ABSTRACT

Title of Dissertation: Statistical and Geometric Modeling of Spatio-Temporal Patterns for Video Understanding

Pavan Turaga, Ph.D. Oral Examination, 2009

Dissertation directed by: Professor Rama Chellappa
Department of Electrical and Computer Engineering

Spatio-temporal patterns abound in the real world, and understanding them computationally holds the promise of enabling a large class of applications such as video surveillance, biometrics, computer graphics and animation. In this dissertation, we study models and algorithms to describe complex spatio-temporal patterns in videos for a wide range of applications.

The spatio-temporal pattern recognition problem involves recognizing an input video as an instance of a known class. For this problem, we show that a first order Gauss-Markov process is an appropriate model to describe the space of primitives. We then show that the space of primitives is not a Euclidean space but a Riemannian manifold. We use the geometric properties of this manifold to define distances and statistics. This then paves the way to model temporal variations of the primitives. We then show applications of these techniques in the problem of activity recognition and pattern discovery from long videos.

The pattern discovery problem on the other hand, requires uncovering patterns from large datasets in an unsupervised manner for applications such as automatic indexing and tagging. Most state-of-the-art techniques index videos according to the global content in the scene such as color, texture and brightness. In this dissertation, we discuss the problem of activity based indexing of videos. We examine the various issues involved in such an effort and describe a general framework to address the problem. We then design a

cascade of dynamical systems model for clustering videos based on their dynamics. We augment the traditional dynamical systems model in two ways. Firstly, we describe activities as a cascade of dynamical systems. This 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 given video sequence may be segmented into different clusters where each cluster represents an activity.

Finally, we show the broader impact of the algorithms and tools developed in this dissertation for several image-based recognition problems that involve statistical inference over non-Euclidean spaces. We demonstrate how an understanding of the geometry of the underlying space leads to methods that are more accurate than traditional approaches. We present examples in shape analysis, object recognition, video-based face recognition, and age-estimation from facial features to demonstrate these ideas.

Statistical and Geometric Modeling of Spatio-Temporal Patterns for Video Understanding

by

Pavan K. Turaga

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in fulfillment
of the requirements for the degree of
Doctor of Philosophy
2009

Advisory Committee:
Professor Rama Chellappa, Chair/Advisor
Professor Larry Davis
Professor Adrian Papamarcou
Professor Min Wu
Professor V. S. Subrahmanian

Dedication

Dedicated to my family.

Acknowledgments

I owe my gratitude to all the people who have made this dissertation a reality.

First and foremost I'd like to thank my advisor, Professor Rama Chellappa for accepting me as a student and guiding me through my research. The extremely high standards he sets for himself and the professional heights he has scaled will always remain an inspiration for me. Inspite of his professional achievements, his polite persona and humility teach valuable lessons.

I have been fortunate to have found an opportunity to work with several great researchers all of whom greatly influenced my graduate experience, both within and outside of research – Prof. V. S. Subrahmanian, Prof. Anuj Srivastava, Dr. Yuri Ivanov, and Dr. Tanveer Syeda-Mahmood. Special gratitude is due to Dr. Ashok Veeraraghavan – colleague and collaborator – whose great mentoring during my early graduate days proved invaluable to me.

The former and current fellow graduate students at the Computer Vision Laboratory who made my everday life pleasant deserve a special mention. Aravind Sundaresan, Ashok Veeraraghavan, Narayanan Ramanathan, Gaurav Aggarwal, Seong-Wook Joo, Naresh Cuntoor, Feng Guo, Arun Mohanchettiar, Kaushik Mitra, Soma Biswas, James Sherman, Aswin Sankaranarayanan, Mahesh Ramachandran, Wu Hao, Ruonan Li, Dikpal Reddy, Nitesh Shroff, Raghuraman Gopalan, Sima Taheri, Mohammed Abdelkader, Ming Du, Ming Liu, Vishal Patel.

I would also like to acknowledge help and support from several staff members who make it possible for us to find our ways through the administrative jungle – Janice Perrone

(CfAR), Maria Hoo (ECE), Meg Richmond (IES), the UMIACS staff (Arlene Schenk, Yerty Valenzuela, Edna Walker) and the UMIACS computing staff.

I do not think I can do justice with words in acknowledging the role of my family members - Mom, Dad, Brother, Sis-in-law and Ramya.

My friends and housemates, what a group ! Ramya Chari, Abhinav Gupta, Swati Jarial, Ravi Tandon, Srikanth Vishnubhotla, Gaurav Agarwal, Surbhi Jain, Aravind Sundaresan, Sudarshan Koushik, Gunjan Sharma, Nitin Kumar, Manish Shukla, Archana Anibha, Anuj Rawat, Himanshu Tyagi, Anshu Sarje, Jishnu, Sravya and so many more.

It is impossible to remember all, and I apologize to those I've inadvertently left out.

Lastly, thank you all and thank God!

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Videos play an ever increasing role in our everyday lives with applications ranging from broadcast news, entertainment, scientific research, security and surveillance. Video is a rich source of patterns in the form of spatio-temporal intensity variations. Since such visual patterns evolve with time, we not need to understand the underlying geometry of the pattern that is evolving, but also need to characterize the dynamics of evolution. The goal of this dissertation is to study the related problems of pattern recognition and pattern discovery from video data with various applications that include modeling and recognizing human activities.

We show that short-segments of videos can be considered as outputs of stationary linear dynamic systems which can be parametrized as first-order Gauss-Markov processes. We show under certain assumptions that the parameter-space can be considered as a Grassmann manifold, which is not a linear space but a Riemannian manifold. In order to develop accurate inference algorithms on these manifolds we need to a) understand the geometric structure of these manifolds b) derive appropriate distance measures and c) develop probability distribution functions (pdf) and estimation techniques that are consistent with the geometric structure of these manifolds. We show how accurate statistical characterization that is tuned to the geometry of these manifolds allows us to design efficient algorithms that compare favorably to the state of the art in various applications.

We further consider the problem of modeling the temporal dynamics that give rise to the wide variety of spatio-temporal patterns. In general, the exact nature of these laws is very difficult to estimate. This is because real patterns arise out of complex non-linear processes that are usually unknown. To simplify the problem we consider two models - a sequential compositional model of primitives, and a smooth time-varying model in the primitive space. For both these models, we show how an understanding of the distance metrics and statistics on the manifold of primitives leads to elegant methods for solving the problem.

We apply these techniques to the problem of pattern discovery from large datasets in an unsupervised manner for applications such as automatic indexing and tagging of videos. We examine the various issues involved in such an effort and describe a general framework to address the problem. We design a cascade of dynamical systems model for clustering videos based on their dynamics. We augment the traditional dynamical systems model in two ways. Firstly, we describe activities as a cascade of dynamical systems. This 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. Further, we generalize this approach to the case of complex patterns where a sequential model is not appropriate due to co-articulatory effects. This is generalized by considering the evolution of the dynamics as a smoothly varying linear system whose parameters vary with time. This is modeled as a trajectory on the Grassmann manifold. The dynamics of this variation can be learnt from the data using the geometry of the manifold.

Finally, we show the applicability of the methods developed here for several other problems in computer vision that involve statistical inference over non-Euclidean spaces. Specifically, we show that linear-subspace constraints appear naturally in several vision problems such as shape analysis, object recognition, video-based face recognition, and age-estimation from facial features. We demonstrate how an understanding of the geometry of the Grassmann manifold leads to methods that are more accurate than traditional approaches. This also provides a principled framework for a wide-class of problems involving statistics over subspaces.

1.1 Organization of the Dissertation

In chapter 2, we start with a comprehensive overview of past work in video analysis focusing on human activities. In chapter 3, we discuss a theory of motion perception that leads naturally to the computational model as a linear dynamic system (LDS). We discuss estimation techniques, and distance metrics on the space of LDS. Further, we also discuss geometric variations such as view and execution rate changes and how they influence the model parameters. In chapter 4, we discuss a cascade of dynamical systems model to describe complex activities that are formed by a sequencing of simpler primitives. We show its utility in activity-based video clustering applications. In chapter 5, we discuss a more general time-varying model that can account for the phenomenon of co-articulation and assimilation of primitives on the boundaries. In chapter 6, we discuss in detail the geometry of the parameter-space of the LDS and show that the parameter-space can be considered as a Grassmann manifold. We develop statistical classification techniques on the manifold and show that it can outperform more traditional nearest neighbor classifiers on several applications. In chapter 7, we discuss the broader impact of these methods on several still-image based recognition applications such as age-estimation from facial features, object recognition from landmarks, and object recognition from image-sets. In chapter 8, we discuss directions for future work.

Chapter 2

Related Work

In this chapter, we provide a comprehensive review of various approaches that have been pursued over the past couple of decades in the computer vision community to understand and model human motion and human activities.

2.1 Introduction

Several related survey papers that deal with action and activity modeling in videos have appeared over the years. Most notable among them are the following: Aggarwal and Cai [10] discuss three important sub-problems that together form a complete action recognition system – extraction of human body structure from images, tracking across frames, and finally, recognizing the action. Cedras and Shah [11] present a survey on motion-based approaches to recognition as opposed to structure-based approaches. They argue that motion is a more important cue for action recognition than the structure of the human body. Gavrila [12] presented a survey of literature which focused mainly on tracking of hands and humans via 2D or 3D models and a discussion of action recognition techniques. Recently, Moeslund et al [13] presented a survey of problems and approaches in human motion capture including human model initialization, tracking, pose estimation and activity recognition. Since the scope of the dissertation is limited to recognizing actions from tracked motion or structure features, this chapter will focus exclusively on reviewing approaches for recognition of action and activities from video, and not on the lower-level modules of detection and tracking which is discussed at length in earlier surveys [10, 11, 12, 13, 14].

The terms ‘Action’ and ‘Activity’ are frequently used interchangeably in the vision literature. In the ensuing discussion, by ‘actions’ we refer to simple motion patterns usually executed by a single person, typically lasting for short durations of time on the order of tens of seconds. Examples of actions include bending, walking etc (for example, see figure 2.1). On the other hand, by ‘Activities’ we refer to the complex sequences of actions performed by several people who could be interacting with each other in a constrained manner. They are typically characterized by much longer temporal durations, for example, two persons shaking hands, a football team scoring a goal or a co-ordinated bank attack by multiple robbers (for example, see figure 2.2). This is not a hard boundary and there is a significant ‘gray-area’ between these two extremes. For example, the gestures of a music conductor conducting an orchestra, or the constrained dynamics of a group of humans (see figure 2.3), is neither as simple as an ‘action’ nor as complex as an ‘activity’ according to the above interpretation. However, this simple categorization provides a starting-point to organize many approaches that have been proposed to solve the problem. A quick preview of the various approaches that fall under each of these categories is shown in figure 2.4.

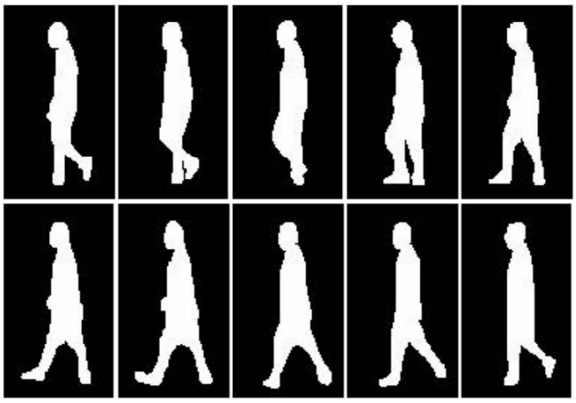
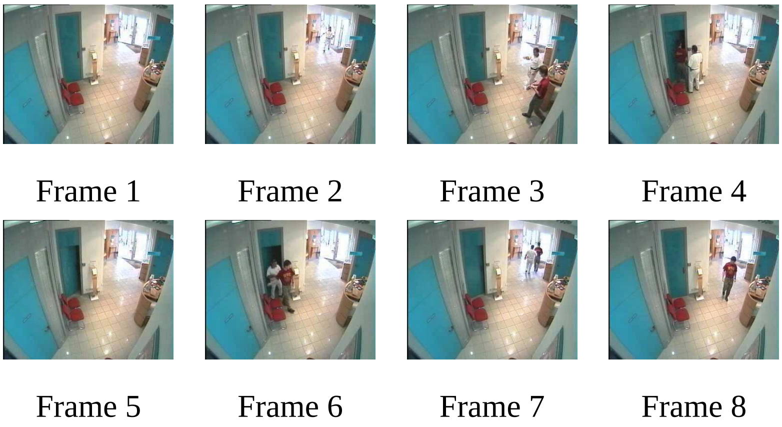


Figure 2.1: Near-field video: Example of Walking action. Figure taken from [4].



Frame 1 Frame 2 Frame 3 Frame 4

Frame 5 Frame 6 Frame 7 Frame 8

Figure 2.2: Medium-field video: Example video sequence of a simulated bank attack.

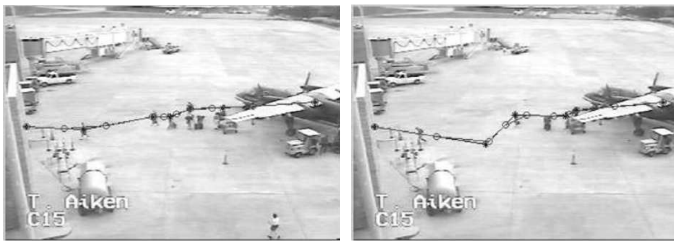


Figure 2.3: Far-field video: Modeling dynamics of groups of humans as a deforming shape. Figure taken from [5].

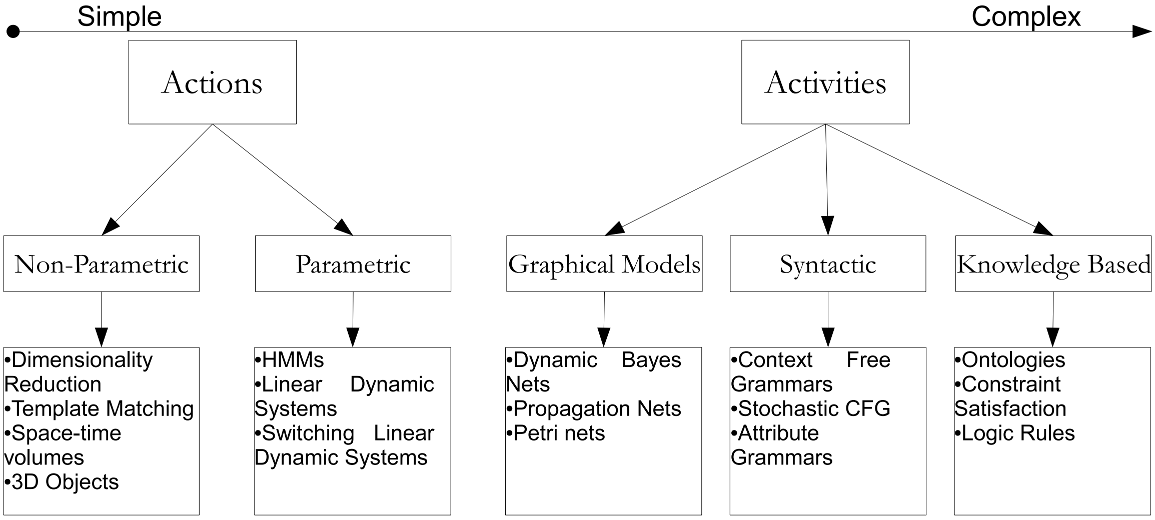


Figure 2.4: Overview of approaches for action and activity recognition.

In this dissertation, we focus on methods of recognition of simple and complex actions. We do not address high-level semantic 'activity' representation and recognition. In this chapter, we review methods for modeling and recognition of simple and complex action classes.

2.2 General Overview

A generic action or activity recognition system can be viewed as proceeding in a series of steps, from a sequence of images to a higher level interpretation. The major steps involved are the following:

1. Input video or sequence of images

2. Extraction of concise low-level features

3. Mid-level action descriptions from low-level features

4. High-level semantic interpretations from primitive actions

Video data consist of massive amounts of raw information in the form of spatio-temporal pixel intensity variations. But, most of this information is not directly relevant to the task of understanding and identifying the activity occurring in the video. A classic experiment by Johansson [15] demonstrated that humans can perceive gait patterns from point light sources placed at a few limb joints with no additional information. Extraneous factors such as the color of the clothes, illumination conditions, background clutter do not aid in the recognition task. We briefly describe a few popular low-level features and refer readers to alternate sources for a more in-depth treatment as we progress.

2.2.1 Optical flow

Optical flow is defined as the apparent motion of individual pixels on the image plane. Optical flow often serves as a good approximation of the true physical motion projected onto the image plane. Most methods to compute optical flow assume that the color/intensity of a pixel is invariant under the displacement from one video frame to the next. We refer the reader to [16] for a comprehensive survey and comparison of optical flow computation techniques. Optical flow provides a concise description of both the regions of the image undergoing motion and the velocity of motion. In practice, computation of optical flow is susceptible to noise and illumination changes. Applications include [17] which used optical flow to detect and track vehicles in traffic.

2.2.2 Point trajectories

Trajectories of moving objects have popularly been used as features to infer the activity of the object. The image-plane trajectory itself is not very useful as it is sensitive to translations, rotations and scale changes. Alternative representations such as trajectory velocities, trajectory speeds, spatio-temporal curvature, relative-motion etc have been proposed that are invariant to some of these variabilities. A good survey of these approaches can be found in [11]. Extracting unambiguous point trajectories from video is complicated by several factors such as occlusions, noise, background clutter etc. Accurate tracking algorithms are needed for obtaining the motion trajectories [14].

2.2.3 Background subtracted blobs

Background subtraction is a popular method to isolate the moving parts of a scene by segmenting it into background and foreground. Several approaches to background modeling exist. One popular approach is to learn a statistical distribution of pixel intensities that correspond to the background as in [18]. By adapting the background model according to new data, the method can also be applied to scenarios with changing background [18].

2.2.4 Shape features

Shape of the human silhouette plays a very important role in recognizing human actions. Several methods have been proposed to quantify shape – global, boundary and skeletal based. Global methods consider the entire shape region to compute the shape-descriptors, for example, shape moments [19]. Boundary methods on the other hand consider only the shape contour as the defining characteristic of the shape. Such methods include chain codes [20] and landmark-based shape descriptors [8]. Skeletal methods represents a complex shape as a set of 1D skeletal curves, for example, the medial axis transform [21]. Applications include shape-based dynamic modeling of the human silhouette as in [22] to perform gait recognition.

2.3 Modeling and Recognizing Actions

Approaches for human action recognition fall into one of the two following categories – a) Methods that rely on human body models, b) Methods that do not rely on human body models. Methods that fall in the first category rely on segmentation of the body into individual parts and extract features such as joint-angles or joint-trajectories. However, segmentation of the human body is a computationally intensive task, and extraction of joints and angles requires good tracking algorithms. These approaches were popular in the early 90s and an excellent survey can be found in [10]. More recently, the focus has shifted to approaches which do not assume a body model, but rely on motion information extracted directly from the images. Motion-based approaches for modeling actions fall into two major classes – parametric and non-parametric. Parametric approaches typically impose a model on the dynamics of the motion. The particular parameters for a class of actions is then estimated from training data. Examples include Hidden Markov Models (HMMs), Linear Dynamical Systems (LDSs) etc. Non-parametric approaches on the other hand do not impose a model, instead relying on coarse representations drawn from data such as action-templates. We will first discuss the non-parametric methods and later, the parametric methods.

2.3.1 Non-Parametric Approaches for Action Recognition

2.3.1.1 2D-templates

One of the earliest attempts at action-recognition that does not depend on 3-D structure estimation was proposed by Polana and Nelson [23]. They first rely on motion-detection and tracking of humans in the scene. After tracking, a ‘cropped’ sequence constraining the human is constructed where scale changes are compensated for. A periodicity index is computed for the given activity and the algorithm proceeds to recognize the action if it is found to be sufficiently periodic. To perform recognition, the periodic sequence is segmented into individual cycles using the periodicity estimate and combined to get an average-cycle. The average-cycle is divided into a few temporal segments and flow-based features are computed for each spatial location in each segment. The flow-features in each segment are averaged into a single frame. The average-flow frames within an activity-cycle form the templates for each action class. Other related approaches for representation and recognition of quasi-cyclic actions have been proposed in [24]. Since, these methods are periodicity-based, they are best suited to quasi-periodic actions such as walking, running, swimming etc.

Bobick and Davis [25] proposed using ‘temporal templates’ as models for actions. In their approach, background subtraction is followed by an aggregation of a sequence of background subtracted blobs into a single static image. They propose two methods of aggregation – the first methods gives equal weight to all images in the sequence, which gives rise to a representation called the ‘Motion Energy Image’ (MEI). The second method gives decaying weights to the images in the sequence with higher weight given to new frames and low weight to older frames. This leads to a representation called the ‘Motion History Image’ (MHI). The MEI and MHI together comprise a ‘template’ for a given action. From the templates, translation, rotation and scale invariant Hu-moments are extracted which are then used for recognition. It was shown in [25] that MEI and MHI have sufficient discriminating ability for several simple action classes such as ‘sitting down’, ‘bending’, ‘crouching’ and other aerobic postures. However, it was noted in [26] that MEI and MHI lose discrimination for complex activities due to overwriting of the motion history and hence are unreliable for matching.

2.3.1.2 3-D Space-time Volumes

While most of the above approaches extract features from individual video frames, direct analysis of actions as 3-D spatio-temporal volumes has also been investigated by several researchers. Chomat et al. [27] model a segment of video as a (x,y,t) spatio-temporal volume and compute local appearance models at each pixel using a Gabor filter bank at various orientation and spatial scales and a single temporal scale. A given action is recognized using a spatial average of the probabilities of individual pixels in a frame. Since, actions are analyzed at a single temporal scale, this method is not applicable to variations in execution rate. As an extension to this approach, local histograms of normalized space-time gradients at several temporal scales are extracted by Zelnik-Manor and Irani [28]. The sum of the chi-square metric between histograms is used to match an input video with a stored exemplar.

Laptev and Lindeberg [29] proposed a spatio-temporal generalization of the well-known Harris interest point detector, which is widely used in object recognition applications, and applied it to modeling and recognizing actions in space-time. Dollar et al. [30] model a video sequence by the distribution of space-time (ST) feature prototypes. The feature prototypes are obtained by k-means clustering of a large set of features – space-time gradients – extracted at ST interest points from the training data. Neibles et al. [31] use a similar approach where they use a bag-of-words model to represent actions. The bag-of-words model is learnt by extracting spatio-temporal interest points and clustering of the features. Since, most of these methods are based on linear operations such as filtering and spatio-temporal gradients, the descriptors are sensitive to changes in appearance, noise, occlusions etc. It has also been noted that interest points are extremely sparse in real-life human actions and certain types of actions do not give rise to distinctive features [31, 30].

2.3.1.3 3D Object models

Successful application of models and algorithms to object recognition problems led researchers in action-recognition to propose alternate representations of actions as spatio-temporal objects. Syeda-Mahmood et al. proposed a representation of actions as generalized cylinders in the joint (x,y,t) space [32]. Yilmaz and Shah [33] represent actions as 3-D objects induced by stacking together tracked 2-D object contours. A sequence of 2-D contours in (x,y) space can be treated as an object in the joint (x,y,t) space. This representation encodes both the shape and motion characteristics of the human. From the (x,y,t) representation, concise descriptors of the object's surface are extracted corresponding to geometric features such as peaks, pits, valleys and ridges. Since this approach is based on stacking together a sequence of silhouettes, accurate correspondence between points of successive silhouettes in the sequences needs to be established. Quasi view-invariance for this representation was shown theoretically by assuming an affine camera model. Similar to this approach, [34] proposed using background subtracted blobs, instead of contours, which are stacked together to create an (x,y,t) binary space-time volume. Since, this approach uses background subtracted blobs, the problem of establishing correspondence between points on contours in the sequence does not exist. From this space-time volume, 3-D shape descriptors are extracted by solving a Poisson equation [34].

2.3.1.4 Manifold Learning Methods

Most approaches in action recognition involve dealing with data in very high-dimensional spaces. Hence, these approaches often suffer from the 'curse of dimensionality'. The feature-space becomes sparser in an exponential fashion with the dimension, thus requiring a larger number of samples to build efficient class-conditional models. Learning the manifold on which the data resides enables us to determine the inherent dimensionality of the data as opposed to the raw dimensionality. The inherent dimensionality contains fewer degrees of freedom and allows efficient models to be designed in the lower-dimensional space. The simplest way to reduce dimensionality is via Principal Component Analysis (PCA) which assumes that the data lies on a linear subspace. Except in very special cases, data does not lie on a linear subspace. This requires methods that can learn the intrinsic geometry of the manifold from a large number of samples. Nonlinear dimensionality reduction techniques allow for representation of data points based on their proximity to each other on nonlinear manifolds. Several methods for dimensionality reduction such as PCA, locally linear embedding (LLE) [35], Laplacian eigenmap [36], and Isomap [37] have been applied to reduce the high-dimensionality of video data in action-recognition tasks (c. f. [38, 39, 40]).

2.3.2 Parametric Methods

The previous section focused on representations and models for the simplest of action classes – known as atomic or primitive actions. The parametric approaches that we will describe in this section are much more powerful modeling tools. Parametric methods such as HMMs, LDSs are well suited to model more complex actions where the underlying process is characterized by complex temporal dynamics. In such cases, simple template matching approaches would either require too many templates or would not capture the dynamics of the action at all. Examples of such complex actions include the steps in a ballet dancing video, juggling a ball or conducting an orchestra using complex hand gestures. Accurate modeling and recognition of this class of complex actions requires more sophisticated methods that explicitly model the temporal dynamics of the action.

The most popular method used for modeling complex temporal dynamics is the so called state-space approach. The state-space approach models the temporal evolution of features as a trajectory in some configuration space, where each point on the trajectory corresponds to a particular ‘configuration’ or ‘state’ – for instance, a particular pose or stance of the actor.

2.3.2.1 Hidden Markov Models

One of the most popular state-space models is the HMM. In the discrete HMM formalism, the state space is considered to be a finite set of discrete points. The temporal evolution is modeled as a sequence of probabilistic jumps from one discrete state to the other. HMMs first found wide applicability in speech recognition applications in the early 80s. An excellent source for a detailed explanation of HMMs and its associated three problems – inference, decoding and learning – can be found in [41]. Beginning in the early 90’s, HMMs have found many applications in computer vision. One of the earliest approaches to recognize human actions via HMMs was proposed by Yamato et al. [42] where they recognized tennis shots such as backhand stroke, backhand volley, forehand stroke, forehand volley, smash etc by modeling a sequence of background subtracted images as outputs of class-specific HMMs. Several successful gesture recognition systems such as in [43, 44], make extensive use of HMMs by modeling a sequence of tracked features such as hand blobs as HMM outputs.

Apart from gesture recognition, HMMs and its extensions have also been used for other action recognition applications such as in Siskind and Morris [45]. HMMs have also found applicability in modeling the temporal evolution of human gait patterns both for action recognition and biometrics (cf. Kale et al. [46], Liu and Sarkar [47]). All these approaches are based on the assumption that the feature sequence being modeled is a result of a single person performing an action. Hence, they are not directly applicable to applications where there are multiple agents performing an action or interacting with each other. To address this issue, Brand et al [48] proposed a coupled HMM to represent the dynamics of interacting targets. They demonstrate the superiority of their approach over conventional HMMs in recognizing two-handed gestures. Incorporating domain knowledge into the HMM formalism has been investigated by several researchers. Moore, Essa and Hayes [49] use HMMs in conjunction with object detection modules to exploit the relationship between actions and objects. Hongeng and Nevatia [50] incorporate *a priori* beliefs of state-duration into the HMM framework and the resultant model is called Hidden semi-Markov Model (semi-HMMs). Cuntoor and Chellappa [51] have proposed a mixed-state HMM formalism to model non-stationary activities, where the state-space is augmented with a discrete label for higher-level behavior modeling.

HMMs are efficient tools for modeling time-sequence data and are useful both for their generative and discriminative capabilities. HMMs are well-suited for tasks that require recursive probabilistic estimates [52] or when explicit segmentation into atomic action units is difficult. However, their utility is restricted due to the simplifying assumptions that the model is based on. Most significantly, the assumption of Markovian dynamics and the time-invariant nature of the model restricts the applicability of HMMs to relatively simple and *stationary* temporal patterns.

2.3.2.2 Linear Dynamical Systems

Linear dynamical systems are a form of HMMs where the state-space is not constrained to be a finite set of symbols but can take on continuous values in $\mathbb{R}^k$ where k is the dimensionality of the state-space. The simplest form of LDS is the first order time-invariant Gauss-Markov process which can be interpreted as a continuous state-space generalization of HMMs with a Gaussian observation model. Several applications such as recognition of humans and actions based on gait (Bissacco et al [53], Veeraraghavan et al [4], Mazzaro et al. [54]) and dynamic texture modeling and recognition [55, 56] have been proposed using LDSs. First order LDSs were used by Vaswani et al [5] to model the configuration of groups of people in an airport tarmac setting by considering a collection of moving points (humans) as a deforming shape. Advances in system identification theory for learning LDS model parameters [57, 58, 59] from data and distance metrics on the LDS space [60] have made LDSs popular for learning and recognition of high-dimensional time-series data.

2.3.2.3 Non-linear Dynamical Systems (NLDS)

While time-invariant HMMs and LDSs are efficient modeling and learning tools, they are restricted to linear and stationary dynamics. Consider the following activity – a person bends down to pick up an object, then he walks to a nearby table and places the object on the table and finally rests on a chair. This activity is composed of a sequence of short segments each of which can be modeled as a LDS. The entire process can be seen as switching between LDSs. To tackle such complex dynamics, a popular approach is to model the process using Switching Linear Dynamical systems (SLDS) or Jump Linear Systems (JLS). An SLDS, consists of a set of LDSs with a switching function that causes model parameters to change by switching between models. Bregler [61] presented a multi-layered approach to recognize complex movements consisting of several levels of abstraction. The lowest level is a sequence of input images. The next level consists of 'blob' hypotheses where each blob is a region of coherent motion. At the third level, blob tracks are grouped temporally. The final level, consists of a HMM which represents the complex behavior. North et al [62] augment the continuous state vector with a discrete state component to make a 'mixed' state. The discrete component represents a mode of motion or more generally a 'switch' state. Corresponding to each switch state, a Gaussian Autoregressive (AR) model is used to represent the dynamics. A maximum likelihood approach is used to learn the model parameters for each motion class. Pavlovic and Rehg [63] model the non-linearity in human motion in a similar framework, where the dynamics are modeled using LDS and the switching process is modeled using a probabilistic finite state-machine. Other applications of this framework include the work of Del Vecchio et al [64] who used this framework for classification of drawing tasks. Though the SLDS framework has greater modeling and descriptive power than HMMs and LDSs, learning and inference in SLDS are much more complicated, often requiring approximate methods [65]. In practice, determining the appropriate number of switching states is challenging and often require large amounts of training data. Apart from maximum likelihood (ML) approaches, algebraic approaches which can simultaneously estimate the number of switching states, the switching instants and also the parameters of the model for each switch state have been proposed by Vidal, Chiuso and Soatto [66]. However, algebraic approaches are often not robust to noise and outliers in the data.

2.3.3 Invariances in Human Action Analysis

One of the most significant challenges in action recognition is to find methods that can explain and be robust to the wide variability in features that is observed within the same action class. Sheikh et. al. [67] have identified three important sources that give rise to variability in observed features. They are

1. Viewpoint
2. Execution Rate
3. Anthropometry

Any real-world action recognition system needs to be invariant to these factors. In this section, we will review some efforts in this direction that have been pursued in the research community.

2.3.3.1 View-Invariance

A fundamental problem in video-based recognition of activities is achieving view invariant representations of actions. While it may be easy to build statistical models of simple actions based on the representations discussed so far from a single view, it is extremely challenging to generalize them to other views even for very simple action classes. This is due to the wide variations in motion-based features induced by camera perspective effects and occlusions. One way to deal with the problem is to store templates from several canonical views as done by Bobick et al. [25] and interpolate across the stored views as proposed by Darrell, Essa and Pentland [68]. This approach however is in general not scalable since one does not know how many views to consider as canonical. Another approach is to assume that point correspondences across views are available as in Syeda-Mahmood et al. [32] and compute a transformation that maps a stored model to an example from an arbitrary view. Seitz and Dyer [24] present an approach to recognize cyclic motion that is affine-invariant by assuming that feature correspondence between successive time-instants is known. It was shown by Rao and Shah [69] that extrema in space-time curvature of trajectories is preserved across views. The extrema in space-time curvature of hand trajectories are denoted as 'dynamic instants'. An action is then considered as a sequence of dynamic instants which is preserved across several views. Another example is the work of Parameswaran and Chellappa [70, 71] who define a view invariant representation of actions based on the theory of 2D and 3D invariants. In their approach, they consider an action to be a sequence of *poses*. They assume that there exists at least one *key-pose* in the sequence in which 5 points are aligned on a plane in the 3-D world coordinates. Using this assumption, they derive a set of view-invariant descriptors. More recently, the notion of motion-history [25] was extended to 3-D by Weinland et al [2] where the authors combine views from multiple cameras to arrive at a three-dimensional binary occupancy volume. Motion history is computed over these 3-D volumes and view-invariant features are extracted by computing the circular FFT of the volume.

2.3.3.2 Execution Rate Invariance

The second major source of observed variability in features arises from the differences in execution rates while performing the same action. Variations in execution style exist both in inter-person and intra-person settings. State-space approaches are robust to minor changes in execution rates, but are not truly rate-invariant since they do not explicitly model transformations of the temporal axis ((c. f. Bobick and Wilson [72], Hoey and Little [73])). Mathematically, the variation in execution rate is modeled as a warping function of the temporal scale. The simplest case of linear time-warps can be usually dealt with fairly easily (c. f. [25, 74]). To model highly non-linear warping functions, common methods methods include Dynamic Time Warping of the feature sequence such as the works of Takahashi et. al [75], Darrel et al [68], Giese and Poggio [76], Rao et al [77] and Veeraraghavan et al [1].

2.3.3.3 Anthropometric Invariance

Anthropometric variations such as those induced by the size, shape, gender etc. of humans is another important class of variabilities that requires careful attention. Unlike viewpoint and execution-rate variabilities which have been well-studied, a systematic study of anthropometric variations has only been receiving interest in recent years. Ad hoc methods which normalize the extracted features to compensate for changes in size, scale etc. are usually employed when no further information is available. Drawing on studies on human anthropometry, Gritai et al. in [78], suggested that the anthropometric transformation between two different individuals can be modeled as a projective transformation of the image co-ordinates of body joints. Based on this, they define a similarity metric between actions, by using epipolar geometry to provide constraints on actions performed by different individuals.

2.4 Modeling and Recognizing Complex Activities

Most activities of interest in applications such as surveillance, content-based indexing etc involve several actors, who interact not only with each other, but also with contextual entities. The approaches discussed so far are mostly concerned with modeling and recognizing actions of a single actor. Modeling a complex scene and the inherent structure and semantics of complex activities require higher-level representation and reasoning methods. The previously discussed approaches are not suited to deal with the complexities of spatio-temporal constraints on actors and actions, temporal relations such as sequencing and synchronization, and the presence of multiple execution threads. Thus, structural and syntactic approaches such as dynamic belief networks, grammars, petri-nets etc are well-suited to tackle these problems. Moreover, some amount of domain knowledge can be incorporated in the design of concise and intuitive structural descriptions of activities. Syntactic and structural methods typically follow a hierarchical approach. At the lower levels are the standard vision modules such as background-foreground segmentation, tracking, object detection etc. At the mid-level are action-recognition modules such as the ones discussed so far. At the high-level are the reasoning engines which encode the activity semantics/structure based on lower level action-primitives.

2.4.1 Graphical Models

2.4.1.1 Belief Networks

A Bayesian network (BN) [79] is a graphical model that encodes complex conditional dependencies between a set of random variables. BNs are directed acyclic graphs where the nodes represent random variables and directed edges represent causality relations. Dynamic Belief networks (DBNs) are a generalization of the simpler Bayesian networks which incorporate temporal dependencies between random variables. DBNs encode far more complex conditional dependence relations among several random variables as opposed to just one hidden random variable in the case of HMMs. Development of efficient algorithms for learning and inference in graphical models (c. f. [80, 81]) have made them popular tools to model structured activities [17]. Methods to learn the topology or structure of Bayesian networks from data [82] have also been investigated in the machine learning community.

2.4.1.2 Petri Nets

Petri Nets were defined by Carl Adam Petri as a mathematical tool for describing relations between conditions and events. Petri Nets are particularly useful to model and visualize behaviors such as sequencing, concurrency, synchronization and resource sharing. Conditions refers to the state of an entity and events refer to changes in the state of the entity. Petri nets have traditionally found use in modeling hybrid systems, where they are well-suited to model complex behavior such as concurrency, synchronization and resource sharing [83, 84]. Petri Nets were used by Castel et al [85] to develop a system for high-level interpretation of image sequences and by Ghanem et al [86] as a tool for querying surveillance videos. Albanese et al [87] have recently proposed the concept of a probabilistic Petri Net (PPN).

2.4.1.3 Other Graphical Models

While DBNs are an attractive means to model relations between several variables, they are not particularly well suited for describing complex temporal relations other than simple sequencing. Researchers have proposed alternate graphical approaches that specifically model more complex temporal relations such as sequentiality, duration, parallelism, synchrony etc. Examples include the work of Pinhanez and Bobick [88] who use a simplified version of Allen's interval algebra to model sophisticated temporal ordering constraints such as past, now, future (PNF). Shi et al [89] represent activities using partially ordered temporal intervals. In their approach, an activity is constrained by temporal and logical ordering, including duration, of the activity intervals.

2.4.2 Syntactic Approaches

Syntactic pattern recognition approaches such as Context-free grammars (CFG) express the structure of a process using a set of production rules. To draw a parallel to grammars in language modeling, the production rules specify how complex sentences (activities) can be constructed in a grammatically sound manner from simpler words (activity primitives), and how to recognize if a given sentence (video) conforms to the rules of a given grammar (activity model). Syntactic approaches are useful when the structure of a process is difficult to learn but may be known a priori. Syntactic pattern recognition approaches were first successfully applied to still-image recognition tasks such as shape modeling [90]. Success in these domains coupled with the success of HMMs and DBNs in action-recognition tasks, led to renewed interest in syntactic approaches for activity recognition.

2.4.2.1 Context free Grammars

One of the earliest use of grammars for visual activity recognition was proposed by Brand [91], who used a grammar to recognize hand manipulations in sequences containing disassembly tasks. They made use of simple grammars with no probabilistic modeling or error analysis. Ryoo and Aggarwal [92] used the CFG formalism to model and recognize composite human activities and multi-person interactions. They followed a hierarchical approach where the lower-levels are composed of HMMs and Bayesian Networks. The higher-level interactions are modeled by CFGs.

2.4.2.2 Stochastic Grammars

Algorithms for detection of low-level primitives are frequently probabilistic in nature. Thus, Stochastic Context-free grammars (SCFGs) which are a probabilistic extension of CFGs were found to be suitable for integration with real-life vision modules. SCFGs were used by Ivanov and Bobick [93] to model the semantics of activities whose structure was assumed to be known. They used HMMs for low-level primitive detection. The grammar production rules were augmented with probabilities and a ‘skip’ transition was introduced. This resulted in increased robustness to insertion errors in the input stream and also to errors in low-level modules. Results on surveillance videos and complex gestures of a music conductor showed promising results. Moore and Essa [94] used SCFGs to model multi-tasked activities – activities that have several independent threads of execution with intermittent dependent interactions with each other, as demonstrated in a Blackjack game with several participants.

2.4.3 Knowledge and Logic-based Approaches

Logic and knowledge based approaches express activities in terms of primitives and constraints on them. These methods can express far more complex constraints than grammar based approaches. While grammars can be efficiently parsed due to their syntactic structure, logical rules can lead to a computational overhead due to constraint satisfaction checks. But, logical rules are often far more intuitive and human-readable than grammatical rules.

2.4.3.1 Logic Based Approaches

Logic-based methods rely on formal logical rules to describe constraints in activities. Logical rules are useful to express domain knowledge as input by a user or to present the results of high-level reasoning in an intuitive and human-readable format. Medioni et al. [95] propose a hierarchical representation to recognize a series of actions performed by a single agent. Symbolic descriptors of actions are extracted from low-level features through several mid-level layers. Then, a rule based method is used to approximate the probability of occurrence of a specific activity, by matching the properties of the agent with the expected distributions (represented by a mean and a variance) for a particular action. In a later work Hongeng, Nevatia and Bremond [96] extend this representation by considering an activity to be composed of several action threads. Each action thread is modeled as a stochastic finite-state automaton. Constraints between the various threads are propagated in a temporal logic network. Shet et al [97] propose a system that relies on logic programming to represent and recognize high-level activities. Low level modules are used to detect primitive events. The high level reasoning engine is based on Prolog, and recognizes activities which are represented by logical rules between primitives.

2.4.3.2 Ontologies

In most practical deployments, that use any of the afore-mentioned approaches, symbolic activity definitions are constructed in an empirical manner. Though empirical constructs are fast to design and even work very well in most cases, they are limited in their utility to the specific deployment for which they have been designed. Hence, there is a need for a centralized representation of activity definitions or ontologies for activities which are independent of algorithmic choices. Ontologies standardize activity definitions, allow for easy portability to specific deployments, enable interoperability of different systems and allow easy replication and comparison of system performance. Chen et al. [98] use ontologies for analyzing social interaction in nursing homes. Hakeem et al have used ontologies for classification of meeting videos [99]. Georis et al [100] use ontologies to recognize activities in a bank monitoring setting. Bremond and Thonnat [101] have investigated the use of contextual information in activity recognition through domain ontologies. As a result of the Video Event Challenge Workshops held in 2003 [102], ontologies have been defined for six domains of video surveillance - 1) Perimeter and Internal Security, 2) Railroad Crossing Surveillance, 3) Visual Bank Monitoring, 4) Visual Metro Monitoring, 5) Store Security, 6) Airport-Tarmac Security. This led to the development of two formal languages - The Video Event Representation Language (VERL) [103], which provides an ontological representation of complex events in terms of simpler sub-events, and the Video Event Markup Language (VEML) which is used to annotate VERL events in videos. Though ontologies provide concise high-level definitions of activities, they do not necessarily suggest the right ‘hardware’ to ‘parse’ the ontologies for recognition tasks.

Chapter 3

Spatio-Temporal Models for Videos

In this chapter, we discuss a hierarchy of perceptual processes that start from low-level pixel intensity variations, towards higher level semantic interpretation of human motion. This will lay the foundations for the computational models and methods that shall be used later in the dissertation.

3.1 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 [104] 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. The dynamic sketch provides a coarse description of shape and motion characteristics of the actor or group of actors involved in the activity. In computational terms, this stage corresponds to the extraction of low-level features from each frame (or pair of frames) of the video. 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 of the sensory field that are relevant for activity based video indexing. Visual encoding mechanisms present in the human brain mimic this phenomenon and is called predictive coding. Barlow [105] and Srinivasan et.al. [106] contend that predictive coding is not just a mechanism for compression but actually goes much further than compression and enables animals to process information in a timely manner. They argue that in the absence of such predictive encoding mechanisms in the neuronal responses, the visual information would flood the brain of these animals and not allow for timely response to these visual stimuli. We refer the interested reader to early works of Barlow, Srinivasan and Marr ([105], [106], [104]) 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 [107]. For example, a car parking activity may be considered to be formed of the following primitives - ‘Car enters parking lot’, ‘Car stops in parking slot’, ‘Person walks away from car’. Such a description requires the ability to segment an activity into its constituents and then develop a model 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 chapter, 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 [108]. 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. There is also evidence from neuroscience about the existence of ‘mirror neurons’ in humans. These neurons fire not just when a particular activity is performed, but also when the same activity is observed by the subject as being performed by someone else [109]. This suggests that there is a strong correlation between the way we perform activities and the way we recognize them. In computational terms, this suggests that the underlying mathematical model for activity recognition and activity synthesis should be the same.

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-thread activity can be said to consist of a linear sequence of a few primitives. An example of a single-thread activity is ‘Person approaches a door’ → ‘Person swipes the access card’ → ‘Person enters a building’. Similarly, a complex multi-thread activity can be seen as a collection of several single-thread 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. As mentioned earlier, evidence from neuroscience [109] suggests the use of a common mathematical framework, that allows for activity recognition as well as activity synthesis. In the context of machine learning, this requires the model to be both discriminative (recognition) and generative (synthesis) in nature. The model should also be rich enough to accommodate the addition of new activities i.e. it should be possible to create representations for new activities using the same general rules of combination, using a different set of primitives.

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.

3.2 Computational Models

There exists a 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.

3.2.1 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 [104]. 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 chapter, 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 3.1 some widely used low-level features and their respective characteristics.

3.2.2 Action-sketches

A significant body of work in activity recognition builds upon extracting action-primitives and modeling the interactions between them. One approach has been to define action-primitives a priori using domain knowledge and user experience. This approach has obvious limitations, since it requires one to enumerate a new list of primitives for every new domain. Thus, techniques for automatic primitive extraction have been gain-

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 ([4, 110])
Shape	Near Field	Single agent	Moderate	Can be incorporated by affine invariance on shapes	Gait Recognition ([4, 22]), Far Field Activity Recognition ([5])
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 ([111, 112], Crowd Monitoring ([113])
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([114], Traffic monitoring ([17]), Far-field surveillance [51])
Circular Fourier Features	Medium and Near field	Single Agent	Moderate	View Invariant	Action recognition [3]

Table 3.1: Various Features for the low-level representation (Dynamic Sketch) and their properties and applicability in various scenarios

ing importance in recent years. 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. [115] 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 [116] 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 [25]. 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 [31]. In a similar approach, [28] 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 [7].

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 [117]. Modeling of complex activities using a switching linear dynamic system, where each system corresponding to an action-primitive was proposed in [62] and [118]. Similarly, human gait patterns have been modeled as linear dynamical systems in [4, 53] and by HMM’s in [46].

We summarize in Table 3.2 some of the well-known tools and their respective characteristics.

3.2.3 Semantic Sketches

In 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 among action primitives [115], [61]. 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 [17]. Complex activities can be modeled as being generated by a switching linear dynamic system as in [62], [118], [119] 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 [56].

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 [93, 120]. Temporal logic networks which encode logical relationships between primitives were used for recognizing events involving multiple objects in [121]. A bag of primitives approach is used in [122] 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 [85].

3.3 Modeling motion primitives with Dynamical Systems

First, we assume that a suitable low-level feature has been chosen that encodes the desired properties such as shape and motion. Given a sequence of these features, we would now like to represent them in a compact manner. In this section, we show that LDS is an appropriate model to describe short-term dynamics. We review the necessary mathematical details and estimation algorithms for LDSs and show that they are well suited to model human actions.

Linear Dynamical System for Action Elements: The dynamics of each action element can be modeled using a time-invariant dynamical system. In several scenarios (like

Property	**CLDS**	SLDS [62] [118]	Grammars [93]	DBNs [17]	Sliding window approaches [7, 28]
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 3.2: Various approaches for activity based mining from video and their characteristics

far-field surveillance, objects moving on a plane etc), it is reasonable to model constant motion in the real world using an LDS model on the image plane. Given the boundaries between action elements, we model each of these segments using an LDS model. Lets 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

$$f(t) = Cz(t) + w(t) \quad w(t) \sim N(0,R) \tag{3.1}$$

$$z(t+1) = Az(t) + v(t) \quad v(t) \sim N(0,Q) \tag{3.2}$$

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 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. We denote the cross correlation between w and v by S. The parameters of the model are given by the transition matrix A and the state matrix C. We note that the choice of matrices A,C,R,Q,S is not unique. However, we can transform these models to their corresponding “innovation representations” [57] which is unique. Similar models have been successfully applied in several tasks such as dynamic texture synthesis and analysis [123], comparing silhouette sequences [4], [53] 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 4.2.1. In summary, the parametric model for each segment consists of the measurement matrix C and the transition matrix A.

3.4 Estimation of the model parameters

It is easily shown that there are infinitely many choices of parameters that give rise to the same sample path $f(t)$. Resolving this ambiguity requires one to impose further constraints and choose a canonical model. The conditions as proposed in [123] are that

$m >> d, rank(C) = d$ and $C^T C = I$. The number of unknowns that need to be solved for are: $md - \frac{d(d+1)}{2}$ for C, d^2 for A, $\frac{d(d+1)}{2}$ for Q: resulting in $md + d^2$ unknowns (we have ignored the observation noise covariance as of now). For each observed frame we get m equations. Hence, $d+1$ linearly independent observations are sufficient to solve for the required parameters ($m(d+1) > md + d^2$ since $m >> d$).

Using these constraints, the parameter estimates can be obtained in closed form. The algorithm is described in [57] and was adopted for texture modeling in [123]. Let observations $f(1), f(2), \ldots f(\tau)$, represent the features for the frames $1, 2, \ldots \tau$. Let $[f(1), f(2), \ldots f(\tau)]$ = $U\Sigma V^T$ be the singular value decomposition of the data. Then $\hat{C} = U, \hat{A} = \Sigma V^T D_1 V (V^T D_2 V)^{-1} \Sigma^{-1}$, 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 4.1 (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 [124] for brief details of the approach. This makes the algorithm for estimating the system and transition matrices, efficient, robust, simple and closed-form.

3.5 Generative Power of the Model

A useful test for a representational model is to synthesize from it, and see how well 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 [125] 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 3.1.

In the next experiment, we generated a bending sequence. During the learning

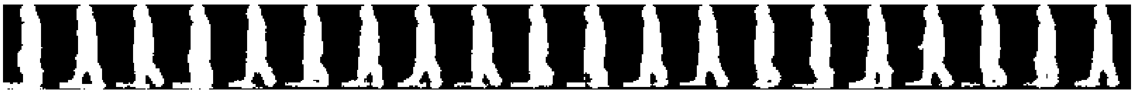


Figure 3.1: A model for the silhouette dynamics for gait was learnt using 1 segment. Shown above is the generated gait sequence from the learnt model.

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 3.2.

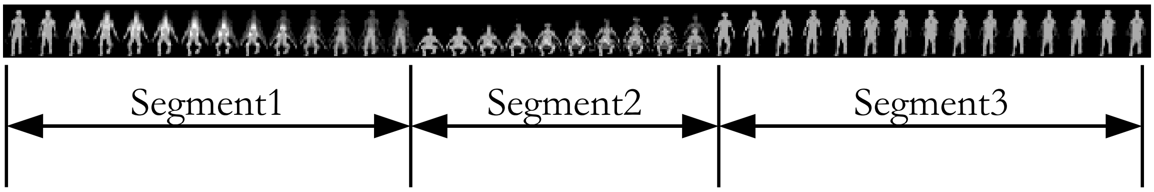


Figure 3.2: A model for silhouette dynamics during 'bending' was learnt using 3 segments. Shown above is the generated bending sequence from the learnt cascade of LTI models.

3.6 Model Order Selection

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 [125] using $d = 5, 10, 15$ on Probes A-G. Results are shown in figure 3.3. 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) [126], Bayesian Information Criteria (BIC) [127], 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.

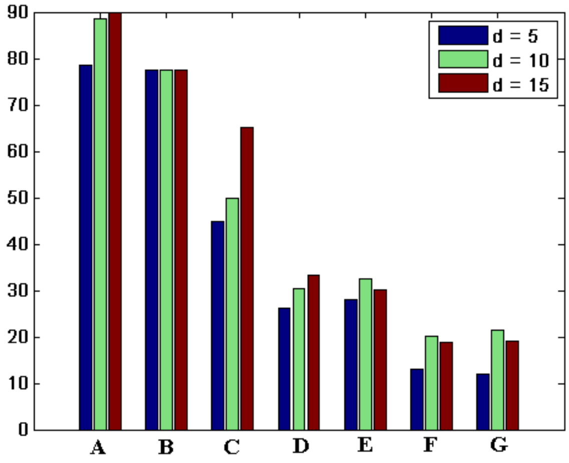


Figure 3.3: Model order selection experiment on the USF gait database. Bar plot shows recognition performance as a function of the hidden state dimension (d) on the 7 different challenge experiments (probes A-G) in the USF gait database.

3.7 Distance Metrics on LDS space

One of the most commonly used distance metrics on the LDS space is based on subspace angles $(\theta_i, i = 1,2,....n)$ between two ARMA models. These are defined in [60] 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 [60]. The subspace angles $(\theta_1, \theta_2, ...)$ between the range spaces of two matrices A and B is recursively defined as follows [60],

$$cos\theta_1 = \max_{x,y} \frac{\left|x^T A^T B y\right|}{\|Ax\|_2 \|By\|_2} = \frac{\left|x_1^T A^T B y_1\right|}{\|Ax_1\|_2 \|By_1\|_2} \tag{3.3}$$

$$cos\theta_k = \max_{x,y} \frac{\left|x^T A^T B y\right|}{\|Ax\|_2 \|By\|_2} = \frac{\left|x_k^T A^T B y_k\right|}{\|Ax_k\|_2 \|By_k\|_2} for \quad k = 2, 3, ... \tag{3.4}$$

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 [60].

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:

$$d_M^2 = \ln \prod_{i=1}^{n} \frac{1}{\cos^2(\theta_i)}, \quad d_g = \sin\theta_{max}, \quad d_F^2 = 2\sum_{i=1}^{n} \sin^2\theta_i \tag{3.5}$$

3.8 Building Invariances into the LDS Distance Metrics Model

The distance metrics defined in the previous section do not take into account geometric transformations that do not alter the perception of the spatio-temporal pattern. When there is a change in viewpoint or there is an affine transformation of the low-level features, the distance metrics will break down. 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 ([25]) 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 makes the algorithm flexible to the choice of feature.

3.8.1 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 in the state equation. As described in section 3.4 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: 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.

$$\int_{D_1} K_X(\overline{p}, \overline{s})\phi_n^X(\overline{s})d\overline{s} = \lambda_n^X \phi_n^X(\overline{p}), \quad \int_{D_1} \phi_m^X(\overline{s})\phi_n^X(\overline{s}) = \delta(m,n) \tag{3.6}$$

where both $\overline{p}, \overline{s} \in D_1$ and $\delta(m,n) = \{1 \text{ if m = n, 0 otherwise}\}$. 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.

$$\int_{D_2} K_G(\overline{q}, \overline{r})\phi_n^X(T(\overline{r}))d\overline{r} = \int_{D_2} K_X(T(\overline{q}), T(\overline{r}))\phi_n^X(T(\overline{r}))d\overline{r} \tag{3.7}$$

$$= \int_{D_1} K_X(\overline{p}, \overline{s})\phi_n^X(\overline{s})\frac{1}{|J_T(\overline{r})|}d\overline{s} \tag{3.8}$$

where (3.8) is obtained by a change of variables given by $\overline{p} = T(\overline{q}), \bar{s} = T(\bar{r})$, and $|J_T(\bar{r})|$ is the determinant of the Jacobian of T with respect to $\bar{r}$ evaluated at $\bar{r} = T^{-1}(\bar{s})$. Now, if $|J_T(\bar{r})| = |J_T| = constant$, then it comes out of the integral in (3.8), and using (3.6) we obtain

$$\int_{D_2} K_G(\overline{q},\bar{r})\phi_n^X(T(\bar{r}))d\bar{r} = \frac{\lambda_n^X}{|J_T|}\phi_n^X(T(\overline{q})) \tag{3.9}$$

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.

3.8.2 Application to Invariances

When two images are related by a general spatial transform (affine, homography etc), they are related by $I_2(x,y) = I_1(T(x,y))$.

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

$$T(\overline{p}) = \begin{bmatrix} a_{11}x + a_{12}y + t_1 \\ a_{21}x + a_{22}y + t_2 \end{bmatrix} \tag{3.10}$$

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 $H = \left[h_{ij}\right]$. In the inhomogeneous coordinates the transformation is given by

$$T(\overline{p}) = \begin{bmatrix} (h_{11}x + h_{12}y + h_{13})/(h_{31}x + h_{32}y + h_{33}) \\ (h_{21}x + h_{22}y + h_{23})/(h_{31}x + h_{32}y + h_{33}) \end{bmatrix} \tag{3.11}$$

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 [128]

$$H = M'(R + \frac{Tn^T}{d_\pi})M^{-1} \tag{3.12}$$

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 $T = [\varepsilon_x, \varepsilon_y, \varepsilon_z]'$ and R using small rotations as [129]

$$R \approx \begin{bmatrix} 1 & -n_3\theta & n_2\theta \\ n_3\theta & 1 & -n_1\theta \\ -n_2\theta & n_1\theta & 1 \end{bmatrix} \tag{3.13}$$

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 (3.12) and simplifying, we obtain the following relations between the required elements of H – h_{31}, h_{32}, h_{33},

$$\frac{h_{31}}{h_{33}} = \frac{a/f}{-ax_0/f - by_0/f + c} \tag{3.14}$$

$$\frac{h_{32}}{h_{33}} = \frac{b/f}{-ax_0/f - by_0/f + c} \tag{3.15}$$

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

$$\lim_{\theta,\varepsilon_x,\varepsilon_y,\varepsilon_z \to 0} \frac{h_{31}}{h_{33}} = 0 \tag{3.16}$$

$$\lim_{\theta,\varepsilon_x,\varepsilon_y,\varepsilon_z \to 0} \frac{h_{32}}{h_{33}} = 0 \tag{3.17}$$

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.

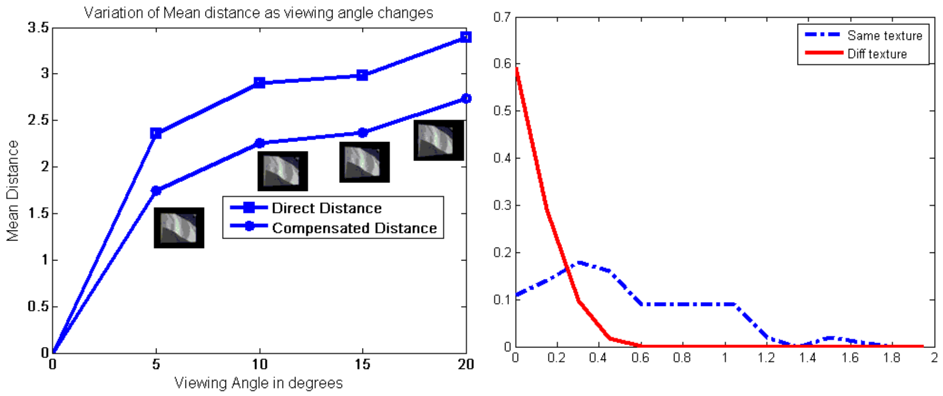
Note: The invariance theorem was proved for continuous random fields. In real images, spatial transforms are not one-to-one maps due to the discrete nature of the underlying lattice. But, our experiments suggest that this theorem can be used to get very good approximations even in the discrete case.

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

$$d_{compensated}(S_1, S_2) = \min_T d(T(S_1), S_2) \tag{3.18}$$

where $d(.,.)$ is any of the distance metrics in (3.5), 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 (3.18). Depending on the complexity of the transformation model, one can use featureless image registration techniques such as [130], [131] 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 4.2.3). We could not arrive at closed form expressions for the gradients. Instead, we resort to using 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 computed or accessed. Even though only limited convergence results for the NM method are known, it is known to work well in practice [132].

To illustrate the effectiveness of our proposed technique, we conducted the following experiment. We took a set of 10 dynamic textures from [133]. 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 3.4(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 3.4(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.



(a)

(b)

Figure 3.4: (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

3.8.3 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 [67]. In the general case, one needs to consider warping functions of the form $g(t) = f(w(t))$ such as in [1] where 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 execution rate well. It can be shown that, under linear warps the stationary distribution of the Markov process in (3.2) 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: $X_1(k) = A_1X_1(k-1) + v(k)$. Ignoring the noise term for now, we can write $X_1(k) = A_1^kX(0)$. 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 $A_1 = V_1D_1V_1^{-1}$, and $A_2 = V_2D_2V_2^{-1}$. Then, for rational q, $A_2 = A_1^q = V_1D_1^qV_1^{-1}$. 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

$$\hat{q} = \frac{\sum_i log\left|\lambda_2^{(i)}\right|}{\sum_i log\left|\lambda_1^{(i)}\right|} \tag{3.19}$$

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 3.8.1. Note that compensation for execution rate is done only for segments which have very similar $\hat{C}$ matrices.

3.9 View Invariance-Simulated Data

We show a recognition experiment based on our modified distance metric. In the next experiment, the setup is the same as described 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 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 features and building models around its neighborhood. We call it Center of Mass Heuristic – CMH. Recognition percentages are shown in table 3.3. The baseline column corresponds to direct application of the Frobenius distance. We see that our method performs better in almost all cases.

	Baseline		CMH		Compensated distance	
	Exemplars		Exemplars		Exemplars	
Activity	1	10	1	10	1	10
1	40	0	40	40	40	50
2	0	0	0	10	70	80
3	0	0	20	40	10	20
4	40	30	10	20	30	60
5	30	30	40	20	40	40
6	10	0	40	50	30	50
7	0	10	0	30	30	70
8	0	10	30	40	0	40
9	0	40	20	20	30	70
10	0	0	10	20	40	40
Average	12	12	21	29	32	52

Table 3.3: 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.

Chapter 4

Sequence of Dynamical Systems for Video Clustering

Parallel to the development of accurate and efficient recognition techniques, there has also been a lot of interest into the discovery of patterns from raw data in the pattern recognition community. 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 [134]. The pattern discovery approach has also been pursued for this problem domain such as in [135]. 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. In the absence of extra information, such as the specific time and location of an unusual activity, 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 cameras 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.

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. [136] showed evidence that supports the notion that infants solve this problem by using coherent patterns of sounds to discover syllables and transitions of syllables within words to distinguish the ends of words. We use a similar framework in the context of activities - where 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.

Most single-agent activities in surveillance settings consist of an actor (subject) executing a series of action elements (verbs) 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-verbs) contributes to a typical activity. Moreover, several multi-agent activities may also be adequately represented by a sequence of actions. Thus, CLDS 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 verbs (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 action-elements interact with each other is used to discover the activities themselves. The overall system overview is shown in figure 4.1. Each of the components will be described in detail in the ensuing discussion.

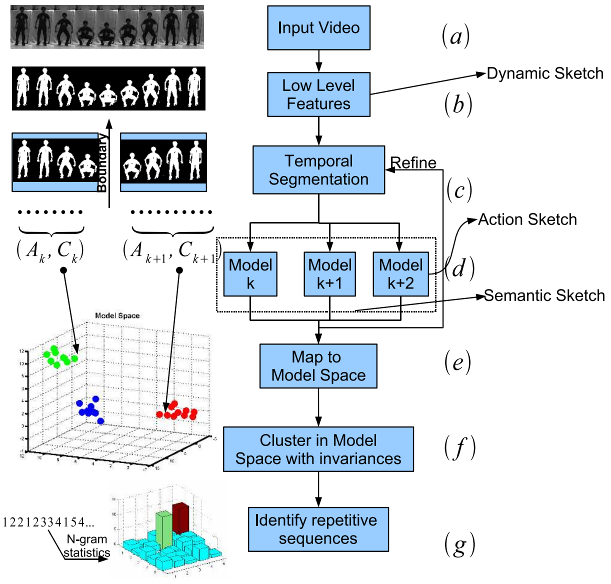


Figure 4.1: System Overview: (a) Input video, (b) Feature extraction (Dynamic Sketch), (c) Temporal segmentation, (d) Build and learn dynamical models, (e,f) Cluster in model space taking into account invariances on the data, (g) Identify repetitive activities

4.1 Sequence of Dynamical Systems

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. 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 LDS. Figure 4.2 illustrates the complete model for such an activity.

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

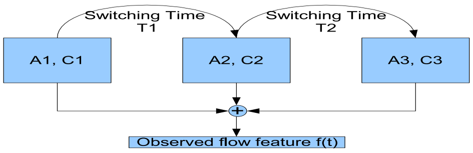


Figure 4.2: Illustration of a cascade of three linear dynamical systems. The temporal order of the execution of these dynamical models and their switching times are shown with arrows.

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

$$g(x;\alpha,\beta) = x^{\alpha-1}\frac{\beta^{\alpha}e^{-\beta x}}{\Gamma(\alpha)} \quad for\ x > 0 \tag{4.1}$$

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

$$\mu = \frac{\alpha}{\beta}, \quad \sigma^2 = \frac{\alpha}{\beta^2} \tag{4.2}$$

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

$$\hat{\alpha} = \frac{\hat{\mu}^2}{\hat{\sigma}^2}, \quad \hat{\beta} = \frac{\hat{\mu}}{\hat{\sigma}^2} \tag{4.3}$$

4.2 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 such as 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.

4.2.1 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, if the features were point-trajectories, a natural metric to discover segment boundaries would be space-time curvature [77]. Similarly, for shape features a reasonable metric would be shape deformation [137]. In this section, we describe a simple method for discovering action boundaries that works well for background subtracted silhouettes (and other image-like features).

During each action segment, the evolution of features is modeled using an affine motion model as is usually the case with traditional tracking algorithms. The crucial difference is that, we do not actually segment and track individual objects in the scene, but instead model the entire feature during a segment using the affine motion model.

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 evaluate whether it is consistent with the predictions of the learnt affine parameters. If so, we add the frame to the current segment. Otherwise, we detect the presence of a boundary. Learning the affine parameters for each segment can be achieved in closed-form using the properties of the fourier-transform [130] (FFT).

This segmentation scheme is suboptimal due to the assumption of affine motion. To overcome this we iterate back and forth between learning the LDS parameters for each segment and tweaking the segment boundaries till convergence is reached. Taking the output of the above scheme as an initial point, we learn the LDS parameters 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 LDS 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 3.4, 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:

$$\Delta(\tau) = \sum_{t=t_1}^{t_b+\tau} \left\| C_1(C_1^T f_t) - f_t \right\|^2 + \sum_{t=t_b+\tau}^{t_2} \left\| C_2(C_2^T f_t) - f_t \right\|^2 \tag{4.4}$$

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 $\arg\min_\tau \Delta(\tau) = 0$. We show some segmentation results on a near-field video sequence of an actor 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 4.3 from two different views.

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

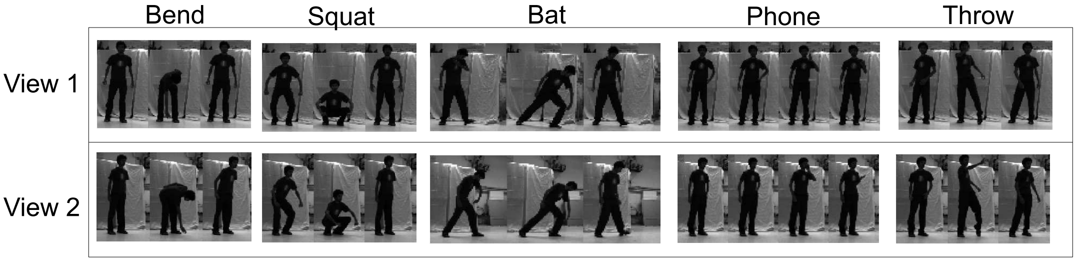
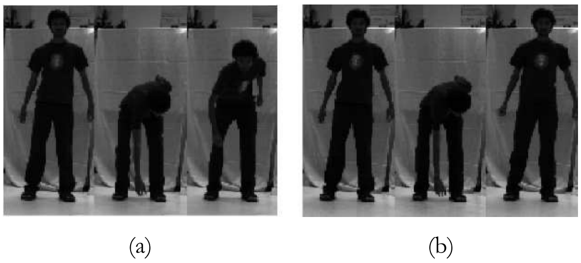


Figure 4.3: Sample segment boundaries for 5 activities. Note that the temporal segmentation algorithm finds a boundary whenever there is a change in the direction of motion. Notice that the segmentation results are consistent across view changes.

inadequate and we found that refinement of these boundaries using feedback significantly improved the results. We show one such example in figure 4.4. 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.



(a)

(b)

Figure 4.4: Bending boundaries (a) Before refinement, (b) After refinement

4.2.2 Relation with Switching Linear Dynamical Systems:

Learning the switching instants between LDS models is also encountered in SLDS. In SLDS, usually an extra hidden state is used to model switches. Any change in this hidden state corresponds to a switch between the LDS models such as in [62] and [63]. Usually, the number of states to switch amongst is assumed to be known (equal to the number of distinct actions), but we do not make any such assumption. An approach was presented in [66] for a special class of systems to estimate the number of states as well as to learn the dynamics of each system. In our experiments, we found that our algorithm for segmentation works reasonably well with a far smaller computational burden.

4.2.3 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 using the method described in section 3.3. Even though a long video might consist of several segments, not all of them will be distinct. We need to cluster these segments (figures 4.1 (e), (f)) to discover the distinct action elements (words). In order to perform this clustering, we need a distance measure on the space of LDS models. We use subspace angles $(\theta_i, i = 1, 2, ...n)$ between two ARMA models which are defined in [60] as discussed in section 3.7.

We use the Frobenius distance in all the results shown in this chapter. 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 the 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, such as k-means etc. The other popular alternative for clustering rely on graph-theoretic methods such as Normalized cuts ([138]). The advantage offered by these approaches is that they do 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 also provide principled means for estimating the number of clusters. A well known result regarding the Laplacian of a graph is briefly summarized as follows.

Let $G = (V, E)$ be an undirected and unweighted graph with vertex set V of cardinality n and edge set E of cardinality m. The existence of an edge between two vertices v_i and v_j is denoted as $(v_i, v_j) \in E(G)$. Let d_j denote the degree of vertex v_j. Let A be the $n \times n$ adjacency matrix of the graph such that $A_{ij} = 1$ if and only if $(v_i, v_j) \in E(G)$. Let D be the diagonal matrix with $D_{jj} = d_j$.

The Laplacian of the graph, L is defined as

$$L = D - A \tag{4.5}$$

and the normalized Laplacian, L_{norm} is defined as

$$L_{norm} = D^{-1/2} L D^{-1/2} \tag{4.6}$$

A connected graph is a graph such that there exists a path between all pairs of vertices. A *connected component* is a maximal subset of the graph that forms a connected graph. The following is a well-known result [139] that relates the number of connected components of a graph to eigenvalues of its Laplacian.

Result: If G is a graph and L its Laplacian as defined above, then the multiplicity of 0 as an eigenvalue of L is equal to the number of connected components of G ([139]).

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 represent 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 similar result as above. A synthetic example is shown in figure 4.5 for the case of two clusters. We generated scalar data from two Gaussian densities with different means and large variances such that there is significant overlap in the pdfs. This overlap is reflected in the similarity matrix as well in figure 4.5(a). The eigenvalues of the normalized Laplacian are shown in figure 4.5(b). The 'elbow' is observed at 2 as shown circled.

Once we have estimated the number of clusters, we can generate the clusters using any standard graph clustering algorithm. We have used normalized cuts in our experiments [138]. Let the K cluster centers thus obtained be given by $C_1, C_2, C_3, \ldots C_K$. The segmented video is then given by a sequence of these labels.

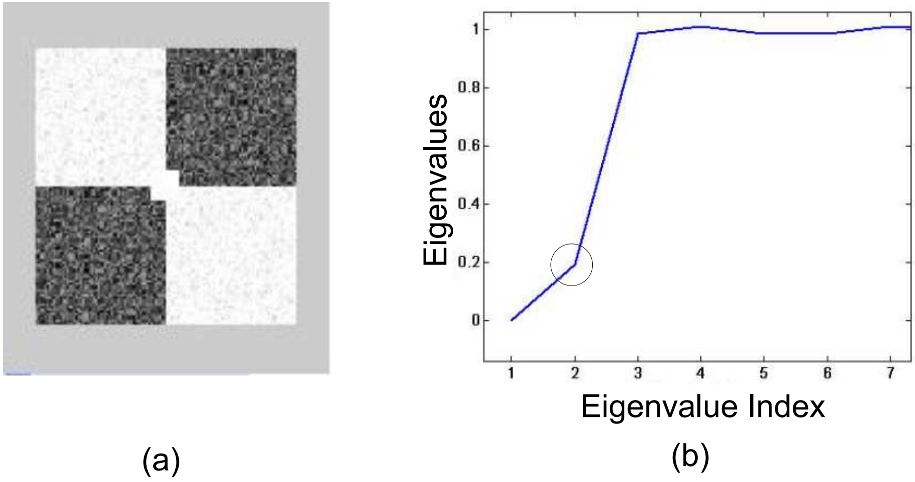


Figure 4.5: Illustrative example for estimating the number of clusters using heuristics based on the eigenvalues of the Laplacian of the similarity matrix (a) Similarity Matrix, (b) Eigenvalues of the normalized Laplacian. The location of the elbow (shown circled) represents the estimate of the number of clusters.

4.2.4 Discovering the Cascade Structure

After clustering the action elements each segment is assigned a label. Suppose we have the following sequence of labels $(C_1, C_3, C_2, C_6, C_7, C_8, C_1, C_3, C_5, C_2, C_6, C_1, C_7, C_8)$. 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 (C_1, C_3, C_2) and (C_6, C_7, C_8) are the repeating patterns, up to one insertion error. To discover the repeating patterns, we build n-gram statistics of the segment labels as shown in figure 4.1 (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 4.1.

4.3 Sequence of Dynamical models for Activity based Video Mining

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. **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.

3. **Torino 2006 figure skating data:** We have used figure skating video from the 2006 Winter Olympics at Torino. 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.

4.3.1 Experiments on UMD Dataset [1]

In the experiment described in section 4.2.1, 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 oversegmentation effects, we merge very similar motifs. Since, a motif is a string of labels, we used the Levenshtein distance [140] as the metric to merge them. The classification of the activities into motifs is tabulated in Table 4.1. 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 among various activities by learning the right cascade structure of the action prototypes.

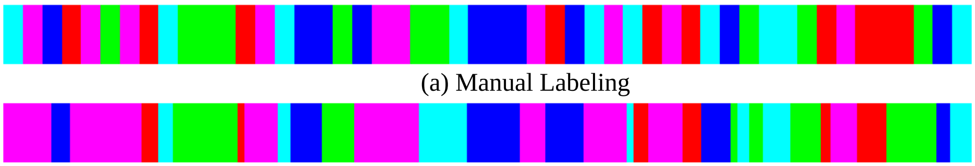
Figure 4.6 shows activity labels for the entire video sequence extracted manually and automatically. Matching of the colors in the figure indicates that the algorithm is able to discover and identify activities in an unsupervised manner. We found that the errors in labeling are typically near the transition 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 4.7(a) and dimensions 4-6 in figure 4.7(b). We see that the trajectories of the same activity are closely clustered together in the Laplacian-space.

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 4.1: Composition of the Discovered Clusters in the UMD database

4.3.2 INRIA - Free-Viewpoint Database [2]

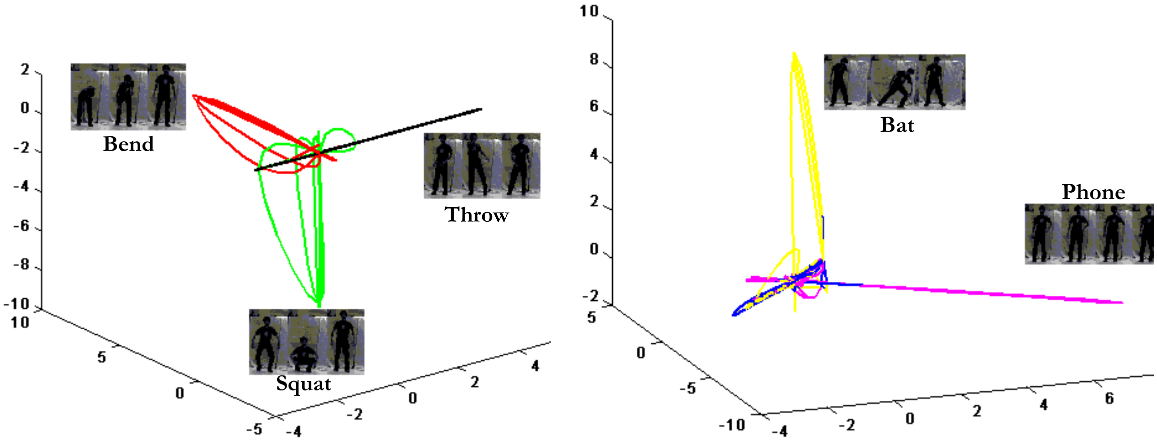
The INRIA multiple-camera multiple video database of the PERCEPTION group consists of 11 daily-live motions performed each 3 times by 10 actors. The actors freely change position and orientation. Every execution of the activity is done at a different rate.



(a) Manual Labeling

(b) Automatically Discovered Labels (unsupervised−clustering)

Figure 4.6: 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)

Figure 4.7: (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.

For this dataset, we extract $16 \times 16 \times 16$ circular FFT features as described in [2]. Instead of modeling each segment of activity as a single motion history volume as in [2], 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 [141].

We performed a clustering experiment on all 30 sequences (10 actors $\times 3$ sequences per actor). Segmentation was performed using the method described in [141]. The clustering results are shown in Table 4.2. 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 maybe attributed to the similar and also sparse motion patterns that are generated by those activities.

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 4.2: Confusion matrix showing view-invariant clustering using the proposed algorithm on the INRIA dataset.

We also show the actual summarization results obtained on two of the actors – ‘Florian’ and ‘Alba’ in figures 4.8 and 4.9.

4.3.3 Torino 2006 Figure Skating data

We performed a clustering and retrieval experiment on the Torino 2006 Winter Olympics figure skating videos. This data is very challenging since it is unconstrained and involves rapid motion of both the actor (skater) and real-world motion of the camera including pan, tilt and zoom. Some representative frames from the raw video are shown in figure 4.10.

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. We perform median filtering followed by connected com-

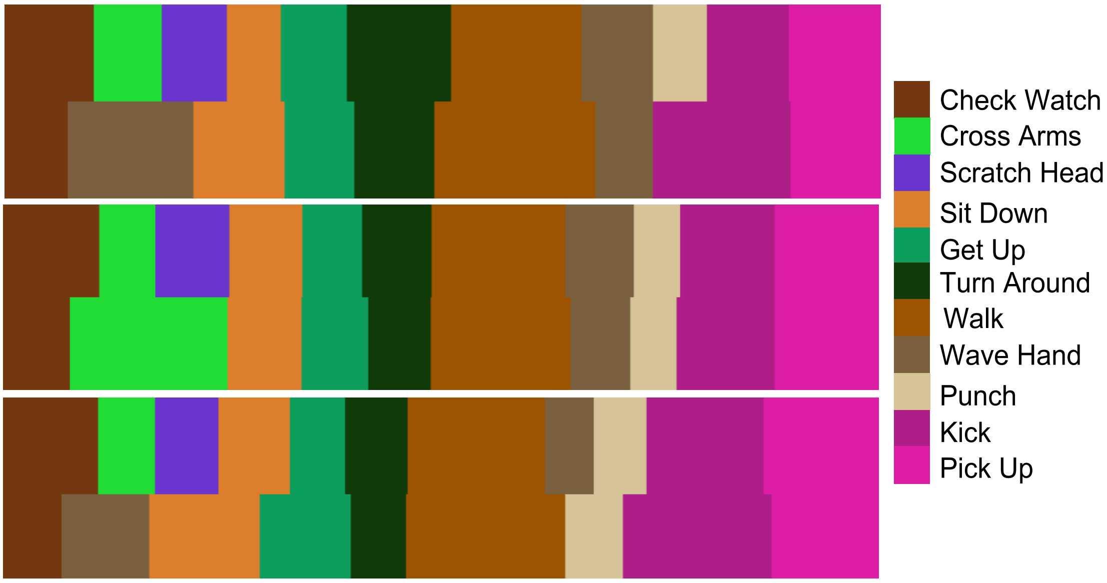


Figure 4.8: Color coded activity labeling for three sequences by actor 'Florian'. First row in each is the groundtruth, second row is the discovered labeling. Image best viewed in color.

ponent analysis 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 smoothing 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: In a setting such as figure skating, 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, we build models for fixed length subsequences using sliding windows. 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. To discover these 'interesting' activities, we apply a two-stage clustering algorithm. 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 4.11 – 4.15. We observe that Clusters 1 - 4 correspond dominantly to 'Sitting Spins', 'Standing

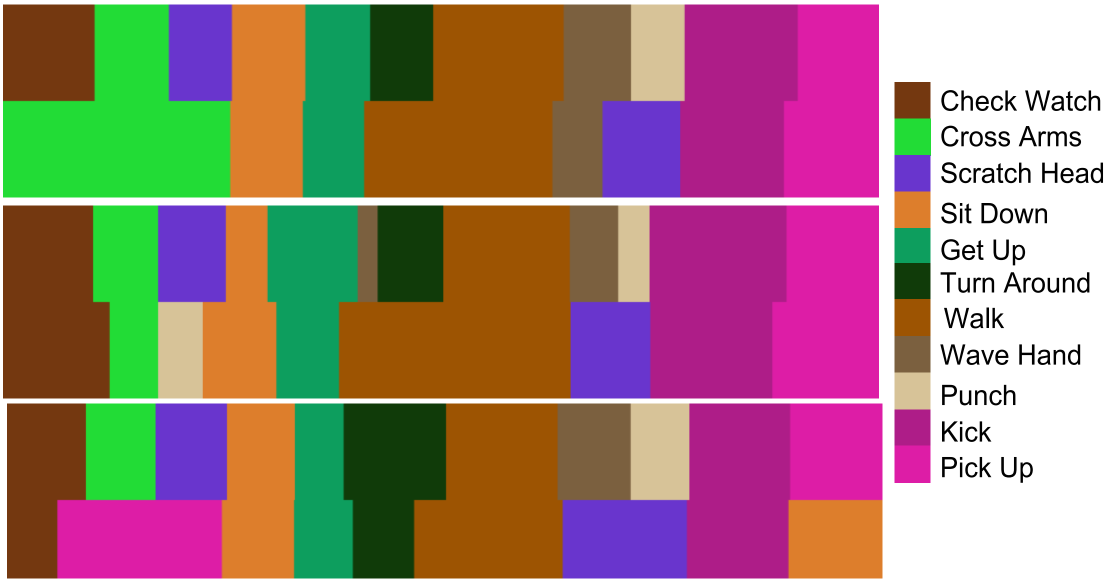


Figure 4.9: Color coded activity labeling for three sequences by actor ‘Alba’. First row in each is the groundtruth, second row is the discovered labeling. Image best viewed in color.

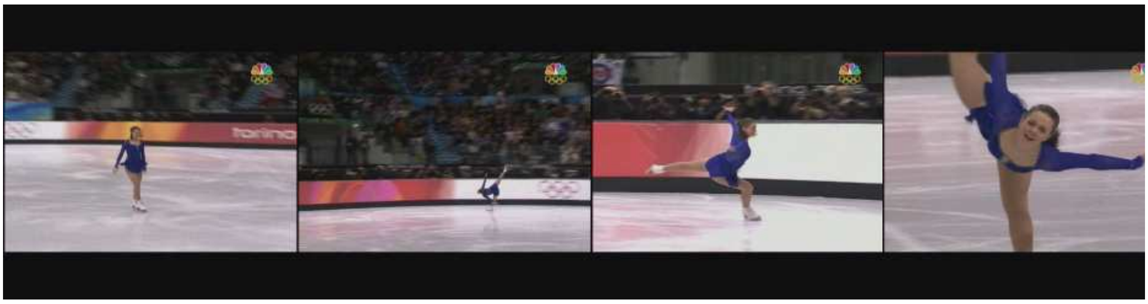


Figure 4.10: Sample images from the skating video of Emily Hughes of USA.

Spins’, ‘Leaping 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.

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 Leap spin and standing spin are shown in figures 4.16 - 4.17.

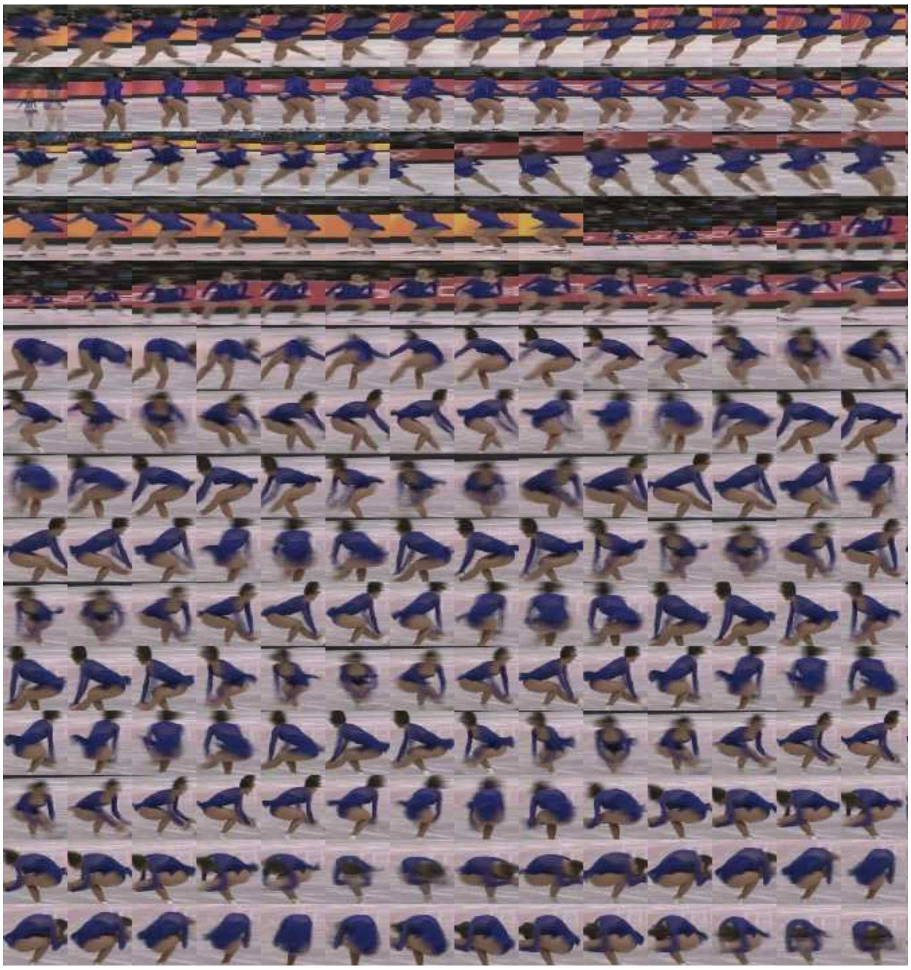


Figure 4.11: Shown above are a few sequences from Cluster1. Each row shows contiguous frames of a sequence. We see that this cluster dominantly corresponds to ‘Sitting Spins’.

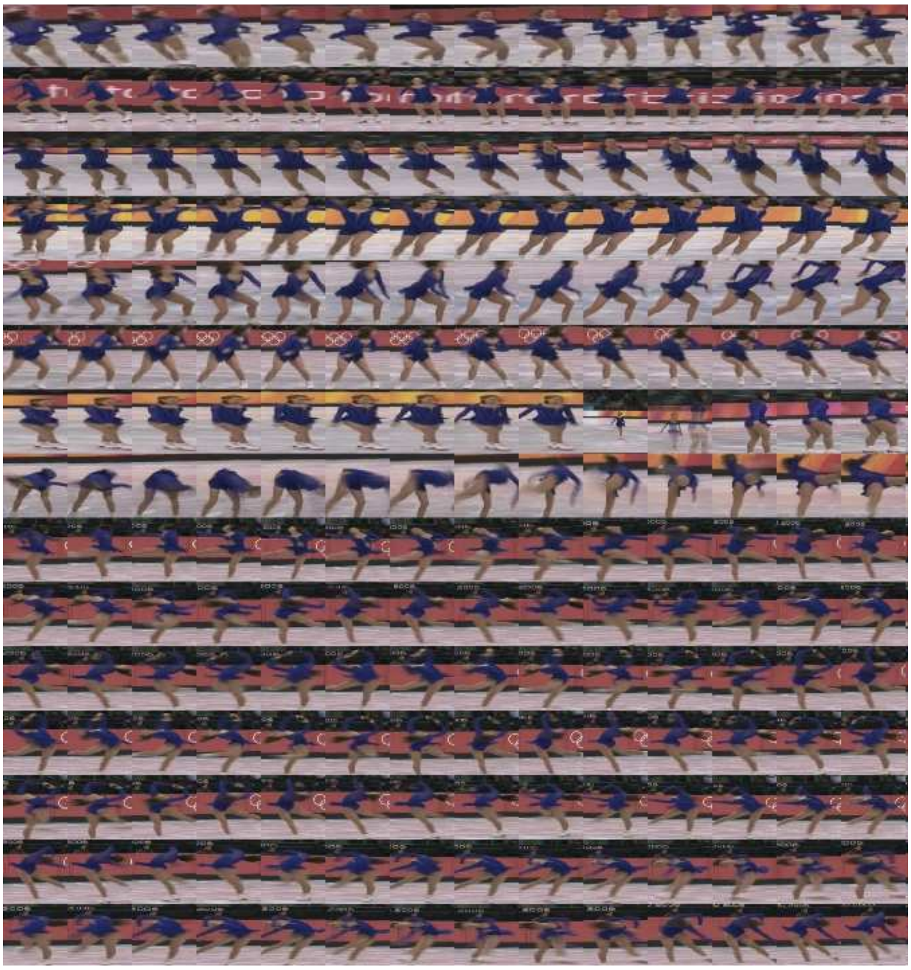


Figure 4.12: Shown above are a few sequences from Cluster2. Each row shows contiguous frames of a sequence. Notice that this cluster dominantly corresponds to ‘Standing Spins’.

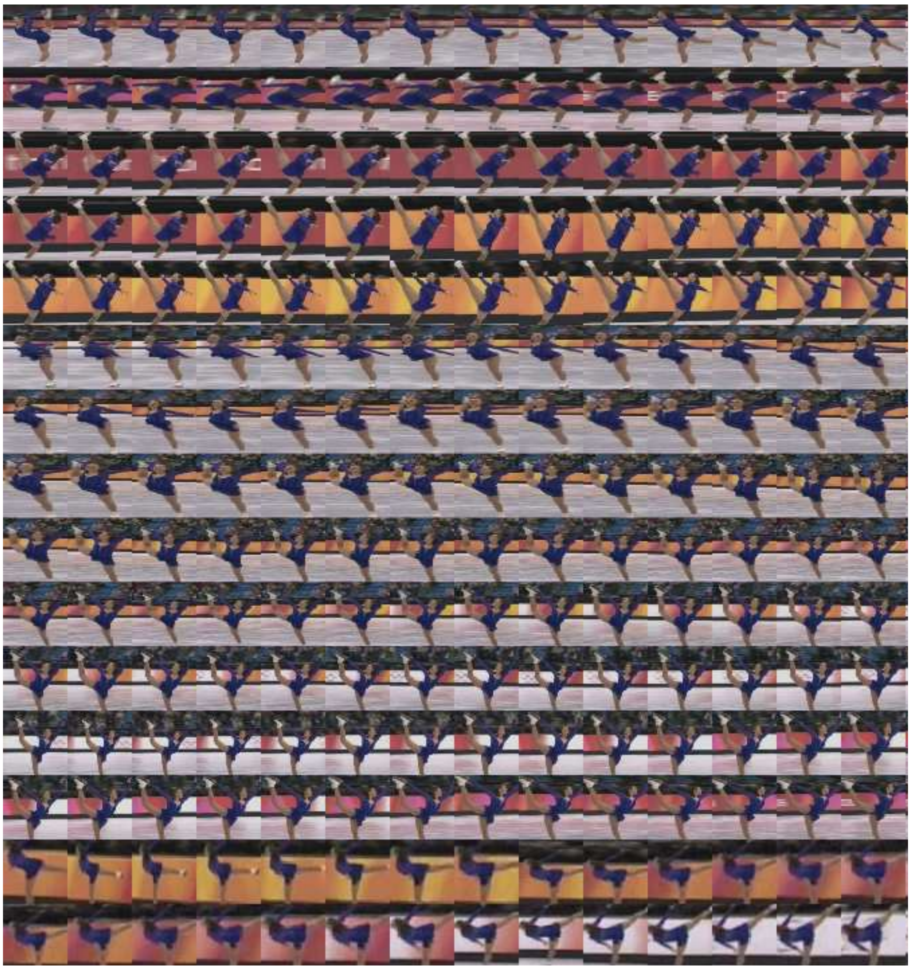


Figure 4.13: Shown above are a few sequences from Cluster3. Each row shows contiguous frames of a sequence. Notice that this cluster dominantly corresponds to ‘Spirals’.

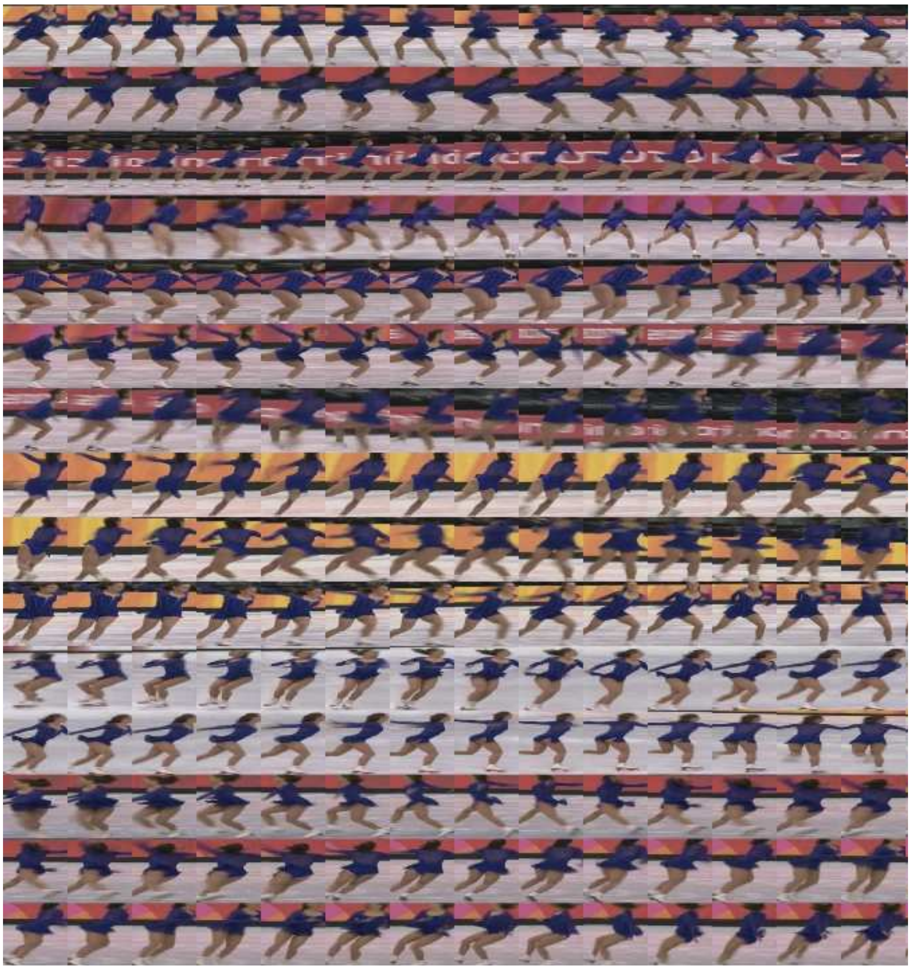


Figure 4.14: Shown above are a few sequences from Cluster4. Each row shows contiguous frames of a sequence. This cluster dominantly corresponds to ‘Leap Spins’.

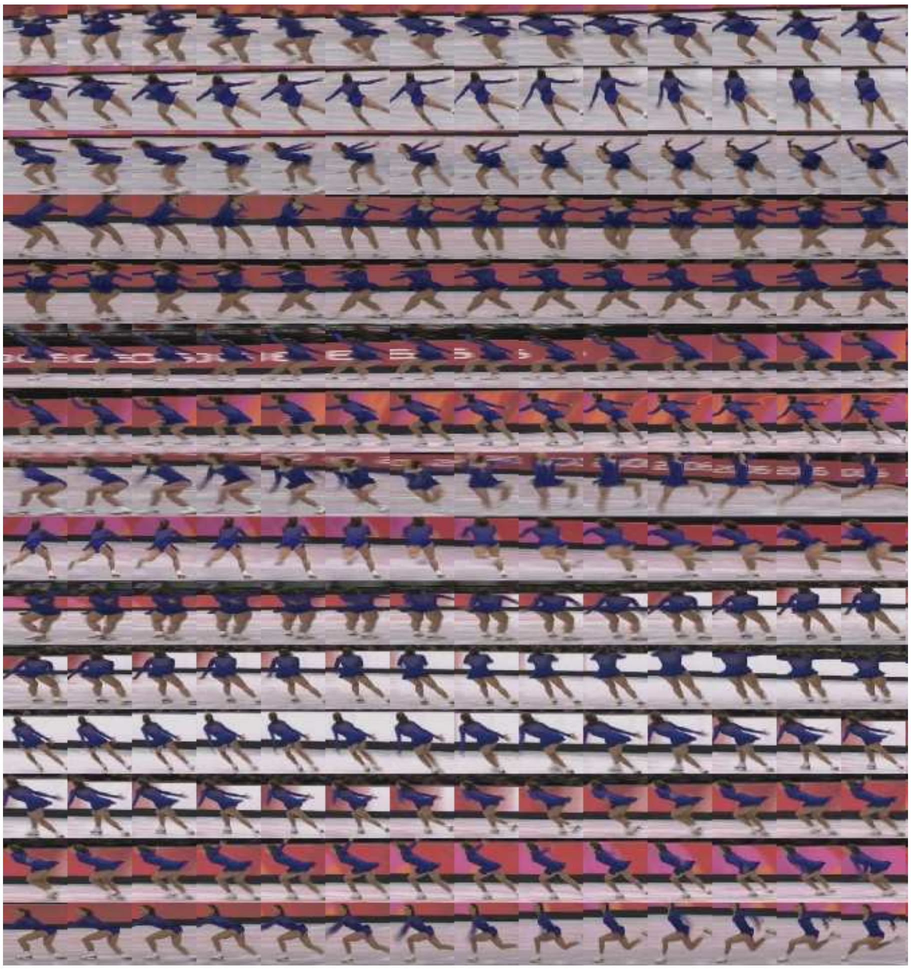


Figure 4.15: Shown above are a few sequences from Cluster5. Each row shows contiguous frames of a sequence. This cluster did not dominantly correspond to any ‘interesting’ skating pose but seemed to capture the ‘usual’ postures. Image best viewed in color.

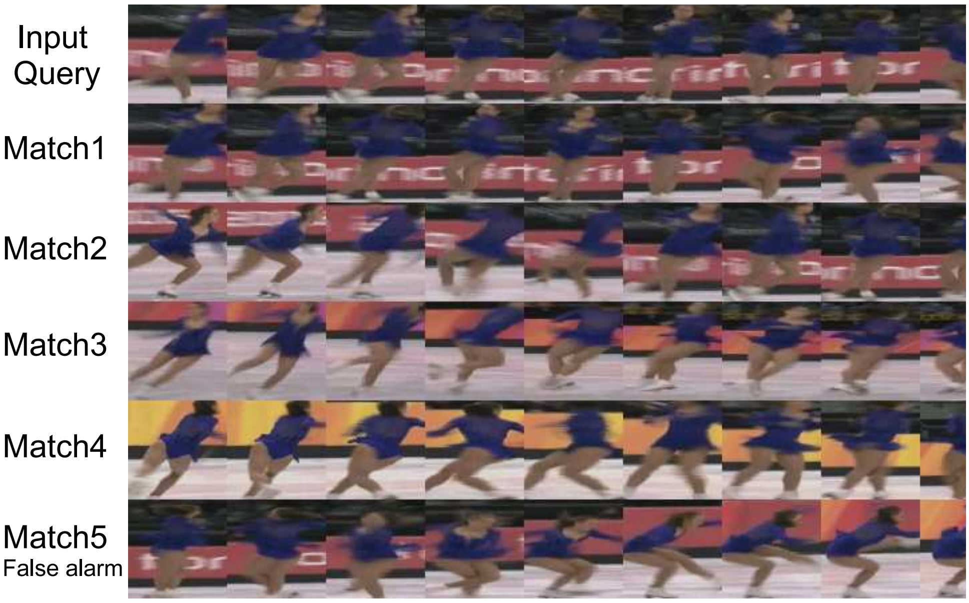


Figure 4.16: Shown above is the input query corresponding to a Leap Spin and the top 5 matches obtained. The last match is a false match. Image best viewed in color.

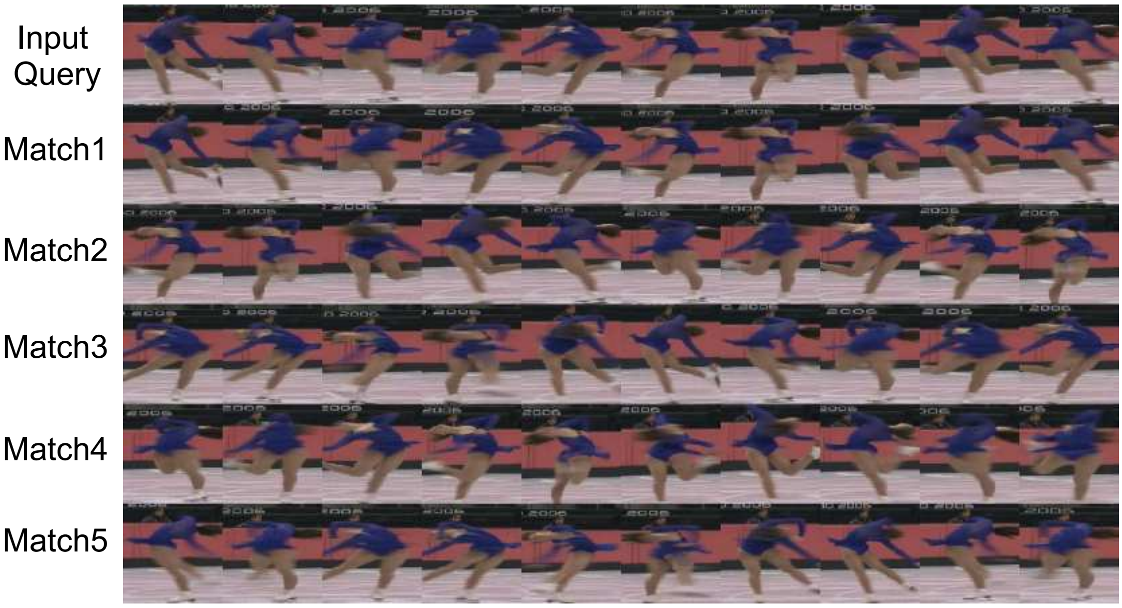


Figure 4.17: Shown above is the input query corresponding to a Standing Spin and the top 5 matches obtained. All the matches correspond to standing spins. Image best viewed in color.

Chapter 5

Temporal Modeling: Time Varying Models

In several domains, it has been observed that human activities are better described as a continuum of actions where the individual boundaries between actions are often blurry [142]. To draw a parallel to language processing, it has been long known in the speech community that words spoken in isolation sound quite different when spoken in continuous speech. This is commonly attributed to 'co-articulation' and 'assimilation' effects. Similarly, when actions appear in a connected form, it is hard to identify precisely where an action ends and where another begins. Consider the action shown in figure 5.1 (a) and a synthesized version which relies on finding segment boundaries and fitting models to each segment in figure 5.1 (b). As can be seen, segmentation followed by modeling causes abrupt changes to appear at segment boundaries during synthesis. This effect is also observed in sign-language where gestures are influenced by adjacent gestures [142], making segmentation and recognition difficult.

Activities may also be viewed from a stochastic process point of view. In this context, 'stationarity' or 'non-stationarity' is an important property of the stochastic process under consideration. Stationarity requires that the ensemble statistics of the process do not change with time. On the other hand, 'time-invariant' and 'time-varying' refer to the properties of the model used to describe a given stochastic process. A good discussion of the relation between stationary processes and time-invariant models is given in [143]. A

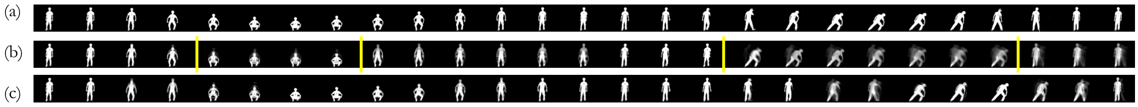


Figure 5.1: (a) Original sequence taken from the common activities dataset [1], (b) Synthesis by a sequence of linear dynamic models with boundaries shown by vertical yellow lines, (c) Synthesis by a continuous time-varying model. It can be seen that when actions are segmented and modeled using switching models, the synthesis results show abrupt changes in pose across boundaries whereas the time-varying model results in a much more natural evolution of poses.

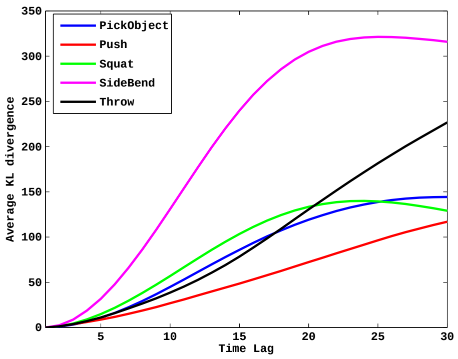


Figure 5.2: Illustration of how statistical properties change with time for 5 activities. The y-axis measures the KL divergence between ensemble statistics as a function of the time-lag. Figure best viewed in color.

key observation is that if a process is stationary, it can be well described by time-invariant models such as the Gauss-Markov model [123]. Now one might ask the question whether activities are stationary or non-stationary. Consider the common activities dataset of [1]. Each activity in the dataset contains 10 executions from 2 views. Considering each execution to be a realization of a random process $X(t)$, we compute the pdf of the random variable at each time instant $f_X(t)$, by fitting a parametric Gaussian estimated from the ensemble. If the activity is indeed stationary, then the pdf's at time-instants t and $t+\delta$ would be identical. We will answer the question using empirical estimates of KL divergence.

We computed the KL-divergence between the pdfs as a function of the lag δ averaged over all time-instants i.e. $KL_{avg}(\delta) = \frac{1}{M}\sum_{t=0}^{M-1} KL(f_X(t), f_X(t+\delta))$. Figure 5.2 shows how KL_{avg} varies with δ for different activities. As is evident, the statistical properties of the activity vary smoothly but significantly over time even for these simple actions. This suggests that complex human activities cannot be considered stationary stochastic processes. Indeed, in this chapter, we consider human actions as quasi-stationary processes. To model such quasi-stationary processes, we notice that the plot in figure 5.2 reveals that we can assume local stationarity, since for small values of δ the statistical properties do not change significantly. Thus, it would suffice to fit locally time-invariant models, but allow the parameters of the model to vary with time. This observation forms the basis for the current work. Note that this approach is widely used in the speech processing community where speech signals are considered short-term stationary in windows of 20-40 milliseconds [144].

We consider human actions as a continuum of dynamical processes, where the parameters change continuously over time as opposed to discrete jumps in time. We represent the LDS at each time-instant as a point on the Grassmann manifold. Then, the overall activity is considered as a trajectory on the Grassmann manifold. Time-varying linear dynamical processes have also been studied in the control literature where they are traditionally used as approximations to non-linear processes [59]. Modeling of time-invariant dynamical systems as points on the Grassmann manifold was considered by [145]. Tracking points on the Grassmann manifold by a Hidden Markov Model on the manifold was proposed by [146] in array-signal processing applications, where a constant velocity model is assumed on the manifold. In contrast to the generative approaches discussed above, there exist discriminative approaches for modeling human actions. An in-depth discussion of discriminative models is beyond the scope of this chapter, and we refer the reader to [147, 148] and references therein.

5.1 Modeling of Complex Activities

An activity is considered as a complex evolution of poses which is governed by an underlying dynamic process. The underlying process is potentially highly non-linear and time-varying. We model complex activities as outputs of a time-varying linear dynamical process. At each time-instant, we assume that the dynamical process is linear. We then allow the parameters of the LDS to vary at each time-instant. Let $f(t) \in \mathbb{R}^m$ denote the observations (flow/silhouette etc) at time-instant t. Then, the time-varying dynamical model is represented as

$$f(t) = C(t)z(t) + w(t), w(t) \sim N(0, R(t)) \tag{5.1}$$

$$z(t+1) = A(t)z(t) + v(t), v(t) \sim N(0, Q(t)) \tag{5.2}$$

where, $z(t) \in \mathbb{R}^d$ is the hidden state vector of dimension d, $A(t)$ is the time-varying transition matrix and $C(t)$ is the time-varying measurement matrix. $w(t)$ and $v(t)$ are noise components modeled as normal with 0 mean and covariance $R(t)$ and $Q(t)$ respectively. When the model parameters A, C, Q, R are constant, the model reduces to the well-known time-invariant LDS which has been successfully applied in several vision tasks [149, 123]. In summary, the model consists of a sequence of parameters: the measurement matrix $C(t)$ and the transition matrix $A(t)$ and the noise covariances $R(t), Q(t)$. Before we discuss the problem of parameter estimation, we show the strength of the model on the synthesis experiment described earlier. The results of synthesis using a continuous time-varying model are shown in figure 5.1(c). It can be seen that the synthesized sequence exhibits a much more realistic evolution of poses.

5.1.1 Estimating the parameters

We first present a brief review of the parameter estimation problem for the time-invariant case before turning to the time-varying case.

The time-invariant case: Consider the time-invariant version of the model in equations (5.1) and (5.2).

$$f(t) = Cz(t) + w(t), w(t) \sim N(0,R) \tag{5.3}$$

$$z(t+1) = Az(t) + v(t), v(t) \sim N(0,Q) \tag{5.4}$$

For the time-invariant case, it is easily shown that there are infinitely many choices of parameters that give rise to the same sample path $f(t)$. Resolving this ambiguity requires one to impose further constraints and choose a canonical model. The conditions as proposed in [123] are that $m >> d, rank(C) = d$ and $C^T C = I$. The number of unknowns that need to be solved for are: $md - \frac{d(d+1)}{2}$ for C, d^2 for A, $\frac{d(d+1)}{2}$ for Q: resulting in $md + d^2$ unknowns (we have ignored the observation noise covariance as of now). For each observed frame we get m equations. Hence, $d+1$ linearly independent observations are sufficient to solve for the required parameters ($m(d+1) > md + d^2$ since $m >> d$).

The parameter estimates can be obtained in closed form using prediction error methods. Several estimation algorithms exist such as the ones described in [57] and [123]. We use the solution derived in [123] here. Let observations $f(1), f(2), \dots f(\tau)$, represent the features for the frames $1, 2, ...\tau$. Let $[f(1), f(2), \ldots f(\tau)] = U\Sigma V^T$ be the singular value decomposition of the data. Then $\hat{C} = U, \hat{A} = \Sigma V^T D_1 V (V^T D_2 V)^{-1} \Sigma^{-1}$, where D_1 = [0 0;$I_{\tau-1}$ 0] and D_2 = [$I_{\tau-1}$ 0;0 0].

The time-varying case: Estimation of time-varying models for time-series has been studied in various domains such as speech processing, econometric data and communication channels. A commonly used assumption in these domains is that the time-varying AR (auto-regressive) and ARMA (auto-regressive moving average) parameters can be expressed as linear combinations of known deterministic functions of time such as the Fourier basis or the exponential basis [144]. Other approaches include Taylor-series expansions of the model parameters such as in [150] for econometric applications. Estimation of time-varying single-input single-output (SISO) AR models has been proposed by estimating an equivalent time-invariant single-input multiple-output (SIMO) process [151], and was applied for channel estimation in communication networks. These approaches are restricted to single-dimensional time-series data. Multi-dimensional time-varying dynamical models traditionally arise as a result of linearizing a non-linear dynamical system. In such cases, the time-varying parameters can be solved for analytically using Taylor series expansions around a ‘nominal trajectory’ [59]. However, in most practical applications including activity modeling, one does not know what the underlying non-linear equations are nor does one have the knowledge of a nominal trajectory. Recently, linear parameter varying (LPV) systems have been proposed to model time-varying processes. In these approaches, the time-varying model parameters are considered to be linear combinations of a small set of time-invariant parameters. The linear combination weights, also called the scheduling weights, change with time [152, 153]. However, identification of LPV systems is computationally very expensive [153]. In the following, we propose a computationally efficient and conceptually simple method to estimate the time-varying parameters of a dynamical system without making strong assumptions on the nature of the time-varying process.

To begin with, it is easily seen that even in the time-varying case there are infinitely many choices of the model parameters that can give rise to the same sample path $f(t)$. So, we impose the same set of conditions as in the time-invariant case i.e.

$m >> d, rank(C(t)) = d$ and $C(t)^T C(t) = I$. Based on the analysis given above, there are $md + d^2$ unknowns for *each* time-instant and m equations per time-instant. Obviously this is an ill-posed problem since there are far more unknowns than there are equations. Hence, we impose another condition that the model parameters stay constant in local temporal neighborhoods. The temporal neighborhood in which the parameters are assumed to stay constant should also ensure that $d+1$ linearly independent observations can be obtained within the neighborhood. In general, it cannot be guaranteed that a fixed $d+1$ sized neighborhood will satisfy this condition. However, in our experience we found that a neighborhood of size $1.5d - 2d$ was sufficient to meet this condition in most real-world human activities. Typically, d is of the order of $5 - 10$ and complex human activities extend to several hundred frames. It is reasonable to assume that in short windows of about $15 - 20$ frames the dynamics can be easily modeled by simple time-invariant dynamical processes.

We now have a sequence of dynamical systems which defines a trajectory on the space of LDS. Before we discuss how we model this trajectory, we first discuss the Grassmann manifold formulation of the LDS space.

5.2 Trajectories on the Model Space

For the time-invariant case, starting from an initial condition $z(0)$, it can be shown that the *expected* observation sequence is given by

$$E\begin{bmatrix} f(0) \\ f(1) \\ f(2) \\ . \\ . \end{bmatrix} = \begin{bmatrix} C \\ CA \\ CA^2 \\ . \\ . \end{bmatrix} z(0) = O_\infty(M)z(0) \tag{5.5}$$

Thus, the expected observation sequence generated by a time-invariant model $M = (A, C)$ lies in the column space S of the extended *observability* matrix given by $O_\infty(M) = [C^T, (CA)^T, (CA^2)^T, ...]^T$. In the time-varying case, we assumed that the model parameters stay constant in short temporal neighborhoods. Let the size of the temporal window be n.

Thus, the n-length expected observation sequence generated by the model $M_t = (C_t, A_t)$ (model at time t) lies in the column space S_t of the *finite* observability matrix given by

$$O_n(M_t) = \left[\ C_t; C_t A_t; \ldots; C_t A_t^{n-1}\ \right] \tag{5.6}$$

Thus, the time-varying model can be viewed as a sequence of subspaces S_t, where each subspace is spanned by the columns of the observability matrix at the corresponding time instant. Finite dimensional subspaces such as these can be identified as points on the Grassmann manifold [154]. Thus, the sequence of subspaces can be mathematically expressed as a trajectory on the Grassmann manifold.

5.3 Statistics and Geometry of the Grassmann manifold

To model and compare trajectories on the Grassmann manifold, we need to understand a) the representation of points, b) distance metrics and c) statistical models on the manifold. In this section, we provide a brief overview of each of these aspects. The Grassmann manifold $G_{m,k}$ is the space whose points are *k-planes* or k-dimensional hyperplanes (containing the origin) in $\mathbb{R}^m$. To each k-plane ν in $\mathbb{R}^m$, we can associate an $m \times k$ orthonormal matrix Y such that the columns of Y form an orthonormal basis for the plane. Note that there exist several choices for the basis Y. Thus, all the choices of basis vectors that span the same subspace need to be considered equivalent. To each k-plane ν in $G_{m,k}$ is associated an equivalence class of $m \times k$ matrices YR in $\mathbb{R}^{m \times k}$, for non-singular R, where Y is an orthonormal basis for the k-plane. This is also called the Procrustes representation. Alternately, one can define a unique projection matrix for the subspace given by $P = YY^T$ which projects points from the ambient Euclidean space onto the given subspace. In applications to human activities, the projection matrix representations leads to large computational overheads since it is a square $m \times m$ matrix. In practice, m is of the order of 10^3 or higher. Thus, we rely on the Procrustes representation of points which relies on storing only tall-thin $m \times k$ matrices.

A point X on $\mathscr{S}_{n,d}$ is represented as a tall-thin $n \times d$ orthonormal matrix. The corresponding equivalence class of $n \times d$ matrices XR in $R_{m,k}$, for $R \in SO(d)$ is also called the Procrustes representation of the Stiefel manifold. Thus, to compare two points in $\mathscr{G}_{n,d}$, we simply compare the smallest squared distance between the corresponding equivalence classes on the Stiefel manifold according to the Procrustes representation. Given matrices X_1 and X_2 on $\mathscr{S}_{n,d}$, the smallest squared Euclidean distance between any pair of matrices in the corresponding equivalence classes is given by

$$d^2_{Procrust}(X_1, X_2) = \min_R tr(X_1 - X_2R)^T(X_1 - X_2R) \tag{5.7}$$

$$= \min_R tr(R^TR - 2X_1^TX_2R + I_k) \tag{5.8}$$

When R varies over the orthogonal group $O(k)$, the minimum is attained at $R = H_1H_2^T = A(A^TA)^{-1/2}$, where $A = H_1DH_2^T$ is the singular value decomposition of A. We refer the reader to [154] for proofs and alternate cases.

Given several examples from a class $(X_1, X_2, \ldots, X_n)$ on the manifold $V_{k,m}$, the class conditional density can be estimated using an appropriate kernel function. We first assume that an appropriate choice of a divergence on the manifold has been made such as the one above. For the Procrustes measure the density estimate is given by [154] as

$$\hat{f}(X;M) = \frac{1}{n}C(M)\sum_{i=1}^{n} K[M^{-1/2}(I_k - X_i^TXX^TX_i)M^{-1/2}] \tag{5.9}$$

where $K(T)$ is the kernel function, M is a $k \times k$ positive definite matrix which plays the role of the kernel width or a smoothing parameter. $C(M)$ is a normalizing factor chosen so that the estimated density integrates to unity. The matrix valued kernel function $K(T)$ can be chosen in several ways. We have used $K(T) = exp(-tr(T))$ in all the experiments reported in this chapter. In this non-parametric method for density estimation, the choice of kernel width M becomes important. Thus, though this is a non-iterative procedure, the optimal choice of the kernel width can have a large impact on the final results. In general, there is no standard way to choose this parameter except for cross-validation.

5.4 Comparing sequences of Subspaces

Given a video of a long activity, first the time-varying model parameters $M_t = (A_t, C_t)$ are estimated using small temporal sliding-windows and the method described in section 5.1.1. Subsequently, for each window the observability matrix $O_n(M_t)$ is computed. Then for each observability matrix, an orthonormal basis is computed using standard SVD based algorithms. So, we now have a sequence of subspaces, or in other words a trajectory on the Grassmann manifold. To compare two subspace trajectories we propose two approaches.

Dynamic time warping: Dynamic time-warping (DTW) only requires an appropriate distance metric between points on the manifold. Given two complex activities and their corresponding subspace sequences $S_1(t)$ and $S_2(t)$, DTW tries to find a warping path $a(t)$ such that $S_1(t) = S_2(a(t))$. To solve the problem we can use any standard DTW algorithm.

Grassmann switching model: In the second approach, we parametrize the trajectory using a switching model akin to the HMM on the Grassmann manifold. Corresponding to an activity class C, suppose we are given M subspace sequences $\{S_i^C(t)\}_{i=1}^M$. We consider the dynamics to be described by a set of K hidden states $L^{(1)}, \ldots L^{(K)}$. The state at time t is denoted by $Q(t)$ and the observation at time t is denoted by $S(t)$. The overall model for the activity consists of the K hidden states, the intra-cluster pdfs $f(S(t)|Q(t) = L^{(i)})$, the transition probability matrix and the prior probability. In general, the Baum-Welch algorithm provides solutions for the above problems in a maximum likelihood sense. This requires one to have analytical expressions for the intra-cluster pdfs and the gradient of the likelihood of a sequence in terms of these parameters. In our case, we solve these problems in a much simpler, although sub-optimal way as follows. Given a sequence of subspaces $\{S_i^C(t)\}_{i=1}^M$, the following procedure is adopted to estimate the switching model.

1. Cluster the points into K clusters or hidden-states $L^{(1)}, \ldots L^{(K)}$.
2. Estimate a pdf within each cluster $f(S(t)|Q(t) = L^{(i)})$.
3. Estimate the transition probabilities $p(Q(t) = L^{(i)}|Q(t-1) = L^{(j)})$ between the clusters.

4. Estimate the prior probability $p(Q(0))$. Any of the distance metrics on the Grassmann manifold can be used to perform clustering. In our experiments, we used a spectral clustering algorithm – Normalized cuts – to get the clusters. Within each cluster, we use the non-parametric density estimate as described in chapter 6 to estimate the intra-cluster pdf. Once the clusters are found, we form the sequence of cluster labels corresponding to the sequence of subspaces. The sequence of labels is used to estimate the transition probabilities by bi-gram counts. Thus, we have now learnt a switching model on the Grassmann manifold for each activity class.

Given a new subspace sequence, we need a method to classify it into one of the action classes. In the case of standard HMMs, this problem is solved by the forward-backward algorithm and its variants. We use a simpler version that works much faster and using fewer computations. Given a sequence $S(t)$ and an activity model, we first assign each $S(t)$ into one of the clusters of the model. Let us denote by $Q(t)$ the sequence of cluster labels thus obtained. Then we compute the likelihood of the sequence as $p(Q(0))\prod_k f(S(k)|Q(k))p(Q(k)|Q(k-1))$. Though this is sub-optimal than the forward-backward algorithm, we found that we obtain significant computational advantages using these approximations.

Relation to Switching Linear Dynamical Systems: SLDS [118, 115, 155, 62] model a complex activity by breaking it down into simpler motion patterns where each motion pattern is modeled using a simple model such as an HMM or an LDS. The overall activity is then modeled by switching amongst a small set of dynamical systems. In the above Grassmann switching model, if we constrain the intra-cluster pdf to be $f(S(t)|Q(t) = L^{(k)}) = \delta(S(t) - \mu_k)$, where μ_k is the cluster center, then the Grassmann switching model reduces to the SLDS model. Thus, the SLDS model is a special case of the proposed Grassmann switching model. Further, in SLDS it is usually assumed that complex human actions can be separated into simpler motion patterns. However, we do not rely on segmentation of activities into primitive actions and thus our approach is applicable even in complex cases when segmentation is difficult.

5.5 Experiments

In the first experiment we performed a synthesis experiment on a skating dataset obtained from [6]. From a segment of video of about a 100 frames that contained fast skating actions as shown in figure 5.3 (a), a discrete-switching model and a time-varying model were estimated. The actions in the sequence exhibit co-articulation effects, where transitions between distinct poses contain intermediate poses that share the appearance of both the starting and the ending pose. The results of synthesis using the models are shown in figure 5.3. The experiment shows that the time-varying model can account for such co-articulatory effects and produce realistic looking sequences.

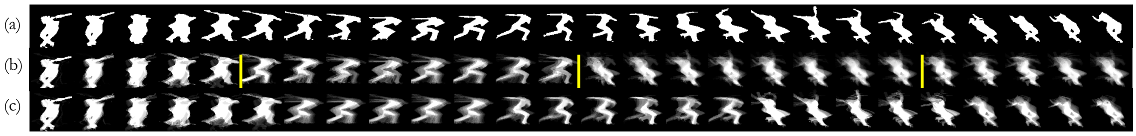


Figure 5.3: (a) Original skating sequence taken from [6], (b) Synthesis by a sequence of linear dynamic models with boundaries shown by vertical yellow lines, (c) Synthesis by a continuous time-varying model. It can be seen that synthesis results show abrupt changes in pose across boundaries whereas the time-varying model results in a smoother evolution of poses.

Next we present experiments demonstrating the strength of the model for summarizing and recognizing complex activities. In the first experiment we show the results of summarizing a long video containing a complex activity – the game of Blackjack. For this, we used the dataset reported in [7].

5.5.1 Blackjack Game Summarization

The game of Blackjack consists of a few elements such as dealing cards, waiting for bids, shuffling the cards etc. We try to estimate a Grassmann switching model for the entire video of Blackjack. The Grassmann switching model would then represent a 'summary' of the game, where the clusters of the model represent various elements of the game and the switching structure represents how the game progresses. This video consists of about 1700 frames. We extracted the motion-histogram features as proposed in [7] for each frame of the video. The time-varying model parameters are estimated in sliding

Figure 5.4: A few sample frames from the Blackjack dataset of [7].

windows of size 10. The dimension of the state vector is chosen to be $d = 5$. To estimate the Grassmann switching model for the game of Blackjack, we manually set the number of clusters to 5. In figure 5.5, we show an embedding of the video obtained from the model parameters using Laplacian eigenmaps. Each point corresponds to a time-invariant model parameter (A, C) pair or equivalently a point on the Grassmann manifold. Each cluster was found to correspond dominantly to a distinct element of the game as shown. The switching structure between the clusters is encoded in the transition matrix and is shown in figure 5.6. As can be seen the switching structure corresponds to a normal game of Blackjack. Since this is a data-driven procedure, it should be noted that the switching structure will not necessarily be the same for every individual Blackjack game. However, given two distinct Blackjack games we can now quantify the notion of how similarly the two games proceeded.

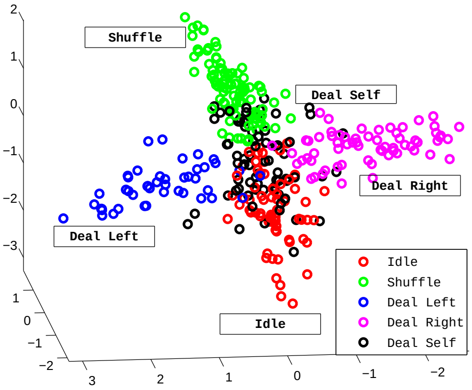


Figure 5.5: An embedding of the entire Blackjack video sequence. Figure best viewed in color.

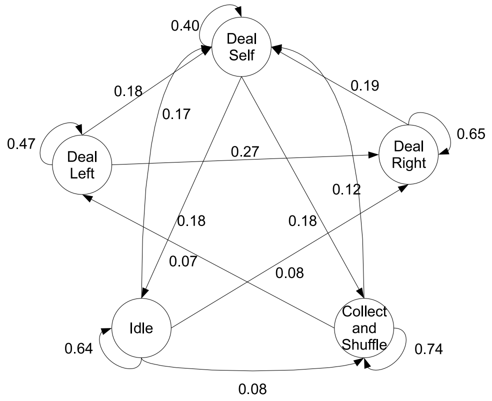


Figure 5.6: Estimated structure of the game of Blackjack. (For the sake of clarity arcs with low weights have not been shown).

5.5.2 Complex Activity Recognition

In the next experiment, we took the common activities dataset described in [1] consisting of 10 simple actions – $\{$Pick Object, Jog, Push, Squat, Wave, Kick, Side Bend, Throw, Turn around, Talk on cellphone$\}$. Each action is performed 10 times each by the same actor under two different viewing angles separated by about 20°. We create more complex actions from this set. We divided the actions into two groups - the first group contains the first 5 actions, the second group contains the next 5 actions. Then, we created compound actions by taking one action from the first group and an action from the second group. Then, we swapped the two constituent actions. This causes the two resulting compound actions to share similar global second-order statistics (the mean and covariance). Thus, we have 10 compound actions as shown in table 5.1. To test the framework, we performed a leave-one-out testing where we trained on 9 executions and tested on the remaining execution. Both views were used in training as well as testing. Since the global second order-statistics of activities such as PickObject-Kick and Kick-PickObject etc are similar, time-invariant linear dynamic systems are expected to show confusion between them. The results of the recognition experiment are shown in table 5.1. As is evident, both the DTW based and the Switching model show 100% recognition since they account for the time-varying dynamics of the compound actions.

Activity Type	LDS	Grass. DTW	Grass. Switching model
PickObject - Kick	100	100	100
Kick - PickObject	50	100	100
Jog - SideBend	100	100	100
SideBend - Jog	50	100	100
Push - Throw	0	100	100
Throw - Push	100	100	100
Squat - TurnAround	100	100	100
TurnAround - Squat	0	100	100
Wave - TalkCellphone	50	100	100
TalkCellphone - Wave	50	100	100
Average	60%	100%	100%

Table 5.1: Recognition percentages on Compound actions

Chapter 6

Detailed analysis of the Geometry of the Primitive Space

Let us now consider the ARMA model in more detail and try to understand the space of the model parameters. The ARMA model equations are given by

$$f(t) = Cz(t) + w(t) \quad w(t) \sim N(0,R) \tag{6.1}$$

$$z(t+1) = Az(t) + v(t) \quad v(t) \sim N(0,Q) \tag{6.2}$$

where, z is the hidden state vector, A the transition matrix and C the measurement matrix. f represents the observed features while w and v are noise components modeled as normal with 0 mean and covariance R and Q respectively. For high-dimensional time-series data (dynamic textures etc), the most common approach is to first learn a lower-dimensional embedding of the observations via PCA, and learn temporal dynamics in the lower-dimensional space.

The model parameters (A,C) do not lie in a vector space. The transition matrix A is only constrained to be stable with eigenvalues inside the unit circle. For the time-invariant ARMA case, starting from an initial condition $z(0)$, it can be shown that the *expected* observation sequence is given by

$$E\begin{bmatrix} f(0) \\ f(1) \\ f(2) \\ . \\ . \end{bmatrix} = \begin{bmatrix} C \\ CA \\ CA^2 \\ . \\ . \end{bmatrix} z(0) = O_\infty(M)z(0) \tag{6.3}$$

Thus, the expected observation sequence generated by a time-invariant model $M = (A,C)$ lies in the column space S of the extended *observability* matrix given by $O_\infty(M) = [C^T, (CA)^T, (CA^2)^T, ...]^T$. Thus, a linear dynamical system can be alternately identified as *subspace* corresponding to the column space of the observability matrix. In experimental implementations, we approximate the extended observability matrix by the finite observability matrix as is commonly done [149].

$$O_n^T = \left[C^T, (CA)^T, (CA^2)^T, \ldots (CA^{n-1})^T\right] \tag{6.4}$$

Finite dimensional subspaces such as these can be identified as points on the **Grassmann manifold**. We provide the definition of the Grassmann manifold next.

The Grassmann Manifold $\mathscr{G}_{n,d}$ [154]: The Grassmann manifold $\mathscr{G}_{n,d}$ is the space whose points are *d-planes* or d-dimensional hyperplanes (containing the origin) in $\mathbb{R}^n$.

On a related note, the Stiefel manifold is the space of d orthonormal vectors in $\mathbb{R}^n$.In the rest of the chapter, we review the geometry of the Grassmann manifold. This will then lead to appropriate distance metrics and statistical modeling methods on the Grassmann manifold. The set Grassmann manifold of d-dimensional subspaces of $\mathbb{R}^n$ will be denoted as $\mathscr{G}_{n,d}$. The set of all $n \times d$ orthonormal matrices shall be denoted as $\mathscr{S}_{n,d}$. On a computer, a linear subspace of $\mathbb{R}^n$ is stored as a tall-thin orthonormal matrix U such that the columns of U span the subspace. However, this choice of U is non-unique, there exist infinite choices of U that span the same subspace. We are interested in understanding the geometry of $\mathscr{G}_{n,d}$ and $\mathscr{S}_{n,d}$. The two underlying spaces – Stiefel $\mathscr{S}_{n,d}$ and Grassmann $\mathscr{G}_{n,d}$ – associated with our application are nonlinear manifolds and any statistical analysis intrinsic to those spaces requires some tools from differential geometry.

Related Work: The geometric properties of general Riemannian manifolds forms the subject matter of differential geometry. A good introduction to it can be found in [156]. Statistical methods on manifolds have been studied for several years in the statistics community. Some of the landmark papers in this area include [157, 158, 159], however an exhaustive survey is beyond the scope of this chapter. The geometric properties of the Stiefel and Grassmann manifolds have received significant attention. A good introduction to the geometry of the Stiefel and Grassmann manifolds can be found in [160] who introduced these methods in the context of eigenvalue problems. These problems mainly involved optimization of cost functions with orthogonality constraints. Issues involved in algorithmic computations of the geometric operations in such problems was discussed in [161]. A compilation of research results on statistical analysis on the Stiefel and Grassmann manifolds can be found in [154].

In certain vision applications involving subspace constraints, the problems have been recast using the Grassmann manifold. Examples include, [162] who performed optimization over the Grassmann manifold for obtaining informative projections. The Grassmann manifold structure of the affine shape space is exploited in [9] to perform affine invariant clustering of shapes. [163] performs discriminative classification over subspaces for object recognition tasks by using Mercer kernels on the Grassmann manifold. Most of these methods do not fully exploit the Riemannian geometry of the Grassmann manifold, or are tuned to specific domains lacking generality. [146] exploited the geometry of the Grassmann manifold for subspace tracking in array signal processing applications. The methods that we present here form a comprehensive (not exhaustive) set of tools that draw upon the Riemannian geometry of the Grassmann manifold. Along with the mathematical formulation, we also present efficient algorithms to perform these computations. Riemannian manifolds have also been explored in the vision community in other contexts such as in [164, 165], where Euclidean mean shift clustering is extended to Riemannian manifolds. Theoretical foundations for manifolds based shape analysis were described in [166, 167]. Statistical learning of shape classes using non-linear shape manifolds was presented in [168] where statistics are learnt on the manifold's tangent space. manifold's tangent space.

Organization of the Chapter: In section 6.1, we discuss the notation and the special orthogonal group that will lay the foundation for deriving results for the Stiefel and Grassmann manifolds. In section 6.2, we discuss the Stiefel and Grassmann manifolds as quotients of the special orthogonal group. In section 6.3, we discuss statistical methods that follow from the quotient interpretation. In section 6.4, we discuss Procrustes methods and non-parametric density estimation on the Grassmann manifold.

6.1 Mathematical Preliminaries: Notation and Definitions

The two underlying spaces – Stiefel $\mathscr{S}_{n,d}$ and Grassmann $\mathscr{G}_{n,d}$ – associated with our applications are nonlinear manifolds and any statistical analysis intrinsic to those spaces requires some tools from differential geometry. Since learning and using such fundamental mathematical tools demands additional effort, we first motivate their need. We are interested in statistical inferences on these spaces, i.e. estimation and analysis of variables taking values in $\mathscr{S}_{n,d}$ and $\mathscr{G}_{n,d}$. Statistical inferences require probability models that are often based on simple statistics, such as means and covariances, learnt from the past data. Let U_1 $U_2, \ldots, U_k$ be some previously estimated points on $\mathscr{S}_{n,d}$ and we seek their sample mean, an average, for defining a probability model on $\mathscr{S}_{n,d}$. These U_is are tall, orthogonal matrices. It is easy to see that the Euclidean sample mean $\frac{1}{k}\sum_{i=1}^{k} U_i$ is not a valid operation, mainly because it is not a vector space. Similarly, many of the standard tools in estimation and modeling theory do not directly apply to such spaces but can be modified to account for their nonlinear geometry. This motivates the need to understand the geometry of $\mathscr{S}_{n,d}$ and $\mathscr{G}_{n,d}$, a task we will try in this section.

The spaces of interest – Stiefel and Grassmann – are often studied as quotient spaces of the special orthogonal group $SO(n)$. So we start by briefly introducing the special orthogonal group, followed by the notion of quotient spaces. Then we shall show how the Stiefel and Grassmann manifolds can be derived as quotient spaces of $SO(n)$.

6.1.1 The Special Orthogonal Group SO(n)

Let $GL(n)$ be the set of $n \times n$ nonsingular matrices; this set is called the *generalized linear group* because it is also a group with the group operation given by matrix multiplication. The set $GL(n)$ possesses some additional structure that makes it more interesting. It is a differentiable manifold. One consequence is that although it is not a vector space, it can be locally approximated as a vector space using smoothly varying Euclidean coordinates. This property is essential to understanding the task of modifying tools from standard Euclidean statistics to nonlinear manifolds. The dual properties of being a group and a differentiable manifold make it a *Lie group*. If we consider the subset of all orthogonal matrices, and further restricting to the ones with determinant $+1$, we obtain a subgroup $SO(n)$, called the *special orthogonal group*. It can be shown that this is a submanifold of $GL(n)$ and, therefore, also possesses a Lie group structure. Since it has n^2 elements and $n+n(n-1)/2$ constraints (unit length columns $\rightarrow$ n constraints and perpendicular columns $\rightarrow$ $n(n-1)/2$ constraints), it is an $n(n-1)/2$-dimensional Lie group.

To perform differential calculus on a manifold, one needs its tangent spaces. On one hand the elements of tangent spaces are velocities of differentiable curves lying on the manifold; on the other hand, they act as differential operators for functions on the manifold and lead to the definitions of the directional derivatives, gradients, optimal points, etc, all essential in optimization problems. For the $n \times n$ identity matrix I, the tangent space $T_I(SO(n))$ is given by ([156]):

$$T_I(SO(n)) = \{X \in \mathbb{R}^{n\times n} : X + X^T = 0\},$$

It is the set of all $n \times n$ skew-symmetric matrices. For an arbitrary point $O \in SO(n)$, the tangent space is obtained by a simple rotation of $T_I(SO(n))$:

$$T_O(SO(n)) = \{OX | X \in T_I(SO(n))\} \ .$$

Define an inner product for any $Y, Z \in T_O(SO(n))$ by $\langle Y, Z\rangle = trace(YZ^T)$, where *trace* denotes the sum of diagonal elements. With this metric $SO(n)$ becomes a Riemannian manifold.

Using the Riemannian structure, it becomes possible to define lengths of paths on a manifold. Let $\alpha : [0,1] \mapsto SO(n)$ be a parameterized path on $SO(n)$ that is differentiable everywhere on $[0,1]$. Then $\frac{d\alpha}{dt}$, the velocity vector at t, is an element of the tangent space $T_{\alpha(t)}(SO(n))$ and its length is defined to be $\sqrt{\langle \frac{d\alpha}{dt}, \frac{d\alpha}{dt}\rangle}$. The length of the path α is then given by:

$$L[\alpha] = \int_0^1 \sqrt{\left(\left\langle \frac{d\alpha(t)}{dt}, \frac{d\alpha(t)}{dt}\right\rangle\right)} dt \ . \tag{6.5}$$

For any two points $O_1, O_2 \in SO(n)$, one can define a distance between them as the infimum of the lengths of all smooth paths on $SO(n)$ which start at O_1 and end at O_2:

$$d(O_1, O_2) = \inf_{\{\alpha:[0,1]\mapsto SO(n)|\alpha(0)=O_1, \alpha(1)=O_2\}} L[\alpha] \ . \tag{6.6}$$

A path $\hat{\alpha}$ which achieves the above minimum, if it exists, is a **geodesic** between O_1 and O_2 on $SO(n)$. Geodesics on $SO(n)$ can be written explicitly using the matrix exponential.

For an $n \times n$ matrix A, define its matrix exponential $\exp(A)$ by:

$$\exp(A) = I + \frac{A}{1!} + \frac{A^2}{2!} + \frac{A^3}{3!} + \dots \tag{6.7}$$

We can see that given any skew-symmetric matric X, $\exp(X) \in SO(n)$. Now we can define geodesics on $SO(n)$ as follows: for any $O \in SO(n)$ and any skew-symmetric matrix X,

$$\alpha(t) \equiv O\exp(tX) \ ,$$

is the unique geodesic in $SO(n)$ passing through O with velocity vector OX at $t = 0$.

An important tool in statistics on a manifold is an exponential map. If M is a Riemannian manifold and $p \in M$, the **exponential map** $\exp_p : T_p(M) \to M$, is defined by $\exp_p(v) = \alpha_v(1)$ where α_v is a constant speed geodesic starting at p. In case of $SO(n)$, the exponential map $\exp_O : T_O(SO(n)) \to SO(n)$ is given by

$$\exp_O(X) = Oexp(X) \ ,$$

where the exponential on the right side is actually the matrix exponential.

6.2 Stiefel and Grassmann Manifolds as Quotient of SO(n)

A quotient of a space defines equivalence relations between points in the space. If one wants to identify certain elements of a set, using an equivalence relation, then the set of such equivalent classes forms a quotient space. This framework is very useful in understanding the geometry of $\mathscr{S}_{n,d}$ and $\mathscr{G}_{n,d}$ by viewing them as quotient spaces, using different equivalence relations, of $SO(n)$.

$\mathscr{S}_{n,d}$ is the set of all d-dimensional orthnormal bases of $\mathbb{R}^n$ and $\mathscr{G}_{n,d}$ is the set of all d-dimensional subspaces of $\mathbb{R}^n$. A d-dimensional basis of $\mathbb{R}^n$ can be represented by an $n \times d$ matrix U such that $U^T U = I_d$, while a d-dimensional subspace is represented by all such matrices whose columns span that subspace. Notice that such a U can be viewed as the first d columns of an element of $SO(n)$. This sets up the equivalence relations needed to form $\mathscr{S}_{n,d}$ and $\mathscr{G}_{n,d}$ as quotient spaces of $SO(n)$.

1. **Stiefel Manifold**: A **Stiefel** manifold is the set of all orthonormal bases of $\mathbb{R}^n$. Since each orthonormal basis can be identified with an $n \times d$ matrix, a Stiefel manifold is also a set of $n \times d$ matrices with orthonormal columns. More interestingly,

$\mathscr{S}_{n,d}$ can be viewed as a quotient space of $SO(n)$ as follows. Consider the subgroup of smaller rotations $SO(n-d)$ as a subgroup of $SO(n)$ using the embedding: $\phi_a : SO(n-d) \mapsto SO(n)$, defined by

$$\phi_a(V) = \begin{bmatrix} I_d & 0 \\ 0 & V \end{bmatrix} \in SO(n) . \tag{6.8}$$

Now define two elements $O_1, O_2 \in SO(n)$ to be equivalent, i.e. $O_1 \sim_a O_2$, if $O_1 = O_2\phi_a(V)$ for some $V \in SO(n-d)$. (The subscript a is used to distinguish it from another equivalence relation used later for studying $\mathscr{G}_{n,d}$.) Note that $\phi_a(SO(n-d))$ consists of those rotations in $SO(n)$ that rotate only the last $(n-d)$ components in $\mathbb{R}^n$, leaving the first d unchanged. Hence, $O_1 \sim O_2$ if and only if their first d columns are identical, irrespective of the remaining columns. The resulting equivalence classes are:

$$[O]_a = \{O\phi_a(V)|V \in SO(n-d)\}.$$

Since all elements of $[O]_a$ have the same first d columns, we will use that submatrix $U \in \mathbb{R}^{n\times d}$ to represent $[O]_a$. $\mathscr{S}_{n,d}$is now viewed as the set of all such equivalence classes and is denoted simply by $SO(n)/SO(n-d)$.

2. **Grassmann Manifold**: A **Grassmann** manifold is the set of all d-dimensional subspace of $\mathbb{R}^n$. Here we are interested in d-dimensional subspaces and not in a particular basis. In order to obtain a quotient space structure for $\mathscr{G}_{n,d}$, let $SO(d) \times SO(n-d)$ be a subgroup of $SO(n)$ using the embedding $\phi_b : (SO(d) \times SO(n-d)) \mapsto SO(n)$:

$$\phi_b(V_1,V_2) = \begin{bmatrix} V_1 & 0 \\ 0 & V_2 \end{bmatrix} \in SO(n). \tag{6.9}$$

Define an equivalence relation on $SO(n)$ according to $O_1 \sim_b O_2$ if $O_1 = O_2\phi_b(V_1,V_2)$ for some $V_1 \in SO(d)$ and $V_2 \in SO(n-d)$. In other words, O_1 and O_2 are equivalent if the first d columns of O_1 are rotations of the first d columns of O_2 and the last $(n-d)$ columns of O_1 are rotations of the last $n-d$ columns of O_2. An equivalence class is given by:

$$[O]_b = \{O\phi_b(V_1,V_2)|V_1 \in SO(d),\ V_2 \in SO(n-d)\} ,$$

and the set of all such equivalence classes is $\mathscr{G}_{n,d}$. Notationally, $\mathscr{G}_{n,d}$ can also be denoted as simply $SO(n)/(SO(d) \times SO(n-d))$.

For efficiency, we often denote the set $[O]_b$ by the set

$$[U] = \{UO \in \mathbb{R}^{n\times d} | O \in SO(d)\} .$$

where U denotes the first d columns of O.

The main advantage of studying the Stiefel and Grassmann manifolds as quotient spaces of $SO(n)$ is that it lets us use well-known results about geodesics and tangent planes of $SO(n)$ in a systematic manner. Using the tangent structure on $SO(n)$, we can derive tangent structures on the quotient spaces $\mathscr{S}_{n,d}$ and $\mathscr{G}_{n,d}$ using the following principle. If M/H is a quotient space of M under the action of a group $H \subset M$ (assuming H acts on M), then, for any point $p \in M$, a vector $v \in T_p(M)$ is also tangent to M/H as long as it is perpendicular to the tangent space $T_p(pH)$. Here, $T_p(pH)$ is considered as a subspace of $T_p(M)$. We will use this idea to find tangent spaces on $\mathscr{S}_{n,d}$ and $\mathscr{G}_{n,d}$, from the corresponding tangent structure of $SO(n)$.

1. **Tangent Structure of $\mathscr{S}_{n,d}$:** Since $\mathscr{S}_{n,d} = SO(n)/\phi_a(SO(n-d))$, set $M = SO(n)$ and $H = \phi_a(SO(n-d))$, with ϕ_a as defined in Eqn. 6.8. The Jacobian of ϕ_a provides a linear map: $d\phi_a : T_{I_{n-d}}(SO(n-d)) \mapsto T_I(SO(n))$ according to:

$$d\phi_a(D) = \begin{bmatrix} 0 & 0 \\ 0 & D \end{bmatrix} \in T_I(SO(n)).$$

Let $J \in \mathbb{R}^{n\times d}$ be a tall-skinny matrix, made up of the first d columns of I_n; J acts as the “identity” element in $\mathscr{S}_{n,d}$. A vector in $T_{I_n}(SO(n))$, that is perpendicular to $d\phi_a(T_{I_{n-d}}(SO(n-d)))$, when multiplied on right by J results in a tangent to $\mathscr{S}_{n,d}$ at J. A simple calculation shows that

$$T_J(\mathscr{S}_{n,d}) = \{ \begin{bmatrix} C \\ B^T \end{bmatrix} | C \in \mathbb{R}^{d\times d} \text{ skew-symm } , B \in \mathbb{R}^{d\times(n-d)} \} . \qquad (6.10)$$

For any other point $U \in \mathscr{S}_{n,d}$, let $O \in SO(n)$ be a matrix that rotates the columns of U to align with the columns of J, i.e. let $U = O^T J$. Note that the choice of O is not unique. It follows that the tangent space at U is given by: $T_U(\mathscr{S}_{n,d}) = \{O^T G | G \in T_J(\mathscr{S}_{n,d})\}$.

2. **Tangent Structure of $\mathscr{G}_{n,d}$:** In this case, set $M = SO(n)$ and $H = \phi_b(SO(d) \times SO(n-d))$, with ϕ_b as given in Eqn. 6.9. Using the same argument as earlier, the tangent space $T_I(H)$ is considered a subspace of $T_I(SO(n))$ under the embedding $d\phi_b$:
$$d\phi_b(A_1, A_2) = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix} \in T_I(SO(n)) \ .$$
The vectors tangent to $SO(n)$ and perpendicular to the space $(T_{I_d}(SO(d)) \times T_{I_{n-d}}(SO(n-d)))$, will also be tangent to $\mathscr{G}_{n,d}$ after multiplication on right by J. The resulting tangent space at $[J] \in \mathscr{G}_{n,d}$ is:
$$T_{[J]}(\mathscr{G}_{n,d}) = \{ \begin{bmatrix} 0 \\ B^T \end{bmatrix} | \ B \in \mathbb{R}^{d \times (n-d)} \} \qquad (6.11)$$
For any other point $[U] \in \mathscr{G}_{n,d}$, let $O \in SO(n)$ be a matrix such that $U = O^T J$. Then, the tangent space at $[U]$ is given by $T_{[U]}(\mathscr{G}_{n,d}) = \{O^T G | G \in T_{[J]}(\mathscr{G}_{n,d})\}$.

For any $O \in SO(n)$, a geodesic flow in a tangent direction, say, OA, is given by $\psi_O(A,t) = O\exp(tA)$ where exp is the matrix exponential. This is a one-parameter curve with t as the parameter. Similarly, in case of $\mathscr{S}_{n,d}$ and $\mathscr{G}_{n,d}$ a geodesic flow starting from a point $U \in \mathscr{S}_{n,d}$ in a direction $O^T AJ \in T_U(\mathscr{S}_{n,d})$, is given by:
$$\psi_U(O^T AJ, \cdot) : t \mapsto O^T \exp(tA) J \ , \qquad (6.12)$$
Recall that in case of $\mathscr{S}_{n,d}$, the skew-symmetric matrix A is of the type $\begin{bmatrix} C & -B \\ B^T & 0 \end{bmatrix}$, whereas for $\mathscr{G}_{n,d}$ it is of the type $\begin{bmatrix} 0 & -B \\ B^T & 0 \end{bmatrix}$.

6.3 Sample Statistics on the Grassmann manifold

The first question that we consider is: What is a suitable notion of a mean on the Riemannian manifold M ? A popular method for defining a mean on a manifold was proposed by Karcher [169] who used the centroid of a density as its mean.

Karcher Mean [169] The Karcher mean μ_{int} of a probability density function f on M is defined as a local minimizer of the cost function: $\rho : M \to \mathbb{R}_{\geq 0}$, where

$$\rho(p) = \int_M d(p,q)^2 f(q)\, dq\ . \tag{6.13}$$

dq denotes the reference measure used in defining the probability density f on M. The value of the function ρ at the Karcher mean is called the **Karcher mean**. How does the definition of the Karcher mean adapt to a sample set, i.e. a finite set of points drawn from an underlying probability distribution ? Let $q_1, q_2, \ldots, q_k$ be independent random samples from the density f. Then, the sample Karcher mean of these points is defined to be the local minimizer of the function:

$$\rho_k(p) = \frac{1}{k}\sum_{i=1}^{k} d(p,q_i)^2\ . \tag{6.14}$$

An iterative algorithm for computing the sample Karcher mean is as follows. Let μ_0 be an initial estimate of the Karcher mean. Set $j = 0$.

1. For each $i = 1, \ldots, k$, compute the tangent vector v_i such that the geodesic from μ_j, in the direction v_i, reaches q_i at time one, i.e. $\psi_1(\mu_j, v_i) = q_i$ or $v_i = \exp_{\mu_j}^{-1}(q_i)$.

2. Compute the average direction $\bar{v} = \frac{1}{k}\sum_{i=1}^{k} v_i$.

3. If $||\bar{v}||$ is small, then stop. Else, update μ_j in the update direction using

$$\mu_{j+1} = \psi_\varepsilon(\mu_j, \bar{v}),$$

where $\varepsilon > 0$ is small step size, typically 0.5. $\psi_t(p,v)$ denotes the geodesic path starting from p in the direction v parameterized by time t. In other words, $\mu_{j+1} = \exp_{\mu_j}(\varepsilon\bar{v})$.

4. Set $j = j+1$ and return to Step 1.

It can be shown that this algorithm converges to a local minimum of the cost function given in Eqn. 6.14 which is the definition of μ_{int}. Depending upon the initial value μ_0 and the step size ε, it converges to the nearest local minimum.

We exploit the fact that the tangent spaces of M are vector spaces and can provide a domain for defining covariances. We can transfer the probability density f from M to a tangent space $T_p(M)$, using the inverse exponential map, and then use the standard definition of central moments in that vector space. For any point $p \in M$, let $p \to \exp_\mu^{-1}(p)$ denote the inverse exponential map at μ from M to $T_\mu(M)$. The point μ maps to the origin $\mathbf{0} \in T_\mu(M)$ under this map. Now, we can define the Karcher covariance matrix as:

$$K_{int} = \int_{T_\mu(M)} vv^T dv, \;\; v = \exp_\mu^{-1}(q) \; .$$

For a finite sample set, the sample Karcher variance is given by

$$\hat{K}_{int} = \frac{1}{k-1} \sum_{i=1}^{k} v_i v_i^T \; , \;\; \text{where} \;\; v_i = \exp_\mu^{-1}(q_i) \; . \tag{6.15}$$

6.3.1 Parametric Densities

In addition to sample statistics such as the mean and covariance, it is possible to define parametric probability distribution functions on manifolds. We shall here discuss intrinsic methods for defining pdfs. The general idea here is define a pdf on the tangent space of the manifold, and then 'wrap' the distribution back onto the manifold. This allows us to draw upon the wealth of methods available from classical multi-variate statistics for the problem at hand.

Suppose, we have n sample points, given by $q_1, q_2, ...q_n$ from a manifold $\mathscr{M}$. Then, we first compute their Karcher mean $\bar{q}$ as discussed before. The next step is to define and compute a sample covariance for the observed q_i's. The key idea here is to use the fact that the tangent space $T_{\bar{q}}(q)$ is a vector space. For a d-dimensional manifold, the tangent space at a point is also d dimensional. Using a finite-dimensional approximation, say $V \subset T_{\bar{q}}(q)$, we can use the classical multivariate calculus for this purpose. The resulting sample covariance matrix is given by:

$$\bar{\Sigma} = \frac{1}{n-1} \sum_{i=1}^{n} v_i v_i^T$$

where each v_i is a d-dimensional sample of the function $exp_{\bar{q}}^{-1} q_i$. Note that by definition, the mean of v_is should be zero. In cases where the number of samples n

is smaller than d, one can apply an additional dimension-reduction tool to work on a smaller space. For instance, we can use the singular value decomposition (SVD) of the sample covariance matrix $\bar{\Sigma}$ and retain only the top m significant singular values and the corresponding singular vectors. In such cases, the covariance matrix is indirectly stored using $\lambda_1, \lambda_2, ...\lambda_m$ singular values and their corresponding singular vectors $u_1, u_2, ...u_m$.

The exponential map: $\exp_{\bar{q}} : T_{\bar{q}}(q) \to \mathscr{M}$ maps this covariance back to $\mathscr{M}$. Specifically, this approach is widely used to define wrapped-Gaussian densities on a given manifold. In general, one can define arbitrary pdfs on the tangent plane such as mixtures of Gaussians, Laplace etc and wrap it back to the manifold via the exponential map. This allows us to experiment with and choose an appropriate pdf that works well for a given problem domain.

6.3.1.1 Some Synthetic Examples

In this section, we illustrate via some simple examples the concepts of karcher mean and wrapped distributions for the Grassmann manifold. To help visualize the results, we choose $\mathscr{G}_{n,d}$ with $n = 2$ and $d = 1$ i.e. 1-dimensional subspaces of $\mathbb{R}^2$. This is easily visualized as the set of all lines passing through of the origin on the X-Y plane. Lines on a plane can be parametrized by their principal angle with the X-axis. Using this parameterization, in the first experiment we randomly sample directions centered around $\theta = \pi/3$ with variance in θ set to 0.2. A set of such samples in shown in figure 6.1 with dotted blue lines. The Karcher mean of this set is shown as a red line in figure 6.1. As can be seen, the Karcher mean corresponds well to the notion of a ‘mean-axis’ in this case.

In the next experiment, we sampled two sets of lines centered at $\theta = \pi/3$ and $\theta = 2 * \pi/3$ once with equal variances as shown in figure 6.2 and once with unequal variances as shown in figure 6.3. In both cases, the karcher mean is vertically oriented as shown in the plots which is the physically meaningful solution we expect.

Finally, in figure 6.4 we illustrate the concept of the wrapped normal distribution. In this experiment, we generated samples from two classes - one centered at $\theta = 0$ and the other centered at $\theta = \pi/2$. Points from each class are shown in different colors. The

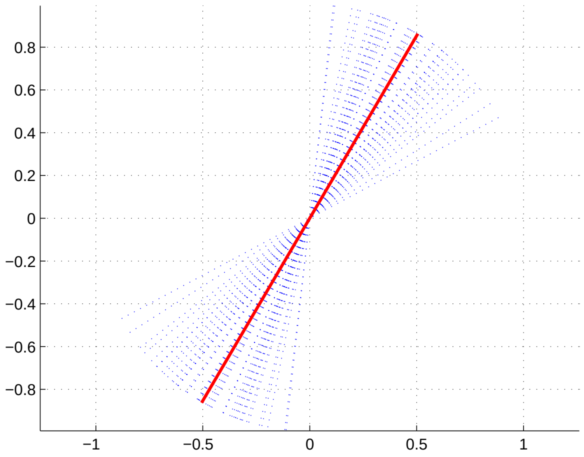


Figure 6.1: Illustration of Karcher mean on the Grassmann manifold. In $\mathbb{R}^2$ the set of all axes (lines passing through the origin) is the Grassmann manifold with $n = 2$ and $d = 1$. Blue dotted lines represent individual points on the Grassmann manifold. The bold red line is the Karcher mean of this set. The Karcher mean corresponds to the notion of a mean axis.

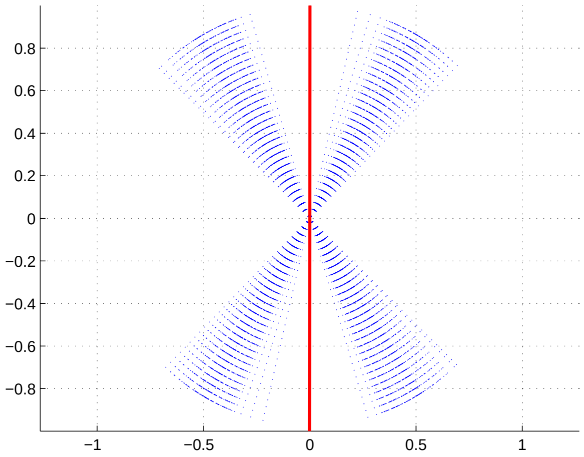


Figure 6.2: Karcher mean of two clusters of lines with equal spread. One cluster is centered at $\theta = \pi/3$ to the X-axis and the other is clustered near $\theta = 2\pi/3$. The bold red line is the Karcher mean of this set. It corresponds to the physically meaningful solution of a vertical axis as the mean.

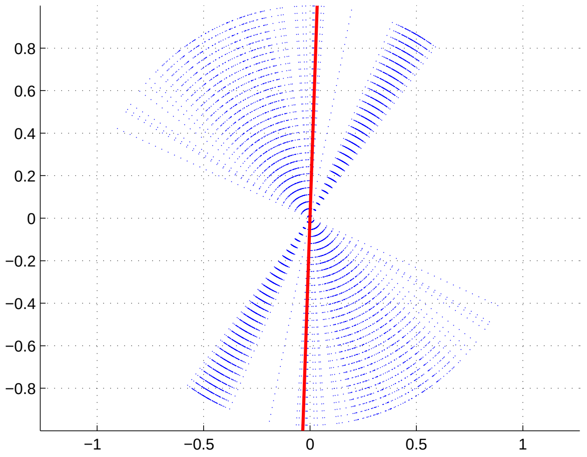


Figure 6.3: Karcher mean of two clusters of lines with unequal spread. One cluster is centered at $\theta = \pi/3$ to the X-axis and the other is clustered near $\theta = 2\pi/3$. The bold red line is the Karcher mean of this set. It corresponds to the physically meaningful solution of a vertical axis as the mean.

Karcher mean of the whole dataset was taken as the pole to compute the tangent vectors for the points. Each of the classes was parameterized by a mean μ and standard-deviation σ on the tangent plane. The points corresponding to μ and and $\mu \pm \sigma$ were then wrapped back onto the manifold. The mean and standard-deviation axes for each of the classes are shown as bold and dashed lines respectively in figure 6.4.

6.3.2 Note on Efficient Computations

To compute the Karcher mean, we need efficient methods for two sub-problems. Given a point S_0 on the manifold, how does one move on the manifold along a specified direction ? and, b) Given two points S_0 and S_1, how does one compute the direction that takes S_0 toward S_1. Efficient methods have been proposed for these two tasks by Gallivan et al [170]. Here we summarize the key results that will be used in this chapter. Recall that geodesic paths on $SO(n)$ are given by one-parameter exponential flows $t \rightarrow exp(tA)$, where $A \in \mathbb{R}^{n\times n}$ is a skew-symmetric matrix. The quotient geometry of the Grassmann manifold implies that geodesics in $\mathscr{G}_{n,d}$ are given by one-parameter exponential flows $t \rightarrow exp(tA)$ where A has a more specific structure given by

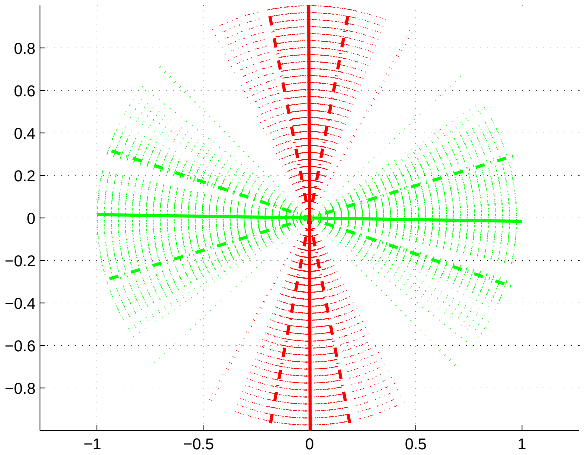


Figure 6.4: Wrapped Normal class conditional-densities of two classes on the Grassmann manifold. Each class is shown in a different color. The mean of each class is shown in bold lines. The wrapped standard-deviation lines are shown in dashed lines for each class.

$$A = \begin{pmatrix} 0 & B^T \\ -B & 0 \end{pmatrix} \tag{6.16}$$

where $B \in \mathbb{R}^{(n-d)\times d}$. The matrix B parameterizes the direction and speed of geodesic flow. We now discuss the solution to the two questions enumerated above.

6.3.3 Moving along the Geodesic

Given a point on the Grassmann manifold S_0 represented by orthonormal basis Y_0, and a direction matrix B, the one-parameter geodesic path emanating from Y_0 in this direction is given by

$$Y(t) = Q\ exp(tA)\ J \tag{6.17}$$

where, $Q \in SO(n)$ and $Q^T Y_0 = J$ and $J = [I_d; 0_{n-d,d}]$. Given Y_0 and A the following are the steps involved in sampling $Y(t)$ for various values of t.

1. Compute the $n \times n$ orthogonal completion Q of Y_0. This can be achieved by the QR decomposition of Y_0.

2. Compute the compact SVD of the direction matrix $B = \tilde{U}_2 \Theta U_1$.

3. Compute the diagonal matrices $\Gamma(t)$ and $\Sigma(t)$ such that $\gamma_i(t) = cos(t\theta_i)$ and $\sigma_i(t) = sin(t\theta_i)$, where θ's are the diagonal elements of Θ.

4. Compute
$$Y(t) = Q \begin{pmatrix} U_1\Gamma(t) \\ -\overline{U}_2\Sigma(t) \end{pmatrix} \tag{6.18}$$
for various values of $t \in [0,1]$.

6.3.4 Computing the Velocity Matrix

Now, given two points on the manifold S_0 and S_1 with orthonormal basis Y_0 and Y_1, we need an efficient way to compute the velocity parameter B such that traveling in this direction from S_0 leads to S_1 in unit-time. Given two subspaces S_0 and S_1 and corresponding $n \times d$ orthonormal basis-vectors Y_0 and Y_1:

1. Compute the $n \times n$ orthogonal completion Q of Y_0.

2. Compute the thin CS decomposition of $Q^T Y_1$ given by

$$\begin{aligned} Q^T Y_1 = \begin{pmatrix} X \\ Y \end{pmatrix} &= \begin{pmatrix} U_1 & 0 \\ 0 & U_2 \end{pmatrix} \begin{pmatrix} \Gamma(1) \\ -\Sigma(1) \\ 0 \end{pmatrix} V_1^T \\ &= \begin{pmatrix} U_1 & 0 \\ 0 & \tilde{U}_2 \end{pmatrix} \begin{pmatrix} \Gamma(1) \\ -\Sigma(1) \end{pmatrix} V_1^T \end{aligned}$$

3. Compute $\{\theta_i\}$ which are given by the arcsine and arcos of the diagonal elements of Γ and Σ respectively. i.e. $\gamma_i = cos(\theta_i)$ and $\sigma_i = sin(\theta_i)$. Form the diagonal matrix Θ containing θ's on its diagonal.

4. Compute $A = \tilde{U}_2 \Theta U_1$.

6.4 Non-parametric methods: Procrustes Representation for the Grassmann manifold

The Stiefel and Grassmann manifolds are endowed with a Riemannian structure that lends itself to computation of distances between points on the manifold via geodesics. The Riemannian computations outlined above are in general computationally expensive for a general manifold. Though efficient algorithms have been proposed for the Stiefel and Grassmann manifolds, Karcher mean computation is an iterative procedure. In recent years the Procrustes methods proposed by [154] have become popular for non-iterative density estimation as an alternative. However, as will be seen later this approach requires a choice of parameters (kernel-width) whose optimal value is not known in advance.

A point X on $\mathscr{S}_{n,d}$ is represented as a tall-thin $n \times d$ orthonormal matrix. The corresponding equivalence class of $n \times d$ matrices XR in $R_{m,k}$, for $R \in SO(d)$ is also called the Procrustes representation of the Stiefel manifold. Thus, to compare two points in $\mathscr{G}_{n,d}$, we simply compare the smallest squared distance between the corresponding equivalence classes on the Stiefel manifold according to the Procrustes representation. Given matrices X_1 and X_2 on $\mathscr{S}_{n,d}$, the smallest squared Euclidean distance between any pair of matrices in the corresponding equivalence classes is given by

$$d^2_{Procrust}(X_1, X_2) = \min_R tr(X_1 - X_2R)^T(X_1 - X_2R) \tag{6.19}$$

$$= \min_R tr(R^TR - 2X_1^TX_2R + I_k) \tag{6.20}$$

When R varies over the orthogonal group $O(k)$, the minimum is attained at $R = H_1H_2^T = A(A^TA)^{-1/2}$, where $A = H_1DH_2^T$ is the singular value decomposition of A. We refer the reader to [154] for proofs and alternate cases.

Given several examples from a class $(X_1, X_2, \ldots, X_n)$ on the manifold $V_{k,m}$, the class conditional density can be estimated using an appropriate kernel function. We first assume that an appropriate choice of a divergence on the manifold has been made such as the one above. For the Procrustes measure the density estimate is given by [154] as

$$\hat{f}(X;M) = \frac{1}{n}C(M)\sum_{i=1}^{n} K[M^{-1/2}(I_k - X_i^T X X^T X_i)M^{-1/2}] \tag{6.21}$$

where $K(T)$ is the kernel function, M is a $k \times k$ positive definite matrix which plays the role of the kernel width or a smoothing parameter. $C(M)$ is a normalizing factor chosen so that the estimated density integrates to unity. The matrix valued kernel function $K(T)$ can be chosen in several ways. We have used $K(T) = exp(-tr(T))$ in all the experiments reported in this chapter. In this non-parametric method for density estimation, the choice of kernel width M becomes important. Thus, though this is a non-iterative procedure, the optimal choice of the kernel width can have a large impact on the final results. In general, there is no standard way to choose this parameter except for cross-validation.

6.5 Experiments on Linear Dynamic Models

6.5.1 Experiments on Activity Recognition

Activity	**Dim. Red.** [3] 16^3 volume	**Best Dim. Red.** [3] 64^3 volume	**Subspace Angles** 16^3 volume	**NN-Procrust** 16^3 volume
Check Watch	76.67	86.66	93.33	90
Cross Arms	100	100	100	96.67
Scratch Head	80	93.33	76.67	90
Sit Down	96.67	93.33	93.33	93.33
Get Up	93.33	93.33	86.67	80
Turn Around	96.67	96.67	100	100
Walk	100	100	100	100
Wave Hand	73.33	80	93.33	90
Punch	83.33	96.66	93.33	83.33
Kick	90	96.66	100	100
Pick Up	86.67	90	96.67	96.67
Average	88.78	93.33	93.93	92.72

Table 6.1: Comparison of view invariant recognition of activities in the INRIA dataset using a) Best DimRed [3] on $16 \times 16 \times 16$ features, b) Best Dim. Red. [3] on $64 \times 64 \times 64$ features, c) Nearest Neighbor using ARMA model distance ($16 \times 16 \times 16$ features), d) Nearest Neighbor using Procrustes distance ($16 \times 16 \times 16$ features)

We performed a recognition experiment on the publicly available INRIA dataset [3]. The dataset consists of 10 actors performing 11 actions, each action executed 3 times at varying rates while freely changing orientation. We used the view-invariant representation and features as proposed in [3]. Specifically, we used the $16 \times 16 \times 16$ circular FFT features proposed by [3]. Each activity was modeled as a linear dynamical system. Testing was performed using a round-robin experiment where activity models were learnt using 9 actors and tested on 1 actor. For the kernel method, all available training instances per class were used to learn a class-conditional kernel density as described in section 6.4. In table 6.1, we show the recognition results obtained using four methods. The first column shows the results obtained using dimensionality reduction approaches of [3] on $16 \times 16 \times 16$ features. [3] reports recognition results using a variety of dimensionality reduction techniques (PCA, LDA, Mahalanobis) and here we choose the row-wise best performance from their experiments (denoted 'Best Dim. Red.') which were obtained using $64 \times 64 \times 64$ circular FFT features. The third column corresponds to the method of using subspace angles based distance between dynamical models [60]. Column 4 shows the nearest-neighbor classifier performance using Procrustes distance measure ($16 \times 16 \times 16$ features). We see that the manifold Procrustes distance performs as well as ARMA model distance.

In table 6.1 we show results of statistical modeling using parametric and non-parametric methods. For the parametric method, we consider two cases - single pole and multiple poles. In the single pole case, the tangent plane is constructed at the Karcher mean of the entire training dataset. In the multiple pole case, we construct a class-specific tangent plane at the Karcher mean of each of the classes. For classification of a test-point, we compute its probability of belonging to a class using the wrapped normal on the class-specific tangent plane. Then, the point is classified into the class that has the highest likelihood. As can be seen in the results in table 6.2, statistical modeling of class conditional densities leads to a significant improvement in recognition performance over simpler methods shown in table 6.1. Note that even though the manifold approaches presented here use only $16 \times 16 \times 16$ features they outperform other approaches that use higher resolution ($64 \times 64 \times 64$ features) as shown in table 6.1.

Activity	Wrapped Normal: Single Pole	Wrapped Normal: Multiple Poles	Procrustes Kernel $M = I$
Check Watch	96.67	100	100
Cross Arms	93.33	100	100
Scratch Head	93.33	90	96.67
Sit Down	90	96.67	93.33
Get Up	100	96.67	96.67
Turn Around	96.67	100	100
Walk	93.33	90	100
Wave Hand	86.67	93.33	100
Punch	90	100	100
Kick	93.33	100	100
Pick Up	93.33	100	100
Average	93.33	96.06	98.78

Table 6.2: Results of Statistical Modeling on recognition of activities in the INRIA dataset using a) Wrapped Normal + Single Tangent Plane b) Wrapped Normal + Class specific tangent plane c) Procrustes Kernel method M = I.

Kernel width: M	$10^{-3} * I$	$10^{-2} * I$	$10^{-1} * I$	$10^{0} * I$	$10^{1} * I$	$10^{2} * I$	$10^{3} * I$
Avg. Performance	90	97.87	97.87	98.78	93.63	90.91	90.91

Table 6.3: INRIA Activity Recognition: Variation of performance of the kernel density estimator with different choices of the width parameter M.

As mentioned before, for the non-parametric case, an appropriate choice of the kernel width M has to be made. In general, cross-validation is suggested to estimate the optimal kernel width. Different classes may have a different optimal kernel width. Hence, cross-validation requires a lengthy training phase. A sub-optimal choice can often lead to poor performance. This is one of the significant drawbacks of non-parametric methods. In table 6.3, we empirically show how the performance depends on the choice of the kernel width. We choose the kernel to be of the form $M = \sigma * I$. We choose $\sigma = 10^{-3}, 10^{-2}, \ldots, 10^{3}$, and show the variation of the performance.

6.5.2 Video-Based Face Recognition

Video-based face recognition (FR) by modeling the 'cropped video' either as dynamical models ([171]) or as a collection of PCA subspaces [172] have recently gained popularity because of their ability to recognize faces from low resolution videos. However, in this case, we focus only on the C matrix of the ARMA model or PCA subspace as the distinguishing model parameter. This is because the C matrix encodes the appearance of the face, whereas the A matrix encodes the dynamic information. The C matrices are orthonormal, hence points on the Stiefel manifold. But, for recognition applications, the important information is encoded in the subspace spanned by the C matrix. Hence, we identify the model parameters (C's) as points on the Grassmann Manifold.

We performed a recognition experiment on the NIST-MBGC Video Challenge dataset. The MBGC dataset consists of a large number of subjects walking towards a camera in a variety of illumination conditions. Face regions are tracked and a sequence of cropped images is obtained. There were a total of 143 subjects with the number of videos per subject ranging from 1 to 5. In our experiments we took subsets of the dataset which contained at least 2 sequences per person denoted as S_2, at least 3 sequences per person denoted as S_3 etc. Each of the face-images was first preprocessed to zero-mean and unity variance. In each of these subsets, we performed a leave-one-out testing. The results of the leave one out testing are shown in table 6.4. Also reported are the total number of distinct subjects and the total number of video sequences in each of the subsets. In the comparisons, we show results using the 'arc-length' metric between subspaces [160]. This metric computes the subspace angles between two subspaces and takes the frobenius norm of the angles as a distance measure [160]. We also show comparisons with the Procrustes measure, the Kernel density estimate with $M = I$ and a wrapped normal density with the Karcher mean of the entire dataset as the pole.

As can be seen, statistical methods outperform nearest neighbor based approaches. As one would expect, the results improve when more examples per class are available. Since the optimal kernel-width is not known in advance, this might explain the relatively poor performance of the kernel density method.

Subset	**Distinct Subjects**	**Total Sequences**	**Arc-length Metric**	**Procrustes Metric**	**Kernel density**	**Wrapped Normal**
S_2	143	395	38.48	43.79	39.74	**63.79**
S_3	55	219	48.85	53.88	50.22	**74.88**
S_4	54	216	48.61	53.70	50.46	**75**
Avg.			45.31%	50.45%	46.80%	**71.22%**

Table 6.4: Comparison of video based face recognition approaches using a) Subspace Angles + Arc-length metric, b) Procrustes Distance, c) kernel density, d) Wrapped Normal on Tangent Plane

Chapter 7

Applications to Still Image based Recognition

Many applications in computer vision such as dynamic textures [123],[56], human activity modeling and recognition [53],[4], video based face recognition [171], shape analysis [166],[167] involve learning and recognition of patterns from exemplars which obey certain constraints. In this chapter, we shall examine a broad class of applications where the underlying constraints on the data have a special structure. The structure under study is the linear subspace structure. Subspace constraints have proved to be a simple yet powerful tool in several applications. While estimating linear subspace models of variation is standard fare in several problems in vision such as linear regression, linear classification, linear subspace estimation etc, much less attention has been devoted to statistical inference on the space of linear subspaces.

In many of these applications, given a database of examples and a query, the following two questions are usually addressed – a) what is the 'closest' example to the query in the database ? b) what is the 'most probable' class to which the query belongs ? A systematic solution to these problems involves a study of the underlying constraints that the data obeys. The answer to the first question involves study of the geometric properties of these constraints, which then leads to appropriate definitions of distance metrics such as geodesics etc. The answer to the second question involves statistical modeling of inter- and intra-class variations. We shall discuss in a later section that the space of linear subspaces can be shown to be a Riemannian manifold. More formally, the space of k-dimensional subspaces in $\mathbb{R}^n$ is called the Grassmann manifold. On a related note, the Stiefel manifold is the space of k orthonormal vectors in $\mathbb{R}^n$. The study of these manifolds has important consequences for applications such as dynamic textures [123, 56], human activity modeling and recognition [53, 4], video based face recognition [171] and shape analysis [166, 167] where data naturally lies either on the Stiefel or the Grassmann manifold.

First, we discuss some motivating examples in vision that illustrate the need to study these manifolds and their geometry.

7.1 Motivating Examples

1. **Spatio-temporal dynamical models:** A wide variety of spatio-temporal data in computer vision are modeled as realizations of dynamical models. Examples include Dynamic textures [123], human joint angle trajectories [53] and silhouette sequences [4]. One popular dynamical model for such time-series data is the autoregressive and moving average (ARMA) model. For the ARMA model closed form solutions for learning the model parameters have been proposed in [57, 123] and are widely used. An ARMA model can be equivalently considered as the subspace spanned by the columns of its observability matrix. A subspace such as this, is a point on the Grassmann manifold. Given several instances, current approaches involve computing the distance between them using well-known distance measures [60] followed by nearest neighbor classification. Instead, given several instances of each class we can learn compact class conditional probability density functions over the parameter space – the Grassmann manifold spanned by the columns of the observability matrix in this case. This is an example of a modeling constraint that leads to linear subspace structure of the data.

2. **Shape Analysis:** Representation and recognition of shapes is a well understood field in statistics and vision [8, 173]. The shape observed in an image is a perspective projection of the original 3D shape. In order to account for this, shape theory studies the equivalent class of all configurations that can be obtained by a specific class of transformation (e.g. linear, affine, projective) on a single basis shape. It can be shown that affine and linear shape spaces for specific configurations can be identified by points on the Grassmann manifold [167]. Given several exemplar shapes belonging to a few known classes, we are interested in estimating a probability distribution over the shape space for each of the classes. These can then be used for problems such as retrieval, classification or even to learn a generative

model for shapes. This is an example of an invariance requirement that leads to a linear subspace structure of data.

3. **Image Matching and retrieval:** In image and object recognition, recent methods have focused on utilizing multiple images of the same object, taken under varying viewpoints or varying illumination conditions, for recognition [174, 163, 175, 176]. e.g. The set of face images of the same person under varying illumination conditions is frequently modeled as a linear subspace of 9-dimensions which is motivated from the nine-points of light model [177]. In such applications, an object 'category' consists of image-sets of several 'instances'. For example, a category of horses would have image-sets of several distinct horses, with several images per distinct horse. A common approach in such applications is to approximate the image-space of a single instance under these variations as a linear subspace [163, 178]. Linear subspaces are points on the Grassmann manifold. Given several image-sets per object category, the goal then is to learn a statistical model over the Grassmann manifold.

4. **On-line Visual Learning via Subspace Tracking:** Applications involving dynamic environments and autonomous agents such as a mobile robot navigating through an unknown space cannot be represented by static models. In such applications it is important to adapt models, that have been learnt offline, according to new observations in an online fashion. One approach is to perform incremental PCA to dynamically learn a better representational model as the appearance of the target dynamically changes as in [179]. Incremental PCA has also been used to recognize abnormalities in the visual field of a robot as in [180]. In an unrelated domain, the theory of subspace tracking on the Grassmann manifold [146] has been developed for array signal processing applications. Since PCA basis vectors represent a subspace which is identified by a point on the Grassmann manifold, subspace tracking lends itself readily to statistical analysis for online visual learning applications.

5. **Projective Geometry:** A fundamental concept inherent in projective geometry is the notion of scale ambiguity [181]. In homogeneous co-ordinates, two points re-

lated by a constant scale factor are considered to be equivalent. Thus, points in 3D are considered as lines in 4D homogeneous space passing through the origin. Similarly points in 2D are considered as lines in homogeneous 3D space. The set of lines passing through the origin is a special case of the Grassmann manifold. The scale ambiguity also manifests in several other quantities such as the fundamental matrix, essential matrix etc. Applications such as estimating fundamental matrices or computing an average fundamental matrix from several independent estimates require statistical methods on the Grassmann manifold.

Contributions: We first show how a large class of problems in computer vision can be recast as statistical inference problems on the Stiefel and/or Grassmann manifolds. Then, we solve these problems using the Riemannian geometry of the manifolds. We also discuss some recently proposed non-Riemannian approaches to statistical modeling on the Grassmann manifold. Finally, we present a wide range of experimental evaluation to demonstrate the effectiveness of these approaches and provide a comprehensive comparison. We show in the chapter that inspite of the ease of use of non-Riemannian approaches, their performance is tied to a good choice of parameters. On the other hand, the performance of the Riemannian approaches is consistent over several applications with minimal tuning of parameters.

Next we present a few application areas and experiments that demonstrate the usefulness of statistical analysis on the manifolds.

7.2 Object and Image Classification

Recent efforts in object recognition, have focused on utilizing multiple images of the same object, taken under varying viewpoints or varying illumination conditions [174, 163, 175, 176]. The most common physical factors that give rise to the multitude of appearances are illumination and view change. There has been significant research in understanding the mathematics of these variations in computer vision. A simplistic model for object appearance variations is a mixture of subspaces. In this section, we shall explore how multiple exemplars can be effectively utilized in a subspace framework for object recognition.

We consider the CMU-PIE face dataset which contains images of 68 persons under varying poses, illumination and expressions. For comparison, we use the methods proposed in [163]. The methods proposed in [163] involve discriminative approaches on the Grassmann manifold using Mercer-kernels. In this approach, a Mercer-kernel is defined on the Grassmann manifold which then enables using kernel versions of SVMs, Fisher Discriminant Analysis etc for classification. In this experiment, we use the experimental protocol suggested in [182]. For each of the 68 subjects, 7 near frontal poses are used in the experiment. For each person under a fixed pose, we approximate the variations due to expressions and illumination as a linear subspace. Thus, for each person we have a set of subspaces corresponding to each pose. This allows us to build a statistical model on the Grassmann manifold for each person. A round-robin experiment is performed in which 6 poses are used for training and the remaining pose is used for testing. The results are shown in table 7.1. The results using the other methods were reported in [182].

Subspace Dimension	m=2	m=3	m=4	m=5	m=6	m=7	m=8	m=9
GDA (Proj) [163]	74.8	**89.8**	87.2	91.7	92.5	93.8	93.6	95.3
GDA (BC) [163]	71.4	82.5	64.8	58.6	47.5	43.1	39.9	36.3
MSM [183]	67.0	65.0	64.6	64.2	64.0	64.6	64.6	64.6
cMSM [184]	71.2	67.6	68.2	69.7	69.9	70.2	72.7	72.5
DCC [174]	**78.9**	66.5	63.8	64.6	67.6	67.6	67.6	65
Kernel Density: $M = I$	78.36	88.44	**89.91**	**93.69**	**95.79**	**97.26**	**96.84**	**97.26**
Wrapped Normal: Single Pole	69.95	76.89	69.74	77.73	79.83	79.20	80.46	76.26
Wrapped Normal: Multiple Poles	69.95	76.89	70.16	77.31	82.56	84.66	85.50	86.97

Table 7.1: CMU-PIE Database: Face Identification using various Grassmann statistical methods. Performance of various methods is compared as the subspace dimension is varied.

As can be seen, the proposed statistical approaches compare well with the state of the art. In particular, the kernel density method outperforms all of the other methods. The discriminative approaches of [163] outperforms the wrapped normal approach. However, the variability of the performance is high depending on what Mercer kernel is chosen. The wrapped normal provides consistent performance and beats most of the other methods.

7.3 Affine Shape Analysis

Algorithmic Details: The representation and analysis of shapes has important applications in object recognition, gait recognition and image registration. Landmark based shape analysis is one of the most widely used approaches for representing shapes. A shape is represented by a set of landmark points on its contour. A shape is represented by the matrix $L = [(x_1,y_1);(x_2,y_2);\ldots;(x_m,y_m)]$, of the set of m landmarks of the centered scaled shape. The *shape space* of a base shape is the set of equivalent configurations that are obtained by transforming the base shape by an appropriate spatial transformation. For example, the set of all affine transformations of a base shape forms the *affine shape space* of that base shape. More rigorously, let $\chi = (x_1,x_2,\ldots,x_m)$ be a configuration of m points where each $x_i \in R^2$. Let γ be a transformation on R^2. For example, γ could belong to the affine group, linear group, projective group etc. Let

$$A(\gamma,(x_1,\ldots,x_m)) = (\gamma(x_1),\ldots,\gamma(x_m)) \tag{7.1}$$

be the *action* of γ on the point configuration.

In particular, the *affine shape space* [166] [185] is very important because the effect of the camera location and orientation can be approximated as affine transformations on the original base shape. The affine transforms of the shape can be derived from the base shape simply by multiplying the shape matrix S by a 2×2 full rank matrix on the right (translations are removed by centering). Multiplication by a full-rank matrix on the right preserves the column-space of the matrix S. Thus, all affine deformations of the same base shape, map to the same point on the Grassmann manifold [185]. Therefore, a systematic study of affine shape space essentially boils down to a study of the points on the Grassmann manifold. We can use both Procrustes distance and kernel density methods

Algorithm	**Rank 1**	**Rank 2**	**Rank 3**	**Rank 4**
SC [186]	20/40	10/40	11/40	5/40
IDSC [186]	40/40	34/40	35/40	27/40
Hashing [187]	40/40	38/40	33/40	20/40
Grassmann Procrustes	38/40	30/40	23/40	17/40

Table 7.2: Retrieval experiment on articulation dataset. Last row is the results obtained using Grassmann manifold Procrustes representation. No articulation invariant descriptors were used.

described earlier for several applications of affine invariant shape analysis such as shape retrieval and recognition.

7.3.1 Articulation Database

We conducted a retrieval experiment on the articulated shape database from [186]. We use the same test scheme proposed in [186]. The database consists of 8 object classes with 5 examples for each class. For each shape, 4 top matches are selected and the number of correct hits for ranks $1, 2, 3, 4$ are reported. Table 7.2 summarizes the results obtained on this dataset. The proposed approach compares well with other approaches. It should be noted however, that this is not a fair comparison, as we do not use any articulation-invariant descriptors such as the ones used in [186] and [187]. In spite of this, manifold-based distance metrics perform very well.

7.3.2 Affine MPEG-7 Database

Since the strength of the approach lies in affine invariant representation of shapes, we conducted a synthetic experiment using the MPEG-7 database. We took one base shape from each of the 70 object classes and created 10 random affine warps of the shapes with varying levels of additive noise. This new set of shapes formed the gallery for the experiment. Sample shapes that were generated are shown in figure 7.1. The test set was created by randomly picking a gallery shape and affine warping it with ad-

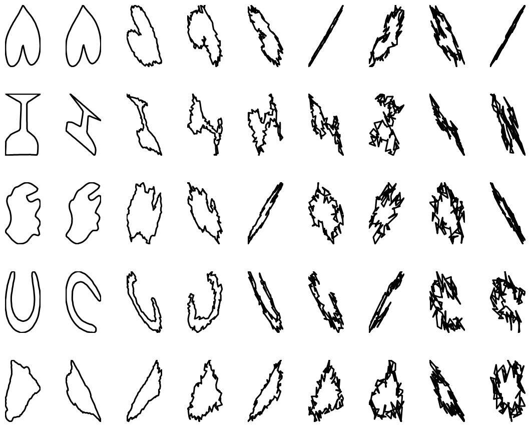


Figure 7.1: Synthetic data generated from the MPEG database. The first column shows base-shapes from the original MPEG dataset for 5 objects. The remaining columns show random affine warps for the base shapes with increasing levels of additive noise.

ditive noise. The recognition experiment was performed using the Procrustes distance and the kernel density methods. For comparison, we used the popular shape Procrustes distance [8] as a baseline measure. We also used the 'arc-length' distance metric used in [9]. The arc-length distance metric is the Frobenius norm of the angle between two subspaces. In all cases, the experiments were repeated with 100 Monte-Carlo trials for each noise level in order to robustly evaluate the performance. The performance of the methods is compared in Figure 7.2 as a function of noise to signal ratio. It can be seen that manifold-based methods perform significantly better than straightforward shape Procrustes measures. Among the manifold methods, the kernel density method outperforms both the Procrustes and the arc-length distance measures. Since the Grassmann manifold based methods accurately account for the affine variations found in the shape, they outperform simple methods that do not account for affine invariance. Moreover, since the kernel methods learn a probability density function for the shapes on the Grassmann manifold, it outperforms distance based nearest neighbor classifiers using Grassmann arc-length and Grassmann Procrustes.

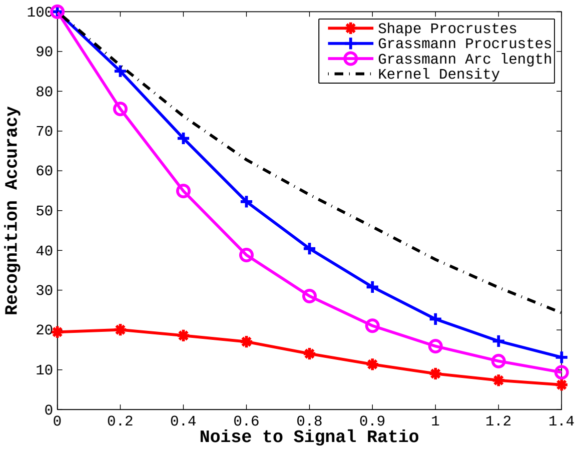


Figure 7.2: Comparison of recognition performance on MPEG-7 database. For comparison we used the shape Procrustes measure [8] and the Grassmann arc-length distance [9]. Manifold based methods perform significantly better than direct application of shape Procrustes measure. Among the manifold methods, statistical modeling via kernel methods outperforms the others.

7.3.3 Sampling from Distributions

Generative capabilities of parametric probability densities can be exploited via appropriate sampling strategies. Once the distribution is learnt, one can synthesize samples from the distribution in a two step process. We first generate a sample from a proposal distribution (we used a matrix-variate normal centered around the class mean), then we use an accept-reject strategy to generate the final shape [154]. We show a sampling experiment using this technique. For this experiment, we took one shape from each of the object classes in the MPEG-7 database and corrupted it with additive noise to generate several noisy samples for each class. We used the Grassmann representation of points as idempotent projection matrices. Then, we learnt a parametric Langevin distribution on the Grassmann manifold for each class. Note that the distribution is learnt on the Grassmann manifold, hence, a sample from the distribution represents a subspace in the form of a projection matrix. To generate an actual shape we need to first choose a $2-frame$ for the generated subspace which can be done via SVD of the projection matrix. Once the $2-frame$ is chosen, actual shapes can be generated by choosing random coordinates in the $2-frame$. We show sampling results in Figure 7.3.

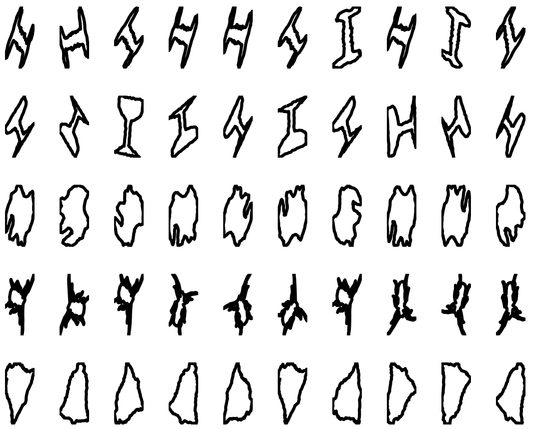


Figure 7.3: Samples generated from estimated class conditional densities for a few classes of the MPEG dataset

7.4 Age Estimation

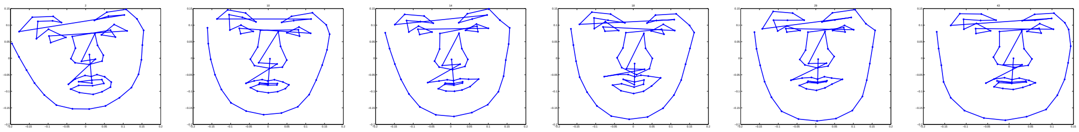
Understanding and modeling of aging in human faces is an important problem in many real-world applications such as biometrics, authentication and synthesis. In this chapter, we provide a Riemannian interpretation of the geometric attributes of faces as they age. Specifically, we consider faces to be described by a set of landmark points on the face whose geometry can be described as a Grassmann manifold. Then the problem of age estimation is posed as a problem of function estimation on the manifold. Further, motivated by studies in neuroscience, we quantify the deformation that warps an 'average' face to a given face. This deformation is then shown to contain important information about the age of the face. The warping of an average face to a given face is then considered to be described by a velocity vector that transforms the average to a given face along a smooth geodesic in unit-time. We show experiments on age estimation using the standard FG-Net dataset and a passport dataset which illustrate the effectiveness of this approach.

The modeling of the appearance of human faces is an important component in several applications such as biometrics, animation, and picture annotation. Faces are deformable 3D objects. As a result of the imaging process, the perceived 2D appearance of a given face exhibits wide variation due to illumination changes, shadows, and pose variations. These variations are usually referred to as structured variations since there exist mathematical models of image formation under these conditions. Unstructured variations such as expressions further increase the space of 2D appearances of a given face. Given a 2D image of a face, humans are capable of factoring out these variations in a manner that has not yet been fully understood. Several computational approaches to account for these variations have been proposed and we refer to [188] for a survey. Facial geometry and texture, both aid in several perception tasks such as recognition, age perception, and matching.

In this chapter, by facial geometry we refer to the location of 2D facial landmarks on images. We discuss how to characterize the 'space' of these facial landmarks. We provide a mathematically well grounded and unified Riemannian framework for modeling facial geometry. The proposed Riemannian interpretation enables the application of a rich class of classification and inference tools. To demonstrate the practical utility and power of these methods, we choose the problem of age-estimation as an example. However, the primary goal is not to provide an algorithm for age-estimation, but to provide a systematic and unified perspective for facial geometric modeling. The theory developed here would prove useful in other face modeling tasks where an accurate description of statistical models on face-spaces is required. We demonstrate in experiments that even with simple learning and regression methods, the results of the proposed framework are comparable to several complex and optimized state-of-the-art systems, and even outperform many of them. Thus, the proposed framework can form the basis of a more principled approach to facial geometric modeling that can be optimized to reach even higher performance levels in several applications.

One might ask, why do we choose age-perception as the example and what is the role of geometry in it ? Aging is a source of variation which has only recently been gaining attention. Understanding the appearance variations induced by aging is important for applications where the claimed identity and the enrolled face may show a large difference in apparent age. Studies in neuroscience have shown that facial geometry is a strong factor that influences age perception [189]. In [189], it is shown that shape-averaged faces are perceived to be younger. Further, the 'distance' from the average is a strong indicator of the apparent age of the person. The regions where a given face shows a large difference in shape from a shape-averaged face when further exaggerated, results in a caricature [190, 191]. Young faces exhibit distinct growth-related anthropometric trends. Anthropometric variations in adults are distinctive to a lesser degree than in children, but nevertheless they do exhibit drifts in facial features surrounding the mouth, eyebrows etc. This is illustrated in figure 7.4 where distinct geometric changes can be observed as a person ages.

To develop appropriate statistical inference methodologies, one needs to understand a) what is the space of these geometric landmarks, and b) What are the appropriate statistical models and distance metrics in this space. We show that an affine-invariant representation of facial landmark geometry can be analytically modeled as a Grassmann manifold. Then, we discuss how to measure distances and model transformations in this space. Further, we describe the warping process of one face to another by a smooth geodesic flow on the Grassmann manifold. Then, these warping parameters are shown to contain age-specific information which can prove useful for estimating the apparent age of a person.



(a) Age 2 (b) Age 10 (c) Age 14 (d) Age 18 (e) Age 29 (f) Age 43

Figure 7.4: Facial geometric variation across ages. Samples shown correspond to individual 2 from the FG-net dataset.

Related Work: Research in modeling aging can be divided into two main classes – physics-based models and data-driven models. The first class concerns itself with computational models to describe the physical process of aging. Examples include the works of Pittenger and Shaw [192] who studied facial growth as a viscalelastic event defined on the craniofacial complex. Mark et al. [193] studied geometric invariants that characterize cardioidal strain transformations and their relation to perception of growth. Todd et al. [194] treated the human head as a fluid filled spherical object and proposed the revised cardioidal strain model to account for craniofacial growth. More recently, Narayanan and Chellappa [195] applied these models in conjunction with anthropometric data to identify different growth parameters for different parts of the face. Physics-based approaches such as these have mostly found use in synthesis applications such as age progression and regression, where it is important to synthesize realistic younger or older looking faces from a given face.

In the data driven approaches, modeling of age progression is typically done by estimating functional forms of the aging process or learning classifiers from training data. Examples include the work of [196], who proposed methods to classify face images as that of babies, young adults and senior adults. Facial anthropometric measurements were used to classify faces as babies and adults. Adult faces were further classified into young or senior adults using texture analysis. Ramanathan and Chellappa [197] proposed a Bayesian age-difference classifier built on a probabilistic eigenspaces framework to perform face verification across age progression. Several regression-based methods have been proposed to estimate the perceived age of a face from images. Lanitis et al. [198, 199] constructed an aging function based on a parametric model for human faces and performed automatic age progression, age estimation, face recognition across aging. Fu et al. [200] combined dimensionality reduction methods such as PCA, LLE, LPP, OLPP etc with regression. Guo et. al. [201] proposed robust regression followed by local adjustments for age estimation and showed that local adjustments improve performance. All these approaches mainly differ in the features used and variations in the choice of regression methods.

7.4.1 Modeling the Geometry of the Face

Representations and recognition of shapes is a well understood field [8, 202]. In this chapter, we are interested in the 2D geometry of facial landmarks. The shape observed in an image of a face is a perspective projection of the 3D locations of the landmarks. Standard approaches to describe shapes involve extracting features such as moments [19], shape context [203] etc. These approaches extract coarse features which correspond to the average properties of the shape. These approaches are particularly useful when landmarks on shapes cannot be reliably located across different images or do not necessarily correspond to physically meaningful parts of the object. However, in the case of faces, there exist physically meaningful locations such as eyes, mouth, nose etc which can be reliably located on most faces. This suggests the use of a representation that exploits the entire information offered by the location of landmarks instead of relying on coarse features. In several face recognition tasks, the locations of the landmarks have been shown to be extremely informative [204, 205]. There exist several automatic methods to locate facial landmarks which work well on constrained images such as passport photos. It is in constrained scenarios such as these that the methods proposed here are applicable.

The drawback of using the locations of landmarks is that they are sensitive to transformations such as affine transforms, view changes etc. In order to account for this, shape theory studies the equivalent class of all configurations that can be obtained by a specific transformation (e.g. linear, affine, projective) from a given base shape. A shape is represented by a set of landmark points, given by a $m \times 2$ matrix $L = [(x_1, y_1); (x_2, y_2); \ldots; (x_m, y_m)]$, of the set of m landmarks of the centered shape. The *shape space* of this base shape is the set of equivalent configurations that are obtained by transforming the base shape by an appropriate spatial transformation. For example, the set of all affine transformations forms the *affine shape space* of that base shape.

The *affine shape space* [166] [185] is very important because small changes in camera location or change in the pose of the subject can be approximated well as affine transformations on the original base shape. The affine transforms of the shape can be derived from the base shape simply by multiplying the shape matrix L by a 2×2 full rank matrix on the right. For example, let A be a $2{\times}2$ affine transformation matrix i.e. $A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$. Then, all affine transforms of the base shape L_{base} can be expressed as $L_{affine}(A) = L_{base} * A^T$. Note that, multiplication by a full-rank matrix on the right preserves the column-space of the matrix L_{base}. Thus, the 2D subspace of $\mathbb{R}^m$ spanned by the columns of the matrix L_{base} is an *affine-invariant* representation of the shape. i.e. $span(L_{base})$ is invariant to affine transforms of the shape. Subspaces such as these can be identified as points on a Grassmann manifold. We now define the Grassmann manifold.

As already known, the Grassmann manifold $G_{k,m}$ is the space whose points are *k-planes* or k-dimensional hyperplanes (containing the origin) in $\mathbb{R}^m$.

7.4.2 Aging on the Manifold

The basic premise of our work is that the perceived age will show a functional dependence on the geometry of the face. Given several faces X_i, along with their respective ages y_i, the goal is to estimate a function $y = f(X)$ that can explain the aging patterns. This can be formulated as a regression problem. Regression problems are mostly studied in Euclidean vector spaces and there exist a wealth of methods for robust regression. Regression has been applied to age-estimation tasks before by assuming that faces, or features extracted from faces lie in a Euclidean space such as in [198, 200, 201]. However, for geometric features considered here, we need to solve the regression problem on the Grassmann manifold. The Grassmann manifold is not a vector space, thus precluding the use of classical techniques. We explore two distinct approaches for solving the regression problem – a differential geometric and a kernel-based machine learning approach. In the differential geometric approach, all points on the manifold are projected onto the tangent plane at a mean-point and standard vector-space methods are applied on the tangent plane, which is a Euclidean vector space [157]. This approach can also be viewed as performing regression on the transformation required to warp an 'average-face' to a given face. Thus this approach is motivated by [189]. Given a face and an 'average-face', we compute the directional velocity vector with which the average-face should move on the manifold so that it reaches the given face in unit time. This velocity vector is then used as an age signature.

On the other hand, kernel methods offer an alternative approach for solving such problems. The assumption is that the kernel provides a mapping into a higher-dimensional Euclidean space, thereby implicitly enabling standard vector space approaches on the higher dimensional space. For the case of the Grassmann manifold, there is an elegant interpretation of points as vectors via the so called Cauchy-Binet embedding [206], which arises from the Cauchy-Binet kernel. However, the differential geometric approach enables a far richer class of statistical estimation techniques to be deployed, whereas the kernel-based method is limited in applicability to those algorithms that admit a kernel interpretation. Since there exist kernel versions of regression algorithms (Ridge, SVM, RVM etc) we shall see in experiments that both approaches offer comparable performance on age-estimation tasks.

7.4.3 Differential Geometric methods for Aging

Given an ‘average-face’ or a shape-normalized face, we would like to quantify the deformation that can warp the average to any given face. We can conveniently model these deformations via geodesics on the Grassmann manifold. We parameterize the deformation between two shapes on the Grassmann manifold as the velocity with which a point on the manifold should move in order to reach the second point in unit-time. We have already discussed in chapter 6 how to compute these parameters. We shall use these velocity parameters as aging signatures. Once these velocity parameters are computed, we can *flatten* them to a vectorial form. Once this is done, we can apply standard Euclidean space regression methods on the velocity parameters. But first, we need to specify what the ‘shape-normalized’ or ‘average’ face is and how to compute it.

The shape-normalized face can be a generic face that is obtained by averaging the shapes of several faces. In the current setup, we need to find the mean of a set of subspaces, or the mean of a set of points on the Grassmann manifold. The problem can be solved by computing the Karcher mean.

7.4.4 Kernel Methods

To discuss how to solve the function estimation problem $y = f(X)$ on the Grassmann manifold using kernels, we first define the Cauchy-Binet embedding. This embedding maps points from the Grassmann manifold to a large dimensional vector space. The Cauchy-Binet embedding [206] is a mapping from $G_{k,m}$ to $\mathbb{R}^n$, where $n = \binom{m}{k}$. The mapping is understood as follows. Let $S \in G_{k,m}$ and Y be an $m \times k$ tall-thin orthonormal matrix such that $span(Y) = S$. Let s be a subset of $\{1, \ldots, m\}$ with k elements $s = \{r_1, \ldots, r_k\}$, and $Y^{(s)}$ be the $k \times k$ matrix whose row indices are given by the vector s. Then, there are $n = \binom{m}{k}$ combinations for the vector s. Let these combinations be given by $s_1, \ldots s_n$. Then, the Cauchy-Binet embedding is a mapping $\Phi : G_{k,m} \to \mathbb{R}^n$ where $\Phi(S) = [det(Y^{(s_1)}), \ldots det(Y^{(s_n)})]$, where $span(Y) = S, Y^T Y = I$. Note that this embedding is independent of the choice of Y as long as it is orthonormal and satisfies $span(Y) = S$. It can be shown that dot-products in the Cauchy-Binet space can be evaluated via a Mercer kernel on the Grassmann manifold [163]. Specifically, if S_1 and S_2 are two subspaces with orthonormal basis Y_1 and Y_2, then

$$\Phi(S_1)^T \Phi(S_2) = det(Y_1^T Y_2)^2 \tag{7.2}$$

Let us denote by $K_{CB}(Y_1, Y_2) = det(Y_1^T Y_2)^2$ the Cauchy-Binet kernel on the Grassmann manifold. This dot-product interpretation makes it feasible to implement standard regression algorithms such as Ridge Regression, SVM-based regression etc. via the 'kernel-trick' on the Grassmann manifold. Further, standard vector-space kernels such as the polynomial, radial basis and sigmoid can be rewritten in terms of the Cauchy-Binet kernel on the Grassmann manifold. As an example, the polynomial kernel in the CB space can be rewritten as

$$\begin{aligned} K_{poly}(\Phi(S_1), \Phi(S_2)) &= (1 + \gamma \Phi(S_1)^T \Phi(S_2))^d \\ &= (1 + \gamma K_{CB}(Y_1, Y_2))^d \end{aligned}$$

Similarly the RBF kernel on the CB-space can be rewritten as

$$\begin{aligned} K_{RBF}(\Phi(S_1), \Phi(S_2)) &= exp^{-\gamma(\Phi(S_1) - \Phi(S_2))^T (\Phi(S_1) - \Phi(S_2))} \\ &= exp^{-\gamma(K_{CB}(Y_1, Y_1) + K_{CB}(Y_2, Y_2) - 2K_{CB}(Y_1, Y_2))} \end{aligned}$$

This gives rise to a new family of kernels on the Grassmann manifold which can also be shown to be Mercer kernels. In practice, we need not compute the large $\binom{m}{k}$ dimensional embedding itself. As shown above, dot products and Mercer Kernels in the CB space can be evaluated using the Cauchy-Binet kernel on the Grassmann manifold. This makes this approach computationally efficient and flexible in the choice of the regression method.

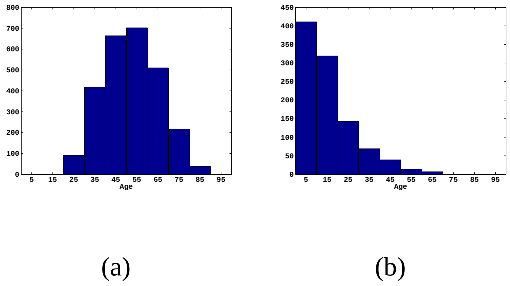
7.4.5 Experiments

We evaluate the strength of the Riemannian framework on age-estimation tasks on two datasets. The first dataset is the Passport dataset [195] which contains mostly adult faces. The age distribution of the faces is shown figure 7.5(a). In this dataset, we used 47 fiducial points marked manually. The second is the publicly available FG-Net dataset [207], which contains both adult and young faces. The distribution of ages is shown in figure 7.5(b). Some sample images from this dataset are shown in figure 7.6. For this dataset, 68 fiducial points are available with each face. Both datasets exhibit wide variations in age ranges of the faces, thus testing the framework on both young and adult faces.

Given a face and its landmarks, we extract the tall-thin Grassmann Procrustes representation using standard SVD methods. Given the matrix of landmarks L we center it and compute its SVD $L = U\Sigma V^T$. The affine-invariant Grassmann Procrustes representation of L is then given by $Y_L = U$. Now given several examples Y_i with corresponding ages y_i, we want to estimate the aging-function $y = f(Y)$ in a robust manner. Given a training set, we compute the shape-normalized face μ as described in section 6.3. For each face in the training set Y_i, we compute the aging signatures using the flattened warping parameters A_i as described in section 6.3.2. Then, we estimate the aging function $y_i = f(A_i)$ using standard regression methods. Further, we also use the Cauchy-Binet kernel on the Grassmann manifold to perform kernel regression.

For performing regression using the Cauchy-Binet kernel, we use ε-SVMs, RVMs, and ridge regression (regularized linear least-squares). We use the ε-SVM with $\varepsilon = 0.02$, the cost parameter $C = 1000$, and regularization parameter $\lambda = 10^{-6}$. For RVMs, there are no parameters to tune except the number of iterations for the RVM optimization routine. We set this to 50 iterations. For ridge regression, the regularization parameter λ is chosen to be $\lambda = 10^{-6}$. To perform regression using the velocity vectors, we use the same regression methods, but with the polynomial kernel of degree 2 i.e. $K(A_1, A_2) = (1 + A_1^T A_2)^2$, where A_1 and A_2 are the vectorial forms of the velocity matrices.

Two metrics have been proposed in literature for quantifying the performance of



(a) (b)

Figure 7.5: Distribution of ages in (a) Passport, (b) FG-Net dataset

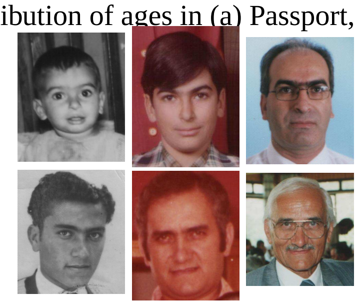


Figure 7.6: Sample images from the FG-Net dataset

age-estimation algorithms. The first criterion measures the mean absolute error (MAE) in age-estimation across the entire dataset. i.e. $MAE = \frac{1}{N}\sum_i |l_i - \hat{l}_i|$, where N is the size of the dataset, l_i is the true age of the i^{th} person being tested, and $\hat{l}_i$ is the assigned age. The second metric is the cumulative match score. The cumulative score is defined as $CS(j) = N_{e \leq j}/N \times 100\%$, where $N_{e \leq j}$ is the number of test-images on which the absolute error in age-estimation is within j years.

Passport dataset: In the passport dataset, we performed a leave-one-out testing in which the regression algorithms are trained on the entire dataset except one sample on which the testing is done. The MAE results using various algorithms is summarized in table 7.3. The SVM and RVM based regression are seen to perform better than the simpler ridge-regression. We see that the lowest MAE was achieved by using velocity vectors with RVM regression and it is 8.84 years. Considering that the average age in this dataset is 42 years, the obtained MAE is quite encouraging. Figure 7.7(a) shows the cumulative score curves as a function of the error-level using the Cauchy-Binet kernel with SVM, RVM, and ridge regression. We see that about 85% of the faces are classified within 15 years of their true age. Similar results are obtained using the velocity parameters as shown in figure 7.7(b).

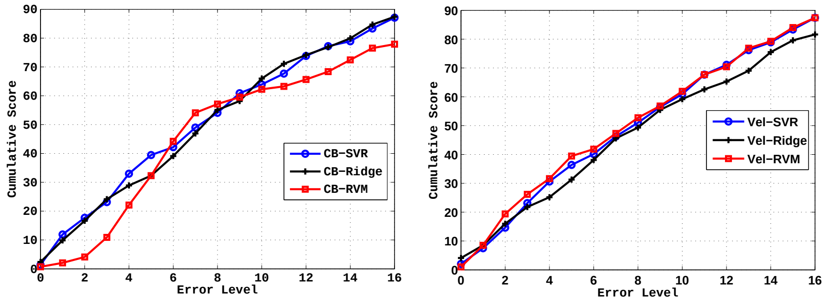
FG-Net dataset: For the FG-Net dataset, we performed a leave-one-person-out testing as has recently been suggested [208]. In this mode, all images corresponding to the same person are used for testing and the remaining images are used for training. The results of the proposed framework on the FG-Net dataset is shown in table 7.4. The lowest MAE was obtained by using SVM + Cauchy-Binet kernel, and also by SVM + polynomial kernel on velocity vectors. MAE in both these cases was 5.89 years. The table also shows a comparison with other recently published methods. The cumulative scores of the proposed methods are shown in figures 7.8(a) and 7.8(b). We see that more than 90% of the faces are classified within 15 years of their true age.

We see that the proposed algorithms are comparable to the state-of-the-art methods and even outperform most of them except RUN1 [209] (MAE = 5.78) and LARR [201] (MAE = 5.07). The work of [209] deals primarily with a new regression method that can deal with uncertain labels. The features used are cropped face images. Our approach is flexible in the choice of regression method, and we can utilize the method of [209] as well. Here, we show that accurate characterization of geometry yields comparable results even with simple, unoptimized regression methods. In [201], a suite of dimensionality reduction approaches – PCA, LLE, LPP etc – etc are empirically evaluated. It was found that Orthogonal LPP (OLPP) performs best in age-estimation tasks. However, there is no principled argument on why this is so. Further, the age estimation results are locally adjusted around the estimated age to tweak estimation results. The proposed method can be combined with the features of [201] and also the suggested local adjustment, but as stated in the introduction the focus of the current work is not to outperform these methods in age-estimation, but to show how a principled method to model the geometric variations of faces can provide comparable results.

It is generally accepted that geometric variations are more pronounced in children than adults. This might explain why the age-estimation error in the passport dataset is larger than in the FG-Net dataset. Further, the published methods compared in table 7.4 rely on some form of joint structure and texture information such as using the whole images themselves, or using Active Appearance models. Inspite of this obvious handicap, it is interesting to note that accurate characterization of geometry provides better results

Method	**Ridge Regression**	**SVM**	**RVM**
Cauchy-Binet	12.49	9.03	9.85
Warping Velocities	15.72	9.78	8.84

Table 7.3: Mean-Absolute Errors using different regression methods using the Cauchy-Binet embedding and the warping velocities on the Passport dataset.



(a)

(b)

Figure 7.7: Passport data Cumulative scores using (a) Cauchy-Binet kernel, (b) velocity parameters with polynomial kernel.

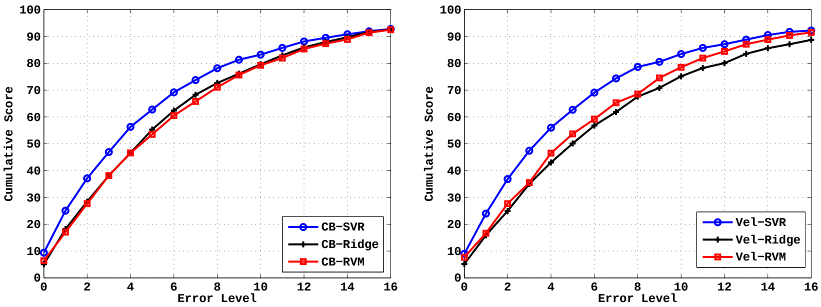
in many cases. This does not downplay the role of texture in age-perception, and the proposed methods may be further combined with textural features.

7.5 Conclusion

In this chapter we have presented a comprehensive set of tools and algorithms for statistical computing on the Grassmann manifold. We have shown that the Grassmann manifold arises naturally in many important applications in computer vision. We have presented statistical modeling methods that are derived from the Riemannian geometry of the manifold. We have also presented Procrustes representation and non-parametric density estimation methods which offer an alternative to the Riemannian approaches. As seen in experiments the Riemannian geometric approaches tend to perform uniformly well over several experiments. However, the performance of the non-parametric approach is

	Method	**MAE**
Cauchy-Binet	Ridge	6.60
	SVM	5.89
	RVM	6.86
Warping Velocities	Ridge	7.57
	SVM	5.89
	RVM	6.69
Other Algorithms	AAS [199]	14.83
	WAS [208]	8.06
	Ages [208]	6.77
	$Ages_{lda}$ [208]	6.22
	QM [198]	6.55
	MLP [198]	6.98
	RUN1 [209]	5.78
	LARR [201]	5.07

Table 7.4: Comparison of Mean-Absolute Errors using proposed methods with state-of-the-art on the FG-Net dataset.



(a)

(b)

Figure 7.8: FG-Net data Cumulative scores using (a) Cauchy-Binet kernel, (b) velocity parameters with polynomial kernel.

strongly tied to the choice of the kernel-width. With a good choice of the kernel-width parameter, it can outperform the wrapped-normal approach. This is because non-parametric methods can provide better fit to the data than imposing a parametric form. Further, the computational cost involved in classification using the non-parametric method is quite high as it involves computing distances to every point in the training dataset. Whereas for the wrapped normal case, classification is much faster. Further, the geometric methods presented in this chapter offer principled solutions to several interesting problems such as smoothing, prediction, and time-sequence modeling on the manifold.

Chapter 8

Directions for Future Work

The problems addressed in this thesis and the methods proposed to solve them lead us to several interesting future research directions. In this chapter we outline a few directions for future research work.

8.1 Indexing the Manifold: Applications to Database Searching

In the preceding discussion, we have represented the members of a database – e.g. actions or shapes – as points on the Grassmann manifold. We always assumed that the dataset was small enough to ignore the complexity involved in nearest neighbor searching. When the size of the dataset is extremely large, searching for the most similar element to a given query can be prohibitively expensive if done in a brute-force linear fashion. Thus, for large datasets, it is necessary to index and organize the data in a form that enables fast-lookup. Two of the most commonly used approaches for organizing a database are based on a) Space-partitioning, and b) Clustering.

Space partitioning methods divide the data-space into distinct regions. The partitioning is done so that each region is made up of 'similar' data points. For example, if the input data lies on a sphere in $\mathbb{R}^3$, a natural way of partitioning the data-space would be in terms of the longitude and latitude of the points. The longitude and latitude are an 'index' into the manifold. For complex manifolds, this process is more commonly known as 'Charting' the manifold. Space-partitioning is well understood for Euclidean spaces and is known to work efficiently for low-dimensions. For high-dimensional spaces, space partitioning methods are known to perform as poorly as simple linear searches. This is due to the fact that in high-dimensional spaces, the number of regions required to cover the entire space grows in an exponential fashion with the dimension, hence requiring an exponentially larger number of similarity checks [210]. Moreover, due to the sparse nature of high-dimensional spaces most of the regions are empty, and thus do not add to the retrieval results. By careful analysis of the underlying manifold on which the data lies, more efficient space partitioning methods can be devised with fewer number of partitions, which are also more populated. Future research would focus on mathematical representations and methods that would enable charting the Grassmann and the Stiefel manifold for fast similarity search applications.

Space partitioning methods are directly related to the geometry of the manifold, and are insensitive to the naturally embedded clusters in a given dataset. By specifically discovering the clusters inherent within the given dataset, one can design more efficient indexing methods rather than charting the entire manifold. Clustering based methods rely on a notion of 'distance' or 'similarity' in the data space. Designing the right clustering algorithm requires both a notion of a natural distance metric on the manifold (geodesics etc) and algorithms for finding clusters that are consistent with the geometry of the manifold. Standard clustering methods such as k-means are designed for euclidean spaces and thus are not directly applicable. Future research would focus on deriving appropriate clustering algorithms that are adapted to the structures of our Manifolds of interest – Grassmann and Stiefel. Further, hierarchical clustering methods such as dendrograms can be employed to organize the data in a hierarchical fashion, where the lower-levels of the hierarchy encode coarse similarity relations and the higher-levels provide successive levels of refinement to the similarity search.

8.2 Separating Style and Content

Visual patterns can be viewed as characterized by two underlying attributes – their style and their content. Traditional pattern recognition approaches attempt to build models for the content of a pattern without specific regard to the style. As an example, in computer generated text, the alphabet 'a' may be rendered in one of several font styles. The style of the font does not change the content itself. However, the style reflects itself in the wide variations of observable features such as corners, edges etc. Similarly in action recognition, the same action such as walking may be performed in several different styles. A choice of features that is invariant to stylistic changes does not usually exist.

However, one can exploit the geometry of the underlying feature space to learn models of stylistic variations and the individual mappings between style and content. We propose to study the problem of separating the style and content of human actions by exploiting the geometry of the Grassmann manifold.

8.3 Geometric Subspace Dynamics

So far, we have treated points on the Grassmann manifold in a ‘static’ manner. We explored statistical modeling methods and distance metrics on the manifold. This naturally leads us to extend these techniques to situations where we are interested in modeling the dynamics of a process on the manifold. This requires accurate modeling of the temporal dependence in a way that is consistent with the geometry of the manifold. As a specific example, consider the problem of shape sequence modeling. A 2-D shape is usually represented in the form of a few landmarks on its contour. The affine-shape space of a shape is the space of all possible affine warps of a given shape. Affine shape spaces can be identified as points on the Grassmann manifold. Preliminary experiments in Chapter 6 have shown promising results on affine-invariant shape classification from still images. Future work would focus on extending this framework to shape sequences. A shape sequence can be modeled as a trajectory on the Grassmann manifold. Parametric and non-parametric methods will be extended to model the evolution of the shape on the Grassmann manifold. Non-parametric methods such as Dynamic Time Warping only require a measure of distance between two points on the manifold, hence they are easily applicable to shape sequence matching on the Grassmann manifold. Parametric methods such as HMMs can also be suitably extended by a careful derivation of each of the components of the model. HMMs consist of two major components – the hidden state space and the observation model. The hidden state space in the current context would consist of a discrete set of points on the Grassmann manifold which can be estimated by clustering algorithms which will be developed as proposed in section 8.1. The observation model would consist of parametric probability density functions on the Grassmann manifold such as the matrix Bingham distribution as described in chapter 6. This would then allow concise parametric models to represent shape sequences that are also consistent with the geometry of the manifold.

8.4 Online Visual Learning

Applications involving dynamic environments and autonomous agents such as a mobile robot navigating through an unknown space cannot be represented by static models. In such applications it is important to adapt models, that have been learnt offline, according to new observations in an online fashion. In the object recognition domain, one common approach is to perform incremental PCA to dynamically learn a better representational model as the appearance of the target dynamically changes as in [179]. Incremental PCA has also been used to recognize abnormalities in the visual field of a robot as in [180]. In an unrelated domain, the theory of subspace tracking on the Grassmann manifold [211] has been developed for array signal processing applications. Since PCA basis vectors represent a subspace which is identified by a point on the Grassmann manifold, subspace tracking can be applied for online visual learning applications. By tracking the evolution of the appearance subspace or the model parameters of an ARMA model on the Grassmann manifold, one can identify points of large changes in the trajectory which can potentially be used for anomaly detection also.

8.5 Anomaly Detection

Detecting anomalies in the field of view of a camera is an important problem with several applications in computer vision and robotics. One of the frequently used strategies for detecting anomalies is based on outlier detection. The dynamic nature of patterns in a the field of view of a stationary or a moving camera can be well described by a sequence of time-varying dynamical systems. A simpler approximation would be to represent only the coarse observation subspaces and model the dynamic nature of patterns are time-varying subspaces. This model can be used to detect anomalies in the field of veiew of the camera and hence can be used for anomaly detection.

Bibliography

[1] A. Veeraraghavan, R. Chellappa, and A. K. Roy-Chowdhury, “The function space of an activity,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 959–968, 2006.

[2] D. Weinland, R. Ronfard, and E. Boyer, “Free viewpoint action recognition using motion history volumes,” *Computer Vision and Image Understanding*, vol. 104, no. 2, pp. 249–257, 2006.

[3] ——, “Free viewpoint action recognition using motion history volumes,” *Computer Vision and Image Understanding*, vol. 104, no. 2, pp. 249–257, 2006.

[4] A. Veeraraghavan, A. Roy-Chowdhury, and R. Chellappa, “Matching shape sequences in video with an application to human movement analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 12, pp. 1896–1909, 2005.

[5] N. Vaswani, A. K. Roy-Chowdhury, and R. Chellappa, “”shape activity”: a continuous-state HMM for moving/deforming shapes with application to abnormal activity detection,” *IEEE Transactions on Image Processing*, vol. 14, no. 10, pp. 1603–1616, 2005.

[6] Y. Wang, H. Jiang, M. S. Drew, Z.-N. Li, and G. Mori, “Unsupervised Discovery of Action Classes,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[7] H. Zhong, J. Shi, and M. Visontai, “Detecting unusual activity in video,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 819–826, 2004.

[8] D. G. Kendall, “Shape manifolds, procrustean metrics and complex projective spaces,” *Bulletin of London Mathematical society*, vol. 16, pp. 81–121, 1984.

[9] E. Begelfor and M. Werman, “Affine invariance revisited,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[10] J. K. Aggarwal and Q. Cai, “Human motion analysis: A review,” *Computer Vision and Image Understanding*, vol. 73, no. 3, pp. 428–440, 1999.

[11] C. Cedras and M. Shah, “Motion-based recognition: A survey,” *Image and Vision Computing*, vol. 13, no. 2, pp. 129–155, 1995.

[12] D. M. Gavrila, “The visual analysis of human movement: a survey,” *Computer Vision and Image Understanding*, vol. 73, no. 1, pp. 82–98, 1999.

[13] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," *Computer Vision and Image Understanding*, vol. 104, no. 2, pp. 90–126, 2006.

[14] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys*, vol. 38, no. 4, 2006.

[15] G. Johansson, "Visual perception of biological motion and a model for its analysis," *Perception and Psychophysics*, vol. 14, no. 2, pp. 201–211, 1973.

[16] S. S. Beauchemin and J. L. Barron, "The computation of optical flow," *ACM Computing Surveys*, vol. 27, no. 3, pp. 433–466, 1995.

[17] T. Huang, D. Koller, J. Malik, G. H. Ogasawara, B. Rao, S. J. Russell, and J. Weber, "Automatic symbolic traffic scene analysis using belief networks," *National Conference on Artificial Intelligence*, pp. 966–972, 1994.

[18] A. M. Elgammal, D. Harwood, and L. S. Davis, "Non-parametric model for background subtraction," *Proceedings of IEEE European Conference on Computer Vision*, pp. 751–767, 2000.

[19] M.-K. Hu, "Visual pattern recognition by moment invariants," *IEEE Transactions on Information Theory*, vol. 8, no. 2, pp. 179–187, 1962.

[20] H. Freeman, "On the encoding of arbitrary geometric configurations," *IRE Transactions on Electronic Computers*, vol. 10, no. 2, pp. 260–268, 1961.

[21] H. Blum and R. N. Nagel, "Shape description using weighted symmetric axis features," *Pattern Recognition*, vol. 10, no. 3, pp. 167–180, 1978.

[22] A. Bissacco, P. Saisan, and S. Soatto, "Gait recognition using dynamic affine invariants," *International Symposium on Mathematical Theory of Networks and Systems*, 2004.

[23] R. Polana and R. Nelson, "Low level recognition of human motion (or how to get your man without finding his body parts)," *Proceedings of the IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, pp. 77–82, 1994.

[24] S. M. Seitz and C. R. Dyer, "View-invariant analysis of cyclic motion," *International Journal of Computer Vision*, vol. 25, no. 3, pp. 231–251, 1997.

[25] A. F. Bobick and J. W. Davis, "The recognition of human movement using temporal templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 3, pp. 257–267, 2001.

[26] A. F. Bobick, "Movement, activity, and action: The role of knowledge in the perception of motion," *Philosophical Transactions of the Royal Society of London B*, vol. 352, pp. 1257–1265, 1997.

[27] O. Chomat and J. L. Crowley, “Probabilistic recognition of activity using local appearance,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 02, pp. 104–109, 1999.

[28] L. Zelnik-Manor and M. Irani, “Event-based analysis of video,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 123–130, 2001.

[29] I. Laptev and T. Lindeberg, “Space-time interest points,” *Proceedings of IEEE International Conference on Computer Vision*, 2003.

[30] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, “Behavior recognition via sparse spatio-temporal features,” *IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, 2005.

[31] J. C. Niebles, H. Wang, and L. Fei Fei, “Unsupervised learning of human action categories using spatial-temporal words,” *British Machine Vision Conference*, 2006.

[32] T. F. Syeda-Mahmood, M. Vasilescu, and S. Sethi, “Recognizing action events from multiple viewpoints,” *IEEE Workshop on Detection and Recognition of Events in Video*, pp. 64–72, 2001.

[33] A. Yilmaz and M. Shah, “Actions sketch: A novel action representation,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 984–989, 2005.

[34] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri, “Actions as space-time shapes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2247–2253, 2007.

[35] S. T. Roweis and L. K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[36] M. Belkin and P. Niyogi, “Laplacian eigenmaps and spectral techniques for embedding and clustering,” *Advances in Neural Information Processing Systems*, pp. 585–591, 2001.

[37] J. B. Tenenbaum, V. D. Silva, and J. C. Langford, “A global geometric framework for nonlinear dimensionality reduction.” *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[38] Y. Yacoob and M. J. Black, “Parameterized modeling and recognition of activities,” *Computer Vision and Image Understanding*, vol. 73, no. 2, pp. 232–247, 1999.

[39] A. M. Elgammal and C. S. Lee, “Inferring 3D body pose from silhouettes using activity manifold learning,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 681–688, 2004.

[40] R. Pless, “Image spaces and video trajectories: Using isomap to explore video sequences,” *Proceedings of IEEE International Conference on Computer Vision*, pp. 1433–1440, 2003.

[41] L. R. Rabiner, “A tutorial on hidden markov models and selected applications in speech recognition,” *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[42] J. Yamato, J. Ohya, and K. Ishii, “Recognizing human action in time-sequential images using hidden markov model,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 379–385, 1992.

[43] A. D. Wilson and A. F. Bobick, “Learning visual behavior for gesture analysis,” *Proceedings of the International Symposium on Computer Vision*, pp. 229–234, 1995.

[44] T. Starner, J. Weaver, and A. Pentland, “Real-time american sign language recognition using desk and wearable computer based video,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 12, pp. 1371–1375, 1998.

[45] J. M. Siskind and Q. Morris, “A maximum-likelihood approach to visual event classification,” *Proceedings of IEEE European Conference on Computer Vision*, pp. 347–360, 1996.

[46] A. Kale, A. Sundaresan, A. N. Rajagopalan, N. P. Cuntoor, A. K. Roy-Chowdhury, V. Kruger, and R. Chellappa, “Identification of humans using gait,” *IEEE Transactions on Image Processing*, vol. 13, no. 9, pp. 1163–1173, 2004.

[47] Z. Liu and S. Sarkar, “Improved gait recognition by gait dynamics normalization,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 6, pp. 863–876, 2006.

[48] M. Brand, N. Oliver, and A. Pentland, “Coupled hidden markov models for complex action recognition,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 994–999, 1997.

[49] D. J. Moore, I. A. Essa, and M. H. Hayes, “Exploiting human actions and object context for recognition tasks,” *Proceedings of IEEE International Conference on Computer Vision*, pp. 80–86, 1999.

[50] S. Hongeng and R. Nevatia, “Large-scale event detection using semi-hidden markov models,” *Proceedings of IEEE International Conference on Computer Vision*, pp. 1455–1462, 2003.

[51] N. P. Cuntoor and R. Chellappa, “Mixed-state models for nonstationary multiobject activities,” *EURASIP Journal of Applied Signal Processing*, vol. 2007, no. 1, pp. 106–119, 2007.

[52] J. Schlenzig, E. Hunter, and R. Jain, “Recursive identification of gesture inputs using hidden markov models,” *Proceedings of the Second IEEE Workshop on Applications of Computer Vision*, pp. 187–194, 1994.

[53] A. Bissacco, A. Chiuso, Y. Ma, and S. Soatto, “Recognition of human gaits,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 52–57, 2001.

[54] M. C. Mazzaro, M. Sznaier, and O. Camps, “A model (in)validation approach to gait classification,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 11, pp. 1820–1825, 2005.

[55] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto, “Dynamic textures,” *International Journal of Computer Vision*, vol. 51, no. 2, pp. 91–109, 2003.

[56] A. B. Chan and N. Vasconcelos, “Classifying video with kernel dynamic textures,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[57] P. V. Overschee and B. D. Moor, “Subspace algorithms for the stochastic identification problem,” *Automatica*, vol. 29, no. 3, pp. 649–660, 1993.

[58] Z. Ghahramani and G. E. Hinton, “Parameter estimation for linear dynamical systems,” Department of Computer Science, University of Toronto, Technical Report, Tech. Rep. CRG-TR-96-2, 1996.

[59] L. Ljung, Ed., *System identification (2nd ed.): theory for the user*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1999.

[60] K. D. Cock and B. D. Moor, “Subspace angles between arma models,” *Systems and Control Letters*, vol. 46, pp. 265–270, 2002.

[61] C. Bregler, “Learning and recognizing human dynamics in video sequences,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, p. 568, 1997.

[62] B. North, A. Blake, M. Isard, and J. Rittscher, “Learning and classification of complex dynamics,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 9, pp. 1016–1034, 2000.

[63] V. Pavlovic, J. M. Rehg, and J. MacCormick, “Learning switching linear models of human motion,” *Advances in Neural Information Processing Systems*, pp. 981–987, 2000.

[64] D. Del Vecchio, R. M. Murray, and P. Perona, “Primitives for human motion: A dynamical approach,” *Proceedings of IFAC World Congress*, 2002.

[65] S. M. Oh, J. M. Rehg, T. R. Balch, and F. Dellaert, “Data-driven mcmc for learning and inference in switching linear dynamic systems,” *National Conference on Artificial Intelligence*, pp. 944–949, 2005.

[66] R. Vidal, A. Chiuso, and S. Soatto, "Observability and identifiability of jump linear systems," *Proceedings of IEEE Conference on Decision and Control*, pp. 3614–3619, 2002.

[67] Y. Sheikh, M. Sheikh, and M. Shah, "Exploring the space of a human action," *Proceedings of IEEE International Conference on Computer Vision*, pp. 144–149, 2005.

[68] T. J. Darrell, I. A. Essa, and A. P. Pentland, "Task-specific gesture analysis in real-time using interpolated views," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 12, pp. 1236–1242, 1996.

[69] C. Rao and M. Shah, "View-invariance in action recognition," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 316–322, 2001.

[70] V. Parameswaran and R. Chellappa, "View invariants for human action recognition," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 613–619, 2003.

[71] ——, "View invariance for human action recognition," *International Journal of Computer Vision*, vol. 66, no. 1, 2006.

[72] A. F. Bobick and A. D. Wilson, "A state-based technique for the summarization and recognition of gesture," *Proceedings of IEEE International Conference on Computer Vision*, pp. 382–388, 1995.

[73] J. Hoey and J. J. Little, "Representation and recognition of complex human motion," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1752–1759, 2000.

[74] P. K. Turaga, A. Veeraraghavan, and R. Chellappa, "From videos to verbs: Mining videos for activities using a cascade of dynamical systems," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[75] K. Takahashi, S. Seki, E. Kojima, and R. Oka, "Recognition of dexterous manipulations from time-varying images," *Proceedings IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, pp. 23–28, 1994.

[76] M. A. Giese and T. Poggio, "Morphable models for the analysis and synthesis of complex motion patterns," *International Journal of Computer Vision*, vol. 38, no. 1, pp. 59–73, 2000.

[77] C. Rao, M. Shah, and T. Syeda-Mahmood, "Invariance in motion analysis of videos," *Proceedings of the eleventh ACM International Conference on Multimedia*, pp. 518–527, 2003.

[78] A. Gritai, Y. Sheikh, and M. Shah, "On the use of anthropometry in the invariant analysis of human actions," *IEEE International Conference on Pattern Recognition*, pp. 923–926, 2004.

[79] J. Pearl, *Probabilistic reasoning in intelligent systems: networks of plausible inference*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1988.

[80] M. I. Jordan, *Learning in Graphical Models*. The MIT Press, 1998.

[81] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, “Factor graphs and the sum-product algorithm,” *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[82] N. Friedman and D. Koller, “Being Bayesian about Bayesian network structure: A Bayesian approach to structure discovery in Bayesian networks.” *Machine Learning*, vol. 50, no. 1–2, pp. 95–125, 2003.

[83] R. David and H. Alla, “Petri nets for Modeling of Dynamic Systems A Survey,” *Automatica*, vol. 30, no. 2, pp. 175–202, 1994.

[84] T. Murata, “Petri nets: Properties, Analysis and Applications,” *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541–580, 1989.

[85] C. Castel, L. Chaudron, and C. Tessier, “What is going on? A High-Level Interpretation of a Sequence of Images,” *ECCV Workshop on Conceptual Descriptions from Images*, 1996.

[86] N. Ghanem, D. DeMenthon, D. Doermann, and L. Davis, “Representation and Recognition of Events in Surveillance Video Using Petri Nets,” *Second IEEE Workshop on Event Mining 2004, CVPR2004*, 2004.

[87] M. Albanese, R. Chellappa, V. Moscato, A. Picariello, V. S. Subrahmanian, P. Turaga, and O. Udrea, “A constrained probabilistic petri net framework for human activity detection in video,” *Submitted to IEEE Transactions on Multimedia*.

[88] C. S. Pinhanez and A. F. Bobick, “Human action detection using pnf propagation of temporal constraints,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, p. 898, 1998.

[89] Y. Shi, Y. Huang, D. Minen, A. Bobick, and I. Essa, “Propagation networks for recognizing partially ordered sequential action,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 862–869, 2004.

[90] K. S. Fu, *Syntactic Pattern Recognition and Applications*. Prentice-Hall Inc., 1982.

[91] M. Brand, “Understanding manipulation in video,” *Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition*, p. 94, 1996.

[92] M. S. Ryoo and J. K. Aggarwal, “Recognition of composite human activities through context-free grammar based representation,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1709–1718, 2006.

[93] Y. A. Ivanov and A. F. Bobick, “Recognition of visual activities and interactions by stochastic parsing,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 852–872, 2000.

[94] D. Moore and I. Essa, “Recognizing multitasked activities from video using stochastic context-free grammar,” *Eighteenth national conference on Artificial intelligence*, pp. 770–776, 2002.

[95] G. Medioni, I. Cohen, F. Brémond, S. Hongeng, and R. Nevatia, “Event detection and analysis from video streams,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 8, pp. 873–889, 2001.

[96] S. Hongeng, R. Nevatia, and F. Bremond, “Video-based event recognition: activity representation and probabilistic recognition methods,” *Computer Vision and Image Understanding*, vol. 96, no. 2, pp. 129–162, 2004.

[97] V. D. Shet, D. Harwood, and L. S. Davis, “Vidmap: video monitoring of activity with prolog,” *Proceedings of IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pp. 224–229, 2005.

[98] D. Chen, J. Yang, and H. D. Wactlar, “Towards Automatic Analysis of Social Interaction Patterns in a Nursing Home Environment from Video,” *MIR 04: Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval*, pp. 283–290, 2004.

[99] A. Hakeem and M. Shah, “Ontology and Taxonomy Collaborated Framework for Meeting Classification.” *IEEE International Conference on Pattern Recognition*, pp. 219–222, 2004.

[100] B. Georis, M. Maziere, F. Bremond, and M. Thonnat, “A Video Interpretation Platform Applied to Bank Agency Monitoring,” *2nd Workshop on Intelligent Distributed Surveillance Systems (IDSS)*, 2004.

[101] F. Bremond and M. Thonnat, “Analysis of Human Activities Described by Image Sequences,” *Intl. Florida AI Research Symposium*, 1997.

[102] Event Ontology Workshop. http://www.ai.sri.com/∼burns/EventOntology.

[103] J. Hobbs, R. Nevatia, and B. Bolles, “An Ontology for Video Event Representation,” *IEEE Workshop on Event Detection and Recognition*, 2004.

[104] D. Marr, *Vision*. W. H. Freeman, 1982.

[105] H. B. Barlow, “The coding of sensory messages,” pp. 331–360, 1961.

[106] M. V. Srinivasan, S. B. Laughlin, and A. Dubs, “Predictive coding: A fresh view of inhibition in the retina,” *Proceedings of the Royal Society of London. Series B, Biological Sciences*, vol. 216, no. 1205, pp. 427–459, 1982.

[107] M. R. Lemke and M. Schleidt, “Temporal segmentation of human short-term behavior in everyday activities and interview sessions,” *Naturwissenschaften*, vol. 86, no. 6, 1999.

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

[109] V. Gallese, L. Fadiga, L. Fogassi, , and G. Rizzolatti, “Action recognition in the premotor cortex,” *Brain*, vol. 119, no. 2, pp. 593–609, 1996.

[110] G. Veres, L. Gordon, J. Carter, and M. Nixon, “What image information is important in silhouette based gait recognition?” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 776–782, 2004.

[111] S. Ali and M. Shah, “A lagrangian particle dynamics approach for crowd flow segmentation and stability analysis,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–6, 2007.

[112] J. Wright and R. Pless, “Analysis of persistent motion patterns using the 3d structure tensor,” *IEEE Workshop on Motion and Video Computing*, pp. 14–19, 2005.

[113] E. L. Andrade, S. Blunsden, and R. B. Fisher, “Hidden markov models for optical flow analysis in crowds,” *IEEE International Conference on Pattern Recognition*, pp. 460–463, 2006.

[114] C. Rao, A. Yilmaz, and M. Shah, “View-invariant representation and recognition of actions,” *International Journal of Computer Vision*, vol. 50, no. 2, pp. 203–226, 2002.

[115] X. Feng and P. Perona, “Human action recognition by sequence of movelet codewords,” *3DPVT*, pp. 717–721, 2002.

[116] A. Ali and J. K. Aggarwal, “Segmentation and recognition of continuous human activity,” *IEEE Workshop on Detection and Recognition of Events in Video*, pp. 28–35, 2001.

[117] D. Del Vecchio, R. M. Murray, and P. Perona, “Decomposition of human motion into dynamics based primitives with application to drawing tasks,” *Automatica*, vol. 39, pp. 2085–2098, 2003.

[118] V. Pavlovic and J. M. Rehg, “Impact of dynamic model learning on classification of human motion,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 788–795, 2000.

[119] S. M. Oh, J. M. Rehg, T. R. Balch, and F. Dellaert, “Learning and inferring motion patterns using parametric segmental switching linear dynamic systems,” *International Journal of Computer Vision*, vol. 77, no. 1–3, pp. 103–124, 2008.

[120] D. Minnen, I. Essa, and T. Starner, “Expectation grammars: Leveraging high-level expectations for activity recognition,” *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 626–632, June 2003.

[121] S. Hongeng and R. Nevatia, “Multi-agent event recognition,” *Proceedings of IEEE International Conference on Computer Vision*, vol. 02, 2001.

[122] R. Hamid, A. Johnson, S. Batta, A. Bobick, C. Isbell, and G. Coleman, “Detection and explanation of anomalous activities: Representing activities as bags of event n-grams,” *CVPR*, vol. 1, pp. 1031–1038, 2005.

[123] S. Soatto, G. Doretto, and Y. N. Wu, “Dynamic textures,” *ICCV*, vol. 2, pp. 439–446, 2001.

[124] M. A. Turk and A. P. Pentland, “Face recognition using eigenfaces,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 586–591, 1991.

[125] S. Sarkar, P. J. Phillips, Z. Liu, I. R. Vega, P. Grother, and K. W. Bowyer, “The humanid gait challenge problem: Data sets, performance, and analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 2, pp. 162–177, 2005.

[126] H. Akaike, “A new look at the statistical model identification,” *IEEE Transactions on Automatic Control*, vol. 19, no. 6, pp. 716– 723, 1974.

[127] G. Schwarz, “Estimating the dimension of a model,” *Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[128] B. Rousso, S. Avidan, A. Shashua, and S. Peleg, “Robust recovery of camera rotation from three frames,” *In Proc. APRA Image Understanding Workshop*, 1996.

[129] J. Q. Fang and T. S. Huang, “Solving three-dimensional small-rotation motion equations,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 253–258, 1983.

[130] B. S. Reddy and B. N. Chatterji, “An fft-based technique for translation, rotation, and scale-invariant image registration,” *IEEE Transactions on Image Processing*, vol. 5, no. 8, pp. 1266–1271, 1996.

[131] S. Mann and R. W. Picard, “Video orbits of the projective group: A simple approach to featureless estimation of parameters,” *IEEE Transactions on Image Processing*, vol. 6, no. 9, pp. 1281–1295, 1997.

[132] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, “Convergence properties of the nelder–mead simplex method in low dimensions,” *SIAM Journal on Optimization*, vol. 9, no. 1, pp. 112–147, 1998.

[133] D. Chetverikov and R. Pteri, “A brief survey of dynamic texture description and recognition,” *Proc. of the International Conference on Computer Recognition Systems*, 2005.

[134] Y. Rui, Z. Xiong, R. Radhakrishnan, A. Divakaran, and T. S. Huang, “A unified framework for video summarization, browsing and retrieval,” *MERL Technical Report TR2004-115*, 2004.

[135] A. Divakaran, K. A. Peker, S.-F. Chang, R. Radhakrishnan, and L. Xie, “Video mining: Pattern discovery versus pattern recognition,” *IEEE International Conference on Image Processing*, vol. 4, pp. 2379–2382, 2004.

[136] J. R. Saffran, R. N. Aslin, and E. L. Newport, “Statistical learning by 8-month-old infants,” *Science*, vol. 274, no. 5294, pp. 1926–1928, 1996.

[137] A. J. Yezzi and S. Soatto, “Deformotion: Deforming motion, shape average and the joint registration and approximation of structure in images,” *International Journal of Computer Vision*, vol. 53, no. 2, pp. 153–167, 2003.

[138] J. Shi and J. Malik, “Normalized cuts and image segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, 2000.

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

[140] V. I. Levenshtein, “Binary codes capable of correcting deletions, insertions, and reversals,” *Doklady Akademii Nauk SSSR*, vol. 163, no. 4, pp. 845–848, 1965.

[141] D. Weinland, R. Ronfard, and E. Boyer, “Automatic discovery of action taxonomies from multiple views,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1639–1645, 2006.

[142] C. Vogler and D. Metaxas, “Asl recognition based on a coupling between hmms and 3d motion analysis,” *Proceedings of IEEE International Conference on Computer Vision*, pp. 363–369, 1998.

[143] T. Claasen and W. Mecklenbrauker, “On stationary linear time-varying systems,” *IEEE Trans. on Circuits and Systems*, vol. 29, no. 3, pp. 169–184, 1982.

[144] M. Hall, A. V. Oppenheim, and A. Willsky, “Time-varying parametric modeling of speech,” *Proceedings of IEEE Conference on Decision and Control*, pp. 1085–1091, 1977.

[145] P. Turaga, A. Veeraraghavan, and R. Chellappa, “Statistical Analysis on Stiefel and Grassmann Manifolds with Applications in Computer Vision,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008.

[146] A. Srivasatava and E. Klassen, “Bayesian geometric subspace tracking,” *Advances in Applied Probability*, vol. 36(1), pp. 43–56, March 2004.

[147] C. Sminchisescu, A. Kanaujia, Z. Li, and D. Metaxas, “Conditional Random Fields for Contextual Human Motion Recognition,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 1808–1815, 2005.

[148] L. P. Morency, A. Quattoni, and T. Darrell, “Latent-Dynamic Discriminative Models for Continuous Gesture Recognition,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[149] P. Saisan, G. Doretto, Y. N. Wu, and S. Soatto, “Dynamic texture recognition,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 58–63, 2001.

[150] T. S. Rao, “The fitting of nonstationary time-series models with time-dependent parameters,” *Journal of the Royal Statistical Society B*, vol. 32, no. 2, pp. 312–322, 1970.

[151] M. Tsatsanis and G. Giannakis, “Subspace methods for blind estimation of time-varying fir channels,” *IEEE Transactions on Signal Processing*, vol. 45, no. 12, pp. 3084–3093, 1997.

[152] L. H. Lee, “Identification and Robust Control of Linear Parameter-Varying Systems,” *PhD thesis, University of California at Berkeley, Berkeley, California*, 1997.

[153] V. Verdult and M. Verhaegen, “Subspace identification of multivariable linear parameter-varying systems,” *Automatica*, vol. 38, no. 5, pp. 805–814, 2002.

[154] Y. Chikuse, *Statistics on special manifolds, Lecture Notes in Statistics*. Springer, New York., 2003.

[155] S. M. Oh, J. M. Rehg, and F. Dellaert, “Parameterized duration modeling for switching linear dynamic systems,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1694–1700, 2006.

[156] W. M. Boothby, *An introduction to differentiable manifolds and Riemannian geometry*. Academic Press Inc, 1975.

[157] R. Bhattacharya and V. Patrangenaru, “Large sample theory of intrinsic and extrinsic sample means on manifolds-I,” *Annals of Statistics*, vol. 31, no. 1, pp. 1–29, 2003.

[158] B. Pelletier, “Kernel density estimation on riemannian manifolds,” *Statistics & Probability Letters*, vol. 73, no. 3, pp. 297–304, 2005.

[159] X. Pennec, “Intrinsic statistics on riemannian manifolds: Basic tools for geometric measurements,” *Journal of Mathematical Imaging and Vision*, vol. 25, no. 1, pp. 127–154, 2006.

[160] A. Edelman, T. A. Arias, and S. T. Smith, “The geometry of algorithms with orthogonality constraints,” *SIAM Journal Matrix Analysis and Application*, vol. 20, no. 2, pp. 303–353, 1999.

[161] P.-A. Absil, R. Mahony, and R. Sepulchre, "Riemannian geometry of Grassmann manifolds with a view on algorithmic computation," *Acta Applicandae Mathematicae*, vol. 80, no. 2, pp. 199–220, 2004.

[162] D. Lin, S. Yan, and X. Tang, "Pursuing informative projection on grassmann manifold," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 02, pp. 1727–1734, 2006.

[163] J. Hamm and D. D. Lee, "Grassmann discriminant analysis: a unifying view on subspace-based learning," *International Conference on Machine Learning*, pp. 376–383, 2008.

[164] R. Subbarao and P. Meer, "Nonlinear mean shift for clustering over analytic manifolds," *International Journal of Computer Vision*, vol. 84, no. 1, pp. 1–20, August 2009.

[165] O. Tuzel, F. Porikli, and P. Meer, "Pedestrian detection via classification on Riemannian manifolds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 10, pp. 1713–1727, October 2008.

[166] C. R. Goodall and K. V. Mardia, "Projective shape analysis," *Journal of Computational and Graphical Statistics*, vol. 8, no. 2, 1999.

[167] V. Patrangenaru and K. V. Mardia, "Affine shape analysis and image analysis," *22nd Leeds Annual Statistics Research Workshop*, 2003.

[168] A. Srivastava, S. H. Joshi, W. Mio, and X. Liu, "Statistical shape analysis: Clustering, learning, and testing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 4, 2005.

[169] Karcher, H., "Riemannian center of mass and mollifier smoothing," *Communications on Pure and Applied Mathematics*, vol. 30, pp. 509–541, 1977.

[170] K. Gallivan, A. Srivastava, X. Liu, and P. VanDooren, "Efficient algorithms for inferences on grassmann manifolds," *12th IEEE Workshop Statistical Signal Processing*, 2003.

[171] G. Aggarwal, A. K. Roy-Chowdhury, and R. Chellappa, "A system identification approach for video-based face recognition," *IEEE International Conference on Pattern Recognition*, pp. 175–178, 2004.

[172] K. C. Lee, J. Ho, M. H. Yang, and D. Kriegman, "Video-based face recognition using probabilistic appearance manifolds," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[173] I. L. Dryden and K. V. Mardia, "Statistical shape analysis," 1998.

[174] T. K. Kim, J. Kittler, and R. Cipolla, "Discriminative learning and recognition of image set classes using canonical correlations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1005–1018, 2007.

[175] O. Arandjelovic, G. Shakhnarovich, J. Fisher, R. Cipolla, and T. Darrell, "Face recognition with image sets using manifold density divergence," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 581–588, 2005.

[176] S. K. Zhou and R. Chellappa, "From sample similarity to ensemble similarity: Probabilistic distance measures in reproducing kernel hilbert space," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 6, pp. 917–929, June 2006.

[177] K.-C. Lee, J. Ho, and D. J. Kriegman, "Acquiring linear subspaces for face recognition under variable lighting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 684–698, 2005.

[178] W. Fan and D.-Y. Yeung, "Locally linear models on face appearance manifolds with application to dual-subspace based classification," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1384–1390, 2006.

[179] M. Artac, M. Jogan, and A. Leonardis, "Incremental PCA for on-line visual learning and recognition," *IEEE International Conference on Pattern Recognition*, 2002.

[180] H. V. Neto and N. Ulrich, "Incremental PCA: An alternative approach for novelty detection," *Towards Autonomous Robotic Systems*, 2005.

[181] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521623049, 2000.

[182] J. Hamm, "Subspace-based learning with grassmann kernels," *PhD Thesis, University of Pennsylvania*, 2008.

[183] K. Fukui and O. Yamaguchi, "Face recognition using multi-viewpoint patterns for robot vision," *Int. Symp. of Robotics Res.*, pp. 192–201, 2003.

[184] O. Yamaguchi, K. Fukui, and K. Maeda, "Face recognition using temporal image sequence," *Proceedings of the 3rd. International Conference on Face & Gesture Recognition*, p. 318, 1998.

[185] G. Sparr, "Depth computations from polyhedral images," *Proceedings of IEEE European Conference on Computer Vision*, 1992.

[186] H. Ling and D. W. Jacobs, "Shape classification using the inner-distance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 2, 2007.

[187] S. Biswas, G. Aggarwal, and R. Chellappa, "Efficient indexing for articulation invariant shape matching and retrieval," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[188] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, "Face recognition: A literature survey," *ACM Computing Surveys*, vol. 35, no. 4, pp. 399–458, 2003.

[189] A. J. O'Toole, T. Price, T. Vetter, J. C. Bartlett, and V. Blanz, "Three-dimensional shape and two-dimensional surface textures of human faces: The role of "averages" in attractiveness and age," *Image and Vision Computing Journal*, vol. 18, no. 1, pp. 9–19, 1999.

[190] S. E. Brennan, "The caricature generator," *Leonardo*, vol. 18, no. 3, pp. 170–178, 1985.

[191] P. J. Benson and D. I. Perrett, "Synthesizing continuous-tone caricatures," *Image and Vision Computing*, vol. 9, no. 2, pp. 123–129, 1991.

[192] J. B. Pittenger and R. E. Shaw, "Aging faces as viscal-elastic events : Implications for a theory of nonrigid shape perception," *Journal of Experimental Psychology : Human Perception and Performance*, vol. 1, no. 4, pp. 374–382, 1975.

[193] L. S. Mark, J. T. Todd, and R. E. Shaw, "Perception of growth : A geometric analysis of how different styles of change are distinguised," *Journal of Experimental Psychology : Human Perception and Performance*, vol. 7, no. 4, pp. 855–868, 1981.

[194] J. T. Todd, L. S. Mark, R. E. Shaw, and J. B. Pittenger, "The perception of human growth," *Scientific American*, vol. 242, no. 2, pp. 132–144, 1980.

[195] N. Ramanathan and R. Chellappa, "Modeling age progression in young faces," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 387–394, 2006.

[196] Y. H. Kwon and N. Vitoria Lobo, "Age classification from facial images," *Computer Vision and Image Understanding*, vol. 74, no. 1, pp. 1–21, 1999.

[197] N. Ramanathan and R. Chellappa, "Face verification across age progression," *IEEE Transactions on Image Processing*, vol. 15, no. 11, pp. 3349–3361, 2006.

[198] A. Lanitis, C. Taylor, and T. Cootes, "Toward automatic simulation of aging effects on face images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 442–455, 2002.

[199] A. Lanitis, C. Draganova, and C. Christodoulo, "Comparing different classifiers for automatic age estimation," *IEEE Trans. Systems, Man and Cybernetics*, vol. 34, no. 1, pp. 621–628, 2004.

[200] Y. Fu, Y. Xu, and T. S. Huang, "Estimating human age by manifold analysis of face pictures and regression on aging features," *IEEE International Conference on Multimedia and Expo*, pp. 1383–1386, 2007.

[201] G. D. Guo, Y. Fu, C. R. Dyer, and T. S. Huang, "Image-based human age estimation by manifold learning and locally adjusted robust regression," *IEEE Transactions on Image Processing*, vol. 17, no. 7, pp. 1178–1188, July 2008.

[202] I. L. Dryden and K. V. Mardia, *Statistical Shape Analysis*. John Wiley & Sons, 1998.

[203] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *PAMI*, vol. 24, no. 4, pp. 509–522, 2002.

[204] L. Wiskott, J.-M. Fellous, N. Krüger, and C. von der Malsburg, "Face recognition by elastic bunch graph matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 775–779, 1997.

[205] J. Shi, A. Samal, and D. Marx, "How effective are landmarks and their geometry for face recognition?" *Computer Vision and Image Understanding*, vol. 102, no. 2, pp. 117–133, 2006.

[206] S. V. N. Vishwanathan and A. J. Smola, "Binet-cauchy kernels," *Advances in Neural Information Processing Systems*, 2004.

[207] "The fg-net aging database," *Available: http://www.fgnet.rsunit.com/*.

[208] X. Geng, Z. Zhou, and K. Smith-Miles, "Automatic age estimation based on facial aging patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2234–2240, 2007.

[209] S. Yan, H. Wang, X. Tang, and T. S. Huang, "Learning auto-structured regressor from uncertain nonnegative labels," *Proceedings of IEEE International Conference on Computer Vision*, no. 7, pp. 1–8, 2007.

[210] R. Weber, H. J. Schek, and S. Blott, "A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces," *Proceedings of the International Conference on Very Large Data Bases*, pp. 194–205, 1998.

[211] A. Srivasatava and E. Klassen, "Bayesian geometric subspace tracking," *Advances in Applied Probability*, vol. 36(1), pp. 43–56, March 2004.