HAL
open science

Multiclass object recognition for driving assistance systems and video surveillance

Fatin Zaklouta

To cite this version:

Fatin Zaklouta. Multiclass object recognition for driving assistance systems and video surveillance. Automatic. École Nationale Supérieure des Mines de Paris, 2011. English. NNT : 2011ENMP0045 . pastel-00657727

HAL Id: pastel-00657727

https://pastel.hal.science/pastel-00657727v1

Submitted on 9 Jan 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY

MINES ParisTech

École doctorale n°432 : Sciences des Métiers de l'Ingénieur

Doctorat ParisTech

T H È S E

pour obtenir le grade de docteur délivré par

l'École nationale supérieure des mines de Paris

Spécialité « Informatique temps réel, robotique et automatique »

présentée et soutenue publiquement par

Fatin ZAKLOUTA

Decembre 2011

Reconnaissance d'objets multiclasses pour des applications d'aide à la conduite et de vidéo surveillance

Multiclass Object Recognition for Driving Assistance Systems and Video Surveillance

Directeur de thèse : **Fawzi NASHASHIBI**
Co-encadrement de la thèse : **Bogdan STANCIULESCU**

Jury

Mme Bernadette Dorizzi	Rapporteur
M. Christian Igel	Rapporteur
M. Patrick Siarry	Examinateur
M. Bogdan Stanciulescu	Examinateur
M. Fawzi Nashashibi	Examinateur

T H È S E

MINES ParisTech
Centre de Robotique
60 Boulevard Saint Michel, 75006 Paris

Abstract

Traffic Sign Recognition (TSR) and pedestrian detection are important components of an Advanced Driver Assistance System (ADAS). The former enhances the safety by informing the driver of speed limits, road regulations and imminent dangers, such as icy roads or pedestrian crossings. The latter warns the driver of the presence of a pedestrian to avoid a potentially dangerous situation. These two ADAS functionalities are particularly important when the driver is tired, distracted or affected by poor visibility conditions. The pedestrian detection is also common to the video surveillance systems and is used to warn residents of intruders. In this thesis, we focus on TSR and pedestrian detection for ADAS and video surveillance applications.

We propose a novel three stage approach for TSR, consisting of a color segmentation, a shape detection and a content classification phase. The segmentation is based on the color enhancement and an adaptive threshold. It extracts the relevant red regions in the image and hereby reduces the search space. The shape detection is performed by a linear Support Vector Machine (SVM) trained on Histograms of Oriented Gradients (HOG). The extracted circular and triangular signs are passed on to the tree classifiers in the third phase. These identify the content of the traffic sign. In our experiments, a recall and precision rate of 90% are attained.

The choice of the classifier depends on the data set at hand. The K-d tree excels on the smaller data set, yet are outperformed by the Random Forest on the larger one. The performance of the K-d tree classifier was improved by up to 20% when introducing the spatial weighting of the feature vectors. It yields an accuracy of 80% on our video sequences. The Random Forest outperforms the K-d tree on the larger German Traffic Sign Recognition Benchmark, achieving a classification rate of 97%.

Moreover, we examine three aspects of the classification process: the features describing the objects, the metrics used to compare them in the classifiers and the combination of these classifiers. The selection of pertinent features using Random Forests and Fisher's Criterion reduces the feature space as well as the memory and processing requirements, while retaining a high classification accuracy.

The choice of the metric has a significant effect on the performance of the Nearest Neighbor classifer. We determine that the correlation is the most suitable metric for the HOG features. The correlation was used to improve the pedestrian detection in a video surveillance application by eliminating immobile false detections, such as trees. The precision is doubled, without affecting the recall rate.

The combination of the tree classifiers with the HOG/linear SVM detector into a heterogenous mixture of experts improves the performance of the pedestrian detection algorithm in ADAS. The false positives per frame rate is decreased by about half, while maintaining a high recall rate.

Résumé

Dans cette thèse, nous abordons la détection des piétons et la reconnaissance des panneaux. Cette étude a pour applications l'aide à la conduite (anglais: *Advanced Driver Assistance Systems (ADAS)*) et la vidéo surveillance.

Nous examinons, d'abord, trois aspects du processus de classification: les caractéristiques qui décrivent les objets, les paramètres utilisés pour les comparer et la combinaison des différents classifieurs. La sélection des caractéristiques pertinentes est faite en utilisant des Random Forest et le critère de Fisher. Ce dernier réduit la dimension des caractéristiques ainsi que la mémoire et le temps de calcul, tout en conservant une précision de classification élevée.

Un des points faibles des systèmes de classification sont les fausses alarmes. Dans une application de vidéo surveillance avec des caméras statiques, la plupart des fausses alarmes sont des objets fixes, comme les lampadaires. Celles-ci sont éliminées grâce à la corrélation sur plusieurs trames. Dans le cadre d'ADAS, les fausses alarmes récurrentes sont supprimées par un filtre complémentaire en forme d'arbre. La combinaison des classifieurs d'arbre avec les histogrammes des gradients orientés (en anglais: *Histogram of Oriented Gradients (HOG)*) et le séparateur à vaste marge (en anglais: *Support Vector Machines (SVM)*) linéaires améliore la performance de l'algorithme de détection des piétons. Le taux de faux positifs par image est diminué de moitié, tout en gardant le taux de rappel.

Pour la reconnaissance des panneaux, nous proposons une nouvelle approche composée de trois étapes: une segmentation de couleur, une détection de forme et finalement la classification de contenu. La segmentation est basée sur la technique de color enhancement et un seuil adaptatif. De cette manière, les régions rouges de l'image sont extraites, mais l'espace de recherche est réduit. La détection de forme est effectuée par un SVM avec les caractéristiques de HOG. Les panneaux circulaires et triangulaires extraits sont ensuite transmis aux arbres de classifications qui identifient le contenu des panneaux. Un taux de rappel et précision de 90% est atteint sur nos séquences de film.

Finalement, nous avons testé les classifieurs sur deux bases de données de panneaux routières de différentes tailles et répartitions des classes. Le *K-d tree* donne de meilleurs résultats sur la plus petite de nos bases. La performance a été améliorée jusqu'à 20% lors de l'implémentation de la pondération spatiale des vecteurs des caractéristiques et une précision de 80% est atteinte sur nos séquences. Par contre, la performance du classifieur *Random Forest* est supérieure sur la grande base *German Traffic Sign Recognition Benchmark*, avec un taux de classification de 97%.

Contents

1 Introduction **3**

1.1 Context 3

1.2 Contributions 4

1.3 Outline 4

2 Learning: From Characteristic to Generic **7**

2.1 Describing the Object 8

2.1.1 Local Features 8

2.1.2 Global Features 12

2.2 Classifying the Object 14

2.2.1 Binary Adaboost 14

2.2.2 Multiclass Adaboost 15

2.2.3 Support Vector Machines 20

2.2.4 Tree Classifiers 25

2.3 Comparing on Benchmarks 28

2.3.1 Data Sets 28

2.3.2 Performance Evaluation 29

2.4 Conclusions 32

3 Classification: Identifying the Unknown **35**

3.1 Importance of the Feature 36

3.1.1 Feature Ranking Techniques 36

3.1.2 Evaluation 39

3.2 Importance of the Metric 41

3.2.1 Similarity Metrics 41

3.2.2 Comparison on Benchmark Data Sets 43

3.2.3 Correlation for Video Surveillance 44
3.3 Mixture of Experts 47
3.3.1 Pedestrian Detection Techniques 47
3.3.2 Our Approach for False Alarm Elimination 49
3.4 Conclusions and Perspectives 52

4 Real-Time Traffic Sign Recognition 55
4.1 Existing Approaches 56
4.1.1 Determining the Location using Segmentation 58
4.1.2 Category Determination using Shape Detection 58
4.1.3 Classification Techniques 59
4.2 Our Approach 62
4.2.1 Image Segmentation 62
4.2.2 Category detection 64
4.2.3 Sign classification 70
4.3 Overall Performance of the TSR System 78
4.4 Feature Space Reduction 79
4.4.1 Feature Selection 79
4.4.2 Evaluation 81
4.5 Conclusions and Perspectives 81

5 Conclusions and Perspectives 85
5.1 Summary 85
5.1.1 Three Aspects of Classification 85
5.1.2 Real-Time Traffic Sign Recognition System 86
5.2 Future Work 86

A Courses and Publications 89
A.1 Publications 89
A.2 Courses 90

List of Figures

1.1 Eliminating false alarms for pedestrian detection 4
1.2 Example of traffic sign recognition. 5

2.1 Examples of Haar-like features 8
2.2 Integral Image representation 9
2.3 Evaluation of Control Points 9
2.4 Examples of control points 10
2.5 Examples of Connected Control Points 10
2.6 Structure of SIFT interest points 11
2.7 Structure of SURF interest points 11
2.8 Gradient responses used in SURF. 12
2.9 Structure of HOG descriptor 13
2.10 Structure of PHOG descriptor 13
2.11 Distance Transform 14
2.12 Adaboost training and classification 15
2.13 Hamming and Loss-based decoding 17
2.14 Linear SVM 21
2.15 Slack variable in SVM 23
2.16 Data transformation using "Kernel trick" 23
2.17 Example of a 2-dimensional K-d tree. 26
2.18 Classification of a sample x in Random Forest 27
2.19 The ETH80 data set 28
2.20 Average images of Caltech 101 classes. 29

3.1 Feature selection process 37
3.2 Feature ranking using Fisher's Criterion on ETH80 data set. 40

3.3 Effect of reducing the feature dimension on SVM accuracy on ETH80. . . . 41
3.4 Feature Ranking of PHOG L0-3 using Fisher's Criterion. 42
3.5 Reducing the PHOG $L3$ feature dimension on SVM accuracy on Caltech 101 43
3.6 Reducing the $L0-3$ feature dimension on SVM accuracy on Caltech 101 . 44
3.7 Effect of the correlation threshold on recall and precision 46
3.8 False alarms eliminated by HOG correlation 46
3.9 Mixture of Experts 48
3.10 Examples of HOG descriptors 50
3.11 False alarms filtered by K-d tree 51
3.12 Pedestrians misclassified by K-d tree 52

4.1 Examples for difficulties facing the Traffic Sign Recognition (TSR) task . . . 56
4.2 Three stage approach for TSR. 62
4.3 Image segmentation for TSR 63
4.4 Examples of traffic sign detection using HOG/linear SVM. 64
4.5 Recall and precision for different HOG detector sizes. 67
4.6 Recall and precision for different HOG detector sizes. 68
4.7 Using the segmentation mask to find ROIs 69
4.8 Effect of segmentation on RGB HOG detection results. 71
4.9 Tight bounding box for HOG 4 feature 73
4.10 Confusion matrices for GTSRB with and without spatial weighting 75
4.11 Comparison of DTs using Bottomhat and Canny 78
4.12 Variable Importance using Random Forests 81
4.13 Feature ranking using Fisher Scores 82
4.14 Effect of feature space reduction on classification accuracy 83

List of Tables

2.1 Class Coloring Codewords 17
2.2 ETH 80 classification results 30
2.3 Classification results on Caltech 101 data set 32

3.1 K-d tree accuracy on ETH80 data set. 44
3.2 K-d tree accuracy on Caltech 101 data set 45
3.3 HOG similarity measures for eliminating static false alarms. 47
3.4 Tree filtering on five test sequences 51

4.1 State-of-the-art segmentation techniques 57
4.2 State-of-the-art detection techniques 60
4.3 State-of-the-art detection techniques 61
4.4 Class sizes of TSR dataset 66
4.5 Effect of segmentation masks on HOG/lin. SVM detection 70
4.6 Parameters of the HOG descriptors for GTSRB 70
4.7 Classification results when varying E_{max} in K-d tree 72
4.8 Effect of similarity measure on K-tree classification 73
4.9 Spatial weighting for 5x5 non-overlapping blocks. 74
4.10 Classification results: K-d tree with spatially weighted HOG 74
4.11 Effect of Spatial Weighting on K-d tree classification 76
4.12 Varying Random Forest Parameters 77
4.13 Overview of classifier accuracies 77
4.14 Classification results using Distance Transform 78
4.15 Overall performance of TSR system 80
4.16 Feature Space Reduction using Fisher's Criterion and Random Forests . . 83

List of Algorithms

1 Binary Adaboost 16
2 Adaboost.M2 16
3 Adaboost.OC 18
4 Adaboost.ECC 19
5 Adaboost.ERP 20
6 Adaboost.ERC 20
7 Determining Variable Importance using Random Forests 39

Chapitre 1

Introduction

Ce chapitre présente le contexte de cette thèse, ses domaines d'applications, ses objectifs ainsi que les contraintes associées. Il donne également un aperçu des contributions scientifiques originales de ce manuscrit.

Les systèmes d'aide à la conduite (en anglais: *Advanced Driver Assistance Systems, ADAS*) améliorent la sécurité du conducteur sur la route. Plusieurs types de capteurs peuvent être utilisés, comme un laser, un lidar ou une caméra. Leurs principales fonctionnalités sont la navigation, la détection de marquages au sol, la détection de piétons et la reconnaissance des panneaux. Cette thèse porte sur ces deux dernières, car elles sont encore à l'heure actuelle perfectibles.

Les trois grandes difficultés pour un système de détection des panneaux sont: i) la mauvaise visibilité due à la basse résolution des images utilisées, à la météo, ou à une faible illumination, ii) la rotation, occlusion ou détérioration des panneaux et iii) les contraintes sur la capacité mémoire ou la puissance de calcul disponible et la nécessité d'un système temps réel pour les dispositifs embarqués. D'autre part, la détection des piétons amène des difficultés supplémentaires comme les mouvements intrinsèques des personnes à repérer et la diversité de leurs vêtements, tailles et postures.

Étant donné que le point faible des approches existantes pour la détection est un taux élevé de fausses alarmes, nous proposons deux méthodes pour filtrer les candidats extraits et améliorer la précision. La première approche réduit le taux de fausses alarmes dans une application de vidéo-surveillance avec des caméras fixes en éliminant les candidats statiques. La deuxième approche améliore les systèmes de l'aide à la conduite en supprimant les fausses alarmes récurrentes comme les arbres et les lampadaires.

Finalement, nous proposons un système complet pour la reconnaissance de panneaux de signalisation. Celui-ci se compose d'une segmentation d'image, suivie d'une détection et d'une classification. La première étape réduit l'espace de recherche tandis que la seconde en extrait les candidats triangulaires et circulaires. Ces panneaux potentiels sont analysés par l'étape suivante de classification où ils sont identifiés par des *K-d tree* et les *Random Forests*. Pour réduire les besoins en mémoire et en calcul, nous examinons deux techniques de sélection des caractéristiques: les *Random Forests* et *Fisher's Criterion*.

Le chapitre 2 présente l'état de l'art des caractéristiques et des algorithmes de classification. Le chapitre 3 évalue trois aspects de la classification: l'importance des caractéristiques, la métrique utilisée dans les classifieurs pour comparer les caractéristiques et la combinaison de différents classifieurs. Un système complet pour la reconnaissance des panneaux routiers est décrit en détail dans le chapitre 4. Enfin les contributions principales de cette thèse dans les domaines de l'*ADAS* et de la vidéo-surveillance sont résumées dans le chapitre 5.

Chapter 1

Introduction

Advanced Driver Assistance Systems (ADAS) aim at enhancing driving safety and comfort through technology integrated on-board and in the infra-structure. One can distinguish between the *autonomous* applications, which rely on the first type, and the *cooperative*, which rely on both. The palette of available sensors includes navigation systems which provide directions and up-to-date traffic information, radars and road sensors for traffic surveillance as well as dynamic message boards to display warnings and traffic information.

Currently, some ADAS technologies are mature enough to be commercialized and implemented in vehicles. For example, lane departure warning systems are embedded in cars, using camera vision to alert the driver when he accidentally crosses over the lane markings. On the other hand, some vision-based systems, such as pedestrian detection and traffic sign recognition, are not yet perfectionized.

Traffic signs inform the driver of speed limits and warn him against possible dangers such as icy roads, imminent road works or pedestrian crossings. In fact, the on-board traffic sign detection challenge, only introduced to the market in 2008 by BMW and Opel, are currently limited to the speed limit and no overtaking sign detection. As for the pedestrian detection, some carmakers, such as Volvo, offer a daylight pedestrian detection system using a fusion of camera vision and radar.

1.1 Context

In this thesis, we focus on the development of monocular vision-based traffic sign recognition and pedestrian detection systems. The advantage of using an embedded camera over other sensors, such as radars, lidars or transmitters integrated in the infrastructure, is that they are less expensive and multifunctional. The images captured can be used for several functionalities including traffic sign recognition and pedestrian detection in ADAS and video surveillance applications.

The bright colors and simplified pictograms of the traffic signs make them easily perceivable and comprehensible. However, four of the main difficulties facing a TSR system are: i) the poor visibility due to bad weather, poor illumination or low resolution, ii) the rotation, occlusion and deterioration of the signs, iii) the limited memory and processing capacities and the real-time requirements of the embedded systems and iv) the variations

(a) Video surveillance (b) ADAS

Figure 1.1: We propose two approaches to eliminate false alarms in video surveillance and ADAS systems. Green: validated detection, red: discarded false alarm.

from one country to another. The pedestrian detection task imposes further difficulties such as the non-rigid nature of humans as well as the diversity of their clothing, sizes and postures.

1.2 Contributions

We develop a classification-based approach to overcome the aforementioned challenges in TSR and pedestrian detection. On the one hand, this allows us to exploit the standardized nature of the traffic signs. On the other, a generic model for the pedestrian detection can be generated. Moreover, the classification approach is feasible in real-time and respects the memory constraints of an embedded system.

The existing pedestrian detection approaches often generate many false alarms. We propose two approaches for the post-filtering of detected candidates to improve the precision. The first approach reduces the false alarm rate of a fixed camera video surveillance system by eliminating static detections. The second approach enhances ADAS by removing recurring false detections such as trees, poles and car tires. Figure 1.1 illustrates some examples of the false alarms eliminated using our proposed approaches.

Moreover, we propose a complete TSR system in Chapter 4. This consists of a segmentation, a detection and a classification step. The first reduces the search space while the second step retrieves candidates based on their shape. These are passed on to the classifier in the third step which identifies the content of the traffic signs found. Figure 1.2 illustrates an example of our TSR system for an embedded camera setup.

Further, to overcome the memory and processing constraints in embedded systems, two feature selection techniques, Random Forest and Fisher's Criterion, are implemented to reduce the feature space by retaining only a subset of the most useful features.

1.3 Outline

This thesis is divided into five chapters:

- This chapter gives an overview of the aims and objectives of the thesis as well as

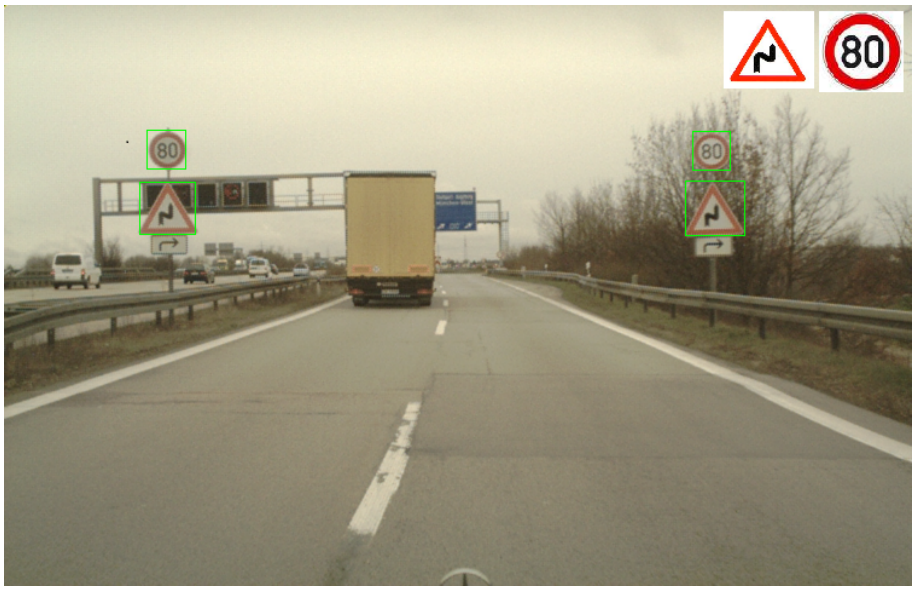


Figure 1.2: An example of Traffic Sign Recognition. We propose a complete TSR system to detect and classify the traffic signs.

its contributions to the Advanced Driving Assistance Systems (ADAS) and Video Surveillance applications.

- An overview of the state of the art, the existing features, machine learning algorithms and applications is presented in Chapter 2.
- In Chapter 3, three aspects of the classification process are evaluated: the importance of the feature, the metric used in the classifier to compare the features and the combination of different classifiers. Moreover, our improvements of the pedestrian detection for video surveillance and ADAS systems are presented
- We propose a three-stage traffic sign recognition system in Chapter 4.
- The main contributions in the fields of ADAS and video surveillance systems as well as the conclusions drawn in this thesis are summarized in Chapter 5.

Chapitre 2

Apprentissage: Du spécifique au générique

Ce chapitre présente l'état de l'art des caractéristiques et des méthodes d'apprentissages existantes les plus utilisées pour la reconnaissance d'objets. Le choix des caractéristiques ou des algorithmes dépend souvent de la qualité des images disponibles, de la nature de l'objet à reconnaitre et des contraintes de l'application finale. Les avantages et désavantages des différentes méthodes sont présentées ici, ainsi que des exemples concrets d'applications.

Concernant les caractéristiques, on peut les répartir en deux catégories: locales et globales. Le premier type, auquel appartiennent les points d'intérêt, décrivent une sous-partie de l'image. Tandis que le second type, parmi lesquels les histogrammes de gradients orientés (en anglais: *Histogram of Oriented Gradients (HOG)*), décrivent l'image entière. Il est montré dans nos expériences et l'état de l'art présenté dans ce chapitre, que les caractéristiques de HOG sont les plus appropriées pour la description d'images d'objets hétéroclites.

Quant à la classification, nous comparons plusieurs approches différentes comme Adaboost, les séparateurs à vaste marge (en anglais: *Support Vector Machines (SVM)*) et les classifieurs à base d'arbres: *K-d tree* et *Random Forests*.

Nous évaluons les différents algorithmes sur deux bases d'images publiques: ETH80 et Caltech 101. Nos expériences montrent que la performance des SVM et des classifieurs à base d'arbres dépend de la cardinalité et la distribution des données d'apprentissage. Les SVMs atteignent des taux de classification élevés, mais au prix de temps de calcul importants. Cependant les arbres de classification sont une alternative efficace. Les *Random Forests* offrent de bonnes performances pour un coût moindre en temps de calcul.

Chapter 2

Learning: From Characteristic to Generic

Contents

2.1 Describing the Object **8**
2.1.1 Local Features 8
2.1.1.1 Haar 8
2.1.1.2 Control Points 9
2.1.1.3 Interest Points 10
2.1.2 Global Features 12
2.1.2.1 Histogram of Oriented Gradients 12
2.1.2.2 Spatial Pyramids 12
2.1.2.3 Distance Transforms 13
2.2 Classifying the Object **14**
2.2.1 Binary Adaboost 14
2.2.2 Multiclass Adaboost 15
2.2.3 Support Vector Machines 20
2.2.3.1 Binary 21
2.2.3.2 Multiclass 24
2.2.4 Tree Classifiers 25
2.2.4.1 K-d tree 25
2.2.4.2 Random Forest 26
2.3 Comparing on Benchmarks **28**
2.3.1 Data Sets 28
2.3.2 Performance Evaluation 29
2.3.2.1 ETH80 29
2.3.2.2 Caltech 101 data set 30
2.4 Conclusions **32**

This chapter gives an overview of the existing image features and machine learning algorithms used for the vision-based object recognition. The choice of the suitable features or algorithms usually depends on the quality of the images, the characteristics of the object and the constraints of the application. The advantages and disadvantages of the

different techniques will be discussed and examples of applications presented. Further, a comparison of various features and classifiers on two classification benchmark data sets is presented.

2.1 Describing the Object

Image features can be divided into two categories: local and global. Local features, such as interest points, describe sparse patches of the image. These patches are often selected to be informative, such as corners and rich texture. An example of such features is the interest point descriptor Speed-Up Robust Features (SURF). Global features, on the other hand, describe the entire image. An example of such features is the Histograms of Oriented Features (HOG).

2.1.1 Local Features

Local features describe patches of the image. One example is the Interest Points. These are patches rich in information such as corners and edges. The gradient and texture information in the patches is encoded using descriptors such as Scale-Invariant Feature Transform (SIFT) and Speed-Up Robust Features (SURF). The sparse representation of the image using patch descriptors requires less computation time and memory. The Haar features, Control Points and Interest Points will be presented in the following.

2.1.1.1 Haar

The Haar-like features introduced by Viola and Jones [Viola 01] compute the difference of the sums of the pixels in the adjacent regions: $region_1$ and $region_2$. A positive sample is detected when this difference exceeds a predefined threshold θ as in equation (2.1).

$$\Sigma(region_1) - \Sigma(region_2) > \theta \tag{2.1}$$

The basic types of Haar-like features are illustrated in Figure 2.1. As described in [Viola 01], the computation of the Haar-like features can be accelerated using integral images. A pixel at a point (x, y) in the integral image is defined as the sum of all pixels above and to its left. This is calculated as shown in equation 2.2. Figure 2.2 illustrates the computation of the pixel sum within an area using this method.

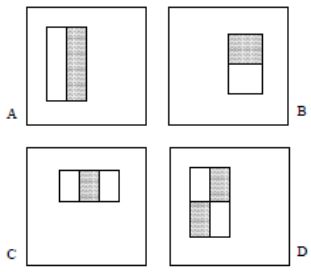


Figure 2.1: Examples of Haar-like features from [Viola 01]. The pixel values of the regions of each color are summed and their difference computed.

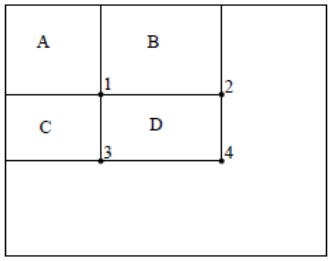


Figure 2.2: Integral Image representation: The pixel sums at the points 1, 2, 3 and 4 are that of the areas A, (A+B), (A+C) and (A+B+C+D) respectively. The sum of the pixels in area D can be easily computed using the values at the points 1, 2, 3 and 4: D = 4-(2+3)+1.

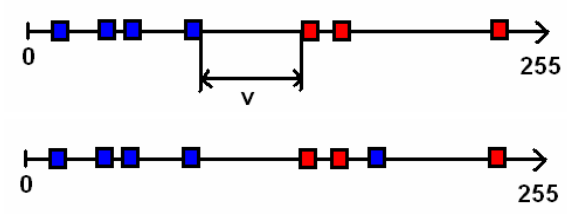


Figure 2.3: A Control Points descriptor is a set of pixels divided into two subsets (red and blue) and a minimum distance v between them. A sample descriptor is considered positive (top) when the $v > \theta$ and negative (bottom) when the points are not linearly separable.

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} src(x', y') \tag{2.2}$$

2.1.1.2 Control Points

Abramson and Steux [Abramson 05] propose pixel-based features, called control points. These are faster and require far less memory than the Haar-like features as they eliminate the need for preprocessing and the storage of the integral images. They are also illumination independent, making them more robust to variations in the image.

A Control Points descriptor consists of a vector $cp = \{\{(x_i, y_i), l_i \in \{a, b\}\}, i \in [1, N], \theta\}$, where N is the total number of control points, $\{(x_i, y_i), l_i\}$ designates their location and $l_i \in \{a, b\}$ is the corresponding subset label. The minimum difference between the pixel values of the subsets a and b is defined as θ. The best descriptors are selected from a randomly generated set of Control Points during the Adaboost training, described in Chapter 2.2.1.

When classifying a new descriptor, the difference v between the values $\{(x_i, y_i)\}$ of the subsets a and b needs to exceed θ for a positive classification as shown in equation 2.3 and illustrated in Figure 2.3.

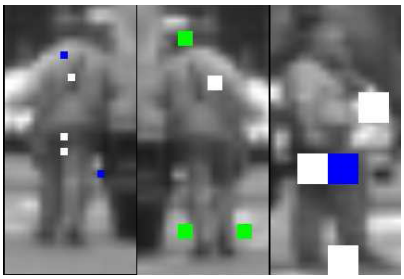


Figure 2.4: Examples of control points features [Abramson 05] on full, half- and quarter-resolution images (from left to right). Two colors designate the Control Point subsets a and b.

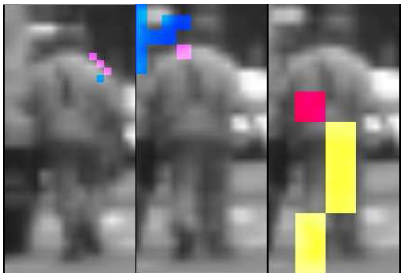


Figure 2.5: Examples of Connected Control Point features [Stanciulescu 07] on full, half- and quarter-resolution images (from left to right). Two colors designate the Control Point subsets a and b.

$$\begin{array}{rc} & \forall i,j \in N \times N \\ \text{Positive:} & min\{value((x_i,y_i)|l_i=a) - value((x_j,y_j)|l_j=b)\} \quad > \theta \\ \text{Negative:} & otherwise \end{array} \tag{2.3}$$

An optimal set size N of six points is empirically determined in [Abramson 05]. Examples of applications are shown in Figure 2.4.

Stanciulescu et al. [Stanciulescu 07] enhanced the Control Points by asserting an 8-connexity, leading to a reduction of the search space. In their experiments on real-time vehicle detection, the Connected Control Points outperformed the classical ones as well as the Haar-like features. Examples of the Connected Control Points used on pedestrian images are illustrated in Figure 2.5.

We published a comparison of the Haar and the Control Points features for pedestrian detection in [Zaklouta 09] showing that the latter yields better results.

2.1.1.3 Interest Points

An interest point is a point in an image which is rich in information, i.e. with strong gradients. These points are found using detectors such as the Harris and the Hessian-based detector. The quality of a detector can be measured by its robustness to variations and its repeatabilty. The repeatability refers to the capability of detecting the same interest points under different view conditions. The information at the interest points is then described

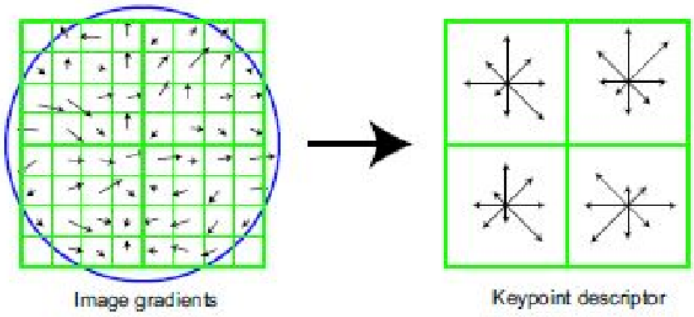


Figure 2.6: Structure of SIFT interest points [Lowe 04]

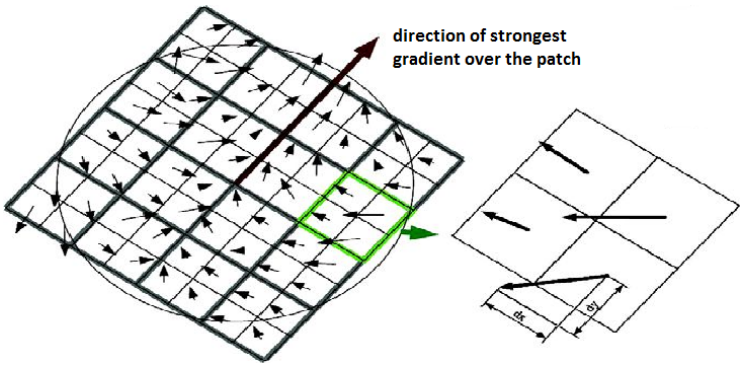


Figure 2.7: Structure of SURF interest points [Bay 08]

using descriptors, such as Scale-Invariant Feature Transform (SIFT) and Speeded-Up Robust Features (SURF).

Detectors

The affine invariant Harris detector [Mikolajczyk 02] locates corner points, i.e. points with significant horizontal and vertical gradients, using the Harris Corner detector. It iteratively analyzes them at different scales and neighborhood patch sizes. The Hessian-based detector [Mikolajczyk 01] iteratively optimizes the location, scale and neighborhood of the points to determine the affine invariant interest points. Bay et al. [Bay 08] introduced a Fast-Hessian detector. It acclerates the computation by using the integral image representation. Published surveys [Tuytelaars 08, Mikolajczyk 05] on local features determined that the Hessian-based detectors are more stable and repeatable than the Harris-based ones.

Descriptors

Once the interest points are detected, they are represented by a descriptor. Lowe et al. [Lowe 04] developed the SIFT descriptor, which stores the local gradient histograms. This proved to be fast and robust to changes in scale, rotation and illumination. Figure 2.6 illustrates the structure of the SIFT descriptor. The patch around the interest point is divided into blocks, four in general. In each block, the histogram of the gradient orientations of the pixels is computed. A 2-D Gaussian is applied to weight the values in the center of the patch. The patch is rotated so that the strongest gradient faces north. This makes the descriptors comparable and hence robust to rotations. The gradient orientation histograms are concatenated to form the SIFT descriptor vector.

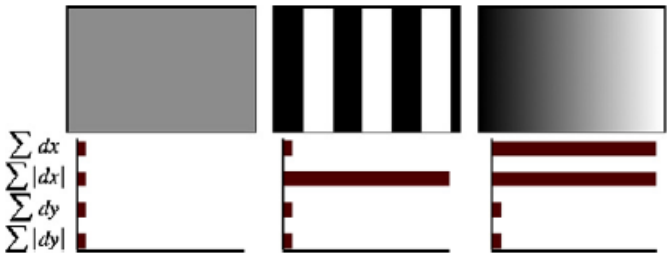


Figure 2.8: Gradient responses used in SURF[Bay 08]. Left: A uniform surface produces low responses. Middle: Stripes invoke a high $\sum |d_x|$. The positive and negative d_x responses counterbalance, keeping $\sum d_x$ low. Right: Gradual increase in intensity augments both $\sum d_x$ and $\sum |d_x|$

The SURF is a 64 value descriptor introduced by [Bay 08], which describes the gradients in a patch around an interest point. Figure 2.7 illustrates the structure of the SURF descriptor.

$$v_{surf} = (\Sigma d_x, \Sigma d_y, \Sigma|d_x|, \Sigma|d_y|) \tag{2.4}$$

where d_x and d_y are the horizontal and vertical gradients respectively. The responses d_x and d_y are illumination invariant as shown in Figure 2.8. The sum of the gradient magnitudes describes the intensity information.

2.1.2 Global Features

Global features describe the entire image. Examples of such features are Distance Transforms (DT) [Gavrila 07, Franke 99] and Histograms of Oriented Gradients (HOG) proposed by Dalal and Triggs [Dalal 05]. Both achieve high performance in pedestrian detection applications.

2.1.2.1 Histogram of Oriented Gradients

Binning is defined as the combining of several pixels into a function unit. It is often used to reduce the noise in an image. The HOG feature, proposed by Dalal and Triggs [Dalal 05] and primarily used for pedestrian detection, is based on this principle. Figure 2.9 illustrates the structure of the HOG descriptor. The image is divided into overlapping blocks. Each block, in turn, is divided into non-overlapping cells. The gradient orientation and magnitude are computed for each pixel. A histogram of these orientations is formed for each cell. The magnitude of the gradient is used as a vote weight. A local contrast normalization is then performed on each block. The resulting histograms are concatenated to form the HOG descriptor.

2.1.2.2 Spatial Pyramids

The concept of spatial pyramids such as Pyramid Histogram Of Visual Words (PHOW) [Lazebnik 06, Bosch 06] and Pyramid of Histograms of Oriented Gradients (PHOG) [Bosch 07b] is to describe the image at several repartition levels. The coarse representation at the low level of the pyramid captures the global form of the object, while the finer partitioning of the image captures the specific details. Both PHOW and PHOG describe the orientations and intensities of the gradients in the image.

Pyramid Histogram of Visual Words (PHOW)
These descriptors [Bosch 07a] are a variant of dense SIFT descriptors. The SIFT de-

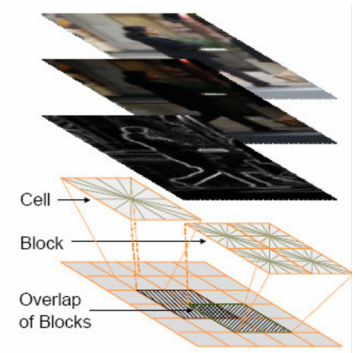


Figure 2.9: Structure of HOG descriptor [Dalal 05]

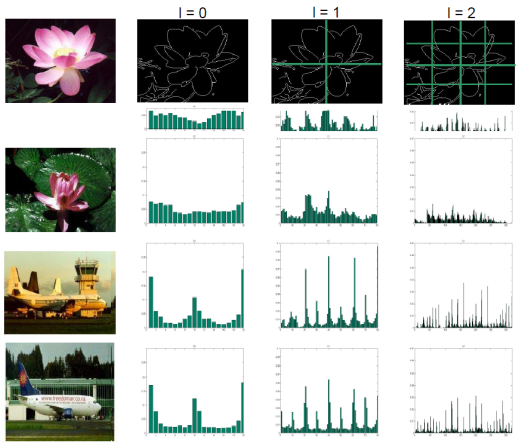


Figure 2.10: Structure of PHOG descriptor [Bosch 07b]

scriptors [Lowe 04] are computed at multiple scales on patches, which are densely distributed over the image. These descriptors are merged into *visual words* using *k-means*, which is a clustering algorithm that divides a sample set into k clusters, resulting in a Voronoi partitioning of the data space. Each sample belongs to the cluster with the nearest mean value. The vocabulary set of each image, usually consisting of hundreds of words, is represented by a histogram of word occurences. This feature is used in [Lazebnik 06, Bosch 06] to classify scenes and in [Bosch 07a, Bosch 07b] for object recognition

Pyramid of Histograms of Oriented Gradients (PHOG)
These descriptors [Bosch 07b] are a concatenation of HOG descriptors, calculated at several partitioning levels of the image. Figure 2.10 illustrates the composition of the PHOG descriptor. At the first level $l = 0$, the HOG descriptor is computed over the entire image. At levels $l = 1$ and $l = 2$, the HOG descriptors of all four and eight image subdivisions are computed and concatenated. The PHOG descriptor of the image is the weighted concatenation of those from all three levels. The weights can be either learned during training or set empirically.

2.1.2.3 Distance Transforms

The Distance Transforms (DT) represent the distance of each pixel to the nearest edge in the corresponding Canny edge image as illustrated in Figure 2.11. The advantage of the DT over the edge image is that the similarity measure is smoother. The correlation

Figure 2.11: Distance Transform. The lighter the pixel, the further it is from an edge.

distance between the template DT and that of the test image determines the classification decision.

2.2 Classifying the Object

Classification is the problem of identifying the class to which a new sample belongs using the knowledge obtained from a set of training samples. There are several approaches to solving this problem. On the one hand, the tree classifiers, such as the K-d tree, use optimized techniques, such as hierarchichal structures and Best Bin First Approximate Search [Beis 02] to efficiently comb through the training samples and determine the Nearest Neighbor of the test sample. The advantage of this type of approach is that they are particularly performant when using unbalanced data sets as shown in [Khoshgoftaar 07].

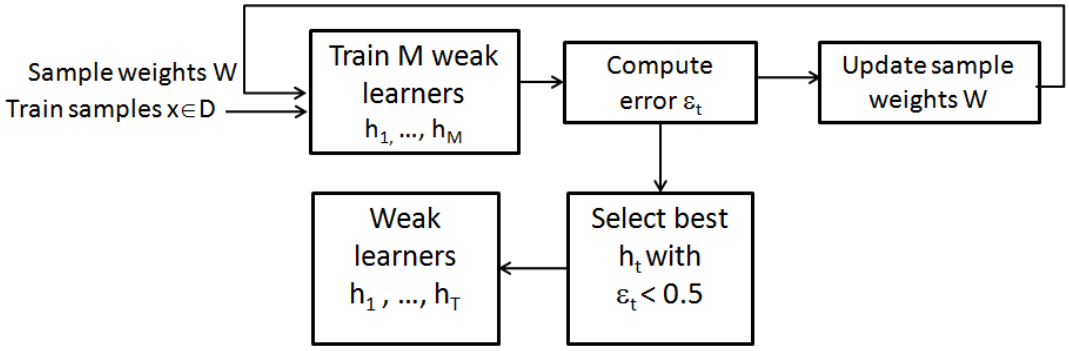
On the other hand, algorithms such as Adaboost and Support Vector Machines (SVM) learn a representative subset of weak learners or training samples. They are less sensitive to outliers and posess a strong generalization capacity as they discover trends in the data. The computation and memory requirements are reduced since only a representative subset is considered.

The choice of the appropriate classification algorithm depends on the quantity and class distribution of the training samples as well as the features used. Different variants of the Adaboost algorithm, the Support Vector Machines as well as the tree classifiers: K-d tree and Random Forest are presented in this section.

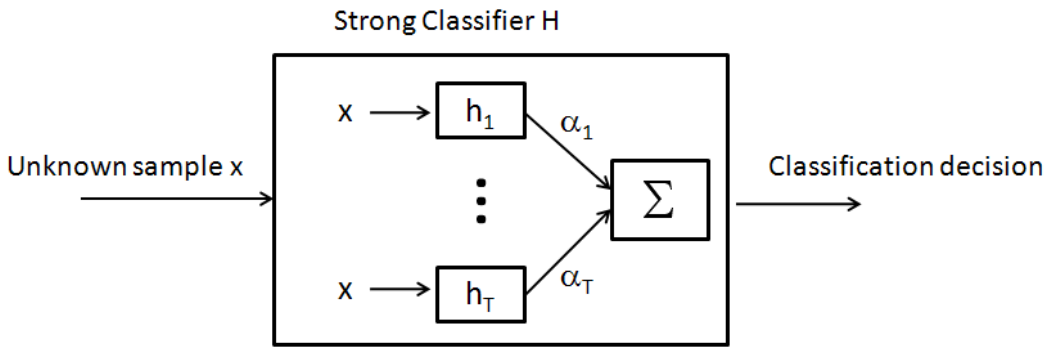
2.2.1 Binary Adaboost

Adaboost [Freund 97, Schapire 99] is a powerful machine learning algorithm. It combines T weak classifiers h_t into a strong classifier H. They are chosen to minimize the error on the samples from the training set. The confidence α_t in their decision is a function of their classification error ϵ_t. In each round, the misclassified training samples obtain a higher weight, so that the weak learner focuses on classifying them correctly. The procedure of the Binary Adaboost algorithm is described in Algorithm 1. Figure 2.12 illustrates the general training and classification schemes of the Adaboost algorithm.

Each weak classifier has an accuracy better than that of random guessing. Dietterich et al. [Dietterich 00] shows that there is a trade-off between accuracy and generalization capacity. The more accurate the classifiers, the more often they agree, which in turn implies that their errors are correlated, increasing the risk of overfitting. The less accurate the weak learners, the more resistant they are to outliers, yet the less accurate they are, increasing the error rate.



(a) Training of weak learners $h_1, \ldots, h_T$



(b) Classification of sample x by strong classifier H

Figure 2.12: Adaboost training and classification

2.2.2 Multiclass Adaboost

Several extensions to the Adaboost algorithm have been proposed over the past decade. This section gives a brief overview of some of these variants.

Freund et al. [Freund 96] propose the Adaboost.M2 algorithm (see Algorithm 2) for solving the multi-class problem. It differs from the original binary version in the weighting of sample-label pairs $\tilde{W}$ as opposed to the sample weighting W in Adaboost, as well as the use of the *pseudo-loss measure* in the error evaluation.

The weak learners in this algorithm predict a set of possible classes for a sample **x**. The final decision is the class with the highest vote. The weak learner's descision is weighted according to the pseudo-loss measure, which penalizes the absence of the correct label in the predicted set as well as the wrong labels. This measure focuses the learning on the hard samples, as well as the labels which are difficult to distinguish. The disadvantage of the pseudo-loss measure is, however, that it is time-consuming to compute. For K classes, it takes $O(K)$ times longer to evaluate the error than in the binary Adaboost algorithm.

Error Correcting Output Codes

Error Correcting Codes (ECOC) were originally used to discover transmission errors in signal processing. They are used to reduce the multi-class problem to several binary ones in [Dietterich 95]. During training, a set of N binary classification functions are generated, each partitioning the $K > 2$ labels into two classes. This repartition of the labels is known as *coloring*. A coloring function μ maps each label $l_i \in \{1, \ldots, K\}$ to +1 or -1.

Algorithm 1 Binary Adaboost

Data $D = \{x_1, \ldots, x_N\}$, Labels $L = \{l_i \in \{1, -1\}\}$, T number of weak learners
- *initialize equal weights w for all N data samples*
  $w = \frac{1}{N}$, $w \in W$

for t=1 to T **do**
- *find classifier h_t with smallest error ϵ_t with respect to W*
  $\epsilon_t = \sum_{i=1}^{N} w_i, l_i \neq h_t(x_i)\}$
- *compute classifier weight*
  $\alpha_t = ln(\frac{1-\epsilon_t}{\epsilon_t})$
- *update sample weights*
  $w_i = \frac{w_i}{Z} \cdot exp(-\alpha_t \cdot l_i \cdot h_t(x_i))$, where $Z = \sum_{i=1}^{n} w_i$

end for

- *Classification of sample x using strong classifier H*
  $H(x) = sign\big(\sum_{t=1}^{T} \alpha_t \cdot h_t(x)\big)$

Algorithm 2 Adaboost.M2

Data $D = \{x_1, \ldots, x_N\}$, Labels $L = \{l_i \in \{1, \ldots, K\}\}$, T number of weak learners
- *initialize weights for all sample-label pairs $\tilde{W_{i,k}}$*

$$\tilde{W_{i,k}} = \begin{cases} \frac{1}{N(K-1)}, & l_i \neq k \\ 0, & l_i = k \end{cases}$$

for t=1 to T **do**
- *find classifier h_t with smallest error ϵ_t with respect to $\tilde{W}$ using pseudo-loss measure*
  $\epsilon_t = \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{K} \tilde{W_{i,k}} \cdot \big([l_i \notin \{h_t(x_i)\}] + [l \in \{h_t(x_i)\}]\big)$
- *compute classifier weight*
  $\alpha_t = \frac{1}{2} ln(\frac{1-\epsilon_t}{\epsilon_t})$
- *update sample-label weights*
  $\tilde{W_{i,k}} = \frac{1}{Z} \cdot \tilde{W_{i,k}} \cdot exp(\alpha_t \cdot \big([l_i \notin \{h_t(x_i)\}] + [l \in \{h_t(x_i)\}]\big))$,
  where $Z = \sum_{i=1}^{N} \sum_{k=1}^{K} \tilde{W_{i,k}}$

end for

- *Classification of sample x using strong classifier H*
  $H(x) = argmax_{l \in Y}\big(\sum_{t=1}^{T} \alpha_t \cdot [l \in h_t(x_i)]\big)$

Each of the K classes is assigned a *codeword* composed of T values. The t-th bit of a codeword indicates the coloring of the class with respect to the t-th binary classification function h_t. The result of the training is a K×T matrix consisting of +1 and -1 values. Table 2.1 shows an example of a coloring of four classes. The binary classifier h_1 learns to distinguish between classes $\{c_1, c_2, c_3\}$ and $\{c_4\}$. The codeword of class c_1 is $< -1, -1, -1, -1, -1, 1 >$.

The Hamming distance $h_d(w_1, w_2)$ between two codewords cw_1, cw_2 is the number of bits in which they differ. The codeword of a new data sample **x** consists of the responses of each of the T classifiers. Its class corresponds to the one with the smallest Hamming distance to its codeword.

		Classifiers			
		l_1	l_2	l_3	l_4
Learners	h_1	-1	-1	-1	1
	h_2	-1	-1	1	1
	h_3	-1	1	1	1
	h_4	-1	1	-1	1
	h_5	-1	1	1	-1
	h_6	1	1	-1	1

Table 2.1: Class Coloring: Codewords for $K = 4$ classes with $T = 6$ values corresponding to the $t_1, \ldots, t_T$ weak learners

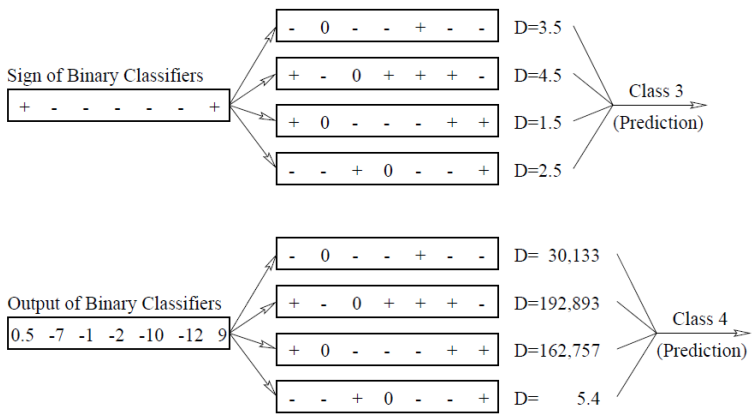


Figure 2.13: The Hamming-decoding (top) uses the Hamming distance for classification. The Loss-based decoding (bottom) uses the an exponential function of the confidence measures of each weak learner. [Allwein 01]

The robustness of the Error Correcting Codes is due to their ability of correcting up to $\lfloor \frac{d-1}{2} \rfloor$ bit errors, where d is the minimum Hamming distance between any two codewords. Given two classes c_1 and c_2 with codewords $(1, 1, 1)$ and $(-1, -1, -1)$ respectively. The minimum Hamming distance is $d = 3$, i.e. one bit error can be corrected. Given samples $x_1, x_2 \in c_1$, with the codewords $(1, -1, 1)$ and $(1, -1, -1)$. The Hamming distances are $h_d(cw(x_1), cw(c_1)) = 1$ and $h_d(cw(x_2), cw(c_1)) = 2$ to c_1 and $h_d(cw(x_1), cw(c_2)) = 2$ and $h_d(cw(x_2), cw(c_2)) = 1$ to c_2. The ECOC was able to correctly classify x_1 as there is only a one bit error. However, x_2 is misclassified as there are two bit errors.

The larger the minimum Hamming distance, the stronger the ECOC. A minimum Hamming distance of 2 is given in a one-vs-all coloring, which does not have the error correcting property.

Allwein et al. [Allwein 01] proposed the *don't care* coloring value 0, which allows the weak learner to ignore a class during the training. A *don't care* value counts as half a vote when computing the Hamming distance. They also introduced the more accurate *Loss-based Decoding* of the vote vectors using a confidence measure to weight the weak learner's decision. The difference between the Hamming-decoding and the Loss-based decoding is illustrated in Figure 2.13.

Algorithm 3 Adaboost.OC

Data $D = \{x_1, \ldots, x_N\}$, Labels $L = \{l_i \in \{1, \ldots, K\}\}$, T number of weak learners

- *initialize weights for all sample-label pairs* $\tilde{W_{i,k}}$

$$\tilde{W_{i,k}} = \begin{cases} \frac{1}{N(K-1)}, & l_i \neq k \\ 0, & l_i = k \end{cases}$$

for t=1 to T **do**

- *compute coloring* μ_t
  $\mu_t : L \rightarrow \{-1, +1\}$
- *compute sample weights* $w \in W$
  $w_i = \frac{1}{U} \cdot \sum_{k=1}^{K} \tilde{W_{i,k}} \left[\mu_t(l_i) \neq \mu_t(l)\right]$, where $U = \sum_{i=1}^{N} w_i$
- *find classifier* h_t *with smallest error* ϵ_t *with respect to* W
  $\epsilon_t = \frac{1}{2} \sum_{m=1}^{N} \sum_{k=1}^{K} \tilde{W_{i,k}} \cdot \left(\left[l_i \notin L_h\right] + \left[l \in L_h\right]\right)$
  $L_h = \{l \in \{h_t(x) = \mu_t(l)\}\}$
- *compute classifier weight*
  $\alpha_t = \frac{1}{2} ln(\frac{1-\epsilon_t}{\epsilon_t})$
- *update sample-label weights*
  $\tilde{W_{i,k}} = \frac{1}{Z} \cdot \tilde{W_{i,k}} exp(\alpha_t \cdot \left(\left[l_i \notin L_h\right] + \left[l \in L_h\right]\right))$
  $Z = \sum_{i=1}^{N} \sum_{k=1}^{K} \tilde{W_{i,k}}$ *is normalization factor*

end for

- *Classification of sample* x *using strong classifier* H
  $H(x) = argmax_{l \in L}\left(\sum_{t=1}^{T} h_t(x) \cdot \alpha_t \cdot \left[l \in L_h\right]\right)$

Adaboost.OC

The Adaboost.OC algorithm [Schapire 97], described in Algorithm 3, combines the Error Correcting Output Codes and boosting. Similarly to Adaboost.M2, it weights sample-label pairs using the distribution $\tilde{W}$. These weights are then used to compute the sample weight distribution W. In each round, a weak learner h_t, minimizing the error with respect to W, is generated. Its weight α_t and the sample-label distribution $\tilde{W}$ are updated with respect to its pseudo-loss error ϵ_t. The sample weights are updated in turn with respect to $\tilde{W}$.

The ECOC generate the coloring μ_t in each training round, reducing the multi-class problem to a binary one. The decision of a weak learner h_t is computed with respect to the coloring μ_t in round t, i.e. a correct classification of a sample x_i premises $\mu_t(l_i) \in h_t(x_i)$. The final decision is the class with the highest vote.

Adaboost.ECC

The Adaboost.ECC algorithm [Guruswami 99], described in Algorithm 4, is similar to Adaboost.OC, except that the time-consuming pseudo-loss error ϵ_t computation is eliminated. Instead, the confidence α_t of each weak learner h_t, is calculated based on its error rate. This simplifies the algorithm and accelerates the training.

One can differentiate between the symmetric and asymmetric case. In the former, there is only one weight α_t per classifier h_t, while in the latter the weight α_t or β_t depends on the classification outcome of h_t.

Algorithm 4 Adaboost.ECC

Data $D = \{x_1, \ldots, x_N\}$, Labels $L = \{l_i \in \{1, \ldots, K\}\}$, T number of weak learners

- *initialize weights for all sample-label pairs* $\tilde{W_{i,k}}$

$$\tilde{W_{i,k}} = \begin{cases} \frac{1}{N(K-1)}, & l_i \neq k \\ 0, & l_i = k \end{cases}$$

for t=1 to T **do**

- *compute coloring* μ_t
  $\mu_t : L \to \{-1, +1\}$
- *compute sample weights* $w \in W$
  $w_i = \frac{1}{U} \cdot \sum_{k=1}^{K} \tilde{W_{i,k}} \big[\mu_t(l_i) \neq \mu_t(l)\big]$
  with $U = \sum_{i=1}^{N} w_i$ *is normalization factor*
- *find classifier* h_t *with smallest error* ϵ_t *with respect to* W
  $\epsilon_t = \sum_{i=1}^{N} w_i, \big[l_i \notin L_h\big]\}$
  $L_h = \{l \in \{h_t(x) = \mu_t(l)\}\}$
- *compute classifier's positive and negative vote* α_t *and* β_t *respectively*

$$g_t = \begin{cases} \alpha_t, & h_t = +1 \\ -\beta_t, & h_t = -1 \end{cases}$$

  - ⋄ *Asymmetric case*
    $\alpha_t = \frac{1}{2} ln\Big(\frac{\sum_{i:h_t(x_i)=\mu(l_i)=+1} w_i}{\sum_{i:h_t(x_i)=+1\neq\mu(l_i)} w_i}\Big)$
    $\beta_t = \frac{1}{2} ln\Big(\frac{\sum_{i:h_t(x_i)=\mu(l_i)=-1} w_i}{\sum_{i:h_t(x_i)=-1\neq\mu(l_i)} w_i}\Big)$
  - ⋄ *Symmetric case*
    $\alpha_t = \beta_t = \frac{1}{2} ln\Big(\frac{\sum_{i:h_t(x_i)=\mu(l_i)} w_i}{\sum_{i:h_t(x_i)\neq\mu(l_i)} w_i}\Big)$

- *update sample-label weights*
  $\tilde{W_{i,k}} = \frac{1}{Z} \cdot \tilde{W_{i,k}} \cdot exp\Big(-\frac{1}{2} \cdot g_t \cdot \big(\mu_t(l_i) - \mu_t(k)\big)\Big)$
  $Z = \sum_{i=1}^{N} \sum_{k=1}^{K} w_i$ *is normalization factor*

end for

- *Classification of sample* x *using strong classifier* H
  $H(x) = argmax_{l \in L}\big(\sum_{t=1}^{T} h_t \cdot g_t \cdot \mu_t(l)\big)$

Algorithm 5 Adaboost.ERP

for r = 1 to R **do**

- *compute sample weights* $w \in W$
  $w_i = \frac{1}{U} \cdot \sum_{k=1}^{K} \tilde{W_{i,k}} \big[\mu_t(l_i) \neq \mu_t(k)\big]$
  with $U = \sum_{i=1}^{N} w_i$ *is normalization factor*
- *find classifier* h_t *with smallest error* ϵ_t *with respect to* W
  $\epsilon_t = \frac{1}{2} \sum_{m=1}^{N} \sum_{k=1}^{K} \tilde{W_{i,k}} \cdot \big(\big[l_i \notin L_h\big] + \big[l \in L_h\big]\big)$
  $L_h = \{l \in \{h_t(x) = \mu_t(l)\}\}$
- *adjust coloring of each label* k
  $\mu_t(k) = sign\big[\sum_{i=1|l_i=k}^{N} \sum_{k=1}^{K} \tilde{W_{i,k}} \cdot h_t(x_i) - \sum_{i=1}^{N} \tilde{W_{i,k}} \cdot h_t(x_i)\big]$

end for

Algorithm 6 Adaboost.ERC

for r=1 to R **do**

- *find classifier* h_t *with smallest error* ϵ_t *with respect to* W
  $\epsilon_t = \frac{1}{2} \sum_{m=1}^{N} \sum_{k=1}^{K} \tilde{W_{i,k}} \cdot \big(\big[l_i \notin L_h\big] + \big[l \in L_h\big]\big)$
  $L_h = \{l \in \{h_t(x) = \mu_t(l)\}\}$
- *adjust sample weight to new classifier* h_t^m
  $w_i = \frac{1}{Z} \cdot w_i \cdot exp\big(h_t^m(x_i) \cdot \mu_t(l_i)\big)$
- *add new classifier* h_t^m *to ensemble*
  $h'_t = h'_t \cup h_t^m$

end for

Adaboost.ERP - ECC with Repartitioning

Li [Li 06] proposes an improvement of the Adaboost.ECC algorithm by adapting the label repartitioning i.e. coloring to minimize the error of the best weak learner in each round. This is done iteratively in a second boosting loop within the Adaboost.ECC algorithm. This sub-loop, shown in Algorithm 5, replaces the search for the classifier h_t with the minimum error ϵ_t.

Adaboost.ERC - ECC with Repeating Codes

Lin et al. [Lin 06] extend the Adaboost.ECC algorithm to the Adaboost.ERC by iteratively training an ensemble of weak learners h'_t on a given coloring μ_t and adjusting the sample weights W in each repetition cycle r. The subloop is described in Algorithm 6.

In each Adaboost round t, a set of weak learners $h_t^1, \ldots, h_t^R$ is trained, with respect to the coloring μ_t, and added to the strong classifier H. When the error of a weak learner h_t^r is low, the sample weights in W hardly change and a similar weak learner h_t^{r+1} is generated in the next round. If it is too weak, the W is adjusted, so that a better one is created in the next round. Therefore, the weighting α_t is obsolete in this algorithm, as the poor learners perish, while the better ones become numerous.

2.2.3 Support Vector Machines

A Support Vector Machine (SVM) [Burges 98] is a binary classifier. The best separation between two classes is represented by a subset of data samples, called *Support Vectors*. A test sample can be classified depending on its distance to the support vectors. The binary SVMs can be combined into an ensemble of one-vs-one or one-vs-all

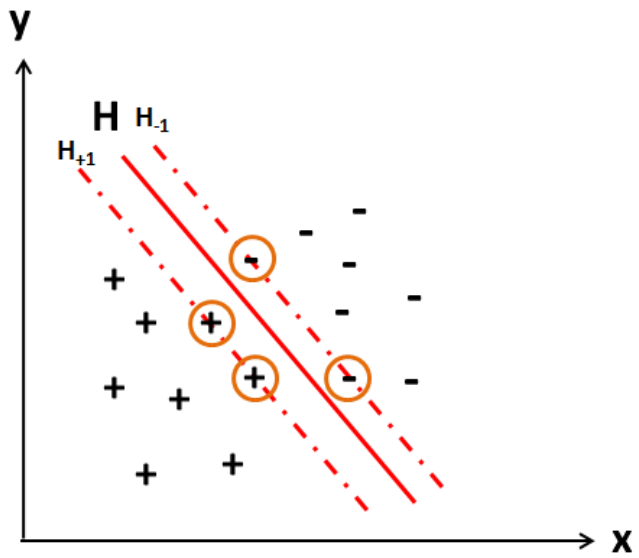


Figure 2.14: Linear SVM: A hyperplane H separates the two classes. The support vectors (encircled) hold up the separating margin.

classifiers to solve multi-class problems.

2.2.3.1 Binary

Given a set of N linearly separable, d-dimensional samples $x_i \in \mathbb{R}^d$ with labels $l_i \in \{-1, +1\}$. A hyperplane H that separates the two classes is defined as $\mathbf{w} \cdot x + b = 0$. The perpedicular distance of H to the origin is then $\frac{|b|}{\|\mathbf{w}\|}$ and the vector $\mathbf{w}$ the normal to H. Figure 2.14 illustrates this data separation.

The shortest distances to the data points of labels -1 and $+1$ are set to d_- and d_+ respectively. The corresponding points are called *Support Vectors*. The width of the margin surrounding H is $d_- + d_+$. Since the training set in this example is linearly separable, one can assume that all the data points x_i satisfy one of the following inequalities:

$$\mathbf{w} \cdot x_i + b \leq -1, \; l_i = -1 \tag{2.5}$$

$$\mathbf{w} \cdot x_i + b \geq +1, \; l_i = +1 \tag{2.6}$$

The inequalities 2.5 and 2.6 can be combined to the following:

$$l_i \cdot (x_i \cdot \mathbf{w} + b) - 1 \geq 0, \; 1 \leq i \leq N \tag{2.7}$$

The support vectors lie on H_{+1} and H_{-1}, with $\mathbf{w} \cdot x + b = \pm 1$. The perpendicular distances from the hyperplanes H_{+1} and H_{-1} to the origin are then $\frac{|-1-b|}{\|\mathbf{w}\|}$ and $\frac{|1-b|}{\|\mathbf{w}\|}$ respectively. Therefore, the margin between H_{+1} and H_{-1} is $\frac{2}{\|\mathbf{w}\|}$ wide.

The Lagrangian Formulation

To optimize the classification performance, the hyperplanes H_{+1} and H_{-1} are chosen to maximize the margin width $\frac{2}{\|\mathbf{w}\|}$, which is equivalent to minimizing $\frac{1}{2}\|\mathbf{w}\|^2$. This can be solved using quadratic programming.

The minimization problem can be defined as a Lagrangian formulation L.

$$L = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i (l_i(x_i \cdot \mathbf{w} + b) - 1) \tag{2.8}$$

where $\alpha_i \geq 0$. The advantage of this form is that the inequality in 2.7 is replaced by Lagrangian multipliers α_i, which are easier to handle.

To find the optimal solution, L needs to be minimized with respect to **w** and b with the Karush-Kuhn-Tucker (KKT) conditions:

$$\frac{\partial L}{\partial \mathbf{w}} = 0 \Leftrightarrow \mathbf{w} - \sum_{i=1}^{N} \alpha_i l_i x_i = 0, \forall i \tag{2.9}$$

$$\frac{\partial L}{\partial b} = 0 \Leftrightarrow -\sum_{i=1}^{N} \alpha_i l_i = 0, \forall i \tag{2.10}$$

$$l_i(x_i \cdot w + b) - 1 \geq 0, \forall i \tag{2.11}$$

$$\alpha_i \geq 0, \forall i \tag{2.12}$$

$$\alpha_i(l_i(w \cdot x_i + b) - 1) = 0, \forall i \tag{2.13}$$

Substituting the equations 2.9 and 2.10 into equation 2.8 results in the dual formulation L_D. This is to be maximized with respect to α_i.

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j l_i l_j x_i \cdot x_j \tag{2.14}$$

The optimal values for **w** and b can be found using:

$$\mathbf{w} = \sum_{i=1}^{N} l_i \alpha_i x_i \tag{2.15}$$

$$b = l_k - \sum_{i=1}^{N} l_i \alpha_i x_i \cdot x_k, \alpha_k > 0 \tag{2.16}$$

where $\alpha_i > 0$ for the support vectors x_i. The classification function $f(x)$ for a new sample x is defined as

$$f(x) = sign(\mathbf{w} \cdot x + b) = sign\Big(\sum_{i=1}^{m} l_i \alpha_i (x_i \cdot x) + b\Big) \tag{2.17}$$

Hence, only the support vectors, $x_1, \ldots, x_m$, are needed to determine the label of an unknown data vector x.

Linearly inseparable Data

White noise, inaccurate measurements or erroneous labeling can lead to inseparable training data. The constraints are relaxed using a slack variable $\xi_i \geq 0,\ 1 \leq i \leq N$, to tolerate the samples within the separating margin. The constraints in 2.5 and 2.6 are adjusted to:

$$x_i \cdot \mathbf{w} + b \geq +1 - \xi_i \tag{2.18}$$

$$x_i \cdot \mathbf{w} + b \geq -1 + \xi_i \tag{2.19}$$

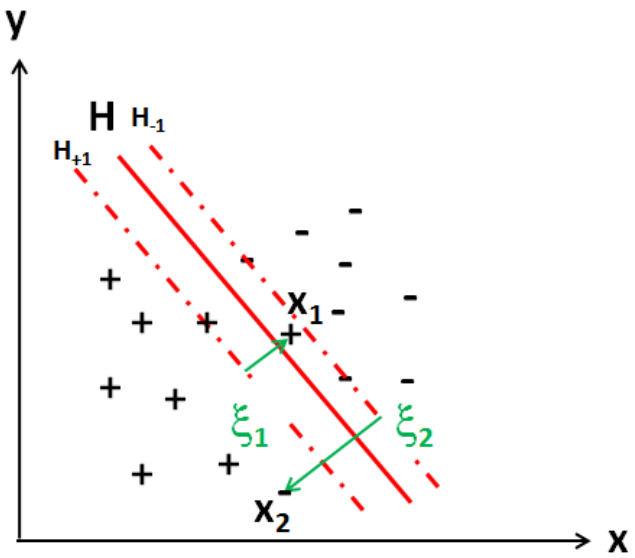


Figure 2.15: Linear separator using slack variable ξ_i. The sample x_1 is classified correctly due to $\xi_1 < 1$, while x_2 is misclassified since $\xi_2 > 1$.

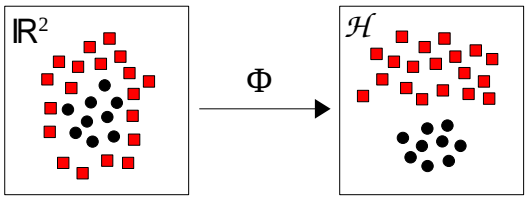


Figure 2.16: The nonlinear function Φ maps linearly inseparable data from $\mathbb{R}^2$ to a higher dimensional Euclidean space $\mathscr{H}$.

A data point x_i on the wrong side of H is considered an error, when ξ_i exceeds 1. The upper bound for the training errors is $\sum_{\forall i} \xi_i$. Figure 2.15 illustrates the use of slack variables.

Nonlinear Support Vector Machines

The linear data classification can be generalized to find a separator for linearly inseparable data. This is done by mapping the data to a different Hilbert space $\mathscr{H}$ using a nonlinear function Φ (see Figure 2.16). The linear classification of the data points in $\mathscr{H}$ is equivalent to their nonlinear classification in $\mathbb{R}^d$.

$$\Phi : \mathbb{R}^d \to \mathscr{H} \tag{2.20}$$

The dot product of the data points $x_i \cdot x_j$ used in training is replaced by $\Phi(x_i) \cdot \Phi(x_j)$. This, however, makes it difficult to handle due to the high dimension of $\mathscr{H}$. To avoid the explicit computation of Φ, the adjusted dot product is replaced by a *kernel function* K.

$$K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j). \tag{2.21}$$

The kernel function K is symmetric ($K(x_i, x_j) = K(x_j, x_i)$) and satisfies the Mercer criteria

$$\int K(x_i, x_j) g(x_i) g(x_j) dx_i dx_j \geq 0 \tag{2.22}$$

for any function g, having a finite value of $\int g(x)^2 dx$ This criteria ensures that the Kernel matrix $K(i, j)$ is positive semi-definite. For more details on the characteristics of kernel functions and their construction refer to [Genton 02].

Some examples of popular kernel functions are:

- **Linear**

$$K(x_i, x_j) = x_i \cdot x_j \tag{2.23}$$

- **Polynomial**

$$K(x_i, x_j) = (\gamma x_i \cdot x_j + r)^d, \ \gamma \in \mathbb{R}^+, \ r \in \mathbb{R}, \ d \in \mathbb{N} \tag{2.24}$$

- **Gaussian Radial Basis Function**

$$K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2} \tag{2.25}$$

To determine the label of an unknown data point x, the classification function defined in 2.17 is adjusted to

$$\begin{aligned} f(x) &= sign(\mathbf{w} \cdot x + b) \\ &= sign\Big(\sum_{i=1}^{m} \alpha_i l_i (\Phi(x_{si}) \cdot \Phi(x)) + b\Big) \\ &= sign\Big(\sum_{i=1}^{m} \alpha_i l_i K(x_{si}, x) + b\Big) \end{aligned} \tag{2.26}$$

where x_{si} are the Support Vectors.

2.2.3.2 Multiclass

Although Support Vector Machines are binary classifiers, they can be employed for multi-class classification in several ways including: one-vs-all or one-vs-one. In the former, one SVM is trained to distinguish each class against all the $N - 1$ others. A test sample is classified by all N trained SVMs. The highest confidence determines the classification decision. In the one-vs-one scenario, $\frac{n \cdot (n-1)}{2}$ SVMs are trained to classify one class against another. To classify a sample, each resulting binary SVM then votes for a class and the highest vote is taken. In both cases, the SVM parameters need to be adjusted to compensate for unbalanced data sets.

In [Weston 99], Weston et al. propose a formulation of the multiclass SVM problem using a piecewise linear separation of the L classes. Their analysis on various benchmarks shows that this approach attains a comparable performance to that of the one-vs-all or one-vs-one SVM variants, however, with a smaller number of support vectors and less kernel calculations.

Kim et al. [Kim 02] compare different construction and evaluation methods for SVM ensembles. They examine bagging and boosting for SVM construction, where the former selects bootstrap samples at random and the latter uses a selection scheme such as Adaboost. The output of the SVMs in the ensemble can be evaluated using different techniques:

- In the majority voting scheme the highest vote determines the class.
- The LSE-based weighting extends this by weighting the SVM output.
- In the double-layer hierarchical combining the outputs of the first-layer classifiers are evaluated by the second-layer classifier.

In their experiments on hand-written digit recognition and fraud detection, Kim et al. [Kim 02] establish that the boosting outperforms bagging. However, the choice of the aggregation method depends on the application. It is asserted that the SVM ensemble techniques outperform a single SVM in terms of classification accuracy.

2.2.4 Tree Classifiers

Binary decision trees organize the data by splitting it hierarchichally. The nodes of the tree recursively divide the data space. To classify a new sample, the tree is traversed down to the leaves. The sample is compared to the information stored in the nodes and the final leaf.

There are many ways to split the nodes in a decision tree. In [Tu 05], the split functions in the nodes are strong classifiers trained using Adaboost. In [Wu 00], the split functions are chosen so that the margin is maximized amongst all possible separating hyperplanes. This improves the average accuracy and ensures a better generalization. Tibshirani et al. [Tibshirani 07] yield comparable results to other state-of-the-art approaches by selecting the largest margin between two classes in each node of the tree classifier. The K-d tree uses median of the feature with the highest variance. The Random Trees in the Random Forest select the feature with the highest Information Gain.

In this thesis, we examine K-d trees and Random Forests and introduce adjustments to improve their performance.

2.2.4.1 K-d tree

A K-d tree is a binary search tree organizing **K**-dimensional data points. Each non-leaf node splits the data into two subspaces according to the i-th feature f_i with the highest variance at that level. In Fig. 2.17, the 2-dimensional sample space is recursively divided along the x and y axes. To ensure the tree is balanced, the median value m_i of f_i is used for the split in each node. The left subtree contains data with values $f_i < m_i$ and the right subtree $f_i > m_i$. This division is repeated until the subtrees are leaves with one sample each.

The K-d tree is a nearest-neighbor-based search tree. To classify a sample, the tree is traversed until a leaf is reached. At each node, the i-th feature f_i sample is compared to the splitting criteria to determine which subtree to descend into. However, this search is computationally expensive in high dimensional feature spaces.

Therefore, Beis et al. [Beis 02] introduce the Best Bin First algorithm [Beis 02], which performs an approximate Nearest Neighbor search. This efficient variant of the K-d tree search algorithm allows for the indexing of higher dimensional spaces which is required

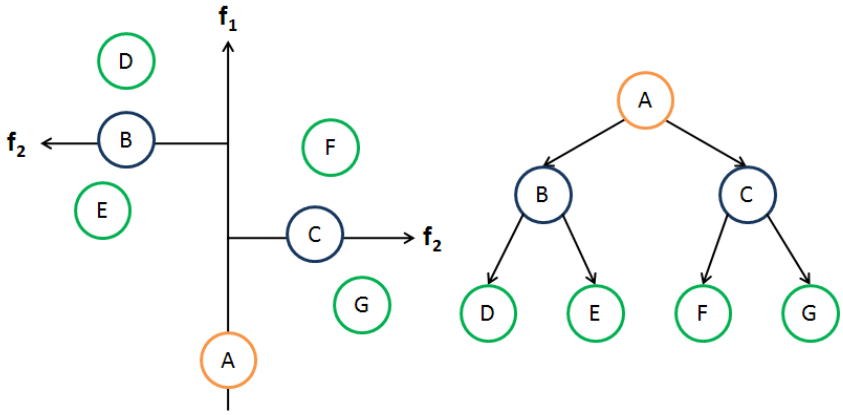


Figure 2.17: Example of a 2-dimensional K-d tree. Root node A divides feature space using feature f_1. Nodes B and C divide the corresponding subsets $\{D, E\}$ and $\{F, G\}$ using feature f_2.

when using HOG vectors or other large descriptors. The search is conducted using a priority queue containing candidate nodes ranked according to their distance to the query sample. The ranking determines the order in which the nodes are examined. During the search, the siblings of the current node being examined are iteratively added to the priority queue. The search is terminated when the algorithm scans a predefined maximum number of nodes E_{max}.

During the testing phase, the k Nearest Neighbors (k_{NN}) are retrieved for each candidate **x**. The vote of the class $l_i \in L$ is incremented for each $k_{NN} \in l_i$. The vote is weighted with the reciprocal of the distance d to **x**. The maximum vote determines the class of **x**.

$$\text{vote}_{l_i} = \sum_{1}^{L} \frac{1}{\text{d}(k_{NN}, \mathbf{x})}, k_{NN} \in l_i \tag{2.27}$$

The Euclidean distance is the most common similarity measure. However, its performance is compared to that of other metrics in Chapter 3.2.

2.2.4.2 Random Forest

Random Forests were introduced by Breiman and Cutler [Breiman 01]. An extensive description is given in [Ho 95]. An ensemble of random trees forms a random forest.

A random tree is grown as follows:

- A subset $I \subset I_N$ of training samples is randomly chosen with replacement. The tree is grown using this subset and is not pruned.

- In each node, a subset F of features is randomly chosen. The current data subset is split into I_l and I_r using the feature $f \in F$ and threshold $t \in [min(f), max(f)]$ with the maximum Information Gain Δ. The entropy E is calculated over the labels

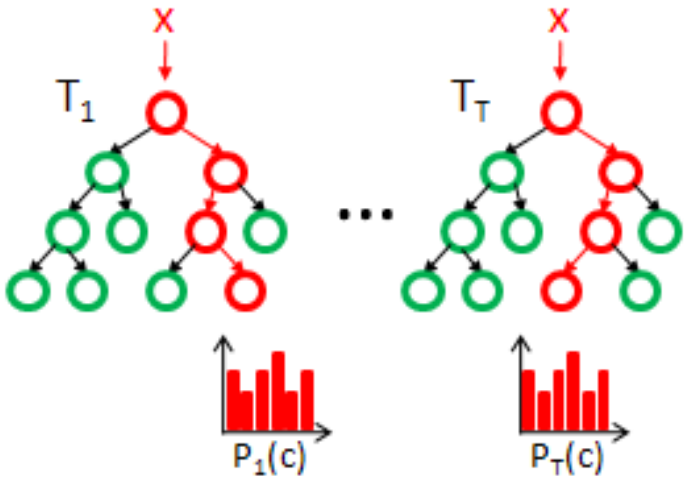


Figure 2.18: Classification of a sample x in Random Forest

frequencies in I.

$$I_l = \{i \in I | f_j < t\}, \forall 1 \leq j \leq |F| \quad (2.28)$$

$$I_r = I \setminus I_l \quad (2.29)$$

$$\Delta = -\frac{|I_l|}{|I|}E(I_l) - \frac{|I_r|}{|I|}E(I_r) \quad (2.30)$$

$$f_j{}^{opt} = max_j \Delta \quad (2.31)$$

The size of F is an empirical choice. Too many features make the training slow and risk over-fitting. The smaller F, the stronger the randomization, the faster the training, yet the higher the risk of under-fitting. In [Geurts 06], the *totally randomized trees*, with $|F| = 1$, yield higher classification accuracies for certain datasets.

Figure 2.18 illustrates the classification of a sample x in a Random Forest. Each of the random trees $T_j \in \{T_1, T_2, \ldots, T_T\}$ in the Random Forest is traversed. In the leaf attained in T_j , the posterior probablity that x belongs to the class $l \in \{1, 2, \ldots, L\}$ is denoted by $P_j(l|x)$. The class l^* of x is determined using the combined decision of the ensemble of trees.

$$P(l|x) = \frac{1}{T}\sum_{j=1}^{T} P_j(l|x) \quad (2.32)$$

$$l^* = max_L P(l_i|x) \quad (2.33)$$

The random forests achieve state-of-the-art performance in many multi-class classification applications. They are used for tracking the keypoints of an object in [Lepetit 06] and clustering visual words in [Moosmann 07]. Shotton et al. [Shotton 08] use Random Forests to cluster and classify *semantic texton*, which represent patches of the image, to segment images and classify the objects associated with the textons.

Using sample subsets for tree training improves the generalization capacity and reduces the memory and processing requirements. In [Bosch 07a], the random forests outperform the SVM in classifying images from the Caltech-101 and Caltech-256 data

Figure 2.19: The ETH80 [Eichhorn 04] data set contains images of 8 classes of objects taken at 41 different angles.

sets. Khoshgoftaar et al. show in [Khoshgoftaar 07] that random forests perform well on binary classification problems with imbalanced data sets and outperform SVM, Naives Bayes, k_{NN} and C4.5 classifiers. A further advantage is that they are fast to build, easy to implement in a distributed computing environment and they enable online learning.

2.3 Comparing on Benchmarks

Two of the major challenges facing classification algorithms are the different view points of the objects and the unbalanced data sets. The first problem requires discriminant features which are robust to rotations and affine transformations. The second problem demands classifiers which can handle classes with unbalanced cardinalities. We evaluate the performance of various features and classifiers using the ETH 80 and the Caltech 101 data sets, which simulate these two difficulties.

2.3.1 Data Sets

The ETH80 data set was first introduced in [Eichhorn 04]. This dataset consists of images of 8 classes, with 10 objects each. Each object was photographed at 41 different angles i.e. a total of 3280 images. Figure 2.19 illustrates some examples of these objects. This data set is often used in literature to benchmark classification algorithms. A drawback, however, is that some of the objects are toys and the background is uniform. This makes it less suitable to simulate a real-world classification problem. However, the 41 different viewpoints make it possible to evaluate the robustness of the features and classifiers to rotations and affine transformations.

The Caltech 101 data set [1] contains images of 101 categories of objects, gathered from the internet. There are between 40 and 800 samples per class, with 50 images on average. This data set is often used for benchmarking classification and segmentation methods. However, Ponce et al. [Ponce 06] constate that although the Caltech 101 data set is very diverse, it lacks variety in terms of orientation, size and background of the objects. Figure 2.20 shows the average images for the Caltech 101 classes.

[1] http://www.vision.caltech.edu/Image_Datasets/Caltech101/

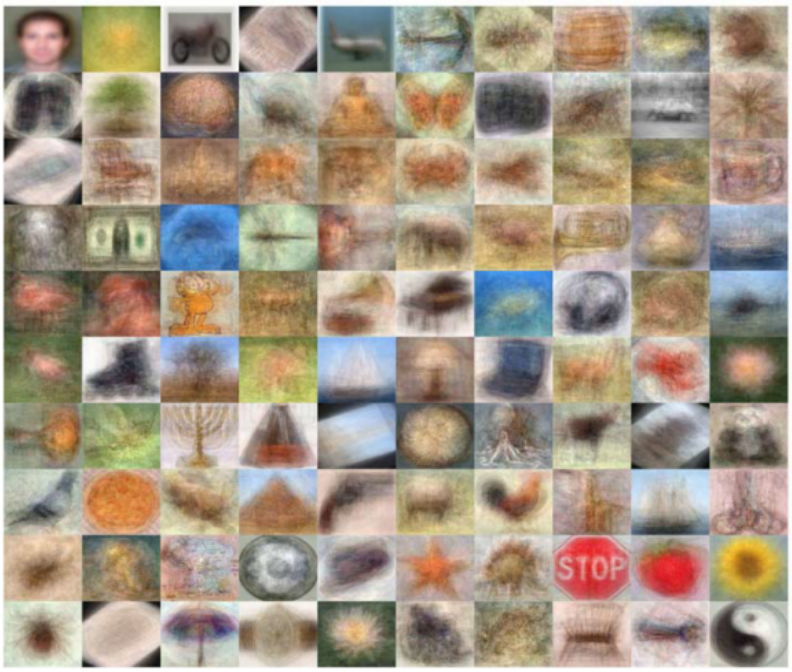


Figure 2.20: Average images of Caltech 101 classes [Ponce 06]. This data set is diverse, yet it lacks variety in orientation, size and background of the objects.

2.3.2 Performance Evaluation

In this section, we compare the features and classification approaches presented in the state-of-the-art using the ETH80 and Caltech data sets. We further evaluate the performance of the HOG feature and tree classifiers: K-d tree and Random Forest. Refer to Sections 2.1 and 2.2 for more details on the features and classifiers used in this section.

2.3.2.1 ETH80

Eichhorn et al. [Eichhorn 04] use eight one-vs-all SVMs with a Bhattacharyya kernel to classify the SIFT features extracted from the ETH80 images. They use the Harris corner detector to extract 40 interest points on average. The classifiers are evaluated using a leave-one-object-out crossvalidation: the training is performed on 79 of the 80 objects and the classifier is tested using the 41 images of the remaining object. This is repeated for all 80 objects in the data set. They achieve an average accuracy of 74%.

Grauman et al. [Grauman 07] repeat the same experiment using one-vs-all SVMs and PCA-SIFT features. They use the pyramid matching kernel, which maps the sets of features to multi-resolution histograms. The similarity of these histograms is defined as their weighted intersection at the different pyramid levels. They show that this method approximates the best partial matching of the two feature sets. The advantage of their method is that it runs in linear time with respect to the number of features m. The Bhattacharyya kernel requires $O(dm^3)$, where d is the feature dimension. They achieve an accuracy of 83% when using 153 interest points on average and 73% when limiting the average number of features to 40 per image.

The same experiment was repeated by Suard et al. [Suard 06] using Histogram of Oriented Gradients (HOG) and graph features with 28 one-vs-one SVMs. The HOG features used consist of 8464 values: 96x96 pixel images, 4x4 pixel cells, 8x8 pixel blocks,

Source	Feature	Classifier	Kernel	Accuracy
[Eichhorn 04]	SIFT (40 points)	one-vs-all SVM	Bhattacharyya	74%
[Grauman 07]	PCA-SIFT (153 points)	SVM	pyramid matching	83%
[Grauman 07]	PCA-SIFT (40 points)	SVM	pyramid matching	73%
[Suard 06]	Graph	one-vs-one SVM	graph	78%
[Suard 06]	HOG (8464)	one-vs-one SVM	linear	**90.1%**
[Suard 06]	HOG (8464) + Graph	one-vs-one SVM	linear and graph	**94.1%**
This thesis	HOG (8464)	K-d tree	-	89.4%
This thesis	HOG (8464)	Random Forest	-	83.2%
This thesis	HOG (784)	SVM (one-vs-all)	linear	87.90%
This thesis	HOG (784)	SVM (one-vs-all)	χ^2	89.02%
This thesis	HOG (784)	K-d tree	-	**92.23%**
This thesis	HOG (784)	Random Forest	-	86.49%

Table 2.2: ETH 80 classification results

4x4 pixel block stride and 4 orientation bins. The graph feature is extracted from the object skeleton. It describes the coordinates of the vertices and the orientation, length, strength and area of the edges. They achieve an accuracy of 78% when using an SVM and the graph kernel. The HOG features are combined with a linear SVM to achieve an accuracy of 90.1%. The combination of both features performed best with an accuracy of 94.1%.

We evaluate the K-d tree and Random Forest classifiers using the same dataset and HOG features as in [Suard 06]. Various K-d tree parameters are tested and $k_{NN} = 5$ and $E_{max} = 1000$ are chosen empirically as they achieve the best results. The Random Forest contains 100 trees, each built with subsets of minimum 10 samples and 10 features. Further, we train the classifiers on smaller HOG descriptors with only 784 values: 64x64 pixel images, 8x8 pixel cells, 16x16 pixel blocks, 8x8 pixel block stride and 4 orientation bins. The average results of the leave-one-object-out crossvalidation of all the approaches are shown in Table 2.2.

A single K-d tree achieves a classification rate of 89.4%, which is comparable to the 90.1% achieved by the 28 one-vs-one SVMs in [Suard 06] using the same sized HOG descriptors. The Random Forest with 100 trees yields a slightly lower accuracy of 83.2%.

When using the only 784 dimensional HOG features, i.e. the coarser spatial partitioning, the performance of the K-d tree was improved to 92.23%. The Random Forests also achieve a high classification accuracy of 86.49%. The eight one-vs-all linear SVMs performed better with a χ^2 than with a linear kernel with 89.02% and 87.9% respectively.

2.3.2.2 Caltech 101 data set

The Caltech 101 is commonly used to benchmark features and classification algorithms. We present the state-of-the-art techniques and perform our own evaluation using SVM, K-d trees and Random Forests. We use the HOG/PHOG features, which will also be used for the pedestrian detection and traffic sign recognition applications in Chapters 3 and 4 respectively. The results of all the approaches presented in the following are obtained by using 30 images for training and the rest for testing.

Lazebnik et al. [Lazebnik 06] extract a predefined number of interest points. They

then compute the corresponding SIFT features. Next, the image is divided into three resolution levels. The histograms of the SIFT features are calculated and concatenated from each of the subregions. They present two methods for generating the SIFT features: "weak" and "strong" features. The former are the SIFT points whose gradient magnitude exceeds a threshold. As for the latter, the SIFT features are densely calculated over a grid. A k-means clustering is then performed on a random subset of the training examples to generate the vocabulary set. A one-vs-all SVM is trained for each class. The pyramid matching kernel is used. This calculates a weighted histogram intersection at each level of the pyramid. The weight of a level is inversely proportional to the respective cell width. A four level pyramid of "strong" features yields a classification rate of 64.6% on the Caltech 101 data set.

The appearance features App, consisting of densely computed and quantized SIFT features, are also used in [Bosch 07b]. Further, they introduce global and class-specific level weights for combining the four layers of the pyramid. The χ^2 kernel is used in the one-vs-all SVM classifiers, achieving a classification rate of 68.1% using the grayscale Caltech 101 images. Bosch et al. [Bosch 07b] also evaluate the four level PHOG feature and yield an accuracy of 69% on the Caltech 101 data set. They further combine the two PHOG features $Shape_{180}$ (gradients orientation in $[0° - 180°]$) and $Shape_{360}$ (gradients orientation in $[0° - 360°]$), with the App_{gray} and App_{color} (App computed on gray and color images respectively). These four features combined achieve an accuracy of 77.8%.

Bosch et al. [Bosch 07a] use the $Shape_{180}$, $Shape_{360}$, App_{gray} and App_{color} features to train Random Forest classifiers. The similarity measure used in their experiments is the exponential of the weighted sum of differences of the histograms at each pyramid level. They grow 25 trees per feature and merge the weighted outputs into a final classification decision. The size of the training set is augmented by generating 10 new images per training class. This methodology yields an accuracy of 80%. The one-vs-all SVM yields 81.3% on the same data set. However, they establish that the computational cost is reduced by factor 40 when using the Random Forests rather than the SVM.

Gehler et al. [Gehler 09] use boosting to learn the combination weights of the different features in the one-vs-all SVM classifiers. The feature weights β are sparse, meaning that for $\beta_f = 0$, the feature f is not considered in the classification function. They evaluate the performance of various feature combinations, including PHOG, the appearance descriptors App mentioned earlier, region covariance and Local Binary Patterns (LBP). They yield a classification rate of 77.7% on the Caltech 101 data set by selecting 7 out of the 39 given feature kernels.

We compare the performance achieved by the SVM, K-d tree and Random Forest classifiers. We use the publicly available PHOG features[2] used in [Gehler 09]. The pyramid level L has 2^L cells in each dimension and the histograms contain K bins. Therefore, the PHOG descriptor vector at the level L contains $K \cdot 4^L$ values. We use the directed gradients, with orientations that lie in $[0° - 360°]$ and $K = 40$ bins. We also combine the descriptors of all four pyramid levels in $L0 - 3$. The results are shown in Table 2.3.

The higher pyramid levels, i.e. the finer image partitioning, achieve the better results. The SVMs yield the best accuracy rates of 68.5% using the PHOG L3 descriptor. The Random Forest has a poorer performance of 45.44% when using the $L3$ level and only 30 training images. However, it is shown in [Bosch 07a], that its performance is improved on

[2]http://people.ee.ethz.ch/ pgehler/projects/iccv09/

	L0	L1	L2	L3	L0-3
PHOG descriptor size	40	160	640	2560	3400
SVM	21.39%	49.13%	65.36%	68.50%	81.16%
K-d tree	18.82%	31.51%	35.84%	25.68%	27.06%
Random Forest	26.41%	40.0%	48.44%	45.44%	52.64%

Table 2.3: Classification results of SVM (χ^2 kernel), K-d tree and Random Forest on Caltech 101 data set using PHOG features (Pyramid levels L0-3).

the more complex Caltech 256 data set when increasing the number of training samples. They use the PHOG and PHOW features in their experiments. One can conclude that the SVM has a higher computational complexity [Bosch 07a], yet is more performant when using small training sets.

The combination of all the pyramid levels with equal weights achieves the highest accuracies when using the SVM and Random Forest classifiers, with 81.16% and 52.64% respectively. The K-d tree performance suffers when increasing the size of the PHOG descriptor, i.e. when using $L3$ and $L0-3$, because of the accumulation of the Euclidean distance over the large dimensionality. For example, two samples may have small differences in several values of their descriptors, yet these are accumulated in the ample dimensionality, resulting in a large Euclidean distance between the two vectors.

2.4 Conclusions

This chapter presents an overview of some of the common features and classification algorithms used in the state-of-the-art. The features can be divided into two main categories: the local, describing patches of the image and the global, describing the entire image. Examples of the former include the interest points, while the latter comprises the Histograms of Oriented Gradients (HOG). As for the classification techniques, we describe the binary and some multi-class variants of Adaboost, the Support Vector Machines (SVM) and the tree classifiers: K-d tree and Random Forest.

The choice of the features and classifiers largely depends on the objects to be classified as well as the performance requirements and system constraints. The HOG feature proved to represent the image classes efficiently both in our evaluation and the state-of-the-art. The performance of the SVMs and the tree classifiers depends on the cardinality and distribution of the training data. The SVM yields high performance rates, yet incurs important computational costs. The tree classifiers are an efficient alternative. The Random Forests attain high accuracy rates at a lower cost. However, additional samples may need to be generated, as they require adequate amounts of training data.

Chapitre 3

Classification: Identifier l'inconnu

Dans ce chapitre, nous examinons trois différents aspects du processus de classification: i) les caractéristiques, ii) la comparaison de ces caractéristiques au sein d'un classifieur à l'aide de métriques et iii) la combinaison de différents classifieurs.

Tout d'abord, nous évaluons la réduction de la dimension des vecteurs caractéristiques pour réduire les besoins en mémoire et le temps de calcul. Les composantes de ces vecteurs sont triées en fonction de leurs importances via l'utilisation de *Random Forests* ou du critère de Fisher. Nous validons cette approche sur deux bases d'images publiques: ETH80 et Caltech 101. Les caractéristiques utilisées ici sont les histogrammes de gradients orientés (anglais: *Histogram of Oriented Gradients (HOG)*) et leurs pyramides (en anglais: *Pyramid Histogram of Oriented Gradients (HOG)*). Nous examinons l'effet de cette réduction sur la performance de classifieurs à vaste marge (en anglais: *Support Vector Machines (SVM)*) et nous obtenons des résultats tout aussi satisfaisants en n'utilisant qu'une sous-partie de l'espace des caractéristiques.

Ensuite, nous examinons l'importance de la métrique utilisée pour comparer les caractéristiques de deux objets. Nous comparons les performances obtenues en utilisant la distance euclidienne, la corrélation et la distance χ^2 comme mesure de similarité entre deux vecteurs caractéristiques. Nous évaluons également leur impact sur la performance des *K- d trees*. Les résultats obtenus indiquent que la corrélation et la distance χ^2 sont plus adequates pour la comparaison d'histogrammes qu'une simple distance euclidienne.

De plus, dans le cadre d'une application de détection de piétons par des caméras statiques, nous filtrons les fausses alarmes fixes comme les arbres et les lampadaires, en utilisant la corrélation entre les vecteurs de caractéristiques. Nous évaluons cette méthode sur les séquences de film CAVIAR, qui ont été prises par une caméra de vidéosurveillance fixe dans un centre commercial. Comme résultat, le taux de précision est doublé tout en gardant un taux de rappel élevé.

Quant à l'ADAS, nous proposons d'utiliser un mélange d'experts consistant en un premier détecteur de piétons suivi d'un arbre de filtrage de fausses alarmes. Nous évaluons cette méthode sur les séquences de film de *Daimler*, qui ont été prises par une caméra embarquée dans une voiture. Le nombre de fausses alarmes par image est réduit de moitié, tout en conservant le taux de rappel.

Nous concluons ce chapitre en proposant des extensions possibles, comme l'étude de l'effet de la réduction de la dimensionalité des vecteurs sur d'autres types de caractéristiques et avec d'autres classifieurs, comme des réseaux de neurones ou Adaboost.

Chapter 3

Classification: Identifying the Unknown

Contents

3.1 Importance of the Feature 36
3.1.1 Feature Ranking Techniques 36
3.1.1.1 Feature Selection 36
3.1.1.2 Algorithms and Applications 36
3.1.2 Evaluation 39
3.1.2.1 ETH80 data set 39
3.1.2.2 Caltech 101 data set 40
3.2 Importance of the Metric 41
3.2.1 Similarity Metrics 41
3.2.2 Comparison on Benchmark Data Sets 43
3.2.3 Correlation for Video Surveillance 44
3.3 Mixture of Experts 47
3.3.1 Pedestrian Detection Techniques 47
3.3.1.1 Single Classifier Detectors 47
3.3.1.2 Mixture of Experts 48
3.3.2 Our Approach for False Alarm Elimination 49
3.3.2.1 HOG/SVM and K-d tree 49
3.3.2.2 HOG/SVM and Random Forest 51
3.4 Conclusions and Perspectives 52

We examine three different aspects of the classification process in this chapter: i) the feature, ii) the comparison of the feature in the classifier and iii) the combination of different classifiers. In Section 3.1 we evaluate the reduction of the feature space to reduce processing and memory requirements. In Section 3.2 we evaluate the importance of the metric used for comparing the descriptors of two objects. We evaluate its effect on the K-d tree classification results and the filtering of false alarms in a pedestrian detection application for static cameras. In Section 3.3, we use a mixture of experts in a pedestrian detection application. The first classifier detects the pedestrians, while the second tree classifier filters the false alarms.

3.1 Importance of the Feature

The goal of feature selection is, on the one hand, to reduce the feature space, which accelerates the training and testing phase and reduces the memory requirements. On the other hand, it improves the classification performance by retaining only the most important features. It also helps in understanding the underlying structure of the generated features.

3.1.1 Feature Ranking Techniques

The general outline of the feature selection process is given in this section. Further, an assortment of four commonly used feature selection algorithms and their respective applications are presented.

3.1.1.1 Feature Selection

The four steps of the feature selection process, as described in [Liu 05] and illustrated in Figure 3.1 are:

I **Subset Generation** consists of selecting a representative subset of features. This search can be performed by progressively adding features to an empty subset (forward search) or by removing features from the entire set (backward search) or in both directions (bidirectional search). There are several ways to perform the search. A complete search, such as Branch and Bound, guarantees an optimal solution. The sequential search, such as the greedy hill-climbing approach, may not find an optimal subset, but is fast and simple. The random search avoids local optimas by either generating random subsets or injecting random features into the subsets generated by other methods.

II **Subset Evaluation** analyzes the quality of the generated feature subsets using the distance, information, dependency or consistency measures.

III **Stopping Criteria** determines the end of the search process. It is often a predefined number of iterations, error rate or no significant improvement over several iterations.

IV **Result Validation** determines the quality of the generated feature subset with respect to a classifier. The decrease of the classification error is used as a goodness measure.

3.1.1.2 Algorithms and Applications

In the following, we present four algorithms for the subset generation and evaluation steps of the feature selection process. Moreover, we describe their applications in the state-of-the-art, as well as their advantages and possible issues.

Relief Algorithm

The Relief algorithm [Kira 92] ranks the features f according to their quality w_f. A data sample x is selected at random. It's contribution to the quality of f is the distance to the nearest correct classification hit, and the nearest misclassification $miss$, as shown in Equation 3.1. This is repeated m times, where m cannot exceed the number of training instances.

$$w_f = \sum_{i=0}^{m} |(x_f)_i - (miss_f)_i| - |(x_f)_i - (hit_f)_i| \tag{3.1}$$

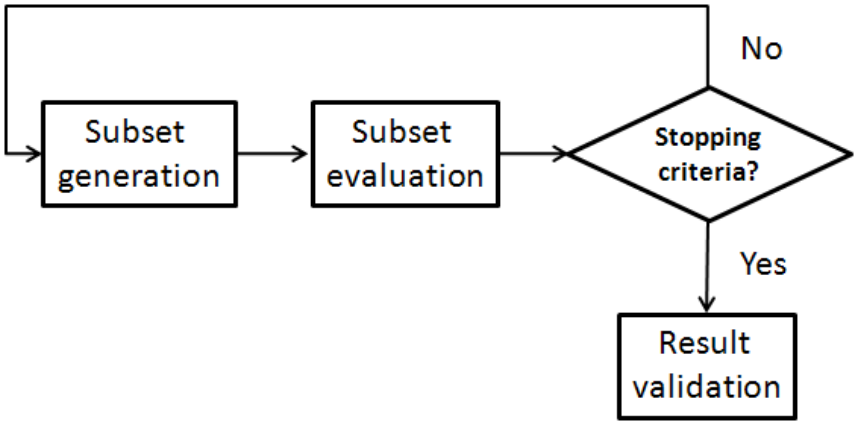


Figure 3.1: Feature selection process

In [Kononenko 94], Kononenko et al. establish that the Relief algorithm is sensitive to outliers and redundant attributes. They overcome this problem by taking the k Nearest Neighbors into account. They develop two multi-class extensions of the Relief algorithm: i) the nearest miss is considered as the nearest neighbor with a different class, ii) the nearest miss from each other class is taken into account.

Ribeiro et al. [Ribeiro 05] select suitable features for low-level, short-term human activity recognition in video sequences. They propose the Brute Search, the Lite-Search and the Lite-lite search algorithms and compare them to the Relief algorithm. They evaluate the generated feature subsets using the Naive Bayes classifier. The Brute search evaluates all possible combinations of the K features. The Lite-search finds the single best feature f_1 using a Brute search and then finds another feature $f_2 \in K \setminus \{f_1\}$ that forms the best pair with f_1. This procedure is repeated until the required subset size n is attained. The Lite-lite search ranks the features individually and selects the best n to form a subset. Ribeiro et al. determine that the Relief algorithm failed in modeling the complex nature of the human activity data and was outperformed by the other proposed approaches. The best performance for this application was attained by the Brute search.

Bins et al. [Bins 01] select relevant features using the Relief algorithm. They then apply the k-means clustering on the correlation amongst the features to remove redundant ones and the Sequential Floating Forward (or Backward) Selection to select the final subset. They yield good results on aerial, handwritten digits and animal images.

In their experiments, Kohavi et al. [Kohavi 97] note that the Relief algorithm shows a high variance in the feature rankings it produces due to the randomness in the generation. They upgrade the Relief to a deterministic version that uses all the samples as well as all the nearest hits and misses. They also state that the Relief algorithm fails to identify redundant features and to remove irrelevant features with a high correlation with the label.

Fisher's Criterion

Fisher's Criterion is a statistical tool for ranking features. One of its advantages is that it is simple to implement and efficient. The Linear Discriminant Analysis (LDA) finds a linear transformation of the feature space that maximizes the Fisher criterion [Duda 01]. This can be used for feature space reduction. Given a sample set X containing L classes and K features. Fisher's criterion $F(j)$ of a variable j is defined as the ratio of the inter-class

variance S_B to the intra-class variance S_W:

$$F(j) = \frac{S_B(j)}{S_W(j)},\ j \in \{1, \ldots, K\} \tag{3.2}$$

$$S_B(j) = \sum_{k=1}^{L} |X^k|(\mu_j^k - \mu_j)^2 \tag{3.3}$$

$$S_W(j) = \sum_{k=1}^{L} \sum_{x \in X^k} (x_j - \mu_j^k)^2 \tag{3.4}$$

where $X^K = \{x \in X | class(x) = k\}$ contains the samples x of the class K, μ_j^k and μ_j are the averages over the j-th feature of the samples X^K and X respectively. The numerator indicates the variance between the classes, while the denominator indicates the variance within each class.

According to [Guyon 03], Fisher's Criterion may be preferable to other approaches, because it is computationally efficient, as it computes and sorts only n scores, and robust against overfitting. However, it does not consider the characteristics of the underlying classifiers, which might be a disadvantage in some applications.

In [Sahoolizadeh 08], the face recognition was performed by sequentially applying the Principal Components Analysis (PCA) to extract the most important features, LDA for selecting the significant features in terms of class separability and a neural networks to classify the images using the reduced feature set. This approach yields high results on the YALE face dataset and outperforms the Eigenfaces and Discrete Cosine Transforms (DCT) approaches. Yang et al. [Yang 08] use a 10-dimensional feature space extracted using LDA to classify human actions using motion sensor data. In [Wu 03], Wu et al. use the Fisher Criterion to reduce the feature space in their palm recognition application and yielding high accuracy rates.

Support Vector Machines

Mutch et al. [Mutch 06] use Support Vector Machines for feature selection. They use $\frac{L(L-1)}{2}$ one-vs-one linear SVMs to learn L classes. The length of the normal of the separating hyperplane in the dth dimension is used to weight the d^{th}feature. The wider the margin in the dth dimension, the better the separability between the two classes and the more important the dth feature. They yield high classification accuracies on the Caltech-101 objects and the UIUC car image dataset.

In [Weston 01], the linear and non-linear SVM feature selection techniques outperform Fisher's Criterion and Pearson's Correlation for face and pedestrian detection as well as cancer classification. In [Rakotomamonjy 03], a wide spectrum of feature selection criteria using SVMs are tested on several benchmarks. They conclude that the change of the weight vector norm $\bigtriangledown||w||^2$ has the best overall performance.

Random Forests

As mentioned earlier, each tree in the Random Forest is constructed using a subset $I \subset I_N$ training samples randomly chosen with replacement. During the validation phase, the out-of-bag (oob) data, which is left out during training, is passed through each tree T_t and the respective classification error ϵ_t is estimated. To compute the variable importance v_k, each feature f_k of the oob data is randomly altered. The permuted samples are passed through each tree again and the difference in classification error is computed.

Algorithm 7 Determining Variable Importance using Random Forests

$oob = \{x_1, \ldots, x_O\}$ with $x = \{f_1, \ldots, f_K\}$
Random Forest $F = \{T_1, \ldots, T_T\}$
compute $\epsilon', \forall x \in oob$
for k=1 to K **do**
  for r=1 to R **do**
    randomly permute f_k for $\forall x \in oob$
    compute ϵ_r for $\forall x \in oob$
  end for
  $v_k = \frac{1}{R} \cdot \sum_{r=1}^{R} |\epsilon_r - \epsilon'|$
end for

The average error over all trees T_t in the Random Forest determines the importance v_k of variable f_k. Algorithm 7 summarizes the variable importance computation using Random Forests.

3.1.2 Evaluation

In this section, we focus on two ranking methods to select a subset of adequate features: Fisher Criterion and Random Forests, as they yield a high performance in the state-of-the-art. We evaluate the effect of reducing the feature dimension on the classification accuracy of the SVM classifiers. For this, we use the HOG and Pyramid of Histograms of Oriented Gradients (PHOG) features, as they yield good results in [Bosch 07b, Bosch 07a]. These are dense features, i.e. they are computed over the entire image and hence, have very large descriptors, making it all the more important to reduce the feature dimension.

The Random Forests and Fisher's Criterion feature selection techniques are also used in the context of traffic sign recognition in Chapter 4, as the memory and processing constraints are important issues in embedded systems.

3.1.2.1 ETH80 data set

As shown in Chapter 2.3, the one-vs-all SVMs yield high accuracy rates on the ETH80 data set when using the 784 value HOG features. We further evaluate the effect of reducing the feature dimension by selecting a subset of the highest ranked features. Figure 3.2 illustrates the feature ranking using Fisher's Criterion. Note that the variable importance peaks recur every 112 values. This coincides with the borders of the block columns (7 blocks x 4 cells x 4 bins = 112 values). Therefore, the borders of the images, such as the legs of the dog or the handle of the cup, are more important for the classification of the ETH80 objects than the central regions.

Figure 3.3 shows the effect of reducing the feature space dimension on the classification accuracy of the different object classes. Note that, with exception of the cow class, the performance remains unchanged when reducing the feature dimension from 784 to 484 values. The average accuracy rates over all the classes when using 784, 684 and 484 dimensional features, chosen using Fisher's Criterion, lie at 89.02%, 88.05%, 88.54% respectively.

The training and testing of the one-vs-all SVM classifiers requires 13.31 seconds on average when using the entire 784 values of the HOG descriptor. This is reduced to only 6.93 seconds when using a subset of 484 values. We use the Matlab implementation of

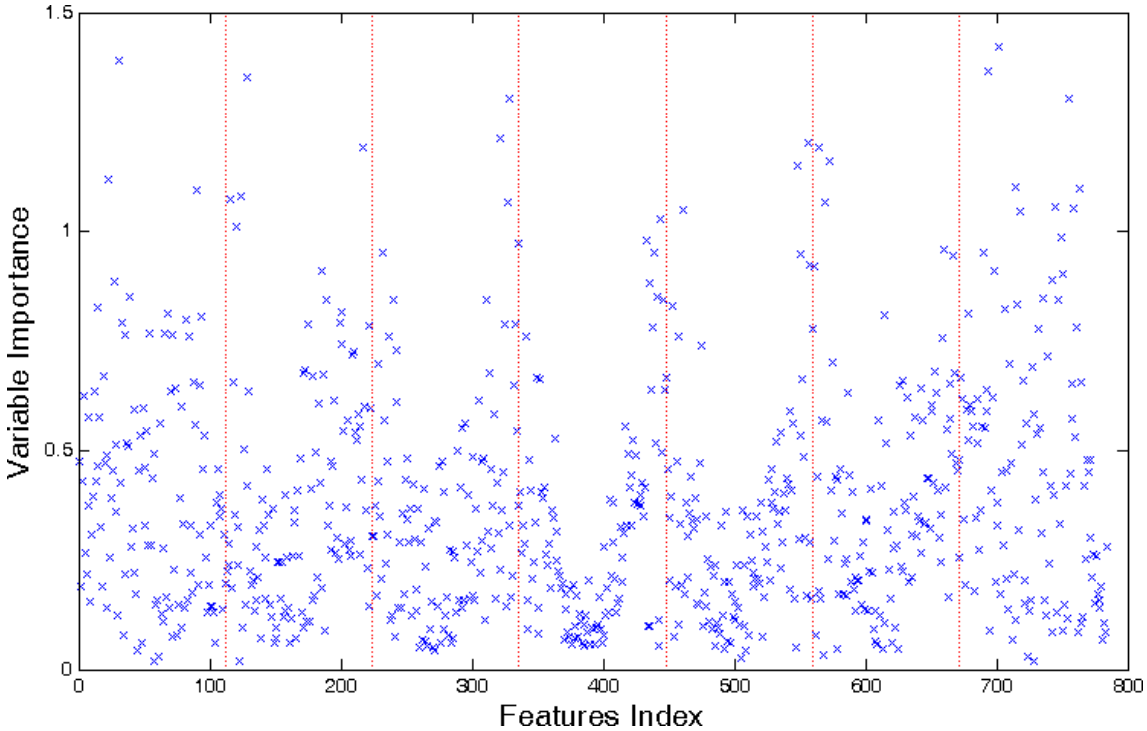


Figure 3.2: Feature ranking using Fisher's Criterion on ETH80 data set. The peaks in the variable importance recur every 112 values (one column of blocks composed of 7 blocks x 4 cells x 4 bins), coinciding with the blocks near the border.

the Pegasos solver used in [Shalev-Shwartz 07], as it optimizes the training process. The experiments were run on a 2.93 GHz Intel Core i7 PC.

3.1.2.2 Caltech 101 data set

As shown in Chapter 2.3 the one-vs-all SVM with the χ^2 kernel also yields high accuracy rates on the Caltech 101 data set when combined with the PHOG features. The four pyramid levels $L0$ to $L3$ as well as their combination $L0-3$ are evaluated. The improvement in classifier accuracy induced by the increasing the image partitioning from level $L2$ to level $L3$ is relatively small compared to that procured by using level $L1$ instead of level $L0$. This can be explained by examining the feature importance of the histogram bins of the four pyramid levels. Figure 3.4 illustrates the feature ranking using Fisher's Criterion. The lower pyramid level bins have a higher ranking than those of the higher levels, i.e. they are more important for the differentiation between the object classes. Hence, we propose to select the most significant features to reduce the feature dimension and processing requirements.

Further, we evaluate the effect of reducing the feature dimensionality on the performance of the one-vs-all SVMs with the χ^2 kernel. As shown in Figures 3.5, the accuracy remains almost constant when reducing the PHOG $L3$ feature subset size from 2560 to about 1000 features. The performance drops steeply when eliminating some of the most important 500 features. A similar trend can be observed when selecting a feature subset from the $L0-3$ feature vector as shown in Figure 3.6. There is a small drop in accuracy of about 1% to 3% when reducing the number of features from 3400 to only 1000. The slope is steeper when selecting less than the 1000 best features.

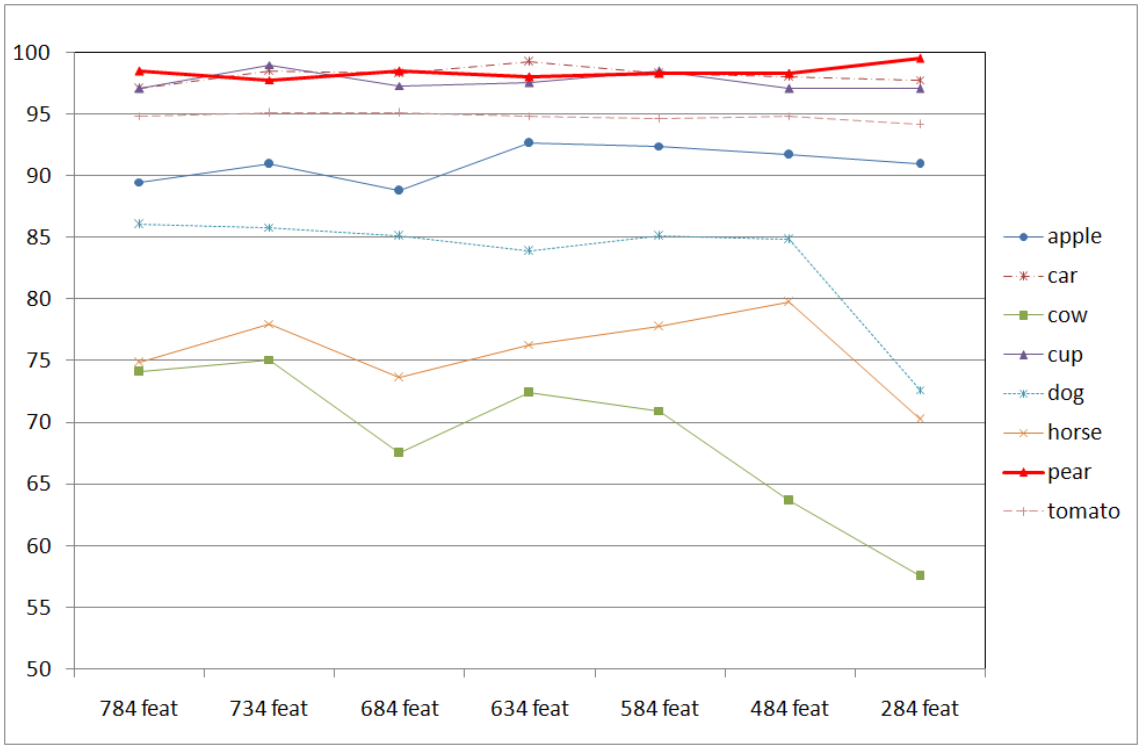


Figure 3.3: Effect of reducing the feature dimension using Fisher's Criterion on SVM accuracy on ETH80. One-vs-All SVMs trained in a leave-one-out scheme for 80 objects of 8 classes. Used 784 value HOG feature and χ^2 kernel.

The total time needed to train the SVM classifiers and test the remaining samples drops from 671 seconds when using all 3400 features of the PHOG $L0-3$ descriptor to only 415 seconds and 243 seconds when using a subset of 2400 features and 1400 features respectively.

Hence, the same high accuracy can be achieved with smaller feature vectors and in less training and testing time for both the ETH80 and Caltech 101 data sets.

3.2 Importance of the Metric

After having established the importance of the feature selection, we now turn to its evaluation in the classification process. In this section, we present three similarity metrics used in the Nearest Neighbor classifiers to compare two object descriptors. We evaluate their effect on the K-d tree classification accuracy. Further, we integrate the similarity measure into the pedestrian detection application for static cameras to eliminate false alarms.

3.2.1 Similarity Metrics

We examine alternative distance measures which can, amongst other applications, be used in the Nearest Neighbor search in the K-d tree: Euclidean d_{Euclid}, Correlation d_{corr} and Chi Squared d_{χ^2}.

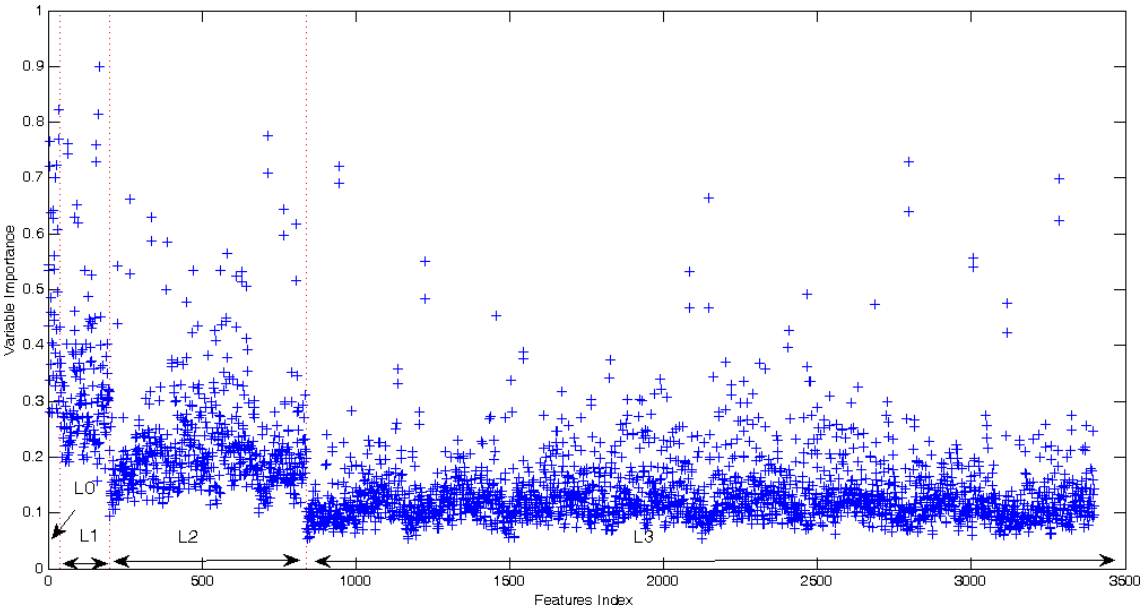


Figure 3.4: Feature Ranking of PHOG $L0-3$ on Caltech 101 data set using Fisher's Criterion. The lower pyramid levels are more important i.e. contribute more to the classification decision than the higher levels. Therefore, the improvement in classification accuracy is larger when using $L1$ instead of $L0$ than when using $L3$ instead of $L2$.

Euclidean

The Euclidean distance is defined as the square root of the sum of the squared differences between the two D-dimensional vectors x and y.

$$d_{Euclid}(x,y) = \sqrt{\sum_{i=1}^{K}(x_i - y_i)^2} \tag{3.5}$$

Correlation

Pearson's correlation coefficient $\rho(x,y)$ is defined as the covariance between two variables x and y divided by the product of their standard deviation σ. The corresponding distance d_{corr} is defined in Eq. 3.6.

$$d_{corr}(x,y) = 1 - \rho(x,y) = 1 - \frac{cov(x,y)}{\sigma_x \cdot \sigma_y} \tag{3.6}$$

It is invariant to linear transformations of the variables x and y, which makes it more suitable for the Nearest Neighbor search when using the HOG feature. For example, two samples with similar trends but different magnitudes have a correlation close to 1, yet a large Euclidean distance. When searching the tree for the Nearest Neighbors of a candidate x, the samples y which minimize the distance $1 - d_{corr}(x,y)$ are retained.

Chi Squared χ^2

The Chi Squared distance measure $d_{\chi^2}(x,y)$ between two variables $x, y \in \mathbb{R}^{\mathbf{K}} \times \mathbb{R}^{\mathbf{K}}$ is defined as

$$d_{\chi^2}(x,y) = \frac{1}{2}\sum_{i=1}^{\mathbf{K}} \frac{(x_i - y_i)^2}{x_i + y_i} \tag{3.7}$$

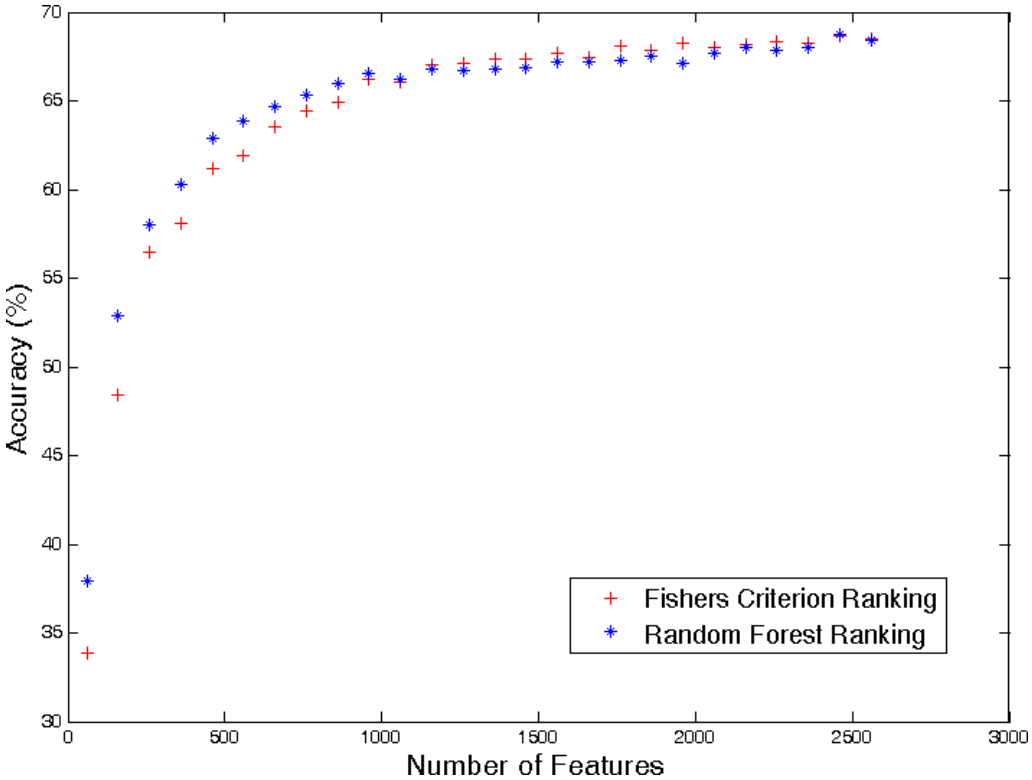


Figure 3.5: Effect of reducing the feature dimension on SVM accuracy on Caltech 101 data set. Feature ranking performed using Random Forest and Fisher's Criterion. One-vs-all SVM with χ^2 kernel and PHOG $L3$ feature used.

Domeniconi et al. [Domeniconi 02] use it to improve the Nearest Neighbor classification of data sets. Further, it is used in [Zhang 07] to compare two histograms for texture classification. It is shown to outperform the Chamfer distance in [Bosch 07b], yielding good results on the Caltech-101 and Caltech-256 image datasets using the PHOG feature.

3.2.2 Comparison on Benchmark Data Sets

We examine the effect of the similarity measure on the performance of the K-d tree classifier. As shown in Chapter 2.3, the K-d tree with the Euclidean similarity measure achieves a classification rate of 92.23% on the ETH80 and 27.06% on the Caltech 101 data set. The results obtained when using the Euclidean, Correlation and χ^2 similarity measures are listed in Tables 3.1 and 3.2 for the ETH80 and Caltech 101 data sets respectively.

The K-d tree performance does not vary significantly on the ETH80 data set when using the different similarity measures. The highest accuracy of 92.71% is attained by the χ^2 similarity measure. As for the Caltech 101 data set, the results are improved significantly, by up to 26%, when using the χ^2 and $correlation$ similarity measures. The improvement is stronger when using the finer image partitioning levels. This is because the histogram differences are relativized when using these similarity measures as opposed to the Euclidean distance. The highest results attained by the K-d tree are obtained when using the $correlation$ similarity measure and the PHOG $L3$ level.

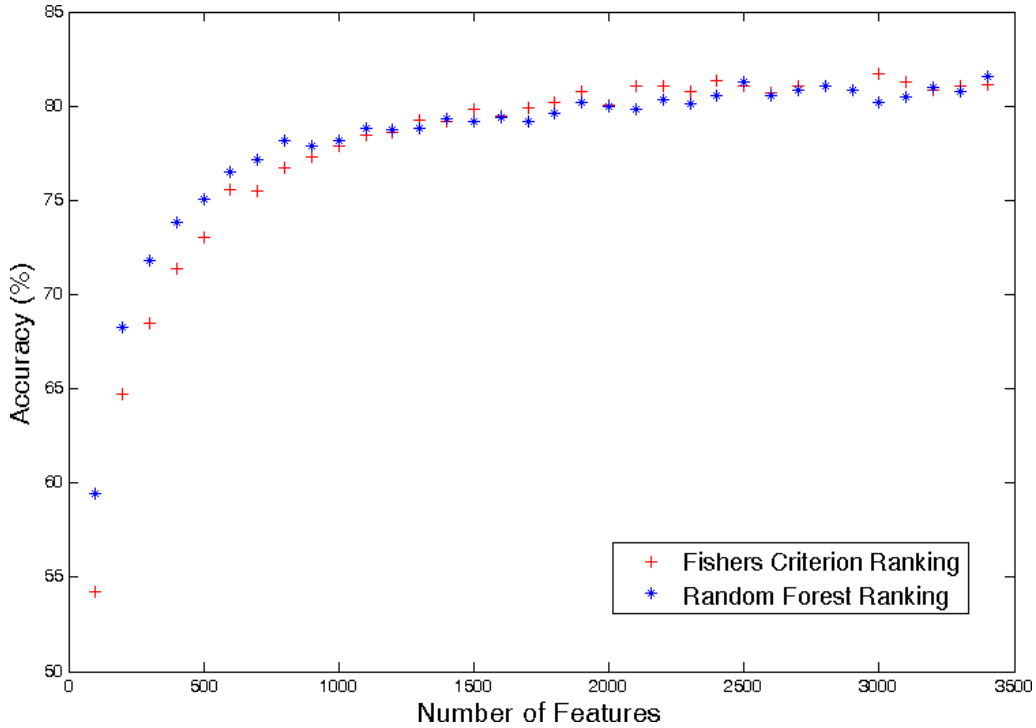


Figure 3.6: Effect of reducing the feature dimension on SVM accuracy on Caltech 101 data set. Feature ranking performed using Random Forest and Fisher's Criterion. One-vs-all SVM with χ^2 kernel and PHOG $L0-3$ feature used.

Similarity Measure	Accuracy
Euclidean	92.23%
Correlation	91.01%
χ^2	92.71%

Table 3.1: K-d tree classification results using different similarity measures on ETH 80 data set using 784 valued HOG descriptor.

3.2.3 Correlation for Video Surveillance

The video surveillance product offered by our project partner, *Connected Objects*, informs the users of the presence of people near their home: whether the children are back from school, the housekeeper arrived or intruders lurking in the backyard. The images from various fixed cameras in and outside the house are captured periodically (one image every few seconds). The ones containing people are stored for further use.

A HOG/linear SVM detector is used to detect people in the images, as it yields the best performance in the state-of-the-art. The advantage of the HOG descriptor is that it is resistant to variations in illumination due to the block normalization of the gradients. This is particularly important during the rapid changes in illumination at dusk and dawn.

In general, the false alarms obtained from the people detector are often static objects which resemble people, such as trees and fences. Hence, we apply a HOG correlation measure to reduce false alarms by eliminating the immobile candidates. The HOG descriptor of a detected bounding box is compared to that in the previous image. If the correlation between these two regions at the same location in consecutive images is suf-

	L0	L1	L2	L3	L0-3
PHOG descriptor size	40	160	640	2560	3400
Euclidean	18.82%	31.51%	35.84%	25.68%	27.06%
Correlation	19.14%	37.71%	49.71%	51.64%	33.91%
χ^2	21.60%	35.84%	46.33%	45.31%	49.40%

Table 3.2: Classification results of K-d tree with different similarity measures on Caltech 101 data set using PHOG features (Pyramid levels L0-3).

ficiently high, the candidate is eliminated as part of the background, since insufficient movement has been determined. The assumption that pedestrians are mobile is applicable due to the fact that the cameras are static and that the images are taken at an interval of a few seconds.

The disadvantage of this approach is that non-moving people are falsely eliminated as background. This problem can be overcome by using a longer history span for the comparison. Another drawback is that it is not applicable in embedded systems as such. A complementary computation of the camera's egomotion is needed to determine the location of the detected candidate in the previous image.

The Detector

This approach is validated on the public benchmark dataset CAVIAR[1], containing 26 video sequences captured by fixed video surveillance cameras at a shopping center in Portugal. Only non-occluded pedestrians are considered. The minimum required overlap ϵ of a candidate found and the ground truth is set to

$$\epsilon = \frac{\text{candidate} \cap \text{ground truth}}{\text{candidate} \cup \text{ground truth}} >= 25\% \tag{3.8}$$

The detector performance is evaluated using the recall and precision values. These are defined as following:

$$\text{recall} = \frac{\text{true positives detected (TP)}}{\text{total true positives}} \times 100\% \tag{3.9}$$

$$\text{precision} = \frac{\text{true positives detected (TP)}}{\text{all detections}} \times 100\% \tag{3.10}$$

The linear SVM/HOG detector is trained on the data set from [Enzweiler 09] using a similar procedure to that described in [Dalal 05]. A positive:negative ratio of 50:50 is used, with 15660 samples for each class. The bootstrap of negative samples is chosen at random. A second SVM is trained using the original 15660 positive samples and a new set of 15660 negative false alarms generated by the first detector on the training set. The training samples are resized to 24x48 pixels. The cell, block and stride sizes are set to 4, 8 and 4 respectively. The histograms contain nine orientation bins for the undirected gradients.

The Results

The effect of the correlation threshold ϵ on recall and precision is shown in Figure 3.7. For example, $\epsilon = 0.1$ requires a detection to be correlated by at least 10% with same

[1] http://homepages.inf.ed.ac.uk/rbf/CAVIAR/

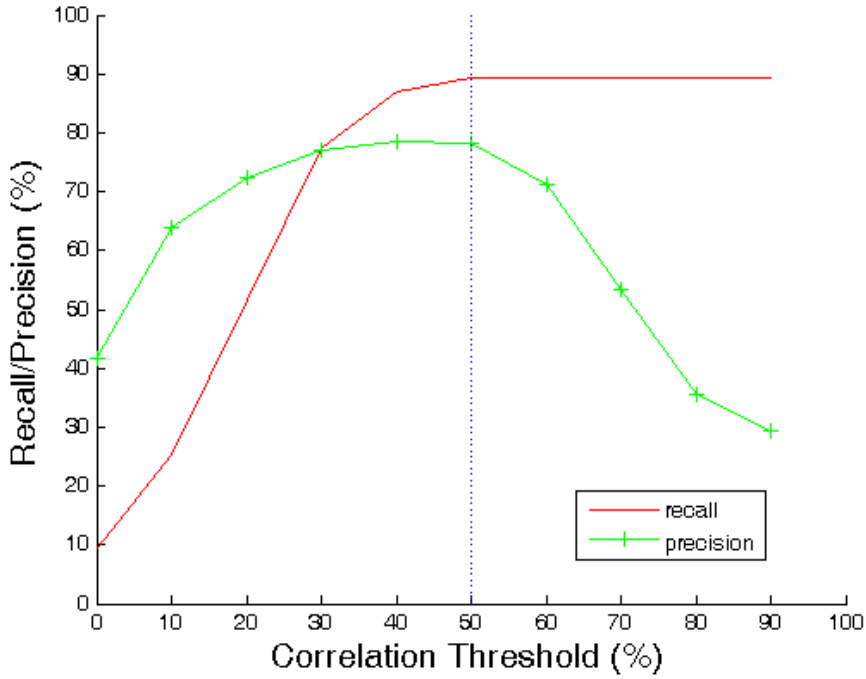


Figure 3.7: The effect of the correlation threshold on recall and precision on One Left Shop Reenter1 (Corridor) *sequence.*

area in the previous frame. This weak constraint falsely eliminates many detections as false alarms. When $\epsilon = 0.9$, a high correlation is necessary, i.e. hardly any movement in this area is tolerated. This eliminates less false alarms, yet retains all the detected pedestrians. Therefore, when increasing the correlation threshold ϵ, the recall rate rises. The precision is increased at first, as the number of correct detections rises, yet drops at a $\epsilon > 0.5$, due to the saturation of the recall and the continuous increase of the number of false alarms.

The HOG correlation threshold ϵ is set to 0.5, as it achieves the best trade-off between recall and precision. The average results over all 26 image sequences are shown in Table 3.3. The precision is increased by 18.94% when using the HOG correlation since many false alarms, such as the railing and the trash can, illustrated in Figure 3.8, are eliminated.

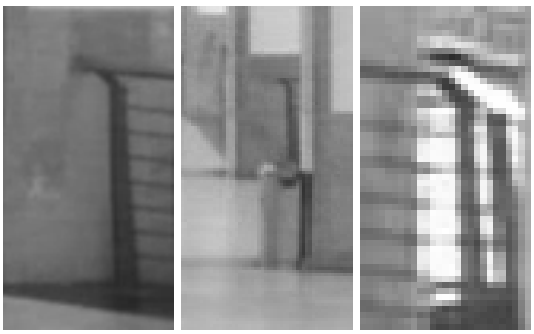


Figure 3.8: The railing and the trash can are examples of false alarms in the CAVIAR dataset eliminated by HOG correlation.

Similarity Measure	Average Recall	Average Precision
None	64.93%	23.22%
Euclidean	58.42%	37.88%
Correlation	**62.22%**	**42.16%**
χ^2	37.98%	39.77%

Table 3.3: Comparison of the HOG similarity measures for eliminating static false alarms on CAVIAR benchmark sequences for pedestrian detection.

The average recall rate dropped slightly, by only 2.71%, when using the HOG correlation. Unfortunately, the immobile pedestrians, especially the distant ones, are also eliminated by the HOG correlation. Table 3.3 shows that the correlation similarity measure yields better results than the Euclidean and χ^2 distances.

3.3 Mixture of Experts

A fast and reliable pedestrian detector is required in ADAS. A large part of the false alarms obtained by the linear SVM/HOG detector are trees, poles, tires and traffic signs, due to their person-like silhouette. These recurring errors can be learned and eliminated. We propose a mixture of heterogenous experts, in which the output of the linear SVM detector is filtered by a subsequent tree classifier. The SVM learns to distinguish the general form of the pedestrians, while the tree classifier compares the detected candidates with known positive or negative samples to eliminate recurring false alarms.

We start this section by presenting the state-of-the-art approaches for pedestrian detection using single classifiers and mixtures of experts in Section 3.3.1. We then describe and evaluate our proposed approach in Section 3.3.2.

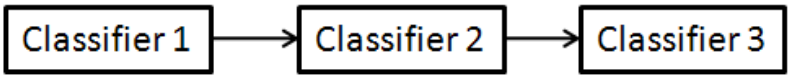
3.3.1 Pedestrian Detection Techniques

The task of pedestrian detection is challenging because of the non-rigid nature and the common occlusions of humans as well as the diversity of their clothing, shapes and sizes. Several algorithms, benchmark datasets and evaluation techniques for pedestrian detection have been developed over the past decades. Amongst the most popular are Adaboost, Hierarchical Template Matching using Distance Transforms [Gavrila 99], as well as Support Vector Machines (SVMs) trained on Histogram of Oriented Gradients (HOG) descriptors [Dalal 05]. Several recent works have combined heterogenous classifiers and features, in a serial or parallel manner, to form a mixture of experts, as illustrated in Figure 3.9. The cascading of heterogenous classifiers refines the candidate set and reduces the false alarms. In this section, we present a brief overview of existing single binary classifiers, used for pedestrian detection, and their combination into ensembles.

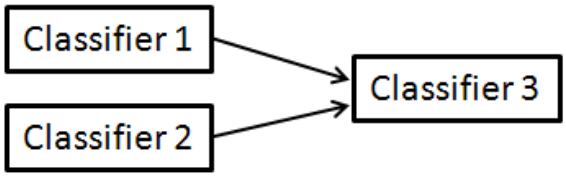
3.3.1.1 Single Classifier Detectors

The Adaboost algorithm [Viola 01] (cf Chapter 2.2.1) is combined with the Haar-like features in [Viola 05] to detect far pedestrians in low resolution images. Their computation is accelerated by using the Integral Image representation. The cascade Adaboost detector is enhanced by integrating the motion information.

In [Moutarde 08], Adaboost is combined with the Control Points features to efficiently



(a) Mixture of Experts in Series



(b) Mixture of Experts in Parallel

Figure 3.9: Mixture of Experts

detect vehicles and pedestrians in an ADAS framework. As elaborated in Chapter 2.1.1.2, a genetic algorithm is integrated in Adaboost to optimize the selection of suitable Control Points. The pixel-based nature of these features simplify their computation and reduce the memory requirements. We published a comparison of performance of the Haar-like and Control Points features in pedestrian detection in [Zaklouta 09], showing that the latter yields the better results.

The efficient linear SVM/HOG people detector, introduced in [Dalal 05], also achieves high performance rates in [Enzweiler 09]. The Histogram of Oriented Gradients (HOG) feature, describing the orientations of the local gradients in the image, are used to train a linear SVM. The image is scanned at different scales and each candidate is classified by the SVM. Details on the HOG/SVM detector are given in Chapter 2.1.2.1. A comprehensive study on pedestrian classification using HOG and Local Receptive Fields (LRF) features with various SVM kernels is presented in [Paisitkriangkrai 07].

Dollar et al. [Dollar 09] introduce the new Caltech pedestrian dataset and benchmark existing approaches. They establish that the HOG/linear SVM detector performs best amongst the state-of-the-art approaches. We also yield high detection rates when using this detector for traffic sign detection as shown in Chapter 4.

The Co-occurence HOG, introduced in [Watanabe 09], describes the distribution of the oriented gradients within a given range. They used it for training a linear SVM and reduce the miss rate by half on the INRIA and Daimler datasets compared to the HOG descriptor. However, these features are more time consuming to compute than HOG.

The LRF features are combined with Neural Networks in [Enzweiler 09, Wohler 99] and further with Support Vector Machines (SVM) in [Munder 06] yielding slightly better results on the same data set.

3.3.1.2 Mixture of Experts

There are two main categories of ensemble classifiers: in serial or parallel as illustrated in Figure 3.9. As shown in [Kim 02], there are several methods for evaluating the combined decision of the classifiers, such as by using a majority vote, a Least Square Error weighting or a double-layer hierarchical combination, where a classifier in the second layer learns to evaluate the combined decisions of the classifiers in the first layer.

In [Li 10], the Haar and HOG features are combined to train a cascade Adaboost detector. The extracted candidates are further filtered using the Edgelet features around

the head and shoulder region. In [Zhu 06], a set of linear SVM classifiers are trained on HOG descriptors with different size blocks in an Adaboost manner, i.e. in each round, the classifiers are chosen to concentrate on the difficult samples from the previous round. These boosted classifiers are in turn cascaded to reduce the false alarm rate. Gavrila et al. [Gavrila 00] detect pedestrians using the template matching of Distance Transforms and verify the candidates using a Radial Basis Function (RBF) classifier.

In [Sotelo 06], several SVMs are trained on different body parts, like the arms, legs and head. The best suited feature, amongst the Texture Units, Normalized Histograms, Haar wavelets, gradient magnitude and orientation, co-occurence matrix and Canny, is selected for each body part. A further SVM classifier evaluates the outputs of the body-part SVMs in parallel to make a final decision. In [Mikolajczyk 04], Adaboost is used to train seven body part detectors. These are then combined using a joint likelihood body model based on the appearance and position of the parts found. In the binocular setup in [Curio 00], the pedestrians are detected using the texture, contour information and a thresholding of the difference between the images from the two cameras. The detection hypothesis is then validated by tracking the pedestrian's torso and modelling his gait using the leg movements.

3.3.2 Our Approach for False Alarm Elimination

In general, the false alarms obtained from the people detectors are often static objects which resemble people, such as trees, poles or fences. Figure 3.10 illustrates some examples of people and other objects and the corresponding HOG descriptors.

We introduce a new method for eliminating false alarms in embedded systems using a serial, heterogenous mixture of experts. The HOG/linear SVM detector is combined with a tree classifier, K-d tree or Random Forest. The latter filters out the false alarms of the former.

The proposed approach is evaluated on the test dataset published in [Enzweiler 09]. It consists of five image sequences of a total duration of 27 minutes, filmed by an embarked camera in a car, driving through tree-lined suburban roads and crowded downtown streets. Only non-occluded pedestrians with a minimum height of 72 pixels are considered. This corresponds to a height of 1.5 m at a 25 m distance for the camera setup in [Enzweiler 09]. Further, we use the performance evaluation measures and learning procedure described in Section 3.2.3. However, the sizes are adapted to the video sequence. The training samples are resized to 48x96 pixels. The cell, block and stride sizes are set to 8, 16 and 8 respectively.

3.3.2.1 HOG/SVM and K-d tree

The K-d tree is built using the same HOG descriptors used for the SVM training, i.e. 15660 positives and 15660 negatives from the random bootstrapping. We add a further set of 15660 false alarms, which were generated by the trained detector on the negative samples and used to train the second SVM. The Euclidean distance, correlation and χ^2 similarity measures in the Nearest Neighbor search are examined. More details on the K-d tree and the similarity measures can be found in Chapter 2.2.4.1.

Table 3.4 lists the results obtained from the linear SVM with the K-d tree filtering on the five test sequences. Note that the false positive per frame rate is halved when concatenating the K-d tree with the correlation measure. There are significantly more

[2] http://opencv.willowgarage.com/wiki/

(a) Pedestrian gradient image (b) Open legs more difficult (c) Weak contrast difficult

(d) Tire resembles a person (e) Default detector in OpenCV (f) Our detector

Figure 3.10: a)-c) Examples of pedestrians and the corresponding gradient images, d) Example of false positive that resembles a pedestrian, e) Positive weights of default HOG descriptor gradients learned by SVM and provided in OpenCV library[2] f) Positive weights of HOG descriptor gradients obtained from our SVM training on [Enzweiler 09]

false alarms in the crowded downtown sequences 2 through 4. On average, the linear SVM generates 0.67 false positives per frame. The K-d tree with the correlation similarity measure reduces this to only 0.33 false positives per frame. Figure 3.11 shows some examples of the false alarms correctly filtered out by the K-d tree with the correlation similarity measure.

Unfortunately, in some sequences the detection rate tends to suffer due to the filtering. On average, the recall of the linear SVM is reduced by only about 6% when using the correlation measure. The largest decrease in recall, of 3% to 10%, occurs in the downtown sequences 2 to 5 due to the frequent partial occlusion and grouping of the pedestrians. Sequence 4 is particularly crowded with groups of pedestrians.

Other misclassifications include pedestrians with open legs, partially occluded or with a weak constrast with the surroundings. The clutter in the background and the inaccurate centering of bounding box around the pedestrian due to the regrouping of several detections also lead to a misclassification of the extracted candidates. Some examples of misclassified detected pedestrians are shown in Figure 3.12. The drop in recall can be avoided by enriching the train data set with samples of pedestrians within a group.

The parameters of the K-d tree were varied and those attaining the best results were

SVM			SVM & K-d tree Correlation		SVM & Random Forest	
Seq	Recall	fppf	Recall	fppf	Recall	fppf
1	71.1%	0.218	70.2%	0.117	69.3%	0.156
2	78.5%	0.977	73.2%	0.492	75.7%	0.831
3	80.4%	0.764	70.7%	0.429	78.1%	0.659
4	77.5%	1.220	74.8%	0.529	76.5%	1.087
5	78.4%	0.176	67.2%	0.086	76.9%	0.134
Average	77.2%	0.67	**71.2%**	**0.33**	75.3%	0.573

Table 3.4: Recall and false positives per frame (fppf) on five test sequences using an SVM with and without tree filtering. $k_{NN} = 4000$ *and* $E_{max} = 40000$*. Random Forest 100 trees.*

Figure 3.11: Examples of false alarms, such as trees, tires and poles, correctly filtered by K-d tree with the correlation similarity measure. $k_{NN} = 4000, E_{max} = 40000$*.*

kept. Hence, the number of nearest neighbors is set to $k_{NN} = 4000$ and the number of candidates examined is set to $E_{max} = 40000$. The average time required to test a detected candidate lies at 298 ms. This high latency is due to the large size of the tree. Therefore, the K-d tree filtering approach is not feasible in real-time in crowded scenes.

The Euclidean and χ^2 similarity measures achieve an average recall of 69.8% and 68.1% respectively. The false positive per frame rates lie at 0.65 and 0.61 respectively. They require 102 ms and 601 ms respectively to classify a candidate.

To conclude, the performance of the K-d tree depends on the extent of the provided training set, the quality of the preceding detector and the number of candidates extracted at runtime.

3.3.2.2 HOG/SVM and Random Forest

The performance of the Random Forest as a post-detection filter is also evaluated. The structure and parameters of the Random Forests are described in Chapter 2.2.4.2. The best results are obtained by a Random Forest with 100 trees, 100 samples each, 100 variables selected at each node and a depth of 100 nodes. The trees are built using the positive samples and the false alarms generated by the final linear SVM classifier. The processing time required per extracted candidate is less than 1 ms. On average, a recall rate of 75.3% and 0.573 false positives per frame are achieved. Hence, although the Random Forest eliminates less false alarms than the the K-d tree with the correlation similarity measure, the recall rate is less affected and the classification is feasible in real-

(a) Weak contrast (b) Sprawled legs (c) Unfamiliar

(d) Bad centering of bounding box (e) Groups of pedestrians

Figure 3.12: Examples of detected pedestrians falsely filtered by K-d tree with the correlation similarity measure. $k_{NN} = 4000, E_{max} = 40000$.

time.

3.4 Conclusions and Perspectives

The contribution in this chapter is three-fold. In Section 3.1, we show that a high performance can be achieved with a fraction of the feature dimension, reducing the memory and processing requirements. The subset of the most important features is selected using the Random Forests or Fisher's Criterion feature ranking methods. We evaluated this feature space reduction approach using two classification benchmark data sets: ETH80 and Caltech 101.

Further, three distance metrics are compared and their effect on the K-d tree classification accuracy is evaluated using the ETH80 and Caltech 101 benchmark data sets. We use the correlation measure to eliminate immobile false alarms in a pedestrian detection application for static cameras in Section 3.2. This technique doubles the precision rate while retaining a high recall rate. Moreover, we propose and evaluate a second method for eliminating false alarms in an embedded ADAS system in Section 3.3.2. The HOG/linear SVM pedestrian detector is complemented with a tree classifier containing the false alarms generated by the former. This mixture of experts reduces the false positives per frame rate by more than half, while maintaining a high recall rate.

Future work could include the evaluation of the effect of the feature space reduction

on other classifiers, such as Neural Networks and Adaboost. One can also consider using the feature space reduction to accelerate and eventually improve the pedestrian detection.

Moreover, the temporal information could be exploited to track the detected pedestrians. This could improve the recall and precision values. The performance of the tree classifiers, used as filters in the post-detection phase, could be improved by enriching the training data set and using online learning, where samples are added to the K-d tree or new random trees are built in the forest. Further, different image features can be used in the tree filtering stage to increase the robustness of the system.

The SVM and tree classifiers, as well as the feature space reduction and the different similarity measures are used in the scope of the traffic sign recognition system described in Chapter 4.

Chapitre 4

Reconnaissance de Panneaux

La reconnaissance de panneaux routiers est un élément crucial dans un système d'aide à la conduite. Elle informe le conducteur des limites de vitesse et des dangers imminents. Malheureusement, tenter de réaliser cette tâche fait apparaître plusieurs obstacles: i) la mauvaise qualité des images disponibles à cause d'une basse résolution, de mauvaises conditions météorologiques ou de variations d'illumination, ii) la rotation, occlusion ou déterioration des panneaux, iii) les contraintes en mémoire et capacités de calcul imposées dans les systèmes embarqués, iv) les variations en taille, couleur et type de panneaux entre les différents pays.

Nous commençons ce chapitre avec un état de l'art spécifique à la reconnaissance de panneaux routiers. Ensuite, nous proposons un système temps réel. Celui-ci se compose de trois étapes: la segmentation, la détection et la classification. La segmentation réduit l'espace de recherche. Nous améliorons l'approche de Ruta et al. [Ruta 10], *color enhancement*, en introduisant un seuil adaptatif. Nous comparons ceci avec des autres approches de segmentation pour la détection de panneaux.

Dans la deuxième étape, un détecteur de cercles et un de triangles en forme de séparateur à vaste marge (en anglais: *Support Vector Machines (SVM)*) linéaires et le caractéristique des histogrammes des gradients orientés (en anglais: *Histogram of Oriented Gradients (HOG)*), sont appliqués pour trouver les panneaux. La combinaison de la segmentation et la détection atteint des taux de rappel et de précision de 90% sur nos séquences vidéo de test.

Le contenu des panneaux est classé par des classifieurs multiclasses. Nous comparons les performances de classifieurs à base d'arbres- *K-d tree* et *Random Forest* - à celles des SVMs. Nous proposons une pondération spatiale des composantes des vecteurs caractéristiques, pour concentrer la mesure de similarité entre vecteurs sur les parties les plus importantes. Ceci améliore la performance des K-d tree de 15% sur le German Traffic Sign Recognition Benchmark. Les *Random Forest* atteignent un taux de classification de 97% sur cette même base.

En outre, nous proposons de réduire la dimension des vecteurs caractéristiques en choisissant les composantes les plus pertinentes à l'aide de *Random Forests* ou du Critère de Fisher. Ceci réduit les besoins en mémoire et en puissance de calcul. La performance des SVMs est même améliorée sur ces sous-ensembles de caractéristiques.

Chapter 4

Real-Time Traffic Sign Recognition

Contents

4.1 Existing Approaches 56
4.1.1 Determining the Location using Segmentation 58
4.1.2 Category Determination using Shape Detection 58
4.1.3 Classification Techniques 59
4.2 Our Approach 62
4.2.1 Image Segmentation 62
4.2.1.1 Color Enhancement 62
4.2.1.2 Chromatic Filter 63
4.2.1.3 Morphological Filters 64
4.2.2 Category detection 64
4.2.2.1 Data set and Evaluation 64
4.2.2.2 Implementation of the Detector 65
4.2.2.3 Use of Segmentation Masks 67
4.2.2.4 Effect of Segmentation on the Detection 68
4.2.3 Sign classification 70
4.2.3.1 German TSR Benchmark 70
4.2.3.2 Comparison of Classifiers 72
4.2.3.3 Comparison of HOG and Distance Transforms 76
4.3 Overall Performance of the TSR System 78
4.4 Feature Space Reduction 79
4.4.1 Feature Selection 79
4.4.2 Evaluation 81
4.5 Conclusions and Perspectives 81

Advanced Driver Assistance Systems (ADAS) play an important role in enhancing car safety and driving comfort. Some of their components include navigation systems to provide directions as well as up-to-date traffic information and vision-based systems such as lane departure warning systems and traffic sign recognition (TSR).

The latter enhances traffic safety by informing the driver of speed limits or possible dangers such as icy roads, imminent road works or pedestrian crossings. The traffic

Figure 4.1: Examples for difficulties facing the Traffic Sign Recognition (TSR) task: over-illumination, under-illumination, rotation, occlussion and deterioration of the signs.

signs can be divided into different categories depending on their color and shape, e.g. red-rimmed circular prohibition signs, triangular warning signs and blue information signs. The simplified pictograms make them easily perceivable and comprehensible.

Traffic sign recognition algorithms face three main difficulties:

i the poor image quality due to low resolution, bad weather conditions, over- or under-illumination,

ii the rotation, occlusion and deterioration of the signs,

iii the limited memory and processing capacities in real-time applications such as ADAS.

iv the size, font, color, etc. of the signs vary from one country to another.

Some examples of the first two difficulties are illustrated in Figure 4.1. We start this chapter by giving a short overview of the state-of-the-art methods for TSR systems, with their advantages and possible issues. We then describe our contributions the TSR process and how we overcome these difficulties.

We start this chapter by giving an overview of the existing TSR approaches in Section 4.1. Our three stage approach is described in Section 4.2 and the overall performance of the proposed system is presented in Section 4.3. The effect of the feature selection on the performance of the traffic sign classification is evaluated in Section 4.4. We conclude this chapter and present an outlook on further possible improvements in Section 4.5.

4.1 Existing Approaches

Most traffic recognition algorithms divide the problem into three stages:

i a rough segmentation to determine the location of the signs

ii category determination and

iii candidate classification to identify the content of the extracted traffic signs using various machine learning techniques.

The segmentation phase is not a mandatory step. However, it is often deployed in approaches using color images. This section gives a brief overview of some of the techniques used in the three TSR stages.

Some approaches, such as [Jimenez 05, Fang 03] reduce the memory and processing requirements by using tracking. The candidates found are tracked over several frames

Technique	Color Space	Reference	Advantages	Possible Issues
Fixed thresholds for Hue and Saturation combined with achromatic decomposition	HSV	[Bascon 10]	Simple and fast	Not resistant to illumination variations and sign deterioration
Shadow-highlight invariant algorithm: thresholding in H, then S and V	HSV	[Fleyeh 06]	Robust to illumination changes	Time-consuming: sliding window post-processing and region growing
HSI thresholding	HSI	[Qingsong 10, Nguwi 08, Kuo 07]	Hue robust to illumination variation	Time-consuming RGB to HSI convesion; Thresholds set empirically
Multiple fixed thresholds	YUV	[Miura 00]	YUV colorspace invariant to illumination	Thresholds set empirically
Classification of RGB channel ratios	RGB	[Le 10, De La Escalera 97]	Robust due to ratios of RGB and blockwise classification	Requires prior thresholding to accelerate
Color enhancement and recursive thresholding	RGB	[Ruta 11, Ruta 10]	Relative dominance of RGB channels robust and efficient	Recursive thresholding and subsequent region growing slow
Color histograms	RGB	[Lim Jr 10]	Relative chromatic map more robust	Empirical thresholding and erosion for sign and pictogram extraction
Chromatic and achromatic thresholding	RGB	[Bascon 07]	Simple and fast	Thresholds set empirically
RGB color Haar wavelets	RGB & normalized RGB	[Bahlmann 05]	Relative difference in color and feature selection more robust	Minimum sign size 20 pixels, Slow due to preprocessing

Table 4.1: Comparison of state-of-the-art segmentation techniques for Traffic Sign Recognition (TSR).

to reinforce the decision made or to reduce the search space in the subsequent frame and therewith accelerate the performance. Some implementations, such as [Ruta 10], only examine every n-th frame, to speed up the overall system.

4.1.1 Determining the Location using Segmentation

As traffic signs need to be easily perceivable, they are brightly colored in red, blue, yellow. Hence, the detection is often based on a pre-segmentation of the image to reduce the search space and retrieve possible Regions of Interest (ROIs).

Since the direct thresholding of the RGB channels is sensitive to changes in illumination, the relation between the RGB (Red Green Blue) colors is often used. In [Ruta 10], the color enhancement is used to extract red, blue and yellow blobs. This transform emphasizes the pixels where the given color channel is dominant over the other two in the RGB color space.

In [Bascon 07], chromatic and achromatic filters are used to extract the red rims and the white interior of the speed limit and warning traffic signs respectively. The HSI model (Hue Saturation Intensity) is used in [Kuo 07] as it is invariant to illumination changes. Empirically determined fixed thresholds define the range of each HSI channel in which lie the red and blue traffic sign candidates. It is pointed out in [De La Escalera 97], however, that HSI is computationally expensive due to its nonlinear formulae.

Table 4.1 gives an overview of some of the existing segmentation techniques, their advantages and possible issues.

4.1.2 Category Determination using Shape Detection

Several detection algorithms are based on edge detection, making them more robust to changes in illumination. These are shape-based methods which exploit the invariance and symmetry of the traffic signs. They are however often sensitive to the quality of the preprocessing, such as edge extraction or color segmentation.

- **Template matching**
  Franke et al. [Franke 99] use Distance Transform (DT) and Template Matching (TM) to detect circular and triangular signs. Similarly, Ruta et al. [Ruta 10] use the Color Distance Transform, where a DT is computed for every color channel separately. The advantage of matching DTs over edge images is that the similarity measure is smoother and robust to slight rotations. It is, however, sensitive to affine rotations and occlusions. Refer to Chapter 2.1.2.3 for more details.

- **Machine Learning**
  In [Bascon 07], four Support Vector Machines (SVMs) are trained on the Distance to Border (DtB) vectors to classify the shape of an extracted candidate ROI. In [Jimenez 05], the FFT signatures of candidate signs are used to detect relevant shapes. This feature is robust to rotation and scaling, yet not to occlusion and deterioration.

- **Hough Transform**
  The Hough Transform is also widely used to detect regular shapes such as circles and triangles [Moutarde 07, Ruta 09, Kuo 07]. The processing time is decreased by the simpler Radial Symmetry Detector [Barnes 08], yet it is limited to circular traffic signs. Ruta et al. [Ruta 09] refine the Hough Transform result using the

Confidence-Weighted Mean Shift to eliminate redundant detections. The Hough transform is combined with an iterative process of median filtering and dilation to refine the candidate set in [Kuo 07].

- **Shape Detection using Histogram of Oriented Gradients (HOG)**
  Many recent approaches use gradient orientation information in the detection phase. Gao et al. [Gao 06] classify the candidate traffic signs by comparing their local edge orientations at arbitrary fixation points with those of the templates. In [Alefs 07], Edge Orientation Histograms (see Chapter 2.1.2.1 for details) are computed over shape-specific subregions of the image. In [Xie 09, Qingsong 10], the Regions of Interest (ROI) obtained from color-based segmentation are classified using a HOG-based classifier. To integrate color information in the HOG descriptor, Creusen et al [Creusen 10] concatenate the HOG descriptors calculated on each of the color channels. The advantages of this feature are its scale-invariance, the local contrast normalization, the coarse spatial sampling and the fine weighted orientation binning.

Table 4.2 gives an overview of some of the existing shape detection techniques and the features used.

4.1.3 Classification Techniques

The classification techniques used to determine the content of the detected traffic signs belong to two general categories: learning and Nearest Neighbor approaches. The learning consists of finding an optimal separation between two or more classes. It includes, amongst others, one-vs-all SVM classifiers, Adaboost and Neural Networks. The Nearest Neighbor approaches seek the most similar existing training sample to the given unknown. They include template matching and tree classifiers such as K-d trees and Random Forests.

Xie et al. [Xie 09] train the HOG descriptors of each class using one-vs-all SVMs. The Forest-ECOC Adaboost classifiers achieved high performance rates in [Baró 09]. Multilayer Perceptrons (MLPs) yield high accuracy rates in [Hoferlin 09, Nguwi 08]. They also achieve low false positive rates when identifying the characters in speed limit signs in [Bargeton 08]. The performance of the Neural Networks is improved by pre-selecting the color-shape features using PCA and Fisher Linear Discriminant in [Lim Jr 10].

The K-d tree is used in [Kuo 07] to identify the content of the sign. The Random Forests used in [Kouzani 07] outperform the one-vs-all SVMs on their dataset. They also generate an ensemble of SVMs using bagging and a boosted ensemble of Naive Bayes classifiers, which improve the performance of the non-ensemble version. However, these do not outperform the Random Forests.

The advantage of the tree classifiers is that they are easy to train and update. The learning approaches tend to be biased towards over-represented classes and generally require a large training set. As described in Chapter 2.2.4, the Best Bin First Approximate Search [Beis 02] enables a rapid search in the K-d tree. The randomness and ensemble voting of the Random Forests make them robust to outliers and unbalanced datasets.

Table 4.3 gives an overview of some of the classification techniques used for traffic signs, their advantages and possible issues.

Technique	**Reference**	**Advantages**	**Possible Issues**
Fixed thresholds on X and Y axis projections	[Miura 00]	Simple, fast	Sensitive to occlusion, translation and rotation
Threshold on sum of absolute differences of Fast Fourier Transform (FFT)	[Jimenez 05]	Fast FFT computation; robust to rotation and illumination variation	Not robust to occlusions; False alarms; Requires good pre-segmentation
Hierarchical matching of Distance Transforms	[Franke 99, Ruta 10]	Fast, DT is robust to variance due to smooth similarity measure	Not robust to occlusions, sensitive to presegmentation and edge extraction
SVM and Distance to Border	[Bascon 07]	2-D rotation invariance	Four SVMs per class; good pre-segmentation required
SVM and Rim shape recognition using XOR derived invariant features	[Fleyeh 08]	Robust, good performance	Sensitive to pre-segmentation
SVM and HOG	[Xie 09, Chen 08]	Robust	Slow due to sliding window and one 1-vs-all SVM per class
SVM and HOG, concatenated HOG on CIELab & YCbCr color spaces	[Creusen 10]	HOG/linear SVM detector robust	Slow due to large concatenated HOG vectors and color space conversion
Gradient, Angular Radial Transform, Histogram of Phase Angles etc	[Kuo 07]	Robust due to RBF training	Computation expensive
Maximum likelihood voting and Radial Symmetry	[Barnes 08]	Simple, fast	Limited to circular road signs
Hough transforms	[Bargeton 08, Hoferlin 09, Garrido 05, Moutarde 07, Ruta 09]	Robust, simple, applicable to all regular shapes	Parameter tuning

Table 4.2: Comparison of state-of-the-art detection techniques for Traffic Sign Recognition (TSR).

Technique	**Reference**	**Advantages**	**Possible Issues**
1-vs-all SVM (HOG)	[Xie 09, Qingsong 10]	SVM robust to outliers	Many classes → many classifiers → slow
Forest-ECOC	[Baró 09]	Fast, robust	Require large training set
Keypoints matching	[Ren 09]	Fast, rotation, scale and illumination invariant	Minimum sign size: 60 pixels; requires pre-segmentation
MLP	[Hoferlin 09, Nguwi 08]	Robust, outperform SVM in [Nguwi 08]	Computationally expensive
MLP (characters)	[Bargeton 08]	Low FP rate, works on LED signs	Requires good character visibility
Neural Networks (Color-Shape features)	[Fang 03, Ohara 02, De La Escalera 97]	High accuracy; [Fang 03]: tracking to reinforce and accelerate	Require large training set
Committee of Neural Networks (Multilayer Perceptron & Convolutional Neural Networks)	[Ciresan 11]	High accuracy; winner of GTSRB challenge	Expensive in terms of computational time and resources
Multiscale Convolutional Networks	[Sermanet 11]	High accuracy; Runner-up at GTSRB challenge	Expensive in terms of computational time and resources
PCA/FLD & RBF-Neural Networks	[Lim Jr 10]	High performance rates	Computationally expensive pictogram and feature extraction & classification
Genetic algorithm	[De La Escalera 03]	Robust: genetically determine fittest sample during training	Require large training sets; possible local minima
K-d tree	[Kuo 07]	Two classifiers more reliable	Computational and memory costs
Random Forests	[Kouzani 07]	Robust; Handle large unbalanced classes; High hit rates	Require large training datasets and memory

Table 4.3: Comparison of state-of-the-art classification techniques for Traffic Sign Recognition (TSR).

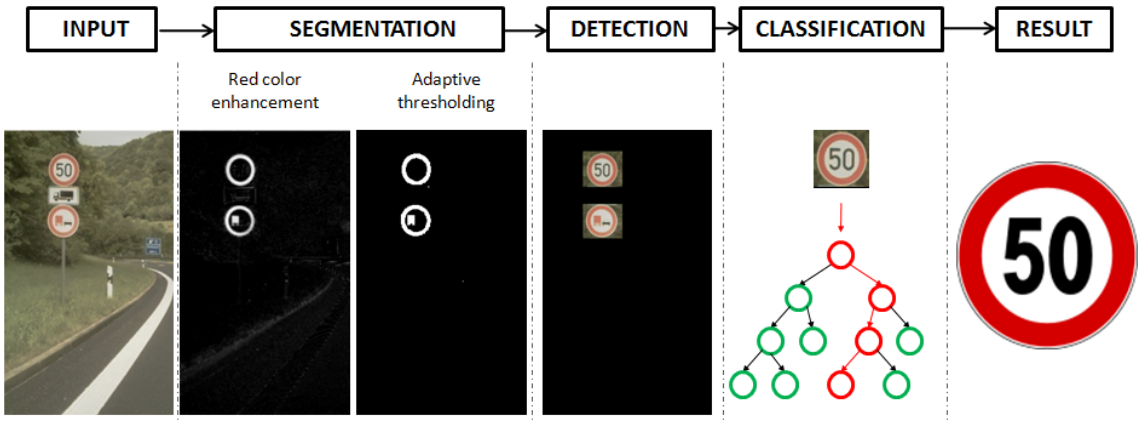


Figure 4.2: Three stage coarse-to-fine approach proposed for Traffic Sign Recognition (TSR): i) Segmentation ii) Detection using HOG/SVM, iii) Classification using tree classifiers.

4.2 Our Approach

We propose a coarse-to-fine traffic sign recognition approach consisting of three stages: i) segmentation, ii) shape detection and iii) classification. These are illustrated in Figure 4.2. The image segmentation reduces the search space to the red regions that potentially contain a traffic sign. We improve the red color enhancement approach used by Ruta et al. in [Ruta 10] by introducing a global threshold. The circular and triangular signs are detected using one linear SVM/HOG detector each. The candidates found are further efficiently classified using tree classifiers. We further introduce spatial weighting techniques to improve the accuracy and feature space reduction for resource optimization.

The performance of each of these three stages is compared to the state-of-the-art techniques. The classification step was also put to the test at the live German Traffic Sign Benchmark Challenge [Stallkamp 11], where our approach was ranked 3rd in terms of accuracy and proved to be suitable for embedded systems as it runs in real-time and is resource efficient.

4.2.1 Image Segmentation

Potential ROIs are extracted in the segmentation phase. We implement the red color enhancement [Ruta 10, Ruta 11], the chromatic filter [Bascon 07] and introduce the morphological filters. After applying one of these filters, the image is thresholded using an empirically determined threshold or the adaptive threshold which we propose. The resulting binary mask is used to reduce the search space and reduce the number of false alarms of the successive detector.

4.2.1.1 Color Enhancement

Color Enhancement is proposed by Ruta et al. in [Ruta 10, Ruta 11]. For each RGB pixel $x = \{x_R, x_G, x_B\}$ in the image, the red color enhancement is provided by

$$f_R(x) = max(0, \frac{min(x_R - x_G, x_R - x_B)}{s}) \quad (4.1)$$

$$s = x_R + x_G + x_B \quad (4.2)$$

(a) Original (b) Enhanced (c) Chromatic (d) Tophat (e) Bottomhat

Figure 4.3: Image segmentation for TSR: ROI extraction using red color enhancement, chromatic, tophat and bottomhat filters to reduce search space and accelerate detection.

The pixels having a dominant red component are extracted, while all others are set to zero. In their approach, the ROIs are found in a subsequent stage by recursive thresholding using a Quad-tree.

In our approach, we accelerate the segmentation stage by eliminating the recursion and applying a global threshold to the enhanced image to generate a binary mask of the ROIs. The threshold used is empirically set to $\mu + 4 \cdot \sigma$, where μ and σ are the mean and standard deviation of the pixel values over the entire filtered image. The global mean helps take into account the illumination of the whole image, while a mean computed locally is more sensitive to small illumination variations in the image.

Figure 4.3b shows the result of the red color channel enhancement. Note that the rims and the red truck pictogram are emphasized. The search space is reduced when applying the binary mask obtained from the red color enhancement. The false alarm rate is also lowered as will be shown in Section 4.2.2.4. The global thresholding approach is faster and more simple than the recursive thresholding using a Quad-tree. It is also robust to local variations in illumination.

4.2.1.2 Chromatic Filter

We further evaluate the chromatic filter [Bascon 07] to determine the ROIs. Bascon et al. [Bascon 07] use the achromatic segmentation masks to extract the white interior of the traffic signs and the chromatic masks for the red rims and yellow construction site signs.

The chromatic color decomposition of an image is computed using

$$f(R, G, B) = \frac{|R - G| + |G - B| + |B - R|}{3D}, \tag{4.3}$$

where D is the degree of extraction of an achromatic color. It is empirically set to $D = 20$ in [Bascon 07]. To extract the chromatic pixels, $f(R, G, B) > 1$ is used.

A threshold of $f(R, G, B) < 1$ extracts the achromatic pixels i.e. those lacking hue. However, the achromatic segmentation is not suitable for our application, since our sequences contain large white areas such as the sky and buildings. We therefore apply the chromatic segmentation to obtain the red parts of the signs. The result of the chromatic filter is shown in Figure 4.3c.

Figure 4.4: Examples of traffic sign detection using HOG/linear SVM. The undirected gradients used in the HOG detector allow for the detection of static and illuminated traffic signs.

4.2.1.3 Morphological Filters

We examine two morphological filters, top-hat and its reciprocal bottom-hat. The former emphasizes light pixels with a high contrast to their local environment, such as the inside of the traffic signs. The latter emphasizes dark pixels with a high contrast to their local environment such as sign rims and some pictograms. The images obtained by these two operators, I_T and I_B respectively, are defined as

$$I_T = I - I_O, I_O = I_D \circ I_E \tag{4.4}$$

$$I_B = I_C - I, I_C = I_E \circ I_D \tag{4.5}$$

where I is the input image and I_O, I_C, I_D, I_E represent the opening, closing, dilation and erosion operators respectively.

The result of the tophat and bottomhat filtering is shown in Figure 4.3d and e. Note that the interiors of the traffic signs are emphasized in the tophat transform. The rims and the pictograms are emphasized in the bottomhat transform. We further apply an empirically determined adaptive threshold $\mu + 4 \cdot \sigma$ to obtain a binary mask designating the ROIs, where μ and σ are the mean and standard deviation of the pixel values over the entire filtered image.

4.2.2 Category detection

In this thesis, we focus on the detection of speed limit and warning signs. To evaluate the proposed detection algorithm and the performance of the color segmentation as a filter, we use image sequences acquired during the daytime in both urban and highway environments under different meteorological conditions. During the detection phase, the image is scanned at multiple scales. Only the areas resulting from the segmentation mask are examined. The resulting candidates are then passed on to the tree classifier in the classification phase.

4.2.2.1 Data set and Evaluation

Our data set contains 24 classes of traffic signs: 12 round and 12 triangular. The train and test data sets contain 14763 and 1584 signs respectively. There is a significant imbalance in the number of training samples for each class as shown in Table 4.4. This imbalance can also be found in the real world, as some signs, such as the speed limits, are more abundant than the $wind$ warning sign for example. The size of the training sets varies from 15 to 3852 images. The *frog*, *traffic jam* and *wind* warning sign have no

testing samples, because they appear in only one sequence, i.e. one physical sign over a period of time. Nevertheless, they are used in the training set to reinforce the shape detector. The image resolution is 752x480 pixels.

A predicted bounding box is considered correct if it overlaps more than 50% of the ground truth.

$$\epsilon = \frac{\text{prediction} \cap \text{ground truth}}{\text{prediction} \cup \text{ground truth}} >= 50\% \tag{4.6}$$

The evaluation is based on the recall and precision values, which are defined as follows:

$$\text{recall} = \frac{\text{true positives detected (TP)}}{\text{total true positives}} \times 100\% \tag{4.7}$$

$$\text{precision} = \frac{\text{true positives detected (TP)}}{\text{all detections}} \times 100\% \tag{4.8}$$

4.2.2.2 Implementation of the Detector

Two detectors are trained, one for the circular and another for the triangular traffic signs. The detectors are trained at an empirically determined positive:negative ratio of 1:10. The false alarms generated by the initial detector on the training dataset are reinjected as new negative samples to train a cascade classifier. This method yields higher recall/precision values than the single layer for pedestrian detection in [Dalal 05].

The Feature: HOG

The HOG descriptor is used, as it is fast to compute and robust to changes in illumination and scale. A further reason for choosing HOG over other features is that when using undirected gradients, i.e. orientations between $0°$ and $180°$, both static (red rim, white interior) and dynamic (illuminated, red rim, black interior) signs can be found with the same detector. Other features, such as Control Points or Haar, would fail in this case due to their use of directed gradients i.e. orientations of $0°$ to $360°$. Examples of static and illuminated signs are illustrated in Figure 4.4.

The Detector: SVM

The computed HOG descriptors are used to train two linear SVM classifiers (one for round signs and another for triangular ones). This significantly reduces the computational costs at runtime compared to the four SVM classifiers trained per shape in [Bascon 07].

The SVMLight [1] library is used. The m resulting support vectors are combined to a single global vector v. Given the SVM classification function $f(x)$ of an unknown sample x from Chapter 2.2.3, with support vectors x_i, Lagrange multipliers α_i and labels l_i.

$$f(x) = sign\Big(\sum_{i=1}^{m} l_i \alpha_i (x_i \cdot x) + b\Big) \tag{4.9}$$

$$\Leftrightarrow f(x) = sign\Big(x \cdot \sum_{i=1}^{m} l_i \alpha_i x_i + b\Big) \tag{4.10}$$

$$\Leftrightarrow f(x) = sign\Big(x \cdot v + b\Big) \tag{4.11}$$

[1] http://svmlight.joachims.org/

Sign	Meaning	Train	Test
	construction	21	2
	danger	1474	149
	deer	30	13
	frog	30	0
	slippery road	338	61
	children	68	25
	no overtaking	15	4
	no overtaking truck	968	179
	pedestrian	78	1
	school	31	1
	sharp curve	131	40
	slope	113	20
	speed 10	23	6
	speed 30	651	111
	speed 40	258	73
	speed 50	1110	207
	speed 60	1499	51
	speed 70	316	46
	speed 80	1900	171
	speed 100	3852	276
	speed 120	1726	126
	traffic jam	18	0
	truck	66	22
	wind	20	0
Total		14763	1584

Table 4.4: The class sizes of Traffic Sign Recognition dataset. There is a large imbalance in the number of train and test samples for 24 traffic sign classes.

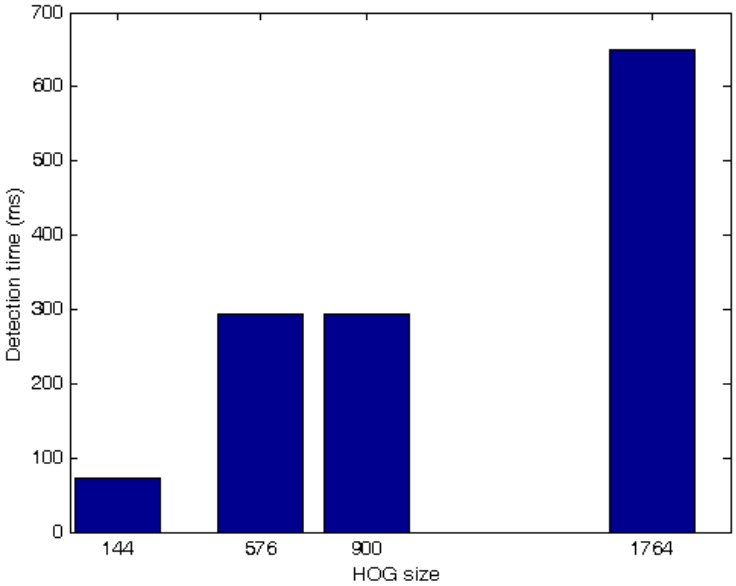


Figure 4.5: Detection times for different HOG descriptor sizes: 144 ($B = 4, S = 4, C = 4$), 576 ($B = 4, S = 4, C = 2$) , 900 ($B = 8, S = 2, C = 4$) , 1764 ($B = 4, S = 2, C = 2$) where B = block size, S = stride, C = cell size in pixels, histograms with 9 bins

This reduction to one support vector accelerates the detection because the HOG descriptor x of each subwindow is compared to a single vector v instead of several.

To determine the optimal size of the HOG descriptor, a preliminary linear SVM is trained on the triangular signs, with a positive:negative ratio of 50:50. The negative samples are generated randomly. The Recall/Precision curves for different HOG descriptor parameters (block, cell and stride sizes) are illustrated in Figure 4.6. All descriptors use 9 bin histograms. Each curve is obtained by varying the classifier threshold in the linear SVM. Note that the finer block and cell partitioning yields higher precision values as they describe the image more precisely.

The drawback of the larger descriptors, however, is that they are more time-consuming to compute and compare. The detection times using the different HOG descriptor sizes and the same SVM threshold are depicted in Figure 4.5. As the speed and memory requirements are important constraints in emarked systems, the smaller and more efficient 144 value vector ($B = 4, S = 4, C = 4$) will be optimized and used for the traffic sign detection in further experiments. Its low precision is improved using the segmentation as shown in Figure 4.8.

4.2.2.3 Use of Segmentation Masks

We propose to use a sliding window on the binary mask resulting from the segmentation in the detection phase. The image is resized and scanned at different scales. The sum of the pixel values $sum(img(x, y, w, h))$ in each subwindow $img(x, y, w, h)$ is computed, where (x, y) and (w, h) are the top left corner, width and height of the subwindow in the image. If the ratio of the white pixels to the area of the subwindow $area(img(x, y, w, h))$ exceeds a given threshold θ as in Equation 4.12, the subwindow is considered to contain a potential sign. The integral image representation of the binary

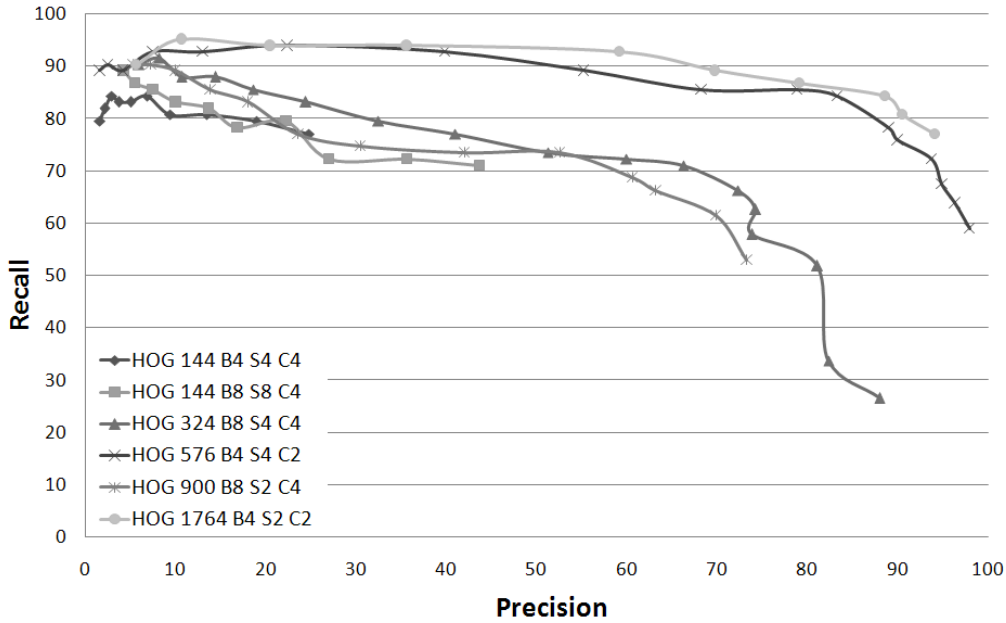


Figure 4.6: Recall and precision for different HOG detector sizes for triangular traffic signs. B = block size, S = stride, C = cell size in pixels.

mask image is used to accelerate the sum computation.

$$\frac{sum(img(x,y,w,h))}{area(img(x,y,w,h))} > \theta \tag{4.12}$$

Hence, the candidate subwindows are passed on to the HOG/linear SVM detectors for cirlces and triangles, which in turn verify the presence of the respective shape.

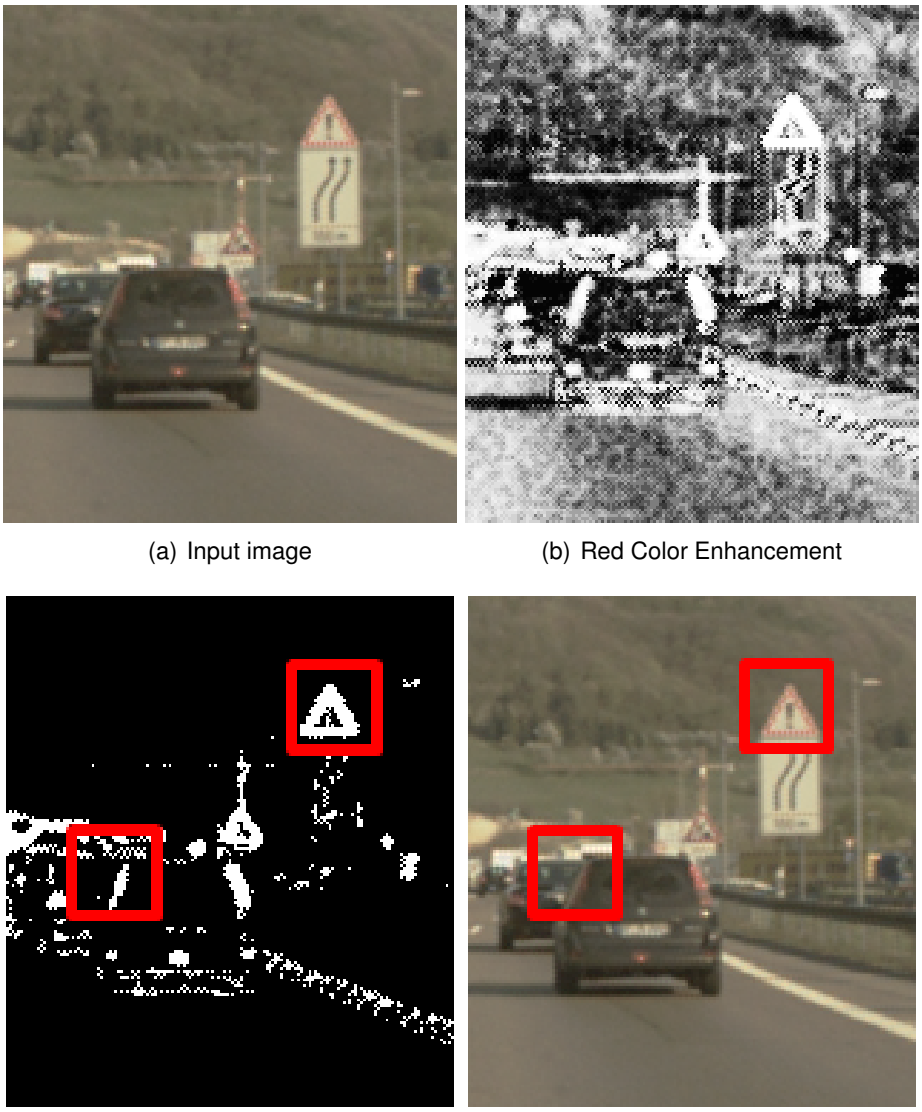
This methodology is illustrated in Figure 4.7. This approach reduces the search space and accelerates the detection. As an alternative, the ROIs can be found using a connected components approach. This could, however, require a longer computation time.

4.2.2.4 Effect of Segmentation on the Detection

The influence of the segmentation masks on the performance of the HOG/linear SVM detectors is illustrated in Figure 4.8. The best trade-off between the recall and precision values as well as the respective average processing times per image are listed in Table 4.5. The red channel color enhancement improves both the recall and the precision of the HOG detector. It yields the best results with a recall of 90.21% at a precision of 90.9%. Note that the HOG descriptors computed on the RGB color space achieve a 10% higher recall than those computed on the grayscale images. This improvement is due to the strong gradients in the red color channel which are diminished in the grayscale images.

Table 4.5 also shows the average time (in ms) required to process a 752x480 pixel frame on a 2.93 GHz Intel Core i7 PC. The grayscale HOG detector achieves real-time performance, running at 35.56 ms per frame (28 frames per second). The improved red enhancement segmentation mask requires a longer processing time, yet can run in near real-time at 48.85 ms per frame (20.5 frames per second).

In general, the segmentation reduces the number of false positives, yet decreases the number of true detections. The morphological operators falsely eliminating signs with a

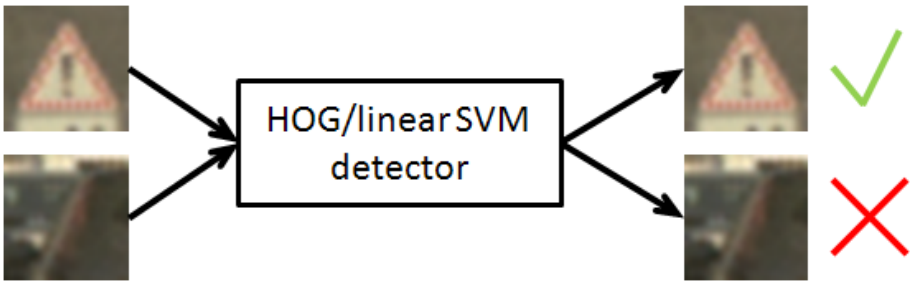


(a) Input image

(b) Red Color Enhancement

(c) Color enhancement image thresholded using adaptive threshold. Sliding window finds candidates in binary segmentation mask at different scales

(d) Pass candidates from input image to detector



(e) Category detection using HOG/linear SVM detectors for cirlces and triangles verify the presence of respective shape category.

Figure 4.7: Using the segmentation mask to find candidate ROIs: The sliding window is passed over the binary image resulting from the red color enhancement and adaptive thresholding. The extracted subwindows contain a sufficient ratio of white pixels to their area. These are passed on to the HOG/linear SVM detectors for circles and triangles.

	HOG Grayscale			HOG RGB		
Mask	Recall	Precision	Time (ms)	Recall	Precision	Time (ms)
No mask	**75.82**	**90.16**	35.56	**84.72**	**89.89**	49.31
Tophat	38.45	92.13	32.89	41.67	88.95	45.78
Bottomhat	42.05	91.48	32.75	45.20	88.40	45.52
Chromatic	42.11	78.29	48.07	38.38	92.68	55.24
Red	**84.66**	**94.77**	48.85	**90.21**	**90.90**	55.54

Table 4.5: Effect of segmentation masks on HOG/lin. SVM detection performance.

Name	Cell	Block	Stride	Bins	Oriented Gradients	Dimension
HOG 1	5x5	10x10	5x5	8	true	1568
HOG 2	5x5	10x10	5x5	8	false	1568
HOG 3	4x4	8x8	4x4	9	true	2916

Table 4.6: Parameters of the HOG 1-3 descriptors provided by GTSRB [Stallkamp 11].

low contrast to the background or within the sign due to poor illumination or deterioration of the sign. The chromatic filter omits distant and poorly illuminated traffic signs due to the lack of color richness of the corresponding pixels. The red color enhancement technique is less sensitive to changes in illumination, since it takes the relative dominance of one channel over the others into consideration.

We choose to combine the linear SVM/HOG detector with the red color enhancement segmentation technique, as it proved to be a good compromise between the resource efficiency and overall performance. The work on traffic sign recognition using segmentation masks and HOG-based SVMs presented in this section was published in [Zaklouta 11a, Zaklouta 11b].

4.2.3 Sign classification

The content of the candidate traffic signs detected is identified using the tree classifiers. For an extensive evaluation of the tree classifiers, we use the German Traffic Sign Recognition Benchmark [Stallkamp 11], which contains a large number of classes and images. We replace and evaluate the Euclidean similarity measure in the K-d tree by the correlation and χ^2 distance of the HOG descriptors. Further, we introduce the spatial weighting, which is particularly adapted for the traffic sign classification. This improvement weights the feature vectors to focus the Nearest Neighbor comparison in the tree classifers on the interior of the sign.

4.2.3.1 German TSR Benchmark

We further evaluate the performance of combining the HOG features with the K-d trees and Random Forests using the German Traffic Sign Recognition Benchmark [Stallkamp 11]. The work presented in this section was published in [Zaklouta 11c].

This image data set contains 43 classes, 26640 train images and 12569 test images. The parameters of the three HOG features used are listed in Table 4.6. All the images used are resized to 40x40 pixels using a bilinear interpolation. The precalculated HOG

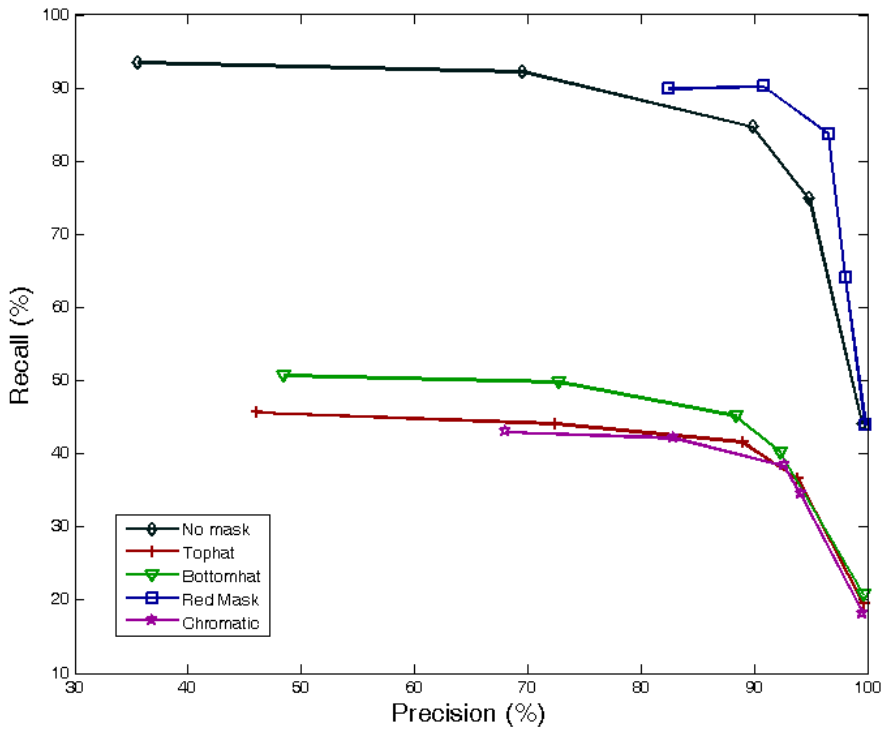


Figure 4.8: Effect of segmentation on RGB HOG detection results.

1, 2, and 3 descriptors for the train and test images are available online[2].

Overview of Approaches at Live Competition

Several teams participated in the GTSRB online and live competitions. The best performance of 99.15% in the online competition and 99.46% at the live competition were achieved using a committee of Neural Networks by Ciresan et al. [Ciresan 11]. They combine Multi-layer Perceptron (MLP) with deep Convolutional Neural Networks (CNN) of alternating convolutional, maximum pooling and fully connected hidden layers to ensure the robust and accurate classification of the input traffic signs. The human classifier correctly classified 98.81% and 98.84% in the online and live competitions respectively. Sermanet et al. [Sermanet 11] achieved 99.17% and 98.31% for these same data sets using Multi-Scale Convolutional Networks. They experiment with various feature subsets, network parameters and structures to optimize the results. Boi et al. [Boi 11] use a hierarchichal ensemble of SVM classifiers trained on shape, color and gradient-based features to achieve a classification accuracy of 96.89%. Rajesh et al. [Rajesh 11] combine the Coherence Vector of Oriented Gradients (CVOG), Color Coherence Vector (CCV) and the HOG features with Neural Network classifiers to achieve a best performance of 94.73%. Our efficient approach using Random Forests yield 97.2% on the online competition dataset and came in 3rd at the live competition with a classification rate of 96.14%.

The memory and processing contraints in embedded systems render the large Neural Networks and extensive preprocessing and feature extraction techniques inapplicable inspite their high recognition rates. Our approach proved to be the most suitable for

[2]http://benchmark.ini.rub.de/

E_{max}	**HOG 1**	**HOG 2**	**HOG 3**	**Avg ms**
500	71,30%	68,84 %	71,08%	1.59
1000	73,03%	70,40%	72,75%	3.29
1500	73,76%	71,25 %	73,65%	4.91
2000	73,94%	71,87 %	74,00%	6.55
2500	74,37%	72,03%	74,53%	8.19
5000	**74,92%**	**73,39%**	**75,03%**	16.36

Table 4.7: Classification results when varying E_{max} in K-d tree. $k_{NN} = 5$

embedded systems, as it runs in real-time and is resource efficient. It is described in detail and compared to other classifiers in the following.

4.2.3.2 Comparison of Classifiers

In this section, we compare the tree classifiers, K-d tree and Random Forests, to the state-of-the-art Support Vector Machine (SVM) classifier. We also evaluate the new approaches proposed: the spatial weighting and the correlation and χ^2 as similarity measures.

A **K-d tree**

The construction and the parameters of the K-d tree are described in Chapter 2.2.4. Five nearest neighbors are retrieved for each test candidate, i.e. $k_{NN} = 5$. This value was determined empirically. Unless otherwise specified the similarity measure used in the K-d tree is the Euclidean distance.

I **Influence of the E_{max} parameter**

The performance of the K-d tree also depends on the value of the E_{max} parameter, which determines the number of neighbors examined during the search. Its effect on the classification results is shown in Table 4.7. Increasing the E_{max} parameter by a factor of 10 increases the classification rate by up to 4.55% for the HOG 2 descriptor. The average time needed to classify a sample, however, also increases significantly by a factor 10 from 1.59 ms to 16.36 ms. The E_{max} parameter in the K-d tree is set to 5000 in further experiments, as it yields the best results.

The HOG 2 descriptor has a poorer overall performance than HOG 1 and HOG 3. The former computes signed gradient orientations i.e. 0° to 360°, while the two latter use unsigned gradients i.e. 0° to 180°. When using the same number of bins, the binning is coarser in the HOG 2 descriptor i.e. the bins are larger (45° per bin) than in the HOG 1 (22.5° per bin) and HOG 3 (20° per bin) descriptors. A finer spatial binning better describes the characteristics of each traffic sign class.

II **Correlation as a Similarity Measure**

The Euclidean distance measure in the K-d tree is replaced by the correlation, which is described in Chapter 3.2.1. The results obtained when checking five Nearest Neighbors ($k_{NN} = 5$) and examining 5000 nodes at most ($E_{max} = 5000$) are shown in Table 4.8. The accuracies are improved by up to 2.44%. This improvement shows that the correlation is a more robust measure for this application, as it takes into account the trend and not the difference between the HOG descriptors.

Weighting	HOG 1	HOG 2	HOG 3
Euclidean	74.92%	73.39%	75.03%
Correlation	75.20%	73.78%	77.47%
Chi Squared χ^2	**76.83%**	**76.01%**	**77.87%**

Table 4.8: Improvement of the K-d tree classification results when using the correlation or χ^2 distance instead of default Euclidean similarity measure on the HOG descriptors. $E_{max} = 5000$, $k_{NN} = 5$

Figure 4.9: Tight bounding box around the traffic sign used for computing the HOG 4 feature

III **Chi Squared χ^2 as a Similarity Measure**
The Euclidean distance measure in the K-d tree is replaced by the χ^2 distance, which is described in Chapter 3.2.1. The classification results obtained using this measure are shown in Table 4.8. There is an improvment of up to 2.84%. This shows that the χ^2 distance measure is a more suitable similarity measure for the histograms, as it relativizes the difference between the bins to their average size. This also makes it more robust to noise. The Euclidean distance measure, on the other hand, does not take into account the fact that the feature differences are not the same in all the dimensions of the feature space.

IV **Spatial Weighting**
In the K-d tree the Nearest Neighbor search extracts the signs which are the most similar to the test sample. The difference between the signs usually lies in the interior region containing the pictogram or speed limit.

To evaluate the effect of the background on the classification performance, the HOG 4 descriptor was calculated on a tight bounding box of the images as shown in Figure 4.9, using the same parameters as the HOG 3. It yields an accuracy of 92.7%, i.e. 17% higher than HOG 3, using the Euclidean distance, $k_{NN} = 5$ and $E_{max} = 5000$. This shows that the effect of the small strip of background around the traffic sign has a significant effect on the performance of the K-d tree classifier.

We propose a new spatial weighting of the features to increase the influence of the central part of the image rather than the borders. When computing the Euclidean distance between a training sample in the tree and the HOG descriptor of the test image, the difference between the individual blocks is multplied by a factor $f \in]0, 1[$, depending on the position of the block. The interior blocks have a larger factor than the ones along the border. Table 4.9 depicts the weighting scheme for 5x5 non-overlapping blocks.

The 30, 50 and 80 speed limit signs are most frequently confused by the K-d tree. The spatial weighting of the HOG descriptor values according to the location of

$$\begin{matrix} 1 & 2 & 3 & 2 & 1 \\ 2 & 4 & 6 & 4 & 2 \\ 3 & 6 & 9 & 6 & 3 \\ 2 & 4 & 6 & 4 & 2 \\ 1 & 2 & 3 & 2 & 1 \end{matrix} \quad \mathbf{X} \ \frac{1}{\sum}$$

Table 4.9: Spatial weighting for 5x5 non-overlapping blocks.

Weighting	HOG 1	HOG 2	HOG 3
No Weighting	74.92%	73.39%	75.03%
With Spatial Weighting	88.53%	88.73%	90.39%

Table 4.10: Improvement of the K-d tree classification results when applying the spatial weighting to the HOG descriptors. $E_{max} = 5000$, $k_{NN} = 5$

the block improves the results of the approximate Nearest Neigbors search in the K-d tree significantly. The prioritizing of the interior helps better distinguish the fine difference between the contents of the traffic signs. As shown in Table 4.10, the overall classification hit rates were increased by about 15% with an $E_{max} = 5000$.

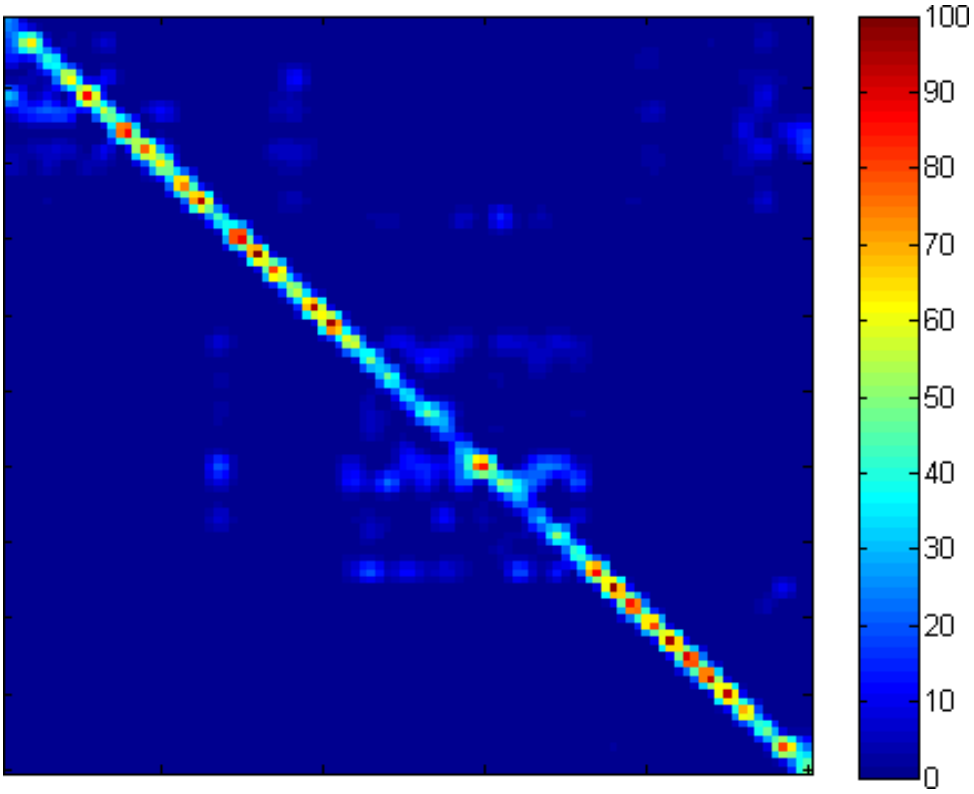
Table 4.11 shows the confusion matrix of some of the largest improvements induced by the spatial weighting. The HOG 3 descriptor is used, as it achieved the best results. The classification accuracy for the $20\ km/h$ speed limit, for example, increased by 25 samples, of which 10 were previously confused with the $70\ km/h$ speed limit. Similarly, 73 less *Right of Way* signs are confused with the *Construction Site* sign when using the spatial weighting, increasing the classification accuracy of this sign by a total of 117 samples.

Figure 4.10 illustrates the effect of the spatial weighting on all 43 classes using the HOG 3 descriptor. Note that the rate of correct classifications along the diagonal is improved. The misclassifications between similar classes, depicted as light blue patches around the diagonal, are reduced significantly.

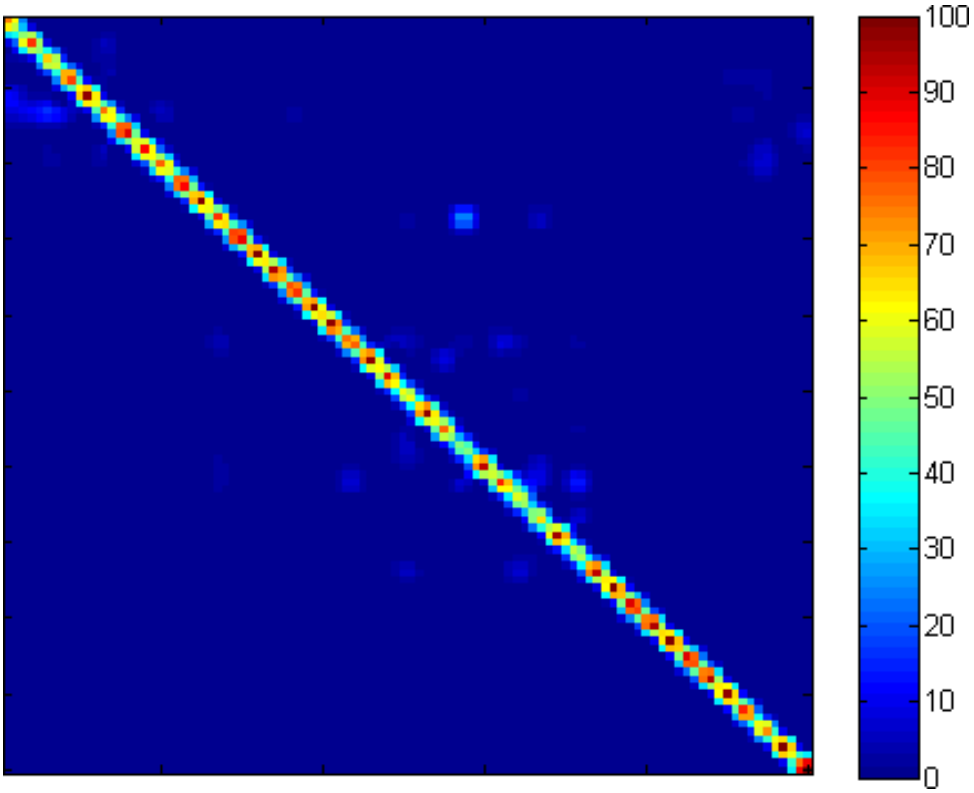
B **Random Forests**

To evaluate the performance of the Random Forest classifier, we vary its parameters: the number of trees in the forest, the number of features to be chosen at random and the size of the training sample subset. The structure of the Random Forest and its parameters are explained in Chapter 2.2.4. The results of changing the parameters are shown in Table 4.12. The HOG 2 descriptor is used as it yields the highest accuracy of 97.2%, compared to 95.1% and 95.2% for the HOG 1 and 3 descriptors respectively. The performance of the Random Forest does not vary significantly (0 to 2%) when the parameters are changed, which makes it more generic and easier to use. The number of trees is set to 500, the number of features and samples to 100, as these parameters yield the best results for these features.

The Random Forest achieves up to 7% higher classification rates than the K-d tree with spatially weighted blocks. Since small subsets of 100 features and 100 training samples are used to construct the random trees, the probability of choosing the HOG



(a) Without applying the spatial weighting



(b) When applying the spatial weighting

Figure 4.10: Confusion matrices for GTSRB with and without spatial weighting in K-d tree classification. The HOG 3 descriptor is used. Note that the classification accuracy along the diagonal is improved when applying the spatial weighting.

Predicted Label \ True Label	20	30	50	60	70	80	100	120	
20	**+25**	0	0	0	0	0	0	0	
30	-5	**+132**	-29	-9	-9	-9	-3	0	⋮
50	-6	-30	**+214**	-9	2	9	0	0	
60	0	-25	-27	**+95**	-4	-48	-5	-1	
70	-10	-11	-57	-3	**+69**	-22	-2	-2	
80	0	-41	-33	-48	-29	**+122**	-21	-26	
100	-1	-21	-35	-24	-8	-37	**+41**	-37	
120	-3	-4	-20	-1	-21	-6	-9	**+68**	
									+117
									-73
	⋮		⋮		⋮		⋮		-24
									-2
									-30
									-5
Total	60	720	750	450	660	600	480	450	420

Table 4.11: Effect of Spatial Weighting on classification accuracy of speed limit and warning signs with K-d tree and HOG 3 descriptor.

descriptors of the 10% border region is small and the perturbation caused by the background is less significant. Hence, the randomness of the Random Forest classifier makes it more robust to variations than the K-d tree, which uses the entire descriptor set.

C **Multi-class SVM**
To extend our study of classifiers for TSR, the performance of the multi-class SVM classifiers is also evaluated. One linear SVM is trained per class to obtain 43 one-vs-all classifiers. The entire ensemble is queried and the highest confidence vote determines the class of a test sample. The Pegasos solver [Shalev-Shwartz 07] is used, as it optimizes the training process. Table 4.13 shows a comparison of the results obtained on the HOG 2 feature. The multi-class SVM achieves an accuracy of 95.04%. The Random Forests outperform the SVM classifiers by 1.6% to 2.1%.

4.2.3.3 Comparison of HOG and Distance Transforms

To further evaluate the effect of the features used on the performance of the K-d tree, we also use Distance Transforms (DTs). This is an efficient method to determine the similarity between two contour images. Various metrics can be used. In our experiments,

	Nb Samples	Nb Features	Nb Trees	Accuracy
Features	100	10	500	95.5%
	100	50	500	97.1%
	100	75	500	97.0%
	100	100	500	97.1%
Samples	10	100	500	97.1%
	100	100	500	**97.2%**
	500	100	500	95.2%
Trees	100	100	50	96.0%
	100	100	100	96.7%
	100	100	300	**97.2%**
	100	100	500	97.1%
	100	100	750	**97.2%**

Table 4.12: Classification results do not fluctuate when varying the parameters of the Random Forests using the HOG 2 descriptor.

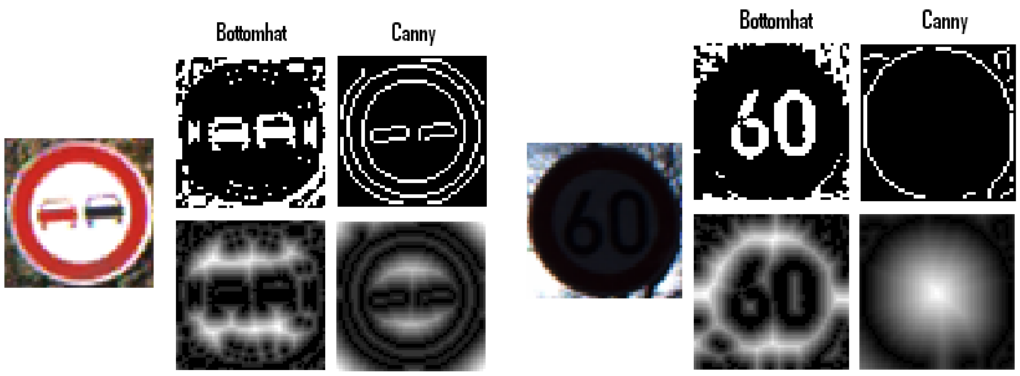
Classifier	Parameters	Accuracy
K-d tree	$E_{max} = 5000, k_{NN} = 5$	73.39%
K-d tree (Correlation)	$E_{max} = 5000, k_{NN} = 5$	73.78%
K-d tree (χ^2)	$E_{max} = 5000, k_{NN} = 5$	76.01%
K-d tree (Spatial Weighting)	$E_{max} = 5000, k_{NN} = 5$	88.73%
Random Forest	100 trees 100 var, 100 samples	96.70%
Random Forest	500 trees 100 var, 100 samples	**97.20%**
1-vs-all SVM (Pegasos)	C=10	95.04%

Table 4.13: Overview of K-d tree, Random Forest and SVM classifier accuracies using the HOG 2 descriptor of GTSRB.

the pixel value in the DT is the Euclidean distance of this pixel to the nearest nonzero pixel in the binary image.

Similarly to the HOG descriptor computation, the images within the bounding box are used. These are resized to 50x50 pixels. We test both the entire resized image and an interior patch of 41x41 pixels, further eliminating the background and capturing only the pictogram or characters inside the traffic sign.

We use the bottom-hat transform with a filter size of 2x2 pixels and an adaptive threshold to obtain the binary image as in Section 4.2.1. The result is a segmentation of the dark regions that are surrounded by light pixels: for example pictograms and characters. Figure 4.11 juxtaposes the result of the bottomhat thresholding and the Canny edge detection with their corresponding DT. Note that the bottomhat thresholding preserves the details of the pictograms. Since the Canny edge detector uses the first derivative of a Gaussian, it considers the weak gradients around the characters in a poorly illuminated



(a) Top: Edge image, Bottom: DT (b) Top: Edge image, Bottom: DT

Figure 4.11: Comparison of Distance Transforms (DT) using Bottomhat thresholding and Canny edge detection.

Feature	Bottom hat	Canny	HOG 2
Entire Image (50x50 pixels)	81.2%	77.8 %	97.20%
Interior (41x41 pixles)	81.8%	77.4%	

Table 4.14: Random Forest classification results using Distance Transform (DT) and HOG.

image as noise. On the other hand, the bottomhat operator considers local gradients and is less sensitive to global variations in illumination.

The results obtained using the Random Forest classifiers and the Distance Transforms are shown in Table 4.14. The number of random trees in the forest is set to 500 and 100 variables and samples were chosen at random for their construction. Removing the background information eliminates irrelevant information and increases the classification rate by about 4%. The bottomhat thresholding approach achieves higher classification rates than that using the Canny edge detection. However, the DT classification results are lower than those obtained using HOG descriptors.

4.3 Overall Performance of the TSR System

The overall performance on the traffic sign recognition data set presented in Section 4.2.2.1 is shown in Table 4.15. The red color segmentation and the 144 value HOG/linear SVM detector presented in Section 4.2.2.4 were used, because they achieved the best results. The parameters of the K-d tree were set to $k_{NN} = 5$ and $E_{max} = 1000$, as they were shown to be a good compromise of processing time, memory requirements and accuracy rate. The Random Forest used contains 100 trees. The parameters of the 1568 value HOG 1 descriptor presented in Section 4.2.3 were used to compute the HOG training and testing features in the classification phase.

As mentioned earlier, the combination of the red color enhancement segmentation and HOG/linear SVM detector achieves a detection rate of 90.21%. The traffic signs found are then recognized by the tree classifiers. The K-d tree with the Euclidean distance

similarity measure yields a classification rate of 59.21%. The correlation and χ^2 metrics improve the accuracy slightly to achieve 61.02% and 66.13% respectively. However, the spatial weighting of the HOG features in the K-d tree improves the classification accuracy by over 20%, attaining the best overall rate of 80.90%. This outperforms the Random Forest by about 10%.

The Random Forests achieve a poorer performance on this data set due to its small size. The random selection of samples when constructing the random trees favors the highly represented classes. The K-d tree classifiers are able to overcome this problem, as they use the entire data set. When considering the results at hand, one can conclude that the choice of the classifier strongly depends on the training data set. The Random Forests outperformed the K-d trees on the German Traffic Sign Recognition Benchmark due to the sufficient amount of 26640 training samples. The K-d tree yields the better performance on the smaller data set containing only 14763 training images.

4.4 Feature Space Reduction

The feature space reduction techniques are designed to minimize the memory and processing requirements as well as yield higher accuracy rates by eliminating less important features. The minimization of resource requirements is particularly important in embedded systems, which are often subject to physical and financial constraints. The feature selection also helps understand the generated features. In the following experiments, the German Traffic Sign Recognition Benchmark and the corresponding HOG 2 feature are used for the evaluation of the Random Forests and Fisher's Criterion for feature space reduction.

4.4.1 Feature Selection

Two selection techniques are implemented for the feature space reduction: Random Forests and Fisher's Criterion. We use the HOG 2 descriptor in the experiments as it yields the best results.

As described in Chapter 3.1.1, the Random Forest can be used to evaluate the feature importance based on the variance of the classification errors on the permuted out-of-bag data. The variable importance of the HOG 2 feature values obtained using a Random Forest of 100 trees, 100 variables and 100 samples is shown in Figure 4.12. Each row or column of blocks of this feature is described by 224 values (7 blocks x 4 cells x 8 bins). In Figure 4.12, the peaks in the variable importance recur every 224 values, coinciding with the interior region of the image. Therefore, the central blocks have a higher variable importance than the marginal ones. This affirms the performance improvement when using the spatial weighting, as the interior of the traffic sign image, containing the pictogram or the speed limit, is more important for the classification than the border regions.

Fisher's Criterion ranks the features according to their ratio of inter-class to intra-class variance. Refer to Chapter 3.1.1.2 for more details. The Fisher Scores of the HOG 2 feature vector are shown in Figure 4.13. A similar trend to the Random Forests ranking can be observed, where the interior blocks yield a higher score than those on the border.

Sign	Train	Test	**Detected**	**Classified**				
				K-d tree (Euclidean)	K-d tree (spatial weighting)	K-d tree (Correlation)	K-d tree (χ^2)	Random Forests
	21	2	2	0	1	0	0	0
	1474	149	145	144	140	144	144	139
	30	13	3	0	1	0	0	0
	30	0	0	0	0	0	0	0
	338	61	33	1	16	1	1	19
	68	25	16	0	0	0	0	0
	15	4	4	0	0	0	0	0
	968	179	173	143	172	142	157	168
	78	1	0	0	0	0	0	0
	31	1	0	0	0	0	0	0
	131	40	2	0	1	0	0	0
	113	20	18	2	15	2	2	13
	23	6	6	0	0	0	0	0
	651	111	83	37	59	37	36	47
	258	73	68	11	27	9	8	12
	1110	207	201	112	181	123	141	163
	1499	51	50	44	44	44	44	39
	316	46	43	12	25	12	13	14
	1900	171	167	64	131	74	97	126
	3852	276	269	259	267	258	268	268
	1726	126	124	23	76	26	34	3
	18	0	0	0	0	0	0	0
	66	22	22	0	0	0	0	0
	20	0	0	0	0	0	0	0
Average			**90.21%**	59.62%	**80.90%**	61.02%	66.13%	70.75%

Table 4.15: Overall performance of the TSR system using Red Color Enhancement, HOG/linear SVM detector and tree classifiers on our data set.

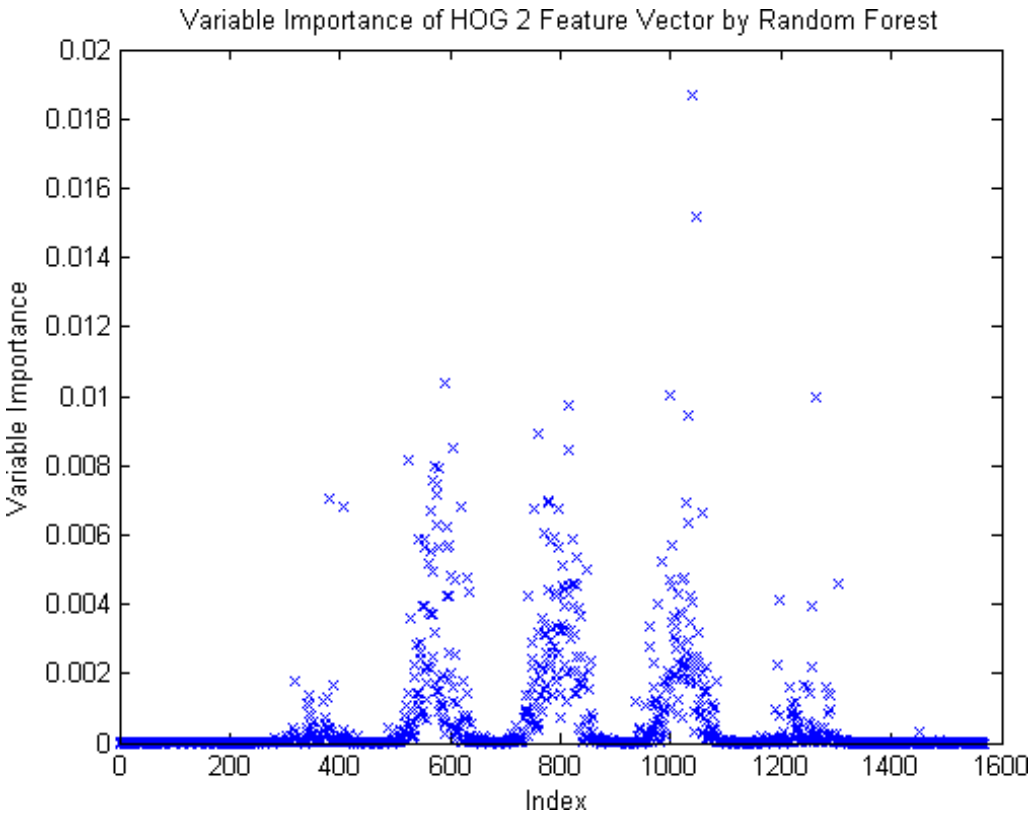


Figure 4.12: Variable Importance of HOG 2 feature using Random Forests. The peaks in the variable importance recur every 224 values (One row or column of blocks is composed of 7 blocks x 4 cells x 8 bins), coinciding with the interior region of the image.

4.4.2 Evaluation

The n most important features are selected from the ranking obtained from a Random Forest with 100 trees or Fisher's Criterion. To evaluate the feature space reduction, we combine the feature selection techniques with the Random Forest and SVM classifiers. A Random Forest with 100 trees or an SVM is then trained using this subset.

Figure 4.14 illustrates the effect of the number of selected features n on the classification accuracy. Note that the Random Forests outperform the SVMs. The former yield higher classification accuracies on subsets with more than 500 features, due to the random variable selection in the random trees. The accuracy of the SVMs, however, is improved when using only 200-500 feature values.

The overall performance is compared in Table 4.16. Note that similarly high accuracy rates are obtained by the Random Forest and SVM when using only about one third of the features.

4.5 Conclusions and Perspectives

A real-time Traffic Sign Recognition system was presented in this chapter. The first step of the three stage approach is the image segmentation to reduce the search space. We improve the color enhancement approach proposed by Ruta et al. [Ruta 10] by in-

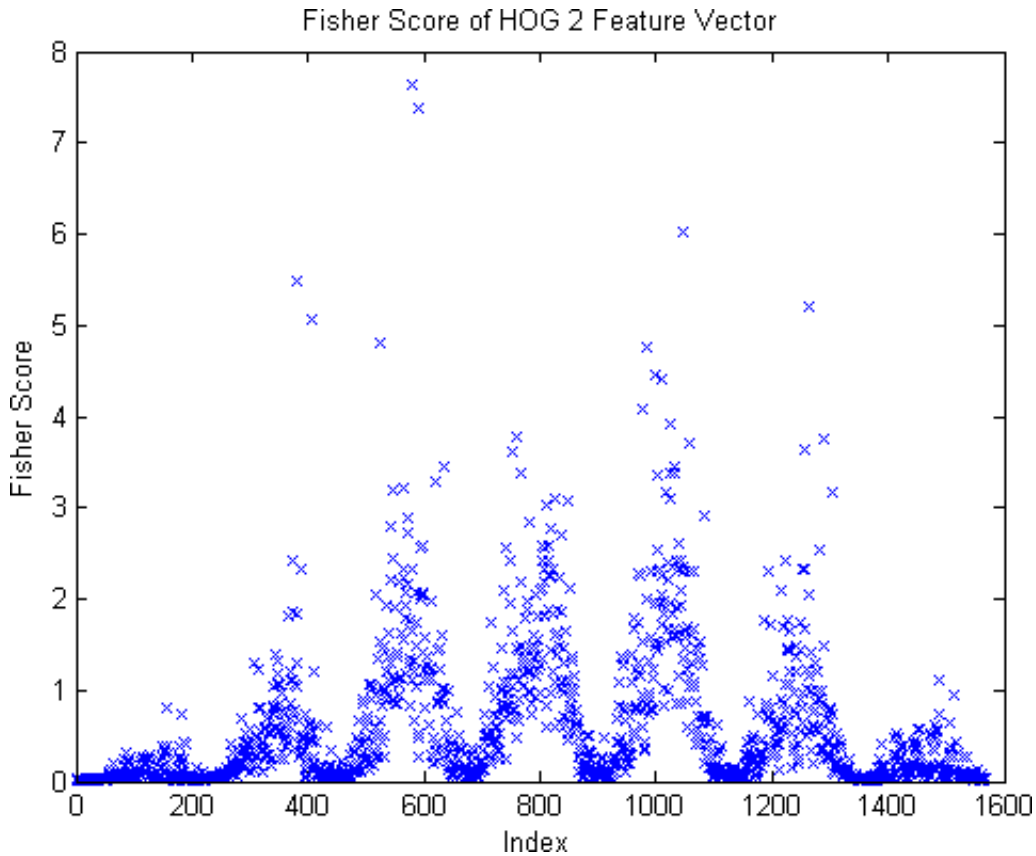


Figure 4.13: Feature ranking using Fisher Scores on HOG 2 feature. The peaks in the variable importance recur every 224 values (One row or column of blocks is composed of 7 blocks x 4 cells x 8 bins), coinciding with the interior region of the image.

troducing an adaptive threshold. In the second stage, the circular and triangular signs are detected using the efficient HOG/linear SVM detector. The combination of these two phases achieves recall and precision rates of over 90% at a processing rate of 18 to 28 frames per second.

The candidates found by the detector are identified using multi-class classifiers. We compare the performance of the K-d trees, the Random Forests and the one-vs-all SVM classifiers. We improve the K-d tree accuracy by up to 15% when applying a spatial weighting to focus the Euclidean similarity measure on the interior of the traffic sign. This technqiue outperforms the K-d tree with the Euclidean, correlation and χ^2 distance metrics on our traffic sign recognition data set with a classification rate of about 81%. However, the Random Forest outperforms the K-d tree and SVM on the larger German Traffic Sign Recognition Benchmark, yielding a classification accuracy of 97%. One can conclude that the choice of the suitable classifier depends on the cardinality of the training data set.

Moreover, we employ the Random Forests and Fisher's Criterion feature selection techniques. The benefit of this is two-fold. On the one hand, the feature space dimension is reduced, minimizing the memory and processing requirements. On the other, the classification accuracy of the SVM is improved when using a well-chosen subset of features.

Feature Selection	# Features	Random Forests	linear SVM (Pegasos)
None	1568	96.69%	95.00%
Random Forest	500	96.63%	96.48%
Fisher Criterion	500	96.63%	96.41%
Random Forest	1000	96.70%	95.94%
Fisher Criterion	1000	96.85%	95.67%

Table 4.16: Feature Space Reduction using Fisher's Criterion and Random Forests. Fisher's Criterion and Random Forest (100 trees, 100 variables and 100 samples) used for feature ranking and classification. Classifiers: Random Forest and linear SVM with C=10.

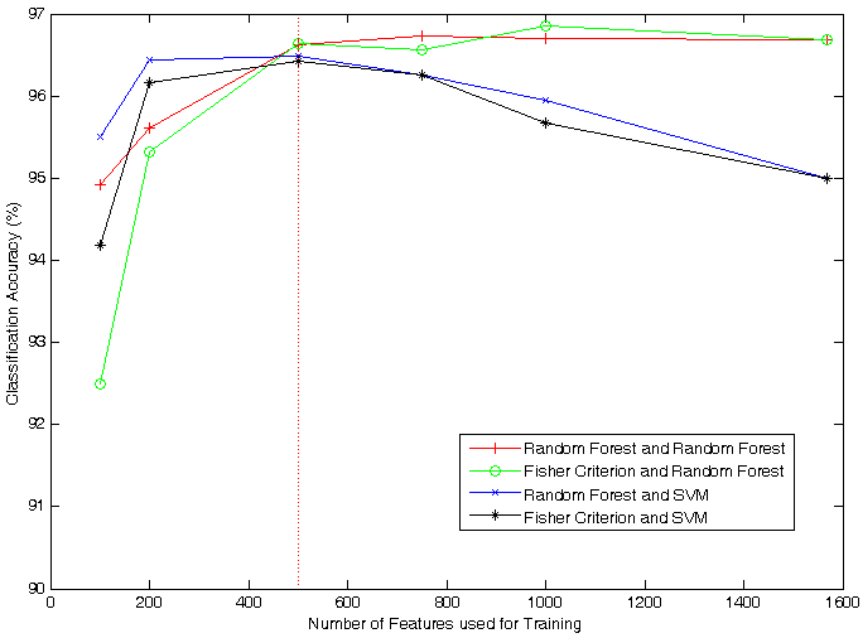


Figure 4.14: Effect of feature selection using Fisher Criterion and Random Forests on Random Forest and SVM classification accuracy. Subsets of n = 100, 200, 500, 750 and 1000 of the 1568 values of HOG 2 descriptors selected using Random Forests or Fisher Criterion.

Future work could integrate the temporal information to track the detected traffic signs and reinforce the decision making process. This would further accelerate the candidate detection by restricting the search space. The feature selection can be employed to accelerate the detection phase by reducing the size of the descriptor vectors. Further, this could be combined with other classifiers, such as the Neural Networks. Moreover, the adaptive threshold can be combined with other color enhancements to detect blue traffic signs or green traffic lights, for example.

Chapitre 5

Conclusion

Ce chapitre conclut le manuscrit avec un résumé des solutions présentées. Nous avons abordé la détection de piétons et la reconnaissance de panneaux pour les applications d'aide à la conduite et la vidéo-surveillance.

Nous avons examiné tout d'abord trois aspects du processus de classification: les caractéristiques, les métriques de comparaison et la combinaison de différents classifieurs. La sélection des caractéristiques réduit leur dimension ainsi que le besoin en temps de calcul et mémoire, tout en gardant un taux de classification élevé.

Dans le cadre de la détection de piétons, la plupart des fausses alarmes sont des objets fixes, comme des lampadaires. Celles-ci sont éliminées grâce à la corrélation sur plusieurs trames en utilisant des caméras statiques et par un filtre complémentaire en forme d'arbre dans des systèmes embarqués.

Pour la reconnaissance de panneaux, nous proposons une nouvelle approche composée de trois étapes: une segmentation de couleur, une détection de forme et une classification. La segmentation a pour but de réduire l'espace de recherche et d'éliminer une partie des fausses alarmes. Nous adaptons la segmentation proposée par [Ruta 10] qui cherche les pixels avec un canal rouge dominant par rapport aux autres. Nous améliorons cette approche en introduisant des seuils adaptatifs qui prennent en compte l'illumination de l'image entière.

La détection est réalisée à l'aide d'un classifieur à vaste marge (en anglais: *Support Vector Machines (SVM)*) travaillant sur des histogrammes de gradients orientés (en anglais: *Histogram of Oriented Gradients (HOG)*). Cette méthode, combinée avec la segmentation, atteint un taux de rappel et de précision de 90% sur nos séquences de film.

La classification se fait quant à elle grâce à des algorithmes à base d'arbres: *K-d tree* et *Random Forest*. La performance du premier est améliorée jusqu'à 20% lors de l'implémentation de la pondération spatiale des vecteurs de caractéristiques, et une précision de 80% est atteinte sur nos séquences. Le *Random Forest* atteint un taux de classification de 97% sur le Benchmark Allemand de la Reconnaissance de Panneaux (*German Traffic Sign Recognition Benchmark*).

Nous concluons cette thèse en proposant des extensions possibles, comme l'intégration de l'information temporelle et spatiale pour suivre les panneaux détectés et réduire les fausses alarmes. De plus, l'approche de la reconnaissance de panneaux peut être étendue à d'autres types de panneaux, comme ceux d'information.

En enrichisant la base d'apprentissage de plus nombreux exemples, il serait possible d'améliorer le résultat. Il serait également intéressant d'examiner d'autres classifieurs comme Adaboost ou des SVMs avec différents noyaux.

L'effet de la réduction de la dimensionalité des vecteurs caractéristiques pourrait quant à lui être étudié sur d'autres types de caractéristiques et avec d'autres classifieurs, comme des réseaux de neurones ou Adaboost. Enfin, son effet sur la vitesse et la performance de la détection de panneaux pourrait être évalué.

Chapter 5

Conclusions and Perspectives

Contents

5.1 Summary **85**

5.1.1 Three Aspects of Classification 85

5.1.2 Real-Time Traffic Sign Recognition System 86

5.2 Future Work **86**

5.1 Summary

The main objectives of this thesis were to develop a robust, real-time Traffic Sign Recognition System (TSR) and to enhance the existing pedestrian detection algorithms for Advanced Driver Assistance Systems (ADAS) and video surveillance applications. This chapter provides a summary of the contributions made in this thesis as well as an outlook for further research.

5.1.1 Three Aspects of Classification

In Chapter 3, we examine three aspects of the classification process: the features describing the object, the metrics used to compare them and the combination of the classifiers which distinguish between the object categories.

The importance of the features for the distinction between the classes can be evaluated using the Random Forest and Fisher's Criterion. Our experiments show that a well chosen subset of the most significant features is sufficient to attain an equally high classification rate as when using the entire feature set. This helps reduce the memory and processing requirements.

Moreover, the importance of the metric used in the K-d tree classifiers is evaluated. We establish that the correlation and the χ^2 distance measures are more suitable for the comparison of the Histogram of Oriented Gradients (HOG) and Pyramid HOG (PHOG) features. We use the correlation of the HOG descriptors to eliminate immobile false alarms, such as trees and poles, in a static camera video surveillance application. The precision rate was doubled using this technique, while maintaining the high recall rate.

A further pedestrian detection application for ADAS was developed by combining the

HOG/linear SVM detector with the K-d tree and Random Forest classifiers. This heterogenous mixture of experts decreased the number of false positives per frame to about half, while maintaining the recall rate.

5.1.2 Real-Time Traffic Sign Recognition System

We propose a three stage approach for TSR consisting of a segmentation, a category detection and a classification phase. We compare four different segmentation techniques: color enhancement, chromatic filtering as well as the tophat and bottomhat morphological filters. We propose to use the red color enhancement technique to reduce the search space of traffic signs. Moreover, we improve the segmentation by introducing an adaptive threshold. The result is combined with the triangular and circular shape detectors. These consist of linear Support Vector Machines (SVM) trained on the Histogram of Oriented Gradients (HOG) features. The combination of the red color enhancement segmentation and the shape detector attained a recall and precision rate of 90%, running at about 18 frames per second.

In the classification phase, we use the K-d tree and Random Forest classifiers. From our experiments, we can conclude that the choice of the classifier depends on the data set at hand. The K-d tree excels on the smaller data set, yet is outperformed by the Random Forest on larger data sets. The performance of the K-d tree was improved by up to 20% when introducing the spatial weighting of the feature vectors, which concentrates the comparison on the interior of the traffic sign. It attains an accuracy of 80% on our film sequences. The Random Forest outperforms the K-d tree on the German Traffic Sign Recognition Benchmark, achieving a classification rate of 97%.

The memory and processing requirements were reduced when applying the aforementioned feature reduction techniques. However, the high classification accuracies of the Random Forest and SVM classifiers were maintained. The accuracy of the latter was even augmented, when removing the redundant features.

5.2 Future Work

Future work could include the use of the temporal information to reinforce both the traffic sign as well as the pedestrian detection algorithms, by tracking the object found. This would also accelerate the process by restricting the search space.

Further improvements can be procured by enriching the data sets used in the K-d tree and Random Forest classifiers. This can be achieved using online learning, where the user contributes to updating the training data set.

The TSR system presented in this thesis can be extended to other sign categories. The adaptive threshold can be combined with various color enhancements to detect blue traffic signs or green traffic lights, for example.

The tree filters used to eliminate false alarms in the pedestrian detection applications can be combined with other features than those used in the detector, such as denser HOG descriptors or Distance Transforms, to better complement it. Other classifiers, such as Adaboost or SVMs with non-linear kernels could also be considered.

Moreover, the effect of the feature space reduction can be evaluated on other dense image descriptors such as the Pyramid Histograms of Visual Words (PHOW). Further, its

influence on other classifiers such as Neural Networks or Adaboost can also be evaluated. Finally, the effect of reducing the dimensionality of the feature vector on the speed and performance of the traffic sign and pedestrian detectors can also be examined.

Appendix A

Courses and Publications

A.1 Publications

- *Real-Time Traffic Sign Recognition in Three Stages*, Fatin Zaklouta, Bogdan Stanciulescu, Invited Paper for Special Issue of Robotics and Autonomous Systems Journal (affiliated with the Intelligent Autonomous Systems (IAS) Society), submitted October 2011

- *Real-time Traffic Sign Recognition using Tree Classifiers*, Fatin Zaklouta, Bogdan Stanciulescu, Special Issue on Machine Learning for Traffic Sign Recognition, IEEE Intelligent Transportation Systems Journal, submitted October 2011

- *Real-time traffic sign recognition using spatially weighted HOG trees*, Fatin Zaklouta, Bogdan Stanciulescu, IEEE International Conference on Advanced Robotics (ICAR) 2011, Best Student Paper Award

- *Traffic Sign Classification using K-d trees and Random Forests*, Fatin Zaklouta, Bogdan Stanciulescu, Omar Hamdoun, IEEE International Joint Conference on Neural Networks (IJCNN) 2011

- *Segmentation Masks for Real-time Traffic Sign Recognition using Weighted HOG-based Trees*,Fatin Zaklouta, Bogdan Stanciulescu, IEEE Intelligent Transportation Systems (ITS) 2011

- *Warning Traffic Sign Recognition using a HOG-based K-d Tree*, Fatin Zaklouta, Bogdan Stanciulescu, IEEE Intelligent Vehicles Symposium (IV) 2011

- *Classifying Bags of Keypoints using HMMs*, Fatin Zaklouta, Bogdan Stanciulescu, IEEE ACS/IEEE International Conference on Computer Systems and Applications (AICCSA) 2010

- *Object Classification Using Bags of Local Features*, Fatin Zaklouta, Bogdan Stanciulescu, Cognitive Systems with Interactive Sensors (COGIS) 2009

- *Performance of Haar-like Features and Control Points on Pedestrian Detection*, Fatin Zaklouta, Bogdan Stanciulescu, Amaury Breheret, Cognitive Systems with Interactive Sensors (COGIS) 2009

A.2 Courses

Attended

- *Morphologie Mathématique*, Centre de Morphologie de Mines ParisTech, Fontainebleau, Oct. 2008
- *Reconnaissance d'Objets et Vision Artificielle*, by Schmid and Ponce, École Normale Supérieur, Nov.-Dec. 2009
- *Colloquium STAtistiques pour le Traitement de l'IMage (STATIM2010)*, Université d'Evry, 11 et 12 mars 2010
- *Computer Vision and Machine Learning (CVML) Summer School 2010*, INRIA Grenoble, July 2010

Given

- *Reconnaissance des Formes*, EFREI, 2010/2011

Bibliography

[Abramson 05] Y. Abramson & B. Steux. *YEF* Real-Time Object Detection*. Proc. Intl Workshop Automatic Learning and Real-Time, 2005.

[Alefs 07] B. Alefs, G. Eschemann, H. Ramoser & C. Beleznai. *Road sign detection from edge orientation histograms*. In 2007 IEEE Intelligent Vehicles Symposium, pages 993–998, June 2007.

[Allwein 01] E.L. Allwein, R.E. Schapire & Y. Singer. *Reducing multiclass to binary: a unifying approach for margin classifiers*. The Journal of Machine Learning Research, vol. 1, pages 113–141, 2001.

[Bahlmann 05] C. Bahlmann, Y. Zhu, V. Ramesh, M. Pellkofer & T. Koehler. *A system for traffic sign detection, tracking, and recognition using color, shape, and motion information*. In Proceedings of the IEEE Symposium on Intelligent Vehicles, pages 255–260. Citeseer, 2005.

[Bargeton 08] A. Bargeton, F. Moutarde, F. Nashashibi & B. Bradai. *Improving pan-European speed-limit signs recognition with a new global number segmentation before digit recognition*. 2008 IEEE Intelligent Vehicles Symposium, 2008.

[Barnes 08] N. Barnes, A. Zelinsky & L.S. Fletcher. *Real-time speed sign detection using the radial symmetry detector*. Intelligent Transportation Systems, IEEE Transactions on, vol. 9, no. 2, pages 322–332, 2008.

[Baró 09] X. Baró, S. Escalera, J. Vitrià, O. Pujol & P. Radeva. *Traffic sign recognition using evolutionary adaboost detection and forest-ECOC classification*. Intelligent Transportation Systems, IEEE Transactions on, vol. 10, no. 1, pages 113–126, 2009.

[Bascon 07] S. Maldonado Bascon, S. Lafuente-Arroyo, P. Gil-Jimenez, H. Gomez-Moreno & F. Lopez-Ferreras. *Road-sign detection and recognition based on support vector machines*. IEEE transactions on intelligent transportation systems, vol. 8, no. 2, pages 264–278, 2007.

[Bascon 10] S. Maldonado Bascon, J. Acevedo Rodriguez, S. Lafuente Arroyo, A. Fernndez Caballero & F. Lopez-Ferreras. *An optimization on pictogram identification for the road-sign recognition task using SVMs*. Computer Vision and Image Understanding, vol. 114, no. 3, pages 373 – 383, 2010.

[Bay 08] H. Bay, A. Ess, T. Tuytelaars & L. Van Gool. *Speeded-up robust features (SURF)*. Computer Vision and Image Understanding, vol. 110, no. 3, pages 346–359, 2008.

[Beis 02] J.S. Beis & D.G. Lowe. *Shape indexing using approximate nearest-neighbour search in high-dimensional spaces*. In Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on, pages 1000–1006. IEEE, 2002.

[Bins 01] J. Bins & B.A. Draper. *Feature selection from huge feature sets*. In Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on, volume 2, pages 159–165. IEEE, 2001.

[Boi 11] Fabio Boi & Lorenzo Gagliardini. *A Support Vector Machines Network for Traffic Sign Recognition*. In International Joint Conference on Neural Networks, 2011.

[Bosch 06] A. Bosch, A. Zisserman & X. Munoz. *Scene classification via pLSA*. Computer Vision–ECCV 2006, pages 517–530, 2006.

[Bosch 07a] A. Bosch, A. Zisserman & X. Munoz. *Image classification using random forests and ferns*. In Proc. ICCV, 2007.

[Bosch 07b] A. Bosch, A. Zisserman & X. Munoz. *Representing shape with a spatial pyramid kernel*. In Proceedings of the 6th ACM international conference on Image and video retrieval, pages 401–408. ACM, 2007.

[Breiman 01] L. Breiman. *Random forests*. Machine learning, vol. 45, no. 1, pages 5–32, 2001.

[Burges 98] C.J.C. Burges. *A tutorial on support vector machines for pattern recognition*. Data mining and knowledge discovery, vol. 2, no. 2, pages 121–167, 1998.

[Chen 08] C.H. Chen, M. Chen & T. Gao. *Detection and Recognition of Alert Traffic Signs*. Publication of Standford University, 2008.

[Ciresan 11] Dan Ciresan, Ueli Meier, Jonathan Masci & Jurgen Schmidhuber. *A Committee of Neural Networks for Traffic Sign Classification*. In International Joint Conference on Neural Networks, 2011.

[Creusen 10] I.M. Creusen, R.G.J. Wijnhoven, E. Herbschleb & P.H.N. de With. *Color exploitation in hog-based traffic sign detection*. In Image Processing (ICIP), 2010 17th IEEE International Conference on, pages 2669 –2672, 2010.

[Curio 00] C. Curio, J. Edelbrunner, T. Kalinke, C. Tzomakas & W. Von Seelen. *Walking pedestrian recognition*. Intelligent Transportation Systems, IEEE Transactions on, vol. 1, no. 3, pages 155–163, 2000.

[Dalal 05] Navneet Dalal & Bill Triggs. *Histograms of Oriented Gradients for Human Detection*. In Cordelia Schmid, Stefano Soatto & Carlo Tomasi, editeurs, International Conference on Computer Vision

& Pattern Recognition, volume 2, pages 886–893, INRIA Rhône-Alpes, ZIRST-655, av. de l'Europe, Montbonnot-38334, June 2005.

[De La Escalera 97] A. De La Escalera, L.E. Moreno, M.A. Salichs & J.M. Armingol. *Road traffic sign detection and classification*. IEEE Transactions on Industrial Electronics, vol. 44, no. 6, pages 848–859, 1997.

[De La Escalera 03] A. De La Escalera, J.M. Armingol & M. Mata. *Traffic sign recognition and analysis for intelligent vehicles*. Image and vision computing, vol. 21, no. 3, pages 247–258, 2003.

[Dietterich 95] Thomas G. Dietterich & Ghulum Bakiri. *Solving multiclass learning problems via error-correcting output codes*. Journal of Artificial Intelligence Research, vol. 2, pages 263–286, 1995.

[Dietterich 00] T.G. Dietterich. *An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization*. Machine Learning, vol. 40, no. 2, pages 139–157, 2000.

[Dollar 09] P. Dollar, C. Wojek, B. Schiele & P. Perona. *Pedestrian Detection: A Benchmark*. 2009.

[Domeniconi 02] C. Domeniconi, J. Peng & D. Gunopulos. *Locally adaptive metric nearest-neighbor classification*. IEEE Transactions on Pattern Analysis and Machine Intelligence, pages 1281–1285, 2002.

[Duda 01] R.O. Duda, P.E. Hart, D.G. Stork*et al.* Pattern classification, volume 2nd Edition. Wiley New York, 2001.

[Eichhorn 04] J. Eichhorn & O. Chapelle. *Object categorization with SVM: kernels for local features*. MPIK, 2004.

[Enzweiler 09] M. Enzweiler & DM Gavrila. *Monocular Pedestrian Detection: Survey and Experiments*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, no. 12, 2009.

[Fang 03] C.Y. Fang, S.W. Chen & C.S. Fuh. *Road-sign detection and tracking*. IEEE transactions on vehicular technology, vol. 52, no. 5, pages 1329–1341, 2003.

[Fleyeh 06] H. Fleyeh. *Shadow And Highlight Invariant Colour Segmentation Algorithm For Traffic Signs*. In Cybernetics and Intelligent Systems, 2006 IEEE Conference on, pages 1 –7, june 2006.

[Fleyeh 08] H. Fleyeh & M. Dougherty. *Traffic sign classification using invariant features and Support Vector Machines*. In Intelligent Vehicles Symposium, 2008 IEEE, pages 530–535. IEEE, 2008.

[Franke 99] U. Franke, D. Gavrila, S. Görzig, F. Lindner, F. Paetzold & C. Wöhler. *Autonomous driving approaches downtown*. IEEE Intelligent Systems, vol. 13, no. 6, pages 1–14, 1999.

[Freund 96] Yoav Freund & Robert E. Schapire. *Experiments with a New Boosting Algorithm*. In In Proceedings of the Thirteenth International Conference on Machine Learning, pages 148–156, 1996.

[Freund 97] Y. Freund & R.E. Schapire. *A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting*. Journal of Computer and System Sciences, vol. 55, no. 1, pages 119–139, 1997.

[Gao 06] X.W. Gao, L. Podladchikova, D. Shaposhnikov, K. Hong & N. Shevtsova. *Recognition of traffic signs based on their colour and shape features extracted using human vision models*. Journal of Visual Communication and Image Representation, vol. 17, no. 4, pages 675–685, 2006.

[Garrido 05] M. Garcia Garrido, M. Sotelo & E. Martin-Gorostiza. *Fast road sign detection using hough transform for assisted driving of road vehicles*. Computer Aided Systems Theory–EUROCAST 2005, pages 543–548, 2005.

[Gavrila 99] DM Gavrila & V. Philomin. *Real-time object detection for smart vehicles*. In The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999, volume 1, 1999.

[Gavrila 00] D. Gavrila. *Pedestrian detection from a moving vehicle*. Computer Vision ECCV 2000, pages 37–49, 2000.

[Gavrila 07] D.M. Gavrila. *A bayesian, exemplar-based approach to hierarchical shape matching*. IEEE Transactions on Pattern Analysis and Machine Intelligence, pages 1408–1421, 2007.

[Gehler 09] P. Gehler & S. Nowozin. *On feature combination for multiclass object classification*. In Computer Vision, 2009 IEEE 12th International Conference on, pages 221–228. IEEE, 2009.

[Genton 02] M.G. Genton. *Classes of kernels for machine learning: a statistics perspective*. The Journal of Machine Learning Research, vol. 2, pages 299–312, 2002.

[Geurts 06] P. Geurts, D. Ernst & L. Wehenkel. *Extremely randomized trees*. Machine Learning, vol. 63, no. 1, pages 3–42, 2006.

[Grauman 07] Kristen Grauman, Trevor Darrell & Pietro Perona. *The pyramid match kernel: Efficient learning with sets of features*. Journal of Machine Learning Research, vol. 8, page 2007, 2007.

[Guruswami 99] V. Guruswami & A. Sahai. *Multiclass learning, boosting, and error-correcting codes*. pages 145–155, 1999.

[Guyon 03] I. Guyon & A. Elisseeff. *An introduction to variable and feature selection*. The Journal of Machine Learning Research, vol. 3, pages 1157–1182, 2003.

[Ho 95] T.K. Ho. *Random decision forests*. In icdar, page 278. Published by the IEEE Computer Society, 1995.

[Hoferlin 09] B. Hoferlin & K. Zimmermann. *Towards reliable traffic sign recognition*. In Intelligent Vehicles Symposium, 2009 IEEE, pages 324–329. IEEE, 2009.

[Jimenez 05] P. Gil Jimenez, S. Lafuente-Arroyo, H. Gomez-Moreno, F. Lopez-Ferreras & S. Maldonado-Bascon. *Traffic sign shape classification evaluation. Part II. FFT applied to the signature of blobs*. In Intelligent Vehicles Symposium, 2005. Proceedings. IEEE, pages 607–612. IEEE, 2005.

[Khoshgoftaar 07] T.M. Khoshgoftaar, M. Golawala & J. Van Hulse. *An Empirical Study of Learning from Imbalanced Data Using Random Forest*. In 19th IEEE International Conference on Tools with Artificial Intelligence, 2007. ICTAI 2007, volume 2, pages 310 –317, 2007.

[Kim 02] Hyun-Chul Kim, Shaoning Pang, Hong-Mo Je, Daijin Kim & Sung Yang Bang. *Pattern classification using support vector machine ensemble*. Pattern Recognition, 2002. Proceedings. 16th International Conference on, vol. 2, pages 160–163 vol.2, 2002.

[Kira 92] K. Kira & L.A. Rendell. *A practical approach to feature selection*. In Proceedings of the ninth international workshop on Machine learning, pages 249–256. Morgan Kaufmann Publishers Inc., 1992.

[Kohavi 97] Ron Kohavi & George H. John. *Wrappers for feature subset selection*. Artificial Intelligence, vol. 97, no. 1-2, pages 273 – 324, 1997. <ce:title>Relevance</ce:title>.

[Kononenko 94] I. Kononenko. *Estimating attributes: analysis and extensions of RELIEF*. In Machine Learning: ECML-94, pages 171–182. Springer, 1994.

[Kouzani 07] A.Z. Kouzani. *Road-sign identification using ensemble learning*. In Intelligent Vehicles Symposium, 2007 IEEE, pages 438–443. IEEE, 2007.

[Kuo 07] Wen-Jia Kuo & Chien-Chung Lin. *Two-Stage Road Sign Detection and Recognition*. In Multimedia and Expo, 2007 IEEE International Conference on, pages 1427 –1430, 2007.

[Lazebnik 06] S. Lazebnik, C. Schmid & J. Ponce. *Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories*. 2006.

[Le 10] T. Le, S. Tran, S. Mita & T. Nguyen. *Real Time Traffic Sign Detection Using Color and Shape-Based Features*. Intelligent Information and Database Systems, pages 268–278, 2010.

[Lee 04] Y. Lee, Y. Lin & G. Wahba. *Multicategory support vector machines*. Journal of the American Statistical Association, vol. 99, no. 465, pages 67–81, 2004.

[Lepetit 06] V. Lepetit & P. Fua. *Keypoint recognition using randomized trees*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 9, pages 1465–1479, 2006.

[Li 06] Ling Li. *Multiclass boosting with repartitioning*. pages 569–576, 2006.

[Li 10] Z. Li, Z. Wei, B. Yin, X. Ji & R. Shan. *Pedestrian Detection Based on a New Two-Step Framework*. In Education Technology and Computer Science (ETCS), 2010 Second International Workshop on, volume 3, pages 56–59. IEEE, 2010.

[Lim Jr 10] K.H. Lim Jr, K.P. Seng Jr & L.M. Ang Jr. *Intra color-shape classification for traffic sign recognition*. ICS, pages 642–647, 2010.

[Lin 06] Y.S. Lin & C.N. Hsu. *Boosting multiclass learning with repeating codes*. pages 591–598, 2006.

[Liu 05] H. Liu & L. Yu. *Toward integrating feature selection algorithms for classification and clustering*. Knowledge and Data Engineering, IEEE Transactions on, vol. 17, no. 4, pages 491–502, 2005.

[Lowe 04] D.G. Lowe. *Distinctive image features from scale-invariant keypoints*. International Journal of Computer Vision, vol. 60, no. 2, pages 91–110, 2004.

[Mikolajczyk 01] K. Mikolajczyk & C. Schmid. *Indexing based on scale invariant interest points*. vol. 1, pages 525–531, 2001.

[Mikolajczyk 02] K. Mikolajczyk & C. Schmid. *An affine invariant interest point detector*. Computer Vision ECCV 2002, pages 128–142, 2002.

[Mikolajczyk 04] K. Mikolajczyk, C. Schmid & A. Zisserman. *Human detection based on a probabilistic assembly of robust part detectors*. Computer Vision-ECCV 2004, pages 69–82, 2004.

[Mikolajczyk 05] K. Mikolajczyk & C. Schmid. *A performance evaluation of local descriptors*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 10, pages 1615–1630, 2005.

[Miura 00] J. Miura, T. Kanda & Y. Shirai. *An active vision system for real-time traffic sign recognition*. Proc. IEEE Inteligent transportation systems, pages 52–57, 2000.

[Moosmann 07] F. Moosmann, B. Triggs & F. Jurie. *Fast discriminative visual codebooks using randomized clustering forests*. Advances in neural information processing systems, vol. 19, page 985, 2007.

[Moutarde 07] F. Moutarde, A. Bargeton, A. Herbin & L. Chanussot. *Robust on-vehicle real-time visual detection of American and European speed limit signs, with a modular Traffic Signs Recognition system*. In 2007 IEEE Intelligent Vehicles Symposium, pages 1122–1126, 2007.

[Moutarde 08] F. Moutarde, B. Stanciulescu & A. Breheret. *Real-time visual detection of vehicles and pedestrians with new efficient adaBoost features*. Workshop on Planning, Perception, and Navigation for Intelligent Vehicles, IROS 2008, June 2008.

[Munder 06] S. Munder & DM Gavrila. *An Experimental Study on Pedestrian Classification*. IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, pages 1863–1868, 2006.

[Mutch 06] J. Mutch & DG Lowe. *Multiclass object recognition with sparse, localized features*. vol. 1, 2006.

[Nguwi 08] Yok-Yen Nguwi & Abbas Kouzani. *Detection and classification of road signs in natural environments*. Neural Computing & Applications, vol. 17, pages 265–289, 2008. 10.1007/s00521-007-0120-z.

[Ohara 02] H. Ohara, I. Nishikawa, S. Miki & N. Yabuki. *Detection and recognition of road signs using simple layered neural networks*. In Neural Information Processing, 2002. ICONIP '02. Proceedings of the 9th International Conference on, volume 2, pages 626 – 630 vol.2, nov. 2002.

[Paisitkriangkrai 07] S. Paisitkriangkrai, C. Shen & J. Zhang. *An Experimental Evaluation of Local Features for Pedestrian Classification*. In Proceedings of the 9th Biennial Conference of the Australian Pattern Recognition Society on Digital Image Computing Techniques and Applications, pages 53–60. IEEE Computer Society, 2007.

[Ponce 06] J. Ponce, TL Berg, M. Everingham, DA Forsyth, M. Hebert, S. Lazebnik, M. Marszalek, C. Schmid, BC Russell, A. Torralba*et al. Dataset issues in object recognition*. Lecture Notes in Computer Science, vol. 4170, page 29, 2006.

[Qingsong 10] X. Qingsong, S. Juan & L. Tiantian. *A detection and recognition method for prohibition traffic signs*. In Image Analysis and Signal Processing (IASP), 2010 International Conference on, pages 583–586. IEEE, 2010.

[Rajesh 11] R. Rajesh, K. Rajeev, K. Suchithra, V.P. Lekhesh, V. Gopakumar & N.K. Ragesh. *Coherence Vector of Oriented Gradients for Traffic Sign Recognition using Neural Networks*. In International Joint Conference on Neural Networks, 2011.

[Rakotomamonjy 03] A. Rakotomamonjy. *Variable selection using svm based criteria*. The Journal of Machine Learning Research, vol. 3, pages 1357–1370, 2003.

[Ren 09] F.X. Ren, J. Huang, R. Jiang & R. Klette. *General traffic sign recognition by feature matching*. In Image and Vision Computing New Zealand, 2009. IVCNZ'09. 24th International Conference, pages 409–414. Ieee, 2009.

[Ribeiro 05] P.C. Ribeiro & J. Santos-Victor. *Human activity recognition from video: Modeling, feature selection and classification architecture*. In Proceedings of International Workshop on Human Activity Recognition and Modelling. Citeseer, 2005.

[Ruta 09] A. Ruta, Y. Li, M. Uxbridge, F. Porikli, S. Watanabe, H. Kage, K. Sumi & J. Amagasaki. *A New Approach for In-Vehicle Camera Traffic Sign Detection and Recognition*. In Proc. IAPR Conference on Machine Vision Applications, Japan, 2009.

[Ruta 10] A. Ruta, Y. Li & X. Liu. *Real-time traffic sign recognition from video by class-specific discriminative features*. Pattern Recognition, vol. 43, no. 1, pages 416–430, 2010.

[Ruta 11] A. Ruta, F. Porikli, S. Watanabe & Y. Li. *In-vehicle camera traffic sign detection and recognition*. Machine Vision and Applications, pages 1–17, March 2011.

[Sahoolizadeh 08] A.H. Sahoolizadeh, B.Z. Heidari & C.H. Dehghani. *A New Face Recognition Method using PCA, LDA and Neural Network*. International Journal of Computer Science and Engineering, vol. 2, no. 4, pages 218–223, 2008.

[Schapire 97] R.E. Schapire. *Using output codes to boost multiclass learning problems*. In MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-, pages 313–321. MORGAN KAUFMANN PUBLISHERS, INC., 1997.

[Schapire 99] R.E. Schapire. *A brief introduction to boosting*. Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence table of contents, pages 1401–1406, 1999.

[Sermanet 11] Pierre Sermanet & Yann LeCun. *Traffic Sign Recognition with Multi-Scale Convolutional Networks*. In International Joint Conference on Neural Networks, 2011.

[Shalev-Shwartz 07] S. Shalev-Shwartz, Y. Singer & N. Srebro. *Pegasos: Primal estimated sub-gradient solver for svm*. In Proceedings of the 24th international conference on Machine learning, pages 807–814. ACM, 2007.

[Shotton 08] J. Shotton, M. Johnson & R. Cipolla. *Semantic texton forests for image categorization and segmentation*. In Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, pages 1–8. IEEE, 2008.

[Sotelo 06] M.A. Sotelo, I. Parra, D. Fernandez & E. Naranjo. *Pedestrian detection using SVM and multi-feature combination*. In Intelligent Transportation Systems Conference, 2006. ITSC'06. IEEE, pages 103–108. IEEE, 2006.

[Stallkamp 11] Johannes Stallkamp, Marc Schlipsing, Jan Salmen & Christian Igel. *The German Traffic Sign Recognition Benchmark: A multi-class classification competition*. In International Joint Conference on Neural Networks, 2011.

[Stanciulescu 07] B. Stanciulescu, A. Breheret & F. Moutarde. *Introducing New AdaBoost Features for Real-Time Vehicle Detection*. Proceedings of

COGIS2007 Cognitive Systems with Interactive Sensors, Stanford University, USA, 2007.

[Suard 06] F. Suard, A. Rakotomamonjy & A. Bensrhair. *Object Categorization using Kernels combining Graphs and Histograms of Gradients*. Lecture Notes in Computer Science, vol. 4142, page 23, 2006.

[Tibshirani 07] R. Tibshirani & T. Hastie. *Margin trees for high-dimensional classification*. The Journal of Machine Learning Research, vol. 8, pages 637–652, 2007.

[Tu 05] Z. Tu. *Probabilistic boosting-tree: Learning discriminative models for classification, recognition, and clustering*. vol. 2, 2005.

[Tuytelaars 08] T. Tuytelaars & K. Mikolajczyk. *Survey on local invariant features*. FnT Computer Graphics and Vision, vol. 1, no. 1, pages 1–94, 2008.

[Viola 01] Paul Viola & Michael Jones. *Rapid object detection using a boosted cascade of simple features*. In IEEE CVPR, volume 1, pages 511–518, 2001.

[Viola 05] P. Viola, M.J. Jones & D. Snow. *Detecting pedestrians using patterns of motion and appearance*. International Journal of Computer Vision, vol. 63, no. 2, pages 153–161, 2005.

[Watanabe 09] T. Watanabe, S. Ito & K. Yokoi. *Co-occurrence histograms of oriented gradients for pedestrian detection*. Advances in Image and Video Technology, pages 37–47, 2009.

[Weston 99] J. Weston & C. Watkins. *Support vector machines for multi-class pattern recognition*. In Proceedings of the seventh European symposium on artificial neural networks, volume 4, pages 219–224, 1999.

[Weston 01] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio & V. Vapnik. *Feature selection for SVMs*. Advances in neural information processing systems, pages 668–674, 2001.

[Wohler 99] C. Wohler & J.K. Anlauf. *An adaptable time-delay neural-network algorithm for image sequence analysis*. Neural Networks, IEEE Transactions on, vol. 10, no. 6, pages 1531–1536, 1999.

[Wu 00] D. Wu, K.P. Bennett, N. Cristianini & J. Shawe-Taylor. *Enlarging the Margins in Perceptron Decision Trees*. Machine Learning, vol. 41, no. 3, pages 295–313, 2000.

[Wu 03] X. Wu, D. Zhang & K. Wang. *Fisherpalms based palmprint recognition*. Pattern Recognition Letters, vol. 24, no. 15, pages 2829–2838, 2003.

[Xie 09] Y. Xie, L.F. Liu, C.H. Li & Y.Y. Qu. *Unifying visual saliency with HOG feature learning for traffic sign detection*. In Intelligent Vehicles Symposium, 2009 IEEE, pages 24–29. IEEE, 2009.

[Yang 08] A.Y. Yang, S. Iyengar, P. Kuryloski & R. Jafari. *Distributed segmentation and classification of human actions using a wearable motion sensor network*. 2008.

[Zaklouta 09] Fatin Zaklouta, Bogdan Stanciulescu & Amaury Breheret. *Performance of Haar-like Features and Control Points on Pedestrian Detection*. In COGIS, 2009.

[Zaklouta 11a] Fatin Zaklouta & Bogdan Stanciulescu. *Real-time traffic sign recognition using spatially weigted HOG trees*. In IEEE International Conference on Advanced Robotics (ICAR). IEEE, 2011.

[Zaklouta 11b] Fatin Zaklouta & Bogdan Stanciulescu. *Warning Traffic Sign Recognition using a HOG-based K-d Tree*. In Intelligent Vehicles, 2011.

[Zaklouta 11c] Fatin Zaklouta, Bogdan Stanciulescu & Omar Hadmoun. *Traffic Sign Classification Using K-d trees and Random Forests*. In International Joint Conference on Neural Networks, 2011.

[Zhang 07] J. Zhang, M. Marszalek, S. Lazebnik & C. Schmid. *Local features and kernels for classification of texture and object categories: A comprehensive study*. In Computer Vision and Pattern Recognition Workshop, 2006. CVPRW'06. Conference on, pages 13–13. IEEE, 2007.

[Zhu 06] Q. Zhu, M.C. Yeh, K.T. Cheng & S. Avidan. *Fast human detection using a cascade of histograms of oriented gradients*. In Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, volume 2, pages 1491–1498. IEEE, 2006.

Reconnaissance d'Objets Multi-Classes pour des Applications ADAS et Vidéo Surveillance

Résumé : La détection de piétons et la reconnaissance des panneaux routiers sont des fonctions importantes des systèmes d'aide à la conduite (anglais : Advanced Driver Assistance System - ADAS). Une nouvelle approche pour la reconnaissance des panneaux et deux méthodes d'élimination de fausses alarmes dans des applications de détection de piétons sont presentées dans cette thèse.

Notre approche de reconnaissance de panneaux consiste en trois phases : une segmentation de couleurs, une détection de formes et une classification du contenu. Le *color enhancement* des regions rouges est amélioré en introduisant un seuil adaptatif. Dans la phase de classification, la performance du K-d tree est augmentée en utilisant un poids spatial. Les *Random Forests* obtienent un taux de classification de 97% sur le benchmark allemand de la reconnaissance des panneaux routieres (*German Traffic Sign Recognition Benchmark*).

Les besoins en mémoire et calcul sont réduits en employant une réduction de la dimension des caractéristiques. Les classifieurs atteignent un taux de classification aussi haut qu'avec une fraction de la dimension des caractéristiques, selectionée en utilisant des *Random Forests* ou *Fisher's Crtierion*. Cette technique est validée sur deux benchmarks d'images multiclasses : ETH80 et Caltech 101.

Dans une application de vidéo surveillance avec des caméras statiques, les fausses alarmes des objets fixes, comme les arbres et les lampadaires, sont éliminées avec la corrélation sur plusieurs trames. Les fausses alarmes récurrentes sont supprimées par un filtre complémentaire en forme d'arbre.

Mots clés : Aide à la Conduite (anglais : Advanced Driver Assistance System - ADAS), reconnaissance de panneaux routiers, Détection de piétons, vidéo surveillance, apprentissage automatique, segmentation de couleurs, réduction de dimension des caractéristiques

Multiclass Object Recognition for Driving Assistance Systems and Video Surveillance

Abstract: Pedestrian Detection and Traffic Sign Recognition (TSR) are important components of an Advanced Driver Assistance System (ADAS). This thesis presents two methods for eliminating false alarms in pedestrian detection applications and a novel three stage approach for TSR.

Our TSR approch consists of three phases: a color segmentation, a shape detection and a content classification. The red color enhancement is improved by using an adaptive threshold. The performance of the K-d tree classifier is augmented by introducing a spatial weighting. The Random Forests yield a classification accuracy of 97% on the German Traffic Sign Recognition Benchmark.

Moreover, the processing and memory requirements are reduced by employing a feature space reduction. The classifiers attain an equally high classification rate using only a fraction of the feature dimension, selected using the Random Forest or Fisher's Criterion. This technique is also validated on two different multiclass benchmarks: ETH80 and Caltech 101.

Further, in a static camera video surveillance application, the immobile false positives, such as trees and poles, are eliminated using the correlation measure over several frames. The recurring false alarms in the pedestrian detection in the scope of an embedded ADAS application are removed using a complementary tree filter.

Keywords: Advanced Driver Assistance Systems (ADAS), Traffic Sign Recognition (TSR), Pedestrian Detection, Video Surveillance, Machine Learning, Color Segmentation, Feature Space Reduction

MINES ParisTech

ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY