2011

Patterns of Motion: Discovery and Generalized Representation

Imran Saleemi
University of Central Florida

PATTERNS OF MOTION: DISCOVERY AND GENERALIZED REPRESENTATION

by

IMRAN SALEEMI
B.S. Ghulam Ishaq Khan Institute
M.S. University of Central Florida

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the Department of Electrical Engineering and Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Spring Term
2011

Major Professor: Mubarak Shah

ABSTRACT

In this dissertation, we address the problem of discovery and representation of motion patterns in a variety of scenarios, commonly encountered in vision applications. The overarching goal is to devise a generic representation, that captures any kind of object motion observable in video sequences. Such motion is a significant source of information typically employed for diverse applications such as tracking, anomaly detection, and action and event recognition. We present statistical frameworks for representation of motion characteristics of objects, learned from tracks or optical flow, for static as well as moving cameras, and propose algorithms for their application to a variety of problems. The proposed motion pattern models and learning methods are general enough to be employed in a variety of problems as we demonstrate experimentally.

We first propose a novel method to model and learn the scene activity, observed by a static camera. The motion patterns of objects in the scene are modeled in the form of a multivariate non-parametric probability density function of spatio-temporal variables (object locations and transition times between them). Kernel Density Estimation (KDE) is used to learn this model in a completely unsupervised fashion. Learning is accomplished by observing the trajectories of objects by a static camera over extended periods of time. The model encodes the probabilistic nature of the behavior of moving objects in the scene and is useful for activity analysis applications, such as persistent tracking and anomalous motion detection. In addition, the model also captures salient scene fea-

tures, such as, the areas of occlusion and most likely paths. Once the model is learned, we use a unified Markov Chain Monte-Carlo (MCMC) based framework for generating the most likely paths in the scene, improving foreground detection, persistent labelling of objects during tracking and deciding whether a given trajectory represents an anomaly to the observed motion patterns. Experiments with real world videos are reported which validate the proposed approach.

The representation and estimation framework proposed above, however, has a few limitations. This algorithm proposes to use a *single* global statistical distribution to represent all kinds of motion observed in a particular scene. It therefore, does not find a separation between multiple semantically distinct motion patterns in the scene. Instead, the learned model is a joint distribution over *all* possible patterns followed by objects. To overcome this limitation, we then propose a superior method for the discovery and statistical representation of motion patterns in a scene. The advantages of this approach over the first one are two-fold: first, this model is applicable to scenes of dense crowded motion where tracking may not be feasible, and second, it distinguishes between motion patterns that are distinct at a semantic level of abstraction. We propose a mixture model representation of salient patterns of optical flow, and present an algorithm for learning these patterns from dense optical flow in a hierarchical, unsupervised fashion. Using low level cues of noisy optical flow, K-means is employed to initialize a Gaussian mixture model for temporally segmented clips of video. The components of this mixture are then filtered and instances of motion patterns are computed using a simple motion model, by linking components across space and time. Motion patterns are then initialized and membership of instances in different motion patterns is established by using KL divergence between mixture distributions of pattern instances. Finally, a pixel level

representation of motion patterns is proposed by deriving conditional expectation of optical flow. Results of extensive experiments are presented for multiple surveillance sequences containing numerous patterns involving both pedestrian and vehicular traffic. The proposed method exploits optical flow as the low level feature and performs a hierarchical clustering to obtain motion patterns; and we observe that the use of optical flow is also an integral part of a variety of other vision applications, for example, as features based representation of human actions. We, therefore, propose a new representation for articulated human actions using the motion patterns. The representation is based on hierarchical clustering of observed optical flow in four dimensional, spatial and motion flow space. The automatically discovered motion patterns, are the primitive actions, representative of flow at salient regions on the human body, much like trajectories of body joints, which are notoriously difficult to obtain automatically. The proposed method works in a completely unsupervised fashion, and in sharp contrast to state of the art representations like bag of video words, provides a truly semantically meaningful representation. Each primitive action depicts the most atomic sub-action, like left arm moving upwards, or right leg moving downward and leftward, and is represented by a mixture of four dimensional Gaussian distributions. A sequence of primitive actions are discovered in the test video, and labled by computing the KL divergence between mixtures. The entire video sequence containing the human action, is thus reduced to a simple string, which is matched against similar strings of training videos to classify the action. The string matching is performed by global alignment, using the well-known Needleman-Wunsch algorithm. Experiments reported on multiple human actions data sets, confirm the validity, sim-

plicity, and semantically meaningful nature of the proposed representation. Results obtained are encouraging and comparable to the state of the art.

To my mother,

Shaista Saleemi,

for her love,

her sacrifices.

$\sim$

To my wife,

Nadia,

for her support,

and her companionship.

ACKNOWLEDGMENTS

I would like to thank my advisor Dr. Mubarak Shah for his guidance and unconditional support. His relentless encouragement, and faith in my ability was the single most important reason for the completion of this work.

I am grateful to Khurram Shafique, and Yaser Sheikh for their advice and inspiration during my first year of graduate studies. I thank my colleagues at the Computer Vision Lab at UCF, for their ideas, advice, and their company: Saad Khan, Saad Ali, Arslan Basharat, Jingen Liu, and Mikel Rodriguez. I would also like to express my gratitude to the current members of our lab, especially Yang, Salman, Dey, Vlad, and Kishore.

My years in Orlando have been made memorable by the friendship of Zain, Sana, Nazim, Azer, Kamran, Asher, and Cheema among others. I am indebted to them for their support through the ups and downs, and their companionship. I am thankful to my brother Umar, and my sister Amna, for their continuous support and encouragement, and for their love.

I thankfully acknowledge Dr. Xin Li for his ideas and advice, during our discussions. I am also grateful to Dr. Marshall Tappen and Dr. Pawel Wocjan for serving on my committee.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

Vision is the primary source of information for awareness, learning, and knowledge in humans. At the lowest level of abstraction, visual input can be categorized into static and motion information. In terms of computer vision, static input, such as, appearance, color, and shape, can be extracted from single images, while motion data require capturing of a sequence of images, or videos. It is therefore, not surprising that a vast majority of literature in computer vision deals with problems arising from, or related to various kinds of motion observed in digital visual data. Fields related to such problems include structure from motion, object tracking, action and event recognition, and 3d reconstruction, among others. Extraction, analysis and representation of motion in videos is therefore, a key building block for a variety of applications in visual surveillance.

This thesis develops algorithms and representations that provide a unifying, generic framework to model and learn key patterns within motion data captured from videos, and proposes methods to exploit these motion patterns to solve a wide range of common problems in the field of computer vision, particularly, in visual surveillance. One of the key aspects of the proposed representations is their statistical, generative nature, which if needed, allows sampling and reconstruction of the observed motion data. Moreover, the learning and representation frameworks are largely independent of the methods employed to infer motion, and may include sparse object tracking data, or dense pixel correspondences from optical flow.

1.1 Overview and Motivation

Traditionally, a significant amount of effort in the vision community has been focused on developing fully automated surveillance and monitoring systems [1, 9, 85]. Such systems have the advantage of providing continuous 24 hour active warning capabilities and are especially useful in the areas of law enforcement, national defence, border control and airport security. The current systems are efficient and robust in their handling of common issues, such as illumination changes, shadows, short-term occlusions, weather conditions, and noise in the imaging process [36]. However, most of the current systems have short or no memory in terms of the observables in the scene. Due to this memory-less behavior, these systems lack the capability of learning the environment parameters and intelligent reasoning based on these parameters. Such learning and reasoning is an important characteristic of all cognitive systems that increases the adaptability and thus the practicality of such systems. A number of studies have provided strong psychophysical evidence of the importance of context for scene understanding in humans, such as, handling long term occlusions, detection of anomalous behavior, and even improving the existing low-level vision tasks of object detection and tracking [4, 75].

Recent works in the area of scene modeling are limited to the detection of entry and exit points and finding the likely paths in the scene [74, 18, 71, 35] . We argue that over the period of its operation, an intelligent tracking system should be able to model the scene from its observables and be able to improve its performance based on this model. The high-level knowledge necessary to make such inferences derives from domain knowledge, past experiences, as well as scene geometry,

learned traffic and target behavior patterns in the area, etc. For example, consider a scene that contains bushes that only allow partial or no observation while the targets pass behind them. Most existing systems only detect the target when it comes out of the bushes and are unable to link the observations of the target before and after the long term occlusion. Given that this behavior of targets disappearing and appearing after a certain interval at a certain place is consistently observed, an intelligent system should be able to infer the correlation between these observations and to use it to correctly identify the targets at reappearance. We believe that the identification, modeling and analysis of target behavior in the scene is the key to achieving autonomous intelligent decision making capabilities. The work presented in this thesis is a step forward in this direction.

In addition to observations made from tracking of interesting objects in a scene, optical flow as a low level feature has remained very popular over the years since the original paper by Horn and Shunck [30]. Despite the numerous methods proposed to improve the quality of optical flow [52], Lucas-Kanade flow [41] with some improvement remains one of the most widely employed methods. However, despite its wide range of applications including structure from motion, reconstruction, and video compression, raw optical flow especially in the regions of relatively small or no motion, is highly noisy and sensitive to clutter, and pixel wise local motion does not provide significant information regarding the general properties of the scene. A few methods have been proposed in the community that exploit optical flow or simpler, spatiotemporal gradients to analyze patterns of motion in a scene. These include topic models [78], bag of words using diffusion maps embedding [84], and quantized spatiotemporal gradients and HMM's [46]. We however, argue that a sufficient condition to have a generic, general purpose, yet powerful representation, is for

it to be generative. It is therefore important to consider statistical properties of optical flow and employ meaningful inference techniques to estimate models of motion patterns since they have proved extremely useful in various surveillance tasks [32, 63].

Moreover, since optical flow is frequently employed as the low level feature in action representation, recognition and classification literature, a meaningful, consolidated representation of motion should be directly applicable to the problem of action recognition as well. This should alleviate the need to have complicated, sophisticated, and computationally expensive processing pipelines for such applications. An example of such techniques include the Bag of Video Words approach, which involves multiple processing steps, including linearization of spatiotemporal gradient features or optical flow, quantization, dimensionality reduction, clustering, and state of the art classifiers like State Vector Machines (SVM) to obtain reasonable action recognition performances. What is worst is that such techniques provide no semantic interpretation, or insight into the inner workings or atomic representations of the actions being modeled. The unifying idea behind this thesis is that a reasonable representation of motion should be such that it simplifies the exploitation of motion, while being applicable to many of the diverse vision problems.

1.2 Contributions

This thesis is an attempt to devise a single, general purpose, generative, statistical framework to represent motion of objects observed in visual data, to solve a variety of diverse applications commonly encountered in the field of computer vision. As mentioned earlier, sensing of motion is an integral part of the human experience and is paramount to carrying out our daily lives. The

same is true for machine vision, where motion of objects is estimated and used to solve problems from object detection and tracking, to learning normalcy models, to recognizing human actions and activities.

In contrast to the many existing approaches to solving the above mentioned problems, which are reviewed in detail in chapter 2, the representations and learning algorithms proposed in this thesis are a step towards a consolidated framework that can be employed towards the solution of common computer vision problems. Moreover, the development of representations and algorithms in this thesis, is heavily inspired by the need to move towards more semantically meaningful solutions. The statistical nature of the proposed representations, coupled with their low dimensionality, allows straightforward visualizations, for example, by marginalization and computation of conditional expectations. This ease of visualization is translated to meaningful representations for common applications like action recognition. This is in sharp contrast to most of the work in the literature, where better performance is preferred over understanding. An example of this prevalent trend is the bag of video words approach, where neither visual nor semantic interpretation of the actions representation (codebook) is available. This thesis proposes frameworks, which bring us much closer to understanding of the underlying structures within motion, even to the extent of potential transformation to natural languages.

1.2.1 Motion Patterns as a Single Non-parametric Distribution

We first propose a novel framework to learn traffic patterns in the form of a multivariate non-parametric probability density function of spatio-temporal variables that include the locations of

objects detected in the scene and their transition times from one location to another. The model is learned in a fully unsupervised fashion by observing the trajectories of objects by a static camera over extended periods of time. The model also captures salient scene features, such as, the usually adapted paths, frequently visited areas, occlusion areas, entry/exit points, etc.

We learn the scene model by using the observations of tracks over a long period of time. These tracks may have errors due to clutter and may also be broken due to short term and long term occlusions, however by observing enough tracks, one can get a fairly good understanding of the scene and infer above described scene properties and salient features. We assume that the tracks of the moving objects are available for training. We use these tracks in a training phase to discover the correlation in the observations by learning the motion pattern model in the form of a multivariate probability density function (pdf) of spatio-temporal parameters (i.e., the joint probability density of pairs of observations of an object occurring within certain time intervals, $(x, y, x', y', \Delta t)$). Instead of imposing assumptions about the form of this pdf, we estimate the pdf using kernel density estimators. Each track on the image lattice can then be seen as a random walk where the probabilities of transition at each state of the walk is given by the learned spatio-temporal kernel density estimate. Once the model is learned, we use a unified Markov Chain Monte-Carlo (MCMC) sampling based scheme to generate the most likely paths in the scene, to decide whether a given path is an anomaly to the learned model, and to estimate the probability density of the next state of the random walk based on its previous states. The predictions based on the model are then used to improve the detection of foreground objects as well as to persistently track targets through short-term and long-term occlusions. We show quantitatively that the proposed system improves

both the precision and recall of the foreground detection and can handle long term occlusions that cannot be handled using constant dynamics models commonly used in the literature. We also show that the proposed algorithm can handle occlusions that were not observed during training. The examples of these type of occlusions include vehicles parked in the scene after training, or a bug sitting on the camera lens.

As opposed to explicitly modeling trajectories or smaller sections of trajectories of objects in the scene, we propose modeling of joint distribution of initial and final locations of every possible transition and the transition times. Our approach is original in the following ways:

- *We propose a novel motion model that not only learns the scene semantics but also the behavior of traffic through arbitrary paths. This learning is not limited like other approaches that work best with well defined paths like roads and walkways.*

- *The learning is accomplished using a joint five dimensional model unlike pixel-wise models and mixture or chain models. The proposed model represents the joint probability of a transition from any point in the image to any other point, and the time taken to complete that transition.*

- *The temporal dimension of traffic patterns is explicitly modeled, and is included in the feature vector, thus enabling us to distinguish patterns of activity that correspond to the same trajectory cluster but have high deviation in the temporal dimension. This is a more generalized method as compared to modeling pixel-wise velocities.*

- *Instead of fitting parametric models to the data, we propose the idea of learning tracks information using Kernel Density Estimation. It allows for a richer model and the density retained at each point in the feature space accurately reflects the training data.*

- *Rather than exhaustively searching for predictions in the feature space based on their probabilities, we propose to use stochastic methods to* sample *from the learned distribution and use it as prediction with a computed probability. Sampling is thus used as the process propagation function in our state estimation framework.*

- *Unlike most of the previous work reported in this section, which is targeted towards one or two similar applications, we apply the proposed probabilistic framework to solve a variety of problems that are commonly encountered in surveillance and scene analysis.*

1.2.2 Motion Patterns as Distinct Mixture Distributions

We, however, observe that the proposed global statistical model of motion of the entire scene, discussed above, does not provide explicit separation between motion patterns that are semantically distinct. For example, the two directions of traffic on a two-way street are not represented by multiple distributions or clusters. Therefore, in the second part of this thesis, we propose an improved representation of motion, where distinct patterns of motion are automatically discovered at multiple semantic abstractions, and represented by multiple mixture distributions.

The goal of this method is scene modeling and understanding by unsupervised inference of motion patterns in static camera scenes, which is a key task in visual surveillance. Scene under-

standing, in general, may refer to *scene layout* in structured environments (e.g., sidewalks, roads, intersections), *motion patterns* (e.g., vehicles turning, pedestrian crossings), and *scene status* (e.g., traffic light status, congestion). The proposed work is an attempt to model and learn motion patterns from static camera videos without any user intervention. Given that the related literature uses various terms to describe their representations (e.g., behavior, activity, event and variants), it should be mentioned here that the term 'motion pattern' in our work refers to a spatial segment of the image, that has a high degree of local similarity of speed as well as flow direction within the segment, and otherwise outside. The patterns may not be disjoint in the image space, and at a semantic level describe the flow of object motion and ideally contain the source as well as the sink of the path described by the pattern. Ideally, the representation of a pattern should not only list the pixels in the pattern, but it is very desirable to also have a statistical model of flow magnitude and direction at each pixel. Models representing scenes and the motion therein are useful in a variety of surveillance related tasks like object tracking and classification, abnormal behavior detection, etc.

We propose a mixture model representation of salient patterns within optical flow observed over multiple frames, and present an algorithm for learning these patterns from dense optical flow in a hierarchical, unsupervised fashion. Using low level cues of noisy optical flow, K-means is employed to initialize a Gaussian mixture model for temporally segmented clips of video. The components of this mixture are then filtered and instances of motion patterns are computed using a simple motion model, by linking components across space and time. Motion patterns are then initialized and membership of instances in different motion patterns is established by using KL

divergence between mixture distributions of pattern instances. Finally, a pixel level representation of motion patterns is proposed by deriving conditional expectation of optical flow.

The proposed method for inference of motion patterns employs simple yet powerful statistical modeling and learning methodologies. The novel contributions of this approach include, (1) introduction of a statistical model of raw unquantized optical flow for motion patterns representation, (2) a hierarchical problem-specific learning method, (3) use of a co-occurrence free measure of spatiotemporal proximity and flow similarity between features, and (4) statistical inference of motion patterns by computation of conditional optical flow expectation.

1.2.3 Discovery of Patterns in Human Actions

Finally, we present a framework that exploits the motion patterns representation proposed above, in order to discover and represent primitive or atomic actions within articulated human actions. When it comes to human action recognition, motion pertaining to articulation of human limbs is a significant source of information. A variety of approaches have attempted to capture this motion by representing it using spatially distributed histograms of orientation, holistic representations, and trajectories of human body parts or joints. In general, typical human actions are adequately recognizable if some description of limb movement in a temporal sequence is available. Of course, detection of limbs in generic videos is a hard problem and far from solved.

It has been observed however, that recognition of human actions in videos, does not require explicit detection of presence, location, or motion of all individual body parts, throughout the video. Indeed, approaches like bag of words have demonstrated, that there often is a clear sepa-

ration between low level features emanating from videos of actions belonging to different classes, regardless of viewpoint. Since recognizing an action is possible, even when all limbs or body joints are not explicitly detected, we propose to capture articulated motion of humans performing actions, by computing flow with respect to the human body, and learning patterns of motion within the computed flow. Ideally, we expect these motion patterns to represent motion of the limbs, but since the patterns are discovered automatically, there is no constraint on the body parts' presence in the FOV.

The proposed approach for action recognition begins with a simple 'centralization' process, whereby the bounding boxes (region of interest) containing the human performing an action are superimposed on top of each other, so as to compensate for rigid (or translational) motion of the entire body. We are therefore left with the motion that is with respect to the body center. An example of this is a video of a human walking, recorded from a profile view. One will observe that although the legs of the performer do not move backward with respect to the ground, they do move back with respect to the performer's body. We then simply compute the optical flow on this centralized region of interest, and estimate distinct patterns of motion. As expected, we successfully obtain representations of atomic actions or primitives, that essentially correspond to the motion of the performer's limbs. For example, consider the action 'waving both arms'. One can break the action down into multiple sub-actions, for example, right arm moving from the side to over the head, left arm moving up, right arm moving down, left arm moving down, and so. These are exactly the kind of primitive actions that we expect to discover automatically given a training video, and represent them as motion patterns.

Given this representation learned from training videos, a test video is similarly processed to detect instances of these primitives, and a temporal sequence is thus obtained which can be written as a simple string. Strings obtained during training are then compared to the test string by global alignment of letters (primitives) in the string using the Needleman-Wunsch algorithm, and the action is classified.

The main contributions of this approach are evident in the simplicity and efficiency of the recognition process. The completely unsupervised method provides a semantically meaningful representation, where the detected motion patterns can actually be given natural language labels. Despite its simplicity, the eventual representation is so powerful, that the actual recognition task is reduced to a simple string matching.

1.3 Organization of the Thesis

The thesis is structured as follows: **Chapter 2** reviews existing literature, and focuses on the many representations of motion in different problem domains, as well as the applications and fields of computer vision that employ such representations for the solution of different problems. **Chapter 3** presents a framework to represent and learn object motion derived from tracking data as a single non-parametric distribution, and describes algorithms to apply this model to solve different common computer vision problems. **Chapter 4** proposes an improved model to represent patterns of motion as Gaussian mixture distributions of optical flow, and reports results of learning this model on static camera surveillance (CCTV) as well as wide area aerial (WAAS) data. **Chapter 5** then presents an unlikely application of learning motion patterns, by employing them to estimate

and recognize atomic actions, for human action recognition and classification. Finally, the thesis is concluded in **Chapter 6** with a summary of contributions and description of future work.

CHAPTER 2: LITERATURE REVIEW

In this chapter we discuss some of the more prominent methods for representing motion by analysis of trajectories or dense flow. Moreover, some of the common computer vision problems and applications that depend on modeling and representation of motion are discussed. An important point to notice here is the vast variety of these representations and exploitation methodologies, which can be contrasted later with the proposed general framework that solve many of the same problems, without the need for application-specific abstraction techniques.

2.1 Trajectory Analysis

One of the important and prevalent types of such abstraction techniques, are the ones derived from object trajectories. Assuming that an object is detected in each frame of a video sequence, and associated over multiple frames, the trajectory of the object can be represented as a sequence of points, $T = \{(x_i, y_i, t_i)\}$, where (x, y) indicates the center of the detected object, and t indicates the time at which the object was observed. Since numerous object tracking methods have been proposed for a variety of scenarios in the computer vision and surveillance literature [85], trajectory and path modeling has become an important step in various applications, many of which are crucial to surveillance and monitoring systems. Such models can be used to filter object tracks, act as prior motion model in tracking, generate likely paths, find locations of sources and sinks in the scene,

and detect anomalous tracks, etc. This kind of modeling can be directly used as a feedback to the initial stages of the tracking algorithm, and applied to solve short and long term occlusions. In recent years, a number of different methods and features for trajectory and path modeling of traffic in the scene have been proposed. These methods differ by their choice of features, models, learning algorithms, applications, and training data. A review of these models is presented in [7]. Some of these methods are briefly discussed below.

Modeling of typical paths and trajectories has been performed by employing Neural Networks based approaches, as proposed in [34, 56, 38]. For example, [34] proposes a modeling of object trajectories at two distinct levels of abstraction, which include, (i) instantaneous (position) or first-order (velocity) information, and (ii) long-term, large scale, trajectory information. The proposed learning algorithm employs a topological map (self organizing feature map), which takes as input an augmented feature vector of the current position, and instantaneous velocity. The network thus essentially acts as a classifier, which makes a binary decision about the normalcy (or otherwise) of an input test trajectory. In addition to the computational complexity and lack of adaptability of Neural Networks, a major disadvantage of these methods is their inability to handle incomplete and partial trajectories. Fernyhough et al. [20] use the spatial model presented in [31] as a basis for a learning algorithm that can automatically learn object paths by accumulating the traces of targets. Koller-Meier et al. [45] use a node-based model to represent the average of trajectory clusters. A similar technique is suggested by Lou et al. [51]. Although both methods successfully identify the mean of common trajectory patterns, no explicit information is derived regarding the distribution of trajectories around the mean. A hierarchical clustering of trajectories is proposed in [26, 70],

where trajectories are represented as a sequence of states in a quantized 6D space for trajectory classification and the method is based on a co-occurrence matrix that assumes that all trajectory sequences are equally long. However, this assumption is usually not true in real sequences.

Detection of sources and sinks in a scene as a pre-step allows robust tracking. In [69], a Hidden Markov Model based scheme of learning sources and sinks is presented, where all sequences are two-state long. The knowledge of sources and sinks is used to correct and stitch tracks in a closed-loop manner. Another HMM-based approach is to model trajectories as transitions between states representing Gaussian distributions on the 2D image plane [60, 77]. Galata et al. [23] propose a mechanism that automatically acquires stochastic models of any behavior. Unlike HMMs, the proposed variable length markov model can capture dependencies that may have a variable time scale.

Another body of work for modeling motion patterns in a scene lies in the literature of multi-camera tracking [35, 33, 44], where the objective is to use these models to solve for hand-off between cameras. The motion patterns are modeled only between the field of views of cameras. These models can be seen as special cases of the proposed model, which is richer in the sense that it captures the motion pattern both in visible and occluded regions. This richness of the proposed model allows it to handle dynamic occlusions, such as person to person occlusions as well as occlusions that occur after the model is learned. Tekalp [14] uses Bayesian networks for tracking and occlusion reasoning across calibrated cameras with overlapping views, where sparse motion estimation and appearance are used as features. Tieu et al. [74] infer the topology of a network of

non-overlapping cameras by computing the statistical dependence between observations in different cameras.

Another interesting piece of work in this area is by Ellis et al. [18] who determined the topology of a camera network by using a two stage algorithm. First the entry and exit zones of each camera are determined using an Expectation-Minimization technique to cluster the start and end points of object tracks. The links between these zones across cameras are then found using the co-occurrence of entry and exit events. The proposed method assumes that correct correspondences will cluster in the feature space (location and time) while the wrong correspondences will generally be scattered across the feature space. The basic assumption is that if an entry and exit at a certain time interval are more likely than a random chance then they should have a higher likelihood of being linked.

Stauffer [71] proposes an improved linking method that tests the hypothesis that the correlation between exit and entry events is similar to the expected correlation when there are no valid transitions. This allows the algorithm to handle the cases where exit-entrance events may be correlated, but the correlation is not due to valid object transitions. Both of these methods assume a fixed set of entry and exit locations after initial training and hence cannot deal with newly formed occlusions without retraining of the model.

Huang and Russell [33] presented a probabilistic approach for tracking vehicles across two cameras on a highway. Transition times were modeled as Gaussian distributions and the initial transition probabilities were assumed to be known. Kettnaker and Zabih [44] used manual input of the topology of allowable paths of movement and the transition probabilities and did not jointly model the highly correlated features of positions, velocities and transition times of objects across

cameras. Javed et al. [35] presented a non-parametric motion model to describe the relationship between spatio-temporal features of observations between two cameras. The model was learned based on the sparse observation during training. The observations for training were also required to be manually labelled.

Hoiem et. al. [29] take a step forward in image and scene understanding and proposed improvement in object recognition by modeling the correlation between objects, surface geometry and camera viewpoints. Rosales et al. [62] estimate motion trajectories using Extended Kalman Filter to enable improved tracking before, during and after occlusions. Kaucic et al. [42] present a modular framework that handles tracking through occlusions and the blind regions between cameras by the initialization, tracking and linking of high-confidence smaller track sections. Wang et al. [79] propose measures for similarity between tracks that take into account spatial distribution, velocity, and object size. The trajectories are then clustered based on object type and its spatial and velocity distributions. Perera et. al. [59] present a method to reliably link track segments during the linking stage. Splitting and merging of tracks is handled to achieve multi-object tracking through occlusion. Recently, Hu et al. [32] have presented an algorithm for learning motion patterns where foreground pixels are first clustered using fuzzy K-means algorithm and trajectories are then hierarchically clustered based on the results of previous step. Trajectory clustering is performed in two layers for spatial and temporal based clustering. Each pattern of clustered trajectories is then assumed to be a link in a chain of Gaussian distributions, the parameters for which are estimated using features of each clustered trajectory. Results of experiments on anomaly detection and behavior prediction are reported.

It is easy to see that, as an underlying feature for motion analysis and representation, exploitation of the trajectory information alone, has generated a vast diversity of proposed methods, even when the application domains are extremely similar or overlapping. In this thesis, we argue against the prevalence of such specialized, problem-specific approaches, by proposing a general representation of observable object motion, and experimentally demonstrate the feasibility, effectiveness and generality of the representation by exploiting it for some of the common vision problems.

2.2 Dense Flow Analysis

A large amount of literature in the field of computer vision that makes use of motion information is categorized as scene modeling and understanding framework. Scene understanding may refer to various characteristics of a visually observable region. These may including static properties, such as, roads, sidewalks, walls, intersections, occlusions, and entry or exit locations; dynamic properties, e.g., vehicles moving straight, making turns, pedestrians crossing the road, etc.; and the scene status, for example, traffic density and congestion, traffic light status, etc.

Much of the above mentioned understanding about a scene can be extracted from the motion information, including even the static characteristics such as presence of intersections or occlusions. However, due to limitations of the current state of the art in tracking, object detection and tracking may be infeasible for a variety of scenarios for which such visual data may be available. From a scene understanding point of view, the various approaches to scene modeling, diverse as they are in methods and applications, can be broadly categorized based on the observations made from the scene. The most commonly used features for scene modeling are low level motion and ap-

pearance features [89, 82, 78, 84]. Examples of such features include sparse or dense optical flows [41], spatiotemporal gradients [46], and object trajectories obtained after detection and tracking [72, 6, 79, 32, 63].

Scene modeling and activity perception approaches based on the traditional pipeline of object detection (based on appearance [11] or motion [72]), and subsequent tracking of those detections [6, 39], are well suited to surveillance scenarios involving far-field settings, where computation of complicated features, such as gestures, appearance or local motion may be infeasible, but sufficient number of pixels on object ensure motion based detection. Such methods are applicable to structured scenes containing clearly discernable distinct paths, like roads and sidewalks, and well defined entry and exit points. Proposed approaches to scene understanding employing trajectory features involve clustering or classification of whole trajectories using proximity in the space of different distance metrics [72, 79, 32, 43, 22, 40]. Recently proposed methods learn dense pixel to pixel transition distributions using trajectories obtained after tracking [63] and attempt abnormality detection, improved tracking, and even motion based foreground background segmentation. The reliance of all the above mentioned approaches on good quality object trajectories is however, a significant drawback.

The second kind of approaches for scene understanding and activity analysis use dense, low level motion or appearance feature vectors directly, instead of trajectories. Such features include multi-resolution histograms [89], spatiotemporal gradients [46], and appearance descriptors and spatiotemporal volumes [48], etc. These approaches can perform action recognition and are use-

ful for detection and separation of co-occurring activities, but are usually supervised, where the training stage involves manual isolation of activities or video clips into distinct activities.

A few methods have recently been proposed that employ similar low level features for motion pattern inference [78, 46, 84]. Wang et al [78] obtain sparse optical flow [41] for pixels with frame difference above a threshold which is then used as the low level observations. Yang et al [84] also employ dense optical flow, while Kratz and Nishino [46] use spatiotemporal gradients as the most basic features. All these methods however share the severe drawbacks of spatiotemporal quantization of videos, and co-occurrence statistics based inference. While temporal division of long video sequences into tractable smaller length video clips is useful without loss of information, spatial division of video frames into cuboids (30×30 in [46], and 10×10 in [78, 84]) results in loss of resolution and therefore coarse estimates of motion at the super-pixel level, as opposed to pixel or sub-pixel levels. Since high density crowd scenes may not have regions of uniform motion, especially when observed over a significant length of time, and owing to the small number of pixels per object, this spatial resolution, which is usually already limited in surveillance scenarios, becomes extremely important. The quantization of optical flow direction into large intervals [78, 84] also poses a significant limitation on the potential applications by introducing abrupt boundary jumps and coarse pattern direction estimation. Furthermore, employment of co-occurrence statistics [84], while useful at higher levels of model abstraction, results in the fact that co-occurring patterns of motion, even when distant in image and motion feature space, cannot be discerned from one another. On the other hand, explicit high level temporal models of atomic patterns with non-quantized representations can easily be employed to deduce co-occurrence.

2.3 Motion Representation for Action Recognition

Another important application in the field of computer vision, that essentially relies on motion information, is the representation, recognition, and classification of human actions. This is a broad and active area of research, and comprehensive reviews of the proposed methods can be found in [8, 81]. Our discussion in this regard is restricted to a few influential and relevant parts of literature. These methods can be categorized based on how they represent actions (and motion), and are summarized below:

2.3.1 Holistic Representations

These methods incorporate global, image-based representation of actions, and do not require detection, labeling, or tracking of individual body parts. The only requirement for these methods is to detect a bounding box enclosing the person performing an action. Dense features computed within the bounding box are then used to represent actions. Examples of these features include intensity images [12], silhouettes composing motion history (MHI) and motion energy (MEI) images [5], spatiotemporal shapes using tracks of body contours [87], and spatiotemporal volumes spanned by silhouette images [67]. Other holistic representations of actions involve optical flow [61], spatiotemporal gradients [88], and HoG [73]. As mentioned earlier, such representations ignore the useful information related to primitive, sub-actions, which can compose multiple actions by spatiotemporal ordering, and are much more discriminative than holistic representations.

2.3.2 *Part based Representations*

Methods based on information derived from knowledge of location or appearance of body parts or joints fall in this category. This is the most intuitive representation, but the most difficult to estimate reliably in practice. Examples include body parts detection, or features of trajectories of landmark human body joints [86], and motion features of detected body parts [57]. Detection of body parts or joints in videos is an extremely challenging problem and even the constrained settings of discriminative background and use of markers does not ensure a completely unsupervised process. The proposed work can be considered to be part of this category, since the proposed primitive actions representation generally corresponds to discriminative motion induced by independently moving body parts. In contrast to traditional methods however, there is no need to explicitly detect any body parts, or even assume presence of such parts or joints. For example, actions of an alien actor with six arms and four legs can be represented just as well as a human's.

2.3.3 *Interest point based Representations*

Another important direction of research which has gained a lot of interest recently is the use of spatiotemporal interest points, and their trajectories or feature descriptors computed around interest points. Works by Dollar et al. [15], Laptev et al. [47], Gilbert et al. [24], and Filipovych and Ribeiro [21] are representative of this large category of methods, which is also loosely termed as 'bag of visual words'. The main strength of this representation is the robustness to occlusion, since there is no need to track or detect the entire human body or its parts, and therefore, impressive results have been obtained using such methods on standard data sets. We argue however, that in

the process of computing such representations, the semantic interpretation of actions is lost, and it is not easy to visualize the underlying spatiotemporal structure of actions. Furthermore, these methods have increased complexity and are sensitive to a number of intermediate processes including interest point detection, choice of descriptors, number of codebook words, and classifiers. We contend that action representation and recognition need not be this complex and should have a semantic interpretation.

2.4 Summary

The review of existing literature related to analysis, exploitation, and representation of object motion, is expected to serve two main purposes. First, it highlights the importance of using motion in addressing a variety of computer vision problems. Foreground (or object) motion is one of the most widely employed input features for diverse applications. Secondly, the variety of applications dictate the proposing and use of many different kinds of representation, from holistic to local levels of abstraction for both dense features like optical flow, and sparse ones like object trajectories. This observation about the multitude of representations, in turn underscores the contributions of this thesis, in proposition of generalized representations.

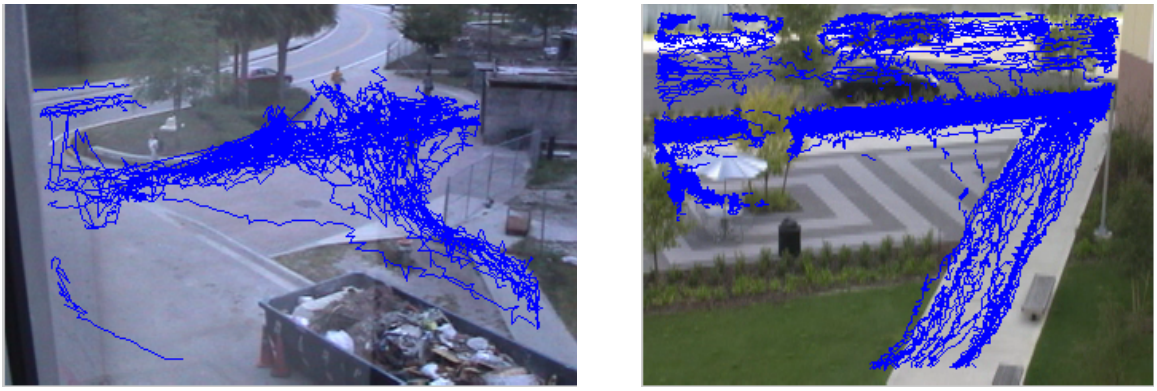
In the following chapters, we formally describe and elaborate on the proposed, novel representations. Specifically, the next chapter introduces a single, generalized model for representation of all trajectories of objects tracked in a scene.

CHAPTER 3: MOTION PATTERNS AS A NON-PARAMETRIC PROBABILISTIC MODEL

In this chapter, we present a framework to automatically learn a probabilistic model of the traffic patterns in a scene, which is represented as a distribution of spatio-temporal transitions across the scene. We show that the proposed model can be applied towards various visual surveillance applications that include, behavior prediction, detection of anomalous patterns in the scene, improvement of foreground detection, and persistent tracking. The chapter is organized as follows: Learning of model, sampling, and state estimation is described in Section 3.1. In Section 3.2, we present our methods to use the learned distribution for different applications. These include behavior prediction using likely path generation(3.2.1), improving foreground detection using motion information (3.2.2), anomalous track detection (3.2.3), and persistent tracking of objects over time (3.2.4). Results are presented in Section 3.3 and some issues raised by the reviewers of the associated publication ([63]) are discussed in Section 3.4.

3.1 Probabilistic Traffic Model

We propose a model that learns traffic patterns as joint probability of the initial and final locations, and duration of object transitions and describe our method of learning and sampling from the distribution. In this section, we discuss the feature to be learned and explain how it is computed. The sampling process and its use in the state estimation framework is also explained.



(a) (b)

Figure 3.1: Two example scenes used for the testing of proposed framework, with some of the tracks generated by the tracker during training.

3.1.1 Learning the Transition Distribution using KDE

We use the surveillance and monitoring system, KNIGHT[36] to collect tracking data. This data consists of time stamps and object labels with locations of their centroid for each frame of video. KNIGHT assigns a unique label to each object present in the scene and attempts to persistently track the object using a joint motion and appearance model. A frame of video may contain multiple objects. Given this data, we associate an observation vector $O(x, y, t, l)$ with each detected object. In this vector, x and y are the spatial coordinates of the object centroid on image lattice, t is the time instant at which O is observed (accurate to a millisecond) and l is the label for object in O as per tracking. We seek to build a five dimensional estimate of the transition probability density function $p\,(X, Y, \tau)$, where X and Y are the initial and final states representing the object centroid locations in $2d$ image coordinates and τ is the time taken to make the transition from X to Y in milliseconds.

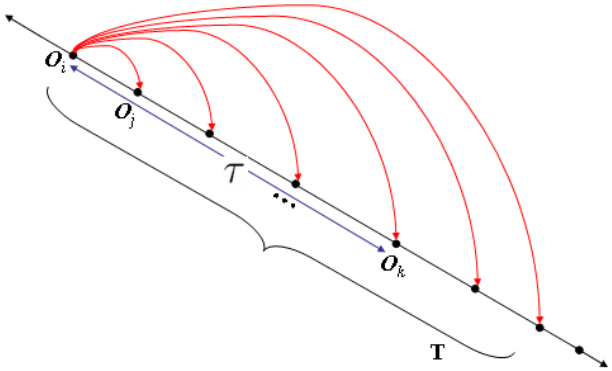


Figure 3.2: A set of observations is shown on a track. τ in this example is the time taken for the transition between observations O_i and O_k. The maximum allowed duration between any two observations is T.

In the proposed work, the analysis is performed on a feature space where the n transitions are represented by $\mathbf{z}_i \in \mathbb{R}^5$, $i = \{1, 2, \ldots, n\}$. The vector $\mathbf{z}$ consists of a pair of $2d$ locations of the centroid of object before and after transition, and time taken to execute the transition. Any two observations O_i and O_j in frames i and j respectively, comprise a feature point $Z(x_i, y_i, x_j, y_j, t_j - t_i)$ for the probability estimate if they satisfy the following:

- $0 < t_j - t_i = \tau < T$, thus the implicit assumption is that O_i and O_j are uncorrelated if occurring simultaneously or beyond a time interval of T. For our experiments, we chose $T = 5000$ms.
- $l_i = l_j$, both O_i and O_j are the observations of the same object as per the tracker's labelling.
- If $l_j \notin L_k$, where L_k is the set of all objects (labels) in frame k, then *all* labels in frame k make valid data points with l_j provided $0 < \tau < T$.

Given the feature (X, Y, τ) as described above, a parametric model like Gaussian can be fitted to the training data. However, it can be noticed that the transition probabilities of moving to final states that are equidistant from initial state are not equal. In other words, some paths are much more likely to be adapted as compared to others. Therefore, very rarely is the probability of transition equally likely in all directions. Although such multimodal distribution can be captured using a mixture of Gaussian distributions, it tends to over-smooth the probability, thus losing the valuable structure therein. Such arbitrarily structured data can be best analyzed using nonparametric methods since they make no underlying assumptions about the shape of the density. Nonparametric estimation methods operate on the principle that dense regions in a given feature space, populated by feature points from some class, correspond to the modes of the actual underlying pdf.

Therefore, in the proposed approach, Kernel Density Estimation ([58], [16]) is used to estimate the pdf $p\,(X, Y, \tau)$ non-parametrically. Suppose we have a sample consisting of n, d dimensional, data points $\mathbf{z_1}, \mathbf{z_2}, \ldots, \mathbf{z_n}$ from a multi-variate distribution $p(\mathbf{z})$, then an estimate $\hat{p}(\mathbf{z})$ of the density at **z** can be computed using

$$\hat{p}(\mathbf{z}) = \frac{1}{n}|\mathbf{H}|^{-\frac{1}{2}} \sum_{i=1}^{n} K(\mathbf{H}^{-\frac{1}{2}}(\mathbf{z} - \mathbf{z_i})), \tag{3.1}$$

where the d variate kernel $K(\mathbf{z})$ is a bounded function satisfying $\int K(\mathbf{z})dz = 1$, $K(\mathbf{z}) = K(\mathbf{-z})$, $\int \mathbf{z}K(\mathbf{z})dz = 0$, $\int \mathbf{z}\mathbf{z}^{\mathbf{T}}K(\mathbf{z})dz = \mathbf{I_d}$ and $\mathbf{H}$ is the symmetric positive definite $d \times d$ bandwidth matrix. The multivariate kernel $K(\mathbf{z})$ can be generated from a product of symmetric univariate kernel K_u, i.e.,

$$K(\mathbf{z}) = \prod_{j=1}^{d} K_u(\mathbf{z}_{\{j\}}). \tag{3.2}$$

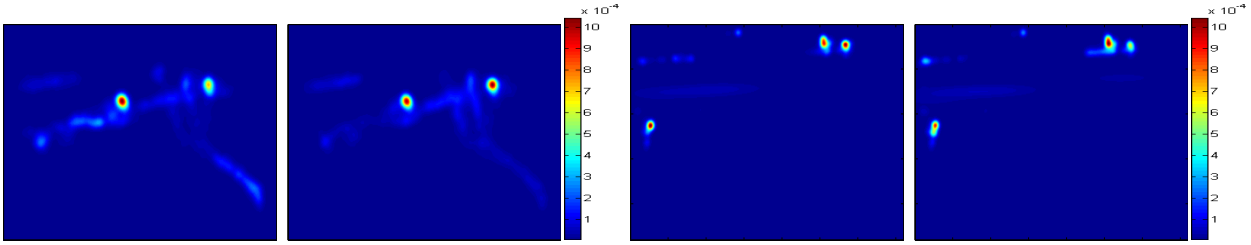
The selection of the kernel bandwidth $\mathbf{H}$ is the single most important criterion in kernel density estimation. Asymptotically, the selected bandwidth $\mathbf{H}$ does not affect the estimate but in practice sample sizes are limited. Theoretically, the ideal or optimal $\mathbf{H}$ that balances the bias and variance globally can be computed by minimizing the mean-squared error, $MSE\{\hat{p}_{\mathbf{H}}(\mathbf{z})\} = E\{[\hat{p}_{\mathbf{H}}(\mathbf{z}) - p_{\mathbf{H}}(\mathbf{z})]^2\}$, where $\hat{p}$ is the estimated density, p is the true density, and the subscript $\mathbf{H}$ indicates the use of the bandwidth $\mathbf{H}$ in computing the density. However, the true density p is not known in practice. Instead, various heuristic approaches have been proposed for finding $\mathbf{H}$ (See [76] for a survey of these approaches). We use the *Rule of Thumb* method [68], which is a fast standard deviation based method that uses the Asymptotic Mean Integrated Squared error (AMISE) criterion,

$$AMISE\{\mathbf{H}\} = \frac{1}{n} R(K) |\mathbf{H}|^{-\frac{1}{2}} + \int_{R^5} [(K_{\mathbf{H}} * p)(\mathbf{z}) - p(\mathbf{z})]^2 dz, \tag{3.3}$$

where $R(K) = \int_{R^5} K(\mathbf{z})^2 dz$ and $*$ is the convolution operator. The data-driven bandwidth selector is then, $\mathbf{H} = \arg\min_{\mathbf{H}} \widehat{AMISE}\{\mathbf{H}\}$. To reduce the complexity, $\mathbf{H}$ is assumed to be a diagonal matrix, i.e., $\mathbf{H} = diag[h_1^2, h_2^2, \ldots, h_d^2]$. We use a product of univariate Gaussian kernels to generate $K(\mathbf{z})$, i.e.,

$$K_u(z_{\{j\}}) = \frac{1}{h_j \sqrt{2\pi}} exp\left(-\frac{z_{\{j\}}^2}{2{h_j}^2}\right), \tag{3.4}$$

where, h_j is the j^{th} non-zero element of $\mathbf{H}$, and K_u is centered at $z_{\{j\}}$. It is emphasized here, that using a Gaussian kernel does not make any assumption on the scatter of data in the feature space. The kernel function only defines the effective region in $5d$ space in which each data point will have an influence while computing the probability estimate. Each time a pair of observations satisfying



(a) (b) (c) (d)

Figure 3.3: These maps represent the marginal probability of an observation, (a) reaching each point in the image, and (b) starting from each point in the image, in any arbitrary duration of time for the scene in Fig. 3.1a. Figs. c and d show similar maps for the scene in Fig. 3.1b.

the described criteria are available during training, the observed feature $\mathbf{z}$ is added to the sample. $p(X, Y, \tau)$ then gives the probability of a transition from point X to point Y in τ milliseconds.

Fig. 3.3 illustrates the maps of $p(X, Y, \tau)$ marginalized over $2d$ vectors X and Y, i.e., Fig. 3.3a represents the probability of an object reaching a point in the map from any location, i.e., $\int_X \int_\tau p(X, Y, \tau)\, d\tau dX$. It is important to note that the areas with higher densities (brighter colors) cannot be labelled as entry or exit points. The intensity at a point in the image only illustrates the marginal probability that an observed state transition has, of $starting$ or $ending$ at that point. The dominant points in the maps only mean that a higher number of object observations were made at these points. For example, in practice, this could possibly mean that these areas are places where people stand or sit while exhibiting nominal motion but significant enough to be noticed by the tracker. Similarly, each point in Fig. 3.3b represents the probability of an object starting from that particular point and ending anywhere in the image, i.e., $\int_Y \int_\tau p(X, Y, \tau)\, d\tau dY$. Notice that the similarity in the two maps manifests a correlation between incoming and outgoing object transitions at the

same locations. It should also be pointed out that since $p(X, Y, \tau)$ is a high dimensional space, the probabilities from a particular location to another location in a specific interval can be very low and a high dynamic range is required to display these maps correctly. So the dark blue regions in these maps are not necessarily absolutely zero. It should be noted that many dark blue regions in these maps can have relatively higher probabilities of transition, if the pdf is marginalized over specific time intervals or a specific starting or ending point, as discussed later. Figures 3.3c and 3.3d show similar maps for the scene in Figure 3.1b.

The probability density function $p(X, Y, \tau)$ now represents the distribution of transitions from any point X in the scene to any other point Y given the time taken to complete the transition as τ. Our next step is to be able to predict the $next$ state of an object, given the $current$ state and an approximate time duration of the jump. This is achieved by sampling the marginal of this distribution function as described in the following subsection.

3.1.2 Construction of Predicted Trajectories

The learned model can be used to find a likely next state given the current state of an object. The motivation is that we want to construct a likely trajectory from the current object position onwards, which would act as the predicted behavior of the object. Instead of finding the probabilities of transitions to different locations, we use the learned distribution as a generative model and $sample$ feature points from the model to construct a sequence of tracks based on the training observations. Thus, the predicted track is propagated using the sampled candidates.

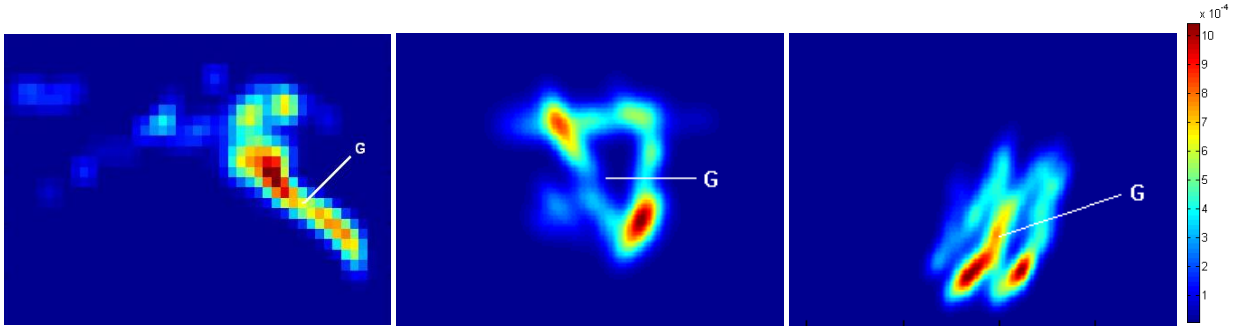


Figure 3.4: Regions of maps that represent the probability of reaching any state in the image from the state G.

Once the distribution has been learned, each track Ω can be viewed as a Markov process, $\Psi_k : \Omega \longrightarrow I$, on the image lattice I, where $k \in \{1, 2, \ldots\}$. By marginalizing out τ or integrating $p(X, Y, \tau)$ over an appropriate value of τ, the learned distribution can then be seen as the transition probability model, i.e., $\hat{p}\left(X_{t+1}|X_t\right)$, where X_t is the current state and X_{t+1} is the next state. Fig. 3.4 illustrates the marginal probability of reaching any state Y in the image starting from state X = G, in any arbitrary duration of time, and can be written as $\int_\tau p(X = G, Y, \tau)\, d\tau$. The relatively bright spots further away show the $possible$ states that a series of transitions can lead to. These correspond to higher values of τ, the time dimension of the distribution, for which the probability of transition from G is obviously low. These maps do not represent the whole scene, but just small regions of it.

Another of this paper's main contributions lies in the method we propose for computation of future states. The simplest approach to generate a likely state would be to choose some location in the neighborhood of the current state and find the probability of a transition between them using the learned distribution. But for practical purposes this approach seems quite naive. Many questions

need to be answered before this approach can be useful, for example, what should be the threshold probability for a prediction to be accepted and what should be size of the neighborhood. Even if these questions are answered, it should be noted that we do not want the most likely next state, rather a possible next state.

To compute a $next$ state given the $current$ state, we propose sampling from the learned transition distribution. Instead of sampling only one prediction, we propose sampling multiple distinct future states from the distribution so that the multi-modal nature of the learned density is fully utilized.

In real world scenarios, the assumption that tracks are first order Markov processes is not usually valid. It can however be noted that in our scene traffic model, the choice of different values of τ (the time duration required to complete a transition), can be used to achieve higher order Markov chains. Furthermore, while dealing with applications such as track filtering, anomaly detection and tracking through occlusions, it must be noted that sampling high probability future states is not always enough, for example in human traffic scenarios where it is possible for people to take the less travelled but possible paths, which we do not want to ignore in our algorithm. So instead of only one prediction for next location, multiple distinct future states are sampled from the distribution.

The track is initialized with any location and the representative future states are initially chosen with uniform distribution in neighborhood of the current state with equal weights. At any given step the weighted mean of all these will give the next candidate state. The transitions of this random walk will not always be the most likely ones but rather possible ones, that are in accordance with

Initialize X_0*; set* $i = 0$.
Repeat,
Let τ *be the time elapsed between* X_{i-1} *and* X_i.
Repeat for each sample u_{i-1}^k*, where* $k \in \{1, 2, \ldots, N\}$,
Let $\acute{Y} \sim q(.|\theta_{i-1}, \tau)$*,* q *being the Gaussian distribution* $\mathcal{N}(\theta_{i-1}, \Sigma)$ *where* θ_{i-1} *is the mean, and covariance matrix,* $\Sigma = diag[\tau^2, \tau^2]$.
Let $r \sim U(0, 1)$.
Let $\hat{p}(\acute{Y}) = \int_X \int_\tau p(X, Y = \acute{Y}, \tau)\, d\tau dX$ *and* $\hat{p}(u_{i-1}^k) = \int_X \int_\tau p(X, Y = u_{i-1}^k, \tau)\, d\tau dX$.
Compute $\alpha(u_{i-1}^k, \acute{Y}, \tau) = \min\left(1, \frac{\hat{p}(\acute{Y}) q(u_{i-1}^k | \acute{Y}, \tau)}{\hat{p}(u_{i-1}^k) q(\acute{Y} | u_{i-1}^k, \tau)}\right)$.
If $r \leq \alpha(u_{i-1}^k, \acute{Y}, \tau)$*, set* $u_i^k = \acute{Y}$*, otherwise, set* $u_i^k = u_{i-1}^k$.
Set $X_i = \sum_{m=1}^N u_i^m w_i^m$*, where* w_i^m *is the weight associated with* u_i^m.
Set $i = i + 1$.

Figure 3.5: Metropolis Hastings Sampling. See section 3.4 for details about the choice of q, the proposal density.

the training data. A sample set containing N future states is sampled from the distribution to represent $P(X_0)$ using the algorithm described in Fig. 3.5. These samples can be written as, $\left\{\left(u_0^{k,-}, w_0^{k,-}\right)\right\}$, where $u_0^{k,-} \sim \mathcal{N}(X_0, \Sigma)$, $w_0^{k,-} = \frac{1}{N}$, N is the number of possible future states, $\Sigma = diag[\tau_{max}^2, \tau_{max}^2]$, and $k = \{1, \ldots, N\}$. τ_{max} is the maximum possible time, which we seek to sample transitions from. The value of τ_{max} is set to the average duration in which a person takes a step. This will correspond to maximum time allowed for a sampled transition to occur. The symbols '$-$' and '$+$' indicate the representation of the state before and after the measurement has arrived respectively. Since only N states are used to represent the whole distribution given X_0 as starting state, weights $w^{1,\ldots,k}$ are normalized such that $\sum_i w_i^{k,+} = 1$, thus $w_0^{k,-} = \frac{1}{N}$, initially. At each step i, we have the representation of the probability density function $P(X_i | X_0, \ldots, X_{i-1})$ in the form of a set of future states $\left\{\left(u_i^{k,-}, w_i^{k,-}\right)\right\}$, where $u_i^{k,-} = f\left(u_{i-1}^{k,+}\right)$, and $w_i^{k,-} = w_{i-1}^{k,+}$, f is a function that samples from state transition distribution according to the conditional probability

$P(.|u_{i-1}^{k,+})$ using the Metropolis-Hastings algorithm, which is described in detail in Fig. 3.5. Given this representation of $P\left(X_i|X_0,\ldots,X_{i-1}\right)$, we update it to $\left\{\left(u_i^{k,+},w_i^{k,+}\right)\right\}$ as, $u_i^{k,+}=u_i^{k,-}$ and,

$$w_i^{k,+}=w_i^{k,-}\int_0^{\tau_{\max}} p\left(X_{i-1},u_i^{k,-},\tau\right)d\tau. \tag{3.5}$$

An estimate of X_i is now given by the weighted mean of all the sampled future states as, $X_i=\sum_{m=1}^{N} u_i^{m,+}w_i^{m,+}$, where X_i serves as the distribution's prediction until the $measurement$ from the tracker arrives which is denoted by ω_i (not to be confused with w_i, which is the weight of a particular sample). The update step then involves the computation of the final estimate of the object's new location, θ_i as a weighted mean of X_i and ω_i as,

$$\theta_i=\frac{X_i p(X_i|\theta_{i-1})+\omega_i p(\omega_i|\theta_{i-1})}{p(X_i|\theta_{i-1})+p(\omega_i|\theta_{i-1})}, \tag{3.6}$$

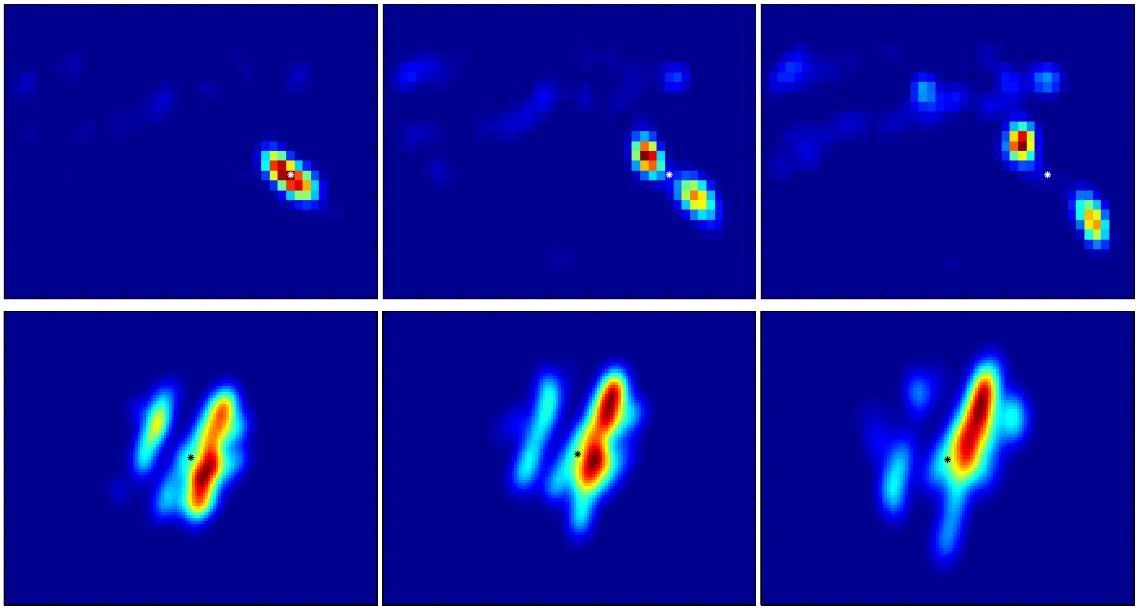
where the probabilities of transition from the previous state estimate θ_{i-1} to the new potential states X_i and ω_i serve as the weights, which are normalized so they sum to 1.

After update, the weights are normalized again so that $\sum_i w_i^{k,+}=1$ and their variance σ_w^2 is calculated. If $\sigma_w^2>\sigma_{th}^2$, a predefined maximum threshold, then a new set of samples is constructed by drawing (sampling) without replacement from the distribution using θ_i as current position. The probability of the sample being drawn is assigned as the initial weights. Thus the predicted states with relatively lower weights are discarded and others are retained to represent the next state. This process is continued to propagate the trajectory.

The choice of $\tau_{\max}$ depends on desired maximum duration between two consecutive points on a track. Assuming first order Markov process in a human traffic environment, $\tau_{\max}$ corresponds to the average time taken by a person to take one step. For most of the experiments reported in

this paper, the value of $\tau_{\max}$ was chosen to be 800 milliseconds since it was empirically found to be the average time taken by a person to take one step. The importance of this value is illustrated in Fig. 3.6. Different values of τ can be used to specify the time interval in transition from X to Y. Fig. 3.6a, 3.6b and 3.6c depict the marginal probabilities of an object starting from G shown by a $*$, and reaching any state in the image in 0-1.5, 1.5-3 and 3-5 seconds respectively. These maps can be represented mathematically as $\int_{0}^{1500} p(X = G, Y, \tau)d\tau$, $\int_{1500}^{3000} p(X = G, Y, \tau)d\tau$, and $\int_{3000}^{5000} p(X = G, Y, \tau)d\tau$. The maps show the paths that objects usually follow in these areas. So the time dimension of the learned distribution encodes additional information that is vital to solving short-term occlusions using the distribution as described in Section 3.2.4. Notice that the modes or the areas of dominant probability in general are not equidistant from G. Furthermore, the locations of these modes and their distances from G can be different for different choices of G depending on the usual speeds of objects in that area, which is in contrast to tracking using Gaussian or other symmetric distributions as process noise functions.

We now have a mechanism of sampling from the distribution that enables us to predict object behavior in space and time, which is then used for likely (typical) path generation, anomaly detection, and persistent tracking of objects through occlusions as described in the next section. The learned transition probabilities are also used to improve Foreground detection.



(a) (b) (c)

Figure 3.6: Marginal probability maps of the learned pdf at different time intervals. The maps represent the probability of reaching any state in the image from the state G (shown by white $*$ in first and black $*$ in second row) in (a) 0-1.5 seconds, (b) 1.5-3 seconds (c) 3-5 seconds. Each row shows a different starting state in the image. These maps show only small regions of the actual scene.

3.2 Applications of Proposed Model

It is important that motion patterns of objects be modeled in a way that enables solution of multiple problems that are encountered in scene analysis and surveillance. We next describe some of these problems and application of the proposed model and state estimation framework for their solution.

The flow of information through the proposed framework is shown in Fig. 3.7.

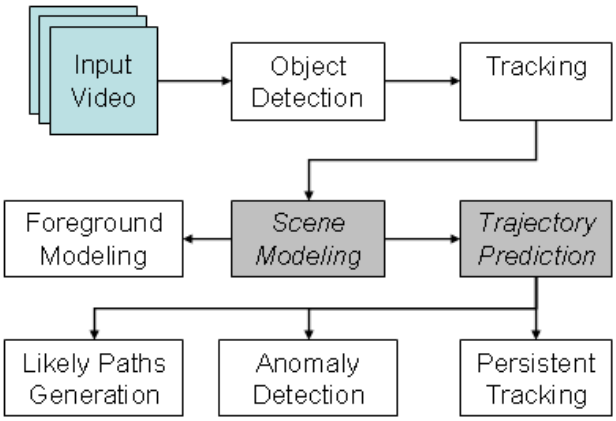


Figure 3.7: Different modules in the system and the data flow between them.

3.2.1 Generating Likely Paths

An important aspect of modeling traffic patterns is generation of likely paths. Given the current location of an object, such a simulated path amounts to a prediction of future behavior of the object. Furthermore, our model allows us to assign a confidence measure to such likely paths since the probability of each transition in the path is available. Likely paths can be loosely defined as the relatively commonly adapted trajectories in a scene, which may or may not have been observed during training and can be perturbations of the actual observed trajectories.

We seek to sample from the learned model of transition patterns to generate behavior predictions. We expect that only a few number of paths should adequately reflect observations of trajectories through walkways and roads, etc. The sampling method is used as described in section 3.1.2 except that there are no $measurements$ available which translates to acceptance of the weighted mean of all samples at each step as the next state. More specifically, equation 3.6 transforms to $\theta_i = X_i$. For our experiments we start by manually choosing X_0 as the initial state in a

region of significant traffic. However, the initial state can be chosen automatically by computing the probability map of a marginal of $p(X, Y, \tau)$ over Y and τ which will reveal the regions with high probability of having a start state, as shown in Fig. 3.3. The trajectory is propagated by accepting $predicted$ states as $next$ states. The chain is stopped whenever $\sigma_w^2 > \sigma_{th}^2$, that is, there is a need to resample candidate predictions or when the Metropolis-Hastings algorithm chain fails to move for more than a specified number of iterations, e.g., we used a maximum of 200 iterations for this purpose.

3.2.2 Improvement of Foreground Detection

The intensity difference of objects from the background has been a widely used criterion for object detection, but it can be noted that temporal persistence is also an intrinsic property of the foreground objects, i.e. unless an object exits from the scene or becomes occluded, it has to either stay at the same place or move to a location within the spatial vicinity of the current observation. Since our proposed transition model incorporates the probabilities of movement of objects from one location to another, it can be used to improve the foreground models. We now present the formulation for this application. It should be noted however that this method alone cannot be used to model the foreground. Instead it needs to be used in conjunction with an appearance based model like mixture of Gaussians [70]. We seek to find the likelihood of a pixel u belonging to the foreground at time instant t. Let $\mathbf{u}$ be a random variable which is true if and only if the pixel u is a foreground pixel. Also, let Λ be the feature set used to model the background (for example, color/gray scale in appearance based model), and $\Phi = \{\phi_1, \phi_2, \ldots \phi_Q\}$ be the set of pixels detected

as foreground at time instant $t-1$, where Q is the total number of the foreground pixels in the previous frame. Then according to Bayes rule,

$$P(\mathbf{u} = true|\Lambda, \Phi) \propto P(\Lambda, \Phi|\mathbf{u} = true)P(\mathbf{u} = true). \tag{3.7}$$

Assuming independence of appearance Λ and the foreground history Φ, we may write,

$$P(\mathbf{u} = true|\Lambda, \Phi) \propto P(\Lambda|\mathbf{u} = true)P(\Phi|\mathbf{u} = true)P(\mathbf{u} = true). \tag{3.8}$$

Here we use the framework of aggregating expert opinions [3] to combine the decisions from different features. Specifically, we consider logarithmic opinion pooling [27], i.e.,

$$P(\Lambda, \Phi|\mathbf{u} = true) \propto P(\Phi|\mathbf{u} = true)^{\beta} P(\Lambda|\mathbf{u} = true)^{1-\beta}, \tag{3.9}$$

where, the weight β represents the expert's reliability. We chose $\beta = \frac{Q}{Q+Q_a}$, where Q is the number of pixels classified as foreground in the previous frame, while Q_a is the number of foreground pixels detected in the last frame using only appearance based model. The reason for this choice of weight is that while motion patterns can help deduce that a pixel is *not* a part of the foreground, they cannot be used to infer the presence of an object. In other words, using the learned density should typically help reduce the number of false positives but may not affect the number of false negatives to a large extent.

The KNIGHT object detection and tracking system ([36]) is used to extract appearance based likelihood, $P(\Lambda|\mathbf{u} = true)$. KNIGHT performs background subtraction at multiple levels using statistical models of gradient and color data. Specifically, a mixture of Gaussians model is used with 4 Gaussians per pixel. The learned pdf $p(X, Y, \tau)$ is used to estimate the remaining terms as

follows:

$$P(\Phi|\mathbf{u} = true) = \sum_{j=1}^{Q} p(\phi_j, u, \Delta t), \tag{3.10}$$

and the marginal (prior) of u being a foreground pixel is,

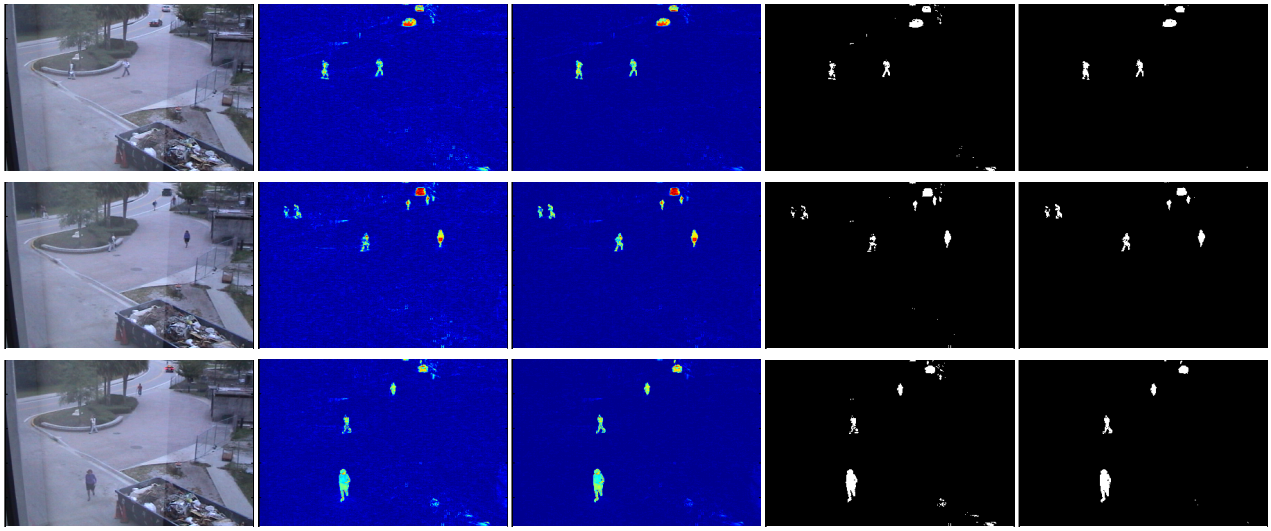
$$P(\mathbf{u} = true) = \int_X \int_\tau p(X, Y = u, \tau) d\tau dX. \tag{3.11}$$

The value of Δt represents the length of time between two consecutive frames. It should be pointed out here that although not all previous foreground pixels should contribute significantly to the pixel u, the value of Δt is typically small enough to inhibit contributions from far away pixels. In other words, only the pixels in Φ that are spatially close to u chip in substantial values to the sum.

Fig. 3.8 shows the results of the proposed foreground model. It can be seen that the number of false positives has decreased and the true silhouettes are much more pronounced. The scheme presents a simple and effective way of reducing errors and improving background subtraction. Many of the commonly encountered false detections like motion of static background objects are removed using this approach. More experiments and a quantitative analysis are reported in Section 3.3.

3.2.3 Anomaly Detection

If tracking data used to model the state transition distribution spans sufficiently large periods of time, it is obvious that a sampling algorithm will not be able to sample a track that is anomalous considering the usual patterns of activity and motion in that scene. Once the model has been



(a) (b) (c) (d) (e)

Figure 3.8: Foreground Modeling: Each row shows an example of improvement on the foreground probability estimation. (a) original image, (b) probability map using only the mixture of Gaussians method and, (c) foreground probability map using proposed model in conjunction with the mixture of Gaussians model, and columns (d) and (e) show foreground masks obtained using the maps in (b) and (c) respectively. Columns (b) and (c) show an improvement of foreground probability modeling while (d) and (e) show reduction in the number of false positives.

learned over a reasonably large period of time, it is representative of the traffic patterns in the scene. This observation forms the basis of our anomaly detection algorithm.

The anomaly detection algorithm generates its own predictions for future states using the method described in section 3.1.2 without using the current observation of the tracker. It then compares the actual measurements of objects with the predicted tracks and computes a difference measure between them. Let the set of predicted states of an object be Θ and the actual measurements as observed by the tracking algorithm be Ω. Using the framework as described earlier, we can compute $\Theta = \{\theta_1, \theta_2, \ldots, \theta_{i+1}\}$, which is the predicted or estimated state. Notice that

at step i, we have a prediction for location at step $i+1$. Then the observed track described by $\Omega = \{\omega_1, \omega_2, \ldots, \omega_i\}$ is labelled as anomalous if, $d_i > d_{th}$, where,

$$d_i = \frac{1}{n+1} \sum_{j=i-n}^{i} \left((\omega_j - \theta_j)^T \Sigma_\Theta^{-1} (\omega_j - \theta_j) \right)^{\frac{1}{2}}, \tag{3.12}$$

where Σ_Θ is covariance matrix of x and y coordinates of all the potential (candidate) next states, n is the number of previous states included in calculation of Mahalanobis distance, and d_{th} is a predefined maximum difference between observed and predicted tracks. Furthermore, the plot of unnormalized distances between observed anomalous and predicted trajectories versus the number of states, n, used in computation of the distance is sublinear with respect to n.

3.2.3.1 Types of Anomalies

Using track prediction described in section 3.1.2, the distance between actual (observed) and usually adapted (predicted) paths can be determined. This translates into classification of unusual behavior owing to mere presence of objects in unusual areas in the scene, i.e., those areas of the scene that are most often not traversed. This type of anomalous tracks are $spatially$ incoherent relative to the commonly observed patterns of motion in the scene. Another important type of unusual activity is related not to the location but to the speed with which the activity is being performed. If the scene is that of human traffic, it is likely that variance among the speeds of different people will not be very significant and hence average speed or time required per step ($\tau_{\max}$ in section 3.1.2) can easily be learned. An object whose speed varies significantly as compared to this average speed is more likely to be engaged in a suspicious activity. The same notion applies to the vehicle traffic as well. In this case the predicted track either falls behind or hurries ahead of

the actual observations. Examples of this kind of anomalous behavior includes persons running or riding a bicycle in an area where such kind of activities do not usually take place. The value of n in equation 3.12 is the number of previous consecutive states that are used in computation of the distance between the tracks. Increasing the value of n helps detect suspicious behavior where activity is not necessarily unusual in either spatial or temporal domains but the object has been in view for a long time and error (distance between actual and predicted paths) has accumulated and crossed the d_{th} barrier. Tracks of objects moving with even slightly slower or faster than usual speeds for an unusually long period of time can be labelled anomalous with this scheme. This proposed approach for anomaly detection is able to handle all kinds of anomalies that can possibly be detected using only the track information. It should be noted that the observed tracks information, received from the tracker has timestamps for each instance of object detection, i.e., time taken for each transition by an object is known.

3.2.4 Persistent Tracking through Occlusions

Finally, we present application of the proposed scene model for persistent tracking through occlusions and completion of the missing tracks. Persistent tracking requires modeling of spatio-temporal and appearance properties of the targets. Traditionally, parametric motion models such as, constant velocity or constant acceleration, are used to enforce spatio-temporal constraints. These models usually fail when the paths adapted by objects are arbitrary. The proposed model of learning traffic parameters is able to handle these shortcomings when occlusions are not permanently present in the scene and the patterns of motion through these occlusions have previously been

learned, e.g., person to person occlusions, large objects like vehicles that hide smaller moving objects from view. We use the proposed distribution to describe a solution to these problems.

Let $\hat{p}(Y|X) = \int_{\tau} p(X, Y, \tau)\ d\tau$ and Ω be an observed track. A sample set containing N samples is generated from the learned pdf to represent $\hat{p}(X_0)$ where X_0 is the initial location of the track. These samples are represented as $\left\{\left(u_0^{k,-}, w_0^{k,-}\right)\right\}$, where $u_0^{k,-} \sim \mathcal{N}(X_0, \Sigma)$, $\mathcal{N}$ represents Gaussian distribution with mean X_0 and covariance matrix, $\Sigma = diag[\tau^2, \tau^2]$, and $k = \{1, \ldots, N\}$. For the initial set of samples, the weight is assigned as,

$$w_0^{k,-} = \frac{\int_X \hat{P}\left(u_0^{k,-}|X\right) dX}{P_\gamma\left(\Gamma = u_0^{k,-}\right)}, \tag{3.13}$$

where, Γ is a random variable that represents Gaussian distribution in the neighborhood of X_0. The symbols '$-$' and '$+$' indicate the representation of the state before and after availability of the measurement respectively. Since only N samples are used to represent the whole distribution, weights $w^{\{1,\ldots,k\}}$ are normalized such that $\sum_i w_i^{k,+} = 1$. At each step i, a representation of the probability density function $\hat{p}\left(X_i|\omega_0, \omega_1, \ldots, \omega_{i-1}\right)$ is retained in the form of weighted candidate samples $\left\{\left(u_i^{k,-}, w_i^{k,-}\right)\right\}$, where $u_i^{k,-} = f\left(u_{i-1}^{k,+}\right)$ and $w_i^{k,-} = w_{i-1}^{k,+}$. f is a function that samples from the state transition distribution according to the conditional probability $\hat{P}(.|u_{i-1}^{k,+})$ and the smoothness of motion constraint using the Metropolis-Hastings algorithm, which is described in detail in Fig. 3.5.

At any time, we want to be able to find correspondences between successive instances of objects. This task proves to be very difficult when the color and shape distributions of objects are very similar or when the objects are in very close proximity of each other, including in person

to person occlusion scenarios. However, in the proposed model, a hierarchical use of transition density enables our scheme to assign larger weights to more likely paths. Assume now that the observed measurements of m objects at time instant i are given by $\Omega = \{\omega_i^1, \omega_i^2, \ldots, \omega_i^m\}$, and the predicted states of previously observed s objects are $\Theta = \{\theta_i^1, \theta_i^2, \ldots, \theta_i^s\}$ respectively. We must find the mapping function from the set $\{\Omega_i^k\}$, $1 \leq k \leq m$ to the set $\{\Theta_i^l\}$, $1 \leq l \leq s$. To establish this mapping function, we use the graph based algorithm proposed in [66]. The weights of the edges corresponding to the mapping Θ_i^l and Ω_i^k are given by the Mahalanobis distance between the estimated position θ_i^l and the observed position ω_i^k, for each l and k. Once the correspondences are established, for each object, given the representation of $\hat{p}\left(\theta_i|\theta_0, \ldots, \theta_{i-1}\right)$, and the corresponding new observation ω_i, we update it to $\left\{\left(u_i^{k,+}, w_i^{k,+}\right)\right\}$ as $u_i^{k,+} = u_i^{k,-}$ and,

$$w_i^{k,+} = w_i^{k,-} \sum_{k=1}^{N} \hat{P}\left(\omega_i|u_i^{k,-}\right) = \prod_{j=0}^{i} \sum_{k=1}^{N} \hat{P}\left(\omega_j|u_j^{k,-}\right). \tag{3.14}$$

The product of old weight with the sum of probabilities of reaching ω_i from each sample state $u_i^{k,-}$ for all k, translates into a product of transition probabilities for all jumps taken by an object. This results into assignment of larger weights for higher probability tracks and in the next time step, this weight helps establish correspondences such that the probability of an object's track is increased. Once updated, the candidate locations are re-sampled as described in the section 3.1.2. This approach enables us to simultaneously solve short-term occlusions and person to person occlusions, as well as persistent labeling of objects.

Table 3.1: Datasets: Observations list the number of times any object was detected and feature samples are the number of 5d features computed and added to the density estimate (regardless of resampling). Learning Time is the time taken to compute the samples and generate their kernel density estimate.

Sequence	**Resolution**	**Duration**	**Observations**	**# of Samples**	**Learning Time**
Scene 1	360 x 240	3 days (approx.)	3.9 million	936,739	194 hrs (approx.)
Scene 2	320 x 240	6 hr 20 min	253,697	66,498	13 hrs (approx.)

3.3 Results

In this section, we present our experimental results. The collected tracking data is derived from two video sequences taken at the scenes shown in Fig. 3.1. Some figures related to these datasets and their density estimates are summarized in table 3.1. The video sequences were acquired by static cameras, and the scenes consist of high human traffic with occlusions and both commonly and sparingly used paths. Initial tracking is done using the algorithm in [36]. As can be seen in Fig. 3.1, there are numerous entry and exit points in the scenes and multiple paths can be chosen between different pairs of entry and exit points.

The organization of this section is as follows: First the results of our methods for generation of likely paths and improvement of foreground modeling and detection are presented. The results of our experiments on anomaly detection are then reported. Finally we show a few examples of persistent tracking of objects through occlusions using the proposed distribution.

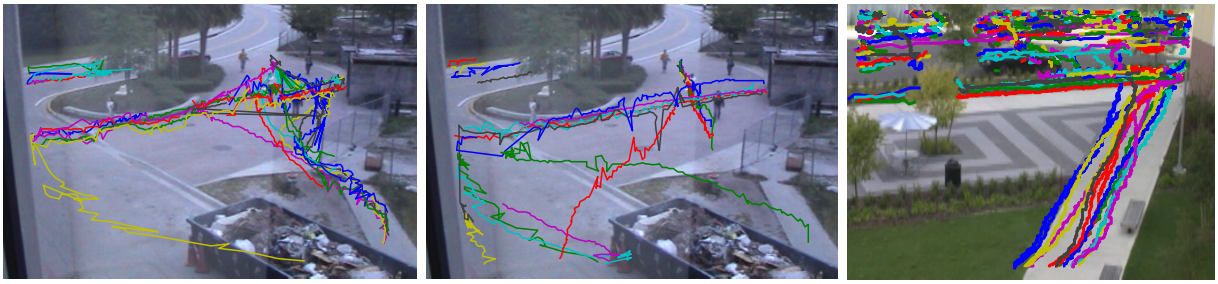


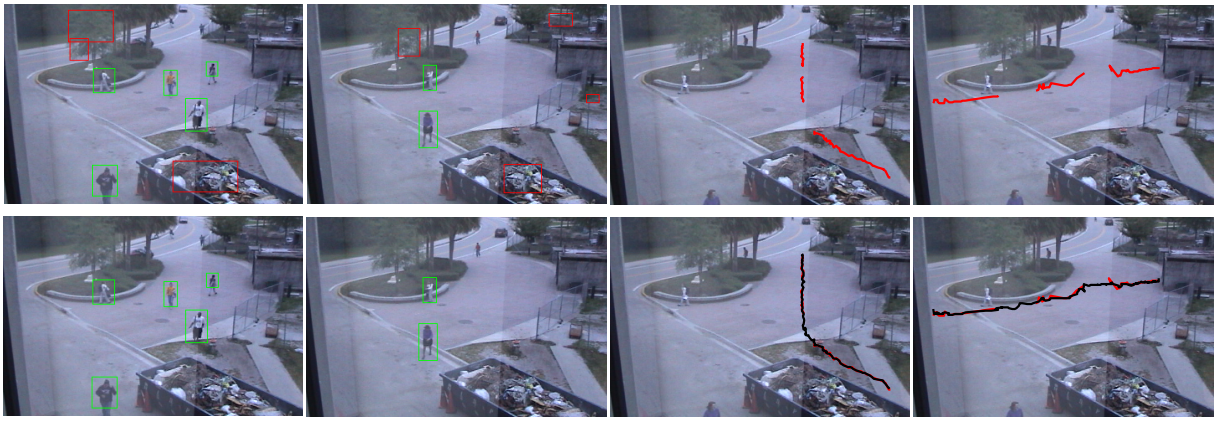
Figure 3.9: Examples of likely paths generated by the proposed algorithm using Metropolis-Hastings sampling. Tracks are initialized from random locations selected manually.

3.3.1 Likely Paths Generation

As described in Section 3.2.1, starting at random initial states in the image, sampling from the distribution gives possible paths that are usually followed by the traffic. Fig. 3.9 shows some of these random walks. It should be noted that no observations from the tracking algorithm have been used in this experiment. The likely paths generated are purely simulated based on the learned distribution. It can however be seen in Fig. 3.9 that some tracks are not very smooth and loop at some locations. But we want to point out here that it is not our purpose to generate smooth paths, rather a depiction of actual tracking scenarios. A comparison of Fig. 3.9 with figure 3.1b, that shows $actual$ tracks used during training, proves the similarity of both and validates the proposed model and sampling method.

3.3.2 Foreground Detection

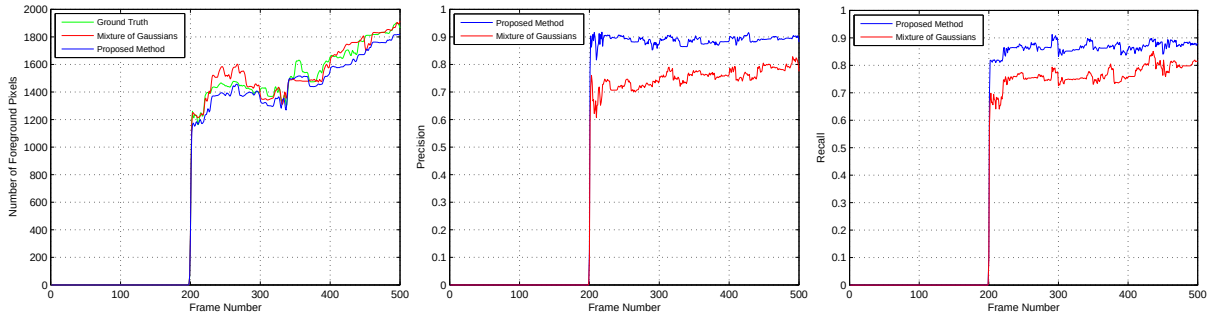
A scheme for improving detection of foreground objects using the proposed model was described in section 3.2.2. Fig. 3.10 presents the results of that scheme. The top row of columns (a) and



(a) (b) (c) (d)

Figure 3.10: Foreground Modeling Results: Columns (a) and (b) show results of object detection . The images in top row are without and bottom row are with using the proposed model. Green and red bounding boxes show true and false detections respectively. (c) and (d) show results of blob tracking. Red tracks are original broken tracks and black ones are after improved foreground modeling.

(b) show images that represent the results of the object detection algorithm of [36]. It can be seen in these images that parts of trees, garbage dump, and grass have been detected as objects due to nominal motion owing of wind. The false detections are shown by red bounding boxes. On the other hand, during the training phase, there were very few object detections in these areas, thus giving low probability of object presence at these locations. The bottom row of columns (a) and (b) show the results after mis-detections have been removed using the combination of appearance and proposed motion model. The scheme has removed these false positives owing to their low probability of transition or existence or both. Columns (c) and (d) show tracks from the tracking algorithm of [36] which are broken in different places because of a significant number

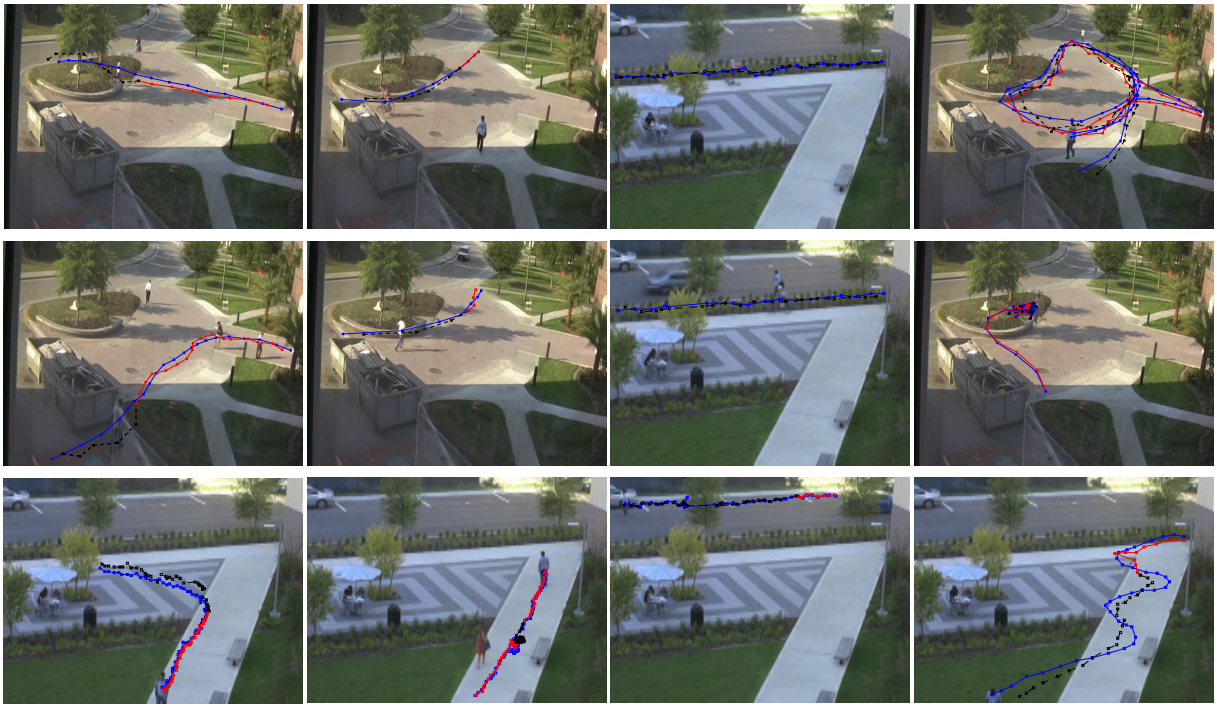


(a) (b) (c)

Figure 3.11: Foreground Modeling Results: (a) Number of foreground pixels detected by the mixture of Gaussians method, the proposed method in conjunction with the mixture of Gaussians method, and the ground truth. The pixel-level foreground detection recall and precision using the mixture of Gaussians approach only, and the proposed formulation are shown in (b) and (c) respectively. As shown, the combined model performs persistently better than the appearance only model in terms of both precision and recall.

of false negatives in foreground detection. Again these missed detections were corrected after incorporating the motion based foreground model.

For a quantitative analysis of the foreground modeling scheme, we manually segmented a 500 frames video sequence into foreground and background regions. The first 200 frames were used to initialize the mixture of Gaussians based background model. Foreground was estimated on the rest of the frames using i) mixture of Gaussians model, and ii) the proposed logarithmic pooling of appearance and motion models. Pixel level precision and recall were computed for both cases. The results of this experiment are shown in Fig. 3.11. It can be seen from the plots that both the precision and recall of the combined model are consistently better than that of the mixture of Gaussians approach alone. The number of false positives for the combined model was reduced by



(a) (b) (c) (d)

Figure 3.12: Results of Anomaly detection: (a) Spatially anomalous, (b) and (c) Temporally anomalous, and (d) Suspicious behavior due to presence over large distance or extended period. Blue track represents the actual (observed) track. Red and black tracks correspond to typical and atypical (anomalous) *predicted* paths respectively.

79.65% on average as compared to the mixture of Gaussians model. The average reduction in the number of false negatives was 58.27%.

3.3.3 Anomaly Detection

Fig. 3.12 shows results of our anomaly detection algorithm described in Section 3.2.3. The normal tracks adapted in each scene can be seen in Fig. 3.1 and Fig. 3.9. In the top row of Fig. 3.12a, a person walks through the paved region and then into the grass. The algorithm separates this trajec-

tory into typical and atypical parts. The blue track shown in the figure is the track of concern, red track is the area where the prediction follows observations while dotted black shows the prediction once an anomaly is detected. Notice how the blue track is closely followed by the red one as long as it is typical behavior. At the time of training, the dump trailer was at a different position as seen in Fig. 3.1a, resulting in classification of the track observed in middle row of Fig. 3.12a as anomalous. The bottom row of Fig. 3.12a shows another $spatially$ incoherent track through an area where no motion is usually observed.

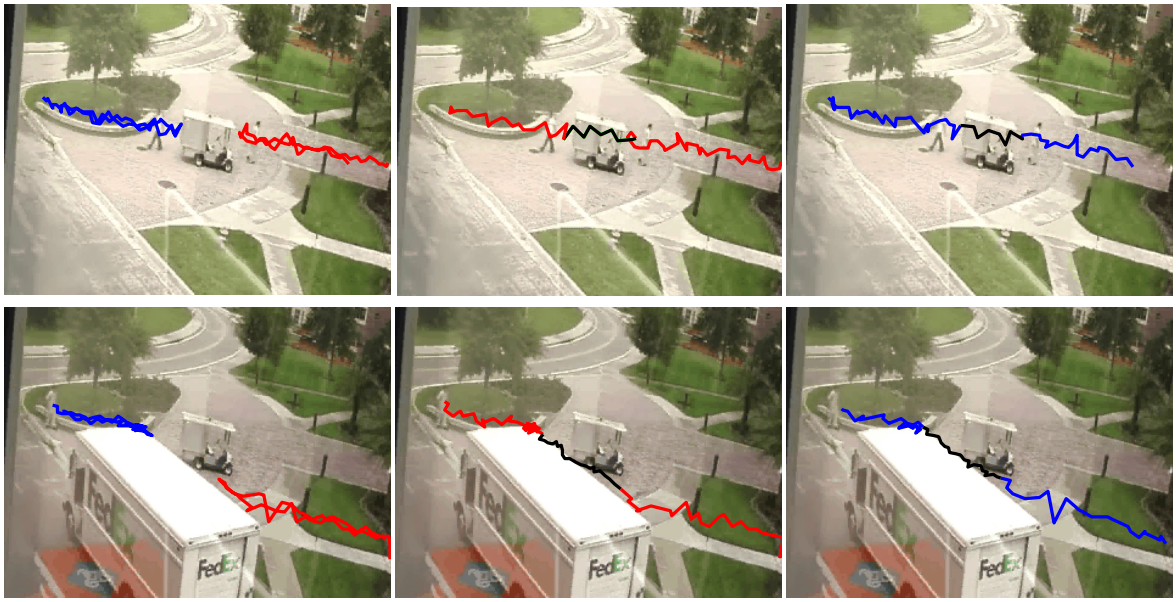
The top row of Fig. 3.12b shows anomaly due to the unusual speed of a person riding a bicycle in an area where people usually only walk. The middle row of Fig. 3.12b shows the results of our algorithm on another example where a person is running in the same paved area. Again, this behavior has not been observed in the tracks used for training because people usually do not run in this area. The bottom row of Fig. 3.12b shows a stopping action where a person sits down to tie their shoelaces. Three more examples of the second type of anomaly (temporal inconsistency) are shown in Fig. 3.12c. The top and middle rows in the third column show two persons skating and riding a bicycle respectively while the bottom row shows a person walking on the road where usually only cars are observed. Since the speeds of objects in Fig. 3.12b and Fig. 3.12c are significantly different from the average speed of objects during training, the prediction either lags behind as compared to the actual measurements from the tracker, or hurries ahead resulting in the observed trajectory being labelled anomalous. Three examples of the third type of anomalies are shown in Fig. 3.12d. In the top row of Fig. 3.12d a person is walking slowly in a circular path for a few minutes. The motion pattern itself is not anomalous to the learned distribution, but the

Table 3.2: Quantitative analysis of anomaly detection results using classification by 3 human observers as the ground truth. The proposed system labelled 14 of the 19 sequences as anomalous.

Ground Truth	**Anomalous**	**Normal**	**Precision**	**Recall**
Human 1	13	6	92.86%	100%
Human 2	15	4	100%	93.34%
Human 3	11	8	78.57%	100%

presence of the object in the scene for a relatively longer period of time results in accumulation of $error$ calculated as d_i using larger values of n in Eq. 3.12, and eventually becomes greater than d_{th} resulting in classification of the sequence as an anomaly. The middle row of Fig. 3.12d shows a person jumping over an area which people usually only sit on. Even though the person is not present in the scene for very long, the error d_i quickly becomes significant due to the incoherence of his actions relative to the training examples. The bottom row shows a a zigzag pattern of walking, a motion that has not been observed before, resulting in the classification of the trajectory as anomalous.

Since the decision as to whether a given trajectory is anomalous is subjective and differs from one observer to another, we asked three human observers to classify 19 sequences as either normal or anomalous. The proposed method of anomaly detection was run on these 19 sequences which labelled 14 of them as anomalous. The quantitative results of these experiments are summarized in Fig. 3.2.

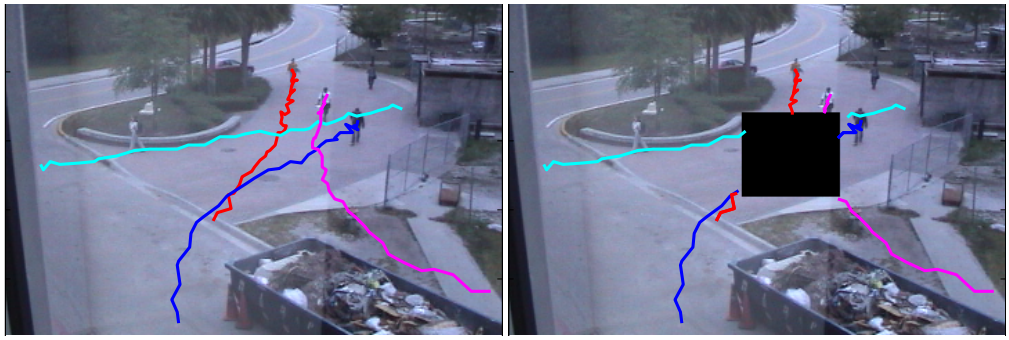


(a) (b) (c)

Figure 3.13: For each row, (a) shows the observed tracks in blue and red that have been labelled wrong, and (b) and (c) show the stitched part of tracks in black, and actual tracks in red and blue respectively.

3.3.4 Persistent Tracking

In Section 3.2.4, we described our method to track objects persistently through occlusions. In absence of measurements, our algorithm propagates the track based on samples from the distribution. Fig. 3.13 shows two examples of tracks that have undergone erroneous labeling by the tracking algorithm because both of them had considerable missing portions due to total occlusion. In the top row of Fig. 3.13, the tracker assumes that two objects walked towards the golf car and then returned to their original locations, which is not correct. Each of these tracks are propagated through predicted locations to join the correct track. The result of propagation based on weighted mean of these samples is shown in top row of Fig. 3.13b and Fig.3.13c, where both tracks are separately



(a) (b)

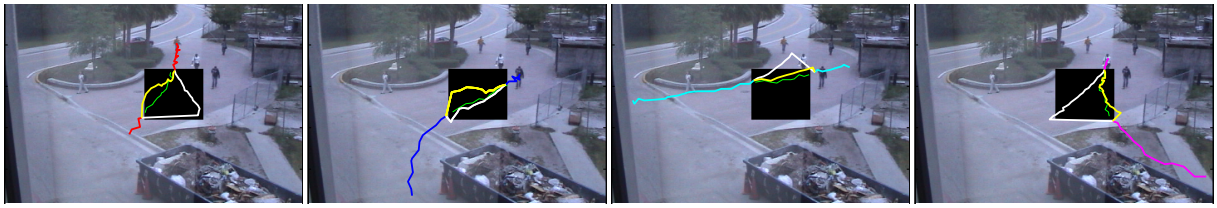
Figure 3.14: Example of persistent tracking for multiple simultaneous objects with overlapping or intersecting tracks undergoing occlusion. (a) Actual original tracks (ground truth) (b) Broken tracks due to simulated occlusion shown as black region.

shown for clarity. Red and blue colors show two different tracks and the black color shows the part where trajectory has been stitched. The bottom row of Fig. 3.13 shows a similar example where a truck is obstructing the camera's view. Here again the objects had undergone total occlusion due to the truck obstructing the camera's view. It should be pointed out here that such kind of persistent tracking is possible only in areas where traffic patterns have already been learned and the occlusions are only short lived. If an object like the truck in bottom row of Fig. 3.13 is occluding a region in the scene that contains crossroads, this scheme can fail in certain conditions. However, we would like to emphasize here that such scenarios will be difficult to resolve even by a human observer, unless shape or appearance distributions are available.

Another example of persistent tracking through occlusions is shown in Figs. 3.14 and 3.15, where we created a considerably large simulated occlusion in a small video sequence as shown in Fig. 3.14b as a black box in the middle of the image. The location of this occlusion makes the task challenging since most of the likely paths in the scene pass through this region. This sequence

contains four tracks that have significant portions under occlusion. For comparison with other algorithms, we also tested Kalman filter based tracking on the same sequence where the process noise function is assumed to be Gaussian, and trajectories are assumed to be locally linear, as in [62]. The results on the four tracks are shown separately in Fig. 3.15. The direction of movement in all tracks is from the top of the image downwards except in Fig. 3.15c where it is from left to right. The results of Kalman filter tracking is shown as white track, the proposed approach as yellow and the ground truth as green tracks. The ground truth trajectories are available since the occlusion is only simulated. The reason for the sharp turns at the end of the predicted tracks is the recovery from error, once the object comes out of occlusion. It is fairly obvious that Kalman filter based tracks tend to follow linear constant motion dynamics, which actually works better than our algorithm in Fig. 3.15b, but in the other three tracks, the proposed approach works significantly better than the Kalman filter. The reason for the deviation of the track generated using motion patterns from the ground truth (in Fig. 3.15b) is that the length of the track $before$ the occlusion is very small. As a result, the algorithm is unable to differentiate between the actual track that is coming from the top, and a track which goes from that position to the left. As a result, the predicted track goes left since the small track length makes it seem that the direction of movement is towards the left.

Finally, we report some of the run time performance data. All the experiments were ran on a Pentium-4 2.53 GHz machine and the coding was done in Matlab without any optimizations for speed. For track generation (used in likely tracks generation, anomaly detection and persistent tracking) the proposed algorithm performed at an average of 0.422 frames per second. The



(a) (b) (c) (d)

Figure 3.15: Results for scenario shown in figure 3.14. Green track is the ground truth. Tracking through occlusion using Kalman filter is shown in white and yellow tracks are generated using the proposed approach. Notice that both methods can recover well once the measurement is available again, but during occlusion the proposed method stays closer to ground truth.

Table 3.3: Mahalanobis distances of tracks generated using proposed approach and kalman filter based tracking, from the ground truth.

Track	**Proposed Approach**	**Kalman Filter**	**% Reduction**
Fig. 3.15a	74.34	165.22	55.01%
Fig. 3.15b	119.20	59.14	-50.39%
Fig. 3.15c	142.74	472.08	69.76%
Fig. 3.15d	52.06	628.69	91.72%

Metropolis-Hastings algorithm failed to sample a new state approximately 11.72% of the time. The average time taken to compute the probability of a transition ($p(X, Y, \tau)$) was 35.78 milliseconds but had a high standard deviation where the time taken was lower for areas with less traffic. The proposed foreground detection algorithm performed at an average of 0.637 frames per second. The performance of the algorithm using a histogram approximation of the kernel density estimate is reported in table 3.4.

Table 3.4: Performance of the proposed algorithm when using histogram approximation of kernel density.

Metric	**KDE**	**Histogram**	**% Improvement**
Track Generation (fps)	0.422	1.451	243.84 %
Metropolis Hastings Failure Rate	11.72 %	6.93 %	40.87 %
Time taken to compute probability (ms)	35.78	10.07	71.86 %
Foreground Detection (fps)	0.637	2.762	333.59 %

3.4 Discussion

In this section, we discuss some of the questions and concerns raised by the anonymous reviewers and the associate editor of the publication ([63]), based on the work proposed in this chapter . We formulate this section as a series of questions and comments, and their corresponding answers that were exchanged during the review process.

Comment 1: The proposed technique to handle problems of object association across short / long-term occlusions is not convincing especially when there are multiple objects within the scope of this association problem. The underlying track breaks may require sophisticated appearance matching over and above the kinematic constraints learned from the track distributions.

Response: We agree that appearance is a key feature for object association and do fully understand that successful object association requires both spatio-temporal and appearance features to complement each other. Traditionally, appearance models are used along with kinematic constraints (imposed by an assumed motion model, such as constant velocity or constant acceleration) for object association. The contribution of the paper is to learn these kinematic constraints directly from

the track distribution instead of assuming some motion model to be valid everywhere in a scene. The kinematic model learned this way not only includes the above mentioned motion properties but also embodies scene-specific knowledge, such as presence of paths, entry/exit points, and people preferences. Similar to traditional kinematic models, the proposed spatio-temporal model can be complemented with more sophisticated appearance matching to solve complex occlusions, e.g.., occlusions due to group motion.

Comment 2: Are you assuming ideal tracking? What influence does imperfect tracking have on your algorithm's performance?

Response: No, we are not assuming the input tracks to be ideal and like other realtime systems, KNIGHT system is also susceptible to errors due to illumination changes, shadows, and severe occlusions. However, the errors do not greatly affect the performance of the proposed algorithm because the system is trained for video sequences collected over extensive periods of time. Over time, the feature samples in the density estimate are re-sampled and outliers are rejected. Consequently, the true patterns of object dynamics are retained.

Comment 3: Although you state that the proposed algorithm is resilient to failures of the underlying tracker, it is not clear from the formulation that this is so. If a tracker, such as KNIGHT, tends to fragment tracks a lot, then the transition times beyond a small value will almost never get good support. This will not reflect the true distributions. It is not clear from the paper how this aspect is handled? Or is it handled at all?

Response: The proposed framework does not rely on tracking information. As a matter of fact, the distributions can be generated by using the detections alone (by assigning a link between each pair

of detection that occurs within a given time period and using it as a data point for the model). Over time, the actual links will persist and the noisy links will average out. The tracking information is used only when it is available to reduce the influence of noise on the model. For example, consider a high precision tracker that links two observations with high precision but may fragment tracks a lot. If the tracker establishes a link between two observations then we ignore other possible links. However, if such a link for an observation is not present then all possible pairs are considered (for example, for time interval of n frames, the first and last n points in a track do not have links established by the tracker). This way, the algorithm is resilient to failures of the underlying tracker.

Comment 4: If there is a static object, like a truck or a building, that occludes objects consistently over the course of their motion, how can the proposed algorithm connect these objects especially in situations when objects close to each other are moving, for instance groups of people?

Response: As mentioned in response to Comment 3, the algorithm does not fully rely on the track links and thus can connect objects across long term occlusions due to a static object. The algorithm however cannot solve occlusions caused by objects moving together in groups. Resolving such occlusions would require use of appearance features along with the proposed spatio-temporal model.

Comment 5: Are there some anomalous trajectories in your training dataset?

Response: There are some anomalous trajectories in our training sets, for example, people walking through the grassy areas. However, by definition, the anomalous trajectories are very sparse and hence have very low probability in the model as compared to normal trajectories or events.

Comment 6: The determination of d_{th} is tricky. How can it be determined automatically?

Response: The threshold d_{th} depends on several factors including the distance of an object from the camera. In other words, the farther the object, lesser should be the value of d_{th}, especially for scenes where the sizes of objects vary greatly from one region to another within the image lattice. The threshold can be determined automatically if either the camera calibration is known or if the object sizes are also used in the feature vector.

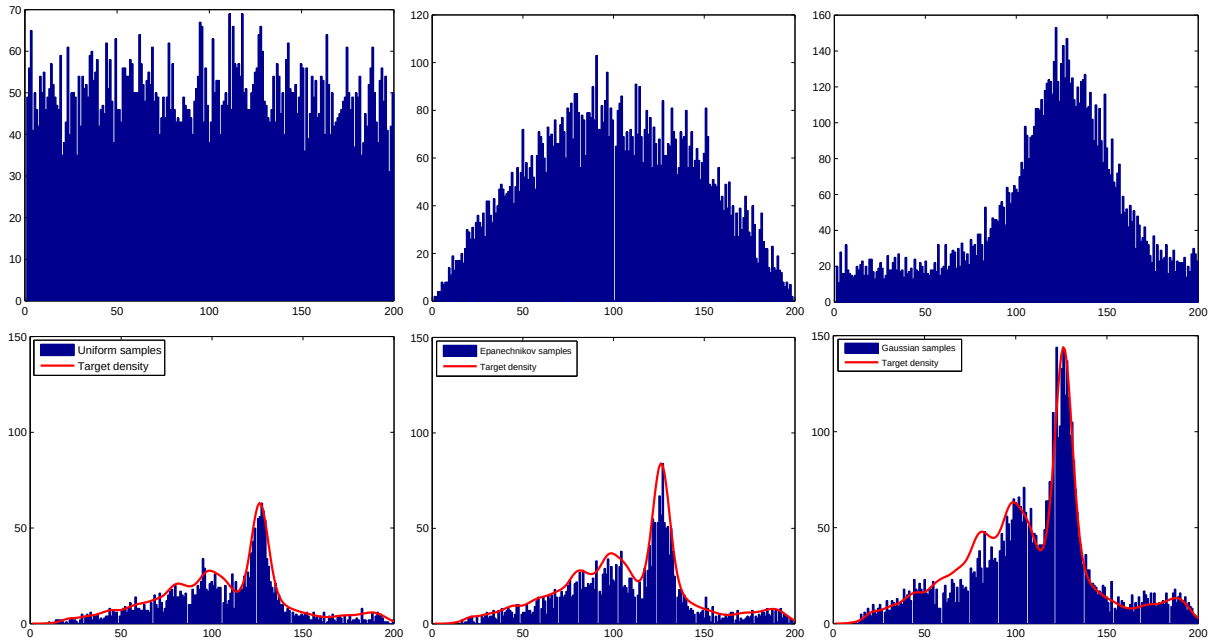
Comment 7: The computational cost of the system is very high.

Response: The system has high computational cost for training, but it can be performed off-line. The run-time performance of the system can be significantly improved by using software optimization and using well known approximations of KDE, such as histograms, Fast Gauss Transform [25, 17], and mixture models. In our experiments, the use of histograms (in an un-optimized MATLAB code) improved the run-time performance of foreground estimation from 0.6 fps to 2.75 fps (See Table 3.4).

Comment 8: How to apply incremental learning to update the model adaptively.

Response: Traditionally, the kernel density estimation is made adaptive to the changes in the data by keeping a fixed size queue of n data points. Once k new data points are observed, the oldest k points in the queue are replaced with the new data points and the model is updated. The model updates can be done on a low priority thread, although online updating will not have any significant effect on performance. Note that the proposed system is based on static scenes and the availability of large training data, and hence does not have to be updated very frequently.

Comment 9: Why use the Gaussian distribution as proposal density for Metropolis-Hastings sampling?



(a) Uniform (b) Epanechnikov (c) Gaussian

Figure 3.16: A comparison of different proposal densities. The first row shows histograms of the 10,000 samples from the respective proposal distributions. The second row shows histograms of samples that were accepted for the target density plotted in red. All histograms contain 200 bins.

Response: The reason we choose Gaussian distribution as the proposal density is the observation that often the true conditional density of object transition given the current state resembles the Gaussian distribution. In addition to Gaussian distribution, we experimented with Uniform and Epanechnikov densities centered on the current location. Assume that the initial location X of a transition can be written as $X = (x_1, x_2)$. The results of sampling from the target density $\int_Y \int_\tau \int_{x_1} p(X, Y, \tau) dx_1 d\tau dY$ are shown in Fig. 3.16 and Fig. 3.5. The 1 dimensional marginal density of the proposed distribution is chosen for ease of visualization. As can be seen in Fig. 3.16, the $quality$ of samples is not affected by the choice of proposal density, however, the performance in terms of speed is significantly reduced due to a bad choice as more samples are rejected.

Table 3.5: Acceptance Rates of Candidate Samples, the time taken to generate a 10 steps trajectory, and the number of times the Metropolis Hastings chain fails to move.

Proposal Density	**Accepted**	**Rejected**	**Accept/Reject Ratio**	**Time (secs)**	**# Chain Fails**
Uniform	1991	8009	0.25	79	2
Epanechnikov	2535	7465	0.34	52	2
Gaussian	5356	4644	1.15	16	0

3.5 Summary

To summarize, this chapter introduces a novel method of modeling traffic activity in scenes that is simple and intuitive. It models the spatio-temporal attributes of traffic without losing any information available from tracking. We have shown that the proposed model and framework are valid, and have the capability of generating likely tracks, and improving foreground detection. It can detect anomalous motion, and persistently track objects through short term occlusions. Moreover, Kernel Density Estimation is chosen as the preferred method for model representation and learning, which provides a rich model which is reliable and accurately reflects the transition likelihoods. We show results of experiments on a scene where motion patterns are not well-defined, unlike in scenes where roads or walkways are usually followed.

The representation and estimation framework proposed in this chapter, however, has a few limitations. This algorithm proposes to use a *single* global statistical distribution to represent all kinds of motion observed in a particular scene. It therefore, does not find a separation between multiple semantically distinct motion patterns in the scene. Instead, the learned model is a joint distribution over *all* possible patterns followed by objects. To overcome this limitation, in the next

chapter, we propose a superior method for the discovery and statistical representation of motion patterns in a scene.

CHAPTER 4: MOTION PATTERNS AS DISTINCT MIXTURE DISTRIBUTIONS

This chapter presents a new method for inference of motion patterns, which overcomes the drawbacks and limitations of the techniques mentioned in the earlier chapters, in two important ways. First, this model is applicable to scenes of dense crowded motion where tracking may not be feasible, and second, it distinguishes between motion patterns that are distinct at a semantic level of abstraction. Figure 4.1 shows a few examples of static camera, surveillance video scenes, where some of the desired patterns of motion are shown as colored arrows. The goal is the automatic discovery and representation of all such patterns directly from the video. The proposed method employs simple yet powerful statistical modeling and learning methodologies. The process flow of the framework employed in this chapter, is outlined in Fig. 4.2.

Our proposed method begins with introduction of a probabilistic model of motion patterns based on optical flow, such that although noise and clutter may still not be clearly separable, the statistical properties of our model of dense flow will help classify it into patterns when observed for extended periods of time. We first compute dense optical flow using the method of [41]. We can then define, $\mathbf{X} = (x, y, \rho, \theta)$, as a random variable representing optical flow at a particular pixel, where (x, y) is the location of the pixel on the image lattice, and (ρ, θ) is the magnitude and direction of the optical flow vector (u, v) such that, $\rho = \sqrt{u^2 + v^2}$ and $\theta = tan^{-1}\left(\frac{v}{u}\right)$. A number of possible values for the variable $\mathbf{X}$ are shown in Fig. 4.3, as observed in a small video clip. The

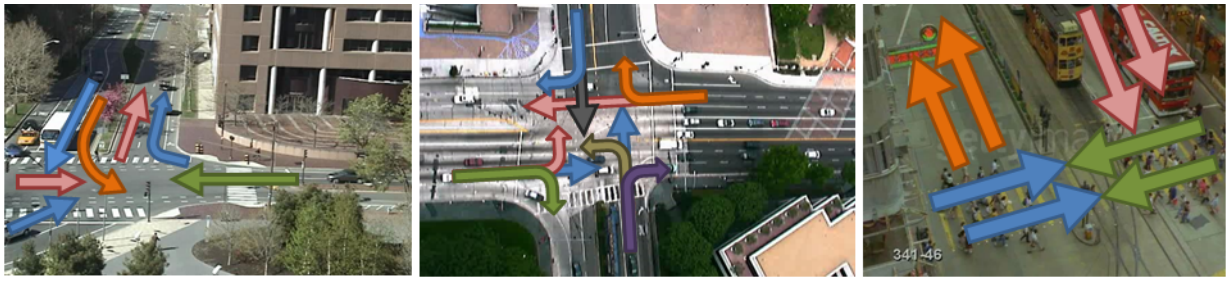


Figure 4.1: Examples of surveillance sequences from static cameras used in our experiments. Left to right: MIT, NGSIM, and Hong Kong sequences. The arrows show some desirable patterns of motion.

goal of learning the model is to evaluate the probability $p_i(\mathbf{X} = \mathrm{x})$, i.e., estimate the likelihood of a pixel and optical flow pair x, of belonging to the motion pattern i.

Since the motion patterns need to be learned over extended periods of time, and the optical flow is dense, only a parametric approximation of the actual distribution is tractable. We employ a mixture of Gaussian model to this end. Furthermore, since estimation of mixture parameters is computationally intensive, we propose a hierarchical learning technique that allows us to bypass parameter optimization without compromising the quality of distribution. The hierarchical learning is performed in a bottom up fashion, starting from pixel level optical flow vectors, to a large number of distributions, which then represent mixtures of distributions. The mixtures then form the eventual representation of motion patterns.

It can be noticed that a single pattern of motion, as well as an 'instance' of that pattern (e.g., an object following the pattern), can be visualized as a worm in the 5-space, where time (frame number) is the fifth dimension, just like the trajectory of an object is a curve in the 3-space, (x, y, t).

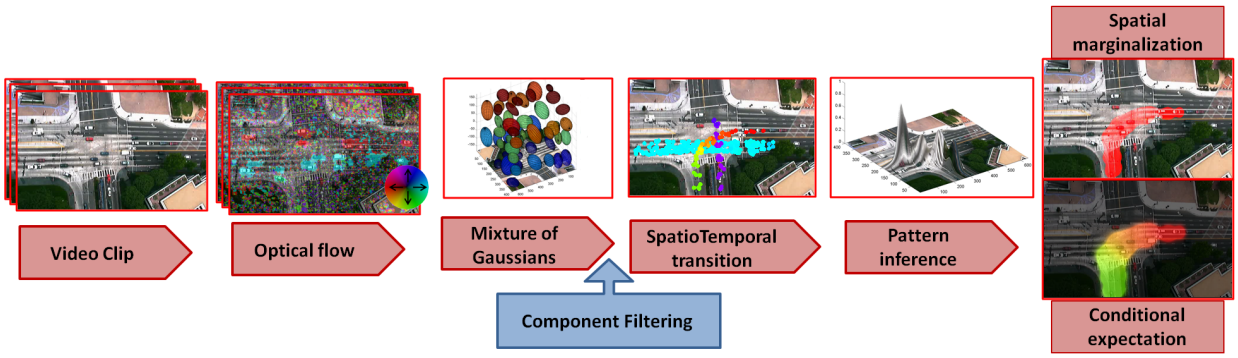


Figure 4.2: Process flow of our approach: Grouping of frames into video clips, optical flow computation, Gaussian mixture model learning by K-means, filtering of noisy Gaussian components, inter-component spatiotemporal transition computation for instance learning, pattern inference using KL-divergence, pattern representation as spatial marginal density, and computation of conditional expected value of optical flow given pixel location.

Obviously the description of a pattern itself is independent of the notion of time (although the *sequence* of inter-pattern occurrences in general would be governed by time, i.e., non-random). However, multiple instances of the same pattern can be significantly spaced out in time, which is very likely to happen since most patterns by definition are recurrent. Therefore, in the learning phase, time is taken into account when learning instances of patterns, but ignored when learning the actual patterns. Since the smallest unit of time, a frame of optical flow, has too few observations (values for the random variable $\mathbf{X}$ to take) to contribute towards a meaningful group of mixture components, we therefore quantize time into disjoint segments of k frames each, referred to as 'video clips' where, in our experiments, k is typically set to the number of frames in 1 second of video.

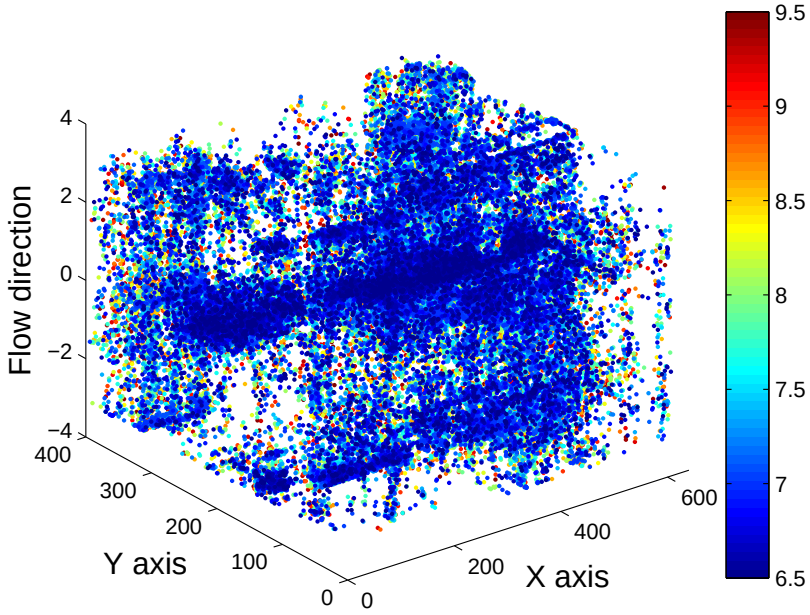


Figure 4.3: 4d low level features (x, y, ρ, θ), where ρ is represented by colors as per the legend. Note that $\theta \in [-\pi, \pi]$.

4.1 Mixture Model

We begin by marginalizing out time in each video clip, and filtering out optical flow observations with unusually small or large magnitudes. The remaining observations can now be considered as data points in the 4d space (x, y, ρ, θ), an example of which, is plotted in Fig. 4.3. Our goal now is to learn this 4d distribution modeled as a Mixture of Gaussian components. The theoretically straightforward way for this is random or approximate initialization of a chosen number of components followed by a parameter optimization algorithm like EM. However, as we will later show, a careful initialization suffices for the purpose of our algorithm and the optimization process can be skipped (significantly improving computation efficiency as the number of data points in each video

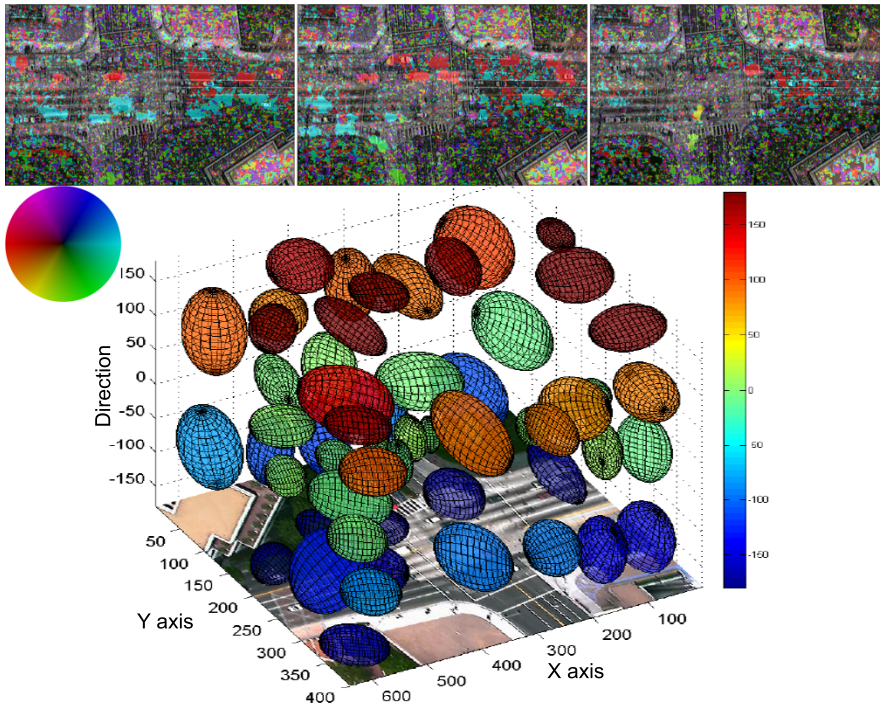


Figure 4.4: Top: 3 examples of optical flow frames in a video clip, where colors show flow direction as per the circle on the left, and magnitude is represented by image brightness. Bottom: A few Gaussian components representing the marginalized distribution $\int p(\mathbf{X})d\rho$ for the video clip, shown as error ellipses with surfaces at 1σ. The colors of ellipses also indicate the mean direction of optical flow, i.e. μ_θ as per the color bar shown on the right.

clip are $\sim 10^5$). We therefore estimate the parameters of the distribution by treating the observations as data points that need to be clustered into N clusters, where the centroid and error ellipse of each cluster serve as the parameters, mean μ, and standard deviation σ, in each dimension for each Gaussian component, which along with the orientation of the error ellipse determines the full covariance matrix, Σ of the component.

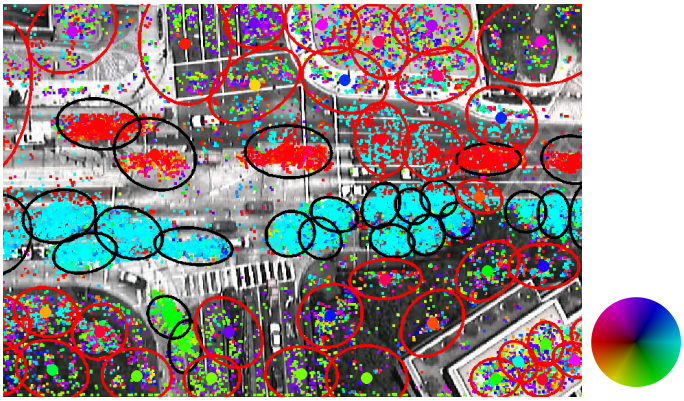


Figure 4.5: Component Filtering: Two dimensional marginalized Gaussian mixture components, $\int\int p(X)d\rho d\theta$, represented by component error ellipses at 1.5σ. Data points are shown as colored dots, where colors correspond to optical flow directions as per the circle shown as legend. Components detected as noise or clutter are shown as red ellipses, while components with low directional variance are depicted as black ellipses. Mean direction is shown as solid dot at each component centroid. Notice the variation in colors in noisy components.

We employ K-means clustering to estimate the parameters of these N components or clusters. Initialization is performed using multiple iterations of K-means on randomly sampled subsets of the data points, and the distance metric is Euclidean, modified to cater for boundaries of direction interval $[-\pi, \pi]$. We can then write, $\mathbf{X} \sim \sum_{i=1}^{N} \ w_i \ \mathcal{N}(\mathbf{X}|\mu_i, \Sigma_i)$, where, w_i, the weight of each component is the percentage of data points in the i^{th} cluster, and $\mathcal{N}$ represents Gaussian distribution.

As mentioned before one of the main goals of the proposed method of motion pattern learning is to be able to avoid object detection and tracking, which can be problematic especially in cases of crowd scenarios and dense traffic flow and clutter. We observe that by filtering non-moving and erroneous optical flow observations, the data points in the 4d space (x, y, ρ, θ) are segmented into

spatially disjoint clusters generally corresponding to moving objects. Although raw optical flow is noisy, observations of true motion occur in dense, low variance clusters of consistent optical flow, which then become components of the mixture model. Since our objective is to find many dense clusters with small variance especially in the direction of optical flow, the number of components need only be large enough. Furthermore, optimization of parameter fitting is not required. An example of K-means clustering as component initialization can be seen in Fig. 4.4.

4.1.1 Gaussian Component Quality Assessment

Despite filtering of optical flow with nominal or unusually large magnitude ρ, a non-negligible number of remaining observations, and therefore Gaussian components will comprise optical flow from noise and background areas. Such components have x and y standard deviations that are similar to other components, but they have high variance in ρ and θ dimensions as shown in Fig. 4.5. The components with unusually high variances especially in flow direction, θ, can therefore be discarded. It can be seen in Fig. 4.5 that such components, shown as red ellipses, have observations with highly varying colors that depict flow direction. While discarding of these components can be construed as loss of information, it should be noted that given the small value of k, the GMM of a single time clip represents instances of atomic events and even if a few correct components are discarded, these atomic examples are repeated periodically multiple times in a video sequence, and therefore do not have an adverse effect on learning of actual patterns. Fig. 4.5 shows an example clip where such components have been filtered out. The filtering is based on directional variance

of a component being larger than other components (relative), or than a fixed threshold (absolute), e.g., mean variance.

Although learning the proposed model using a non-parametric distribution is intractable as explained earlier, it should be noticed that components in the Gaussian mixture behave as representative sampled data points of the underlying data, and adequately retain the multimodal structure of the true distribution.

4.1.2 Inter-Component Spatio-Temporal Transition

We now have the representation of a single video clip in the form of N Gaussian mixture components, our goal is to do the same for motion patterns using multiple video clips. We observe that since motion pattern instances should appear as continuous worms in the spatio-temporal volume of optical flow, and because each worm has been divided into components by mixture learning (as shown in Fig. 4.6), the clusters belonging to the same pattern form a representation of a pattern 'instance'. We therefore, propose a method of linking components in temporally adjacent (or proximal) video clips to form multiple mixtures corresponding to instances of motion patterns.

Since the GMM components capture the local shape of a pattern instance, by linking components through time, the full structure of the pattern can be recovered. We treat components from all video clips as nodes in an undirected graph and place an edge between two nodes according to the following conditions: (1) the components belong to proximal video clips, and (2) one component is 'reachable' from the other. The term 'reachability' can be defined in terms of transition probabilities of a random walk, where these transitions reflect an underlying motion model. A graphical

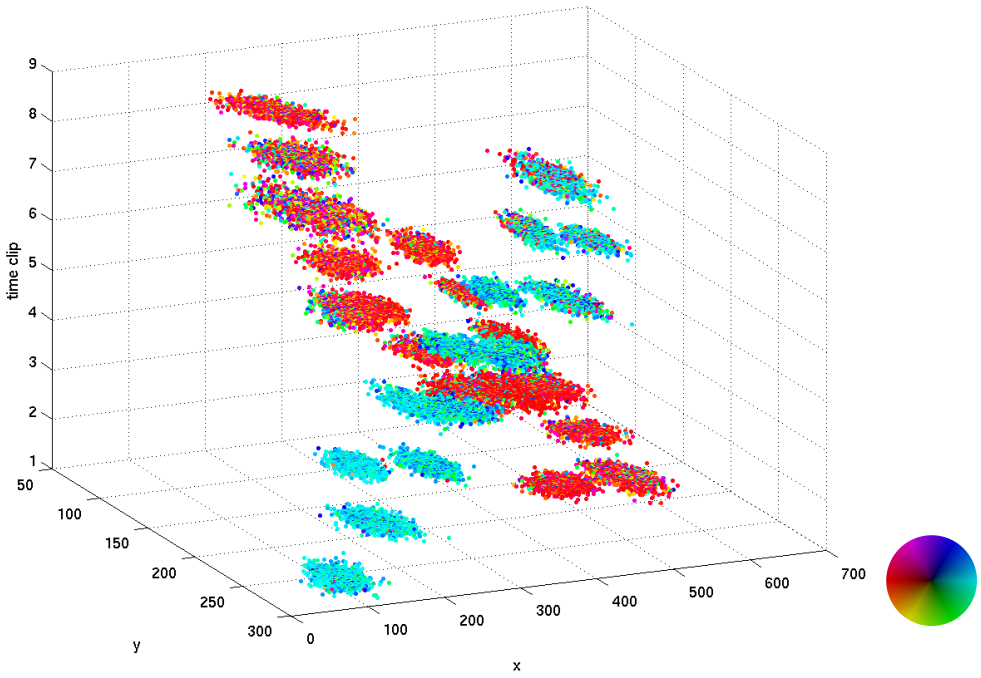


Figure 4.6: Components drawn as collection of points appear as worms in the spatiotemporal volume (x, y, t), where t is quantized into video clips. Colors represent the optical flow direction as per the legend, while magnitude is not shown. One instance each is shown for two patterns; eastward (cyan), and westward (red) traffic.

illustration of this process can be seen in Fig. 4.7. Given components i and j from video clips t and $t + \xi$ respectively, where ξ is a small positive integer, the probability of transition of a feature from component i to j can be computed as the probability that a *predicted* data point $\hat{\mathrm{x}}_i$ belongs to component j, and can be written as,

$$p_j(\mathbf{X} = \hat{\mathrm{x}}_i) = \frac{1}{4\pi^2\sqrt{|\Sigma_j|}} \quad exp\left[-\frac{1}{2}\left(\hat{\mathrm{x}}_i - \mu_j\right)^\top \Sigma_j^{-1}\left(\hat{\mathrm{x}}_i - \mu_j\right)\right], \tag{4.1}$$

and $\hat{\mathrm{x}}_i = (\hat{x_i}, \hat{y_i}, \rho_i, \theta_i)$ is the transition prediction for component i, where, $\hat{x_i} = x_i + \xi k \rho_i \cos\theta_i$, and $\hat{y_i} = y_i + \xi k \rho_i \sin\theta_i$.

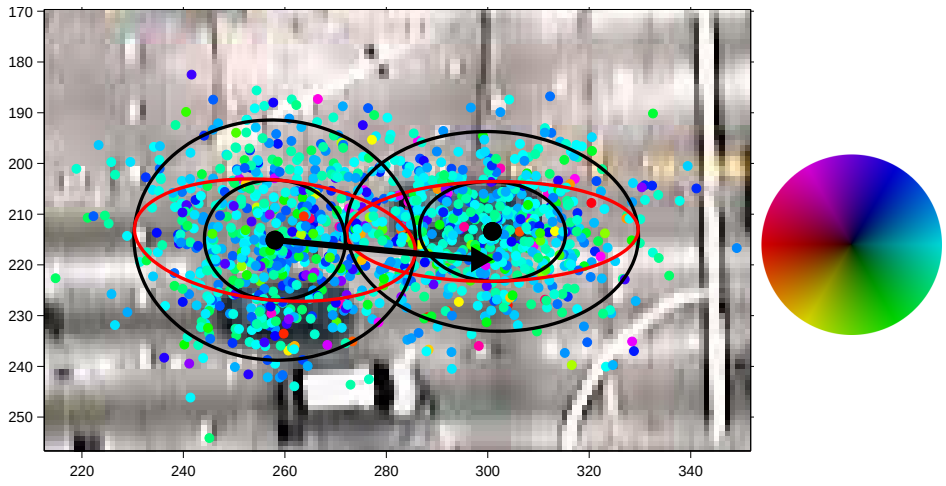


Figure 4.7: Illustration of 'reachability' from left component to right component. Two sets of black concentric ellipses show 1σ and 2σ error ellipses for each component, with mean location shown as large black dots in the center. Cyan colored dots show data points in each component, and represent eastward motion. The red ellipses depict hypothetical transformed error ellipses as described in text. The black line starting from left component's centroid shows the prediction $(\hat{x_i}, \hat{y_i})$, represented by the arrow.

The transition prediction is set up such that the underlying motion model follows the constant velocity model, in that smoothness of motion is exhibited in terms of both magnitude and direction of velocity, whereas the factor ξk, the number of frames between the video clips ensures that the prediction is meaningful for evaluation using component j. The weight of the edge connecting nodes i and j is computed using both forward (Eq. 5.1) and backward likelihoods, where the backward likelihood is computed using the opposite direction $(\theta_j + \pi)$. Since these two events are independent, the edge weight is defined as their product.

The edge weights are subsequently thresholded by using Mahalanobis distance between the destination node's distribution and the source node's prediction. However, for the purpose of

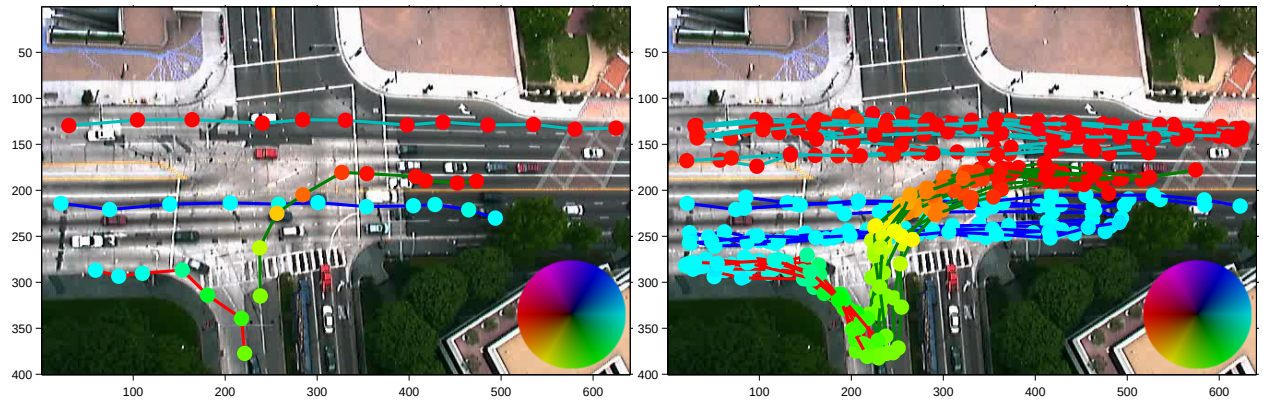


Figure 4.8: Examples of pattern instances. Left: one instance each of four patterns, westward, eastward, left and right turns. Colored circle locations represent Gaussian component's X and Y means, while their colors represent mean optical flow direction as per the circle in the bottom right. Lines connecting the components represent membership in an instance. Right: Multiple instances for each pattern shown in the left image.

computing the edge weight, the destination node's covariance, Σ_j is transformed so that while the error ellipse's orientation in 4d remains unchanged, the spatial variance of the component (variance in x and y dimensions), reflects more variation in the direction parallel to its mean optical flow direction, compared to the direction orthogonal to mean flow. In other words, component j is reachable from component i if its prediction falls within a transformed error ellipse of cluster j, where the transformed ellipse reflects standard deviation 2σ in direction of mean flow and 1σ in the perpendicular direction. This reachability test is illustrated in Fig. 4.7, where the transformed covariance is depicted by red ellipses. When all edges have been placed between appropriate nodes, the connected components of the graph represent distinct motion pattern instances that occurred throughout the video. Some examples of such pattern instances are shown in Fig. 4.8.

4.1.3 Motion Patterns Inference

We now have numerous instances of each motion pattern and the final goal is to merge all instances of each distinct pattern into a single mixture of Gaussian components, which can be achieved by a comparison between all pairs of instance probability distributions (represented by GMMs). We use the Kullback-Leibler (KL) divergence to compare these distributions. Note that the previous step of computing pattern instance distributions incorporated a time constraint, such that each instance is bounded by, and is continuous across a temporal segment. However, it is now reasonable to compare instances from different time periods in the video. In other words, if we consider two mixtures of temporally linked Gaussian components, the KL divergence between them will be low if they represent the same action (eg. a car moving eastward) because of similarity in location (x, y), as well as motion (ρ, θ).

The KL divergence between two Gaussian mixtures is only computable through approximation, for which we use Monte Carlo sampling of instance distributions. We employ Eq. 4.2 where p_f and p_g are probability density functions representing two pattern instances f and g, that we wish to compare. We draw n samples $\{\mathrm{x}_i\}_{i=1}^{n}$ from the first distribution p_f. If the samples also have a high probability of occurring in distribution p_g, then the divergence is low, and we consider the instance g to be a subset of the same pattern as f. We can therefore write for a two instance divergence,

$$D(f \parallel g) \approx \frac{1}{n} \sum_{i=1}^{n} \log \frac{p_f(\mathrm{x}_i)}{p_g(\mathrm{x}_i)}. \tag{4.2}$$

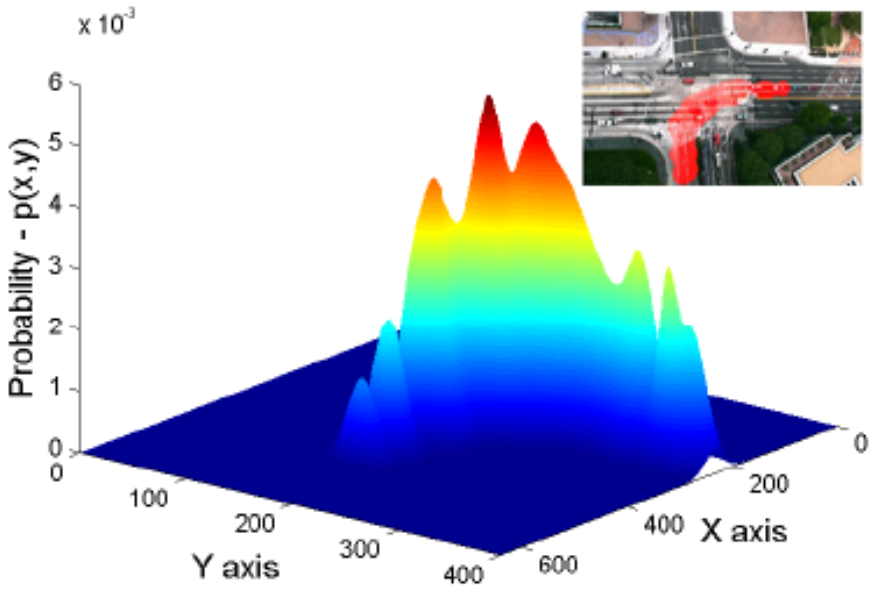


Figure 4.9: Probability density for a single motion pattern, left turn. Assuming this is the m^{th} pattern in the set of all patterns, the figure shows the marginal density surface, $\int\int p_m(x, y, \rho, \theta)d\rho d\theta$. Inset: The same density thresholded and overlayed on the scene image, shows the extent of the motion pattern.

The divergence between all possible instance pairs is computed and assembled into a non-symmetric square matrix which can be considered as edges of a fully connected graph, where the edges can be binarized by choosing different thresholds. A graph connected component analysis of the matrix then gives multiple final motion patterns. The automatic or manual choice of various binarization levels represent a multiscale analysis of the scene, such that a high level corresponds to few, large patterns, and vice versa. The components of individual instances are merged into the connected component (the motion pattern) that they belong to. Patterns with a single or very few instances can be discarded.

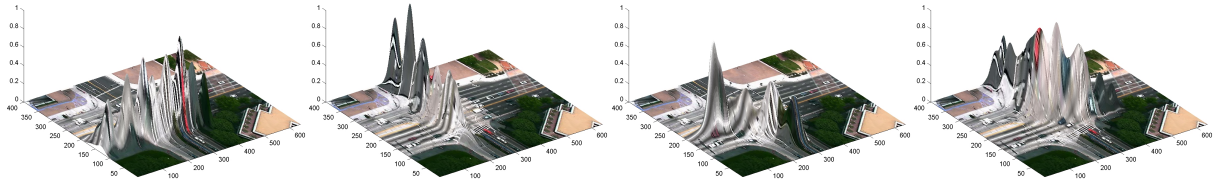


Figure 4.10: Probability surfaces of four motion patterns, textured by an image of the scene showing, (L-R) eastbound motion, southbound motion, right turning traffic, and left turn.

The result is a number of self-similar groups of motion instances that have occurred numerous times throughout the video. Each such group then represents an individual Gaussian mixture. As shown in Fig. 4.9, a marginalized probability map of each of these new GMMs indicates the probability of a pixel belonging to that motion pattern. By thresholding this probability map, it is possible to give a binary label to each pixel in the image, indicating membership in the pattern, as shown in Fig. 4.9(inset).

4.1.4 Conditional Expectation of Optical Flow

The proposed representation of a motion pattern is richer than giving a binary label to each pixel because it also contains per pixel motion information. The majority of applications in scene understanding and surveillance require that probability is bound to pixels, i.e., the statistical distribution is computable conditional on spatial location. The proposed probability distribution of a motion pattern given a pixel is easily computable by marginalization over the optical flow dimensions, ρ and θ, that is $\int \int p(x, y, \rho, \theta) d\rho d\theta$ as can be seen in figures 4.9 and 4.10. However, we are not only

interested in the probability as shown in Fig. 4.9, but also in the distribution of different optical flow values given a pixel.

Furthermore, the most useful representation of a pattern is not the probability density function of optical flow given a pixel but the value of optical flow that is most expected for that pattern at a given location. The estimation of such a value involves the computation of conditional expected value of optical flow magnitude and direction given a pixel location. Assuming conditional independence given a pixel, we compute the conditional expectations for magnitude ρ and direction θ separately. The expectation for direction given a pixel location, is a two dimensional function with (possibly real valued) domain equal to the size of the image, and range in the interval $[-\pi, \pi]$, and for a motion pattern comprised of M Gaussian mixture components, indexed by $i \in \{1, \ldots, M\}$, this function can be computed as follows:

$$
\begin{aligned}
\mathbb{E}[\Theta|\mathbf{X}, \mathbf{Y}] &= \int_{-\infty}^{\infty} \theta p(\theta|x, y) d\theta \\
&= \int_{-\infty}^{\infty} \frac{\theta p(x, y, \theta)}{p(x, y)} d\theta \\
&= \frac{1}{p(x, y)} \int_{-\infty}^{\infty} \theta \left[\sum_{i=1}^{M} w_i p_i(x, y, \theta) \right] d\theta \\
&= \sum_{i=1}^{M} w_i \frac{\int_{-\infty}^{\infty} \theta p_i(x, y, \theta) d\theta}{p_i(x, y)} \\
&= \sum_{i=1}^{M} w_i \mathbb{E}_i[\Theta|\mathbf{X}, \mathbf{Y}] \qquad (4.3) \\
&= \tan^{-1} \left\{ \frac{\sum_{i=1}^{M} w_i \sin\left(\mathbb{E}_i[\Theta|\mathbf{X}, \mathbf{Y}]\right)}{\sum_{i=1}^{M} w_i \cos\left(\mathbb{E}_i[\Theta|\mathbf{X}, \mathbf{Y}]\right)} \right\}, \qquad (4.4)
\end{aligned}
$$

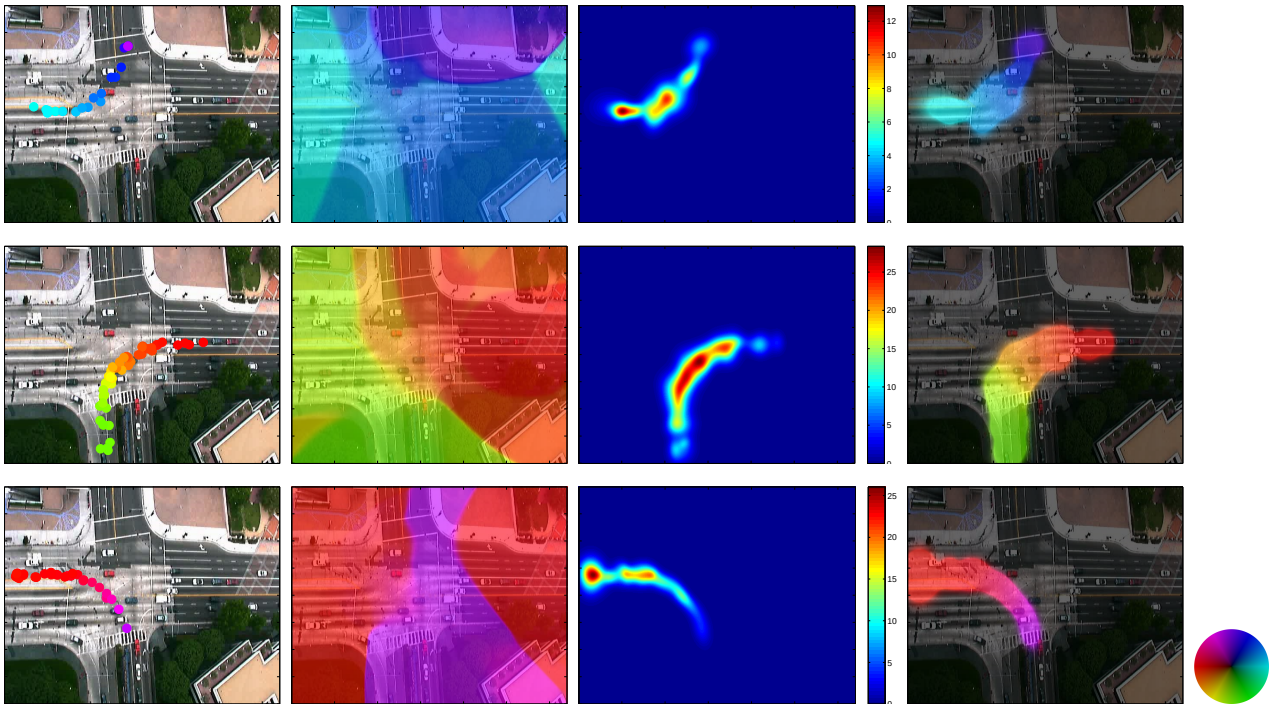


Figure 4.11: Three example patterns in 3 rows, with each row showing: (L-R) components in the pattern, expected orientation per pixel, expected magnitude per pixel, and combined expected optical flow per pixel, with magnitude indicated by brightness.

where $\mathbb{E}_i[\Theta|\mathbf{X}, \mathbf{Y}]$ is the expected direction at a pixel indicated by (x, y) as per the Gaussian distribution of the i^{th} component of the pattern. Eq. 4.3 shows that the conditional expectation of a weighted mixture distribution is the weighted mean of individual conditional expectations, whereas Eq. 4.4 is specific to direction expectation, so as to avoid problems at phase boundaries.

Fixing x and y by choosing a particular pixel, this distribution is a one dimensional density represented by values along a vertical line in the three dimensional space (x, y, θ). It is known that for any multivariate Gaussian distribution of dimension k, any sub-vector of dimension $l < k$, itself represents a multivariate Gaussian distribution, and it can be shown that the distribution

of any dimension of the multivariate Gaussian density, given all the other dimensions is a one dimensional Gaussian probability density. Therefore, the pdf of $p_i(\theta|x,y)$ is Normally distributed and its mean is given by,

$$\mathbb{E}_i[\Theta|\mathbf{X},\mathbf{Y}] = \mu_\theta + \begin{bmatrix} \sigma_{\theta X} \sigma_{\theta Y} \end{bmatrix} \bullet \begin{bmatrix} \sigma_{XX} \sigma_{XY} \\ \sigma_{YX} \sigma_{YY} \end{bmatrix}^{-1} \bullet \begin{bmatrix} x - \mu_X \\ y - \mu_Y \end{bmatrix}. \tag{4.5}$$

The conditional expectation of magnitude ρ can be computed in a similar fashion using Eq. 4.3. The conditional expected optical flow provides the dominant magnitude and direction of the motion pattern at the pixel level, instead of quantized or interpolated estimates of direction at super-pixel levels (quantized image space) as in [78, 84]. Since GMM pdf is a continuous function, the proposed representation can be used to computed probability or expectation with sub-pixel accuracy. Furthermore, given the range of expected optical flow magnitude over the entire image plane, the need for thresholding of probability is avoided as can be seen in Fig. 4.11. It is worth mentioning here that the boundary or segmentation of the patterns shown in Fig. 4.11 are not defined by any thresholding of the probability, but the rather sharp contours are actually a manifestation of the steep drop in expected flow magnitude values, reducing the brightness of the flow image.

4.2 Experimental Results

We tested the proposed model and learning method on three data sets. For the first, NGSIM data set [54], we used a 15 minute video of a segment of road that includes an intersection as can be seen in Fig. 4.1. This is a challenging video for scene understanding because, for the region in the

center of the intersection, there is no single dominant pattern. Fig. 4.14 shows the expected optical flow of some of the patterns discovered, by overlaying them on an image of the scene field of view. Each pixel in the image shows the expected direction with a different color, and the expected flow magnitude with varying brightness. Figure 4.14 first row depicts examples of, (a) southbound, (b) east to north left turn, (c) south to east left turn, and (d) eastbound motion patterns. The second row shows, (e) west to south left turn, (f) eastbound, (g) east to south right turn, and (f) northbound traffic patterns.

The second data set used in our experiments is the MIT data set [78]. Some of the smaller patterns discovered in this dataset are shown in figure 4.12. These patterns correspond to atomic behaviors of vehicular and pedestrian traffic, e.g., notice the patterns corresponding to pedestrians entering and exiting a building in the top row (2^{nd} and 3^{rd} patterns). The larger patterns discovered in this sequence are shown in figure 4.15, where the patterns correspond to vehicular traffic such as turn, and behaviors like pedestrians on sidewalks and crosswalks. The third data set we tested our algorithm on is also very challenging, depicting dense crowds of pedestrians walking in opposite directions through the same spatial regions. As can be seen in figures 4.13 and 4.16, the proposed method is able to detect multiple semantically meaningful patterns of motion in a completely unsupervised manner. This sequence is much more challenging than those used in [78, 84, 46], because of crowded traffic, and multiple co-occurring patterns in the same region. This sequence is only 250 frames which is a very small number for learning patterns. The results of this sequence also employ the multiscale analysis of motion patterns by varying thresholds of the KL divergence graph edges. There are only three main patterns in the small sequence, i.e.,

pedestrians on crosswalk walking in opposite directions simultaneously, and the vehicles on the top right coming to a stop. However, a low threshold of KL divergences yields many small patterns, as shown in figure 4.13, while a higher one results in only three (figure 4.16, that represent motion at a higher semantic level. This aspect of our approach is akin to the multiscale analysis of Yang et al [84]. The proposed method is readily usable in a variety of surveillance applications, for example anomaly detection, and use as prior motion model for tracking.

4.3 Motion Patterns in Aerial Videos

A significant amount of visual surveillance data that is collected for defense as well as civilian purposes, is comprised of aerial videos. It is therefore important to ascertain commonly adapted routes, and estimate models of normalcy in such data. One significant advantage of analyzing aerial video data is that it has the potential to cover a much larger swath of territory as compared to static surveillance sensors (CCTV, etc.). One such source is the CLIF dataset [10]. CLIF stands for Columbus Large Image Format, and consists of sequences captured from a UAV around the OSU campus. The sequences are captured from an altitude of about 7000 feet (2.1 km) at a rate of 2 frames per second. The sensor consists of 6 cameras arranged in a 2×3 array. The typical resolution per frame per camera is 4008×2672, so a single mosaic frame covers on the order of square kilometers of area.

Obviously the foremost problem in motion patterns estimation in such data is the platform motion since the sensor is aboard a UAV. Therefore camera motion compensation is first performed by aligning consecutive frames using Harris corners, SIFT feature descriptors and matching, and

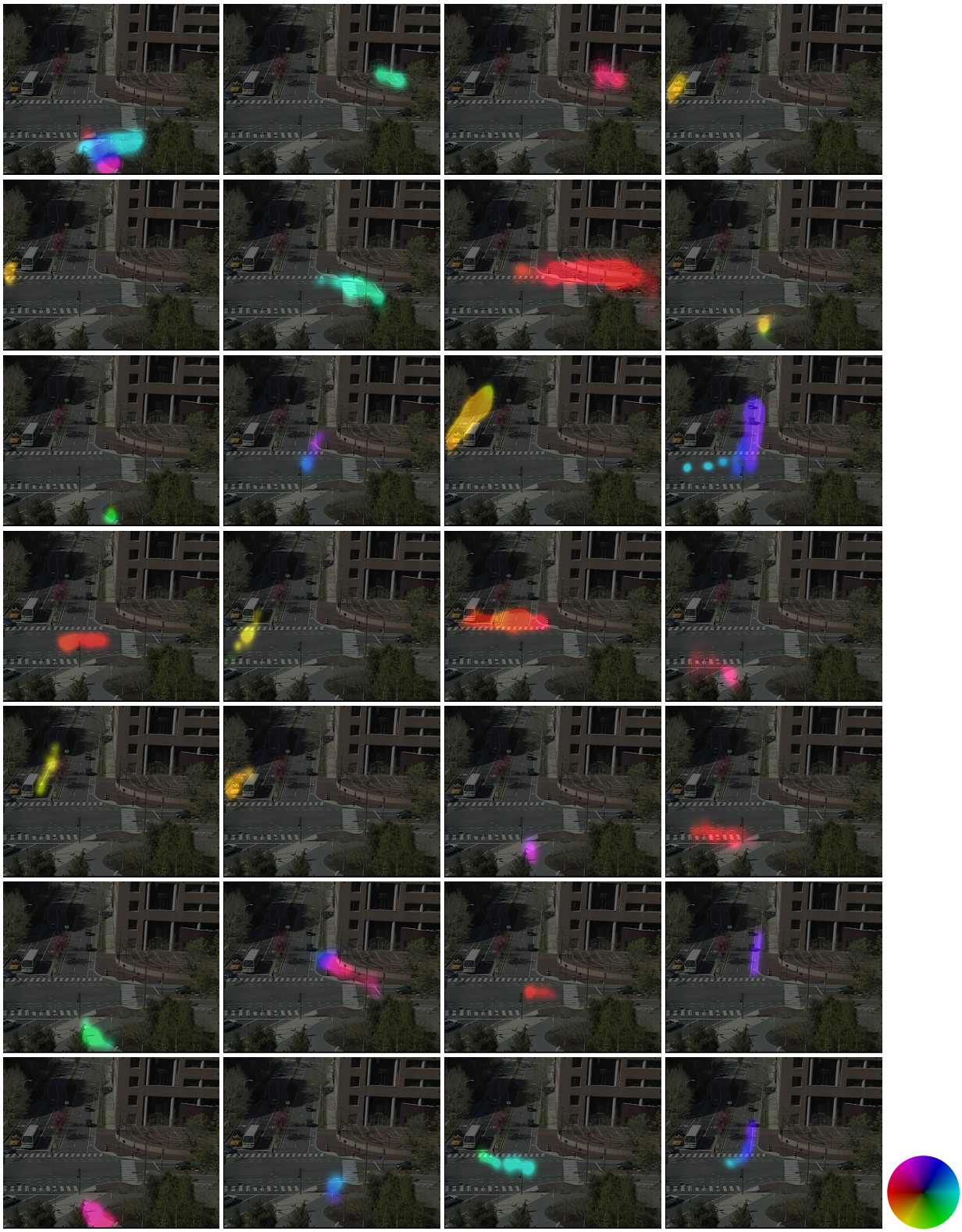


Figure 4.12: Some of the 51 smaller motion patterns detected in the MIT dataset (thresholding the KL-divergence graph weights at a low level).

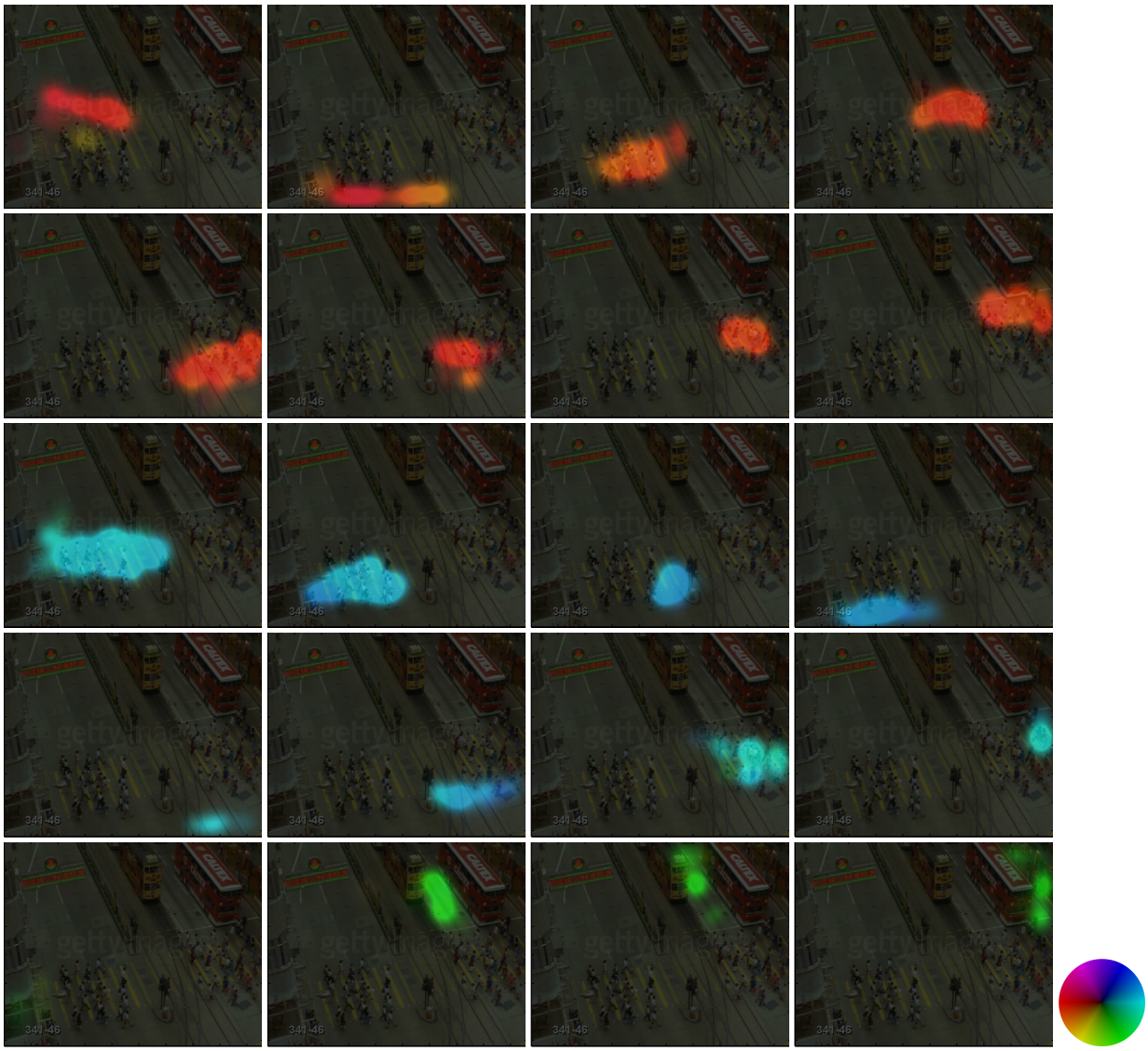


Figure 4.13: The 20 smaller motion patterns detected in the Hong Kong dataset.

RANSAC based Homography fitting. It is observed however, that the estimated transformations between frames are not perfect due to problems like parallax. Therefore, alignment reside, artifacts introduced during image warping, and low frame rate of video, cause the optical flow computed on aligned frames, to be especially noisy. A simple and viable alternative to using optical flow as

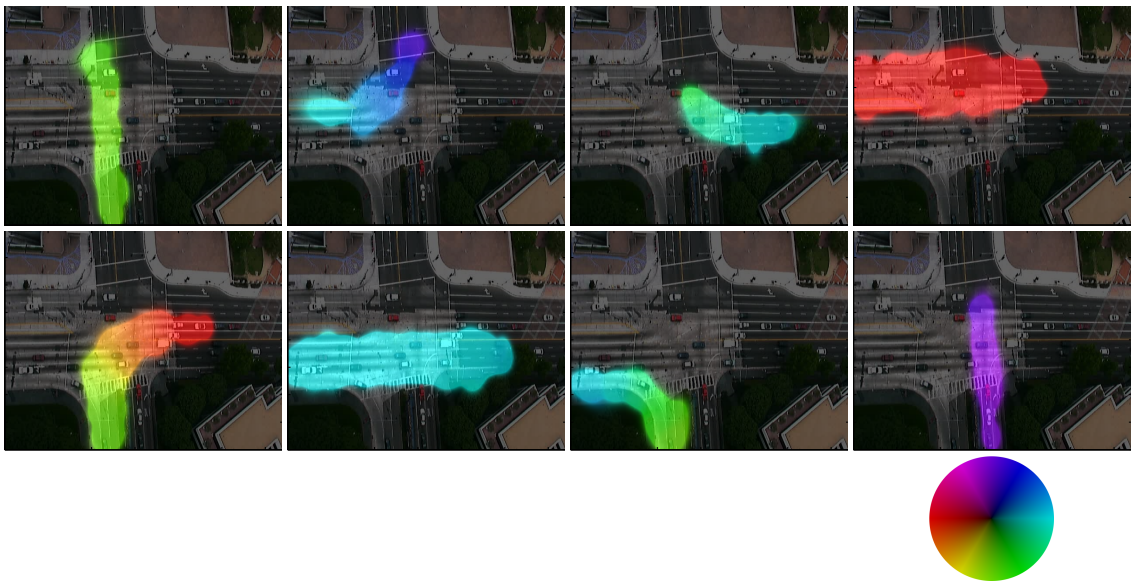


Figure 4.14: Some of the larger motion patterns detected in the NGSIM dataset (merging at high KL-divergence).

Figure 4.15: 9 larger motion patterns detected in the MIT dataset (merging at high KL-divergence).

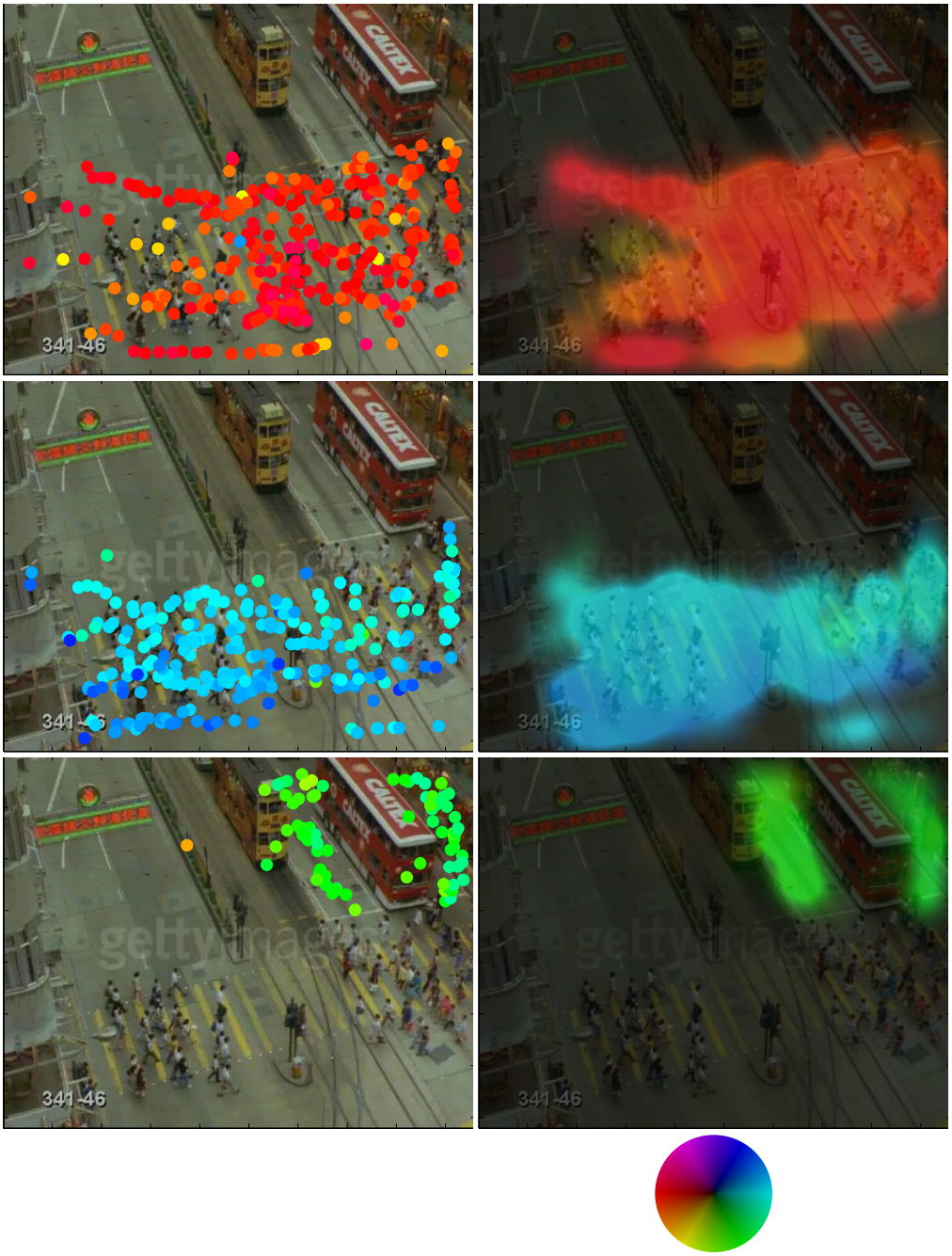


Figure 4.16: Merging of pattern instances at a high KL divergence results in 3 larger, semantically meaningful patterns.

features is to use the output of tracking data. Using a reasonably performing tracking algorithm, we first perform motion based object detection, and tracking on moving objects using the aligned video frames. The object trajectories thus obtained are essentially sequences of (x, y, t) points, and can easily be converted to sparse flow, e.g., given a two point track, $\{(x_1, y_1, t_1), (x_2, y_2, t_2)\}$, a single flow vector, (x_1, y_1, u_1, v_1) can be obtained, where, $u_1 = x_2 - x_1$, and $v_1 = y_2 - y_1$. Figure 4.17 shows tracking results as flow features on 3 aerial sequences.

Given such sparse flow vectors, we can test the motion patterns estimation algorithm without any modifications. The qualitative results of a few such experiments are shown.

4.4 Summary

To summarize, this chapter presents a novel scene understanding framework, and introduces a new representation and learning approach for motion pattern discovery in static as well as aerial sensor video sequences, where an optical flow based mixture model is representative of salient patterns of traffic, and is learned without parameter optimization. Our representation avoids any quantization and loss of information in the feature space, and we have presented results of motion patterns discovery in diverse, and challenging scenarios, making use of both optical flow and object tracks.

The experiments and results reported in this chapter have focused on scenes where the dominant foreground motion has been that of vehicles, imaged from reasonably nadir views. Some scenes (e.g., MIT dataset) do contain motion emanating from pedestrians, but that is mostly whole body, rigid motion, due to the distance from the sensor. In other words, articulated motion of humans

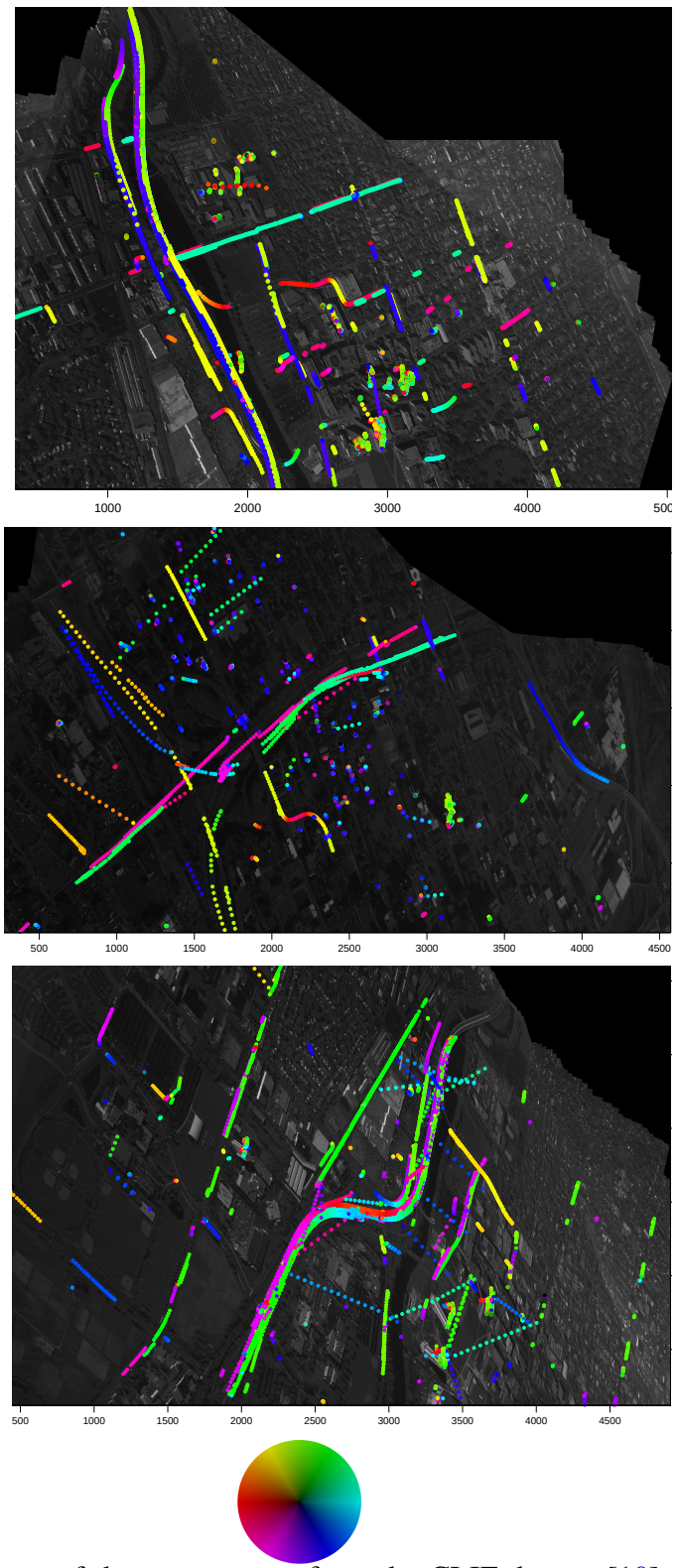


Figure 4.17: Mosaic images of three sequences from the CLIF dataset [10] showing object trajectories converted to instantaneous motion flow feature vectors, where the color of a data point indicates the flow direction, and brightness encodes magnitude.

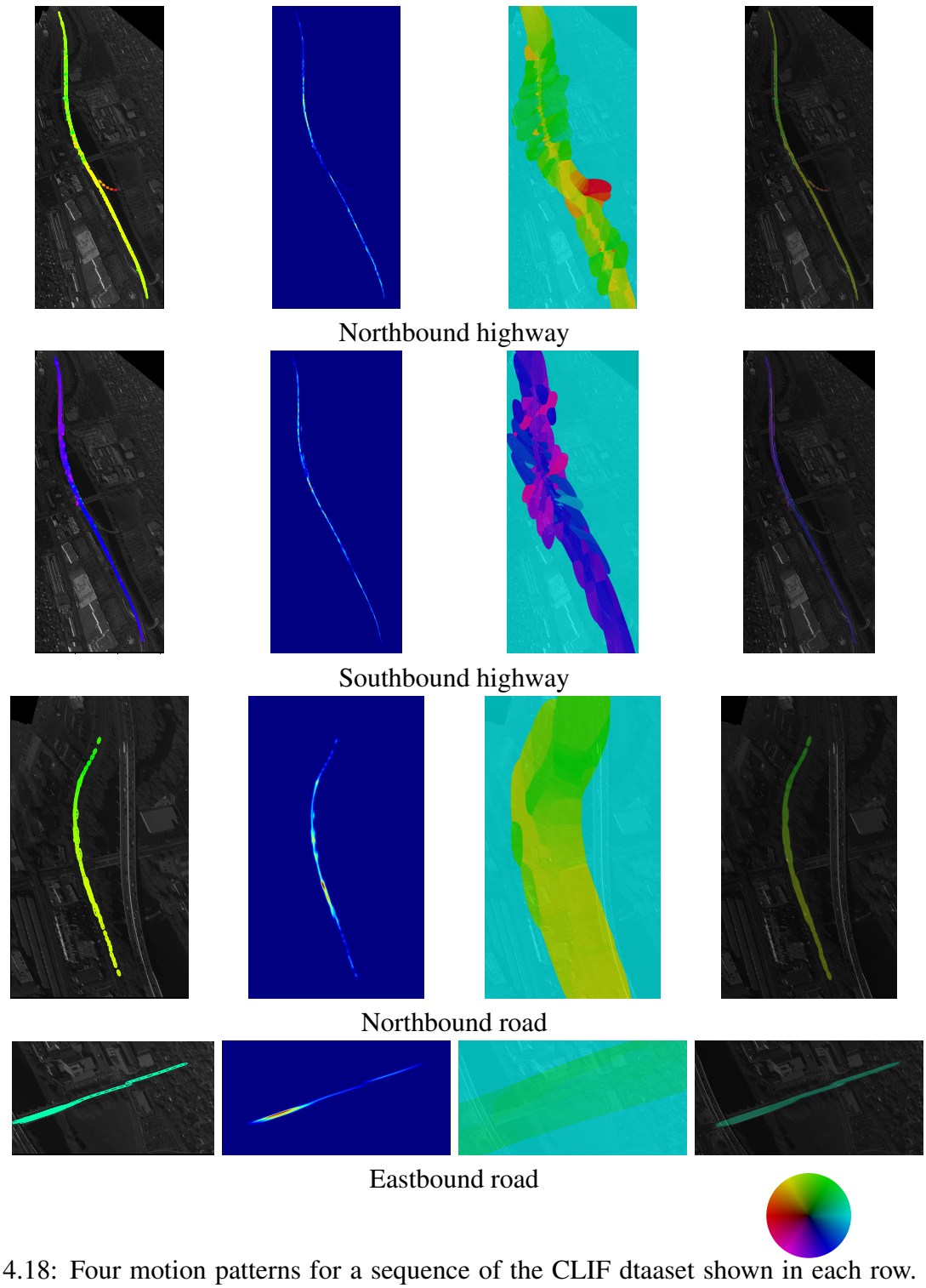
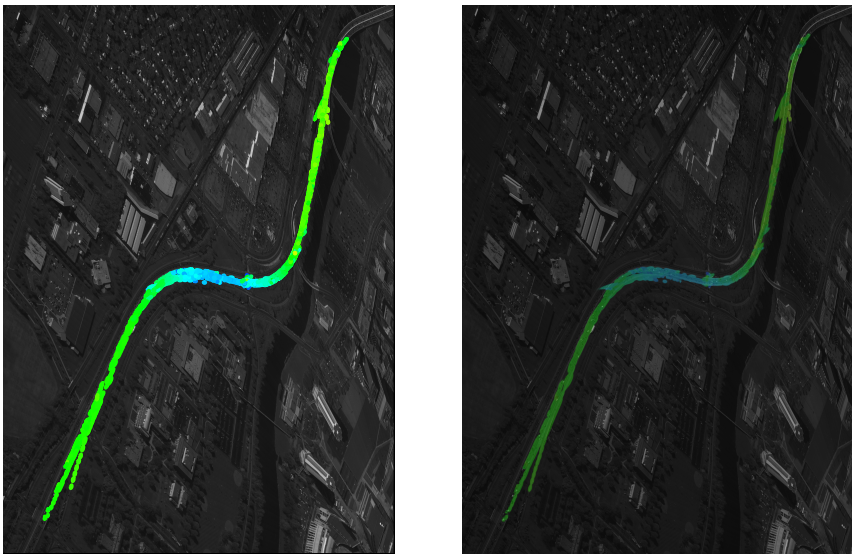
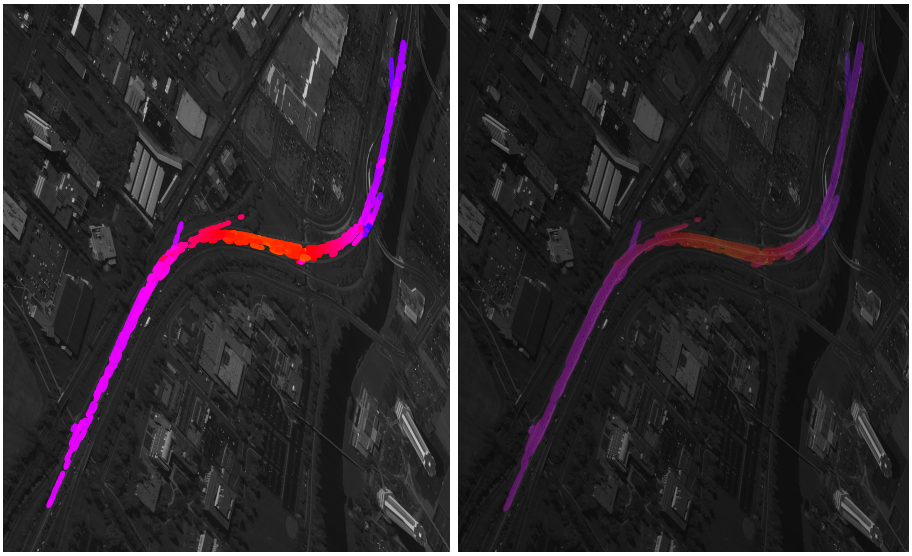


Figure 4.18: Four motion patterns for a sequence of the CLIF dtaaset shown in each row. Each column shows (L-R): Gaussian components making up the motion pattern shown as error ellipses; expected magnitude; and expected orientation of the motion patterns conditioned on location; and expected flow shown by color and brightness according to the color wheel. Notice that the mixture components are too small and numerous ($\sim$1000) to be discerned individually.



Northbound highway



Southbound highway

Figure 4.19: Two motion patterns for a sequence of the CLIF dtaaset shown in each row. Each column shows (L-R): Gaussian components making up the motion pattern shown as error ellipses; and conditional expected flow shown by color and brightness according to the color wheel.

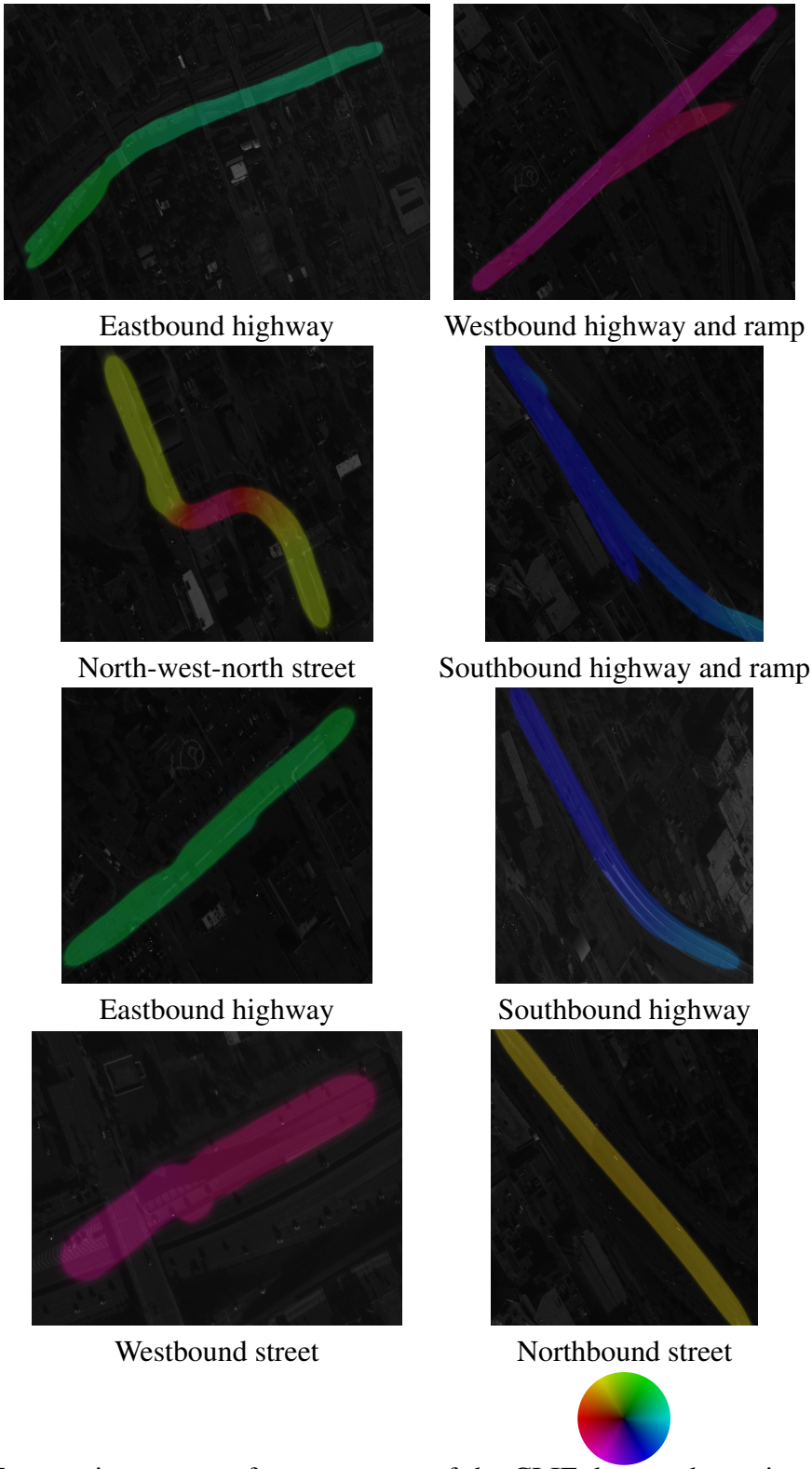


Figure 4.20: Two motion patterns for a sequence of the CLIF dtaaset shown in each row. Each column shows (L-R): Gaussian components making up the motion pattern shown as error ellipses; and conditional expected flow shown by color and brightness according to the color wheel.

is not captured by the optical flow. The next chapter proposes a direct, and unlikely application of discovery and representation of motion patterns, by applying the proposed framework to videos containing humans performing actions, imaged from frontal, almost oblique views. The discovery, semantic interpretation, and potential uses of the motion patterns are discussed.

CHAPTER 5: MOTION PATTERNS AS ACTIONS REPRESENTATION

Imagine you were asked, "*describe* the *action* in Figure 5.1 in *words*." A reasonable description would be that the action is performed by simultaneous upward motion of both arms, followed by a downward motion. In order to represent and recognize actions, one must then ask, How can arms be detected from videos?, Do the arms always move together?, if not, What is the smallest unit of an action?, How many such units are present in an action?, and How can the spatial information, shape, and motion of an action unit be learned and represented?

In this chapter, we attempt to answer these questions, and propose a method that conforms to the guidelines set forth by the answers. One can observe, that actions, even by natural language definition, comprise motion. It is therefore, meaningful to try and put the previously described motion representations to test, and evaluate the feasibility of representing human actions and primitives, as 'motion patterns'.

Recognition of human actions observed in videos is an important problem in computer vision and has received increased attention from the community in recent years [81]. The large number of applications of human action recognition in surveillance, retrieval, and analysis of videos has made this effort worthwhile and numerous methods have been proposed to this end. Some of these methods have performed impressively on standard action datasets available to the community. A majority of such methods follow the general, two-step paradigm of feature extraction, and recog-

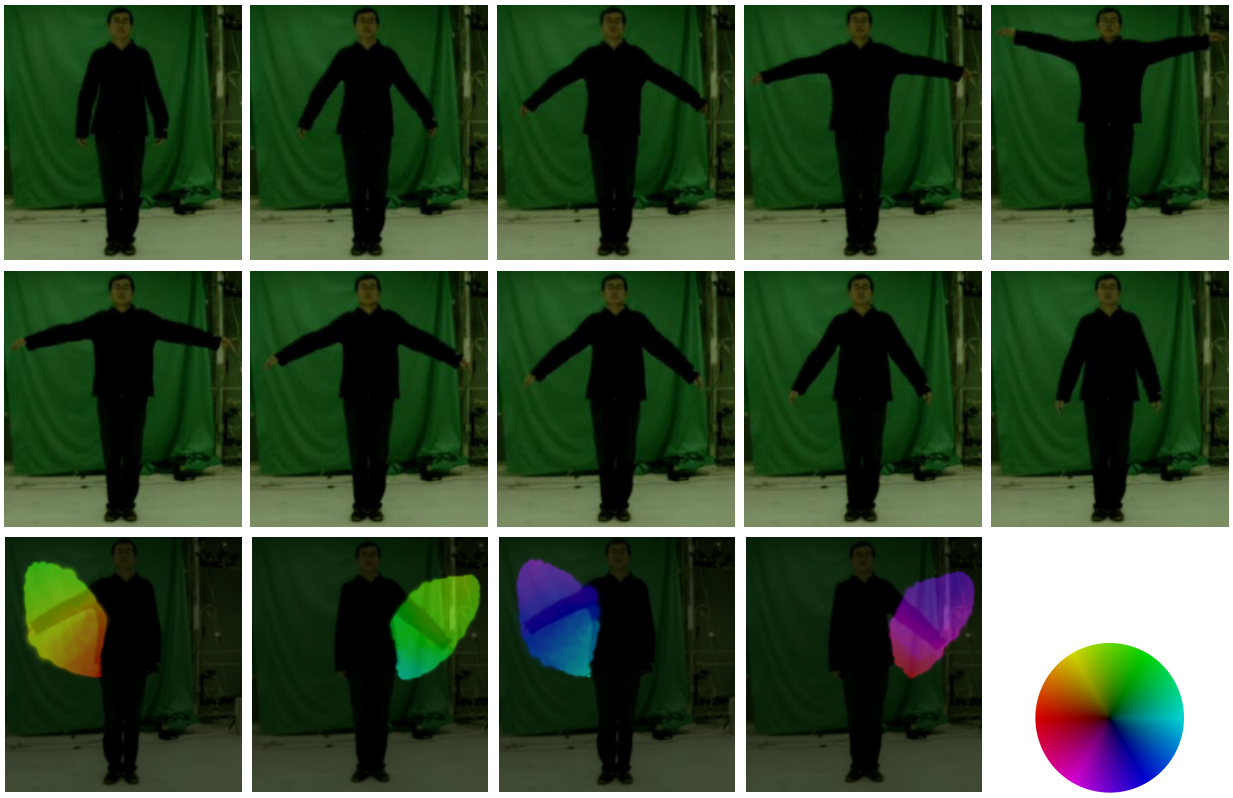


Figure 5.1: Top two rows: A single cycle of 'attention both' action [49], comprising upward and downward motion of both arms. Bottom row: four primitives representing the spatial extent and flow (as per color wheel) of the action. The primitives correspond to, (L-R) right arm moving up, left arm moving up, right arm moving down, and left arm moving down, resp.

nition using trained classifiers (e.g., SVM). Features include those derived from joints or parts trajectories [57], spatiotemporal interest points [47], dense features of actor bounding boxes [12], and those extracted from silhouettes [13], contours [87], and volumes [67], etc.

Descriptions of videos extracted in this way are often collapsed into a single feature vector, losing valuable spatial, temporal, and composition information. In terms of the level of abstraction, such representations, can be categorized from local or distributed (interest points, bag of words),

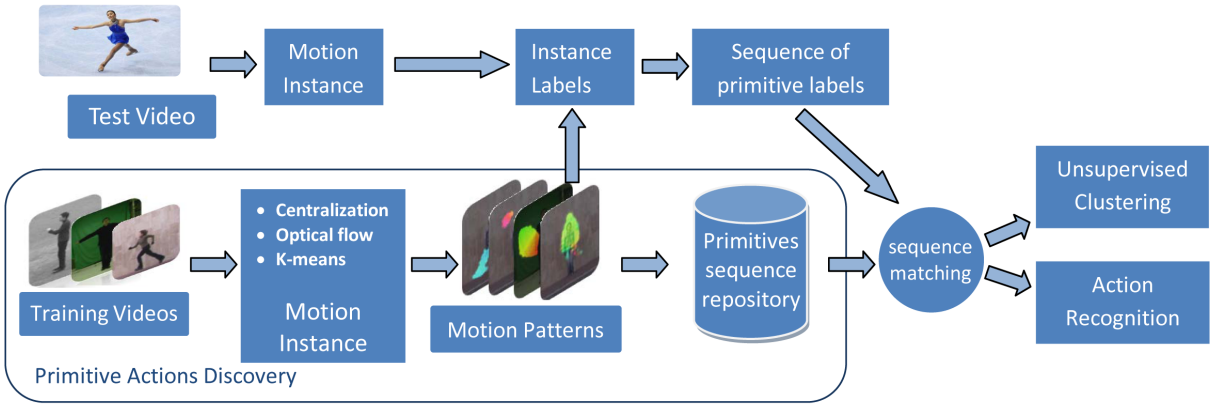


Figure 5.2: Process flow of the proposed approach for action representation and recognition.

to global or holistic (volumes, motion history images). We argue that both extremes are counterintuitive and semantically meaningless. In other words, on the one hand, bag of video words based approaches [47] generate codebook derived histogram representations, that have no semantic interpretation and cannot be illustrated visually, or described textually. On the other hand, holistic or global approaches [67] ignore the intuitive compositional hierarchy of action primitives.

In view of the traditional direction of research, we can summarize the intuitive questions asked earlier as, *What should be the representation of actions?*, and *How, if at all, can it be extracted from example videos?* We make a few key observations as an attempt to answer them. Firstly, a meaningful representation or data extraction and summarization method should incorporate the domain knowledge, that human actions are composed of primitives and the spatial and temporal relationships between these primitives can encode multiple discernable actions. Secondly, these primitives can be shared among actions, and an action's representation need not be completely independent of the rest. Thirdly, the representation of these primitive actions should reflect the real

world interpretation (semantic labeling) of the sub-actions. In other words, a meaningful action primitive is one which can be illustrated visually, and described syntactically, for example, 'left arm moving upwards'. We argue and show experimentally, that given such a representation, the task of recognition is almost trivial, which is what would be expected from a human analyst. In other words, if the vocabulary of actions is indeed semantically meaningful, one should be able to describe it using textual labels, and given such a sequence of descriptions, a human should be able to recognize the action, without actually observing any visual data at all. This paper proposes such a representation for human actions.

Given a video containing an action, an algorithmic overview of the proposed approach is as illustrated in fig 5.2: 1) a frame difference based foreground estimation, and 'centralization' of the actor to remove translational motion, thus resulting in a stack of rectangular image regions centered around the human; 2) computation of optical flow to obtain 4d feature vectors (x, y, u, v); 3) clustering of feature vectors to obtain components of a Gaussian mixture; 4) spatio-temporal linking of Gaussian components resulting in instances of primitive actions; 5) merging of primitive action instances to obtain final statistical representation of the primitives. Given a test video, instances of action primitives are detected in a similar fashion, which are labeled by comparing against the learned primitives. Sequences of observed primitives in training and test videos are represented as strings and matched using simple alignment [53] to classify the test video. The contributions of the proposed work are:

- completely unsupervised discovery of representative and discriminative action primitives without assuming any knowledge of the number of primitives present, or their interpretation,

- a novel representation of human action primitives that captures the spatial layout, shape, temporal extent, as well as the motion flow of a primitive,
- statistical description of primitives as motion patterns, thus providing a generative model, capable of estimating confidence in observing a specific motion at a specific point in space-time, and even sampling, to reconstruct very accurate representative flow of training data,
- highly abstract, semantically meaningful representation of primitives which can be labeled syntactically as components of an action, thus making the recognition task extremely straightforward.

In light of this discussion, we now present our method for representing human actions, which is completely unsupervised, semantically meaningful, and greatly simplifies the recognition task. Specifically, the next section describes in detail, our method for discovering primitive actions, by exploiting the framework proposed in Chapter 4, and Section 5.2 elaborates on the process of recognizing and classifying actions observed in unseen, test videos. Experiments performed on two human action recognition datasets are reported in Section 5.3.

5.1 Action Representation

The goal of the proposed method is to represent primitive actions as statistical description of regions of similar motion flow in videos containing human actions. Chapter 4 has introduced such a method ([64]), for estimating traffic patterns for surveillance videos. A common approach to obtaining motion is by quantizing it in terms of optical flow [41], computed from sequences of im-

ages depicting the action (as in [83, 28]). It can be observed that whole-body translational motion is not helpful in discerning human actions, e.g., the difference between running and walking is due to the distinct patterns of articulations as compared to difference in speeds which is subjective and depends on factors such as viewpoint, etc. On the other hand, computed optical flow in videos of such actions tends to be dominated by whole body translation, rather than the articulated motion. It is therefore, desirable to compensate for such translational motion by approximate alignment of the agent performing the action.

5.1.1 Actor Centralization

The proposed method begins by performing such an alignment where necessary. This is a simple process which includes computation of a simple intensity difference image for consecutive frames, and thresholding of this difference image to obtain the motion blob, which is then represented as a bounding box. These bounding boxes obtained in successive frames are then stacked together to obtain a sequence of cropped image frames for the entire training data set. The training is therefore performed in an unsupervised fashion, i.e., estimation of action primitives does not make use of the labels in the training data. We make sure that these bounding boxes are the same size for a given data set. We call this process 'centralization'. Figure 5.3 contrasts the optical flow computed from original frames, versus centralized bounding boxes of actors, where the latter captures articulated motion, while the former captures only the rigid body translational motion.

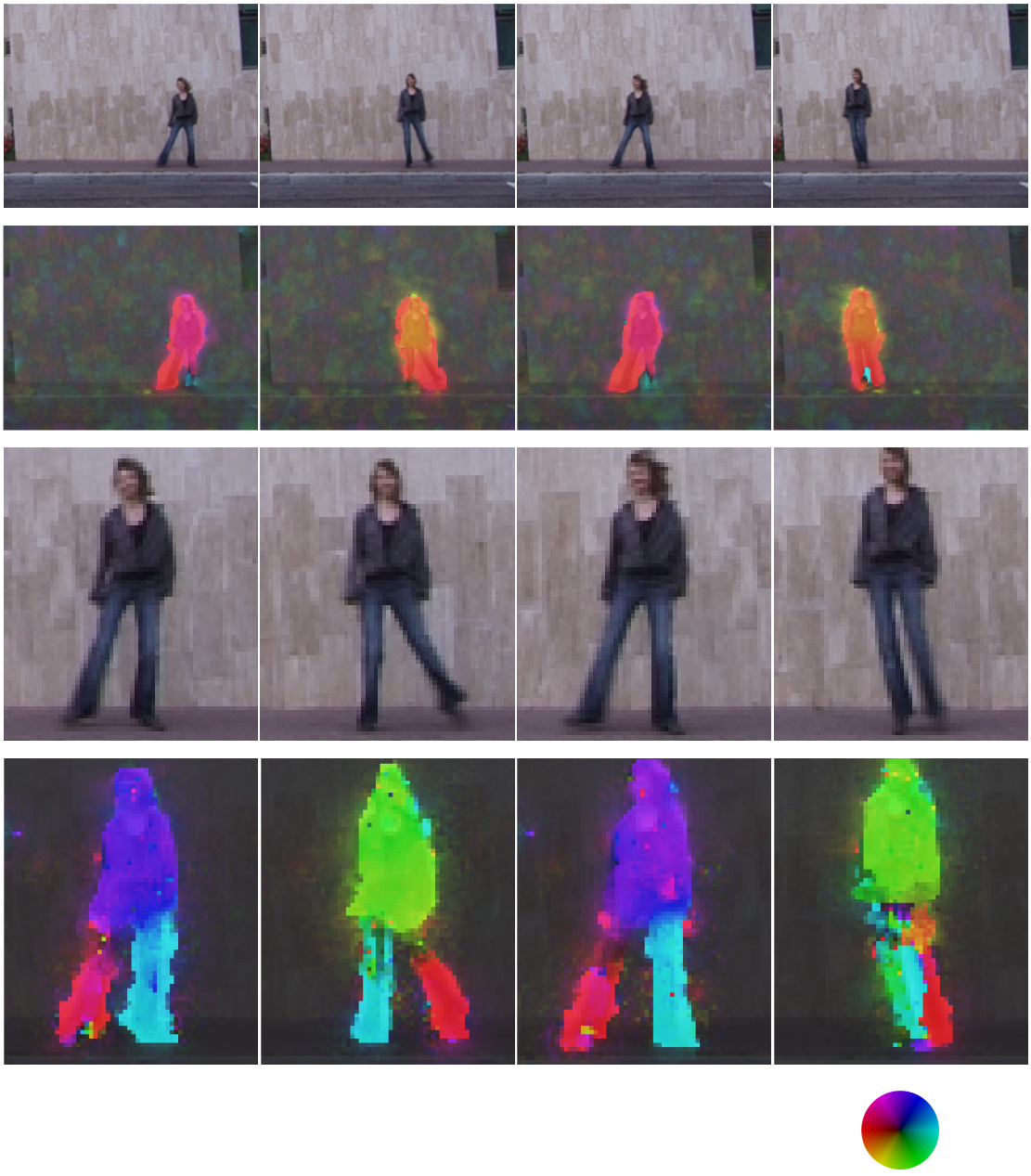


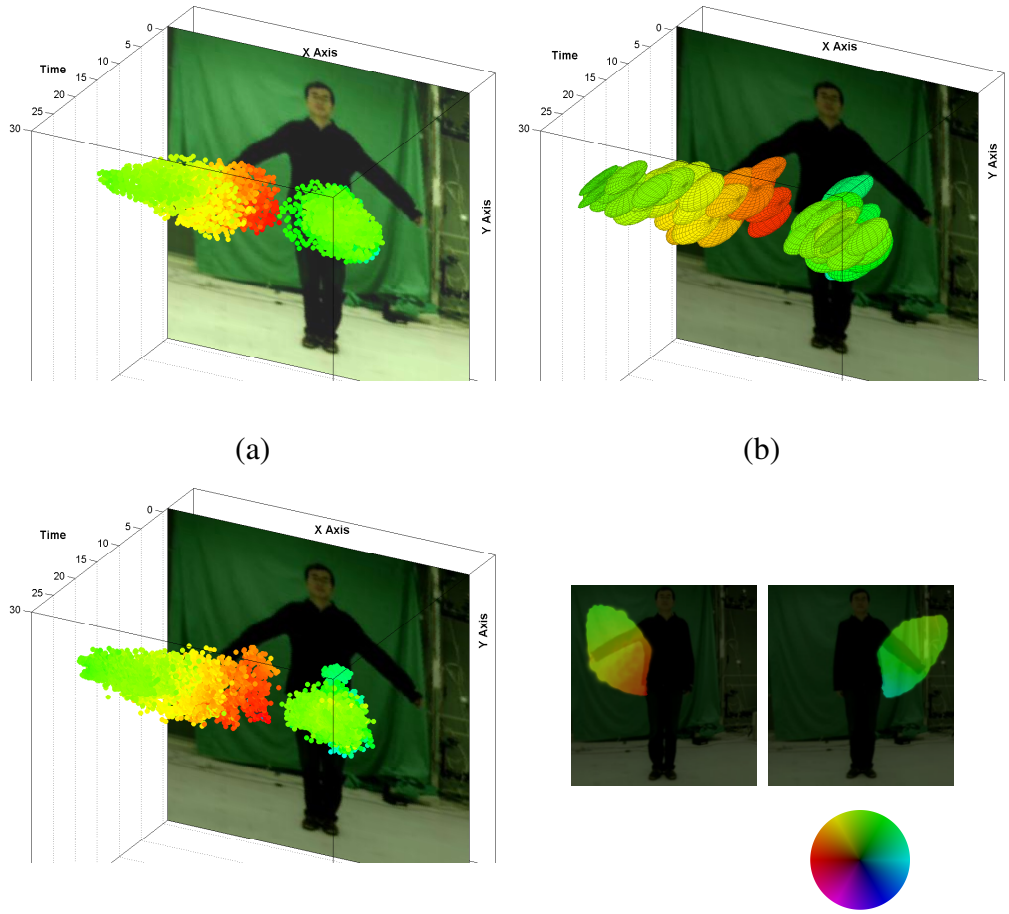
Figure 5.3: Process of optical flow computation. Top row shows 4 frames from Weizmann 'Side' action, and second row shows corresponding optical flow, which essentially captures rigid body, translational motion. The proposed method computes coarse bounding boxes around actors in each frame, as shown in third row. The resulting image patches are then stacked together, and optical flow is computed as shown in the fourth row. Notice that our aim of capturing *articulated* motion is well satisfied using this simple approach. No foreground or motion segmentation is shown here.

5.1.2 Optical Flow Computation

Next, Lucas-Kanade optical flow [41] is computed for the centralized training videos. Some of the noise in optical flow is eliminated by removing flow vectors with magnitude below a small threshold. The remaining flow vectors are then assumed to be values taken by the random variable, $\mathbf{F} = (x, y, u, v)$, where (x, y) is a location in the image, and (u, v) are the horizontal and vertical components of the optical flow vector at (x, y). We propose that each action primitive be described as a statistical distribution over the random variable $\mathbf{F}$, where each distribution is represented as a mixture of Gaussians. The goal of the training phase then, is to estimate the parameters of each such mixture distribution, where the number of primitives (motion patterns) as well as the number of components in each pattern's mixture distribution are unknown.

5.1.3 Gaussian Mixture Initialization

We adapt the method of [64] for this purpose, and begin by performing a K-means clustering of all the 4d feature vectors obtained. The value of K is not crucial and the goal is to obtain many, low variance clusters, which will become the Gaussian components in the motion patterns mixture distributions. The clustering is performed separately for each group of k frames, representing a short clip of video. The K-means clustering is performed using simple Euclidean distance, and the time of occurrence of each data point (optical flow vector) is ignored. The clustering results in a set of N Gaussian components, $\mathbf{C} = \{C_i\}$, for the entire training set, and the mean, μ_i and covariance matrix Σ_i for the i^{th} component are computed.



(a) (b)

(c) (d)

Figure 5.4: Illustration of primitive discovery from optical flow. (a) optical flow features from 30 frames of 'attention both' action, shown as colored dots where color represents flow magnitude and orientation by brightness and color resp. according to the color wheel on the right. Two primitives are intuitively discernable. (b) k-means clustering and component linking results in two primitive instances which meaningfully describe the two sub-actions. Gaussian distributions are shown as error ellipses at 2.5σ, in $(x, y, magnitude)$, but placed in $(x, y, time)$. (c) optical flow data points *sampled* from the two mixture distributions representing the primitives are almost identical to the original data. (d) Expected optical flow of conditioned on location, for the two primitives.

5.1.4 *Instances of Action Primitives*

The eventual goal is to find out which of these components belong to each primitive action's distribution. We notice that the action primitive, represented as a motion pattern, repeats itself within the video of an action (because actions are cyclic), as well as within the training data set (because there are multiple examples of each action). Therefore, we first attempt to further group the Gaussian components, such that each repetition of a primitive is represented by a such high level cluster. A simple similarity constraint is proposed in [64] to group components into an instance. Following this proposed measure, we define a weighted graph, $G = \{\mathbf{C}, E, W\}$, where E and W are $N \times N$ matrices corresponding to edges and their weights. The element $e_{i,j}$ is 1, if an edge exists between two components, C_i and C_j. This will be the case whenever C_i and C_j occurred in consecutive, k-frames long, video clips. The edge weight, $w_{i,j}$ for the edge $e_{i,j}$ is given as,

$$\frac{1}{4\pi^2\sqrt{|\Sigma_j|}} exp\left[-\frac{1}{2}\left(\hat{\mathrm{f}}_i - \mu_j\right)^\top \Sigma_j^{-1}\left(\hat{\mathrm{f}}_i - \mu_j\right)\right], \tag{5.1}$$

where $\hat{\mathrm{f}}_i = (\hat{x}_i, \hat{y}_i, u_i, v_i)$ is the transition prediction for component C_i, where, $\hat{x}_i = x_i + ku_i$, and $\hat{y}_i = y_i + kv_i$. The weighted graph G, is then converted into an un-weighted one, by removing edges with weights below a certain threshold. The threshold is chosen as the Gaussian probability of a data point at 1.5σ. A connected components analysis of this un-weighted graph gives P sequences of Gaussian components, each of which is assumed to be a single occurrence of an action primitive, e.g., one example of the right hand moving to the right while waving. Each such sequence of components (action primitive instance) is a Gaussian mixture, $S_i = \{C_m\}$, where $1 \leq i \leq P$, and $1 \leq m \leq M_i$, where M_i is the number of Gaussian components in the i^{th} mixture,

S_i. We observe that these action primitives are shared in multiple similar actions, e.g., 'right arm going upwards' is shared by 'one hand waving' as well as 'both hands waving' (Table 5.2).

5.1.5 Action Primitives Estimation

As mentioned earlier, the instances computed are multiple occurrences of the same primitive actions. The final step in training is to merge the representations of these occurrences into a single Gaussian mixture for each primitive action. This is done by computing the KL divergence between each Gaussian mixture, and merging the ones with low divergences. The components of the merged distributions then define the mixture model of the action primitive. The set of Q primitives can be represented as $\mathbf{A} = \{a_q\}$, where $1 \leq q \leq Q$.

5.2 Action recognition

Given the mixture distributions of action primitives, we deduce the sequence of primitives occurring in the training examples of a labeled action class, by indexing into the stack of training videos. The result is a temporal sequence of primitives that represent a particular class. These primitives can also co-occur in which case, an arbitrary order (e.g., ascending order of primitive number, q) is used. A synthetic example of such a sequence is $\{a_1, a_5, (a_3, a_4), a_1, a_5, (a_3, a_4), \ldots\}$, where within each cycle, the primitive action a_5 is preceded by a_1, and followed by the co-occurrence of primitives a_3 and a_4. The sequence depicts two cycles of the action. We observe in our experiments, that most actions are adequately represented by very few primitives. This is due to the nature of the primitive discovery process and representation, where a specific sub-action of

a limb is usually decomposed into at most two primitives. The motion patterns thus represent the sub-actions at the highest possible level of abstraction, e.g., a continuous motion of the right arm moving upwards need not be broken further. Most actions are therefore represented by two primitives (or groups of co-occurring primitives) as can be seen in Tables 5.2, and 5.3.

A process similar to the training phase is performed for the test video. However, since a test video is typically short, and contains at most a few cycles of the action, we do not perform the final step of primitive instance merging. This is because, for most test videos, only a single instance of action primitives is observed. The problem of action recognition then reduces to a very simple string matching problem, where the letters in the string represent the primitive number. We perform string matching using the well known Needleman-Wunsch algorithm [53], thus obtaining a confidence score in a particular matching. This process starts after the action primitives have been learned across all training videos. Since the information about which video clip a Gaussian was observed in, is available, we can index back into that clip of the training video and extract the action label for that video clip. We, therefore, obtain a sequence of primitives for each example video used in training. For each action class, we have as many strings as the number of examples. These strings are compared among themselves, and the average matching score is obtained. The string with the highest score is chosen as the final representation of the action class. In other words, this final string is the one that is most similar across the training videos of that particular action. The primitives (motion patterns or Gaussian mixture distributions) that are not part of any of the action strings are then discarded. These are the primitives that are not shared across multiple

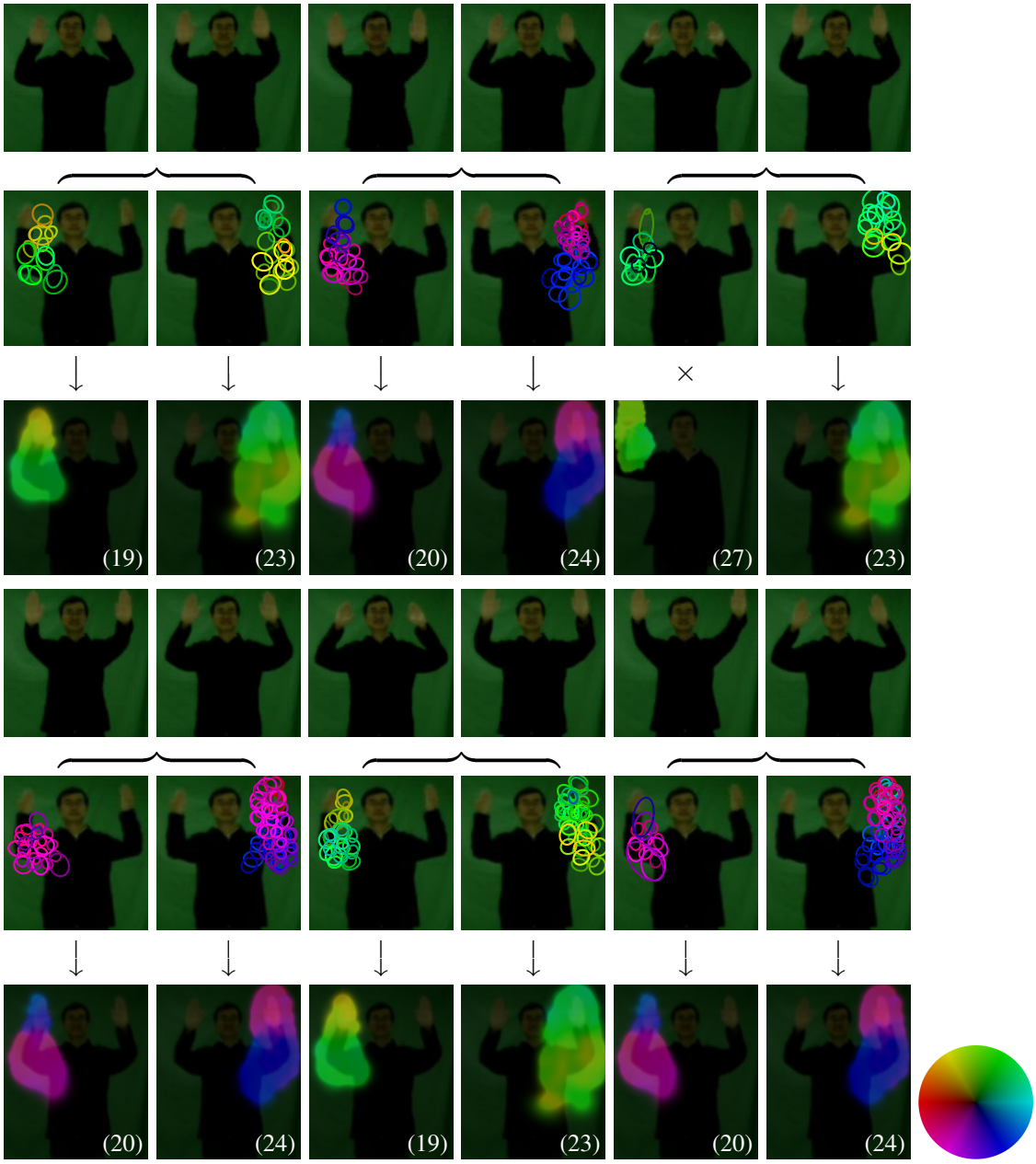


Figure 5.5: Process of action recognition: (rows 1,4): 12 frames from a test video of action 'go back', representing 3 cycles. (rows 2,5): 6 *pairs* of co-occurring primitive instances obtained for the test video, shown as Gaussian error ellipses drawn at 1.5σ, with color and brightness corresponding to mean directions, and magnitudes resp., and horizontal braces ($\{$) indicating co-occurring pairs. (rows 3,6): Corresponding learned primitives. The downward arrows indicate the correctness of matching for the test video. The only incorrect label ($\times$) is of the 5^{th} detected primitive, labeled as 27 instead of 19. Matching score for test video is 91.67%, for the 'go back' action.

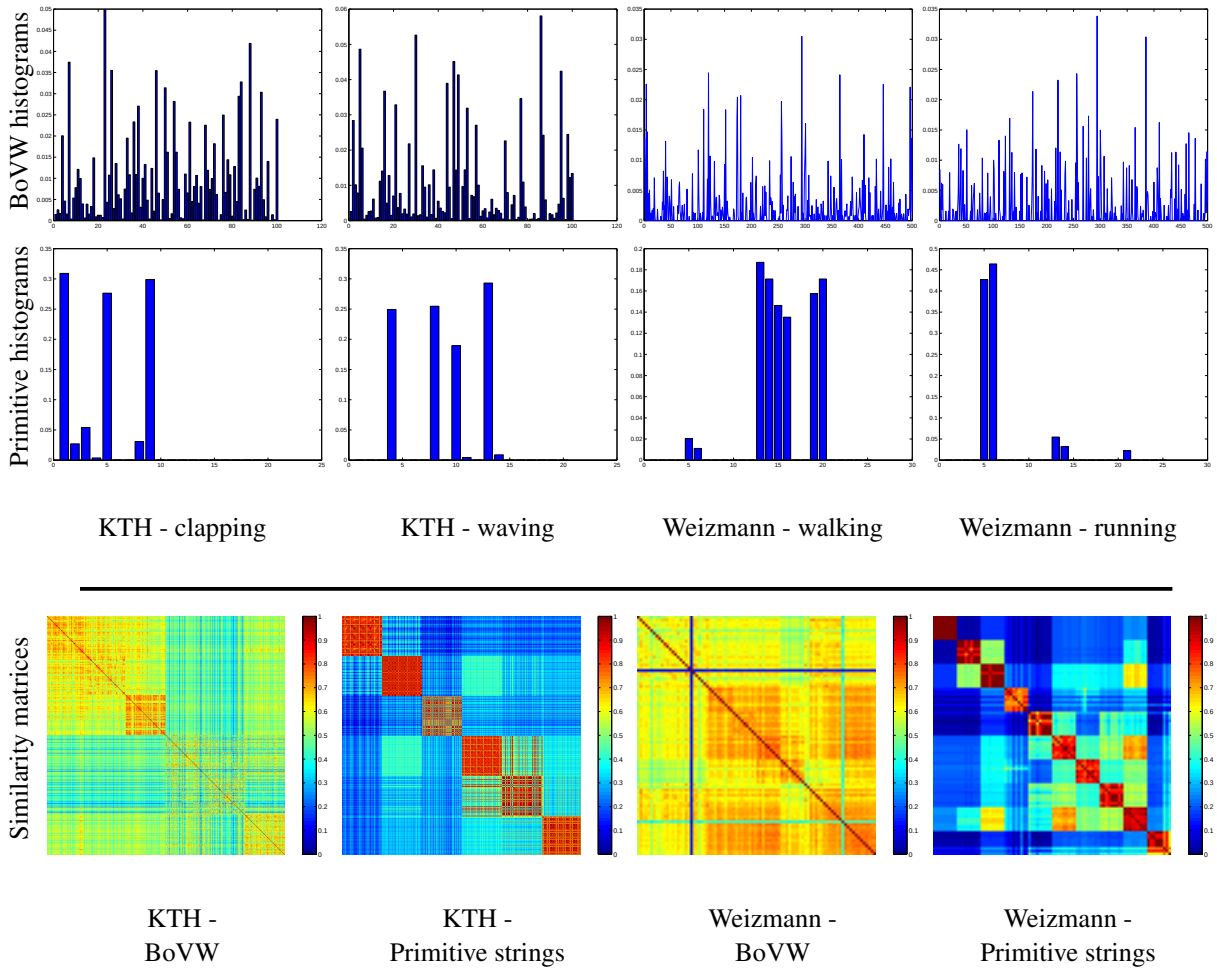


Figure 5.6: Comparison of discriminative power of histograms of video words (BoVW) versus histograms of proposed action primitives. First two rows show average histograms over 4 classes from 2 datasets. Notice the sparsity and discriminatory nature of the primitive histograms. Notice that histogram is a weaker representation than strings, and is only used here for visualization. The third row shows similarity matrices for all the examples in the datasets using histogram intersection for BoVW, and string matching for primitive strings. Observe how the distinct classes are clearly visible along the diagonal, for matrices of string matching scores.

examples of the same action class, for any action. The co-occurring pairs (or groups) are sorted arbitrarily, and for string matching, all possible permutations are tried, and the best chosen.

The string representing the test video is similarly compared against final strings of all action classes and the one with the highest confidence is chosen as the classification of the test video. Despite the simplicity of the recognition approach, very encouraging results are obtained as reported in section 5.3. This is a direct consequence of the semantically meaningful nature and high level of abstraction of the proposed action representation. In actuality, we observed that even a literal string matching performs reasonably well without global alignment of primitive labels.

One way to describe the goal of the proposed method is to emphasize estimation of improved, semantic vocabularies and representations, rather than on complex classification schemes. In order to test if the motion patterns based representation is discriminative enough, we visualize similarities between examples of actions, some of which are shown in figure 5.6.

5.3 Experiments

The proposed approach has been evaluated for 4 human action datasets, as well as composite datasets created from these 4 sets, to show representation and recognition quality and performance. For all datasets, we demonstrate the superiority of the proposed representation with the two main types of experiments, detailed in the following subsections. Moreover, we compare our representation to the well known bag of video words using both experimental setups, and also quantify the ability of temporal extent estimation, number of cycles estimation, and effect of the parameter K

Table 5.1: Experiment statistics: Number of motion pattern instances, initial number of primitives estimated, and final number of primitives after filtering of unshared patterns. The value of K used for each experiment is also shown. Notice that as few as 50 action primitives sufficiently represent even very large datasets (e.g., composite dataset combined 25 action classes).

Supervised	Value of K	# primitive instances	# initial primitives	# final primitives
Kecklab	20	620	45	31
Weizmann	30	2000	52	35
Unsupervised				
Weizmann	50	2500	43	26
KTH	50	15500	51	20
IXMAS	50	820	48	30
Composite	50	19800	88	55

used in K-means. Some statistics related to learning of action primitives as motion patterns, are reported in table 5.1.

5.3.1 Supervised Classification

In this experiment, a dataset is divided into training and testing sets. The primitive actions representation is learned from examples in the training sets. The training videos are then represented as strings of the primitive labels. Given a test video, pattern instances are estimated using the proposed approach, which are then represented as Gaussian mixtures. These distributions are then compared against the learned primitive distributions using KL divergence, and labeled. The test video is thus also represented as a string of learned primitives. Finally, a string matching based nearest neighbor classifier is employed to assign an action label to the test video.

The first dataset used in our experiments is the Kecklab Gesture dataset [49]. This video dataset has 14 actions related to gestures, and the test sequences in this set involve camera motion, moving objects, occlusions, and background clutter. Coarse detection of moving objects is first performed to obtain bounding boxes which are scaled to the same size (250×200), and are stacked together. Optical flow is then computed, and K-means clustering is performed using 20 Gaussian components, in each 5 frames long clip. Using the proposed method, about 620 motion instances are extracted and 45 action primitives are obtained from those. Primitives that are not well-shared across examples of the same class are also discarded and finally, 31 primitives are obtained from the training videos, which are shown in figures 5.7 and 5.8. It can be observed that each primitive is semantically meaningful and can be given a textual label as described in the caption. Sequences of these primitives define the actions in the dataset as listed in table 5.2. Using the proposed action representation and recognition method, an average accuracy of **94.64%** is reached, which is comparable to the work of [49], who proposed the dataset.

The second dataset used in our experiments is the Weizmann dataset [67], which consists of videos of 10 actions performed by 9 individual actors, thus resulting in 90 videos. Background difference is used to extract a bounding box around actors. Hence, the optical flow only captures the local motion information (wrt actor's body) instead of the global (translational) motion. The bounding box is extracted and resized to 250×200. A continuous video of the entire dataset is divided into 5890 clips, in which around 2000 instances, and 52 action primitives are obtained. Finally 35 primitives are found after discarding non-shared primitives. Some of these primitive

actions are shown in figures 5.9 and 5.10. Notice the similarity between primitives 17, 18, 21, 22 of Gesture dataset (figure 5.8), and primitives 1, 2, 3, and 4, of the Weizmann dataset (figure 5.9).

It is observed that due to partial, spatial similarity of some actions (e.g., around the hand in single-hand waving and both hand waving, and around the torso, in para-jumping and sideways jumping), there is an increased mutual sharing of primitives among actions. Furthermore, due to the increased level of human body articulation in the Weizmann dataset, e.g., lower body motion, an action needs more primitives to be represented well. For example, the 'side' action, depicting translation by jumping sideways, is represented by the string, $\{7, 11, 12, 8, 9, 10\}$ where primitives are shown in figures 5.9 and 5.10. Primitives 7 and 8 correspond to torso moving upwards and downwards respectively, while 11 and 12 represent leftward motion of the left and right legs respectively. Primitives 9 and 10 depict the corresponding rightward movement of the legs.

Given the representation of these 10 action classes as primitives, recognition using leave-one-out cross-validation was performed, and an average accuracy of **99%** was obtained, which is comparable to some very complex state of the art methods that employ sophisticated classifiers for recognition, given elaborate representations [67, 49]. The encouraging results, thus obtained, verify our claims about the feasibility of a simple recognition approach, while the qualitative results of primitive action representation (figures 5.9 and 5.10) clearly show the intuitive, compact, and meaningful nature of the proposed representation.

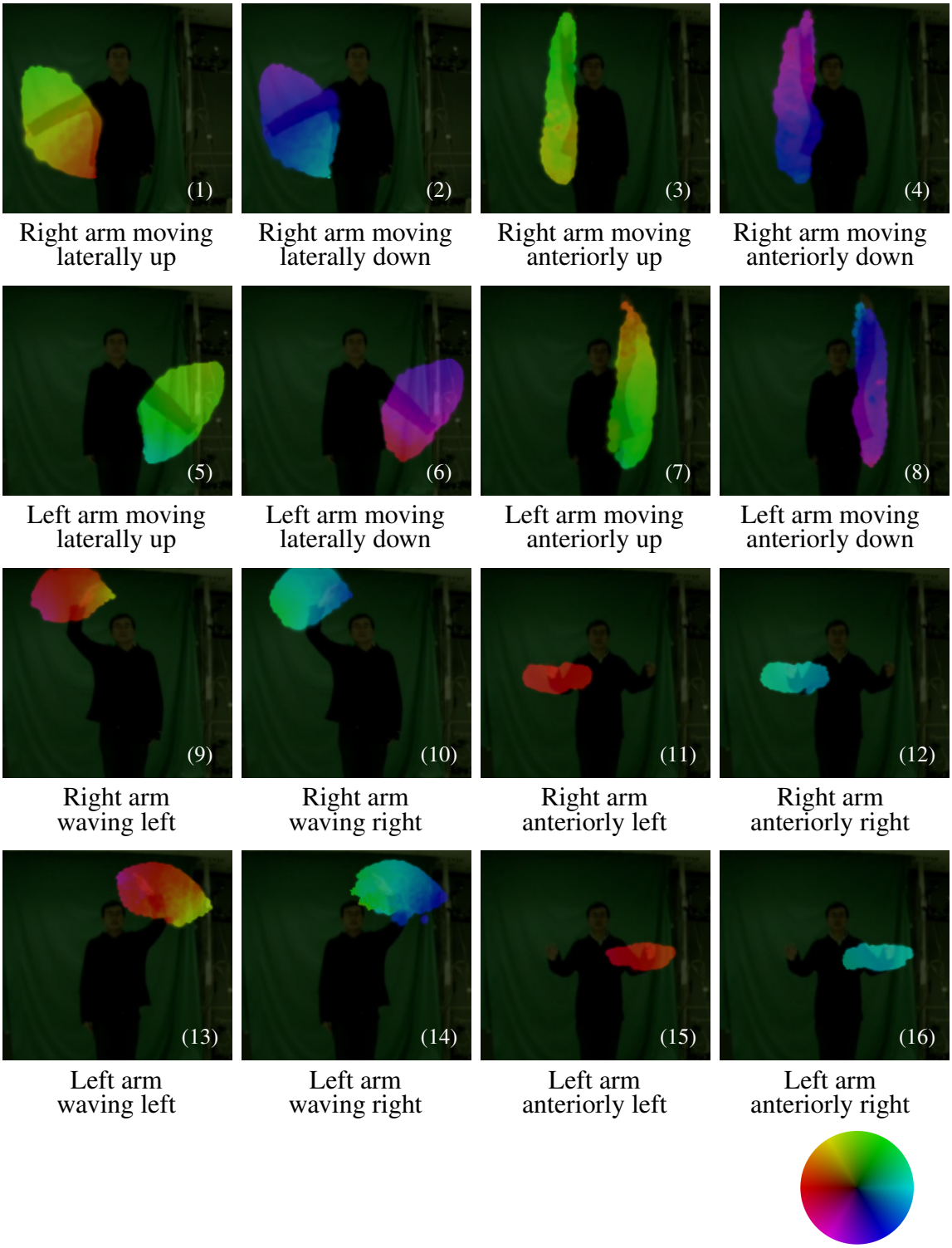


Figure 5.7: Action primitives 1 to 16, discovered in the gesture dataset [49] are shown by the conditional expected optical flow of the Gaussian mixtures that represent them. The direction is encoded by color as shown in the color wheel on bottom right, while the increasing magnitude is depicted by increasing brightness. See table 5.2 for the list of actions represented by each primitive.

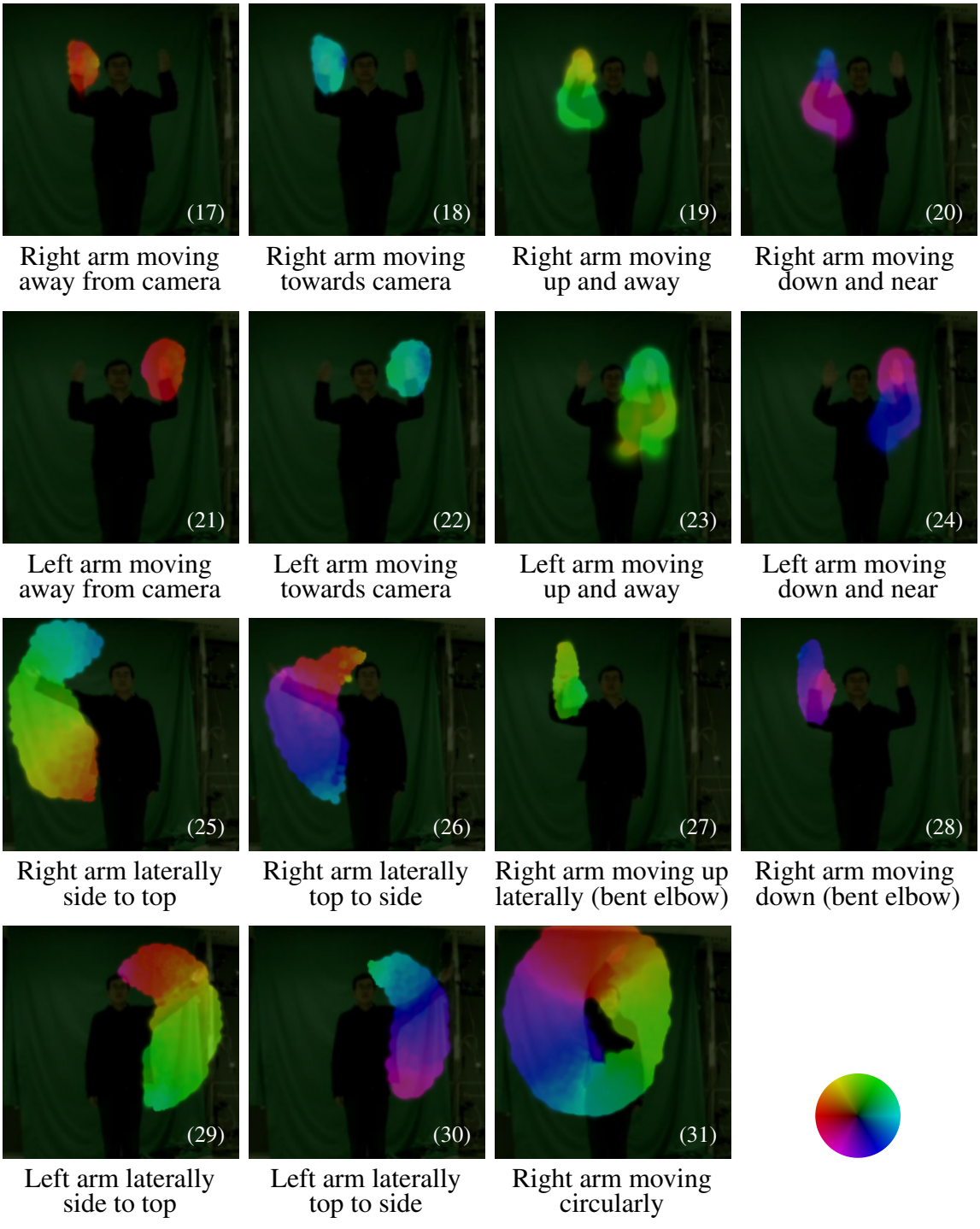


Figure 5.8: Primitive actions 16 to 31 for the gesture dataset [49]. Colors and brightness encode flow similar to previous figures. Observe that the direction of motion in (31) is captured by the primitive's representation (CCW for observer). See table 5.2 for the list of actions represented by each primitive.

Figure 5.9: First 12 of the 25 Weizmann dataset primitives. The captions refer to limbs (right/left) from actor's POV (true left/right), and the direction of motion from the viewer's POV. (1-4) are used to described both wave1 and wave2; (5-6) depict running; (7-8) comprise jumping in place; (9-12) and particularly interesting since they show that the proposed method captures the true articulation. Notice that when the actor perform 'sideways galloping' going to *her* right, although the left foot stays in place, the entire left leg is extended to the right with respect to the body center.

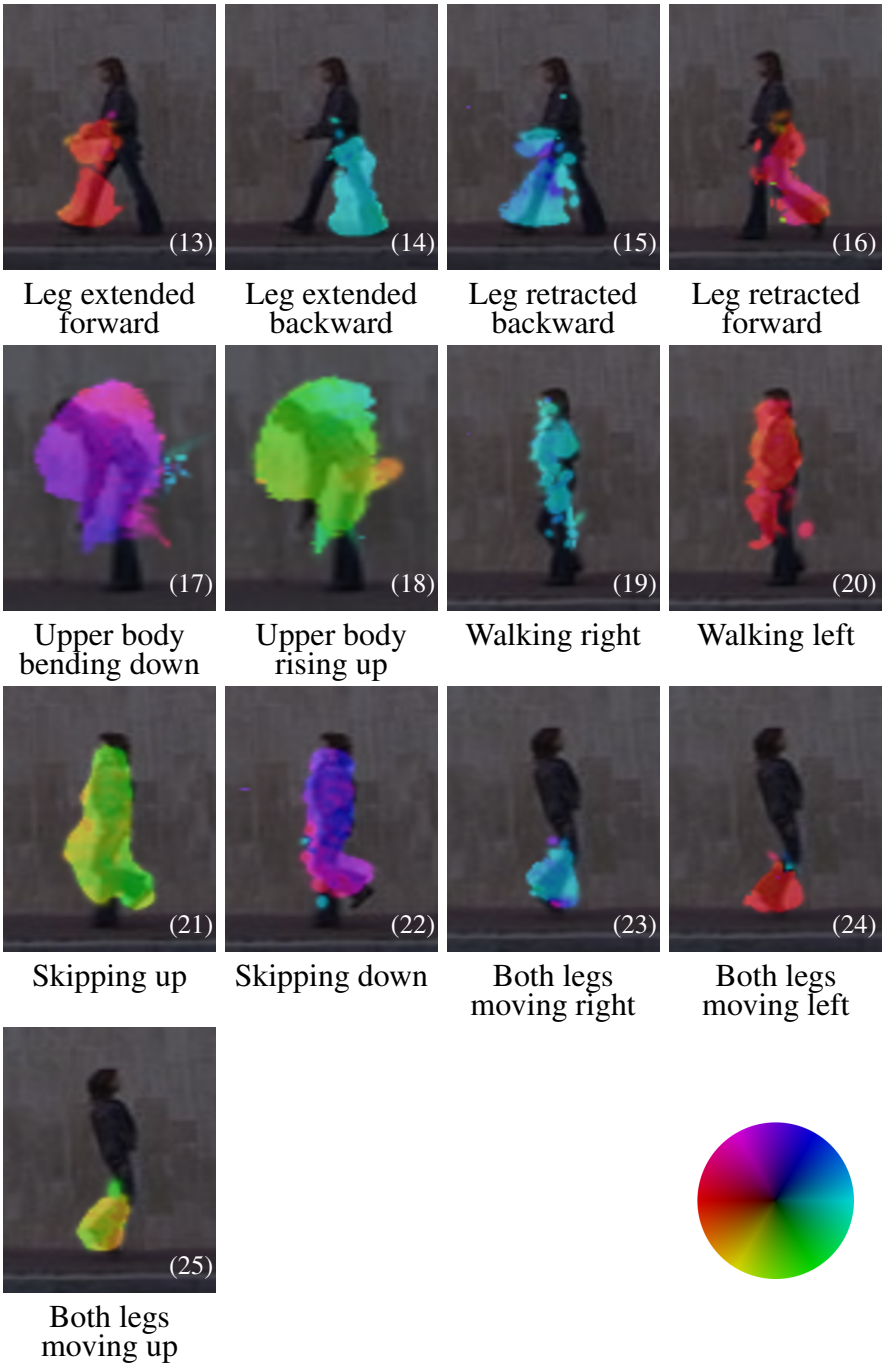


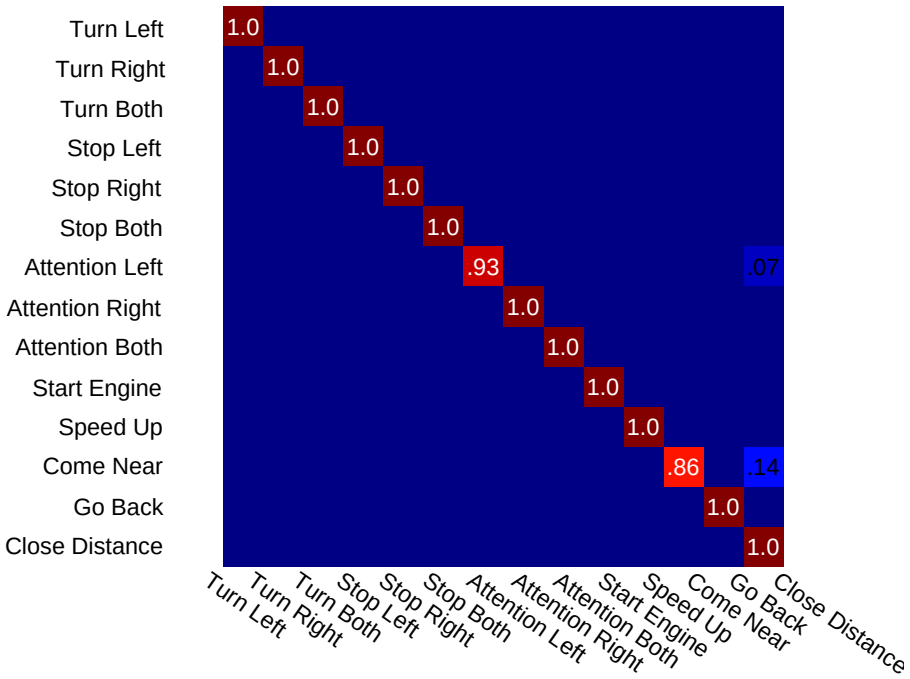
Figure 5.10: Primitives 13-25 for the Weizmann dataset: (13-16) correspond to walking. Notice that the 'retraction' is due to centralization; (17-18) depict the bending action; (19-20) represent residual motion after centralization for 'walking'; (21-22) comprise 'skipping'; (23-25) represent 'jumping' and capture the leftward, rightward and upward joint motions of the lower legs, in addition to primitives (7-8). Notice that primitives like 16, 21, and 22 readily capture the *shape* of the atomic action unit, which is helpful for recognition. See table 5.3 for details about the sequence of primitives used by each action.

Table 5.2: Kecklab gesture dataset: Primitives (shown in figures 5.7, and 5.8) used by each of the actions. Notice that a primitive can be part of multiple actions (e.g., 1), while an action may be represented by a single primitive (e.g., 31). Groups of primitives (shown by { }) co-occur. The last row represents an undefined observed action wherein a person moves their arm(s) upwards on the way to performing the 'attention' actions.

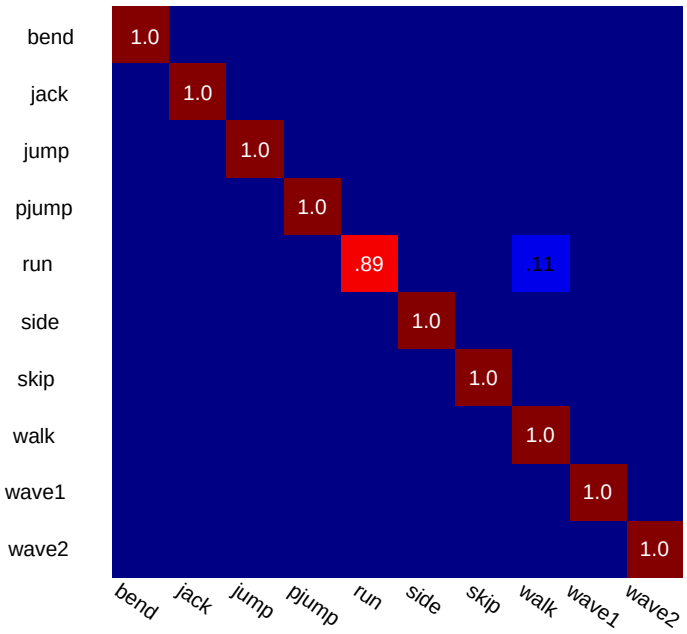
Action	Primitives	Action	Primitives
Turn left	1, 2	Attention right	13, 14
Turn right	5, 6	Attention both	{9, 14}, {10, 13}
Turn both	{1, 5}, {2, 6}	Speed up	27, 28
Stop left	3, 4	Come near	{19, 23}, {20, 24}
Stop right	7, 8	Go back	{17, 21}, {18, 22}
Stop both	{3, 7}, {4, 8}	Close distance	{11, 16}, {12, 15}
Attention left	9, 10	Start engine	31
Non-action	25, 26, 29, 30		

Table 5.3: Weizmann dataset: Primitives (shown in figures 5.9, and 5.10) used by each of the actions.

Action	Primitives
Bending	17, 18
One hand Waving	1, 2
Two hand waving	{1, 4}, {2, 3}
Running	5, 6
Para-jumping	7, 8
Sideways gallop	7, {9, 10}, 8, {11, 12}
Walking	{13, 14}, {15, 16}, 19, 20
Jumping	23, 24, 25, 7, 8
Jumping Jacks	7, {9, 10}, {1, 4}, 8, {11, 12}, {2, 3}
Skipping	21, 22



Kecklab Gesture dataset



Weizmann dataset

Figure 5.11: Confusion tables for action recognition performance on the Kecklab gesture and Weizmann datasets.

5.3.1.1 *Describing Complicated Unseen Actions*

One of the advantages of the primitives being semantically meaningful, is that it can describe complicated actions. To verify, we construct a composite video which is composed of pieces of frames from the Kecklab Gesture dataset, such that the left and right halves of the original sequences are joined to define a new, composite, unseen action that is more complex in terms of the number of primitives. The process of primitive detection is in such a video illustrates the following contributions: The proposed method has the ability to represent a hitherto unseen action (not trained for), by automatically detecting the primitive or atomic actions it contains. It can be used to classify it as unknown, or retrieve the most similar action by string matching. Even a complex action, involving multiple, simultaneously moving actuators (or limbs), is easily representable and recognizable. Figure 5.12 shows an example of temporal extent estimation for primitive actions, in a composite action video.

5.3.1.2 *Temporal Extent*

As mentioned earlier, one of the advantages of the proposed approach is that the nature of action representation allows easy detection of cycle extents, and counting of the number of cycles in test videos. We demonstrate this point by treating test examples, as a single long video and counting the number of times a cycle of an action appears in the video. The quantitative output of this experiment, performed on the Kecklab Gesture dataset, is reported in figure 5.13.

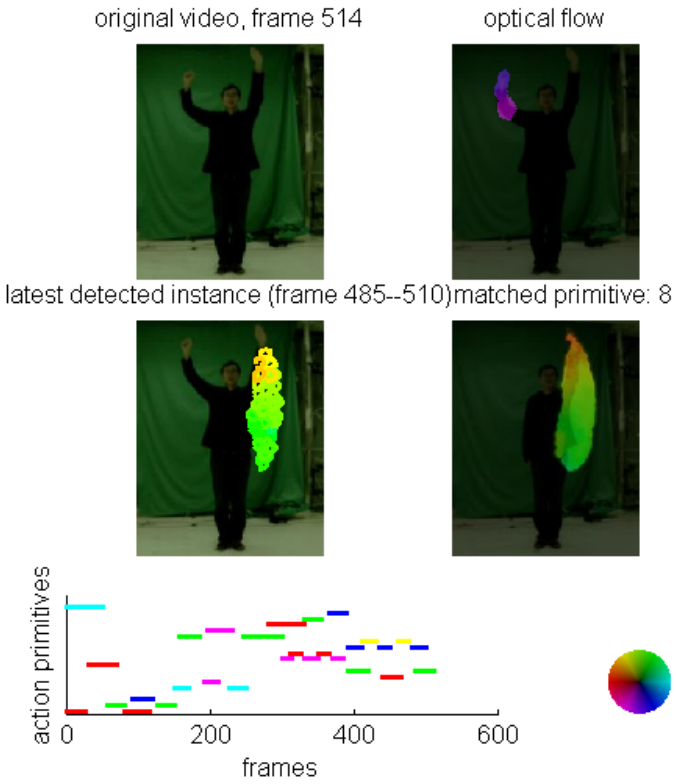


Figure 5.12: A frame of output video showing original video, optical flow, primitive action detected, and corresponding learned primitive action for a video of a composite action (see text for details). A timeline at the bottom shows the primitive actions detected so far in the video, along with their temporal extents in terms of start and end frames.

5.3.1.3 One-shot learning

To verify our claim of semantic representation, we attempted action recognition using as few as possible training examples using the proposed method. This experiment was performed for both Kecklab gesture and Weizmann datasets, and the results are plotted in figure 5.14, also showing comparison to BoVW method. For Kecklab dataset, 9 examples per action are available, and we performed incremental learning and recognition increasing the number of available videos from 1

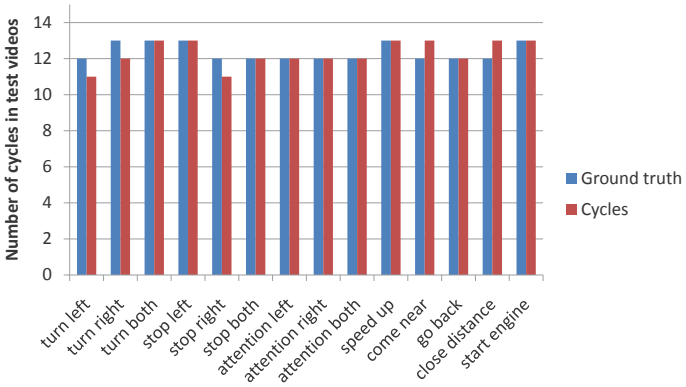


Figure 5.13: Counting the number of cycles of each action in a single, long test video. This is an additional benefit of the proposed representation, and the results are very close to the ground truth. Due to the meaningful nature of the representation, estimation of number of action cycles, and their extents is easy.

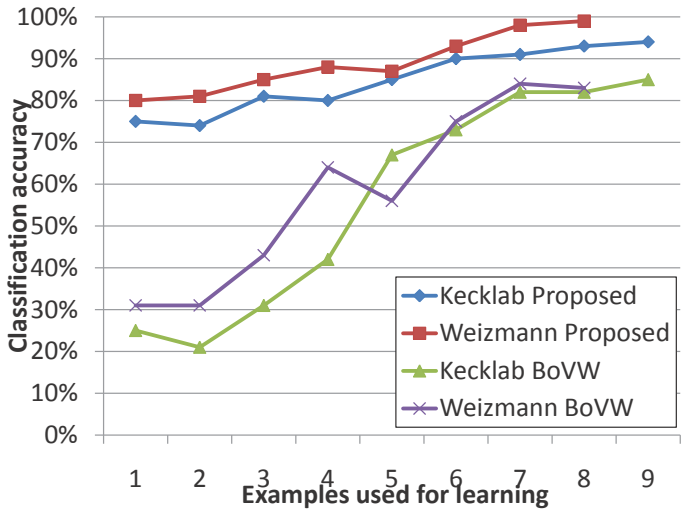


Figure 5.14: Classification of actions using a variable number of training examples. The values corresponding to 1 on the X-axis are essentially results of *one-shot learning*. Using our method, an average accuracy of around 80% is obtained for both the Kecklab gesture and Weizmann datasets, using a *single* training example for model learning. The proposed method outperforms BoVW by a large margin.

to 9 (testing on test set). Similarly for Weizmann dataset, we trained on a variable number videos (1 to 8), as we added videos from each of the 9 performers incrementally. For both datasets, using even a *single* example, around 80% recognition rate is achieved, compared to about 30% for BoVW.

5.3.2 Unsupervised Classification

In this experiment, *all* videos in the dataset are used to learn the action primitives representation, and the videos are represented as strings of primitive labels. A string matching similarity matrix of all videos is then constructed (figure 5.6, row 3), and spectral clustering is performed to obtain clusters of videos. Each video is assigned the label of the dominating cluster, and comparison of the assigned label with the ground truth label, provides classification accuracy.

Unsupervised clustering over the entire Kecklab gesture dataset was performed using the proposed action primitives representation, as described before, and the proposed method obtained an average classification accuracy of **91.64%**. The same experiment was als performed for the Weizmann dataset, and the string representation exploiting proposed action primitives, reached an average classification accuracy of **89.7%** compared to 67% for histograms of video-words (BoVW) using a codebook of size 500, and employing Mean-shift clustering. The reason for this large difference in performance is the discriminative power of the primitive representation (figure 5.6-3rd row).

Encouraging results were observed for experiments on KTH and IXMAS datasets as well. The KTH dataset has 600 videos, and consists of 2391 sequences from six action classes. Unsupervised

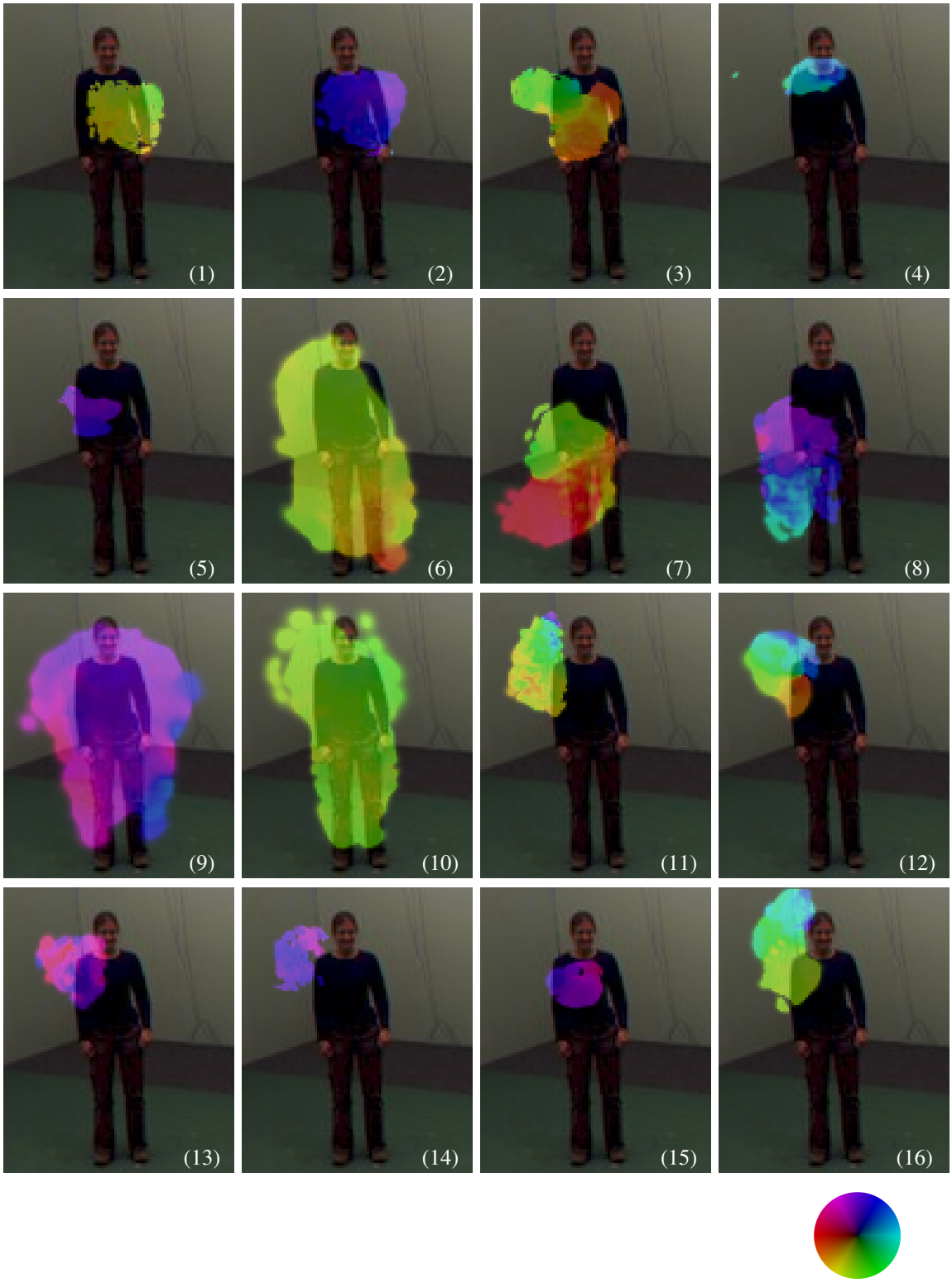


Figure 5.15: Some of the primitive actions for the IXMAS dataset. These images show the conditional expectation of optical flow. See table 5.4 for the list of actions represented by each primitive.

Table 5.4: IXMAS dataset: Primitives shown in figure 5.15, used by each of the actions.

Action	Primitives	Action	Primitives
Check watch	1, 2	Scratch head	14
Crossarm	3, 4, 5	Throw overhead	15, 16
Getup	6	Kick	7, 8
Pick up	6, 9	Sit down	9
Point	11	Punch	12, 13

clustering of strings representing motion patterns obtained from these sequences resulted in an accuracy of **91%** when using a maximum of 50 Gaussian components per video clip. Accuracy for the BoVW framework using vocabulary size of 100, and histogram intersection as distance measure was close to 65%.

IXMAS is a much more challenging dataset, and using 30 discovered action primitives, accuracy of unsupervised clustering on this dataset was around **63%**. The same experiment repeated for BoVW using a vocabulary size of 500 (more than 15 times the proposed vocabulary), reached an accuracy of only 53%. All of these results are summarized in figure 5.16.

5.3.3 Cross-dataset Unsupervised Clustering

We experimentally demonstrate and quantify our claim that the proposed representation is meaningful and allows simple but robust recognition and classification of actions, by exploitation of the core, underlying structure of the spatiotemporal flows induced by motion. In addition to recogni-

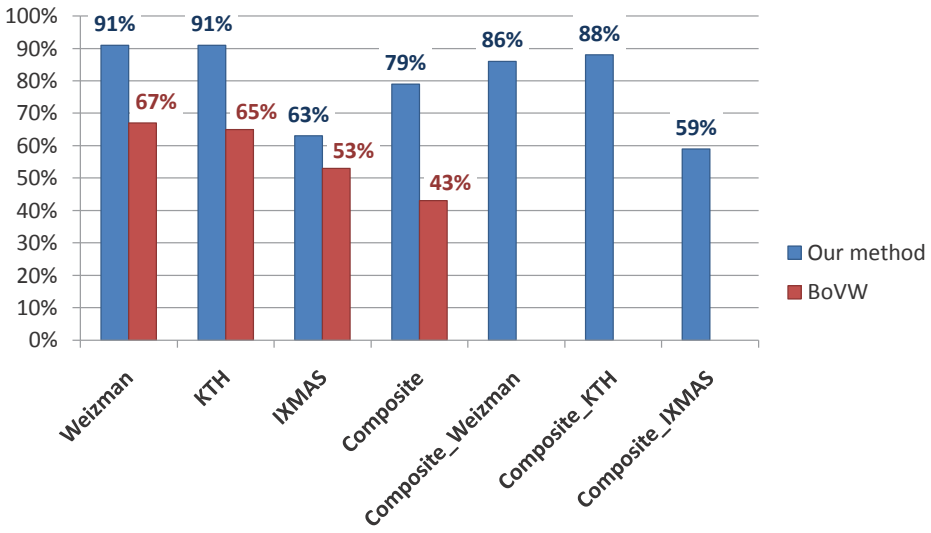


Figure 5.16: Average accuracy of classification by unsupervised clustering of strings representing action videos. The last 3 bars correspond to clustering of videos within these individual datasets (like the first three), but using the shared primitives representation, obtained from the composite dataset.

tion from very few examples (figure 5.14), the ultimate test of these claims is to perform action recognition across multiple datasets.

We perform this experiment by combining three diverse sources, namely, the KTH, IXMAS, and the Weizmann datasets. We performed a variety of experiments with this dataset. First, the classification accuracy for the composite dataset, is obtained by attempting unsupervised (unlabeled) clustering of data points, where a data point represents a single video of an unlabeled action, as a string of action primitives. The action primitives are learned using all the videos of the composite set, and 55 primitive actions are discovered. Spectral clustering over a graph of string similarity scores resulted in a classification accuracy of **79%**, compared to only 43% for BoVW

using a codebook of size 500 (>9 times the size of our vocabulary). Figure 5.17 shows the performance of action recognition on the Composite dataset consisting of **25** action categories from KTH, IXMAS, and Weizman datasets. These results correspond to unsupervised classification, via spectral clustering of string matching score based similarity matrix.

Another interesting experiment performed using this dataset was, to use the 55 action primitives learned in the *composite* dataset (Weizmann + KTH + IXMAS), to represent and classify videos in the 3 datasets individually, and compare the performance against the representation using dedicated, indigenous primitives learned within each dataset (i.e., against results reported in the previous subsection). Naturally, it is expected that the performance would degrade using the composite primitives, due to larger variance in shared actions, and more noise. However, a meaningful high-level representation should be largely robust to these problems, which is what we observed in our experiments. The performance using composite primitives on Weizmann, KTH, and IXMAS, was 86%, 88%, and 59% respectively, compared to, 91%, 91%, and 63% respectively for the individual datasets, which is not a significant deterioration. These results are summarized in figure 5.16.

We also quantified the effect of the parameter, K, the number of components in each video clip obtained using the K-means algorithm. The larger values of K essentially correspond to increased granularity of representation (even though the statistical distribution is defined continuously spatially and in motion space). We observed that as conjectured earlier, a high, computationally reasonable value of K, lets the performance of our method peek and flatten very quickly. The quantified results of analysis of the effect of parameters, is shown in figure 5.18.

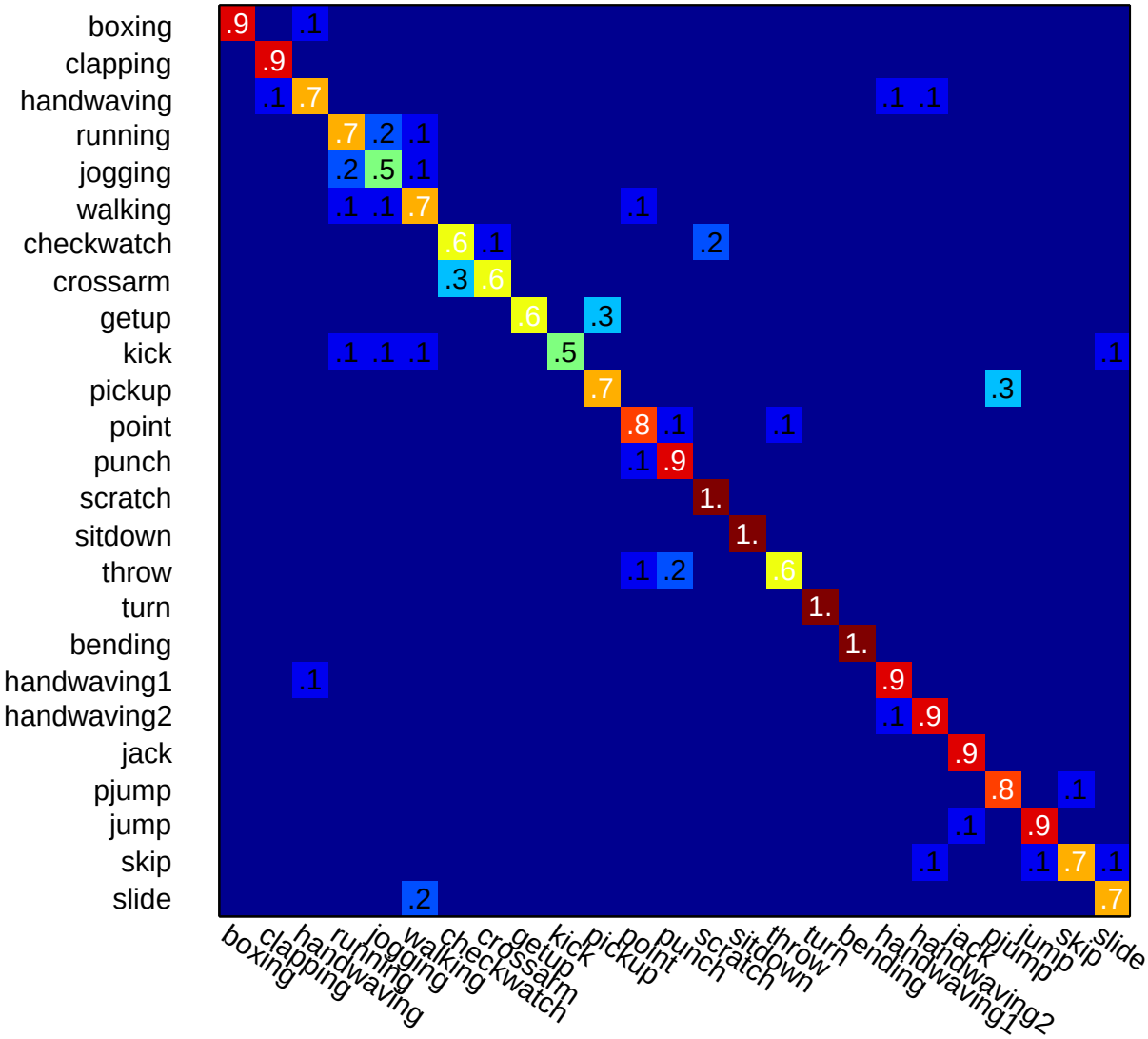


Figure 5.17: Confusion table for the Composite dataset consisting of 25 action classes from the KTH, IXMAS and Weizmann datasets. The confusion table shows results of unsupervised clustering of primitive action strings, representing the action videos. The unsupervised classification was performed by spectral clustering of the string matching score similarity matrix. The value of K (in K-means) used for this experiment was 50. The average accuracy obtained for classification of videos in the Composite dataset was around **79%**.

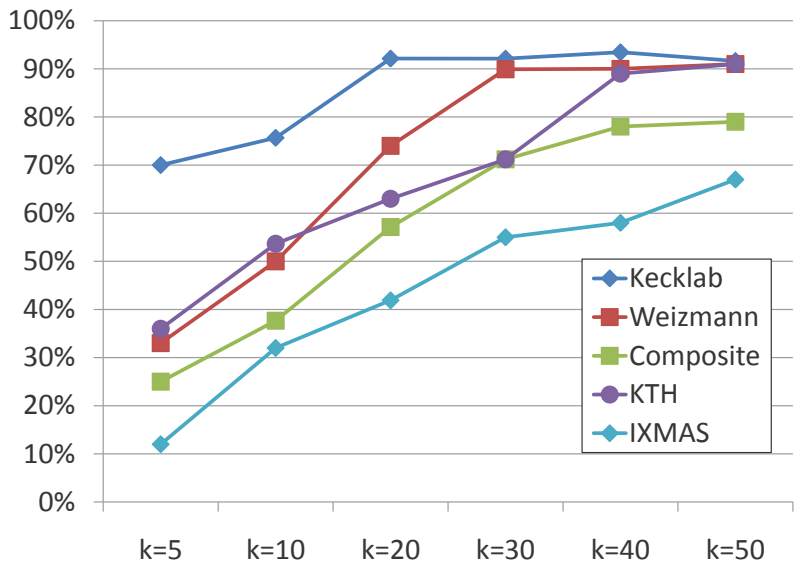


Figure 5.18: Average accuracy of classification by unsupervised clustering of strings representing action videos. In addition to high classification accuracy, the figure also shows that as claimed in the text, the performance of our approach reaches a plateau as the value of K, in the K-means clustering is increased. The proposed primitive representation allows simple clustering framework to achieve non-trivial results, i.e., around 90% for each of the datasets, Kecklab, Weizmann, and KTH, for K = 50.

5.4 Summary

The work proposed in this chapter is a novel method of primitive action representation where the number of action primitives is computed and the primitives estimated in a completely unsupervised manner. The semantically meaningful nature of the representation, allows us to employ a very simple recognition scheme which has numerous advantages, like recognition of unseen composite action, insensitivity to occlusions (partial primitive list), invariance to splitting of primitive during

Table 5.5: Quantitative comparison of the proposed action classification method with some of the state of the art techniques. We note however, that not all of these methods employ the same experimental settings, and evaluation methods, e.g., leave one out versus splitting. Nevertheless, these statistics provide a bird's eye view of where our framework stands with respect to the related work, despite its simplicity. The result figures of related methods are quoted from respective papers and [49].

Kecklab	Accuracy (%)	Weizmann	Accuracy (%)	KTH	Accuracy (%)
Proposed	94.6	Proposed	99	-	-
-		Proposed (spectral clustering)	91.3	Proposed (spectral clustering)	91
Lin [49]	91	Wang [80]	97.2	Liu [50]	94.1
		Jhuang [37]	98.8	Jhuang [37]	91.6
		Thurau [73]	94.4	Lin [49]	95.7
		Lin [49]	100	Ahmad [2]	87.6
		Fathi [19]	100	Wang [80]	92.4
		Schindler [65]	100	Nowozin [55]	87

learning, detection of cycle extents and number, etc. Most importantly, the method is semantically meaningful to the extent that it can even be used to come up with a natural language description.

CHAPTER 6: CONCLUSION

The unifying theme behind the work presented in this thesis has been the development of generic, and generative models representing object motion in general scenes. Examples of such motion include rigid, holistic motion of vehicles and humans, as well as non-rigid, articulated motion. To this end, we have investigated two different but closely related motion representations, and demonstrated their feasibility and usefulness by learning and applying them to a variety of diverse problems, such as anomaly detection, use as prior for object tracking, and human action recognition. Specifically, the differences between these two representations can be summarized as follows:

- The global distribution proposed in Chapter 3 represents all observable motion as a single probability density function of spatiotemporal transitions, while the framework proposed in Chapter 4 simultaneously learns and separates distinct patterns of motion into multiple Gaussian mixture distributions.

- The single probability density function is a non-parametric one, represented using Kernel Density estimation while the motion patterns representation is in the form of the parametric Gaussian mixtures. The parametric representation is obviously easier to manipulate and analyze, for example, in terms of sampling and expectation computation, etc. There is however, no limitations on using one in place of the other.

- The first proposed representation also explicitly encodes the temporal aspect of the motion (i.e., instantaneous versus long term motion) by extracting transitions over a range of temporal durations (e.g., 0 to 5000 milliseconds). On the other hand, the mixture distributions for motion patterns employ optical flow which by definition is instantaneous. It can however be analyzed over longer time durations by interest point tracking or particle advection, without explicitly incorporating the notion of objects. Therefore, once again, this is not an inherent limitation on either of the representations.

Some of the specific contributions of the work proposed in this thesis are outlined below.

6.1 Summary of Contributions

1. Global Representation of Motion Patterns
   (a) Non-parametric representation
   (b) Explicit modeling of temporal dimension for multiscale analysis
   (c) Point sampling for trajectory prediction
   (d) Application of representation to multiple problems
2. Gaussian Mixtures for Motion Patterns
   (a) Statistical representation of raw optical flow
   (b) Simultaneous learning and separation of semantically distinct patterns

(c) Robust measure of spatiotemporal affinity between flow points and Gaussian components

(d) Representation of four dimensional representation as two dimensional conditional expected flow

3. Motion Patterns Representation of Actions

(a) Novel representation of articulation in human actions

(b) Unsupervised discovery of primitive actions

(c) Capturing of spatial layout, shape, temporal extent, as well as motion flow of primitive

(d) Generative representation of articulated human motion

(e) Reduction of action recognition to simple string matching

(f) Classification via nearest neighbor and spectral clustering

6.2 Future Directions

Representation and learning of motion from video sequences has been a traditionally important aspect of research in computer vision. However, the main premise of this thesis is generalization of multiple methods into a single, general-purpose representation. The frameworks proposed in this thesis are only the first steps in this direction, and the alleviation of existing assumptions and limitations, provides fertile ground for future research. Moreover, discovery of new computer vision problems that can be solved robustly and efficiently using representations similar to the

proposed ones, is also an interesting direction for future work. We describe some open problems, potential solution, and future directions.

6.2.1 Inferring Invisible Motion Patterns

Even with an explosion in the number of visual sensors deployed in a multitude of scenarios, e.g., urban surveillance, a 100% coverage of all regions of importance is not a reasonable expectation. It is therefore important to devise representations and algorithms that *infer* the required information about an invisible region, from the visible ones. A traditional area of research in this regard is the well known problem of non-overlapping sensors. A typical application in this regard is the track correspondences across such cameras, usually attempted by learning transformation (or lack thereof) in appearance, and temporal aspects of the transitions from one field of view to another. One possible extension of the work proposed in this thesis is the unsupervised learning of patterns of motion expected within such invisible regions. In addition to the obvious motivation and usefulness of such techniques, it can be noticed that this is a reasonable question to ask. After all, there is usually enough information in the visible fields of view, e.g., vehicle entry and exit times, etc., that should allow learning a coarse sketch of what is going on in the invisible region.

6.2.2 View Invariant Representation of Motion Patterns

One obvious limitation of the representations proposed in this thesis is that they are not view invariant. For example, two motion patterns corresponding to left turns can be very different due to the length of distance covered during the turn (scale), absolute direction of motion (rotation),

and location of the turn (translation). Similarly, primitive human actions with the same semantic interpretation, can vary significantly, when represented as motion patterns. There is, therefore, a need to devise ways to mitigate the effects of simple geometric transformations on the motion patterns, in order to make them more generalized, and view-invariant. We have some work in progress in this direction, where we are attempting to automatically estimate affine transformations between two mixture distributions representing motion patterns. We first observe that the effect of an affine transformation on the mean of a 4d Gaussian representing location and flow in 2d is the piecewise multiplication of the location and flow with the transformation in homogenous coordinates. The covariance matrix of such a Gaussian is similarly transformed by computing, $H \times \Sigma \times H^T$, where H represents the 3×3 transformation matrix and Σ is the covariance matrix of the Gaussian. The goal then, is to find the transformation H between two Gaussian mixtures, such that some error measure is minimized. The two directions of research in this regard is the minimization of KL-divergence between the distributions, or an EM algorithm. The KL-divergence however, is a non-symmetric measure. The EM algorithm can be set up such that the goal is to only decide memberships of sampled points of one mixture, in the components of the other mixture. In other words, this problem can be compared to K-means, where the cluster centers, covariances, as well as weights are pre-determined and fixed. The goal is to decide which data points belong to which cluster, such the clusters' specifications remain unchanged.

LIST OF REFERENCES

[1] "Special issue on video communications, processing, and understanding for third generation surveillance systems". *Proceedings of the IEEE*, 89(10), Oct 2001.

[2] Mohiuddin Ahmad and Seong-Whan Lee. Human action recognition using shape and clg-motion flow from multi-view image sequences. *Pattern Recogn.*, 41:2237–2252, July 2008.

[3] J. A. Benediktsson and P. H. Swain. "Consensus theoretic classification methods". In *IEEE Transactions on Sys. Man and Cybernetics*, volume 22, pages 688–704, 1992.

[4] I. Biederman. "On the semantics of a glance at a scene". In *Perceptual Organization*, pages 213–253. Hillsdale, NJ: Lawrence Erlbaum Associates, 1981.

[5] A. Bobick and J. Davis. Real-time recognition of activity using temporal templates. In *WACV*, 1996.

[6] Matthew Brand and Vera Kettnaker. Discovery and segmentation of activities in video. *PAMI*, 22(8):844–851, 2000.

[7] H. Buxton. "Generative models for learning and understanding dynamic scene activity". In *Generative Model Based Vision Workshop*, 2002.

[8] R. Chellappa. Machine Recognition of Human Activities: A Survey. *Circuits and Systems for Video Technology, IEEE Trans. on*, 18(11):1473–1488, September 2008.

[9] R. Collins, J. Lipton, and T. Kanade. "Introduction to the special section on video surveillance". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), Aug 2000.

[10] Columbus Large Image Format (CLIF) 2006 Dataset. Available at https://www.sdms.afrl.af.mil/datasets/clif2006/.

[11] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[12] T. Darrell and A. Pentland. Space-time gestures. In *CVPR*, 1993.

[13] J.W. Davis and A.F. Bobick. The representation and recognition of action using temporal templates. In *CVPR*, 1997.

[14] S. L. Dockstader and A. M. Tekalp. "Multiple camera fusion for multi-object tracking". In *IEEE Workshop on Multi-Object Tracking*, 2001.

[15] P. Dollar. Behavior recognition via sparse spatio-temporal features. In *PETS*, 2005.

[16] R. Duda, P. Hart, and D. Stork. *"Pattern Classification"*. Wiley Interscience, 2nd edition, 2001.

[17] A. Elgammal, R. Duraiswami, and L. Davis. "The fast Gauss transform for efficient kernel density evaluation with applications in computer vision". *IEEE Trans. PAMI.*, 25, Nov 2003.

[18] T. J. Ellis, D. Makris, and J. K. Black. "Learning a multi-camera topology". In *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, 2003.

[19] A. Fathi and G. Mori. Action recognition by learning mid-level motion features. In *CVPR*, 2008.

[20] J.H. Fernyhough, A.G. Cohn, and D.C. Hogg. "Generation of semantic regions from image sequences". In *ECCV*, 1996.

[21] R. Filipovych and E. Ribeiro. Learning human motion models from unsegmented videos. In *CVPR*, 2008.

[22] Zhouyu Fu, Weiming Hu, and Tieniu Tan. Similarity based vehicle trajectory clustering and anomaly detection. In *ICIP*, 2005.

[23] A. Galata, N. Johnson, and D. Hogg. "Learning variable length markov models of behaviour". *Computer Vision and Image Understanding: CVIU*, 81(3):398–413, 2001.

[24] Andrew Gilbert, John Illingworth, and Richard Bowden. Scale invariant action recognition using compound features mined from dense spatio-temporal corners. In *ECCV*, 2008.

[25] L. Greengard and J. Strain. "The fast Gauss transform". *SIAM J. Sci. Statist. Comput.*, 12(1):79–94, 1991.

[26] W.E.L. Grimson, C. Stauffer, R. Romano, and L. Lee. "Using adaptive tracking to classify and monitor activities in a site". In *International Conference on Computer Vision and Pattern Recognition*, 1998.

[27] G. Hinton. "Products of experts". In *ICANN*, pages 1–6, 1999.

[28] J. Hoey and J. Little. Representation and recognition of complex human motion. In *CVPR*, 2000.

[29] D. Hoiem, A. Efros, and M. Hebert. "Putting Objects in Perspective". In *CVPR*, 2006.

[30] Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[31] R.J. Howard and H. Buxton. "Analogical representation of spatial events, for understanding traffic behaviour". In *10th European Conference On Artificial Intelligence*, 1992.

[32] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank. "A system for learning statistical motion patterns". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1450–1464, September 2006.

[33] T. Huang and S. Russell. "Object identification in a bayesian context". In *Proceedings of IJCAI,*, 1997.

[34] A. Hunter, J. Owens, and M. Carpenter. "A neural system for automated CCTV surveillance". In *IEE Intelligent Distributed Surveillance Systems*, 2003.

[35] O. Javed, K. Shafique, Z. Rasheed, and M. Shah. "Modeling inter-camera space-time and appearance relationships for tracking across non-overlapping views". *Computer Vision and Image Understanding (Elsevier)*, 109(2), Feb 2008.

[36] Omar Javed, Khurram Shafique, and Mubarak Shah. "Automated Visual Surveillance in Realistic Scenarios ". In *IEEE Multimedia*, pages 30–39, January - March 2007.

[37] H. Jhuang, T. Serre, L. Wolf, and T. Poggio. A biologically inspired system for action recognition. In *ICCV*, 2007.

[38] N. Johnson and D.C. Hogg. "Learning the distribution of object trajectories for event recognition". *Image and Vision Computing*, 14(8):609–615, August 1996.

[39] Neil Johnson and David Hogg. Learning the distribution of object trajectories for event recognition. In *BMVC*, 1995.

[40] I.N. Junejo, O. Javed, and M. Shah. Multi feature path modeling for video surveillance. In *ICPR*, 2004.

[41] T. Kanade and B.D. Lucas. An iterative image registration technique with an application to stereo vision. In *IJCAI*, 1981.

[42] R. Kaucic, A. Perera, G. Brooksby, J. Kaufhold, and A. Hoogs. "A unified framework for tracking through occlusions and across sensor gaps". In *CVPR*, 2005.

[43] Eamonn J. Keogh and Michael J. Pazzani. Scaling up dynamic time warping for datamining applications. In *ACM Int. Conf. on Knowledge Discovery and Data Mining*, 2000.

[44] V. Kettnaker and R. Zabih. "Bayesian multi-camera surveillance". In *CVPR*, 1999.

[45] E.B. Koller-Meier and L. Van Gool. "Modeling and recognition of human actions using a stochastic approached". In *2nd European Workshop on Advanced Video-Based Surveillance Systems*, 2001.

[46] L. Kratz and K. Nishino. Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models. In *CVPR*, 2009.

[47] I. Laptev. Learning realistic human actions from movies. In *CVPR*, 2008.

[48] Ivan Laptev and Tony Lindeberg. Space-time interest points. In *ICCV*, 2003.

[49] Zhe Lin, Zhuolin Jiang, and Larry S. Davis. Recognizing actions by shape-motion prototype trees. In *ICCV*, 2009.

[50] J. Liu and M. Shah. Learning human action via information maximization. In *CVPR*, 2008.

[51] J. Lou, Q. Liu, T. Tan, and W. Hu. "Semantic interpretation of object activities in a surveillance system". In *ICPR*, 2002.

[52] E. Memin and P. Perez. Dense estimation and object-based segmentation of the optical flow with robust techniques. *IEEE Transactions on Image Processing*, 7(5):703–719, 1998.

[53] S. Needleman and C. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, March 1970.

[54] Next Generation Simulation (NGSIM) dataset. Available at http://www.ngsim.fhwa.dot.gov/.

[55] S. Nowozin, G. Bakir, and K. Tsuda. Discriminative subsequence mining for action classification. In *ICCV*, 2007.

[56] J. Owens and A. Hunter. "Application of the self-organising map to trajectory classification". In *3rd IEEE International Workshop on Visual Surveillance*, 2000.

[57] V. Parameswaran and R. Chellappa. View invariants for human action recognition. In *CVPR*, 2003.

[58] E. Parzen. "On the estimation of a probability density function and mode". *Ann. Math. Stati.*, 33:1065–1076, 1962.

[59] A. Perera, C. Srinivas, A. Hoogs, G. Brooksby, and W. Hu. "Multi-Object Tracking Through Simultaneous Long Occlusions and Split-Merge Conditions". In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 666–673, 2006.

[60] P. Remagnino and G.A. Jones. "Classifying surveillance events from attributes and behaviour". In *BMVC*, 2001.

[61] N. Robertson and I. Reid. Behaviour understanding in video: a combined method. In *ICCV*, 2005.

[62] R. Rosales and S. Sclaroff. "Improved tracking of multiple humans with trajectory prediction and occlusion modeling". *In CVPR Workshop on the Interpretation of Visual Motion*, 1998.

[63] I. Saleemi, K. Shafique, and M. Shah. Probabilistic modeling of scene dynamics for applications in visual surveillance. *PAMI*, 31(8):1472–1485, 2009.

[64] Imran Saleemi, Lance Hartung, and Mubarak Shah. Scene understanding by statistical modeling of motion patterns. In *CVPR*, 2010.

[65] K. Schindler and L. van Gool. Action snippets: How many frames does human action recognition require? In *CVPR*, 2008.

[66] K. Shafique and M. Shah. "A non-iterative greedy algorithm for multi-frame point correspondence". In *ICCV*, 2003.

[67] E. Shechtman. Actions as space-time shapes. *PAMI*, 29(12):2247 –2253, 2007.

[68] B. W. Silverman. *"Density Estimation for Statistics and Data Analysis"*. Chapman and Hall, 1986.

[69] C. Stauffer. "Estimating tracking sources and sinks". In *Second IEEE Event Mining Workshop*, 2003.

[70] C. Stauffer and W.E.L. Grimson. "Learning patterns of activity using real time tracking". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):747–767, 2000.

[71] Chris Stauffer. "Learning to track objects through unobserved regions". In *Proceedings of the IEEE Workshop on Motion and Video Computing*, volume 2, pages 96–102, 2005.

[72] Chris Stauffer and Eric Grimson. Learning patterns of activity using real-time tracking. *PAMI*, 22(8):747–757, 2000.

[73] C. Thurau and V. Hlavac. Pose primitive based human action recognition in videos or still images. In *CVPR*, 2008.

[74] Kinh Tieu, G. Dalley, and W.E.L. Grimson. "Inference of Non-overlapping Camera Network Topology by Measuring Statistical Dependence". In *IEEE Int. Conf. on Computer Vision*, pages 1842–1849, 2005.

[75] A. Torralba. "Contextual influences on saliency". In *Neurobiology of Attention*, pages 586–593. Academic Press / Elsevier. 2005.

[76] Berwin A. Turlach. Bandwidth Selection in Kernel Density Estimation: A Review. In *CORE and Institut de Statistique*, pages 23–493, 1993.

[77] M. Walter, A. Psarrou, and S. Gong. "Learning prior and observation augmented density models for behaviour recognition". In *British Machine Vision Conference*, 1999.

[78] Xiaogang Wang, Xiaoxu Ma, and Eric Grimson. Unsupervised activity perception by hierarchical bayesian models. In *CVPR*, 2007.

[79] Xiaogang Wang, Kinh Tieu, and Eric Grimson. Learning semantic scene models by trajectory analysis. In *ECCV*, 2006.

[80] Yang Wang, Payam Sabzmeydani, and Greg Mori. Semi-latent dirichlet allocation: A hierarchical model for human action recognition. In *In ICCV Workshop on Human Motion Understanding, Modeling, Capture and Animation*, 2007.

[81] D. Weinland. A survey of vision-based methods for action representation, segmentation and recognition. *CVIU*, 2010.

[82] T. Xiang and S.G. Gong. Video behaviour profiling and abnormality detection without manual labelling. In *ICCV*, 2005.

[83] Y. Yacoob and M.J. Black. Parameterized modeling and recognition of activities. In *ICCV*, 1998.

[84] Yang Yang, Jingen Liu, and Mubarak Shah. Video scene understanding using multi-scale analysis. In *ICCV*, 2009.

[85] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 2006.

[86] A. Yilmaz and M. Shah. Recognizing human actions in videos acquired by uncalibrated moving cameras. In *ICCV*, 2005.

[87] Alper Yilmaz and Mubarak Shah. Actions sketch: a novel action representation. In *CVPR*, 2005.

[88] L. Zelnik-Manor and M. Irani. Event-based analysis of video. In *CVPR*, 2001.

[89] H. Zhong et al, J.B. Shi, and M. Visontai. Detecting unusual activity in video. In *CVPR*, 2004.