FAST AND ROBUST OPTIMIZATION APPROACHES FOR PEDESTRIAN DETECTION

VICTOR HUGO CUNHA DE MELO

FAST AND ROBUST OPTIMIZATION APPROACHES FOR PEDESTRIAN DETECTION

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

ORIENTADOR: WILLIAM ROBSON SCHWARTZ
COORIENTADOR: DAVID MENOTTI

Belo Horizonte
Janeiro de 2014

VICTOR HUGO CUNHA DE MELO

FAST AND ROBUST OPTIMIZATION APPROACHES FOR PEDESTRIAN DETECTION

Dissertation presented to the Graduate Program in Computer Science of the Universidade Federal de Minas Gerais in partial fulfillment of the requirements for the degree of Master in Computer Science.

ADVISOR: WILLIAM ROBSON SCHWARTZ
CO-ADVISOR: DAVID MENOTTI

Belo Horizonte
January 2014

Melo, Victor Hugo Cunha de

M528f Fast and Robust Optimization Approaches for Pedestrian Detection / Victor Hugo Cunha de Melo. — Belo Horizonte, 2014
xxvi, 62 f. : il. ; 29cm

Dissertação (mestrado) — Universidade Federal de Minas Gerais
Orientador: William Robson Schwartz
Coorientador: David Menotti

1. Computação - Teses. 2. Visão por Computador - Teses. 3. Filtragem Aleatória. 4. Partial Least Squares. 5. Variable Importance on Projection. 6. Detecção de Pedestres. I Orientador. II Coorientador. Título.

CDU 519.6*82.10(043)

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

Fast and robust optimization approaches for pedestrian detection

VICTOR HUGO CUNHA DE MELO

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

PROF. WILLIAM ROBSON SCHWARTZ - Orientador
Departamento de Ciência da Computação - UFMG

PROF. DAVID MENOTTI GOMES - Coorientador
Departamento de Computação - UFOP

PROF. CLÁUDIO ROSITO JUNG
Departamento de Informática Aplicada - UFRGS

PROF. JEFERSSON ALEX DOS SANTOS
Departamento de Ciência da Computação - UFMG

PROF. MARIO FERNANDO MONTENEGRO CAMPOS
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 25 de fevereiro de 2014.

Acknowledgments

First and foremost, I would like to express my gratitude to my advisor William Schwartz. From the beginning, William has always been there to listen and to advise me. William's enthusiasm, knowledge, and patience has always been a great inspiration to me. He has patiently helped me in refining my inarticulate ideas, teaching me what questions should be asked and how to express in scientific communication. He has been a great advisor, a mentor, and a friend. I have learned a lot under his guidance.

A special thanks goes to my coadvisor, David Menotti, who has always been an inspiration to me. David has always been a friend, supporting me and pushing me since my undergraduation. During this Master's, David provided insightful thoughts, helping me everytime I thought I had reached a dead end. Thank you for inspiring me the interest for computer science and computer vision, and to pursue the academic career.

I would like to thank the Master's thesis committee, Cláudio Rosito, Jefferson Santos, and Mario Campos. Thank you for carefully reviewing the text and for your insights on how this work could be improved.

During this Master's, I have counted on the assistance of several friends. Thank you to my colleague Samir. Without him, this work would not be possible. To my long time friends, Antonio Carlos and Suellen, who always supported me. To the SSIG group: Artur, Cássio, César, Cristiane, Jéssica, Ricardo, and all the members who were part of it, for the insightful discussions and the fun moments. To the friends I made in Verlab: Balbino, Cláudio, Douglas, Drews, Elerson, Erickson, Fernando, and Vinicius, thank you for presenting me a whole new world and for making my transition more smoothly. To the PPGCC secretariat, who always attended me kindly and patiently. Finally, I would like to specially thank my roomate, Fernando, for being a great friend and making everything more amusing.

This research would not have been possible without the financial support of the Brazilian National Research Council – CNPq (Grant #487529/2013-8) – and the

Minas Gerais Research Foundation - FAPEMIG (APQ-01294-12), and I would like to express my gratitude to them.

In conclusion, I would like to thank my parents Sebastião Porfírio and Ana Vera, my first advisors. Thank you for being the best parents that I could ever have. My brother Caio Hess, my first friend and colleague, thank you for always supporting me. And my final words go to my love, Mariana. Mariana probably suffered the most, watching closely my sleepless nights, patiently skipping weekends and vacations. She was also there to share with me the good moments, making this ride more joyful. I cannot express how grateful I am for everything you did for me.

"Regard all dharmas as dreams"
(Geshe Chekhawa)

Abstract

The large number of surveillance cameras available nowadays in strategic points of major cities provides a safe environment. However, the huge amount of data provided by the cameras prevents its manual processing, requiring the application of automated methods. Among such methods, pedestrian detection plays an important role in reducing the amount of data by locating only the regions of interest for further processing regarding activities being performed by agents in the scene. However, the currently available methods are unable to process such large amount of data in real time. Therefore, there is a need for the development of optimization techniques. Towards accomplishing the goal of reducing costs for pedestrian detection, we propose in this work two optimization approaches. The first approach consists of a cascade of rejection based on Partial Least Squares (PLS) combined with the propagation of latent variables through the stages. Our results show that the method reduces the computational cost by increasing the number of rejected background samples in earlier stages of the cascade. Our second approach proposes a novel optimization that performs a random filtering in the image to select a small number of detection windows, allowing a reduction in the computational cost. Our results show that accurate results can be achieved even when a large number of detection windows are discarded.

Keywords: Computer Vision, Pedestrian Detection, Rejection Cascade, Random Filtering, Partial Least Squares, Variable Importance on Projection, Visual Surveillance.

Resumo

O grande número de câmeras de vigilância hoje em dia disponíveis em pontos estratégicos das principais cidades fornece um ambiente seguro. No entanto, a enorme quantidade de dados geradas por estas câmeras impede o processamento manual, exigindo a aplicação de métodos automatizados. Entre estes métodos, a detecção de pedestres desempenha um papel importante na redução da quantidade de dados por localizar apenas as regiões de interesse para o tratamento posterior sobre as atividades a serem realizadas pelos agentes na cena. No entanto, os métodos de detecção de pedestres disponíveis atualmente são incapazes de processar tal quantidade de dados em tempo real. Portanto, é necessário utilizar técnicas de otimização para permitir a detecção em tempo real, mesmo quando grandes volumes de dados têm de ser processados. Para cumprir a meta de redução de custos para a detecção de pedestres, este trabalho propõe duas abordagens de otimização. A primeira abordagem consiste em uma cascata de rejeição baseada no método *Partial Least Squares* (PLS) e no método de *Variable Importance in Projection* (VIP), combinada com a propagação de variáveis latentes através dos estágios. Os resultados mostram que o método reduz o custo computacional, aumentando o número de amostras pertencentes ao fundo rejeitadas nos estágios iniciais da cascata. A segunda abordagem consiste em uma otimização baseada em uma filtragem aleatória na imagem para descartar um grande número de janelas de detecção rapidamente, permitindo uma redução do custo computacional. A avaliação experimental demonstra que pode ser obtido uma grande acurácia, mesmo quando um grande número de janelas de detecção é descartado.

Palavras-chave: Visão Computacional, Detecção de Pedestres, Cascata de Rejeição, Filtragem Aleatória, *Partial Least Squares*, *Variable Importance on Projection*.

List of Figures

List of Tables

List of Abbreviations

PLS Partial Least Squares

PCA Principal Component Analysis

FPS frames per second

FPPW False Positives Per Window

FPPI False Positives Per Image

PPS Pedestrian Protection Systems

SVM Support Vector Machines

VIP Variable Importance on Projection

HOG Histograms of Oriented Gradients

ADAS Advanced Driver Assistance Systems

DET Detection Error Tradeoff

RF Random Filtering

LR Location Regression

Contents

Chapter 1

Introduction

VIDEO SURVEILLANCE has been around us for almost a century and recently it suffered a huge growth due to the dropping prices of the cameras and the increasing network connectivity [Porikli et al., 2013]. Nowadays, we have a growing availability of visual data captured by surveillance cameras, which provides safer environments for people whom attend monitored environments. However, the large number of cameras to be monitored and consequently the large number of images that must be interpreted, precludes an effective manual processing and require a significant number of people dedicated to analyzing visual data. The ubiquity of video surveillance is advantageous for protection, but it is harder to monitor.

Although the large amount of visual data may provide more secure environments, their analysis becomes challenging when performed manually. In addition, most of the data do not present interesting events from the surveillance standpoint, turning it into a repetitive and monotonous task for humans. Hence, automatic understanding and interpretation of activities performed by humans in videos show great interest because such information can assist the decision making process of security agents.

Most surveillance systems usually employ human operators to monitor activities of interest. However, human operators are more susceptible to fatigue after a certain time of video monitoring. Hampapur et al. [2003] show that after 20 minutes of watching and evaluating monitor screens, the operator's focus weakens to well bellow acceptable levels. In addition, we cannot forget that operators are not always well-intentioned, such as recently happened on Araraquara (Brazil) in which the operators were using the surveillance cameras to inappropriately look at women[1]. On the other hand, smart surveillance tries to minimize these problems and can be used

[1]Story available at `http://folha.com/no1384502` (in Portuguese).

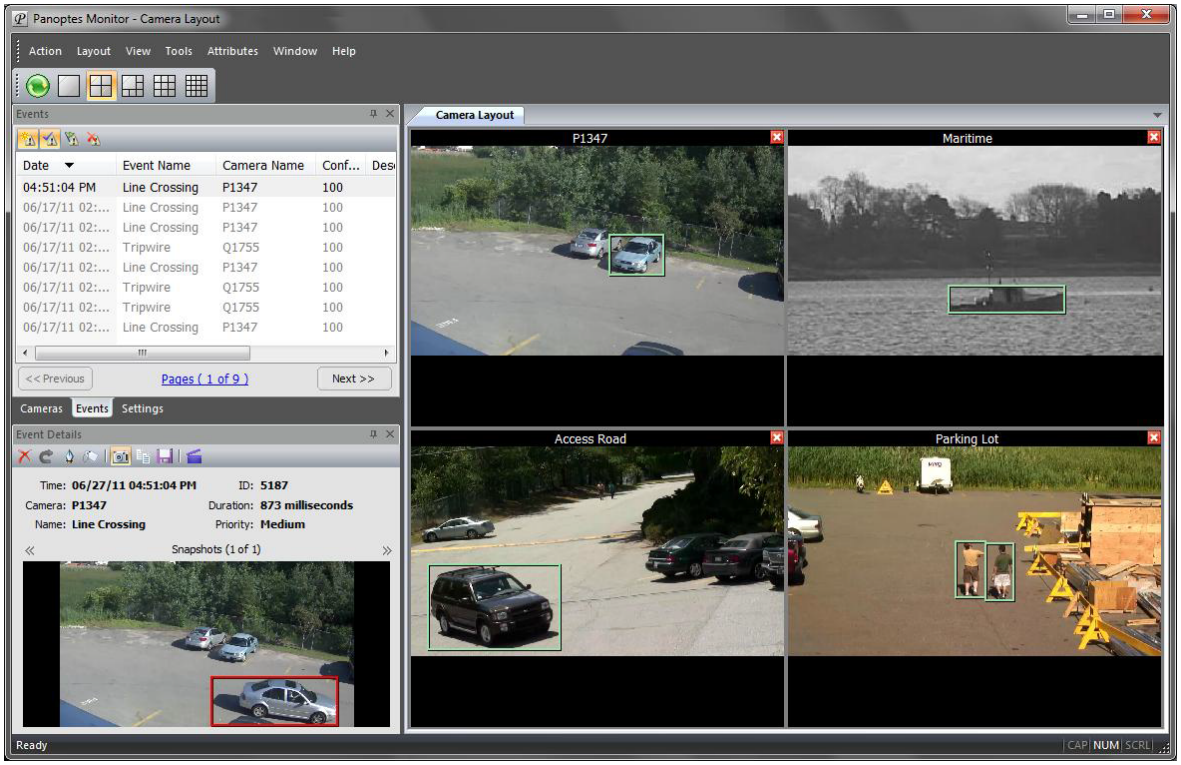


Figure 1.1. Panoptes: an example of a smart surveillance system, developed by intuVision. Source: `http://www.intuvisiontech.com/products/panoptes.php`.

to assist the operators, showing only relevant content instead of the full video.

Smart surveillance is defined by the employment of automatic video analysis technologies in video surveillance applications [Hampapur et al., 2003]. Smart surveillance systems can be designed to small processing tasks, such as detection of unusual events, or to more complex ones, such as semantic understanding of the activies happening on a video. In addition, it may be used to assist surveillance operators to maintain their focus only on important events. For instance, Figure 1.1 depicts an operating smart surveillance system, detecting pedestrians who are crossing dangerous regions, cars in parking lots, and ships. Regardless the range of operation of those systems, the first step is to detect the pedestrians' position in an image, since they are the most important agents in the scene.

Humans can be found in several environments, representing a key information for numerous applications. Given their importance, we are interested in monitoring them to determine how they interact with the environment. Therefore, we want to know their location and what activities they are performing to infer whether they may harm someone or something might harm them. This knowledge can be then

used to take preventive decisions to maintain the well being of people.

This work focuses on pedestrian detection, a subproblem of human detection. The term *pedestrian detection* differs from *human detection* in the range of poses a person may assume. We can distinguish them as the following. Human detection is more broad, including people in any pose, while pedestrian detection is restricted to bodies close to an upright position. Dalal and Triggs [2005] define pedestrian detection as "*the detection of mostly visible people in more or less upright poses, usually standing.*" For now on, we will only refer to pedestrian detection throughout this work. We focus on pedestrian detection because, besides of capturing the most important poses in video footages, pedestrians are the main agents interacting with the environment in surveillance videos. Hence, we define pedestrian detection as the following:

Definition 1.1 (Pedestrian Detection). *Given an input image, a sequence or a video footage, the Pedestrian Detection problem consists of locating the position of all persons close to an upright pose, covering their exactly height and weight.*

Pedestrian detection methods can be divided into two categories: holistic and part-based methods [Schwartz et al., 2009]. *Holistic methods* statistically analyze a detection window combined with the feature extraction to classify whether this window contains a pedestrian or not (see Figure 1.2(a) for an example). On the other hand, *part-based methods* consist of a generative process where detected parts of the human body are combined based on a prior human model (Figure 1.2(b)).

Several challenges are faced by the pedestrian detection problem [Geronimo et al., 2010]. Among them, there are changes in appearance due to different types of clothing, illumination changes and pose variations, low quality of the data acquired, and the small size of the pedestrian, which makes the detection process harder. In addition, a large number of applications require a high performance and reliable detection results, outlining the need for efficient and accurate pedestrian detection approaches.

1.1 Motivation

In some applications, pedestrian detection is one of the first steps of a larger system. For instance, consider the applications of Looking at People, which concerns tasks of visual analysis of humans including pose estimation, action recognition, and person reidentification. Figure 1.3 illustrates the relationship of these tasks with pedestrian detection. One can see that pedestrian detection is an elementary task, which other

(a) (b)

Figure 1.2. Example of holistic and part-based methods. (a) Holistic method. Source: Dalal and Triggs [2005]; (b) Part-based method. Source: Lin and Davis [2008].

tasks depend upon. Hence, pedestrian detection must be fast enough, otherwise the tasks depending on it will be executed after the events of interest have already happened and then being useless to take preventive actions. In addition, it must be accurate since the high-level tasks depends upon the data provided by detection (e.g., if poorly locations are used for action recognition, one might miss an action).

Real-time applications require accurate and fast answers regarding the presence of a pedestrian to take preventive actions. The automotive industry and the scientific community, for example, are currently researching intelligent systems integrated into the vehicle aimed at anticipating accidents to avoid or lessen its severity. These systems are called Advanced Driver Assistance Systems (ADAS) [Geronimo et al., 2010], in which Pedestrian Protection Systems (PPS) play an important role. The goal of a PPS is to detect the presence of pedestrians in a specific region around the vehicle, as illustrated in Figure 1.4, in such way that the driver can be notified about this presence or an automatic preventive action is taken to stop the vehicle.

Pedestrian detection can also be subject to several constraints, such as low processing power or a large amount of data. When multiple cameras are used, the amount of recorded data increases significantly, which is even more concerning if we consider high resolution cameras. For example, the bar graph in Figure 1.5 illustrates the amount of data generated by a single camera, at different frame rates. A camera widely used, of 640×480 pixels and operating at 30 frames per second (FPS), generates almost 10 megapixels per second. In higher definition, of 1920×1080 and 30 FPS, it generates more than 60 megapixels per second; the detection problem becomes intractable for the current pedestrian detection methods [Dollar et al., 2012]. Due to such constraints, these scenarios require fast methods with high detection

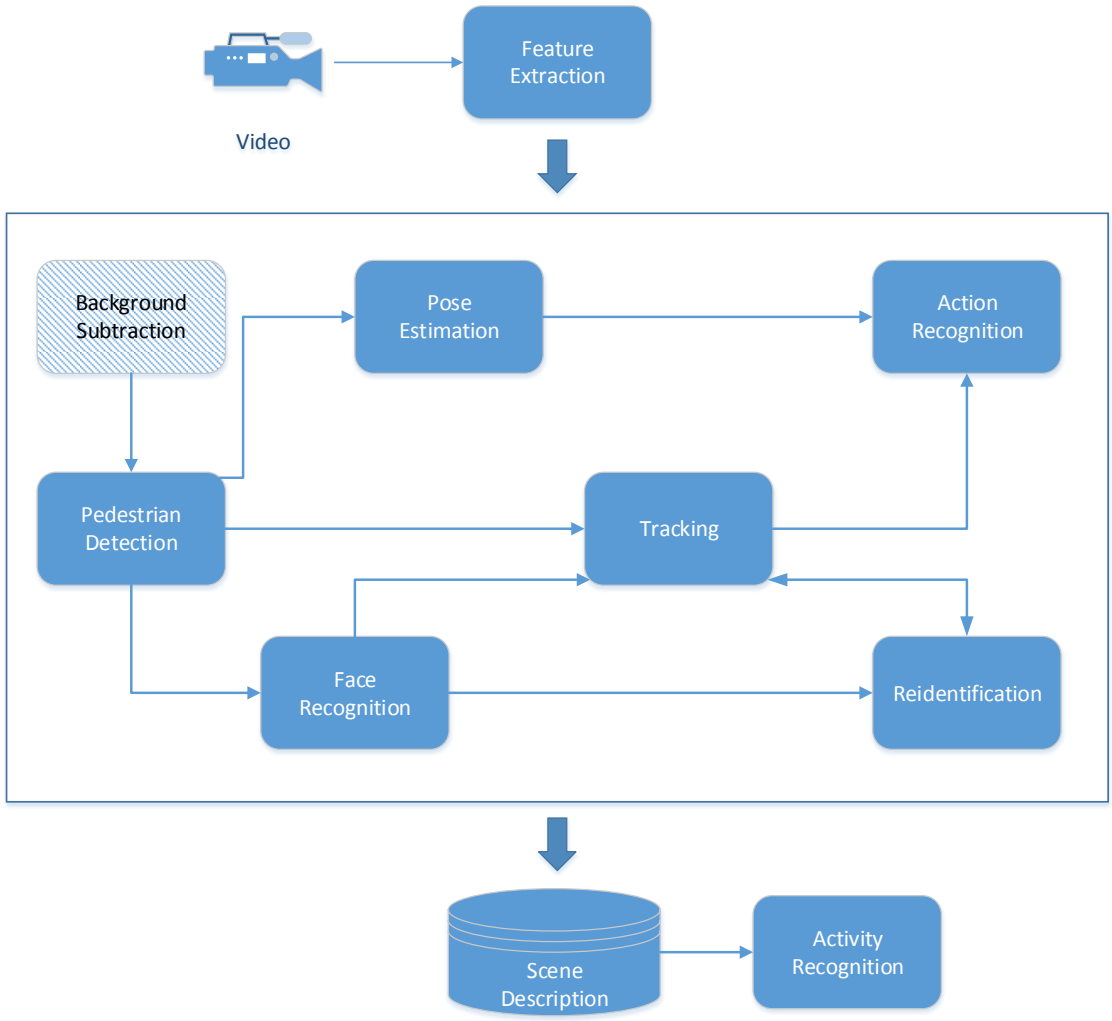


Figure 1.3. Looking at people and applications [Schwartz, 2012a]. Background subtraction is shaded because it is optional to the execution of the other tasks.

rate. Nevertheless, the majority of pedestrian detection methods are not fast enough for these applications [Dollar et al., 2012]. Therefore, it is desirable the development of methods to significantly reduce the computational cost. One way of achieving that is to focus on optimization approaches.

There are several optimization approaches to reduce the computational cost. The most common are those based on the computation of efficient feature descriptors [Dollár et al., 2014], cascade of rejection that incrementally increases the feature complexity [Marin et al., 2013], region of interest filtering using saliency detection to reduce the number of detection windows to be evaluated [Cheng et al., 2013], GPU-based approaches to parallelize the processing [Benenson et al., 2012a]. However,

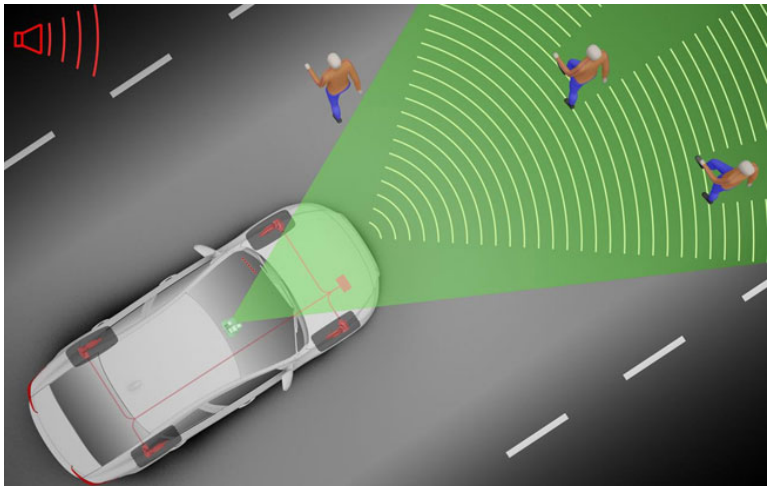


Figure 1.4. Example of car with a pedestrian protection system. Source: `http://www.caradvice.com.au/`.

such approaches still are not enough to achieve real time processing in most cases.

1.2 Dissertation's Goal

In this work we study the pedestrian detection problem, focusing at reducing its computational cost. The problem we are tackling can be defined in the following question:

Dissertation's Problem. *How to reduce the computational cost of pedestrian detection methods while keeping a high accuracy?*

This work addresses the aforementioned problem by proposing two different optimization approaches. In our first approach, described in details in Section 3.2, we propose a novel optimization of pedestrian detection methods based on sliding windows. The idea is to perform a random filtering in the image to select a very small number of detection windows and discard the remaining ones. Differently from other optimization techniques, the proposed approach does not perform any kind of processing in the discarded windows, which provides a significant speed-up. To the best of our knowledge, this is the first time such approach is employed to optimize pedestrian detection. Due to the random nature of the choice, the selected windows might be slightly shifted from the person's body, which need to be fixed before presenting them to a classifier. Therefore, a regression, referred to as *location regression*, is executed to adjust the location of each detection window in the image.

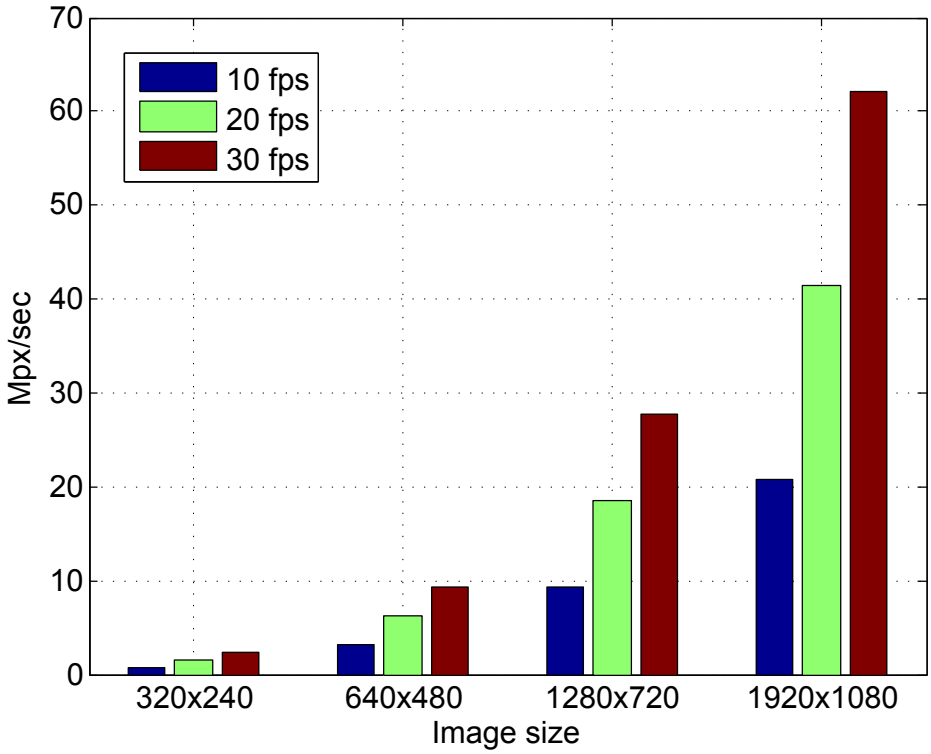


Figure 1.5. Amount of data recorded by a single camera at different frame rates and resolutions.

In our second approach, described in Section 3.3, we reduce the computational cost of the PLS Detector [Schwartz et al., 2009] by applying the Partial Least Squares (PLS) [Rosipal and Kramer, 2006] in a rejection cascade framework to reduce the number of projections and the amount of feature descriptors extracted (responsible for the majority of the computational cost). Variable Importance on Projection (VIP) [Wold et al., 1993] is applied to rank features according to their discriminative power, allowing the rejection of more samples in earlier stages of the cascade which effectively reduces the number of projections and extracted feature descriptors. In addition, this work also proposes the propagation of latent variables, estimated by PLS, from one stage to another, aiming at achieving high accuracy. The cascade resulting from the methodology described in this work is referred to as *PLS Cascade*.

1.3 Contributions

In this work, we propose algorithms to optimize pedestrian detection methods. The main contributions of this work are:

- Reduction of the computational cost of the PLS Detector, a widely employed

pedestrian detector;

- The application of Variable Importance on Projection (VIP) for feature ordering for fast training cascades of rejection;
- A new filtering approach, which can be applied on any sliding window based detector;

During the development of this work, we were awarded as one of the best works at the Seminar Week of the Graduation Program in Computer Science at Universidade Federal de Minas Gerais. In addition, we have produced some technical papers which have been submitted for publication. The following list provides references to these documents.

- Melo, V., Leão, S., Campos, M., Menotti, D., and Schwartz, W. (2013). *Fast pedestrian detection based on a Partial Least Squares Cascade*. In IEEE International Conference on Image Processing.
- Melo, V., Leão, S., and Schwartz, W. (2013). *Pedestrian Detection Optimization Based on Random Filtering*. In Workshop of Works in Progress (WIP) at Conference on Graphics, Patterns and Images (SIBGRAPI).
- Melo, V., Leão, S., Menotti, D., and Schwartz, W. (accepted). *An Optimized Sliding Window Approach to Pedestrian Detection*. In International Conference on Pattern Recognition.

1.4 Dissertation Organization

We have organized this dissertation into the following chapters. Chapter 2 reviews previous work on pedestrian detection and approaches to decrease the computational cost. Chapter 3 describes our proposed approaches for reducing the computational cost of the PLS Detector using a cascade of rejection and the random filtering approach. Chapter 4 shows our experimental evaluation. Finally, Chapter 5 points our final remarks.

Chapter 2

Related Work

Several pedestrian detection approaches have been proposed in the past years. In this chapter, we review mainly works that focus on reducing the computational cost and improving the detection rate. Initially, Section 2.1 addresses the main pedestrian detection methods and the state-of-the-art solutions in the literature. Section 2.2 describes the computational cost issue of pedestrian detection. Section 2.3 reviews different approaches employed for reducing the computational cost, including cascade of rejection and region of interest filtering, subjects directly related to our two solutions proposed in this work.

2.1 Pedestrian Detection Approaches

We present in this section the state-of-the-art on pedestrian detection methods based on Computer Vision. These methods may be divided into two classes, holistic and part-based [Schwartz et al., 2009]. We focus mainly on monocular detectors rather than stereo ones once stereo pairs of cameras are not always available, e.g, in surveillance systems.

2.1.1 Holistic-Based Detectors

Most pedestrian detection methods proposed in the literature are holistic. Such approaches learn a model for the whole target object. When performing detection, they try to match the entire model with the target object, in our case, the whole body of a pedestrian. In other words, the pedestrian is described by a single feature vector and is classified at once [Marin et al., 2013]. Holistic detectors can collect more dis-

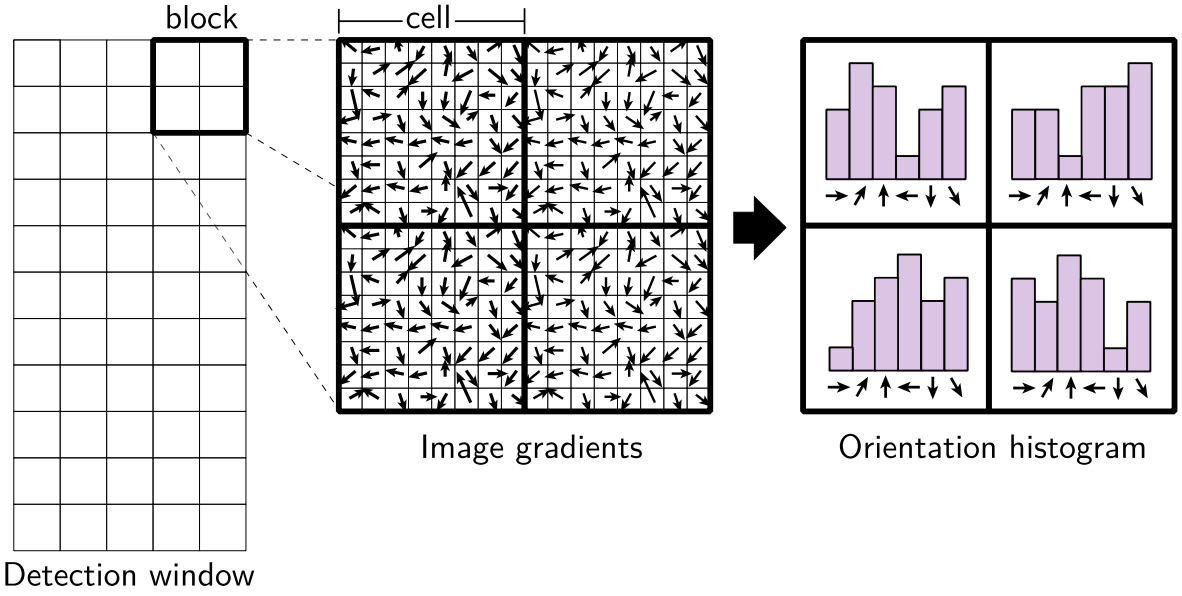


Figure 2.1. Illustration of HOG computation.

criminative information due to the large size of whole body, when compared to the sizes of the parts [Schwartz et al., 2011].

Among these methods, Dalal and Triggs [2005] contributed with a remarkable work for allowing a large improvement on pedestrian detection and other object detectors as well. They proposed a new detector based on a novel feature descriptor[1] called Histograms of Oriented Gradients (HOG) which was able to achieve better results than the top performers at that time, such as detectors based on *wavelets* [Viola et al., 2003] and PCA-SIFT [Ke and Sukthankar, 2004]. The HOG is extracted from a detection window divided into cells of size 8×8. Each group of 2×2 cells is integrated into a block in a sliding fashion, making a dense grid of overlapping blocks [Zhu et al., 2006]. Next, the algorithm computes a histogram of oriented gradients for each cell, followed by a L2-norm with an optional clipping. Each block contains the concatenation of all its cells. We give details about this procedure on Figure 2.1. The HOG can be optimized by applying integral images [Viola and Jones, 2001], in which each histogram bin corresponds to an integral image. The HOG descriptor still is one of the most used and has inspired several others [Wang et al., 2009; Prisacariu and Reid, 2009; Dollár et al., 2009].

Based on integral images, Dollár et al. [2009] proposed linear and non-linear transformations to compute multiple registered image channels, called *Integral Channel Features*. Authors employed these descriptors into their CHNFTRS detec-

[1] In this work, we refer to both feature and descriptor interchangeably.

tor. Using 6 orientation bins, 1 gradient magnitude, and 3 LUV color channels are enough to reach state-of-the-art results. In Dollár et al. [2010], it is proposed a feature extraction that exploits the interpolation of features in different image scales, significantly reducing the cost and producing faster detectors when coupled with cascade classifiers. The *Integral Channel Feature* has demonstrated to be one of the fastest, yet simpler, feature descriptor, being used by several works. Our proposed approaches can also benefit from this feature descriptor since they are independent of the feature descriptors used.

Schwartz et al. [2009] noticed that using HOG by itself may lead to false positives due to the spatial distribution of edge orientations. Thus, objects with a similar spatial distribution to pedestrians, such as trees and light poles, may be misclassified as pedestrians. They observed that this problem can be avoided if one considers other important sources of information that inherently belong to pedestrians and do not belong to these false positives. The authors explored this insight proposing the use of information to complement the one extracted by HOG, such as clothing homogeneity and skin color. This information set increases the feature space, making the problem intractable by conventional machine learning techniques, such as Support Vector Machines (SVM). Hence, Schwartz et al. applied the supervised statistical approach called *Partial Least Squares* (PLS) to project the feature vectors onto a smaller subspace, allowing the use of SVM and quadratic classifiers. Along with the detector, the authors also proposed the use of Variable Importance on Projection, a feature selection tool based on PLS. The VIP provides a score for each feature, allowing to rank them according to their discriminative power in the PLS model [Schwartz et al., 2009]. This is important for our work because we employ it on the training phase of the cascade, allowing to considerably reduce the training computational cost and reduce a high number of detection windows on the early stages of the cascade.

Benenson et al. [2012b] proposed a monocular and, optionally, stereo detector with four incremental approaches. Their method is able to reach 50 FPS on a monocular system and 135 FPS on its stereo form. The authors use CHNFTRS [Dollár et al., 2010] as the baseline detector and their optimizations approaches are incorporated into it. The first optimization is a *single scale detection*, a slightly modified version of FPDW, proposed by Dollár et al. [2010], that allows object detection without image resizing and extraction of features for each image. The second approach is the use of a optimized version of the attentional cascade, known as *soft-cascade* [Bourdev and Brandt, 2005]. Besides, Benenson et al. [2012b] also employed a GPU-compatible implementation of CHNFTRS. In addition, as stereo information allows to reduce

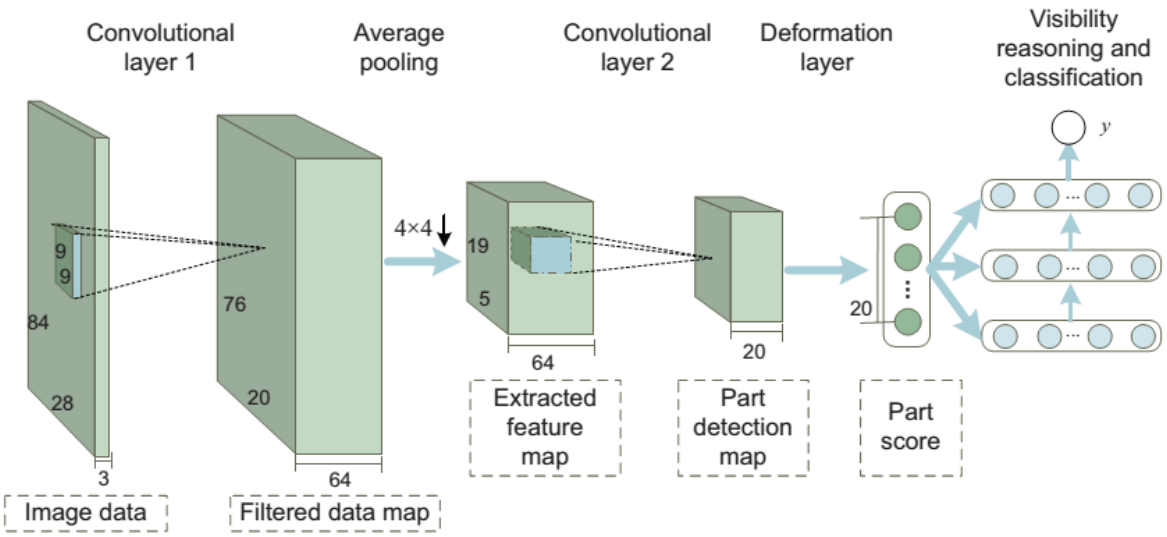


Figure 2.2. Example of a deep learning model. Source: Ouyang and Wang [2013].

the search area, the authors employ depth information extracted by an estimation of *stixels* [Benenson et al., 2011] and *ground-plane*.

Marin et al. [2013] proposed a combination of multiple local experts by means of a Random Forest ensemble. Each tree of the forest corresponds to a local expert, classifiers trained for feature vectors extracted from the same rectangular area across several windows. Their method works with rich block-based descriptors which are reused by the different experts of the ensemble. To obtain complementary and discriminant local experts, the authors employ a feature selection based on a random principle. They extract K rectangular areas with random values of width and height, and then the most discriminant ones are selected. They also show how to integrate the ensemble with a soft-cascade, in which the initial layer is compound of a number of trees and each following layer has an additional tree.

Recently, there is an increasing interest on deep learning methods applied on computer vision [Hinton et al., 2006; Hinton, 2007]. Several authors have proposed pedestrian detectors based on these methods [Zeng et al., 2013; Ouyang and Wang, 2012, 2013] achieving competitive results with the state-of-the-art. Figure 2.2 depicts an example of a convolutional neural network applied on pedestrian detection. However, the main focus of works on deep learning is not on their computational cost, since they require high computational power.

2.1.2 Part-Based Detectors

Besides holistic detectors, another approach to perform detection consists of modeling the pedestrians' constituents into a part-based fashion. In general, part-based detectors consists of a bottom-up approach that first detects the parts of a pedestrian until it gradually matches the whole pedestrian's body. They can be combined with a holistic detector, which is referred as *root* within this context [Felzenszwalb et al., 2010b]. The body parts might be either real, such as head, torso, arms, and legs, or only body-inspired, in which the detection window is virtually splitted into regions corresponding to body parts. In addition, they can be assumed at fixed locations or searched in a range of allowed locations. The latter method, known as *deformable*, allows the detection of unseen poses during the training phase and the removal of misclassified parts [Geronimo and López, 2013].

Part-based detectors show better performance on high-resolution images and are more suitable to handle conditions such as pose variation and partial occlusions [Dollar et al., 2012]. However, they are harder to train because they often make use of latent information, which is hard to retrieve because most datasets do not have body parts' annotations and it is daunting to make it manually [Felzenszwalb et al., 2010b]. Because of these weakly labeled data, deformable models are often outperformed by models such as holistic on difficult datasets, containing low quality images.

Among the classical part-based detectors, Weber et al. [2000] and Fergus et al. [2003] proposed constellation structures that restrict parts of the body to a sparse set of localizations, determined by a keypoint detector. The constellations capture the geometric arrangement by using a Gaussian distribution. In contrast, pictorial structure models [Felzenszwalb and Huttenlocher, 2005] define a matching problem whose body parts have an individual correspondence cost in a dense set of localizations. Amit and Trouvé [2007] employed a similar approach, but the authors consider explicitly as a appearance model with overlapping parts.

Deformable 2D models have difficulties to capture significant variations in appearance and form, such as the ones caused by extreme point of view changes. One possible solution employs aspect graphs [Plantinga and Dyer, 1986], allowing to capture significant point of view changes, in which mixture models reveal to be a simple solution for this problem. For example, it is common to use multiple models to encode frontal and lateral views of faces and cars [Schneiderman and Kanade, 2000]. Mixture models were also used to capture other aspects of appearance variation, such as when there are multiple natural subclasses in one object

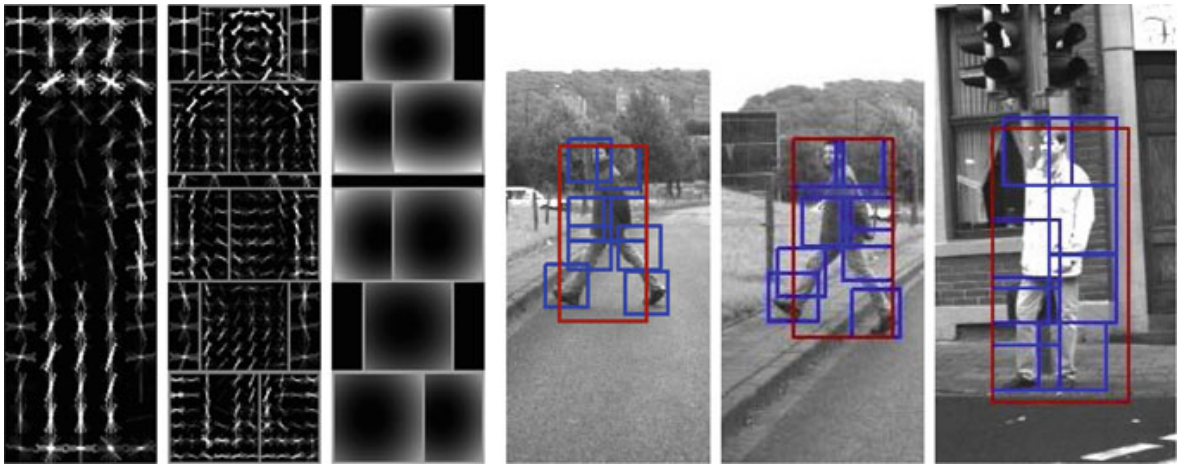


Figure 2.3. Part-based pedestrian model and detections. From left to right, HOG model of the root; HOG model of eight parts; spatial layout cost function of the parts (darker regions represents lower deformation cost); detections in Daimler A.G. dataset, shown boxes frame the detected root and parts. Source: [Geronimo and López, 2013], with kind permission of Springer Science+Business Media.

category [Schneiderman and Kanade, 2000].

Tosato et al. [2010] defined an hierarchy of overlapping parts with different size levels. The hierarchy is structured in such a way that the number of details is increased at each level. For example, the first level takes into account the full pedestrian's body. In the second level, the authors split the candidate window into three virtually body-inspired partitions, which are head-shoulder, torso and legs. The third level considers seven parts, the head, left and right shoulders, arms and legs. The authors combine the full hierarchy, composed of 11 parts altogether, by means of an ensemble using LogitBoost and using covariance as feature.

Felzenszwalb et al. [2010b] proposed a discriminatively trained, multi-scale, and deformable part model for object detection. The method reveals to obtain competitive results for pedestrian detection as well. Their method includes both a root model covering the entire pedestrian and part models in higher resolution. As explained earlier, the training phase is harder for part-based detectors due to the lack of annotated parts. In that work, the authors presented a new methodology for learning parts from weakly-labeled data based on generalization of SVMs, referred to *latent SVM*, which handles latent variables such as part positions. In addition, Felzenszwalb et al. [2010b] presented a new bootstrap method for data mining hard negative samples during training. The final detector, LATSVM, showed the highest detection rates among the part-based detectors and competitive scores regarding other holistic detectors [Dollar et al., 2012]. The final location of the

parts allows to compute a more precise location of the detected pedestrians, than when using only the root model. Moreover, an extension of this work is proposed by Park et al. [2010] to use multiresolution models, in which the method automatically switches to parts only at sufficiently high resolutions, since LATSVM scores better within this scenarios.

2.2 Computational Cost Issues

As illustrated in Figure 2.4, a general pedestrian detector is commonly composed of the following steps. The first step generates a set of detection windows that sample an input image. The second step extracts a set of feature descriptors from each detection window. Finally, in the third step, the feature vector of the detection windows are presented to a classifier, which will output high responses for those detection windows likely to contain pedestrians.

So far, we have reviewed several methods that addressed the pedestrian detection problem. Such methods still present a high computational cost related to two steps in common to every detector, namely, the feature extraction and the classification.

Feature extraction focuses on extracting relevant cues that better describe a given object. This step would not be necessary if we decided to use the image pixels by themselves. However, we cannot use pixel intensities solely since they are subject to noise, changes in illumination, curse of dimensionality, among others. Hence, we employ feature extractors to compute measurements that are meaningful and invariant to certain properties, such as localization and illumination [Trucco and Verri, 1998].

In the training phase, the classifier learns the pedestrian model from a set of feature descriptors extracted from several examples of pedestrians. In the testing phase, the algorithm repeats the procedure of extracting features from each detection window, and then presents them to the classifier, which provides a confidence score for this sample. As a detector must extract features for every test sample, which can easily be more than 60,000 for a single 640x480 image, feature extraction is responsible for a considerably amount of the computational cost. Therefore, we are interested in extracting meaningful features for pedestrians, constrained by their low computational cost.

The *classifier* employed by the detector also has a strong influence on the computational cost since it is also executed for each sample. Classifiers are built by

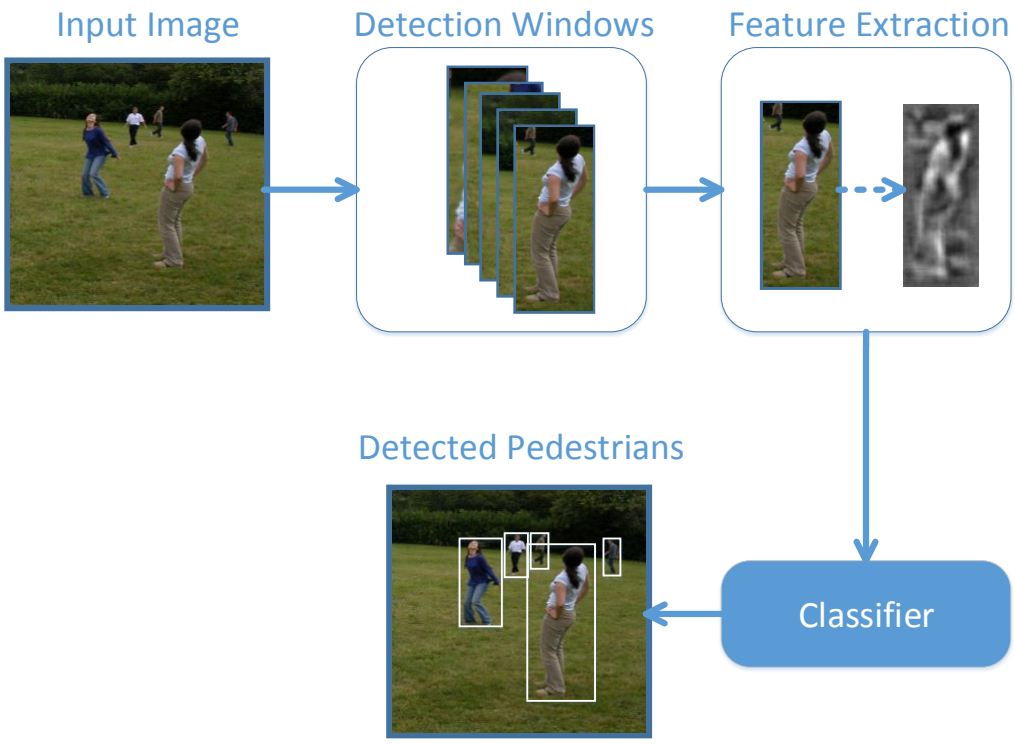


Figure 2.4. Scheme of a general pedestrian detector. We propose optimizations related to both steps of detection window generation and classification.

taking a set of labelled examples and using them to come up with a rule that will assign a label to any new example. In the general problem, we have a training dataset $(\mathbf{x}_i, y_i)$; each of the $\mathbf{x}_i$ consist of the feature vector for the i-th sample, and the y_i is a label giving the type of the object that generated the example [Forsyth and Ponce, 2011]. We need a classifier that best separates data into either pedestrian or background classes. While using more complex functions might lead to better results, they also increase the computational cost. Thus, we must deal with the trade-off of the computational cost versus the detection rate. Moreover, if the detector must deal with high dimensional feature spaces, it might be necessary to reduce the number of dimensions. One may apply dimensionality reduction tools, such as Principal Component Analysis (PCA) and PLS, before presenting the sample to the classifier, which influences the computational cost, as well.

2.3 Optimization Approaches

There are several optimization approaches to reduce the computational cost of pedestrian detectors. On the one hand, the approaches might focus on reducing

the amount of data processed, which allows the extraction of fewer features and fewer samples, consequently reducing the computational cost. On the other hand, the optimization approaches resort to specialized hardware to speed up the computation.

In this section, we explore the main optimization approaches for pedestrian detection, grouping them in three main categories: *cascades of rejection* (Section 2.3.1), *parallelization and GPUs* (Section 2.3.2), and *filtering* (Section 2.3.3) [Benenson et al., 2012b]. In addition to these categories, *prior knowledge* also deserves mentioning. It takes advantage from information of the environment and the camera setup, such as ground plane estimation and fixed cameras, allowing to restrict the search area instead of scanning the full image. Since such approaches depend on the inherent characteristics of the data set, they are complementary to our work. Hence, we do not extend on this topic.

2.3.1 Cascade of Rejection

Rejection cascades are a widely employed approach to reduce the computational cost in object detection. They are composed of multiple stages, each one composed of a classifier or an ensemble of them. The most common cascade employs AdaBoost to build an ensemble of weak classifiers in each stage, as illustrated by Figure 2.5. They are trained by adding features until the target detection and false positives rates are met. A cascade of rejection may also be referenced by other names, such as cascade of classifiers [Bourdev and Brandt, 2005], since it has multiple succeeding classifiers; or even attentional cascade [Viola and Jones, 2001], as it focus attention on samples harder to classify.

The main idea behind this approach is to use simple classifiers to discard detection windows that are easy to classify, while the remaining windows advance through the cascade, where more complex classifiers are used. The key insight behind the usage of cascades is that most of the regions of natural images belong to the background and present simple characteristics, such as walls, sky, or roads, while few regions belong to pedestrians, which results in an unbalanced distribution with much more counter-examples than pedestrians. Therefore, the early stages of the cascade focus on those simple background regions, which can be discarded using simple classifiers with few feature descriptors. Then, a small number of regions is left for later stages, which are composed of more complex classifiers, with more feature descriptors. This process leads to a significant reduction in computational cost.

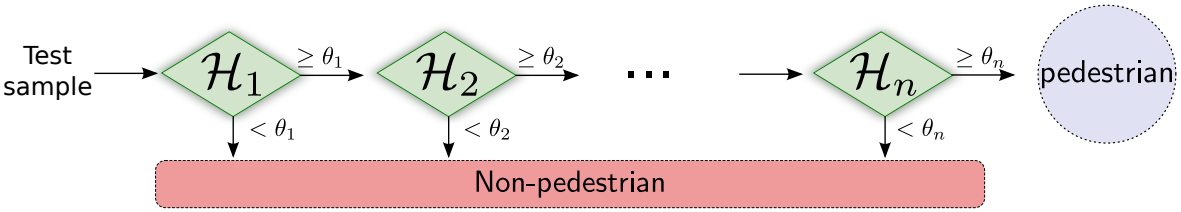


Figure 2.5. Example of a cascade of rejection. The i-th stage is composed of an ensemble of weak classifiers, creating a strong classifier $\mathcal{H}_i$. Along with the ensemble, a stage has a threshold θ_i to reject or propagate a window.

The cascade framework was initially employed in face detection by Viola and Jones [2001]. In their seminal work, they proposed a face detector using this technique by successively combining classifiers with increasing complexity by means of *AdaBoost* [Freund and Schapire, 1995] to build the cascade stages to allow rejection of a large amount of windows in early stages.

In the pedestrian detection domain, the rejection cascade was firstly used by Zhu et al. [2006] with the extraction of HOG features [Dalal and Triggs, 2005], resulting in detection rates comparable to the state-of-the-art of the literature at that moment with substantial speed improvement. However, the training time considerably increased since AdaBoost needs to select the most discriminative features to compose each of its weak classifiers.

Tuzel et al. [2007] improved the results obtained by Dalal and Triggs [2005] applying low-level features including intensity, gradient, and spatial location along with a covariance matrix. As covariance matrices do not lie in a vector space, the authors applied LogitBoost classifiers combined with a rejection cascade that contains points lying on a Riemannian manifold. As an extension to Tuzel et al. [2007], Paisitkriangkrai et al. [2008] proposed a classification cascade that evaluates weak classifiers in a Euclidean space, instead of a Riemannian manifold, which results in a faster method.

A cascade of classifiers was also employed to build deformable part models. Felzenszwalb et al. [2010a] proposed a cascade of classifiers in which partial hypotheses are eliminated by a sequence of thresholds determined by a set of positive examples, which theoretically guarantee the performance of this cascade method. The detection algorithm of the cascade for general classification models is formally defined by one grammar.

For training object detectors, an easier and faster training cascade variation, called *soft-cascade*, which uses fewer features, was described by Bourdev and Brandt [2005]. Zhang and Viola [2007] proposed the multiple-instance pruning (MIP) algo-

rithm for soft-cascades. It computes thresholds to terminate the computation with no reduction in detection rate or increase in false positive rate on the training set.

Dollár et al. [2012] reduced the computational cost of their previous method [Dollár et al., 2010], reaching a speed detection of 35–65 FPS using the *crosstalk cascades*. In this cascade variation, the authors explore the correlations among the adjacent detection windows, introducing two opposite mechanism: detector excitation of promising neighbors and inhibition of inferior neighbors. Due to this communication between detectors, this method reaches a 4–30x speedup.

In general, we train one cascade of rejection for the domain of interest, e.g pedestrians or faces. However, the cascade may fail when presented to a different domain from the one learned. For instance, consider a cascade of classifiers trained for a domain of adult faces. This cascade might be unable to generalize the classification for baby faces. A common solution consists in training one cascade for each domain, i.e one cascade for adult faces and a second cascade for baby faces. Nevertheless, this is not scalable for a large number of data domains because it requires huge data annotation and computational effort. With this in mind, Jain and Farfade [2013] proposed a cascade variation that easily allows the adaptation of a pre-trained cascade to a new similar domain, instead of training one for each domain from scratch. The solution requires a small number of labeled positive samples from a different yet similar data domain. The results are better than the baseline cascade and the one trained from scratch using the given training examples.

Different from the previous approaches, our detector, the PLS Cascade (described in Section 3.3), proposes a cascade of classifiers using a combination of PLS and variable selection approach VIP aiming at reducing the number of projections required by the PLS detector [Schwartz et al., 2009]. In addition, different from approaches such as Viola and Jones [2001] and Zhu et al. [2006], the proposed cascade propagates information (without increasing the computational cost) to later stages to increase the discriminability of the classifiers instead of maintaining all feature descriptors as candidates during all stages. Our resulting approach is faster to train than the conventional cascades; the usage of the VIP allows to reject more samples in the earlier stages, and the computational cost of the PLS Detector is considerably reduced.

2.3.2 Parallelization and GPUs

Graphics Processing Units (GPUs) are highly parallelizable architectures, with many cores and large amounts of memory in a single unit. Applying parallelization tech-

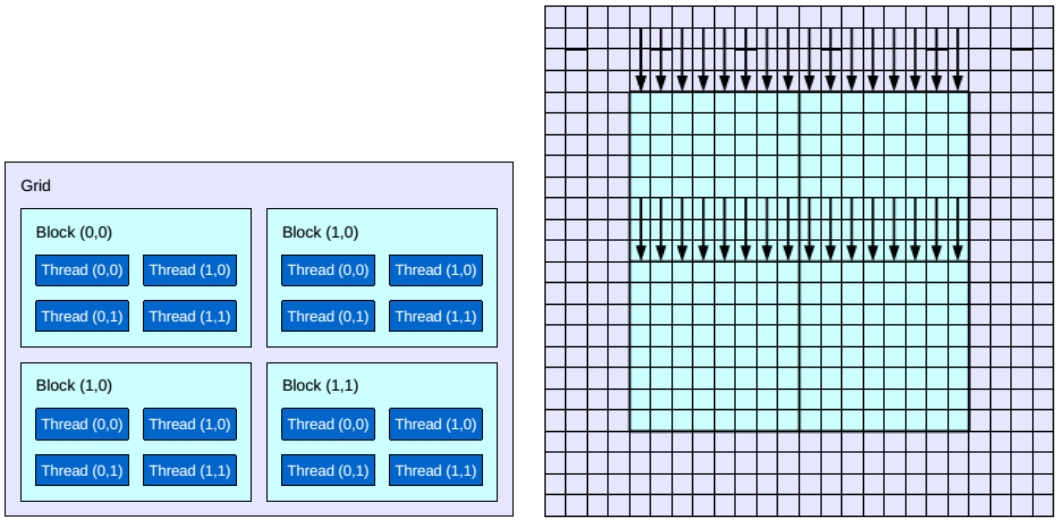


Figure 2.6. Example of a GPU's architecture and feature extraction. Left image illustrates the thread architecture, while the right one exemplifies HOG extraction using GPU. Each vertical arrow represents one thread and the direction they follow to process the pixels in the cell. Source: Prisacariu and Reid [2009].

niques allows to significantly increase the performance. Although Moore's Law predicts that processing capacity doubles every two years, the processing speed for a single CPU is apparently stuck for a few years, making it unpractical to depend only on such architecture. However, Moore's Law is still valid for highly parallelizable architectures, such as CPUs with multiple cores and GPUs [Leibe et al., 2008]. Thus, new methods for object detection must explore the capabilities of this growing architecture.

There are many methods for object detection that explore these characteristics and are able to achieve higher processing speed in conventional methods. Particularly in pedestrian and face detection some methods use GPU to extract descriptors, a significantly cost of a detection method in terms of computational time. Masaki et al. [2010]; Wojek et al. [2008]; Zhang and Nevatia [2008]; Prisacariu and Reid [2009] showed in their works efficient ways to extract descriptors using GPU such as HOG, as illustrated in Figure 2.6. Benenson et al. [2012b] showed that it is possible to achieve a high speedup in detection, and in some cases, better accuracy compared to other state-of-the-art methods. The authors adapted the CHNFTRS detector, proposed method by Dollár et al. [2009], to a parallel architecture and also extracted descriptors based on GPU. The authors claim to have achieved a seven times higher speed than the original detector, just by changing it to GPU processing. Oro et al. [2011] presented a real-time face detector based on Haar that uses the

GPU to calculate integral images and filter evaluation, reaching 35 frames per second in resolutions up to 1960×1080 and using the sliding window approach with shifting of one pixel.

Although parallelization and GPU algorithms are not addressed by this work, our proposed approaches may benefit from them since they are complementary. For example, we could employ feature extraction using GPU, which would allow a significantly speedup to our method.

2.3.3 Region of Interest Filtering

Filtering regions of images consists in removing elements that do not belong to the targeted object, reducing the region of search and keeping only potential objects of interest, as illustrated in Figure 2.7. With smaller searchable regions, a robust classifier is applied onto a smaller number of windows. Therefore, instead of using this classifier in the entire image, it is applied only in small regions, reducing the computational cost of detection. Several methods address the aforementioned problem induced by dense search in sliding window approaches. Some proposed heuristics evaluate windows in fixed sizes, which are subsampled for certain strides [Dalal and Triggs, 2005; Ferrari et al., 2008]. Depending on the stride, one might obtain a more sparse or dense sampling of the image. For example, larger strides yields more sparse sampling of the image.

Focusing on searching only promising regions of the image, Lampert et al. [2008] proposed a method to perform object localization relying on a branch-and-bound approach that finds the global optimum of a quality function over every possible subimage. It returns the same object locations that an exhaustive sliding window approach would, but requiring fewer classifier evaluations than there are candidate regions in the image, typically running in linear time or faster in function of the number of images. Related to this work, Lampert [2010] described a divide and conquer method to accelerate the evaluation of classification cascades for object detection. A set of candidate regions, in contrast with individuals regions, permit a large number of potential locations to be discarded, reducing the computational cost compared to other cascade strategies for object detection.

Saliency detectors are able to detect regions of interest by simulating the behavior of the human visual system. In the first phase, called pre-attentive visual search, they quickly detect the possible positions of proto-objects in the image. The obtained saliency map suggests the position of the proto-objects. Feng et al. [2011] proposed a filtering method that finds the salience of each window and segments

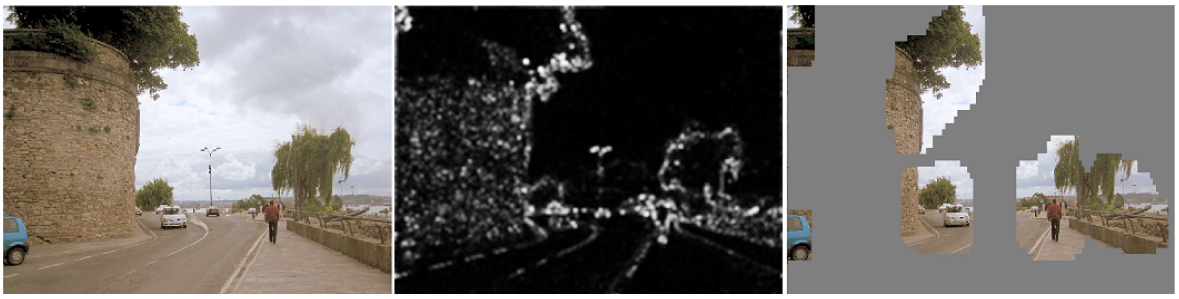


Figure 2.7. An example of saliency detector. From left to right, the original image; the saliency map; and candidate regions in the saliency map. Source: Silva Filho et al. [2012].

the image into regions based on their similarities. To find the most probable window to contain a salient object, it is employed the difference among regions given their LAB color histograms and spatial distances.

To detect objects in different sizes, Itti et al. [1998]; Harel et al. [2007] proposed the direct analysis of features extracted in multiple scales of the image. Based on saliency detectors, Silva Filho et al. [2012] proposed a method based on multi-scale Spectral Residual Analysis (MSR), in which an image is resized several times by a factor to cover different scales. In each resizing, a saliency map is created and a sliding window approach is applied, then a quality function is computed in each map in order to discard regions. Figure 2.7 illustrates the application of MSR on an image. In comparison with a regular sliding window approach, the MSR method was able to reduce in 75% the number of windows to be evaluated by an object detector and improving the detection rate in most cases.

Recently, Cheng et al. [2013] applied a soft image abstraction representation to split an image into large scale and homogeneous elements for salient region detection. The authors considered both appearance similarity and spatial distribution of image pixels, abstracting unnecessary image details and allowing to compare saliency values across similar regions. Such approach produces perceptually accurate salient detection. Margolin et al. [2013] proposed a novel method that combines patterns, colors, and high-level cues and priors. The experiments show that the method outperforms most state-of-the-art methods on five data sets.

The full extent of these approaches contains several solutions to reduce the amount of data to be processed. These approaches are based on branch-and-bound techniques, saliency detectors, among others. Although most of the techniques allows to reduce the amount of data and may be used as a preliminary step to the classifier, they still might present unnecessary evaluations. In contrast, our random

filtering approach, described in Section 3.2, aims at rejecting detection windows by evaluating only a few of them; consequently, a large amount of windows are preemptively discarded without cost. Later, we correct the misplaced windows using location regression, which has a low computational cost since it requires the extraction of simple and sparse features. Therefore, our proposed filtering method is able to achieve a considerably speedup.

Chapter 3

Methodology

This chapter describes our proposed methodology, composed of two novel optimization approaches for reducing the computational cost of pedestrian detection, namely, the *random filtering* and the *PLS Cascade*. These optimization approaches focus on the generation of the detection windows and on the classifier.

As observed in the preceding chapter, filtering approaches such as saliency detectors require to extract and evaluate features for all detection windows, at least once, while most cascades of rejection do not take advantage of feature selection for fast training and increased rejection of detection windows in earlier stages. In this work, one of our focus is to avoid the feature extraction for most of the detection windows.

As shown in Figure 3.1, our proposed optimization methodology consists of the following steps. Given an input image, in the first step we apply the traditional sliding window algorithm, which scans the input image with a window of fixed size in a range of scales, generating a set of detection windows (described in Section 3.1). Such detection windows are presented to the random filtering which selects a random set of detection windows and adjusts them properly using a location regression (described in Section 3.2). Later, the filtered and adjusted set of detection windows is presented to the last step of our methodology, the PLS Cascade, to reject detection windows that are easily classified as background, while windows that are harder to predict advances through the stages of the cascade (described in Section 3.3).

It is worth noting that the proposed optimization approaches are mutually independent, such that the random filtering is optional for the execution of the PLS Cascade, and the converse is also true. Therefore, we may use random filtering with any other detector based on sliding window, and the PLS Cascade might be employed stand-alone. In this work, we focus on the PLS Detector [Schwartz et al.,

2009] since it is widely used in the literature and achieves high detection rates on several pedestrian detection data sets.

The remainder of this chapter describes each step of our methodology, starting with a brief review of the generation of the detection windows based on the sliding window algorithm [Forsyth and Ponce, 2011].

3.1 Sliding Window Algorithm

Object detection is closely related to the task of object classification. Given a test sample, the classification task focuses on assigning a class to an object, while the detection task aims at localizing the objects in an image. In other words, the main difference between the two is that object detection requires to output the tuple (x_0, y_0, w, h, r), in which x_0 and y_0 are the left-upper coordinate of the object; w and h are its respective width and height; and r its rotation. In contrast, object classification just requires to assign a label for a test sample. Hence, the question is how to tackle the object detection problem?

A widely employed approach to solve the object detection problem is reducing it to a classification problem. Such approach is called sliding window [Forsyth and Ponce, 2011], and it works by exhaustively scanning an input image to generate a set of coordinates of several detection windows in multiple scales. In this work, we define a detection window as the following:

Definition 3.1 (Detection Window). *A detection window consists of a candidate window to contain a pedestrian, defined by a tuple (x_0, y_0, w, h), in which x_0 and y_0 are the left-upper coordinate of the window, and w and h its respective width and height.*

One may notice that this definition does not include the rotation r. It is not required because the rotation is assumed to be always the same, as pedestrian detection is restricted to pedestrians in more or less upright poses.

After generating this set of windows, the detection is handled as a classification problem, i.e., the detection windows are presented to a classifier that predicts whether a candidate window belongs to either the pedestrian or the background class. For each scale, the algorithm shifts the detection window s_x and s_y pixels, in the horizontal and vertical axis, respectively. The range of scales starts from a minimum to a maximum value, aiming at covering pedestrians of all sizes in the image, and each image is rescaled by a scaling factor α. The values of s_x and s_y are usually estimated as a percentage of the detection window's width w and height h scaled

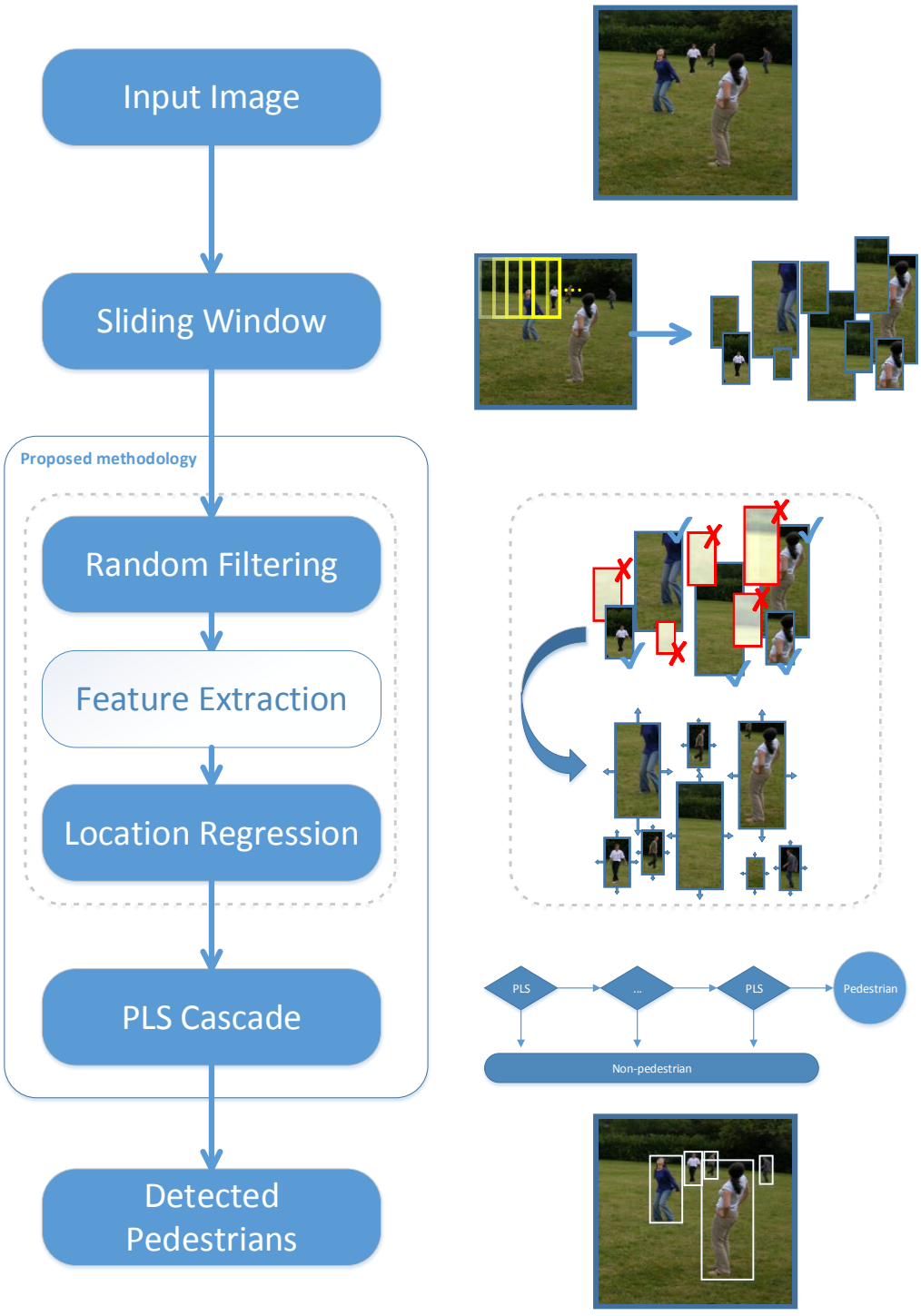


Figure 3.1. Fluxogram describing the general steps of the optimization methodology proposed in this work. The balloon represents our proposed approach, divided into two smaller steps, namely, random filtering combined with location regression (Section 3.2), and the PLS Cascade (Section 3.3).

by α. Figure 3.2 illustrates the algorithm. For instance, a 640×480 image, scanned across 10 scales, easily generates more than 60,000 detection windows.

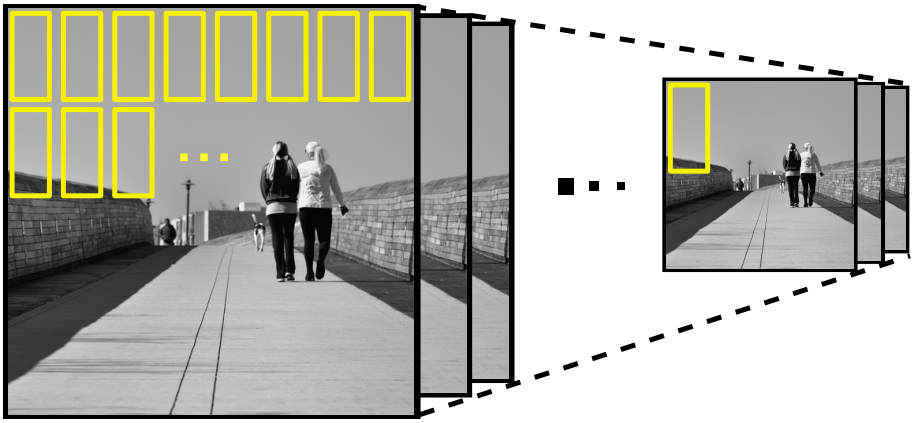


Figure 3.2. Sliding window algorithm. The input image is scanned in all possible locations and in multiple scales by a detection window, whose size is kept fixed in our work. This example illustrates non-overlapping windows, but real systems sets the strides such that the windows overlap with each other, avoiding to skip a pedestrian.

3.2 Random Filtering and Location Regression

The sliding window algorithm generates the detection windows in a wide range of scales and strides, yielding a set of overlapping windows with high redundancy, which highlights the need for a filtering approach. To reduce the amount of data processed by the pedestrian detector, we propose a method based on a random filtering followed by adjustments on the detection window locations. Here, we randomly select a fraction of windows that will be presented to a classifier (details in Section 3.2.1). However, the selected windows might be slightly displaced from the pedestrian's location, which may result in lower responses by the classifier. Therefore, before presenting them to the classifier, a regression is employed to adjust the window location to increase their responses (Section 3.2.2). An overview of the steps of this approach is depicted in Figure 3.3.

3.2.1 Random Filtering

As mentioned earlier, the first step of any pedestrian detector consists of applying the sliding window algorithm in a given number of scales and strides. However, if a large number of scales and small strides are used to move the detection win-

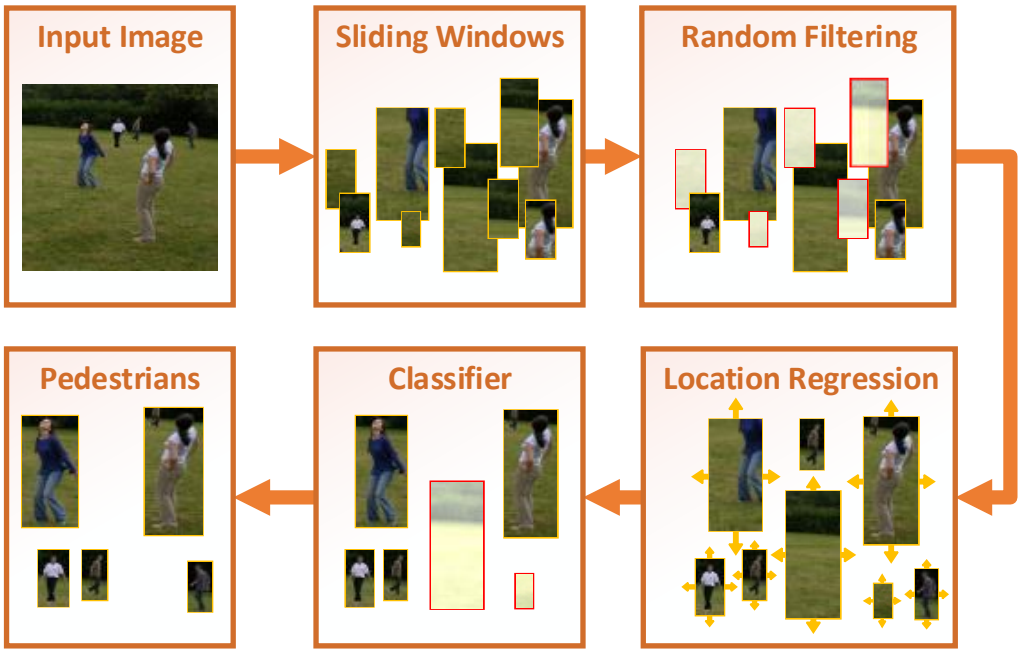


Figure 3.3. Overall layout of the proposed approach based on random filtering and location regression.

dow, we might have a large amount of windows to be classified, demanding high computational power. On the other hand, if few scales with large strides are considered, pedestrians might be skipped. Therefore, the former option is more suitable for achieving accurate results.

After generating an initial set of detection windows, instead of presenting all windows to a classifier, our approach performs a random selection of the windows, i.e., a percentage of the total number of detection windows is selected. To ensure that every pedestrian is still detected, we use the Maximum Search Problem theorem [Schölkopf and Smola, 2002, pp. 180]. The problem of classifying windows as containing pedestrians or not may be seen as the task of finding a subset of windows containing pedestrians from a finite set of windows. As most maximum search problems, the exact solution is computationally expensive (every sample has to be evaluated). Instead, it is possible to find almost optimal approximate solutions by using probabilistic methods as the one described as follows.

The problem at hand might be formulated as follows. Given a set of m windows, where $M = \{f_1, \ldots, f_m\}$ and $\mathcal{Q}[f]$ is a criterion to evaluate whether a detection window is covering image region with a pedestrian, i.e., the classifier response. Then, the problem can be stated as finding a window $\hat{f}_i$ that maximizes $\mathcal{Q}[f]$. In the pedestrian detection context, one is interested in finding not only the window $\hat{f}_i$

that maximizes $\mathcal{Q}[f]$, but also a subset of windows with large $\mathcal{Q}[\hat{f}_i]$, since more than one pedestrian might be in the image.

To solve the aforementioned problem, all terms $\mathcal{Q}[f_i]$ must be computed, which demands m detection window evaluations. Due to the multiple scales and strides considered to locate all pedestrians in an image, the number of extracted windows is large for a given image, rendering this operation too expensive. For instance, for an image with 640×480 pixels, there are approximately 60,000 detection windows that need to be evaluated to detect pedestrian in multiple scales. Therefore, it is imperative to find a cheaper approximate solution.

Schölkopf and Smola [2002] demonstrated that by selecting a random subset $\tilde{M} \subset M$ sufficiently large, one can take the maximum over $\tilde{M}$ as an approximation of the maximum over M. If a small fraction of $\mathcal{Q}[f_i]$ ($i = 1, 2, \ldots, m$), whose values are significantly smaller or larger than the average do not exist, one can obtain a solution that is close to the optimum with high probability.

To compute the required size, $\tilde{m} = |\tilde{M}|$ ($\tilde{M} \subset M$), of a random subset to achieve a desired degree of approximation, Schölkopf and Smola [2002] showed that one can use the following equation

$$\tilde{m} = \frac{\log(1-\eta)}{\ln(n/m)} \tag{3.1}$$

where η is the desired confidence and n denotes the number of elements in M having $\mathcal{Q}[f_i]$ smaller than the maximum of $\mathcal{Q}[f_i]$ among the elements in $\tilde{M}$.

Equation 3.1 states that at least one element $f_i \in \tilde{M}$ will have $\mathcal{Q}[f_i]$ higher than the $\mathcal{Q}[f_i]$ of m elements in the original set M with a confidence of η. Therefore, this result would not be helpful when the element $f_i \in M$ with only the maximum $\mathcal{Q}[f_i]$ is required ($\tilde{m}$ would be very close to m). However, in the pedestrian detection problem based on sliding windows, we can take advantage of the fact that one pedestrian is covered by more than one detection window leading to a correct detection. This behavior is due to the redundancy resulting from the small strides in s_x and s_y and multiple scales (Figure 3.4 illustrates that, even when very few windows are randomly selected, there are a number of windows covering the pedestrian location). Therefore, n in Equation 3.1 can be fairly large, which reduces $\tilde{m}$ significantly.

To illustrate the usage of the above result, we performed the following experiment. Given an image with $m = 60{,}000$ detection windows uniformly sampled from an 640×480 image pixels at multiple scales, we found that 583 windows contain the correct location of a pedestrian (windows that should lead to high values of $\mathcal{Q}[f_i]$,

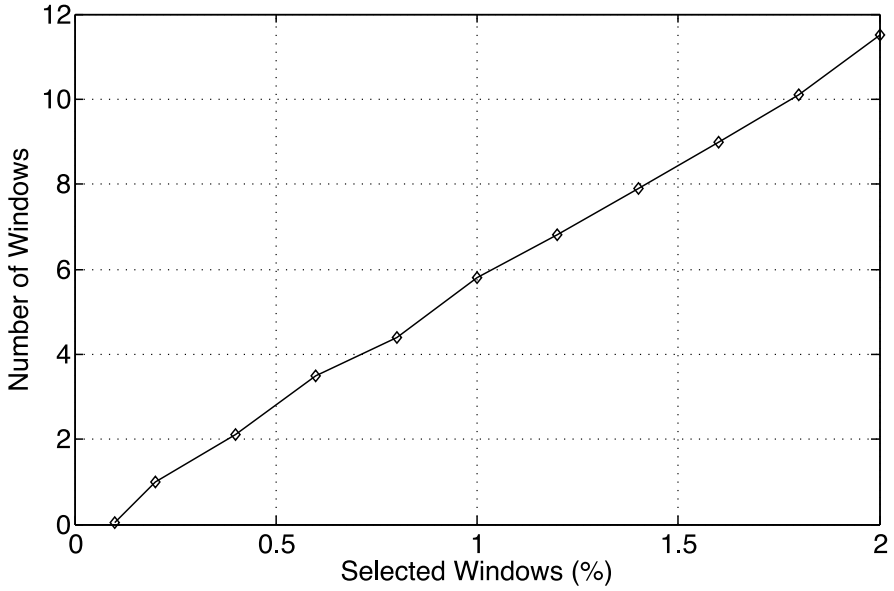


Figure 3.4. Mean number of detection windows covering a pedestrian in the INRIA dataset as a function of the percentage of randomly selected windows. As standard approach to determine whether a pedestrian is covered by a detection window [Dollar et al., 2012], we consider that a window A covers a ground-truth window B when their intersection divided by their union (Jaccard coefficient) is greater than 0.5.

depending on the classifier being considered). Therefore, according to Equation 3.1, a random sampling with $\hat{m} = 133$ (0.22% of the total windows) should contain at least one pedestrian with a probability of 95%, which is compatible with the plot shown in Figure 3.4.

Although the random filtering can provide a small subset of detection windows, such that almost every person in the image is covered, these windows might not provide the exactly location of a pedestrian. Hence, this pedestrian might be missed due to the low response achieved by the classifier. Therefore, we employ an extra step before presenting the window to the classifier to adjust the window location to the pedestrian, as will be described in the next section.

3.2.2 Location Regression

Aiming at adjusting the bounding box delimited by a detection window, we learn a regression model (referred to as *location regression*) to correct it to the pedestrian's location. In this problem, we want to find displacements Δx and Δy such that, when added to the centroid (G_x, G_y) of a given window, they move the detection window to the correct position of a pedestrian. Other variables may also be considered, such

as the tuple $(\Delta w, \Delta h)$, in which Δw and Δh denotes width and height of a detection window, respectively. Nonetheless, in this work we consider only the $(\Delta x, \Delta y)$ tuple to demonstrate the detection improvement.

Previously, location regression has been applied on detection, but with a different purpose. Schwartz et al. [2013a] noticed that multiple redundant windows could be over a pedestrian, which should be presented to a generic classifier at the end of a detector stage. Hence, they applied location regression to reduce even further the number of detection windows that would be considered by the classifier by finding the corresponding pedestrian to each detection window. In this work, on the other hand, we only have few windows that were selected by random filtering and we want to predict their correct location. In other words, while the former wants to reduce redundant windows, the latter aims at finding the "best" location for the windows.

Unlike Schwartz et al. [2013a], our proposed method learns the regression model during an offline phase. First, we need to generate a training set to be presented to the learning algorithm. Given a training sample, we generate a set of displaced windows with the respective differences $(\Delta x, \Delta y)$ to their correct position. This set of displaced windows is generated in all directions, as long as the Jaccard coefficient between the ground-truth bounding box and the displaced window is greater than 50% [Dollar et al., 2012]. This ensures that we have a portion of the pedestrian within the window. The Jaccard coefficient $J(d_1, d_2)$ between two windows d_1 and d_2 is defined as

$$J(d_1, d_2) = \frac{|d_1 \cap d_2|}{|d_1 \cup d_2|}. \tag{3.2}$$

For instance, consider the sample shown in Figure 3.5. The blue bounding boxes represent the ground-truth annotation, while the green bounding boxes represent the generated samples. A bounding box that correctly matches a pedestrian, i.e., it positioned exactly over a pedestrian, $(\Delta x, \Delta y)$ equal to $(0, 0)$. Another example is a bounding box displaced one pixel to the left, such as the first one, must add $(1, 0)$ to match the ground-truth centroid. Finally, a bounding box displaced one pixel above and to the right, such as the second one, must add $(-1, -1)$ to correct its location.

Another possibility is to model negative samples along with the positive ones. In this regression model, we may represent their displacement correction as infinity, i.e., (∞, ∞), since they cannot be adjusted to any pedestrians' location. This would allow us to build a decision surface that could improve the accuracy of location re-

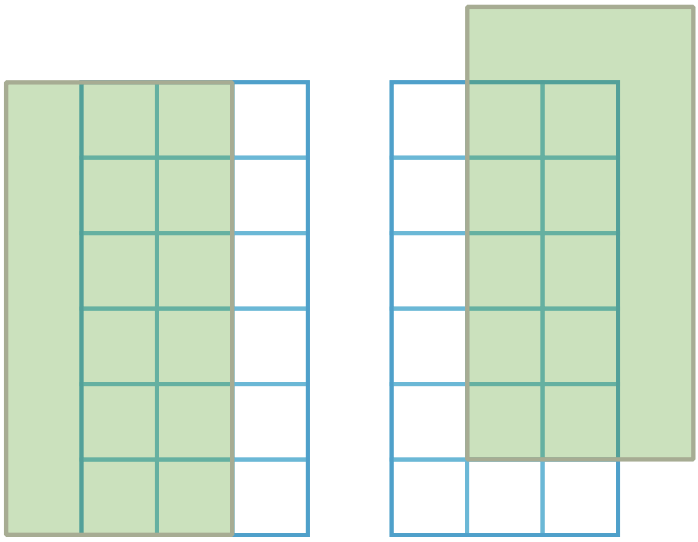


Figure 3.5. Examples of sample generation used in the learning phase of the location regression.

gression and possibly reject negative windows in advance, as they would be shifted to outside of the image's boundaries. This modeling was not considered in this work and it will be addressed as future work.

Once the training set is created, features descriptors are extracted from the windows and associated to the displacements. Ideally, such descriptors should be simple enough to preserve a low computational cost. Then, a regression with two dependent variables, Δx and Δy, is learned. Even though we have employed a regression based on Partial Least Squares due to its numerical stability and robustness to multicollinearity [Rosipal and Kramer, 2006], other methods could have been applied.

During the testing phase, the location regression corrects the detection windows' location before presenting them to the classifier, as illustrated in Figure 3.6.

3.3 Partial Least Squares Cascade

Although PLS allows accurate detection in high-dimensional feature sets, the method presents a high computational cost [Dollar et al., 2012; Schwartz et al., 2013b]. To reduce this cost, we propose the application of Partial Least Squares method in the context of a cascade framework, referred to as *PLS Cascade*.

In the proposed cascade, the feature descriptors are ranked by Variable Importance on Projection (VIP) so that more discriminative descriptors are used first in the cascade aiming at the rejection of a large number of samples in early stages. We

Figure 3.6. Real example of performing location regression to adjust the detection window location.

also propose to propagate the latent variables from one stage to the next such that discriminative information is also available in later stages without the need for reconsideration of feature descriptors that were already used in previous stages. The training and testing phases are illustrated in Figure 3.7 and Figure 3.8, respectively, and are described in Section 3.3.2, after a overview on the Partial Least Squares and its derived feature selection method, VIP (Section 3.3.1).

3.3.1 Partial Least Squares Analysis

Designed to model relations between observed variables, PLS constructs a set of predictor variables (latent variables) as a linear combination of the original predictors, represented in a matrix $\mathbf{X}$ (feature matrix), containing one sample per row [Wold, 1985]. The responses associated with the samples are stored in a vector $\mathbf{y}$, which are the class labels in the pedestrian detection problem.

Given an m-dimensional feature space and a scalar denoting the class label, a set with N samples is represented by the feature matrix $\mathbf{X}_{N\times m}$ and by the vector $\mathbf{y}_{N\times 1}$. PLS decomposes $\mathbf{X}$ and $\mathbf{y}$ as

$$\mathbf{X} = \mathbf{T}\mathbf{P}^T + \mathbf{E}, \quad \mathbf{y} = \mathbf{U}\mathbf{q}^T + \mathbf{f} \tag{3.3}$$

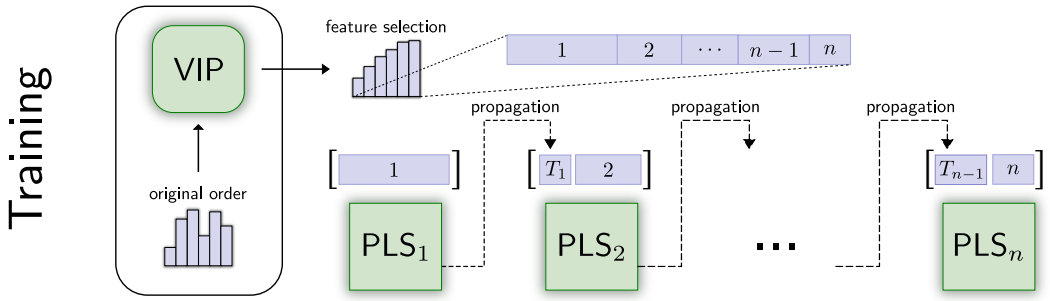


Figure 3.7. Overall layout of the proposed *PLS cascade* using Partial Least Squares with latent variable propagation and Variable Importance on Projection (VIP) for feature ranking. Initially, the descriptors are extracted from the image and sorted using VIP, which ranks variables by their discriminative power. According to their rankings, the variables are set to stages, which allows to increase the number of discarded samples in the early stages. Each stage adds features until it reaches a desired false positive and miss rates. Hence, a PLS model is created using these features to classify the samples presented to this stage. Since features that have already been considered are not used in the later stages, the low-dimensional feature set (latent variables) are propagated to avoid using only features with less discriminative power.

where $\mathbf{T}_{N\times p}$ and $\mathbf{U}_{N\times p}$ stand for latent matrices containing p extracted latent vectors, the matrix $\mathbf{P}_{m\times p}$ and the vector $\mathbf{q}_{1\times p}$ represent the loadings, and $\mathbf{E}_{N\times m}$ and $\mathbf{f}_{N\times 1}$ store the residuals from the decomposition. The PLS method employs the Nonlinear Iterative Partial Least Squares (NIPALS) algorithm to estimate a set of projection vectors $\mathbf{W} = \{\mathbf{w}_1, \mathbf{w}_2..., \mathbf{w}_p\}$, referred to as projection vectors. Each of these vectors is estimated to maximize the covariance between the predictor and the response variables [Rosipal and Kramer, 2006], such as

$$|cov(\mathbf{t}_i, \mathbf{u}_i)|^2 = \max_{|w_i|=1} |cov(\mathbf{X}\mathbf{w}_i, \mathbf{y})|^2 \tag{3.4}$$

where $\mathbf{t}_i$ is the i-th column of matrix $\mathbf{T}$, $\mathbf{u}_i$ the i-th column of matrix $\mathbf{U}$, and $cov(\mathbf{t}_i, \mathbf{u}_i)$ is the sample covariance between latent vectors $\mathbf{t}_i$ and $\mathbf{u}_i$. This process is repeated until the desired number of latent vectors had been extracted. Therefore, while performing the dimension reduction, the PLS focuses on the discrimination among the classes, providing a low dimensional feature space suitable for classification and regression.

To perform the dimensionality reduction, the feature vector $\mathbf{x}_i$ is projected onto the projection vectors $\mathbf{W} = \{\mathbf{w}_1, ..., \mathbf{w}_p\}$, which gives as result the latent vector $\mathbf{t}_i$

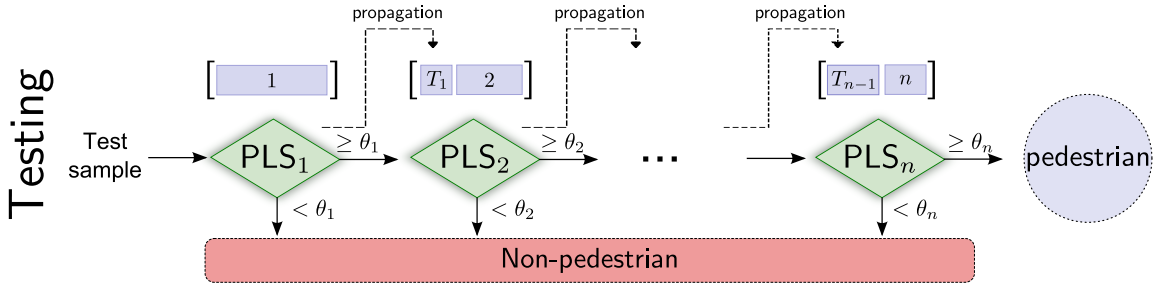


Figure 3.8. Testing phase. The detection window is evaluated through each stage of the cascade. If the score of a detection window is higher than the rejection threshold θ_i, it triggers the classifier of the next stage; otherwise, it is rejected. This procedure repeats until the test sample reaches the last stage. When computing the regression, PLS gerenates a set of latent variables T_i, which are propagated to the subsequent stages.

$(1 \times p)$, as

$$\mathbf{t}_i = \mathbf{x}_i \mathbf{W} \tag{3.5}$$

Finally, the matrix $\mathbf{B}$ of regression coefficients for the model $\mathbf{y} = \mathbf{U}\mathbf{q}^T + \mathbf{f}$ is given as

$$\mathbf{B} = \mathbf{W}\mathbf{q}^T = \mathbf{W}(\mathbf{T}^T\mathbf{T})^{-1}\mathbf{T}^T\mathbf{y} \tag{3.6}$$

where $\mathbf{B} = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_p\}$.

In this work, we use the PLS approach to both classify a sample considering the PLS regression [Schwartz, 2012b] and reduce the dimensionality of the data so that we can use the latent variables in the remaining stages of the rejection cascade to improve their discriminative power (described in Section 3.3.2).

Derived from PLS, the VIP provides a score for each variable on the original feature space (matrix X), so that it is possible to rank the variables according to their predictive power in the PLS model [Wold et al., 1993]. A higher score indicates that the variable is more important. The VIP for the j-th variable is defined as

$$\mathbf{VIP}_j = \sqrt{m \sum_{i=1}^{p} \mathbf{b}_i^2 \mathbf{w}_{ji}^2 / \sum_{i=1}^{p} \mathbf{b}_i^2} \tag{3.7}$$

where $\mathbf{w}_{ji}$ is the j-th element of vector $\mathbf{w}_i$, and $\mathbf{b}_i$ is the regression weight for the i-th projection vector ($\mathbf{b}_i = \mathbf{U}_i^T\mathbf{T}_i$). VIP is employed to rank the variables to add the most discriminative in the early stages of the cascade increasing the rejection of background samples.

3.3.2 PLS Cascade

As pointed out by Viola and Jones [2001], the intuition behind a rejection cascade is to build smaller, but efficient classifiers, such that the simpler and faster ones are applied in samples which are easy to classify as background and the more complex and slower ones are employed on samples harder to classify. In this way, background detection windows are discarded in earlier stages and the detector will be likely to focus on more promising detection windows, the one more likely to contain pedestrians.

The PLS cascade focuses on two aspects to reduce the computational cost for object detection. First, the VIP ranks the feature descriptors so that the most discriminative variables are assigned to the initial positions of the feature vector. Hence, differently from other cascades [Viola and Jones, 2001; Zhu et al., 2006] that need to learn weak classifiers for every feature to choose the best, our method learns classifiers by adding features incrementally, following their positions in the feature vector. Second, to avoid considering all features as candidates for each stage, we remove descriptors that have been already used. This aspect significantly reduces the training time. Nevertheless, this approach may lead to a reduction in the detection rate, considering that the later classifiers use less discriminative features. To improve the detection rate in later stages of the cascade, we propose two strategies for propagation of the low-dimensional feature set (*latent variables*). The *cumulative strategy* propagates the latent variables from stages $1, 2, \ldots, i$ to $i + 1$, while the *noncumulative strategy* propagates from stage i to $i + 1$. PLS requires the projection of the feature set onto a low-dimensional space when classifying a sample, hence the propagation does not increase the computational cost of the method since this information has already been extracted to run the earlier stages.

In the proposed approach, instead of pooling all descriptors and learning weak classifiers for all of them (or a random subset of them) to choose the ones that are more suitable for a stage, as in [Viola and Jones, 2001; Zhu et al., 2006], we first apply the VIP to select the most discriminative descriptors and then learn the classifiers following the descriptors rank, which reduces the computational cost for the training.

Another important feature of the proposed cascade is that each stage contains only a single classifier based on PLS, instead of a set of weak classifiers. Therefore, to learn each stage, descriptors are incrementally added following the rank of the variables. A stage is completed when the learned classifier is able to reject a certain number of negative samples and keep a specified maximum miss rate. Once a stage

is finished, the latent variables estimated by the PLS are propagated to the next stage to be used together with the descriptors. This process ends when all descriptors have been considered, as illustrated in Figure 3.7 for a cascade with n stages.

In the testing phase, the PLS Cascade evaluates a detection window d_k through each of the n stages learned in the training phase, until it either rejects or classifies the window as a pedestrian. As illustrated in Figure 3.8, the feature descriptors of a detection window are splitted among the stages of the cascade, according to the order determined during the training phase using VIP. While PLS estimates the confidence score $\mathbf{B}_{(i)}(d_k)$ of a test sample for the i-th stage of the cascade, it also generates the set of latent variables $\mathbf{T}_i$, which are propagated to the next stage using either the cumulative or the noncumulative strategy. The confidence score $\mathbf{B}_{(i)}(d_k)$ of the i-th stage is measured against a rejection threshold θ_i, determined during the learning phase aiming at the minimization of the false positive rate. Scores higher than the threshold, i.e. $\mathbf{B}_{(i)}(d_k) \geq \theta_i$, triggers the next stage of the cascade, in which the procedure is repeated. Otherwise, the detection window is rejected as non-pedestrian. Finally, a detection window d_k is classified as a pedestrian if it succeeds throughout the n stages of the cascade.

Chapter 4

Experimental Results

This chapter evaluates the approaches proposed in Chapter 3. Section 4.1 presents the data set employed in our experimental evaluation. Section 4.2 addresses the random filtering approach and evaluates the effectiveness of location regression. Section 4.3 explores the PLS Cascade. In the Section 4.4, we discuss the optimization approaches and the results obtained.

As several works in pedestrian detection, we report the performance results using Detection Error Tradeoff (DET) curves. They show the miss rate versus the amount of false positives on a log-log scale. Hence, in such curves, lower and left-most values denote better performance. Miss rate is defined as

$$\text{Miss rate} = 1 - \text{Recall} = \frac{\#\text{FalseNegatives}}{\#\text{TruePositives} + \#\text{FalseNegatives}} \tag{4.1}$$

The amount of false positives may be either *per window* (false positives per window, FPPW) or *per image* (false positives per image, FPPI). In this work we report the random filtering results using FPPI and the PLS Cascade using FPPW.

4.1 Data Sets

We conducted our experiments on the widely employed INRIA Person data set [Dalal and Triggs, 2005]. It is composed of several images in a wide range of places and backgrounds, and under different weather conditions. The images contain persons mostly standing, but they can assume any pose or orientation. The data set is split into training and testing sets. The positive training set contains 1208 samples, normalized to 64×128 pixels, as illustrated in Figure 4.1. With the left-right reflection, this number doubles yielding 2416 positive training samples. The

Figure 4.1. Positive training samples for the INRIA Person data set, normalized to 64 × 128 pixels.

negative training set contains 1218 images, which might be randomly sampled to generate negative samples of size 64 × 128. The testing set has 288 images containing pedestrians, yielding 589 pedestrians.

4.2 Random Filtering and Location Regression

In this section, we evaluate the performance of random filtering and location regresion, employed to adjust the detection window's location before presenting it to a classifier.

Considering the experimental setup described in Section 4.2.1, we evaluate the proposed approach by focusing on five points: evaluation of the detection windows selected by random filtering regarding the ground-truth (described in Section 4.2.2); adjustment of the selected windows using location regression (described in Section 4.2.3); evaluation considering a real classifier (described in Section 4.2.4); evaluation of the computational cost of each step (described in Section 4.2.5); and study of the pedestrians' distribution (described in Section 4.2.6).

Initially, we are interested to determine if random filtering is able to detect every pedestrian in an image, for which we evaluate the windows selected by random filtering regarding the ground-truth – i.e., a perfect classifier (Section 4.2.2). Although in this experiment random filtering misses only a few persons in the image, when these windows are presented to a real classifier (e.g., the PLS Detec-

tor [Schwartz et al., 2009]), the detection rate decreases due to slight displacements in the detection windows. Hence, we employ location regression to adjust the detection windows selected by random filtering (Section 4.2.3) and, then, present them to a classifier (Section 4.2.4). We explore how location regression improves the detection rate and how it is influenced by larger distances between a sample and a detection window (Section 4.2.3). Finally, as we have considered up so far that pedestrians are uniformly distributed in the images, we study their distribution which might yield a better filtering approach (Section 4.2.6).

4.2.1 Experimental Setup

From the training partition of the INRIA Person data set, we generated 500,000 negative samples to be used in the learning phase of the detector. Note that, differently from Dalal and Triggs [2005], no bootstrapping is performed to obtain hard negative samples for training the PLS Detector, which may reduce the recall achieved. Since in this work we are only interested in the relative comparisons, this does not affect the experiments.

We consider the following setup to execute the experiments. The detection window size is set up to 64×128 pixels. The images are resized by a range of scales (increased by 10%), so that pedestrians with sizes between 60 and 700 pixels can be detected. The detection window is shifted by a stride of 12% and 4% of the object width and height, respectively.

We use the PLS Detector [Schwartz et al., 2009] as our baseline (any other sliding window based detector could be used instead). The detector was trained using the same Histograms of Oriented Gradients (HOG) setup used by Dalal and Triggs [2005], i.e, a feature vector with $3,780$ dimensions.

To execute the location regression, we consider two feature descriptors, namely, pixel intensity and HOG. While the former simply considers the pixel intensity within the detection windows, which is extremely fast to compute, the latter is extracted following setup proposed by Dalal and Triggs [2005]. As we show in Section 4.2.4, the number of detection windows remaining after the random filtering is small, hence the HOG extraction does not increase significantly the computational cost.

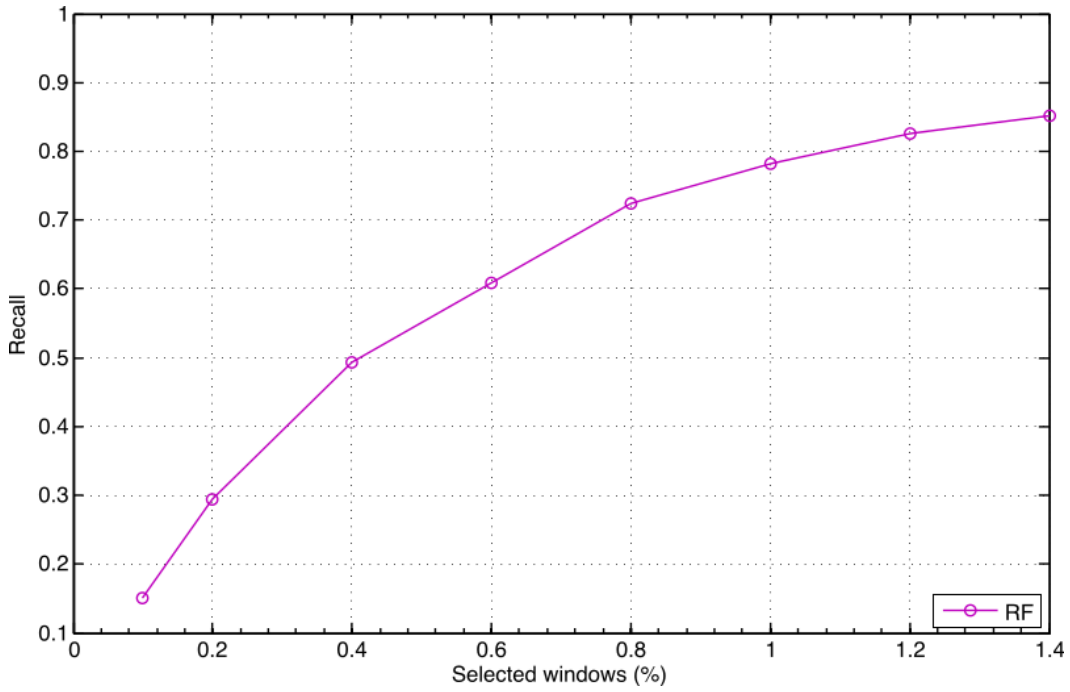


Figure 4.2. Achievable recall as a function of the number of selected windows, evaluated on the INRIA data set (RF: random filtering).

4.2.2 Ground-truth Comparison

To verify the applicability of the Maximum Search Problem theorem, presented in Section 3.2.1, this experiment determines the ratio of pedestrians that are covered[1] by at least one detection window as a function of the percentage of selected windows. This can be verified according to their correct position given by the ground-truth.

Figure 4.2 shows that a random selection of 1.4% of detection windows is enough to detect 83% of the pedestrians on the INRIA data set (if the classifier provided perfect results). Note that according to the theorem, approximately 0.2% would be enough to approximate the maximum (find at least one pedestrian). However, since, on average, two people are present in each image, this value increases. In addition, we cannot achieve maximum recall score in this experiment because we are not padding the images, which means that people near to the edge of the images cannot be fit within a detection window.

In the next section, we compare the results of random filtering (RF) when coupled with location regression (LR). In addition, we evaluate different feature setups for random filtering, for the best one is employed with the detector (described in

[1] A window is considered covered when the Jaccard coefficient (Equation 3.2) is greater than 0.5.

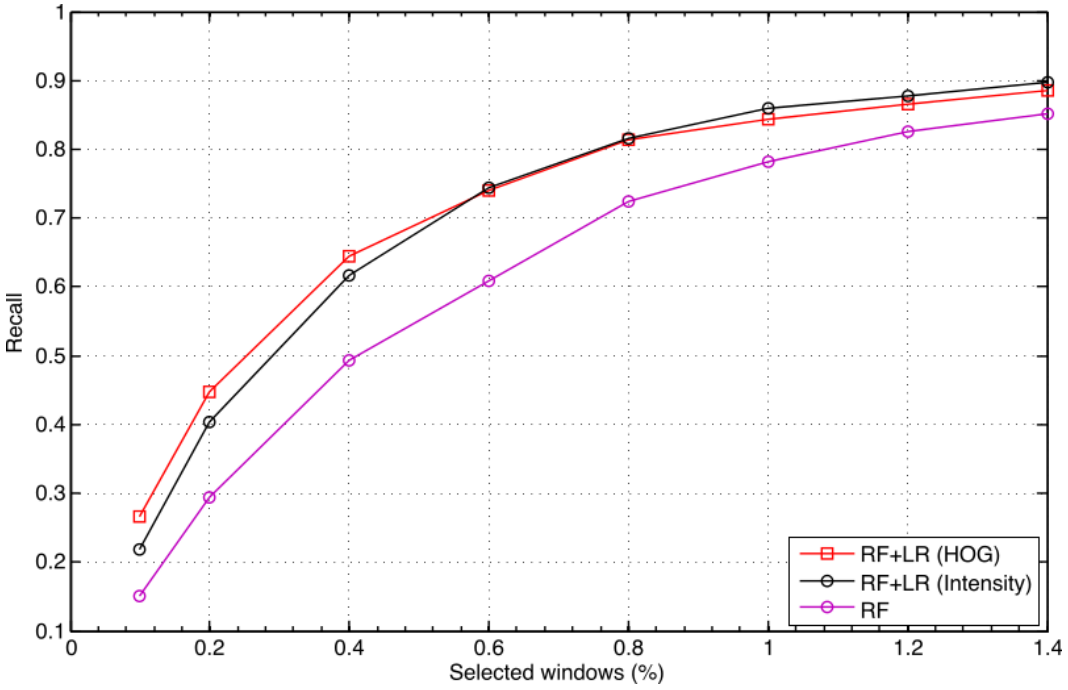


Figure 4.3. Achievable recall as a function of the number of selected windows, evaluated on the INRIA data set (RF: random filtering, LR: location regression).

Section 4.2.4).

4.2.3 Location Regression

After applying the random filtering, we adjust the detection windows using location regression. This section evaluates the detection rate obtained by incorporating this approach. Figure 4.3 reports the results achieved when applying the technique, using either pixel intensity or HOG as feature descriptor. As we can see, the regression is able to correct the position of the detection windows and, consequently, increase the recall achieved by the random filtering to a recall of 0.9 when 1.4% of the detection windows are selected. Note that these results show the maximum achievable recall if the detector provided perfect results.

In addition, this section evaluates which feature descriptors yields the best results for location regression. Since it must be fast, we trained the regression models with two simple feature descriptors. The first model uses pixel intensity as feature, which consists of concatenating all pixels within the detection window into a feature vector. The second model is based on HOG, also using the Dalal and Triggs' setup. Figure 4.3 shows that both models present comparable results.

We also evaluated the variance of the random filtering and location regres-

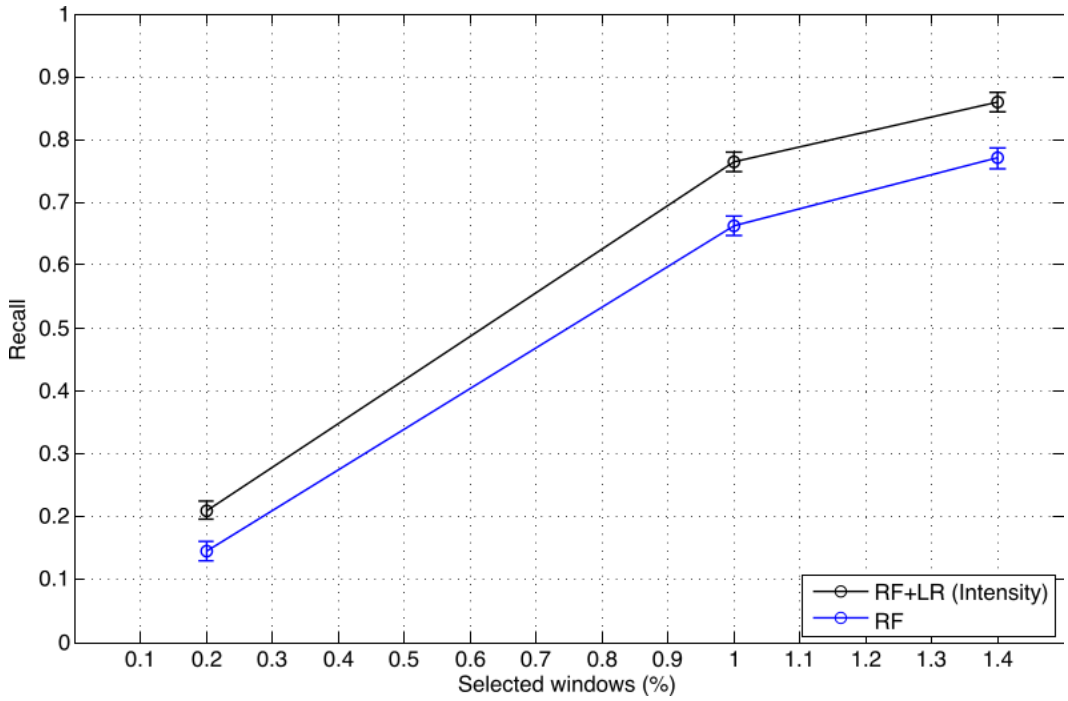


Figure 4.4. Variance of the random filtering and location regression.

sion. We repeated this experiment 33 times for random selections of 0.2%, 1%, 1.4%. In this experiment, location regression was employed with pixel intensity features since we are only interested in the variance. Figure 4.4 shows that the variance of the approaches is negligible, even considering its randomness. Hence, we can consider that the random filtering is reliable to achieve a given recall score as a function of the selected windows.

In the next experiment, we evaluated the predicted values of location regression (also using pixel intensity as feature) when the detection window moves aways from a pedestrian. For easier comprehension of their behavior, we analysed the horizontal and vertical predictions independently. Figures 4.5 and 4.6 shows that the error slightly increases when the detection window moves farther, demonstrating that location regression becomes less accurate for windows with large displacements. In these Figures, the x-axis presents the values of the ground-truth and the y-axis presents the absolute error of the regression regarding the ground-truth.

4.2.4 Pedestrian Detector

We may notice in Figure 4.2 that the selected windows miss only a few pedestrians in the data set after performing the random filtering. However, these windows still need to be presented to a classifier, which may not obtain high accuracy due to

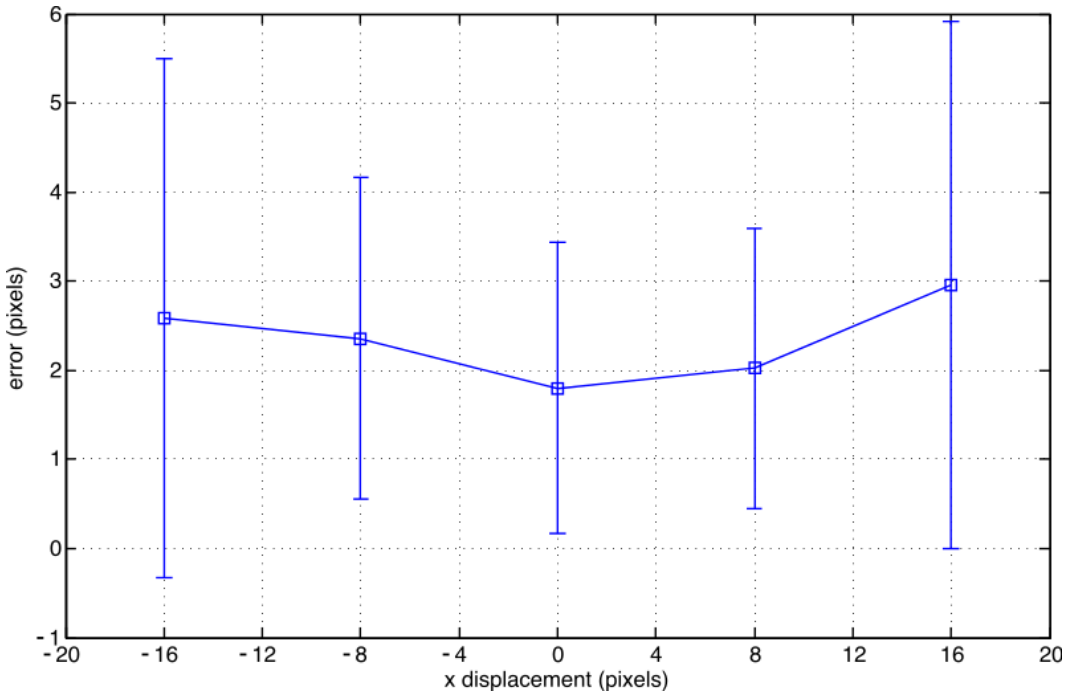


Figure 4.5. Horizontal prediction of location regression when a detection window moves away from a pedestrian. x-axis represents the horizontal movement, while the y-axis presents the absolute error regarding the ground-truth.

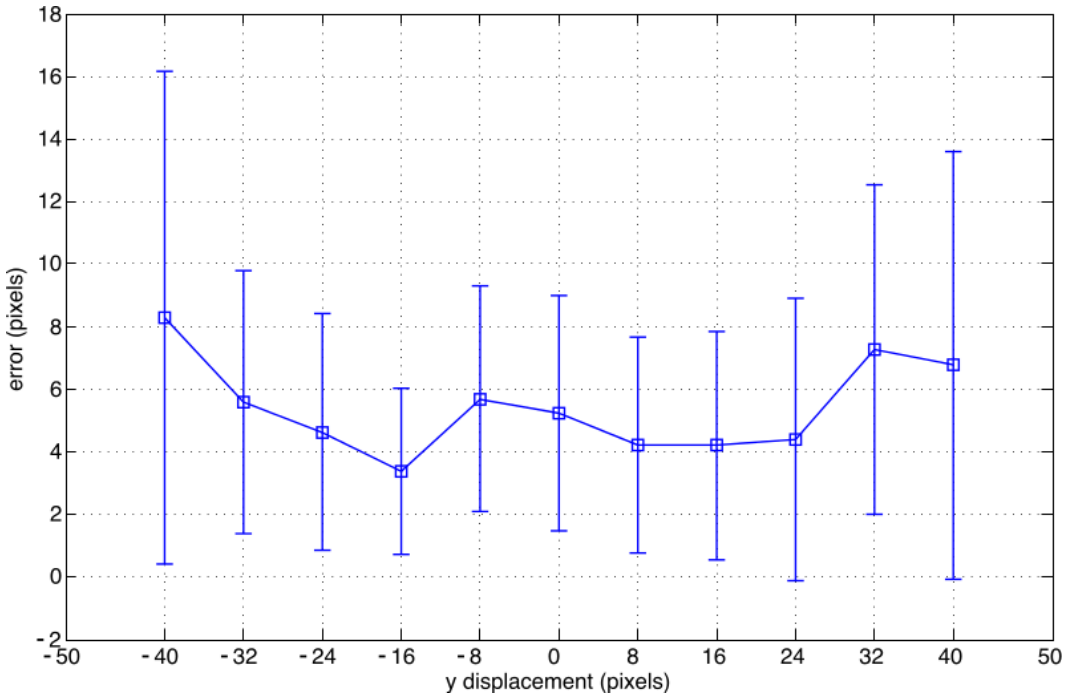


Figure 4.6. Vertical prediction of location regression when a detection window moves away from a pedestrian. x-axis represents the vertical movement, while the y-axis presents the absolute error regarding the ground-truth.

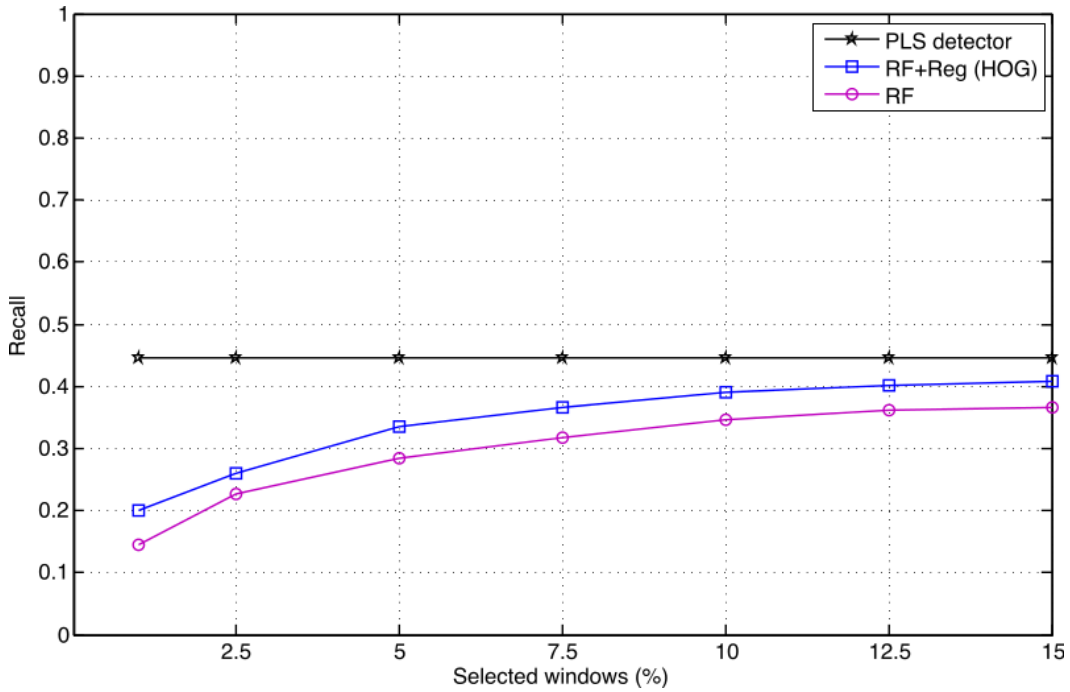


Figure 4.7. Recall achieved at 1 FPPI when the selected detection windows are presented to the PLS detector. The PLS detector is shown as line because it is executed with 100% of the detection windows (without filtering).

some displacement of windows regarding to the person's location so that is does not result in high responses. This experiment evaluates how that may affect the accuracy of the detector/classifier. In the following experiments, we discuss only results achieved with HOG, because it has lower dimensionality and consequently is less subject to issues regarding the curse of dimensionality.

The results in Figure 4.7 show the recall obtained at one false positive per image (FPPI). Even after executing the random filtering, the accuracy is still comparable to the original detector, which considers 100% of the detection windows (no windows are discarded). However, to achieve similar results, the number of selected detection windows had to be larger than the result achieved by the ground truth experiment described in Section 4.2.2. This indicates that, although the correct detection windows have been selected, the PLS detector does not provide high responses for all the correct windows. By using the location regression, we could improve the random filtering results, increasing the recall to 40.9% (close to the 45% achieved by the original detector). It is worth noting that, for this experiment, the PLS Detector was executed with a single stage. Therefore, it is not the same version employed by Schwartz et al. [2009].

The results achieved with this experiment demonstrate the high influence of

the classifier in the accuracy. Even though a random selection of 1.4% detection windows covers more than 85% of the pedestrians in this data set (Figure 4.2), when a classifier was employed to the same number of windows, the results decreased significantly. According to Figure 4.7, the maximum achievable by the PLS Detector at 1 FPPI is 45%, but when 1.4% detection windows were selected, the recall achieved is approximately 15%, which was increased to 20% when the location regression was considered. If it had followed the results in Figure 4.2, it should have obtained at least 40% of recall (90% of the maximum achievable). Therefore, these results emphasize the need for further studies to increase the robustness of classifiers to small displacements of the detection windows to allow the fully exploitation of techniques such as random filtering.

4.2.5 Computational Cost

This section reports the speedup achieved when compared to the execution of the detector PLS alone. The results in Table 4.1 show the speedup for the results reported on Figure 4.7. The random filtering was able to achieve significant reduction in the computational cost, which also justifies its usage. In addition, by comparing the last two rows in Table 4.1 one may note that the employment of the location regression presents a low overhead.

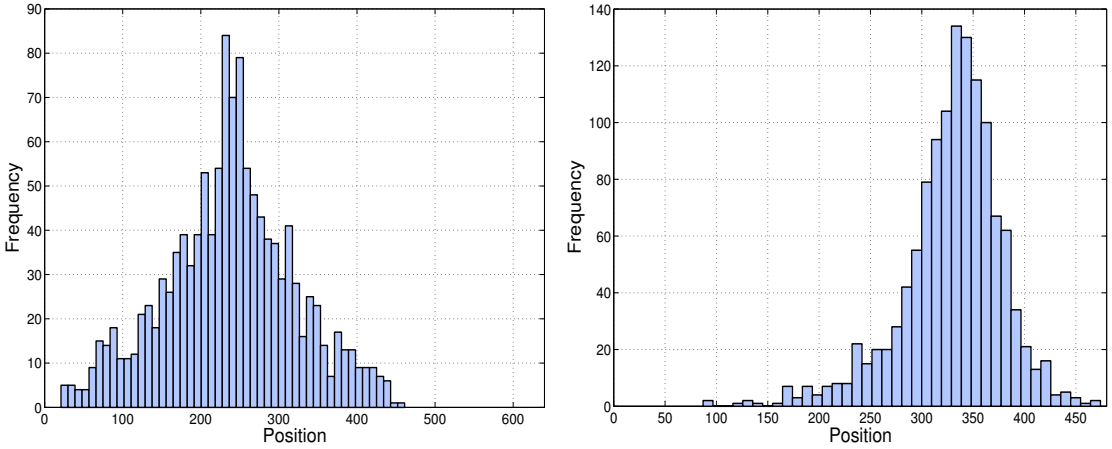
Table 4.1. Relative speedup achieved with the proposed method when compared to original detector alone (RF: random filtering, LR: location regression using HOG).

SETUP	PERCENTAGE OF SELECTED WINDOWS							
	1%	2.5%	5%	7.5%	10%	12.5%	15%	100%
PLS Detector	–	–	–	–	–	–	–	1.00×
RF	67.91×	37.64×	21.81×	15.58×	12.10×	9.62×	8.45×	–
RF+LR	64.83×	35.13×	20.40×	14.59×	11.39×	9.09×	7.97×	–

4.2.6 Pedestrians' Distribution

Up to now, we have considered that pedestrians are uniformly distributed over an image. However, this is not necessarily true. This assumption may lead to more erroneous or inefficient sampling.

We have built histograms of the x- and y-coordinates of the pedestrians' centroids for every image. These histograms were collected from INRIA Person data



(a) INRIA (x-axis) (b) INRIA (y-axis)

Figure 4.8. Histogram of the distribution of the pedestrian according to the image coordinates in the x and y axes.

set. According to the histograms displayed in Figure 4.8, it is clear that the distribution of the pedestrians in the frames is not uniform. We intend to study such characteristic in future works aiming at reducing even more the number of detection windows that have to be randomly sampled to detect all pedestrians in the image.

4.3 PLS Cascade

In this section, we analyze the impact of the PLS technique in a cascade structure using VIP to sort the feature descriptors according to their discriminative information as well as the contribution of the propagation of the latent variables to improve the detection rates.

The experimental evaluation of the proposed method focuses on four main points: the results achieved by the baseline cascade, i.e., the cascade without VIP or propagation of latent variables (described in Section 4.3.2); the contribution of the VIP to rank the features so that a large number of detection windows are rejected in early stages (described in Section 4.3.3); the reduction in the miss rate achieved by the propagation of latent variables to the next stage (described in Section 4.3.4); and comparison with other approaches (described in Section 4.3.5). These points are discussed in more details in the next sections, after the description of the experimental setup.

4.3.1 Experimental Setup

The experimental evaluation was based on the extraction of a set of HOG features with the same setup as in Dalal and Triggs [2005]. The HOG features are extracted in blocks of size 16×16, divided into 2×2 cells of size 8×8. The experiments were performed in the INRIA Person data set [Dalal and Triggs, 2005] (described in Section 4.1) resulting in a total of 3,780 feature descriptors per detection window. For all experiments, we employed a 5-fold cross-validation to estimate the threshold used to reject a sample in a given stage and to estimate the number of latent variables for the PLS models. The minimum detection rate for each stage was set to 0.9 and the maximum false positive rate was 0.8. Following other works [Schwartz et al., 2009; Zhu et al., 2006; Dalal and Triggs, 2005], we have reported the miss rate at 10^{-4} FPPW. In addition, we report the rate of discarded samples considering the first stage of the cascade.

4.3.2 Baseline Cascade

Initially, we have learned a cascade without using VIP and without applying the propagation of latent variables to have a baseline, referred to as *baseline cascade*. This cascade is able to reject a certain number of samples in the earlier stages (20% by the first stage, as shown in Figure 4.9), which results in a lower number of projections when compared to the PLS detector (9.64% of the projections are required), as shown in Figure 4.11. However, it suffers with a high miss rate of 75.32% at 10^{-4} FPPW (Figure 4.10).

4.3.3 Application of the VIP

Aiming at rejecting a larger number of samples in the earlier stages, the VIP has been applied to rank the features according to their discriminative power. The features are added to the stages following their ranks, as discussed in Section 3.3.2. We consider two approaches with the VIP. The first, referred to as *VIP once*, the VIP is applied once before the beginning of the cascade considering all variable at once and the second consists of applying VIP before each stage of the cascade, not considering the variables already used in previous stages.

On the one hand, the results achieved by both approaches increased significantly the number of samples rejected in the first stage of the cascade (around 70%, as shown in Figure 4.9), compared to the baseline cascade. According to Figure 4.11, the number of projection also has been reduced to 6.63% compared to the PLS de-

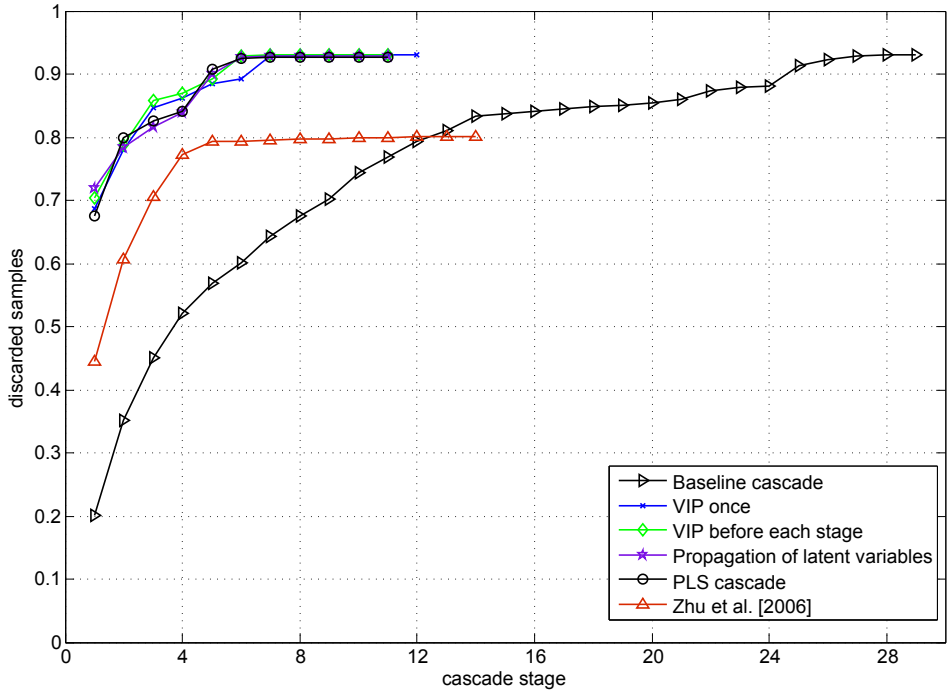


Figure 4.9. Cumulative rate of discarded samples as a function of the stages, reported for different setups and methods. The setup referred to as *PLS cascade* is composed of VIP once and incremental propagation of latent variables.

tector Schwartz et al. [2009]. On the other hand, the miss rate has increased to more than 88.91% at 10^{-4} FPPW (Figure 4.10). The reason for this poor result is that the discriminative feature descriptors have been employed in the earlier stages, and in the later stages (responsible for discarding complex samples), features with low discriminative power are used.

Since both approaches present similar results, we have chosen to use the *VIP once* throughout the remaining experiments because it is applied only once, which reduces the computational cost to learn the cascade.

4.3.4 Propagation of Latent Variables

To incorporate more discriminative information in the remaining stages, we evaluate the benefits of propagating the latent variables to the later stages of the cascade. According to experimental evaluation, the best number of latent variables propagated to the next stage is 11, number used in the experiments.

Propagating latent variables from one stage to the next (referred to as *propagation of latent variables*) shows a positive effect by reducing the miss rate when compared to the previous experiments (36.01% at 10^{-4} FPPW, as shown in Figure 4.10),

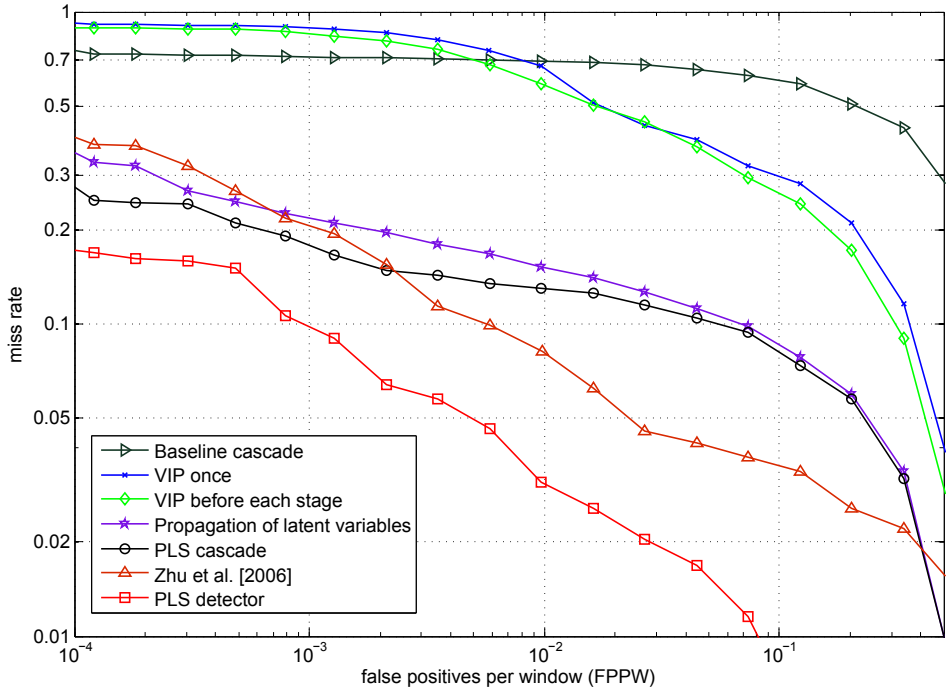


Figure 4.10. Results achieved with different setups and methods, reported in a detection error tradeoff plot.

which is expected once the stages have more information to classify the samples. Even though there is a trade off between the size of the set propagated and the number of projections (the more variables are propagated, the more projections are needed), this number is still low: only 7.50% of the projections required by the PLS detector, as shown in Figure 4.11.

Finally, we considered the cumulative propagation of latent variables. Therefore, instead of propagating the variable only from stage i to $i+1$, the model in the i-th stage uses the variables from stages 1, 2, ..., $i-1$. The setup considering VIP once and the cumulative propagation of latent variables is referred to as *PLS cascade* and it will be set as the final configuration for the proposed cascade. With the PLS cascade, the miss rate reduces to 28.35% at 10^{-4} FPPW (Figure 4.10) at a slight increment of 3.63% in the number of projections when compared to propagation only to the next stage.

4.3.5 Comparisons

We compared the PLS cascade with the cascade proposed by Zhu et al. [2006] (using PLS for classification, instead of SVM) and with the PLS detector proposed by Schwartz et al. [2009]. In order to establish a fair comparison, we have used

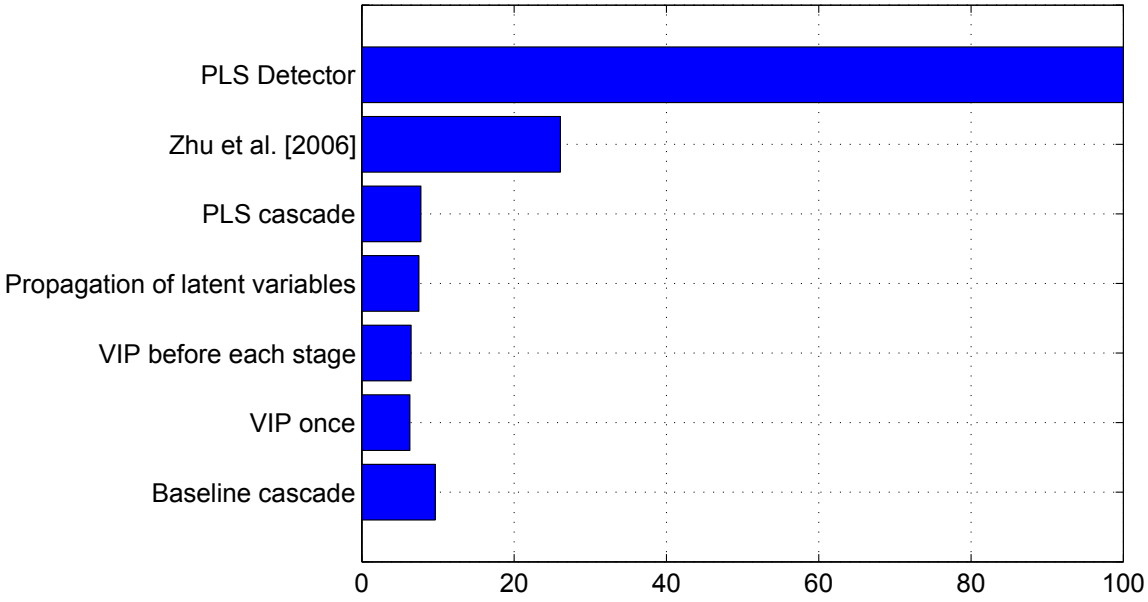


Figure 4.11. Percentage of projections performed by each method, normalized by the number of projection required by the PLS detector.

the same 3,780 feature descriptors employed to learn the PLS cascade to learn Zhu's cascade and the PLS detector.

Regarding the cascades, the results in Figure 4.10 show that the miss rate achieved by the PLS cascade (28.35% at 10^{-4} FPPW) is smaller than the one achieve by Zhu's cascade (40.16%). In addition, according to Figure 4.9, the number of samples discarded in the early stages is greater when the proposed cascade is considered (e.g., 67.45% of the detection windows are rejected by PLS cascade at the first stage and 44.46% by the Zhu's cascade), which makes the PLS cascade a faster and more accurate method.

When compared to the PLS detector, the proposed cascade achieved a higher miss rate at 10^{-4} (17.38% for the PLS detector and 28.35% for the PLS cascade), according to Figure 4.10. Even though the miss rate is higher, the proposed cascade performs only 7.49% of the projections required by the PLS detector (Figure 4.11), which makes the PLS cascade a promising approach for further investigation, which should focus mainly on the use of a larger number of feature descriptors, which is usually necessary for cascade approaches (e.g., as much as 98,928 descriptors were used by Zhu et al. [2006] to achieve similar results obtained by Dalal and Triggs [2005] with only 3,780 descriptors with their SVM-based detector).

4.4 Discussion and Remarks

This chapter has presented the experimental evaluation of our proposed optimization approaches. In this section, we present a discussion and remarks of the achieved results.

Random filtering allowed to greatly reduce the amount of detection windows processed, without significantly increasing the computational cost. The percentage of selected windows, estimated in Section 3.2.1, might be seen as a lower limit of the real estimation of the number of selected windows. Applying location regression to correct the windows selected by random filtering allows to increase the detection rate, which could be explored to select an even smaller number of windows without greatly affecting the computational cost.

Random filtering was not able to achieve the same recall obtained by the PLS Detector stand-alone, mainly due to the generalization of the classifier for non-centralized pedestrians. We will address this issue in future works by exploring other classifier's setup. In addition, based on our analysis of the pedestrians' distribution, random filtering might take advantage of this prior knowledge to perform an improved random selection. For example, in a video sequence, it would not need to select windows from known background regions, such as the sky, which would allow a higher speed up and detection rates. It is worth noting that, even though the images from the INRIA Person data set [Dalal and Triggs, 2005] are not a sequence, they still present peak-centered distributions, meaning that not all regions require to be sampled equally.

PLS Cascade allowed to reduce the number of projections performed by the PLS Detector. The experiments have shown that VIP allows faster training and rapid window rejection in earlier stages of the cascade. The cumulative strategy of propagation has obtained better results than the noncumulative, since it incorporates features of every previous stage. Finally, there is no extra cost on computing the feature space onto a low dimensional one, since it is already done when performing the PLS regression.

Finally, as both approaches focus on reducing the computational cost, we could integrate them to reduce even further the computational cost. We will address this method in future work.

Chapter 5

Conclusions

In this work, we proposed two novel optimization approaches for reducing the computational cost of pedestrian detection.

The first optimization is based on random filtering approach to discard a large number of detection windows, which is further improved by the application of a regression to correct the window location to fit the persons in the image. This approach can be applied as a early step of any sliding window based detector. For instance, it could be used as the first step of a cascade of rejection. Compared to the application of a detector method alone, our experimental evaluation showed that accurate results at a reduced computation cost may be achieved by our method even when a large number of detection windows are discarded.

The second optimization approach addresses the computational cost of the Partial Least Squares (PLS) Detector [Schwartz et al., 2009] by proposing the usage of a rejection cascade based on PLS. This method allows reducing the computational cost by discarding less promising samples earlier. In order to discard even more samples in earlier stages of the cascade, we proposed the use of the PLS-based feature sorting method VIP and to improve the detection rate, a latent variable propagation scheme is employed. Results showed that the combination of VIP and propagation of latent variables is promising due to the significant reduction on the number of projections, even when compared to a well-known cascade approach [Zhu et al., 2006].

5.1 Future Works

As future works, we intend to evaluate the integration of our proposed optimization approaches, i.e., random filtering and the PLS Cascade. As both approaches focus

on reducing the computational cost, their integration might lead to a large reduction in the cost. In addition, we will evaluate our proposed methodology in other data sets, such as the ETHZ Pedestrian data set [Ess et al., 2007].

Regarding the random filtering approach, we intend to evaluate the influence of modeling negative samples along with the positive ones. Our hypothesis is that it would allow the location regression to build a decision surface that might improve its accuracy and, possibly, reject negative windows in advance, as they would be shifted to outside of the image's boundaries. Moreover, we will extend the study on how detection window displacements affects the classifier, as it would allow to further improve the location regression to achieve exactly the same result achieved by an sliding window based detector alone at a much smaller computational cost. In addition, we will exploit the pedestrians' distribution in a data set and measure the gain of this approach.

Bibliography

Amit, Y. and Trouvé, A. (2007). Pop: Patchwork of parts models for object recognition. *International Journal of Computer Vision*, 75(2):267–282.

Benenson, R., Mathias, M., Timofte, R., and Van Gool, L. (2012a). Pedestrian Detection at 100 Frames per Second. In *IEEE Intl. Conference on Computer Vision and Pattern Recognition*, pages 1–8.

Benenson, R., Mathias, M., Timofte, R., and Van Gool, L. (2012b). Pedestrian Detection at 100 Frames per Second. In *IEEE Intl. Conference on Computer Vision and Pattern Recognition*.

Benenson, R., Timofte, R., and Van Gool, L. (2011). Stixels estimation without depth map computation. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 2010–2017.

Bourdev, L. and Brandt, J. (2005). Robust Object Detection via Soft Cascade. In *IEEE Intl. Conference on Computer Vision and Pattern Recognition*, pages 236–243.

Cheng, M.-M., Warrell, J., Lin, W.-Y., Zheng, S., Vineet, V., and Crook, N. (2013). Efficient salient region detection with soft image abstraction. In *IEEE Intl. Conference on Computer Vision*, pages 1–8.

Dalal, N. and Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. In *IEEE Intl. Conference on Computer Vision and Pattern Recognition*, pages 886–893.

Dollár, P., Appel, R., Belongie, S., and Perona, P. (2014). Fast Feature Pyramids for Object Detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, pages 1–14.

Dollár, P., Appel, R., and Kienzle, W. (2012). Crosstalk Cascades for Frame-Rate Pedestrian Detection. In *European Conference on Computer Vision*.

Dollár, P., Belongie, S., and Perona, P. (2010). The Fastest Pedestrian Detector in the West. In *British Machine Vision Conference*.

Dollár, P., Tu, Z., Perona, P., and Belongie, S. (2009). Integral Channel Features. In *British Machine Vision Conference*.

Dollar, P., Wojek, C., Schiele, B., and Perona, P. (2012). Pedestrian Detection: An Evaluation of the State of the Art. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 34:743–761.

Ess, A., Leibe, B., and Gool, L. V. (2007). Depth and appearance for mobile scene analysis. In *IEEE Intl. Conference on Computer Vision*, pages 1–8.

Felzenszwalb, P., Girshick, R., and McAllester, D. (2010a). Cascade Object Detection with Deformable Part Models. In *IEEE Intl. Conference on Computer Vision and Pattern Recognition*.

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

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., and Ramanan, D. (2010b). Object Detection with Discriminatively Trained Part Based Models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, pages 1627–1645.

Feng, J., Wei, Y., Tao, L., Zhang, C., and Sun, J. (2011). Salient object detection by composition. In *IEEE Intl. Conference on Computer Vision*, pages 1028–1035.

Fergus, R., Perona, P., and Zisserman, A. (2003). Object class recognition by unsupervised scale-invariant learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 264–271, Madison, Wisconsin.

Ferrari, V., Fevrier, L., Jurie, F., and Schmid, C. (2008). Groups of adjacent contour segments for object detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 30(1):36–51.

Forsyth, D. A. and Ponce, J. (2011). *Computer Vision: A Modern Approach*. Prentice Hall Professional Technical Reference.

Freund, Y. and Schapire, R. E. (1995). A Decision-Theoretic Generalization of Online Learning and an Application to Boosting. In *Proceedings of the Second European Conference on Computational Learning Theory*, pages 23–37, London, UK. Springer-Verlag.

Geronimo, D. and López, A. M. (2013). *Vision-based Pedestrian Protection Systems for Intelligent Vehicles*. Springer.

Geronimo, D., Lopez, A. M., Sappa, A. D., and Graf, T. (2010). Survey of Pedestrian Detection for Advanced Driver Assistance Systems. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 32(7):1239–1258.

Hampapur, A., Brown, L., Connell, J., Pankanti, S., Senior, A., and Tian, Y. (2003). Smart surveillance: applications, technologies and implications. In *Information, Communications and Signal Processing*, volume 2, pages 1133–1138.

Harel, J., Koch, C., and Perona, P. (2007). Graph-based visual saliency. In *Advances In Neural Information Processing Systems*, pages 545–552.

Hinton, G. E. (2007). Learning multiple layers of representation. *Trends in Cognitive Sciences*, pages 428–434.

Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, pages 1527–1554.

Itti, L., Koch, C., and Niebur, E. (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259.

Jain, V. and Farfade, S. S. (2013). Adapting Classification Cascades to New Domains. In *IEEE Intl. Conference on Computer Vision*.

Ke, Y. and Sukthankar, R. (2004). PCA-SIFT: a more distinctive representation for local image descriptors. In *IEEE Intl. Conference on Computer Vision and Pattern Recognition*.

Lampert, C. H. (2010). An Efficient Divide-and-Conquer Cascade for Nonlinear Object Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA, USA.

Lampert, C. H., Blaschko, M. B., and Hofmann, T. (2008). Beyond sliding windows: Object localization by efficient subwindow search. In *IEEE Intl. Conference on Computer Vision and Pattern Recognition*, pages 1–8.

Leibe, B., Leonardis, A., and Schiele, B. (2008). Robust object detection with interleaved categorization and segmentation. *International journal of computer vision*, 77(1):259–289.

Lin, Z. and Davis, L. S. (2008). A Pose-Invariant Descriptor for Human Detection and Segmentation. In *European Conference on Computer Vision*, pages 423–436.

Margolin, R., Tal, A., and Zelnik-Manor, L. (2013). What Makes a Patch Distinct? In *IEEE Intl. Conference on Computer Vision and Pattern Recognition*, pages 1139–1146.

Marin, J., Vazquez, D., Amores, J., Lopez, A., and Leibe, B. (2013). Random forests of local experts for pedestrian detection. In *IEEE Intl. Conference on Computer Vision*, pages 1–8.

Masaki, I., Horn, B. K., Bilgiç, B., et al. (2010). *Fast human detection with cascaded ensembles*. PhD thesis, Massachusetts Institute of Technology.

Oro, D., Fernández, C., Saeta, J. R., Martorell, X., and Hernando, J. (2011). Real-time gpu-based face detection in hd video sequences. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 530–537. IEEE.

Ouyang, W. and Wang, X. (2012). A Discriminative Deep Model for Pedestrian Detection with Occlusion Handling. In *IEEE Intl. Conference on Computer Vision and Pattern Recognition*, pages 3258–3265.

Ouyang, W. and Wang, X. (2013). Joint deep learning for pedestrian detection. In *IEEE Intl. Conference on Computer Vision*, pages 1–8.

Paisitkriangkrai, S., Shen, C., and Zhang, J. (2008). Fast Pedestrian Detection Using a Cascade of Boosted Covariance Features. *IEEE Trans. Circuits and Systems for Video Technology*, pages 1140–1151.

Park, D., Ramanan, D., and Fowlkes, C. (2010). Multiresolution Models for Object Detection. In *ECCV*.

Plantinga, W. and Dyer, C. (1986). An algorithm for constructing the aspect graph. In *Foundations of Computer Science, 1986., 27th Annual Symposium on*, pages 123–131. IEEE.

Porikli, F., Bremond, F., Dockstader, S., Ferryman, J., Hoogs, A., Lovell, B., Pankanti, S., Rinner, B., Tu, P., and Venetianer, P. (2013). Video Surveillance: Past, Present, and Now the Future. *Signal Processing Magazine*, pages 190–198.

Prisacariu, V. and Reid, I. (2009). fastHOG - a real-time GPU implementation of HOG. Technical report, Department of Engineering Science, Oxford University.

Rosipal, R. and Kramer, N. (2006). Overview and Recent Advances in Partial Least Squares. *Lecture Notes in Computer Science*, 3940:34–51.

Schneiderman, H. and Kanade, T. (2000). A statistical method for 3d object detection applied to faces and cars. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 1, pages 746–751. IEEE.

Schölkopf, B. and Smola, A. J. (2002). *Learning with kernels: support vector machines, regularization, optimization and beyond*. MIT press.

Schwartz, W., Kembhavi, A., Harwood, D., and Davis, L. (2009). Human Detection Using Partial Least Squares Analysis. In *IEEE Intl. Conference on Computer Vision and Pattern Recognition*.

Schwartz, W. R. (2012a). Aplicação de Observação de Pessoas em Computação Forense. In *Workshop de Forense Computacional (WFC) in SBSEG 2012 (XII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais)*, pages 1–14.

Schwartz, W. R. (2012b). Scalable People Re-Identification Based on a One-Against-Some Classification Scheme. In *IEEE Intl. Conference on Image Processing*.

Schwartz, W. R., Davis, L. S., and Pedrini, H. (2011). Local response context applied to pedestrian detection. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, Lecture Notes in Computer Science. Springer Berlin Heidelberg.

Schwartz, W. R., de Melo, V. H. C., Pedrini, H., and Davis, L. S. (2013a). A Data-Driven Detection Optimization Framework. *Neurocomputing*, 104:35–49.

Schwartz, W. R., de Melo, V. H. C., Pedrini, H., and Davis, L. S. (2013b). A Data-Driven Detection Optimization Framework. *Neurocomputing*. (to appear).

Silva Filho, J. G. d., Schnitman, L., and Oliveira, L. R. d. (2012). Multi-scale spectral residual analysis to speed up image object detection. In *Brazilian Symposium on Computer Graphics and Image Processing*, pages 79–86.

Tosato, D., Farenzena, M., Cristani, M., and Murino, V. (2010). Part-based human detection on Riemannian manifolds. In *IEEE Intl. Conference on Image Processing*.

Trucco, E. and Verri, A. (1998). *Introductory Techniques for 3-D Computer Vision*. Prentice Hall PTR.

Tuzel, O., Porikli, F., and Meer, P. (2007). Human detection via classification on riemannian manifolds. In *IEEE Intl. Conference on Computer Vision and Pattern Recognition*, pages 1–8.

Viola, P. and Jones, M. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. In *IEEE Intl. Conference on Computer Vision and Pattern Recognition*.

Viola, P., Jones, M., and Snow, D. (2003). Detecting pedestrians using patterns of motion and appearance. In *IEEE Intl. Conference on Computer Vision and Pattern Recognition*.

Wang, X., Han, T. X., and Yan, S. (2009). An HOG-LBP Human Detector with Partial Occlusion Handling. In *IEEE Intl. Conference on Computer Vision*, pages 1–8.

Weber, M., Welling, M., and Perona, P. (2000). Towards automatic discovery of object categories. In *IEEE Intl. Conference on Computer Vision and Pattern Recognition*, pages 101–108.

Wojek, C., Dorkó, G., Schulz, A., and Schiele, B. (2008). Sliding-Windows for Rapid Object Class Localization: A Parallel Technique. In *Proceedings of the 30th DAGM symposium on Pattern Recognition*, pages 71–81. Springer-Verlag.

Wold, H. (1985). Partial Least Squares. In *Encyclopedia of Statistical Sciences*, volume 6, pages 581–591. Wiley, New York, NY, USA.

Wold, S., Johansson, W., and Cocchi, M. (1993). PLS - Partial Least-Squares Projections to Latent Structures. In Kubinyi, H., editor, *3D QSAR in Drug Design: Volume 1: Theory Methods and Applications*, pages 523–550. Springer Verlag.

Zeng, X., Ouyang, W., and Wang, X. (2013). A cascaded deep learning architecture for pedestrian detection. In *IEEE Intl. Conference on Computer Vision*, pages 1–8.

Zhang, C. and Viola, P. A. (2007). Multiple-Instance Pruning For Learning Efficient Cascade Detectors. In *Neural Information Processing Systems*.

Zhang, L. and Nevatia, R. (2008). Efficient scan-window based object detection using GPGPU. In *IEEE Computer Vision and Pattern Recognition Workshops*, pages 1–7.

Zhu, Q., Avidan, S., Yeh, M.-C., and Cheng, K.-T. (2006). Fast Human Detection using a Cascade of Histograms of Oriented Gradients. In *IEEE Intl. Conference on Computer Vision and Pattern Recognition*, pages 1491–1498.