Title:
Recognizing Human Group Activities in Video through Mining Optimal Features

Abstract approved: ____________________________

Sinisa Todorovic

Given a video, we would like to recognize group activities, localize video parts where these activities occur, and detect actors involved in them. To this end, we propose a novel, mid-level feature, called control point, for representing group activities. The control points are aimed at summarizing visual cues, lifting from the noisy low-level features, and jointly providing visual evidence of actors and their group activity to higher-level inference algorithms. We formulate a generative model, called chains model, to organize a huge number of video features in an ensemble of chains of control points, representing a group activity. The chains may have arbitrary length, ideally, starting and ending at the beginning and end of the time interval occupied by the activity. We derive an efficient MAP inference, which is a new, EM-like algorithm that iterates two steps: warps the chains of control points to their expected locations so they can better summarize visual cues, and then maximizes their posterior probability. Our
evaluation on benchmark UT-Human Interaction and Collective Activities datasets demonstrates that we outperform the state of the art with reasonable running times.
Recognizing Human Group Activities in Video through Mining Optimal Features

by

Mohamed R. Amer

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented May 23, 2011
Commencement June 2011
Master of Science thesis of Mohamed R. Amer presented on May 23, 2011.

APPROVED:

__________________________
Major Professor, representing Electrical and Computer Engineering

__________________________
Director of the School of Electrical Engineering and Computer Science

__________________________
Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

__________________________
Mohamed R. Amer, Author
ACKNOWLEDGEMENTS

I would like to thank everyone who has helped me on my project, specially my advisor Sinisa Todorovic. Also, I would like to thank my committee: Alan Fern, Bechir Hamdaoui, Huaping Liu, and Guenter Schneider. I would like to acknowledge the support of the NSF for funding this research under grant IIS-1018490.
TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>2 Literature Review</td>
<td>4</td>
</tr>
<tr>
<td>2.1 Features</td>
<td>4</td>
</tr>
<tr>
<td>2.1.1 Low Level Features</td>
<td>5</td>
</tr>
<tr>
<td>2.1.2 Mid-Level Features</td>
<td>10</td>
</tr>
<tr>
<td>2.2 Activity Recognition</td>
<td>12</td>
</tr>
<tr>
<td>3 Overview of Our Approach</td>
<td>18</td>
</tr>
<tr>
<td>3.1 Features</td>
<td>18</td>
</tr>
<tr>
<td>3.2 Activity Recognition</td>
<td>20</td>
</tr>
<tr>
<td>3.3 Comparison</td>
<td>20</td>
</tr>
<tr>
<td>4 Control Points and Their Descriptors</td>
<td>23</td>
</tr>
<tr>
<td>4.1 Extracting Bounding Boxes</td>
<td>23</td>
</tr>
<tr>
<td>4.2 Context Descriptor</td>
<td>24</td>
</tr>
<tr>
<td>5 The Chains Model</td>
<td>27</td>
</tr>
<tr>
<td>6 Inference</td>
<td>29</td>
</tr>
<tr>
<td>6.1 The MAP Inference</td>
<td>29</td>
</tr>
<tr>
<td>6.2 Extracting Features</td>
<td>31</td>
</tr>
<tr>
<td>7 Learning</td>
<td>37</td>
</tr>
<tr>
<td>8 Results</td>
<td>39</td>
</tr>
<tr>
<td>8.1 Quantitative Results</td>
<td>41</td>
</tr>
<tr>
<td>8.2 Qualitative Results</td>
<td>43</td>
</tr>
<tr>
<td>9 Conclusion</td>
<td>49</td>
</tr>
<tr>
<td>Bibliography</td>
<td>50</td>
</tr>
<tr>
<td>------------------------</td>
<td>----</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Optical Flow</td>
<td>6</td>
</tr>
<tr>
<td>2.2 Trajectories</td>
<td>7</td>
</tr>
<tr>
<td>2.3 Foreground Pixels</td>
<td>8</td>
</tr>
<tr>
<td>2.4 Spatio-Temporal Interest Points</td>
<td>8</td>
</tr>
<tr>
<td>2.5 Histogram of Oriented Gradients</td>
<td>9</td>
</tr>
<tr>
<td>2.6 Motion History Image</td>
<td>10</td>
</tr>
<tr>
<td>2.7 Accumulated Poses</td>
<td>11</td>
</tr>
<tr>
<td>2.8 Sequencing Code Map</td>
<td>12</td>
</tr>
<tr>
<td>3.1 Control Point</td>
<td>19</td>
</tr>
<tr>
<td>3.2 Block Diagram</td>
<td>21</td>
</tr>
<tr>
<td>4.1 Descriptor</td>
<td>26</td>
</tr>
<tr>
<td>6.1 Chain Model</td>
<td>31</td>
</tr>
<tr>
<td>8.1 Confusion Matrix for Collective Activity Dataset</td>
<td>44</td>
</tr>
<tr>
<td>8.2 Confusion Matrix for UT-Interaction Dataset</td>
<td>44</td>
</tr>
<tr>
<td>8.3 Accuracy ROC</td>
<td>45</td>
</tr>
<tr>
<td>8.4 Detection ROC</td>
<td>46</td>
</tr>
<tr>
<td>8.5 Qualitative Results</td>
<td>47</td>
</tr>
<tr>
<td>8.6 Qualitative Results</td>
<td>47</td>
</tr>
<tr>
<td>8.7 Qualitative Results: Failure Example</td>
<td>48</td>
</tr>
<tr>
<td>Table</td>
<td>Page</td>
</tr>
<tr>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>8.1 Variants Comparison</td>
<td>41</td>
</tr>
<tr>
<td>8.2 Average Classification Accuracy</td>
<td>43</td>
</tr>
<tr>
<td>8.3 Average Classification Accuracy</td>
<td>43</td>
</tr>
</tbody>
</table>
DEDICATION

To my family and friends.
Chapter 1 – Introduction

We study recognition of human interactions and group activities in videos. Given a video, we would like to recognize group activities, localize video parts where these activities occur, and detect actors involved in them. This is challenging because human interactions and group activities are typically characterized by huge variations in structural, motion, and appearance properties. For example, running in a group can be performed by a varying number of people, in diverse, dynamically changing spatial configurations, in which the runners may partially occlude one another. Also, each runner’s speed and sportswear may be different from those of the other runners in the group. In the face of these challenges, we seek answers to the following questions. What features should be extracted from the video to facilitate accurate inference? How to model group-activity classes, so that their representations allow for efficient and robust learning and inference of group activities?

Regarding activity features, we depart from the common practice in activity recognition to extract a set of (arguably good) features, and then conduct inference on this fixed set, without ever revisiting the video. We believe that the complexity of group activities requires a more symbiotic interaction between high-level inference algorithms and low-level feature extractors than seen in existing work. We here specify this symbiosis as an iteration in which inference guides low-level algorithms in their search for the optimal features, and, conversely, adaptive feature extraction facilitates reasoning about many
competing hypothesis. Instrumental to this interaction is a new concept that we introduce here, referred to as control point. The control points are aimed at summarizing visual cues, lifting from the noisy low-level features, and jointly providing visual evidence of actors and their group activity to higher-level inference algorithms. The control points can be interpreted as spotlights that shed light on certain space-time areas in the video. If an activity occurs, it has to be in the spotlight of many control points so that they can collectively provide a strong support of this hypothesis. The main difference from other common features which summarize their neighborhood (e.g., SIFT, shape context) is that the control points are movable, laying on a deformable space-time grid throughout the video. Top-down feedback from inference algorithms warps the grid so the control points could better summarize relevant visual information for activity recognition.

Regarding modeling, we propose to extend the chains model — a generative model, recently used for detecting a person’s hand in a still image [23]. The chains model has been demonstrated as suitable for handling non-rigid configurations of a relatively small number of features in the image. This motivates us to seek its generalization to videos, so it can address the huge variability of group activities. Intuitively, our chains model will be aimed at handling a large number of video features by organizing them in an ensemble of chains of control points, representing a group activity. Our chains will have arbitrary length, ideally, starting and ending at the beginning and end of the time interval occupied by the activity. Since the start and end frames of the activity are typically unknown, we will relax the restrictive assumption of the original chains model that the feature chains must start from some known reference point. More importantly, while the
original chains model is designed to probabilistically generate low-level image features (observables), our chains model will generate video features by creating chains of the mid-level features, i.e., control points between the start and end frames of the activity. For such a more general model, we will derive a new inference algorithm which will be able to efficiently interact top-down with the control points.
Chapter 2 – Literature Review

There is a large volume of literature that discusses activity recognition. Reviewing all that work is beyond our scope. We focus on the most related approaches. In this chapter we will give an overview of prior work on activity recognition and the different features and activity models. We will also motivate the use of our proposed features and activity model.

There are different settings for activity recognition. Some approaches focus on exploiting contextual information in a video including: scenes and objects [34], objects and objects [44, 11], human-object interaction [12], objects and attributes [60], human poses and actions [63]. Other approaches focus on exploiting contextual information in an image including: object-action context [12, 16, 64, 65, 6, 27], and human-object interaction [23, 24]. Our main focus is to detect group activities in videos. In the following sections we describe the different features and activity models used to solve this problem.

2.1 Features

There are different types of features that are used for activity recognition. We divide them into two categories: low-level, and mid-level. Low-level features are computed directly from pixels. They include: Optical Flow, Trajectories, Foreground Pixels, Spatio-
Temporal Interest Points, and Histogram of Oriented Gradients. Mid-level features synthesize low-level features into richer features for activity recognition. They include: Motion History Image, Accumulated Poses, Sequence Code Map. In the following subsections we will discuss each of these features in greater detail.

2.1.1 Low Level Features

The most common low level feature include: Optical flow, Trajectories, Foreground Pixels, Spatio-Temporal Interest Points, and Histogram of Oriented Gradients.

**Optical Flow** is a pattern of pixel motion caused by a relative motion between the camera and the scene. It is an approximation of motion that happens in two successive frames in case of 2D, and $n$ successive frames in case of 3D as described in [31, 21]. It helps detect relevant areas in the video where there is motion. Figure 2.1 shows a 2D representation of optical flow using [31]. The equation of motion is described as $\nabla I \cdot \vec{v} = -I_t$ where $\vec{v}$ is the flow direction, $\nabla I$ is the gradient of the image, and $I_t$ is the derivative of the image with respect to time.

Optical Flow is considered one of the most useful features in activity recognition. However, it is a noisy feature. It captures first-order motion and often fails when the motion has sudden changes. There are multiple sources of noise, for example blur and jitter in the frames, also background texture. They give rise to noisy components. Different techniques have been proposed to remove that noise, but at the price of increased computation. The latest version of Optical Flow is described in [56] where they provide a formulation with median filtering heuristic, that gives more robust Optical Flow.
Figure 2.1: 2D optical flow on two frames from UT-Interaction dataset [48] using [56]. The cyan circles shows the areas where motion happens between the two frames.

**Trajectories** of points are a contiguous sequence of spatial locations of tracked points in the video (e.g. KLT tracker [51]). They help model spatio-temporal relationships. However, they can be noisy and be affected by rotation and scale changes. Also since they rely on point features, the noise of the point features affects the tracker. Feature trackers often assume a constant appearance of image patches over time and may fail when this appearance changes (e.g. when two objects in the video overlap [54, 2], or when features appear or disappear in the domain). Figure 2.2 shows an example of the output of KLT tracker [51].

**Foreground Pixels** identifies moving objects from the portion of a video frame that differs significantly from a background model. There are different approaches for background subtraction including: frame difference, running average, and fitting one or more Gaussian distributions [9, 26, 55, 13]. Once the background is modeled, it is easy to extract foreground objects. The following constraints limit this feature: the camera has to be static and a background model is specific for a certain video and cannot be reused for other videos. Also, if the background contains dynamically changing objects then the learned model is noisy.
There is a trade off between accuracy, time, and memory for these different approaches. Figure 2.3 shows an example of extracted foreground objects using background subtraction with the running average approach.

**Spatio-Temporal Interest Points** are computed using Gaussians, and derivative of Gaussians of pixels along the spatial and time axes. They indicate video areas where motion happens. Common interest points are STIP [28] and 3D SIFT [50]. To extract the value of STIP we compute a $3 \times 3$ matrix composed of pixel gradients along $x, y, t$ axes at each pixel location the matrix is averaged with Gaussian kernel. The pixel locations in the video characterized by large eigenvalues of the second moment matrix are declared as STIP points. 3D SIFT is computed by the difference of Gaussians at different scales. The problem with these features is that they do not capture any global information in the scene. Figure 2.4 shows an example of these two interest points.
Figure 2.3: Foreground pixels obtained by a running average approach on a frame from UT-Interaction dataset [48].

Figure 2.4: The middle image shows STIP features extracted from a video sequence, and to the right a 3D SIFT feature computed at the marked location showed in the video frame from UT-Interaction dataset [48] on the left.
Figure 2.5: 2D HOG computed on a single frame from UT-Interaction dataset [48] we can see that the computed HOG is very informative and could be used for object and pose detection. The cyan boxes show the location of the two persons.

Histogram of Oriented Gradients (HOG) is a count of occurrences of specific gradient orientation in particular portions of an image or a video called cells. There are 2D [10] and 3D [5] HOG which use bins in space and space and time, respectively. To compute HOG, we first compute the pixel gradient, and then estimate gradient histograms over cells. Each pixel within the cell gives a weighted vote for an orientation-based histogram. The cells themselves can either be rectangular or radial in shape, and the histogram bins are evenly spread over the cell.

The main advantage of HOG is its stability. Also, it is more global than STIP or SIFT and it is very useful to detect human and objects (e.g. Cars, and buses) [14, 5]. Figure 2.5 shows the result of applying 2D HOG to a frame from a video sequence.

After our brief summary of the low level features. In the following subsection we describe mid-level features.
2.1.2 Mid-Level Features

To bridge the gap between the low-level features and the higher level semantic interpretation, a mid-level description is needed. There are multiple ways for computing mid-level features:

**Motion History Image** [3] uses poses obtained from a low level feature extractor, specifically foreground objects from a background subtraction algorithm. Then it accumulates them in time by giving lower weights to old frames and gradually increase the weight for new frames. The obtained image is a 2D template of a certain action.

The main disadvantage of this feature is that it requires the presence of only one person in the scene with a uniform background, since it uses background subtraction. Also, it can not handle complex activity, since it overwrites the motion history. Figure 2.6 shows an example of Motion History Image.
Accumulated Poses [15] is a more advanced version of the Motion History Image. It accumulates poses in time to construct a 3D volume instead of a 2D template. The poses are extracted using the foreground pixels. This 3D volume is then used for matching. This feature also handles only a single person with a uniform background. Figure 2.7 shows an example of Accumulated Poses.

Sequencing Code Map [32] uses trajectories as a low level feature, where each trajectory is broken up into fragments and then quantized into one direction depending on the number of bins in the Sequencing Code Map. This feature is computationally efficient and lifts up the level of trajectories to be more discriminative. It does not keep the spatio-temporal relationship of the video since the trajectories are presented in 2D angular map. Its main disadvantages are the accumulated noise from the feature extraction and tracking, and the quantization error and it is a heuristic. Figure 2.8 shows an example of a SCM.
After describing the different features and motivating the use of our proposed feature. In the following section we describe the different activity models that use these features to reason about an activity.

2.2 Activity Recognition

Approaches for activity recognition use as an input low-level or mid-level features. There are different ways to discriminate between activity classes. In this section we present some of these approaches including: Bag of Words, Generative Graphical Models, Discriminative Graphical Models, Context Free Grammars, and Logic Based Approaches.

**Bag of Words (BOW)** [53, 8] is a model that was first used for classification of documents. It was adopted in the computer vision community for classification of images and videos. It allows a dictionary-based modeling, where each video is treated as a document which contains some words from the dictionary. BOW estimates the histogram of
occurrences of the words, which could be low-level, such as STIP or SIFT. For a certain activity a new video is classified by estimating the histogram distance from the BoW model. For example, a Support Vector Machine or a Naive Bayes classifiers could be used to classify BoW.

The main advantages of the method includes: its simplicity, computation efficiency, and its intrinsic invariance. Also it is robust to background clutter and produces good categorization accuracy even without exploiting geometric information. As for its disadvantages, classification using BoW is global and does not account for spatio-temporal relations of words, and it is good for classifying periodic activities but not complex ones. It is a good baseline to compare to.

**Generative Graphical Models** are graphical models where the state space is considered to be a number of states and the time is modeled as a sequence of probabilistic jumps. There are single layered and hierarchical generative models. Both consist of either Hidden Markov Models (HMM) [7] and Dynamic Bayesian Networks (DBN) [42]. HMMs and DBNs are the common used generative graphical models in activity recognition. In a single layered generative model case, an activity is represented in terms of a set of hidden states where a human is assumed to be in one state at each time frame, and each state generates an observation. Once transition and observation probabilities are trained for the models, activities are commonly recognized by solving the evaluation problem. The evaluation problem is the problem of calculating the probability of a given sequence generated by a particular state-model. There are different variants of the HMM that are used in activity recognition such as [58, 20, 41, 62]. More complex models were proposed by [35] to model human-human interaction by using a coupled
HMMs (CHMMs), where each person has HMM model and interactions between persons are modeled by coupling between the different HMMs. A DBN is an extension of an HMM, composed of multiple conditionally independent hidden nodes that generate observations at each time frame directly or indirectly. DBN is considered more powerful in encoding complex conditional dependencies than HMM.

Generative graphical models provide more flexibility than the BOW approach for the activity recognition system because it models temporal dependencies. However, they tend to require a large number of training videos proportional to the complexity of the activity. A limitation of single layered generative models is that they cannot recognize activities with complex temporal structures, such as an activity composed of concurrent subevents. This problem was solved by introducing hierarchical HMMs (LHMMs) [40] where the bottom layer HMMs recognize atomic actions of a single person by matching the models with the sequence of feature vectors extracted from videos. The upper layer HMMs treat recognized atomic actions as observations generated by the upper layer HMMs. This hierarchical addition increases the model complexity.

**Discriminative Graphical Models** such as Conditional Random Field (CRF) [43, 39] is a Markov random field that is trained discriminatively. They condition on the entire observation sequence, which avoids the need for independence assumptions between observations. CRF models the conditional probability of the label sequence rather than the joint probability of both the labels and observations, as done by HMMs. It does not model prior distribution over observed variables, which makes it possible to include complicated features. The probability of a variable configuration in an undirected graph is proportional to the product of a series of non-negative potential functions, with one
potential function for each clique of the graph, where a clique is a set of nodes that are fully connected to each other. The Hammersley-Clifford Theorem proves that such a composition of clique potentials produces a distribution that obeys the conditional independence assumptions encoded by the graph structure [19]. The probability of a variable in an undirected graph is $P(V) = \frac{1}{Z} \prod_{c \in \text{cliques}(V)} \kappa(c)$ where $Z$ is the normalization term, $\kappa(C) = \exp(w \cdot f)$ is the potential function, which could be unary, pairwise, or higher order potential which model relationships between nodes of the graph. CRF models are commonly trained by maximizing the conditional likelihood of a labeled training set to estimate the weight vector $w$.

Discriminative graphical models can model complex relationships between nodes with pairwise and higher order potentials. It performs as well as or better than generative graphical models even when the model features do not violate the independence assumptions of the HMM. However, it is very complex and computationally inefficient.

**Context-Free Grammars (CFG)[18, 33]** are used to model a composite human activities and interactions. Human activities are represented as a set of production rules generating a string of atomic actions. It is a hierarchical approach where the lower-levels are either HMMs or DBNs, and the higher levels are CFG. They only need an enumerated list of lower-level events and a set of rules that define the higher level, then once obtained there are multiple algorithms such as the EarleyStolcke algorithm used in [22] that parse them. CFGs and stochastic CFGs (SCFGs) have been used by previous researchers to recognize high-level activities [36, 47].

The main advantage of CFGs is the flexibility of definition of rules. [17] represented the rules in the form of AND-OR graphs where CFGs consist of a string of low-level
activities, the temporal ordering of atomic-level activities has to be sequential. The assumption that all observations are parsed makes it difficult for the system when an unknown observation interferes with the system. To avoid this problem, [25] developed an algorithm to learn, with no supervision, the grammar rules from observations.

**Logic Based Approaches (LBA)** [1] is a description-based approaches model a human activity as an occurrence of its low-level event that satisfies certain relations, which rely on common sense in interpreting the mid-level actions. LBA are hierarchical approaches and they are able to handle activities with concurrent structures. A time interval is usually associated with an occurring low-level event to specify necessary temporal relationships among low-level event. Allens temporal predicates [1] have been adopted for these approaches [52, 37, 46, 17, 4], to specify relationships between time intervals with operators including: before, meets, overlaps, during, starts, finishes, and equals. Some approaches describe all activities in terms of events or scene structures. The approach described in [17] aims to recognize atomic-level actions more reliably by modeling causality among the actions. A tree structured AND-OR graph has been used to represent a storyline of a sports game and a label is assigned to each action that fits the storyline. Also, Markov-Logic Networks [45] is another approach that incorporates a probabilistic model in addition to the logical model. Each logical rule is presented as a first order logic formula, which gets provided a weight – also referred to as a belief.

LBA approaches can incorporate human knowledge into the systems and require less training data. Also they can represent and recognize human activities with complex temporal structures. The main disadvantage of the LBA is that the recognition is performed by developing an approximation algorithm to solve the constraint satisfaction
problem which is NP-hard.
Chapter 3 – Overview of Our Approach

After describing the different low-level and mid-level features used for activity recognition in (section 2.1), we motivate the use of our proposed mid-level feature, the control point.

3.1 Features

Control Points are pose based descriptors that use a HOG based object detector [14] as a low-level feature. Control points consist of a description of the surrounding space of the feature, from bounding boxes locations in the neighborhood of the point, and within each of those bounding boxes we have the descriptor of it which consists of poses of persons performing an activity, so we construct a histogram of poses in a spatio-temporal manner. The reason why we use such feature is that it bridges the gap between higher level reasoning and low-level features, which allows us to describe and detect complex activities.

It is more robust and rich in its description than the previously mentioned features, since it is computed from a reliable low-level feature (HOG). Also our claim that human activities are best described in form of poses, and Control Point collects poses in its neighborhood. It is robust against transient occlusions, because the control points are not affected by occlusion of a particular actor. This is important because mutual occlusions
of actors are frequent in group activities. Figure 3.1 shows an example of a Control Point.

After describing the different models used for activity recognition in (section 2.2) we motivate the use of our proposed model, the Chains Model, and give an overview of the approach.
3.2 Activity Recognition

**Chains Model** is a discriminative graphical model that localize activities in a principled manner using the MAP inference on Markovian Chain Model proposed by [23]. We seek to infer the start and end of the activity, as well as detect all participants. Our approach consists of two main steps, as illustrated in Figure 3.2. The control points are placed on a deformable grid in the video. The goal is to jointly collect visual evidence if any activity of interest is present. The MAP inference of the chains model identifies chains of the control points. The chains are adaptively warped to better summarize visual information for: recognizing and localizing activities, and detecting their participants. We derive an efficient MAP inference, which is a new, EM-like algorithm that iterates two steps: warps the chains of control points to their expected locations so they can better summarize visual cues, and then maximizes their posterior probability.

Our approach is able to detect and localize multiple co-occurring activities, and their respective actors. Also it is data driven and robust against occlusion.

In the following section we compare our proposed features and model against prior work and show its advantage over each of them.

3.3 Comparison

Graphical models offer a powerful computational framework for representing and reasoning about many important properties of activities, including motion and appearance properties of people involved in the activity, and temporal configurations of their interactions. For example, activities has been modeled by DBNs [61], prototype trees
Our approach: The control points are placed on a deformable grid in the video. Their goal is to jointly collect visual evidence if any activity of interest is present. The MAP inference of the chains model identifies chains of the control points. The chains are adaptively warped to better summarize visual information for: recognizing and localizing activities, and detecting their participants. (a) The initial control points on a regular grid. (b) The chains found after multiple iterations and the control points grid is deformed to fit the description better. (c) The resulting activity and the actors are localized.

Most of these approaches, however, do not address group activities. They can typically handle only sanitized environments with static background, where actors are prominently featured (with few exceptions, e.g., [17, 39]). We address more difficult settings. A few methods seek to learn temporal structure of activities from data [39]), or relevant contextual relations within a group activity [27]. However, their model structure permits only a fixed number of actors, or a fixed number of primitive actions defining the group activity. We do not suffer from these limitations, but allow arbitrary number of participants in the activity, and arbitrary number of primitives. This complexity is summarized in our approach by the means of control points, and their associated context descriptors. First-order logic can also be used for representing human interactions [48]. They localize activities in the video. However, this is
done through a heuristic voting procedure and its inference is NP-hard. In our approach, we localize activities in a principled manner using the MAP inference. Spatio-temporal relations within a group activity are modeled in [6]. However, they can only classify videos, whereas we additionally seek to infer the start and end of the activity, as well as detect all participants.
Chapter 4 – Control Points and Their Descriptors

The control points are aimed at jointly providing contextual cues for activity recognition and detecting actors involved in the activity. A descriptor of each point \(i\) summarizes a relatively large space-time neighborhood around \(i\), in terms of a histogram of human poses detected in that neighborhood, as illustrated in Figure 4.1. Below, we first explain how to detect people and their poses in the video, and then we describe the context descriptor of the control points.

4.1 Extracting Bounding Boxes

Given a video, we first run the efficient people detector of [14]. The detector parameters are set such that the detector yields high recall, under the average running time of 2s per frame. The resulting detections are noisy bounding boxes, which represent our input. In the sequel, we will use the terms detections and bounding boxes interchangeably to refer to the output of the people detector. High recall is desirable for our purposes to ensure detection of all people in the video, where we expect that our higher-level algorithms will be able to discard false positives. Note that the detector also localizes characteristic parts of a detected person (e.g., legs, arms). Each detection can be characterized by a space-time pose descriptor that captures: (i) the distances and orientations of localized human parts relative to the major axis; and (ii) the mean direction of the optical flow
within the bounding box computed by the approach of [56]. Next, we use these pose descriptors to map the detections to a dictionary of codewords, where each codeword represents a characteristic human pose. The dictionary consists of \( d = 300 \) codewords for each class, learned by the K-means algorithm on training videos for each class with labeled bounding boxes around people performing various group activities, and where the bounding boxes are described by the same pose descriptor. In chapter 7) we explain how to learn the dictionary in a weekly supervised manner.

4.2 Context Descriptor

For detecting actors in the video, it is necessary to identify foreground and background bounding boxes. We use the control points to efficiently solve this problem. Specifically, we place the control points throughout the video on a deformable grid, as illustrated in Figure 3.2. The points jointly collect evidence of candidate foreground bounding boxes, as follows. For each control point \( i \), we compute a context descriptor, \( h_i \). The descriptor is a histogram of codewords detected in a spatiotemporal neighborhood of \( i \).

Most importantly, \( h_i \) accounts only for those detections that are estimated as foreground. We allow for certain deformations of the grid, such that the control points may assume optimal locations at which their \( h_i \)’s will collectively provide strong evidence about foreground.

To compute \( h_i \), we avoid enumerating exponentially many choices of figure/ground labeling of noisy people detections. Rather, we compute it as a linear function of indicator vector \( b \in \{0, 1\}^k \), where \( k \) is the total number of detections in the video. Each
element of \( b \) is set to 1 if the corresponding bounding box is estimated as foreground, or 0, otherwise. To define this linear function, for each \( i \), we specify a binary, \( d \times k \) matrix \( V_i \), where \( d \) is the size of the dictionary of human poses. \( V_i \) can be readily computed from all noisy detections around \( i \). Specifically, each element \( (V_i)_{uv} = 1 \) if \( u \)th bounding box “falls” within the neighborhood of \( i \), and the bounding box is mapped to \( u \)th codeword. Neighborhood is a space-time ellipsoid centered at \( i \), whose scale and principal axes are adaptively estimated as for standard scale-invariant space-time interest points. In our experiments, we observe ellipsoids that span between 2 to 6 frames. Given \( V_i \) and an estimate of foreground bounding boxes \( b \), the context descriptor is defined as \( h_i \triangleq V_i b \).

The control points are also characterized by their space-time locations:

\[
q_i = [x_i, y_i, t_i, \cos \varphi_i, \sin \varphi_i]^T
\]

where \( x_i \) and \( y_i \) are coordinates in the frame at time \( t_i \), and \( \varphi \) is the direction of optical flow at \( i \) computed by [56].
Figure 4.1: The context descriptor of control point $i$ summarizes a relatively large space-time neighborhood around $i$, in terms of a histogram of human poses, $h_i = V_i b$, detected in that neighborhood.
Chapter 5 – The Chains Model

The chains model is a generative model of control points observed in the video, \( \mathbb{F} = \{ F_i = (h_i, q_i) : i = 1, \ldots, n \} \). Hidden variables that govern the generative sampling of control points belonging to the activity (referred to as foreground points) include: (i) the activity’s start and end frames in the video, \( L_S \) and \( L_E \); (ii) an ordered list (chain) of foreground control points \( O = (O(1), \ldots, O(m)) \), where \( 1 \leq O(i) \leq n \); and (iii) the total number of foreground points \( M \) in the video. The joint distribution of all random variables, \( P(M, O, L_S, L_E, F) \), is specified as

\[
P(M, O, L_S, L_E|\mathbb{F})P(\mathbb{F}) = P(M, O)P(L_S|M, O, \mathbb{F}) \cdot P(L_E|M, O, \mathbb{F}) \prod_{i=2}^{m-1} P(F_{O(i+1)}|F_{O(i)}) \prod_{i \in \mathbb{F}\setminus O} P_G(F_i). \tag{5.1}
\]

Below, we explain each distribution. We assume that \( P(M, O) = P(M)P(O) \). \( M \) has a Poisson distribution, \( P(M = m) \triangleq \frac{\lambda_M^m}{m!}e^{-\lambda_M} \), governing that the expected number of control points in the activity chains is \( \lambda_M \). This prevents inference of unrealistically short and very long chains. The distribution of point orderings \( P(O) \) is uniform if for all point pairs \( (O(i), O(i+1)) \) the frame of point \( O(i) \) does not happen after the frame of \( O(i+1) \), and \( P(O) = 0 \), otherwise. This prevents the chains to move backwards in time. Next, \( P(L_S|M, O, \mathbb{F}) \) and \( P(L_E|M, O, \mathbb{F}) \) are defined as the conditional probability that the chain \( O \) starts and ends at points \( 1 \leq O(1) \leq n \) and \( 1 \leq O(m) \leq n \). Specifically, let
\(t_{O(1)}\) and \(t_{O(m)}\) be the frame number (time) of the start and end of the chain. Then, we specify that the probability of the start exponentially decreases as \(t_{O(1)}\) becomes larger, 
\[ P(L_S|M,O,\mathbb{F}) \triangleq \lambda_S \exp(-\lambda_S \frac{t_{O(1)}}{t_{\text{video}}}) \]
Also, we define that the probability of the end exponentially decreases as \(t_{O(m)}\) happens before the end of the video, 
\[ P(L_E|M,O,\mathbb{F}) \triangleq \lambda_E \exp(-\lambda_E \frac{(t_{\text{video}} - t_{O(m)})}{t_{\text{video}}}) \]
Next, \(P(F_{O(i+1)}|F_{O(i)})\) denotes the transition probabilities between two consecutive control points \(O(i)\) and \(O(i+1)\) in the chain. Since, all legal orderings have a uniform distribution, in the following, we will simplify notation and write \(P(F_j|F_i)\) to denote the probability of transitioning from foreground point \(i\) to \(j\). Finally, \(P_G(F_i)\) is the probability that control point \(i\) falls on the background clutter (i.e., represents a ground point). \(P_G\) is used to generate all control points that lie outside the chain \(O\). We assume that \(P_G(F_i)\) is uniform, since our videos have huge variability.

From above, the set of parameters of the chains model is \(\Theta = \{\lambda_S, \lambda_E, \lambda_M, P(F_j|F_i)\}\).
Chapter 6 – Inference

In inference, we recognize the activity class, identify the start and end frames of the activity, and detect foreground bounding boxes of people who perform the activity. In the following two sections, we first describe our MAP inference algorithm given video features, and then explain how to mine relevant features from the video.

6.1 The MAP Inference

The MAP inference is illustrated in Figure 6.1. For inference, we compute the marginal posterior distribution of $L_s$ and $L_e$ over all possible chains, $P(L_s, L_e | F)$. From (5.1), we have

$$P(L_s, L_e | F) \propto \sum_{M,O} P(M, O, L_s, L_e, F),$$

where $P(M, O, L_s, L_e, F)$ is the joint probability of the activity class, control points, and features.

$$\propto \sum_{M,O} P(M)P(L_s, L_e | M, O, F) \prod_{i,j} P(F_j | F_i).$$

Similar to Markov chain model we compute (6.1) by organizing all transition probabilities in a $n \times n$ matrix $X = [P(F_j | F_i)]$. Also, we organize all conditional probabilities $P(L_s, L_e | M, O, F)$, for each control point, into $n$-dimensional vectors $\omega = [P(L_s=1 | \cdot), ..., P(L_s=n | \cdot)]^T$, and $\gamma = [P(L_e=1 | \cdot), ..., P(L_e=n | \cdot)]^T$. Note that each row $i$ of $X$ contains the probabilities of transitioning, in one step, from the corresponding point $i$ to other points $j$ in the video. The sum of these probabilities along each row of $X$ must be 1, $X1_n = 1_n$, where $1_n$ is the $n$-dimensional vector with all elements equal to 1. From the Markov chain theory, we have that the probability of transitioning...
from $i$ to $j$ in $m$ steps is equal to $(X^m)_{ij}$. It is straightforward to show that the marginal posterior in (6.1) can be computed as

$$P(L_S, L_E | \mathcal{F}) \propto \omega^T \left[ \sum_{m=1}^{m_{\text{max}}} P(M = m) X^m \right] \gamma,$$

(6.2)

where $m_{\text{max}}$ is the maximum chain length equal to the number of the control points along the time axis. Note that the summation in (6.2) means that the marginal posterior is computed by summing over all paths of length $m$ – not only over simple paths. This approximation has a justifiable complexity vs. accuracy trade off in our experiments, because computing $m$ shortest simple paths is very expensive. Assuming that $m_{\text{max}}$ is a very large number, i.e., that the activity spans a large number of video frames, $m_{\text{max}} \to \infty$, the bracketed term in (6.2) can efficiently be approximated as

$$\sum_{m=1}^{m_{\text{max}}} P(M = m) X^m \approx e^{-\lambda_M} \sum_{m=0}^{\infty} \frac{\lambda_M^m}{m!} X^m = e^{-\lambda_M} e^{\lambda_M X}.$$

(6.3)

By plugging in the parameters of different activity models, $\Theta_a, a = 1, 2, ..., $ from (6.2) and (6.3), we have that the MAP inference recognizes activity class $a$ as

$$a = \arg\max_{\Theta_a, a=1,2,...} \omega_a^T \cdot \exp(\lambda_{M,a} \cdot X) \cdot \gamma_a.$$

(6.4)

Also, from (6.2) and (6.3), the optimal start and end of the recognized activity $a$ can be simply estimated as indices $1 \leq i \leq n$ and $1 \leq j \leq n$ of the maximum product: $\arg\max_{i,j} \omega_{i:a} \cdot \gamma_{j:a} \cdot \left[ \exp(\lambda_{M,a} \cdot X) \right]_{ij}$. When multiple activity instances need to be detected in a video, we simply choose the second, third, etc. best pair $(i, j)$ for their start and end frames.
Figure 6.1: MAP inference finds the start (S) and end (E) control points, by estimating the most likely Markov chains of the video’s control points. The control points are placed on a deformable grid which is warped so the points can extract relevant visual cues for the MAP inference. The optimal space-time layout of the control point is used to estimate the matrix $X$ of transition probabilities.

From (6.4), the MAP inference requires estimates of transition probabilities $X$ over a given set of observables, i.e., the control points. In the next section, we explain how to estimate the optimal space-time layout of the control points, and use this information to estimate $X$ for our inference.

6.2 Extracting Features

This section presents our approach to mining relevant video features. Our features are control points, described in chapter 4. Given a set of control points in the video, we estimate their transition probabilities $X$, and plug them in (6.4) for conducting the MAP inference.

To this end, we initially place a number of control points on a grid in the video, as illustrated in Figure 6.1. The initial point locations form a regular grid, where 16
points are placed in every fifth frame. The grid is deformable along the spatial and temporal axes. Our goal is to estimate the optimal locations of the points, so their context descriptors jointly provide the most relevant visual cues for the MAP inference. Note that, at different locations, the control points may include in their context descriptors different sets of bounding boxes (i.e., people detections). Thus, when the points are in the optimal layout they should all jointly include only foreground bounding boxes, i.e., correctly provide evidence about people involved in the activity.

Finding the optimal point locations and estimating their transition probabilities $X$ is specified as an EM-like algorithm. In the E-step, the grid is warped to its expected layout. Then, in M-step, each $X_{ij}$ is estimated as confidence that points $i$ and $j$ are best matches, i.e., that $j$ is the best one-step transition from $i$ along the activity chains. The E-step and M-step are iterated until convergence. Below, we first explain the E-step, and then present the M-step.

In the E-step, given a current estimate of $X$, the expected location of each point $q_i$ can be found as \( \hat{q}_i = \sum_j q_j X_{ij} \), where $q_i$ is defined in chapter 4. Let $Q$ denote an $n \times 5$ matrix whose rows are $Q_i \triangleq q_i^T$, $i = 1, ..., n$. Then, the expected locations of all points can be expressed as $\hat{Q} = XQ$.

In the M-step, we estimate $X$. To this end, we extend the matching algorithm of [29]. They match two images, so matching is invariant to locally affine transformation. However, they treat descriptors of image features as fixed vectors. They do not account that, along with locations, the descriptors also change under affine transformations. Instead, in our approach, as the points change positions, their context descriptors “cover” different sets of bounding boxes in the video, and thus the descriptors change, as desired.
Below, we first briefly review [29], for completeness, and then point out our novelty.

Using the definition of $X$, given in chapter 6, we formulate the M-step as the standard linear assignment problem:

$$
\text{minimize } \sum_{i=1}^{n} \sum_{j=1}^{n} X_{ij} C_{ij} \\
\text{subject to } \forall i, j, \ X_{ij} \geq 0, \ X_{1n} = 1_n
$$

(6.5)

where $C_{ij}$ is dissimilarity between the descriptors of points $i$ and $j$. As explained in chapter 4, the descriptor of $i$ is estimated as $h_i = V_i b$, where $b$ is the indicator vector of foreground bounding boxes, $b \in \{0, 1\}^k$, and $V_i$ and $V_j$ are $d \times k$ matrices that count bounding boxes in neighborhoods of $i$ and $j$. Thus, we define dissimilarities $C_{ij}$ as

$$
C_{ij} \triangleq \| (V_i - V_j) b \|_\Sigma = [(V_i - V_j) b]^T \Sigma^{-1} (V_i - V_j) b,
$$

(6.6)

where $\| . \|_\Sigma$ is the Mahalanobis distance, parameterized by $d \times d$ matrix $\Sigma$. $\Sigma$ is learned for each activity class, and encodes the activity-specific covariance of human poses. To ensure that matches cannot occur backwards in time, $C_{ij}$ is set to infinity (large number) whenever $i$ occurs after $j$ in the video. We organize all $C_{ij}$ in $n \times n$ matrix $C_b$, where $b$ in subscript indicates that each $C_{ij}$ is a function of $b$.

In addition to matching, i.e., estimating $X$, we also wish to identify foreground bounding boxes, i.e., estimate $b$. Thus, from (6.5), and $\sum_{i,j} X_{ij} C_{ij} = \text{tr}\{C_b^T X\}$, our
linear assignment problem extends that of [29] as:

\[
\begin{align*}
\text{minimize} & \quad \text{tr}\{C_b^T X\} \\
\text{subject to} & \quad X \succeq 0, \quad X 1_n = 1_n, \quad b \succeq 0, \quad \|b\|_2^2 = 1.
\end{align*}
\] (6.7)

where minimization is done with respect to \(X\) and \(b\).

We would also like to constrain neighboring points to preserve their neighbor relations after reallocating to their expected positions \(\hat{Q}\) in the E-step. To this end, we define the \(n \times n\) adjacency matrix \(W\), where each \(W_{ij}\) is inversely proportional to the Euclidean distance between \(q_i\) and \(q_j\), and \(W_{ii} = 0\). Then, our goal is to minimize expected distances of each point to its neighbors, defined as L1 norm \(\|\hat{Q} - W \hat{Q}\|_1\). This constraint is added to (6.5), yielding

\[
\begin{align*}
\text{minimize} & \quad \text{tr}\{C_b^T X\} + \alpha \| (I - W) X Q \|_1 \\
\text{subject to} & \quad X \succeq 0, \quad X 1_n = 1_n, \quad b \succeq 0, \quad \|b\|_2^2 = 1.
\end{align*}
\] (6.8)

We also want to constraint the absolute distance from the initial fixed position not only these relative to neighbors. This is done by adding this constraint \(\| (I - X) Q \|_1\) to 6.8, yielding

\[
\begin{align*}
\text{minimize} & \quad \text{tr}\{C_b^T X\} + \alpha \| (I - W) X Q \|_1 + \beta \| (I - X) Q \|_1 \\
\text{subject to} & \quad X \succeq 0, \quad X 1_n = 1_n, \quad b \succeq 0, \quad \|b\|_2^2 = 1.
\end{align*}
\] (6.9)

where optimal \(\alpha = 0.3\) and \(\beta = 0.9\) are empirically estimated.

As a standard step in optimization [29], the M-step in (6.9) can be exactly linearized
by introducing an auxiliary $n \times 5$ matrix $Z$ to replace the L1-norm constraint as

$$\begin{align*}
\text{minimize} & \quad \text{tr}\{C_b^T X\} + \alpha 1_n^T Z 1_5 + \beta 1_n^T Y 1_5 \\
\text{subject to} & \quad X \geq 0, \; X 1_n = 1_n, \; b \geq 0, \; \|b\|_2^2 = 1, \; Z \geq 0, \; Y \geq 0 \\
& \quad (I-W)XQ \leq Z, \; (I-W)XQ \geq -Z \\
& \quad (I-X)Q \leq Y, \; (I-X)Q \geq -Y \end{align*}$$

where minimization is over $X$, $b$, $Z$, and $Y$. From (6.10), it follows that our M-step can be efficiently solved as the linear program (LP). The number of variables in our LP model (6.10) is proportional to $n^2$. LP with tens of thousands of variables and thousands of constraints can be solved within seconds on a standard PC using state-of-the-art solvers, such as CVX. Typically, for $10^3$ control points, each our LP iteration takes less than 5s on a 2.66GHz, 3.49GB RAM PC.

To solve (6.10), we first compute the LP program to find $X$, $Z$, and $Y$, using the estimate of $b$ from the previous EM iteration. Then, the obtained $X$ is plugged in (6.10) to find $b$, which amounts to solving the following convex problem:

$$\begin{align*}
\text{minimize} & \quad \sum_{i=1}^n \sum_{j=1}^n X_{ij}(V_i - V_j)b\Sigma \\
\text{subject to} & \quad b \geq 0, \; \|b\|_2^2 = 1 \end{align*}$$

Since (6.11) is a convex quadratically constrained problem, it can be efficiently solved by standard solvers (e.g., CVX). The obtained $b$ is discretized to take binary values in $\{0, 1\}^k$, which results in detecting foreground bounding boxes, i.e., people participating in the activity. In our implementation, for $b$ with size $10^4$, it takes typically less than 10s
to solve (6.11) on a 2.66GHz, 3.49GB RAM PC.

After finding $X$ and $b$, in the M-step, the points are moved to their expected locations $\hat{Q} = XQ$, in the next E-step. This, in turn, changes their layout. Therefore, for the next M-step, we recompute: (i) $W$ to capture new neighbor relations between the points, and (ii) descriptors $h_i = V_ib$ to account for any new bounding boxes indicated by the new $b$ that fall within their neighborhood. The EM iterations are repeated until changes of the objective of (6.10) become close to zero. We usually run 3-5 iterations. In our experiments, the results of the EM are typically not affected by a specific choice of the initial layout of control points.
Chapter 7 – Learning

As specified in chapter 5 and (section 6.2), the set of parameters of the chains model is
\( \Theta = \{ \lambda_S, \lambda_E, \lambda_M, P(F_j|F_i) \} \), where the transition probabilities \( P(F_j|F_i) \) depend on the
covariance matrix of human poses \( \Sigma \). These parameters are learned from training videos
showing instances of an activity class. We assume that the \( R \) training videos are labeled
with the start, end frames of the activity, and the length of the video \( \{ (t_{S,r}, t_{E,r}, t_r) : r = 1, \ldots, R \} \), and with a bounding box around all participants in the activity in each frame.

Then, we compute \( \lambda_S = \frac{R}{\sum_r t_{S,r} t_r} \), \( \lambda_E = \frac{R}{\sum_r (1 - t_{E,r} t_r)} \), and \( \lambda_M = \frac{R}{\sum_r (t_{E,r} - t_{S,r})} \). For \( \Sigma \), we
first run the people detector of [14], and keep only those detections that fall within the
labeled bounding boxes. Then, we map the remaining detections to codewords of the
dictionary of human poses. Finally, each element \( \Sigma_{uu'} \) is computed as the covariance of
coop-occurrence of human poses \( u \) and \( u' \) in neighboring frames. Where, \( u_{ij} = (V_i - V_j)b \),
and the mean \( \mu \) for \( N \) values of \( u \) is computed as \( \mu = \frac{1}{N} \sum_N u \), and the variance \( \Sigma \) is
computed as \( \mu = \frac{1}{N-1} \sum_N (u - \mu)(u - \mu)^T \).

For learning the dictionary of poses we employed the approach used in [59]. Given
a set of example videos of all activity classes, they extract poses boxes, and then learn
a sparse dictionary of most discriminative poses. Dictionary learning is cast within the
large-margin framework with an L-1 regularization for sparse dictionary. In equation 7.1
\( e = [a_1^2, a_2^2, \ldots, a_k^2, \ldots] \) is the weights of the dictionary elements and \( a_k \) are auxiliary
variables, and \( k \) is the index over all codewords.
This gives 7.2 which is the objective function for learning the weights, where $\eta$ is the learning rate.

\[
\arg\max_e \quad z^T \frac{e}{\|e\|} \\
\text{subject to} \quad e \geq 0
\] (7.1)

It is solved using a gradient ascent with the update function in 7.3.

\[
\arg\max_e \quad \frac{1}{\hat{R}(a)} \sum_k z_k a_k^2, \quad e = [a_1^2, a_2^2, \ldots, a_k^2, \ldots], \quad \hat{R}(a) = \|a\|_2^2 \\
\text{subject to} \quad e \geq 0
\] (7.2)

\[
a_k \leftarrow a_k + \eta \frac{z_k \sqrt{\hat{R}(a)} - \sum_k z_k a_k^2}{\hat{R}(a)} \cdot a_k
\] (7.3)
Chapter 8 – Results

Our approach is evaluated on two benchmark datasets. First, the collective activity dataset [6] consists of 75 short videos of crossing, waiting, queuing, walking, talking, running, and dancing. This dataset tests our performance on collective behavior of individuals under realistic conditions, including background clutter, and transient mutual occlusions of actors. The video frames are 640x480 pixels. 66.6% and 33.3% of the videos from each class are used for training and testing, respectively. The dataset provides labels of every 10th frame, in terms of bounding boxes around people performing the activity, their pose, and activity class. Second, the UT-Interaction dataset [49] consists of 20, 1-minute videos of continuous executions of 6 classes of human interactions: shaking-hands, pointing, hugging, pushing, kicking, and punching. Each video shows at least one instance of every interaction class, where in some cases distinct activities may co-occur. The video frames are 720x480 pixels. The videos show two scene types: 10 videos are taken on a parking lot, and the other 10 videos are captured in a natural setting by a moving camera. The UT-Interaction dataset presents a number of challenges: the jittery camera motion, simultaneous performance of several activities, activities may begin and end at arbitrary times, presence of people who are not involved in the activity, etc. The dataset provides ground truth labels in terms of time intervals of the activities, and bounding boxes around all actors. Our evaluation setup is the same as that presented in [48]. Specifically, for training, we use 20% of the available manual segmentations of
the videos into 60 intervals, each occupied by a unique activity instance. We test on the full (unsegmented) sequences.

We test different aspects of our approach through the four following variants:

**Var1** is our default. It runs the MAP inference (section 6.1) on the optimally warped grid of control points (section 6.2), whose context descriptors (section 4.2) represent histograms of human poses detected by the detector of [14].

**Var2** serves to evaluate how much the use of control points as activity features contributes to our performance, by replacing the control points with raw detections of people in the video. Var2 uses the same algorithms as Var1, except that instead of the control points we run our MAP inference directly on bounding boxes detected by the people detector of [14]. These detections are described by the pose descriptor (section 4.1). Note that, in Var2, we cannot search for the optimal locations of video features, because the features are fixed people detections. Thus, instead of running the iterative EM estimation (section 6.2), we compute the transition probabilities $X$ in one-shot matching of these detections from (6.10), where detection similarities $C_{ij}$ are treated as constant differences of the pose descriptors. Note that Var2 also helps us evaluate the sole performance of the proposed chains model, without boosting its inference with searching for optimal features.

**Var3** is designed to evaluate how much the use of the sophisticated people detector of [14] affects our performance, by replacing the bounding boxes with more noisy, lower-level features. To this end, given a video, we compute HOGHOF descriptors at a dense grid of space-time interest points (STIP) using the recent approach of [28]. As in Var1, we here use K-means to build a dictionary of 300 codewords of HOGHOF descrip-
Table 8.1: A summary of the four variants of our approach.

<table>
<thead>
<tr>
<th>Variant</th>
<th>Object Detector</th>
<th>STIP</th>
<th>Shape Context</th>
<th>Raw Detections</th>
<th>Warping</th>
</tr>
</thead>
<tbody>
<tr>
<td>Var1</td>
<td>✓</td>
<td>-</td>
<td>✓</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Var2</td>
<td>✓</td>
<td>-</td>
<td>-</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Var3</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Var4</td>
<td>✓</td>
<td>-</td>
<td>✓</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

tors extracted from training videos. Similar to Var1, STIPs detected in test videos are mapped to the codewords, which enables estimation of the context descriptors of our control points (section 4.2). As in Var1, we search for the optimal warping of these control points capturing the context of HOGHOF-based codewords.

Var4 helps us evaluate whether enabling invariance to locally affine transformations while warping the grid of the control points plays a significant role for activity recognition. Var4 uses all the steps of Var1, except that we set $\alpha = 0$ and $\beta = 0$ in (6.10). Note that our complexity is not reduced by setting $\alpha = 0$ and $\beta = 0$. Table 8.1 gives a summary of the four variants.

8.1 Quantitative Results

We use three metrics for evaluation: activity classification accuracy, recall and precision of activity detection (a-recall, a-precision), and recall precision of detecting people involved in the activity (p-recall, p-precision). For evaluating a-detection, we compute a ratio, $\rho_a$, of the intersection and union of detected time intervals and ground-truth time intervals of activities. If the activity is correctly recognized, and $\rho_a > 0.5$ then the detected interval is declared true positive (a-TP), otherwise it is false positive (a-FP). Note that evaluating a-detection amounts to evaluating detection of the start and
end frames of the activity. For evaluating p-detection, we compute a ratio, $\rho_p$, of the intersection and union of detected bounding boxes and ground-truth bounding boxes of people participating in activities. If the activity is correctly recognized, and $\rho_p > 0.5$ then the detected person is declared p-TP, otherwise they are p-FP.

Table 8.2 compares our classification accuracy with that of the state of the art approaches [57, 6, 38] on the Collective Activity Dataset. For running and dancing classes, no previous results have been reported. Table 8.2 also shows the average running times of our different variants. The reported running times include only the MAP inference, and do not include the time it takes to run the people detector, and compute other features and descriptors. As can be seen, Var1 outperforms [57, 6, 38] in reasonable running times. Var2 is the fastest, but also the worst of all our variants, because it does not search for the optimal features, but takes fixed people detections as activity features. Nevertheless, the inference of our chains model on “raw” features, in Var2, compares favorably against the competing methods. From Table 8.2, searching for the optimal features in Var1 improves performance by 3.9% relative to using “fixed” features in Var2, with reasonable increase in running time. The results for Var3 and Var4 suggest that our approach performs competitively well even with “poorer” STIP features, and that enabling invariance to locally affine transformations of features in Var1 improves performance by 3.5% over Var4. The confusion matrices are shown in figures 8.1-8.2.

Table 8.3 compares our classification accuracy and a-detection FP rate with those of [48] on the UT-Interaction dataset. We here use Var3 for fair comparison, because [48] does not use any people detector. We outperform [48] in both metrics. Var3’s area under ROC curve for a-detection is 0.94 which outperforms 0.91 of [48]. Also,
Table 8.2: Average classification accuracy, and running times of the MAP inference on the Collective Activity Dataset [6]. Our Var1 outperforms [38], [6] and [57] on all classes.

<table>
<thead>
<tr>
<th>Class</th>
<th>Var1</th>
<th>Var2</th>
<th>Var3</th>
<th>Var4</th>
<th>[57]</th>
<th>[6]</th>
<th>[38]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Walk</td>
<td>72.2%</td>
<td>68.2%</td>
<td>68.8%</td>
<td>68.5%</td>
<td>68%</td>
<td>57.9%</td>
<td>25.5%</td>
</tr>
<tr>
<td>Cross</td>
<td>69.9%</td>
<td>65.1%</td>
<td>67.3%</td>
<td>66.4%</td>
<td>65%</td>
<td>55.4%</td>
<td>38.9%</td>
</tr>
<tr>
<td>Queue</td>
<td>96.8%</td>
<td>96%</td>
<td>96.5%</td>
<td>96.2%</td>
<td>96%</td>
<td>63.3%</td>
<td>25.5%</td>
</tr>
<tr>
<td>Wait</td>
<td>74.1%</td>
<td>70%</td>
<td>72.0%</td>
<td>71.1%</td>
<td>68%</td>
<td>64.6%</td>
<td>24.4%</td>
</tr>
<tr>
<td>Talk</td>
<td>99.8%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>83.6%</td>
<td>43.0%</td>
</tr>
<tr>
<td>Run</td>
<td>87.6%</td>
<td>80%</td>
<td>82.7%</td>
<td>81.3%</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Dance</td>
<td>70.2%</td>
<td>65%</td>
<td>67.2%</td>
<td>67.6%</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Avg</td>
<td>81.5%</td>
<td>77.8%</td>
<td>79.0%</td>
<td>78.0%</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Time</td>
<td>55s</td>
<td>42s</td>
<td>54s</td>
<td>51s</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Table 8.3: Average classification accuracy and false positive rates for a-detection on the UT Interaction Dataset [49].

Var3’s area under ROC curve for p-detection is 0.87. Our ROC curves are presented in figures 8.3-8.4.

8.2 Qualitative Results

For generating qualitative results we use our default Var1. Figures 8.5-8.6 show our successful results on few example frames from the UT-Interactions and Collective Activity datasets. Figure 8.7 shows a failure example, where Var1 correctly detects activity *hugging*, but has FP for one of the actors. The FP is simply too close to the two truly hugging people.
Figure 8.1: Confusion Matrix for Collective Activity Dataset

Figure 8.2: Confusion Matrix for UT-Interaction Dataset
Figure 8.3: ROC curve for activity detection on UT-Interaction Dataset. We compute a ratio, \( \rho_a \), of the intersection and union of detected time intervals and ground-truth time intervals of activities. If the activity is correctly recognized, and \( \rho_a > 0.5 \) then the detected interval is declared true positive, otherwise it is false positive. Note that evaluating activity detection amounts to evaluating detection of the start and end frames of the activity.
Figure 8.4: ROC curve for people detection on UT-Interaction Dataset. We compute a ratio, $\rho_p$, of the intersection and union of detected bounding boxes and ground-truth bounding boxes of people participating in activities. If the activity is correctly recognized, and $\rho_p > 0.5$ then the detected person is declared true positive, otherwise they are false positive.
Figure 8.5: Results of Var1 on example frames from the UT-Interaction Dataset. Our MAP inference correctly recognizes and detects *pushing* and *kicking* even when these activities co-occur in the video (top); identifies foreground control points (only a few examples shown as colored ellipses, for clarity); and detects people who are involved in the activity (colored bounding boxes); the control points and people detections that are inferred to belong to the same activity are marked with the same color; the other people detections that are not associated with any activity are marked with black bounding boxes.

Figure 8.6: Results of Var1 on example frames from the Collective Activity Dataset [6]: See caption of Figure 8.5. Var1 is able to detect distinct co-occurring activities. Sometimes, Var1 infers two activities instead of one, when the true activity is spatially spread-out (e.g., walking).
Figure 8.7: A failure example on the UT-Interaction Dataset: See caption of Figure 8.5. Var1 correctly detects the activity *hugging*, but wrongly identifies one bounding box as the actor (red); the other actor is correctly identified (magenta); this error happened because the FP is very close to the two truly hugging people.
Chapter 9 – Conclusion

Complexity of group activities requires a more symbiotic interaction between high-level inference algorithms and low-level feature extractors than seen in existing work. We here specify this symbiosis as an iteration in which inference guides low-level algorithms in their search for the optimal features, and, conversely, adaptive feature extraction facilitates reasoning about many competing hypothesis. Our evaluation of benchmark UT-Human Interaction and Collective Activities datasets demonstrates that we outperform the state of the art with reasonable running times.
Bibliography


