Improving Representation and Classification of Image and Video Data for Surveillance Applications

Andres Sanin
BSc(Biol), MSc(Biol), MSc(CompSc)

A thesis submitted for the degree of Doctor of Philosophy at The University of Queensland in 2012

School of Information Technology and Electrical Engineering
Abstract

Due to global security issues, the use of surveillance cameras has increased dramatically over the last decade. However, security cameras have not been altogether successful for preventing crime, mainly because the security videos are monitored by human operators who can become tired and distracted. Automatic visual surveillance systems have great potential for overcoming the bottleneck caused by humans in the surveillance process. A typical surveillance system includes four modules: (i) foreground extraction, (ii) object detection, (iii) tracking, and (iv) scene analysis.

The main aim of this thesis is to contribute towards automatic visual surveillance. We approach this by identifying potential areas of improvement in three of the above modules. First, for the foreground extraction module we recognise shadow removal as the best way of improving the results of current foreground extraction methods, which in turn helps with object detection and tracking. Second, in the case of object detection, as humans are the main objects of interest in surveillance applications, we focus our work on improved pedestrian detection (i.e., locating pedestrians within images). Third, in the scene analysis module, we focus on the action and gesture recognition task due to its broad range of applications in surveillance scenarios, such as detecting suspicious behaviour.

Our work on shadow detection and removal includes a thorough review of shadow detection methods published during the last decade. We also propose a novel shadow detection and removal algorithm. The new method is tested with a detailed comparative evaluation against a selection of related state-of-the-art approaches. We also contribute a dedicated dataset for the purpose of evaluating shadow detection performance. In the comparative evaluation the proposed method shows superior shadow detection performance.

In the case of pedestrian detection, we identify the potential of covariance descriptors, a form of feature representation which has been proved to produce excellent detection results. As the descriptors are represented by covariance matrices, which do not lie in a Euclidean space, taking into account the non-linear nature of the data may lead to improved detection results. A state-of-the-art method in the literature uses covariance descriptors.
and manifold-based learning using tangent spaces for pedestrian detection. We propose an improvement of this approach which combines multiple tangent spaces for improved mapping of the manifold data into Euclidean data. With this method, termed \textit{k-tangent spaces}, we show improved results on two standard datasets designed for measuring pedestrian detection performance.

We present the problem of action and gesture recognition as an extension of the object detection problem, by adding the temporal dimension. For this reason, we propose extending the two-dimensional covariance descriptors into spatio-temporal covariance descriptors, which we call \textit{Cov3D}. Additionally, we extend a manifold to Euclidean mapping which does not rely on tangent spaces and has shown to be useful for action recognition. This mapping approach is called Riemannian locality preserving projection (RLPP). Because learning with descriptors is best suited for boosting-based approaches, we propose a weighted version of the mapping approach which we call \textit{WRLPP}. The use of \textit{Cov3D} descriptors in conjunction with boosting and \textit{WRLPP} results in a state-of-the-art action and gesture recognition system. We demonstrate the superior performance of the system compared to related approaches on four challenging datasets.

The three proposed algorithms above can be embedded into current automatic surveillance systems for increased accuracy and robustness. More importantly, they provide individual contributions which are useful to any other computer vision application which requires the automatic processing and analysis of digital image and video information.
Declaration by Author

This thesis is composed of my original work, and contains no material previously published or written by another person except where due reference has been made in the text. I have clearly stated the contribution by others to jointly-authored works that I have included in my thesis.

I have clearly stated the contribution of others to my thesis as a whole, including statistical assistance, survey design, data analysis, significant technical procedures, professional editorial advice, and any other original research work used or reported in my thesis. The content of my thesis is the result of work I have carried out since the commencement of my research higher degree candidature and does not include a substantial part of work that has been submitted to qualify for the award of any other degree or diploma in any university or other tertiary institution. I have clearly stated which parts of my thesis, if any, have been submitted to qualify for another award.

I acknowledge that an electronic copy of my thesis must be lodged with the University Library and, subject to the General Award Rules of The University of Queensland, immediately made available for research and study in accordance with the Copyright Act 1968.

I acknowledge that copyright of all material contained in my thesis resides with the copyright holder(s) of that material. Where appropriate I have obtained copyright permission from the copyright holder to reproduce material in this thesis.
Publications During Candidature

**Journal Articles**


**Conference Papers**


**Co-authored Papers**


Publications Included in this Thesis


<table>
<thead>
<tr>
<th>Contributor</th>
<th>Statement of contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Andres Sanin (Candidate)</td>
<td>Conception and design of algorithm (80%)</td>
</tr>
<tr>
<td></td>
<td>Design of experiments (90%)</td>
</tr>
<tr>
<td></td>
<td>Paper writing (70%)</td>
</tr>
<tr>
<td>Conrad Sanderson</td>
<td>Conception and design of algorithm (20%)</td>
</tr>
<tr>
<td></td>
<td>Design of experiments (10%)</td>
</tr>
<tr>
<td></td>
<td>Paper writing and editing (30%)</td>
</tr>
<tr>
<td>Brian C. Lovell</td>
<td>Paper review</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Contributor</th>
<th>Statement of contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Andres Sanin (Candidate)</td>
<td>Conception and design of algorithm (80%)</td>
</tr>
<tr>
<td></td>
<td>Design of experiments (90%)</td>
</tr>
<tr>
<td></td>
<td>Paper writing (70%)</td>
</tr>
<tr>
<td>Conrad Sanderson</td>
<td>Conception and design of algorithm (20%)</td>
</tr>
<tr>
<td></td>
<td>Design of experiments (10%)</td>
</tr>
<tr>
<td></td>
<td>Paper writing and editing (30%)</td>
</tr>
<tr>
<td>Brian C. Lovell</td>
<td>Paper review</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Contributor</th>
<th>Statement of contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Andres Sanin (Candidate)</td>
<td>Conception and design of algorithm (70%)</td>
</tr>
<tr>
<td></td>
<td>Design of experiments (80%)</td>
</tr>
<tr>
<td></td>
<td>Paper writing (70%)</td>
</tr>
<tr>
<td>Conrad Sanderson</td>
<td>Conception and design of algorithm (20%)</td>
</tr>
<tr>
<td></td>
<td>Design of experiments (10%)</td>
</tr>
<tr>
<td></td>
<td>Paper writing and editing (20%)</td>
</tr>
<tr>
<td>Mehrtash Harandi</td>
<td>Conception and design of algorithm (10%)</td>
</tr>
<tr>
<td></td>
<td>Design of experiments (10%)</td>
</tr>
<tr>
<td></td>
<td>Paper writing and editing (10%)</td>
</tr>
<tr>
<td>Brian C. Lovell</td>
<td>Paper review</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Contributor</th>
<th>Statement of contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Andres Sanin (Candidate)</td>
<td>Conception and design of algorithm (70%)</td>
</tr>
<tr>
<td></td>
<td>Design of experiments (80%)</td>
</tr>
<tr>
<td></td>
<td>Paper writing (70%)</td>
</tr>
<tr>
<td>Conrad Sanderson</td>
<td>Conception and design of algorithm (20%)</td>
</tr>
<tr>
<td></td>
<td>Design of experiments (10%)</td>
</tr>
<tr>
<td></td>
<td>Paper writing and editing (20%)</td>
</tr>
<tr>
<td>Mehrtash Harandi</td>
<td>Conception and design of algorithm (10%)</td>
</tr>
<tr>
<td></td>
<td>Design of experiments (10%)</td>
</tr>
<tr>
<td></td>
<td>Paper writing and editing (10%)</td>
</tr>
<tr>
<td>Brian C. Lovell</td>
<td>Paper review</td>
</tr>
</tbody>
</table>
Contributions by Others to the Thesis

The work contained in this thesis was carried out by the author under the guidance and supervision of his advisors, Prof. Brian C. Lovell and Dr. Conrad Sanderson. Part of the work contained in this thesis was carried out by the author under the collaboration and discussions with Dr. Mehrtash T. Harandi.

Statement of Parts of the Thesis Submitted to Qualify for the Award of Another Degree

None
Acknowledgements

First and foremost, I wish to express my sincere gratitude to my advisors, Prof. Brian C. Lovell and Dr. Conrad Sanderson, for their guidance, encouragement, and support throughout my PhD studies. The drive and work-discipline of both Brian and Conrad has been exemplary, and I sincerely hope that I picked up some along the way. Brian was always available when I needed him, and I am still trying to figure how Conrad always managed to reply to my emails with his help and feedback in minutes.

I would also like to thank my friends and colleagues, in particular my manifold advisor, Dr. Mehrtash T. Harandi, and my two PhD co-conspirators, Yongkang Wong and Vikas Reddy, for their invaluable insights and countless favours. I would like to acknowledge the support of The University of Queensland and NICTA by awarding scholarships and providing necessary resources and lovely environment to undertake this work.

Finally, I am indebted to my family, my parents, my sister and lovely nieces, and to Adri, my partner in life, good roads and bad times. I would not have achieved this milestone without their strong support and understanding.
Keywords

computer vision, machine learning, pattern recognition, automatic surveillance, shadow detection, pedestrian detection, action recognition, gesture recognition, covariance descriptors, Riemannian manifolds.

Australian and New Zealand Standard Research Classifications (ANZSRC)

ANZSRC code: 080104, Computer Vision, 40%
ANZSRC code: 080106, Image Processing, 30%
ANZSRC code: 010401, Applied Statistics, 30%

Fields of Research (FoR) Classification

FoR code: 0801, Artificial Intelligence and Image Processing, 70%
FoR code: 0104, Statistics, 30%
## Contents

1 Introduction 21
  1.1 Goals and Challenges .................................. 24
  1.2 Contributions .......................................... 25
    1.2.1 Shadow detection and removal contributions ...... 26
    1.2.2 Pedestrian detection contributions ............... 26
    1.2.3 Action and gesture recognition contributions .... 26
  1.3 Thesis Outline ........................................ 27
    1.3.1 Composite literature review ........................ 28

2 Shadow Detection and Removal 30
  2.1 Review ................................................ 32
    2.1.1 Useful features for shadow detection ............. 33
    2.1.2 Taxonomy of recent shadow detection methods .... 36
  2.2 Methods Selected for Implementation .................. 36
    2.2.1 Chromacity-based method .......................... 38
    2.2.2 Physical method .................................. 38
    2.2.3 Geometry-based method ............................ 39
    2.2.4 Texture-based method ............................ 40
  2.3 Proposed Shadow Detection and Removal Method ....... 41
    2.3.1 Creation of candidate shadow regions ............. 42
    2.3.2 Classification of candidate shadow regions ....... 45
  2.4 Comparative Evaluation ................................ 47
    2.4.1 Shadow detection performance ..................... 49
    2.4.2 Effect on tracking performance .................... 56
  2.5 Main Findings and Future Directions ................... 60
3  Pedestrian Detection
3.1  Review ......................................................... 65
  3.1.1  Features ..................................................... 66
  3.1.2  Classification ................................................. 71
3.2  Manifolds ...................................................... 73
  3.2.1  Space of covariance matrices ............................. 74
  3.2.2  Classification on manifolds ................................. 75
3.3  Proposed Pedestrian Detection Method ....................... 77
  3.3.1  Covariance descriptors ..................................... 78
  3.3.2  $k$-tangent spaces model ................................ 78
  3.3.3  Detection algorithm ........................................ 81
3.4  Experiments ................................................... 82
  3.4.1  Preliminary experiments ................................... 85
  3.4.2  Comparative evaluation .................................... 86
3.5  Main Findings and Future Directions .......................... 88

4  Action and Gesture Recognition ................................. 91
4.1  Review ......................................................... 93
  4.1.1  Features ..................................................... 94
  4.1.2  Classification ............................................... 96
4.2  Cov3D Descriptors ............................................ 98
  4.2.1  Fast Cov3D computation ................................... 100
  4.2.2  Cov3D for action recognition ................................ 102
4.3  Mapping Approach ............................................. 104
  4.3.1  Riemannian locality preserving projection ................ 105
  4.3.2  Weighted Riemannian locality preserving projection .... 106
4.4  Proposed Action and Gesture Recognition Method ........... 107
4.5  Experiments .................................................. 110
  4.5.1  Preliminary experiments ................................... 111
  4.5.2  Comparative evaluation .................................... 113
4.6  Extended Method with Hybrid Classification .................. 118
  4.6.1  Hybrid multi-class classification .......................... 118
  4.6.2  Experiments ................................................ 120
4.7  Main Findings and Future Directions .......................... 122
5 Conclusion

5.1 Shadow Detection and Removal .................................. 124
5.1.1 Future work .................................................. 125
5.2 Pedestrian Detection .............................................. 125
5.2.1 Future work .................................................. 125
5.3 Action and Gesture Recognition ................................. 126
5.3.1 Future work .................................................. 126

Bibliography .................................................................. 127
List of Figures

1.1 Examples of automatic video processing . . . . . . . . . . . . 22
1.2 Video processing modules . . . . . . . . . . . . . . . . . . . . 23

2.1 Improving tracking with shadow removal . . . . . . . . . . . . 31
2.2 Proposed shadow removal method . . . . . . . . . . . . . . . 43
2.3 Shadow detection performance dataset . . . . . . . . . . . . . 48
2.4 Ground truth creation steps . . . . . . . . . . . . . . . . . . . 50
2.5 Shadow detection accuracy per algorithm . . . . . . . . . . . . 52
2.6 Effect of desaturation on shadow detection performance . . 54
2.7 Qualitative shadow detection results . . . . . . . . . . . . . . 55
2.8 Effect of shadow removal methods on tracking performance . 59

3.1 Applications that use object detection . . . . . . . . . . . . . . 63
3.2 Haar wavelets . . . . . . . . . . . . . . . . . . . . . . . . . . . 66
3.3 Histogram of gradient orientations . . . . . . . . . . . . . . . 67
3.4 HOG grid normalisation . . . . . . . . . . . . . . . . . . . . . 68
3.5 Obtaining a covariance descriptor . . . . . . . . . . . . . . . . 70
3.6 Tangent space illustration . . . . . . . . . . . . . . . . . . . . 73
3.7 The complex notion of linear classification on manifolds . . . 76
3.8 Examples of pedestrian images . . . . . . . . . . . . . . . . . . 84
3.9 Performance effect of the number of tangent spaces . . . . . . 86
3.10 Comparison of tangent space representations . . . . . . . . . . 87
3.11 Detection performance on the INRIA dataset . . . . . . . . . . 88
3.12 Detection performance on the DaimlerChrysler dataset . . . . 89

4.1 Applications that use action or gesture recognition . . . . . . 91
4.2 Obtaining a Cov3D descriptor . . . . . . . . . . . . . . . . . . 99
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.3</td>
<td>Spatio-temporal window in integral video</td>
<td>102</td>
</tr>
<tr>
<td>4.4</td>
<td>Example Cov3D descriptors</td>
<td>103</td>
</tr>
<tr>
<td>4.5</td>
<td>Action and gesture recognition datasets</td>
<td>110</td>
</tr>
<tr>
<td>4.6</td>
<td>Comparison of Cov3D features</td>
<td>112</td>
</tr>
<tr>
<td>4.7</td>
<td>Comparison of mapping approaches</td>
<td>113</td>
</tr>
<tr>
<td>4.8</td>
<td>Comparison of multi-class classification approaches</td>
<td>121</td>
</tr>
</tbody>
</table>
List of Tables

2.1 Taxonomy of shadow detection methods .......................... 37
2.2 Average processing time per algorithm ......................... 51
2.3 Summary of the comparative evaluation ......................... 57

4.1 Recognition performance on the KTH dataset ................. 114
4.2 Confusion matrix on the KTH dataset ....................... 114
4.3 Recognition performance on the UCF dataset ................. 115
4.4 Confusion matrix on the UCF dataset ....................... 115
4.5 Recognition performance on the CK+ dataset ................. 116
4.6 Confusion matrix on the CK+ dataset ....................... 116
4.7 Recognition performance on the Cambridge dataset ........... 117
4.8 Confusion matrix on the Cambridge dataset ................. 117
4.9 Hybrid approach on the KTK dataset .......................... 121
4.10 Hybrid approach on the UCF dataset ....................... 122
4.11 Hybrid approach on the CK+ dataset ....................... 122
4.12 Hybrid approach on the Cambridge dataset .................. 122
# List of Algorithms

3.1 Training the *k-tangent spaces* model ........................................ 81
3.2 Pedestrian detection using *k-tangent spaces* ............................. 83

4.1 *WRLPP* projection model .......................................................... 107
4.2 Boosting with *WRLPP* ................................................................. 109
4.3 Training the hybrid classifier ......................................................... 119
4.4 Predicting with the hybrid classifier .............................................. 120
## Acronyms and Abbreviations

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAM</td>
<td>Active Appearance Model</td>
</tr>
<tr>
<td>ANN</td>
<td>Artificial Neural Network</td>
</tr>
<tr>
<td>BoW</td>
<td>Bag of Words</td>
</tr>
<tr>
<td>CLM</td>
<td>Constrained Local Model</td>
</tr>
<tr>
<td>Cov3D</td>
<td>Spatio-temporal Covariance</td>
</tr>
<tr>
<td>CRF</td>
<td>Conditional Random Field</td>
</tr>
<tr>
<td>FPPW</td>
<td>False Positives Per Window</td>
</tr>
<tr>
<td>GMM</td>
<td>Gaussian Mixture Model</td>
</tr>
<tr>
<td>HDN</td>
<td>Hierarchy of Discriminative space-time Neighbourhood features</td>
</tr>
<tr>
<td>HMM</td>
<td>Hidden Markov Model</td>
</tr>
<tr>
<td>HOF</td>
<td>Histogram of differential Optical Flow</td>
</tr>
<tr>
<td>HOG</td>
<td>Histogram of Oriented Gradients</td>
</tr>
<tr>
<td>HOG3D</td>
<td>3-Dimensional Histogram of Oriented Gradients</td>
</tr>
<tr>
<td>HOSVD</td>
<td>High Order Singular Value Decomposition</td>
</tr>
<tr>
<td>LBP</td>
<td>Local Binary Pattern</td>
</tr>
<tr>
<td>MKL</td>
<td>Multiple Kernel Learning</td>
</tr>
<tr>
<td>MLP</td>
<td>Multi-Layer Perceptron</td>
</tr>
<tr>
<td>MOTA</td>
<td>Multiple Object Tracking Accuracy</td>
</tr>
<tr>
<td>MOTP</td>
<td>Multiple Object Tracking Precision</td>
</tr>
<tr>
<td>MR</td>
<td>Miss Rate</td>
</tr>
<tr>
<td>MRF</td>
<td>Markov Random Field</td>
</tr>
</tbody>
</table>
MS  Mean Shift
NN  Nearest Neighbour
PCA Principal Component Analysis
PF  Particle Filter
PM  Product Manifolds
RLPP Riemannian Locality Preserving Projection
SIFT Scale-Invariant Feature Transform
SVD  Singular Value Decomposition
SVM  Support Vector Machine
TB  Tangent Bundle
TCCA Tensor Canonical Correlation Analysis
TMS  Temporal Modelling of Shapes
WRLPP Weighted Riemannian Locality Preserving Projection
Mathematical Notation

- $x$ a vector (lower-case, bold typeface)
- $A$ a matrix (upper-case, bold typeface)
- $\mathcal{M}$ a manifold (upper-case, calligraphy typeface)
- $\mathbb{Q}$ a tensor or a special set (upper-case, blackboard typeface)
- $x(i)$ $i$-th element of vector $x$
- $A(i,j)$ element of matrix $A$, located at row $i$ and column $j$
- $\|x\|$ norm of vector $x$: $\sqrt{x(1)^2 + \ldots + x(D)^2}$
- $A^T$ transpose of matrix $A$
- $A^{-1}$ inverse of matrix $A$
- $|A|$ determinant of matrix $A$
- $\emptyset$ an empty set
- $\{u_k\}_{k=1}^N$ set of $N$ elements: $\{u_1, \ldots, u_N\}$
- $\bigcup$ set union operator
- $\text{Sym}^+_d$ space of $d \times d$ dimensional symmetric positive definite matrices
Chapter 1

Introduction

Computer vision is a branch of computer science dedicated to acquiring, processing, analysing and interpreting digital image information. The images can be obtained with various devices such as digital and infra-red cameras, ultrasound scanners, X-ray machines and microscopes. Current mobile phones are a practically ubiquitous means of capturing images and recording video. The increasing presence of capture devices encourages the creation of applications that automatically process and analyse digital image and video information.

Automatic processing and analysis of digital image and video data has potential applications in numerous fields such as security, transport, marketing, medicine, gaming, sports and entertainment. Particular applications are shown in Figure 1.1. One promising application is the development of automatic surveillance systems. Below, we describe the motivation of developing automatic surveillance systems to assist or replace human intervention. We then present a generic architecture which abstracts a surveillance system as a sequence of modules. Our goal is to make individual contributions to a subset of these modules. Although the work in this thesis is focused on surveillance applications, the results can be potentially applied to other areas which require automatic processing of image and video data.

Over the last decade the use of surveillance cameras has increased dramatically in response to global security issues. A notable example of this is the U.K., where there are more than 4 million cameras and it is estimated that
Chapter 1. Introduction

Figure 1.1: Selected examples of automatic video processing. From left to right, row by row: crowd tracking and analysis, augmented reality in sports, vision-based fire detection, human-computer interaction for gaming, automatic cell counting for medical imaging, self-driving car.

there is one camera for every 14 people\(^1\). It was believed that the presence of CCTV cameras would be enough to deter crime. However, evidence shows that this is not the case\(^2\), and in several occasions the cameras have become a target of attack themselves. Having not met the expectations as a preventive measure, visual surveillance has been more useful as an evidential measure, but even then it has been useful for solving less than three percent of crimes\(^3\).

Perhaps the main reason why CCTV cameras have not been successful as a preventive measure is that they are monitored mainly by humans. In the presence of suspicious individuals, a single human operator, who has to oversee dozens of monitors, can generally only keep track of one suspect at a time. Furthermore, since most scenarios are uneventful the vast majority of the time, it is difficult for the person to keep focused and alert. Increasing the number of human operators to cope with the increasing number of cameras is expensive and unrealistic. Furthermore, having more human operators does not guarantee successful surveillance. Automatic visual surveillance seems to be a good way to handle the increasing amount of information to be monitored, by either replacing or assisting human workers and reducing the cost of operation. There are several ways in which automatic surveillance


\(^3\)http://www.guardian.co.uk/uk/2008/may/06/ukcrime1 (Retrieved: 20-Nov-2012).
Chapter 1. Introduction

Visual surveillance systems may have several modules. In Figure 1.2 we present a common system architecture. We use this example to introduce four main components, or modules, that can be present in such systems. Foreground extraction is the stage where regions of interest, like moving objects, are marked inside each frame to produce a foreground mask. The object detection module is in charge of segmenting the objects of interest, such as people, from the foreground mask. Once the objects of interest are marked, the tracking module is in charge of finding the frame-by-frame correspondence of the objects. Finally, the scene analysis module produces the high-level result, which is application dependent. Examples of this last module include behaviour classification, identity recognition or database annotation. A broad list of commercial and academic surveillance systems is given in [VV05].

In general, surveillance systems can be described with modifications of this general architecture. Possible modifications include variations in the input, output, connectivity and number of modules. For instance, additional input such as human shape models or camera models may be used [ZNW08], the modules may be divided according to object types such as single person, people in groups or non-people [HHD00], or the output may be product of a single camera in a distributed system, where the data is consolidated in a multi-camera approach [SJS07]. The number of modules may change since not all systems require all modules to be present. For example, pedestrian detection can be done directly from the input frames without the need of foreground detection [VJ04], and recent systems for detecting anomalies in very crowded scenarios use only the foreground extraction and scene analysis modules [RSL11]. In other cases, the detection module is used to initialise
the tracking process and then the tracking module can directly detect the objects in the following frames [ARS08] or provide feedback to improve further detection [HSHG04].

1.1 Goals and Challenges

The main goal of this thesis is to contribute towards automatic visual surveillance. Instead of developing a complete system, we aim for individual contributions to the modules presented in Figure 1.2. Here, we briefly describe the current approaches used in each module and identify the main areas of improvement.

As mentioned before, the purpose of the foreground extraction module is to separate moving objects from a static background. Two common tasks in this module are to generate a model of the background, a process known as background estimation (e.g., [RSL09]), and separating segments of the video which differ from that background model into a foreground mask, a process commonly referred to as background subtraction (e.g., [SG00]) or foreground segmentation (e.g., [RSSL10]). Research in these two areas has been extensive and is out of the scope of this thesis. However, a common third step involves post-processing the foreground mask. In particular, current background subtraction methods tend to be sensitive to moving cast shadows, and we identify shadow detection and removal as an area of improvement to make foreground extraction more reliable.

Object detection can be split in two broad categories. In the first category are the approaches that use the segments, or blobs, detected as foreground and then proceed to classify them, selecting only the objects of interest (e.g., [ZLYX07]). Methods in the second category search the images directly using models specifically trained to detect objects of interest (e.g., [VJ01]). Our goal is to improve on current state-of-the-art algorithms in the second category, which do object detection directly on the video (i.e., without foreground extraction). As humans are the principal objects of interest of surveillance applications, we focus our work on the task of pedestrian detection.

In the review by Yilmaz et al. [YJS06], tracking is defined as the problem of estimating the trajectory of an object in the image plane as it moves
around a scene. Among the various issues tracking algorithms have to cope with, occlusions affect performance the most, mainly because it is difficult to determine the position and velocity of objects that are not clearly visible. In these situations, the common approach is to develop algorithms that are robust to occlusion. Examples include the use of models that maintain multiple hypotheses of each object (e.g., [SWTO04]), or segmenting the objects into multiple parts to be tracked individually (e.g., [GCS04]). In general, dealing with occlusion involves increasing the number of computations per object, which results in methods that are complex and computationally expensive. Some tracking systems adopt a hybrid approach where a simple tracking algorithm is normally used, and switch to a more complex algorithm in the presence of occlusion (e.g., [KBSL08, TKLL10]). In this fashion, tracking is a field which has been extensively studied, and where, in our view, incremental research is common.

The final module, which we call scene analysis, is an abstraction of the final, high-level, step in the pipeline which produces system output. The actual task of this module is application dependent. We are particularly interested in action and gesture recognition due to its broad range of applications such as detection of suspicious behaviour (e.g., [ASR07]). Other possible tasks in this module include recognition of individuals (e.g., [KPLKP11]), anomaly detection (e.g., [RSL11]), as well as video scene classification (e.g., [ZMZ08]) and segmentation (e.g., [WLTW05]). Because actions and gestures are patterns that occur in space as well as time, we aim to contribute towards spatio-temporal based action and gesture recognition.

1.2 Contributions

Our main goal is to improve automatic video processing and analysis. Specifically, we provide contributions to the following three computer vision tasks:

- Shadow detection and removal
- Pedestrian detection
- Action and gesture recognition
Besides providing state-of-the-art algorithms to address these three tasks, we made a series of contributions in terms of novel representation and classification approaches which could prove useful in alternative applications. Below we enumerate our contributions per task. Note that our purpose here is to give a brief overview of the contributions; further explanation of the topics will be given later in the chapters attributed to each of the three tasks.

### 1.2.1 Shadow detection and removal contributions

- We present a state-of-the-art shadow detection and removal method with the best shadow detection performance achieved to date.
- We include a thorough review of shadow detection methods published in the last decade, categorised in a feature-based taxonomy.
- We perform a detailed qualitative and quantitative comparative evaluation of the proposed method and a selection of prominent reviewed approaches.
- Additionally, we provide a dedicated image dataset for the evaluation of moving cast shadow detection algorithms, along with C++ source code for the proposed and compared methods.

### 1.2.2 Pedestrian detection contributions

- We present a state-of-the-art pedestrian detection algorithm that shows superior performance on two standard datasets.
- Along with the proposed algorithm, we include a new model for mapping manifold data into Euclidean data resulting in improved learning. The model, which we term \( k\)-tangent spaces, can be useful for any application that involves data represented as points on a manifold.

### 1.2.3 Action and gesture recognition contributions

- We present a state-of-the-art action and gesture recognition algorithm which shows superior performance on four challenging datasets.
• We propose a novel representation of spatio-temporal data based on covariance descriptors. We term the novel representation as Cov3D.

• We include a fast calculation method of the spatio-temporal Cov3D descriptors.

• We also propose a novel manifold to Euclidean mapping approach called weighted Riemannian locality preserving projection (WRLPP).

• Additionally, we suggest an extended version of the new method with a hybrid multi-class classifier, combining one-vs-all and one-vs-one classifiers for improved recognition results.

1.3 Thesis Outline

The rest of this thesis is comprised of four major parts. Chapter 2, Chapter 3 and Chapter 4 present our work on shadow detection and removal, pedestrian detection and action and gesture recognition, respectively. The above three chapters represent independent tasks in automatic visual surveillance and include their own literature review, proposed algorithms and relevant experiments. Our concluding remarks and possible future directions are presented in Chapter 5. These four chapters are summarised below:

• **Chapter 2: Shadow Detection and Removal.** The chapter begins with a description of the shadow detection task and mentions the approaches. Next it includes a thorough review of shadow detection methods published during the last decade. Based on the strengths and weaknesses of recent approaches, we design a novel shadow detection algorithm. We compare our proposed approach with a selection of the state-of-the-art approaches in a qualitative and quantitative comparative evaluation. A dedicated dataset was created for this purpose and is described in this chapter. The comparative evaluation shows that our proposed approach leads to significantly better shadow detection results.

• **Chapter 3: Pedestrian Detection.** The chapter first gives an overview of classification-based detection approaches. Second, it introduces co-
variance descriptors, a form of feature representation which has proved useful for pedestrian detection. These descriptors are represented with covariance matrices, which do not lie in a Euclidean space. Instead, they lie in special spaces known as manifolds, which we proceed to introduce. Then, we explain how learning techniques can exploit the non-linear nature of the data to obtain improved classification. We choose one particular state-of-the-art pedestrian detection approach which uses covariance descriptors in combination with manifold-based learning. We propose a way of improving the manifold representation of the data in the aforementioned approach, resulting in significantly better pedestrian detection.

• **Chapter 4: Action and Gesture Recognition.** The chapter first reviews the main related approaches and recognises the potential of methods which take into account spatio-temporal features. As covariance descriptors have shown superior results for representing images, we extend their definition to include the temporal dimension to form spatio-temporal covariance descriptors, which we call Cov3D. An extension for the fast calculation of Cov3D descriptors is presented. For learning, we recognise the potential of mapping approaches which do not rely on tangent spaces. We choose the Riemannian locality preserving projection (RLPP) and extend it into a weighted version (WRLPP) designed for boosting-based learning. Using Cov3D in conjunction with WRLPP results in improved action and gesture recognition, compared to related state-of-the-art approaches on four challenging datasets.

• **Chapter 5: Conclusion.** The chapter summarises the contributions of this thesis and enumerates new avenues and improvements for future research.

1.3.1 Composite literature review

As this work covers several distinct yet related topics, each topic has its own literature review. The overall literature review comprises:

• Section 2.1, which explains the main features that are used for detecting
moving cast shadows. The features are then used to categorise recent shadow detection methods.

- Section 2.2, which details a subset of prominent moving cast shadow detection methods chosen for implementation.

- Section 3.1, which covers the literature review of object detection methods and categorises the most relevant approaches for pedestrian detection, according to their selection of features.

- Section 3.2, which first introduces Riemannian manifolds and then gives an overview of classification approaches on manifold data.

- Section 4.1, which covers the literature review of action and gesture recognition methods.
Chapter 2

Shadow Detection and Removal

Many computer vision applications dealing with video require detecting and tracking moving objects. When the objects of interest have a well defined shape, template matching or more sophisticated classifiers can be used to directly segment the objects from the image. These techniques work well for well defined objects such as vehicles but are difficult to implement for non-rigid objects such as human bodies. A more common approach for detecting people in a video sequence is to detect foreground pixels, for example via Gaussian mixture models [RSSL10, SG00]. However, current techniques typically have one major disadvantage: shadows tend to be classified as part of the foreground. This happens because shadows share the same movement patterns and have a similar magnitude of intensity change as that of the foreground objects [NB04].

Because cast shadows can be as big as the actual objects, their incorrect classification as foreground results in inaccurate detection and can decrease tracking performance. Example scenarios where detection and tracking performance are affected include: (i) several people are merged together because of their cast shadows, and (ii) the inclusion of shadow pixels decreases the reliability of the appearance model for each person, increasing the likelihood of tracking loss. Both scenarios are illustrated in Figure 2.1. As such, removing shadows has become an unavoidable step in the implementation of robust tracking systems [MYC07].

The last review of shadow detection and removal techniques was done in 2003 by Prati et al. [PMTC03]. The review categorised shadow detection
methods in an algorithm-based taxonomy. The authors selected one algorithm from each class to do a comparative evaluation. The main conclusion was that only the simplest methods were suitable for generalisation, but in almost every particular scenario the results could be significantly improved by adding assumptions. As a consequence, there was no single robust shadow detection technique and it was better for each particular application to develop an appropriate technique according to the nature of the scene.

Since the review by Prati et al. [PMTC03], many new methods have been proposed. We extend their survey to include methods published during the last decade. Although progress has been made to produce more accurate methods, robustness is still an issue; we observe that current methods tend to have at least one scenario where they perform poorly. We propose a new shadow detection method which combines chromacity and texture features and aims to be robust and accurate at the same time. The method first uses chromacity information to segment large candidate shadow regions and then
classifies these regions as shadows or foreground objects using region-level gradient correlation. An extensive comparative evaluation shows that the proposed method outperforms a selection of prominent methods in a variety of scenarios, at the cost of increased computation.

We continue the chapter as follows. In Section 2.1 we present the updated survey, which describes the main features that are used for detecting moving cast shadows, and then uses them to categorise recent shadow detection methods. In Section 2.2, a subset of prominent methods is chosen for implementation, with each selected method explained in detail. In Section 2.3 we detail our proposed shadow removal method. In Section 2.4 we perform an extensive quantitative and qualitative comparative evaluation of the selected methods and the proposed method. In Section 2.5 we present our concluding remarks.

The proposed shadow detection and removal algorithm presented in this chapter has been published in [SSL10], and the review and comparative evaluation of recent methods has been published as a survey article in [SSL12].

2.1 Review

Prati et al. [PMTC03] classified shadow detection methods in an algorithm-based taxonomy. As a secondary classification, they mentioned the types of features used by each method among three broad classes: spectral, spatial and temporal features. We have observed that the choice of features has greater impact on shadow detection results compared to the choice of algorithms. Therefore, we present a feature-based taxonomy with a secondary mention of the types of algorithms. Furthermore, we divide spectral features into intensity, chromacity and physical properties. We divide spatial features into geometry and textures.

In Section 2.1.1, we explain how each type of feature can be used to detect shadows. In Section 2.1.2, we categorise recent shadow detection methods into a feature-based taxonomy.
2.1.1 Useful features for shadow detection

Most of the following features are more useful for detecting shadows when the frame, which contains objects and their shadows, can be compared with an estimation of the background, which has no objects or moving cast shadows. This review focuses in detecting shadows produced by moving objects in video sequences, where it is reasonable to assume that a clear view of the background can be obtained, or that the background can be estimated even in the presence of foreground objects [TKBM99, RSSL11].

Intensity

The simplest assumption that can be used to detect cast shadows is that regions under shadow become darker as they are blocked from the illumination source. Furthermore, since there is also ambient illumination, there is a limit on how much darker they can become. These assumptions can be used to predict the range of intensity reduction of a region under shadow, which is often used as a first stage to reject non-shadow regions [LD07, HC09, HHCC03, TLH05, ZFX06]. However, there are no methods which rely primarily on intensity information for discriminating between shadows and objects.

Chromacity

Most shadow detection methods based on spectral features use colour information. They use the assumption that regions under shadow become darker but retain their chromacity. Chromacity is a measure of colour that is independent of intensity. For instance, after a green pixel is covered by shadow it becomes dark-green, which is darker than green but has the same chromacity. This colour transition model where the intensity is reduced but the chromacity remains the same is normally referred to as colour constancy [HHD99] or linear attenuation [HC09]. Methods that use this model for detecting shadows often choose a colour space with better separation between chromacity and intensity than the RGB colour space (e.g., HSV [CGPP03], c1c2c3 [SCE04], YUV [CSK10], normalised RGB [CSE05]), or a combination of them [SL10]. Most of these methods are simple to implement and computationally inexpensive. However, because they make comparisons at the pixel-level,
they are susceptible to noise [PMTC03]. Furthermore, they are sensitive to strong illumination changes and fail with strong shadows [NB04].

Physical properties

The linear attenuation model assumes that the illumination source produces pure white light [NB04], which is often not the case. In outdoors environments, the two major illumination sources are the sun (white light) and the light reflected from the sky (blue light). Normally, the white light from the sun dominates any other light source. When the sun’s light is blocked, the effect of sky illumination increases, shifting the chromacity of the region under shadow towards the blue component. Nadimi and Bhanu [NB04] proposed a dichromatic model which takes into account both illumination sources to better predict the colour change of shadowed regions. Further work has been done to create more general non-linear attenuation models accounting for various illumination conditions in both indoor and outdoor scenarios [MBZ08, HC09]. Alternatively, some methods address the non-linear attenuation problem by learning the appearance that every pixel has under shadow without explicitly proposing an attenuation model [PT05, LHTW07, MBZ07, JP08]. These methods that try to model or learn the specific appearance of shadow pixels are typically referred to as physical approaches. By learning or modelling particular scenarios, these methods tend to be more accurate than chromacity methods (direct comparisons are reported in [LHTW07, JP08]). However, since they are still limited to spectral properties, their main disadvantage involves dealing with objects having similar chromacity to that of the background [HC09].

Geometry

In theory, the orientation, size and even shape of the shadows can be predicted with proper knowledge of the illumination source, object shape and the ground plane. Some methods use this information to split shadows from objects [HHCC03, YYK03, NP06, FQS08, CA10]. The main advantage of geometry features is that they work directly in the input frame; therefore, they do not rely on an accurate estimation of the background reference. However,
methods that use geometry features impose scene limitations such as: specific object types, typically pedestrians (i.e., standing people) [HHCC03, CA10] or vehicles [YYK03, FQS08]; requiring objects and shadows to have different orientation [HHCC03, CA10]; and assuming a unique light source [NP06] or a flat background surface [FQS08]. Additionally, current geometry-based methods are not designed to deal with objects having multiple shadows or (except for [HHCC03]) multiple objects detected as a single foreground blob.

**Textures**

Some methods exploit the fact that regions under shadow retain most of their texture. Texture-based shadow detection methods typically follow two steps: (1) selection of candidate shadow pixels or regions, and (2) classification of the candidate pixels or regions as either foreground or shadow based on texture correlation. Selection of the shadow candidates is done with a weak shadow detector, usually based on spectral features. Then, each shadow candidate is classified as either object or shadow by correlating the texture in the frame with the texture in the background reference. If a candidate’s texture is similar in both the frame and the background, it is classified as shadow. Various methods perform this correlation with various techniques (e.g., normalised cross-correlation [TLH05], gradient or edge correlation [JS02, XLLY05], orthogonal transforms [ZFX06], Markov or conditional random fields [WLW06, QLLL10], Gabor filtering [LD07]). Texture correlation is a potentially powerful method for detecting shadows as textures are highly distinctive, do not depend on colours, and are robust to illumination changes. However, texture-based shadow detection methods tend to be slow as they often have to compute one or several neighbourhood comparisons for each pixel.

**Temporal features**

Finally, since moving cast shadows share the same movement pattern as the objects that produce them, the same temporal consistency filters that have been applied to the objects can be applied to the shadows [CSE05, NB04, LHTW07, NP06]. This filtering usually enhances the detection results by
keeping only the pixels that are consistent in time. However, as with the intensity features, there are no methods which rely primarily on temporal features for shadow detection.

2.1.2 Taxonomy of recent shadow detection methods

We categorise shadow detection methods published during the last decade according to their feature choice. Although some of the methods use more than one feature, we take into account the feature that makes the dominant contribution to the detection results or the novelty of the paper. As mentioned in Section 2.1.1, intensity features are used mainly as a first step for detecting shadows, and temporal features are mainly used for filtering the detection results. Therefore, all the reviewed methods were classified into one of four categories: (i) chromacity-based methods, (ii) physical methods, (iii) geometry-based methods, (iv) texture-based methods. Our taxonomy is detailed in Table 2.1. Methods are sorted by year in the first column and grouped by category. The additional columns show secondary classifications within each category. Chromacity-based methods are subdivided according their colour space, level of granularity and additional spatial or temporal verification. Physics-based methods are subdivided according to their physical shadow model, learning algorithm and additional spatial or temporal cues. Geometry-based methods are subdivided according to their supported object type, whether they support multiple objects per blob, their main geometrical cue and additional cues. Texture-based methods are subdivided according to their weak shadow detector, texture correlation method and the size of the regions used in the correlation. The highlighted methods were chosen for the comparative evaluation and are explained in Section 2.2.

2.2 Methods Selected for Implementation

For practical reasons, only a subset of the reviewed methods was implemented. In this section, we detail one prominent method from each category. These methods were implemented and used for the comparative evaluation in Section 2.4.
<table>
<thead>
<tr>
<th>Paper</th>
<th>Model</th>
<th>Learning</th>
<th>Spatial or temporal cues</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nadimi and Bhanu 2004 [NB04]</td>
<td>Dichromatic</td>
<td>—</td>
<td>Spatio-temporal test</td>
</tr>
<tr>
<td>Porikli and Thornton 2005 [PT05]</td>
<td>—</td>
<td>Shadow flow</td>
<td>—</td>
</tr>
<tr>
<td>Liu et al. 2007 [LHTW07]</td>
<td>—</td>
<td>Gaussian mixture model</td>
<td>Markov random fields and tracking</td>
</tr>
<tr>
<td>Martel-Brisson and Zaccarin 2007 [MBZ07]</td>
<td>—</td>
<td>Gaussian mixture model</td>
<td>—</td>
</tr>
<tr>
<td>Martel-Brisson and Zaccarin 2008 [MBZ08]</td>
<td>General</td>
<td>Kernel based</td>
<td>—</td>
</tr>
<tr>
<td>Joshi and Papanikolopoulos 2008 [JP08]</td>
<td>—</td>
<td>Semisupervised (SVM)</td>
<td>Edges</td>
</tr>
<tr>
<td>Huang and Chen 2009 [HC09]</td>
<td>General</td>
<td>Gaussian mixture model</td>
<td>Gradients (attenuation)</td>
</tr>
</tbody>
</table>

### Geometry-based methods

<table>
<thead>
<tr>
<th>Paper</th>
<th>Objects</th>
<th>Blob segmentation</th>
<th>Main cue</th>
<th>Other cues</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hsieh et al. 2003 [HHCC03]</td>
<td>People</td>
<td>Via head detection</td>
<td>Orientation</td>
<td>Intensity and location</td>
</tr>
<tr>
<td>Yoneyama et al. 2003 [YYK03]</td>
<td>Vehicles</td>
<td>—</td>
<td>2D models</td>
<td>Vanishing point</td>
</tr>
<tr>
<td>Nicolas and Pinel 2006 [NP06]</td>
<td>Any</td>
<td>—</td>
<td>Light source</td>
<td>Temporal filter</td>
</tr>
<tr>
<td>Fang et al. 2008 [FQS08]</td>
<td>Vehicles</td>
<td>—</td>
<td>Wave transform</td>
<td>Spectral</td>
</tr>
<tr>
<td>Chen and Aggarwal 2010 [CA10]</td>
<td>People</td>
<td>—</td>
<td>Log-polar coordinates</td>
<td>Colour and oriented gradients</td>
</tr>
</tbody>
</table>

### Texture-based methods

<table>
<thead>
<tr>
<th>Paper</th>
<th>Weak detector</th>
<th>Texture correlation</th>
<th>Correlation level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Javed and Shah 2002 [JS02]</td>
<td>Colour segmentation</td>
<td>Gradient direction correlation</td>
<td>Medium region</td>
</tr>
<tr>
<td>Xu et al. 2005 [XLHY05]</td>
<td>—</td>
<td>Static edge correlation</td>
<td>Pixel</td>
</tr>
<tr>
<td>Tian et al. 2005 [TLHY05]</td>
<td>Intensity range</td>
<td>Normalised cross-correlation</td>
<td>Pixel</td>
</tr>
<tr>
<td>Wang et al. 2006 [WLHY06]</td>
<td>—</td>
<td>Intensity and edge DCRF filter</td>
<td>Small region</td>
</tr>
<tr>
<td>Zhang et al. 2006 [ZFXY06]</td>
<td>—</td>
<td>Orthogonal transforms</td>
<td>Small region</td>
</tr>
<tr>
<td>Leone and Distant 2007 [LD07]</td>
<td>Photometric gain</td>
<td>Gabor filter</td>
<td>Small region</td>
</tr>
<tr>
<td>Zhang et al. 2007 [ZHYW07]</td>
<td>Intensity constraint</td>
<td>Ratio edge test</td>
<td>Small region</td>
</tr>
<tr>
<td>Nghiem et al. 2008 [NBZT08]</td>
<td>Chromacity based</td>
<td>Intensity reduction ratio</td>
<td>Small region</td>
</tr>
<tr>
<td>Shoab et al. 2009 [SDY09]</td>
<td>—</td>
<td>Gradient background subtraction</td>
<td>Pixel</td>
</tr>
<tr>
<td>Pei and Wang 2009 [PW09]</td>
<td>—</td>
<td>PCA based</td>
<td>Small region</td>
</tr>
<tr>
<td>Nakagami and Nishikawa 2010 [NN10]</td>
<td>—</td>
<td>Walsh transform domain</td>
<td>Small region</td>
</tr>
<tr>
<td>Panicker and Wiley 2010 [PW10]</td>
<td>—</td>
<td>Foreground edge detection</td>
<td>Pixel</td>
</tr>
<tr>
<td>Qin et al. 2010 [QLLL10]</td>
<td>Shadow colour model</td>
<td>Local ternary pattern MRF</td>
<td>Small region</td>
</tr>
</tbody>
</table>

Table 2.1: Taxonomy of recently published shadow detection methods (see text for details).
2.2.1 Chromacity-based method

Among the chromacity methods, the most important factor is to choose a colour space with a separation of intensity and chromacity. Several colour spaces such as HSV [CGPP03], c1c2c3 [SCE04] and normalised RGB [CSE05] have proved to be robust for shadow detection [SYW07]. We chose the HSV approach proposed by Cucchiara et al. [CGPP03], since that colour space provides a natural separation between chromacity and luminosity. This shadow detection method has been widely used in surveillance applications (e.g., [MP08, FS10]). Since the value ($V$) is a direct measure of intensity, pixels in the shadow should have a lower value than pixels in the background. Following the chromacity cues, a shadow cast on background does not change its hue ($H$) and the authors noted that shadows often lower the saturation ($S$) of the points. Therefore, a pixel $p$ is considered to be part of a shadow if the following three conditions are satisfied:

$$\beta_1 \leq \left( \frac{F_p^V}{B_p^V} \right) \leq \beta_2 \quad (2.1)$$

$$\left( F_p^S - B_p^S \right) \leq \tau_S \quad (2.2)$$

$$\left| F_p^H - B_p^H \right| \leq \tau_H \quad (2.3)$$

where $F_p^C$ and $B_p^C$ represent the component values, $C$, of HSV for the pixel position $p$ in the frame ($F$) and in the background reference image ($B$), respectively. $\beta_1$, $\beta_2$, $\tau_S$ and $\tau_H$ represent thresholds that are optimised empirically. Working with alternative colour spaces may produce different but not necessarily better results [SYW07]. However, extending the pixel-level based analysis to an observation window improves results bycountering pixel-level noise [SCE04]. In our implementation of the HSV method we used a 5-by-5 observation window rather than treating each pixel separately.

2.2.2 Physical method

Research in physical models for cast shadow removal has been done incrementally. The more recent papers are extensions of previous physical models, typically removing some assumptions and improving on previous results. We
chose a recent approach by Huang and Chen [HC09] which does not make prior assumptions about the light sources and ambient illumination, and reports better results than similar methods. For a pixel $p$, given the vector from shadow to background value denoted as $v(p)$, the colour change is modelled using the 3D colour feature $x(p) = [\alpha(p), \theta(p), \phi(p)]$. Here, $\alpha(p)$ represents the illumination attenuation, while $\theta(p)$ and $\phi(p)$ indicate the direction of $v(p)$ in spherical coordinates:

\[
\alpha(p) = \frac{||v(p)||}{||BG(p)||} \quad (2.4)
\]

\[
\theta(p) = \arctan\left(\frac{v_G(p)}{v_R(p)}\right) \quad (2.5)
\]

\[
\phi(p) = \arccos\left(\frac{v_B(p)}{||v(p)||}\right) \quad (2.6)
\]

where $BG(p)$ is the background value at the pixel $p$, and the superscripts $R$, $G$, $B$ indicate the components in the RGB colour space. This colour feature describes the appearance variation induced by the blocked light sources on shaded regions. The model is learned in an unsupervised way. First, a weak shadow detector identifies pixels in the foreground that have reduced luminance and different saturation from that of the background. Then, the attenuation of these candidate shadow pixels is used to update a Gaussian mixture model of the 3D colour features, penalising the learning rate of pixels with larger gradient intensities than the background, which are more likely to be foreground objects. Finally, posterior probabilities of the model are used to classify each pixel in the foreground as either object or shadow.

### 2.2.3 Geometry-based method

Most geometry methods assume that each foreground blob contains a single object and shadow, which is not guaranteed in many computer vision applications. For this reason, we chose the method proposed by Hsieh et al. [HHCC03], which separates the blobs into individual objects before doing the geometric analysis. As in most geometry-based methods, their work assumes that the objects of interest are persons and that their shadows have a
different orientation. First, they analyse the vertical peaks on each blob to detect potential heads, and then use this information to split the blobs into person-shadow pairs. Given a person-shadow region \( R \), its centre of gravity \((\bar{x}, \bar{y})\) and orientation \(\theta\) are found as follows:

\[
(\bar{x}, \bar{y}) = \left( \frac{1}{|R|} \sum_{(x,y) \in R} x, \frac{1}{|R|} \sum_{(x,y) \in R} y \right)
\]

\[
\theta = \frac{1}{2} \arctan \left( \frac{2\mu_{1,1}}{\mu_{2,0} - \mu_{0,2}} \right)
\]

where \(|R|\) is the area of the region in pixels, and \(\mu_{p,q}\) are the correspondent central moments. The point below the centre of gravity with the maximum vertical change is considered to be the pixel where the shadow begins, and a segment oriented according to \(\theta\) is used to roughly split a candidate shadow region \(R_2\). Then, the following Gaussian model is built from the pixels in \(R_2\):

\[
G(s, t, g) = \exp \left[ - \left( \frac{w_s s^2}{\sigma_s^2} + \frac{w_t t^2}{\sigma_t^2} + \frac{w_g (g - \mu_g)^2}{\sigma_g^2} \right) \right]
\]

where \(s\) and \(t\) are the elliptical coordinates of each pixel and \(g\) its intensity (i.e., \(g = I(s, t)\)), and \(w\) and \(\sigma^2\) are the weight and variance of each component in the Gaussian. This model summarises the intensity of the shadow pixels and includes the coordinates where the shadow is expected to be. Once the model is built, every pixel in the original region \(R\) is classified as object or shadow, according to whether it agrees with the Gaussian model or not.

### 2.2.4 Texture-based method

Texture-based methods present the greatest diversity among the various categories. We chose the method proposed by Leone and Distante [LD07], as it correlates textures in small regions using Gabor functions. Region-level correlation is more robust than pixel-level correlation and Gabor functions can provide optimal joint localisation in the spatial/frequency domains [RH99]. As in the majority of the texture-based shadow detection methods, the method first creates a mask with the potential shadow pixels in the foreground. Then, if the textures of small region centred at each pixel are corre-
lated to the background reference, the pixels are classified as shadow. In this case, the potential shadow pixels are found using a photometric gain measure which gives higher probability to pixels whose intensity is lower than the background. The texture analysis is performed by projecting a neighbourhood of pixels onto a set of Gabor functions with various bandwidths, orientations and phases, and the matching between frame and background is found using Euclidean distance. Since a full Gabor dictionary may be expensive to compute, a sub-dictionary with the most representative kernels can be first found using the matching pursuit strategy [BM95].

2.3 Proposed Shadow Detection and Removal Method

As Prati et al. [PMTC03] concluded in their review, simple methods with less assumptions are better for general purpose shadow detection, while for better performance in specific environments, methods with additional assumptions yield better results. Although progress has been made in the last years to provide robust and accurate methods, this tendency is still visible. Simple approaches such as the chromacity-based methods work well in various scenarios but are not accurate enough, while more sophisticated methods, such as physical or texture-based methods, can be accurate only if certain conditions are met (i.e., saturated colours and rich textures, respectively). In this section, we propose a shadow detection method that aims to be robust and accurate at the same time.

The first step to improve accuracy while keeping the method robust is to combine spectral and spatial features so that when one fails, the other works. A common feature combination approach is to integrate the probability of pixels being shadow due to spectral and spatial features into a single cast shadow posterior probability (e.g., [MBZ08]). Alternatively, the features can be applied in sequence, where spectral features are typically used to select candidate shadow pixels which are then classified using spatial features (e.g., [NBT08]). These two approaches suffer when both features perform weakly at the same time.

The second step is to maximise the effectiveness of each particular feature for shadow detection. Geometry features impose several limitations and are
not suitable for general purpose shadow detection. Physical methods adapt locally and therefore use intensity and chromacity features as effectively as possible. Texture features, on the other hand, have only been so far used at the pixel or small region level (e.g., an arbitrary 4-by-4 pixel neighbourhood [LD07]). The problem of using small regions is that they are not guaranteed to contain significant textures. Using larger regions, which are more likely to contain significant textures, should increase the reliability of texture features for shadow detection.

We propose a new shadow detection method that uses colour features and edges to create large candidate shadow regions\(^1\) (ideally containing the whole shadow area), which are then discriminated from objects using texture features. The proposed method has two major parts: (1) creation of candidate shadow regions, and (2) classification of the candidate regions as objects or shadows.

In the first part, we use spectral features to select all the pixels which are likely to be shadows and form candidate shadow regions. Foreground edges are used to split regions containing both object and shadow pixels. In the second part, for each region we correlate its texture in the frame with its texture in the background reference. The texture correlation is done by comparing the gradient direction of pixels inside the region. Only pixels with significant gradient magnitude are used and the pixel-level comparisons are combined to form a region-level texture correlation. The regions with a significant texture correlation are classified as shadows. These two parts are detailed below, and the whole process is illustrated in Figure 2.2.

### 2.3.1 Creation of candidate shadow regions

The purpose of this first step is to create groups of pixels as large as possible which contain shadow pixels or object pixels, but not both. First, we use intensity and chromacity features to pre-select shadow pixels from the foreground mask, and group them into regions using connected component analysis. In most cases, the resulting regions will break separately into shadow or object regions. However, in some cases the same region will contain both

\(^1\)In this thesis we use the phrase “large candidate shadow regions” to indicate candidate shadow regions of non-arbitrary size which contain as many shadow pixels as possible.
Figure 2.2: Processing in the proposed method: (a) mask obtained after foreground extraction, marked in blue; (b) pre-selection of shadow pixels based on spectral features, marked in red; (c) internal foreground edges used to split the pixels into regions; (d) candidate shadow regions; (e) gradient direction of pixels in the frame with significant gradient magnitude; (f) gradient direction of pixels in the background reference with significant gradient magnitude; (g) gradient direction correlation for each candidate region (only the region in the left pane is above the threshold $\tau_c$); (h) final foreground object pixels marked in blue, omitting the shadow.
shadow and object pixels. We use the edges that occur between objects and shadows to split this regions. Below we explain these steps in detail.

**Candidate shadow pixels**

For a shadow pixel, we expect its intensity to decrease within a predictable range and its chromacity to be maintained, when comparing the input frame to the background reference. To obtain the features, we use the HSV colour space, which provides a natural separation of intensity and chromacity. Intensity is directly measured by the value (V) component, and chromacity is mainly measured by the hue (H) and saturation (S) components. For each pixel \( p \) in the foreground mask, we obtain \( F^V_p, F^S_p \) and \( F^V_p \), the three components of the pixel in the input frame, and \( B^H_p, B^S_p \) and \( B^V_p \), the three components of the pixel in the background reference. The pixel \( p \) is considered to be a candidate shadow if:

\[
\beta_1 \leq \frac{F^V_p}{B^V_p} \leq \beta_2 \quad \text{with} \quad \beta_1, \beta_2 \propto \frac{F^V_s}{B^V_s}, \tag{2.10}
\]
\[
\text{and} \quad \frac{F^S_p - B^S_p}{\tau_S} \leq \tau_S \quad \text{with} \quad \tau_S \propto \frac{1}{F^S}, \tag{2.11}
\]
\[
\text{and} \quad |F^H_p - B^H_p| \leq \tau_H \quad \text{with} \quad \tau_H \propto \frac{1}{F^S} \tag{2.12}
\]

where, \( \beta_1, \beta_2, \tau_S \) and \( \tau_H \) represent thresholds that are optimised empirically for high detection accuracy, and adapted according to the average attenuation of shadow pixels, \( \frac{F^V_s}{B^V_s} \), and the average saturation of the scene, \( \overline{F^S} \), in recent frames.

Our approach for selecting candidate shadow pixels based on spectral features is similar to the shadow detection approach in [CGPP03], with two key differences: (1) we tune the thresholds to ensure high detection accuracy, and (2) we use global scene information to adapt the thresholds.

Classifying all shadow pixels as shadows is important in this step because it will impose the upper bound of the final detection accuracy. At this stage it is common to have object pixels misclassified as shadows. The next section describes how we deal with this problem. Adapting the thresholds increases the effectiveness of the spectral features. In a scene with strong shadows, the intensity thresholds (\( \beta_1 \) and \( \beta_2 \)) should have smaller values. In a scene with
saturated colours, chromacity features are more reliable and the chromacity thresholds ($\tau_S$ and $\tau_H$) should be more strict (i.e., smaller values).

**Candidate shadow regions**

A mask is constructed with all the candidate shadow pixels selected in the previous step and the connected components are used as regions. Some regions may contain both object and shadow pixels. Since our final classification is made at the region level, this will necessarily introduce detection and/or discrimination errors. Therefore, it is necessary to split these regions so that they contain object pixels or shadow pixels, but not both.

Typically, there is an edge between foreground objects and their shadows. This edge can be used to split problematic regions. However, one of our goals is to generate large shadow regions and it is important to avoid breaking regions that already contain only shadows. For this reason, we break the regions using only foreground edges, that is, edges introduced by the foreground objects at each frame which do not exist in the background reference. We assume that shadow regions misclassified as foreground do not introduce new edges (other than their contours) to the scene.

The foreground edges are obtained by subtracting the edges in the background reference from the edges in the frame, both detected using Canny edge detection [Can86]. As the boundaries of strong shadows may introduce edges which are not present in the background reference, better results are likely to be obtained if outer edges are ignored from the foreground edges.

### 2.3.2 Classification of candidate shadow regions

At this stage, there are a number of candidate shadow regions which should contain only object or shadow pixels. For each region, we correlate the texture between the frame and the background reference. Since shadows preserve the underlying textures, shadow regions should have a high texture correlation.

Due to the variable size and shape of our candidate shadow regions, as well as for speed and simplicity, we generate a region level correlation value by comparing gradient directions at the pixel-level, similar to [JS02]. How-
ever, we restrict the correlation to pixels with a significant gradient magnitude because intensity derivatives (such as gradient direction) are more susceptible to noise [OH07]. Additionally, our equations take into account the circular nature of the gradient direction variable [Fis93]. The classification process is detailed in the following steps.

**Significant gradients**

For each candidate region, the gradient magnitude $|\nabla_p|$ and gradient direction $\theta_p$ at each pixel $p = (x, y)$ are calculated using:

\begin{align*}
|\nabla_p| &= \sqrt{\nabla_x^2 + \nabla_y^2} \quad (2.13) \\
\theta_p &= \arctan2\left(\frac{\nabla_y}{\nabla_x}\right) \quad (2.14)
\end{align*}

where $\nabla_y$ is the vertical gradient (difference in intensity between the pixel and the pixel in the next row), while $\nabla_x$ is the horizontal gradient. The function $\arctan2(\cdot)$ is a variant of $\arctan(\cdot)$ that returns an angle in the full angular range $[-\pi, \pi]$, allowing the gradient direction to be treated as a true circular variable [Fis93]. Only the pixels with $|\nabla_p|$ greater than a certain threshold $\tau_m$ are taken into account to avoid the effects of noise, which is stronger in the smooth regions of the frame.

**Gradient direction difference**

Since the gradient direction is a circular variable, the difference has to be calculated as an angular distance. For each pixel $p = (x, y)$ that was selected due to significant magnitude, the difference in gradient direction between the frame $F$ and the background reference $B$ is calculated using:

\begin{equation}
\Delta \theta_p = \arccos \frac{\nabla_x^F \nabla_x^B + \nabla_y^F \nabla_y^B}{\sqrt{(\nabla_x^F)^2 + (\nabla_y^F)^2}(\nabla_x^B)^2 + (\nabla_y^B)^2}} \quad (2.15)
\end{equation}
Chapter 2. Shadow Detection and Removal

Gradient direction correlation

The gradient direction correlation between the frame and the background is estimated using:

\[ c = \frac{\sum_{p=1}^{n} H(\tau_a - \Delta \theta_p)}{n} \]  

(2.16)

where \( n \) is the number of pixels selected in the candidate shadow region and \( H(\cdot) \) is the unit step function which, in this case, evaluates to 1 if the angular difference is less than or equal to the threshold \( \tau_a \), and 0 otherwise. In essence, \( c \) is the fraction of pixels in the region whose gradient direction is similar in both the frame and the background. If \( c \) is greater than threshold \( \tau_c \), the candidate region is considered a shadow region and it is removed from the foreground mask.

2.4 Comparative Evaluation

In this section, we present two sets of experiments to compare the performance of the four methods selected in Section 2.2 and the approach proposed in Section 2.3. The first set of experiments compares direct measures of shadow detection performance. We show both quantitative and qualitative results. The second set of experiments is an applied empirical study that shows the improvement in tracking performance after using each of the five compared methods. Below, we describe the sequences used for the experiments and the steps for creating the ground truth frames. In the following subsections, we explain each experiment in detail and present the results.

The sequences used in our experiments are summarised in Figure 2.3. We used a wide range of scenes with variations in the type and size of objects and shadows. The first six sequences were introduced in [PMTC03] and [MBZ08], and have been widely used for testing shadow detection performance. The last entry summarises 25 sequences from the CAVIAR dataset.

\[^2\]C++ source code for the five implemented methods can be obtained from http://arma.sourceforge.net/shadows/.


Table 2.3: Image sequences used in the comparative evaluation. Sequences are described in terms of: number and size of frames and number of examples with manually labelled shadows; scene type, surface and noise; type, size (relative to the frame size) and speed (in pixels/frame) of the objects; and size, strength and direction of the shadows. The main features or challenges presented by each sequences are highlighted using bold text. Note that the last column summarises the properties of 25 sequences from the CAVIAR dataset, all of which are different recordings of the same scene.

<table>
<thead>
<tr>
<th>Frames</th>
<th>Campus</th>
<th>Hallway</th>
<th>Highway 1</th>
<th>Highway 3</th>
<th>Lab</th>
<th>Room</th>
<th>CAVIAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>1179</td>
<td>1800</td>
<td>440</td>
<td>2227</td>
<td>887</td>
<td>300</td>
<td>1388 (×25)</td>
</tr>
<tr>
<td>Labelled</td>
<td>53</td>
<td>13</td>
<td>8</td>
<td>7</td>
<td>14</td>
<td>22</td>
<td>45 (×25)</td>
</tr>
<tr>
<td>Size</td>
<td>352 × 288</td>
<td>320 × 240</td>
<td>320 × 240</td>
<td>320 × 240</td>
<td>320 × 240</td>
<td>320 × 240</td>
<td>384 × 288</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scene</th>
<th>Type</th>
<th>Surface</th>
<th>Noise</th>
<th>Type</th>
<th>Surface</th>
<th>Noise</th>
<th>Type</th>
<th>Surface</th>
<th>Noise</th>
<th>Type</th>
<th>Surface</th>
<th>Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frames</td>
<td>outdoor</td>
<td>asphalt</td>
<td>high</td>
<td>indoor</td>
<td>asphalt</td>
<td>medium</td>
<td>outdoor</td>
<td>asphalt</td>
<td>medium</td>
<td>indoor</td>
<td>white</td>
<td>indoor</td>
</tr>
<tr>
<td>labelled</td>
<td>textured</td>
<td>asphalt</td>
<td>low</td>
<td>medium</td>
<td>carpet</td>
<td>medium</td>
<td>labelled</td>
<td>carpet</td>
<td>medium</td>
<td>reflective</td>
<td>reflective</td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>352 × 288</td>
<td>320 × 240</td>
<td>320 × 240</td>
<td>320 × 240</td>
<td>320 × 240</td>
<td>320 × 240</td>
<td>384 × 288</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Objects</th>
<th>Type</th>
<th>Size</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frames</td>
<td>vehicles/people</td>
<td>medium</td>
<td>5 − 10</td>
</tr>
<tr>
<td>labelled</td>
<td>people</td>
<td>variable</td>
<td>5 − 15</td>
</tr>
<tr>
<td>Size</td>
<td>large</td>
<td>30 − 35</td>
<td></td>
</tr>
<tr>
<td>Speed</td>
<td>small</td>
<td>10 − 15</td>
<td></td>
</tr>
<tr>
<td>large</td>
<td>medium</td>
<td>2 − 5</td>
<td></td>
</tr>
<tr>
<td>small</td>
<td>medium</td>
<td>1 − 4</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Shadows</th>
<th>Size</th>
<th>Strength</th>
<th>Direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frames</td>
<td>very large</td>
<td>weak</td>
<td>horizontal</td>
</tr>
<tr>
<td>labelled</td>
<td>medium</td>
<td>strong</td>
<td>multiple</td>
</tr>
<tr>
<td>Size</td>
<td>large</td>
<td>very strong</td>
<td>multiple</td>
</tr>
<tr>
<td>Strength</td>
<td>medium</td>
<td>weak</td>
<td>multiple</td>
</tr>
<tr>
<td>Direction</td>
<td>medium</td>
<td>vertical</td>
<td>multiple</td>
</tr>
</tbody>
</table>
Each sequence presents a different challenge for the shadow detection methods to test their robustness. The Campus sequence is a particularly noisy outdoor sequence where some of the shadows are extremely long. The Hallway sequence has a textured background and the size of the people changes significantly according to their distance to the camera. The Highway 1 and Highway 3 sequences show a traffic environment with two different lighting conditions and vehicle sizes. In particular, the Highway 3 has very small vehicles which could be misclassified as shadows. The Lab and Room indoor sequences show two laboratory rooms in two different perspectives and lighting conditions. From the CAVIAR dataset, we selected 25 sequences that show people walking in various patterns inside a shopping mall.

We decided to create an extended moving cast shadow detection dataset for two main reasons: (1) to have an increased number of sequences and labelled frames, and (2) to provide standard estimations of foreground and background so that this and future comparative evaluations can focus exclusively on shadow detection.

The labelling process is summarised in Figure 2.4. We first used the standard Gaussian mixture model (GMM) foreground extraction method from OpenCV 2.0 [BK08] to extract a foreground mask on each frame. The resulting mask should contain only object and shadow pixels. By superimposing the original frame on the foreground mask, pixels in the mask were manually marked as object (white) or cast shadow (grey).

Our extended dataset, which includes labelled foreground masks, original frames and the backgrounds estimated with the GMM method, is a further contribution of this thesis6. The original sequences can be used for learning-based methods which require all the intermediate frames.

2.4.1 Shadow detection performance

First, we measured the shadow detection performance of each method in every test sequence. We then gradually decreased the colour information of each sequence to test the dependency of each method on colour features. Fi-

---

6The ground truth masks can be obtained from http://arma.sourceforge.net/shadows/.
Chapter 2. Shadow Detection and Removal

Figure 2.4: Ground truth creation steps: (a) original frame; (b) foreground mask obtained with the standard Gaussian mixture model (GMM) foreground extraction method from OpenCV 2.0 [BK08]; (c) frame superimposed on foreground mask to facilitate labelling; (d) manually labelled mask with object pixels marked in white and shadow pixels in grey.

Finally, we show qualitative results for each sequence and summarise their observed behaviour. All the methods were faithfully implemented as described in Section 2.1. The thresholds for all methods were selected to obtain the best overall performance on the test sequences, using one setting for all sequences (i.e., no specific tuning for each sequence).

Quantitative results

To test the shadow detection performance of the five methods we used the two metrics proposed by Prati et al. [PMTC03], namely shadow detection rate ($\eta$) and shadow discrimination rate ($\xi$):

\[ \eta = \frac{TP_S}{TP_S + FN_S} \]  \hspace{1cm} (2.17)
\[ \xi = \frac{TP_F}{TP_F + FN_F} \]  \hspace{1cm} (2.18)

where $TP$ and $FN$ stand for true positive and false negative pixels with respect to either shadows ($S$) or foreground objects ($F$). The shadow detection rate is concerned with labelling the maximum number of cast shadow pixels as shadows. The shadow discrimination rate is concerned with maintaining the pixels that belong to the moving object as foreground. In this chapter we often use the average of the two rates as a single performance measure.
Table 2.2: Average frame processing time (in milliseconds) per sequence for various shadow detection methods. The first value shown for the texture-based method corresponds to the processing time using 48 Gabor kernels. This configuration was used in the previous experiments for maximum accuracy. The second value in brackets represents the time when using a reduced set of 16 Gabor kernels (for faster processing).

<table>
<thead>
<tr>
<th></th>
<th>Chromacity</th>
<th>Geometry</th>
<th>Physical</th>
<th>Textures</th>
<th>Proposed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Campus</td>
<td>8.72</td>
<td>9.44</td>
<td>10.00</td>
<td>156.46</td>
<td>(48.81)</td>
</tr>
<tr>
<td>Hallway</td>
<td>11.28</td>
<td>8.91</td>
<td>12.81</td>
<td>223.64</td>
<td>(77.37)</td>
</tr>
<tr>
<td>Highway 1</td>
<td>10.73</td>
<td>24.75</td>
<td>16.93</td>
<td>341.32</td>
<td>(116.28)</td>
</tr>
<tr>
<td>Highway 3</td>
<td>6.82</td>
<td>6.49</td>
<td>7.15</td>
<td>120.36</td>
<td>(37.07)</td>
</tr>
<tr>
<td>Lab</td>
<td>8.95</td>
<td>17.68</td>
<td>15.34</td>
<td>253.82</td>
<td>(82.48)</td>
</tr>
<tr>
<td>Room</td>
<td>7.14</td>
<td>8.41</td>
<td>8.51</td>
<td>144.87</td>
<td>(47.07)</td>
</tr>
<tr>
<td>Caviar</td>
<td>10.82</td>
<td>13.94</td>
<td>14.08</td>
<td>243.07</td>
<td>(82.98)</td>
</tr>
<tr>
<td>Average</td>
<td><strong>9.21</strong></td>
<td><strong>12.80</strong></td>
<td><strong>12.12</strong></td>
<td><strong>211.93</strong></td>
<td>(70.30)</td>
</tr>
</tbody>
</table>

Figure 2.5 shows the average shadow detection and discrimination rates on each test sequence. Each bar represents the average of the detection and discrimination rates on each sequence, while the individual detection and discrimination rates are indicated by a square and a circle, respectively. In all cases, the proposed approach performs considerably better than all the others, obtaining high values for both the detection and discrimination rates in all sequences. We discuss these sequence-related results more deeply in the qualitative results section.

The average processing time per frame of each shadow detection method is shown in Table 2.2. Times are shown in milliseconds, obtained on a 32-bit Intel CPU running at 2.6 GHz. Four of the methods, excluding the texture-based, have the same asymptotic complexity of constant operations per pixel. The chromacity-based method was the quickest to implement and run. The geometry and physical-based methods need more operations for calculating the central moments and updating the shadow models, respectively. The proposed method requires extra steps to generate the candidate shadow regions and calculate the gradients for each pixel, and is the slowest of these four. The amount of operations per pixel in the texture-based method depends on the size and number of kernels used for correlating the textures. This texture-based method has considerably higher computational load than the rest, even when the number of kernels and their size are minimised.
Figure 2.5: Comparison of shadow detection results by sequence. The performance measure is the average between the shadow detection and discrimination rates. Each bar represents the average performance with a vertical segment whose length indicates the average difference between the detection rate (marked with a square) and the discrimination rate (marked with a circle).
The proposed method was designed firstly to increase the effectiveness of texture-based features by selecting large regions, and secondly to use spectral features to improve results in the absence of significant textures. However, it is hard to conclude how well the method achieves these two goals by simply observing the shadow detection performance on various sequences. For this reason, we tested the performance of each method while gradually decreasing the colour information of each sequence until reaching greyscale frames. This experiment is used for two purposes: (1) to make a fair comparison between the proposed and texture-based methods when colour information is lost and thus determine which one is using texture information more effectively; and (2) to observe how the colour information is used when available, and how dependent are the methods on colour features.

Figure 2.6 shows the average performance of each method across all sequences as the colour information is manually decreased. The performance results indicate the average between shadow detection and discrimination rates. The desaturation rate indicates the reduction in colour information used to modify the frames as follows:

\[
F_{\text{desat}} = (1 - \lambda)F_{\text{orig}} + \lambda F_{\text{grey}}
\]

where \( \lambda \) is the desaturation rate, \( F_{\text{orig}} \) is the original frame and \( F_{\text{grey}} \) is the frame converted to greyscale. In other words, when the desaturation is 0% the original frames are used, when it is 100% the frames are converted to greyscale, and the rest are interpolations between the two, created by gradual blending to avoid colour distortions. Note that alternative ways of reducing the colour information can also be used. For example, it is possible to represent the images in HSV colour space and gradually decrease the saturation (\( S \)) component to remove colour.

Several things can be concluded from this figure. First, as expected, methods which use colour features perform better when all the colour information is available, and their performance drops as the frames are desaturated. Second, the decrease in performance of the physical method is lower since it is locally adaptive; the performance of the proposed method also decreases, but with a magnitude lower than that of the chromacity method, since it is glob-
Figure 2.6: Shadow detection performance when the colour information is gradually reduced. Each line shows the average performance of a shadow detection method ranging from the original sequences (0% desaturated) to the greyscale-converted sequences (100% desaturated).

ally adaptive. Third, without colours the proposed method depends on textures while performing considerably better than the texture-based method. Last, the proposed method performs better than all others regardless of the degree of desaturation; while it uses colour information to improve performance, without such information its performance is nevertheless still good.

Qualitative results

We show the qualitative results in Figure 2.7. The first column shows an example frame for each sequence, the second column shows the expected results, where foreground pixels are marked in blue and shadow pixels are marked in green. The remaining columns show the observed results with each shadow detection method. For symmetry, we show results for all sequences and all methods. However, it is important to note that the geometry-based method was designed for pedestrian shadow detection and, apart from mentioning the limitation, we do not take into account the sequences with objects other than pedestrians when comparing to this particular method. In
Figure 2.7: Qualitative shadow detection results. The first column shows an example frame for each sequence. The second column shows the expected results where object pixels are marked in blue and shadow pixels are marked in green. The remaining columns show the observed results for each method. The geometry-based method works well for long shadows with a well defined orientation (Campus example) but does not work well for spread shadows (Lab example). Both the chromacity-based and physical methods are affected by pixel-level noise, and fail when objects are darker and have similar colours to the background (Campus example), although the physical method adapts better to the different scenes. The texture-based method fails for pixels located in non-textured neighbourhoods (Highway 1 example). The proposed method works reasonably well in all cases.

In general, as was also shown in the qualitative results, the proposed method performs better in all the examples.

Specific observations can be also done for each method. The geometry-based method only works when each shadow has a unique orientation which differs to the object’s orientation, as happens in the Campus example, but fails when the shadows have the same orientations as the objects or when shadows have multiple directions as in the Lab example. The chromacity-based
method is affected by pixel-level noise, and fails when objects are darker and have similar colours to the background as in the Campus example. Although the physical method also uses chromacity features, it has the ability to adapt to each scene and it is less likely to fail when the chromacity-based method fails. The texture-based method works well in scenes with textured backgrounds but fails for pixels located in non-textured neighbourhoods (as can be seen in the Highway 1 example). Finally, the proposed method works well in most cases, although it can distort the object contours as in the Lab example.

The results from both the quantitative experiments and qualitative observations are summarised in Table 2.3. We assign scores to each method according to several criteria. The geometry-based method has strong assumptions regarding the object and shadow shape, but when these assumptions are met the method works well independently on the quality of spectral and texture features. The chromacity-based method is simple and fast, and as Prati et al. [PMTC03] concluded, its few assumptions lead it to work reasonably well in most scenes. However, it has a strong trade-off between shadow detection and discrimination. The physical method reduces the limitations of the chromacity-based method, provided there are sufficient examples to learn and adapt to the appearance of moving cast shadow in the background. However, as the chromacity-based method, it is sensitive to pixel-level noise and scenes with low saturated colours. The texture-based method is robust to various illumination conditions and is easy to implement, but it requires the background to be textured and needs a costly texture correlation operation per pixel. The proposed method is not sensitive to pixel-level noise, and it is independent to the type of shadows, objects and scenes. It presents the best results at the cost of additional computational load.

2.4.2 Effect on tracking performance

Given that several methods perform relatively well (i.e., detection and discrimination rates above 75%), will pursuing higher rates translate into improved object detection and tracking? Although measuring shadow detection accuracy pixel-by-pixel is an objective measure, applications rarely have the need of explicitly detecting shadow pixels. Rather, shadow detection meth-
Table 2.3: Summary of the comparative evaluation. The shadow detection methods are rated as low, medium or high according to seven criteria. For the first five criteria, high indicates that the method performed well in all cases. A method with low scene, object or shadow independence, only works well for a narrow set of scene, object or shadow types. Low penumbra detection means that a method is not appropriate for detecting shadow borders. Low robustness to noise means that shadow detection performance is significantly affected by scene noise. For the last two criteria (i.e., 6 and 7) lower means better. A high detection/discrimination trade-off means that tuning a method to increase one of the rates will significantly decrease the other. In terms of computational load, the methods are ranked by their average processing time per frame.

ods are typically used in tracking applications to clean the detection results and improve tracking performance. It is important to prove that increasing the detection rate and/or discrimination rate of shadow pixels will result in better tracking results. For this reason, we performed a second set of experiments to measure tracking performance after applying each method to remove the shadows in the foreground masks (i.e., setting the shadow pixels to zero).

Using various tracking algorithms is important since they may use the shadow-removed foreground masks differently. For instance, the masks can be used prior to tracking (e.g., for initialisation of particle filters [ZCM04]) or during tracking (e.g., for updating appearance models in blob matching [FV06, ZA06]).

We used five tracking algorithms implemented in the video surveillance module of OpenCV 2.0 [BK08]: (i) blob matching or connected component tracking (CC), (ii) mean-shift tracking (MS), (iii) mean-shift tracking with foreground feedback (MSFG), (iv) particle filtering (PF), and (v) blob matching/particle filter hybrid (CCPF). The foreground masks are used while tracking in CC, prior to tracking in MS and PF, and both while and prior to tracking in MSFG and CCPF.
For the experiments, we used the tracking ground truth data available for the 50 sequences in the second set of the CAVIAR dataset. We performed 30 tracking evaluations by combining six shadow removal options (no shadow removal, geometry-based method, chromacity-based method, physical method, texture-based method and proposed method) with the five tracking algorithms. The tracking performance was measured with the two metrics proposed by Bernardin and Stiefelhagen [BS08], namely multiple object tracking accuracy (MOTA) and multiple object tracking precision (MOTP):

\[
\text{MOTA} = 1 - \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t} \quad (2.20)
\]

\[
\text{MOTP} = \frac{\sum_{i,t} (d^i_t)}{\sum_t c_t} \quad (2.21)
\]

Briefly, MOTA is an inverse measure of the number of missed objects \(m_t\), false positives \(fp_t\) and mismatches \(mme_t\). The higher the MOTA, the better. MOTP measures the average pixel distance \(d^i_t\) between the ground-truth locations of objects and their locations according to a tracking algorithm. The lower the MOTP, the better. Ground truth objects and hypotheses are matched using the Hungarian algorithm [Mun57].

The tracking results are presented in Figure 2.8. Each bar represents the performance result of a particular tracking algorithm after removing the shadows detected by one of the shadow detection methods, averaged for the 50 test sequences. In all cases, the proposed method results in better tracking performance, with considerable improvements over the next best method (texture-based).

Three other things can be observed in these results. First, in most cases, tracking performance is significantly improved with shadow removal regardless of the tracking algorithm. In some cases, the geometry-based method led to a decrease in tracking performance. Recall that the geometry method relies on the assumption that objects and shadows have different orientations, which is not met in the CAVIAR dataset. Second, the improvement in tracking performance by using a better shadow remover (e.g., proposed method instead of the chromacity-based method) is comparable to the im-
Figure 2.8: Effect of shadow removal methods on: (a) multiple object tracking accuracy (MOTA), where taller bars indicate better accuracy; and (b) multiple object tracking precision (MOTP), where shorter bars indicate better precision. Results are grouped by tracking algorithm: blob matching or connected component tracking (CC), two mean shift trackers (MS and MSFG), particle filter (PF) and hybrid tracking (CCPF).
improvement by using a better tracking algorithm (e.g., particle filter instead of blob matching). Last, improving shadow detection performance (as shown for the CAVIAR dataset in Figure 2.5) leads to a proportional improvement in tracking performance. Therefore, regardless of the tracking algorithm, it is worth pursuing better shadow detection methods to obtain more accurate tracking results.

2.5 Main Findings and Future Directions

In this chapter we presented a review of shadow detection methods published during the last decade, proposed a new shadow detection method and compared it with a selection of previous methods. The review and comparative evaluation follows the work of Prati et al. in 2003 [PMTC03] but with recent publications, a more comprehensive set of test sequences and more detailed experiments.

In the review the shadow detection methods are classified in a feature-based taxonomy. Methods that use mainly spectral features are classified as either chromacity-based or physical methods. Methods that use mainly spatial features are classified as either geometry-based or texture-based methods. Four of the reviewed methods, one from each category, and a proposed shadow detection approach are tested in an extensive qualitative and quantitative comparative evaluation. Additionally, we observed the effect of low saturation on shadow detection performance and evaluated the practical link between shadow detection and tracking performance.

The quantitative and qualitative comparative results can serve as a guide (for both practitioners and researchers) to select the best method for a specific purpose. All shadow detection approaches make different contributions and all have individual strengths and weaknesses. Out of the selected methods, the geometry-based technique has strict assumptions and is not generalisable to various environments, but it is a straightforward choice when the objects of interest are easy to model and their shadows have different orientation. The chromacity-based method is the fastest to implement and run, but it is sensitive to noise and less effective in low saturated scenes. The physical method improves upon the accuracy of the chromacity method by adapting
to local shadow models, but fails when the spectral properties of the objects are similar to that of the background. The texture-based method is especially robust for pixels whose neighbourhood is textured, but may take longer to implement and is the most computationally expensive. The proposed method produces the most accurate results, but has a significant computational load due to its multiple processing steps.

In addition to measuring traditional shadow detection performance via detection and discrimination rates, we proposed to use tracking results as an unbiased way to measure the usefulness of shadow removal. Using tracking results is also useful when there is no shadow ground truth but information on object locations is available.

Traditionally, simple and fast shadow detection and removal methods have been favoured in computer vision applications such as tracking systems [FS10, LX06, MP08, WS07a]. It is hence pertinent to note that when the shadow detection performance is relatively poor (e.g., < 60% for the geometry-based technique on the CAVIAR dataset), shadow removal can in fact lead to tracking performance which is worse than not using shadow removal. In contrast, more elaborate shadow detection algorithms lead to considerably better tracking performance, regardless of the tracking algorithm.

A logical future direction is to use extra features in the existing methods, as all the currently used features largely provide independent contributions. For instance, geometry and temporal features can be added to the physical or texture-based approaches. Alternatively, physical or texture features can be used to pre-select candidate shadow pixels and feed them to geometry-based methods for shadow remodelling. Lastly, even if there are considerable differences in the computational load across the various shadow detection methods, all of them can be optimised to meet real-time requirements. In particular, all the compared approaches are parallelisable at either the pixel or region level.

Finally, the proposed shadow detection approach could be extended in several ways. First, a physical method, such as [HC09], tuned for high detection rate can be used for a more accurate selection of candidate shadow pixels. Second, multi-scale processing [OH07] can be used to calculate gradient directions of blocks using multiple resolutions instead of single pixels. This
can be used to filter out the effect of unreliable gradients that may arise from
pixel-level noise, and to preserve gradients that may disappear at the pixel
level when attenuated by shadows. Third, a more robust procedure for cal-
culating the region level texture correlation can be used, such as comparing
histograms of oriented gradients [DT05]. This would be particularly useful
in the presence of a shaking camera where pixel-based gradient correlation
may fail.
Chapter 3

Pedestrian Detection

Automatic surveillance systems often require locating objects of interests in every frame or when they first enter the scene. The nature of the objects is application dependent; for example, they may be whole entities such as humans, animals or vehicles, or they may be sections such as faces, hands or license plates. Depending on the application, the system may simply stop after the objects have been localised (e.g., for tagging or image segmentation) or it may feed the information to subsequent modules for further processing (e.g., for tracking, face recognition or number plate recognition). Figure 3.1 illustrates some applications where object detection is required.

In this chapter, we first review object detection methods with emphasis on pedestrian detection. We then propose a new algorithm and test it on two challenging datasets. The review starts with an overview of object detection approaches and then categorises the methods which are relevant to pedestrian detection. From the reviewed methods, of particular relevance is the

![Figure 3.1: Example applications that use object detection. From left to right: tracking for sports [XALL11], face recognition [LCB+08], and number plate recognition [CdCR09].]
work presented by Tuzel et al. [TPM08]. In their work, the authors use covariance descriptors for feature representation. Covariance matrices do not lie in a vector space, and it has been shown that taking this non-linear nature of the data into account leads to improved learning [Lui12a]. We present a brief introduction to the concept of manifolds, in particular to $\text{Sym}^+_d$, a Riemannian manifold which describes the space of positive definite symmetric matrices, to which covariance matrices belong.

Performing classification directly on a manifold is rather complicated. The typical approach is to first map the manifold data into Euclidean space so traditional machine learning approaches can be used. Tuzel et al. [TPM08] used the concept of tangent spaces, which are Euclidean projections of the manifold that locally preserve the structure of the data. More accurate mapping approaches have been proposed [LZ08], but these tend to be computationally expensive and can become impractical for object detection, where thousands of sample images are used for training and thousands of windows have to be scanned inside each image in search of the objects. We propose a new mapping model which combines multiple tangent spaces for improved data representation. We describe this model, denoted $k$-tangent spaces, and embed it in a cascaded boosting approach for pedestrian detection. We demonstrate empirically that the proposed model leads to markedly improved pedestrian detection performance. Although slower than using a single tangent space, the speed of the proposed method is still practical for object detection.

We continue the chapter as follows. In Section 3.1 we present a literature review of object detection methods and categorise the most relevant approaches for pedestrian detection according to their selection of features. In Section 3.2 we introduce Riemannian manifolds and give an overview of classification approaches on manifold data. The proposed pedestrian detection method is described in Section 3.3. Results from pedestrian detection experiments on two challenging datasets are given in Section 3.4. The main findings and possible future directions are summarised in Section 3.5.

The work in this chapter has been published in [SSHL12].
Chapter 3. Pedestrian Detection

3.1 Review

Yilmaz et al. [YJS06] classify object detection methods into four categories: (i) point detectors, (ii) image segmentation, (iii) background subtraction, and (iv) supervised learning. Point detectors are used to find interest points in images which have salient textures in their neighbourhood (e.g., [HS88, Low04, Mav10]). Image segmentation is used to segment images into perceptually similar regions (e.g., [CKS95, CM99, SM00]). The main problem with these two approaches is that interest points or similar regions in images do not necessarily belong to the objects of interest.

Background subtraction is the process of building a model of the scene and then detecting deviations from the model for each incoming frame (e.g., [SG00, LHGT03, RSSL10]). Exclusively using the foreground for object detection has several limitations. First, it is usually limited to video as, in most cases, training the background model requires an image sequence. Second, it is assumed that a reliable model of the background can be obtained, which can be challenging in the presence of moving cameras or non-stationary background objects such as waving trees. Third, elements in the foreground moving next to each other are detected as single objects. Fourth, not all the elements that move are considered objects of interest in a given application. These last two limitations normally imply implementing extra steps to segment foreground elements containing multiple objects and to classify objects of interest among other moving objects, respectively.

Supervised learning uses a large set of example objects of interest to train a model using machine learning techniques. Then, a sliding-window approach is normally used to find regions in the images whose representations have a high likelihood of belonging to the trained model. We are more interested in methods belonging to this category, commonly referred to as sliding-window approaches, as they have been the favoured choice for pedestrian detection [DWSP12].

Recent pedestrian detection surveys show that despite significant progress, performance still has much room for improvement [WMSS10, DWSP12]. Sliding-window based pedestrian detection has two main components: feature representation and classification. Based on the material reviewed, pa-
Chapter 3. Pedestrian Detection

![Figure 3.2: Examples of horizontal (a), vertical (b), line (c) and diagonal (d) Haar wavelets. The Haar-based features are evaluated by subtracting the intensity sum of the pixels in the dark grey region(s) from the intensity sum of the pixels in the light grey region(s).](image)

Persons tend to be mainly focused on proposing a novel feature representation and then choosing a classification approach which is more compatible with the features (sometimes the choice of feature representation and extraction will limit the option of applicable machine learning techniques), or the one which leads to better detection results.

Below, we first list the most common options of feature representation and use it to categorise related pedestrian detection methods. Then we summarise the main classification options used in conjunction with these features for pedestrian detection.

### 3.1.1 Features

We categorise related pedestrian detection methods according to five common feature representations: (i) intensity features, (ii) gradient features, (iii) motion features, (iv) shape features, and (v) mixed features.

**Intensity features**

One of the first approaches was the work of Papageorgiou and Poggio [PP00], where features are obtained using Haar wavelets. Haar-based features are quantisation of intensity differences at various locations, scales and orientations. The example wavelets shown in Figure 3.2 account for horizontal, vertical, line and diagonal intensity differences.

Viola and Jones [VJ01] built upon that approach to include a method of fast feature calculation based on integral images, a cascaded classification
Chapter 3. Pedestrian Detection

Figure 3.3: Constructing a histogram of gradient orientations [FR95]. (a) First, the region of interest containing the gesture is defined. (b) Second, at each pixel, the vertical and horizontal intensity gradients are used to calculate gradient directions. (c) Third, the histogram of gradient orientation is constructed taking into account all the pixel observations inside the region.

Gradient features

Large gains in pedestrian detection came with the adoption of gradient-based features [DWSP12], in particular the use of oriented gradients. Freeman and Roth [FR95] used orientation gradients for hand gesture recognition. Given a sample image containing a gesture, vertical and horizontal intensity gradients at each pixel are extracted to calculate gradient directions. Then a histogram is constructed to represent and match gestures to each other. These steps are shown in Figure 3.3. Similarly, Lowe [Low04] proposed a sparse scale-invariant feature transform (SIFT) to match various views of objects or scenes. SIFT is basically a sparse version of oriented gradients where the histograms are obtained and normalised locally (i.e., inside regions). The sparsity is achieved by using only subsets of the image, located with an interest-point detector.

Dalal and Triggs [DT05] popularised the use of histograms of oriented
Chapter 3. Pedestrian Detection

Figure 3.4: HOG grid normalisation. The input image (a) is divided into an uniform grid (b). For each cell in the grid a histogram of oriented gradients is constructed and locally normalised with the histogram calculated from a larger sliding block (c). The normalised cell histograms are concatenated to form the final representation.

gradients (HOG) for pedestrian detection. Their work evaluates normalised local histograms of image gradient orientations in a dense grid. The extraction and normalisation process is illustrated in Figure 3.4. The intuition behind the grid approach is that local object appearance and shape can be represented by the distribution of gradient orientations independently of their specific location. First, the input image is evenly divided into small cells. For each cell, a histogram is constructed from the gradient orientations at each pixel. A second set of histograms is calculated over larger regions, referred to as blocks by the authors. The block histograms are used to contrast-normalise their respective cell histograms for better illumination invariance. The normalised histograms are concatenated to form the final representation.

This procedure was shown to be successful for pedestrian detection. Since then, most pedestrian detection approaches use HOG in some form and these have been considered state-of-the-art [EG09, DWSP12]. Similar to the cascaded extension on Haar-based features proposed by Viola and Jones [VJ01], Zhu et al. [ZYCA06] extended the HOG approach for fast feature calculation
using integral histograms, proposed a cascaded version for fast detection and included automatic feature selection for optimised classification using adaptive boosting.

**Motion features**

Most sliding-window techniques are designed to work on still images. However, since many applications involve detecting pedestrians in videos, some authors propose incorporating temporal information to capture human motion. For instance, Viola *et al.* [VJS05] extended Haar wavelet representations and local shape filters to incorporate changes in intensity over time.

Although most of these approaches work well for stationary cameras, they are not designed to handle the motion of the background caused by non-static camera setups. Dalal *et al.* [DTS06] modelled motion statistics based on optical flow features to compensate for uniform camera movement. Optical flow captures the apparent displacement of every pixel over time. They collect features such as horizontal and vertical flow, as well as flow direction and magnitude, into histograms of differential optical flow (which other authors often term HOF).

**Shape features**

Other approaches have used spatial information for pedestrian detection. These approaches typically incorporate the spatial configuration of salient regions. For instance, Gavrila and Philomin [GP99] used the Hausdorff distance transform and a template tree to capture salient image regions and compare them to a hierarchy of shape templates.

More recently, Wu and Nevatia [WN05] manually designed a large set of short lines and curve segments, known as edgelets, which are used to locally represent edge-like shapes. They combined these edgelets to learn body part detectors as well as full body detectors. In a similar approach, Sabzmeydani and Mori [SM07] generate shapelets, which are descriptors learned from low-level oriented gradient responses. These shapelets are then combined to form pedestrian detectors.

---

1By definition a pedestrian is a person travelling on foot, which implies movement under the assumption of a fixed camera.
Chapter 3. Pedestrian Detection

Figure 3.5: Obtaining a covariance descriptor [TPM08]. A region $R$ is defined inside the input image. For each pixel $i$ in $R$ a number of features such as location, intensity and gradients are extracted to form the feature vector $z_i$. The feature vectors are then used to compute the covariance matrix $C_R$.

**Mixed features**

According to recent surveys [EG09, DWSP12], no single feature has been shown to outperform HOG. However, they note that improved results can be obtained by combining features in a single representation. Examples include: combining colour self-similarity (CSS) with HOF [WMSS10]; combining HOG, edgelets and covariance features [WN08]; and combining local binary patterns (LBP [OPM02]) with HOG [WHY09].

Instead of representing features as normalised histograms, Tuzel et al. [TPM08] represent features using covariance descriptors. To represent a given image region, a number of features is calculated for every pixel inside that region. For each pixel, a vector is constructed where each element in the vector is a feature extracted at the image location. Although the selection of features may vary, in the mentioned work a combination of gradients and intensity derivatives was used. The covariance matrix obtained from all the feature vectors is used to represent the region. This process is illustrated in Figure 3.5.

Covariance descriptors have become increasingly popular for feature representation. This is due to their relative low sensitivity to translations and scale variations, as well as being quick to compute through integral image representations [GPC11, PSZ08, TPM06]. It is also important to note that covariance matrices do not lie on a vector space. Rather, they lie on the Riemannian manifold of symmetric positive definite matrices $\text{Sym}_d^+$. Tuzel et al. [TPM08] proposed a boosting framework for learning manifold data. Their
method achieved higher performance compared to using HOG descriptors\textsuperscript{2}, possibly attributed to both the robustness of covariance descriptors as features and exploiting their non-linear nature for improved classification.

The method we propose in Section 3.3 is an extension of the work by Tuzel et al. [TPM08], where we use the same features and aim to improve classification with an improved representation of the manifold data. Before describing our classification approach, it is necessary to understand the basic concepts of manifold geometry. In Section 3.2, we introduce these concepts focused on $\text{Sym}_d^+$, the Riemannian manifold of $d \times d$ dimensional symmetric positive definite matrices, where covariance descriptors belong.

### 3.1.2 Classification

Once an image region has been converted to feature representation, the detection task becomes a binary classification problem. The most common classification approaches used for pedestrian detection are artificial neural networks (ANN) [RBK98], support vector machines (SVM) [POP98], and variants of boosting [VJ01].

**Artificial neural networks**

Artificial neural networks, often just called neural networks, are mathematical models inspired by biological neural networks. Neural networks are formed by nodes connected by links. Each link has an associated weight determining the strength of the connection. The nodes are typically organised in layers.

The simple multi-layer perceptron (MLP) architecture consists of three or more layers of nodes: one input layer, one or more hidden layers and one output layer. Nodes in the input layer, depending on the connections and weights, can activate nodes in the hidden layers. In turn, nodes in the hidden layers can activate nodes in the output layer where the classification decision is produced. Learning is done by adjusting the network connections.

\textsuperscript{2}Although the pedestrian detection method based on covariance descriptors has been mentioned in recent surveys [DWSP12, EG09], it has not been included in their comparative evaluation. Possible reasons for this are the recency of the publication and the difficulty of implementing manifold-based classification.
and their weights. Although they can be slow to train, neural networks tend to be fast to run [BK08].

**Support vector machines**

Support vector machines typically project the data into a higher-dimensional space where a distance function between any pair of data points is defined. The intuition is that at higher dimensions the data points are more likely to be linearly separable. Then, training an SVM involves searching for a maximum margin hyperplane that best separates the training data. Support vector machines are among the classifiers with higher discriminative power; they usually perform the best when the amount of data is limited, and are only outperformed by approaches such as boosting with large training datasets [Vap95].

**Boosting**

Boosting is a machine learning meta-algorithm to produce a final strong classifier by combining a series of simple weak classifiers. A key aspect of boosting is that weights are assigned to the training samples. The strong classifier is trained through several iterations. Each iteration trains a weak classifier and modifies the weights of the training samples, giving more emphasis to samples where errors were made. This iterative process continues until the total classification error over the dataset falls below a predefined threshold.

Boosting only requires the weak classifiers to be slightly better than random guessing. The weak classifiers are combined to produce a strong classifier that is well correlated with the true classification [FS95]. This is more likely to happen with large amounts of training data. For detection, boosting approaches have additional advantages. First, if feature representation differs between individual weak learners, boosting will also perform automatic feature extraction. Second, weak classifiers can be grouped into layers to form a rejection cascade [VJ01], where the first layers are designed to quickly reject negative samples. This leads to faster detection as most examples in sliding-window approaches do not contain objects of interest.
Figure 3.6: The tangent space $T_X$ is a Euclidean representation of the differentiable manifold $M$. The tangent space is defined by the tangent vectors ($\Delta$) of the curves of the form ($XY$) in the manifold.

### 3.2 Manifolds

Although the concept of non-Euclidean geometry has been critical in physics over the last two centuries\(^3\), only now it is becoming popular in other areas such as computer vision. Intuitively, we can think of a manifold as a continuous surface lying in a higher-dimensional Euclidean space ($\mathbb{R}^n$) [KSM99]. Although manifolds have a wide variety of sizes and shapes, they all have in common the fact that locally they are relatively flat and can be approximated by Euclidean space. Take the surface of the Earth for example; while it is in reality a spheroid, from our perspective (i.e., locally), it looks flat\(^4\).

Differentiable manifolds locally resemble a linear space where it is possible to use calculus. Each point $X$ in an $n$-differentiable manifold $M$ has a tangent space $T_X$, an $n$-dimensional Euclidean space. The point defining the tangent space is termed the tangent pole. The tangent space is defined by the tangent vectors of the curves that pass through the tangent pole. Figure 3.6 shows $T_X$, the tangent space of manifold $M$ at point $X$. It also illustrates how the tangent vector $\Delta$ is defined by the curve between $X$ and $Y$.

For manifolds to be useful in computer vision applications, it should be

\(^3\)As an interesting fact, the discovery of non-Euclidean geometry was considered to be among the major scientific breakthroughs in the 2012 National Geographic special publication “100 scientific discoveries that changed the world” [Bro12].

\(^4\)Although we now know that the Earth is not truly flat, most modern maps are still flat due to the convenience and simplicity of local Euclidean approximations to non-Euclidean manifolds.
possible to do simple operations such as computing distances and angles. This can be achieved using Riemannian metrics. A metric at point $X$ is a map $\langle u, v \rangle_X$ that smoothly associates pairs of vectors $(u,v)$ in the tangent space $T_X$ to distance values. The metric is said to be Riemannian if it is symmetric (i.e., $\langle u, v \rangle_X = \langle v, u \rangle_X$) and positive definite (i.e., $\langle u, u \rangle_X \geq 0$) for all points in the manifold. A Riemannian manifold is a differentiable manifold where a Riemannian metric can be defined [Jos05].

3.2.1 Space of covariance matrices

We are particularly interested in the space of $d \times d$ dimensional symmetric positive definite matrices $Sym_d^+$, where covariance matrices belong. $Sym_d^+$ can be formulated as a Riemannian manifold using the metric defined in [PFA06]:

$$\langle u, v \rangle_X = \text{trace} \left( X^{-\frac{1}{2}} u X^{-1} v X^{-\frac{1}{2}} \right) \quad (3.1)$$

Below we describe the operations in $Sym_d^+$ which will be used in the rest of this chapter. Two important operators, namely the exponential map $\exp_X(\cdot)$ and the logarithm map $\log_X(\cdot) = \exp_X^{-1}(\cdot)$, allow switching between the manifold and the tangent space at $X$. The exponential operator maps a tangent vector to a point on the manifold. Inversely, the logarithm operator maps a point on the manifold to the tangent space. The exponential and logarithm maps vary as point $X$ moves along the manifold. For $Sym_d^+$, these operators are defined as follows [PFA06]:

$$\exp_X(y) = X^{\frac{1}{2}} \exp \left( X^{-\frac{1}{2}} y X^{-\frac{1}{2}} \right) X^{\frac{1}{2}} \quad (3.2)$$

$$\log_X(Y) = X^{\frac{1}{2}} \log \left( X^{-\frac{1}{2}} Y X^{-\frac{1}{2}} \right) X^{\frac{1}{2}} \quad (3.3)$$

where $\exp(\cdot)$ and $\log(\cdot)$ are matrix exponential and logarithm operators, respectively. For symmetric positive definite matrices they can be computed through singular value decomposition (SVD). Let $X = U \Sigma U^T$ be the SVD of the symmetric matrix $X$, then:
Chapter 3. Pedestrian Detection

\[ \exp (X) = U \exp (\Sigma) U^T \]  \hspace{1cm} (3.4)
\[ \log (X) = U \log (\Sigma) U^T \]  \hspace{1cm} (3.5)

where \( \exp (\Sigma) \) and \( \log (\Sigma) \) are two diagonal matrices where the diagonal elements are respectively equivalent to the exponential or logarithms of the diagonal elements of matrix \( \Sigma \).

It is important to have a distance measure on the manifold. The minimum length curve connecting two points on the manifold is called the geodesic \([TVSC11]\). The geodesic distance \( d(X, Y) \) is the length of this curve. Using the tangent space defined at \( X \), the geodesic distance between \( X \) and \( Y \) can be calculated via:

\[ d^2(X, Y) = \text{trace} \left[ \log^2 \left( X^{-\frac{1}{2}} Y X^{-\frac{1}{2}} \right) \right] \]  \hspace{1cm} (3.6)

It is also important to calculate a central tendency measure for a set of points on the manifold. Given a set \( \{X_i\}_{i=1}^N \) of points on manifold \( M \), the Karcher mean \([Kar77]\) is the point on \( M \) that minimises the sum of distances:

\[ \mu = \arg \min_{X \in M} \sum_{i=1}^N d^2(X_i, X) \]  \hspace{1cm} (3.7)

For \( \text{Sym}_d^+ \), the Karcher mean can be iteratively found using:

\[ \mu^{[t+1]} = \exp_{\mu^{[t]}} \left[ N^{-1} \sum_{i=1}^N \log_{\mu^{[t]}} (X_i) \right] \]  \hspace{1cm} (3.8)

3.2.2 Classification on manifolds

While the geometry of Riemannian manifolds has been extensively studied, with well defined metrics and operations for \( \text{Sym}_d^+ \), classification directly on manifolds (e.g., dividing a manifold into two or more distinct parts) is still a challenging problem \([HSWL12, SHSL12]\). Consider the case of linear binary classification on the two-dimensional Euclidean space \( \mathbb{R}^2 \). A point and a direction vector in \( \mathbb{R}^2 \) define a line that separates \( \mathbb{R}^2 \) into two. An equivalent curve can be defined in a two-dimensional differentiable manifold using a point in the manifold and a tangent vector in the tangent space of that point.
Chapter 3. Pedestrian Detection

Figure 3.7: The geodesic curves on the 2-torus never divide the manifold into two. Rather, they produce a single surface and therefore are not suitable for linear classification.

Using such curve for classification is a complicated notion even for simple manifolds. As shown in Figure 3.7 the curves on the 2-torus never divide the manifold into two.

As a consequence, learning manifold data usually involves first mapping the points into vector space where traditional machine learning techniques can be applied. Current methods vary in terms of the mapping approach and how it is embedded in the classification algorithms. The simplest approach is to map the manifold data into an arbitrary tangent space, usually $T_I$ where $I$ is the identity matrix [GIK10]. Processing a manifold through a single tangent space is necessarily restrictive, as only distances to the tangent pole are true geodesic distances. Distances between arbitrary points on the tangent space do not represent true geodesic distances, and hence do not represent the manifold structure accurately [Jos05].

As a partial workaround to the above limitation, Tuzel et al. [TPM08] recently proposed a pedestrian detection method based on a LogitBoost cascade [FHT00] over Riemannian manifolds. Linear regression was used as a weak classifier within the cascade, after mapping the selected training points to a tangent space that occurs at the weighted Karcher mean of the points. This approach mitigates the problem of incorrect distance representation described above. Here, distances between arbitrary points are better represented because, on average, all points are closer to the tangent pole where geodesic distances are locally preserved.

A more complex approach involves using training data to create coordinate charts that try to preserve the relations between points [LZ08, HSWL12]. The main drawback of these approaches is that they tend to be computation-
ally expensive as they require repetitive calculation of geodesic distances, which involve costly SVD operations. Finally, we note that geodesic distances could be directly used in conjunction with distance-based learning techniques like nearest-neighbour classification. However, this approach also requires repetitive calculation of geodesic distances and has been shown to be less reliable than using coordinate charts [HSLW12].

3.3 Proposed Pedestrian Detection Method

As mentioned before, with both the robustness of covariance descriptors as features and exploiting their non-linear nature for improved classification, Tuzel et al. [TPM08] show superior pedestrian detection performance compared to related state-of-the-art methods. However, as discussed in the previous section, tangent space representations (even the one at the weighted Karcher mean) do not retain true geodesic distances between points other than the tangent pole. Furthermore, considering more complex manifold representations such as coordinate charts becomes impractical for detection applications where thousands of examples are processed for training and as many windows have to be scanned inside every image in search of pedestrians.

In this section, we propose a general discriminative model based on the combination of several tangent spaces, in order to preserve more details of the manifold structure while still maintaining efficiency of computation. We shall refer to this model as \(k\)-tangent spaces. The model can be used as a weak classifier in a boosting-based framework, allowing the learning algorithm to focus on better classification rather than finding the best representation of the manifold.

For ease of comparison, we used the same features and cascaded approach as in [TPM08], but replacing each weak classifier with a \(k\)-tangent spaces model. An overview of the features and construction of the covariance descriptors is presented in Section 3.3.1. Section 3.3.2 details the proposed \(k\)-tangent spaces model and the overall classification algorithm is summarised in Section 3.3.3.
3.3.1 Covariance descriptors

To calculate the covariance descriptor representing a subwindow $r$ inside some region of interest $R$, a feature vector is extracted at each pixel inside the subwindow using the following 8 gradient-based features:

$$
\mathbf{z} = \begin{bmatrix} x & y & |d_x| & |d_y| & |d_{xx}| & |d_{yy}| & \sqrt{d_x^2 + d_y^2} & \arctan \frac{|d_y|}{|d_x|} \end{bmatrix}
$$

(3.9)

where $x$ and $y$ indicate the location of the pixel, $d_x = \frac{\partial I(x,y)}{\partial x}$, $d_{xx} = \frac{\partial^2 I(x,y)}{\partial x^2}$ (with $I(x,y)$ representing the pixel intensity), and the last two terms represent edge magnitude and orientation. The covariance matrix $C_r$ is calculated from the $S$ feature vectors inside the subwindow $r$ and then normalised as $\hat{C}_r$ following [TPM08]:

$$
C_r = \frac{1}{S-1} \sum_{i=1}^{S} (\mathbf{z}_i - \mu)(\mathbf{z}_i - \mu)^T
$$

(3.10)

$$
\hat{C}_r = \text{diag}(C_R)^{-\frac{1}{2}} C_r \text{diag}(C_R)^{-\frac{1}{2}}
$$

(3.11)

where $\mu$ is the mean vector and $C_R$ is the covariance matrix of the feature vectors inside the entire region of interest $R$ containing subwindow $r$.

3.3.2 $k$-tangent spaces model

Intuitively speaking, the larger the area covered by the tangent space, the worse the representation. To address this issue, we propose the combination of multiple tangent spaces into a single discriminative model. The poles of these tangent spaces are the centres of previously clustered data.

The concept of combining results from various tangent spaces has been partially explored before. For instance, Lui [Lui12b] used tangent bundles on special manifolds for action recognition. Actions are represented by fragments of video, which can be expressed as a third order data tensor $A \in \mathbb{R}^{W \times H \times T}$, where $W$, $H$ and $T$ are the video width, height and length, respectively. To represent a single action, the author uses high order singular value decomposition (HOSVD) of the tensor data to produce a set of three orthogonal matrices. In the context of action videos, these matrices represent
appearance, horizontal motion and vertical motion. The learning approach chosen by the author is nearest neighbour (NN) classification, which needs a distance measure between video samples. The concept of tangent bundles is used to obtain a single distance value by comparing the sets of matrices.

Formally, the tangent bundle of a differentiable manifold $\mathcal{M}$ is the disjoint union of the tangent spaces of $\mathcal{M}$:

$$TM = \bigcup_{x \in \mathcal{M}} T_x \mathcal{M}$$  \hspace{1cm} (3.12)

which, in the case of $[\text{Lui12b}]$, is essentially the set of orthogonal matrices. The author then defines the intrinsic distance on a tangent bundle as:

$$d_{TM} = \sum_{k=1}^{N} g_c(\Delta_k)$$  \hspace{1cm} (3.13)

where $\Delta_k$ is the tangent vector computed from the $k$-th tangent plane, $g_c$ is the associated canonical metric used to measure the length of the vector, and $N$ is the order of the data tensor (3 in the case of action videos).

Similarly, Tosato et al. $[\text{TFCM10}]$ perform part-based human detection by selecting a hierarchy of 11 body part regions represented as covariance descriptors. For each body part they train a separate LogitBoost classifier on a single tangent space. The final classifier is represented as a weighted sum of the body part classifiers.

Although we use a similar model combination approach to generate our $k$-tangent spaces model, it is important to note that we are attempting to solve a conceptually different problem compared to the two approaches discussed above. In $[\text{Lui12b}]$ and $[\text{TFCM10}]$, each sample is represented as a collection of manifold points obtained from semantically meaningful feature groups (appearance, horizontal motion and vertical motion in $[\text{Lui12b}]$; head, torso, legs and arms in $[\text{TFCM10}]$). Therefore, model combination is needed to produce a single distance value to compare sets of matrices or to generate a single classifier, respectively.

In our case, the multiple models are not produced by manifold points created from various feature groups. Instead, we represent each point with a single covariance descriptor and generate multiple models trained on various
tangent spaces obtained after clustering the training data. In other words, the two previous approaches deal with multiple covariance descriptors per sample, trained on a single tangent space. In contrast, our approach deals with a single covariance descriptor per sample, trained on multiple tangent spaces.

We combine multiple tangent spaces into a single discriminative model as follows:

$$\lambda = \{(\mu_k, n_k, g_k)\}_{k=1}^{K}$$

(3.14)

where, for the $k$-th entry, $\mu_k$ is the tangent pole (centre), $n_k$ is the number of samples associated with the centre, and $g_k$ is a discriminative model associated with the centre. By having several spaces, the likelihood of each sample point having at least one appropriate vector space representation is increased.

For a given test sample $Z \in \mathcal{M}$, the combined output of the $K$ discriminative models can be obtained using a basic probabilistic mixture:

$$G(Z|\lambda) = \sum_{k=1}^{K} \frac{n_k}{\sum_{l=1}^{K} n_l} g_k \left(\log_\mu_k(Z)\right)$$

(3.15)

where each $n_k$ is used as a mixing coefficient, emphasising more dense clusters that are less likely to be affected by outliers.

Within the context of pedestrian detection, the proposed model is trained by first grouping positive samples into $K$ clusters. Negative samples are not taken into account at this stage as they are not well characterised (i.e., they do not represent a coherent class). To obtain the cluster centres, a form of the $k$-means algorithm [Elk03] explicitly adapted to manifolds can be used. Specifically, the geodesic distance in (3.6) is used as the distance measure, with each cluster centre taken to be a Karcher mean, as in (3.7).

Once the cluster centres are obtained, all samples are projected onto each tangent space $k$ and a discriminative model $g_k$ is trained. The process is summarised in Algorithm 3.1.

The $k$-tangent spaces model can be used to replace the weak classifier (which uses a single tangent space) within the boosting-based algorithm.
Algorithm 3.1: Training the \textit{k-tangent spaces} model for pedestrian detection

\textbf{Input:} Training samples with labels \( \{(X_i, y_i)\}_{i=1}^{N}, X_i \in \mathcal{M}, y_i \in \{-1, +1\} \)

- Cluster positive samples to obtain \( K \) cluster centres and the number of samples associated with each centre: \( \{\mu_k, n_k\}_{k=1}^{K} \)

- For \( k = 1, \ldots, K \)
  - Map all samples to the tangent space at \( \mu_k \) via \( \log_{\mu_k}(\cdot) \) defined in (3.3)
  - Train discriminative model \( g_k \) using the mapped samples and their labels

\textbf{Output:} The model \( \lambda = \{\mu_k, n_k, g_k\}_{k=1}^{K} \)

proposed by Tuzel \textit{et al.} [TPM08]. The discriminative models within each \textit{k-tangent spaces} model are the same as the original weak learners (regression functions) used in [TPM08]. The resulting strong classifier produced by boosting is hence comprised of a cascade of \textit{k-tangent spaces}.

### 3.3.3 Detection algorithm

The final detector is a series of 30 LogitBoost classifiers (levels) with rejection cascade. Before training each cascade level, 10,000 negative examples are randomly sampled and the ones which are not rejected by the previous levels are added to the training set. The LogitBoost classifier at each level is trained by iteratively selecting the best \textit{k-tangent spaces} model among 200 random subwindows, until at least 35 percent of the negatives examples are rejected, while correctly detecting 99.8 percent of the positive examples.

Briefly, LogitBoost classifies points in the \( d \)-dimensional vector space to binary outputs \( y_i \in \{0, 1\} \). In the algorithm, a set of \( M \) weighted least squared regressions \( g_m(x) \) of training points \( x_i \in \mathbb{R}^d \) is fitted to response values \( z_i \in \mathbb{R} \), with weights \( w_i \). The response values and weights are calculated as follows:

\[
    z_i = \frac{y_i - p(x_i)}{p(x_i)(1 - p(x_i))} \quad (3.16)
\]

\[
    w_i = p(x_i)(1 - p(x_i)) \quad (3.17)
\]

where \( p(x_i) \) is the probability of \( x_i \) belonging to the positive class (i.e., \( y_i = 1 \))
and is calculated using:

\[
p(x) = \frac{e^{F(x)}}{e^{F(x)} + e^{-F(x)}} \quad (3.18)
\]

\[
F(x) = \frac{1}{M} \sum_{m=1}^{M} g_m(x) \quad (3.19)
\]

A detailed description of LogitBoost can be found in [FHT00].

For every training sample, the normalised covariance matrix is obtained inside each subwindow \( r \). For each subwindow a \( k \)-tangent spaces model is constructed using Algorithm 3.1. The best model is chosen and added to the current level as a weak classifier. Weak classifiers are added until the target rates are satisfied. As in [TPM08], the target rates are accompanied by a margin condition defined as follows. Let \( p_l(R) \) be the learned probability function of a sample being positive at cascade level \( l \). We continue to add weak classifiers to level \( l \) until \( p_l(R_p) - p_l(R_n) < \text{margin} \), where \( R_p \) is the positive example with the \((0.998N_p)\)-th largest probability, \( R_n \) is the negative example with the \((0.35N_n)\)-th smallest probability, and \( \text{margin} = 0.2 \).

The overall approach is summarised in Algorithm 3.2. We reiterate that, for ease of comparison, it follows the same steps as the algorithm proposed in [TPM08], but replacing each weak classifier with a \( k \)-tangent spaces model.

Once the detector has been trained, a test example \( R \) can be classified as positive or negative using:

\[
\text{class}(R) = \text{sign}[F_l(R) - \text{thrd}_l]
\]

\[
= \text{sign} \left[ \sum_{m=1}^{M_l} G_{l,m}(\hat{C}_{r_{l,m}}) - \text{thrd}_l \right] \quad (3.20)
\]

where \( G_{l,m} \) refers to (3.15), with \( F_l \) and \( \text{thrd}_l \) defined in Algorithm 3.2.

### 3.4 Experiments

We used two datasets of pedestrian images for the experiments: INRIA person dataset [DT05] and DaimlerChrysler pedestrian classification benchmark.
Algorithm 3.2: Pedestrian detection using \(k\)-tangent spaces

**Input:** Training set \(\{(R_i, y_i)\}_{i=1}^N\), \(y_i \in \{0, 1\}\)

- For \(l = 1, \ldots, L\)
  - Sample negative examples and keep false positives only
  - Start with \(w_i = 1/N, F_l(R) = 0\) and \(p_l(R_i) = 1/2\)
  - While \(p_l(R_p) - p_l(R_n) < \text{margin}\)
    * Compute the response values and weights \(z_i = \frac{w_i - p_l(R_i)}{p_l(R_i)(1 - p_l(R_i))}, w_i = p_l(R_i)(1 - p_l(R_i))\)
    * Sample \(\{r_{l,t}\}_{t=1}^{200}\) subwindows and construct normalised covariance descriptors:
      \(X_{i,l,t} = \hat{C}_{i,r_{l,t}}\)
    * For each subwindow, construct a \(k\)-tangent spaces model \(G_{l,t}\) using Algorithm 3.1, with linear regression as the models to fit, using the response values \(z_i\) and weights \(w_i\)
    * Update \(F_l(R) \leftarrow F_l(R) + \frac{1}{2} G_{l,m}(R)\), where \(G_{l,m}\) is the best model among \(\{G_{l,t}\}_{t=1}^{200}\)
      which minimises the negative binomial log-likelihood:
      \[-\sum_{i=1}^N [y_i \log(p(x_i)) + (1 - y_i) \log(1 - p(x_i))]\]
    * Update \(p_l(R) \leftarrow \frac{p_l(R_i)}{\sum_{k=1}^{M_l} p_l(R_k)}\)
  - Sort positive and negative samples according to descending probabilities and find samples at the decision boundaries \(R_p = (0.998 N_p)\)-th \(R^+\), \(R_n = (0.35 N_n)\)-th \(R^-\)
  - Store \(F_l = \{(r_{l,m}, G_{l,m})\}_{m=1}^{M_l}\) and \(\text{thrd}_{l} = F_l(R_n)\)

**Output:** The rejection cascade \(\{F_l\}_{l=1}^L\)

The INRIA dataset contains 2416 pedestrian samples and 1218 person-free images for training, as well as 1126 pedestrian samples and 453 person-free images for testing. The size of the samples is \(64 \times 128\) with a margin of 16 pixels around the pedestrians. The main challenges of the INRIA dataset are variations in pose, background and partial occlusions.

The DaimlerChrysler dataset contains three training sets and two test sets. Each training and test set contains 4800 pedestrian samples and 5000 negative samples. Additionally, each training set contains 1200 person-free images where further negative samples can be obtained. The size of the positive samples is \(18 \times 36\) with a margin of 2 pixels around the pedestrians. Detection on this dataset is more difficult due to the small size of the samples and the challenging negative set.

In our experiments, as per [TPM08], each pixel \(I_{(x,y)}\) located at \((x, y)\) was represented by the 8-dimensional feature vector described in (3.9).
Chapter 3. Pedestrian Detection

![Figure 3.8: Examples of pedestrian images in (a) INRIA and (b) DaimlerChrysler datasets.](image)

The covariance descriptor of a given region of pixels (selected by the boosting framework) is hence an $8 \times 8$ matrix. In all cases the proposed $k$-tangent spaces model was used as a weak learner within the boosting-based pedestrian detection framework proposed by Tuzel et al. [TPM08].

For ease of comparison with previous approaches, the results of all our experiments are presented as detection error trade-off curves on a log-log scale, following the framework of [TPM08]. The curves plot miss rate (MR) against false positives per window (FPPW)$^5$, calculated using:

$$
\text{MR} = \frac{FN}{FN + TP} \quad (3.21)
$$

$$
\text{FPPW} = \frac{FP}{FP + TN} \quad (3.22)
$$

where $TP$, $TN$, $FP$, $FN$ stand for true positives, true negatives, false positives and false negatives, respectively. Each trade-off curve was generated

---

$^5$False positives can be measured at the window level (i.e., testing a single, usually cropped, sample) or at the image level (i.e., testing multiple regions inside each image). Authors use the terms per-window and per-image to differentiate between both rates.
by adding one cascade level at a time (i.e., from using only the first level of the rejection cascade until using all 30 levels)\(^6\). For all curves the best performance point is at the bottom-left corner of the graph (i.e., minimal error rates).

We first performed two preliminary experiments to justify our parameter selection, and then a comparative evaluation against several notable pedestrian detection methods in the literature.

### 3.4.1 Preliminary experiments

Here we present two preliminary experiments on the INRIA dataset to justify the two most important parameters of our pedestrian detection method: (i) the number of tangent spaces used in the \(k\)-tangent spaces model, and (ii) the tangent space representation used for mapping manifold points to a vector space. To avoid training on the background, the margin around the pedestrians in the positive samples was reduced from 16 to 2 pixels.

#### Number of tangent spaces

In the first preliminary experiment we varied the number of tangent spaces from 1 to 5. The results, presented in Figure 3.9, show that the detection performance generally increases as the number of tangent spaces increases to 3 or 4. Using 5 tangent spaces degrades performance. We conjecture that the number of training samples affects the optimal number of tangent spaces. The more spaces are used, the lower the number of samples is available for each cluster, and hence it is more likely that the clusters are affected by outliers.

#### Tangent space representations

In the second preliminary experiment, we compared the performance of four tangent-based mapping approaches: (i) directly converting the covariance descriptors into vectors (i.e., without projection to a tangent space); (ii) projecting the points to the tangent space with the identity matrix as the pole;

---

\(^6\)For ease of comparison with related methods the curves were cropped at MR = 0.01.
(iii) projecting the points to the tangent space with the Karcher mean of positive samples as the pole; and (iv) using the proposed \( k \)-tangent spaces model with 3 tangent spaces\(^7\).

The results, presented in Figure 3.10, indicate that any tangent space representation leads to better performance compared to the direct vector space representation. The Karcher mean mapping outperforms the identity matrix mapping, in agreement with the results presented in [TPM08]. The proposed \( k \)-tangent spaces model leads to the best performance.

3.4.2 Comparative evaluation

We compared the proposed approach using 3 tangent spaces against several notable pedestrian detection methods in the literature. We restricted the selection to methods with reported performance on either the INRIA or the DaimlerChrysler dataset.

\(^7\)The best results were obtained using 3 or 4 tangent spaces. Using 3 tangent spaces requires less computation and we consider it the best trade-off between detection accuracy and speed.
Chapter 3. Pedestrian Detection

Figure 3.10: Pedestrian detection using various tangent space representations on the INRIA dataset. MR and FPPW are described in (3.21) and (3.22).

**INRIA dataset**

For the INRIA dataset, we selected three techniques based on HOG features: two SVM based methods [DT05], and an AdaBoost based method [ZYCA06]. We have also compared against the method of Tuzel et al. [TPM08], where mapping based on the Karcher mean was used (i.e., single tangent space).

To allow a direct comparison with previously published results, the original images in the INRIA dataset were used (i.e., with a margin of 16 pixels around the pedestrians), rather the margin reduced images as used in the first and second experiments. The results, presented in Figure 3.11, indicate that considerably better detection performance is obtained by the proposed $k$-tangent spaces approach.

**DaimlerChrysler dataset**

For the more challenging DaimlerChrysler dataset, we selected the best results for the three descriptors proposed in [MG06]: PCA coefficients, Haar wavelets, and local receptive fields (LRF). We have also compared against the method of Tuzel et al. [TPM08], where mapping based on the weighted
Chapter 3. Pedestrian Detection

Figure 3.11: Comparison with other pedestrian detection methods on the INRIA dataset. The configuration of pedestrian images was different than used in Figure 3.10, to allow direct comparison with previously published results [DT05, MG06, TPM08, ZYCA06]. MR and FPPW are described in (3.21) and (3.22).

Karcher mean was used (i.e., single tangent space). The results, presented in Figure 3.12, indicate that markedly better detection performance is obtained by the proposed k-tangent spaces approach.

3.5 Main Findings and Future Directions

Covariance descriptors have been recently demonstrated to be useful features for pedestrian detection, due to their relative low sensitivity to translations and scale variations. By interpreting the descriptors as points on Riemannian manifolds, and by exploiting the curvature of such manifolds, higher performance can be achieved than vector space approaches such as histograms of oriented gradients.

Current manifold-based classification approaches typically map the points to a tangent space, where traditional machine learning techniques are used. However, such treatment is restrictive as distances between arbitrary points on the tangent space do not represent true geodesic distances, and hence do
In this chapter, we proposed a general discriminative model based on the combination of several tangent spaces, in order to preserve more details of the structure. By having several spaces, the likelihood of a sample point having at least one appropriate vector space representation is increased.

The model was used as a weak learner within the boosting-based pedestrian detection framework proposed by Tuzel et al. [TPM08]. Experiments on the challenging INRIA and DaimlerChrysler datasets indicate that the proposed model (with just a few tangent spaces) leads to considerably improved performance.

It is worth noting that, as a consequence of boosting, the final inference on the manifold in [TPM08] was made through multiple tangent spaces. However, there is an inherent limitation in that each boosting iteration has to deal with two problems at the same time: finding the best tangent representation and finding the best discriminative model. This limitation is not present in our approach where the best tangent representation is found with the $k$-tangent spaces model so that boosting is dedicated to finding the best weak classifier.
Future work includes: adapting the model to more general classification problems (i.e., not tied to covariance matrices or pedestrian detection), as long as the samples can be represented as points on Riemannian manifolds; and exploring alternative ways of fusing information [SP04, Bis06] of individual tangent spaces to create the \textit{k-tangent spaces} model.
Chapter 4

Action and Gesture Recognition

As we described in Chapter 1, automatic surveillance systems may comprise one or more modules. In the previous two chapters we presented contributions towards two of these modules; foreground extraction and object detection. In this chapter we present our contribution towards the last module: scene analysis. This name generalises the task of processing information from input images or video to generate the final output of the system. A common task in this module is to recognise the actions or gestures performed by the objects of interest.

Besides automatic visual surveillance, both action and gesture recognition have been useful in areas such as content-based video analysis, human-computer interaction, behavioural sciences and animation. Some examples are shown in Figure 4.1.

Figure 4.1: Example applications that use action of gesture recognition. From left to right: scene classification by activity [LFF07], human-computer interaction for medical imaging [WKSE11], and automated monitoring and analysis of social behaviour of insects [DWH+09].
In this chapter we propose the use of spatio-temporal covariance descriptors for action and gesture recognition tasks. Flat region covariance descriptors were first proposed for the task of object detection and classification in images [TPM08]. Each covariance descriptor represents the features inside an image region as a normalised covariance matrix. They have led to improved results over related descriptors such as histograms of oriented gradients, in terms of detection performance as well as robustness to translation and scale [TPM08]. Furthermore, covariance matrices provide a low-dimensional representation which enables efficient comparison between sets of covariance descriptors.

The proposed spatio-temporal descriptors, which we name Cov3D, belong to the group of symmetric positive definite matrices which do not form a vector space. They can be formulated as a connected Riemannian manifold, and taking into account the non-linear nature of the space of the descriptors may lead to improved classification results. The most common approach for classification on manifolds is to first map the points into an appropriate Euclidean representation [LZ08] and then use traditional machine learning methods. A recent example of mapping is the Riemannian locality preserving projection (RLPP) technique [HSWL12].

The Cov3D descriptors are extracted from spatio-temporal windows inside sample videos, with the number of possible windows being very large. As such, we use a boosting approach to search the subset of windows which is most useful for classification. We propose to extend RLPP by weighting (WRLPP), in order to take into account the weights of the training samples. This weighted projection leads to a better representation of the neighbourhoods around the most critical training samples during each boosting iteration. The proposed Cov3D descriptors, in conjunction with the classification approach based on WRLPP boosting, lead to a state-of-the-art method for action and gesture recognition.

The proposed method includes a boosting framework based on binary LogitBoost classifiers [FHT00]. Since the action and gesture recognition tasks have multiple classes, we create a multi-class classifier from the full set of one-vs-one classifiers. However, later in this chapter we also propose a hybrid classification approach which combines one-vs-all and one-vs-one binary
classifiers. We show how this new multi-class classifier can be used to extend our action and gesture recognition method for improved performance.

We continue the chapter as follows. In Section 4.1 we present a literature review of action and gesture recognition methods. In Section 4.2 we describe the spatio-temporal covariance descriptors (Cov3D), and use the concept of integral video to enable fast calculation inside any spatio-temporal window. In Section 4.3, we propose the weighted Riemannian locality projection approach (WRLPP) used to map manifold points in vector space. In Section 4.4, the boosting classification approach is shown using both our proposed features and weighted mapping approach. In Section 4.5, we compare the performance of the proposed method against several recent state-of-the-art methods on four benchmark datasets. In Section 4.6, we propose an extension of the classification algorithm using an hybrid approach which combines one-vs-one and one-vs-all binary classifiers. Concluding remarks and possible future directions are given in Section 4.7.

The work in this chapter has been published in [SSHL13].

4.1 Review

Human motion analysis can be performed at various levels of abstraction. Moeslund et al. [MHK06] presented a hierarchy comprised of action primitives, actions and activities. Action primitives are atomic movements at the limb level, actions are sequences of action primitives describing particular whole-body movements (e.g., walking, running, jumping), and activities comprise a set of actions which give an interpretation of the movement in the scene (e.g., playing golf, jumping hurdles, stealing). For simplicity, we refer to both whole-body movements and activities as actions, and we use the term gestures in reference to movements of body parts or their interpretation in the scene (e.g., hand-waving, smiling).

As with typical computer vision applications, action recognition can be viewed as a combination of: (i) feature extraction for image representation, and (ii) classification. Based on the material reviewed, similar to object detection, action recognition literature tends to mainly focus on proposing a novel feature representation and then choosing the classification approach.
which is more compatible with the features, or the one which leads to better recognition results. Below, we first list the most common options of feature representation and use it to categorise related methods for action and gesture recognition. Then we summarise the main classification options used in conjunction with these features for action and gesture recognition.

4.1.1 Features

According to a recent survey on recognition of human activities [TCSU08], the focus has shifted to methods that do not rely on human body models, where the information is extracted directly from the images and hence are less dependent on reliable segmentation and tracking algorithms. Such image representation methods can be categorised into global and local based approaches [Pop10]. We use this categorisation to review related methods in action and gesture recognition.

**Global representations**

Methods with global image representation encode visual information as a whole. Global representations for action recognition are commonly derived from silhouettes, optical flow or edges.

Davis and Bobick [DB97] constructed a temporal template from an action sequence by extracting silhouettes and aggregating the differences between subsequent frames. The vector values at each point in the templates are formed by two methods of aggregation. The first indicates the presence of motion and the second is a function of the recency of motion. These templates have sufficient discriminating ability for simple actions but have low discriminative power for complex activities [Bob97]. Wang et al. [WHT07] applied the $\mathcal{R}$ transform to the extracted silhouettes to create scale and translation invariant templates. The $\mathcal{R}$ transform, an adaptation of the Radon transform, is invariant to common geometrical transformations [TWS06].

Optical flow is the pattern of apparent motion of objects in a visual scene. A common starting point for optical flow estimation is to assume that pixel intensities are translated from one frame to the next [FW06]. Ali and Shah [AS10] extracted a series of kinematic features based on optical flow. A
Chapter 4. Action and Gesture Recognition

group of kinematic modes is found using principal component analysis (PCA). Guo et al. [GIK10] encode the same kinematic features using sparse representation of covariance matrices.

Several methods first divide the region of interest into a fixed spatial or temporal grid, extract features inside each cell and then combine them into a global representation. For example, this can be achieved using local binary patterns (LBP) [KZP08], or histograms of oriented gradients (HOG) [TH08]. Global representations are sensitive to viewpoint, noise and occlusions which may lead to unreliable classification; furthermore, global representations depend on reliable localisation of the region of interest [Pop10].

Local representations

Local representations are designed to deal with the abovementioned issues by describing the visual information as a collection of patches, usually at the cost of increased computation. Laptev and Lindeberg [LL03] extracted interest points using a 3D Harris corner detector and use the points for modelling the actions. Spatio-temporal approaches based on interest points have been extensively used in the past (e.g., [PGC08, BGX09]). One of their major drawbacks is the low number of interest points that are able to remain stable across an image sequence. A common solution is to work with windowed data, extracting salient regions which can be represented using Gabor filtering [DRCB05], discrete wavelet transform [RAK07] or Hessian matrices [WTVG08].

Wang et al. [WUK+09] showed that dense sampling approaches tend to perform better compared to interest point based approaches. Dense sampling is typically done for a set of patches inside the region of interest. Features are extracted from each patch to form a descriptor. These descriptor representations differ from grid-based global representations in that they can have an arbitrary position and size, and that the patches are not combined to form a single representation but a set of multiple representations. Examples are subspaces of correlated movement [WC07], wavelet-based descriptors [KSH05], as well as HOG and HOF descriptors [LMSR08], SIFT descriptors [Low04], and their respective spatio-temporal versions: HOG3D [KMS08] and 3D SIFT [SAS07]. Because of the likely large number of descriptors and/or their high
dimensionality, comparing sets of descriptors is often not straightforward. This has led to compressed representations such as formulating sets of descriptors as bags-of-words [NWFF08].

As discussed in the previous chapter, covariance descriptors are successful representations for detection on still images compared to related state-of-the-art approaches such as HOG. We follow the logical approach for adapting image-based classification approaches to video-base classification (which is the case for actions and gestures) by presenting a spatio-temporal version of covariance descriptors, along with a classification approach which works best with them.

### 4.1.2 Classification

Once an action or gesture sequence has been converted to feature representation, the recognition task becomes a classification problem. Poppe [Pop10] divides classification approaches into direct classification and temporal state-space models. We discuss these two categories below. Additionally, because several classification approaches are binary in nature and the action and gesture recognition problem usually involves multiple classes, we include a third section discussing common ways of combining binary classifiers into multi-class classifiers.

**Direct classification**

Direct classification approaches classify feature representation without explicitly modelling variations in time. A common previous step to this approaches is to perform dimensionality reduction using PCA [MP03] or non-linear mappings to lower-dimensional spaces [WS08].

When a reliable distance measure between feature representation can be obtained, a common approach is to use nearest neighbour (NN) classification. This approach can be used for global features [BGS+05], local features such as histograms of codewords [BCS08], or in combination with dimensionality reduction [WS07b, TVSC11, HSWL12].

Another approach is to use discriminative classifiers which are designed to separate action of gesture classes instead of modelling them. The most com-
mon options are support vector machines (SVM) [JSWP07, LCSL07, SLC04] and boosting frameworks [FM08, LP07, OCKI06, NBT07]. An advantage of boosting approaches is that they can also be used for automatic feature selection\(^1\).

**Temporal state-space models**

Temporal state-space approaches model variations in time of actions or gestures. They represent instances of actions or gestures with states and model their transition probabilities using edges. As with direct classification approaches, temporal state-space models can also be generative or discriminative.

Generative approaches aim to model each action class with all its variations. The most known generative approaches are hidden Markov models (HMM), which have been extensively used for action and gesture recognition [YOI92, WBR07, LN07]. On the other hand, discriminative models focus on the differences between action or gesture classes, for which the most representative approaches are conditional random fields (CRF) [SKLM05, WS07c, ZG10].

A possible third class off temporal state-space models is dynamic time warping [VRCC05, YZ09], where distances between whole action or gesture sequences are generated. These approaches could be considered generative but they function differently as they are used between pairs of sequences.

**Multi-class classification**

Several classifiers which are common for action and gesture recognition, such as SVM and some forms of boosting, are binary in nature. Although some of these methods can be directly extended to take into account multiple classes, the multi-class classification problem can also be divided into several binary classification tasks which can be solved efficiently [ASS01]. Two important concepts define how a multi-class classifier is generated using binary classifiers. First is the selection of classes or groups or classes the classifiers will

\(^1\)Boosting is explained in Section 3.1.2.
discriminate from. Second is how a consensus is reached from the decisions of individual binary classifications to produce the overall output.

The simplest way to reduce classification of $K$ classes is to generate $K$ binary classifiers. Each classifier discriminates one class from the other $K-1$ classes. This approach is known as one-vs-all and, although simple, provides good classification performance if the classes are reasonably easy to separate [TD02]. Another common approach is to generate binary classifiers to discriminate between each pair of classes. This approach is known as one-vs-one (or all-vs-all) and it results in a group of $\frac{K(K-1)}{2}$ binary classifiers. It is more computationally expensive as each sample needs to be tested against more classifiers, but it usually leads to better classification results, compared to the one-vs-all approach [HL02].

There are two common ways of combining the decision of the individual binary classifiers into a single output\(^2\). The first and simplest approach is voting, where each sample is labelled as the class with the highest number of votes produced by the binary classifiers. The main problem of this approach is that voting can produce inconsistent results due to ties or contradicting votes [RDS01]. The second approach, confidence value estimation, aims to overcome this problem. This approach takes into account the probability of the sample being of the positive class of each binary classifier, to form a real-valued confidence value. Using probability estimates produces more stable results compared to voting [WLW04].

In our work, as we use a boosting framework based on binary LogitBoost classifiers [FHT00] for automatic feature selection, we opt for the one-vs-one approach. However, in Section 4.6 we propose a hybrid classification approach which combines one-vs-all and one-vs-one binary classifiers. We show how this new multi-class classifier can be used to extend our action and gesture recognition method for improved performance.

\subsection*{4.2 Cov3D Descriptors}

In this section we first present the general form of the proposed spatio-temporal covariance descriptors (Cov3D), an algorithm for their fast calcu-\(^2\)A more complete categorisation can be found in [SP04].
Chapter 4. Action and Gesture Recognition

Figure 4.2: Conceptual demonstration for obtaining a spatio-temporal covariance descriptor (Cov3D). A spatio-temporal window $R$ is defined inside the input video. For each pixel in $R$ a feature vector $z_i$ is calculated. The feature vectors are then used to compute the covariance matrix $Cov3D_R$.

Let $V$ be the sequence of images $\{I_t\}_{t=1}^T$, and $F$ be the $W \times H \times T \times d$-dimensional feature video extracted from $V$:

$$F(x, y, t) = \Phi(V, x, y, t)$$  

where the function $\Phi$ can be any mapping such as intensity, colour, gradients, or optical flow. For a given spatio-temporal window $R \subset F$, let $\{z_i\}_{i=1}^S$ be the $d$-dimensional feature vectors inside $R$. The region $R$ is represented with the $d \times d$ covariance matrix of the feature vectors:

$$Cov3D_R = \frac{1}{S-1} \sum_{i=1}^S (z_i - \mu)(z_i - \mu)^T$$  

where $\mu$ is the mean vector. Figure 4.2 shows the construction of a covariance descriptor inside a spatio-temporal window. Examples of feature vectors specific for action and gesture recognition are given in Section 4.2.2.

Representing a spatio-temporal window with a covariance matrix has several advantages: (i) it is a low-dimensional representation which is independent on the size of the window, (ii) the impact of noisy samples is reduced through the averaging during covariance computation, (iii) it is a straightforward method of fusing correlated features.
4.2.1 **Fast Cov3D computation**

Viola and Jones [VJ01] introduced the concept of integral images, an intermediate image representation used for the fast calculation of region sums. Ke et al. [KSH05] extended the concept to image sequences, where the integral images are stacked to form an integral video, which can be used to compute spatio-temporal region sums in constant time. For a video $V$, its integral video $IV$ is defined as:

$$\text{IV}(x', y', t') = \sum_{x \leq x'} \sum_{y \leq y'} \sum_{t \leq t'} V(x, y, t)$$  \hfill (4.3)

Tuzel et al. [TPM08] used the integral image representations for fast calculation of flat region covariances. Here we extend the idea for fast calculation of covariance matrices inside a spatio-temporal window using the integral video representation. Following [TPM08], the $(i, j)$-th element of the covariance matrix defined in (4.2) can be expressed as:

$$\text{Cov3D}_R(i, j) = \frac{1}{S - 1} \left[ \sum_{k=1}^{S} z_k(i)z_k(j) - \frac{1}{S} \sum_{k=1}^{S} z_k(i) \sum_{k=1}^{S} z_k(j) \right]$$  \hfill (4.4)

where $z_k(i)$ refers to the $i$-th element of the $k$-th vector.

To find the covariance in a given spatio-temporal window $R$, we have to compute the sum of each feature dimension, $z(i)^d_{i=1}$, as well as the sum of the multiplication of any two feature dimensions, $z(i)z(j)^d_{i,j=1}$. With $d$ representing the number of dimensions, we construct $d + d^2$ integral videos for each feature dimension $z(i)$ and multiplication of any two feature dimensions $z(i)z(j)$. Then, the covariance of any spatio-temporal window can be computed in $O(d^2)$ time, as follows:

Let $\mathbb{P}$ be the $W \times H \times T \times d$ tensor of the integral videos:

$$\mathbb{P}(x', y', t', i) = \sum_{x \leq x'} \sum_{y \leq y'} \sum_{t \leq t'} F(x, y, t)(i)$$  \hfill (4.5)

where $F(x, y, t)(i)$ is the $i$-th element of vector $F(x, y, t)$. Furthermore, let $\mathbb{Q}$ be the $W \times H \times T \times d \times d$ tensor of the second-order integral videos:
\[
Q(x', y', t', i, j) = \sum_{x \leq x'} \sum_{y \leq y'} \sum_{t \leq t'} F(x, y, t)(i) \cdot F(x, y, t)(j) \quad (4.6)
\]
for \(i, j = 1, \ldots, d\). The complexity of calculating these tensors of integral videos is \(O(d^2WHT)\). The \(d\)-dimensional feature vector \(p_{x,y,t}\) and the \(d \times d\) dimensional matrix \(Q_{x,y,t}\) can be obtained from the above tensors using:

\[
p_{x,y,t} = \begin{bmatrix} P(x, y, t, 1) & \cdots & P(x, y, t, d) \end{bmatrix}^T \quad (4.7)
\]

\[
Q_{x,y,t} = \begin{pmatrix} Q(x, y, t, 1, 1) & \cdots & Q(x, y, t, 1, d) \\
\vdots & \ddots & \vdots \\
Q(x, y, t, d, 1) & \cdots & Q(x, y, t, d, d) \end{pmatrix} \quad (4.8)
\]

Let \(R(x_1, y_1, t_1; x_2, y_2, t_2)\) be the spatio-temporal window containing all points in the set \(\{(x, y, t)|x_1 \leq x \leq x_2, y_1 \leq y \leq y_2, t_1 \leq t \leq t_2\}\), as shown in Figure 4.3. The covariance of the spatio-temporal window bounded by \((0, 0, 0)\) and \((x, y, t)\) is:

\[
\text{Cov3D}_{R(0,0,0; x,y,t)} = \frac{1}{S-1} \left[ Q_{x,y,t} - \frac{1}{S} p_{x,y,t} p_{x,y,t}^T \right] \quad (4.9)
\]

where \(S = x \cdot y \cdot t\). Similarly, after a few arrangements, the covariance of the region \(R(x_1, y_1, t_1; x_2, y_2, t_2)\) can be computed as:

\[
\text{Cov3D}_{R(x_1,y_1,t_1; x_2,y_2,t_2)} = \\
\frac{1}{S-1} \left[ \hat{Q}_{x_2,y_2} + \hat{Q}_{x_1-1,y_1-1} - \hat{Q}_{x_2,y_1-1} - \hat{Q}_{x_1-1,y_2} \right. \\
\left. - \frac{1}{S} (\hat{p}_{x_2,y_2} + \hat{p}_{x_1-1,y_1-1} - \hat{p}_{x_2,y_1-1} - \hat{p}_{x_1-1,y_2}) \right] \\
(\hat{p}_{x_2,y_2} + \hat{p}_{x_1-1,y_1-1} - \hat{p}_{x_2,y_1-1} - \hat{p}_{x_1-1,y_2})^T \quad (4.10)
\]

where:

\[
\hat{p}_{x,y} = p_{x,y,t_2} - p_{x,y,t_1} \quad (4.11)
\]

\[
\hat{Q}_{x,y} = Q_{x,y,t_2} - Q_{x,y,t_1} \quad (4.12)
\]
Figure 4.3: Integral feature video. The spatio-temporal window $R$ is bounded by $(x_1, y_1, t_1)$ and $(x_2, y_2, t_2)$. Each point in $R$ is a $d$-dimensional vector, where $d$ is the number of features.

and $S = (x_2 - x_1 + 1) \cdot (y_2 - y_1 + 1) \cdot (t_2 - t_1 + 1)$. The covariance of any spatio-temporal window can be computed in $O(d^2)$ time.

### 4.2.2 Cov3D for action recognition

Commonly used features for action and gesture recognition include intensity gradients (e.g., [SI07]) and optical flow (e.g., [AS10]). Previous studies have shown the benefit of combining both types of features [DRCB05, WUK+09]. We define the feature mapping $\Phi(V, x, y, t)$, present in (4.1), as the following combination of gradient and optical-flow based features, extracted from pixel location $(x, y, t)$:

$$
\Phi(V, x, y, t) = \begin{bmatrix} x & y & t & g & o \end{bmatrix}^T \tag{4.13}
$$

where:

$$
g = \begin{bmatrix} |d_x| & |d_y| & |d_{xx}| & |d_{yy}| & \sqrt{d_x^2 + d_y^2} & \arctan \left( \frac{d_y}{d_x} \right) \end{bmatrix} \tag{4.14}
$$

$$
o = \begin{bmatrix} u & v & \frac{\partial u}{\partial t} & \frac{\partial v}{\partial t} & \left( \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) & \left( \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right) \end{bmatrix} \tag{4.15}
$$
Chapter 4. Action and Gesture Recognition

Figure 4.4: Two examples of Cov3D windows that, together, can be useful for the recognition of face expressions.

The first four gradient based features in (4.14) represent the first and second order intensity gradients at pixel location \((x, y)\). The last two gradient based features correspond to the gradient magnitude and gradient orientation. The optical-flow based features in (4.15) represent, in order: the horizontal and vertical components of the flow vector obtained using [Far03], the first order derivatives of the flow components with respect to \(t\), and the spatial divergence and vorticity of the flow field as defined in [AS10]. Each descriptor is hence a \(15 \times 15\) matrix, as \(\Phi(V, x, y, t)\) has 15 dimensions.

For reliable recognition, several regions (and hence several descriptors) are typically used. Figure 4.4 shows the spatio-temporal windows of two descriptors which can be used for recognition of face expressions. With the defined mapping, the input video \(V\) is mapped to \(F\), a 15-dimensional feature video. Since the cardinality of the set of spatio-temporal windows \(\{R \subset F\}\) is very large, we only consider windows of a minimum size and increment their location and size by a minimum interval value. Further specifics on the constraints and selection of the windows used in the experiments are given in Section 4.5.
Also following [TPM08], each covariance descriptor $\text{Cov3D}_R$ is normalised with respect to the covariance descriptor of the region containing the full feature video, $\text{Cov3D}_F$, to improve the robustness against illumination variations:

$$\text{Cov3D}_{\hat{R}} = \text{diag}(\text{Cov3D}_F)^{-\frac{1}{2}} \text{Cov3D}_R \text{diag}(\text{Cov3D}_F)^{-\frac{1}{2}}$$  \hspace{1cm} (4.16)

where $\text{diag}(\text{Cov3D}_F)$ is equal to $\text{Cov3D}_F$ at the diagonal entries and the rest is set to zero.

### 4.3 Mapping Approach

In Section 3.2 we introduced the concept of manifolds. We noted that, as classification directly on manifolds is a complex notion, the usual approach is to first map the manifold data into vector space so that classical machine learning techniques can be used. We summarised the usual approaches, where tangent spaces are the simplest choice but do not maintain the manifold structure of points far from the tangent pole. Although there are more complex charting techniques designed for better mapping, in the case of pedestrian detection we decided not to use them for speed and efficiency considerations. Object detection involves learning from tens of thousands of samples and searching for objects usually needs to match thousands of sliding windows in a single image. As these approaches are computationally expensive, they were not practical for the task.

In the case of action recognition, training sets tend to be smaller and, more importantly, we assume that the action has already been localised, so that the classifier has to be only applied to one instance to recognise the type of action or gesture. Therefore, we now take into account the more powerful charting approaches, and we base our mapping approach in the Riemannian locality preserving projection (RLPP) proposed in [HSWL12] as it has been shown useful for action recognition. In Section 4.5.1 we test the performance of our recognition method using various mapping approaches to justify our selected approach.

In this section, we first summarise the Riemannian locality preserving pro-
jection described in [HSWL12]. Second, we present a modified version of the projection which incorporates prior weights of the training samples. This new charting approach, which we name weighted Riemannian locality preserving projection (WRLPP), is designed for improved classification results while using boosting for feature selection.

4.3.1 Riemannian locality preserving projection

RLPP is based on Laplacian eigenmaps [BN03]. Given $N$ training points $\{X_i\}_{i=1}^N$ from the underlying Riemannian manifold $\mathcal{M}$, the local geometrical structure of $\mathcal{M}$ can be modelled by building an adjacency graph $G$. The simplest form of $G$ is a binary graph obtained based on the nearest neighbour properties of Riemannian points: two nodes are connected by an edge if one node is among the $k$ nearest neighbours of the other node$^3$.

From the adjacency graph $G$ we can find the degree and Laplacian matrices, respectively:

\[
D(i, i) = \sum_k G(i, k) \quad (4.17)
\]
\[
L = D - G \quad (4.18)
\]

where the degree matrix $D$ is a diagonal matrix of size $N \times N$, with diagonal entries indicating the number of edges of each node in the adjacency graph.

The aim is to find a mapping from $\mathcal{M}$ to $\mathcal{M}'$ to preserve the local geometry of the manifold. A suitable transform would place the connected points of $G$ as close as possible, while being flexible to some extent for the unconnected points of $G$. Such a mapping can be described by optimising the following objective function:

\[
f = \min \frac{1}{2} \sum_{i,j} (Y_i - Y_j)^2 W(i, j) \quad (4.19)
\]

which punishes connected neighbours if they are mapped far away in $\mathcal{M}'$.

RLPP uses a heat pseudo-kernel matrix $K$, with the $(i,j)$-th element constructed via:

$^3$More complex affinity graphs could also be used to encode distances between points on Riemannian manifolds [STCO4].
Chapter 4. Action and Gesture Recognition

\[ K(i, j) = e^{-\frac{d(X_i, X_j)}{\sigma}} \]  \hspace{1cm} (4.20)

where \( d(\cdot, \cdot) \) is the geodesic distance defined in (3.6).

The final mapping can be found through the following generalised eigenvalue problem [HSLW12]:

\[ K_L K T A = \lambda K D K T A \]  \hspace{1cm} (4.21)

where the eigenvectors with the \( r \) smallest eigenvalues form the projection matrix \( A \).

### 4.3.2 Weighted Riemannian locality preserving projection

As mentioned before, the number of possible \( \text{Cov3D} \) windows inside a sample video is very large. Therefore, we use boosting to search the windows which are most appropriate for classification. We could use the original RLPP mapping approach to map the matrices as vectors at each boosting iteration. However, as shown in [TPM08], the sample weights can be used to generate a mapping which is more appropriate for the critical training samples. Therefore, we propose a modified projection, specifically designed to be used during boosting, which uses sample weights to generate the final mapping. We refer to this approach as weighted Riemannian locality preserving projection (WRLPP).

In the modified projection, the adjacency graph \( G \) is replaced with a weighted adjacency graph \( \hat{G} \), defined as:

\[ \hat{G} = WGW \]  \hspace{1cm} (4.22)

where \( W \) is a diagonal matrix with diagonal values that correspond to the vector of sample weights \([w_1, w_2, \ldots, w_N]\). Using the weighted adjacency graph, edges involving critical samples (i.e., samples with higher weights) become more important and their geometrical structure is better preserved. The modified projection approach is detailed in Algorithm 4.1.

Once the projection matrix \( A \) has been obtained, a given point \( C \) (a \( \text{Cov3D} \) matrix) on the manifold can then be mapped to Euclidean space via:
Chapter 4. Action and Gesture Recognition

**Algorithm 4.1:** Obtaining the WRLPP projection model

**Input:** Training samples (manifold points), labels and weights \(\{(X_i, y_i, w_i)\}_{i=1}^N, X_i \in \mathcal{M}\)

- Create Riemannian pseudo-kernel matrix:
  \[ K(i,j) = e^{-\frac{d(X_i, X_j)}{\sigma}}, \text{ using (3.6) as } d(\cdot, \cdot) \]

- Construct weighted adjacency graph:
  \[ \tilde{G}(i,j) = \begin{cases} w_i \cdot w_j & \text{if } y_i = y_j \text{ and } X_j \text{ is among the } k \text{ nearest neighbours of } X_i \text{ in } K \\ 0 & \text{otherwise} \end{cases} \]

- Obtain the weighted degree \(N \times N\) diagonal matrix \(\tilde{D}(i,i) = \sum_k \tilde{G}(i,k)\)

- Calculate the weighted Laplacian matrix \(\tilde{L} = \tilde{D} - \tilde{G}\)

- The eigenvectors with the \(r\) smallest eigenvalues of the Rayleigh quotient \(\frac{x^T \tilde{D} x}{x^T \tilde{L} x}\) form the projection matrix \(A\)

**Output:** The projection model \(\lambda = \{A, \{X_i\}_{i=1}^N\}\)

\[ WRLPP(C) = A^T K_C \quad (4.23) \]

where \(K_C = [k(X_1, C), k(X_2, C), \ldots, k(X_N, C)]^T\), with \(k(\cdot, \cdot)\) defined in (4.20), and \(\{X_i\}_{i=1}^N\) representing the training points.

### 4.4 Proposed Action and Gesture Recognition Method

As mentioned in the preceding sections, we have chosen to use boosting to find a subset of the best descriptors for classification, as the number of possible Cov3D descriptors inside a sample video is large. For simplicity, we use a combination of one-vs-one LogitBoost classifiers [FHT00] to achieve multi-class classification\(^4\).

We start with a brief review of binary LogitBoost classification\(^5\), with class labels \(y_i \in \{0, 1\}\). The probability of sample \(x\) belonging to class 1 is represented by:

\[ p(x) = \frac{e^{F(x)}}{e^{F(x)} + e^{-F(x)}} \quad (4.24) \]

\(^4\)Alternative multi-class classifiers are explored in Section 4.6.

\(^5\)We previously described LogitBoost classification in Section 3.3.3.
where \( F(x) = \frac{1}{2} \sum_{m=1}^{M} g_m(x) \), with \( g(x) \) representing a weak learner.

The LogitBoost algorithm learns a set of \( M \) weak learners by minimising the negative binomial log likelihood of the data. A weighted least squares regression \( g_m(x) \) of training points \( x_i \in \mathbb{R}^d \) is fitted to response values \( z_i \in \mathbb{R} \), with weights \( w_i \), where:

\[
w_i = p(x_i)(1 - p(x_i)) \quad (4.25)
\]
\[
z_i = \frac{y_i - p(x_i)}{p(x_i)(1 - p(x_i))} \quad (4.26)
\]

As we are using Cov3D descriptors (covariance matrices) as input data, we adapt the weak learners \( g_m(\cdot) \) to use the projected descriptors. In other words, \( g_m(x) \) is replaced with \( g_m(WRLPP(X)) \), with \( X \) representing a covariance matrix.

For each unique pair of classes \((k, l)\), we train a one-vs-one LogitBoost classifier \( C_{<k,l>} \) as follows. Only the samples belonging to the current pair of classes are used for training the binary classifier. One class is selected to be the positive class and the other as the negative class. For each boosting iteration, we search for the region whose Cov3D descriptor best separates positive from negative samples. The descriptor is calculated for all the training samples and mapped to vector space with WRLPP, using the sample weights calculated for the current boosting iteration. Once in vector space, we fit a linear regression and use it as the weak LogitBoost classifier.

To prevent overfitting, the number of weak classifiers on each one-vs-one classifier is controlled by a probability margin between the last accepted positive and the last rejected negative. Both margin samples are determined by the target detection rate \((dr)\) and the target false positive rejection rate \((rr)\). The final multi-class classifier is a set of one-vs-one classifiers. Each one-vs-one classifier \( C_{<k,l>} \) has a positive label \( y_{<k,l>} \) and a threshold \( \tau_{<k,l>} \). The positive label is the label of the class deemed to be positive and the threshold is found via boosting. Algorithm 4.2 summarises the training process.

A sample video \( V \) is classified as follows. Given a one-vs-one classifier \( C_{<k,l>} \), the probability of a sample video \( V \) belonging to the positive class \( y_{<k,l>} \) is evaluated using:
Algorithm 4.2: Boosting with WRLPP for action and gesture recognition

**Input:** Training videos with labels \( \{(V_i, y_i)\}_{i=1}^{N} \) belonging to \( N_c \) classes

- For each unique pair of class labels \(<k,l>\), train the one-vs-one classifier \( C_{<k,l>} \)
  - Restrict the training set to \( \{(V_j, y_j)\}_{j=1}^{N} \mid y_j \in \{k,l\} \)
  - Let either \( k \) or \( l \) be the positive label \( y_{<k,l>} \)
  - Create binary labels \( y_j' \leftarrow (y_j = y_{<k,l>}) \)
  - Start with \( w_j = 1/N \), \( F(V) = 0 \), \( p(V_j) = \frac{1}{2} \), \( m = 1 \)
  - While \( p(V_p) - p(V_n) < \text{margin} \)
    - Compute the response values and weights \( z_j = \frac{y_j' - p(V_j)}{p(V_j)(1-p(V_j))} \), \( w_j = p(V_j)(1-p(V_j)) \)
    - For each spatio-temporal window \( R \)
      - Construct the descriptors \( X_{j,s} = \text{Cov3D}_{j,R_s} \)
      - From \( \{X_{j,s}, y_j, w_j\}_{n=1}^{N} \) obtain the projection model WRLPP, using Algorithm 4.1
      - Map the data points \( x_{j,s} = \text{WRLPP}_s(X_{j,s}) \) using (4.23)
      - Fit function \( g_s(x) \) by weighted least-squares regression of \( z_j \) to \( x_{j,s} \), using weights \( w_j \)
    - Update \( F(V) \leftarrow F(V) + \frac{1}{2} f_m(V) \), where \( f_m \) is the best classifier among \( \{f_s\} \) which minimises the negative binomial log-likelihood:
      \[ -\sum_{j=1}^{N} [y_j' \log(p(x_j)) + (1 - y_j') \log(1 - p(x_j))] \]
    - Update \( p(V) \leftarrow \frac{e^{F(V)}}{1 + e^{F(V)}} \)
    - Sort positive and negative samples according to descending probabilities and find samples at the decision boundaries \( V_p = (dr \cdot N_p) \)-th \( V^+ \), \( V_n = (rr \cdot N_r) \)-th \( V^- \), where \( dr \) and \( rr \) are the desired detection and false positive rejection rates
    - \( m \leftarrow m + 1 \)
  - Store \( C_{<k,l>} = \{(R_m,\text{WRLPP}_m,g_m)\}_{m=1}^{M} \), threshold \( \tau_{<k,l>} = F(V_n) \) and positive label \( y_{<k,l>} \)

**Output:** The set of \( \frac{N_c(N_c-1)}{2} \) one-vs-one classifiers

\[
C_{<k,l>}(V) = \sum_{m=1}^{M} g_m \left( \text{WRLPP}_m(\text{Cov3D}_{R_m}) \right) - \tau_{<k,l>} \tag{4.27}
\]

After evaluating \( V \) with all the one-vs-one classifiers in the set, the sample is labelled as the class \( a \) which maximises:

\[
C(V) = \arg \max_a \sum_{i \neq a} C_{<a,i>}(V) \text{sign}_a(C_{<a,i>}(V)) \tag{4.28}
\]

where \( \text{sign}_a(C_{<a,i>}(V)) \) is \( \text{sign}(C_{<a,i>}(V)) \) if \( a \) is the positive label \( y_{<a,i>} \), and \( 1 - \text{sign}(C_{<a,i>}(V)) \) otherwise. That is, \( V \) is labelled as the class with greatest probability sum of the one-vs-one classifiers that evaluate to that class.
Chapter 4. Action and Gesture Recognition

<table>
<thead>
<tr>
<th>Type</th>
<th>activities</th>
<th>sports</th>
<th>emotions</th>
<th>hand gestures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classes</td>
<td>6</td>
<td>10</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>Subjects</td>
<td>25</td>
<td>—</td>
<td>123</td>
<td>2</td>
</tr>
<tr>
<td>Scenarios</td>
<td>4</td>
<td>—</td>
<td>—</td>
<td>5</td>
</tr>
<tr>
<td>Video samples</td>
<td>600</td>
<td>150</td>
<td>593</td>
<td>900</td>
</tr>
<tr>
<td>Resolution</td>
<td>160 × 120</td>
<td>variable</td>
<td>640 × 480</td>
<td>320 × 240</td>
</tr>
</tbody>
</table>

Figure 4.5: Overview of datasets.

4.5 Experiments

We used four benchmark datasets for the experiments: KTH action database [SLC04], UCF sports action dataset [RAS08], CK+ facial expression dataset [LCK+10], and Cambridge hand gesture dataset [KC09]. An overview of the datasets shown in Figure 4.5.

Unless otherwise stated, no pre-processing was performed in the input sequences and all the recognition results were obtained using 5-fold cross validation to divide the samples into training and testing sets. In all cases we used the following parameters: 0.95 detection rate, 0.95 false positive rejection rate, 0.5 margin.

Furthermore, since the search space of spatio-temporal windows is very large, we restricted the minimum size of the windows, as well as the minimum increment on location and size of the windows, to $\frac{1}{8}$ of the frame size. Note that smaller window sizes and steps can lead to improved recognition results but the training time increases dramatically due to the vast number of possible windows.

We first performed two preliminary experiments to justify our parameter selection, and then a comparative evaluation against baseline approaches as well as several state-of-the-art methods in the literature, using the four datasets.
4.5.1 Preliminary experiments

Here we present two preliminary experiments to justify the two most important parameters of our action and gesture recognition method: (i) the feature combination used for the construction of the Cov3D descriptors, and (ii) the Riemannian to Euclidean space mapping options.

Since the feature choice affects individual binary classifiers, we show results per classifier with detection error trade-off curves. The curves plot miss rate (MR) against false positives per window (FPPW) in a log-log scale. MR and FPPW are described in Section 3.4. We chose the one-vs-one classifiers between conflicting class pairs (where samples of one class are misclassified as the other class) on the challenging Cambridge hand gesture recognition dataset. Each point on a curve represents the average of all the chosen classifiers. The curves were generated by varying the classification threshold $\tau$ in Algorithm 4.2. For all curves the best performance point is at the bottom-left corner of the graph (i.e., minimal error rates).

Comparison of Cov3D features

As mentioned in Section 4.2.2, we construct the Cov3D descriptors using a combination of gradient and optical flow features. Gradient features alone are good for texture discrimination while optical-flow features are better for motion pattern analysis. However, the Cov3D descriptors include the correlation of both types of features with spatial and temporal location. Therefore, it is possible that both feature types end up encoding information on both texture and motion, providing possibly redundant information. For this reason, we compare the performance of the mixed feature approach against gradient or optical-flow based features only.

The results are shown in Figure 4.6. In all cases, using mixed features leads to better results as the interaction between gradients and motion adds rich information which is useful for action and gesture recognition. This result confirms the observations in [WUK+09].
Comparison of mapping approaches

In Figure 4.7, we compare the following six Riemannian to Euclidean space mapping \((Sym_d^+ \rightarrow \mathbb{R})\) approaches which can be used during boosting: (i) no mapping \((i.e., \text{using a vectorised representation of the upper-triangle of the covariance matrix})\), (ii) projection to a fixed tangent space \([GIK10]\), (iii) projection to the weighted Karcher mean of the samples \([TPM08]\), (iv) projection using the \(k\)-tangent spaces model from Section 3.3.2, (v) mapping the points with the original RLPP method \([HSWL12]\), and (vi) mapping the points with the proposed \(WRLPP\) approach.

Similar to the previous experiment, since the mapping approach affects individual binary classifiers, we show results per classifier with detection error trade-off curves. With the exception of the original RLPP method, incrementally better results are obtained by using the mapping approaches in the mentioned order, as they provide increasingly better vector representations of the manifold space. Although RLPP is designed to provide a better representation compared to tangent-based approaches, it appears not to be appropriate for boosting as it does not take into account the sample weights of critical training points. The proposed \(WRLPP\) method addresses this problem, resulting in the best overall performance.
4.5.2 Comparative evaluation

We compared the proposed approach against several notable action and gesture recognition methods in the literature. We restricted the selection to methods with reported performance on one of the four datasets in Figure 4.5.

KTH action dataset

The KTH human motion dataset [SLC04] contains six types of human actions (see Table 4.2), performed several times by 25 subjects in four scenarios: outdoors, outdoors with scale variation, outdoors with different clothes, and indoors. We first run an automatic pre-processing step to track and stabilise the video sequences so that all of the figures appear in the centre of the field of view.

We compared the Cov3D approach against the following methods. Spatial-temporal words (STW) [NWFF08], bag of words in conjunction with multiple kernel learning (BoW-MKL) [BXG12], and tensor canonical correlation analysis (TCCA) [KC09]. In STW, a video sequence is represented by a set of spatial-temporal words, extracted from space-time interest points. The algo-
Chapter 4. Action and Gesture Recognition

<table>
<thead>
<tr>
<th>Method</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>STW [NWFF08]</td>
<td>83.33%</td>
</tr>
<tr>
<td>BoW-MKL [BXG12]</td>
<td>94.33%</td>
</tr>
<tr>
<td>TCCA [KC09]</td>
<td>95.33%</td>
</tr>
<tr>
<td>Cov3D</td>
<td>98.61%</td>
</tr>
</tbody>
</table>

Table 4.1: Average correct recognition rate on the KTH [SLC04] dataset.

<table>
<thead>
<tr>
<th></th>
<th>Boxing</th>
<th>Hand clapping</th>
<th>Hand waving</th>
<th>Jogging</th>
<th>Running</th>
<th>Walking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boxing</td>
<td>96.7</td>
<td>3.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Hand clapping</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Hand waving</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Jogging</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>96.7</td>
<td>0</td>
<td>3.3</td>
</tr>
<tr>
<td>Running</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>Walking</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.7</td>
<td>0</td>
<td>98.3</td>
</tr>
</tbody>
</table>

Table 4.2: Confusion matrix (in %) on the KTH [SLC04] dataset.

algorithm then utilises latent topic models to learn the probability distributions of the spatial-temporal words. BoW-MKL exploits global spatio-temporal distribution of interest points by extracting holistic features from clouds of interest points, accumulated over multiple temporal scales. The extracted features are fused using MKL. TCCA is an extension of canonical correlation analysis (a principled tool to inspect linear relations between two sets of vectors) to tensor spaces and measures video-to-video volume similarity. The classification results are reported in Table 4.1. Our approach shows the best recognition accuracy. The confusion matrix between actions in the KTH dataset is shown in Table 4.2.

UCF sport dataset

The UCF sport action dataset [RAS08] consists of ten categories of human actions, containing videos with non-uniform backgrounds where both the camera and the subject might be moving. We use the regions of interest provided with the dataset.

We compared the Cov3D approach against the following methods: HOG3D [WUK+09], hierarchy of discriminative space-time neighbourhood features (HDN) [KG10], and augmented features in conjunction with multiple kernel learning (AF-MKL) [WXDL11]. HOG3D is the extension of histogram of oriented gradient descriptor [LMSR08] to the spatio-temporal case. HDN learns
Chapter 4. Action and Gesture Recognition

<table>
<thead>
<tr>
<th>Method</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>HOG3D [WUK+09]</td>
<td>85.60%</td>
</tr>
<tr>
<td>HDN [KG10]</td>
<td>87.27%</td>
</tr>
<tr>
<td>AF-MKL [WXDL11]</td>
<td>91.30%</td>
</tr>
<tr>
<td>Cov3D</td>
<td>93.91%</td>
</tr>
</tbody>
</table>

Table 4.3: Average recognition rate on the UCF dataset [RAS08].

<table>
<thead>
<tr>
<th>Diving</th>
<th>Golf swing</th>
<th>Kicking</th>
<th>Horse riding</th>
<th>Running</th>
<th>Skate-boarding</th>
<th>Pommel horse</th>
<th>Horizontal bar</th>
<th>Walking</th>
<th>Weight-lifting</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>0</td>
<td>20</td>
<td>0</td>
<td>10</td>
<td>70</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>13.3</td>
<td>0</td>
<td>0</td>
<td>80</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.4: Confusion matrix (in %) on the UCF [RAS08] dataset.

shapes of space-time feature neighbourhoods that are most discriminative for a given action category. The idea is to form new features composed of the neighbourhoods around the interest points in a video. AF-MKL exploits appearance distribution features and spatio-temporal context features in a learning scheme for action recognition. As shown in Table 4.3, the proposed Cov3D-based approach achieves the highest accuracy. The confusion matrix between sports in the UCF dataset is shown in Table 4.4.

**CK+ facial expression dataset**

The extended Cohn-Kanade (CK+) facial expression database [LCK+10] contains 593 sequences from 123 subjects. We used the sequences with validated emotion labels, among 7 possible emotions. The image sequences vary in duration (i.e., 10 to 60 frames) and incorporate the onset (which is also the neutral frame) to peak formation of the facial expressions.

We compared the Cov3D method against the baseline approach introduced with the dataset [LCK+10], constrained local models (CLM) [CLL+11], and temporal modelling of shapes (TMS) [JHA11]. The baseline approach
Chapter 4. Action and Gesture Recognition

<table>
<thead>
<tr>
<th>Method</th>
<th>angry</th>
<th>contempt</th>
<th>disgust</th>
<th>fear</th>
<th>happy</th>
<th>sadness</th>
<th>surprise</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAM [LCK\textsuperscript{+}10]</td>
<td>75.0%</td>
<td>84.4%</td>
<td>94.7%</td>
<td>65.2%</td>
<td>100%</td>
<td>68.0%</td>
<td>96.0%</td>
</tr>
<tr>
<td>CLM [CLL\textsuperscript{+}11]</td>
<td>70.1%</td>
<td>52.4%</td>
<td>92.5%</td>
<td>72.1%</td>
<td>94.2%</td>
<td>45.9%</td>
<td>93.6%</td>
</tr>
<tr>
<td>TMS [JHA11]</td>
<td>76.7%</td>
<td>—</td>
<td>81.5%</td>
<td>94.4%</td>
<td>98.6%</td>
<td>77.2%</td>
<td>99.1%</td>
</tr>
<tr>
<td>Cov3D</td>
<td>94.4%</td>
<td>100%</td>
<td>95.5%</td>
<td>90.0%</td>
<td>96.2%</td>
<td>70.0%</td>
<td>100%</td>
</tr>
</tbody>
</table>

Table 4.5: Recognition rate per emotion on the CK+ dataset [LCK\textsuperscript{+}10].

<table>
<thead>
<tr>
<th></th>
<th>angry</th>
<th>contempt</th>
<th>disgust</th>
<th>fear</th>
<th>happy</th>
<th>sadness</th>
<th>surprise</th>
</tr>
</thead>
<tbody>
<tr>
<td>angry</td>
<td>94.4</td>
<td>0</td>
<td>5.6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>contempt</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>disgust</td>
<td>4.5</td>
<td>0</td>
<td>95.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>fear</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>0</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>happy</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3.8</td>
<td>96.2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>sadness</td>
<td>10</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>70</td>
<td>10</td>
</tr>
<tr>
<td>surprise</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.6: Confusion matrix (in \%) on the CK+ [LCK\textsuperscript{+}10] dataset.

uses active appearance models (AAM) to track the faces and extract the features, and then uses support vector machines (SVM) to classify the facial expressions. The CLM approach is an improvement on AAM, designed for better generalisation to unseen objects. The TMS approach uses latent-dynamic conditional random fields to model temporal variations within shapes.

We show the performance per emotion in Table 4.5, in line with existing literature. The proposed Cov3D approach achieves the highest average recognition accuracy of 92.3\% (averaged over the 7 classes). The next best method (TMS) obtained an average accuracy of 87.92\%. The confusion matrix between emotions in the CK+ dataset is shown in Table 4.6.

Cambridge hand gesture dataset

The Cambridge hand-gesture dataset [KC09] consists of 900 image sequences of 9 gesture classes. Each class has 100 image sequences performed by 2 subjects, captured under 5 illuminations and 10 arbitrary motions. The 9 classes are defined by three primitive hand shapes and three primitive motions. Each sequence was recorded with a fixed camera having roughly isolated gestures in space and time. We followed the test protocol defined in [KC09]. Sequences with normal illumination were considered for training while tests were performed on the remaining sequences.
The proposed method was compared against tensor canonical correlation analysis (TCCA) [KC09], product manifolds (PM) [LBK10] and tangent bundles (TB) [Lui12b]. TCCA is the extension of canonical correlation analysis to multiway data arrays or tensors. Canonical correlation analysis and principal angles are standard methods for measuring the similarity between subspaces. In the PM method a tensor is characterised as a point on a product manifold and classification is performed on this space. The product manifold is created by applying a modified high order singular value decomposition on the tensors and interpreting each factorised space as a Grassmann manifold [TVSC11]. In the TB method, video data is represented as a third order tensor and factorised using high order singular value decomposition, where each factor is projected onto a tangent space and the intrinsic distance is computed from a tangent bundle for action classification.

We report the recognition rates for the four test sets in Table 4.7, where the proposed Cov3D-based approach obtains the highest performance. The confusion matrix is shown in Table 4.8.

### Table 4.7: Average recognition rate on the Cambridge dataset [KC09].

<table>
<thead>
<tr>
<th>Method</th>
<th>Set1</th>
<th>Set2</th>
<th>Set3</th>
<th>Set4</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCCA [KC09]</td>
<td>81%</td>
<td>81%</td>
<td>78%</td>
<td>86%</td>
<td>82% (±3.5)</td>
</tr>
<tr>
<td>PM [LBK10]</td>
<td>89%</td>
<td>86%</td>
<td>89%</td>
<td>87%</td>
<td>88% (±2.1)</td>
</tr>
<tr>
<td>TB [Lui12b]</td>
<td>93%</td>
<td>88%</td>
<td>90%</td>
<td>91%</td>
<td>91% (±2.4)</td>
</tr>
<tr>
<td>Cov3D</td>
<td>92%</td>
<td>94%</td>
<td>94%</td>
<td>93%</td>
<td>93% (±1.1)</td>
</tr>
</tbody>
</table>

### Table 4.8: Confusion matrix (in %) on the Cambridge [KC09] dataset.

<table>
<thead>
<tr>
<th>class1</th>
<th>class2</th>
<th>class3</th>
<th>class4</th>
<th>class5</th>
<th>class6</th>
<th>class7</th>
<th>class8</th>
<th>class9</th>
</tr>
</thead>
<tbody>
<tr>
<td>class1</td>
<td>91.3</td>
<td>0</td>
<td>0</td>
<td>6.2</td>
<td>0</td>
<td>0</td>
<td>2.5</td>
<td>0</td>
</tr>
<tr>
<td>class2</td>
<td>0</td>
<td>90.0</td>
<td>0</td>
<td>0</td>
<td>2.5</td>
<td>0</td>
<td>0</td>
<td>7.5</td>
</tr>
<tr>
<td>class3</td>
<td>0</td>
<td>1.3</td>
<td>93.7</td>
<td>0</td>
<td>1.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>class4</td>
<td>1.3</td>
<td>0</td>
<td>98.7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>class5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>98.7</td>
<td>0</td>
<td>0</td>
<td>1.3</td>
<td>0</td>
</tr>
<tr>
<td>class6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>92.5</td>
<td>0</td>
<td>0</td>
<td>7.5</td>
</tr>
<tr>
<td>class7</td>
<td>6.25</td>
<td>0</td>
<td>6.3</td>
<td>0</td>
<td>0</td>
<td>87.5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>class8</td>
<td>0</td>
<td>6.2</td>
<td>0</td>
<td>0</td>
<td>1.3</td>
<td>0</td>
<td>0</td>
<td>92.5</td>
</tr>
<tr>
<td>class9</td>
<td>1.3</td>
<td>0</td>
<td>1.2</td>
<td>0</td>
<td>1.2</td>
<td>0</td>
<td>0</td>
<td>96.3</td>
</tr>
</tbody>
</table>
4.6 Extended Method with Hybrid Classification

For simplicity, in Section 4.4 we chose a combination of one-vs-one classifiers to create the multi-class classifiers used in the proposed action and gesture recognition method. We chose this simple approach to prove that our Cov3D descriptors and WRLPP mapping are the main contributors to the high recognition performance. In this section, we propose an extension of our recognition approach which combines one-vs-all and one-vs-one binary classifiers into a more robust hybrid multi-class classifier.

Below, in Section 4.6.1, we detail the hybrid multi-class classification approach. Then, in Section 4.6.2, we test the action and gesture recognition performance of the extended method.

4.6.1 Hybrid multi-class classification

Some actions or gestures are easier to discriminate than others. Using one-vs-all classifiers is fast and good for discriminating classes which are easily separable from the rest. One-vs-one classifiers are good for discrimination between pairs of conflicting classes, but are not appropriate for labelling examples which belong to classes other than the pair they were trained with. Therefore, we propose a hybrid approach where one-vs-all classifiers are first used to quickly select a set of candidate classes to which a sample may belong, and then one-vs-one classifiers are used to discriminate between the classes in the set to select the most likely output.

The hybrid classifier is as a matrix $H$ of $N_c \times N_c$ binary classifiers, where $N_c$ is the number of classes. The diagonal entries of the matrix are filled with one-vs-all classifiers and off-diagonal entries are filled with one-vs-one classifiers. For instance, the entry $H_{i,i}$ contains the one-vs-all classifier trained to discriminate class $i$ from the rest, while the entry $H_{i,j}$ contains the one-vs-one classifier trained to discriminate between classes $i$ and $j$ (i.e., $i$-vs-$j$ classifier).

The training of the hybrid classifier is therefore reduced to training all the binary classifiers (i.e., the entries in $H$), as described in Algorithm 4.3. Since the one-vs-all classifiers are designed for quick selection of easily discriminated classes, their training parameters can be less strict. Also, since the
Chapter 4. Action and Gesture Recognition

Algorithm 4.3: Training the hybrid classifier

**Input:** Training videos with labels \( \{(V_i, y_i)\}_{i=1}^{N_c} \) belonging to \( N_c \) classes

- Let \( H \) be a matrix of \( N_c \times N_c \) classifiers
- For \( i, j = 1, \ldots, N_c \)
  - if \( i = j \)
    * Train one-vs-all classifier \( C_{<i,i>} \) following Algorithm 4.2 with:
      \[ \{(V_k, y_k)\}_{k=1}^{N} \mid y_p = y_i \]
    * \( H_{i,j} \leftarrow C_{<i,i>} \)
  - else \( (i \neq j) \)
    * Train one-vs-one classifier \( C_{<i,j>} \) following Algorithm 4.2 with:
      \[ \{(V_k, y_k)\}_{k=1}^{N} \mid y_k \in \{y_i, y_j\}, y_p = y_i \]
    * \( H_{i,j} \leftarrow C_{<i,j>} \)

**Output:** The hybrid classifier \( H \)

The majority of the computation is done by these classifiers and they are exposed to more training samples, the one-vs-all classifiers can be trained in a cascaded fashion such as [TPM08]. If all the one-vs-one classifiers are trained using the same parameters, only half their number needs to be trained since the \( i \)-vs-\( j \) classifier will be the same as the \( j \)-vs-\( i \) classifier and \( H \) will be symmetric.

Once the hybrid classifier has been trained, a sample video \( V \) can be classified in the following way. First, the sample is evaluated using all the one-vs-all classifiers. If the output is positive for only one of the classifiers, for instance \( H_{i,i} \), \( V \) is labelled as class \( i \). Otherwise, a set of candidate classes, \( S \), is created from all the classes whose one-vs-all classifiers evaluated to positive\(^6\). Then, the sample is evaluated using all the one-vs-one classifiers between pairs of the classes in \( S \). Finally, \( V \) is labelled as the class with the highest evaluation sum, as described in Algorithm 4.4.

In terms of the amount of binary classifications, the minimum computational complexity of our hybrid approach is \( O(N_c) \), as every one-vs-all classifier has to be evaluated in the first stage. In the worst case, where all classifiers in the matrix have to be evaluated, the approach is \( O(N_c^2) \). However, typically only a few one-vs-all classifiers evaluate to positive and we expect

\(^6\)If none of the classifiers evaluates to positive then \( S \) will contain all the classes.
Algorithm 4.4: Predicting with the hybrid classifier

**Input:** A sample video $V$ and the hybrid classifier $H$

- Initialise the set of candidate classes $S = \emptyset$
- For $i = 1, \ldots, N_c$
  - Get the one-vs-all classifier $C_{<i,i>} = H_{i,i}$
  - if $C_{<i,i>}(V) > 0$, $C_{<i,i>}()$ defined in (4.27)
    - $S \leftarrow S \cup \{i\}$
- if $|S| = 1$
  - $c = S(1)$
- else
  - Let $S = \{i\}_{i=1}^{N_c}$ if $|S| = 0$
  - Initialise $p_i = 0$
  - For $i, j \in S, i \neq j$
    - Get the one-vs-one classifier $C_{<i,j>} = H_{i,j}$
    - $p_i \leftarrow p_i + C_{<i,j>}(V), C_{<i,j>}()$ defined in (4.27)
  - $c = S(\arg \max_i(p_i))$

**Output:** The class label of $C_{<c,c>} = H_{c,c}$

the amortised complexity to be closer to $O(N_c)$.

### 4.6.2 Experiments

In this section we compare various ways of producing a multi-class classifier from binary classifiers. We consider the hybrid classification approach as well as one-vs-all and one-vs-one classification. The one-vs-all approach consists on training one binary classifier for each class using the samples of all other class as negatives. A test example is classified with the same class of the classifier which evaluates to the higher probability using (4.27). The one-vs-one approach consist on training a classifier for each pair of classes as described in Section 4.4.

Figure 4.8 compares the recognition performance of the three classification approaches on the four datasets introduced in Section 4.5. In all cases the combination of one-vs-one binary classifiers yields better results than simply selecting the best one-vs-all classifier, and further improvement
Chapter 4. Action and Gesture Recognition

![Comparison of multi-class classification approaches.](image.png)

### Table 4.9: Confusion matrix (in %) on the KTH \([\text{SLC04}]\) dataset.

<table>
<thead>
<tr>
<th></th>
<th>Boxing</th>
<th>Hand clapping</th>
<th>Hand waving</th>
<th>Jogging</th>
<th>Running</th>
<th>Walking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boxing</td>
<td>96.7</td>
<td>3.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Hand clapping</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Hand waving</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Jogging</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>96.7</td>
<td>0</td>
<td>3.3</td>
</tr>
<tr>
<td>Running</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>Walking</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
</tr>
</tbody>
</table>

is obtained using the hybrid approach. However, it is important to note that even with the worse classification approach (i.e., one-vs-all), the recognition accuracy still outperforms related state-of-the-art algorithms. Therefore, although better results are obtained with the hybrid approach, the choice of multi-class classifier is secondary to the descriptor and mapping choices.

We also provide confusion matrices on the KTH (Table 4.9), UCF (Table 4.10), CK+ (Table 4.11), and Cambridge (Table 4.12) datasets, using the hybrid classification approach. All the results show improvements compared to the confusion matrices for the one-vs-one classification approach presented in Section 4.5.
Chapter 4. Action and Gesture Recognition

<table>
<thead>
<tr>
<th>Diving</th>
<th>Golf swing</th>
<th>Kicking</th>
<th>Horse riding</th>
<th>Running</th>
<th>Skateboarding</th>
<th>Pommel horse</th>
<th>Horizontal bar</th>
<th>Walking</th>
<th>Weightlifting</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Golf swing</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Kicking</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Horse riding</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Running</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>Skateboarding</td>
<td>0</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>80</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Pommel horse</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Horizontal bar</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Walking</td>
<td>0</td>
<td>6.7</td>
<td>0</td>
<td>0</td>
<td>6.7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>86.6</td>
</tr>
<tr>
<td>Weightlifting</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 4.10: Confusion matrix (in %) on the UCF [RAS08] dataset.

<table>
<thead>
<tr>
<th>angry</th>
<th>contempt</th>
<th>disgust</th>
<th>fear</th>
<th>happy</th>
<th>sadness</th>
<th>surprise</th>
</tr>
</thead>
<tbody>
<tr>
<td>angry</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>contempt</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>disgust</td>
<td>4.5</td>
<td>0</td>
<td>95.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>fear</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>0</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>happy</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>sadness</td>
<td>10</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>80</td>
<td>12.5</td>
</tr>
<tr>
<td>surprise</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.11: Confusion matrix (in %) on the CK+ [LCK+10] dataset.

<table>
<thead>
<tr>
<th>class1</th>
<th>class2</th>
<th>class3</th>
<th>class4</th>
<th>class5</th>
<th>class6</th>
<th>class7</th>
<th>class8</th>
<th>class9</th>
</tr>
</thead>
<tbody>
<tr>
<td>91.3</td>
<td>0</td>
<td>6.2</td>
<td>0</td>
<td>0</td>
<td>2.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>90</td>
<td>0</td>
<td>0</td>
<td>2.5</td>
<td>0</td>
<td>0</td>
<td>7.5</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1.3</td>
<td>93.7</td>
<td>0</td>
<td>1.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3.7</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>98.7</td>
<td>0</td>
<td>0</td>
<td>1.3</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>92.5</td>
<td>0</td>
<td>0</td>
<td>7.5</td>
</tr>
<tr>
<td>6.25</td>
<td>0</td>
<td>0</td>
<td>6.3</td>
<td>0</td>
<td>0</td>
<td>87.5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>6.2</td>
<td>0</td>
<td>0</td>
<td>1.3</td>
<td>0</td>
<td>0</td>
<td>92.5</td>
<td>0</td>
</tr>
<tr>
<td>1.3</td>
<td>0</td>
<td>0</td>
<td>1.2</td>
<td>0</td>
<td>1.2</td>
<td>0</td>
<td>0</td>
<td>96.3</td>
</tr>
</tbody>
</table>

Table 4.12: Confusion matrix (in %) on the Cambridge [KC09] dataset.

4.7 Main Findings and Future Directions

In this chapter, we first extended the flat covariance descriptors proposed in [TPM08] to spatio-temporal covariance descriptors termed Cov3D, and then showed how they can be computed quickly through the use of integral video representations.

The proposed Cov3D descriptors belong to the group of symmetric positive definite matrices, which can be formulated as a connected Riemannian man-
ifold. Prior to classification, points on a manifold are generally mapped to a Euclidean space, through a technique such as Riemannian locality preserving projection (RLPP) [HSLW12].

The Cov3D descriptors are extracted from spatio-temporal windows inside sample videos, with the number of possible windows being very large. We used a boosting approach to find a subset which is the most useful for classification. In order to take into account the weights of the training samples, we further proposed to extend RLPP by incorporating weighting during the projection. The weighted projection (termed WRLPP) leads to a better representation of the neighbourhoods around the most critical training samples during each boosting iteration.

Combining the proposed Cov3D descriptors with the classification approach based on WRLPP boosting leads to a state-of-the-art method for action and gesture recognition. The proposed Cov3D-based method performs better than several recent approaches on four benchmark datasets for action and gesture recognition. The method is robust and does not require additional processing of the videos, such as foreground detection, interest-point detection or tracking. To our knowledge, this is the first approach proving to be equally suitable (i.e., > 90% recognition accuracy) for both action and gesture recognition.

Additionally, we proposed an extension of the method which uses a hybrid classification approach. One-vs-all and one-vs-one binary classifiers are combined to produce a fast and robust multi-class classifier. We repeated the experiments on the four datasets and showed improvements in all the recognition results after using the hybrid classification approach.

Further avenues of research include: extending the algorithm for action and gesture detection [NHH07, SI07]; adapting the method for related tasks, such as anomaly detection in surveillance videos [MLBV10, RSL11], where there is often a shortage of positive examples; and exploring more complex information fusion techniques [SP04] as alternatives to the hybrid multi-class classification approach.
Chapter 5

Conclusion

In this thesis we contribute towards automatic visual surveillance. In Figure 1.2 we introduced a series of modules which can be present in surveillance systems. We make individual contributions to three of the modules: (i) foreground extraction, (ii) object detection, and (iii) scene analysis; by working in the following respective applications: (i) shadow detection and removal, (ii) pedestrian detection, and (iii) action and gesture recognition.

For these applications we propose algorithms with novel features and machine learning approaches that outperform current state-of-the-art approaches. Although the experiments target automatic surveillance, the proposed algorithms could also be used in related computer vision tasks, in particular those involving automatic analysis of digital image and video information.

Apart from gathering the individual contributions to develop an automatic visual surveillance system, as part of future research, the proposed algorithms can be further extended or modified to handle new features and requirements. In the following sections we summarise the contributions, possible extensions, and future research directions per application.

5.1 Shadow Detection and Removal

We present a state-of-the-art shadow detection and removal method with the best shadow detection performance to date. Along with the proposed algorithm, we include a thorough review of shadow detection methods published
Chapter 5. Conclusion

in the last decade, categorised in a feature-based taxonomy. The review is followed by a detailed qualitative and quantitative comparative evaluation of the proposed method and a selection of notable reviewed approaches. Additionally, we provide a dedicated dataset for the evaluation of moving cast shadow detection as well as C++ source code for the evaluated methods.

5.1.1 Future work

- A physical shadow detection method, such as [HC09], tuned for high detection rate can be used for a more accurate selection of candidate shadow pixels.

- With multi-scale processing [OH07], gradient directions are calculated for blocks of multiple resolutions instead of single pixels. This can be used to filter out the effect of unreliable gradients that may arise from pixel-level noise, and to preserve gradients that may disappear at the pixel level when attenuated by shadows.

- A more robust procedure for calculating the region level texture correlation can be used, such as comparing histograms of oriented gradients [DT05]. This would be particularly useful in the presence of a shaking camera where pixel-based gradient correlation may fail.

5.2 Pedestrian Detection

We present a state-of-the-art pedestrian detection algorithm which shows superior performance in two standard datasets. Along with the proposed algorithm, we include the \textit{k-tangent spaces} model, a novel way of efficiently mapping manifold data into Euclidean data resulting in improving learning. This can be useful for any application that involves data represented as manifold points where both representation quality and speed are important.

5.2.1 Future work

- The \textit{k-tangent spaces} model can be adapted to more general classification problems (\textit{i.e.}, not tied to covariance matrices or pedestrian detec-
(function), as long as the samples can be represented as points on Riemannian manifolds.

- Alternative ways of fusing information [SP04, Bis06] can be explored to replace our basic probabilistic mixture used to create the \( k \)-tangent spaces model from individual tangent spaces.

### 5.3 Action and Gesture Recognition

We present a state-of-the-art algorithm which shows superior performance in four challenging datasets. We propose a novel representation of spatio-temporal data called Cov3D, which is based on covariance descriptors. Along with the Cov3D descriptors we include a method for their fast calculation based on integral videos. We also propose a novel manifold to Euclidean mapping approach called weighted Riemannian locality preserving projection (WRLPP). Additionally, we suggest an extended version of the method with a hybrid multi-class classifier, combining one-vs-all and one-vs-one classifiers for improved recognition results.

#### 5.3.1 Future work

- The algorithm can be extended to perform action and gesture detection [NHH07, SI07].

- The method can also be adapted for related tasks, such as anomaly detection in surveillance videos [MLBV10, RSL11], where there is often a shortage of positive examples.

- More complex information fusion techniques [SP04] can be explored as alternatives to the hybrid multi-class classification approach.
Bibliography


Bibliography


Bibliography


Bibliography


<table>
<thead>
<tr>
<th>Reference</th>
<th>Authors</th>
<th>Title</th>
<th>Conference/Journal Details</th>
</tr>
</thead>
</table>


Bibliography


Bibliography


[SWTO04] C. Shan, Y. Wei, T. Tan, and F. Ojardias. Real time hand tracking by combining particle filtering and mean shift. In *IEEE Inter-
Bibliography


<table>
<thead>
<tr>
<th>Reference</th>
<th>Authors</th>
<th>Title</th>
<th>Conference/Journal</th>
<th>Pages/Volume</th>
</tr>
</thead>
</table>


