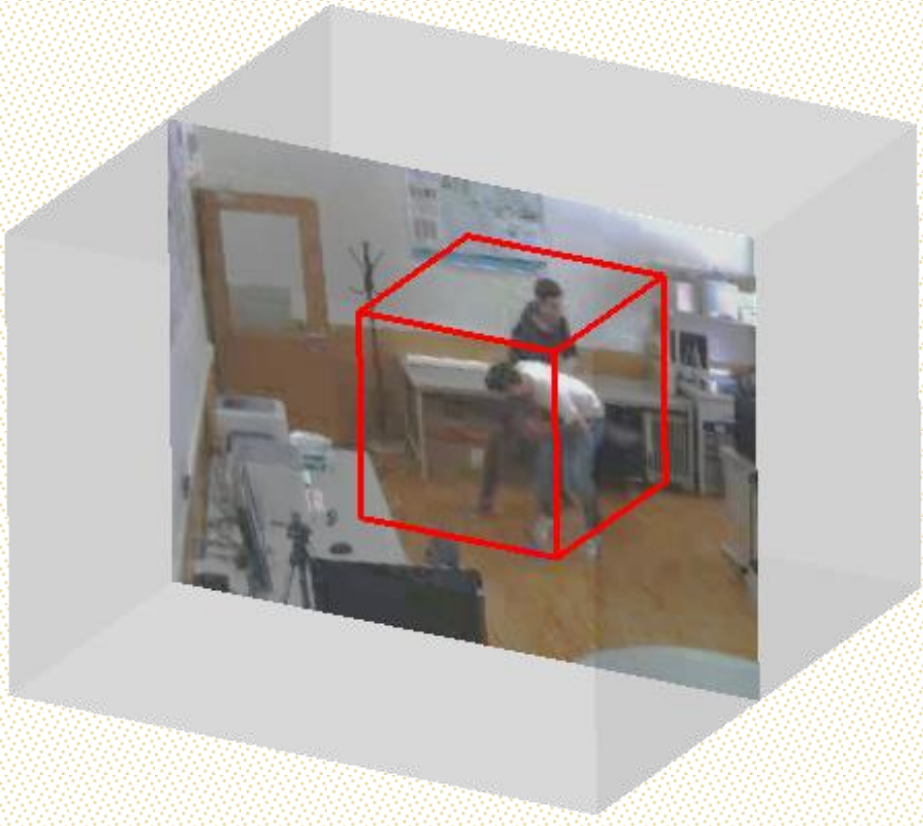

João Faro

Detection of Unusual Events in Videos via Dynamic Sparse Coding

September 2013

· U C ·

Universidade de Coimbra

FCTUC

University of Coimbra
Faculty of Sciences and Technology
Department of Electrical and Computer Engineering of University of Coimbra

Detection of Unusual Events in Videos via Dynamic Sparse Coding

João António Pereira Faro

Juri:

Prof. Doutor Helder de Jesus Araújo

Prof. Doutor Jorge Manuel Moreira de Campos Pereira Batista

Prof. Doutor João Pedro de Almeida Barreto

Thesis submitted in partial fulfillment of the requirements for the Master's Degree in the Department of Electrical and Computer Engineering.

September, 2013

This thesis was done under the supervision of Dr. Jorge Manuel Pereira Batista

Department of Electrical and Computer Engineering, University of Coimbra

Acknowledgements

First of all, I would like to thank Doctor Jorge Batista for his confidence in my capabilities to do this dissertation and for all the help given throughout the past months.
I would like to thank all my colleagues in the Computer Vision Laboratory for their availability in aiding me.
Finally, I thank to all my friends and family, specially to my parents, brother and girlfriend, for their support and patience along these past months.

Resumo

Os sistemas de detecção de eventos não usuais em tempo-real têm como maior desafio a volatibilidade da definição de normalidade de um evento e a necessidade de avaliar o mesmo de forma rápida. Numa altura em que a video-vigilância é cada vez mais utilizada como forma de aumentar a segurança, a capacidade de detecção de situações anómalas de forma eficiente e adaptativa é uma grande vantagem. Os sistemas comerciais existentes requerem um processo de treino inicial exaustivo, ou então necessitam da definição inicial do que são eventos usuais, não se adaptando a possíveis alterações de comportamento do meio em observação. Nesta dissertação é apresentado um sistema que se baseia em codificação esparsa dinâmica para reconstruir sinais a partir de um dicionário aprendido apenas de eventos considerados normais. Assumindo que eventos usuais são mais fáceis de reconstruir a partir de um dicionário de eventos usuais, este algoritmo permite a reconstrução dos eventos e classificação dos mesmos usando codificação esparsa, bem como a actualização online do dicionário para incorporar os novos eventos observados. Este processo permite uma adaptação constante às mudancas comportamentais do meio. A obtenção dos sinais caracterizadores de cada evento é feita através de descritores espaço-temporais calculados em regiões de interesse (ROIs) centradas em pontos de interesse da imagem. O trabalho desenvolvido inclui o uso de diferentes descritores espaço-temporais como é o caso dos Histogramas de Gradientes Orientados/Histogramas de fluxo óptico (HOG/HOF), Gradientes Espaço-Temporais Orientados Direcionalmente (HOG-NSP) e Descritores de Covariâncias Espaço-Temporais (COV3D). Os resultados obtidos mostram que com o método implementado é possível obter um detector de eventos não usuais capaz de avaliar correctamente a normalidade de um evento em diferentes condições. Foi também concluído que o descritor HOG/HOF é o que apresenta melhores resultados, apesar de o descritor HOG-NSP ser o mais indicado para soluções em tempo-real.

Palavras-chave: Classificação de Eventos, Representação Esparsa, Codificação Esparsa, Gradiente, Fluxo Óptico, Matriz de Covariâncias.

Abstract

Real-time unusual event detection systems has been a difficult challenge due to the volatility of the definition for normal activity and the need to classify them in a short period of time. At a time when video surveillance has increasingly been used to enhance the security, the capacity to detect anomalous situations in an efficient and adaptive way is a big advantage. The existing unusual event detectors either need an exhaustive initial training or require an initial definition of what an usual event may look like, which fails to adapt to behavioural changes. This thesis presents a system based on dynamic sparse coding which reconstructs signals from dictionary learned only from event considered usual. Based on an intuition that usual events are more likely to be reconstructible from an event dictionary, this algorithm performs the reconstruction and classification of events using sparse coding as well as an online dictionary update to incorporate newly observed events. To acquire the features of an event spatio-temporal descriptors are computed in different regions of interest (ROI) centered in interest points of the image. For this purpose, three different spatio-temporal descriptors were used: Histogram of Oriented Gradients/Histogram of Optical Flow (HOG/HOF), Directional Space-Time Oriented Gradients (HOG-NSP) and Spation-Temporal Covariance Descriptors (COV3D). The experimental results show that that with the methodology implemented in this dissertation it is possible to achieve an unusual event detector capable of evaluate the normality of an event in different environments. Also, it has been concluded that the HOG/HOF descriptor yields the best results. Despite this, the HOG-NSP descriptor has proved to be more appropriate for real-time solutions.

Keywords: Event Classification, Sparse Representation, Sparse Coding, Gradients, Optical Flow, Covariance Matrix.

Contents

List of Figures

List of Tables

List of Acronyms

HOG	Histogram of Oriented Gradients
HOF	Histogram of Optical Flow
HOG/HOF	Histogram of Oriented Gradients/Histogram of Optical Flow
HOG-NSP	Directional Space-Time Oriented Gradients
BoF	Bag of Features
RCM	Region Covariance Matrix
COV3D	Spation-Temporal Covariance Descriptors
ROI	Region of Interest
STIP	Spatio-Temporal Interest Point
DSI-STIP	Dense and Scale-Invariant Spatio-Temporal Interest Point
SIFT	Scale-Invariant Feature Transform
SURF	Speeded Up Robust Features
HOG3D	3D Histogram of Oriented Gradients
LBP-TOP	Local Binary Pattern on Three Orthogonal Planes
CD	Correct Unusual Event Detection
FA	False Alarm
MISS	Missed Unusual Event Detection

Chapter 1

Introduction

Computers play a fundamental role in our lives. They perform repetitive and mathematically complex tasks more efficiently and more accurately than humans. With growing number of surveillance cameras there is an increased need for automated surveillance systems. Thus one goal of computer vision and machine learning researchers has been to grant computers with the ability to analyse and interpret behaviour in real-time videos. Of the many possible tasks, detecting unusual events from video sequences is of considerable practical importance, leading to a significantly improvement in the efficiency of video analysis, saving human attention for only the most salient contents. Such a capability would have many applications such as houses and stores surveillance, highway and subway monitoring, psychology, etc. However, most of these systems are incapable to change their knowledge of what an usual may look like and take too much time to determine the normality of an event. The main goal of this thesis is to study and evaluate a solution that can easily adapt to behavioural changes like the speed increase of people in a subway or the formation of queues in the highway at peak times, classifying the normality of events efficiently. Furthermore, response time is also considered as an important factor. While in some cases the response time is not a critical factor, there are cases where the consequences of a delayed response can be catastrophic, like when a person fall into the subway line or in traffic accidents. Thus, in this work will be presented a system with different algorithms that when combined will contribute to minimize the above-mentioned problems, generating an efficient and unsupervised unusual event detector with no prior models of what could be an usual event, that runs faster than the others and in a cheap equipment.

1.1 Previous and Related Works

Recently, there has been growing interest in developing systems that evaluate behavior in order to classify their normality.

Initial approaches like [4, 5], tried to classify events in different classes like *"walk"* or *"run"*, considering as unusual the ones that doesn't belong to any class. These approaches fail when object detection, tracking or recognition do not work well, especially in crowded scenes.

Some systems [6, 7], model primitive events, such as *"move"*, *"stop"* or *"turn right"*, and use these primitives to model complicated activities. These primitives are learned from labeled training examples. However, when switching the scene, all primitives must be re-learned. Furthermore, in crowded scenes with occlusions it is complicated to obtain these primitives.

Different approaches [8, 9], directly uses motion feature vectors instead of tracks to a describe video clip. These approaches treat a video clip as an integral entity and classify the whole clip as normal or abnormal. This type of systems are often used to simple datasets where there is only one kind of event in a video clip and can't be used in a real time event detector.

Boiman *et. al* [10] proposes a database indexing algorithm where the new observed data is composed using spatio-temporal patches extracted from previous visual examples. Regions composed by large amounts of data from the data the database are considered normal. Although this algorithm shows good performance, it faces scalability issues as memory and time problems.

In order to reduce memory and scalability issues, Piotr Dollár *et al.* [11] proposed a sparse representation of behaviour through the characterization of cuboids, each one centered in a Spatio-Temporal Interest Point (STIP). These interest points are normally associated to areas with spatially distinguishing features. Different interest point detectors have been implemented in order to find these regions of interest. [12] demonstrated that Harris detector was the one who had a better performance. An extension of Harris detector to the 3D has been proposed by [13]. The basic idea is to find regions containing a reversing in the direction of the gradient (spatio-temporal corners). This approach have shown to be very effective at detecting spatio-temporal corners. Despite this, spatio-temporal corners are insufficient to descriminate some behaviors, like a spinning wheel for example. Later, Willems *et al.* [14] introduced the Dense and Scale-Invariant Spatio-Temporal Interest Point (DSI-STIP).

However this detector detect too many interest points and require the full video analysis to obtain them. A simple spatio-temporal interest point detector was proposed by [11], who resort to 2D Gaussian filters and 1D Gabor filters to produce a detector tuned to fire to periodic motions and also spatio-temporal corners. Other approaches like [15, 16, 17] also shown good performance using sparse representation for action recognition. Sparse representations provide a compact video representation and tolerance to background clutter, occlusions and scale changes.

To describe a spatio-temporal region of interest, different descriptors have been proposed over the last years. For example [18] proposed Scale-Invariant Feature Transform (SIFT) descriptor, [19] proposed Speeded Up Robust Features (SURF), [20] introduced HOG descriptor. [21] proposed Principal Components Analysis - Scale-Invariant Feature Transform (PCA-SIFT), [22] proposed a Spatio-Temporal Descriptor based on 3D Gradients, the 3D Histogram of Oriented Gradients (HOG3D) descriptor and [23] introduced the Histogram of Oriented Gradients/Histogram of Optical Flow (HOG/HOF) descriptor. [24] tested some of the most widely used and concluded that the HOG/HOF descriptor introduced by Laptev *et al.* [23] showed the best results.

Different methods have been proposed as a solution to evaluate the normality of events based on their sparse representation. For example, [11] creates cuboid prototypes from the training data and evaluate events based on the Euclidean distance between the cuboids that characterizes an event and the cuboid prototypes. Other approaches, like [23] resort to the *k-means* algorithm to build a spatio-temporal bag-of-features. Zhao *et al.* [25] proposed a different approach, that evaluate events based on a sparse coding formulation. Recently, sparse coding [26] has become popular in computer vision and has shown promising results when applied to natural images, video and speech [27, 28, 29, 30]. As regards to event detection, [31] uses sparse coding to reconstruct dynamic textures described by local binary patterns from three orthogonal planes. However, this method is applied in a dense form, increasing time complexity. The methodology presented in this work follows the line of reasoning of the one proposed by Zhao *et al.* [25]. In addition, spatio-temporal descriptors presented in [3] and [32] were implemented.

1.2 Thesis Description

The main idea behind this dissertation is to detect events and classify them as usual or unusual based on the reconstruction vectors obtained from the sparse coding formulation.

Each event is represented by a group of spatio-temporal volumes (cuboids), each one centered in an interest point. To obtain these points, a response function is calculated in every frame. The response function is calculated by the application of a Gaussian filter applied along the spatial dimensions and a quadrature pair of Gabor filters applied along the temporal axes. At each interest point (local maxima of the response function above) a cuboid is extracted and described as a vector computed with one of the following spatio-temporal descriptors: HOG/HOF, HOG-NSP and COV3D.

In the initial portion of the video these descriptors are used to learn the dictionary using sparse coding, assuming that an unusual event is unlikely to occur in the small initial portion of the video. In the rest of the time, the descriptors obtained are reconstructed from the dictionary using sparse coding.

Given a dictionary of bases corresponding to usual events, an usual event should be reconstructible from a small number of such bases. On the other hand, an unusual event is either not reconstructible with a small error, or even if it was it will be necessary a large number of bases of the dictionary. Finally, the dictionary is updated with the newly observed event using an algorithm based on stochastic approximations. Figure 1.1 illustrates, step by step, how the algorithms are ordered.

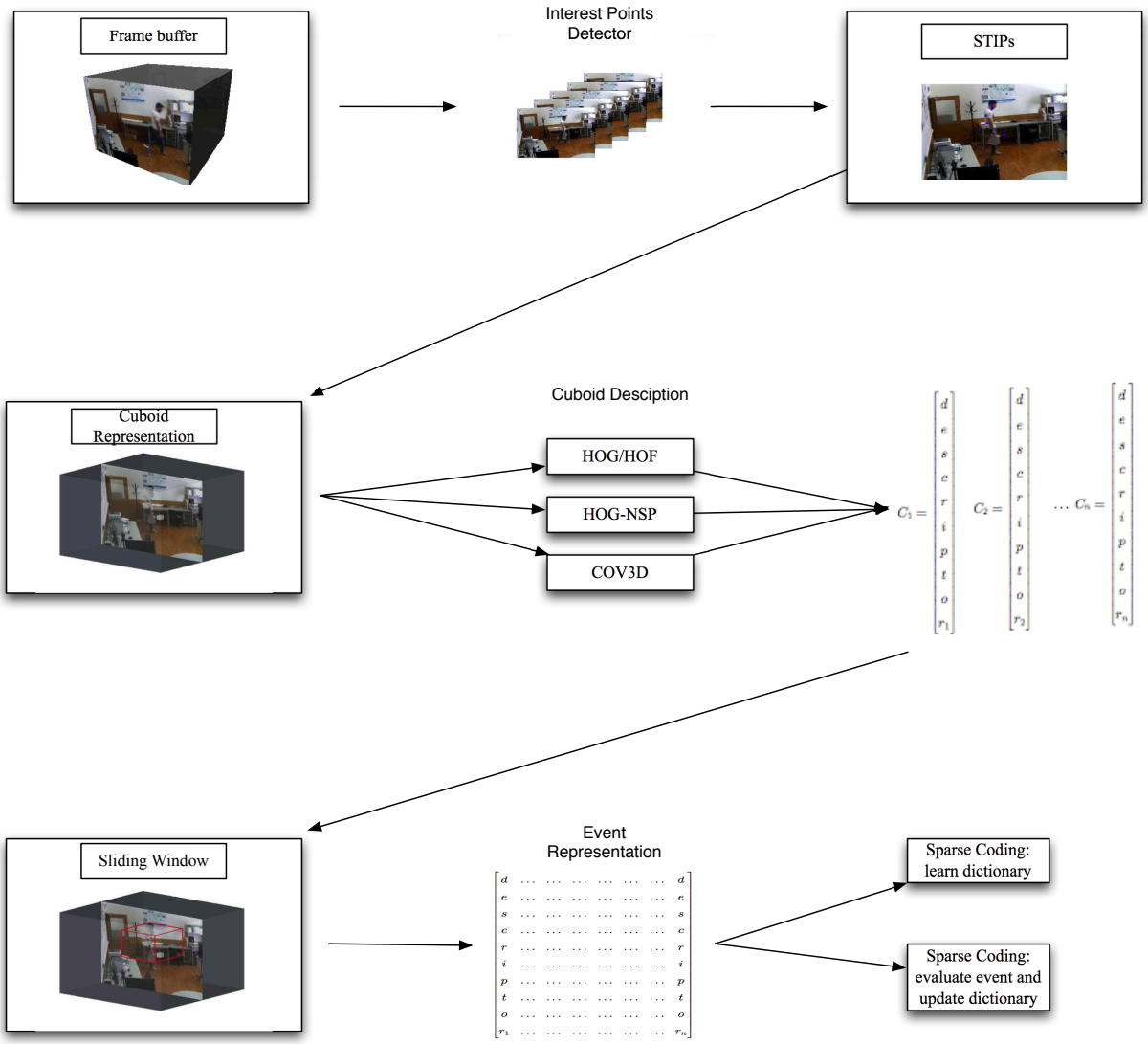

Figure 1.1: Scheme of the system developed.

1.2.1 Work Environment

This work was developed in C++, on UNIX operating system (Ubuntu) with OpenCV 2.3.1 installed. All speed tests were carried with a 2.3 GHz Intel Core i5 with 4GB RAM. A linear algebra library called Armadillo[1] was used.

1.3 Overview

The organisation of this dissertation was done with the objective of introduce the procedures in the same order as the unusual event detector applies them, thus giving the reader an easier perception of the application structure. Chapter 1 introduces the topic, presenting

[1] http://arma.sourceforge.net/

the research already developed in this area as well as its importance and main applications. Chapter 2 explains the procedures to locate different events in a video sequence. In Chapter 3 are presented three different techniques to describe the events in order to differentiate each one of them. Chapter 4 presents the model used to learn what is considered an usual event as well as the method used to classify events as usual or unusual. In each one of these are presented not only the exploited techniques but also the theoretical background used to reach them. In Chapter 5 are presented the experimental results of the work developed. Lastly, Chapter 6 gives important conclusions as well as the direction of future work.

Chapter 2

Event Detection

The representation of events in videos can be separated in two different main approaches: dense and sparse. Generally, dense methods are conceptually simple to implement. Some dense approaches treat a video clip as an integral entity and classify the whole clip as normal or abnormal which is not practical when dealing with stream videos. Other approaches use a segmentation step to locate events. However, for complex scenes with occlusions and multiple events, it is difficult to achieve an efficient segmentation. On the other hand, sparse approaches search for regions with spatially distinguishing features that can be used to describe an event in a succinct way, which reduces the amount of memory needed to store the event and making it more robust against occlusions. Thus, in this work, an event is described in a sparse fashion, represented by a group of cuboids. A cuboid is a spatio-temporal volume that contains the data of regions with spatially distinguishing features. More concretely, each cuboid is centered in STIP. To obtain these STIPs, different interest point detectors have been proposed. For example, Laptev et al. [13] proposed a space-time interest point detector that have been widely used for action recognition [17, 23]. This detector is an extension of the Harris corner detector to the 3D case, that searches for spatio-temporal corners. However, the interest points detected are usually quite sparse. Another interest point detector, the DSI-STIP detector has been introduced by [14]. Although, the interest points detected by DSI-STIP are quite dense and require the entire video to be obtained. In this work, the STIPs are extracted from a response function calculated by the application of an 2D Gaussian smoothing kernel along the spatial dimensions and a quadrature pair of 1D Gabor filters applied temporally, as introduced by Dollár *et al.* [11]. For that, it is assumed a stationary camera. This chapter is organized as follows: Section 2.1 presents the response function used to obtain the STIPs, while Section 2.2 gives an theoretical explanation of

Gabor filters, used to calculate the response function and, finally, Section 2.3 describe the cuboid extraction and how they are grouped to represent an event.

2.1 Response Function

As mentioned before, a representation based on interest points have several advantages, like the amount of memory used and the robustness against occlusions. Moreover, the interest point representation eliminates the need of segmentation and tracking required by many other related approaches and is still robust with respect to different subjects, lighting conditions and scale variations [11].

Spatio-temporal interest point are extracted from a response function, calculated by the application of different linear filters. Each STIP is obtained from a local maxima of the response function. Therefore, areas with spatially distinguishing features should induce the higher responses.

The response function R has the form:

$$R = \sum_{t=-\tau\backslash 2}^{\tau\backslash 2} (I_t * g * h_{ev})^2 + (I_t * g * h_{od})^2 \tag{2.1}$$

where $g(x, y; \sigma)$ is the 2D Gaussian smoothing kernel applied along the spatial dimensions, and h_{ev} and h_{od} represents a quadrature pair of 1D Gabor filters applied along time dimension and defined as:

$$h_{ev} = -\cos(2\pi tw)e^{-t^2/\tau^2} \tag{2.2}$$

and

$$h_{od} = -\sin(2\pi tw)e^{-t^2/\tau^2} \tag{2.3}$$

where $w = 4/\tau$. σ and τ represents the spatial and temporal scale of the detector, respectively.

Figure 2.1 shows an example of a response function calculated for a frame extracted in an outdoor environment, where the highest peaks corresponds to a car hitting another car.

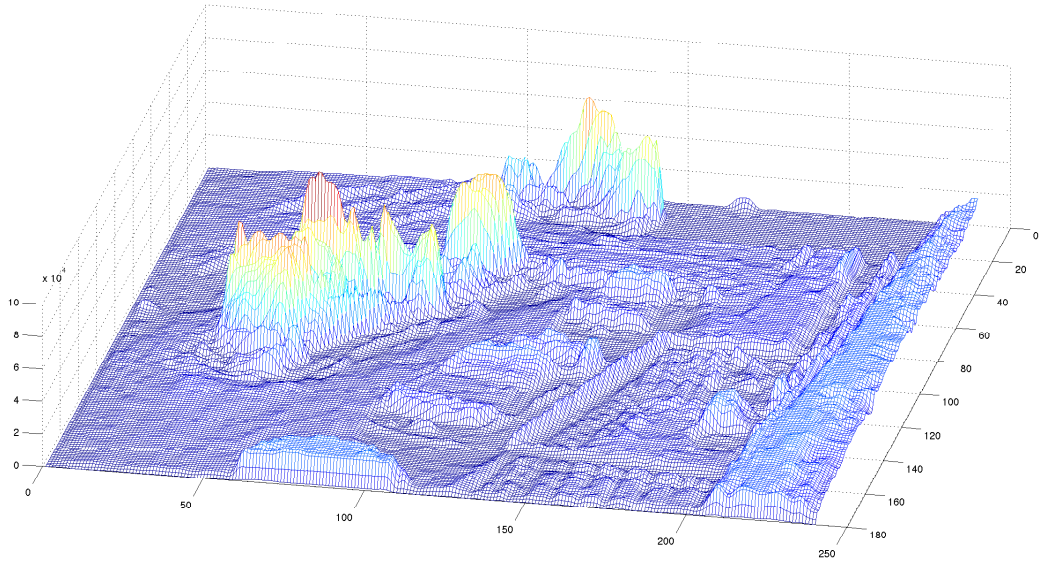

Figure 2.1: Response function for a frame extracted in an outdoor environment.

The STIPs obtained by calculating the local maxima of the chart shown in Figure 2.1 are plotted in the correspondent frame, as shown by Figure 2.2.

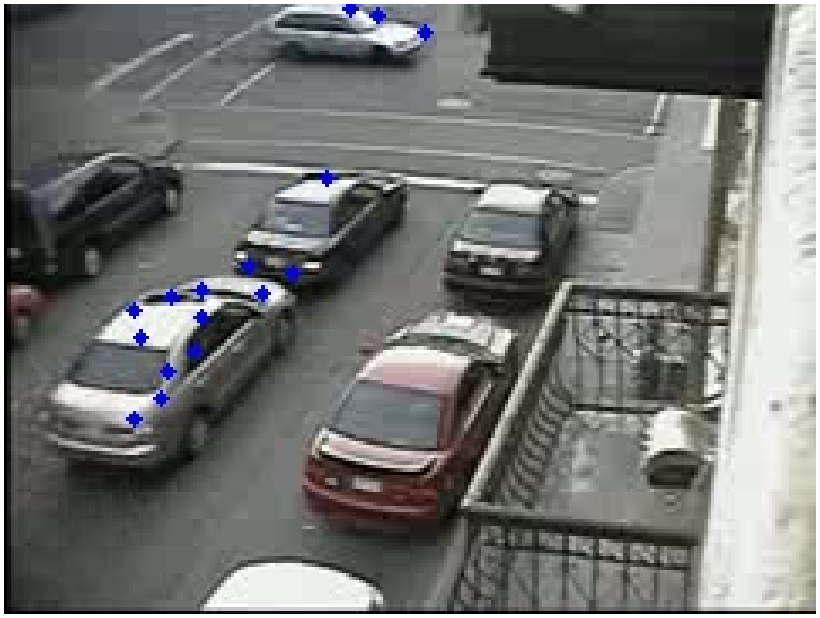

Figure 2.2: STIPS extracted from the response function of Figure 2.1.

The detector is tuned to induce higher responses whenever variations in local image intensities contain periodic frequency motions components. Despite this, the detector also responds strongly to a range of other motions, like spatio-temporal corners [11].

2.2 Gabor Filters

A Gabor filter is a linear filter whose impulse response is defined by a harmonic function multiplied by a Gaussian function. Gabor Filters became popular because their frequency and orientation representations are similar to the characteristics of certain cells in the visual cortex of mammals and have been used in many applications, such as texture segmentation, fractal dimension management, document analysis, edge detection, retina identification, image coding and image representation. In addition, these filters have some interesting mathematical properties. Due to the multiplication-convolution property (convolution theorem), the Fourier transform of a Gabor filter's impulse response is the convolution of the Fourier transform of the harmonic function and the Fourier transform of the Gaussian function [33].

For the purpose of this work, a quadrature pair of $1D$ Gabor filters are used. These filters can be used as excellent band-pass filters for one-dimensional signals. A quadrature pair is a set of two linear operators with the same amplitude response but phase responses shifted by 90°. Thus, complex Gabor filter has two out of phase filters continently allocated in the real and complex part of a complex function, as follows:

$$g(x) = g_e(x) + i\, g_o(x) \tag{2.4}$$

where g_e is the even part defined as:

$$g_e(t) = \frac{1}{\sqrt{2\pi\tau}} e^{-\frac{t^2}{2\tau^2}} \cos(2\pi\omega_0 t) \tag{2.5}$$

and g_o is the odd part defined as:

$$g_o(t) = \frac{1}{\sqrt{2\pi\tau}} e^{-\frac{t^2}{2\tau^2}} \sin(2\pi\omega_0 u) \tag{2.6}$$

Figure 2.3 shows an example of one-dimensional Gabor filter.

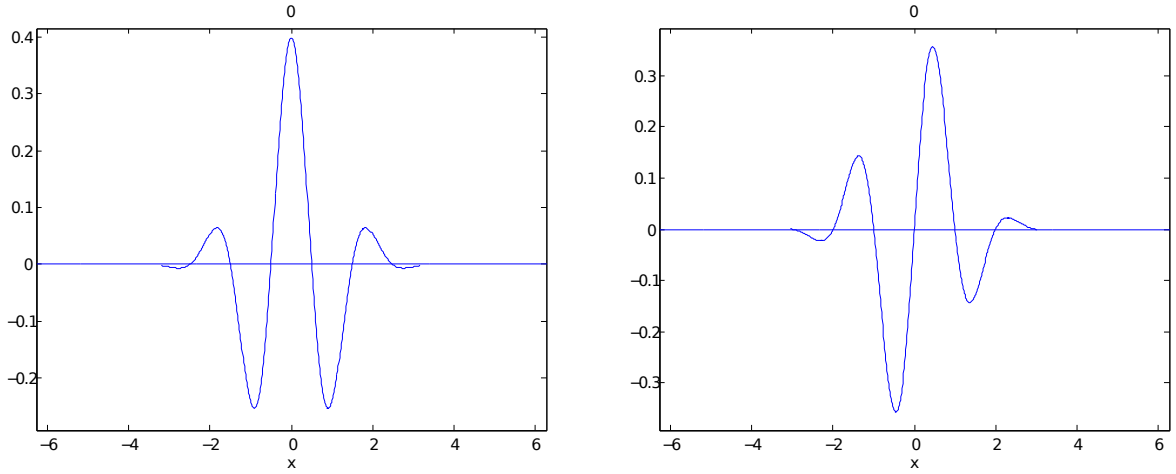

Figure 2.3: One-dimensional Gabor filter. At the left the Gabor cosine (even) and at the right the Gabor sin (odd).

As can be observed in Figure 2.3 sine and cosine Gabor operators are not quadrature pairs because cosine phase Gabors have some DC response, whereas sine Gabors do not. However, the sine and cosine Gabor pair is commonly referred to as a quadrature pair. The overall output of the two out of phase Gabor filters is calculated by adding the squared output of each filter.

2.3 Cuboids

At this point, a definition of event is necessary. Considering a spatio-temporal sliding window with size $SW_x \times SW_y \times SW_t$ that scans along the spatial and temporal axes, each group of cuboids residing in the same sliding window define an event, like in [25].

A cuboid is extracted for each STIP resulting from the response function described in Section 2.1. As a cuboid is centred in the STIP location (x, y, t), the system need to work with a frame buffer.

Considering t as the frame acquired from the stream at a certain instant and s the size of the cuboid, the response function will be computed for the $t - s/2$ frame. In practice, a cuboid is a local video sequence representing actions like a knee bending or a hand moving for example.

An event, X_i, is then represented as:

$$X_i = \{X_i^1, \ldots, X_i^{n_j}\} \tag{2.7}$$

where X^j is a cuboid and n_j is the number of cuboids of the event X_i.

Figure 2.4 shows fifty different cuboids. For simplicity each cuboid is represented in the figure only by one plane (with t fixed).

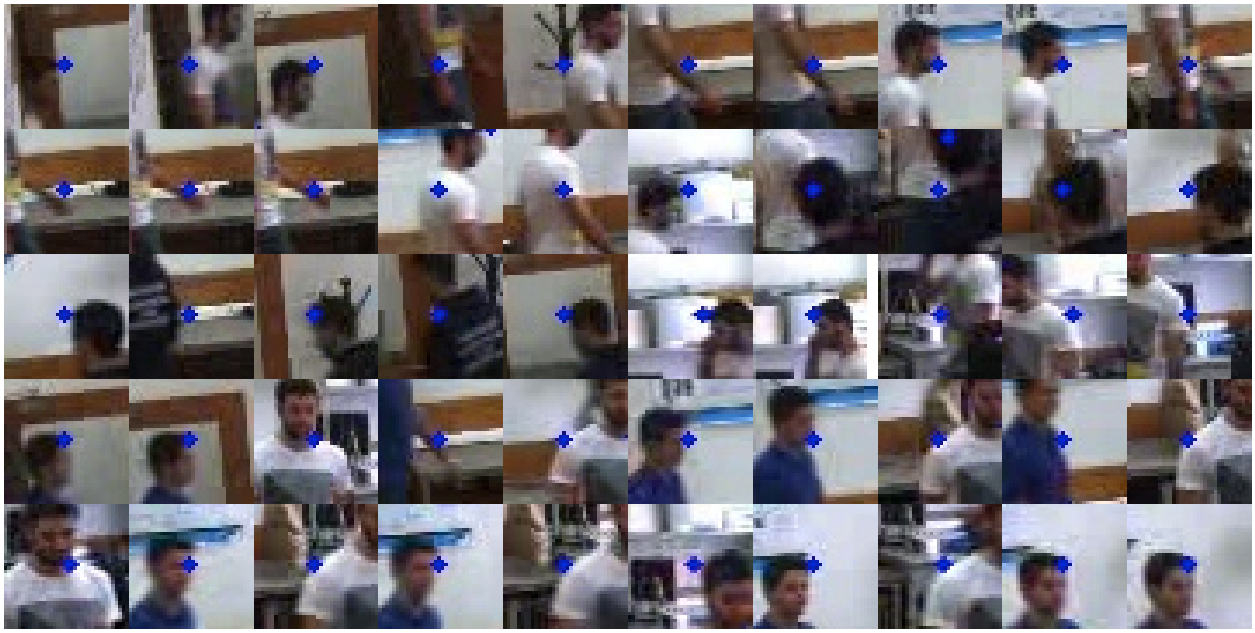

Figure 2.4: Planes of different cuboids (with t fixed).

Figure 2.5 shows one example of a sliding window containing different cuboids.

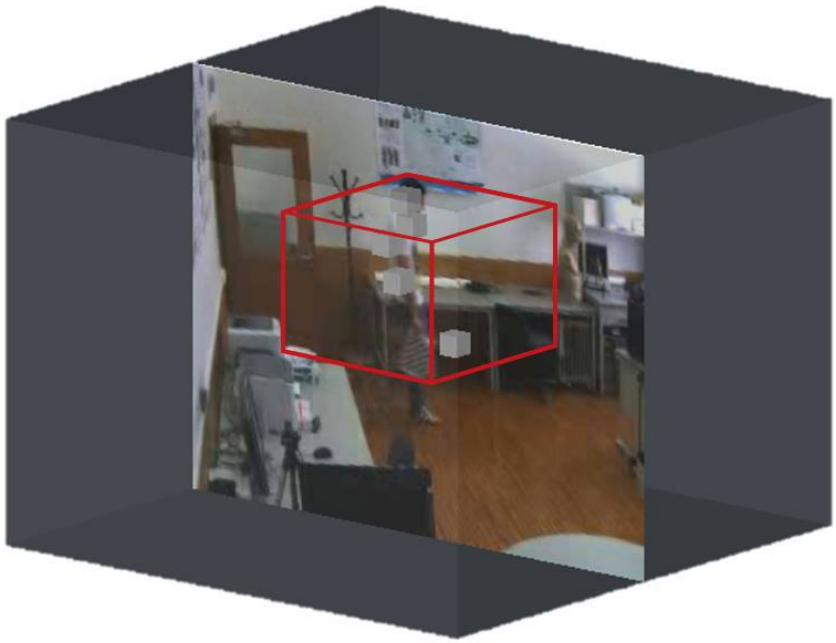

Figure 2.5: Example of a spatio-temporal sliding window (red) with a group of cuboids inside (small gray cubes).

Chapter 3

Event Descriptors

Local descriptors have been increasingly used as a representation method for action recognition. As mentioned in section 2, dense representations methods are conceptually simple to implement. However, for complex scenes with occlusions and multiple events, such methods require a complementary and robust segmentation step, which may be difficult to achieve, specially in a real time system. In this respect, local descriptors-based methods have been shown to be robust to occlusions, cropping and geometric distortions [24]. Moreover, they have shown good results when applied to event detection and action recognition [24, 25, 22]. With the purpose of verify the robustness of the sparse coding applied to event detection, three different spatio-temporal descriptors were implemented to describe the cuboids.

The first one, HOG/HOF descriptor proposed by [23], has proven to be efficient in several reviews, like [24] and [34]. HOG/HOF represents a cuboid based on HOG and Histogram of Optical Flow (HOF) computed in subcuboids and concatenated as a descriptor vector.

The other two descriptors, HOG-NSP and COV3D are recent approaches proposed by [3] and [32]. HOG-NSP uses HOG through nine different planes extracted from a video or a spatio-temporal volume to reach the final descriptor. COV3D is based in Region Covariance Matrixs (RCMs). Although this descriptor compute the RCM using the entire video, the method proposed by [32] was adapted to represent cuboids without the entire clip.

This chapter is divided in five sections: Section 3.1 and Section 3.2 introduce the concepts of HOG and HOF respectively, while Section 3.3, 3.4 and 3.5 describe the three implemented descriptors (HOG/HOF, HOG-NSP and COV3D respectively).

3.1 Histograms of Oriented Gradients

Local object appearance can often be characterized rather well by the distribution of local intensity gradients. Gradient is a vector that points in the direction of the greatest increase of the scalar values in the neighbourhood. Considering an image I, the gradient vector at a point *(x,y)* is given by:

$$\nabla I(x,y) = \left[\frac{\partial I(x,y,t)}{\partial x}, \frac{\partial I(x,y,t)}{\partial y}\right] \tag{3.1}$$

The gradient magnitude can be found from:

$$|\nabla f| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \tag{3.2}$$

The gradient direction can be calculated by the formula:

$$\theta = \arctan\left(\frac{\partial f}{\partial y} \Big/ \frac{\partial f}{\partial x}\right) \tag{3.3}$$

In order to compute HOG, the image (or Region of Interest (ROI)) is divided into small cells of size s_{Ch} and a histogram of gradient orientations with N bins is obtained for each cell. These cells can be rectangular or circular. For better invariance to illumination it is useful to contrast-normalize the local responses before using them. This process is done by accumulating local histogram "energy" over a larger regions (blocks) with size s_{Bh} to normalize all of the cells in the block. The combined histogram entries are used as the feature vector describing the image (or ROI). Since HOG descriptor operates on localized cells, the method upholds invariance to geometric and photometric transformations [20]. It is also possible to compute HOG in spatio-temporal windows, dividing each cuboid into small cuboids of size s_{Cubh} and computing a histogram of gradient orientations for each one. This process is illustrated in figure 3.1.

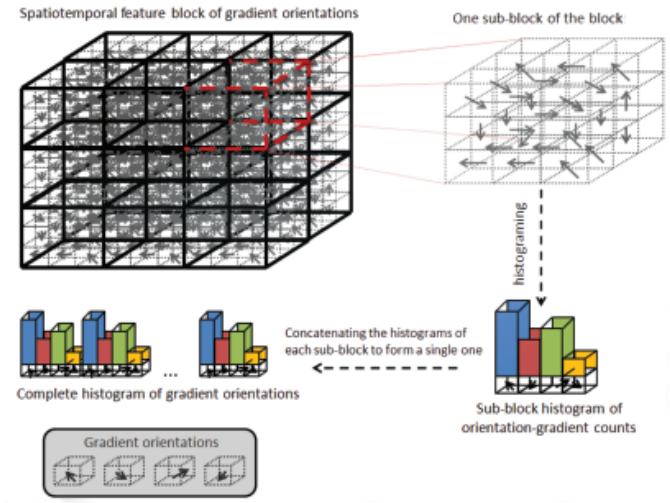

Figure 3.1: HOG construction. Figure from [1].

3.2 Histograms of Optical Flow

Optical flow is used to calculate the motion between two frames taken at times t and $t + \delta t$. Assuming that a point does not vary instantly its appearance and does not move very far between two consecutive images and considering $I(x, y, t)$ the luminance level of pixel (x, y) at time t, the *brightness constancy equation* can be expressed as:

$$I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t) \tag{3.4}$$

where δ represent small variations. Considering $I(x, y, t) = I$, the Taylor expansion of $I(x + \delta x, y + \delta y, t + \delta t)$ results in:

$$I(x + \delta x, y + \delta y, t + \delta t) \approx I + \frac{\delta I}{\delta x}\delta x + \frac{\delta I}{\delta y}\delta y + \frac{\delta I}{\delta t}\delta t \tag{3.5}$$

From Equation 3.4 it follows that $\frac{\delta I}{\delta x}\delta x + \frac{\delta I}{\delta y}\delta y + \frac{\delta I}{\delta t}\delta t = 0$. After dividing by δt and considering $v_\gamma = \frac{\gamma}{\delta t}$ and $I_\gamma = \frac{\delta I}{\delta \gamma}$, $\gamma = x, y, t$, results in:

$$\frac{\delta I}{\delta x}v_x + \frac{\delta I}{\delta y}v_y + \frac{\delta I}{\delta t} = 0 \Rightarrow \begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} = -I_t \tag{3.6}$$

Assuming that if a point moves, its neighbours exhibit the same behaviour, it follows:

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} = - \begin{bmatrix} I_x I_t \\ I_y I_t \end{bmatrix} \tag{3.7}$$

The solution of this system is the velocity vector $[v_x v_y]$.

Analogous to HOG, the magnitude and phase of each pixel can be calculated, respectively, from:

$$|F| = \sqrt{v_x^2 + v_y^2} \tag{3.8}$$

and

$$\theta_F = \arctan\left(\frac{v_y}{v_x}\right). \tag{3.9}$$

To compute HOF, the image (or ROI) is divided into small cells of size s_{Ch} and a histogram of optical flow with N bins is obtained for each cell. The combined histogram entries are used, after normalization, as the feature vector describing the image (or ROI), like in HOG. Figure 3.2 demonstrates how HOF is obtained.

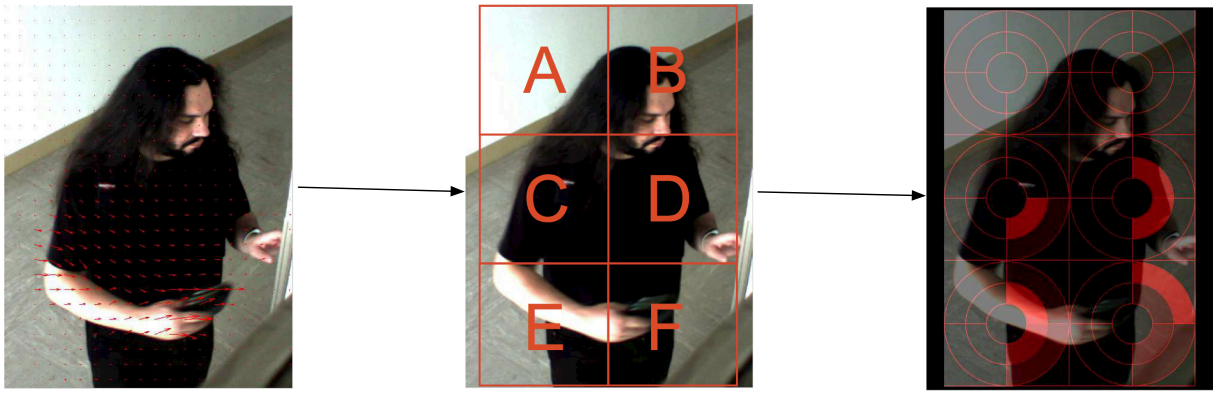

Figure 3.2: HOF construction. After determine the flow vectors (1st), the image is divided in cells (2nd) and an histogram of optical flow is computed for each cell (3rd). Figure from [2].

In order to compute HOF for cuboids, the same procedure as HOG is applied (Figure 3.1).

3.3 HOG/HOF

Laptev *et al.* [23] proposed a local feature descriptor that compute histogram descriptors of space-time volumes (cuboids) centred in an space-time interest point. Each cuboid is subdivided into a grid with $n_x \times n_y \times n_t$ spatio-temporal blocks and HOG and HOF are computed for each one. Normalized histograms are concatenated in a descriptor vector and used to characterize motion and appearance of a space-time volume.

3.4 HOG-NSP

The HOG-NSP descriptor proposed by [3] represents each $3D$ video patch (cuboid) by histograms of oriented gradients over different spatio-temporal symmetry planes, reaching the final descriptor by the Bag of Features (BoF) framework. This descriptor is an extension of the Local Binary Pattern on Three Orthogonal Planes (LBP-TOP) descriptor, proposed by Zhao *et al.* [35]. The LBP-TOP was the first attempt to create a descriptor with no direct extension from 2D to 3D, unlike HOG/HOF [23] or HOG3D [22]. The ideia behind LBP-TOP is to encode a video patch by computing local binary patterns over three orthogonal planes XY, XT and YT planes. According to [3], these three orthogonal planes are not able to optimally capture the dynamical proprieties of a 3D patch.

The nine planes of HOG-NSP descriptor can be obtained by rotating XY, XT and YT planes by 45°, 90° and 135°. For example, by rotating the XY plane around the X axis by 45°, 90° and 135°, three orthogonal planes to the YT plane are attained and named as $P_{(X,45)}$, $P_{(X,90)}$ and $P_{(X,135)}$. The same process is repeated for XT and YT rotations, resulting in twelve planes. Figure 3.3 illustrates this process.

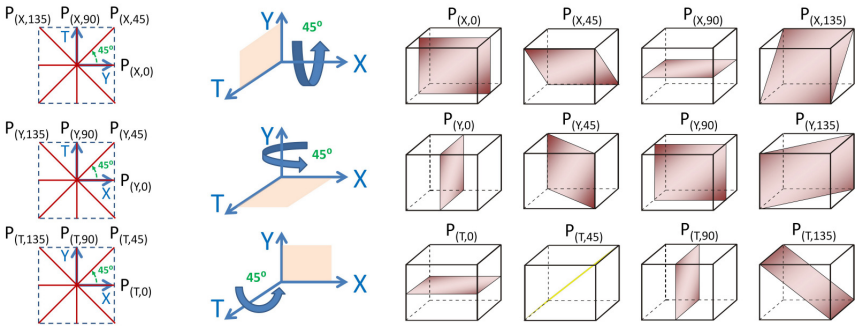

Figure 3.3: HOG-NSP planes. Figure from [3].

As shown in Figure 3.3, some planes are repeated twice ($P_{(X,0)} = P_{(Y,90)}$, $P_{(Y,0)} = P_{(T,90)}$ and $P_{(T,0)} = P_{(X,90)}$). The repeated planes are discarded, resulting in nine distinct planes. In this work the HOG descriptors of these nine planes are concatenated to form the final descriptor of the cuboid, instead of BoF solution followed by [3].

3.5 COV3D

Region covariance descriptors were first introduced by [36] for object detection and classification. The representation of a spatio-temporal window with a RCM has several advantages such as the low-dimensional representation, the independence of the window size and the

reduced impact of noisy samples. Covariance matrices are a natural way of fusing multiple features which might be correlated. The diagonal entries of the covariance matrix represent the variance of each feature and the non-diagonal entries represent the correlations.

Harandi *et al.* [32] proposed a spatio-temporal covariance descriptor applied densely along the entire video. In this work, that idea is explored in a sparse representation, where each cuboid is represented by a covariance matrix. For that, intensity gradients and optical flow are combined, since previous studies have shown the benefit of combining these two types of features [11, 34].

Let V be the sequence of local images that belongs to a video C, s the cuboid size and d the feature vector of each pixel. The $s^3 \times d$ dimensional feature video extracted from V, represented as F is given by:

$$F(x, y, t) = \Phi(V, x, y, t) \tag{3.10}$$

where x, y and t are the coordinates of each pixel on the cuboid and Φ the feature mapping, given by:

$$\Phi(V, x, y, t) = \begin{bmatrix} x & y & t & g & o \end{bmatrix}^T \tag{3.11}$$

with

$$g = \begin{bmatrix} |I_x| & |I_y| & |I_x x| & |I_y y| & \sqrt{I_x^2 + I_y^2} & \arctan \frac{|I_y|}{|I_x|} \end{bmatrix} \tag{3.12}$$

$$o = \begin{bmatrix} u & v & \frac{\partial u}{\partial t} & \frac{\partial v}{\partial t} & \left(\frac{\partial u}{\partial t} + \frac{\partial v}{\partial t}\right) & \left(\frac{\partial u}{\partial t} - \frac{\partial v}{\partial t}\right) \end{bmatrix} \tag{3.13}$$

Equation 3.12 represent the gradient based features. The first four elements correspond to the first and second order intensity gradients at pixel location (x, y) and the last two to the gradient magnitude and orientation.

On the other hand, 3.13 represent optical flow based features, in order: the horizontal and vertical components of the flow vector, the first order derivatives of the flow components with respect to t and the spatial divergence and vorcity of the flow field.

Considering $z_{i_{i=1}}^S$ as the d-dimensional feature vector inside C and S the cuboid volume, the region defined by C can be represented with the $d \times d$ covariance matrix of the feature vectors:

$$Cov3D_C = \frac{1}{S} \sum_{i=1}^{S} (z_i - \mu)(z_i - \mu)^T \tag{3.14}$$

where μ is the mean of the points. In order to accelerate the process, the concept of integral videos is introduced. An integral video is nothing more than a stack of integral

images. For a video V, its integral video IV is defined as:

$$IV(x', y', t') = \sum_{x \leq x'} \sum_{y \leq y'} \sum_{t \leq t'} V(x, y, t) \tag{3.15}$$

Therefore, the covariance matrix defined in 3.14 can be defined as:

$$Cov3D_C(i, j) = \frac{1}{S-1}\left(\sum_{k=1}^{S} z_k(i) z_k(j) - \frac{1}{S} \sum_{k=1}^{S} z_k(i) \sum_{k=1}^{S} z_k(j) \right) \tag{3.16}$$

To find the covariance in a given spatio-temporal window, the sum of each feature dimension, $z(i)_{i=1}^{d}$, defined as P, and the sum of the multiplication of any two feature dimensions, $z(i)z(j)_{i,j=1\ldots d}$, defined as Q, should be calculated. This is achieved using integral videos:

$$P(x', y', t', i) = \sum_{x \leq x'} \sum_{y \leq y'} \sum_{t \leq t'} F(x, y, t)(i) \tag{3.17}$$

and

$$Q(x', y', t', i, j) = \sum_{x \leq x'} \sum_{y \leq y'} \sum_{t \leq t'} F(x, y, t)(i).F(x, y, t)(j) \tag{3.18}$$

The d-dimensional feature vector $p_{x,y,t}$ and the $d \times d$ dimensional matrix $Q_{x,y,t}$ can be obtained from:

$$p_{x,y,t} = \Big[P(x, y, t, 1), \ldots P(x, y, t, d) \Big]^T \tag{3.19}$$

and

$$Q_{x,y,t} = \begin{bmatrix} Q(x, y, t, 1, 1) & \ldots & Q(x, y, t, 1, d) \\ \vdots & \ddots & \vdots \\ Q(x, y, t, d, 1) & \ldots & Q(x, y, t, d, d) \end{bmatrix}^T \tag{3.20}$$

Considering the coordinates inside the cuboid as $\{(x, y, t) | 0 \leq x \leq x', 0 \leq y \leq y', 0 \leq t \leq t'\}$ with $x', y', t' = s$, the covariance of the spatio-temporal window is:

$$Cov3D_{C(0,0,0;x',y',t')} = \frac{1}{S-1}\left(Q_{x,y,t} - \frac{1}{S} p_{x,y,t} p_{x,y,t}^T \right) \tag{3.21}$$

Considering a STIP k centred at (x_k, y_k, t_k), the origin of the cuboid C_k related to the video coordinate system is:

$$C_k(0, 0, 0) = V(x_k - \frac{s}{2}, y_k - \frac{s}{2}, t_k - \frac{s}{2}) \tag{3.22}$$

Figure 3.4 shows how coordinates are related in a cuboid.

Figure 3.4: Cuboid coordinate system

The resultant covariance matrices are symmetric positive definite matrices of size $d \times d$, which can be formulated as a connected Riemannian manifold. A manifold is a topological space which is locally similar to an Euclidean space. A point $\mathbf{Y}$ on the manifold can be mapped to a vector in the tangent space T_x (plane tangent to the surface of the manifold at point X) using the logarithm map operator, as follows:

$$\log_X Y = X^{\frac{1}{2}} \log(X^{-\frac{1}{2}} Y X^{-\frac{1}{2}}) X^{\frac{1}{2}} \tag{3.23}$$

In this work, all covariance matrices will be projected to the tangent space at the identity point, since it was observed to be enough to obtain good results. Thus, equation 3.23 can be simplified as:

$$\log_I Y = \log(Y) \tag{3.24}$$

Given the eigenvalue decomposition of a symmetric matrix, $M = U \log(D) U^T$, the matrix logarithm can be obtained from:

$$\log(M) = U \log(D_m) U^T \tag{3.25}$$

where $\log(D_m)$ is a diagonal matrix with each diagonal element equal to the logarithm of the corresponding element in D.

The resultant matrix M is also symmetric positive definite. The final COV3D descriptor corresponds to the upper triangle of M, resized as a vector, like in [44]. Other approaches resort to more complex mappings that try to preserve the relations between points of the manifold, like [32, 37]. Alternatively, [38] proposed a sparse decomposition of positive definite matrices, thus enabling the use of a sparse coding formulation directly in the manifold.

Chapter 4

Event Evaluation

The evaluation of the normality of an event is based on a sparse coding formulation.

Sparse coding has recently attracted notable attention in computer vision. Its advantage is to represent succinctly large amounts of data, decreasing the memory used when compared to dense representations and describing complex data in a way that is easier to interpret. The aim of sparse coding is to find a set of basis vectors from an initial learned dictionary such that an input vector can be represented as a linear combination of these basis vectors, in the same way as human neurons are activated to encode sensory information [39].

Assuming that an unusual event is unlikely to occur in the small initial portion of the video, a dictionary is learned from the events occurred in that period. Thereafter, the descriptors obtained are reconstructed from the dictionary and an objective function is computed. Small values of the objective function indicate the presence of an usual event while high values mean that an unusual event is involved. After that process, the dictionary is updated in order to include the newly observed event. This step provides the possibility of using a small amount of data to learn the initial dictionary, since the system have the ability to change the knowledge of what is an unusual event. The alternative is to ”show” all the possible usual events while the initial dictionary is constructed, which requires a huge amount of training data.

This chapter is organized as follows: Section 4.1 introduces the sparse coding formulation and explains how each term influences the objective function; Section 4.2 demonstrates the dictionary learning step as well as the reconstruction of an event from the dictionary bases; Section 4.3 covers the online dictionary update algorithm; Section 4.4 shows how events are classified with respect to their normality based on their reconstruction from the dictionary.

4.1 Sparse Coding Formulation

As explained in chapter 2, an event is represented by descriptor vectors extracted from a group of cuboids residing in the same sliding window. From that, an event X_i composed by n_j cuboids, can be defined as $X_i = \{X_i^1, \ldots, X_i^{n_j}\}$.

Given an initial dictionary D (details about learning D in section 4.2), the objective function J that measures the normality X is defined as:

$$J(X, \alpha_i, D) = \frac{1}{2}\sum_{j}^{n_j} ||X_i^j - D\alpha_i^j||_2^2 + \lambda_1 \sum_{j}^{n_j} ||\alpha^j||_1 + \frac{1}{2}\lambda_2 \sum_{j}^{n_j} ||\alpha^j||_2^2 \tag{4.1}$$

where $\alpha_i = \{\alpha^1, \ldots, \alpha^{n_j}\}$ represent the reconstruction weight vectors for the event X_i.

The first term in equation 4.1 is the reconstruction error. Since the dictionary was learned from usual event data, this term should be small for usual events. On the other hand, unusual events usually imply high reconstruction errors. Although, it is possible that unusual events show small reconstruction errors. Assuming a dense reconstruction weight vector it is possible to achieve a weight values combination that induce a small reconstruction error. The next term prevents this situation.

The second term is the sparsity regularization. Since the dictionary is learned in order to maximize the sparsity of weight vectors for usual events, this term enforces the sparsity of α. Thus, usual events cause a sparse reconstruction weight vector. In cases where unusual events have a small reconstruction error, it will result in a dense reconstruction weight vector and, consequently, a higher objective function.

The third term is the smoothness regularization. The quadratic penalty encourages the grouping effect (similar motions at neighbouring patches are more likely to be involved in a usual event) and removes the limitation on the number of selected variables verified when $\lambda_2 = 0$ [40].

Figure 4.1 presents the reconstruction weight vectors for two events constituted by four cuboids. The first is usual and the second is unusual. The last column represents the sum of the weights computed for each cuboid. It can be observed that the reconstruction vectors for an usual event are sparse, while the ones for unusual events are dense.

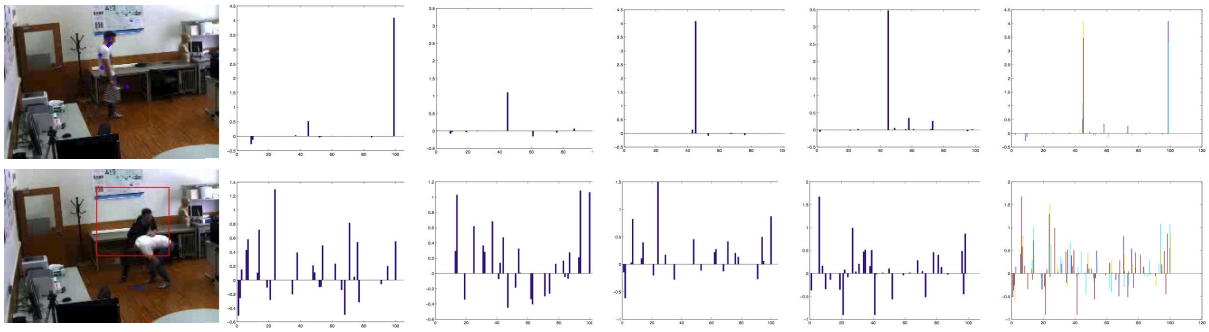

Figure 4.1: Reconstruction weight vectors comparison for an usual event (first row) and an unusual event (second row).

4.2 Optimization

The objective function of Equation 4.1 measures the normality of an event X_i for any dictionary D and any weights α_i. Since usual events corresponds to lower J values, one needs to find the optimal dictionary D^* and the optimal reconstruction weight vectors α_i^* which minimize the objective function for the event X_i. Therefore, and taking into account Equation 4.1, the optimal reconstruction weight vectors α_i^* and dictionary D^* are learned from the following optimization problem:

$$(\alpha_i^*, D^*) = \underset{\alpha_i, D}{\operatorname{argmin}}\, J(X_i, \alpha_i, D) \tag{4.2}$$

It can be observed that the optimization problem is not jointly convex with respect to D and α. However, this problem can be solved alternating between these two variables, minimizing one while holding the other until converge to a local optimum.

With the dictionary D fixed, the reconstruction weight vectors α for the event X_i are obtained from:

$$\min_{\alpha_i^1,\ldots,\alpha_i^{n_j}} \frac{1}{2}\sum_{j}^{n_j} ||X_i^j - D\alpha_i^j||_2^2 + \lambda_1 \sum_{j}^{n_j} ||\alpha^j||_1 + \frac{1}{2}\lambda_2 \sum_{j}^{n_j} ||\alpha^j||_2^2 \tag{4.3}$$

where λ_1 and λ_2 are regularization parameters.

To solve this optimization problem, the LARS algorithm introduced in [46] and implemented by [45] is used.

Holding α, the dictionary D can be obtained from:

$$\min_{D} \frac{1}{2m} \sum_{i=1}^{m} \sum_{j=1}^{n_j} ||X_i^j - D\alpha_i^j||_2^2 \tag{4.4}$$

$$\begin{aligned} s.t.\, D \in \mathbb{R}^{m \times k}, \\ \forall j = 1, \ldots, k,\, d_j^T d_j \leq 1 \end{aligned} \tag{4.5}$$

where m is the number of events and k the size of each descriptor. The constraint in 4.5 is introduced to prevent terms in D from being arbitrarily large which will result in small values of the weights α.

To solve this optimization problem, the Lagrange Dual algorithm documented in [26] and implemented by [45] is used.

4.3 Online Dictionary Update

Over time, the typical behaviour recorded in the environment under analysis can change. For example, in a metro station, at peaks time, it is normal that people to move faster than the rest of the time. The most common approach is to learn the initial dictionary with all possible usual events, including these behaviour changes. Although, an efficient video-vigilance system has to learn to adapt to these changes event if they do not belong to the initial observed data, incorporating them in the dictionary and considering these new events as usual.

As shown by Equation 4.4, the optimal dictionary after t events is the solution of the following optimization problem:

$$\min_{D \in C} \frac{1}{2t} \sum_{i=1}^{t} \sum_{j=1}^{n_j} ||X_i^j - D\alpha_i^j||_2^2 \tag{4.6}$$

with $C = \{D \in \mathbb{R}^{m \times k} : d_j^T d_j \leq 1, \forall j = 1, \ldots, k\}$.

To solve this problem, all t events are needed and the optimization problem have to be solved from scratch, causing memory and time overconsumption. Therefore, the projected first order stochastic gradient descent update proposed by [41] is used. This update only needs the event X_t and the previous dictionary D_{t-1} and is obtained from:

$$D_t = \Pi_C \left[D_{t-1} - \frac{\eta}{t} \nabla_D l(X_t, D_{t-1}) \right] \tag{4.7}$$

where η is the learning rate, Π_C the projection onto C. $l(X_t, D_{t-1})$ is given by:

$$l(X_t, D_{t-1}) = \frac{1}{2}\sum_{j=1}^{n_j} ||X_t^j - D_{t-1}\alpha_t^j||_2^2 \tag{4.8}$$

Following the stages in [41], the dictionary update algorithm is summarized in Algorithm 1. To exemplify, it is considered a random variable $X \in \mathbb{R}^m$. The inner loop appraise one event X_t at a time, as in stochastic gradient descent. The weight vectors α are computed from X_t over the dictionary D_{t-1}, obtained at the previous iteration. The new dictionary D_t is obtained by minimizing over C the function:

$$\hat{f}_t(D) \triangleq \frac{1}{t}\sum_{j=1}^{n_j} \frac{1}{2}||X_i - D\alpha_i||_2^2 + \lambda||\alpha_i||_1 \tag{4.9}$$

where the vectors α_i are computed during the previous steps of the algorithm. The quadratic function $\hat{f}_t$ aggregates the past information (vectors α_i). As demonstrated by [41], $\hat{f}_t(D_t)$ and $f_t(D_t)$ converges almost surely to the same limit and thus $\hat{f}_t(D_t)$ acts as a surrogate for $f_t(D_t)$. For that, the uniqueness of the sparse coding solution should be ensured. This condition can be forced using an elastic net penalization [41], replacing $||\alpha||_1$ by $||\alpha||_1 + \frac{k_2}{2}||\alpha||_2^2$. Since $\hat{f}_t$ is close to $\hat{f}_{t-1}$, D_t can be obtained efficiently using D_{t-1} as a warm restart.

The update of the dictionary is done using block-coordinate descent with warm restarts, making it parameter-free, without any learning rate tuning. Algorithm 2 sequentially updates each column of D. Equation 4.12 gives the solution of the dictionary update with respect to the j-th column, d_j, keeping the other ones fixed under the constraint $d_j^T d_j \leq 1$. [41] also demonstrate that the convergence to a global optimum is guaranteed and that since D_{t-1} is used as a warm restart for computing D, a single iteration of the algorithm has empirically been found to be enough.

An alternative to this dictionary update algorithm is to use a Newton method on the dual of Equation 4.11. Although, this requires inverting a $k \times k$ matrix at each iteration, which is unattainable for an online algorithm.

4.4 Unusual Event Detection

As previously mentioned, given a newly observed event X and the current dictionary D, the correspondent optimal reconstruction weight vectors α are calculated. X is classified as

Algorithm 1 Online Dictionary Learning

Require: Initial dictionary $D_0 \in \mathbb{R}^{m\times k}$, $\lambda \in \mathbb{R}$, T (number of iterations), $X \in \mathbb{R}^m \sim p(x)$ (random variable and an algorithm to draw samples of p).
$A_0 \leftarrow 0$
$B_0 \leftarrow 0$
for t=1 to T **do**
Draw X from $p(x)$
Sparse coding: compute α_t for event X_t

$$\alpha_t = \underset{\alpha\in\mathbb{R}^k}{\operatorname{argmin}} \frac{1}{2}||X_t - D_{t-1}\alpha_t||_2^2 + \lambda_1||\alpha||_1 \tag{4.10}$$

$A_t \leftarrow A_{t-1} + \alpha_t\alpha_t^T$
$B_t \leftarrow B_{t-1} + x_t\alpha_t^T$
Calculate D_t using Algorithm 2, with D_{t-1} as warm restart, so that:

$$\begin{aligned} D_t &= \underset{D\in C}{\operatorname{argmin}} \frac{1}{t}\sum_{i=1}^{t}\frac{1}{2}||X_i - D\alpha_i||_2^2 + \lambda_1||\alpha_i||_1 \\ &= \underset{D\in C}{\operatorname{argmin}} \frac{1}{t}\Big(\frac{1}{2}\operatorname{Tr}(D^TDA_t) - \operatorname{Tr}(D^TB_t)\Big) \end{aligned} \tag{4.11}$$

end for

Algorithm 2 Dictionary Update

Require: Input Dictionary $D = [d_1, \dots d_k] \in \mathbb{R}^{m\times k}$
$A = [a_1, \dots a_k] \in \mathbb{R}^{k\times k} = \sum_{i=1}^{t}\alpha_i\alpha_i^T$
$B = [b_1, \dots b_k] \in \mathbb{R}^{m\times k} = \sum_{i=1}^{t}x_i\alpha_i^T$
repeat
for j=1 to k **do**
Update the j-th column to optimize for

$$\begin{aligned} u_j &\leftarrow \frac{1}{A_{jj}}(b_j - Da_j) + d_j \\ d_j &\leftarrow \frac{1}{\max(||u_j||_2, 1)}u_j \end{aligned} \tag{4.12}$$

end for
until converge

unusual if the following criterion is verified:

$$J(X, \alpha, D) > \varepsilon \tag{4.13}$$

where ε is a user defined threshold that controls the sensivity of the algorithm to unusual events and J is the objective function defined in Equation 4.1.

Algorithm 3 present the main steps of the unusual event detection algorithm:

Algorithm 3 Unusual Event Detection Using Sparse Coding

Input: Video data, threshold ε
  Learn initial dictionary using first N frames in video
  repeat
    Use sliding window to obtain event X_t
    Compute optimal reconstruction weight vectors α_t for event X_t, solving Equation 4.3 with $D = D_{t-1}$
    if $J(X, \alpha, D) > \varepsilon$ **then**
      Fire alarm for event X_t. Event considered as unusual
    end if
  until reach the end of the video

The STIP calculation explained in Chapter 2 and the event description presented in Chapter 3 were omitted from Algorithm 3 for simplicity.

Chapter 5

Experimental Results

This chapter presents the results of the algorithms developed along the project.

The unusual event detector system has been tested with three different type of videos. In the first place, the KTH Human Action dataset was used. This dataset contains six types of human actions (walking, jogging, running, boxing, hand waving and hand clapping). Each video sequence have only one single event played by only one person at a time. Thereafter, the system was tested with three videos created as part of the present work, referred as *LabBrisa Videos*. Finally, the system was tested for *real world* videos extracted from the *YouTube*. These videos typically consist in crowded scenes with multiple events, extracted from real video-surveillance cameras. All videos have a frame size of 240×180 pixels. The entire processing is done using grayscale images.

This chapter is divided in four main sections. Firstly, Section 5.1 contains a brief description of the video datasets used along this dissertation. Secondly, Section 5.2 presentes the parameters used in the event detection. Afterwards, in Section 5.3 are shown some examples of the descriptors obtained for different cuboids. The figures shown in these two sections are extracted from the videos mentioned above without any particular order. In Section 5.4 are presented and compared the results of the unusual event detector for the three datasets and also the comparison between the different spatio-temporal descriptors used in this work.

5.1 Video Datasets

This section introduces the datasets used to test the unusual event detector. For each video is presented a Figure with examples of usual and unusual events (except for KTH Human Actions where the definition of usual event depends on the sequence used to learn the dictionary). Events with a green box are considered as usual while events with a red

box are designed as unusual.

5.1.1 KTH Human Actions Dataset

The KTH human actions dataset is constituted by six types of human actions (walking, jogging, running, handwaving, handclapping and boxing) distributed along 2391 sequences. Each video sequence has only one single event played by only one person at a time, without any occlusion. Figure 5.1 shows some examples of the human actions covered by this dataset.

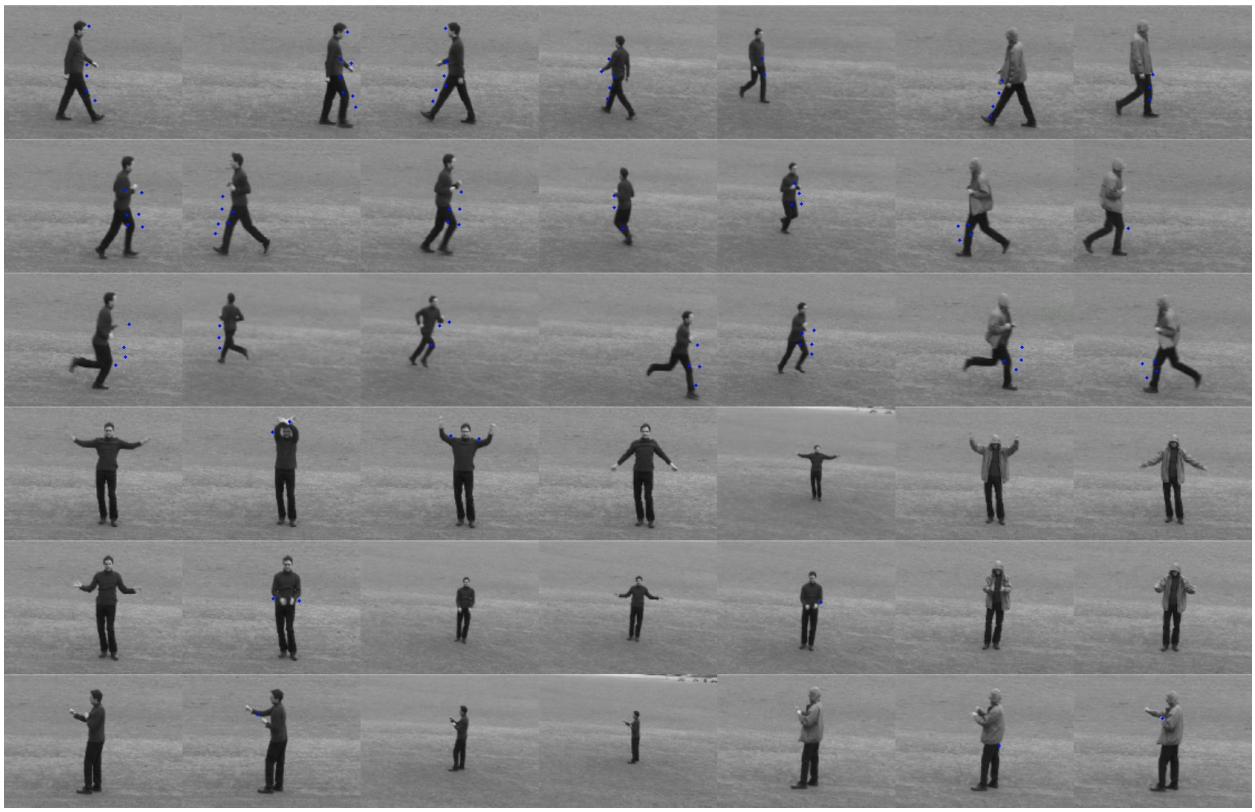

Figure 5.1: KTH human actions examples. Each row represents one different action, ordered as: walking; jogging; running; handwaving; handclapping; boxing.

5.1.2 *LabBrisa* Videos

As a part of this dissertation, three videos were made in order to test the unusual event detector.

The first video, referred as *LabBrisa* Video 1, have different persons walking in a small area, originating many occlusions. The initial dictionary was learned from the first two minutes from the ten minutes video and then the detector is tested along the full video length. This initial dictionary was used for all *LabBrisa* videos. Ideally, the detector should not fire any unusual event alarm, since the video is only constituted by people walking in

the same way as verified during the initial dictionary construction. Figure 5.2 shows the examples of usual events verified in *LabBrisa* Video 1.

Figure 5.2: Video with people walking, referred as *LabBrisa* Video 1.

The second video, referred as *LabBrisa* Video 2, has a fight scene captured in the same environment. Figure 5.3 shows the type of events found in this video sequence.

Figure 5.3: Video with a fight scene, referred as *LabBrisa* Video 2.

Lastly, the third video, referred as *LabBrisa* Video 3, has a person dropping a box and returning back to pick up the box. As demonstrated in Figure 5.4 the unusual events include the box falling to the ground and the movements of stop, go down and get up made by the person.

Figure 5.4: Video with a person leaving a box behind and turning around to catch it, referred as *LabBrisa* Video 3.

5.1.3 *YouTube* Videos

This dataset is constituted by a number of videos downloaded from *YouTube*. These videos contain different categories of targets (human, vehicles, etc.) and covers a wide variety of activities and environmental conditions (indoor and outdoor). For each video, the initial dictionary is learned in the initial portion of the video. This portion are, in general, 1/5 of the video. Thereafter, the unusual event detector search for unusual events in the entire video.

The first video, referred as *YoutubeVideo* 1, was captured with a video-surveillance camera in an outdoor environment. The events considered as unusual are the car crash and people gathered in the middle of the road. Figure 5.5 shows examples of some usual and unusual events of *YoutubeVideo* 1.

Figure 5.5: Video with a car accident downloaded from *Youtube*, referred as *YoutubeVideo* 1.

The second video, referred as *YoutubeVideo* 2, was captured by a video-surveillance camera inside a convenience store. Figure 5.6 shows examples of usual and unusual events. The

most obvious unusual event found in this video is a person falling from the roof. Although this video has an interesting detail. As can be observed in the last image of the second row of Figure 5.6, the system mark a person walking to the end of the store as unusual event. This is due to the fact that the initial dictionary only contain a person walking in front of the camera and leaving the store by the door (Figure 5.6, last image of the first row).

Figure 5.6: Video with a person falling from the roof in a store downloaded from *Youtube*, referred as *YoutubeVideo* 2.

The third video, referred as *YoutubeVideo* 3, captured in a tyre company. The initial dictionary was learned only when a person is walking to the company's office. The unusual events include a person transporting tyres in a wheeled structure and, after that, using that wheeled structure to skate. Figure 5.7 shows examples of the events of this video. In the fourth image of the second row it is possible to observe that a truck on the road was marked as an unusual event. This event is indeed unusual, since in the initial portion of the video only regular cars have been seen.

Figure 5.7: Video with a person skating and falling, downloaded from *Youtube*, referred as *YoutubeVideo* 3.

Lastly, the fourth video was captured in a judo class and is referred as *YoutubeVideo* 4. The usual events in this video include people fighting and falling in the floor. This video

is important to test the robustness of the detector because has crowded scenes with many occlusions. The unusual events are caused by an abnormal luminosity in the room caused by a meteor. Figure 5.5 shows examples of some usual and unusual events of *YoutubeVideo* 4.

Figure 5.8: Video with an unusual luminosity caused by a meteor, in Russia, downloaded from *Youtube* and referred as *YoutubeVideo* 4.

5.2 Event Detection

Recalling Chapter 2, each STIP is extracted from a local maxima of a response function calculated with a combination of a $2D$ Gaussian smoothing kernel with a spatial scale σ applied spatially and a quadrature pair of $1D$ Gabor filters with temporal scale τ applied temporally.

Figure 5.9 presents the response function for different σ and τ scales.

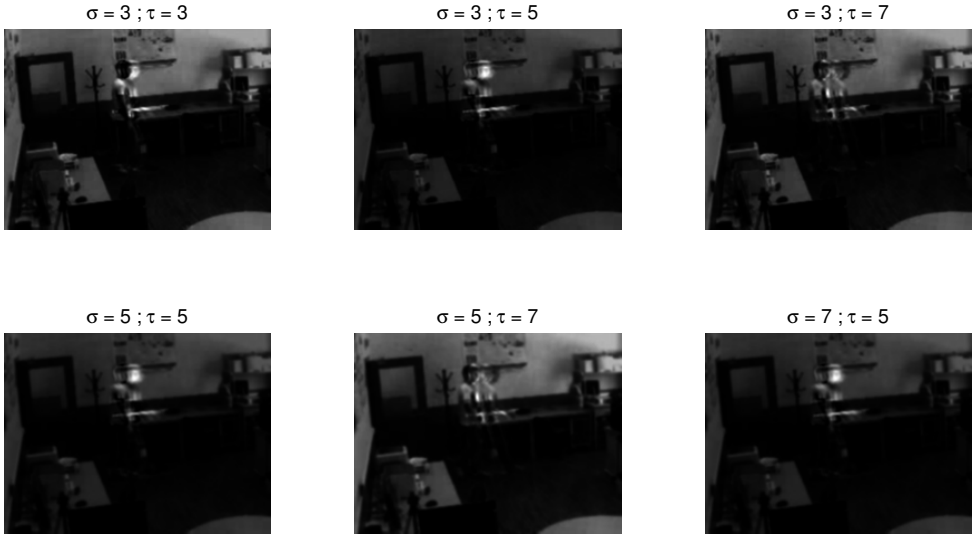

Figure 5.9: Response functions for different values of σ and τ.

The best results are achieved with $\tau = 5$. Since small variations of σ does not induce a relevant difference in the response function, the chosen value for the spatial scale was $\sigma = 3$. Despite this, bigger values of σ reduce the level of detail of the image, compromising the results. The spatio-temporal interest points are then extracted from the local-maxima of the resulting response function.

For each STIP, a cuboid is extracted. According to [11], cuboids have a side length of approximately six times the temporal scale at which they were detected. Since $\tau = 5$, the side length of the cuboids is $s = 33$ pixels (the extra pixels are to ensure that a cuboid contain all the volume of data that contributed to the response function at that interest point).

Figure 5.10 shows the resultant STIPs for two different environments and the correspondent cuboids. Like in Figure 2.4, each resultant cuboid is represented only by one xy plane (with t fixed).

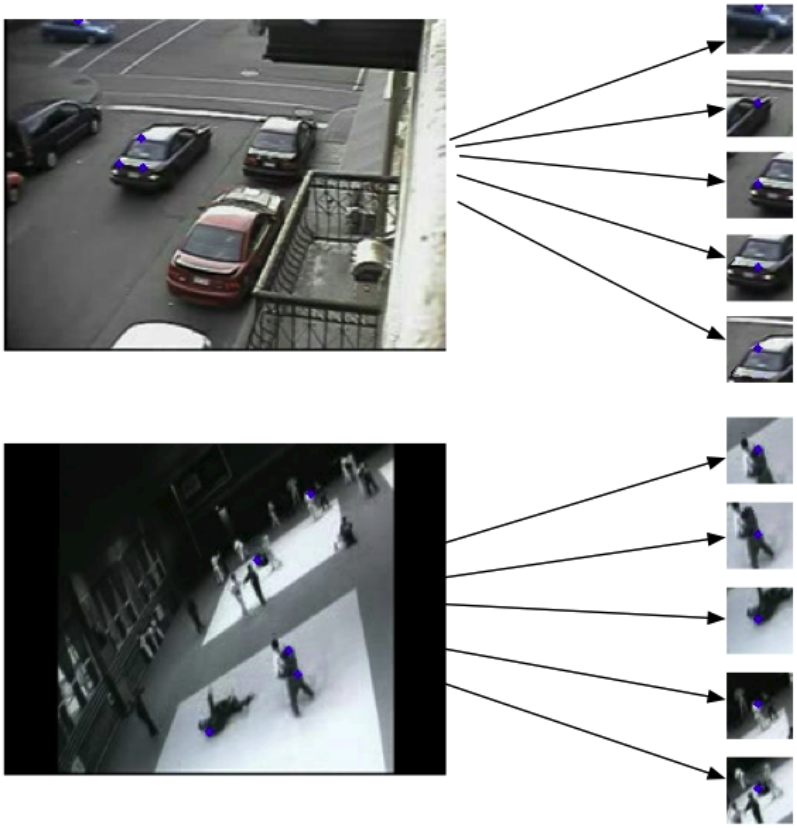

Figure 5.10: Resultant STIPs for two different environments.

An example of the different slices of a cuboid (xy planes with t fixed) is shown in Figure 5.11.

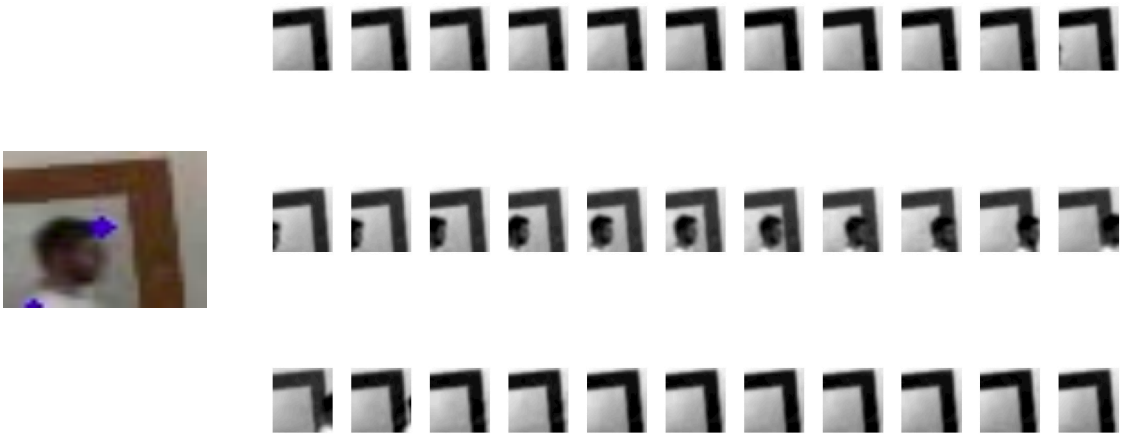

Figure 5.11: Slices of a cuboid (xy planes with t fixed).

As explained in Section 2.3, the cuboids are grouped by a sliding window that scans

along the spatial and temporal axes. The size of this sliding window is $40 \times 40 \times 40$ voxels. This spatio-temporal window slides without overlap in both spatial and temporal axes. For each sliding window, the centroid $C(x_c, y_c, t_c)$ of the cuboids residing there is calculated and considered as the center of the event. If two event centers are not spaced more than a half of the side length of a sliding window, they are considered as the same event.

5.3 Event Descriptors

In this section are shown the parameters used to calculate each descriptors, as well as some examples of the results of each descriptor applied to different cuboids. The comparative results between each one are presented in Section 5.4.

5.3.1 HOG/HOF

The binary file with the implementation of HOG/HOF descriptor was downloaded from Laptev's website[1]. This binary file contain also the interest point detector proposed in [13]. Despite this, this tool is only used to compute HOG/HOFF descriptor. All interest points are calculated with the response function described in Section 2.1.

Each cuboid is partitioned into a grid with $3 \times 3 \times 3$ spatio-temporal blocks. For all blocks, a 4 bins HOG descriptor and a 5 bins HOF descriptor are computed and concatenated into a 72 element HOG descriptor and a 90 element HOF descriptor. The number of bins is not a user defined parameter of the tool.

Figure 5.12 shows the HOG/HOF descritors of two different cuboids. As in previous figures, a cuboid is represented only by one xy plane.

[1] http://www.di.ens.fr/ laptev/download.html

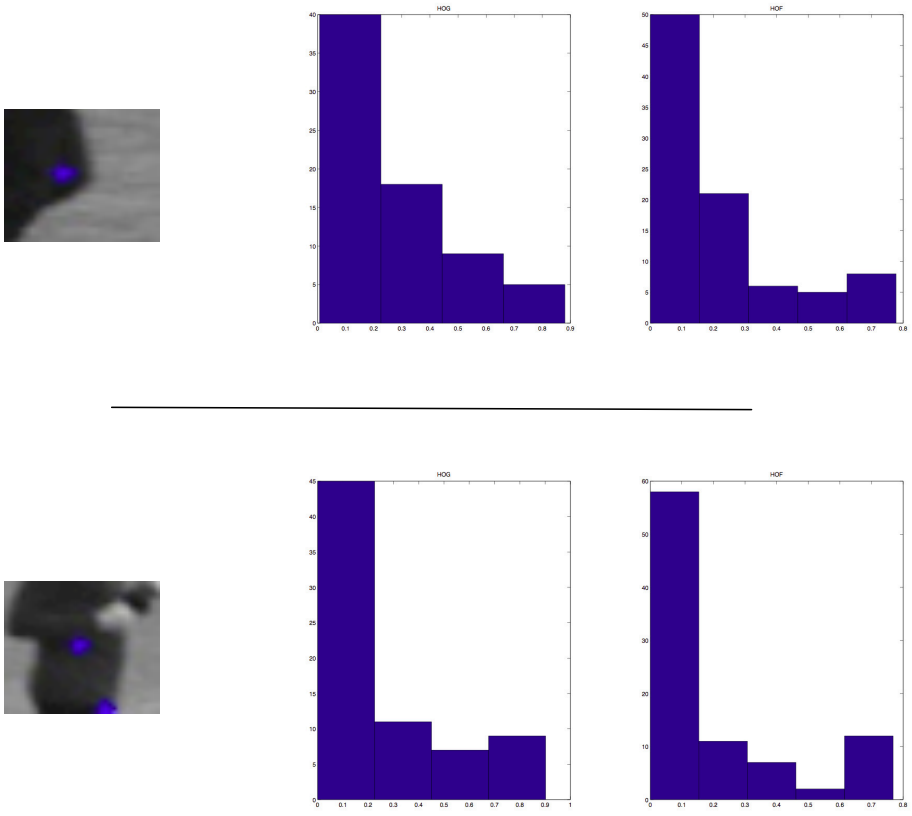

Figure 5.12: HOG/HOF computed for two different cuboids.

The final descriptor vector also include the (x, y, t) normalized coordinates, thus forming a 165 element final descriptor.

5.3.2 HOG-NSP

As explained in Section 3.4 each cuboid is partitioned in nine different planes and the HOG descriptor is computed for each plane. For that, the OpenCV HOG descriptor was used.

The choice of the parameters used to calculate HOG has taken into account the size of the cuboid, the size of the final descriptor and the restrictions of the OpenCV HOG descriptor. The window size W is adjusted to the size of each plane. Since OpenCV HOG descriptor does not accept the size $W = 33 \times 33$, the window size used is the closest to the cuboid size: $W = 32 \times 32$. After multiple tests the remaining parameters used to compute HOG were:

$$s_{cell} = 8 \times 8 \tag{5.1}$$

$$s_{block} = 32 \times 32 \tag{5.2}$$

$$N_{bins} = 5 \tag{5.3}$$

Figure 5.13 shows the nine symmetry planes obtained for two different cuboids. As in previous figures, a cuboid is represented only by one xy plane.

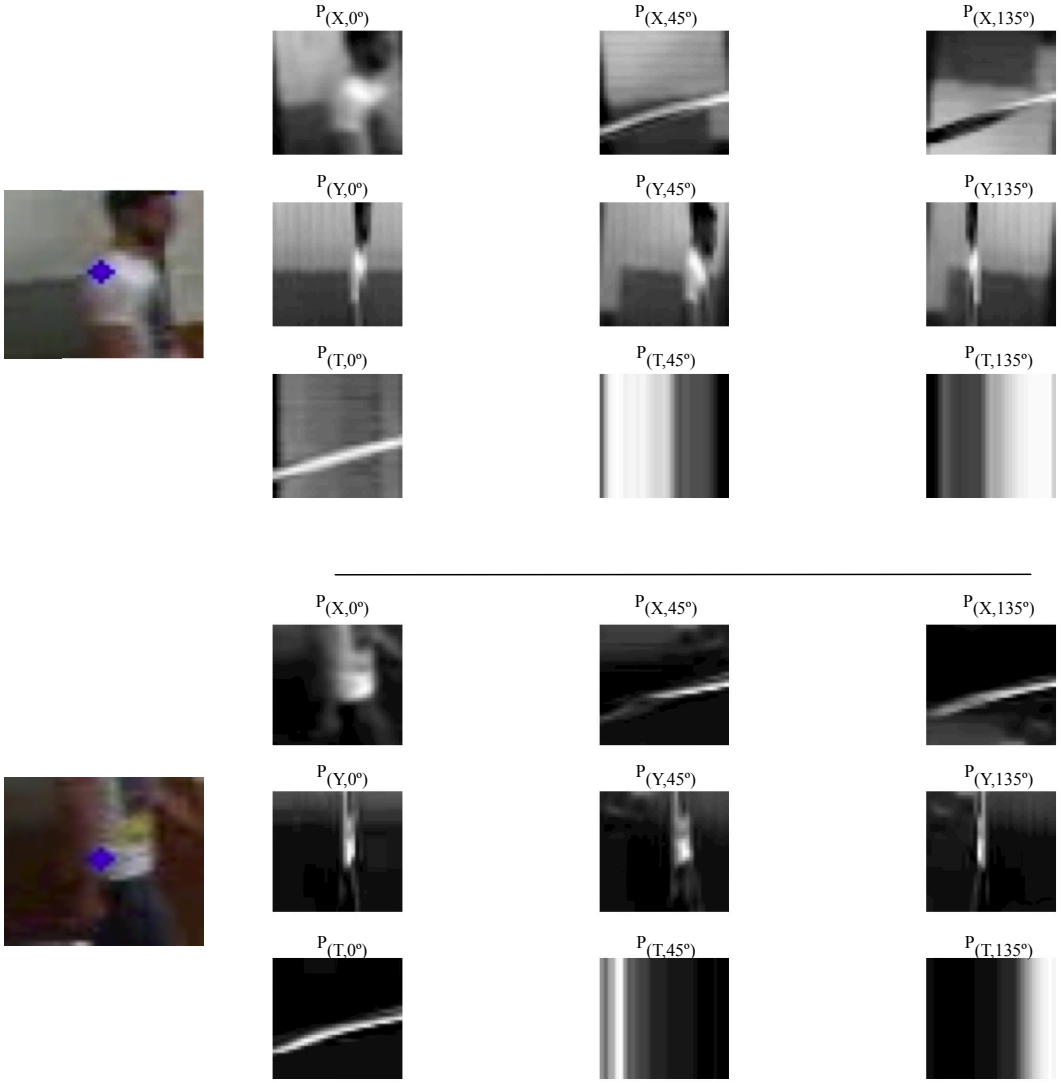

Figure 5.13: The nine symmetry planes for two different cuboids.

The correspondent HOG descriptors are shown in Figure 5.14.

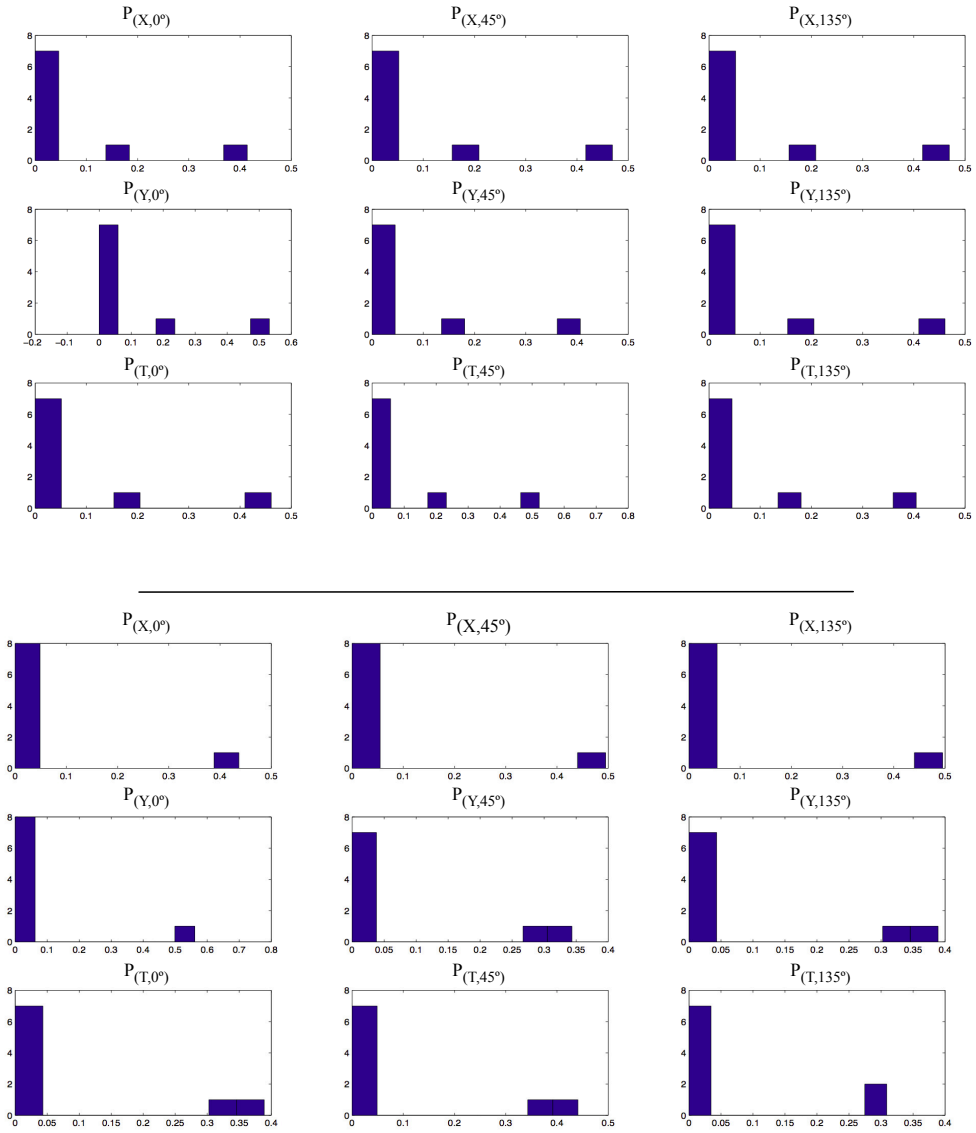

Figure 5.14: Histograms of Oriented Gradients for each plane of Figure 5.13.

The HOG descriptor of each plane is a 80 element vector. Concatenating the nine different descriptors in one vector results in a final descriptor vector with 720 elements.

5.3.3 COV3D

Recalling Section 3.5, a RCM is obtained from each cuboid, where each voxel is represented by a feature vector, defined in Equation 3.11.

The gradient features g (Equation 3.12) were calculated with the Sobel operator from OpenCV, with a kernel size of $k = 3$. On the other hand, the flow features f (Equation 3.13) were determined with the dense Farneback's Optical Flow operator from OpenCV.

Figure shows the RCM resulting from two different cuboids.

Figure 5.15: The RCM matrix of two different cuboids.

As mentioned in Section 3.5 the final descriptor vector is extracted from the upper triangle of the covariance matrix, reshaped as a vector. Since the covariance matrix has a 15×15 size, the final COV3D descriptor is a 120 element vector.

5.4 Event Evaluation

This work follow the same annotation used in [25] and [42], where a frame range is defined for each unusual event. Once the algorithm detects at least one frame in the annotated range, the detection is counted as correct. On the other hand, false alarm is also measured in the same way: at least one frame is fired outside the annotated range, then it is counted as false alarm.

5.4.1 Sparse Coding Formulation

In order to evaluate the normality of an event, it was assumed the existence of an initial dictionary learned from what are considered as usual events. The dictionary is a matrix $D_{s_d \times b}$, where s_d is the size of the descriptor and b is the number of bases of the dictionary. Different number of bases has been tested. Figure 5.16 shows the accumulated reconstruction weight vectors for two different videos after 1000 frames with only usual events.

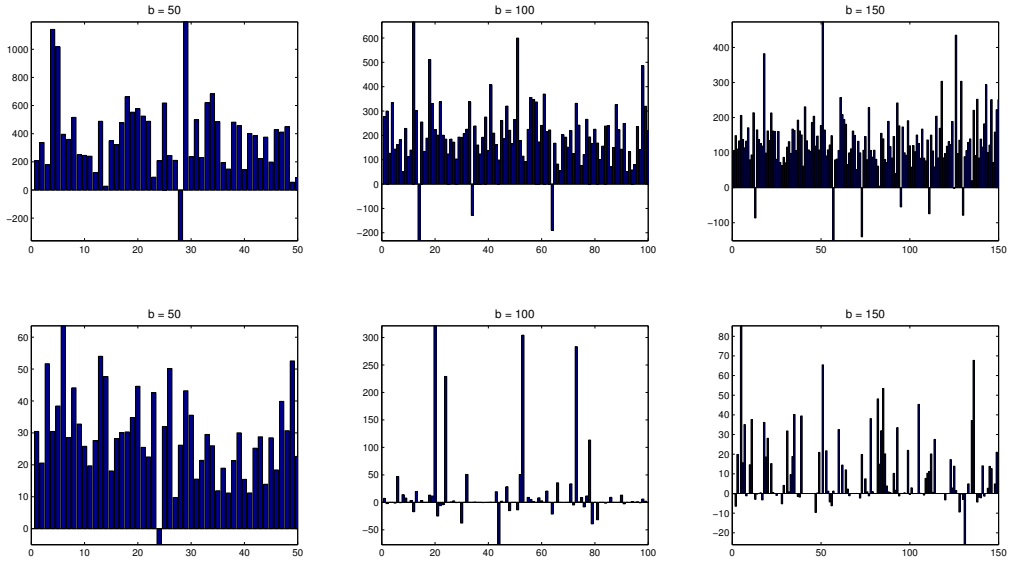

Figure 5.16: Reconstruction weight vectors for different number of bases b. The first row is verified for video where the dictionary is learned from multiple events, while the second row is observed for a dictionary learned from a video with one single event.

For $b = 150$ the dictionary learned for the second video was created with 54 bases initialized randomly. This is due to the small number of different descriptors vectors when compared to the number of required bases. This affect the results, since the random bases can be used to reconstruct unusual events, classifying them as usual. However, using a dictionary with few bases also affect the results. For example, in videos with multiple events, a small number of bases can be insufficient to represent multiple events in discriminatory manner. This situation is verified for *LabBrisa* Video 1 with a learned dictionary of 50 bases. Table 5.1 compares the number of False Alarm (FA) and the mean of the objective function J (Equation 4.1) for the $b = 50$ and $b = 100$ cases.

	FA	$\bar{J}$
$b = 50$	11	1.32
$b = 100$	2	1.11

Table 5.1: Comparison between the results obtained with different number of bases of the dictionary for *LabBrisa* 1 using HOG/HOF descriptor.

An increase of $\bar{J}$ was observed when $b = 50$ as well as an increased number of false alarms. Based on this results the value adopted was $b = 100$.

The influence of λ_1 and λ_2 values (Section 4.1) are shown in Figure 5.17. High values of λ_1 increase the sparsity of the reconstruction weight vector. However,excessively high values induce a limitation on the number of the selected bases, leading to a reconstruction with only one base (Figure 5.17, $\lambda_1 = 0.3$ case). The presence of λ_2 remove this limitation. Furthermore, the penalization introduced by λ_2 ensure the uniqueness of the sparse coding solution which is required for the online dictionary update algorithm (Section 4.3). Despite this, a high λ_2 value decrease the sparsity of the reconstruction weight vector.

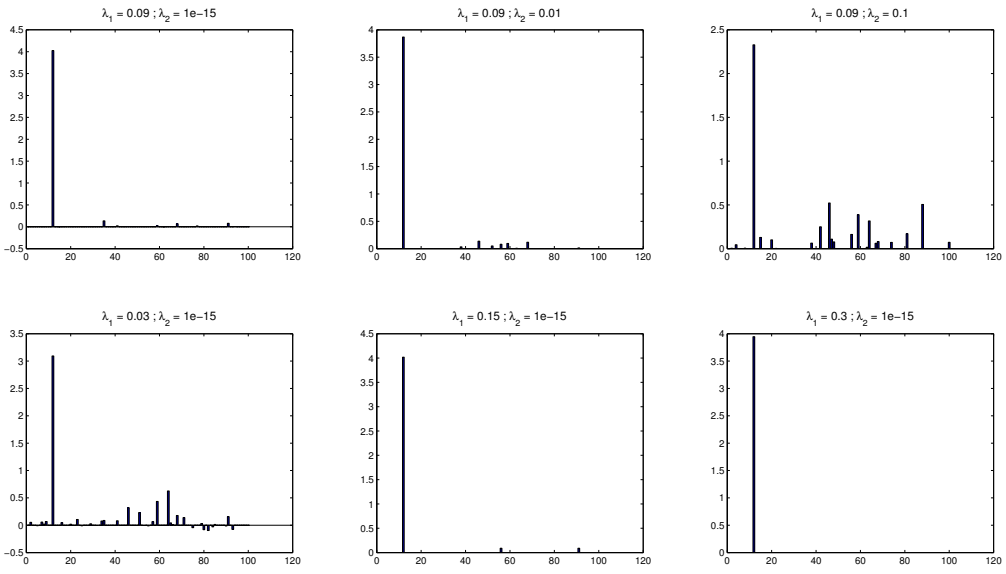

Figure 5.17: Reconstruction weight vectors for the cuboid with different λ_1 and λ_2 values.

The values used for λ_1 and λ_2 were:

$$\lambda_1 = 0.09 \tag{5.4}$$

$$\lambda_2 = 1 \times 10^{-15} \tag{5.5}$$

As regards the unusual event detector, the threshold ε defined in Section 4.4 was obtained from:

$$\varepsilon = 1.9 J_D \tag{5.6}$$

where J_D is obtained from Equation 4.1 using the initial dictionary as D and the data used to learn the dictionary as X.

5.4.2 Online Dictionary Update

As explained in Section 4.3, the dictionary is updated in an online fashion as the algorithm observes more data. In this section are compared the results of the unusual event detection keeping the dictionary fixed along the entire video sequence with the ones with the dictionary update. Tables 5.2, 5.3 and 5.4 show the results using videos *LabBrisa* 1, *YouTube* 1 and *YouTube* 4 respectively. The *LabBrisa* 1 video only has usual events while *YoutubeVideo* 1 and *YoutubeVideo* 4 has usual and unusual events (details about these videos are provided in Sections 5.4.4 and 5.4.5). The tables with the results have three categories: Correct Unusual Event Detection (CD), False Alarm (FA) and Missed Unusual Event Detection (MISS) and $\bar{J}$ (mean J along the entire sequence, obtained from Equation 4.1). To maintain the coherence between the results, the HOG/HOF descriptor was used in all experiments.

	CD	FA	MISS	$\bar{J}$
Fixed D	0	3	0	1.38
Updated D	0	2	0	1.11

Table 5.2: Comparison between a fixed D and an updated D for *LabBrisa* 1 using HOG/HOF descriptor.

	CD	FA	MISS	$\bar{J}$
Fixed D	48	0	0	1.69
Updated D	48	0	0	1.62

Table 5.3: Comparison between a fixed dictionary and an updated dictionary for *Youtube-Video* 1 using HOG/HOF descriptor.

	CD	FA	MISS	$\bar{J}$
Fixed D	8	3	0	1.35
Updated D	10	1	0	1.19

Table 5.4: Comparison between a fixed dictionary and an updated dictionary for *Youtube-Video* 4 using HOG/HOF descriptor.

It can be observed that the fixed dictionary method produce more false alarms than the one with the updated dictionary. This is due to the inability for adapting to the changing contents of the video.

According to [41], the algorithm efficiency can be improved by initializing A_t and B_t as:

$$A_0 = t_0 I \tag{5.7}$$

$$B_0 = t_0 D \tag{5.8}$$

where t_0 constant. After several tests, the value choosed was $t_0 = 10$. Although this initialization produce slightly better results, it is not critical for the overall performance of the dictionary update.

5.4.3 Unusual Event Detector with KTH Human Actions Dataset

The KTH Human Actions Dataset can be used to evaluate the unusual event detector under favourable conditions, without occlusions. Although, some human actions are physiognomically similar to others, like *jogging vs running* or *handwaving vs handclapping*. To evaluate the unusual event detector, the initial dictionary was learned from sequences of one type of human action at a time and the unusual event detector was tested with sequences of the 6 types of actions in the dataset. This process was repeated for the three descriptors presented in Chapter 3.

The results are shown in Tables 5.5, 5.6 and 5.7. Each row represent the human action used to learn the initial dictionary, while each column represent the human action used to test the unusual event detector. Since the video sequences in this dataset contains only one event, the results are presented in a error rate r as follows:

$$r = \frac{F}{T} \tag{5.9}$$

where F is the number of incorrect evaluations and T is the number of sliding windows tested along the video sequence. It was considered as an incorrect evaluation the false alarms and the missed unusual event detections. The highest values are in bold text.

	Walking	Jogging	Running	Handwaving	Handclapping	Boxing
Walking	0	0.09	0	0	0	0
Jogging	0.06	0.12	0.13	0	0	0
Running	0	0.1	0.06	0	0	0
Handwaving	0	0	0	0.16	0.19	0
Handclapping	0	0	0	**0.27**	0.15	0
Boxing	0	0	0	0	0	0

Table 5.5: Results of the unusual event detector using HOG/HOF descriptor for KTH Human Action dataset

	Walking	Jogging	Running	Handwaving	Handclapping	Boxing
Walking	0.05	0.14	0	0.03	0	0.11
Jogging	0.14	0.3	0.18	0.21	0.15	0.18
Running	0.14	0.25	0.29	0.23	0.08	0.14
Handwaving	0	0	0	0.17	0.38	**0.39**
Handclapping	0	0	0	0.29	0.15	**0.39**
Boxing	0	0	0	0.17	0	0.14

Table 5.6: Results of the unusual event detector using HOG-NSP descriptor for KTH Human Action dataset

	Walking	Jogging	Running	Handwaving	Handclapping	Boxing
Walking	0.03	0.53	0.41	0	0	0
Jogging	0.28	0.06	0.51	0	0	0
Running	0.21	**0.64**	0.06	0	0	0
Handwaving	0	0	0	0	0.16	0.19
Handclapping	0	0	0.05	0.26	0	0.29
Boxing	0	0.09	0.11	0.27	0.11	0

Table 5.7: Results of the unusual event detector using COV3D descriptor for KTH Human Action dataset

Observing Tables 5.5, 5.6 and 5.7 it is possible to conclude that the HOG/HOF descriptor yields the best results. It can also be noticed that the relation *jogging vs running* was the one that caused the worst results. More specifically, COV3D descriptor shown many difficulties to distinguish between walking, jogging and running.

Lastly, different human actions were combined for the dictionary learning step. The idea is to know if the detector detracts the ability to correctly evaluate the normality of events. The test video clip, referred as *KTH-All Actions* contains different sequences of all human actions presents in the dataset, in a total 6000 frames. Table 5.8 shows the results using a dictionary learned from walking and handwaving sequences, while for Table 5.9 it was used a dictionary learned from running and boxing sequences. The results are presented in three different categories: CD, FA and MISS.

	CD	FA	MISS
HOG/HOF	104	7	7
HOG-NSP	79	28	11
COV3D	73	19	26

Table 5.8: Results of the unusual event detector for *KTH-All Actions* video with a dictionary learned from walking and handwaving sequences.

	CD	FA	MISS
HOG/HOF	106	11	1
HOG-NSP	79	19	20
COV3D	63	14	41

Table 5.9: Results of the unusual event detector for *KTH-All Actions* video with a dictionary learned from running and boxing sequences.

The results presented in Tables 5.8 and 5.9 can be evaluated with a $F1$ score [43]. The $F1$ score measures the test's accuracy as follows:

$$F1 = \frac{2TP}{2TP + FP + FN} \tag{5.10}$$

where TP are the true positives (correct unusual event detections), TN the true negatives (false alarms) and FN the false negatives (missed unusual event detections). An $F1$ score equal to 1 represent the best value while $F1 = 0$ is the worst value.

Taking into account the results of Tables 5.8 and 5.9, the $F1$ scores for *KTH-All Actions* video are shown in Table 5.10.

	HOG/HOF	HOG-NSP	COV3D
$F1$	0.94	0.80	0.73

Table 5.10: $F1$ score for KTH Human Actions Dataset with multi-event dictionaries.

From Tables 5.8, 5.9 and 5.10 it is clear that the HOG/HOF descriptor yields the best results. As mentioned above, the HOG-NSP and specially the COV3D descriptors have some difficulties to distinguish actions like running and jogging. This handicap is aggravated when the dictionary has multiple events, leading to a reduction of the performance of these descriptors.

5.4.4 Unusual Event Detector with *LabBrisa* Videos

The *LabBrisa* videos are important to evaluate the robustness of the detector when dealing with multiple events and occlusions. Tables 5.12 and 5.13 present the results in three different

categories: CD, FA and MISS. For *LabBrisa* 1 the results are only evaluated by the number of FAs.

Table 5.11 exhibit the unusual event detection results for *LabBrisa* 1.

	HOG/HOF	HOG-NSP	COV3D
FA	2	9	9

Table 5.11: Number of false alarms obtained for *LabBrisa* 1.

The results of the detector applied to *LabBrisa* Video 2 are shown in Table 5.12.

	CD	FA	MISS
HOG/HOF	15	0	0
HOG-NSP	13	0	2
COV3D	14	0	1

Table 5.12: Results of the unusual event detector for *LabBrisa* 2.

Table 5.13 shows the results of the unusual event detector applied to *LabBrisa* 3.

	CD	FA	MISS
HOG/HOF	17	1	1
HOG-NSP	15	1	3
COV3D	18	1	0

Table 5.13: Results of the unusual event detector for *LabBrisa* 3.

Taking into account the results of Tables 5.12 and 5.4, the $F1$ score defined in subsection 5.4.4 for *LabBrisa* Videos are shown in Table 5.14.

	HOG/HOF	HOG-NSP	COV3D
$F1$	0.97	0.90	0.97

Table 5.14: $F1$ score for *LabBrisa* Videos.

Observing Table 5.14 it can be verified that the $F1$ scores are quite similar for the three descriptors. Despite some MISS for the HOG-NSP descriptor case (Tables 5.12 and 5.13), the detector has properly gauged the normality of the majority of the events.

5.4.5 Unusual Event Detector with *Youtube* Videos

In this section, the unusual event detector is applied to a number of videos downloaded from *YouTube*, after learn the initial dictionary from a small initial portion of the video. As in Section 5.4.4 the results have three categories: CD, FA and MISS.

The results of the unusual event detector applied to *YoutubeVideo* 1 are shown in Table 5.15.

	CD	FA	MISS
HOG/HOF	48	0	0
HOG-NSP	48	0	0
COV3D	48	0	0

Table 5.15: Results of the unusual event detector for *YoutubeVideo* 1.

Table 5.16 shows the results of the unusual event detector applied to *YoutubeVideo* 2.

	CD	FA	MISS
HOG/HOF	25	2	0
HOG-NSP	25	1	1
COV3D	26	1	0

Table 5.16: Results of the unusual event detector for *YoutubeVideo* 2

The results of the unusual event detector applied to *YoutubeVideo* 3 are shown in Table 5.17.

	CD	FA	MISS
HOG/HOF	37	4	0
HOG-NSP	38	2	1
COV3D	39	2	0

Table 5.17: Results of the unusual event detector for *YoutubeVideo* 3.

Table 5.18 shows the results of the unusual event detector applied to *YoutubeVideo* 4.

	CD	FA	MISS
HOG/HOF	10	1	0
HOG-NSP	9	2	0
COV3D	10	0	1

Table 5.18: Results of the unusual event detector for *YoutubeVideo* 4.

Taking into account the results of Tables 5.15, 5.16. 5.17 and 5.4, the $F1$ score defined in subsection 5.4.4 for *LabBrisa* Videos are shown in Table 5.19.

	HOG/HOF	HOG-NSP	COV3D
$F1$	0.97	0.97	0.98

Table 5.19: $F1$ score for *LabBrisa* Videos.

It can be observed that the $F1$ score is practically identical for the three descriptors. Furthermore the acquired values show a quite good accuracy in the evaluation of the normality of events in *"real-world"* videos.

5.4.6 Time Complexity

In addiction to the capacity to correctly evaluate the normality of an event, the response time of the system is another factor with particular relevance. In some situations one delayed unusual event detection can be as severe as one missed detection. Therefore, this subsection

compares the temporal performance of the unusual event detector for the three implemented descriptors.

Table 5.20 compares the speed of the detector for each descriptor. The values presented are an average calculated for all videos in one dataset.

	KTH Dataset	*LabBrisa* Videos	*YouTube* Videos
HOG/HOF	$15.8fps$	$7.6fps$	$9.7fps$
HOG-NSP	$33fps$	$19.4fps$	$30.5fps$
COV3D	$8.3fps$	$2.4fps$	$5.9fps$

Table 5.20: Temporal performance of the unusual event detector.

The descriptor for which was verified the best temporal performance was HOG-NSP. One of the main reasons for that is the need to calculate the optical flow in the HOG/HOF and COV3D descriptors, which is significantly slower than the HOGs computation. It was also noted that the COV3D descriptor was the one with the worst speed results.

Chapter 6

Conclusion

This dissertation presents a methodology that is capable of detect unusual events on video sequences or video streams. Resorting to a sparse coding formulation, the proposed methodology reconstructs the observed events from a dictionary learned only from usual events and evaluate their normality based on the correspondent reconstruction error. Each event is represented in a sparse manner, by a group of spatio-temporal regions of interest, called cuboids. This sparse representation of events leads to a memory saving and an increased processing speed, when compared to dense representations. A cuboid is described by a vector, obtained by a local features descriptor. In this work, three different descriptors have been implemented: HOG/HOF, HOG-NSP and COV3D. It was observed that the local descriptor performance is essential to obtain an efficient unusual event detector, since cuboids need to be described in a discriminatory manner in order to be distinguished from each other. Finally, the dictionary is updated with the newly observed event. This online update grant the unusual event detector with the capacity to adapt his knowledge of what is an usual event to behaviour changes of the environment.

Throughout the accomplishment this dissertation, important conclusions have been made. Firstly, it was observed that the three descriptors used in this work are capable to describe cuboids in a discriminatory manner, giving the system the capacity to efficiently classify unusual events. This also proves the robustness of the sparse coding formulation when used for event classification. Secondly, it has been concluded that the HOG/HOF descriptor yields to the best results, when compared to HOG-NSP and COV3D. Nevertheless, their processing time gives rise to certain hindrances when applying for a real-time system. This problem is also verified in COV3D descriptor. However, with some additional work, this system could be implemented using distributed and parallel processing, which will lead to shorter execu-

tion times. Furthermore, having better hardware can always decrease the execution time. Moreover, it has been observed that the combination of intensity gradients and optical flow features is an efficient approach when used as a local descriptor (HOG/HOF and COV3D). Concerning to HOG-NSP, its main advantage is the speed processing which is considerably greater than the ones showed by HOG/HOF and COV3D.

Concluding, this work presents an unusual event detector that can evaluate the normality of events in different environments (indoor, outdoor) and containing different categories (human, vehicles, etc.) with promising results.

In the future, similar experiments could be made with longer videos extracted from real video-surveillance scenarios, like underground stations, airports, etc. Another important point is to search for a descriptor with the discriminatory capacity of HOG/HOF descriptor and the speed performance of HOG-NSP in order to make this unusual event detector a solution that could be applied in real-time video-surveillance systems with superior results.

Bibliography

[1] F. Souza, "An evaluation on color invariant based local spatiotemporal features for action recognition," Master's thesis, University of Minas Gerais, 2011.

[2] N. Dalal, B. Triggs, and C. Schmid, "Human detection using oriented histograms of flow and appearance," in *Proceedings of the 9th European conference on Computer Vision - Volume Part II*, ECCV'06, (Berlin, Heidelberg), pp. 428–441, Springer-Verlag, 2006.

[3] E. Norouznezhad, M. T. Harandi, A. Bigdeli, M. Baktash, A. Postula, and B. C. Lovell, "Directional space-time oriented gradients for 3d visual pattern analysis," in *Proceedings of the 12th European conference on Computer Vision - Volume Part III*, ECCV'12, (Berlin, Heidelberg), pp. 736–749, Springer-Verlag, 2012.

[4] N. M. Oliver, B. Rosario, and A. P. Pentland, "A bayesian computer vision system for modeling human interactions," *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, vol. 22, no. 8, pp. 831–843, 2000.

[5] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, vol. 22, pp. 747–757, 2000.

[6] S. Hongeng and R. Nevatia, "Multi-agent event recognition.," in *ICCV*, pp. 84–93, 2001.

[7] N. Ghanem, D. Dementhon, D. Doermann, and L. Davis, "Representation and recognition of events in surveillance video using petri nets," in *In: Proceedings of Conference on Computer Vision and Pattern Recognition Workshops CVPRW*, p. 2004, 2004.

[8] H. Zhong, J. Shi, and M. Visontai, "Detecting unusual activity in video," in *Proceedings of the 2004 IEEE computer society conference on Computer vision and pattern recognition*, CVPR'04, (Washington, DC, USA), pp. 819–826, IEEE Computer Society, 2004.

[9] P. Smith, Niels, and M. Shah, "Temporal Boost for Event Recognition," in *10th IEEE International Conference on Computer Vision*, IEEE, Oct. 2005.

[10] O. Boiman and M. Irani, "Detecting irregularities in images and in video," *Int. J. Comput. Vision*, vol. 74, pp. 17–31, Aug. 2007.

[11] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *Proceedings of the 14th International Conference on Computer Communications and Networks*, ICCCN '05, (Washington, DC, USA), pp. 65–72, IEEE Computer Society, 2005.

[12] C. Schmid, R. Mohr, and C. Bauckhage, "Evaluation of interest point detectors," *Int. J. Comput. Vision*, vol. 37, pp. 151–172, June 2000.

[13] I. Laptev and T. Lindeberg, "Space-time interest points," in *IN ICCV*, pp. 432–439, 2003.

[14] G. Willems, T. Tuytelaars, and L. Gool, "An efficient dense and scale-invariant spatio-temporal interest point detector," in *Proceedings of the 10th European Conference on Computer Vision: Part II*, ECCV '08, (Berlin, Heidelberg), pp. 650–663, Springer-Verlag, 2008.

[15] H. Jhuang, T. Serre, L. Wolf, and T. Poggio, "A biologically inspired system for action recognition," in *In ICCV*, pp. 1–8, 2007.

[16] J. C. Niebles, H. Wang, and L. Fei-Fei, "Unsupervised learning of human action categories using spatial-temporal words," *Int. J. Comput. Vision*, vol. 79, no. 3, pp. 299–318, 2008.

[17] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: A local svm approach," in *Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 3 - Volume 03*, ICPR '04, (Washington, DC, USA), pp. 32–36, IEEE Computer Society, 2004.

[18] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, pp. 91–110, Nov. 2004.

[19] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Comput. Vis. Image Underst.*, vol. 110, pp. 346–359, June 2008.

[20] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *In CVPR*, pp. 886–893, 2005.

[21] Y. Ke and R. Sukthankar, "Pca-sift: a more distinctive representation for local image descriptors," in *Proceedings of the 2004 IEEE computer society conference on Computer vision and pattern recognition*, CVPR'04, (Washington, DC, USA), pp. 506–513, IEEE Computer Society, 2004.

[22] A. Klaser, M. Marszalek, and C. Schmid, "A spatio-temporal descriptor based on 3d-gradients," in *British Machine Vision Conference*, pp. 995–1004, sep 2008.

[23] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *CVPR'08*, pp. –1–1, 2008.

[24] P. Bilinski and F. Bremond, "Evaluation of local descriptors for action recognition in videos," in *Proceedings of the 8th international conference on Computer vision systems*, ICVS'11, (Berlin, Heidelberg), pp. 61–70, Springer-Verlag, 2011.

[25] B. Zhao, L. Fei-Fei, and E. P. Xing, "Online detection of unusual events in videos via dynamic sparse coding," in *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '11, (Washington, DC, USA), pp. 3313–3320, IEEE Computer Society, 2011.

[26] H. Lee, A. Battle, R. Raina, and A. Y. Ng, "Efficient sparse coding algorithms," in *In NIPS*, pp. 801–808, NIPS, 2007.

[27] B. A. Olshausen, "Sparse coding of time-varying natural images," in *IN PROC. OF THE INT. CONF. ON INDEPENDENT COMPONENT ANALYSIS AND BLIND SOURCE SEPARATION*, pp. 603–608, 2000.

[28] M. S. Lewicki and T. J. Sejnowski, "Learning overcomplete representations," *Neural Comput.*, vol. 12, pp. 337–365, Feb. 2000.

[29] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, pp. 607–609, 1996.

[30] B. A. Olshausen and D. J. Fieldt, "Sparse coding with an overcomplete basis set: a strategy employed by v1," *Vision Research*, vol. 37, pp. 3311–3325, 1997.

[31] J. Xu, S. Denman, S. Sridharan, C. Fookes, and R. Rana, "Dynamic texture reconstruction from sparse codes for unusual event detection in crowded scenes," in *Proceedings of the 2011 joint ACM workshop on Modeling and representing events*, J-MRE '11, (New York, NY, USA), pp. 25–30, ACM, 2011.

[32] M. T. Harandi, C. Sanderson, A. Sanin, and B. C. Lovell, "Spatio-temporal covariance descriptors for action and gesture recognition," in *Proceedings of the 2013 IEEE Workshop on Applications of Computer Vision (WACV)*, WACV '13, (Washington, DC, USA), pp. 103–110, IEEE Computer Society, 2013.

[33] J. R. Movellan, "Tutorial on Gabor Filters," *Tutorial paper http://mplab.ucsd.edu/tutorials/pdfs/gabor.pdf*, 2008.

[34] H. Wang, M. M. Ullah, A. Klaser, I. Laptev, and C. Schmid, "Evaluation of local spatio-temporal features for action recognition," in *University of Central Florida, U.S.A*, 2009.

[35] G. Zhao and M. Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, pp. 915–928, June 2007.

[36] O. Tuzel, F. Porikli, and P. Meer, "Pedestrian detection via classification on riemannian manifolds," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, pp. 1713–1727, Oct. 2008.

[37] M. T. Harandi, C. Sanderson, A. Wiliem, and B. C. Lovell, "Kernel analysis over riemannian manifolds for visual recognition of actions, pedestrians and textures," in *Proceedings of the 2012 IEEE Workshop on the Applications of Computer Vision*, WACV '12, (Washington, DC, USA), pp. 433–439, IEEE Computer Society, 2012.

[38] R. Sivalingam, D. Boley, V. Morellas, and N. Papanikolopoulos, "Tensor sparse coding for region covariances," in *Computer Vision - ECCV 2010, 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV* (K. Daniilidis, P. Maragos, and N. Paragios, eds.), vol. 6314 of *Lecture Notes in Computer Science*, pp. 722–735, Springer, 2010.

[39] B. Olshausen and D. Field, "Sparse coding of sensory inputs," *Current Opinion in Neurobiology*, vol. 14, pp. 481–487, Aug. 2004.

[40] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society, Series B*, vol. 67, pp. 301–320, 2005.

[41] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, (New York, NY, USA), pp. 689–696, ACM, 2009.

[42] J. Kim and K. Grauman, "Observe locally, infer globally: A space-time mrf for detecting abnormal activities with incremental updates," in *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pp. 2921–2928, IEEE, 2009.

[43] Wikipedia, "Receiver operating characteristic — Wikipedia, the free encyclopedia," 2013. [Online; accessed 03-September-2013].

[44] J. a. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu, "Semantic segmentation with second-order pooling," in *Proceedings of the 12th European conference on Computer Vision - Volume Part VII*, ECCV'12, (Berlin, Heidelberg), pp. 430–443, Springer-Verlag, 2012.

[45] R. R. Curtin, J. R. Cline, N. P. Slagle, W. B. March, P. Ram, N. A. Mehta, and A. G. Gray, "MLPACK: A scalable C++ machine learning library," *Journal of Machine Learning Research*, vol. 14, pp. 801–805, 2013.

[46] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *Annals of Statistics*, vol. 32, pp. 407–499, 2004.