Utrecht University

Master Thesis

Human Interaction Recognition from Video

Jos Wind
ICA-3345173

March 2017

Department of Information and Computing Sciences
Faculty of Science
Utrecht University

Jos Wind (3345173)
j.wind@students.uu.nl

Supervised by:
Coert J. van Gemeren, MSc
dr. ir. Ronald W. Poppe
prof. dr. Remco C. Veltkamp

Abstract

Human action recognition has received much attention during the last decade. Inspired by the work by Patron-Perez et al. [36, 37], we construct a pipeline that recognizes 7 person-to-person interactions from video. We focus on urban settings and use videos that have been recorded using consumer hand-held digital cameras. Full bodies are visible, but occlusions do occur over time.

We use the part based models by Felzenszwalb et al. [22] to detect each person in every frame of the video. Human detections are tracked over time using the human tracking framework by Choi et al. [7, 8]. Similar to Wang et al. [52], we compute dense trajectories for each track. The orientation of each person is estimated and pairs of people are formed that are simultaneously on screen. We classify the interaction of each pair using a multiclass SVM.

We look into the performance of each part of the pipeline in order to understand its relative strengths and weaknesses. The human tracking approach leads to good results on the ETH data set [18], but the results on the Collective Activity data set [10] leave room for improvement. We obtain an accuracy of 34.97% while we try to classify 8 classes of orientations of the Collective Activity data set. Many wrong classifications lie in closely related classes. For the final interaction recognition we obtain an overall accuracy of 53.57%. Orientation estimation has an important influence on the overall performance, but the influence of the dense trajectories and the distance between the targets is limited.

Contents

Chapter 1

Introduction

Much progress has been made in the area of video understanding during the last decade. Video understanding is a subfield of computer vision that is concerned with the automatic interpretation of videos. One of its main goals is to recognize human actions from video. Various actions have been subject of research, ranging from simple motions in motion capture labs using multiple high-end cameras to complex real-world settings captured using a single hand-held camera. These different settings all have their own characteristics, making a general solution very challenging.

Human actions may take place at different levels of complexity. Aggarwal and Ryoo make the following distinction [1]:

1. *Gestures* involve an elementary movement of a body part, such as moving an arm.
2. *Actions* are composed of a complex interplay between gestures, such as walking.
3. *Interactions* involve two people that interact with each other, such as a couple hugging.
4. *Group actions* involve multiple people who all take part in the same action, such as a group playing football.

In this project we will focus on analyzing interactions between pairs of people. In the next section we will first introduce the background, followed by an overview of our approach.

1.1 Background

Action recognition can be viewed as a subfield of general video understanding. We distinguish between two general approaches:

1. With *detection* we try to localize *where* and *when* a certain action takes place in a video.
2. With *classification* we try to determine *which* action takes place in a video containing a single action.

For complete action recognition we require both detection and classification, because we wish to know where, when and which actions take place in a video. Nevertheless, both types are often studied separately because of their different characteristics. With detection we have to search through each entire frame of the video at multiple scales to find possible matches. We face the risk of missing actions or generating false positives. With classification we already know when and where the action occurs, but we have to classify to which class it belongs.

With both detection and classification we face similar fundamental challenges. First of all, humans have different appearances and move continuously. A large variety of poses, including continuous articulations over time, are difficult to model. There are also anthropomorphic differences between individuals. For example, not everyone walks in the same manner. We wish to generalize over such variations, but at the same time we still have to be able to distinguish between the different actions. Other challenges arise when we use realistic scenarios, such as complex environments and recording settings. Dynamic and cluttered environments are difficult to work with. Challenging lightning conditions, camera motion and illumination changes over time also require careful attention.

We consider interaction recognition to be a subfield of high-level action recognition. It is concerned with the detection and classification of interactions between persons. Human interaction recognition has received less attention than general action recognition. One reason for this is that it is fundamentally more complex: instead of recognizing the action of a single person, we have to take two persons into account. When two people interact on a single camera, body parts will almost certainly be occluded over time. Given the promising results of action recognition over the last years, human interaction recognition seems like a natural next step in the area of video understanding.

Many interesting applications exist for action and interaction recognition, such as surveillance systems in public places (automatic detection of suspicious behavior), real-time monitoring of people in need of help, gesture-based human computer interaction and automatic search methods for video [39].

1.2 Focus

In this project we will focus on interactions that take place in urban settings. We aim to both detect and classify seven classes of person-to-person interactions, such as people approaching each other, passing by and walking side-by-side. We use a data set containing 44 videos, which are recorded in urban settings with consumer hand-held digital cameras with varying view point [10]. Similar to related work, we use 32 videos for training and 12 for testing. The videos contain multiple persons whose full bodies are visible, but occlusions do occur over time. Multiple interactions may occur simultaneously in a single video. We do not focus on a real-time solution and expect full videos to be available at the start.

We will construct a pipeline for human interaction recognition. We use part based models to detect every person in every frame of the video. The human detections are then tracked

over time. We estimate the orientation of each human detection and use these to generate an orientation histogram for each human track. Next we form pairs of people that are simultaneously on screen and compute their distance. Finally we compute dense trajectories for each human track and use these to train a multiclass interaction classifier.

Our approach is based on the work by Patron-Perez et al. [36, 37]. Their goal is to detect and classify four classes of person-to-person interactions from TV shows, such as hand shakes and high fives. They use local features to find every person in every frame of the video. The human detections are tracked over time and a discrete head pose classifier is trained to distinguish between five classes of head poses. Local spatial and temporal features are extracted in order to predict an individual action class for each person in every frame. Pairs of people are formed based on their relative location. A structured learning approach is then used to link the individual action classes to interactions between pairs of people [47, 48]. More details can be found in Section 4.1.

While Patron-Perez et al. restrict their approach to videos from TV shows, which contain only upper bodies, we focus on more general videos containing full bodies. Our setting contains a larger variety of poses and the camera conditions are more challenging. Because of this, we use a more sophisticated approach for human detection and tracking and we focus on full orientation estimation instead of just head pose estimation. We do not compute intermediate action classes, but we generate dense trajectories for the human tracks and use them directly for classification. While Patron-Perez et al. feed the structured learning framework information from all persons in a frame, we classify each pair of tracks independently.

1.3 Research questions

Our human interaction recognition pipeline consists of several parts. We will look into the performance of each part in order to understand its relative strengths and weaknesses.

Additionally, we pose the following research questions:

1. Which type of dense trajectory descriptor is most suited for interaction classification? Does this depend on the interaction class?
2. How does the use of orientation estimation affect the interaction classification performance?
3. How does the use of distance between the targets affect the interaction classification performance?

This thesis is organized as follows. We will first discuss related work in Chapter 2. We will describe the theory behind the techniques that we use in Chapter 3. Next we will discuss our human interaction pipeline in detail in Chapter 4 and present experimental results in Chapter 5. Finally, we will discuss our conclusions and directions for future work in Chapter 6.

Chapter 2

Related work

An impressive amount of work has been done in the area of action recognition. Interaction recognition has received less attention, as it is fundamentally more complex. Instead of recognizing the action of a single person, two or more persons have to be considered.

Various approaches to interaction recognition have been proposed. Early methods often use a sequential approach, which models human tracks using local features and uses gross body movements and proximity cues in a high-level representation. More recently there has been a focus on the explicit modeling of body parts in order to deal with fine-grained interactions. There are also approaches that try to unify human tracking and interaction recognition.

2.1 Sequential approaches

With sequential approaches human tracks are typically found using an (external) tracking-by-detection approach. The local context of the human tracks is described using spatial and temporal features, such as HOG and HOF descriptors [13, 14] and dense trajectories [52]. A bag of words approach is often used to learn an interaction classifier.

Several variations to the sequential approach have been proposed. For example, Patron-Perez et al. introduce a pipeline that focuses on recognizing person-to-person interactions from TV shows [36, 37]. First human tracks are found using a tracking-by-detection approach. A per-person descriptor is constructed that combines the head orientation and the local spatial and temporal context of each person. Spatial relationships between the interacting individuals are captured in a structured learning framework. A complete overview of their pipeline can be found in Section 4.1.

Sener and Ikizler-Cinbis worked on the same setting as Patron-Perez et al. [42]. Human tracks are found using an off-the-shelf person detector and tracking method. They form pairs of interacting persons and jointly incorporate their appearances and relative spatial positions.

For each pair of persons a two-person descriptor is constructed that contains both shape and motion. Unlike Patron-Perez et al., a weakly supervised learning approach is used. They supply the input videos only with the action class label and do not use any other annotation, which makes training much less cumbersome. This does lead to much noise in the data, which is addressed by jointly leveraging visual and spatial characteristics of the interaction within a multiple-instance learning (MIL) framework.

Prest et al. focus on human-object interactions [40]. They use a tracking-by-detection approach to obtain tracks and model interactions by localizing the object and person in both space and time. Actions are represented as the relative trajectory of the object with respect to the person. They show that their explicit human-object model is an informative cue for action recognition and complements traditional low-level descriptors.

Deng et al. worked on collective activity recognition, which involves groups of people. They use a standard human detection approach and introduce a joint framework for multi-person interaction recognition, which integrates graphical models and deep neural networks [16]. This way, the gap between low-level classifications and the classification of high-level concepts is bridged. A neural network classifies low-level image inputs, refines the classifications using message passing between the outputs and performs structured learning with gating functions.

Park and Trivedi present the *Track-Body Synergy (TBS)* framework, which focuses on the analysis of multi-person interactions across multiple views [35]. The framework consists of two stages: track-level analysis deals with tracks of moving bounding boxes around each person, while body-level analysis focuses on coordinated postures and gesture patterns of body parts. Key to the system is the integration of the bottom-up vision process and a context-driven top-down process that generates hypotheses about possible human interactions.

2.2 A focus on body parts

Although the sequential approach produces state-of-the-art results, it has several disadvantages. Much information from the context and interactions is discarded. The movements of body parts are only implicitly modeled, which may not be sufficient for fine-grained actions.

Coert van Gemeren detects fine-grained, coordinated interactions between pairs of people using a pictorial structures model that incorporates joint movements [50]. Poselets are used to estimate the locations of key joints. When two joint locations are close, the area is described using HOG and HOF features. An SVM classifier is trained to recognize the interaction. In a follow-up project van Gemeren models fine-grained interactions using a spatio-temporal deformable part model (DPM), which models the interacting individuals in a single graph that contains combinations of feature descriptors [49].

Park and Aggarwal use a bottom-up process to simultaneously track and segment multiple body parts [34]. At pixel level a Gaussian mixture model is used to train and classify individual pixels based on color. At blob level multiple pixels are merged into coherent blobs. Blobs are

tracked between frames and associated to body parts within a single frame. At object level a coarse model of the human body is applied as empirical domain knowledge. The human body model is able to resolve ambiguity due to occlusions and image noise and is able to recover from intermittent tracking failures.

Kong and Jia present an interesting approach that unifies a sequential approach with a focus on body parts [29]. They unify body part level features with the action level of person-to-person interactions, which allows them to cope with interactions that involve complicated pose configurations. The body part level models motion constraints of body parts, while the action level introduces semantic relationships between the action classes. A hierarchical model is used with a hidden conditional random field to unify the mutual dependencies at both levels.

2.3 Unified tracking and interaction recognition

Some methods try to unify human tracking and interaction recognition in a single framework. For example, Choi and Savarese argue that there is a strong correlation between an individual's motion, his activity, and the motion and activities of other nearby people [9]. In order to leverage this, they treat human tracking and interaction recognition as a single problem. They introduce a hierarchy of activity types that creates a natural progression from an individual's motion to the activity of a group as a whole. Their framework is able to simultaneously track multiple people and detect their individual activities, person-to-person interactions and collective activities.

Bagautdinov et al. introduce a unified framework for multi-person social scene understanding [2]. Similar to Choi, they do not treat human detection and tracking as a separate problem but introduce an architecture to jointly detect multiple persons, infer their individual social actions and estimate the collective actions using a single pass through a neural network.

Chapter 3

Theory

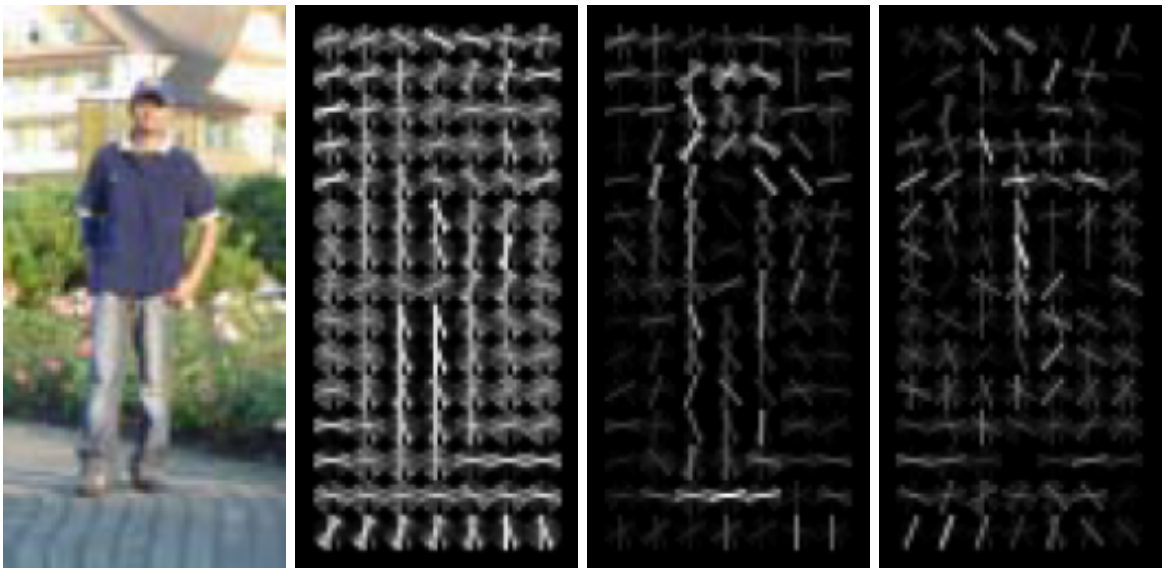
In this chapter we will discuss the theory behind the techniques that we use. We will discuss several feature description methods in Section 3.1 and classification in Section 3.2.

3.1 Feature description

To be able to detect objects in an image, we first have to transform the image into a representation that is useful for a learning algorithm. This preprocessing step is necessary to make the pattern recognition problem easier, to deal with nuisance factors and to speed up the computation. It involves two steps:

1. Feature detection. For each image point we decide whether it is a feature point or not. We distinguish between *dense* and *sparse* approaches. With the dense approach we place a regular shaped grid on top of the image and declare all vertices features points. With the sparse approach we compute *interest points*, which may be unevenly distributed over the image. These points have certain mathematical properties to ensure that they can be detected in two or more images of the same scene. Common examples are edge points and corner points. Feature points can be computed at different scales.
2. Feature description. Once we have a set of feature points, the next step is to describe the local appearance of each. We use a local image patch around a feature point to compute the *feature descriptor*, which is a compact representation that summarizes the contents of the image patch. Many feature descriptors exist for various applications. In general two characteristics are desirable. The feature descriptor has to generalize over small variations in appearance, background and viewpoint, but it also has to be sufficiently rich to allow for classification.

In the following subsections we will present various feature descriptors. The learning algorithms that use the feature descriptors will be discussed in Section 3.2.



(a) Test image (b) HOG descriptor (c) Positive weights (d) Negative weights

Figure 3.1: Example of the HOG descriptor. Images by Dalal and Triggs [13].

3.1.1 Histogram of Oriented Gradients

The basic idea of the Histogram of Oriented Gradients (HOG) descriptor by Dalal and Triggs [13] is to characterize the local appearance and shape of an image by the distribution of local intensity gradients. The HOG descriptor is often used for pedestrian detection because it captures coarse spatial structure well and it is invariant to small local deformations. HOG uses a dense approach with a static grid representation that is very usable for visualization (see Figure 3.1).

The descriptor is computed on a *bounding box* of the object. It consists of a collection of normalized histograms that are computed over spatially offset patches. First the image gradients are computed using a simple 1-dimensional $[-1, 0, 1]$ kernel that is used both horizontally and vertically. The gradient orientations are quantized into nine bins that are spread over $0° - 180°$. A detection window of 64×128 pixels is divided into a regular grid of 8×16 cells. The contribution to the histogram is weighted by the gradient amplitude and the distance from the center of the cell (i.e., more central pixels contribute more). Block descriptors are then formed by concatenating and contrast-normalizing the descriptors in blocks that consist of 2×2 cells. Finally the block descriptors are concatenated to form the final HOG descriptor.

3.1.2 Features based on Optical Flow

Unlike the HOG descriptor, which describes the local appearance of a single image, there are also descriptors that describe the motion of an object over time. These descriptors are based on optical flow, which is the pixel-wise oriented difference between subsequent frames.

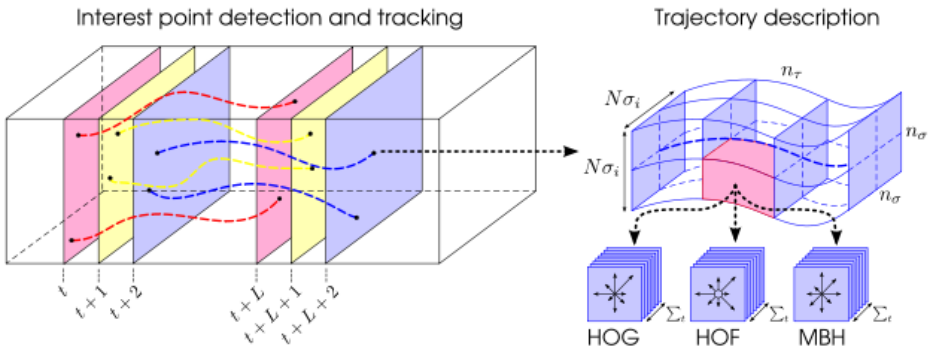


Figure 3.2: HOG, HOF and MBH descriptor as used by Kläser et al. [28]

Optical flow can be thought of as the apparent motion of the objects in a frame, which is caused by the relative motion between the observer and the scene. It basically describes what the location of an image point is in the next frame.

Optical flow can be computed using sparse and dense methods. Sparse optical flow methods track a number of stable corner points between subsequent frames, while dense methods track every single pixel to the next frame. Shi and Tomasi showed that for a stable corner point both eigenvalues of its autocorrelation matrix have to be large enough [43]. The Shi-Tomasi corner points are often in conjunction with the sparse Lucas-Kanade optical flow algorithm [33], which assumes that the optical flow is constant in the local neighborhood of a pixel. The OpenCV library [12] offers an iterative version of the Lucas-Kanade algorithm, which uses pyramids at different scales in order to be able to detect both small and large motions [6].

Dense optical flow can be computed using the Horn-Schunck [26] or Farnebäck method [21], both of which are also available in the OpenCV library. A dense optical flow field $\omega = (u_t, v_t)$ consists of a horizontal part u_t and a vertical part v_t at time t. Given two subsequent frames at time t and $t+1$, the field ω supplies every pixel of frame t with a vector that indicates its estimated location in frame $t+1$.

Well known feature descriptors based on optical flow are the Histogram of Optical Flow (HOF) descriptor [14, 31] and the Motion Boundary Histogram (MBH) descriptor [14, 52], both illustrated in Figure 3.2. The HOF descriptor is similar to the HOG descriptor, but it is computed using a dense optical flow field instead of image gradients. It is a good representation of the absolute motion between frames. The MBH descriptor is similar to the HOF descriptor but uses the horizontal and vertical derivatives of the optical flow field. That way only the change in motion is captured, canceling any constant motion such as camera movements.

3.1.3 Scale-Invariant Feature Transform

The Scale-Invariant Feature Transform (SIFT) [32] can be used for both feature detection and description. First interest points are detected along with their scales and rotations. Next

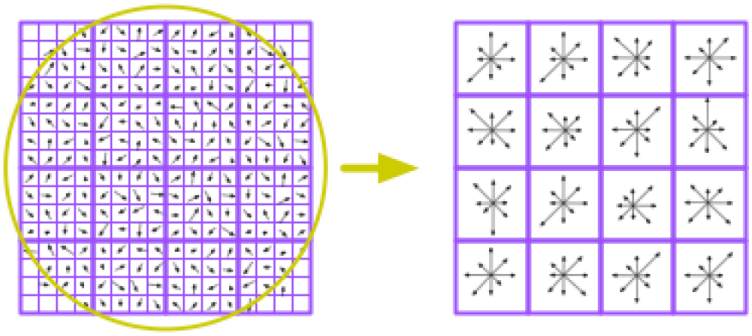


Figure 3.3: Keypoint description: from image gradients to SIFT descriptor [32]

the area surrounding the interest points is described such that it is partially invariant to local shape distortion and changes in illumination.

Lowe distinguishes four stages of computation:

1. Scale-space extrema detection. All image scales and locations are searched for potential interest points that are invariant to scale and orientation using a difference-of-Gaussian function. The difference-of-Gaussian function can be computed from the difference of two nearby scales of the Gaussian function of the image separated by a constant factor. The local extrema of the difference-of-Gaussian function are computed by comparing each sample point to its 26 neighbors in a 3D volume. The extrema are identified at locations where the 26 neighbors are either all smaller or larger than the current value.

2. Keypoint localization. At each potential interest point a detailed model is fit to determine its location and scale at subpixel accuracy. This is done by applying a local quadratic approximation (using a Taylor expansion of the point) and returning the position of the peak. Only stable keypoints are selected by filtering out points in smooth regions and on edges.

3. Orientation assignment. We assign an orientation to each keypoint based on the local image properties. In all future steps we use transformed image data relative to the assigned orientation, scale and location in order to achieve invariance to these properties.

4. Keypoint description. The magnitude and orientation of the image gradients are sampled around the keypoint location at the selected scale. The SIFT descriptor is then formed from a vector with all orientation histogram entries. We use a 4×4 array of histograms with 8 orientation bins each, leading to a (normalized) descriptor vector of length $4 \times 4 \times 8 = 128$ for each keypoint (see Figure 3.3).

The SIFT descriptor is partially invariant to changes in intensity, contrast and small geometric deformations [41]. In contrast to the HOG descriptor, the SIFT descriptor is also invariant to scale and rotation.

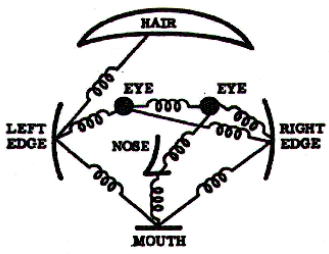


Figure 3.4: Spring-like connections are used in pictorial structures [23]

3.1.4 Deformable Part Models

Felzenszwalb et al. [22] introduce a sophisticated model for object detection that is able to represent highly articulated objects. Mixtures of multiscale deformable part models are used, which build on the pictorial structures framework [23]. Pictorial structures represent objects as a collection of parts in a deformable configuration. So, the object is not modeled as a single (rigid) part but as a collection of its parts. Mixture models are used because a single deformable model may not be expressive enough to represent rich object categories, in which objects may have a different appearance or orientation. A mixture can, for example, capture various views of the same object.

Felzenszwalb's star-structured model consists of a coarse root filter and a set of high resolution part filters with associated deformation models. The root filter covers the entire object and the part filters each cover a small part of the object. The root filter and the part filters capture the local appearance information using HOG descriptors. Certain pairs of parts are connected with spring-like connections, as illustrated in Figure 3.4. The score of a star model at a certain location and scale in a test image is the score of the root filter plus a sum over parts that maximizes the part score minus a deformation cost that measures the deviation of the part from its ideal location relative to the root [22]. The star models are finally combined in a mixture model. The score of the mixture model is simply the maximum of the scores of the star models.

3.1.5 Bag of Visual Words

The Bag of Visual Words model characterizes an entire image by summarizing the statistics of local image features. It depends on local feature descriptors such as the SIFT descriptor to describe the local appearance around interest points. However, unlike the static grid representation of the HOG descriptor, it completely ignores any spatial relationships between features. The model is especially useful when the number of features varies, for example with SIFT. Despite its simple nature, surprisingly good results can be obtained [44], mainly because richer models are much more difficult to train.

The bag of visual words model originated from the information retrieval field. In document classification a bag of words is a sparse vector that denotes how often a word occurs. So, it

is basically a sparse histogram over the vocabulary. Similarly, in computer vision a bag of visual words is a histogram of visual words. We treat images as documents (collections of detected patches) and image features as (visual) words.

The bag of words descriptor is a histogram of codeword frequencies. The codewords belong to a codebook of size k and are determined by clustering local feature descriptors (such as SIFT) and selecting the cluster centers. The patches are often clustered using the k-means algorithm, which partitions the data set into k clusters in such a way that data points in a cluster have small distance compared to points outside the cluster.

The k-means algorithm is a specific form of the well-known EM algorithm [15]. It tries to find a set of cluster centers such that the sum of squares of the Euclidean distances of each data point to its closest cluster center is a minimum [5]. It computes the cluster centers in an iterative manner, either until convergence or until a maximum number of iterations has taken place. The number of clusters is a trade-off between generalization and specificity and is usually chosen experimentally. One of the drawbacks of k-means is that, even though convergence is assured, it may converge to a local minimum. In order to better approximate the global minimum, k-means is often run multiple times and the result with the lowest error is used.

3.2 Classification

Once the feature descriptors have been calculated, a typical classification problem remains. We wish to train a model that is able to correctly categorize new examples that differ from those used for training. To train the model a supervised learning approach is often used. This means that we use an annotated data set of which we know the correct classes (ground truth) to train a model. The learning algorithm analyzes the feature descriptors of the annotated data and adjusts the weights, thresholds and other parameters of the classifier to maximize the performance of the model on the training set.

3.2.1 Support Vector Machine

The Support Vector Machine (SVM) is a non-probabilistic discriminative classifier first introduced by Vapnik [51]. It has since been used extensively for various binary classification tasks. Given an annotated data set of examples belonging to one of two classes, the SVM computes a model that is able to predict the class of new examples.

The SVM is a linear classifier that tries to find a hyperplane in the feature space such that the margin between the hyperplane and the closest examples (the support vectors) is maximized (see Figure 3.5). The optimal hyperplane is calculated using Lagrangian optimization and is described by a weighted combination of the support vectors. Because most data sets are not linearly separable, we typically allow some examples to be on the wrong side of the decision boundary. These examples are penalized using *slack variables* [5]. The penalty increases with

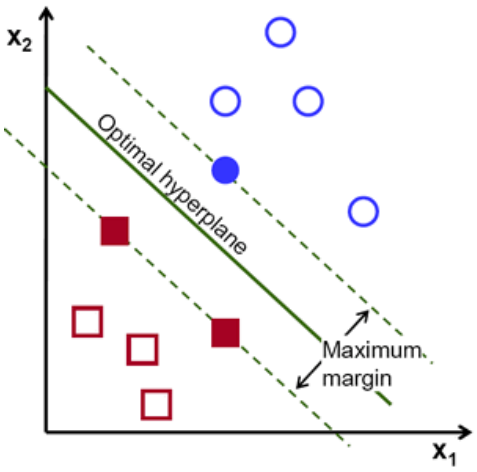


Figure 3.5: Optimal hyperplane of the Support Vector Machine. The filled circle and squares are support vectors. Image by OpenCV [12].

the distance from the boundary.

Various extensions to the SVM have been constructed. While the original SVM was a linear classifier, it is also possible to create nonlinear classifiers by applying the kernel trick [5]. A nonlinear kernel function can be used to transform the input space into a high dimensional feature space. The SVM still calculates an optimal linear hyperplane in the feature space, but with the kernel trick it can be nonlinear in the input space.

3.2.2 Multiclass SVM

A multiclass SVM works similarly as a binary SVM, but it predicts one of M classes instead of just two. It is typically constructed using multiple binary SVMs. There are two common approaches that have comparable performance [17]:

1. The one-vs-one approach uses max-wins voting (MWV). One binary classifier is constructed for every pair of distinct classes, leading to a total of $M(M-1)/2$ binary classifiers. Each classifier makes a vote and the class which has the most votes is predicted.

2. The one-vs-all approach uses a winner-takes-all (WTA) strategy. A total of M binary classifiers are constructed and each binary classifier distinguishes between a single class (positive) and all other classes (negative). The class with the largest classification score is predicted.

3.2.3 Structured SVM

The structured SVM is a generalization of the standard SVM. Unlike binary and multiclass SVMs, the structured SVM is able to predict general structured outputs [47, 48]. It is a very generic method and the output space can have various formats, such as parse trees or even more complex data structures. A multiclass SVM usually struggles with such applications, because the number of classes is typically very large and it does not take advantage of the structure of the output space.

The goal of a structured SVM is to learn functions $f : X \to Y$ from inputs $\boldsymbol{x} \in X$ to complex outputs $\boldsymbol{y} \in Y$ given a set of N training examples $(\boldsymbol{x_1}, \boldsymbol{y_1}), \ldots, (\boldsymbol{x_N}, \boldsymbol{y_N})$. In order to do this, the structured SVM tries to learn a *discriminant function* $F : X \times Y \to \mathbb{R}$ over input/output pairs that can be used for prediction. Given an input $\boldsymbol{x}$ and weight vector $\boldsymbol{w}$, the prediction is given by

$$f(\boldsymbol{x}; \boldsymbol{w}) = \arg\max_{\boldsymbol{y} \in Y} F(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{w}). \tag{3.1}$$

The discriminant function F is assumed to be linear in a feature mapping of inputs and outputs $\boldsymbol{\Psi}(\boldsymbol{x}, \boldsymbol{y})$:

$$F(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{w}) = \boldsymbol{w}^T \boldsymbol{\Psi}(\boldsymbol{x}, \boldsymbol{y}). \tag{3.2}$$

The exact implementation of a structured SVM depends on the details of the classification problem. The feature mapping $\boldsymbol{\Psi}$ codifies the underlying structure of the output space and is typically designed specifically for the problem at hand. Additionally, a custom loss function $\Delta : Y \times Y \to \mathbb{R}$ is used to penalize wrong assignments. The loss function $\Delta(\boldsymbol{y}, \hat{\boldsymbol{y}})$ quantifies the loss of predicting $\hat{\boldsymbol{y}}$ while the true value is $\boldsymbol{y}$.

Chapter 4

Pipeline

We will discuss the details of our pipeline in this chapter. Because we will build upon the work of Patron-Perez et al. [36, 37], we will first give an introduction to their pipeline in Section 4.1. Next we will give a general overview of our adapted pipeline in Section 4.2. In the subsequent sections we will discuss the details of each part.

4.1 Pipeline by Patron-Perez et al.

Patron-Perez et al. built a pipeline for human interaction recognition that produces impressive results [36, 37]. The goal of their system is to detect and classify four symmetric interactions from TV shows: hand shakes, high fives, hugs and kisses (see Figure 4.1). The interactions are called symmetric because both persons participating in the interaction have mirrored poses and contribute with the same movements.

Patron-Perez et al. use a data set based on real TV shows [38], which contains mostly upper bodies. Instead of completely analyzing every frame for possible human interaction cues, and thereby possibly learning information from background clutter, they take a person-centered approach similar to Ferrari et al. [24, 25]. Figure 4.2 shows an overview of the pipeline. The first step is to locate every person in both space and time using a tracking-by-detection approach. They try to find a bounding box for each person in every frame of the video, which is then tracked over time. Two upper body detectors are constructed that are based on a

Figure 4.1: Patron-Perez et al. distinguish between four types of interactions [38]

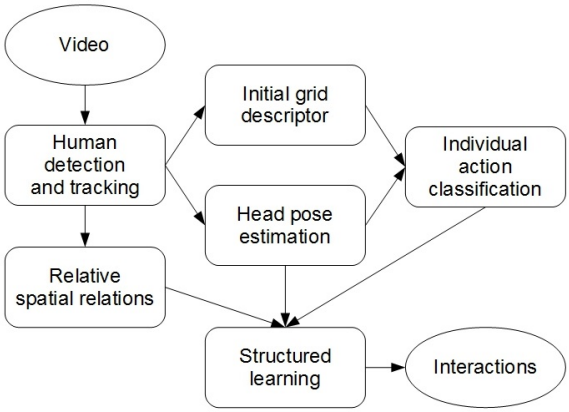


Figure 4.2: An overview of the pipeline by Patron-Perez et al. [36, 37]

HOG descriptor and a linear SVM. They use two detectors in order to handle different scales. The detectors look for upper bodies in windows of 64×64 and 128×128 pixels.

In order to track a single person through the video, the upper body bounding boxes are tracked over time. A *human track* is defined as a sequence of human bounding boxes of the same person in consecutive frames. Patron-Perez et al. track feature points [43] inside each upper body bounding box both forwards and backwards in time using a KLT tracker [33, 46]. The KLT tracker uses Shi-Tomasi corner points and the Lucas-Kanade optical flow algorithm, as explained in Section 3.1.2. Two detections are linked if the overlap ratio between the tracked points that pass through both of them is higher than a certain threshold. In order to filter out background noise, only the head region in the upper body bounding boxes is used for the linking (similar to the work by Everingham et al. [19]). Next clique partitioning is used to interpolate broken tracks [24], which may exist because of short occlusions and fast movements. Clique partitioning uses an overlap measure between the last and first bounding box of every pair of non-overlapping tracks. The position and size of the bounding boxes in a track are smoothed over time using quadratic smoothing. Tracks are split at shot changes (changes of the camera viewpoint), which are detected using simple pixel-by-pixel frame subtraction and the difference between the color distributions of the frames. Tracks that are deviant in some way are considered false positives and are filtered out using a linear classifier similar to the work by Kläser et al. [28]. A total of 20 statistical measures are used, such as the relative length of the track and the relative sizes of the bounding boxes.

Because interacting people often look at each other, Patron-Perez et al. argue that head poses contain important information about possible interactions. In order to use the information contained in the head poses, they estimate the head pose of every person in every frame after the human tracks have been computed. Similar to the method of Benfold and Reid [3, 4], a discrete head pose classifier is trained using HOG descriptors to distinguish between five different head pose classes. However, while Benfold and Reid use an ensemble of randomized ferns for classification, Patron-Perez et al. train a set of five linear one-vs-all SVM classifiers. For every bounding box of a track a score for each head pose is calculated and these scores are smoothed over time, leading to a final head pose estimation θ for each bounding box.

Given the human tracks and the corresponding head poses, Patron-Perez et al. model the local context of each bounding box by computing a person-centered descriptor $\mathbf{d}$ composed of the head pose and the spatial and temporal context. The local context describes the local appearance and motion in the video and is used to capture cues like hand and arm movements indirectly. First a uniform 8×8 grid is placed on top of the bounding box, such that the size of the cells depends on the size of the bounding box. In each cell HOG and HOF descriptors are computed, both of which are composed of five bins. The descriptors are independently normalized and concatenated, which leads to an initial grid descriptor $\mathbf{g}$ of length $8 \times 8 \times 5 \times 2 = 640$ bins.

In order to achieve some form of view invariance, Patron-Perez et al. compute the person-centered descriptor $\mathbf{d}$ by combining the head pose information with the initial grid descriptor $\mathbf{g}$. They wish to learn a different classifier for each head pose θ. The final descriptor is given by $\mathbf{d} = [\mathbf{g}^+; \mathbf{g}]$, with $\mathbf{g}^+ = \mathbf{g} \otimes \delta_\theta$. Here $\otimes$ is the Kronecker product and δ_θ is an indicator vector corresponding to the discrete head poses, having a one at position θ and zero everywhere else. This basically creates a very sparse vector of length $(5+1) \times 640 = 3840$, with the local context $\mathbf{g}$ at two different locations. The first location is used to build a different model for each viewpoint, and the last location is used to account for any information that is independent of the head pose.

Patron-Perez et al. use the person-centered descriptor $\mathbf{d}$ to classify the action of each person. The result is an *interaction score* for each individual, which denotes the interaction the person is most likely involved in. A linear one-vs-all SVM classifier is trained for each interaction using the descriptor as a data vector.

Up to this point Patron-Perez et al. have only modeled characteristics of each individual. The next step is to make pairs of people and model global characteristics of the interactions. Given the assumption that people look at each other while interacting, they combine the head orientations with the relative spatial location between each pair of people in the frame. The spatial area between two persons is divided into six discrete regions based on their horizontal and vertical position in the frame: near left, adjacent left, overlap, adjacent right, near right and far. These cues, together with the bounding boxes and interaction scores from the previous steps, are used in a structured SVM framework to classify the final interactions [48].

As explained in Section 3.2.3, the structured SVM consists of a feature mapping and loss function designed specifically for the problem at hand. Patron-Perez et al. use a feature mapping that consists of three parts:

1. The head orientation potential uses the scores of the head pose classifier. The higher the score, the greater the confidence in the classification.
2. The local context potential is similar to the head orientation potential, but uses the interaction scores instead of the head pose classifier scores.
3. The global context potential codifies the relative spatial relations between people. A configuration label is used to denote which pairs of people are interacting.

Each potential function contains bias terms that denote the confidence. Additionally, the potentials are scaled to make sure that their influence is balanced. Similar to the feature

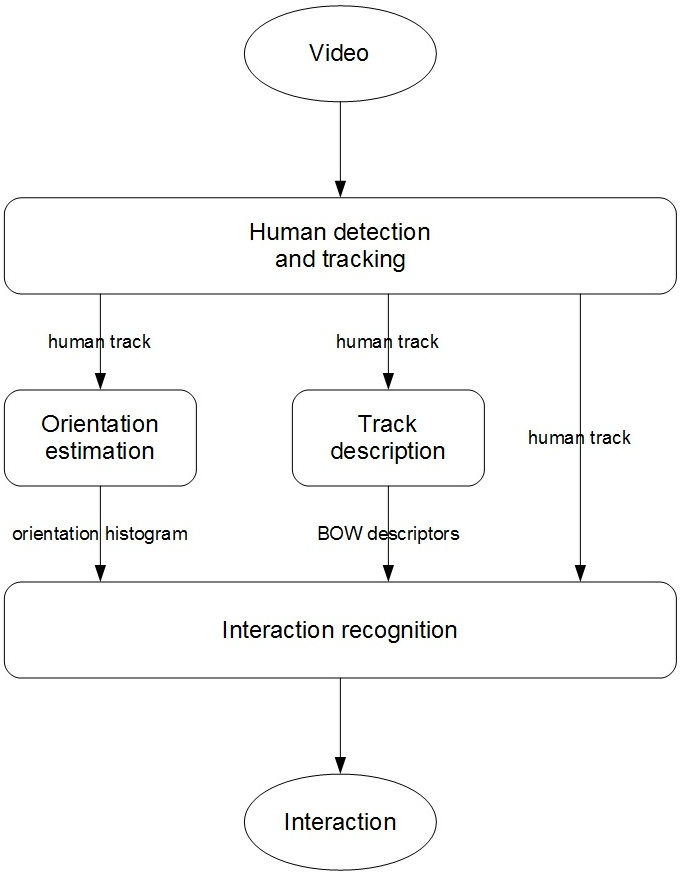


Figure 4.3: An overview of our interaction recognition pipeline

mapping, the loss function consists of three parts. The first part penalizes incorrect head orientation classes, the second part penalizes incorrect individual action classes and the last part penalizes incorrect configuration label assignments.

The structured output consists of the (classified) head poses, the pairs of interacting people (using the configuration label) and the corresponding action classes. Note that the head poses appear both as input and output. However, the input consists of the scores from the pose classifiers and the output consists of the ground truth pose classes.

4.2 Our approach

Figure 4.3 gives an overview of our pipeline. We use a similar approach as Patron-Perez et al., but there are some key differences. The main difference is that we focus on a more general setting where full bodies are visible, while Patron-Perez focus on TV shows using videos that contain only upper bodies. Because we are dealing with full bodies, the approach of Patron-Perez will not work out of the box. We use the publicly available Collective Activity data

Figure 4.4: Choi's data set contains people in urban environments [10]

set by Choi et al. [10], which contains videos of people walking in urban environments (see Figure 4.4). We distinguish between seven interaction classes: approaching, leaving, passing by, facing each other, walking side by side, standing in a row and standing side by side.

Patron-Perez et al. use a fairly straightforward approach to human detection and tracking. Upper bodies are detected using a HOG descriptor and a linear SVM, and are subsequently tracked over time using KLT tracking. While this approach works fine for upper body detection, we expect that it will not suffice for our application because of a larger variation of poses and more challenging camera conditions. We therefore use a different implementation of the tracking-by-detection approach. We detect full bodies using the part based models by Felzenszwalb et al., as discussed in Section 3.1.4. The human detections are used in the human tracking framework by Choi et al. [7, 8], which outperforms the state-of-the-art tracking methods on the publicly available ETH data set [18] in terms of track detection accuracy. Choi's probabilistic formulation is able to cope with false detections by rejecting observations that have an unlikely small or large value for a certain target class.

Experimentation by Patron-Perez et al. shows that head poses are a valuable cue for interaction recognition. However, since we have full bodies at our disposal, it seems rather limited to focus on just the head poses. We therefore use full body orientation estimation with eight discrete orientations. Our approach to orientation estimation is similar to Patron-Perez, but instead of HOG descriptors we use SIFT descriptors because of their scale and rotation invariance. Instead of focusing on a singe frame, we construct an 8-bins orientation histogram for each human track.

According to an evaluation of local spatio-temporal features by Wang et al. [54], tracking densely sampled interest points on a uniform grid in both space and time performs best for action recognition. Because of this, we choose to use Wang's *dense trajectories* [52] instead of computing local descriptors using a uniform grid directly on top of the bounding boxes (like Patron-Perez). The dense trajectories consist of sampled points that are tracked over time using a dense optical flow field. The local spatial and temporal context of the tracks is described using HOG, HOF and MBH descriptors.

Patron-Perez et al. compute an intermediate action class for each individual of every frame. The structured learning framework takes the data of all persons in a frame as input and predicts the possible interactions. A drawback of this approach is that there is no temporal consistency. The interactions are independently recognized in each separate frame. Instead of focusing on individual frames, we choose to focus on full tracks instead. We find every

pair of persons that is simultaneously on screen and look for interactions during the temporal overlap. We do not use a structured learning approach, but train a multiclass SVM that predicts the interaction between two persons.

4.3 Human detection and tracking

Similar to Patron-Perez et al., we start by tracking people. We use a tracking-by-detection approach: we first detect every person in every frame and then track the detections over time to form human tracks. We use the full human body detection and tracking approach by Choi et al. because of its impressive results [7, 8].

Choi et al. introduce two key contributions. First they automatically estimate the camera's extrinsic parameters (panning, location and velocity) while tracking the objects. They use a simplified camera model in order to reduce the number of parameters that have to be estimated. Initial camera parameters are assumed to be given, which are continuously improved by the detected targets in the image plane. Additionally, a KLT tracker is used to track feature points from the static background in order to help estimate the variations in the camera parameters over time [33].

The second contribution is that they do not ignore possible interactions between targets. Existing tracking methods usually assume that targets move independently from each other. In practice this is rarely the case: people try to avoid possible collisions and often walk in groups. Therefore Choi et al. assume that the motion of the targets may be interrelated. They introduce an interaction model that consists of a repulsion model and an attraction model. The repulsion model is used so that targets cannot collide with each other and the attraction model is used to model groups of people walking together.

Choi's tracking algorithm assumes that a bounding box is available for every person in every frame of the video. The part based models by Felzenszwalb et al. are used to obtain the human bounding boxes, as discussed in Section 3.1.4. The human detections are used to initiate new tracks and to gather evidence for existing tracks. Choi et al. introduce a probabilistic formulation to estimate stable and accurate tracks and uniquely associate them to a specific object. The system is able to handle complex scenes where multiple people move simultaneously and may occlude each other.

The goal of the tracking algorithm is to estimate and separate each trajectory in time from the other trajectories. Choi initiates a target hypothesis for each detection that does not match an existing track. If enough matching detections are found for a target hypothesis in N_i consecutive frames (where i is the index of the target), we start a new track. Detections are linked with an overlap ratio between the detections [30]. A mean-shift tracker is assigned to each target and applied to each frame of the track. If not enough detections are found for a target in a number of consecutive frames, the track is terminated. Additional rough trajectories in the image plane are computed using mean-shift [11]. For each target hypothesis a separate mean-shift tracker is applied to each frame. These rough trajectories are used as

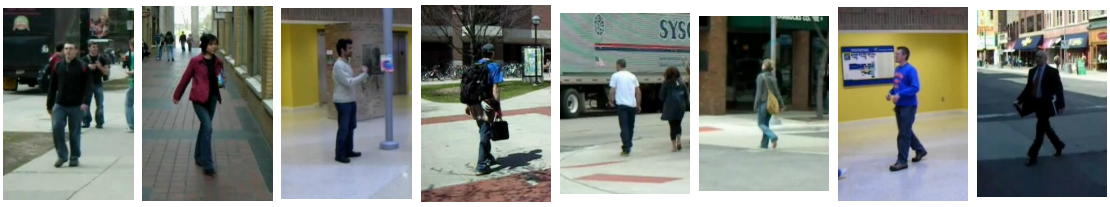


Figure 4.5: Choi's data set contains people with different orientations [10]

an additional cue to better locate the targets in the image plane and to handle situations where targets are not properly detected by the detector.

Each track is modeled as a hidden variable Z_i in 3D space. The 3D trajectories of the tracks are estimated by projecting them onto the 2D image plane (using the estimated camera model), which represents the observation variables X_i. Given the observation variables X_i, we track Z_i by searching for the most likely parameters for the camera and track states. Because of the complexity of the probabilistic formulation, it is challenging to find an analytical inference method to estimate the maximum a posteriori (MAP) solution. Choi et al. approximate the MAP using a Markov Chain Monte Carlo (MCMC) particle filter to generate randomized samples, which are used to propagate the posterior distribution in the framework [27].

The output of Choi's algorithm consists of a track for every person, which supplies every frame of the track with the time, 3D world coordinates, a 3D velocity vector and a 2D bounding box. We will only use the 2D bounding boxes, but the approach in 3D is useful because estimating trajectories in 3D is more robust than just using the image plane [8].

4.4 Orientation estimation

For each human track we wish to know the orientation of the person involved. Because the orientation may change over time, we estimate the orientation in each separate frame and use that to form an orientation histogram for the complete track. As shown in Figure 4.5, we use a discrete set of 8 orientations: front, front-right, right, back-right, back, back-left, left and front-left.

Because of the scale and rotation invariance, we use SIFT descriptors for feature description (see Section 3.1.3). Unlike the standard SIFT method, which detects sparse interest points in the image, we find keypoints on a regular tiled grid across the image. We describe the area surrounding these keypoints such that it is partially invariant to local shape distortion and changes in illumination.

We use the vector quantized SIFT descriptors in a bag of visual words model, as explained in Section 3.1.5. The first step is to set up the codebook: we generate the SIFT descriptors for each bounding box and cluster them using k-means. We use the codebook to compute the bag of words descriptor for each image and use a multiclass SVM classifier for prediction, as explained in Section 3.2.2.

4.5 Track description

In order to describe the appearance and motion of the target, we use the human tracks to extract features that model the local context. We use the dense trajectories by Wang et al. [52], which show a significant improvement over the state of the art in action classification.

Tracking interest points is challenging. First of all there is the drifting problem: trajectories tend to drift from their initial location during tracking. Wang et al. simply avoid this problem by limiting the length of a trajectory to $L = 15$ frames. Because they are mainly interested in dynamic information, static trajectories are pruned. Trajectories that contain sudden large displacements are also removed because they are likely to be erroneous. Finally there is the aperture problem: it is only possible to align patches along the direction normal to the edge direction [45]. In homogeneous areas without much structure it is hard (or even impossible) to track points. Wang et al. prune feature points in those areas in order to achieve stable results. Similar to Shi and Tomasi [43], the eigenvalues of each sampled feature point are computed and any point is pruned for which the smaller eigenvalue is below a certain threshold. This way only stable feature points remain.

On top of the human bounding boxes Wang et al. place a uniform grid spaced by $W = 5$ pixels. If at some moment no track is found in a $W \times W$ region, they sample a feature point on the grid and track it in 8 spatial scales, each spaced by a factor of $1/\sqrt{2}$. The points $P_t = (x_t, y_t)$ at frame t are tracked to the next frame $t + 1$ by median filtering in a dense optical flow field ω, which has been computed using the Farnebäck algorithm in OpenCV (see Section 3.1.2). Given a median filtering kernel M, we have

$$P_{t+1} = (x_{t+1}, y_{t+1}) = (x_t, y_t) + (M \cdot \omega)|_{(\bar{x}_t, \bar{y}_t)}. \tag{4.1}$$

Here $\cdot$ is the standard convolution operator and $(\bar{x}_t, \bar{y}_t)$ is the rounded position of (x_t, y_t). The points in subsequent frames form a trajectory $(P_t, P_{t+1}, \ldots, P_{t+L-1})$ with length L.

In order to encode the local motion pattern of the trajectory, Wang describes its shape by a sequence of normalized displacement vectors, called the *trajectory descriptor*. We have the sequence $S = (\Delta P_t, \ldots, \Delta P_{t+L-1})$ with the displacement vectors $\Delta P_t = (P_{t+1} - P_t) = (x_{t+1} - x_t, y_{t+1} - y_t)$. We obtain the trajectory descriptor by normalizing this vector using the magnitudes of the displacement vectors:

$$S' = \frac{(\Delta P_t, \ldots, \Delta P_{t+L-1})}{\sum_{j=t}^{t+L-1} \|\Delta P_j\|}. \tag{4.2}$$

The trajectory descriptor only describes the motion of the trajectory, similar to the output of the KLT tracker. Wang et al. use local descriptors in a space-time volume around the interest points to describe the appearance and motion of the neighborhood. A 3D volume of size $N \times N$ pixels and L frames surrounding each interest point is subdivided into a spatio-temporal grid of size $n_\sigma \times n_\sigma \times n_\tau$, with $N = 32, n_\sigma = 2, n_\tau = 3$. Wang et al. use this grid to compute three additional types of descriptors.

For the static appearance information an 8-bins HOG descriptor is computed with length 96. Local motion information is captured using a 9-bins HOF descriptor with length 108. The additional bin is used to handle the case of no-motion, so each bin contains a range of 45 degrees. Wang et al. finally compute a horizontal and vertical 8-bins MBH descriptor, each with length 96. All descriptors are independently normalized with the L_2-norm.

We use Wang's dense trajectories in a bag of words approach to generate our final track descriptor. We cluster each descriptor independently using k-means, so we end up with four separate codebooks. The track descriptor simply consists of the bag of words histograms.

4.6 Human interaction recognition

Given the human tracks, orientation histograms and track descriptors, we are now ready to perform the actual interaction recognition. Using the human tracks we first find every pair of persons that is simultaneously on screen. For a given track x, we find all tracks y that satisfy $y_{start} \leq x_{end}$ and $y_{end} \geq x_{start}$. We compute the temporal overlap for each pair and look for an interaction in that time interval.

For each pair we construct a feature vector that consists of the position (x and y), size (width and height), orientation histogram and track descriptor of both targets. Additionally, we include the interaction length and the distance between both targets. We use the feature vector train a multiclass SVM classifier that is able to distinguish between the seven different interaction classes.

Chapter 5

Results

In this chapter we will discuss our experiments and results. First we will give an overview of data sets in Section 5.1. In the next sections we will discuss each part of the pipeline.

Please note that all parts of the pipeline are responsible for the overall performance. If one part does not perform well, it has an important impact on all subsequent parts. Therefore we will also look into the performance of each part independently in order to understand its relative strengths and weaknesses.

5.1 Data sets

We will use two publicly available data sets to train and test parts of the pipeline. We use the Collective Activity data set by Choi et al. [10] for human detection, tracking, orientation estimation and the final interaction recognition. Additionally, we test the performance of our human detection and tracking approach using the ETH data set by Ess et al. [18].

5.1.1 Collective Activity data set

Choi et al. introduced the Collective Activity data set [10], which consists of 44 video sequences (mostly 640x480 pixels) that contain multiple people in urban settings. The videos are recorded under unconstrained real-world conditions, some of them using consumer handheld digital cameras with varying viewpoint. Some example images can be seen in Figure 4.4.

Various types of annotation are available. Choi offers annotated human tracks and has labeled each bounding box with the orientation of the person (8-bins), the atomic action (walking or standing) and a collective activity label. There are 5 classes of collective activities: crossing, waiting, queueing, walking and talking. Additionally, each pair of people has received an

Figure 5.1: The ETH videos contain people walking in urban environments [18]

interaction class: no interaction (NA), approaching (AP), leaving (LV), passing by (PB), facing each other (FE), walking side by side (WS), standing in a row (SR) and standing side by side (SS).

5.1.2 ETH data set

The ETH data set by Ess et al. consists of 8 urban video sequences (640x480 pixels) that have been recorded using a pair of cameras mounted on a chariot and a car [18]. The videos consist of people walking on the street, as can be seen in Figure 5.1. Human tracks have been annotated and camera calibrations and depth maps are available. The ETH data set is widely used to compare the results of pedestrian detection algorithms. The actions of the people have not been labeled.

5.2 Human detection and tracking

We use the default parameters of Choi's human detection and tracking system. Tracking takes place at 15 fps with two person detection models trained by Choi. One model is trained using the INRIA Person data set [13] and the other using the VOC2006 Person data set [20]. The INRIA data set contains people in various urban settings, as can be seen in Figure 5.2. The VOC2006 data is more general and contains people in a wide range of settings (see Figure 5.3). The INRIA data only contains full bodies, while the VOC2006 data also contains people who are partially visible.

5.2.1 Track matching

For track matching we use the common "intersection over union" overlap measure both in the spatial and temporal domains [55]. We compare all ground truth tracks with all found tracks. Two tracks match if both their temporal and spatial overlap exceed certain thresholds.

We first check if the temporal overlap TO between ground truth track T_g and found track T_s

Figure 5.2: The INRIA images contain people in various urban settings [13]

Figure 5.3: The VOC2006 images contain people in a wide range of settings [20]

exceeds the temporal threshold TR_{temp}:

$$TO(T_g, T_s) = \frac{Length(T_g \cap T_s)}{Length(T_g \cup T_s)} \geq TR_{temp}. \tag{5.1}$$

If the temporal overlap does not exceed the temporal threshold, we conclude that the tracks do not match. If it does exceed the threshold, we continue by calculating the spatial overlap SO. The spatial overlap between ground truth bounding box BB_{gi} and found bounding box BB_{si} of frame i is given by

$$SO(BB_{gi}, BB_{si}) = \frac{Area(BB_{gi} \cap BB_{si})}{Area(BB_{gi} \cup BB_{si})}. \tag{5.2}$$

The average spatial overlap of all N frames from the temporal intersection has to exceed the spatial threshold $TR_{spatial}$:

$$\frac{\sum_{N}^{i=1} SO(BB_{gi}, BB_{si})}{N} \geq TR_{spatial}. \tag{5.3}$$

Note that each T_g and T_s may be matched at most once. If there are multiple matches, we choose the one with the largest overlap. Unmatched found tracks count as false positives and unmatched tracks from the ground truth count as false negatives.

5.2.2 Tracking results

In order to verify that our implementation works as expected, we first test it on the ETH data. Similar to Choi, we use the tracking algorithm on frames 350 to 800 of sequence 2,

	CAD09		ActivityTracks	
Spatial threshold	30%	50%	30%	50%
True positive tracks	173	138	288	255
False positive tracks	3998	4033	3883	3916
False negative tracks	293	328	408	441
Precision	4%	3%	7%	6%
Recall	37%	30%	41%	37%

Table 5.1: Tracking results using Choi's Collective Activity data set. In all cases the temporal threshold is 50%.

which contain a total of 33 pedestrians. Choi found a total of 47 trajectories, of which 28 are considered mostly hit (track coverage larger than 80%). Using the model trained on the INRIA Person data we find a total of 144 trajectories. The model based on the VOC2006 data leads to a total of 134 trajectories. The trajectories generated using the VOC2006 data are identical to Choi's trajectories, so the human detection part is working as expected. Unfortunately we do find substantially more trajectories than Choi. However, our results are similar when we leave out all tracks shorter than 5 frames.

Next we test using Choi's Collective Activity data set. We use both the publicly available CAD09 annotation (containing 466 target tracks) and ActivityTracks annotation (696 tracks) as ground truth [10]. Table 5.1 summarizes the results for the INRIA model and both annotation types. The INRIA model generates the best results, but there is still much room for improvement. The VOC2006 model performs substantially worse. We expect that the results can be improved by optimizing the parameters.

Looking at the results, we notice that the system mostly fails when a person is in a cluttered environment or gets occluded for a few frames. Under these circumstances tracks are often split or simply cut short, as can be seen in Figure 5.4. Not many errors occur when a person is clearly in view. Figure 5.5 shows some examples of tracks that do not contain a person, but these tracks are rare and typically very short. Tracks that are split or cut short may pose a problem at later stages of the pipeline. We expect that it will be difficult to recognize an interaction if it is only partially visible in a track.

5.3 Orientation estimation

For the orientation estimation we use an implementation by Coert van Gemeren. For each orientation of Choi's Collective Activity data set we randomly select 100 test images. We compute a bag of words codebook by clustering SIFT descriptors of the images using k-means. We run k-means 8 times and experiment with different values for k. It turns out that $k = 256$ leads to good results. We generate codewords for 625 randomly selected training images for each orientation and use these to train a multiclass linear SVM.

To verify that everything works as expected, we first run the SVM on the original 100 images

(a) Frame 4: all 4 persons are tracked correctly

(b) Frame 20: all 4 persons are still tracked correctly

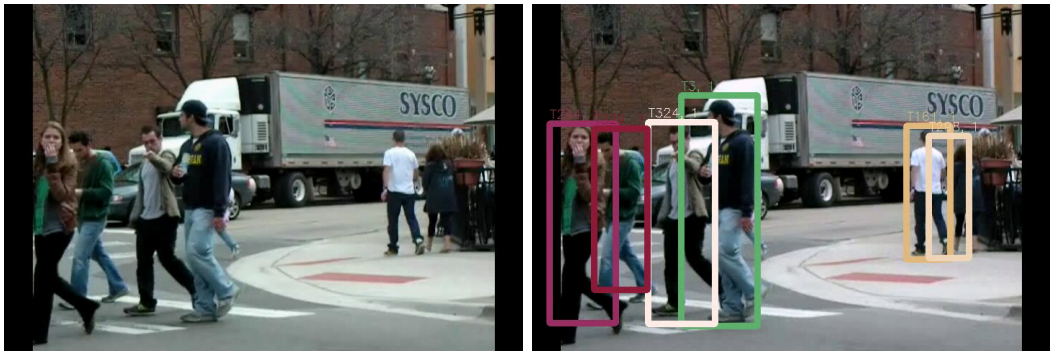
(c) Frame 21: due to partial occlusion, the track of the third person is terminated

(d) Frame 24: there is a new track for the third person, which only lasts 3 frames

(e) Frame 40: the previous third person is fully occluded and no longer tracked

(f) Frame 89: due to clutter, the track of the fourth person is also terminated

(g) Frame 128: all persons are tracked again. The first and fourth person still have their initial track.



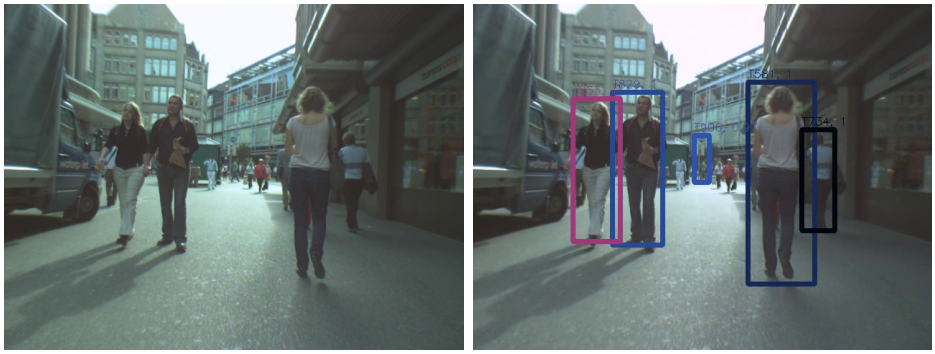
(h) Frame 170: due to clutter and occlusion, the first three persons have received new tracks

Figure 5.4: Tracking errors in sequence 1 of Choi's Collective Activity data set [10]

(a) Frame 1305: reflections in the window are tracked for 11 frames

(b) Frame 1506: an arm is tracked for 2 frames

(c) Frame 1637: part of a wall is tracked in a single frame



(d) Frame 1674: a road sign is tracked for 2 frames

Figure 5.5: Some false tracks in Sequence 2 of the ETH data set [18]

	R	FR	F	FL	L	BL	B	BR	Accuracy
R	21	18	9	5	10	4	2	10	27%
FR	14	24	14	7	2	6	2	10	30%
F	3	5	40	8	2	9	11	1	51%
FL	11	5	14	23	14	2	8	2	29%
L	3	9	2	5	31	18	6	5	39%
BL	4	2	6	4	22	27	12	2	34%
B	1	1	9	13	1	13	36	5	46%
BR	8	3	5	8	6	4	26	19	24%

Table 5.2: Confusion matrix for orientation estimation. The overall accuracy is 34.97%. Actual classes are shown vertically, predicted horizontally.

that were used to generate the codebook. This leads to an almost perfect accuracy of 99.88%. Next we run the SVM on test data that has not been used before. Using 79 randomly selected images for each orientation, we obtain an accuracy of 34.97%.

As can be seen in Table 5.2, many wrong classifications lie in closely related classes. When we consider misclassifications in neighboring classes to be correct (e.g., front-left and front-right are also correct if the true class is front), the accuracy is 67.56%. Interestingly, we observe that mirrored orientations have slightly higher confusions. For example, back-left has an accuracy of 34% while back-right only has 24%.

5.4 Human interaction recognition

We use Choi's Collective Activity data set to experiment with the final interaction recognition. For each pair of targets that is simultaneously on screen, we first compute the temporal overlap and trim the human tracks to that time interval. Next we perform the orientation estimation and form the orientation histograms, as explained in Section 4.4.

Similar to Wang et al., we compute dense trajectories for each track and sample 100 000 random features for each descriptor type. We cluster the features independently for each descriptor type to obtain the bag of words codebooks. For this we run k-means 8 times with $k = 4000$. Since Wang et al. observe that the trajectory descriptor does not improve the overall performance significantly [53], we choose not to use it. We form the final track descriptor by concatenating the three bag of words histograms, leading to a length of 12 000.

As explained in Section 4.6, we construct the feature vector for each pair based on the human tracks, orientation histograms and track descriptors. We only consider pairs that are actually interacting. We train a multiclass SVM classifier with an RBF kernel to distinguish between the 7 interaction classes. We use a grid search with 10-fold cross validation on the parameter space of the SVM to get a good fit. We end up using $C = 6.25$ and $\gamma = 10^{-5}$.

Similar to Choi, we use sequences 1, 2, 4, 5, 6, 7, 8, 10, 11, 26, 28 and 35 for testing and the

	Training data		Test data	
	Absolute	**Relative**	**Absolute**	**Relative**
AP	322	13%	128	19%
LV	248	10%	96	14%
PB	456	18%	84	13%
FE	40	2%	20	3%
WS	508	20%	202	30%
SR	658	26%	84	13%
SS	256	10%	58	9%
Total	2488	100%	672	100%

Table 5.3: Distribution of training and test samples for each class

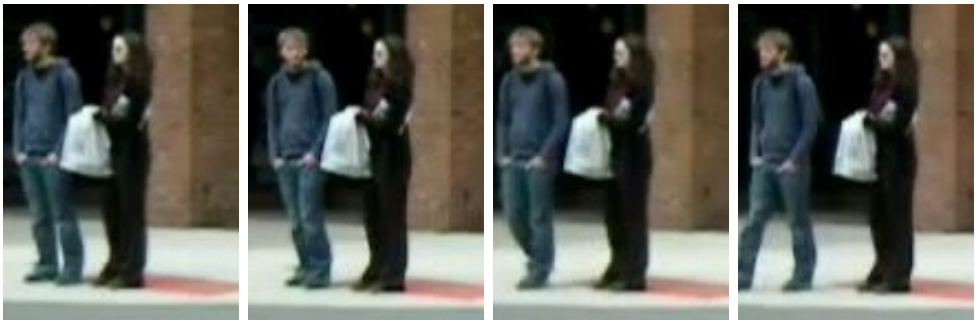
	AP	**LV**	**PB**	**FE**	**WS**	**SR**	**SS**	**Accuracy**
AP	56	2	44	0	14	4	8	44%
LV	0	38	52	0	4	2	0	40%
PB	0	18	46	2	2	12	4	55%
FE	0	0	2	0	0	18	0	0%
WS	10	4	46	0	108	30	4	53%
SR	0	0	0	0	0	84	0	100%
SS	4	0	4	0	14	8	28	48%

Table 5.4: Confusion matrix for interaction recognition using full features. The overall accuracy is 53.57%.

remaining sequences for training. We include a mirrored version of each interaction in our training and test data, which leads to a total of 2488 pairs for training and 672 for testing. Table 5.3 shows the distribution of the training and test samples for all classes.

Table 5.4 give an overview of our results and Figure 5.6 shows some example images. Please note that the results are based on processing in all steps, so errors are propagated through the pipeline. We obtain an overall accuracy of 53.57% using our test data. What immediately stands out is the accuracy of the facing each other (FE) interaction: not even a single interaction is recognized correctly. We expect that this is a direct result of the low number of training examples for this interaction. It is interesting to see that almost all misclassifications of this interaction go to standing in a row (SR), which is conceptually similar. Both interactions do not involve much motion and the targets are standing close to each other. There is however one important difference: when facing each other (FE) the targets have mirrored orientations, while targets that are standing in a row (SR) have the same orientation. Note that the orientation estimation confuses mirrored poses more often.

Another interesting observation is that relatively many interactions are mistakenly classified as passing by (PB). This is the case for approaching (AP), leaving (LV) and walking side by side (WS). All these interactions involve motion and are conceptually close. The remaining interactions do not involve much motion and are almost never misclassified as passing by (PB).



(a) Pair 136: leaving (LV) is misclassified as passing by (PB)

(b) Pair 626: standing side by side (SS) is misclassified as standing in a row (SR)

(c) Pair 1437: correctly classified as standing in a row (SR)

(d) Pair 1981: correctly classified as standing side by side (SS)

Figure 5.6: Some examples of classified interactions between pairs of persons

	Full	No dense	Only dense	Only HOG	Only HOF	Only MBH
AP	44%	44%	6%	0%	5%	14%
LV	40%	35%	0%	0%	0%	0%
PB	55%	48%	14%	31%	31%	43%
FE	0%	10%	40%	0%	10%	0%
WS	53%	51%	54%	69%	48%	43%
SR	100%	90%	88%	95%	1%	92%
SS	48%	38%	34%	14%	31%	38%
Overall	54%	49%	35%	38%	22%	36%

Table 5.5: Overview of accuracies for all combinations of descriptors

5.4.1 Dense trajectory descriptors

In order to understand the influence of the dense trajectory descriptors on the interaction recognition, we perform several tests with various combinations of descriptors. Table 5.5 gives an overview of all results.

We obtain an accuracy of 49.41% on our test data when we do not use any dense trajectories, as can be seen in Table 5.6. Surprisingly, the results are only slightly worse compared to the full features. When we compare both confusion matrices, we see mostly similar errors. The influence of the dense trajectories seems small compared to the other elements of the feature vector.

When we only use dense trajectories, and no other elements of the feature vector, we obtain an accuracy of 34.52%. The confusion matrix can be seen in Table 5.7. The approaching (AP), leaving (LV) and passing by (PB) interactions do not perform well compared to the other interactions. Approaching (AP) and leaving (LV) are often mistaken for walking side by side (WS), which seems intuitive if you do not know the orientations of the targets. We also notice that passing by (PB) is often misclassified as standing in a row (SR). This seems counterintuitive, as we would have expected a significant difference in motion.

Tables 5.8, 5.9 and 5.10 show the results using only dense trajectories based on HOG, HOF and MBH respectively. When we only use HOG we obtain an accuracy 37.80%, with HOF 22.17%, and with MBH 35.56%. It is remarkable how well standing in a row (SR) is recognized using HOG and MBH descriptors, while it completely fails using HOF descriptors. With HOF all standing in a row (SR) interactions are misclassified as walking side by side (WS). This surprises us, given the difference in motion. Although the distance between the targets is constant during both interactions, the walking side by side (WS) interaction contains motion with respect to the background. That is not the case for standing in a row (SR). Most videos do contain quite some camera motion, which might explain the problem.

	AP	LV	PB	FE	WS	SR	SS	Accuracy
AP	56	6	42	0	12	4	8	44%
LV	0	34	58	0	4	0	0	35%
PB	4	22	40	2	0	10	6	48%
FE	2	0	0	2	4	12	0	10%
WS	20	10	46	2	102	20	2	51%
SR	0	0	0	2	6	76	0	90%
SS	2	6	4	8	10	6	22	38%

Table 5.6: Confusion matrix for interaction recognition without dense trajectories. The overall accuracy is 49.41%.

	AP	LV	PB	FE	WS	SR	SS	Accuracy
AP	8	0	12	0	58	48	2	6%
LV	2	0	30	0	46	18	0	0%
PB	2	0	12	10	16	28	16	14%
FE	2	0	0	8	8	2	0	40%
WS	4	4	20	0	110	36	28	54%
SR	0	0	0	0	10	74	0	88%
SS	6	0	6	0	18	8	20	34%

Table 5.7: Confusion matrix for interaction recognition using only dense trajectories. The overall accuracy is 34.52%.

	AP	LV	PB	FE	WS	SR	SS	Accuracy
AP	0	0	8	0	92	22	6	0%
LV	4	0	4	0	74	8	6	0%
PB	2	1	26	0	29	18	8	31%
FE	2	0	10	0	4	0	4	0%
WS	8	2	10	0	140	30	12	69%
SR	0	0	4	0	0	80	0	95%
SS	2	2	10	0	30	6	8	14%

Table 5.8: Confusion matrix for interaction recognition using only HOG trajectories. The overall accuracy is 37.80%.

	AP	LV	PB	FE	WS	SR	SS	Accuracy
AP	6	5	18	0	51	48	0	5%
LV	2	0	24	0	32	38	0	0%
PB	0	2	26	8	6	36	6	31%
FE	2	0	2	2	8	6	0	10%
WS	4	0	18	0	96	46	38	48%
SR	0	0	0	0	83	1	0	1%
SS	6	0	6	0	16	12	18	31%

Table 5.9: Confusion matrix for interaction recognition using only HOF trajectories. The overall accuracy is 22.17%.

	AP	LV	PB	FE	WS	SR	SS	Accuracy
AP	18	2	20	0	21	41	26	14%
LV	12	0	24	0	32	24	4	0%
PB	0	0	36	0	8	24	16	43%
FE	0	0	4	0	0	8	8	0%
WS	6	0	26	0	86	56	28	43%
SR	0	0	0	0	1	77	6	92%
SS	2	0	4	0	2	28	22	38%

Table 5.10: Confusion matrix for interaction recognition using only MBH trajectories. The overall accuracy is 35.56%.

5.4.2 Orientation estimation

We expect that the orientations of the targets play an important role in the interaction recognition. In order to verify this, we do a test without orientation histograms. This leads to an accuracy of 32.74% on the test data, as can be seen in Table 5.11.

Compared to the full features, the results are much worse. The classifier has many difficulties distinguishing between approaching (AP), leaving (LV), passing by (PB) and walking side by side (WS). This seems intuitive, as these interactions are all quite similar, except for the directions the targets move in.

5.4.3 Target distance

Up to this point we did not explicitly use the distance between the targets, but we expect that it may help the classification. Therefore, we compute the Euclidean distance between the centers of the bounding boxes of the targets. Table 5.12 shows the distribution of target distances for all classes.

When we include the target distance in the (full) feature vector, we obtain an accuracy of 51.19%. Table 5.13 gives an overview of the results. The results are quite similar to the results without target distance, but the passing by (PB) interaction is more often misclassified as walking side by side (WS). A reason for this might be the differences in target distance between our training and test data. It might be interesting to look at other methods, for example by compensating for the difference in scale between the targets.

	AP	LV	PB	FE	WS	SR	SS	Accuracy
AP	28	22	38	0	24	10	6	22%
LV	28	18	30	0	6	14	0	19%
PB	13	26	20	0	5	12	8	24%
FE	0	0	2	0	0	18	0	0%
WS	32	10	56	0	74	28	2	37%
SR	0	0	0	0	20	64	0	76%
SS	2	0	18	0	8	14	16	28%

Table 5.11: Confusion matrix for interaction recognition without target orientations. The overall accuracy is 32.74%.

	Training data	Test data	Difference
AP	155	122	−21%
LV	211	223	+6%
PB	133	225	+69%
FE	91	111	+22%
WS	142	100	−30%
SR	218	205	−6%
SS	181	95	−48%
Average	172	150	−13%

Table 5.12: Average target distance for each class. The last column shows the difference between the training and test data.

	AP	LV	PB	FE	WS	SR	SS	Accuracy
AP	54	6	48	0	10	2	8	42%
LV	0	40	48	0	6	2	0	42%
PB	4	16	32	2	14	10	6	38%
FE	0	0	2	0	0	18	0	0%
WS	10	6	50	0	106	26	4	52%
SR	0	0	2	0	0	82	0	98%
SS	2	0	6	0	12	8	30	52%

Table 5.13: Confusion matrix for interaction recognition using target distance. The overall accuracy is 51.19%.

Chapter 6

Discussion

In this project we have analyzed the automatic recognition of human interactions from video. We have focused on interactions in urban settings that take place between pairs of persons. A pipeline for human interaction recognition was constructed, which consists of several parts. Part based models are used to detect every person in every frame of the video. Human detections are tracked over time and the orientations of the persons are estimated. Pairs of people are formed and dense trajectories are computed for each human track. The final interaction recognition is done using a multiclass classifier for each pair of persons.

We have based our approach on the work by Patron-Perez et al. [36, 37], but there are some key differences. While Patron-Perez et al. restricted their approach on videos containing only upper bodies, we focused on more general videos containing full bodies. Our setting has a larger variety of poses and the camera conditions are more challenging. Because of these differences, we have used full body human detection, tracking and orientation estimation. We do not compute intermediate action classes, but use the dense trajectories directly for classification. We classify each pair of tracks individually, while Patron-Perez et al. use a structured learning approach that incorporates all persons in a frame.

Each part of the pipeline was tested independently to learn its relative strengths and weaknesses. We have used the human detection and tracking approach by Choi et al. "as is" [7, 8]. We have shown that the implementation leads to good results on the ETH data set [18], but that the results on the Collective Activity data set [10] leave room for improvement. We expect that the results can be improved by optimizing the parameters for this data set.

Our orientation estimation approach leads to an accuracy of 34.97% on test data, while we try to classify 8 classes of orientations. We observe that many wrong classifications lie in closely related classes.

While we try to classify 7 classes of person-to-person interactions, we obtain an overall accuracy of 53.57% on test data. We immediately observe that the facing each other (FE) interaction is almost always misclassified as standing in a row (SR), probably due to a lack

of training data. The standing in a row interaction seems a logical alternative. We also observe that the approaching (AP), leaving (LV) and walking side by side (WS) interactions are quite often misclassified as passing by (PB). This also seems intuitive, as these interactions all involve motion and are conceptually similar.

6.1 Research questions

6.1.1 Use of dense trajectory descriptors

Which type of dense trajectory descriptor is most suited for interaction classification? Does this depend on the interaction class?

We have performed tests with various types of trajectory descriptors in order to understand their influence on the interaction recognition. First we have performed the interaction recognition without any dense trajectory descriptors. To our surprise, the results were only slightly worse compared to the full features. The influence of the dense trajectories therefore seems small compared to the other elements of the feature vector.

While using only dense trajectories, and no other elements of the feature vector, we obtain an accuracy of 34.52% on test data. The approaching (AP) and leaving (LV) interactions are often mistaken for walking side by side (WS), which seems intuitive without the target orientations. We observe that the passing by (PB) interaction is often misclassified as standing in a row (SR), which seems counterintuitive because of the difference in motion.

We have also independently analyzed the performance of the different types of dense trajectory descriptors. Using HOG we obtain an accuracy of 37.80% on test data, using HOF 22.17% and using MBH 35.56%. The confusion matrices are quite similar, but there is one important difference. To our surprise, the standing in a row (SR) interaction is recognized quite well using HOG and MBH, but it completely fails with the HOF descriptor.

Given the other elements of our feature vector, we conclude that the dense trajectories have limited influence on the performance of the interaction recognition. We do notice some differences between the different types of descriptors, but we do not find any significant patterns.

6.1.2 Use of orientation estimation

How does the use of orientation estimation affect the interaction classification performance?

We have tested our interaction recognition pipeline both with and without the use of orientation estimation. As expected, orientations play an important role in the performance. If we do not include the orientation histograms in the feature vector, we obtain an accuracy of 32.74% on test data. This is a significant decline of more than 20%.

Without the orientation histograms, the classifier is not able to properly distinguish between the approaching (AP), leaving (LV), passing by (PB) and walking side by side (WS) interactions. This seems intuitive, since the interactions are similar except for the directions the targets move in.

6.1.3 Use of distance between the targets

How does the use of distance between the targets affect the interaction classification performance?

We have done tests with and without the explicit use of the distance between the targets. When we include the distance between the targets in the feature vector, we notice that the passing by (PB) interaction is quite often misclassified as walking side by side (WS). A reason for this might be the differences in target distance between our training and test data. Overall, the results are quite similar and we conclude that the explicit use of the distance between the targets does not have much influence on the interaction recognition performance.

6.2 Suggestions for future work

Many possibilities exist for future work. We will discuss some interesting directions.

Our pipeline is quite flexible, in the sense that any full body human detection, tracking and orientation estimation system can be used. These are all active research fields on their own. An improvement to any of these parts will contribute to the overall performance of the human interaction recognition pipeline. Similarly, we expect that the use of additional information will improve the overall performance of the pipeline. Hands play an important role in many interactions, so explicit hand tracking might be beneficial.

We observe that the dense trajectory descriptors have a rather limited influence on the interaction recognition performance. It might be interesting to experiment with different types of feature descriptors. We would like to mention that Wang et al. have continued their work on dense trajectories and have presented a new approach that uses Fisher vectors instead of bag of words [53]. They also use RootSIFT instead of L_2-normalization and estimate the global background motion in order to further improve the results.

Although our results show that the explicit use of the distance between the targets does not have much influence on the interaction recognition performance, it may be interesting to further investigate this. We expect that the results can be improved by compensating for the difference in scale between the targets. While we have only used the distance between the targets, Patron-Perez et al. divide the spatial area between the targets into discrete regions based on their horizontal and vertical positions in the frame [36, 37]. Such a representation is likely to include more information than our approach.

Acknowledgments

This master thesis has been carried out at the Interaction Technology Group of the Department of Information and Computing Sciences of Utrecht University. The project is part of the "Sensing Emotion in Video" project of the COMMIT program, which has the objective to perform research on aspects of the extraction of distinguishing features to describe and categorize motion styles, arousal, happiness, etc.

I would like to thank my supervisors Coert J. van Gemeren, MSc, dr. ir. Ronald W. Poppe and prof. dr. Remco C. Veltkamp for assisting me throughout the project. I would also like to thank my former supervisor dr. Robby T. Tan for introducing me to this field of research.

Bibliography

[1] J. K. Aggarwal and M. S. Ryoo. Human activity analysis: A review. *ACM Computing Surveys*, 43(3):16:1–16:43, April 2011.

[2] T. Bagautdinov, A. Alahi, F. Fleuret, P. Fua, and S. Savarese. Social scene understanding: End-to-end multi-person action localization and collective activity recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '17. IEEE Computer Society, 2017.

[3] B. Benfold and I. Reid. Colour invariant head pose classification in low resolution video. In *British Machine Vision Conference*, BMVC '08. BMVA Press, September 2008.

[4] B. Benfold and I. Reid. Guiding visual surveillance by tracking human attention. In *British Machine Vision Conference*, BMVC '09. BMVA Press, September 2009.

[5] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc., 2006.

[6] J. Bouguet. Pyramidal implementation of the Lucas Kanade feature tracker. *Intel Corporation, Microprocessor Research Labs*, 2000.

[7] W. Choi, C. Pantofaru, and S. Savarese. A general framework for tracking multiple people from a moving camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(7):1577–1591, 2013.

[8] W. Choi and S. Savarese. Multiple target tracking in world coordinate with single, minimally calibrated camera. In *European Conference on Computer Vision*, ECCV '10, pages 553–567. Springer-Verlag, 2010.

[9] W. Choi and S. Savarese. A unified framework for multi-target tracking and collective activity recognition. In *European Conference on Computer Vision*, volume 4 of *ECCV '12*, pages 215–230. Springer-Verlag, 2012.

[10] W. Choi, K. Shahid, and S. Savarese. Collective activity dataset. http://www.eecs.umich.edu/vision/activity-dataset.html, 2009.

[11] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, May 2002.

[12] Intel Corporation, Willow Garage, and Itseez. OpenCV (open source computer vision). http://opencv.willowgarage.com.

[13] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '05, pages 886–893. IEEE Computer Society, June 2005.

[14] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In *European Conference on Computer Vision*, volume 3 of *ECCV '06*, pages 428–441. Springer-Verlag, 2006.

[15] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society Series B (Statistical Methodology)*, 39(1):1–38, January 1977.

[16] Z. Deng, A. Vahdat, H. Hu, and G. Mori. Structure inference machines: Recurrent neural networks for analyzing relations in group activity recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '15. IEEE Computer Society, November 2015.

[17] K. B. Duan and S. S. Keerthi. Which is the best multiclass SVM method? an empirical study. In *International Conference on Multiple Classifier Systems*, MCS '05, pages 278–285. Springer-Verlag, 2005.

[18] A. Ess, B. Leibe, K. Schindler, and L. van Gool. A mobile vision system for robust multi-person tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '08. IEEE Computer Society, June 2008.

[19] M. Everingham, J. Sivic, and A. Zisserman. Taking the bite out of automated naming of characters in TV video. *Image and Vision Computing*, 27(5):545–559, April 2009.

[20] M. Everingham, A. Zisserman, C. K. I. Williams, and L. Van Gool. The PASCAL visual object classes challenge 2006 (VOC2006) results. http://www.pascal-network.org/challenges/VOC/voc2006/results.pdf.

[21] G. Farneback. Two-frame motion estimation based on polynomial expansion. In *Scandinavian Conference on Image Analysis*, SCIA '03, pages 363–370. Springer-Verlag, 2003.

[22] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, September 2010.

[23] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, January 2005.

[24] V. Ferrari, M. Marin-Jimenez, and A. Zisserman. Progressive search space reduction for human pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '08, pages 1–8. IEEE Computer Society, June 2008.

[25] V. Ferrari, M. Marin-Jimenez, and A. Zisserman. Pose search: Retrieving people using their pose. In *IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '09, pages 1–8. IEEE Computer Society, 2009.

[26] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1–3):185–203, August 1981.

[27] Z. Khan, T. Balch, and F. Dellaert. MCMC-based particle filtering for tracking a variable number of interacting targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11):1805–1918, November 2005.

[28] A. Klaser, M. Marszalek, C. Schmid, and A. Zisserman. Human focused action localization in video. In *European Conference on Computer Vision*, volume 6553 of *ECCV '10*, pages 219–233. Springer-Verlag, 2012.

[29] Y. Kong and Y. Jia. A hierarchical model for human interaction recognition. In *IEEE International Conference on Multimedia and Expo*, pages 1–6, July 2012.

[30] H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quaterly*, 2(1–2):83–97, 1955.

[31] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '08, pages 1–8. IEEE Computer Society, June 2008.

[32] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2), November 2004.

[33] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence*, IJCAI '81, pages 674–679. Morgan Kaufmann Publishers Inc., April 1981.

[34] S. Park and J. K. Aggarwal. Simultaneous tracking of multiple body parts of interacting persons. *Computer Vision and Image Understanding*, 102(1):1–21, April 2006.

[35] S. Park and M. M. Trivedi. Understanding human interactions with track and body synergies (TBS) captured from multiple views. *Computer Vision and Image Understanding*, 111(1):2–20, July 2008.

[36] A. Patron-Perez, M. Marszalek, I. Reid, and A. Zisserman. Structured learning of human interactions in TV shows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(12):2441–2453, December 2012.

[37] A. Patron-Perez, M. Marszalek, A. Zisserman, and I. Reid. High five: Recognising human interactions in TV shows. In *British Machine Vision Conference*, BMVC '10, pages 50.1–50.11. BMVA Press, 2010.

[38] A. Patron-Perez, M. Marszalek, A. Zisserman, and I. Reid. TV human interaction dataset. http://www.robots.ox.ac.uk/~vgg/data/tv_human_interactions, 2010.

[39] R. Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976–990, June 2010.

[40] A. Prest, V. Ferrari, and C. Schmid. Explicit modeling of human-object interactions in realistic videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(4):835–848, April 2013.

[41] S. J. D. Prince. *Computer Vision: Models, Learning and Inference*. Cambridge University Press, 2012.

[42] F. Sener and N. Ikizler-Cinbis. Two-person interaction recognition via spatial multiple instance embedding. In *Visual Communication and Image Representation*, volume 32, pages 63–73. Academic Press, Inc., October 2015.

[43] J. Shi and C. Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '94, pages 593–600. IEEE Computer Society, June 1994.

[44] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering objects and their location in images. In *IEEE International Conference on Computer Vision*, volume 1 of *ICCV '05*, pages 370–377. IEEE Computer Society, 2005.

[45] R. Szeliski. *Computer Vision Algorithms and Applications*. Springer Science & Business Media, 2011.

[46] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical report, Carnegie Mellon University, April 1991.

[47] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *International Conference on Machine Learning*, ICML '04, pages 104–. ACM, 2004.

[48] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, December 2005.

[49] C. J. van Gemeren, R. W. Poppe, and R. C. Veltkamp. Spatio-temporal detection of fine-grained dyadic human interactions. In *International Workshop on Human Behavior Understanding*, volume 9997, pages 116–133. Springer International Publishing, October 2016.

[50] C. J. van Gemeren, R. T. Tan, R. W. Poppe, and R. C. Veltkamp. Dyadic interaction detection from pose and flow. In *International Workshop on Human Behavior Understanding*, volume 8749, pages 101–115. Springer International Publishing, September 2014.

[51] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., 1995.

[52] H. Wang, A. Klaser, C. Schmid, and C. L. Liu. Action recognition by dense trajectories. In *IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '11, pages 3169–3176. IEEE Computer Society, 2011.

[53] H. Wang, D. Oneata, J. Verbeek, and C. Schmid. A robust and efficient video representation for action recognition. *International Journal of Computer Vision*, 119(3):219–238, 2016.

[54] H. Wang, M. M. Ullah, A. Klaser, I. Laptev, and C. Schmid. Evaluation of local spatio-temporal features for action recognition. In *British Machine Vision Conference*, BMVC '09. BMVA Press, September 2009.

[55] F. Yin, D. Makris, and S. A. Velastin. Performance evaluation of object tracking algorithms. *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, October 2007.