation effects, we merge very similar motifs. Since, a motif is a string of labels, we used the Levenshtein distance [66] as the metric to merge them. The classification of the activities into motifs is tabulated in Table 3. We see that the table has a strong diagonal structure indicating that each of the discovered motifs corresponds to one of the activities in the dataset. Motifs 1-5 correspond to 'bending', 'squatting', 'throwing', 'pick up phone' and 'batting' respectively. This demonstrates that the algorithm does indeed discover semantically meaningful boundaries and also is able to distinguish between various activities by learning the right cascade structure of the action prototypes.

Figure 8 shows activity labels for the entire video sequence extracted manually and automatically. Matching of the colors in the figure indicates that the algorithm is able to discover and identify activities in an unsupervised manner. We found that the errors in labeling are typically near the transition between two activities, where the actual labeling of those frames is itself subject to confusion. To visualize the clusters and to see the *trajectories* of each activity, we embedded each segment into a six-dimensional Laplacian eigenspace. Dimensions 1-3 are shown in figure 9(a) and dimensions 4-6 in figure 9(b). We see that the trajectories of the same activity are closely clustered together in the Laplacian-space.

7.1.1 View Invariance-Simulated Data

We show a few more recognition experiments based on our modified distance metric given in equation (24). In the next experiment, the setup is the same as described

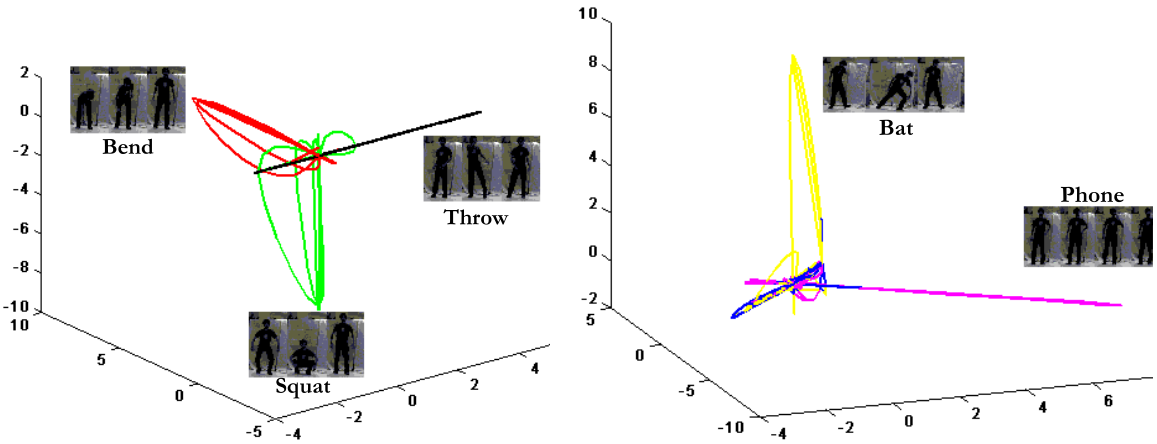
Activity Type	Motif 1	Motif 2	Motif 3	Motif 4	Motif 5
Bending	10	1	0	2	1
Squatting	2	8	2	0	0
Throwing	0	0	7	0	1
Pick Phone	3	0	0	9	0
Batting	0	0	0	1	9

Table 3
Composition of the Discovered Clusters in the UMD database

(a) Manual Labeling

(b) Automatically Discovered Labels (unsupervised−clustering)

Fig. 8. Color coded activity labeling for a 4000 frame video sequence of the UMD database (a) Manual Labeling (b) Unsupervised Clustering result. Image best viewed in color.



(a) (b)

Fig. 9. (a)Visualization of the Clusters in Laplacian Space dimensions 1-3. (b) Visualization of Clusters in Laplacian Space dimensions 4-6. Best viewed in color.

above. But, this time we have 10 activities – *Bend*, *Jog*, *Push*, *Squat*, *Wave*, *Kick*, *Batting*, *Throw*, *Turn Sideways*, *Pick Phone*. Each activity is executed at varying rates. For each activity, a model is learnt and stored as an exemplar. The features (flow-fields) are then translated and scaled to simulate a camera shift and zoom. Models were built on the new features, and tested using the stored exemplars. For the recognition experiment, we learnt only a single LTI model for the entire duration of the activity instead of a sequence. We also implemented a heuristic procedure in which affine transforms are compensated for by locating the center of mass of the

	Baseline		CMH		Compensated distance	
	Exemplars		Exemplars		Exemplars	
Activity	1	10	1	10	1	10
Pick Up Object	40	0	40	40	40	50
Jog in Place	0	0	0	10	70	80
Push	0	0	20	40	10	20
Squat	40	30	10	20	30	60
Wave	30	30	40	20	40	40
Kick	10	0	40	50	30	50
Bend to the side	0	10	0	30	30	70
Throw	0	10	30	40	0	40
Turn Around	0	40	20	20	30	70
Talk on Cellphone	0	0	10	20	40	40
Average	12	12	21	29	32	52

Table 4
Recognition experiment simulated view change data on the UMD database. Table shows a comparison of recognition performance using (a) Baseline technique - direct application of system distance, (b) Center of Mass heuristic, (c) Proposed Compensated distance metric.

features and building models around its neighborhood. We call it Center of Mass Heuristic – CMH. Recognition percentages are shown in table 4. The baseline column corresponds to direct application of the Frobenius distance. We see that our method performs better in almost all cases.

7.1.2 Far-Field Surveillance Data

We also conducted a recognition experiment on a 10 minute video sequence obtained from a far-field surveillance camera. There were 4 different walking patterns (in the location and direction of walk). A model for each of these activities was built and a recognition experiment was run over the entire video sequence and results were manually verified. There were 3 segments that were misclassified from a total of 24 meaningful segments. All of these 3 errors resulted because of a confusion between activities that are co-located but vary only in the local direction of motion. Note that the feature used in this experiment was smoothed flow and therefore did not involve tracking of individual targets.

7.2 Model Order Selection on USF Gait Database [53]

A practical issue in learning the LTI model parameters is to choose an appropriate value for the hidden state dimension d. The answer to this is tied to the domain, and there is no general selection rule. The number d represents the number of basis vectors to project the data on to (the number of principal components). Usually,

the higher the dimension d, the more accurate the representation will be. But, the higher the d, the more the data required for robust estimation of the parameters and the higher the computational cost. Higher-order models also tend to over fit the training data with poor generalization to test instances. One needs to make a trade-off between these issues. To see the effect of varying d, we conducted recognition experiments on the USF dataset using $d = 5, 10, 15$ on Probes A-G. Results are shown in figure 10. We see that the recognition accuracies show an increasing trend as d increases from 5 to 10, but the increase from $d = 10$ to $d = 15$ is only marginal and in some cases even negative. This can be attributed to over fitting of the training data which does not generalize well to test instances. In general, criteria such as Akaike Information Criteria (AIC) [67], Bayesian Information Criteria (BIC) [68], etc may also be used to estimate the optimal number of free parameters (in our case d). In our experiments, we empirically found that using $d = 10$ gives good results across various domains and activity classes.

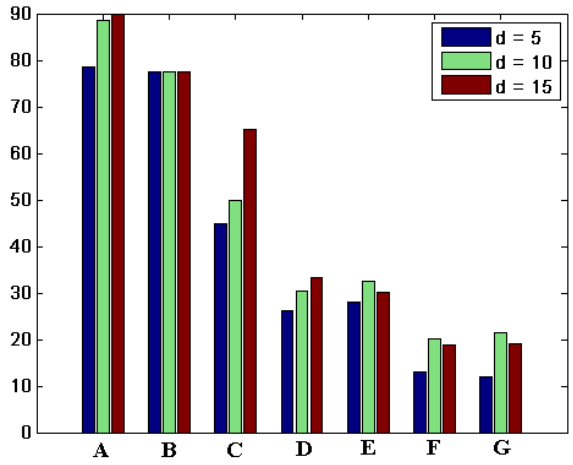


Fig. 10. Model order selection experiment on the USF gait database. Bar plot shows recognition performance as a function of the hidden state dimension (d) on the 7 different challenge experiments (probes A-G) in the USF gait database.

7.3 INRIA - Free-Viewpoint Database [21]

The INRIA multiple-camera multiple video database of the PERCEPTION group consists of 11 daily-live motions performed each 3 times by 10 actors. The actors freely change position and orientation. Every execution of the activity is done at a different rate. For this dataset, we extract $16 \times 16 \times 16$ circular FFT features as described in [21]. Instead of modeling each segment of activity as a single motion history volume as in [21], we build a time series of motion history volumes using small sliding windows. This allows us to build a dynamic model for each segment. We use the segmentation method proposed in [69]. Using these features, we first performed a recognition experiment on the provided data. For the recognition experiment, we used only one segment for each activity which best represented that

	Activity	PCA[21]	Mahalanobis [21]	LDA[21]	System Distance
1	Check Watch	53.33	73.33	76.67	93.33
2	Cross Arms	23.33	86.67	100	100
3	Scratch Head	46.67	86.67	80	76.67
4	Sit Down	66.67	93.33	96.67	93.33
5	Get Up	83.33	93.33	93.33	86.67
6	Turn Around	80	96.67	96.67	100
7	Walk	90	100	100	100
8	Wave Hand	50	70	73.33	93.33
9	Punch	70	86.67	83.33	93.33
10	Kick	50	86.67	90	100
11	Pick Up	60	90	86.67	96.67
	Average	61.21	87.57	88.78	93.93

Table 5
Comparison of view invariant recognition of activities in the INRIA dataset using our approach (system distance) with the approaches proposed in [21].

Motifs	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Sit Down	28	3	0	0	0	1	0	0	0	0	0
Get Up	0	31	0	0	0	0	0	0	0	0	0
Turn Around	0	0	28	0	0	0	1	0	0	0	0
Check Watch	0	0	0	17	5	2	0	6	4	0	0
Cross Arms	0	0	0	0	16	3	0	10	1	0	1
Scratch Head	1	0	0	3	9	3	0	7	4	0	1
Walk	0	0	0	0	0	0	30	0	0	0	0
Wave Hand	0	0	0	6	0	4	0	10	1	0	0
Punch	0	0	0	0	0	4	0	7	9	5	0
Kick	0	0	0	1	0	1	0	0	2	26	0
Pick Up	2	2	0	1	0	1	0	0	4	0	23

Table 6
Confusion matrix showing view-invariant clustering using the proposed algorithm on the INRIA dataset.

activity as in [69]. The recognition results are summarized in table 5. The numbers in the columns corresponding to PCA, LDA and Mahalanobis were obtained using the algorithm reported in [21]. Note that the feature used here [21] itself provides the required invariance to view.

Next, we performed a clustering experiment on all 30 sequences (10 actors $\times 3$ sequences per actor). Segmentation was performed using the method described in [69]. The clustering results are shown in table 6. The strong diagonal structure of the table indicates that meaningful clusters are found. We also see that some activities such as 'Check Watch' and 'Cross Arms' are confused. Similarly, 'Scratch Head' is most often confused with 'Wave Hand' and 'Cross Arms'. Such a confusion may be attributed to the similar and also sparse motion patterns that are generated by those activities.

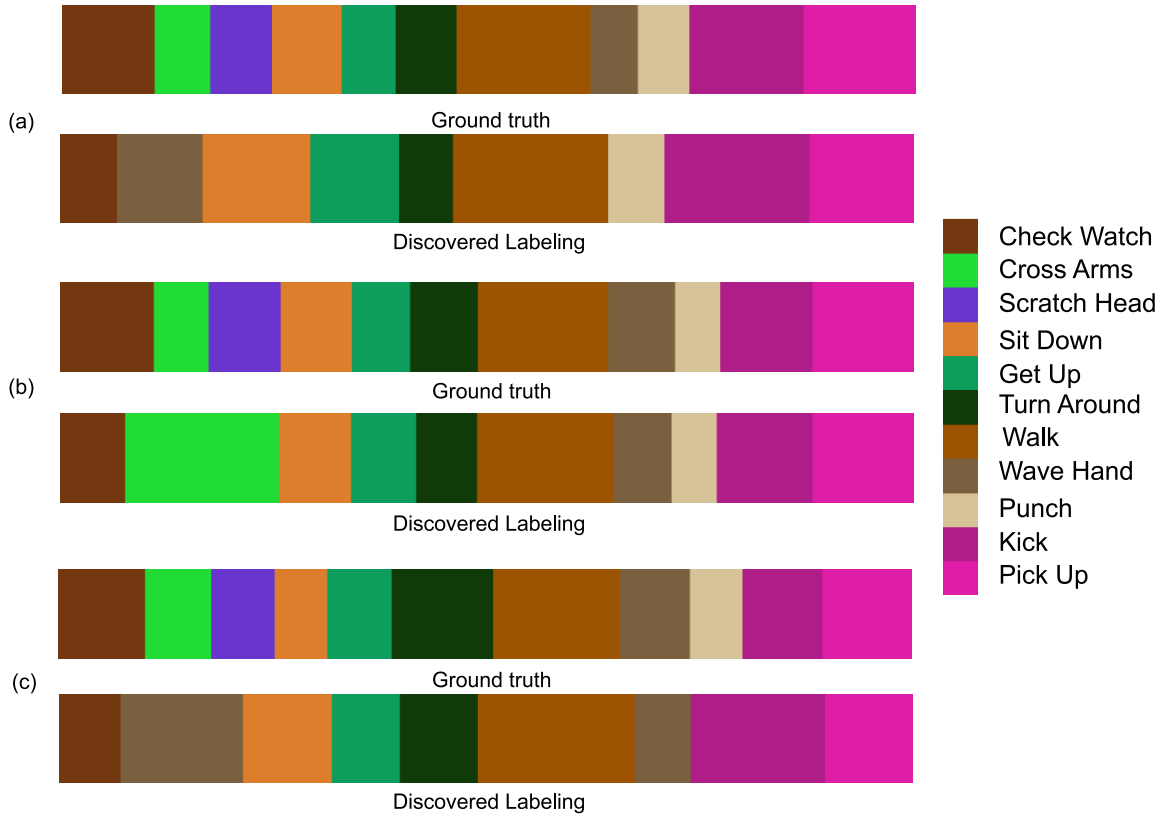


Fig. 11. Color coded activity labeling for three sequences by actor 'Florian'. First row in each is the ground truth, second row is the discovered labeling. Image best viewed in color.

We also show the actual summarization results obtained on two of the actors – 'Florian' and 'Alba' in figures 11 and 12.

7.4 Figure Skating data [42]

We performed a clustering and retrieval experiment on the figure skating dataset reported in [42]. This data is very challenging since it is unconstrained and involves rapid motion of both the skater and real-world motion of the camera including pan, tilt and zoom. Some representative frames from the raw video are shown in figure 13. It should be noted that the authors of [42] consider discovering action classes from static images. Since, they do not use temporal information, the results of our method based on dynamic models cannot be directly compared to [42].

Low-level processing: We built color models of the foreground and background using normalized color histograms. The color histograms are used to segment the background and foreground pixels. Median filtering followed by connected component analysis is performed to reject small isolated blobs. From the segmented results, we fit a bounding box to the foreground pixels by estimating the 2D mean and second order moments along x and y directions. We perform temporal smooth-

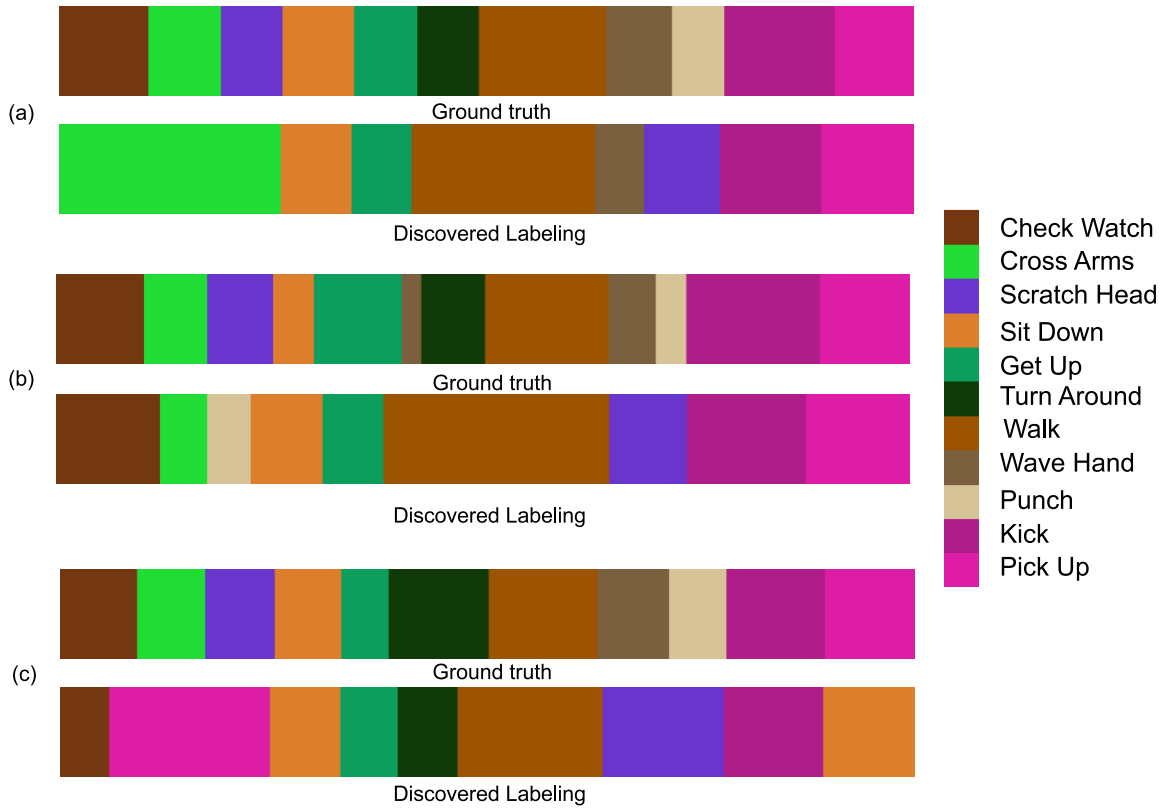


Fig. 12. Color coded activity labeling for three sequences by actor 'Alba'. First row in each is the ground truth, second row is the discovered labeling. Image best viewed in color.

Fig. 13. Sample images from the skating video from [42].

ing of the bounding box parameters to remove jitter effects. The final feature is a rescaled binary image of the pixels inside the bounding box.

Clustering Experiment: Most figure skating videos consist of a few established elements or moves such as jumps, spins, lifts and turns. A typical performance by skater or pair of skaters includes several of these elements each performed several times. Due to the complex body postures involved it is a challenge even for humans to identify clear boundaries between atomic actions. It was difficult even for us to semantically define temporal boundaries of an activity, let alone define a metric for temporal segmentation. Thus, this makes it very difficult to break the video into temporally consistent segments. Instead of performing explicit segmentation, we build models for fixed length subsequences using sliding windows. The results of a temporal segmentation algorithm that can split such a complex video into meaning-

Fig. 14. Shown above are a few sequences from Cluster1. Each row shows contiguous frames of a sequence. We see that this cluster dominantly corresponds to ‘Sitting Spins’. Image best viewed in color. Please see http://www.umiacs.umd.edu/∼pturaga/VideoClustering.html for video results.

ful segments, can be easily plugged in. We use 20 frame long overlapping windows for building models of the video. Also, most of the ‘interesting’ activities such as sitting spins, standing spins, leaps etc are usually few and far between. Further, due to the subsequence approach, there will necessarily be several segments that do not contain any meaningful action. As a simple example, a subsequence that contains the transition from a spin to a jump will not fit into either of these action-clusters. To discover the ‘interesting’ activities, we first need to remove these outlier segments. First, we cluster all the available subsequences into a fixed number of clusters (say 10). Then, from each cluster we remove the outliers using a simple criterion of average distance to the cluster. Then, we recluster the remaining segments. We show some sample sequences in the obtained clusters in figures 14 – 18. We observe that Clusters 1 - 4 correspond dominantly to ‘Sitting Spins’, ‘Standing Spins’, ‘Camel Spins’ and ‘Spirals’ respectively (in a spiral the skater glides on one foot while raising the free leg above hip level). Cluster 5 on the other hand seems to capture the rest of the ‘uninteresting’ actions.

Fig. 15. Shown above are a few sequences from Cluster2. Each row shows contiguous frames of a sequence. Notice that this cluster dominantly corresponds to 'Standing Spins'. Image best viewed in color. Please see http://www.umiacs.umd.edu/∼pturaga/VideoClustering.html for video results.

Retrieval Experiment: We performed a retrieval experiment in which a query segment was selected by the user and provided as input to the matching algorithm. The top 5 matches for two different queries corresponding to Standing spin and Sprial are shown in figures 19 - 20.

8 Conclusions

In this paper, we have proposed a framework to explain perception of activities. We then proposed a CLDS model for representing activities and presented an algorithm for unsupervised learning of the cascade model from long video sequences. We demonstrated the effectiveness of the approach using both far-field surveillance and near-field videos. We also presented a technique for incorporating affine and view-invariance into the clustering algorithm. The results are promising and show that our technique can be used for unsupervised activity indexing as an initial filter for further processing.

Acknowledgments: We wish to thank Dr. Naresh Cuntoor and Mr. Aswin C. Sankara-

Fig. 16. Shown above are a few sequences from Cluster3. Each row shows contiguous frames of a sequence. Notice that this cluster dominantly corresponds to 'Spirals'. Image best viewed in color. Please see http://www.umiacs.umd.edu/∼pturaga/VideoClustering.html for video results.

narayanan for helpful insights. Our utmost thanks are due to Mr. Daniel Weinland for his kind help in providing us with the dataset from [21] and also the code for their recognition and temporal segmentation algorithms. Thanks are also due to Mr. Yang Wang for his help in providing us the skating dataset from [42].

References

[1] P. K. Turaga, A. Veeraraghavan, R. Chellappa, From videos to verbs: Mining videos for activities using a cascade of dynamical systems, IEEE International Conference on Computer Vision and Pattern Recognition (2007) 1–8.

[2] G. Johansson, Visual motion perception, Scientific American.

[3] Y. Rui, Z. Xiong, R. Radhakrishnan, A. Divakaran, T. S. Huang, A unified framework for video summarization, browsing and retrieval, MERL Technical Report TR2004-115.

[4] A. Divakaran, K. A. Peker, S.-F. Chang, R. Radhakrishnan, L. Xie, Video mining: Pattern discovery versus pattern recognition, IEEE International Conference on Image Processing (ICIP) 4 (2004) 2379–2382.

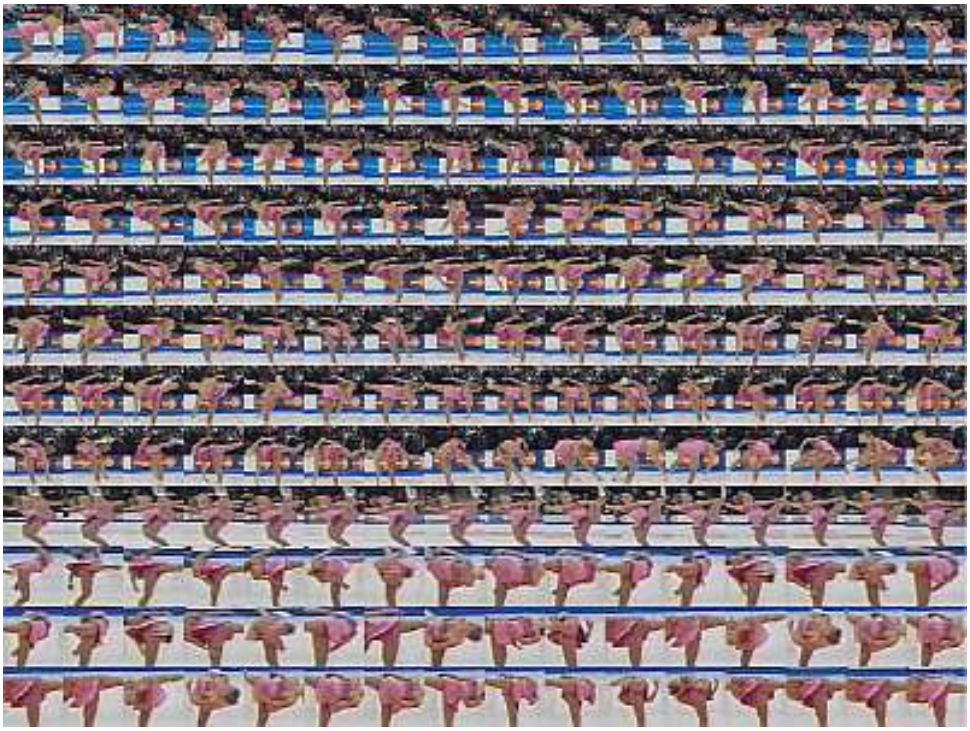


Fig. 17. Shown above are a few sequences from Cluster4. Each row shows contiguous frames of a sequence. This cluster dominantly corresponds to ‘Camel Spins’. Image best viewed in color. Please see http://www.umiacs.umd.edu/∼pturaga/VideoClustering.html for video results.

[5] D. Marr, Vision, W. H. Freeman, 1982.

[6] H. B. Barlow, The coding of sensory messages, Current Problems in Animal Behaviour (1961) 331–360.

[7] M. V. Srinivasan, S. B. Laughlin, A. Dubs, Predictive coding: A fresh view of inhibition in the retina, Proceedings of the Royal Society of London. Series B, Biological Sciences 216 (1205) (1982) 427–459.

[8] M. R. Lemke, M. Schleidt, Temporal segmentation of human short-term behavior in everyday activities and interview sessions, Naturwissenschaften.

[9] V. S. N. Prasad, V. Kellokumpu, L. S. Davis, Ballistic hand movements, Proceedings of Conference on Articulated Motion and Deformable Objects (AMDO) 2006.

[10] D. Marr, E. Hildreth, Theory of edge detection, Proceedings of the Royal Society of London. Series B, Biological Sciences 207 (1167) (1980) 187–217.

[11] A. Veeraraghavan, A. Roy-Chowdhury, R. Chellappa, Matching shape sequences in video with an application to human movement analysis, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (12) (2005) 1896–1909.

[12] G. V. Veres, L. Gordon, J. N. Carter, M. S. Nixon, What image information is important in silhouette based gait recognition?, IEEE International Conference on Computer Vision and Pattern Recognition 2 (2004) 776–782.

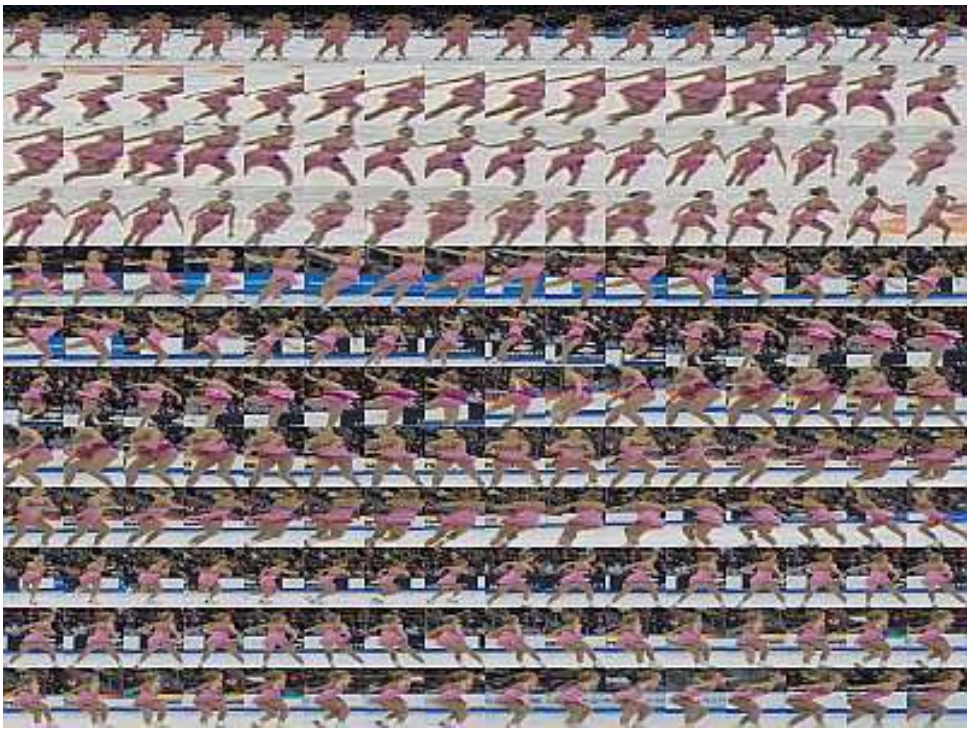


Fig. 18. Shown above are a few sequences from Cluster5. Each row shows contiguous frames of a sequence. This cluster did not dominantly correspond to any ‘interesting’ skating pose but seemed to capture the ‘usual’ postures. Image best viewed in color. Please see http://www.umiacs.umd.edu/∼pturaga/VideoClustering.html for video results.

Fig. 19. Shown above is the input query corresponding to a Standing Spin and the top 5 matches obtained. Image best viewed in color.

Fig. 20. Shown above is the input query corresponding to a Spiral and the top 5 matches obtained. The last match is a false match. Image best viewed in color.

[13] A. Bissacco, P. Saisan, S. Soatto, Gait recognition using dynamic affine invariants, Proc. Of MTNS 2004, Belgium.

[14] N. Vaswani, A. RoyChowdhury, R. Chellappa, "Shape Activities" : A continuous state HMM for moving/deforming shapes with application to abnormal activity detection, IEEE Transactions on Image Processing 14 (10) (2005) 1603– 1616.

[15] S. Ali, M. Shah, A lagrangian particle dynamics approach for crowd flow segmentation and stability analysis, IEEE International Conference on Computer Vision and Pattern Recognition (2007) 1–6.

[16] J. Wright, R. Pless, Analysis of persistent motion patterns using the 3d structure tensor, IEEE Workshop on Motion and Video Computing (2005) 14–19.

[17] E. L. Andrade, S. Blunsden, R. B. Fisher, Hidden markov models for optical flow analysis in crowds, IEEE International Conference on Pattern Recognition (2006) 460–463.

[18] C. Rao, A. Yilmaz, M. Shah, View-invariant representation and recognition of actions, International Journal of Computer Vision 50 (2) (2002) 203–226.

[19] T. Huang, D. Koller, J. Malik, G. H. Ogasawara, B. Rao, S. J. Russell, J. Weber, Automatic symbolic traffic scene analysis using belief networks, National Conference on Artificial Intelligence (AAAI) (1994) 966–972.

[20] N. P. Cuntoor, R. Chellappa, Epitomic representation of human activities, IEEE International Conference on Computer Vision and Pattern Recognition (2007) 1–8.

[21] D. Weinland, R. Ronfard, E. Boyer, Free viewpoint action recognition using motion history volumes, Computer Vision and Image Understanding 104 (2) (2006) 249–257.

[22] X. Feng, P. Perona, Human action recognition by sequence of movelet codewords, 3DPVT (2002) 717–721.

[23] A. Ali, J. Aggarwal, Segmentation and recognition of continuous human activity, IEEE Workshop on Detection and Recognition of Events in Video (2001) 28–35.

[24] A. F. Bobick, J. W. Davis, The recognition of human movement using temporal templates, IEEE Transactions on Pattern Analysis and Machine Intelligence 23 (3) (2001) 257–267.

[25] J. C. Niebles, H. Wang, L. Fei Fei, Unsupervised learning of human action categories using spatial-temporal words, British Machine Vision Conference.

[26] L. Zelnik-Manor, M. Irani, Event-based analysis of video, IEEE International Conference on Computer Vision and Pattern Recognition.

[27] H. Zhong, J. Shi, M. Visontai, Detecting unusual activity in video, IEEE International Conference on Computer Vision and Pattern Recognition (2004) 819–826.

[28] D. Del-Vecchio, R. M. Murray, P. Perona, Segmentation of human motion into dynamics based primitives with application to drawing tasks, Proc. of American Control Conference (2003) 1816 – 1823.

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

[30] V. Pavlovic, J. M. Rehg, Impact of dynamic model learning on classification of human motion., IEEE International Conference on Computer Vision and Pattern Recognition (2000) 1788–1795.

[31] A. Bissacco, A. Chiuso, Y. Ma, S. Soatto, Recognition of human gaits, IEEE International Conference on Computer Vision and Pattern Recognition 2 (2001) 52–57.

[32] A. Kale, A. Rajagopalan, Sundaresan.A., N. Cuntoor, A. Roy Cowdhury, V. Krueger, R. Chellappa, Identification of humans using gait, IEEE Transactions on Image Processing 13 (9) (2004) 1163–1173.

[33] C. Bregler, J. Malik, Learning appearance based models: Mixtures of second moment experts, Advances in Neural Information Processing Systems 9 (1997) 845.

[34] S. M. Oh, J. M. Rehg, T. R. Balch, F. Dellaert, Learning and inferring motion patterns using parametric segmental switching linear dynamic systems, International Journal of Computer Vision 77 (1–3) (2008) 103–124.

[35] A. B. Chan, N. Vasconcelos, Modeling, clustering, and segmenting video with mixtures of dynamic textures, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (5) (2008) 909–926.

[36] Y. A. Ivanov, A. F. Bobick, Recognition of visual activities and interactions by stochastic parsing, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (8) (2000) 852–872.

[37] D. Minnen, I. Essa, T. Starner, Expectation grammars: Leveraging high-level expectations for activity recognition, IEEE International Conference on Computer Vision and Pattern Recognition (2003) 626–632.

[38] S. Hongeng, R. Nevatia, Multi-agent event recognition, IEEE International Conference on Computer Vision 02.

[39] R. Hamid, A. Johnson, S. Batta, A. Bobick, C. Isbell, G. Coleman, Detection and explanation of anomalous activities: Representing activities as bags of event n-grams, IEEE International Conference on Computer Vision and Pattern Recognition 1 (2005) 1031–1038.

[40] C. Castel, L. Chaudron, C. Tessier, What is going on? A High-Level Interpretation of a Sequence of Images, ECCV Workshop on Conceptual Descriptions from Images.

[41] C. Stauffer, W. E. L. Grimson, Learning patterns of activity using real-time tracking, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (8) (2000) 747–757.

[42] Y. Wang, H. Jiang, M. S. Drew, Z. N. Li, G. Mori, Unsupervised discovery of action classes, IEEE International Conference on Computer Vision and Pattern Recognition (2006) 1654–1661.

[43] S. Soatto, G. Doretto, Y. N. Wu, Dynamic textures, IEEE International Conference on Computer Vision 2 (2001) 439–446.

[44] S. M. Oh, J. M. Rehg, F. Dellaert, Parameterized duration modeling for switching linear dynamic systems, IEEE International Conference on Computer Vision and Pattern Recognition (2006) 1694–1700.

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

[46] V. Pavlovic, J. Rehg, Impact of dynamic model learning on classification of human motion, IEEE International Conference on Computer Vision and Pattern Recognition (2000) 788–795.

[47] R. Vidal, S. Soatto, Y. Ma, S. Sastry, An algebraic geometric approach to the identification of a class of linear hybrid systems, In Proc. of IEEE Conference on Decision and Control 1 (2003) 167– 172.

[48] A. Yezzi, S. Soatto, Deformotion: Deforming motion, shape average and the joint registration and approximation of structure in images, International Journal of Computer Vision 53(2) (2003) 153–167.

[49] B. S. Reddy, B. N. Chatterji, An fft-based technique for translation, rotation, and scale-invariant image registration, IEEE Transactions on Image Processing 5 (8) (1996) 1266–1271.

[50] P. V. Overschee, B. D. De Moor, N4sid: Subspace algorithms for the identification of combined deterministic-stochastic systems, Automatica 30 (1994) 75–93.

[51] A. B. Chan, N. Vasconelos, Probabilistic kernels for the classification of auto-regressive visual processes, IEEE International Conference on Computer Vision and Pattern Recognition 1 (2005) 846–851.

[52] M. A. Turk, A. P. Pentland, Face recognition using eigenfaces, IEEE International Conference on Computer Vision and Pattern Recognition (1991) 586–591.

[53] S. Sarkar, P. J. Phillips, Z. Liu, I. R. Vega, P. Grother, K. W. Bowyer, The humanID gait challenge problem: Data sets, performance, and analysis, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (2) (2005) 162–177.

[54] R. J. Martin, A metric for arma processes, IEEE Transactions on Signal Processing 48 (4) (2000) 1164–1170.

[55] K. D. Cock, B. D. Moor, Subspace angles between arma models, Systems and Control Letters 46 (2002) 265–270.

[56] S. V. N. Vishwanathan, A. J. Smola, R. Vidal, Binet-cauchy kernels on dynamical systems and its application to the analysis of dynamic scenes, International Journal of Computer Vision 73 (1) (2007) 95–119.

[57] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (8).

[58] F. R. K. Chung, Spectral Graph Theory, American Mathematical Society, 1997.

[59] B. Rousso, S. Avidan, A. Shashua, S. Peleg, Robust recovery of camera rotation from three frames, In Proc. APRA IU Workshop, 1996.

[60] J. Q. Fang, T. S. Huang, Solving three-dimensional small-rotation motion equations, IEEE International Conference on Computer Vision and Pattern Recognition (1983) 253–258.

[61] S. Mann, R. W. Picard, Video orbits of the projective group: A simple approach to featureless estimation of parameters, IEEE Transactions on Image Processing 6 (9) (1997) 1281–1295.

[62] J. C. Lagarias, J. A. Reeds, M. H. Wright, P. E. Wright, Convergence properties of the nelder–mead simplex method in low dimensions, SIAM Journal on Optimization 9 (1) (1998) 112–147.

[63] D. Chetverikov, R. Pteri, A brief survey of dynamic texture description and recognition, Proc. of the International Conference on Computer Recognition Systems.

[64] Y. Sheikh, M. Sheikh, M. Shah, Exploring the space of a human action., IEEE International Conference on Computer Vision (2005) 144–149.

[65] A. Veeraraghavan, R. Chellappa, A. K. Roy-Chowdhury, The function space of an activity, IEEE International Conference on Computer Vision and Pattern Recognition (2006) 959–968.

[66] V. I. Levenshtein, Binary codes capable of correcting deletions, insertions, and reversals, Doklady Akademii Nauk SSSR 163(4) (1965) 845–848.

[67] H. Akaike, A new look at the statistical model identification, IEEE Transactions on Automatic Control 19 (6) (1974) 716– 723.

[68] G. Schwarz, Estimating the dimension of a model, Annals of Statistics 6 (2) (1978) 461–464.

[69] D. Weinland, R. Ronfard, E. Boyer, Automatic discovery of action taxonomies from multiple views, IEEE International Conference on Computer Vision and Pattern Recognition (2006) 1639–1645.