ABSTRACT

Title of dissertation: Model-driven and Data-driven Approaches for some Object Recognition Problems

Raghuraman Gopalan, Doctor of Philosophy, 2011

Dissertation directed by: Professor Rama Chellappa
Department of Electrical and Computer Engineering

Recognizing objects from images and videos has been a long standing problem in computer vision. The recent surge in the prevalence of visual cameras has given rise to two main challenges where, (i) it is important to understand different sources of object variations in more unconstrained scenarios, and (ii) rather than describing an object in isolation, efficient learning methods for modeling object-scene 'contextual' relations are required to resolve visual ambiguities.

This dissertation addresses some aspects of these challenges, and consists of two parts. First part of the work focuses on obtaining object descriptors that are largely preserved across certain sources of variations, by utilizing models for image formation and local image features. Given a single instance of an object, we investigate the following three problems. (i) Representing a 2D projection of a 3D non-planar shape invariant to articulations, when there are no self-occlusions. We propose an articulation invariant distance that is preserved across piece-wise affine transformations of a non-rigid object 'parts', under a weak perspective imaging model, and then obtain a shape context-like descriptor to perform recognition; (ii)

Understanding the space of 'arbitrary' blurred images of an object, by representing an unknown blur kernel of a known maximum size using a complete set of orthonormal basis functions spanning that space, and showing that subspaces resulting from convolving a clean object and its blurred versions with these basis functions are equal under some assumptions. We then view the invariant subspaces as points on a Grassmann manifold, and use statistical tools that account for the underlying non-Euclidean nature of the space of these invariants to perform recognition across blur; (iii) Analyzing the robustness of local feature descriptors to different illumination conditions. We perform an empirical study of these descriptors for the problem of face recognition under lighting change, and show that the direction of image gradient largely preserves object properties across varying lighting conditions.

The second part of the dissertation utilizes information conveyed by large quantity of data to learn contextual information shared by an object (or an entity) with its surroundings. (i) We first consider a supervised two-class problem of detecting lane markings from road video sequences, where we learn relevant feature-level contextual information through a machine learning algorithm based on boosting. We then focus on unsupervised object classification scenarios where, (ii) we perform clustering using maximum margin principles, by deriving some basic properties on the affinity of 'a pair of points' belonging to the same cluster using the information conveyed by 'all' points in the system, and (iii) then consider correspondence-free adaptation of statistical classifiers across domain shifting transformations, by generating meaningful 'intermediate domains' that incrementally convey potential information about the domain change.

Model-driven and Data-driven Approaches for some Object Recognition Problems

by

Raghuraman Gopalan

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2011

Advisory Committee:
Professor Rama Chellappa, Chair/Advisor
Professor Larry Davis
Professor David Jacobs
Professor Ankur Srivastava
Professor Min Wu

© Copyright by
Raghuraman Gopalan
2011

Dedication

To my parents, *Smt.* Lakshmi, and *Sri.* Gopalan, and my uncles *Sri.* Magesh, and *Sri.* Sridhar

Acknowledgments

Firstly I would like to thank my advisor Prof. Rama Chellappa for taking me as his student, and being patient with me over the last six years. There have been many times where I would go to his office to discuss not-well-thought-of ideas, and he would calmly listen through them and gently make sure that I realize what a problem definition means, and how to think about them so that it will be a useful work to the community. His dedication to work, and positive attitude towards research and life in general has been an immense source of inspiration that motivated me to do a good thesis. Secondly, I would like to thank Prof. Larry Davis for his critical comments on my research proposal which had a positive impact on my thesis. I thank Prof. David Jacobs, and Prof. Ankur Srivastava for giving me an opportunity to work with them during early part of my Ph.D., where I learnt many important aspects of doing good work. I also thank Prof. Min Wu for serving on my Ph.D. defense committee, and Prof. Shihab Shamma for his comments on my proposal exam. I thank all my teachers, in graduate school, in undergrad, and in my primary and secondary school for the values and knowledge they have imparted to me.

Being a part of the Center for Automation Research gives a good opportunity to discuss your ideas with other graduate students and post-docs who have slightly different background. I am very fortunate to have interacted with some of them, and I sincerely thank Prof. Pavan Turaga, Dr. Jagan Sankaranarayanan, and Prof. Ashok Veeraraghavan for all the discussions I had with them which have helped me very much. I also thank all my groupmates for providing a friendly environment, and

thank my officemates Nitess, Sima, Yi-Chen, Nazre, and Amon for putting up with me over the last several years. I thank Ms. Janice Perrone for all the administrative help, which she always does with a smiling face.

Outside school, I am deeply thankful to my room-mates and some of my very good friends: Kaka, Jishnu, Kadhu, RK, Bong bang, Bargav, Chandru, Balaji, and Rajesh. These are the guys with whom I spent most of my time with, and I can not find words to thank them for all the good times we have had during these years. A special thanks to Bargav for all his help in making sure my transfer from a day-scholar in India to the grad school at UMD was smooth. I would also like to thank my other good friends, to name a few, Srikanth, Sravya, Chamar, Shalabh, Taoo, Harita, Aparna, Rashi, Shitu, Alankar, Shraddha, Promita, and Ahmad for many happy moments we have shared. I also thank my undergrad friends, Vignesh, Raja and Ramnath – although we have taken different career paths, we still make it a point to meet when in India. I thank all my family friends and well-wishers, Ravi *chittapa* and family, Natarajan *periyappa* and family, and "Honorable mention" to my sweet (almost-) sister Gayathri, and her parents Kalyani *chitti* and Sankaranarayanan *chittappa*. At this point, I would like to mention 'Superstar' Rajnikanth, the famous actor from Tamil Cinema, and 'Phenom- The Undertaker' of the WWE for giving me the much needed positive energy at times of stress.

Finally, I am deeply indebted to the following four people who have played a very important role in various aspects of my life. First, I thank Sridhar uncle for all his positive thoughts, and affection he has shown to me. Next, I thank Magesh uncle for being a friend, a mentor, and a person with very high qualities that I

always admire. I am really fortunate to know a person like him, with so much love and affection, and all his good thoughts on my education and life. I then thank my father for instilling discipline in me during my early school days, where I was a very naughty kid, and gradually transforming into a very good friend with whom I can talk about anything in life without thinking. With his inherently kind approach in bringing me up, providing all necessities to me, and having a very honest ambition of seeing me in a good position in life and career, I am really grateful to have a father like him. Lastly, I thank my mother for all the love she has showered on me, for all the pains she has taken to make sure that I have a comfortable life, and for being very kind and polite in cultivating good habits in me. Being a very good and dedicated student in school, she did not have the opportunity to pursue higher education, and I humbly hope that this thesis will bring her lots of happiness (a small thing from a son to his loving mother).

In all, I thank God... for everything...

Table of Contents

List of Tables x

List of Figures xii

1 Introduction 1
- 1.1 Overview of the Dissertation . . . 2

2 Articulation-Invariant Representation of Non-planar Shapes 7
- 2.1 Related Work . . . 10
- 2.2 Problem Formulation . . . 11
- 2.3 Proposed Method . . . 12
  - 2.3.1 Approximate Convex Decomposition . . . 14
    - 2.3.1.1 A New Area-based Measure of Convexity . . . 16
    - 2.3.1.2 An Algorithm to Obtain Approximate Convex Segments . . . 17
  - 2.3.2 Shape Representation Invariant to Non-planar Articulations . 19
    - 2.3.2.1 Affine Normalization . . . 21
    - 2.3.2.2 Articulation Invariance . . . 21
- 2.4 Experiments . . . 22
  - 2.4.1 Non-planar Articulations . . . 22
    - 2.4.1.1 Intra-class articulations . . . 23
    - 2.4.1.2 Inter-class variations . . . 24
  - 2.4.2 Shape Retrieval . . . 26
  - 2.4.3 Experiment on the Convexity Measure . . . 27
- 2.5 Discussion . . . 28
- 2.6 Appendix: Properties of the convexity measure . . . 28

3 A Blur-robust Descriptor with Applications to Face Recognition 31
- 3.1 Space of Blur and Blur-Invariants . . . 34
- 3.2 Face Recognition Across Blur . . . 37
  - 3.2.1 Grassmann Manifold: Definition and some methodologies for recognition . . . 37
    - 3.2.1.1 Finding distance between points on $\mathbb{G}_{N,d}$ . . . 38
    - 3.2.1.2 Learning from data on $\mathbb{G}_{N,d}$ . . . 39
  - 3.2.2 Performing Recognition . . . 41
    - 3.2.2.1 Spatially uniform blur . . . 41
    - 3.2.2.2 Spatially varying blur . . . 41
- 3.3 Experiments . . . 43
  - 3.3.1 Effect of Quantization Noise . . . 45
    - 3.3.1.1 Uniform blur . . . 45
    - 3.3.1.2 Spatially varying blur . . . 47
  - 3.3.2 With Other Facial Variations . . . 48
    - 3.3.2.1 Comparison with existing methods . . . 49

3.3.2.2 Learning η_f from data 52
3.4 Discussion 54
3.5 Appendix: Robustness of the blur invariant - An analysis 54

4 Comparing and Combining Lighting Insensitive Approaches for Face Recognition 57
4.1 Description of Algorithms 61
4.2 Setting 1: No training set (on the possible lighting conditions) 63
4.2.1 Initial Comparisons 64
4.2.2 Facial Sub-regions 66
4.2.2.1 Planar models with albedo variations. 69
4.2.2.2 Shape variations in smooth objects. 71
4.2.2.3 Shape variations in objects with discontinuities . . . 73
4.2.3 Classifier combination 76
4.3 Setting 2: With prior training on different lighting conditions 78
4.3.1 Initial Comparisons 78
4.3.2 Classifier combination 80
4.3.3 Experiments on faces with more controlled lighting 82
4.4 Discussion 85
4.4.1 Comparison with the work of Tan and Triggs [177] 85
4.4.2 Comparison with the algorithm for face recognition using Sparse representations [200] 87

5 A Learning Approach Towards Detection and Tracking of Lane Markings 91
5.1 Detection of Lane Markings 96
5.1.1 Problem Definition 96
5.1.2 Modeling the spatial context of lane markings 97
5.1.2.1 A pixel-hierarchy feature descriptor 98
5.1.3 Learning the relevant contextual features through Boosting - Training the classifier 99
5.1.3.1 The Problem of Outliers in Training Set 100
5.1.3.2 Related work 101
5.1.3.3 Proposed method 102
5.1.4 Test phase: Detection (Localization) 105
5.2 Tracking and Learning some variations in road scene 107
5.2.1 Formulation of Particle Filters to Track Lane Markings 108
5.2.2 Learning the Road Scene using Tracked Parameters 109
5.2.2.1 Static world 111
5.2.2.2 Change in lateral motion of vehicle 111
5.2.2.3 Change in road geometry 112
5.2.2.4 Change in traffic pattern ahead of vehicle 112
5.3 Experiments 113
5.3.1 Detection of Lane Markings 113
5.3.1.1 Computations involved in determining f 115
5.3.2 Learning the Road Scene Variations 116

5.4 Discussion 119
5.5 Appendix: Outlier-robust boosting algorithm 120
5.5.1 Outlier Robustness of Adaboost - Discussion 120
5.5.2 Empirical evaluation 121

6 Max-margin Clustering: Detecting Margins from Projections of Points on Lines 127
6.1 Properties of projection of $\mathbb{X}$ on L 129
6.1.1 Case A: Two clusters 129
6.1.2 Case B: Multiple clusters 134
6.1.2.1 Existence of $SI^\star$ - Information conveyed by x_{i_p} . . . 135
6.1.2.2 Role of distance of projection d_{i_p} 139
6.2 A Maximum-margin clustering algorithm 141
6.2.1 Design Issues 143
6.3 Experiments 145
6.3.1 Synthetic data 145
6.3.2 Comparison with existing methods on real data 146
6.3.3 Experiments on vision problems 150
6.3.3.1 Face recognition across lighting variations 150
6.3.3.2 2D Shape matching 151
6.4 Discussion 152
6.5 Appendix: On detecting margins with a restricted analysis on line intervals between all pairs of points 153

7 Domain Adaptation for Object Recognition: An Unsupervised Approach 154
7.1 Related Work 158
7.2 Proposed Method 160
7.2.1 Motivation 160
7.2.2 Problem Description 160
7.2.3 Generating Intermediate Subspaces 161
7.2.4 Performing Recognition Under Domain Shift 163
7.2.5 Extensions 164
7.2.5.1 Semi-supervised Domain Adaptation 164
7.2.5.2 Adaptation Across Multiple Domains 165
7.3 Experiments 166
7.3.1 Comparison with Metric Learning Approach [161] 168
7.3.2 Comparison with Semi-supervised SVM's [6] 169
7.3.3 Studying the information conveyed by intermediate subspaces, and multi-domain adaptation 170
7.3.4 Comparison with unsupervised approaches on non-visual domain data 172
7.4 Discussion 174

8 A Computationally Efficient Method for Contour-based Object Detection 176
8.1 Baseline Face Detection Algorithm - A Brief Overview 179
8.1.1 Computational Complexity of the Baseline Algorithm 181
8.2 The Line Integral Image Representation 182
8.2.1 Speed-up 1 183
8.2.2 Speed-up 2 184
8.2.3 Speed-up 3 186
8.2.4 Speed-up 4: The Line Integral Image 187
8.3 Generalizability of the Line Integral Image Representation in Detecting Arbitrary Objects 190
8.4 Performance Analysis 191
8.4.1 Computational Efficiency - Detecting the Facial Contour . . . 192
8.4.2 Face Detection Accuracy 194
8.4.2.1 Feature Selection for Face Detection 194
8.4.2.2 Experimental Setting 197
8.4.3 Detecting Arbitrary Object Contours using the Line Integral Image Representation 199
8.5 Discussion 201

9 Future Work 204
9.1 Representing and matching non-planar shapes invariant to articulations 204
9.2 Unconstrained face recognition using subspace representations 205
9.3 Alternate strategies for max-margin clustering 205
9.4 Encoding transformation priors for unsupervised domain adaptation . 206

Bibliography 208

List of Tables

2.1 Shape matching costs on the dataset with an articulating human. The cost for our descriptor is around one-tenth of that of [115]. . . . 24
2.2 Recognition across inter-class non-planar articulations. 26
2.3 Retrieval results on MPEG-7 dataset [105]. 27

3.1 Performance comparison on FERET dataset [149] with different synthetic blurs. 49
3.2 Performance comparison on the FRGC 1.0 dataset [148] with real blurred images. 50

4.1 Performance of algorithms on homogenous gallery and heterogeneous gallery 66
4.2 Performance of algorithms on the rectangle model 71
4.3 Performance of algorithms on the cylinder model 73
4.4 Performance of algorithms on the triangle model 75
4.5 Performance comparison of combined classifier with the best individual algorithms 76
4.6 Recognition rates on the ORL face database [163] 84
4.7 Comparing the overall recognition rates of Tan and Triggs algorithm [177] with that of Gradient direction algorithm [40] on the Extended YaleB dataset [80] 86
4.8 Comparing the overall recognition rates of Tan and Triggs algorithm [177] with that of Gradient direction algorithm [40] on PIE dataset [173] 87
4.9 Performance of SRC based face recognition algorithm [200] on the extended Yale-B dataset 88
4.10 Performance of gradient direction algorithm [40] on the extended Yale-B dataset 88

5.1 Detection accuracy of Algorithm 4 with context in terms of the position error in the location of detected lane markings. The results show the position error in terms of neighborhood windows around the true lane marking locations in which the detection results occur. Performance across different false positive rates are given. 114
5.2 Statistics of the variance of polynomial coefficients for the scenarios discussed in Sections 5.2.2.1 to 5.2.2.3. 117
5.3 Statistics of the variance of bounding box locations and polynomial coefficients for the occlusion model discussed in Section 5.2.2.4. 118
5.4 Comparison of different tracking models on a set of 2000 frames . . . 119
5.5 Boosting methods on UCI Dataset [11]: comparing the proposed algorithm with other methods for outlier robustness of Adaboost. Results correspond to the mean and standard deviation of the generalization error. 126

5.6 UCI Dataset [11]: comparing the individual components our proposed algorithm. Results correspond to the mean and standard deviation of the generalization error. 126

6.1 (a) Clustering accuracy (in %) on a synthetic dataset of around 100 samples with $X \in \mathbb{R}^2$ and $X \in \mathbb{R}^3$.(b),(c) Comparison with max-margin clustering methods. Clustering accuracy (in %) for, (b): two-cluster problems, and (c): multi-cluster problems. (d) Comparison with methods that integrate dimensionality reduction and clustering on multi-cluster problem. 149

6.2 Clustering accuracy (in %) on datasets for face recognition across lighting condition, and shape matching. As before, the result for our method correspond to kernel parameters with least NCut cost, whereas for the other two methods we report the maximum clustering accuracy. 152

7.1 Comparison of classification performance with [161]. (a) with labels for all target domain categories. (b) with labels only for partial target categories. asymm and symm are two variants proposed by [161]. . . 166

7.2 Performance comparison across multiple domains in source or target, using the data from [161]. 173

7.3 Performance comparison with some unsupervised DA approaches on language processing tasks [27]. Key: B-books, D-DVD, E-electronics, and K-kitchen appliances. Each row corresponds to a target domain, and three separate source domains. 173

8.1 Face detection - Experimental results on CMU+MIT dataset. (Top) Dataset A, (Bottom) Dataset B 200

8.2 Computational requirements for different matching methods on the subset of five images from the ETHZ dataset [65]. 201

List of Figures

2.1 (a): Comparing distances across 2D projections of non-planar articulating shapes. (L-R) Shape 1 and 2 belong to the same 3D object, whereas shape 3 is from a different one. For a pair of points with same spatial configuration (yellow dots), Top: Inner distance [115] yields $\|d_{11} - d_{12}\|_2 > \|d_{12} - d_{13}\|_2$, whereas our method (bottom) gives $\|d_{21} - d_{22}\|_2 < \|d_{22} - d_{23}\|_2$. (b) Keypoints with similar shape description obtained from our method. Points were picked in the first frame, and their 'nearest neighbors' are displayed in other two frames. No holistic shape matching was done, emphasizing the importance of a shape representation. *(All figures are best viewed in color)* 8

2.2 (a): Error ϵ_{S_k} (2.5) illustrated by 2D projections, p_{ik}, with the camera parallel to planes 1 and 2. (b): Our model of an articulating object with two approximate convex parts p_1 and p_2, connected by a non-convex junction q_{12}. (c): Variation between ID and ED for a pair of points (green dots). $ID - ED$ is large for non-convex points, with the yellow dots indicating junction regions. (d): Information conveyed by (2.6) on the potential convex neighbors of u_l. The shape is enclosed by dashed red line. Color of other points u_m is given by $\frac{ED(u_l,u_m)}{ID(u_l,u_m)}$, with value 1 (white) for convex neighbors and tending towards 0 (black) for non-convex neighbors. 15

2.3 (a): Result of the segmentation algorithm (Section 2.3.1.2) on a 2D shape. Junction detection (yellow dots), initial segmentation, followed by the refined segmentation using the desired convexity (=0.85 here) as the user input. (b) Results on shapes from Brown [169] (Top row) and MPEG-7 [105] (Bottom row) datasets. (c): Segmenting a shape represented by voxel-sets using the same algorithm. 20

2.4 Dataset with non-planar articulations: Intra-class variations of an articulating human. (a): A set of actions observed from a single camera. (b): The same action observed by 4 cameras. The regions obtained from segmentation (Section 2.3.1) along with the points having similar shape representation (Section 2.3.2), are color-coded. 23

2.5 Dataset of non-planar articulations of different subjects. Four robots and human, with a total of 50 shapes. 25

2.6 Performance of our convexity measure on the dataset of [150]. Given at bottom of each shape are the convexity measures of [150] followed by ours (2.6). Our measure is insensitive to intra-class shape variations (text in redand *blue*), and is more sensitive when a part of the shape is disconnected from other parts (text in **green**). 28

3.1 Representing the blur-invariants as points on the Grassmann manifold $\mathbb{G}_{N,d}$. Given two subjects, y_1 and y_2, and the blurred test face $\tilde{y}$ belonging to subject 1, we illustrate how the $\mathcal{Y}_i$'s, created from them lie on the Grassmann. $\mathcal{Y}_1 = span(D(y_1))$ and $\tilde{\mathcal{Y}} = span(D(\tilde{y}))$ map to the same point, while $span(D(y_1) + \eta)$, where the noise is due to different lighting, lies closer to $span(D(y_1))$ than $span(D(y_2))$. 36
3.2 Analyzing the effect of noise due to quantization and sensor-related factors with uniform blur. (a): Comparing the mean error for intra-class faces, and the difference in errors between inter-class faces and intra-class faces on YaleB dataset, for d_G. The scores are normalized with maximum possible error for the inter-class faces. Range: minimum, mean, maximum. (b): Recognition rates for d_G on PIE and YaleB datasets. Range: minimum, mean, max. Noise settings (1-6) with clean gallery: no noise, with η_q, with η_q and AWGN with SNR=50, 20, 10, and 5 dB. (7-12): same as previous six settings, but with a blurred gallery. 44
3.3 Analyzing the effect of quantization noise with spatially varying blur. Recognition rates for d_G, on both PIE and YaleB datasets, with a patch-based classification (3.9) and with a uniform blur assumption while performing classification (3.7). Range: minimum, mean and maximum. 47
3.4 Comparison of our method with the existing approaches Nishiyama et al [136], Hu and Haan [90] on FERET dataset. Variations, in addition to synthetic Gaussian blur include, expression and alignment. 51
3.5 Examples of images, clean and blurred (both medium and extreme) from the UMD remote face dataset. Other facial variations include lighting, occlusion, expression and alignment. 52
3.6 Analyzing performance under unconstrained face variations in the UMD remote face dataset. Studying the impact of distance measures with learning (3.6) and without learning (3.5). (a): Testing on medium blurred images, recognition rates with and without lighting compensation (histogram equalization). Range: minimum, mean and maximum. (b): Same analysis, but with test images pertaining to extreme blur. 53

4.1 Sample images from the CMU-PIE dataset [173] 59
4.2 Sample images from the extended Yale-B dataset [80] 60
4.3 Performance comparison of all the class-independent algorithms on the entire face (without training). 65
4.4 Face Sub-regions 67
4.5 Performance comparison of class-independent algorithms on different regions without training 68
4.6 Rectangle model. Top: Variation in illumination. Bottom: Variation in albedo 70

4.7 Cylinder model. Top: Variation in illumination. Bottom: Variation in curvature (see that as the curvature increases from left image to right image in the bottom row, the change in the lighting pattern gets slower . . . 72
4.8 Triangle model. Top: Variation in illumination. Bottom: Variation in shape . . . 74
4.9 Performance comparison of combined classifier on all facial regions . . 78
4.10 Performance comparison of class-based, and class-independent algorithms on entire face. The same gallery lighting condition was used for all subjects . . . 79
4.11 CAR-CRR curves for PIE heterogeneous gallery experiments 81
4.12 CAR-CRR curves for the extended Yale-B heterogeneous gallery experiments . . . 82
4.13 Sample images from ORL face database [163] . . . 83

5.1 Sample road scenes; day and night-time images illustrating the effect of lighting variations and occluding vehicles as observed by the visual sensor. . . . 92
5.2 Pipeline of the proposed approach: detection with boosting on contextual features, and particle-filter based tracking to learn some road scene variations. . . . 92
5.3 L-R: An illustration of computing the pixel hierarchy descriptor h^x for a pixel x on hierarchical circles R_i, with the underlying F_i corresponding to the intensity image, edge map, and texture response (magnitude patterns of texture response are shown in this figure. 4 wavelet scales with 6 Gabor filter orientations were used). The weak learners h_j correspond to Haar filters, which when applied on $R \times F$ result in $h^x = \{h_j^x\}_{j=1}^{M_2} = f^x(I_o, I_s)$. h^x is the pixel-hierarchy descriptor of a pixel x. h^x computed for $x \in O, O'$ are used to train the boosting algorithm to compute the strong classifier $g^\star(x)$. This is then used to classify pixels in a test image corresponding to lane markings and others. . . . 96
5.4 Localized lane markings L_i obtained by performing outlier-robust boosting on the pixel-hierarchy contextual features. The number of hierarchy levels M_2 for R_i was determined by the least circle enclosing the entire image, for each pixel. All images were of size 240×320, and those regions R_i that did not contribute across all pixels were excluded from the feature set h. The pixels detected as lane marking by the boosting algorithm are grouped using the generalized Hough transform [13]. The parameterized result correspond to the polynomials enclosing the detected lane marking pixels. . . . 108

5.5 Results of lane tracking on day and night time video sequences. Images (a) through (d) illustrate lateral motion of the vehicle w.r.t the lane marking (Section 5.2.2.2). The parameters of polynomial fit [p_2 p_1 p_0] for these images are as follows: [-0.0009565 5.204 -411.4], [-0.009422 3.218 -92.47], [-0.0009464 1.8893 -2.416], [-0.0009211 0.4853 140.7] indicating substantial changes in p_1 and p_0. Image (e) has the following parameters: [-0.3199 0.5179 363.8], where the large variation in p_2 is due to the change in road geometry from straight to curved (Section 5.2.2.3). Images (f) and (g) are used to illustrate the effect of an occluding vehicle (Section 5.2.2.4). The polynomial coefficients of the middle lane markings in both images are [-0.0002544 0.94 -86.4], [-0.0002133 0.96 -90.4]. But the bounding box parameters [x_{bl} y_{bl} x_{tr} y_{tr}] are given by [100 1 225 268] [100 1 225 208]; The missing area R_m does not satisfy (5.18) due to the presence of the vehicle. 123
5.6 ROC curves for lane marking detection: comparing different learning methods on an internally collected dataset of 400 day/ night-time road images using a 5 fold cross-validation. The detection results correspond to pixel error of detected lane markings within a 3×3 neighborhood around the true pixel location. 124
5.7 Computing the contextual features f using Integral images [189]. Given an image representation F_i, to compute the cumulative information within the region D, we only need the value of $I^\star$ for the four corner points 1,2,3 and 4. The information can be computed according to the pattern of Haar-filters. 124
5.8 Road scenarios under inclement weather conditions. (a): Sample road images under rainy, snowy and foggy conditions collected from internet. We collected around 150 such images. Let us call them I_{web}. We retained other training images that we collected before (explained in Section 5.3.1). (b): (L-R) input test image; output of our algorithm without including I_{web} for training; output of our algorithm after including I_{web} in training (the test images shown in (b) were not used in training). We can see that under these conditions, the visual sensors are not adequate to detect lane markings completely, however learning does produce some improvement. 125

6.1 Left: A four class, linearly separable problem with $\mathbb{X} \in \mathbb{R}^2$. With known class labels, a max-margin classifier produces margins (shaded regions) with the separating hyperplanes indicated by the dashed lines. Right: In an unsupervised setting, how to identify these margin regions? Consider two lines L_1 and L_2, and project $\mathbb{X}$ on them (small yellow dots). Interval a_1 of L_1 has no projected points since it lies in margin region $\perp$ to the hyperplane that separates a cluster from *all other clusters*; whereas interval a_2 of L_2 (whose margin separates *only a pair* of clusters) has projected points from other clusters, with their minimum distance of projection d_1 more than that of d_2 for points projected elsewhere on L_2. In this work, we study the statistics of location and distance of projections of $\mathbb{X}$ on all lines L, to identify margins and perform clustering. 130

6.2 An illustration of projection of points on different line segments. The two clusters are represented by ellipses. Assume that points $\mathbb{X}$ are present everywhere inside the ellipses. When labels of $\mathbb{X}$ are available, S will be the separating hyperplane, and H_1 and H_2 are tangents to support vectors of either classes. M_S denotes the margin region (bounded by H_1, and H_2). $SI^\star = \gamma$ is the margin, and w is the normal to S. In a clustering scenario, where labels of $\mathbb{X}$ are unknown, consider two lines ($A, B \in L$ in $\mathbb{R}^2$). L_p refers to the segment of L enclosing all projections x_{i_p} (dots in black). It can be seen that on intervals in $A_p \perp S$, there is no x_{i_p} in the region corresponding to margin M_S; hence, there exist $SI^\star$. For any other line segment not perpendicular to S, say B_p, maximum possible $SI < \gamma$. 132

6.3 Partitioning the space of $\mathbb{X}$ into different regions. The data $\mathbb{X}$, shown in yellow circles, belong to three linearly separable clusters ($k = 3$). With known class labels, a supervised classifier $\hat{S}$ produces decision regions $R_i, i = 1$ *to* 3 belonging to the three classes, shown in red, blue, and green respectively. The margins regions M'_{S_i} are bounded by solid black lines, with their corresponding margins denoted by γ_i. The separating hyperplanes S_i are given in black dashed lines. We now divide the space of $\mathbb{X}$ into, (i) cluster regions CL_i in white dotted lines, and (ii) non-cluster regions that comprise of margin regions M'_{S_i} and T. 136

6.4 Illustrating the data-dependent nature of projections in intervals Int in T. Consider a three cluster problem, where the ellipses are completely filled with points. (a): Since $M'_{S_i} \subset M_{S_i}$, $SI^\star$ does not exist. However, the maximum possible SI occurs for intervals both in margin regions, and in T (shown with double head arrows). (b): When $M'_{S_i} \equiv M_{S_i}$, $SI^\star$ exists, and such intervals only belong to the margin regions (as in the case of a two-cluster problem). 138

6.5 Clustering results on synthetic data $\mathbb{X} \in \mathbb{R}^2$. (a),(b): Results using our method showing robustness to outliers, and in characterizing margin properties. (c): the first two figures shows sample mis-clustering result from KM, and the last two from NC - to illustrate the sensitivity of these algorithms to cluster center initialization, and parameter tuning respectively. (Data magnified by a factor of 5.) 147
6.6 Clustering results on synthetic data $\mathbb{X} \in \mathbb{R}^3$. (a) original data. Data is randomly distributed in x- and y- directions from 0 to 30, and has three splits in the z- direction: 0 to 15, 23 to 38, and 45 to 60. Clustering results are shown in the y-z plane. (b) our method, (c) KM, (d) NC. 148

7.1 Say we are given labeled data X from source domain corresponding to two classes + and $\times$, and unlabeled data $\tilde{X}$ from target domain belonging to class $\times$. Instead of assuming some relevant features or transformations between the domains, we characterize the domain shift between X and $\tilde{X}$ by drawing motivation from incremental learning. By viewing the generative subspaces S_1 and S_2 of the source and target as points on a Grassmann manifold $G_{N,d}$ (green and red dots respectively), we sample points along the geodesic between them (dashed lines) to obtain 'meaningful' intermediate subspaces (yellow dots). We then analyze projections of labeled $\times$, + (green) and unlabeled $\times$ (red) onto these subspaces to perform classification. 156
7.2 Sample retrieval results from our unsupervised method on the dataset of [161]. Left column: query image from target domain. Columns 2 to 6: Top 5 closest matches from the source domain. Source/ target combination for rows 1 to 4 are as follows: *dslr/amazon*, *webcam/dslr*, *amazon/webcam*, *dslr/webcam*. 167
7.3 (a), (b): Performance comparison with [6]. (a) Number of labeled source data = 300. (b) Number of labeled target data = 10. Semi-supervised SVM refers to the top performing SVM variant proposed in [6]. *Please note that our method also has an unsupervised working point (at position 0 on the horizontal axis).* (c) Studying the effect of N' on data from [161, 6]. Naming pattern refers to source domain/ target domain. Accuracy for $N' > 2$ is more than that for $N' = 2$, which says that the intermediate subspces do provide some useful information. 171

8.1 Speed-up methods: (i) A 5*5 DODE filter, (ii) Replacing ellipse by hexagon, (iii) Reusing hexagon values, (iv) The proposed line integral image I_l for three orientations of the hexagon - RE(Rising Edge - blue, long dashed lines in bold), FE(Falling Edge - red, dashed lines), SE(Straight Edge - green, solid lines). Each location on the lines denote the cumulative sum of pixels at that point, along the specified direction, (v) Pipeline for detecting frontal faces using the proposed method: Input image, preprocessing to compute I_l by overlaying three line orientations on the edge image, and detected face represented by a hexagon. 185
8.2 Comparison of the speed-up methods discussed in Section 8.2 by varying the hexagonal base length L, the filter size N, and the image size f. Speed-up 4 (8.16) results in substantial decrease in the computational requirements of the baseline algorithm [129], and is not affected much by the varying the parameters for polygon fitting. I_l, therefore, reduces the edge strength computations, without bringing a heavy overload from preprocessing. *Please note different y-axis scales for graphs in the last row.* 195
8.3 The proposed face detection system. Contour detection using I_l, followed by analyzing $R_i's$ by combining three appearance based descriptors using Support vector machines. 196
8.4 ROC curves of our face detection algorithm on the CMU+MIT dataset [166] on both frontal and profile faces. 198
8.5 Sample face detection results on the internally collected maritime face dataset 198
8.6 Detection results on ETHZ shape dataset [65]. Column orderings: (i) original image, (ii) edge image from the dataset, (iii) localization result (in blue/red) using I_l, and (iv) ground truth localization. . . . 203

Chapter 1

Introduction

Recognizing objects from images and videos is a fundamental problem in computer vision that has received significant attention over the last five decades. Starting from the early attempts on constrained recognition of object templates using a computer in 1960's and 70's, considerable advances have been made by understanding the projective geometry of objects through the eighties, followed by statistical learning methods in the nineties that modeled object variations by leveraging the information conveyed by large quantities of representative data exemplars, and more recently by analyzing object-scene contextual interactions, and by using local features/ attributes to represent objects. However, the increase in the prevalence of cameras and smart phones witnessed in recent years have accelerated the demand for systems with 'unconstrained' visual capabilities, where it is important to understand object variations by relaxing some existing model assumptions. Moreover, the ubiquitousness of these devices translates into the availability of large quantity of data, where efficient learning algorithms are required to extract 'relevant' information for recognition. These challenges collectively form the basic thrust of this dissertation in using models and data to address some problems related to object recognition.

We address this problem from two standpoints. First, given a single instance

of an object, how to utilize models for image formation, and local image features to obtain descriptions that are largely preserved across certain sources of object variations? Towards this end, in Chapters 2 to 4, we study variations in objects due to articulation, blur and lighting, and propose robust descriptors to perform recognition. A main part of this study involves the use of concepts from shape analysis and differential geometry. Secondly, keeping in pace with the ever-increasing availability of vision data, we propose algorithms to learn object-scene contextual interactions to perform recognition. By integrating perceptual observations with statistical models, in Chapters 5 to 7, we investigate the problems of detecting lane markings for autonomous vehicle navigation, unsupervised discovery of object categories using max-margin principles, and correspondence-free domain adaptation for statistical classifiers. Finally, since most computer vision applications demand real-time performance, we address the problem of efficiently representing image contours in Chapter 8. We propose an intermediate representation for piece-wise linear contours called the line integral image, and use it with off-the-shelf contour matching algorithms to demonstrate substantial reduction in computational requirements.

1.1 Overview of the Dissertation

In Chapter 2 we consider the problem of representing a 2D projection of a 3D non-planar shape invariant to 3D articulations, under no self-occlusion. By viewing an articulating object as a set of convex parts connected by non-convex junctions, we approximate articulations of the object as affine transformations of

its constituent parts. We then propose a distance metric that is largely preserved across articulations, by assuming a weak perspective camera to describe the imaging process. Using this distance metric, we design a shape context descriptor to represent the object, and evaluate it on shape retrieval tasks, and articulation-invariant object recognition settings.

Chapter 3 deals with representing an object invariant to the effects of arbitrary blurring, without imposing any restrictions on the parametric form of the blur function. Assuming that we know the maximum possible size of blur kernels, we represent the unknown blur kernel using a complete set of orthonormal basis functions and create a subspace that contains the set of all blurred versions of an object. We show that, under some assumptions, the subspaces created from a clean version of an object, and its blurred versions are the same. We then identify the space of these invariant subspaces with the Grassmann manifold, and use statistical methods defined on this manifold to perform object recognition across blur.

The focus of Chapter 4 is to empirically analyze the robustness of different local image features to object variations resulting from changing lighting conditions. We consider a face recognition setting with single image per person in the gallery. The variation between the gallery and probe is due to lighting, where the gallery itself had different lighting conditions across subjects. We analyze the lighting insensitive features on different facial regions, which have variations in albedo, surface normals and curvature, and empirically demonstrate that the orientation of image gradient is a good feature to perform recognition across large variations in illumination. Although these three studies were primarily designed for cases where only a single

instance of the object is available, they were extended to utilize the availability of more data through statistical modeling.

Chapter 5 marks a shift towards increasing the dependence on large quantity of data to learn contextual information between objects and the surrounding scene. We first study the problem of detecting lane markings from road video sequences. We propose a pixel hierarchy context descriptor that analyzes the visual features in concentric circles around each pixel, for both lane markings and non-lane markings, and then propose an outlier-robust boosting algorithm to learn relevant contextual features to perform detection. The detected lane markings are then tracked using a particle filter, without the knowledge of vehicle speed, by incorporating a static motion model for lane markings and learning relevant road scene variations from the statistics of the tracked parameters.

We then address the problem of clustering 'points' using maximum margin principles in Chapter 6. Unlike many existing methods that address this unsupervised problem by executing a supervised classifier with different label combinations to select the optimal cluster grouping, we perform a more basic study on the relationship between points belonging to a cluster and the margin regions that separate different clusters. By analyzing the projections of all points on the set of lines in the data space, we derive some basic properties that the projections on a line interval will satisfy, if and only if that line interval lies outside of a cluster. By transforming this problem from an integer optimization routine to one that detects the grouping of a 'pair of points' using information conveyed by 'all other points' (context), we demonstrate improved object identification on several machine learning datasets

and computer vision problems such as face recognition under illumination variations, and articulation-invariant shape retrieval.

Chapter 7 studies an interesting problem of unsupervised domain adaptation for object recognition, which hasn't received much attention in the literature. Here we look at the case where the underlying data distribution on which a classifier has been trained is different from that of the test data, while the conditional distribution of labels remains the same across training and testing. Instead of assuming the availability of certain discriminative features across domains, or using certain class of transformations to model the change in the marginal, we propose a framework, motivated by incremental learning, which generates several intermediate domains to help explain the unknown domain shift between the training data distribution and the test data distribution. To model 'contextual' information conveyed by intermediate domains on the transformation between the training and testing domains, we project the labeled training data on *all* domains to learn a discriminative classifier, and then classify the unlabelled test data by analyzing their projections on these domains.

In Chapter 8 we address computationally efficient image representations for contour-based object recognition. We propose a line integral image representation, which is a pre-processing stage that accumulates the edge strength of an image at different line orientations. Using this information, we compute the likelihood of a piece-wise linearly approximated contour in O(1) computatations for each linear side, in contrast to computations across as many pixels that make up the contour side. We then perform object recognition by using this representation with off-

the-shelf contour matching algorithms based on correlation, and Hough voting, and demonstrate substantial improvement in computational speed of these algorithms.

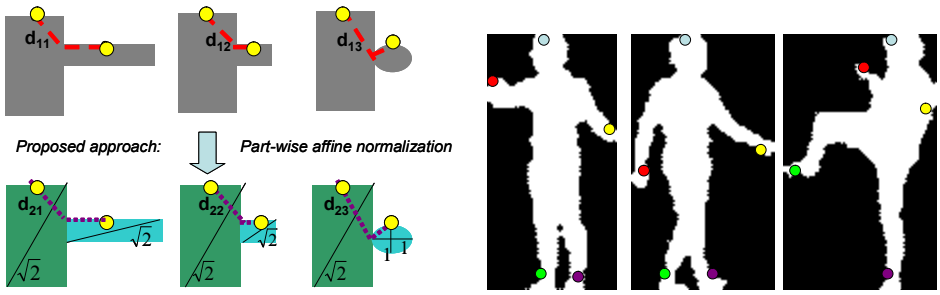
We finally conclude the dissertation in Chapter 9, by discussing potential directions in which subsequent questions raised by this dissertation can be addressed.

Chapter 2

Articulation-Invariant Representation of Non-planar Shapes

Understanding objects undergoing articulations is of fundamental importance in computer vision. For instance, human actions and hand movements are some common articulations we encounter in daily life, and it is henceforth interesting to know how different 'points' or 'regions' of such objects transform under these conditions. This is also useful for vision applications like, inferring the pose of an object, effective modeling of activities using the transformation of parts, and for human computer interaction in general.

Representation and matching of articulating shapes is a well-studied problem, and the existing approaches can be classified into two main categories namely, those based on appearance-related cues of the object (eg. [214]), and those using shape information which can be contours or silhouettes or voxel-sets (eg. [115, 32, 124]). Our work corresponds to the latter category, wherein we represent an object by a set of points constituting its silhouette. Although there have been many efforts ([169, 63, 167]) on deformation invariant 'matching' of shapes, there is relatively less work on 'representing' a shape invariant to articulations, eg. [115, 61, 160]. Among the above-mentioned efforts only [115] deals with 2D shapes and their representation mainly addresses planar articulations. However, most articulating shapes, such as a human, are non-planar in nature and there has been very little effort focusing on



(a) (b)

Figure 2.1: (a): Comparing distances across 2D projections of non-planar articulating shapes. (L-R) Shape 1 and 2 belong to the same 3D object, whereas shape 3 is from a different one. For a pair of points with same spatial configuration (yellow dots), Top: Inner distance [115] yields $\|d_{11} - d_{12}\|_2 > \|d_{12} - d_{13}\|_2$, whereas our method (bottom) gives $\|d_{21} - d_{22}\|_2 < \|d_{22} - d_{23}\|_2$. (b) Keypoints with similar shape description obtained from our method. Points were picked in the first frame, and their 'nearest neighbors' are displayed in other two frames. No holistic shape matching was done, emphasizing the importance of a shape representation. *(All figures are best viewed in color)*

this problem. This leads us to the question we are addressing in this work.

Given a set of points corresponding to a 2D projection of an articulating shape, how to derive a representation that is invariant/insensitive to articulations, when there is no self-occlusion? An example where this question is relevant is shown in Figure 2.1, along with results from our proposed shape representation. Such situations also arise when multiple cameras are observing a scene containing non-planar objects, where the projection of a particular 'region' of an object will depend on its relative orientation with the cameras. Accommodating for such variations, in addition to articulations (for which, each object can have different degrees of freedom) makes this a very hard problem.

Contributions: Under the assumption that a 3D articulating object can be expressed as a combination of rigid convex parts connected by non-rigid junctions that are highly non-convex, and there exists a set of viewpoints producing 2D shapes with all parts of the object visible; given one such instance of the 2D shape, we are interested in obtaining an invariant representation across articulations and view changes. We address this problem by,

1. Finding the parts of a 2D articulating shape through approximate convex decomposition, by introducing a robust area-based measure of convexity.

2. Performing part-wise affine normalization to compensate for imaging effects, under a weak perspective camera model, and relating the points using inner distance to achieve articulation invariance (upto a data-dependent error).

After reviewing the prior work in Section 2.1, we formally define the problem in

Section 2.2. We then present our proposed method in Section 2.3 by providing detailed analysis on the model assumptions. We evaluate our shape descriptor in Section 2.4 through experiments for articulation invariance on a dataset with non-planar shapes, including both intra-class and inter-class studies, and for standard 2D shape retrieval using the MPEG-7 [105] dataset. Section 2.5 concludes the chapter.

2.1 Related Work

Representation and matching of shapes described by a set of N-dimensional points has been extensively studied, and the survey paper by Veltkamp and Hagedoorn [187] provides a good overview of the early approaches. More recently, there have been advances in matching two non-rigid shapes across deformations. For instance, Felzenszwalb and Schwartz [63] used a hierarchical representation of the shape boundary in an elastic matching framework for comparing a pair of shapes. Yang et al [209] used a locally constrained diffusion process to relate the influence of other shapes in measuring similarity between a pair of shapes. Registering non-rigidly deforming shapes has also been addressed by [167] and [194]. Mateus et al [124] studied the problem of articulation invariant matching of shapes represented as voxel-sets, by reducing the problem into a maximal sub-graph isomorphism. There are also efforts, for instance by Bronstein et al [33], on explaining partial similarity between the shapes.

Though there has been considerable progress in defining shape similarity metrics and matching algorithms, finding representations invariant to a class of non-rigid

transformations has not been addressed extensively. This is critical for shape analysis because, rather than spending more efforts in matching, we stand to gain if the representation by itself has certain desirable properties. Some works towards this end are as follows. Elad and Kimmel [61] construct a bending invariant signature for isometric surfaces by forming an embedding of the surface that approximates geodesic distances by Euclidean distances. Rustamov [160] came up with a deformation invariant representation of surfaces by using eigenfunctions of the Laplace-Beltrami operator. However in this work, we are specifically interested in articulation insensitive representation of 3D shapes with the knowledge of its 2D projection alone. A key paper that addresses this particular problem is that of Ling and Jacobs [115]. They propose the inner distance, which is the length of the shortest path between a pair of points interior to the shape boundary, as an invariant descriptor of articulations when restricted to a set of translations and rotations of object parts. But such an assumption is applicable only for planar shapes, or when the shape is viewed using an ideal orthographic camera. Since neither of these two settings hold true in most real world scenarios, representing a 2D projection of a 3D non-planar shape invariant to articulations becomes an important problem, which we formalize in the following section.

2.2 Problem Formulation

An articulating shape $X \subset \mathbb{R}^3$ containing n parts, $\{P_i\}_{i=1}^n$, together with a set of Q junctions, can be written as $X = \{\bigcup_{i=1}^n P_i\} \bigcup \{\bigcup_{i \neq j,\ 1 \leq i,j \leq n} Q_{ij}\}$, where

1. $\forall i, 1 \leq i \leq n$, $P_i \subset \mathbb{R}^3$ is connected and closed, and $P_i \bigcap P_j = \phi, \forall i \neq j, 1 \leq i, j \leq n$

2. $\forall i \neq j, 1 \leq i, j \leq n, Q_{ij} \subset \mathbb{R}^3$, connected and closed, is the junction between P_i and P_j. If there is no junction between P_i and P_j, then $Q_{ij} = \phi$. Otherwise, $Q_{ij} \bigcap P_i \neq \phi$, $Q_{ij} \bigcap P_j \neq \phi$. Further, the volume of Q_{ij} is assumed to be small when compared to that of P_i.

Let $A(.)$ be the set of articulations of X, wherein $A(P_i) \in E(3)$ belong to the rigid 3D Euclidean group, and $A(Q_{ij})$ belong to any non-rigid deformation. Further, let V be the set of viewpoints, and $M \subset (A \times V)$ denote the set of conditions such that the 2D projection of X, say $S \subset \mathbb{R}^2$, has all parts visible; i.e. $S_k = \{\bigcup_{i=1}^{n} p_{ik}\} \bigcup \{\bigcup_{i \neq j,\ 1 \leq i,j \leq n} q_{ijk}\}, \forall k = 1\ to\ M$, where $p_{ik} \subset \mathbb{R}^2$ and $q_{ijk} \subset \mathbb{R}^2$ are the corresponding 2D projections of P_i and Q_{ij} respectively. The problem we are interested now is, given an instance of S, say S_1, how to obtain a representation $\tilde{R}(.)$ such that,

$$\tilde{R}(S_1) = \tilde{R}(S_k),\ \forall k = 1\ to\ M \tag{2.1}$$

2.3 Proposed Method

In pursuit of (2.1), we make the following assumptions. (i) X has approximate convex parts P_i that are piece-wise planar, and (ii) X is imaged using a weak-perspective (scaled orthographic) camera to produce $\{S_k\}_{k=1}^{M}$. Let each S_k be represented by a set of t points $\{u_{lk}\}_{l=1}^{t}$. Given two such points $u_{1k}, u_{2k} \in S_k$, we

would now like to obtain a distance D such that

$$D(u_{1k}, u_{2k}) = c, \forall k = 1 \ to \ M \tag{2.2}$$

where c is a constant, using which a representation $\tilde{R}(.)$ satisfying (2.1) can be obtained. Now to preserve distances D across non-planar articulations, we need to account for (atleast) two sources of variations. First, we compensate for changes in the 2D shape S due to changes in viewpoint V and due to the varying effect of imaging process on different regions of a non-planar X, by performing separate affine normalization to each part $p_{ik} \in S_k$. Let T denote the transformation that maps each part p_{ik} to p'_{ik}. Inherently, every point $u_{lk} \in S_k$ gets transformed as $T(u_{lk}) \rightarrow u'_{lk}$, where the transformation parameters depend on the part to which each point belongs. Next, to account for changes in S_k due to articulations A, we relate the two points $u'_{1k}, u'_{2k} \in S_k$ using the inner distance ID [115] which is unchanged under planar articulations. Essentially, we can write (2.2) as

$$D(u_{1k}, u_{2k}) = ID(u'_{1k}, u'_{2k}), \forall k = 1 \ to \ M \tag{2.3}$$

which, ideally, can be used to construct $\tilde{R}$ (2.1). But, in general,

$$D(u_{1k}, u_{2k}) = c + \epsilon_k, \forall k = 1 \ to \ M \tag{2.4}$$

where,

$$\epsilon_k = \epsilon_{P_k} + \epsilon_{D_k} + \epsilon_{S_k}, \forall k = 1 \ to \ M \tag{2.5}$$

is an error that depends on the data S_k. ϵ_{P_k} arises due to the weak perspective approximation of a real-world full-perspective camera. ϵ_{D_k} denotes the error in

the inner distance when the path between two points, u_{1k} and u_{2k}, crosses the junctions $q_{ijk} \in S_k$; this happens because the shape change of q_{ijk}, caused by an arbitrary deformation of the 3D junction Q_{ij}, can not be approximated by an affine normalization. But this error is generally negligible since the junctions q_{ijk} are smaller than the parts p_{ik}. ϵ_{S_k} is caused due to changes in the shape of a part p_{ik}, while imaging its original piece-wise planar 3D part P_i that has different shapes across its planes. An illustration is given in Figure 2.2(a).

Under these assumptions, we propose the following method to solve for (2.1). By modeling an articulating shape $S \subset \mathbb{R}^2$ as a combination of approximate convex parts p_i connected by non-convex junctions q_{ij}, we

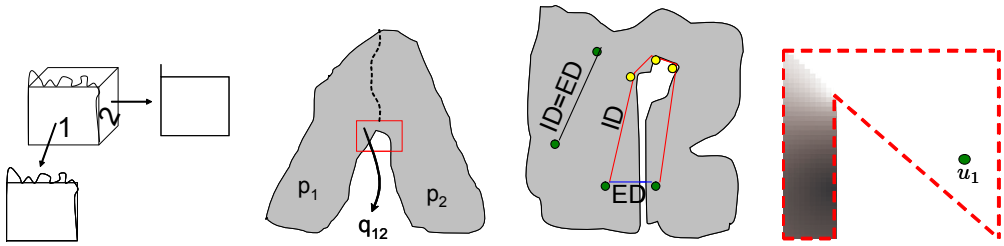
1. Determine the parts of the shape by performing approximate convex decomposition with a robust measure of convexity.

2. Affine normalize the parts, and relate the points in the shape using inner distance to build a shape context descriptor.

We provide the details in the following sub-sections.

2.3.1 Approximate Convex Decomposition

Convexity has been used as a natural cue to identify 'parts' of an object [88]. An illustration is given in Figure 2.2(b), where the object consists of two approximate convex parts p_1 and p_2, connected by a non-convex junction q_{12}. Since exact convex decomposition is NP-hard for shapes with holes [117], there are many approximate solutions proposed in the literature (eg. [114]). An important component of this



(a) (b) (c) (d)

Figure 2.2: (a): Error ϵ_{S_k} (2.5) illustrated by 2D projections, p_{ik}, with the camera parallel to planes 1 and 2. (b): Our model of an articulating object with two approximate convex parts p_1 and p_2, connected by a non-convex junction q_{12}. (c): Variation between ID and ED for a pair of points (green dots). $ID - ED$ is large for non-convex points, with the yellow dots indicating junction regions. (d): Information conveyed by (2.6) on the potential convex neighbors of u_l. The shape is enclosed by dashed red line. Color of other points u_m is given by $\frac{ED(u_l,u_m)}{ID(u_l,u_m)}$, with value 1 (white) for convex neighbors and tending towards 0 (black) for non-convex neighbors.

problem is a well-defined measure of convexity for which there are two broad categories of approaches namely, contour-based and area-based. Each has its own merits and limitations, and there are works addressing such issues (eg. [155, 218, 150]). But the fundamental problems, that of the intolerance of contour-based measures to small boundary deformations, and the insensitivity of area-based measures to deep (but thin) protrusions of the boundary, have not been addressed satisfactorily.

2.3.1.1 A New Area-based Measure of Convexity

In this work, we focus on the problem with existing area-based measures. We start from the basic definition of convexity. Given t points constituting an N-dimensional shape S', the shape is said to be convex if the set of lines connecting all pairs of points lie completely within S'. This definition, in itself, has been used for convex decompositions with considerable success (eg. [170, 190]). What we are interested here is to see if a robust measure of convexity can be built upon it.

We make the following observation. Given two points $u_1, u_2 \in S'$, let $ID(u_1, u_2)$ denote the inner distance between them, and $ED(u_1, u_2)$ denote their Euclidean distance. For a convex S', $ID = ED$ for any given pair of points, whereas for a non-convex S' this is not the case, as shown in Figure 2.2(c). We can see that, unlike the Euclidean distance, the inner distance inherently captures the shape's boundary and hence is sensitive to deep protrusions along it. Whereas, the difference between ID and ED is not much for minor boundary deformations. Using this property, which significantly alleviates the core issue of the existing area-based convexity measures, we propose a new measure of convexity as follows

$$1 - \frac{1}{(t^2 - t)} \sum_{u_l \in S'} \sum_{u_m \in S', m \neq l} \Big(1 - \frac{ED(u_l, u_m)}{ID(u_l, u_m)}\Big) \tag{2.6}$$

where t is the number of points in S', and $1 \leq l, m \leq t$. For a perfectly convex object, this measure will have a value one. We evaluate the robustness of this measure in Section 2.4.3, and discuss how it conforms to the properties that a convexity measure should satisfy in the Appendix.

2.3.1.2 An Algorithm to Obtain Approximate Convex Segments

We now use (2.6) to segment an articulating shape S into approximate convex regions p_i. We first study if $\frac{ED(u_1,u_2)}{ID(u_1,u_2)}$, in addition to saying whether points u_1 and u_2 belong to a convex region, can shed more information on the potential 'convex neighbors' of a particular point u_1. We proceed by considering a 2D shape S_1' having two convex regions, shown in Figure 2.2(d), and measure how $\frac{ED(u_1,.)}{ID(u_1,.)}$ from u_1 to all other $t-1$ points in S_1' vary. We observe that for those points lying in the same convex region as u_1 this term has a value one, whereas its value decreases for points that lie deeper into the other convex region. Hence (2.6) also gives a sense of ordering of convex neighbors around any specific point of interest. This is a very desirable property. Based on this, we formulate the problem of segmenting an articulating shape $S \subset \mathbb{R}^2$ as,

$$\min_{n,p_i} \sum_{i=1}^{n} \sum_{u_l \in p_i} \sum_{u_m \in p_i, u_l \neq u_m} \Big(1 - \frac{ED(u_l, u_m)}{ID(u_l, u_m)}\Big) \tag{2.7}$$

where $1 \leq l, m \leq t$, n is the desired number of convex parts, and p_i are the corresponding convex regions. We then obtain an approximate convex decomposition of S by posing this problem in a Normalized cuts framework [171] and relating all points belonging to S using the information conveyed by (2.6). The details are provided in Algorithm 1, which is applicable for any N-dimensional shape S'.

Estimate of the Number of Parts: We automatically determine the potential number of parts n using the information contained in (2.6). We do this by identifying junctions $q_{ij}, i \neq j, 1 \leq i, j \leq n$, which are the regions of high non-convexity. For those pair of points with $ID \neq ED$, we analyze the shortest path SP using which

Given a set of points t corresponding to an N-dimensional articulating shape S' (which can be a contour or silhouette or voxel-sets, for instance), an estimate $n(> 0)$ of the number of convex parts, and the desired convexity (a number between 0 and 1) for the parts,

(i) Connect every pair of points $(u_l, u_m) \in S'$ with the following edge weight

$$w_{u_l u_m} = exp^{-(\#junctions(u_l,u_m))} * exp^{\frac{-\|1-\frac{ED(u_l,u_m)}{ID(u_l,u_m)}\|_2^2}{\sigma_I^2}} * \begin{cases} exp^{\frac{-\|ID(u_l,u_m)\|_2^2}{\sigma_X^2}} & if \| ID(u_l,u_m) - ED(u_l,u_m) \|_2 \leq T_2 \\ 0 & otherwise \end{cases} \tag{2.8}$$

(ii) Do: Number of segments from $n - \eta$ to $n + \eta$ (to account for possible errors in junction estimates, see Figure 2.3(a) for example)

(iii) Perform segmentation using Normalized cuts [171]

(iv) Until: The resulting segments satisfy the desired convexity (2.6).

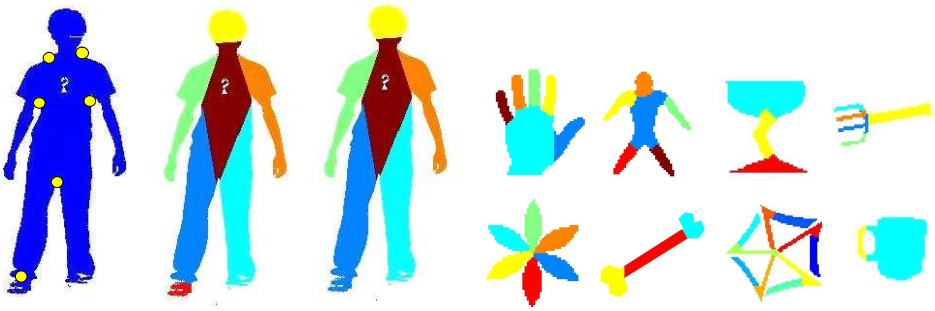
Algorithm 1: Algorithm for segmenting an N-dimensional shape into approximate convex parts.

their inner distance is computed. This SP is a collection of line segments, and its intermediate vertice(s) represent points, which by the definition of inner distance [115], bridge two potentially non-convex regions. This is illustrated in Figure 2.2(c) (see the yellow dots). We then spatially cluster all such points using a sliding window along the contour, since there can be many points around the same junction. Let the total number of detected junctions be n_j. The initial estimate of the number of parts n is then obtained by $n = n_j + 1$, since a junction should connect at least two parts.

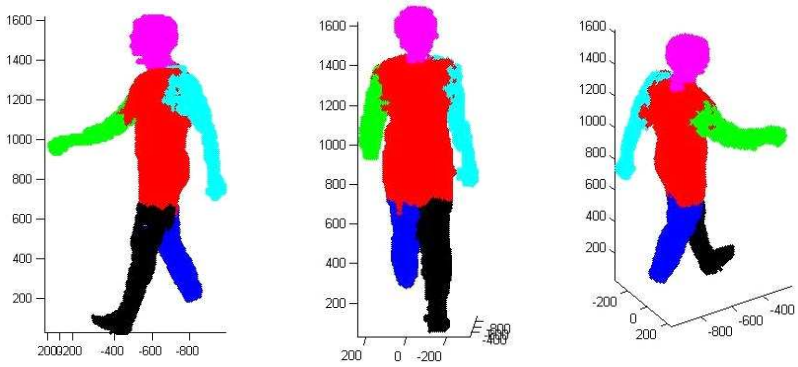
With this knowledge, we define the edge weight between a pair of points in (2.8) where the first two terms collectively convey how possibly can two points lie in the same convex region, and the third term denotes their spatial proximity. T_2, σ_I and σ_X are thresholds chosen experimentally. T_2 governs when two nodes need to be connected, and is picked as the mean of $ID(u_l, u_m) - ED(u_l, u_m), 1 \leq l, m \leq t$. σ_I and σ_X are both set a value of 5. We chose $\eta = 2$ and the desired convexity to be 0.85 in all our experiments. Sample segmentation results of our algorithm on silhouettes and voxel data are given in Figure 2.3.

2.3.2 Shape Representation Invariant to Non-planar Articulations

We now have an approximate convex decomposition of the articulating shape $S \subset \mathbb{R}^2$, i.e. $S = \{\bigcup_{i=1}^{n} p_i\} \bigcup \{\bigcup_{i \neq j,\ 1 \leq i,j \leq n} q_{ij}\}$. Given a set of M 2D projections of the 3D articulating shape X, $\{S_k\}_{k=1}^{M}$ with all n parts visible, we want to find a representation $\tilde{R}$ that satisfies (2.1). As before, let $\{u_{lk}\}_{l=1}^{t}$ be the number of points



(a) (b)



(c)

Figure 2.3: (a): Result of the segmentation algorithm (Section 2.3.1.2) on a 2D shape. Junction detection (yellow dots), initial segmentation, followed by the refined segmentation using the desired convexity (=0.85 here) as the user input. (b) Results on shapes from Brown [169] (Top row) and MPEG-7 [105] (Bottom row) datasets. (c): Segmenting a shape represented by voxel-sets using the same algorithm.

constituting each S_k. Let $u_{1k}, u_{2k} \in S_k$, be two such points. We now compute a distance $D(u_{1k}, u_{2k})$ satisfying (2.2) using a two step process,

2.3.2.1 Affine Normalization

To compensate for the change in shape of S_k due to the varying effect of the imaging process on different parts of the non-planar X and due to the changes in viewpoint V, we first perform part-wise affine normalization. This essentially amounts to finding a transformation T such that,

$$T(p_{ik}) \rightarrow p'_{ik} \tag{2.9}$$

where T fits a minimal enclosing parallelogram [168] to each p_{ik} and transforms it to a unit square. Hence this accounts for the affine effects that include, shear, scale, rotation and translation. This is under the assumption that the original 3D object X has piece-wise planar parts P_i for which, the corresponding 2D part $p_{ik} \in S_k$ can be approximated to be produced by a weak perspective camera.

2.3.2.2 Articulation Invariance

Let u'_{1k}, u'_{2k} be the transformed point locations after (2.9). As a result of T, we can approximate the changes in S_k due to 3D articulations A, by representing them as articulations in a plane. Hence, we relate the points u'_{1k}, u'_{2k} using inner distance (ID) and inner angle (IA) [115] that are preserved under planar articulations. We then build a shape context descriptor [20] for each point u'_{lk}, which is a histogram

h_{lk} in log-polar space, relating the point u'_{lk} with all other $(t-1)$ points as follows

$$h_{lk}(z) = \#\{u'_{mk}, m \neq l, 1 \leq m \leq t : ID(u'_{lk}, u'_{mk}) \times IA(u'_{lk}, u'_{mk}) \in bin(z)\} \quad (2.10)$$

where z is the number of bins. We now construct the representation $\tilde{R}(S_k) = [h_{1k}\ h_{2k}\ .\ .\ h_{tk}]$ that satisfies (2.1) under the model assumptions of Section 2.3.

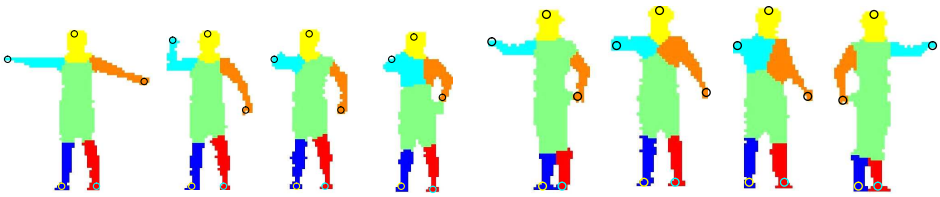
2.4 Experiments

We performed two categories of experiments to evaluate our shape descriptor (2.10). The first category measures its insensitivity to articulations of non-planar shapes on an internally collected dataset, since there is no standard dataset for this problem. Whereas, the next category evaluates its performance on 2D shape retrieval tasks on the benchmark MPEG-7 [105] dataset. We then validated the robustness of our convexity measure (2.6) on the dataset of Rahtu et al [150].

For all these experiments, given a shape $S \subset \mathbb{R}^2$, we model it as $S = \{\bigcup_{i=1}^{n} p_i\} \bigcup \{\bigcup_{i \neq j,\ 1 \leq i,j \leq n} q_{ij}\}$. We then sample 100 points along its contour, by enforcing equal number of points to be sampled uniformly from each affine normalized part p'_i. Then to compute the histogram (2.10), we used 12 distance bins and 5 angular bins, thereby resulting in total number of bins $z = 60$. The whole process, for a single shape, takes about 5 seconds on a standard 2GHz processor.

2.4.1 Non-planar Articulations

We did two experiments, one to measure the variations in (2.10) across intra-class articulations, and the other to recognize five different articulating objects.



(a) (b)

Figure 2.4: Dataset with non-planar articulations: Intra-class variations of an articulating human. (a): A set of actions observed from a single camera. (b): The same action observed by 4 cameras. The regions obtained from segmentation (Section 2.3.1) along with the points having similar shape representation (Section 2.3.2), are color-coded.

2.4.1.1 Intra-class articulations

We collected data of an articulating human, observed by four cameras, with the hands undergoing significant out-of-plane motion. The silhouettes, shown in Figure 2.4, were obtained by performing background subtraction, where the parts p_i of the shape (from Section 2.3.1) along with some points having similar representation (2.10) are identified by color-codes.

We divided the dataset of around 1000 silhouettes, into an unoccluded part of about 150 silhouettes (where there is no self-occlusion of the human) and an occluded part, and compared our representation (2.10) with the inner distance shape context (IDSC) [115] that is insensitive to articulations when the shape is planar. We chose to compare with this method since, it addresses articulation invariance in 2D shapes from the 'representation' aspect rather than matching. We used dynamic programming to obtain point correspondences between two shapes. Given in Table

Method	Matching cost (mean ± standard deviation)	
	Without occlusion	With occlusion
IDSC [115]	0.48 ± 0.21	3.45 ± 1.63
Ours	0.025 ± 0.0012	0.46 ± 0.11

Table 2.1: Shape matching costs on the dataset with an articulating human. The cost for our descriptor is around one-tenth of that of [115].

2.1 are the mean and standard deviations of the difference (in L_2 sense) of the descriptions (2.10) of the matched points. We do this for every pair of shapes in our dataset, with and without occlusion.

It can be seen that the matching cost for our descriptor is significantly less for the unoccluded pair of shapes, and is noticeably lower than [115] for the occluded pair too. This, in a way, signifies that our model assumptions (Section 2.3) is a good approximation to the problem of representing a shape invariant to non-planar articulations (Section 2.2).

2.4.1.2 Inter-class variations

We now analyze how our representation (2.10) can be used for recognition across the 2D shapes produced by different 3D non-planar articulating objects. We collected silhouettes of five different objects, a human and four robots, performing articulations observed from different viewpoints. There were ten instances per subject, with significant occlusion, leading to fifty shapes in total as shown in Figure

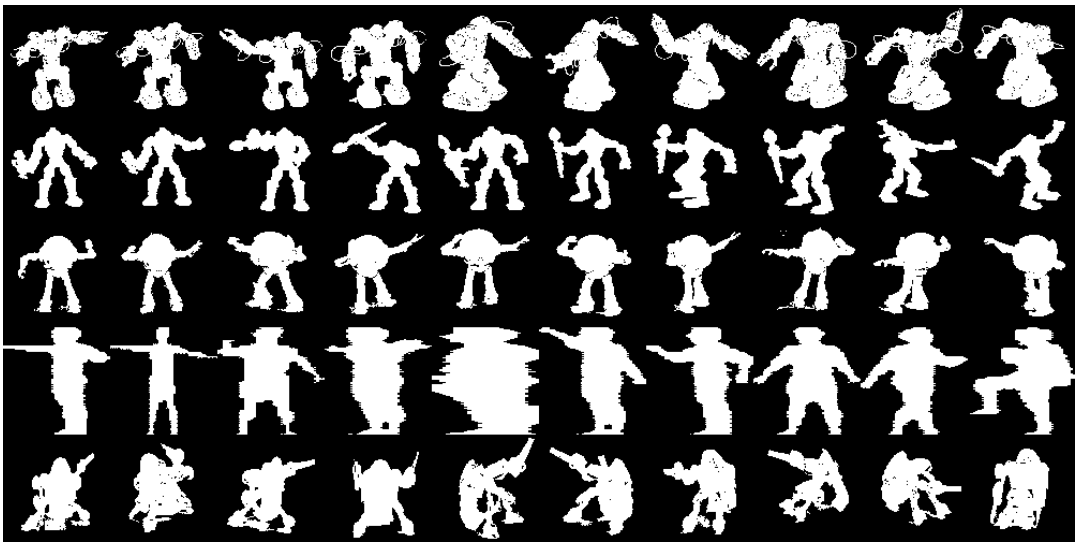


Figure 2.5: Dataset of non-planar articulations of different subjects. Four robots and human, with a total of 50 shapes.

2.5. We compared our algorithm with IDSC in both a leave-one-out recognition setting by computing the Top-1 recognition rate, and also in a validation setting using the Bulls-eye test that counts how many of the 10 possible correct matches are present in the top 20 nearest shapes (for each of the 50 shapes). We report the results in Table 2.2. It can be seen that our descriptor, in addition to handling non-planar articulations, can distinguish different shapes. The errors in recognition are mostly due to occlusions, which our model can not account for. It is an interesting future work to see how to relax our assumptions to address the more general problem stated in Section 2.2.

Method	Top-1 Recognition rate (in %)	BullsEye score (in %)
IDSC [115]	58	39.4
Ours	80	63.8

Table 2.2: Recognition across inter-class non-planar articulations.

2.4.2 Shape Retrieval

We then evaluated our descriptor for the 2D shape retrieval task to study its ability in handling general shape deformations, in addition to pure articulations. We used the benchmark MPEG-7 dataset [105], which contains 70 different shape classes with 20 instances per class. This is a challenging dataset with significant intra-class shape deformations. Some example shapes are given in Figure 2.3(b). The recognition rate is calculated using the Bulls-Eye test by finding the top 40 closest matches for each test shape, and computing how many of the twenty possible correct matches are present in it. The retrieval rates are given in Table 2.3, and we compare with the most recent and other representative methods.

Almost all shapes in this dataset are planar. So the least we would expect is to perform as well as [115], since but for handling non-planar articulations our representation resembles IDSC. The improvement using our representation is mainly due to cases where the shapes have distinct part structure, and when the variations in the parts are different. A part-driven, holistic shape descriptor can capture such variations better. It is interesting to see that we perform better than methods like [208, 209] that use sophisticated matching methods by seeing how different shapes in the dataset influence the matching cost of a pair of shapes. Hence through

Algorithm	BullsEye score (in %)
SC+TPS [20]	76.51
Generative models [181]	80.03
IDSC [115]	85.40
Shape-tree [63]	87.70
Label Propagation [208]	91.00
Locally constrained diffusion [209]	93.32
Ours	93.67

Table 2.3: Retrieval results on MPEG-7 dataset [105].

this study, we would like to highlight the importance of a good underlying shape representation.

2.4.3 Experiment on the Convexity Measure

Finally, we performed an experiment to evaluate our convexity measure (2.6) by comparing it with the recent work by Rahtu et al [150]. Since there is no standard dataset for this task, we provide results on their dataset in Figure 2.6. We make two observations. 1) For similar shapes (text in red and blue), the variation in our convexity measure is much smaller than that of [150]. This reinforces the insensitivity of our measure to intra-class variations of the shape, which is very desirable. 2) It can also been seen that our convexity measure is very sensitive to lengthy disconnected parts (text in green). This is mainly because, we compute pair-wise variations in ID and ED for all points in the shape, which will be high in such cases. These results, intuitively, are more meaningful than that of [150].

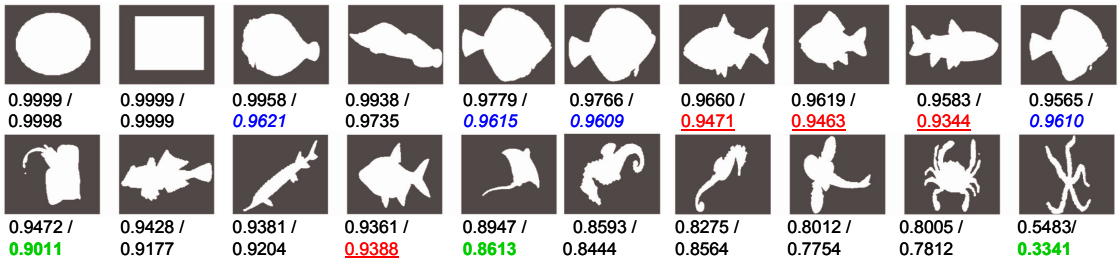


Figure 2.6: Performance of our convexity measure on the dataset of [150]. Given at bottom of each shape are the convexity measures of [150] followed by ours (2.6). Our measure is insensitive to intra-class shape variations (text in red and *blue*), and is more sensitive when a part of the shape is disconnected from other parts (text in **green**).

2.5 Discussion

We presented a method to represent a 2D projection of a non-planar shape invariant to articulations, when there is no occlusion. By assuming a weak perspective camera model, we showed that a part-wise affine normalization can help preserve distances between points, upto a data-dependent error. We then studied its utility through experiments for recognition across non-planar articulations, and for general shape retrieval. It is interesting to see how our assumptions can be relaxed to address this problem in a more general setting.

2.6 Appendix: Properties of the convexity measure

Here we verify the conformity of our convexity measure (2.11),

$$1 - \frac{1}{(t^2 - t)} \sum_{u_l \in S'} \sum_{u_m \in S', m \neq l} \Big(1 - \frac{ED(u_l, u_m)}{ID(u_l, u_m)}\Big) \tag{2.11}$$

to the set of four properties that such a measure must satisfy [218].

(1) *A convexity measure must be a number from* $(0, 1]$

It can be seen that the maximum value of (2.11) can not exceed 1 since $ED \leq ID$, and hence, the minimum value of (2.11) can not be less than 0.

(2) *The convexity measure of the given shape is 1 iff the shape is convex*

It can be seen that (2.11) will have the value 1 only if $ED = ID$ for all point-pairs, and this by definition of ED and ID happens only for convex objects.

(3) *There are shapes whose convexity measure is arbitrarily close to 0, implying that there is no gap between 0 and the minimal possible value of the measure*

We show this by example. Consider a star with n_s thin legs connected together by a very small junction (like the last shape in Figure 2.6, when the thickness of the parts approaches zero). For such a shape, $ED - ID$ will be arbitrarily large for most of the points, except for those small number of points lying within the same leg. Hence, the overall convexity measure will be close to zero in such cases.

(4) *The convexity measure is invariant under similarity transformation of the shape*

We will first deal with scaling. The results for translation and rotation follow from this. Let $\tilde{S}_1$ be a shape bounded by closed contour, and let $\tilde{S}_2$ denote its scaled version by a factor s. Let the area of two shapes be $f(\tilde{S}_1)$ and $f(\tilde{S}_2)$. Then, $f(\tilde{S}_2) = s^2 f(\tilde{S}_1)$. Moving into polar co-ordinates, let $\tilde{S}_1(r, \theta)$ be the continuous space equivalent of the term $\sum_{u_l, u_m \in \tilde{S}_1, l \neq m} (1 - \frac{ED(u_l, u_m)}{ID(u_l, u_m)})$ in (2.11). We now compute the ensemble mean (E_1) of $\tilde{S}_1(r_1, \theta_1)$ and compare it with that of $\tilde{S}_2(r_2, \theta_2)$, E_2, to verify if the convexity measure is invariant to scaling. Assuming points to be sampled

according to uniform distribution from $\tilde{S}_1$ and $\tilde{S}_2$,

$$E_1 = \frac{1}{f(\tilde{S}_1)} \int_{\tilde{S}_1} \tilde{S}_1(r_1, \theta_1) r_1 dr_1 d\theta_1, \;\; E_2 = \frac{1}{f(\tilde{S}_2)} \int_{\tilde{S}_2} \tilde{S}_2(r_2, \theta_2) r_2 dr_2 d\theta_2 \tag{2.12}$$

Since $r_2 = sr_1$, and $\tilde{S}_2(sr_1, \theta_2) = \tilde{S}_1(r_1, \theta_1)$ we have

$$E_2 = \frac{1}{s^2 f(\tilde{S}_1)} \int_{\tilde{S}_1} \tilde{S}_1(r_1, \theta_1) sr_1 s dr_1 d\theta_1 = E_1 \tag{2.13}$$

Since the ensemble means E_1 and E_2 are the same, given large number of samples, their empirical means obtained from (2.11) will converge to the same value (from the Central Limit Theorem). Hence, (2.11) is invariant under scale. Rotation and translation will not change (2.11) since the distances (ED and ID) are preserved. Hence proved.

Chapter 3

A Blur-robust Descriptor with Applications to Face Recognition

Understanding the effects of blur, which normally arises due to out-of-focus lens, atmospheric turbulence, and relative motion between the sensor and objects in the scene, is an important problem in image analysis applications such as face recognition. The image formation equation modeling the blurring process can be written as,

$$\tilde{y}(n_1, n_2) = (y * k)(n_1, n_2) + \eta(n_1, n_2) \tag{3.1}$$

where (n_1, n_2) denotes the pixel location at which a 2D convolution $*$ is performed between a $d_1 \times d_2$ clean image $y_{(d_1 \times d_2)}$ and an unknown blur point-spread function (PSF) $k_{(b_1 \times b_2)}$, to result in a blurred image[1] $\tilde{y}_{(d_1 \times d_2)}$. The ubiquitous noise present in the system, which can be due to quantization, or other sensor-induced errors, is represented by $\eta_{(d_1 \times d_2)}$. Existing approaches to handle the effects of blur can be classified as: (i) inverse methods based on deblurring, and (ii) direct methods based on invariants.

The goal of deblurring is to estimate the clean image y from the observed blurred image $\tilde{y}$. Even with complete knowledge of the blur kernel k, though this is an assumption which is hardly true in practice, inverting (3.1) to obtain y is an

[1] Although convolution of two signals results in a new signal whose size is larger than the other two signals, we are interested in the effects of convolution *only* on the region pertaining to spatial support of the input signal with larger size.

ill-posed problem due to the unknown nature of noise. Techniques for performing image restoration have been actively studied by the image processing community from the early 70's [9], and some of the prominent methodologies include: blind deconvolution [111] that does not assume any knowledge of the blur kernel, and thereby attempts to solve an under-constrained problem of estimating both k and y from $\tilde{y}$, non-blind deconvolution which assumes models for blur [212], learning priors on clean image statistics [64, 110], and using coded-computational photography techniques [2]. Regularization methods based on total variation [158] and Tikhonov regularization [179] constitute an integral part of this process. Such ideas have also been applied for recognizing faces across blur [90, 137, 136, 174].

In contrast to this, direct methods based on invariants search for those properties of the original image that are preserved across blur (under the assumption of zero noise). This is suited for applications where the goal is *not* to recover the *entire* clean image, but to extract some pertinent features using which further analysis can be done. Most efforts in this line of research are devoted to the specific class of centrally symmetric blur PSF's, which account for blur due to out-of-focus lens and atmospheric effects. The main observation behind these methods is as follows: Let $\tilde{y}_F$, y_F, and k_F denote the Fourier transform of $\tilde{y}$, y, and k respectively. Then, under no noise, (3.1) can be written as, $\tilde{y}_F(u,v) = y_F(u,v)k_F(u,v)$, where (u,v) denote the co-ordinates in frequency domain. The phase of these signals are related by, $\angle\tilde{y}_F(u,v) = \angle y_F(u,v) + \angle k_F(u,v)$. Since centrally symmetric kernels have a phase of either 0 or π, the tangent of phase, $\tan(\angle\tilde{y}_F(u,v)) = \tan(\angle y_F(u,v))$, is invariant to blur. Using this property, moment-based invariants were derived both

in spatial and Fourier domain, e.g. [69, 68]. Deriving invariants for linear motion blur has been addressed by [70]. There have been extensions of these works, which in addition to blur, accommodate invariance to rotation, similarity, and affine transformations [67, 176], and have been used for recognizing objects/ faces in distorted images [138, 4]. Robustness to noise is generally studied empirically.

Contributions: Our method belongs to the latter category. Unlike other methods that impose restrictions on the parametric form of the blur kernel, we represent an arbitrary blur kernel as a linear combination of orthonormal basis functions that span its space, and propose:

- A new blur invariant that can handle more general class of blurs, by creating a subspace that results from convolution of an image with each individual basis function, which thereby contains (but not limited to) the set of all blurred versions of that image;
- We provide a Riemannian geometric interpretation of the space spanned by these blur invariants, by studying them as points on the Grassmann manifold;
- We then utilize algorithms derived from this interpretation to perform face recognition across blur, where we demonstrate superior performance of the proposed method over various state-of-the-art methods.

Outline of the chapter: We derive the proposed blur invariant in Section 3.1. In Section 3.2, we study the utility of the invariant for the problem of recognizing faces under arbitrary blur, by considering degradation due to spatially uniform blur and spatially varying blur. We discuss the non-Euclidean nature of the space of

blur-invariants and show that it can be studied as a Grassmann manifold. Section 3.3 presents experiments, where we study the robustness of the invariant under different proportions of quantization noise and other facial variations such as, lighting, alignment, and expression between the gallery and probe. Section 3.4 concludes the chapter.

3.1 Space of Blur and Blur-Invariants

The goal of this section is to obtain a representation of an image y that is invariant to blurring with arbitrary k, under three assumptions: (i) there is no noise in the system ($\eta = 0$), (ii) the maximum size of the blur kernel $b_1 \times b_2 = N$ is known, and (iii) the $N \times N$ BTTB matrix corresponding to the unknown blur PSF, under zero boundary conditions for convolution, is full rank. More discussions on these assumptions are provided in the later part of this section.

For the case of 2D signals[2], we write any square-integrable, shift-invariant kernel k of size $b_1 \times b_2$ as, $k = \sum_{i=1}^{N} \alpha_i \phi_i$, where ϕ_i's are a *complete* set of orthonormal basis functions for $\mathbb{R}^{b_1 \times b_2}$, and α_i's are their combining co-efficients. Hence, under no noise, (3.1) becomes,

$$\tilde{y} = y * \sum_{i=1}^{N} \alpha_i \phi_i \tag{3.2}$$

where the specific form of k is determined by α_i's. We now create a dictionary

$$D(y) = [(y * \phi_1)^v (y * \phi_2)^v \ . \ . \ (y * \phi_N)^v] \tag{3.3}$$

[2]Although the following argument holds for convolution in higher dimensions, our primary focus will be on 2D signals.

of size $d \times N$, where $d = d_1 \times d_2$ with $d > N$, and $(.)^v$ denotes the vectorization operation. The column span of $D(y)$ is a subspace containing the set of convolutions of y with arbitrary kernels of maximum[3] size $b_1 \times b_2$. (i.e.) $span(D(y)) = \mathcal{Y} = \{y * k | k \in \mathbb{R}^{b_1 \times b_2}\}$, which is an N-dimensional subspace in $\mathbb{R}^d$. It is important to note here that the set of all blurred images of y (produced by convolving y with physically realizable blur kernels that have all their co-efficients non-negative, and summing to one), span only a part of this subspace.

Proposition 3.1.1 *$span(D(y))$ is a blur-invariant of y. In other words, $span(D(y)) = span(D(\tilde{y}))$, where $\tilde{y}$ is the blurred version of y.*

Proof Let $\Phi = [(\phi_1)^v \ (\phi_2)^v \ . \ . \ (\phi_N)^v]$ denote the $N \times N$ orthonormal matrix created from the basis functions. By writing convolution as matrix multiplication, (3.3) becomes $D(y) = Y\Phi$, where Y is a $d \times N$ matrix. The rows of Y are created by arranging the elements of y such that their multiplication with a ϕ_i will realize the effect of convolution (of y with ϕ_i) at all d corresponding pixels. Since Φ is full rank, $span(D(y)) = span(Y)$.

Now to prove Proposition 3.1.1, let us consider k_S to be the unknown blur kernel of (maximum) size $b_1 \times b_2$ that produced $\tilde{y}$ from a clean image y. From (3.3) we have,

$$D(\tilde{y}) = [(\tilde{y} * \phi_1)^v \ . \ . \ (\tilde{y} * \phi_N)^v] = [(y * k_S * \phi_1)^v \ . \ . \ (y * k_S * \phi_N)^v]$$

$$= Y[(k_S * \phi_1)^v (k_S * \phi_2)^v \ . \ . \ (k_S * \phi_N)^v] = Y K_S \Phi \tag{3.4}$$

[3]The only user-defined parameter is the size of the kernels, $b_1 \times b_2 = N$, and we discuss more on it in Sec 3.3.

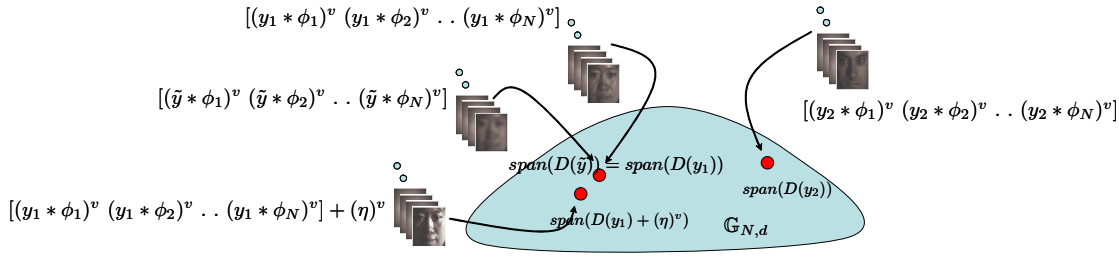


Figure 3.1: Representing the blur-invariants as points on the Grassmann manifold $\mathbb{G}_{N,d}$. Given two subjects, y_1 and y_2, and the blurred test face $\tilde{y}$ belonging to subject 1, we illustrate how the $\mathcal{Y}_i$'s, created from them lie on the Grassmann. $\mathcal{Y}_1 = span(D(y_1))$ and $\tilde{\mathcal{Y}} = span(D(\tilde{y}))$ map to the same point, while $span(D(y_1) + \eta)$, where the noise is due to different lighting, lies closer to $span(D(y_1))$ than $span(D(y_2))$.

where K_S is the BTTB matrix of size $N \times N$ corresponding to the kernel k_S. Since the column span of $D(\tilde{y})$ and $D(y)$ are same if K_S is full rank, $span(D(y)) = \mathcal{Y}$ is a blur-invariant.

Discussion: (i) Intuitively, what we are claiming here is: given a clean image y and its corresponding blurred image $\tilde{y}$, we can use the basis functions to either generate a blur function that converts y to $\tilde{y}$, or to produce a deblur function that brings $\tilde{y}$ to y. Further, since the basis functions can span *any* blur or deblur function of a known maximum size, we do not have constraints on the parametric form of blur functions that can be handled (unlike other invariants). (ii) Regarding the assumption on the rank of K_S, we would like to stress that although some blur PSF's are not invertible, their BTTB matrices are generally full rank (see [85], and the references therein). These BTTB matrices, however, can be extremely ill-conditioned at times.

But since we do not invert these matrices, we do not encounter problems related to high condition numbers of matrix inversion that are prevalent in deblurring-based approaches. (iii) Having said that, there always exist practical scenarios such as the non-zero measurement noise that render some of our assumptions invalid. We present an analysis on the robustness of the invariant to additive perturbations in the Appendix.

3.2 Face Recognition Across Blur

We now study the utility of invariant $\mathcal{Y}$ for the problem of recognizing faces across blur, where we empirically evaluate its robustness to sensor-related noise and the presence of other facial variations between the gallery and probe. Let us consider an M class problem with $\{y_i\}_{i=1}^{B}$ denoting the gallery faces, either clean or blurred, belonging to all subjects. Let $\tilde{y}$ denote the blurred probe image which belongs to one of the M classes. The problem we are looking at is, given y_i's and $\tilde{y}$, find the identity $i^\star \in \{1, 2, ..., M\}$ of $\tilde{y}$. From the gallery and probe images, we first create their respective dictionaries $D(y_i)$'s and $D(\tilde{y})$ using (3.3). We now compare $\tilde{\mathcal{Y}}$ with $\mathcal{Y}_i$ to perform recognition.

3.2.1 Grassmann Manifold: Definition and some methodologies for recognition

Computing similarity measures between subspaces is a well-studied problem. Among such similarity measures, those that account for the underlying geometry of

the problem imposed by some physical constraints, are more meaningful. The space of blur-invariants can be identified with the Grassmann manifold $\mathbb{G}_{N,d}$, which is the space of N-dimensional subspaces in $\mathbb{R}^d$ containing the origin. The blur-invariant $\mathcal{Y}$ is a point on $\mathbb{G}_{N,d}$. An illustration is provided in Figure 3.1. Understanding the geometric properties of the Grassmann manifold have been the focus of works like [199, 60, 1], and these have been utilized in some vision problems with subspace constraints, e.g. [17, 84, 120]. A compilation of statistical analysis methods on this manifold can be found in [42]. Since a full-fledged explanation of these methods is beyond the scope of this chapter, we refer the interested readers to the papers mentioned above. We now use some of these results to compute the distance between the blur-invariants. We specifically focus on the following two cases.

3.2.1.1 Finding distance between points on $\mathbb{G}_{N,d}$

The first method is to use the distance between points on the manifold for classification, which has more relevance when the gallery contains only one image per person. Formally, the Riemannian distance between two subspaces, say $\mathcal{Y}_1$, and $\mathcal{Y}_2$, is the length of the shortest geodesic connecting those points on the Grassmann manifold. One way of obtaining this length is to compute the direction (velocity) matrix A such that the geodesic along that direction, while starting at $\mathcal{Y}_1$, reaches $\mathcal{Y}_2$ in unit time. A is computed using the inverse exponential map. However, since the expression for the inverse exponential map is not available analytically for the Grassmann manifold, we use a numerical method [79] as given in Algorithm 2. The

length of A gives the distance[4] d_G between $\mathcal{Y}_1$ and $\mathcal{Y}_2$, and we use $trace(AA^T)$, where $(.)^T$ is the transpose operator, as the metric to compute the length. More formally, if $A_{\mathcal{Y}_1,\mathcal{Y}_2}$ is the direction matrix between $\mathcal{Y}_1$ and $\mathcal{Y}_2$,

$$d_G(\mathcal{Y}_1, \mathcal{Y}_2) = trace(A_{\mathcal{Y}_1,\mathcal{Y}_2} A^T_{\mathcal{Y}_1,\mathcal{Y}_2}) \tag{3.5}$$

We then perform recognition using d_G, in a nearest-neighbor fashion, as given in (3.7,3.9).

3.2.1.2 Learning from data on $\mathbb{G}_{N,d}$

In cases where there is more data available for each person in the gallery portraying other facial variations, it paves the way for performing statistics on the point cloud on $\mathbb{G}_{N,d}$. Since the blur-invariants have a resultant dimension of $(d-N) \times N$ [60], with d significantly higher than N, it would require large number of samples to learn class-specific distributions. Hence we pursued the method of Hamm and Lee [84] that performs kernel linear discriminant analysis on the blur-invariants using the projection kernel $k_P(\bar{D}(y_1), \bar{D}(y_2)) = \|\bar{D}(y_1)^T \bar{D}(y_2)\|_F^2 = trace[(\bar{D}(y_1)\bar{D}(y_1)^T)(\bar{D}(y_2)\bar{D}(y_2)^T)]$, which is a Mercer kernel that implicitly computes the inner product between $\bar{D}(y_i)'s$ in the space obtained using the following

[4]**Note:** We also note that the distance between $\mathcal{Y}_i's$ can be obtained by studying the differential geometry of $\mathbb{G}_{N,d}$ through a study of geometry of N-planes in the Euclidean space $\mathbb{R}^d$ [199]. An example of such a distance is the arc-length metric $d^2_{arc}(\mathcal{Y}_1, \mathcal{Y}_2) = \sum_{i=1}^{N} \theta_i^2$, a function of principal angles θ_i between the two subspaces spanned by the (orthonormalized) columns of $d \times N$ matrices $D(y_1)$ and $D(y_2)$ respectively, which can be derived from the intrinsic geometry of the Grassmann manifold [199, 60].

Given two dictionaries $D(y_1)$ and $D(y_2)$ whose column space is a point on $\mathbb{G}_{N,d}$, we determine the velocity matrix A such that travelling in this direction from $\mathcal{Y}_1$ leads to $\mathcal{Y}_2$ in unit-time. Let $\bar{D}(y_1)$ and $\bar{D}(y_2)$ denote the $d \times N$ matrices obtained by orthonormalizing the columns of $D(y_1)$ and $D(y_2)$ respectively.

- Compute the $d \times d$ orthogonal completion Q of $\bar{D}(y_1)$.
- Compute the thin CS decomposition of $Q^T \bar{D}(y_2)$ given by
$$Q^T \bar{D}(y_2) = \begin{pmatrix} X_C \\ Y_C \end{pmatrix} = \begin{pmatrix} U_1 & 0 \\ 0 & \tilde{U}_2 \end{pmatrix} \begin{pmatrix} \Gamma(1) \\ -\Sigma(1) \end{pmatrix} V_1^T$$
- Compute $\{\bar{\phi}_i\}$ which are given by the arcsine and arccos of the diagonal elements of Γ and Σ respectively, i.e. $\gamma_i = \cos(\bar{\phi}_i)$, $\sigma_i = \sin(\bar{\phi}_i)$. Form the diagonal matrix $\bar{\Phi}$ containing $\bar{\phi}_i$'s as diagonal elements.
- Compute $A = \tilde{U}_2 \bar{\Phi} U_1$.

Algorithm 2: Numerical computation of the velocity matrix: The inverse exponential map [79].

embedding; $\omega_P : \mathbb{G}_{N,d} \rightarrow \mathbb{R}^{d \times d}, span(\bar{D}(y_i)) \rightarrow \bar{D}(y_i)\bar{D}(y_i)^T$. To make the chapter self-contained, we present the details of this method in Algorithm 3.

3.2.2 Performing Recognition

3.2.2.1 Spatially uniform blur

In the case when k remains unchanged over all pixels (n_1, n_2) of a $d_1 \times d_2$ image y (3.1), we perform recognition with the two subspace distances (SD), d_G (3.5) and KLDA (Algorithm 3), using the nearest neighbor classification method. The identity of probe $\tilde{y}$ is determined by,

$$i^\star = \arg\min_i SD(D(\tilde{y}), D(y_i)) \tag{3.7}$$

3.2.2.2 Spatially varying blur

We now study the more difficult problem, where the blur kernel k is spatially varying. This occurs when different parts of the scene are affected differently by blur, with some common examples being, out-of-focus blur in objects with depth discontinuities, and motion blur when there is a sudden change in intensity values of a *region* due to object movements. The image formation equation for this case can be written as,

$$\tilde{y}_n = y_n * k_n \tag{3.8}$$

where the subscript n indicates the pixel location. Since a blur kernel acts on a local spatial *neighborhood*, allowing it to change at *every* pixel location makes the problem severely under-constrained. A common assumption made to overcome this

From the gallery faces y_i's constituting M classes, and probe faces $\tilde{y}_i$, compute their respective dictionaries $D(y_i)$ and $D(\tilde{y}_i)$. Orthonormalize their columns to obtain $\bar{D}(y_i)$ and $\bar{D}(\tilde{y}_i)$.

Training:

- Compute the matrix $[K_{train}]_{ij} = k_P(\bar{D}(y_i), \bar{D}(y_j))$ for all $\bar{D}(y_i), \bar{D}(y_j)$ in the training set, where k_P is the projection kernel defined earlier.
- Solve $\max_\gamma L(\gamma)$ by eigen-decomposition (3.6), with $K^\star = K_{train}$.
- Compute the (M-1)-dimensional coefficients, $F_{train} = \gamma^T K_{train}$

Testing:

- Compute the matrix $[K_{test}]_{ij} = k_P(\bar{D}(y_i), \bar{D}(\tilde{y}_j))$ for all $\bar{D}(y_i)$ in training, and $\bar{D}(\tilde{y}_j)$ in testing.
- Compute (M-1)-dimensional coefficients, $F_{test} = \gamma^T K_{test}$, by solving for (3.6) with $K^\star = K_{test}$.
- Perform 1-NN classification from the Euclidean distance between F_{train} and F_{test}

The Rayleigh quotient $L(\gamma)$ is given by,

$$L(\gamma) = \max_\gamma \frac{\gamma^T K^\star (\bar{V} - 1_B 1_B^T / B) K^\star \gamma}{\gamma^T (K^\star (I_B - \bar{V}) K^\star + \sigma^2 I_B) \gamma} \tag{3.6}$$

where $K^\star$ is the kernel (Gram) matrix (K_{train} or K_{test}), 1_B is a uniform vector $[1...1]^T$ of length B corresponding to the number of gallery images, $\bar{V}$ is the block-diagonal matrix whose m^{th} block ($m = 1$ *to* M) is the uniform matrix $1_{B_m} 1_{B_m}^T / B_m$, and B_m is the number of gallery images in the mth class, and $\sigma^2 I_B$ is a regularizer to make computations stable.

Algorithm 3: Kernel Linear Discriminant Analysis (KLDA) [84].

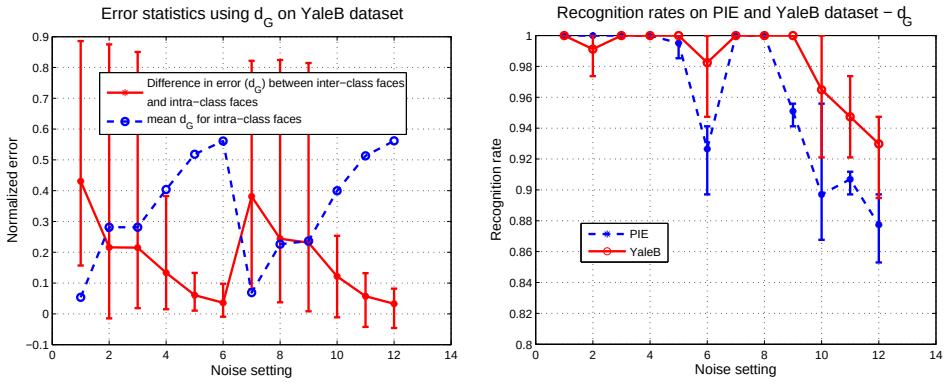
condition is to assume the blur to be locally uniform [37], which is valid in most practical cases. Along these lines, if the blur is assumed to be uniform over a region of size $d_1' \times d_2'$ (with $d_1' > b_1$, and $d_2' > b_2$), we can perform recognition by dividing the image into T overlapping patches of size $d_1' \times d_2'$ each, and rewriting (3.7) as,

$$i^\star = \arg\min_i \sum_{t=1}^{T} SD(D(\tilde{y})_t, D(y_i)_t) \tag{3.9}$$

where the subscript t denotes the patch at which the quantities in (3.9) are computed, and $D(.)_t$'s are points on $\mathbb{G}_{N,d'}$, $d' = d_1' \times d_2'$. The inherent assumption while matching patches is that the faces are aligned. However, for those patches where there is a transition between blur kernels, the column space of $D(.)_t$ will not be invariant to blur. The percentage of such instances depends on the nature of spatially varying blur, and we perform an empirical study in Section 3.3.

3.3 Experiments

We performed two sets of experiments to study the robustness of the blur invariant to noise. For the purpose of illustration, we express noise in the system (3.1) as $\eta = \eta_q + \eta_f$, where η_q denotes the noise due to quantization (which is relevant while studying synthetically created blur images) and other sensor-related issues, and η_f denotes facial variations other than blur such as, lighting, expression, alignment and occlusion, which are common in an unconstrained face recognition setting. We now study, (i) the effect of η_q on recognition when blur is the only source of variation between the gallery and probe (i.e. $\eta_f = 0$), and (ii) when there are other facial variations η_f between the gallery and probe, where we analyze the



(a) (b)

Figure 3.2: Analyzing the effect of noise due to quantization and sensor-related factors with uniform blur. (a): Comparing the mean error for intra-class faces, and the difference in errors between inter-class faces and intra-class faces on YaleB dataset, for d_G. The scores are normalized with maximum possible error for the inter-class faces. Range: minimum, mean, maximum. (b): Recognition rates for d_G on PIE and YaleB datasets. Range: minimum, mean, max. Noise settings (1-6) with clean gallery: no noise, with η_q, with η_q and AWGN with SNR=50, 20, 10, and 5 dB. (7-12): same as previous six settings, but with a blurred gallery.

role of learning from data representative of such conditions.

A note on constructing D(.): In all these experiments, the only user-controlled parameter is the maximum size of blur kernel, which determines the number of columns N of the dictionary $D(.)$. If $(b_1^\star, b_2^\star)$ denote the maximum of b_1 and b_2 over all possible blur kernels, then the value of N should be $b_1^\star \times b_2^\star$. At the same time, $N < d$ since otherwise, the basis will span the entire image space. Hence, in our experiments (except those on spatially varying blur), we chose $N = d_1/2 \times d_2/2$. The columns of the identity matrix I_N were used to represent $\{\phi_i^v\}_{i=1}^N$, although any

complete set of orthonormal basis functions can be used.

3.3.1 Effect of Quantization Noise

We first study how the recognition rates vary in the presence of quantization noise (η_q), and different levels of additive white Gaussian noise (AWGN) that model sensor-induced errors. We compute the distance between subspaces using d_G (3.5) with one image per person in the gallery, in the presence of *synthetically created* uniform blur, spatially varying blur and blurred gallery. We performed experiments using two datasets, the CMU-PIE [173] and the extended YaleB [108], to verify the generalizability of these results. We used the illumination subset of PIE dataset that has 68 subjects[5] with 21 different lighting conditions, and the YaleB dataset that has 38 subjects under 64 illumination settings. In this experiment, both gallery and probe have the same lighting conditions.

3.3.1.1 Uniform blur

We synthetically created blurred images corresponding to the following four categories; motion blur, out-of-focus blur, Gaussian blur, and non-parametric blur (created by generating random non-negative values for a kernel that sum to one). Experiments were performed across different blur kernel sizes (with the maximum size being 24×20) and lighting condition of images (same for both gallery and

[5] All the images were resized to 48×40, resulting in $d = 1920$. This results in $N = 24 \times 20$.

probe), for the following settings: no noise ($\eta = 0$), with quantization noise[6] η_q, with η_q and AWGN resulting in the following four SNR values (in dB), 50, 20, 10 and 5. These settings were first used for clean gallery, and then repeated for blurred gallery, thereby leading to twelve noise settings. Recognition was performed in a one-vs-all fashion by comparing each probe with all gallery images using d_G (3.5,3.7). Hence for an experimental trial, independent of the twelve different noise settings, the gallery and probe contain 68 images each for PIE dataset, and 38 each for YaleB. The results averaged over several such trials for recognition rates on PIE and YaleB datasets, and the statistics of error (distance d_G) on YaleB dataset are given in Figure 3.2.

Observations: From these results, we see that, (i) the difference in error between inter-class faces and intra-class faces reduce as noise increases. At the same time, the mean error for intra-class faces increases with noise. This explains the reason why recognition rate goes down with increasing noise levels. (ii) Even under no noise, irrespective of a clean or blurred gallery, the mean error for correct matches is non-zero. Although, theoretically, the span of the dictionary created from blurred face of a subject is the same as that created from its clean face, the presence of various system-related noise is the reason for such errors. (iii) For noise settings under a blurred gallery, the matching error statistics follow similar trends as that of clean gallery. This is primarily because, the invariant $span(D(y))$ is a subspace containing the set of all blurred versions of an image y, and hence does not depend

[6]Since we synthetically generate blurred images, they are stored explicitly as image files before processed by the algorithm so that the quantization effect is simulated.

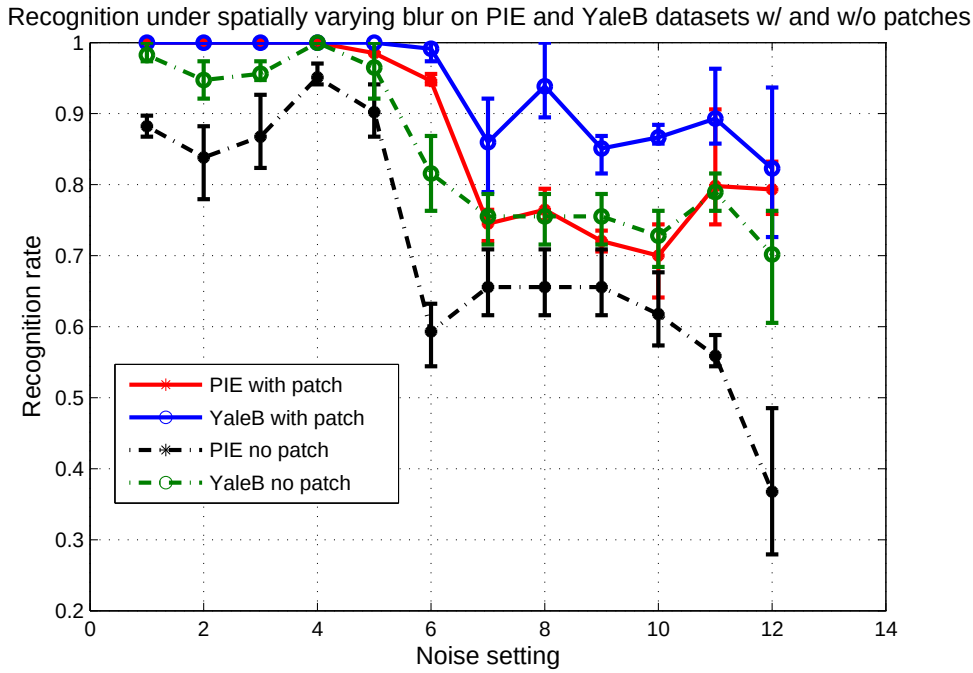


Figure 3.3: Analyzing the effect of quantization noise with spatially varying blur. Recognition rates for d_G, on both PIE and YaleB datasets, with a patch-based classification (3.9) and with a uniform blur assumption while performing classification (3.7). Range: minimum, mean and maximum.

on whether y is clean or blurred (Proposition 3.1.1). (iv) Computational time for recognition on the PIE dataset: To compute the dictionary $D(\tilde{y})$ for a probe face and to evaluate (3.5,3.7), it takes about 0.5 seconds on a 4GHz processor.

3.3.1.2 Spatially varying blur

We then performed the same set of experiments when the nature of blur is spatially varying (even for the blurred gallery). We selected arbitrary sized patches from images, on which one of the four above-mentioned kernels were used to create synthetic blurred images. To perform recognition, T different overlapping patches of

size $d' = d_1/2 \times d_2/2$ were extracted from the image, and their respective dictionaries were created with $N = d_1/4 \times d_2/4$ columns. Also note that, many kernels used to create blurred images were of size more than $d_1/4 \times d_2/4$. This was done to preserve generalization with respect to the nature of spatially varying blur. With this setup, recognition was performed using d_G (3.5,3.9), and the results are presented in Figure 3.3. We also compared with recognition that does not use patches to account for the spatially varying nature of the blur, but perform recognition by approximating the blur as uniform (as in Section 3.3.1.1).

Observations: We can see that, (i) as in the case of uniform blur, the recognition rates decrease as noise increases. For the same noise setting, a further reduction in recognition rates is observed when compared with uniform blur, especially for the blurred gallery. This is primarily due to two cases where, (a) the size of blur kernel is more than $d_1/4 \times d_2/4$ for which, the span of the dictionary $D(.)$ will not contain the blur kernel; and (b) even otherwise, in the regions experiencing a transition between blur kernels, $span(D(.))$ is not an invariant. (ii) Assuming the blur function to be patch-wise uniform is better than approximating the spatially varying blur to be uniform throughout the image. However, there is no standard solution for determining the patch size.

3.3.2 With Other Facial Variations

We now study a more practical setup where we allow for other facial variations, in addition to blur. The main focus, however, is not on explicitly accounting for

Method	Recognition rate (in %) - Type of blur		
	Gaussian	Linear motion	Both
Nishiyama et al [136]	88.3	82.3	82.9
Ours (d_G)	97.21	97.15	97.12

Table 3.1: Performance comparison on FERET dataset [149] with different synthetic blurs.

such variations, but rather to study the robustness of the invariant in their presence.

3.3.2.1 Comparison with existing methods

We used synthetically blurred faces from the FERET dataset [149], and real blurred faces from the FRGC 1.0 dataset [148] for comparison, by following the experimental setup presented in [137, 136]. We used the 'fa'-gallery, and 'fb'-probe subsets of the FERET dataset, which has faces of 1001 subjects, with one image per subject. Faces of the same person across 'fa' and 'fb' have small variations in expression and alignment. The original image size of 128× 128 was resized to 64× 64. We created nine different synthetically blurred sets of 'fb' using Gaussian kernels of size 5×5 with σ ranging from 0 to 8 in steps of 1, and added 30 dB white Gaussian noise to perform recognition. The recognition rates are presented in Figure 3.4, where we compare with existing deblurring-based methods. We then created synthetic Gaussian blur with random values for σ in the range $(0, 8]$, and synthetic linear motion blur with the length of blur having values 5,10,15,20, at

Method	**Recognition on** a subset (in %)	**Recognition on** the full dataset (in %)
Hu and Haan [90]	67.1	-
Nishiyama et al [137]	73.5	-
Ahonen et al [4]	-	45.9
Ahonen et al (with lighting compensation) [4]	-	74.5
Ours- d_G (3.5,3.7)	87.1	69.6
Ours- d_G (with lighting compensation)	-	84.2

Table 3.2: Performance comparison on the FRGC 1.0 dataset [148] with real blurred images.

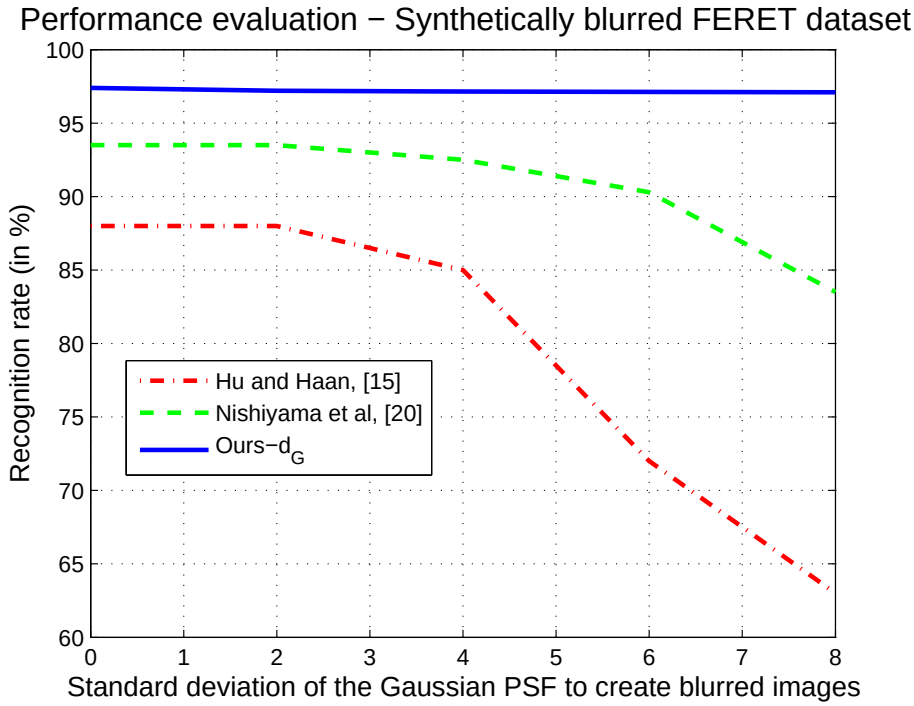


Figure 3.4: Comparison of our method with the existing approaches Nishiyama et al [136], Hu and Haan [90] on FERET dataset. Variations, in addition to synthetic Gaussian blur include, expression and alignment.

angles 0, 0.25π, 0.5π, and 0.75π. White Gaussian noise was added to result in a 30dB SNR, and the recognition rates are given in Table 3.1. The main observation is that the performance of our method is almost the same across different types of blur in the test image (Proposition 3.1.1).

We then evaluated our method on the FRGC 1.0 dataset that has real blurred images. We used the Expt. 4 protocol that has 152 subjects, with one clean face per person in the gallery taken under controlled lighting. The test set contains 608 images, under uncontrolled lighting, of which the 366 affected by blur (mostly due to out-of-focus camera) were chosen for the experiment. In addition to blur

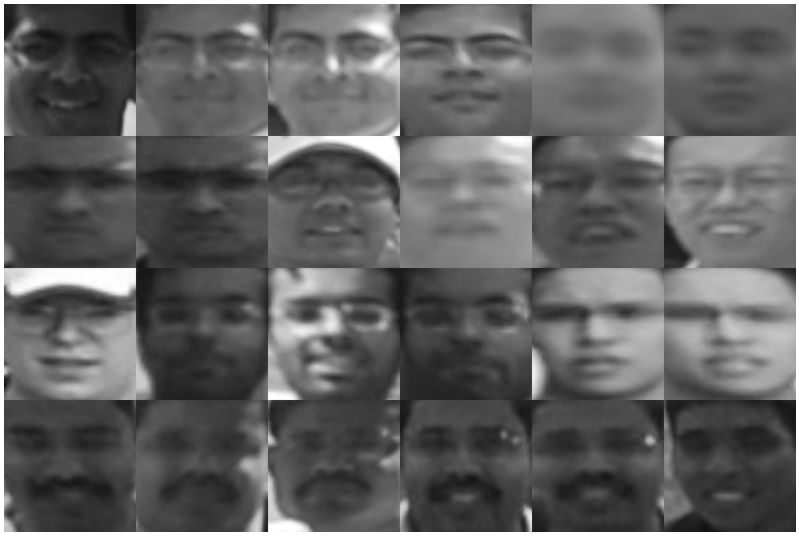


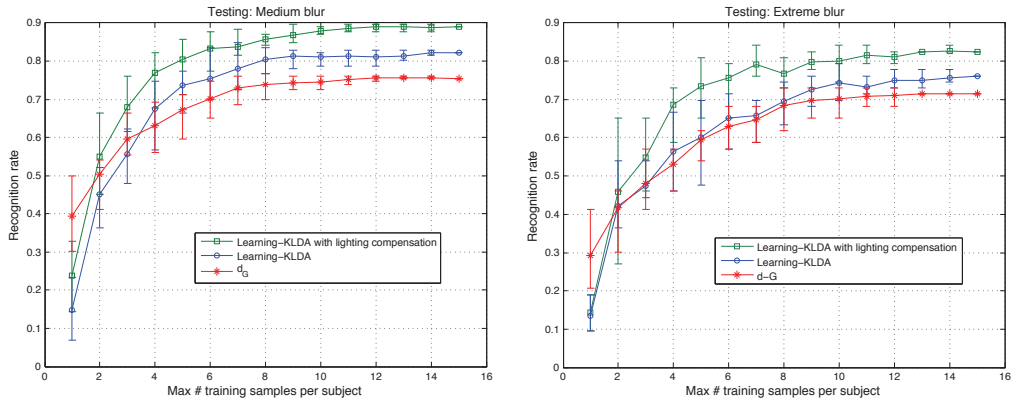
Figure 3.5: Examples of images, clean and blurred (both medium and extreme) from the UMD remote face dataset. Other facial variations include lighting, occlusion, expression and alignment.

and lighting, the gallery and test images have small variations in expression and alignment. We resized the images to 64×64, and present the recognition results in Table 3.2. We also compared with the invariant-based method of Ahonen et al [4] on the same dataset, but with all 608 probe images. We present recognition rates with and without lighting compensation (histogram equalization) in Table 3.2. We observe an improved performance due to explicit accounting for lighting variations.

3.3.2.2 Learning η_f from data

We now consider the availability of data portraying different instances of η_f to perform recognition. We used the UMD remote face dataset[7] comprising of 17 subjects, where in addition to moderate-to-severe blur, there are moderate changes

[7]The dataset will be publicly available soon.



(a) (b)

Figure 3.6: Analyzing performance under unconstrained face variations in the UMD remote face dataset. Studying the impact of distance measures with learning (3.6) and without learning (3.5). (a): Testing on medium blurred images, recognition rates with and without lighting compensation (histogram equalization). Range: minimum, mean and maximum. (b): Same analysis, but with test images pertaining to extreme blur.

in lighting, alignment, expression and occlusions. The experimental setup is as follows. The gallery has 168 clean face images (i.e. no blur) with different examples of η_f, with a *maximum* of 15 images per subject. We have two test sets with real blur, one with 146 moderately blurred images, and an extremely blurred set of 63 images. Gallery and probe have different variations of η_f. Sample images are shown in Figure 3.5. Recognition was performed using the nearest neighbor rule, (i) without learning, using d_G (3.5,3.7), and (ii) with learning using kernel discriminant analysis (3.6,3.7). The results, as a function of maximum number of gallery images per person, is shown in Figure 3.6. It can be seen that the recognition accuracy does improve when more data is used, even when η_f is not explicitly accounted for.

3.4 Discussion

We have made an attempt at understanding the space of blurred versions of an image. We created a subspace resulting from convolutions of the image with a complete set of orthonormal basis functions that could represent the blur kernel, which under some assumptions, was shown to be invariant to blur. We demonstrated the utility of this representation for face recognition under blur through experiments on standard datasets, and analyzed its robustness to the presence of noise and facial variations other than blur. From the point of view of performing robust face recognition under *unconstrained* settings, it is interesting to study the integration of *explicit formulations* of other facial variations such as lighting and pose, with this blur-invariant.

3.5 Appendix: Robustness of the blur invariant - An analysis

Given a dictionary $D(y)$, we are primarily interested in how the column span of $\bar{D}(y)$ (obtained by orthonormalizing the columns of $D(y)$ using an economical SVD) is affected by perturbations. The following analysis is adapted from [175]. Let A be a $m \times n$ matrix, and let $\tilde{A} = A + E$ refer to its perturbed version, where E is the additive noise. Let

$$(U_1\ U_2\ U_3)^H\ A(V_1\ V_2) = \begin{pmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \\ 0 & 0 \end{pmatrix}$$

denote the singular value decomposition of A, in which the singular values are not necessarily in the descending order. The singular subspaces we will bound are the column spaces of U_1 and V_1. The perturbed subspaces will be the column spaces of $\tilde{U_1}$ and $\tilde{V_1}$ in the decomposition

$$(\tilde{U_1}\ \tilde{U_2}\ \tilde{U_3})^H\ \tilde{A}(\tilde{V_1}\ \tilde{V_2}) = \begin{pmatrix} \tilde{\Sigma}_1 & 0 \\ 0 & \tilde{\Sigma}_2 \\ 0 & 0 \end{pmatrix}$$

Let Φ be the matrix of canonical (principal) angles between the column space of U_1 and $\tilde{U_1}$, and let Θ be the matrix of canonical angles between the column space of V_1 and $\tilde{V_1}$. [175] now derives bounds on Φ and Θ.

The bounds will not be cast in terms of E, but in terms of the residuals $R = A\tilde{V_1} - \tilde{U_1}\tilde{\Sigma}_1$, and $S = A^H\tilde{U_1} - \tilde{V_1}\tilde{\Sigma}_1$.

Note that if E is zero, then R and S are zero. More generally,

$$\|R\| \leq \|(\tilde{A} - E)\tilde{V_1} - \tilde{U_1}\tilde{\Sigma}_1\| \leq \|E\tilde{V_1}\| \leq \|E\|$$

with a similar bound for S. We now state the following theorem due to Wedin.

Theorem 3.5.1 ***(Wedin)*** *If there is a $\delta > 0$ such that*

$$\min |\sigma(\tilde{\Sigma}_1) - \sigma(\Sigma_2)| \geq \delta \tag{3.10}$$

and

$$\min \sigma(\tilde{\Sigma}_1) \geq \delta, \tag{3.11}$$

then

$$\sqrt{\|\sin\Phi\|_F^2 + \|\sin\Theta\|_F^2} \leq \frac{\sqrt{\|R\|_F^2 + \|S\|_F^2}}{\delta} \tag{3.12}$$

where $\|.\|_F$ is the Frobenius norm, and σ refer to singular values. Since δ is unknown, we empirically evaluated the robustness of the invariant (which is primarily the column span of the left singular vectors, obtained from an economical singular value decomposition) to additive noise, as well as other facial variations.

Chapter 4

Comparing and Combining Lighting Insensitive Approaches for Face Recognition

There are many algorithms in the literature that address the problem of lighting insensitive 2D face recognition. This is a challenging problem because lighting conditions drastically affects the appearance of a face. In this chapter, we attempt to address this problem by understanding the relative merits of different lighting insensitive representations. We make two main contributions. First, we compare a number of algorithms (both class-based, and class-independent) from the perspective of how well they capture different properties of the human face such as, changes in albedo, and changes in surface normal orientations. After analyzing the relative strengths of these algorithms, we propose effective classifier combination schemes that encode such information to produce better recognition performance.

Relation to Prior Work. There are quite a few works in the literature that provide a comparative study of lighting invariant face recognition algorithms. For instance, Ruiz-del-Solar and Quinteros [159] investigate a set of illumination compensation and normalization approaches in an eigenspace-based face recognition setup. They compare the algorithms based on the modeling stages required, simplicity, speed and recognition rates. France and Nanni [71] compare the recognition rates of a set of image based and 3D model based algorithms, and then propose a

simple fusion algorithm based on the sum rule to highlight the advantage of classifier fusion.

We differ from existing surveys in two aspects. First, we study how robust different representations are, in capturing face properties (such as changes in albedo, and surface normal orientation) under lighting variations. Next, we are specifically interested in performing recognition when there is only one exemplar for each person in the gallery. On top of this, we consider galleries with both homogenous and heterogeneous lighting across different subjects. This setting, though restrictive, applies to many real-life conditions wherein we may have only one sample picture of a person (with arbitrary lighting condition) for recognition. This makes the problem much more challenging. We consider two experimental settings. One (in Section 4.2), when there is no prior training information on the effect of typical lighting changes on faces, where we analyze the performance of five class-independent representations. And the other (in Section 4.3), which provides some training data showing possible lighting conditions, wherein we also include four class-based algorithms in the analysis, since they can use available lighting information to learn to perform classification.

Contributions of this chapter. Given this experimental setup, we make the following three observations to enable better understanding of lighting-insensitive face recognition. First, after reviewing nine algorithms we evaluate in Section 4.1 (spanning both class-based and class-independent approaches), in Section 4.2.1 (and in Section 4.3.1) we compare their performance on the PIE data set [173]. We find that two very simple methods perform the best. Overall a very simple comparison

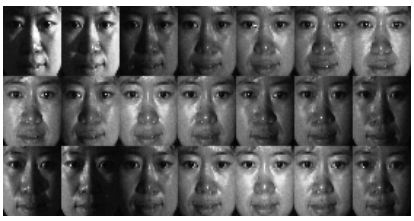


Figure 4.1: Sample images from the CMU-PIE dataset [173]

method using the direction of the image gradient performs better than a number of more recent approaches.

Second, we note that a face contains different sources of information, including albedo changes (e.g., eyebrows), regions of rapid change in surface orientation (e.g., nose) and smooth regions (e.g., cheeks). By looking at individual regions, we can get a better understanding of how well each algorithm makes use of each source of information. In Section 4.2.2 we show experimentally that the relative performance of different class-independent algorithms varies between different regions of the face. To gain some useful intuition, we then consider very simple idealizations of different face regions, and highlight extreme differences of performance for different surfaces.

Finally, these results suggest that we may be able to achieve better performance by combining different representations, benefiting from their different strengths. We show that this is indeed true, demonstrating performance gains with a very simple combination scheme (in Section 4.2.3) that adaptively integrates information from different class-independent representations on the various facial subregions, and then (in Section 4.3.2) by combining information from both class-based and class-independent methods using an SVM that automatically learns the relative importance of these algorithms.

Figure 4.2: Sample images from the extended Yale-B dataset [80]

It will be an interesting topic of future research to determine how best to integrate these representations into recognition algorithms that allow for small changes in pose and facial expression, such as those seen in the recent FRGC data set. However, in this chapter we wish to isolate the effect that lighting change alone has, and to understand this effect thoroughly. For this reason, we experiment using the illumination portion of the CMU PIE data set [173] (shown in Fig. 4.1), and the Extended Yale-B dataset [80] (shown in Fig. 4.2), which controls other sources of variation. We then evaluate the scalability of these representations on images with more controlled lighting conditions, but with other image variations, using the ORL face database [163]. In addition to a standard experimental set-up, in which all gallery images are created with identical lighting, we also consider the more challenging case, in which every gallery image is produced by randomly chosen lighting condition. This simulates some of the challenges of real-world data sets.

4.1 Description of Algorithms

We compared nine algorithms, including both class based approaches and class independent approaches, in our experiments. Although this set of algorithms is certainly not exhaustive, it does give a good sample of different approaches to lighting insensitive face recognition. A brief description of these approaches is given below.

Eigenfaces [184] is a standard benchmark for face recognition. It projects face images into a low-dimensional linear subspace found using principal component analysis. Although not especially well suited to handling lighting variation, it provides a useful point of comparison.

The **Fisherfaces** algorithm [18] (see also [62]) projects images into a direction that not only separates different classes, but also minimizes the within-class scatter. This was explicitly proposed as an effective way to capture variations due to lighting.

Bayesian Face Recognition [128] models variations between images from the same or different individuals using mixtures of Gaussians. The similarity measure is computed based on the maximum-a-posteriori rule, as opposed to the Euclidean norm. In principle, it can model changes due to lighting.

Correlation filters [102] introduce the use of spatial frequency domain methods for lighting insensitive face recognition. A separate filter is trained for every subject (using their 2D Fourier transform representation) such that it produces sharp correlation peaks for the images belonging to that subject, and low values otherwise.

Instead of modeling the illumination variations using face-specific information (as in [102]), the **Image Preprocessing algorithm** [83] estimates the luminance

map present in the image in order to compensate for it, and thereby produces the reflectance map that contains the true information about the facial features of the subjects. This preprocessed image can then be fed into any classifier. We used Eigenfaces [184] to perform classification, as suggested in [83].

Along similar lines, the **Self quotient** image [193] estimates the reflectance of the image by convolving the image with a smoothing kernel and then dividing the original image by the smoothed image (which mostly contains the low frequency components that correspond to illumination effects), and has shown very good performance on the PIE data. In this work we used a much simpler isotropic smoothing instead of anisotropic smoothing (as suggested in [193]). In this form, the algorithm amounts to smoothing the image with a Gaussian, and then pixel-wise dividing the original image by the smoothed image. We obtained the same results given by the authors for the original algorithm, but the results could be different on other datasets.

Another algorithm that displays insensitivity to illumination is the **Eigenphases** [164] method. This algorithm uses the phase information from the frequency domain representation of the image for classification. It is known that the phase information retains most of the intelligibility of the image when compared to the magnitude information of the spectral components, and the authors demonstrate this for the task of face recognition.

The **Whitening** approach described in [139] is specialized for smooth regions wherein the albedo and the surface normal of the neighboring pixels are highly correlated. This means that the pixel independence assumption made implicitly in

computing the sum of squared distances (SSD) is not optimal. This algorithm tries to increase the dissimilarity between the images of different objects by decorrelating the image intensities by applying a Whitening operator. We use the simple Laplacian of Gaussian operator for whitening, as suggested in [139].

Finally, classification based on the **Gradient direction** of the images has also been shown to work well on surfaces, including faces, having properties that change much more rapidly in one direction than in the other (eg., [139], [40] reviews many papers that use this method, going back to the early 1990s). We implemented this method by computing the SSD between the gradient directions in two images. There are other methods that perform well for lighting invariant recognition such as, Gabor Jets [103] and Normalized correlation [92] using small windows. However these two methods have been shown to be quite similar to gradient direction in [139] and hence they are not included in our experiments

4.2 Setting 1: No training set (on the possible lighting conditions)

In this section, we analyze the performance of the algorithms in the absence of any prior information on the lighting conditions present in the scene. Under such conditions, since the class-specific algorithms do not have sufficient exemplars to learn the lighting variations present in the scene, we consider only the class-independent representations. We then divide the face into several regions to study the relative performance of these algorithms. We provide intuitive explanations for the variations in their performance, and then use this information to design an

effective classifier combination algorithm.

4.2.1 Initial Comparisons

We compare the five class-independent algorithms using a standard experimental protocol for PIE data. Each image in this dataset contains one of 68 individuals viewed from the frontal pose and illuminated by a point source of light from one of 21 different directions, without the ambient lighting conditions as shown in Fig 4.2. For all the experiments we used properly cropped faces (by removing the scene background present in the dataset images and retaining only the facial region).

In many applications we do not have access to multiple images of a person with the same pose under different illumination conditions. Hence algorithms that perform well with a minimum number of images of a person are normally preferred. Therefore for all the 68 subjects, we use one illumination condition as the gallery (which contains sample images of the subjects) and the remaining 20 illumination conditions as the probes (which will be compared with all the images in gallery). This is a standard set-up, adopted in many previous papers.

The result of this experiment is given in Fig. 4.3, which shows recognition rates when each lighting condition is used as the gallery. It can be observed that whitening [139] performs much worse than the other class-independent algorithms. But we retain [139] for the experiments involving sub-regions of the face because it is supposed to perform better in smooth regions.

One difficulty with these results is that the performance of the best algorithms

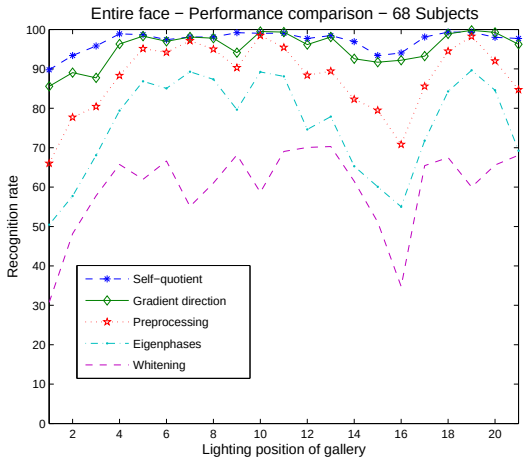


Figure 4.3: Performance comparison of all the class-independent algorithms on the entire face (without training).

is perfect in many cases, making it difficult to distinguish among them. To address this, we also consider a much more challenging recognition task, in which each individual's gallery image is randomly chosen. This makes recognition much more difficult, since it is likely that faces of different individuals taken under similar lighting will appear to be more similar than faces of the same individual taken under very different lighting conditions. However, this difficulty reflects the challenges of many real-world problems, such as sorting personal photos, in which gallery images taken under controlled conditions are not available. Results, averaged over twenty different trials, are given in the Table 4.1.

Overall, the self quotient and gradient direction produce the best performance with a homogenous gallery, with gradient direction performing much better with a gallery formed from heterogeneous lighting. This is rather surprising, since gradient direction is very simple, and is the earliest of these approaches. These results suggest

Table 4.1: Performance of algorithms on homogenous gallery and heterogeneous gallery

Algorithm	Homogenous gallery	Heterogeneous gallery
Self Quotient	97	64
Gradient Direction	95	78
Preprocessing	88	66
Eigenphases	74	60
Whitening	60	40

that gradient direction would be an appropriate benchmark algorithm when new methods are proposed.

4.2.2 Facial Sub-regions

Next, we explore the performance of these algorithms in more detail. As noted, the face provides different sorts of information due to variations in albedo and shape. To get an idea of how different algorithms make use of this information, we coarsely divided the face into seven regions (Fig. 4.4): eyes, nose, lips, two cheek regions and two chin regions. We experiment with the algorithms in these regions, and then provide simple models, to gain a better intuitive understanding of the results.

Some existing works on studying the contributions of different face regions include Nanni and Maio[132], and Nanni and Lumini[131]. In [132], features are extracted from different sub-windows of a face using a bank of Gabor filters and

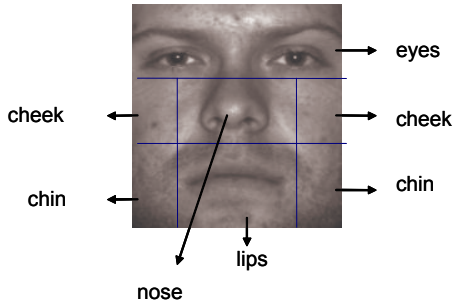


Figure 4.4: Face Sub-regions

Karhunen-Loeve transform. The features obtained by each pattern are used to train a Parzen window classifier to perform face recognition. On the other hand, [131] combines wavelet coefficients from selected sub-bands of several wavelet families and performs face authentication.

3.2.1 Experiments

For each region, both the gallery and the probe contain the same facial features cropped from the face. For all 68 subjects, one illumination condition was used as the gallery and the remaining lighting conditions were used as probes.

The results of recognition experiments on different facial features are provided in Fig. 4.5. We show results for all 68 individuals using gallery images that contain the same lighting. This avoids the need to average over random trials, and still provides sufficient difficulty to evaluate the methods without a ceiling effect, because recognition using a single face region is quite difficult. We performed recognition as described in the last section, but using isolated facial regions.

We find that the relative performances of the different algorithms vary in the

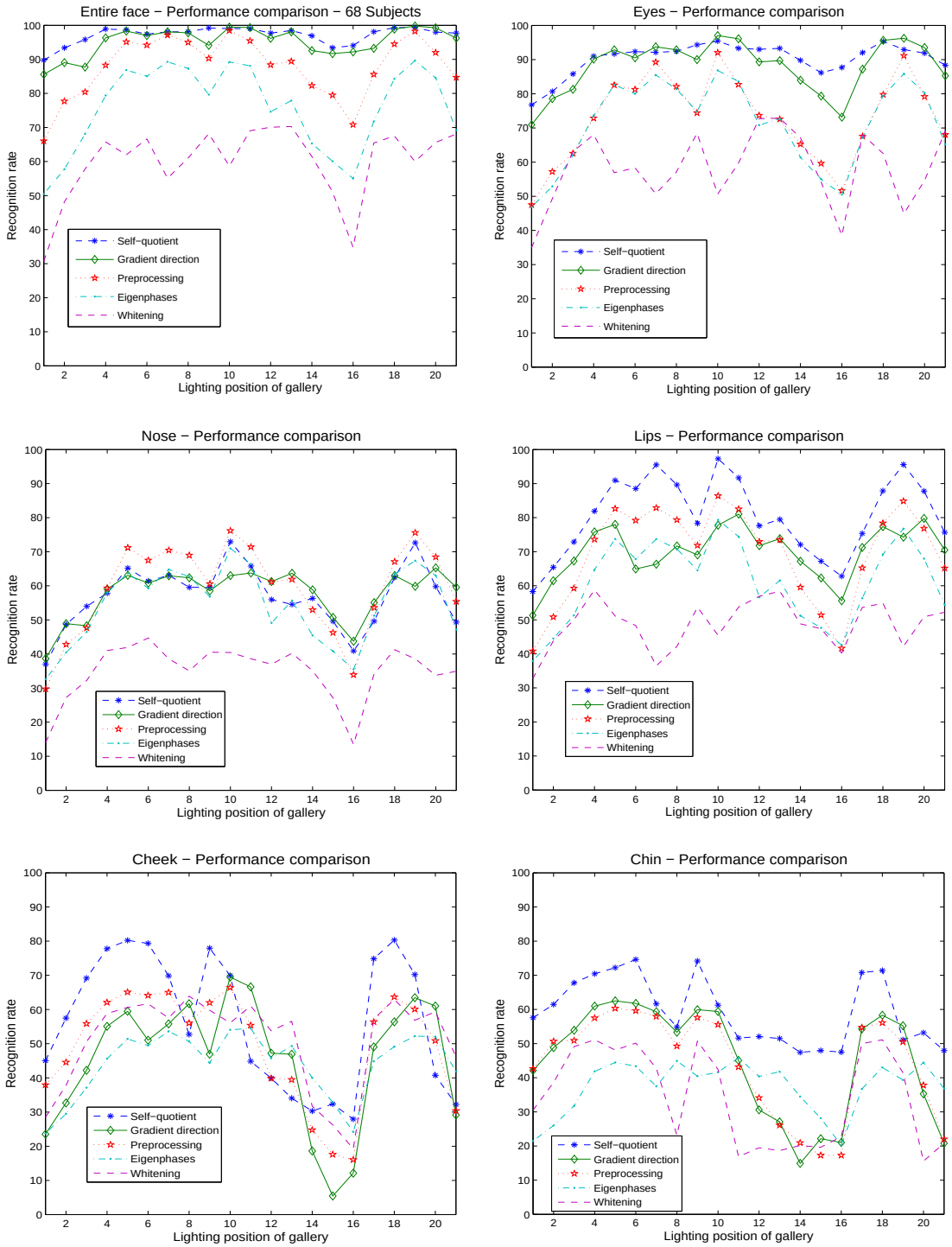


Figure 4.5: Performance comparison of class-independent algorithms on different regions without training

different facial sub-regions. Some of the most noticeable effects are: the Self quotient image algorithm ([193]) performs the best in all regions except for the nose region; Gradient Direction performs well everywhere except for the cheek region; Whitening ([139]) performs poorly, but relatively better in the cheek region.

3.2.2 Analysis of Simple Models

To analyze these results, we model the effects of lighting variation on three simple types of scenes. These are related to important facial characteristics. We make the following observations. First, the face contains albedo variations, especially in regions surrounding the eyes, eyebrows and lips. Second, we consider regions of very high curvature or discontinuity in surface normals, especially at the nose. Finally, the remainder of the face contains regions of smooth variation in shape with little change in albedo. We model these three types of regions with very simple, synthetic models, for which it is easier to understand algorithm performance. We do not expect results with these simple models to perfectly match experiments on faces, since any one face region contains a mix of all three effects. However, we do see that our models explain some of the general trends of our experiments.

4.2.2.1 Planar models with albedo variations.

Through this model, we would like to characterize planar objects that exhibit very large variations in albedo. Towards this end, we create images containing an outer rectangular box of fixed size and an inner rectangular box of variable size

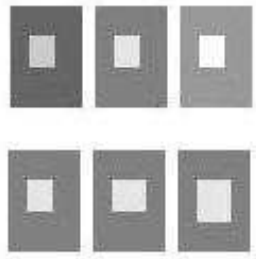


Figure 4.6: Rectangle model. Top: Variation in illumination. Bottom: Variation in albedo

(Fig. 4.6).

This representation has some degree of correlation with the eye region of the face, which has large variations in albedo, due to the eye and eyebrow, while it has much smaller variation in shape. Specifically, the inner rectangle can be related to the human eye while the outer rectangle corresponds to the region surrounding the eyes. We assume that the rectangular surface is Lambertian and that the point light source is at a far distance from the object. The illumination conditions of the two rectangles are varied by changing the position of the point light source. To capture variations between individuals, the position and size of the inner rectangle are changed by small amounts for all possible illumination conditions. 90 different illumination conditions were generated for 400 possible positions and sizes of the inner rectangle. Based on this synthetic dataset, the following results were obtained (Table 4.2). A recognition setup, like the one discussed in 4.2.2, was adopted.

As in the case of the human face, self quotient and gradient direction based methods perform very well in these synthetic conditions. The gradient direction method works very well due to the presence of rich information of the gradient

Table 4.2: Performance of algorithms on the rectangle model

Algorithm	Recognition rate (Rectangle model)	Recognition rate (Human eyes)
Self Quotient	100	90.3
Gradient Direction	100	88
Preprocessing	49.4	73
Eigenphases	49.4	71.4
Whitening	17.6	58.2

angle change in the boundary between the two rectangles.

The self quotient image algorithm works well because there is no change in the surface normal and there is a sizable change in the albedo. In these conditions, [193] points out that self quotient is invariant to lighting changes. This algorithm is shown to capture the albedo changes very well. Whitening does badly as the albedo is not smooth, and is not whitened by the filter we use.

4.2.2.2 Shape variations in smooth objects.

In this model, we attempt to simulate the case wherein the object is predominantly smooth, with gradual variations in its shape. Such a variation can be captured by a small piece of a smooth cylinder (Fig. 4.7). We construct this model by considering cylinders of different radii (accounting for the different subjects) and varying the position of the point light source for each cylinder. This representation

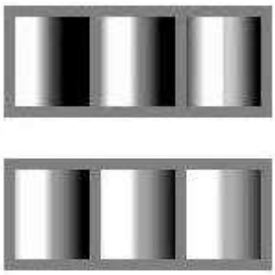


Figure 4.7: Cylinder model. Top: Variation in illumination. Bottom: Variation in curvature (see that as the curvature increases from left image to right image in the bottom row, the change in the lighting pattern gets slower

correlates with the human cheeks where different human cheeks vary in curvature, without discontinuities in shape or much variation in albedo. Again we assume that the cylindrical surface is Lambertian and that the point light source is distant from the object.

The dataset contains cylinders of 11 different radii with 9 illumination conditions and the results are given in Table 4.3. We see that the gradient direction based method performs very poorly, matching the fact that it is also the least effective method on human cheeks. Even though the gradient direction is invariant to lighting for a cylinder, there is no variation in direction of gradient between subjects, while the gradient direction does not capture the changes in curvature. The self quotient algorithm works well because the Gaussian kernel which is used to filter the image attenuates different frequencies in different ways. The intensity is basically a sine wave, and when we divide it by the smoothed sine, we get a constant function whose magnitude encodes the cylinder's curvature. The inten-

Table 4.3: Performance of algorithms on the cylinder model

Algorithm	Recognition rate (Cylinder model)	Recognition rate (Human cheek)
Self Quotient	100	56.5
Gradient Direction	9.1	41.5
Preprocessing	100	49.2
Eigenphases	100	44.1
Whitening	100	50.8

sity of the resulting representation therefore captures the dominant frequency of the initial image. The algorithm uses this criterion to classify these images and is thereby invariant to changes in illumination. Whitening's good recognition rates are in line with the prediction in [139] that it will perform well on smooth surfaces. Eigenphases performs well because the phase spectrum of the signal will be a function of the frequency information present in the signal. This frequency information helps this algorithm to classify the query images properly and thereby give good recognition rates.

4.2.2.3 Shape variations in objects with discontinuities

Through this model, we capture the variations in the shape of an object that has some discontinuities. The motivation behind this model is to obtain an approximate representation of the human nose, which can be modeled as a prism. We

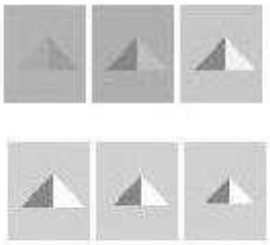


Figure 4.8: Triangle model. Top: Variation in illumination. Bottom: Variation in shape

consider the two sides visible from the frontal view of a prism to represent the nose (Fig. 4.8). This model, however, does not exactly represent the nose because, we don't consider the effects caused by the nostrils. A human nose may be a combination of our simple prism model, and a model of albedo variation, such as our eye model.

The shape of this pyramidal surface is changed to represent different individuals and the position of the light source is moved to create different lighting conditions. The experiment consisted of 12 subjects with 10 illumination conditions each and the results are given in Table 4.4.

The self quotient image does not perform well due to the change in the orientation of the surface normal between the different regions in the triangular model. Lighting variations can change the ratio of the intensity in two regions of the prism, and the self quotient cannot undo this. Thus we find the self quotient algorithm to be not very effective in capturing shape variations, as predicted in [193]. This matches the fact that the nose is the only region in which the self quotient is not the

Table 4.4: Performance of algorithms on the triangle model

Algorithm	Recognition rate (Triangle model)	Recognition rate (Human nose)
Self Quotient	42.7	57
Gradient Direction	100	57.6
Preprocessing	57.8	59.1
Eigenphases	46.7	54.3
Whitening	28.7	34.8

best. Gradient direction works well because it captures the variation in the shape of the triangles. Whitening does not perform well due to the absence of smooth variations in the surface. The preprocessing algorithm is formulated in such a way that, it controls the illumination variations both in regions where the luminance changes smoothly and in regions where there are discontinuities. So, this algorithm performs relatively well in all the regions.

Our results are related to, but also differ somewhat from the discussion in [40] and [139]. They point out that representations related to the direction of the gradient are insensitive to lighting variation for surfaces that change rapidly in shape or albedo in one direction but not another, while whitening approaches are better suited for smooth surfaces that vary slowly in both directions. First, we show that performance of the algorithm can vary depending on whether variations occur in shape or in albedo. Second, we show with our cylinder example that

Table 4.5: Performance comparison of combined classifier with the best individual algorithms

Region	Recognition rate of the combined classifier	Recognition rate of the best individual algorithm
Entire face	99.1	97.1
Eyes	95.7	90.3
Lips	83.3	80.6
Nose	69.3	59.1
Chin	64.9	59.5
Cheek	62.8	56.5

variations within a class must also be considered when determining the effectiveness of a representation. In some cases, the gradient direction may not discriminate within a class, while features such as curvature do.

4.2.3 Classifier combination

The fact that different approaches perform well on different parts of the face suggests that we can improve the overall performance by combining some or all of the methods. To demonstrate this, we experimented with a simple method for combining representations.

First, the outputs of the two top performing algorithms for every feature (including the entire face) are combined by normalizing the SSD for every gallery-probe

combination and then adding the normalized results of the top two algorithms. For example, self quotient image and gradient direction algorithms were combined for the entire face and eyes, self quotient Image and preprocessing algorithms were combined for the nose region and so on. We show the results of adaptively integrating different representations on various facial regions in Tab1e 4.5 and Fig. 4.9, for the task of recognition with a homogenous gallery.

It can be seen that our classifier combination algorithm results in a substantial improvement in situations, like the nose, where the best individual algorithm doesn't perform that well. For regions such as the entire face, the performance improvement is only moderate since the best individual algorithm by itself has recognition rates close to the ceiling. These results, in effect, drive home the point that an effective classifier combination algorithm should take into account the relative strengths of the individual classifiers in capturing different characteristics of the object of interest.

With this encouraging result in hand, we would like to formulate a combination algorithm that automatically learns the relative strengths of the individual algorithms, rather than having a user specifying which algorithms to combine based on observation. Towards this end, in Section 4.3, we consider the setup of having some prior information on the different lighting conditions present in the scene such that one can get a feel of the relative performance of different classifiers and thereby learn the ideal combination strategy before testing it out on the subjects of interest. Since we have a representative training set, we now include the four class-based algorithms (discussed in Section 4.1) into our analysis.

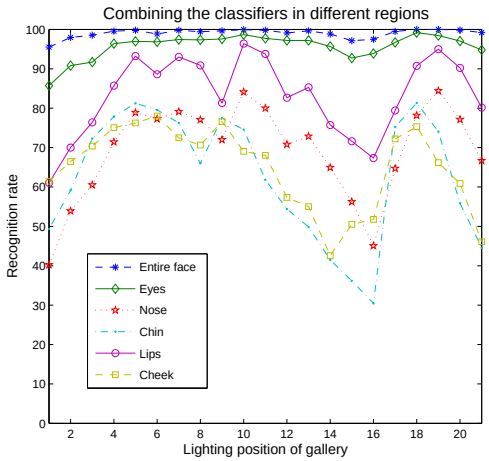


Figure 4.9: Performance comparison of combined classifier on all facial regions

4.3 Setting 2: With prior training on different lighting conditions

In this section we first analyze the performance of different algorithms in the presence of training data that contains representative lighting conditions present in the scene. We then combine the most informative algorithms using a support vector machine (SVM) framework, which learns the combination parameters automatically.

4.3.1 Initial Comparisons

We compare the five class-independent algorithms, along with the four class-based algorithms on the PIE dataset. We use all 21 illumination conditions of the first 34 subjects for training. The algorithms were then tested on the remaining 34 subjects, with one homogenous exemplar lighting condition (for all the subjects) in the gallery. This test is done mainly to analyze the effect of training on the different class-based algorithms. The class-independent algorithms, of course, were tested directly on the second half of the 34 subjects.

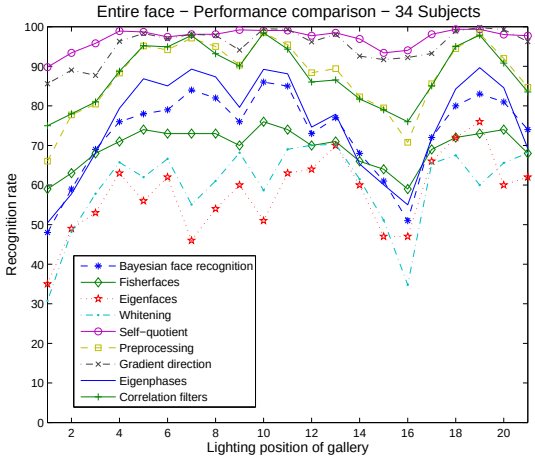


Figure 4.10: Performance comparison of class-based, and class-independent algorithms on entire face. The same gallery lighting condition was used for all subjects

The result of this experiment is given in Figure 4.10, which shows the recognition rates when each lighting condition was used as the gallery. Similar to the experiments conducted without training data (in Section 4.2.1), the algorithms based on the self quotient image [193], and the direction of image gradients [40] perform the best. Yet another observation is that the three class-based methods (Fisherfaces [18], Bayesian face recognition [128], Eigenfaces [184]), and whitening ([139]) perform worse than other algorithms. So, we exclude [18], [128], and [184] from the experiments for classifier combination. However, we retain [139] since it adds considerable value in the cheek region (as shown in Section 4.2.2). For the correlation filters algorithm [102] (and for the class-based algorithms in general), we do not perform the analysis on different facial sub-regions because it is a learning algorithm, and it is difficult to give intuitive explanation of its performance variations (if any) on different facial regions. Now that we have representative algorithms from both class-based and class-independent streams, we discuss our proposed combina-

tion strategy in the next sub-section. Specifically, we consider the recognition setup wherein the gallery and the probes have heterogeneous lighting conditions, in order to overcome the ceiling effect in the recognition rates of certain algorithms, and also because this setup simulates a more representative real-world setting.

4.3.2 Classifier combination

Along similar lines with the discussion in Section 4.2.3, we expect that we can achieve better performance by using learning to determine the best way of combining information. We do this by training a support vector machine (SVM) [39] to perform a verification task, as done previously by [147], for instance. Given a pair of images, the SVM is trained to determine whether they come from the same or different individuals. The radial basis function (RBF) kernel was used to map the inputs to a higher dimensional space. The SVM was trained using intra-personal pairs and extra-personal pairs from the first 34 subjects of the PIE dataset, and tested with randomly generated pairs from the remaining subjects. The lighting conditions used for training and testing were also disjoint. The input to the SVM is the (absolute) difference between the two images after processing them to create six different representations based on gradient direction, self quotient, eigenphases, whitening, image preprocessing and correlation filters. We contrast the performance of an SVM that uses all six representations with six SVMs that each use just one of the representations. The authors of [116] have used a similar approach, training an SVM with just differences in gradient direction.

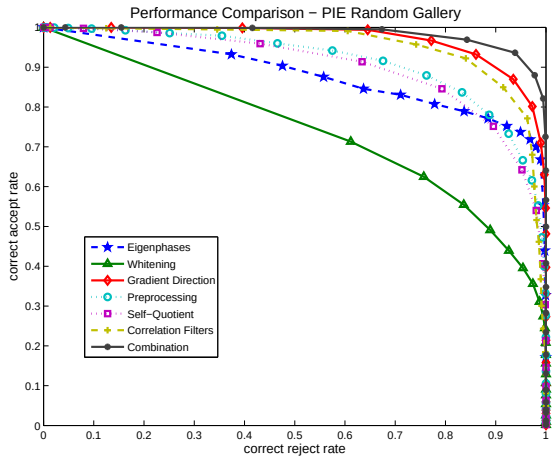


Figure 4.11: CAR-CRR curves for PIE heterogeneous gallery experiments

The result of the SVM combination is given in Fig. 4.11 in the form of Correct Accept Rate (CAR) vs Correct Reject Rate (CRR) curves; It can be seen that the combination results in a good improvement in verification accuracy. For example, the combined method has an Equal Error rate of 7 percent, compared to 10 percent for the best individual algorithm (using the gradient direction). In order to check the generalizability of these results, we experimented with the extended Yale-B dataset [80]. This dataset has cropped faces of 38 subjects under 64 different lighting conditions. All the other variations such as pose, and expressions are fixed. We then performed a similar verification experiment, by training the SVM using the lighting conditions corresponding to the first 18 individuals, and tested it using the pair-wise differences obtained from the remaining 20 subjects. The CAR-CRR curves for the SVM combination, as well as the individual algorithms are given in figure 4.12. Once again, the representation based on the direction of image gradient is the best individual algorithm, followed by the correlation filters. The proposed classifier combination algorithm again results in a substantial improvement in performance

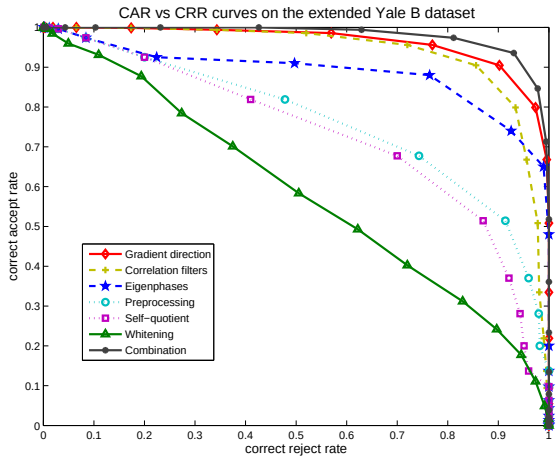


Figure 4.12: CAR-CRR curves for the extended Yale-B heterogeneous gallery experiments

over the individual algorithms. These results again reinforce our observation that, combining different representations by learning their relative strengths is crucial to obtain good performance improvement.

4.3.3 Experiments on faces with more controlled lighting

Our main focus is on understanding the role of different representations when lighting changes. However, it is also important to determine the relative sensitivity of these methods to other image variations. If a representation is insensitive to lighting variation, but highly sensitive to changes in expression, for example, it may be less useful in a general face recognition system.

To evaluate this we experimented using the ORL dataset [163], which has large variations in pose and expression, but small variations in lighting. Fisher discriminant analysis (LDA) [18] has been shown to be efffective on this data ([72]),

Figure 4.13: Sample images from ORL face database [163]

helping to compensate for the correspondence problem that this data gives rise to. Therefore we use LDA as the base method, in combination with lighting insensitive representations. Due to the challenges of this data, we adopt the widely used leave-one-out testing protocol ([206], [72]).

Fisher discriminant analysis [18] was performed on the training images using the six different lighting insensitive representations used in Section 4.3.2. An optimal set of parameters was determined for each representation to learn the inter-class and intra-class variations. For the combined classifier, learning was done by concatenating all the six representations. The test images were then projected onto the learned subspace to perform recognition. This setup was repeated ten times, by taking one of the possible ten images per person in the test set. The average recognition rate over these ten trials are reported in Table 4.6 below. We also compare our results with the previously reported results on this dataset, from [206], and [72], which primarily uses the intensity image of the face to learn the classifier.

Table 4.6: Recognition rates on the ORL face database [163]

Algorithm	Recognition rates
Gradient Direction [40]	95.75
Eigenphases [164]	90.75
Preprocessing [83]	81.75
Self Quotient [193]	80
Whitening [139]	94.25
Correlation filters [102]	96.25
Combination	98.5
Fisherfaces [206]	98.5
ICA [206]	93.8
Eigenfaces [206]	97.5
Kernel Eigenfaces [206]	98
2DPCA [72]	98.3

It can be seen that some of the lighting invariant representations, like gradient direction [40] perform well under general imaging conditions, and the combined representation does provide improvements in the recognition rate. At the same time it seems that the self quotient image [193] is particularly sensitive to non-lighting variations. But we would like to make a point here regarding the amount of training data used. All our previous experiments (until Section 4.3.2) were done with just single image per person in the training set. Our study mainly focuses on how lighting invariant a representation can be, given it sees just one image of the person in arbitrary illumination. But when there are other sources of variations in the dataset, such as expression, registration, scale and pose, we need to have

multiple images of a person in the training set in order to learn a good classifier. It is an interesting future work to design classifiers capturing the contributions of different representations, to perform robust face recognition (with very few training examples) under multiple sources of variations.

4.4 Discussion

Besides the performance gains obtained through classifier combination, another interesting observation of this work is that a single classifier based on the direction of image gradient works very well. Throughout the experiments discussed here, the gradient direction algorithm clearly performs the best with just one exemplar per person in the gallery (with both homogenous and heterogeneous gallery lighting conditions). In order to emphasize the significance of this observation, we compare our results with two recently reported algorithms from the literature.

4.4.1 Comparison with the work of Tan and Triggs [177]

First we consider the work by Tan and Triggs [177], which proposes enhanced local texture feature sets for illumination robust face recognition. They introduce Local Ternary Patterns (LTP), a generalization of the Local Binary Pattern (LBP) texture descriptor [3], and show it to be more discriminant and less sensitive to noise. They then couple this descriptor with a preprocessing step that compensates for lighting, and use a distance transform based similarity metric to obtain good recognition results. We now compare the results of gradient direction with those

reported by [177].

For the experiments on the extended YaleB dataset [80], the frontal face images with most neutral lighting sources ('A+000E+00') were used as the gallery. The probe was divided into five subsets, according to the angle between the light source direction and the central camera axis (12°, 25°, 50°, 77°, 90°), containing frontal images of all 38 subjects. The results obtained by the Tan and Triggs algorithm [177], and by using gradient direction based classifier (with l_1-Norm as the distance measure) [40] are given in Table 4.7.

Table 4.7: Comparing the overall recognition rates of Tan and Triggs algorithm [177] with that of Gradient direction algorithm [40] on the Extended YaleB dataset [80]

Algorithm	Subset # (Number of probes)				
	1 (263)	2 (456)	3 (455)	4 (526)	5 (714)
Tan and Triggs [177]	100%	100%	100%	99.2%	97.2%
Gradient direction [40]	100%	100%	100%	100%	99.73%

It can be seen that we obtain slightly better results using the gradient direction algorithm [40]. We then compare our results on the PIE dataset [173], wherein again, images of all 68 subjects with neutral lighting sources were used as gallery, and the remaining images were used as the probe. In this setup, we obtain the maximum possible recognition rates like [177], as shown in Table 4.8. Through these experiments, we observe that a simple classifier based on the image gradient

orientation offers similar (and in some cases, better) recognition performance.

Table 4.8: Comparing the overall recognition rates of Tan and Triggs algorithm [177] with that of Gradient direction algorithm [40] on PIE dataset [173]

Algorithm	Recognition rates
Tan and Triggs [177]	100
Gradient direction [40]	100

4.4.2 Comparison with the algorithm for face recognition using Sparse representations [200]

Next, we consider a more recent work by Wright et al [200], using the theory of sparse representations for face recognition. The main motivation behind this work is to represent a test face image as a sparse combination of the 'most identical' images present in the training set, so that the occlusions present in the test data can be effectively factored out. The authors also illustrate the potential applications of such an approach for handling variations in illumination. They provide results for lighting invariant face recognition on the extended Yale-B dataset [80] by using sparse representation-based classification (SRC) on different sets of features including, Eigenfaces [184], Fisherfaces [18], Laplacianfaces [87], Randomfaces (obtained by performing random projections on the input faces), and downsampled faces. The gallery for their experiments contained half of the available lighting conditions (i.e. 32 per subject), with the lighting chosen randomly for different subjects. The results obtained using their best image representation (E-random faces), with different

dimensions for the face image, is reproduced in Table 4.9 given below.

Table 4.9: Performance of SRC based face recognition algorithm [200] on the extended Yale-B dataset

Dimension of the face image	Recognition rate of E-random faces [200]
30	90.72
56	94.12
120	96.35
504	98.26

We now compare these results with those obtained using the direction of image gradient [40]. We used the l_1-Norm to compute the distance (since it gave better performance than the l_2-Norm, of about 5% improvement in the recognition rate). We varied the number of (random) lighting conditions for every subject in the gallery, and the results averaged over multiple trials are given in Table 4.10. The input image dimensions used for our experiment is 1920 (i.e. 48*40).

Table 4.10: Performance of gradient direction algorithm [40] on the extended Yale-B dataset

# heterogeneous gallery lighting per subject	Recognition rate of gradient direction algorithm [40]
1	59.1
2	75.8
4	93.5
6	98.6

The important result, as we see, from the tables 4.9 and 4.10 is, although the input image dimensions of our experiment is higher than that of [200], the simple classifier based on the direction of image gradients [40] performs better than [200] with just six lighting conditions in the gallery (when compared with 32 in the case of [200]). Overall, the message we would like to convey from the comparisons given in Section 4.4.1, and Section 4.4.2 is that the gradient orientations [40] retain most of the person-specific information even under very challenging lighting conditions, and it is interesting to see how this information can be better utilized in dealing with more challenging face recognition settings.

To conclude, we have compared a number of approaches to illumination insensitive face recognition, both experimentally and using an analysis of simple idealizations of face features. Based on all the results obtained, we make the following observations. 1) Gradient direction works very well under both homogenous gallery and heterogeneous gallery settings. We suggest that it should be a baseline algorithm for future methods, especially since it is so simple to implement. 2) The self quotient image and gradient direction-based algorithms work extremely well under homogenous gallery lighting conditions. 3) Not all the methods that use training data perform better than simpler methods that use general image processing. This suggests that these methods do not get as much out of training as might be possible. An exception to this is the correlation filters algorithm, which offers better recognition rates than most of the class-independent algorithms, but still is not as good as the direction of gradient (even when the training data has a very good representation of different lighting conditions). 4) Different representations work well

in different parts of the face. For example, the self quotient image is less effective in the nose region, while the gradient direction performs poorly in the cheek region. We are able to explain these results using a simple idealization of facial features. 5) Consequently, it is possible to improve performance by combining different representations. We demonstrated this using two classifier combination algorithms. The first algorithm adaptively integrates information from individual classifiers on various facial regions, whereas the other learns the best combination strategy using a SVM framework. It remains an interesting topic for future work to characterize the strengths and weaknesses of these approaches when both lighting and pose or facial expression vary.

Chapter 5

A Learning Approach Towards Detection and Tracking of Lane Markings

Autonomous navigation of road vehicles is a challenging problem that has wide-spread applications in intelligent systems, and robotics. Detection of lane markings assumes importance in this framework since it gives the driver a sense of the road ahead, such as if it is straight or curved, how many lanes are present, and so on. It is a hard problem due to the variations present in: (i) **the appearance of lane markings -** solid lines, dotted lines, circular reflectors and their color (yellow or white); (ii) **the type of road** on which the vehicle is traveling, such as highways and city streets, and objects that can occlude the lane markings, like vehicles and pedestrians; (iii) **the time of day** in which the scene needs to be analyzed; for instance at night the most visible regions are those that are just ahead of the vehicle, whereas during the day all regions in the field of view of the camera need to be analyzed by the detector; and (iv) **the presence of shadows** in the scene due to objects such as trees that might affect the appearance of lane markings. Some examples illustrating these challenges are given in Figure 5.1.

To deal with the above-mentioned conditions, many approaches have been proposed in the literature. These can be broadly classified based on the type of sensors which could be (visual) cameras, internal vehicle state sensors, GPS sensors, laser

Figure 5.1: Sample road scenes; day and night-time images illustrating the effect of lighting variations and occluding vehicles as observed by the visual sensor.

scanners or radar sensors. Each sensor has its own advantages and limitations. For instance, foggy conditions on the road affect the reliability of cameras, whereas a GPS sensor might be more robust. Although using multiple sensors will certainly help the detection process [43], in this work we are primarily interested in analyzing visual inputs from a single camera mounted in front of a moving vehicle. We now review some related work pertaining to this category.

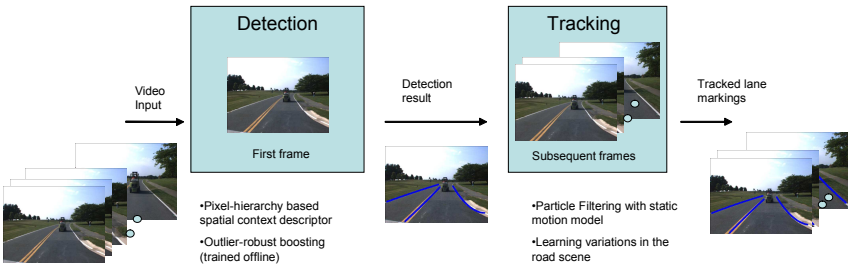


Figure 5.2: Pipeline of the proposed approach: detection with boosting on contextual features, and particle-filter based tracking to learn some road scene variations.

Prior Work: Detecting road lane markings using image analysis has been an area of active research for the last two decades. The recent survey paper by McCall and Trivedi [125] provides a comprehensive summary of existing approaches. Most of the methods propose a three-step process, (i) **extracting features** to initialize

lane markings such as edges [141], texture [152], color [178], and frequency domain features [101]; (ii) **post-processing** the extracted features to remove outliers using techniques like Hough transform [113] and dynamic programming [96], along with computational models explaining the structure of the road using deformable contours [195], and regions with piecewise constant curvatures [133]; and then (iii) **tracking** the detected lane markings in subsequent frames using a Kalman filter [54] or particle filters [10, 99] by assuming motion models (constant velocity or acceleration) for the vehicle. There are also methods that use stereo cameras (e.g. [25, 48]) to enforce similarity of points observed from both cameras. More recently, there has been increased focus on building real-time systems [8] on challenging urban scenarios [31, 41], including night-time driving conditions [30]. Machine learning methods with a single classification boundary such as, neural networks and support vector machines [99] have also been used for detection. However, two main aspects of this problem have not yet been satisfactorily addressed; (i) Since the visual properties of lane markings undergo arbitrarily large variations, using *local features* to describe their appearance, and learning the decision boundary from a *single classifier* may not be robust and scalable; (ii) The assumption of a *pre-specified motion model* for the vehicle breaks down when the vehicle displays a random motion pattern. This is especially critical in the scenario we are interested in where the inputs are obtained only from the visual sensor, without having any inertial information from the vehicle.

Motivated by these challenges, we propose a learning-based approach[1] to de-

[1] An initial version of this work appeared in [81], where we discuss the detection algorithm using

tect lane markings without requiring a pre-defined road model, and track them without the knowledge of vehicle speed. For the two-class object **detection** problem, corresponding to lane markings and non-lane markings, we collect a set of representative training data with which, (i) instead of using *local* features to describe the object in *isolation*, we learn the relationship shared by the object with its surrounding scene. To model this source of information, often referred to as spatial context [56], we propose a *pixel-hierarchy* descriptor in which different visual features such as intensity patterns, texture, and edges are analyzed in an hierarchy of regions surrounding each pixel corresponding to the object; (ii) given a *bag* of contextual features for exemplar pixels corresponding to the two classes, we learn their relevance for decision-making using a machine learning algorithm based on boosting [76] that determines a *final strong* classifier by combining several *weak learners*. In this process, we address the outlier-sensitivity problem of boosting algorithms through methods that jointly optimize the detection error rate and the balance in weight distribution of training exemplars. (iii) Then, we represent the detected lane markings using polynomials, and **track** them in a particle filtering framework [91]. However, since we do not have information about vehicle motion to predict the new position of lane markings, we look at a slightly different problem by assuming the lane markings to be static over the video sequence and then characterize deviations in the tracked model parameters to infer their causes. Assuming the road to be flat, we illustrate this by learning three sources of variations in the road scene, namely;

a boosting variant that learns the weights of training samples before selecting the weak learners (first part of Sec 5.1.3.3).

the change in road geometry (straight or curved), changes in the lateral motion of the vehicle, and the presence of occluding objects in the road ahead.

Contributions: We propose,

- A pixel-hierarchy feature descriptor to model the spatial context information shared by the lane markings and the surrounding scene;
- An outlier-robust boosting algorithm to learn relevant contextual features for detecting lane markings, without assuming any prior road model;
- Learning possible variations in the road scene, by assuming the lane markings remain static through the video, and characterizing the tracked model parameters.

Organization of the chapter: We discuss the detection (localization) of lane markings by extracting contextual features and modeling them with boosting in Section 5.1. Section 5.2 deals with tracking and learning the variations in road scene without any prior knowledge of the vehicle's motion pattern. We then present experimental validation of detection and tracking algorithms in Section 5.3, using data from both daylight and night-time road sequences. Section 5.4 concludes the chapter. A block diagram explaining the flow of the proposed approach is given in Figure 5.2.

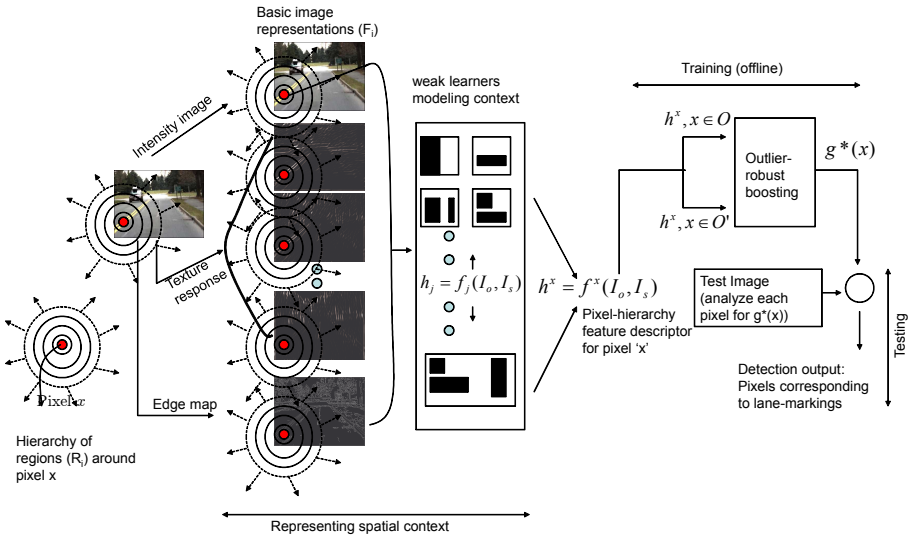


Figure 5.3: L-R: An illustration of computing the pixel hierarchy descriptor h^x for a pixel x on hierarchical circles R_i, with the underlying F_i corresponding to the intensity image, edge map, and texture response (magnitude patterns of texture response are shown in this figure. 4 wavelet scales with 6 Gabor filter orientations were used). The weak learners h_j correspond to Haar filters, which when applied on $R \times F$ result in $h^x = \{h_j^x\}_{j=1}^{M_2} = f^x(I_o, I_s)$. h^x is the pixel-hierarchy descriptor of a pixel x. h^x computed for $x \in O, O'$ are used to train the boosting algorithm to compute the strong classifier $g^\star(x)$. This is then used to classify pixels in a test image corresponding to lane markings and others.

5.1 Detection of Lane Markings

5.1.1 Problem Definition

We are studying a two-class object detection problem, where the goal is to classify lane markings O from non-lane markings O'. In Bayesian terms, the posterior $P(O|I)$ modeling the probability of presence of an object (class) O given an

observation (or measurement) I is given by

$$P(O|I) \propto P(I|O)P(O) \tag{5.1}$$

where $P(I|O)$ denotes the likelihood, and $P(O)$ is the prior for the class O. $P(O'|I) = 1 - P(O|I)$. We take a *data-driven* discriminative approach where the posterior is modeled directly, instead of separately modeling the likelihood and prior (generative). Further, we consider the observations $I = f(I_o, I_s)$, where I_o is the information about the object in isolation, and I_s is the information conveyed by the surrounding scene. Given a set of M training exemplars (pixels) for O, and O' pertaining to different road conditions, the goal of this work is to find functions f and g such that,

$$P(O|I) = g(f_j^i(I_o, I_s)) \tag{5.2}$$

$\forall i = 1 \; to \; M, \forall j = 1 \; to \; M_1$, where $h_j^i = f_j^i(I_o, I_s)$ is one of the M_1 different realizations of spatial context between the object (O or O') and their surrounding regions portrayed for an ith training sample, and g is a feature selection method that optimally computes the relevance of h_j in classifying pixels belonging to O from O'. The details of f and g form the focus of the next two sub-sections.

5.1.2 Modeling the spatial context of lane markings

Context, a loosely defined term in itself, refers to all *pertinent* information conveyed by the visual scene about the existence of an object [56]. Although the complementary information provided by context has been acknowledged since the early 70's [142], only in recent years have we seen its explicit modeling in the main-

stream object detection literature [180]. Since, (i) the lane markings share a rich neighborhood portrayed by the road regions, and (ii) existing methods, at large, characterize the appearance of lane markings *in isolation*, understanding the role of context for this problem attains prominence.

Classification of low-level contextual sources, which do not use any higher-level semantic information on the *structured grouping* of pixels representing different objects, belongs to one of the following categories; (i) top-down methods that compute the *gist* of the scene by computing some global image statistics, e.g. [180], and (ii) bottom-up methods that correlate the properties of a pixel with its *immediate* adjoining region, e.g. [47]. Both have relative advantages/disadvantages depending on the application of interest. In this work, we propose a hierarchical descriptor that encodes information of both types.

5.1.2.1 A pixel-hierarchy feature descriptor

Given a pixel x corresponding to O or O', we consider a hierarchy of regions represented by concentric circles centered at that pixel. Let $R = \{R_i\}_{i=1}^{M_2}$ represent the regions enclosed by circles of increasing radius. We now model the visual information present in R. The exact definition of 'information' depends on the application, and for this problem, motivated by the existing work, we use the intensity image, an edge map output from the Canny operator [36], and texture patterns (obtained from Gabor filters [109] that compute the magnitude and dominant direction of textures at different scales). Let us refer to them as $F = \{F_i\}_{i=1}^{M_3}$. From this, we

compute our basic contextual features

$$h_j^x = f_j^x(I_o, I_s) : R \times F \rightarrow \mathbb{R} \tag{5.3}$$

by analyzing the pattern of F on different regions R. We used a set of rectangular Haar-like filters [145] for this purpose. These filters have different positive and negative regions, where the positive regions replicate the values of the underlying region $R \times F$, and the negative regions correspond to a value of zero. The values underneath the positive regions (for each Haar filter) are then accumulated to result in h_j^x, and the set of all such features, $h^x = \{h_j^x\}_{j=1}^{M_1}$, denotes the pixel-hierarchy feature descriptor of a pixel x. By this way, h_x jointly models I_o by extracting F in the immediate neighborhood of a pixel x, and I_s by describing F in regions belonging to larger concentric circles around x. Typically, the number of features M_1 is in the order of 1000's depending on the precision with which the pattern of positive and negative rectangular regions are varied. An illustration is given in Figure 5.3.

5.1.3 Learning the relevant contextual features through Boosting - Training the classifier

We now require a principled way of selecting relevant features among h (or $f(I_o, I_s)$) that are most discriminative is classifying pixels x corresponding to O from O'. This requirement paves the way to adapt the principles of boosting [76], a machine learning method that determines the optimal set of features to classify objects with provable detection error bounds. Boosting algorithms have been used previously for two-class problems by Viola and Jones [189] for detecting faces, and

Wu and Nevatia [202] for human detection, where the weak learners h modeled the appearance information of objects *in isolation*.

5.1.3.1 The Problem of Outliers in Training Set

Before getting into the details of detecting lane markings using boosting, we address an important problem in the learning stage of boosting algorithms: that of the presence of outliers in the training set. To make the chapter self-contained, we present the basic version of the Adaboost algorithm [76], referred as Discrete Adaboost, below.

Given: $(x_1, y_1), ..., (x_M, y_M)$ where $x_i \in \mathbb{R}^N$ denote the feature representation of an object x_i, and $y_i \in \{-1, +1\}$ denoting its class label, and classifier pool $\mathbb{H}$ consisting of weak learners h_i, $i = 1, ..., M_1$.

Initialize the training samples with uniform weights $D_1(i) = 1/M, \forall \{x_i\}_{i=1}^{M}$.

For iterations t=1,....,T:

- Train the base learners using the weight distribution D_t.
- Get the base classifier $h_t : \mathbb{R}^N \rightarrow \mathbb{R}$, which minimizes the error

$$\epsilon_t^\star = P_{i \sim D_t}[h_t^{x_i} \neq y_i] = \sum_{i=1}^{M} D_t(i) I(y_i \neq h_t^{x_i}) \tag{5.4}$$

  where $I(.)$ is an indicator function.

- Choose $\alpha_t \in \mathbb{R}$, which is a function of the classification accuracy.
- Update:

$$D_{t+1}(i) = \frac{D_t(i) exp(-\alpha_t y_i h_t^{x_i})}{Z_t} \tag{5.5}$$

where Z_t is a normalization factor (chosen such that D_{t+1} will be a distribution).

Output the final classifier:

$$g^{\star}(x) = sign\Big(\sum_{t=1}^{T} \alpha_t h_t^x\Big) \tag{5.6}$$

In summary, given a set of labeled training samples belonging to two classes with the *same initial weights*, and a pool of weak learners, the goal of boosting is to find a sequence of best weak classifiers by adaptively increasing (and decreasing) the weights of wrongly (and correctly) classified samples in each stage. However, when there are outliers present in the training set, say due to mislabeled samples or due to samples that are very different from other neighbors of their class, this process will result in a substantial increase in their weights and thereby force the weak learners to concentrate much more on these samples. This might end up being detrimental to the performance of Adaboost, as demonstrated convincingly by [55].

5.1.3.2 Related work

There is a considerable amount of work addressing this issue. For instance, [77] suggested a variant called 'Gentle Adaboost' by interpreting boosting as an approximation to additive modeling on the logistical scale. [153] showed how to regularize Adaboost to handle noisy data: instead of achieving a hard margin distribution (by concentrating on a few hard to be classified samples, like that of Support Vectors), they propose several regularization methods to achieve a soft margin that reduces the effect of outliers. [74] suggested an algorithm called 'BrownBoost' that takes

a more radical approach by de-emphasizing outliers when it seems clear that they are 'too hard' to be classified correctly. Other approaches include LPBoost [53], WeightBoost [118], SoftBoost [196], and several other references in [127].

5.1.3.3 Proposed method

We make two observations. (i) A common theme among the previous approaches is that they start with equal initial weights for all the training samples before learning the boosting classifier, and (ii) the error $\epsilon_t^\star$, which is minimized to select weak learners h_t, does not account for the (undesired) unbalanced weight distribution of the samples. We now address these two issues in more detail.

Learning prior information about training sample weights[2]: We use statistics to prioritize the training data, rather than assuming uniform initial weights for all samples. This has two advantages: (i) The most representative samples will be preferred for classification in the early stages of boosting since they start with higher initial weights. (ii) At the same time, the rate of increase in weights of hard-to-be classified samples (which can be outliers) will be slow, since these samples start with lower initial weights.

Towards this end, we perform kernel discriminant analysis [18, 62, 5, 130] on

[2]The idea of different initial weights has been used in Asymmetric boosting [188] in a different context where the missed detections are penalized more heavily than false accepts. Hence the positive examples are weighted more than the negative samples to start with. But, unlike our proposed method, *all* positive samples are given the same weight, as are the negative samples. In some sense this is like uniform weighting, with the weights different for the two classes.

the training data, and analyze the distance of projection of samples with respect to their projected class means to determine the initial weights. Essentially, given the set of labeled training data $\{(x_i, y_i)\}_{i=1}^{M}$, we first determine the projection directions α that maximize the following criterion,

$$J(\alpha) = \frac{\alpha^T S_B \alpha}{\alpha^T S_W \alpha} \tag{5.7}$$

S_B and S_W are the between-class scatter and within-class scatter matrices computed using the kernel trick [5], which instead of using a non-linear mapping Φ to explicitly transform x_i from their original space $\mathbb{R}^N$ to a high dimensional feature space $\mathbb{F}$ (allowing linear separability of x_i's), performs an implicit transformation using Mercer kernels [162] of the form $k(x_i, x_j) = \Phi(x_i).\Phi(x_j)$ by reformulating the problem in terms of dot products. The Fisher directions α are then used to project x_i to obtain z_i.

We now analyze these z_i's to learn prior information on the training sample weights. Let $\mu_{y_i}, y_i \in \{-1, 1\}$ denote the class mean of the projected samples. Then for each sample, z_i, we compute a parameter

$$\epsilon_i = \frac{|z_i - \mu_{y_i}|}{\sum\limits_{\forall k: y_k = y_i} |z_k - \mu_{y_k}|} \tag{5.8}$$

which is a function of the distance between a sample and its class mean in the projected space. Then, if $w_i = 1/M$ denote the uniform initial weights of all training samples x_i, the new initial weights ($\tilde{w_i}$) are obtained by,

$$\tilde{w_i} = w_i \exp(-\delta \epsilon_i) \tag{5.9}$$

where δ is the factor controlling the importance of weights learned from (5.8). δ

is the total classification accuracy of kernel discriminant analysis on the training samples, which gives an idea of the reliability of the learned weights. For instance, if the classification accuracy is very low (i.e $\delta \approx 0$), then $\tilde{w}_i$ reduces to w_i=1/M which is the same as the standard boosting setup. This new set of weights is then normalized to make it a distribution, and the classification function is learned using boosting as described in Algorithm 4.

A new cost function for $\epsilon_t^\star$: The error $\epsilon_t^\star$ which is minimized to select the weak learners h_t is, in its basic form (5.4), a function of the classification rate. But, as mentioned before, the problem of outliers leads to a situation where the weights of certain samples become significantly higher than others. Hence to avoid this situation, there have been efforts on modifying the cost function (5.4). The recent work by [196] addresses this issue by defining $\epsilon_t^\star$ as the relative entropy of the distribution of weights at the tth iteration, D_t, with that of the uniform initial distribution D_1. But the problem with this cost is that, D_1 need not be the best reference distribution since not all samples may be of the same quality to compare their current weights D_t with.

Since the undesirable condition caused by outliers is an uneven distribution of the sample weights, we propose to minimize the following cost function, instead of (5.4),

$$\tilde{\epsilon}_t^\star = \frac{(M - \sum_{i=1}^{M} D_t(i) y_i h_t^{x_i})}{M} + \lambda_R f_P(D_{t+1}) \tag{5.10}$$

where the first term measures the error in classification, and the second term $f_P(.)$ measures how sparse the distribution D_{t+1} (5.5) produced by the weak learner h_t

will be. λ_R is a regularization parameter. From the study of the problem of outliers [55], we deduce that $f_P(.)$ should not be sparse. In other words, the weights should not be concentrated on only a few training samples. Hence we define

$$f_P(D_{t+1}) = \frac{\sum_{i=1}^{M} I(D_{t+1}(i) < \lambda_{cost})}{M} \tag{5.11}$$

where $I(.)$ is an indicator function, and λ_{cost} is a threshold. Values of λ_R and λ_{cost} are learned using cross-validation, as explained in the Appendix. With these two modifications, we present our outlier-robust boosting algorithm in Algorithm 4. We empirically evaluated its efficacy on different UCI datasets [11] and present the results in the Appendix. At that point, we also discuss about the convergence bounds of the algorithm.

5.1.4 Test phase: Detection (Localization)

Having computed the desired functions f from (5.3) and $g^\star$ from (5.18), we localize lane markings in a test image by computing the pixel-hierarchy descriptor (5.3) for all pixels, and then classifying them using (5.18). $g^\star(x) = 1$ if the test pixel $x \in O$ (lane markings), and $g^\star(x) = -1$ otherwise. The computations needed to obtain f are performed efficiently using the concept of integral images and attentional cascade [189]. We provide the implementation details, and validation on images corresponding to daylight and nighttime road scenes in Section 5.3.1. The subsets of pixels in a test image classified as lane markings are then grouped and parameterized by a second order polynomial using the generalized Hough transform

Given: $\{(x_i, y_i)\}_{i=1}^{M}$, where $x_i \in \mathbb{R}^N$ is the training data, and $y_i \in \{-1, +1\}$ its class label, and a pool of weak learners $\mathbb{H}$,

Initialize the weight distribution of training samples D_1 from the weights learned from (5.9), i.e.

$$D_1(i) = \frac{1}{M} \exp(-\delta\epsilon_i), \forall i = 1 \; to \; M \tag{5.12}$$

For iterations t=1,...,T:

(i) $\forall h \in \mathbb{H}$, compute the classification error,

$$E_h = \frac{M - \sum_{i=1}^{M} D_t(i) y_i h^{x_i}}{M} \tag{5.13}$$

(ii) Compute an intermediate weight distribution, $\tilde{D}_{t+1}^{h}$, which the weak classifiers $h \in \mathbb{H}$ will produce,

$$\tilde{D}_{t+1}^{h}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h^{x_i})}{Z_t} \tag{5.14}$$

where $\alpha_t \in \mathbb{R}$, and Z_t is a normalization term to make $\tilde{D}_{t+1}^{h}$ a distribution.

(iii) Select the weak learner h_t with the minimum error $\tilde{\epsilon}_t^{\star}$, using the cost proposed in (5.10) as follows,

$$\tilde{\epsilon}_t^{\star} = \min_{h \in \mathbb{H}} E_h + \lambda_R f_P(\tilde{D}_{t+1}^{h}) \tag{5.15}$$

$$h_t = \arg\min_{h \in \mathbb{H}} E_h + \lambda_R f_P(\tilde{D}_{t+1}^{h}) \tag{5.16}$$

(iv) Compute the new weight distribution,

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t^{x_i})}{Z_t} \tag{5.17}$$

Output the final classifier:

$$g^{\star}(x) = sign\Big(\sum_{t=1}^{T} \alpha_t h_t^{x}\Big) \tag{5.18}$$

which is a binary value corresponding to the thresholded posterior probability $g(.)$ in (5.2).

Algorithm 4: Proposed boosting algorithm that reduces overfitting and the effect of outliers in the training set.

[13] as follows,

$$L_i = p_2\bar{x}^2 + p_1\bar{x} + p_0 \tag{5.19}$$

where L_i denote the ith lane marking, and $\bar{x}$ its horizontal coordinates in the image plane. Such a parameterization provides a coarse description of the structure of lane markings, and can differentiate between curved roads and straight roads. Having said that, it is interesting to see if a non-parametric representation using splines or a piece-wise model would provide more discriminative information about different types of lane markings. This set of L_i denote the final detection (localization) result. An illustration is provided in Figure 5.4.

5.2 Tracking and Learning some variations in road scene

We use the particle filtering framework [91] to track the localized lane markings in a video sequence. One challenging aspect of this problem comes from the non-availability of knowledge about motion patterns of lane markings. This is because the positions of lane markings in the image plane will depend on how the viewer (the camera/ vehicle) is moving, and we do not have this information since we use only the visual input from a camera mounted on the vehicle. This is the main difference between our tracking formulation and existing lane tracking algorithms like [54, 10, 99], which either assume a motion model for the vehicle, or use information from inertial vehicle sensors to update the state transition model.

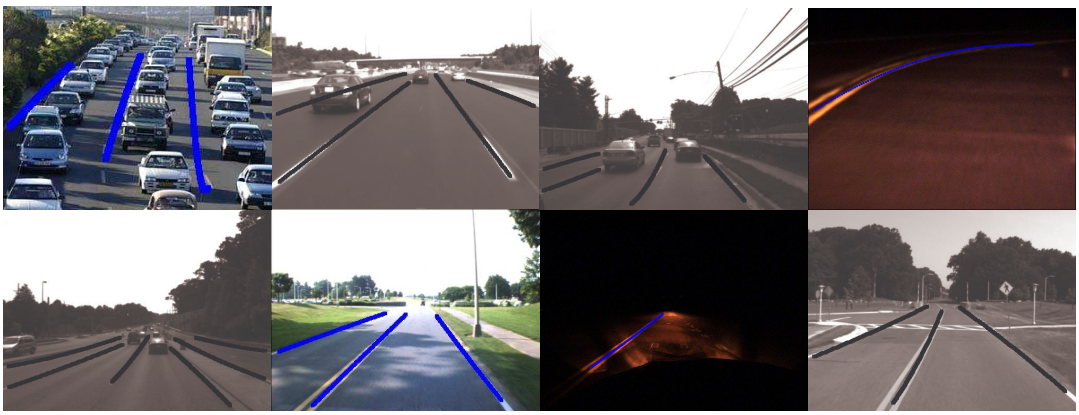


Figure 5.4: Localized lane markings L_i obtained by performing outlier-robust boosting on the pixel-hierarchy contextual features. The number of hierarchy levels M_2 for R_i was determined by the least circle enclosing the entire image, for each pixel. All images were of size 240×320, and those regions R_i that did not contribute across all pixels were excluded from the feature set h. The pixels detected as lane marking by the boosting algorithm are grouped using the generalized Hough transform [13]. The parameterized result correspond to the polynomials enclosing the detected lane marking pixels.

5.2.1 Formulation of Particle Filters to Track Lane Markings

To handle this situation, we propose a static motion model to represent the state of the particles. In other words, we always expect to see the lane markings at their initial position, and if there are any deviations from this hypothesis, we learn the causes for it. More formally,

$$State\ transition\ model : x_t = x_{t-1} + u_t \tag{5.20}$$

$$Observation\ model : y_t = G_t(x_t, v_t) \tag{5.21}$$

where x_t is the state of the system, a 7 dimensional vector $[p_2\ p_1\ p_0\ \bar{x}_{bl}\ \bar{y}_{bl}\ \bar{x}_{tr}\ \bar{y}_{tr}]^T$, where p_i are the coefficients of the polynomial characterizing a lane marking L_i (5.19), and $\{\bar{x}_*, \bar{y}_*\}$ corresponds to the location of the bottom left and top right corners of the window enclosing the lane marking. u_t corresponds to the system noise that has a fixed variance of the form $u_t = R_0 * U_0$, with R_0 being a fixed constant measuring the extent of noise, and U_0 a standardized random variable/vector. For instance, a larger value of R_0 makes the tracker search for the object in a bigger area around the location predicted by the particles. The particles are generated through sequential importance sampling, and we propose around 200 particles to approximate the system dynamics. The observation model (5.21) is characterized by v_t which corresponds to the observation noise, and $G(.)$ is a function that defines the similarity of the object at the region predicted by the particles, with that of its true (initial) appearance. Let H_1 and H_2 denote all H_c points sampled uniformly along the reference particle state, and the proposed particle state respectively. Let $\bar{H}_i = \sum_{x \in H_i} I(g^\star(x) = 1); i = 1, 2$, where $I(.)$ is an indicator function. We then compute the similarity G as,

$$G(H_1, H_2) = \frac{\bar{H}_1}{H_c}(1 - \frac{1}{H_c}\|\bar{H}_1 - \bar{H}_2\|_1) \tag{5.22}$$

5.2.2 Learning the Road Scene using Tracked Parameters

We will now discuss what this tracking model conveys. As before, let the parameterized lane markings detected from the initial frame of the video sequence

be denoted by $\{L_i\}_{i=1}^{k}$ (5.19), where k denotes the number of lane marking groups present in that frame. For instance, in a two lane road there will be three lane markings denoting the left and right edges, and the central median. We define separate particle filters to track each of the k lane markings, and then analyze the variations in its state parameters $[p_2\ p_1\ p_0\ \bar{x}_{bl}\ \bar{y}_{bl}\ \bar{x}_{tr}\ \bar{y}_{tr}]^T$ to understand the causes behind it.

From now on, let us consider one such lane marking and analyze how to interpret its tracked parameters, though this discussion is valid for all lane markings. We also assume that the road is (piecewise) flat, since otherwise the presence of slopes can lead to sudden appearance/disappearance of lane markings that are hard to track. Let the number of frames in the video sequence where the lane marking is successfully tracked[3] be denoted by $\hat{N}$, and the tracked parameters over all the $\hat{N}$ frames be represented by $[p_2^i\ p_1^i\ p_0^i\ \bar{x}_{bl}^i\ \bar{y}_{bl}^i\ \bar{x}_{tr}^i\ \bar{y}_{tr}^i]^T$, $i = 1, 2, ...\hat{N}$. Let $var(.)$ denote the variance of a set of normalized observations of a state variable, computed at k' equal intervals $\{\hat{N}_j\}_{j=1}^{k'}$. We now analyze the variance of each of the seven state parameters to infer the changes in the road scene. We do this in each of the k' intervals. After every such interval, the reference particle state is updated with that of H_2 in that interval with the largest G (5.22). Let us now consider the first interval, for instance.

[3]The detector is run again if the tracking error, $G(H_1, H_2) < \xi_g$, where ξ_g is a threshold learned using cross-validation. $\xi_g = 0.55$ in these experiments.

5.2.2.1 Static world

If the lane markings are present in almost the same location in the tracked frame as their initial position, then there will not be a substantial variation in any of the seven tracked parameters. Formally, let $\mathbf{p}_i = [p_i^1 \; p_i^2 \; .... \; p_i^{\hat{N}_1}]$ and if

$$var(\mathbf{p_i}) < \xi_t, \forall i = 0, 1, 2 \tag{5.23}$$

it implies that, irrespective of speed, the vehicle is maintaining its relative distance with respect to the lane markings (i.e. negligible lateral motion), and the road structure is also remaining constant (i.e. a straight road remains straight, and a curved road remain curved). ξ_t is a threshold learned from cross-validation, and $\xi_t = 20$ in these experiments.

5.2.2.2 Change in lateral motion of vehicle

If there are variations only in the first (p_1) and zeroth (p_0) order coefficients of the parameterized lane marking, i.e.

$$[var(\mathbf{p_1}) > \xi_t] \;\vee\; [var(\mathbf{p_0}) > \xi_t] \tag{5.24}$$

then this is due to the lateral motion of the vehicle with respect to the lane marking, while the road structure remains the same. Specifically, increase in the value of p_0 will be caused by the lateral motion of vehicle rightwards of the lane markings, and a decrease in p_0 is due to a leftwards lateral movement.

5.2.2.3 Change in road geometry

The second order coefficient (p_2) provides some information about the road geometry. For instance, if the road is straight, so will be the lane markings, and hence p_2 will be close to zero. If the road begins to curve, the change in the second order coefficient will get significant. Similar variations occur when a curved road becomes straight. Hence, if

$$var(\mathbf{p_2}) > \xi_t \tag{5.25}$$

then it might be due to changes in the road geometry. This, when coupled with variations in p_1 and p_0 (5.24), can be used to infer simultaneous changes in lateral motion of the vehicle.

5.2.2.4 Change in traffic pattern ahead of vehicle

If there is a significant variation only in (any of) the four boundary points of the area enclosing the lane markings, $\{\bar{x}_{bl}, \bar{y}_{bl}, \bar{x}_{tr}, \bar{y}_{tr}\}$, we analyze the pixels x belonging to the missing area (say, R_m) for (5.18). If

$$\sum_{x \in R_m} I(g^\star(x) = 1) < M_4/2 \tag{5.26}$$

then we classify R_m to belong to non-lane marking. $I(.)$ is an indicator function, and M_4 is the number of pixels uniformly sampled in R_m. This can be used to alert the driver about the traffic pattern ahead of the vehicle. On the other hand, if there are variations in any of the p_i also, the change in the area of bounding box might be due to the change in road geometry as well. Hence, the analysis of (5.18) in the missing region R_m provides some information on occluding objects. It is interesting

to study the scope of learning applications when the state information of all lane markings are used jointly.

We tested our hypotheses by collecting video sequences pertaining to the above four scenarios. Sample results from the tracked sequences illustrating our learning approach are given in Figure 5.5. We present the results of our experiments in Section 5.3.2.

5.3 Experiments

We first evaluate the proposed detection algorithm on day and night time images in Section 5.3.1, and then discuss the learning applications using our tracking model in Section 5.3.2.

5.3.1 Detection of Lane Markings

We tested our outlier-robust boosting classifier (5.18) on road images collected during both day and night, over a period of twelve months. Separate classifiers were used for grayscale and color images (since it changes the information contained in $F = \{F_i\}_{i=1}^{M_3}$). We collected a set of 400 images[4], for both daytime and night time scenarios, and divided them into 5 equal sets. The overlap between these sets, in terms of the nature of road scene, was kept to a minimum (of around 25%) to study the generalization ability. For each of the five trials, one set of images was used for training, and the other four for testing. The detector processes

[4]We will release our datasets for detection and tracking to the community.

Position error in detected pixels (neighborhood around the true pixel location)	Mean correct detection rate of Algorithm 4 with context									
	False positive rate									
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
1× 1	0.85	0.88	0.915	0.94	0.956	0.968	0.979	0.983	0.995	1
2× 2	0.872	0.895	0.92	0.948	0.967	0.975	0.982	0.993	1	1
3× 3	0.91	0.925	0.942	0.957	0.976	0.98	0.99	1	1	1
4× 4	0.932	0.94	0.951	0.964	0.985	1	1	1	1	1
5× 5	0.935	0.947	0.958	0.965	0.985	1	1	1	1	1

Table 5.1: Detection accuracy of Algorithm 4 with context in terms of the position error in the location of detected lane markings. The results show the position error in terms of neighborhood windows around the true lane marking locations in which the detection results occur. Performance across different false positive rates are given.

240×320 images at 15 frames per second, on a 4 GHz processor. We then counted the fraction of lane marking pixels in the regions corresponding to boosting results (which intersects with those hand-marked by the user) to determine correct detection rate, and counted the points that are not marked by the user which have been classified as the lane marking class by the algorithm to compute the false positive rate.

Based on this criterion, we present the performance curves in Figure 5.6 and study the role of spatial context and outlier robust boosting. Accuracy in terms

of the position error of detected lane markings is given in Table 5.1. Since there are no standard datasets for this problem, we implemented another machine learning approach based on [99] which learns a *single* classifier using support vector machines and artificial neural networks trained on both intensity images and contextual features f (5.3). Best performing kernel parameters were used, and the same experimental setup was followed. From these results, we make the following observations,

1. Spatial context information helps the detection process. This can be seen from the performance curves with the contextual features (5.3) and those with only the intensity image;

2. The proposed method for outlier robustness (Algorithm 4) improves detection accuracy of Adaboost. Boosting methods perform better than methods which learn single classifiers, like SVM and neural networks.

These results, overall, support the intuition behind using boosting to learn contextual information for lane marking detection.

5.3.1.1 Computations involved in determining f

Given the required image representations $F = \{F_i\}_{i=1}^{M_3}$ pertaining to the original intensity image, edge map (from Canny operator [36]), and texture responses [109], we use the integral images concept [189] to compute the contextual features

f. We perform an one-time operation to obtain the integral image $I^\star$,

$$I^\star(a,b) = \sum_{a' \leq a, b' \leq b} F_i(a', b') \tag{5.27}$$

where (a, b) denote the pixel location, using which *all* computations of a rectangular region in Haar-filter can be obtained using the knowledge of $I^\star$ belonging to the four corners of rectangle. An illustration is provided in Figure 5.7. Detection across scale is performed by using different sized Haar-filters, rather than analyzing the image across multiple resolutions. Hence, we obtain real-time performance, in line with the first boosting application to object detection [189]. Further increase in computational speed is possible reducing the image resolution.

5.3.2 Learning the Road Scene Variations

We now evaluate our hypotheses about learning variations in the road scene from the tracked model parameters. We collected video sequences pertaining to lateral motion of the vehicle, road curves, and static world models discussed in Sections 5.2.2.1 to 5.2.2.3. The lane markings are detected (localized) and parameterized (5.19) in the first frame of the videos using the approach presented in Section 5.1, and then tracked using the particle filtering framework by assuming a static motion model for the lane markings (Section 5.2). The tracker processes 240×320 images at 25 frames per second, on a 4 GHz processor. Given in Table 5.2 are the statistics of the variance of polynomial coefficients $[p_2\ p_1\ p_0]$ under different scenarios. Five video segments were used for each of the three scenarios listed below. The numbers indicate how the variance of each parameter computed using all frames in a video

varies, indicated by its mean and standard deviation across different videos.

Scenario	Mean±standard deviation of the variance on different videos		
	p_2	p_1	p_0
Static world	1.15±0.11	0.95±0.08	1.12±0.17
Lateral motion	1.85±0.25	33.12±5.66	25.93±6.01
Road geometry	45.22±12.4	2.77±0.55	3.91±1.2

Table 5.2: Statistics of the variance of polynomial coefficients for the scenarios discussed in Sections 5.2.2.1 to 5.2.2.3.

It can be seen that the variance in p_2 is high when the road geometry changes, and p_1, p_0 show a higher variance for lateral motion of the vehicle, whereas for the static world model all the three coefficients have very little variance. We also collected video sequences to test the occlusion model (Section 5.2.2.4). One such video contained three lane markings, and the middle lane marking was occluded by a vehicle (as shown in Figure 5.5). We provide the variation in the bounding box locations of the three lane markings $[x_{bl}\ y_{bl}\ x_{tr}\ y_{tr}]$ in Table 5.3. In this particular video, there is no change in lateral motion of the vehicle or the road geometry. An occluding vehicle shortens the length of the middle lane, and the missing area R_m is then analyzed for (5.18). It can be seen from these tables that the variance of lane marking parameters convey different road scene variations under our tracking framework. We then compared our tracking model with the commonly used constant velocity

Occlusion model: Position of the lane marking	Variance of the bounding box parameters and the polynomial coefficients						
	x_{bl}	y_{bl}	x_{tr}	y_{tr}	p_2	p_1	p_0
Left	1.45	1.62	1.77	1.95	1.12	0.98	1.12
Center	1.43	1.52	1.67	21.95	1.08	0.92	1.22
Right	1.22	1.52	1.83	1.65	1.22	1.98	1.43

Table 5.3: Statistics of the variance of bounding box locations and polynomial coefficients for the occlusion model discussed in Section 5.2.2.4.

model for vehicles in a particle filter framework [10, 99] and with Kalman filtering [54]. Only the visual inputs were used. We tested the tracking accuracy over 2000 hand-marked frames on both day and night images. We present the detection rate and the false positive rate in Table 5.4. The criteria used for correct detection was to check if at least TrD% of tracked points overlap with the ground truth. If the detection is less than TrD%, we consider it a mis-detection. Whereas the false positive is computed if at least TrF% of the tracked result is not included in the ground truth. We used the following values for $(TrD, TrF) = \{(80, 20), (90, 10), (80, 10), (90, 20)\}$. Although there are many different ways to validate tracking algorithms [203], we chose this method mainly to understand the performance of our proposed tracking model (Section 5.2). It can be seen that at these operating points, the performance of our model is comparable with other models, with a significant reduction in the

false positive rate.

Tracking model	Performance criteria (in %) Mean±standard deviation of	
	Correct tracking rate	False positive rate
Particle filtering [10]	82.1 ± 3.5	22.3 ± 3.2
Kalman filtering [54]	76.5 ± 3.7	32 ± 4.8
Ours	83.2 ± 3.1	15.8 ± 2.5

Table 5.4: Comparison of different tracking models on a set of 2000 frames

5.4 Discussion

Through this work, we have studied the utility of learning approaches for the detection and tracking of lane markings using visual inputs from a camera mounted in front of a vehicle. We illustrated the advantages of modeling spatial context information through an outlier-robust boosting formulation, and inferring some variations in the road scene from the statistics of tracked model parameters under a static motion model for the lane markings. Without any assumptions on the road structure, or the motion pattern of the vehicle, we demonstrated some results on challenging daylight and night-time road scenes.

At the core of our approach is the importance placed on the quality of data. Although our data for training and testing had several non-common exemplars, there can be instances such as foggy or rainy road conditions where the visual inputs alone

are insufficient to detect lane markings. An illustration is provided in Figure 5.8. Hence, in order to obtain robust performance under varied road conditions, one could use complementary information from different sensing modalities such as the vehicle's inertial sensors, GPS information and models for road geometry. Towards that end, we hope that the results from this study will provide some insights into the capabilities of *learning contextual information* from *visual* data.

5.5 Appendix: Outlier-robust boosting algorithm

5.5.1 Outlier Robustness of Adaboost - Discussion

We now analyze the iteration bounds of the proposed outlier robust boosting algorithm (Algorithm 4) in converging to the output hypothesis with optimal classification margin for the training data. For the class of boosting algorithms that study outlier robustness by modifying the cost function to achieve balanced weight distribution, results pertaining to the maximum achievable margin, and the number of iterations required for it were established by [197, 196]. Specifically, these results apply to methods where the cost function pertaining to weight distribution of samples is generally expressed as the relative entropy between the predicted weight distribution D_{t+1} and the desired weight distribution, say $D^{\star}$.

We now adapt the results of [197] by rewriting our proposed cost function f_P (5.11) in terms of relative entropy follows,

$$\bar{f}_P(D_{t+1}) = \frac{1}{\lambda_{norm}} \sum_{i=1}^{M} \bar{D}_{t+1}(i) \log \frac{\bar{D}_{t+1}(i)}{\bar{D}^{\star}(i)} \tag{5.28}$$

where $\bar{D}_{t+1}$ and $\bar{D}^{\star}$ are obtained by transforming the predicted weight distribution D_{t+1} and desired weight distribution $D^{\star}$ (that penalizes sparse non-zero weights using the parameter λ_{cost}) as follows: $\bar{D}^{\star}(i) = 1/M, \forall i = 1 \ to \ M$, $\bar{D}_{t+1}(i) \approx 0, \forall i \ s.t. \ D_{t+1}(i) \geq \lambda_{cost}$, and $\bar{D}_{t+1}(i) \approx 1/M', \forall i \ s.t. \ D_{t+1}(i) < \lambda_{cost}$. $M' < M$ is the number of samples for which $D_{t+1}(i) < \lambda_{cost}$, and λ_{norm} is a normalization constant whose value equals $\frac{M}{M'} \log \frac{1/M'}{1/M}$. When $\bar{f}_P$ is used in (5.15) instead of f_P, the optimization problem obtains the form for which the convergence results of [197] apply (since, the main difference between our method and [197, 196] is in the definition of the two distributions whose relative entropy is being computed).

Hence, the proposed boosting algorithm terminates after at most $O(\lceil \frac{2}{\Delta^2} \log(M/\nu) \rceil)$ iterations with a convex combination $g^{\star}$ (5.18) that is at most Δ below the optimum classification accuracy Δ_1 (available to the system). ν is a capping parameter that handles hard-to-be classified samples using soft margins. The effect of parameters Δ_1 and ν on the classification accuracy are studied empirically in the following section.

5.5.2 Empirical evaluation

We used ten UCI benchmark datasets [11] to evaluate the proposed boosting algorithm. The data comes in 100 predefined splits, categorized into training and testing sets. For each split, we used 5-fold cross-validation to select the best kernel and its parameters, and the regularization parameters λ_R and λ_{cost} (5.15). This leads to 100 estimates of the generalization error for each dataset. The means

and the standard deviations are given in Table 5.5. We experimented with three types of Mercer Kernels, namely - Gaussian RBF $k(x_i, x_j) = \exp(-||x_i - x_j||_2^2/c_e)$, polynomial $k(x_i, x_j) = (x_i.x_j)^d$ and sigmoid $k(x_i, x_j) = \tanh(\kappa(x_i.x_j) - \delta_e)$, where x_i and x_j are a pair of data points. For each dataset, without the loss of generality, the best performing kernel (5.9) was used since this step needs to be done *separately* for every experiment.

It can be seen from Table 5.5 that our algorithm gives better performance when compared with the existing approaches on most datasets, and is close to the best algorithm on the others. Based on this study, we have three observations,

1. The weight learning process depends on the classification accuracy obtained from kernel discriminant analysis (the parameter δ in (5.9)). It would be interesting to see how the results vary when the bag of kernels is increased, and when a classifier better than kernel discriminant analysis is used;

2. The individual effect of the two components of our algorithm is studied in Table 5.6. It can be seen that the cost function argument performs slightly better than weight learning, while when used jointly, they produce the least generalization error;

3. Finally, the modifications suggested in our algorithm can be used in tandem with existing methods that focus on other aspects of boosting in handling outliers.

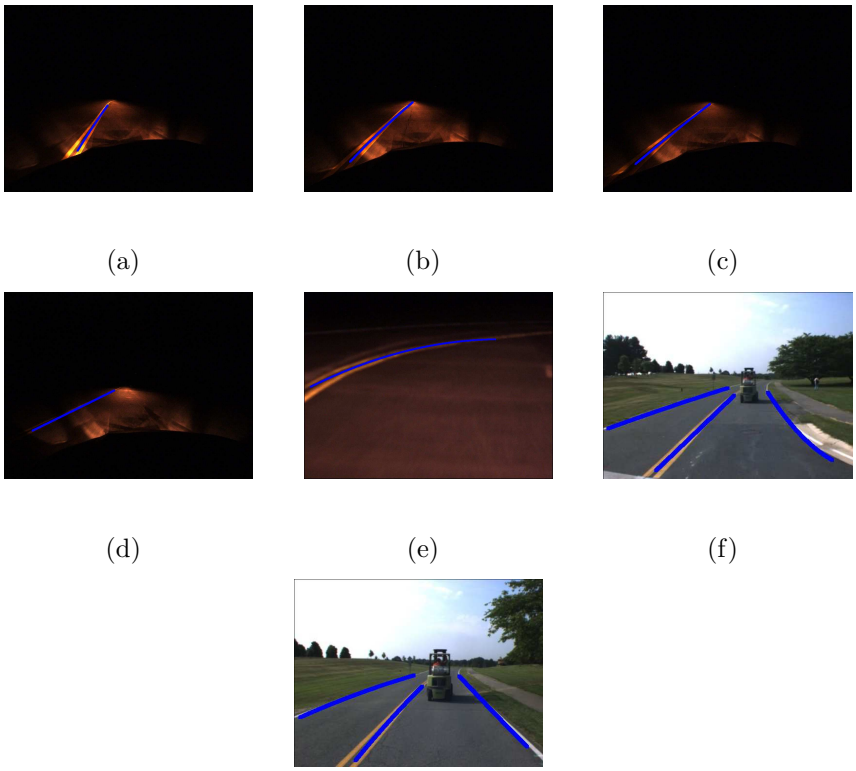


Figure 5.5: Results of lane tracking on day and night time video sequences. Images (a) through (d) illustrate lateral motion of the vehicle w.r.t the lane marking (Section 5.2.2.2). The parameters of polynomial fit $[p_2\ p_1\ p_0]$ for these images are as follows: [-0.0009565 5.204 -411.4], [-0.009422 3.218 -92.47], [-0.0009464 1.8893 -2.416], [-0.0009211 0.4853 140.7] indicating substantial changes in p_1 and p_0. Image (e) has the following parameters: [-0.3199 0.5179 363.8], where the large variation in p_2 is due to the change in road geometry from straight to curved (Section 5.2.2.3). Images (f) and (g) are used to illustrate the effect of an occluding vehicle (Section 5.2.2.4). The polynomial coefficients of the middle lane markings in both images are [-0.0002544 0.94 -86.4], [-0.0002133 0.96 -90.4]. But the bounding box parameters $[x_{bl}\ y_{bl}\ x_{tr}\ y_{tr}]$ are given by [100 1 225 268] [100 1 225 208]; The missing area R_m does not satisfy (5.18) due to the presence of the vehicle.

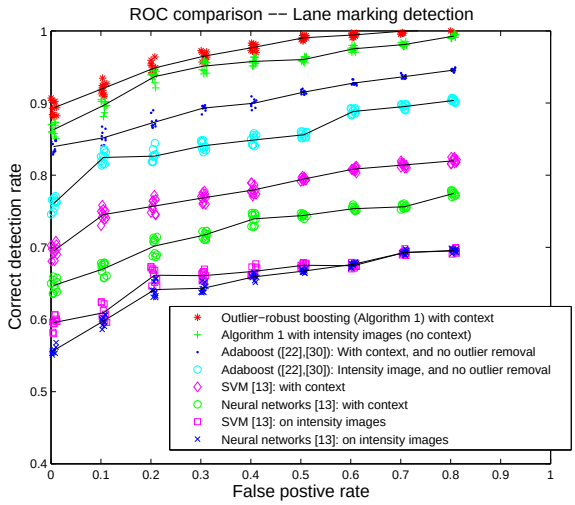


Figure 5.6: ROC curves for lane marking detection: comparing different learning methods on an internally collected dataset of 400 day/ night-time road images using a 5 fold cross-validation. The detection results correspond to pixel error of detected lane markings within a 3×3 neighborhood around the true pixel location.

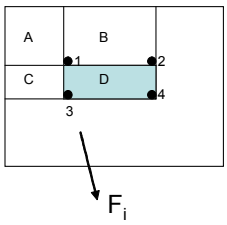


Figure 5.7: Computing the contextual features f using Integral images [189]. Given an image representation F_i, to compute the cumulative information within the region D, we only need the value of $I^\star$ for the four corner points 1,2,3 and 4. The information can be computed according to the pattern of Haar-filters.

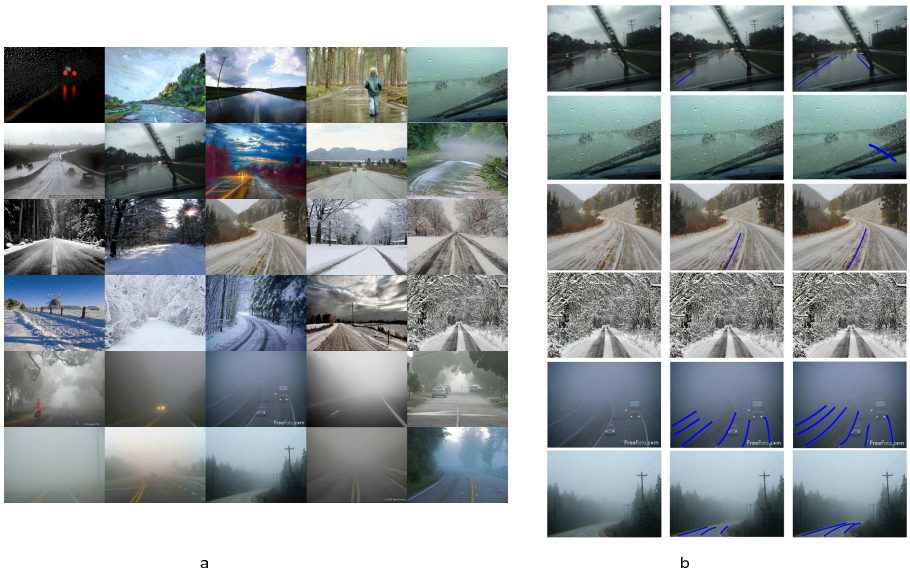


Figure 5.8: Road scenarios under inclement weather conditions. (a): Sample road images under rainy, snowy and foggy conditions collected from internet. We collected around 150 such images. Let us call them I_{web}. We retained other training images that we collected before (explained in Section 5.3.1). (b): (L-R) input test image; output of our algorithm without including I_{web} for training; output of our algorithm after including I_{web} in training (the test images shown in (b) were not used in training). We can see that under these conditions, the visual sensors are not adequate to detect lane markings completely, however learning does produce some improvement.

Datasets	Adaboost [76]	LPBoost [53]	SoftBoost [196]	BrownBoost [74]	Adaboost reg [153]	Ours (Algorithm 4)
Banana	13.3 ± 0.7	11.1 ± 0.6	11.1 ± 0.5	12.9 ± 0.7	11.3 ± 0.6	**10.1 ± 0.3**
B.Cancer	32.1 ± 3.8	27.8 ± 4.3	28.0 ± 4.5	30.2 ± 3.9	27.3 ± 4.3	**26.2 ± 3.2**
Diabetes	27.9 ± 1.5	**24.4 ± 1.7**	**24.4 ± 1.7**	27.2 ± 1.6	24.5 ± 1.7	24.5 ± 1.2
German	26.9 ± 1.9	24.6 ± 2.1	24.7 ± 2.1	24.8 ± 1.9	25.0 ± 2.2	**23.4 ± 1.1**
Heart	20.1 ± 2.7	18.4 ± 3.0	18.2 ± 2.7	20.0 ± 2.8	17.6 ± 3.0	**16.9 ± 2.2**
Ringnorm	1.9 ± 0.3	1.9 ± 0.2	1.8 ± 0.2	1.9 ± 0.2	1.7 ± 0.2	**1.65 ± 0.2**
F.Solar	36.1 ± 1.5	35.7 ± 1.6	35.5 ± 1.4	36.1 ± 1.4	34.4 ± 1.7	**33.7 ± 1.2**
Thyroid	4.4 ± 1.9	4.9 ± 1.9	4.9 ± 1.9	**4.6 ± 2.1**	4.9 ± 2.0	**4.6 ± 2.1**
Titanic	22.8 ± 1.0	22.8 ± 1.0	23.0 ± 0.8	22.8 ± 0.8	22.7 ± 1.0	**21.5 ± 1.0**
Waveform	10.5 ± 0.4	10.1 ± 0.5	9.8 ± 0.5	10.4 ± 0.4	10.4 ± 0.7	**9.12 ± 0.5**

Table 5.5: Boosting methods on UCI Dataset [11]: comparing the proposed algorithm with other methods for outlier robustness of Adaboost. Results correspond to the mean and standard deviation of the generalization error.

Datasets	Ours - weight learning only (5.12)	Ours - cost function only (5.15)	Ours - Algorithm 4 (both (5.12) and (5.15))
Banana	10.6 ± 0.3	10.4 ± 0.5	10.1 ± 0.3
B.Cancer	26.5 ± 3.2	26.45 ± 3.0	26.2 ± 3.2
Diabetes	24.5 ± 1.2	24.5 ± 1.2	24.5 ± 1.2
German	23.9 ± 1.1	23.6 ± 0.9	23.4 ± 1.1
Heart	17.2 ± 2.2	17.15 ± 2.2	16.9 ± 2.2
Ringnorm	1.8 ± 0.2	1.8 ± 0.2	1.65 ± 0.2
F.Solar	34.1 ± 1.2	33.75 ± 1.2	33.7 ± 1.2
Thyroid	4.7 ± 1.6	4.7 ± 1.6	4.6 ± 2.1
Titanic	21.5 ± 1.0	21.5 ± 1.0	21.5 ± 1.0
Waveform	9.5 ± 0.5	9.5 ± 0.1	9.12 ± 0.5

Table 5.6: UCI Dataset [11]: comparing the individual components our proposed algorithm. Results correspond to the mean and standard deviation of the generalization error.

Chapter 6

Max-margin Clustering: Detecting Margins from Projections of Points on Lines

Unsupervised identification of patterns in data, broadly referred to as clustering, is an important problem that has been extensively studied [78, 59] over the last several decades. Existing approaches can be characterized based on pattern representation, criteria for similarity between patterns, and cost functions that determine the grouping mechanism [94, 93]. The goal of this work is to find maximally separable clusters, given the knowledge of number of clusters, and an appropriate representation of data that depends on the specific application of interest.

There are two broad approaches to this problem, both of which draw inspiration from supervised classification. The first class of methods performs clustering by reducing the original dimensionality of data. Subspace selection is performed using discriminative methods such as linear discriminant analysis (LDA) [78], which starts with random assignments of class labels, or using generative methods such as principal component analysis (PCA) [78], locally linear embedding (LLE) [156] and Laplacian Eigenmaps [19]. Standard clustering algorithms like K-means [121] and spectral methods [135, 171] are then applied in the resulting subspace to determine the cluster assignments. However, the absence of 'true' data labels makes this a chicken-and-egg problem, and there are methods addressing this issue by studying

the feedback between subspace selection and clustering (e.g. [51, 211, 210]). The second class of approaches is based on obtaining clusters with maximum separating margins [205, 24, 146], and are primarily motivated by the paradigm of max-margin supervised classifiers, such as support vector machines (SVM) [35]. Most of these methods can be visualized as implicitly running an SVM with different possible label combinations to obtain a final cluster assignment having maximum margin. However, as this process results in a non-convex integer optimization problem, subsequent efforts [217, 215, 185, 192, 216] have proposed approximation strategies that obtain a solution in polynomial time.

Contributions: Our approach belongs to the latter category. However, unlike most existing solutions that optimize over all possible cluster assignments, we seek a more basic understanding of the relationship between data points and margins. Since regions corresponding to the separating margins have (ideally) no data points, our goal is to identify these sparse regions by analyzing the projections of unlabelled points $\mathbb{X} \in \mathbb{R}^N$ on the set of all possible lines L in $\mathbb{R}^N$. In this process,

- We first derive certain properties which the projections of $\mathbb{X}$ on a line interval will satisfy, if and only if that interval lies *outside* of a cluster, under assumptions of linear separability of clusters and absence of outliers;

- We extend these results to define a similarity measure, which computes the probability of finding a margin in the line interval between a pair of points, and use it to perform global clustering. We relax the assumption of linear separability of clusters using kernel methods, and address the problem of outliers

through methods that emphasize a balance between cluster sizes.

Outline of the chapter: Section 6.1 studies the properties of projections of data on line intervals for two cluster and multi-cluster cases. Section 6.2 proposes a method to determine cluster assignments. Section 6.3 validates the proposed method through experiments on standard UCI datasets [73], and on computer vision applications, such as face recognition under illumination variations [173, 80], and 2D shape matching [105, 82]. Section 6.4 concludes the chapter. Figure 6.1 provides an illustration of our approach.

6.1 Properties of projection of $\mathbb{X}$ on L

Let the input $\mathbb{X}$ contain a set of M unlabelled data points, $\{x_i\}_{i=1}^{M} \in \mathbb{R}^N$, belonging to k clusters. For the ease of discussion, we make the following assumptions that will be relaxed later; (i) Points in $\mathbb{X}$ belong to clusters that are (pair-wise) linearly separable in their input space, and (ii) No outliers are present in the data (specific details regarding this assumption will be provided in the following sections). In what follows, we try to detect the presence of margins by studying the patterns in projections of $\mathbb{X}$ on the set of lines L. We will motivate our method by drawing parallels to the supervised max-margin classification scenario.

6.1.1 Case A: Two clusters

We first study a two-cluster problem, i.e. when $k = 2$. For now, let us assume that the true labels of x_i, $y_i \in \{-1, +1\}$, are available. A max-margin classifier,

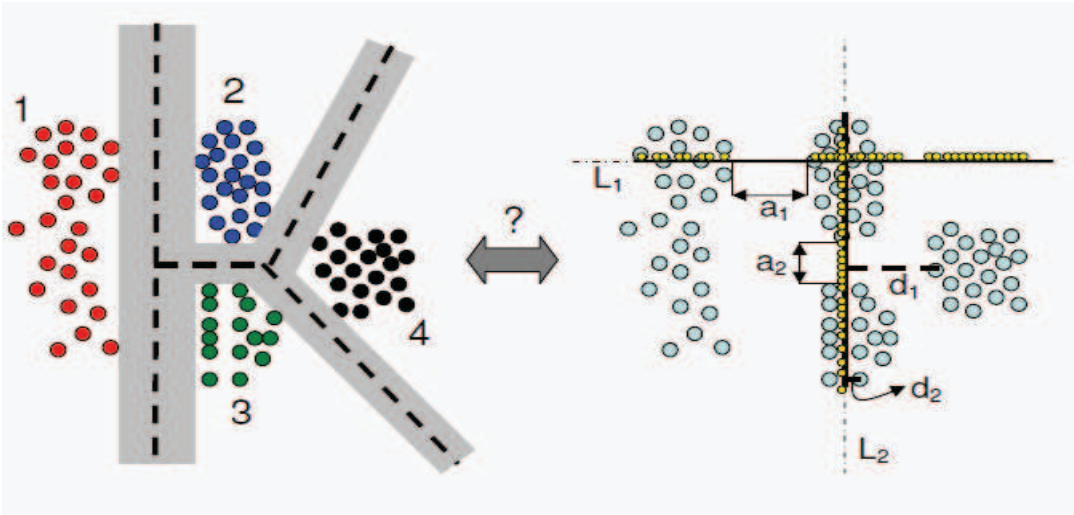


Figure 6.1: Left: A four class, linearly separable problem with $\mathbb{X} \in \mathbb{R}^2$. With known class labels, a max-margin classifier produces margins (shaded regions) with the separating hyperplanes indicated by the dashed lines. Right: In an unsupervised setting, how to identify these margin regions? Consider two lines L_1 and L_2, and project $\mathbb{X}$ on them (small yellow dots). Interval a_1 of L_1 has no projected points since it lies in margin region $\perp$ to the hyperplane that separates a cluster from *all other clusters*; whereas interval a_2 of L_2 (whose margin separates *only a pair* of clusters) has projected points from other clusters, with their minimum distance of projection d_1 more than that of d_2 for points projected elsewhere on L_2. In this work, we study the statistics of location and distance of projections of $\mathbb{X}$ on all lines L, to identify margins and perform clustering.

such as a linear support vector machine (LSVM), produces a decision boundary that optimizes the following objective function,

$$\min_{w,b} \frac{1}{2}\|w\|_2^2, \ s.t. \ y_i(w^t x_i + b) \geq 1, \ \forall i = 1 \ to \ M \tag{6.1}$$

where $(.)^t$ is the transpose operator. Essentially, the separating hyperplane S : $w^t x + b = 0$, where w is the normal to S, is chosen such that it has a maximum separation of $1/\|w\|_2$ from the tangent of support vectors from either classes given by $H_1 : w^t x + b = 1$, and $H_2 : w^t x + b = -1$, respectively. The margin region bounded by parallel hyperplanes H_1 and H_2 is denoted by M_S, which is characterized by no data points $\mathbb{X}$, and therefore provides a separating margin of $\gamma = 2/\|w\|_2$ between the two classes. An illustration is provided in Figure 6.2.

To identify M_S from an unlabelled set of points $\mathbb{X}$, we now consider the projections[1] of $\mathbb{X}$ onto the set of all lines L in $\mathbb{R}^N$. Let x_{i_p} denote the location of projection of $x_i \in \mathbb{X}$ on a line. It is not hard to visualize that the projection of $\mathbb{X}$ creates *patterns* on the line, which is shown using black dots in Figure 6.2 for lines A, and B. Notice that for line A, the projection of $\mathbb{X}$ on them creates two dark patterns with a sparse region in between, which clearly captures the margin between the left and right clusters. On the other hand, line B due to its orientation fails to capture the margin, which makes it unsuitable for our purposes. The intuition behind our algorithm is that if we draw sufficient number of lines between points in $\mathbb{X}$, we may be able to capture the margins that separate the clusters, which in

[1] We are interested in the shortest (perpendicular) projection. Let x_1 and x_2 be any two points through which w passes. To project a new point x_i onto w, we first compute the line passing through x_1 and x_i, say w_{x_1}, and then obtain the location of its projection, $x_{i_p} = x_1 + \frac{w_{x_1}.w}{w.w}$. The distance of projection, d_{i_p}, is given by $\|w_{x_1} - x_{i_p} w\|_2$.

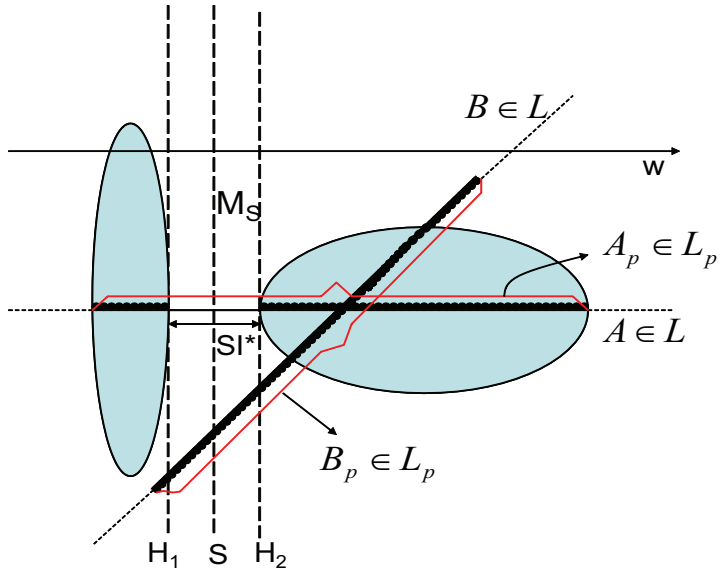


Figure 6.2: An illustration of projection of points on different line segments. The two clusters are represented by ellipses. Assume that points $\mathbb{X}$ are present everywhere inside the ellipses. When labels of $\mathbb{X}$ are available, S will be the separating hyperplane, and H_1 and H_2 are tangents to support vectors of either classes. M_S denotes the margin region (bounded by H_1, and H_2). $SI^\star = \gamma$ is the margin, and w is the normal to S. In a clustering scenario, where labels of $\mathbb{X}$ are unknown, consider two lines ($A, B \in L$ in $\mathbb{R}^2$). L_p refers to the segment of L enclosing all projections x_{i_p} (dots in black). It can be seen that on intervals in $A_p \perp S$, there is no x_{i_p} in the region corresponding to margin M_S; hence, there exist $SI^\star$. For any other line segment not perpendicular to S, say B_p, maximum possible $SI < \gamma$.

turn would aid in clustering of $\mathbb{X}$. Furthermore, we can now discard A and B, in favor of line segments A_p and B_p, which are obtained by walking on those lines and truncating their bounds to lie between the first and last projected points of $\mathbb{X}$ that we encountered. Let the set of these truncated line segments across all L be referred as L_p.

Before analyzing L_p in pursuit of M_S, under the assumption of no outliers in the data, we constrain the maximum margin γ to exist only between points belonging to different clusters, and not otherwise. We now define the following.

Definition Sparsity index of a line segment $z \in L_p$, $SI(z) \in \mathbb{R}$, is the maximum distance[2] travelled along z where there are no projected points x_{i_p}. Let $SI^\star = \max_{z \in L_p} SI(z)$.

Proposition 6.1.1 *$SI^\star = \gamma$ is realized only by those set of line segments $C \subset L_p$ that are normals to the separating hyperplane S, and the intervals in C where $SI^\star$ occurs are those that correspond to the margin region M_S. Furthermore, $\forall \bar{C} = L_p \backslash C$, $SI(\bar{C}) < \gamma$.*

Proof Follows directly from (6.1), provided there exist a unique max-margin separating hyperplane S. ∎

Hence in an unsupervised setting, we *directly* obtain cluster assignments of $\mathbb{X}$ by identifying a line segment (in C) with maximum SI, where the *minimum* distance between a pair of points belonging to *different* clusters is $SI^\star$.

[2] By distance, we refer to the standard Euclidean norm $\|.\|_2$ between the end points of the interval of z containing no x_{i_p}. Further, we might occasionally drop the argument for $SI(.)$ for sake of simplicity.

6.1.2 Case B: Multiple clusters

We now consider the general case where the number of clusters $k \geq 2$. We again draw motivation from the supervised max-margin classification problem, for which there are two popular strategies; (i) directly solve for the multi-class problem by optimizing a single objective function (e.g. [45]), and (ii) decompose the problem into one that *combines* several binary classifiers (e.g. [7]). We will motivate our study using the latter strategy, where we are primarily interested in understanding the information conveyed by a margin, its effect on the distribution of x_{i_p} on w, and the existence of $SI^\star$ to perform clustering.

Consider a set of points $\mathbb{X} = \{x_i\}_{i=1}^M$ with known labels $y_i \in \{1, 2, ..., k\}$ belonging to one of the k linearly separable classes. A supervised classifier produces the final decision boundary $\hat{S}$ by combining several independent binary separating hyperplanes S_i,

$$\hat{S} = g(S_1, S_2, ..., S_l) \tag{6.2}$$

where g is a combination function[3] that determines the piece-wise linear boundaries of the decision regions $R_i, i = 1 \; to \; k$, belonging to the k classes. An illustration is provided in Figure 6.3 for a three-class problem.

Notations: Let $\mathbb{X}_i \subset \mathbb{X}$ represent the set of points that are separated by S_i. Let w_i be the normal to S_i, and let the length of the corresponding margins be denoted by γ_i. Let M_{S_i} denote the margin region corresponding to γ_i when S_i is considered

[3]The value of l depends on the type of binary classifiers used, for instance, one-vs-all or one-vs-one, and the mode of combination g. The maximum number of such classifiers, l', is therefore k for one-vs-all, and $\binom{k}{2}$ for one-vs-one. Since not all hyperplanes might contribute to decision making, $l \leq l'$.

in isolation (i.e. a two-class problem with $\mathbb{X} = \mathbb{X}_i$), and let $M'_{S_i} \subseteq M_{S_i}$ denote the bounded margin region in a multi-class setting where S_i *independently* classifies $\mathbb{X}_i$ (6.2). For subsequent analysis, we partition the space of $\mathbb{X}$ into two regions; (i) cluster regions $CL = \cup_{i=1}^{k} CL_i$, where CL_i is the convex hull of all points belonging to the ith cluster, and (ii) non-cluster regions CL' that include $\cup_{i=1}^{k} M'_{S_i}$ pertaining to margins, and T comprising of $\cup_{i=1}^{k} R_i \setminus CL_i$, and regions where more than one S_i is involved in decision making. Figure 6.3 illustrates this for $k = 3$.

To study the validity of Proposition 6.1.1 in clustering unlabelled $\mathbb{X}$ belonging to multiple linearly separable groups, we first seek to understand the interference of $\mathbb{X}'_i = \mathbb{X} \setminus \mathbb{X}_i$ on the pair of clusters an S_i separates. To visualize what we mean by this, consider the line interval $a_2 \in L_p$ in Figure 6.1 that lies in a margin region perpendicular to S_i which separates $\mathbb{X}_i$ belonging to group 2 and 3. Although a_2 does not contain any points from $\mathbb{X}_i$, many points $\mathbb{X}'_i$ belonging to group 1 and 4 get projected on a_2. Therefore, we first analyze the *relevance* of $SI^\star$ for a multi-cluster problem.

6.1.2.1 Existence of $SI^\star$ - Information conveyed by x_{i_p}

Instead of analyzing the projections of $\mathbb{X}$ directly on L_p, we consider the set of all continuous intervals that are contained in L_p. Let $Int = \{Int^{CL}\} \cup \{Int^{CL'}\}$ be a set, such that Int^{CL} denotes intervals within the cluster regions $CL_i, \forall i = 1 \; to \; k$, and $Int^{CL'}$ denotes those outside the cluster. For example, the line segment $A_p \in L_p$ in Figure 6.2 has Int^{CL} corresponding to its intervals within the ellipses, and $Int^{CL'}$ corresponding to those in M_S. We now analyze the existence of $SI^\star$, in this case,

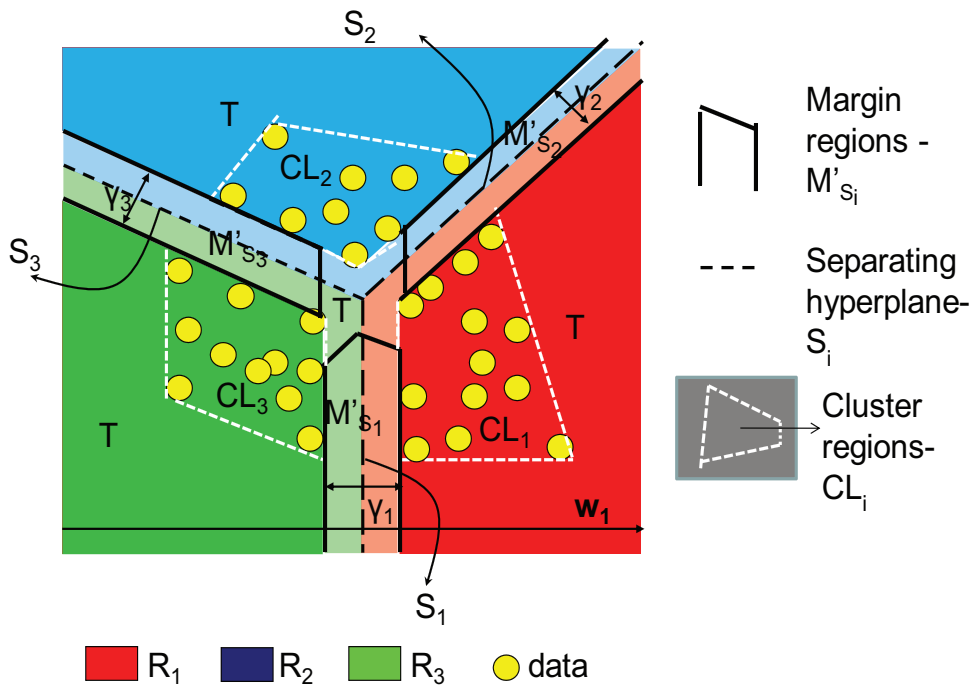


Figure 6.3: Partitioning the space of $\mathbb{X}$ into different regions. The data $\mathbb{X}$, shown in yellow circles, belong to three linearly separable clusters ($k = 3$). With known class labels, a supervised classifier $\hat{S}$ produces decision regions $R_i, i = 1\ to\ 3$ belonging to the three classes, shown in red, blue, and green respectively. The margins regions M'_{S_i} are bounded by solid black lines, with their corresponding margins denoted by γ_i. The separating hyperplanes S_i are given in black dashed lines. We now divide the space of $\mathbb{X}$ into, (i) cluster regions CL_i in white dotted lines, and (ii) non-cluster regions that comprise of margin regions M'_{S_i} and T.

$SI = \gamma_i, \forall i = 1$ *to* l, for intervals in the corresponding margin regions M'_{S_i}. In doing so, we assume that there are no outliers in the data; (i.e.) if the maximum margin between points belonging to a same cluster is M_m, we require that $M_m < \min_{i=1}^{l} \gamma_i$.

Proposition 6.1.2 *For any* $Int^{CL'}$ *in* $M'_{S_i} \perp S_i$, *a* $SI^\star = \gamma_i$ *will be realized iff* $M'_{S_i} \equiv M_{S_i}$.

Proof The basic criteria for $SI^\star = \gamma_i$ to exist is that there should be no $\mathbb{X}$ in M_{S_i}. From the definition of the margin of a separating hyperplane, M_{S_i} will not contain $\mathbb{X}_i$. If $\exists\ \mathbb{X}'_i$ in M_{S_i}, then there will exist a $S_j, j \neq i$ (as determined by g), to classify $\mathbb{X}'_i$ from $\mathbb{X}_i$. This, in turn, leads to an $M'_{S_i} \subset M_{S_i}$ containing no $\mathbb{X}$, which results in a maximum realizable $SI < \gamma_i$ for any $Int^{CL'}$ in $M'_{S_i} \perp S_i$. ∎

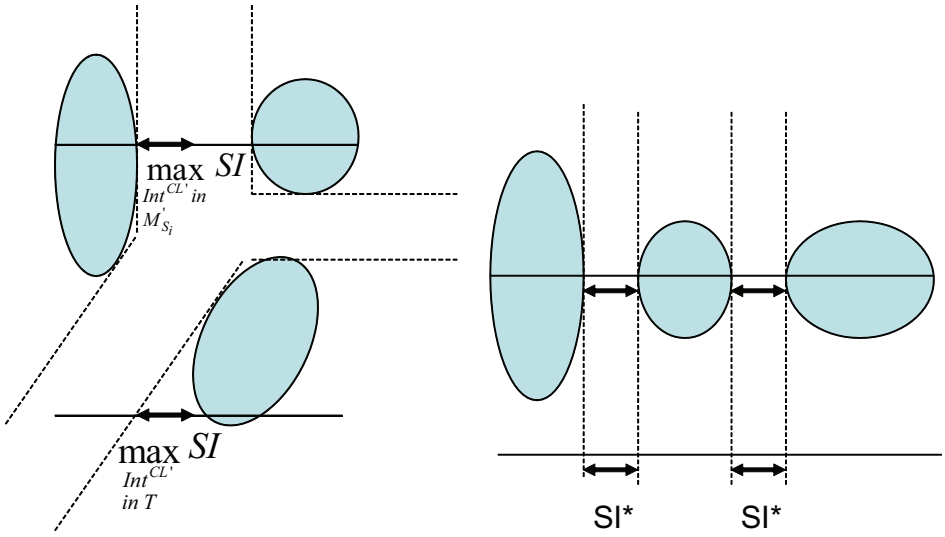
We now focus on intervals belonging to other regions.

Corollary 6.1.3 *For any interval* Int^{CL} *within the cluster region,* $SI < \min_{i=1}^{l} \gamma_i$*; and for an interval* $Int^{CL'}$ *in* $M'_{S_i} \not\perp S_i$, $SI < \gamma_i$.

Proof Follows from Propositions 6.1.1 and 6.1.2. ∎

However, SI for intervals belonging to T is completely dependent on the spatial configuration of the data. Unless otherwise $M'_{S_i} \equiv M_{S_i}, \forall i = 1$ *to* k, the maximum SI realizable at an interval in M'_{S_i} can be realized for intervals in T also. An illustration is provided in Figure 6.4. Hence, with regard to the information conveyed by x_{i_p}, we finally state the following without a proof.

Corollary 6.1.4 *Irrespective of whether* $M'_{S_i} \equiv M_{S_i}$, *an interval with* $SI \geq \min_{i=1}^{l} \gamma_i$ *can exist only outside a cluster.* ∎



(a) (b)

Figure 6.4: Illustrating the data-dependent nature of projections in intervals Int in T. Consider a three cluster problem, where the ellipses are completely filled with points. (a): Since $M'_{S_i} \subset M_{S_i}$, $SI^\star$ does not exist. However, the maximum possible SI occurs for intervals both in margin regions, and in T (shown with double head arrows). (b): When $M'_{S_i} \equiv M_{S_i}$, $SI^\star$ exists, and such intervals only belong to the margin regions (as in the case of a two-cluster problem).

6.1.2.2 Role of distance of projection d_{i_p}

Since existence of $SI^\star$ is itself dependent on the data, the location information of projected points x_{i_p} alone is insufficient to characterize margin properties for a multi-cluster problem. We make the following observation. M'_{S_i}, the informative subset of M_{S_i}, is obtained by *spatially bounding* M_{S_i} to remove the interactions of $\mathbb{X}'_i$. Hence, one way of translating this spatial neighborhood information for clustering is to use the distance of projection of points d_{i_p}. To understand the role of d_{i_p}, let us define D_{min} of a line interval to be the minimum[4] d_{i_p} of all x_{i_p} projected in *that* interval. In similar vein, let D_{max} of an interval denote the maximum d_{i_p} of all x_{i_p} from that interval.

Proposition 6.1.5 *D_{min} for intervals within a cluster is less than that for all intervals $Int^\star$ in a margin region perpendicular to their corresponding separating hyperplanes, i.e. $Int^\star = \cup_i \{Int^{CL'}$ in $M'_{S_i} \perp S_i\}$. Specifically, for intervals:*

1. *within a cluster region,* $\max\limits_{Int^{CL}} D_{min} \leq M_m/2$;
2. *in the margin region perpendicular to the separating hyperplane,* $\min\limits_{Int^\star} D_{min} \geq \min\limits_{i=1}^{l} \gamma_i$.

Proof (i) This result comes directly from the no-outlier assumption in $\mathbb{X}$. When the maximum margin between any two points belonging to a cluster $M_m < \min\limits_{i=1}^{l} \gamma_i$, for any Int^{CL} there will exist an x_{i_p} with $0 \leq d_{ip} \leq M_m/2$. Furthermore, there exists a

[4]To facilitate later discussion, for intervals with a $D_{min} = 0$, we set $D_{min} = \epsilon_{min}$, where ϵ_{min} is a positive real number slightly greater than zero. $\epsilon_{min} = .001$ in our experiments.

pair of projections $(x_{i_p}, x_{j_p}), j \neq i$ such that, $0 \leq d_{i_p} \leq M_m/2$, $0 \leq d_{j_p} \leq M_m$ and $0 \leq |d_{i_p} - d_{j_p}| \leq M_m$. (ii) For intervals $Int_i^\star \in Int^\star$ in $M'_{S_i} \perp S_i$, the points $\mathbb{X}'_i$ need to travel a *minimum* distance of their corresponding margins to interfere with $Int_i^\star$. Hence, across all such intervals $Int^\star$, $D_{min} \geq \min_{i=1}^{l} \gamma_i$. ∎

The salient points of these discussions are captured by Figure 6.1 for a four-cluster problem where, (i) the intervals $a_1, a_2 \in Int^\star$ illustrate that $SI^\star$ need not be realized at all margin regions, and (ii) D_{min} for intervals belonging to $Int^\star$, for instance $a_2 \in L_2$ whose $D_{min} = d_1$, is *always* larger than that for intervals within a cluster. Hence, d_{i_p} conveys much more data-independent[5] information than that portrayed by x_{i_p} alone. We now define the following.

Definition Sparsity index of a line interval for a multi-cluster problem, $SI_m = [SI]_{\mathcal{D}} \in \mathbb{R}$, is the maximum distance travelled on that interval in which there exist no projected points x_{i_p} with $d_{i_p} < \mathcal{D}$. The dependency of SI_m on d_{i_p} is controlled by $\mathcal{D}$, which can take any value in the closed interval $[D_{min}, D_{max}]$.

As in the case of two-cluster problem, where $D_{min} = D_{max} = \infty$ for an interval with $SI^\star$ (Proposition 6.1.1), SI_m can be used to determine if an interval is associated within a cluster or outside cluster regions, as follows.

Proposition 6.1.6 *An interval with* $[SI]_{D^\star_{min}} \geq \min_{i=1}^{l} \gamma_i$ *can lie only outside a cluster, where* $D^\star_{min} \geq \min_{i=1}^{l} \gamma_i$.

[5]The properties of D_{min} for intervals in T, however, are completely dependent on data, as was the case with x_{i_p} (Figure 6.4).

Proof Follows from Corollary 6.1.4 and Proposition 6.1.5. Since $\max_{Int^{CL}} D_{min} < \min_{Int^{\star}} D_{min}$, intervals satisfying the above condition, say $\tilde{Int}$, can belong only to, (i) $Int^{\star}$, and (ii) an interval in T depending on the data configuration. ∎

However, unlike the two-cluster problem, such informative intervals $\tilde{Int}$ do not provide the cluster assignments of $\mathbb{X}$ directly. This is due to the inherent limitation of linear classifiers which, at the most, can separate only a pair of classes. Figures 6.1 and 6.2 illustrate this contrast, where although the interval a_1 on L_1, and the interval in A_p pertaining to M_S realize a $SI^{\star}$, only the latter interval could provide the cluster assignments. Methods by which the information contained in $\tilde{Int}$ can be modeled to estimate the cluster assignments is the focus of the following section. At that point, we will also relax our assumption of requiring linear separability of clusters in their input space, and address the issue of outliers in data.

6.2 A Maximum-margin clustering algorithm

Determining the minimum value of $D^{\star}_{min}$ and the corresponding lower bound of $[SI]_{D^{\star}_{min}}$, in an unsupervised setting, would require identifying a line perpendicular to separating hyperplane with the *least margin*. This is an ill-posed problem because the notion of $D^{\star}_{min}$ and $[SI]_{D^{\star}_{min}}$ are relative with respect to the data configuration. Further, this process would *ideally* necessitate an analysis of projections of $\mathbb{X}$ on all possible lines, and is therefore computationally intensive. Hence, we evaluate the *probability of presence* of $\tilde{Int}$ between *all pair of points* in $\mathbb{X}$ using Proposition 6.1.6, and perform global clustering using it to obtain the cluster assignments.

Since an interval belonging to $\tilde{Int}$ will have a D_{min} (and the corresponding $[SI]_{D_{min}}$) greater than that for all intervals within a cluster, we define a pair-wise similarity measure,

$$f(x_i, x_j) = \exp(-\max_{\mathcal{D}:Int_{ij}} \mathcal{D}[SI]_{\mathcal{D}}) \tag{6.3}$$

which determines how probable is the absence of $\tilde{Int}$ between the points x_i and x_j. Int_{ij} is the line interval between x_i and x_j containing projections of $\mathbb{X}$, from which the bounds for $\mathcal{D}$ are determined to compute (6.3). Since Int_{ij} can contain intervals belonging to both Int^{CL} and $Int^{CL'}$, maximization over $\mathcal{D}$ helps to identify the presence of $\tilde{Int}$ (Proposition 6.1.6). We now make the following observations,

- Maximum value $f(x_i, x_j) = 1$ occurs only when $x_i = x_j$ since, (i) there exist no 'interval' between them ($SI = 0$), and (ii) for any point-pair $(x_i, x_j), j \neq i$, one can always find an infinitesimal interval (up to a discretization error) in Int_{ij} with $D_{min} > 0$, which would make $f(x_i, x_j) < 1$;

- Minimum value $f(x_i, x_j) \approx 0$ occurs only if x_i and x_j belong to different clusters, and Int_{ij} is perpendicular to the hyperplane that separates x_i and x_j, i.e., $Int_{ij} \in Int^{\star} \subset \tilde{Int}$. Such cases will have a large $\max_{\mathcal{D}} \mathcal{D}[SI]_{\mathcal{D}}$, and from previous discussions, this value will be much higher than those when x_i and x_j belong to same cluster.

Essentially, the most significant edges connecting nodes from different clusters are those with least weights, $f(x_i, x_j) \approx 0$, which need to be 'cut' in order to obtain the cluster assignments. We use normalized cuts [171] for this purpose. Details are presented in Algorithm 5. Since we restrict our analysis to Int_{ij} between a pair of

points (instead of using L_p in $\mathbb{R}^N$), we examine the fraction of such 'meaningful' edges obtained from each x_i in the Appendix.

Input: Set $\mathbb{X}$ of M unlabelled points $\{x_i\}_{i=1}^{M} \in \mathbb{R}^N$, and number of clusters $k(> 0)$.

Output: The cluster assignments $y_i \in \{1, 2, ..., k\}, \forall i = 1 \; to \; M$, providing the maximum separating margin (Proposition 6.1.6). Do:

1. Compute projections of $\mathbb{X}$ on the set of line intervals between all possible points pairs, $Int_{ij} : (x_i, x_j), \forall 1 \leq i, j \leq M$.
2. Compute a symmetric $M \times M$ similarity matrix $S^\star$, with its entries $f(x_i, x_j), \forall 1 \leq i, j \leq M$ obtained from (6.3).
3. Perform normalized cuts (NCut) [171]; $y = NCut(S^\star, k)$ to obtain the cluster assignments.

Algorithm 5: Maximum-margin clustering algorithm.

6.2.1 Design Issues

Computing f: Since we use an exponential function to compute (6.3), we first normalize SI and d_{i_p} with the maximum distance between two data points, and the maximum value of d_{i_p} across projections of $\mathbb{X}$ on lines between all pairs of points, respectively. Then while evaluating (6.3), we need to account for the possibility of existence of no x_{i_p} in a small interval ($SI \approx 0$, and/or $SI < \gamma$) within a cluster. Such a condition results in $D_{min} = \infty$, which makes $f = 0$. To avoid such instances, we place an upper bound on the maximum value of D_{min}: $\bar{D}_{min} = n_1 \; {}^*(\max_{Int \in L_p} d_{i_p}), n_1 > 1$. Since we normalize d_{ip}, $\bar{D}_{min} = n_1$, and we chose $n_1 = 7$ for our experiments. This choice would make the minimum value of $f \approx 10^{-3}$,

when the corresponding (normalized) $SI \approx 1$. From previous results, the instance with $SI \approx 1$ and $D_{min} = n_1$ will happen only when x_i and x_j belong to different clusters.

When data is not linearly separable: Since the basic information needed to compute (6.3) comes from x_{i_p} and d_{i_p}, and these computations involve dot products, we accommodate non-linearly separable data using the kernel trick [5].

Effect of outliers: The choice of normalized cuts to perform clustering based on (6.3) is primarily to obtain balanced clusters, which offers some resistance to outliers. Hence, our method is less prone to the presence of *isolated points* belonging to a cluster. An illustration is given in Figure 6.5. However, unlike outlier-robust *supervised* max-margin classifiers that use slack variables (eg. [35]), we cannot deal with conditions where a point belonging to cluster 1 is present inside cluster 2, and both clusters are well-balanced.

Computational complexity: Obtaining the projections of $\mathbb{X}$ on line segments between all pairs of points has a cost of $O(M)$ for each line segment, and $O(M^2)$ for all point-pairs, thereby yielding a total cost of $O(M^3)$. To compute f for a point-pair (6.3) with this information, we need to analyze the maximum distance between adjacent x_{i_p}'s (to compute $[SI]_{\mathcal{D}}$) for a maximum of M possible values of d_{i_p}. However, we discretized the d_{i_p} values into five equal intervals between 0 and 1. Hence, this stage has a cost of $O(M \log M)$ to sort x_{i_p}'s between a point-pair, and when performed for all point pairs incurs a cost of $O(M^3 \log M)$. These two stages, though, permit parallelization to improve efficiency. We then perform normalized cuts, which involves eigen-decomposition with M nodes, and thereby

has a maximum cost of $O(M^3)$. Hence, the overall computational complexity of our method is $O(M^3 \log M)$, which is slightly more than that of normalized cuts.

6.3 Experiments

We performed experiments both on synthetic, and real data to evaluate our method. In all these experiments, we used the following set of kernels: linear, polynomial, RBF (radial basis function), and sigmoid. We then chose the kernel with least Ncut cost (Algorithm 5) to determine the cluster assignments. These results are then matched with the ground truth to compute the clustering accuracy. On the whole, we saw an improvement in clustering accuracy of about 6% on average, and up to a maximum 15% using our method on several synthetic, and real datasets. For cases where we did not perform the best, we were outperformed by an average of about 1% and a maximum of 3.5%.

6.3.1 Synthetic data

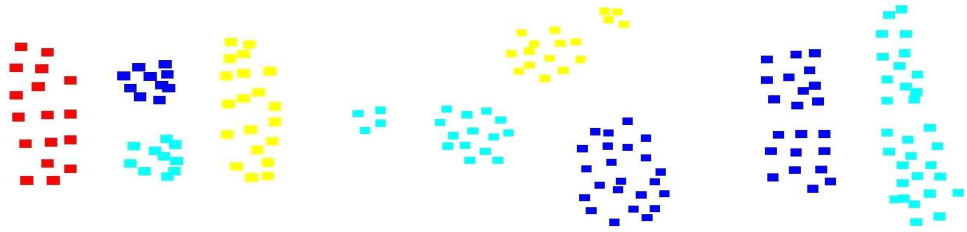
We experimented with synthetic data[6] containing multiple clusters (with maximum $k = 10$), and with cases where the clusters are not linearly separable in their input space. We generated 100 synthetic data, where the first set of 50 samples had outliers, and the second set of remaining samples had no outliers. The outlier instances were not restricted to cases with isolated points, which normalized cuts can handle relatively well. We tested the sensitivity of our algorithm to outliers, and

[6] www.umiacs.umd.edu/users/raghuram/Datasets/MaxMarginClustering.zip

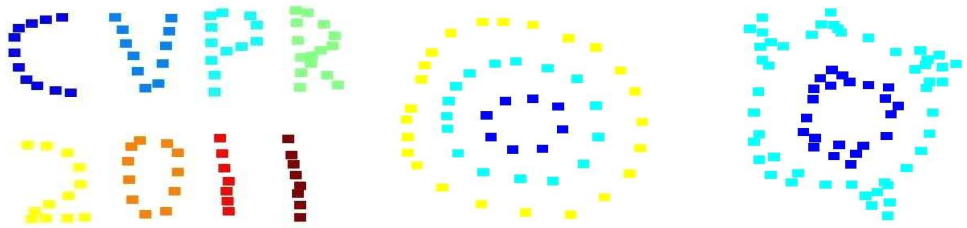
to the absence of the exact value of k (we ran the algorithm for $2 \leq k < 10$) on this dataset and present the results of clustering accuracy in Table 6.1(a). Some clustering results using our method are shown in Figures 6.5 and 6.6, where $\mathbb{X} \in \mathbb{R}^2$, and $\mathbb{X} \in \mathbb{R}^3$. We also show some results using K-means (KM) [121] and using normalized cuts (NC) [171] with the pairwise-similarity measure $f'(x_i, x_j) = \exp(-\|x_i - x_j\|_2^2/\sigma)$, to illustrate the sensitivity of these algorithms to cluster-center initialization, and the value of σ, respectively. Hence, one advantage of our approach is its reduced dependence on parameter tuning. The results of KM and NC, for each kernel parameter setting (and number of clusters), were averaged over 50 trials, and different values of σ (set by a exhaustive search over the distance between all point-pairs in the data) respectively, and the mean and standard deviation of clustering accuracy for the best parameter set are reported. With an improvement of around 9% in accuracy, our method has better tolerance to outliers. It also shows much better performance when the exact value of k is unknown.

6.3.2 Comparison with existing methods on real data

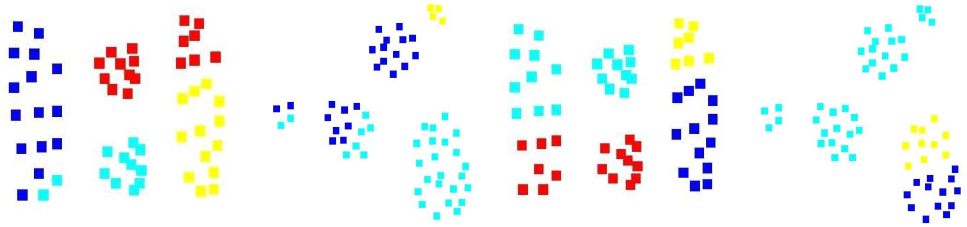
We then evaluated our method on the experimental setup of Wang et al. [192] that performs maximum margin clustering by optimizing over all possible cluster assignments, and that of Ye et al. [211] which performs discriminative clustering by integrating linear discriminant analysis-based dimensionality reduction and K-means clustering. The datasets belonged to the UCI repository [73], text data



(a)



(b)



(c)

Figure 6.5: Clustering results on synthetic data $\mathbb{X} \in \mathbb{R}^2$. (a),(b): Results using our method showing robustness to outliers, and in characterizing margin properties. (c): the first two figures shows sample mis-clustering result from KM, and the last two from NC - to illustrate the sensitivity of these algorithms to cluster center initialization, and parameter tuning respectively. (Data magnified by a factor of 5.)

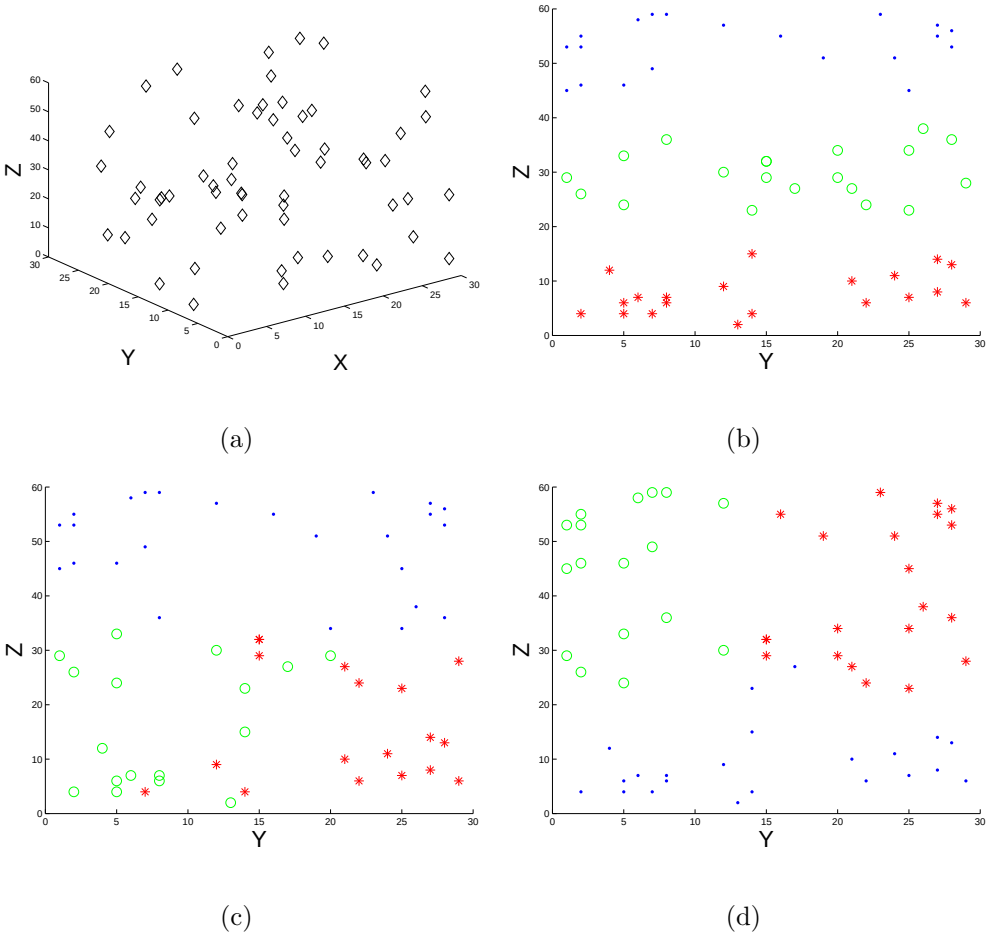


Figure 6.6: Clustering results on synthetic data $\mathbb{X} \in \mathbb{R}^3$. (a) original data. Data is randomly distributed in x- and y- directions from 0 to 30, and has three splits in the z- direction: 0 to 15, 23 to 38, and 45 to 60. Clustering results are shown in the y-z plane. (b) our method, (c) KM, (d) NC.

(a)

Data	KM [121]	NC [171]	Ours
set 1, k known	72.2±3.41	67.5±4.22	**76.5**
set 1 & 2, k known	76.6±2.50	77.1±3.02	**85.5**
set 2, k unknown	70.9±3.28	78.55±2.31	**92.3**
set 1 & 2, k unknown	58.78±3.96	61.43±4.55	**78.1**

(b)

Data	KM [121]	NC [171]	MMC [205]	GMMC [185]	IterSVR [215]	CPMMC [192]	Ours
UCI-Iono.	54.28	75.00	78.75	76.50	77.70	75.48	**86.17**
UCI-Let.	82.06	76.80	-	-	92.80	95.02	**97.25**
UCI-Sat.	95.93	95.79	-	-	96.82	**98.79**	98.35
Text-1	50.53	93.79	-	-	97.02	95.00	**98.56**
Text-2	50.38	91.35	-	-	93.99	**97.21**	96.98
Digits 3-8	94.68	65.00	90.00	94.40	96.64	96.88	**97.33**
Digits 1-7	94.45	55.00	68.75	97.8	99.45	**100.0**	**100.0**
Digits 2-7	96.91	66.00	98.75	99.50	**100.0**	**100.0**	**100.0**
Digits 8-9	90.68	52.00	96.25	84.00	96.33	98.12	**99.56**
UCI-Digit	96.38	97.57	-	-	98.18	99.40	**99.52**
MNIST	89.21	89.92	-	-	92.41	96.21	**98.55**

(c)

Data	KM [121]	NC [171]	CPM3C [192]	Ours
UCI-digits 0689	42.23	93.13	**96.74**	96.11
UCI-digits 1279	40.42	90.11	94.52	**96.54**
USPS	92.15	92.81	95.03	**97.11**
Cora-DS	28.24	36.88	44.15	**56.31**
Cora-HA	34.02	42.00	59.80	**69.86**
Cora-ML 3-8	27.08	31.05	**45.49**	42.33
Cora-OS 1-7	23.87	23.03	59.16	**75.87**
Cora-PL 2-7	33.80	33.97	47.21	**53.33**
WebKB-Corn. 8-9	55.71	61.43	72.05	**73.11**
WebKB-Texas	45.05	35.38	69.10	**75.60**
WebKB-Wash.	53.52	32.85	78.17	**82.43**
WebKB-Wisc.	49.53	33.31	74.25	**79.23**
20-newsgroup	35.27	41.89	71.34	**71.44**
Reuters-RCVI	27.05	-	62.35	**72.81**

(d)

Data	DisKmeans [51] (max,mean)	DisCluster [211] (max,mean)	LLE [156]	LEI [19]	Ours
banding	(77.1,76.8)	(77.1,76.7)	64.8	76.4	**83.2**
soybean	(64.1,63.4)	(63.3,63.2)	63.0	64.9	**68.7**
segment	(68.7,66.4)	(67.6,67.2)	59.4	66.3	**73.1**
pendigits	(69.9,69.0)	(69.6,69.0)	59.9	69.7	**70.1**
satimage	**(70.1,65.1)**	(65.4,64.2)	62.7	66.3	69.5
leukemia	**(77.5,76.3)**	(73.8,73.8)	71.4	68.6	77.4
ORL	(74.4,73.8)	(73.9,73.8)	73.3	31.7	**79.1**
USPS	(71.2,62.8)	(69.2,68.3)	63.1	70.0	**75.3**

Table 6.1: (a) Clustering accuracy (in %) on a synthetic dataset of around 100 samples with $X \in \mathbb{R}^2$ and $X \in \mathbb{R}^3$.(b),(c) Comparison with max-margin clustering methods. Clustering accuracy (in %) for, (b): two-cluster problems, and (c): multi-cluster problems. (d) Comparison with methods that integrate dimensionality reduction and clustering on multi-cluster problem.

(20-newsgroup[7], WebKB[8], Cora [126] and RCVI [112]), digits data (USPS[9], and MNIST [106]), and ORL face dataset[10]. The results of clustering accuracy comparison with max-margin clustering methods are given in Tables 6.1(b) and 6.1(c), and the comparison with the discriminative clustering methods is given in Table 6.1(d). It can be seen that our method compares favorably with other methods on many datasets, offering an overall improvement of 3 to 4%.

6.3.3 Experiments on vision problems

6.3.3.1 Face recognition across lighting variations

We used the YaleB dataset [80] and CMU-PIE Illumination dataset [173]. The YaleB dataset has images of 38 subjects under 64 different lighting conditions, and the PIE dataset has 68 subjects with 21 lighting conditions. No other facial variations such as pose, alignment etc. were present. The images were resized to 48×40, and the gradient orientation information was computed at each pixel. This feature, which was shown to be robust against lighting changes [40], was vectorized to constitute x_i's. Clustering was then performed using normalized cuts on the pair-wise information f (6.3).

[7] people.csail.mit.edu/jrennie/20Newsgroups/

[8] www.cs.cmu.edu/∼WebKB/

[9] www.kernel-machines.org/data.html

[10] www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html

6.3.3.2 2D Shape matching

We used the MPEG-7 shape retrieval dataset [105] and an articulation dataset [82]. The MPEG-7 dataset contains 70 classes of shapes with 20 instances per class containing general shape deformations. The articulation dataset contains 5 classes, with 10 shapes per class, where the main source of variation is non-planar articulations. The underlying shape representation was a shape context descriptor invariant to non-planar articulations. For each aligned shape (2D silhouettes), 100 points were sampled uniformly along the contour, and a log-polar histogram was associated with each point using the method of [82]. We used 5 radial bins, and 12 angular bins resulting in a 100×60 shape descriptor for each shape. The vectorized form of this descriptor represents $x_i \in \mathbb{X}$, using which clustering is done.

We compared our method with Zhang et al. [215], and Ye et al. [211], and the results are given in Table 6.2. We used the publicly available source code for [215][11], implemented [211], and verified results on the datasets used for this work. These results, with roughly a 7% improvement, demonstrate potential applications of our method towards unsupervised pattern discovery in vision problems. As a final note, although it is desirable to have a good grouping mechanism, we would like to emphasize the equally important component of 'data representation' on which we operate on.

[11] www.cse.ust.hk/∼twinsen

Data	DisKmeans [51]	IterSVR [215]	Ours
Face-YaleB	65.6	68.1	**77.4**
Face-PIE	69.2	71.0	**79.5**
Shape-MPEG7	55.9	51.2	**59.3**
Shape-Articulation	42.3	38.5	**51.4**

Table 6.2: Clustering accuracy (in %) on datasets for face recognition across lighting condition, and shape matching. As before, the result for our method correspond to kernel parameters with least NCut cost, whereas for the other two methods we report the maximum clustering accuracy.

6.4 Discussion

We addressed the problem of obtaining clusters with maximum separating margins, by studying the pattern of projections of points on all possible lines in the data space. By drawing parallels with supervised max-margin classification, we derived properties that projections on a line interval would satisfy *if and only if* that interval lies outside a cluster, under assumptions on linear separability of clusters and absence of outliers. We then proposed a pair-wise similarity measure to model this information to perform clustering, by accommodating non-linearly separable data using kernel methods, and (partially) handling outliers by placing emphasis on the cluster size. The experiments illustrated the utility of our method when applied to standard datasets, and to problems in computer vision.

6.5 Appendix: On detecting margins with a restricted analysis on line intervals between all pairs of points

We analyze the consequence of a restricted analysis on line intervals between a pair of points in $\mathbb{X}$, rather than all L_p in $\mathbb{R}^N$. Let $Int_{ij}^{full} \subset L_p$ denote line intervals between all pair of points $(x_i, x_j) \in \mathbb{X}$ containing the projections of $\mathbb{X}$. Let $Int_{ij}^{full} = Int_{ij}^1 \cup Int_{ij}^2$ comprise of two disjoint sets that denote line intervals between a pair of points belonging to same, and different clusters, respectively. From Proposition 6.1.6, only those intervals in $Int_{ij}^2 \perp S_i$, where S_i is the hyperplane separating the pair of points connected by Int_{ij}^2, can have a $D_{min} \geq D_{min}^{\star}$. Let us now analyze the possibility of obtaining such intervals in Int_{ij}^2.

Let $0 < \theta \leq 90°$ denote the angle[12] between an interval Int_{ij}^2 with its corresponding S_i. Let us analyze the distribution of θ tended by the set of all lines joining a point x_i to all points x_j belonging to a different cluster. Since this is a data-dependent analysis, without the loss of generality, let us assume θ to be uniformly distributed between 0 and (including) 90°. Let us split this into n_2 equally spaced angular bins. Essentially, $\forall x_i \in \mathbb{X}$, at least $1/n_2$ of its connections with points x_j in other clusters will *almost surely* realize $D_{min}^{\star}$ (since their $\theta \approxeq 90°$). Hence, for each point, these are the connections that need to be 'cut' in order to group points ($f << 1$).

[12] θ can not equal zero, since the line between a pair of points belonging to different clusters can never be parallel to the hyperplane that separates them.

Chapter 7

Domain Adaptation for Object Recognition: An Unsupervised Approach

As the role of data becomes increasingly important in pattern classification problems, we are often confronted with situations where the data we have to train a classifier is 'different' from that presented during testing. Of the several schools of thought addressing this problem, two prominent ones are transfer learning (TL) [144], and domain adaptation (DA) [21]. These two strategies primarily differ on the assumptions of 'what' characteristics of data are changing between the training and testing conditions. Specifically, TL addresses the problem where the marginal distribution of the data in the training set X (source domain) and the test set $\tilde{X}$ (target domain) are same, while the conditional distributions of the labels, $P(Y|X)$ and $P(\tilde{Y}|\tilde{X})$ with Y and $\tilde{Y}$ denoting labels in either domain, are different. On the other hand, DA pertains to the case where $P(Y|X) = P(\tilde{Y}|\tilde{X})$, but $P(X) \neq P(\tilde{X})$. This specific scenario occurs very naturally in unconstrained object recognition settings, where a domain shift can be due to change in pose, lighting, blur, resolution, among others, and thereby forms the main focus of this work.

Understanding the effects of domain change is a relatively new topic, which has been receiving substantial attention from the natural language processing community over the last few years (e.g. [21, 26, 50]). Although many fundamental

questions still remain on the assumptions used to quantify a domain shift, there are several practical methods that have demonstrated improved performance under some domain variations. Given lots of labeled samples from the source domain, these methods can be broadly classified into two groups depending on whether the target domain data has some labels or it is completely unlabeled. The former is referred to as semi-supervised DA, while the latter is called unsupervised DA. While semi-supervised DA is generally performed by utilizing the correspondence information obtained from labeled target domain data to learn the domain shifting transformation (e.g. [50]), unsupervised DA is based on the following strategies: (i) imposing certain assumptions on the class of transformations between domains [191], or (ii) assuming the availability of certain discriminative features that are common to both domains [26, 122].

In the context of object recognition, the problem of matching source and target data under some pre-specified transformations has been extensively studied. For instance, given appropriate representation of objects such as contours or appearance information, if it is desired to perform recognition invariant to similarity transformations, one can use Fourier descriptors [213], moment-based descriptors [98] or SIFT features [119]. Whereas in a broader setting where we do not know the exact class of transformations, the problem of addressing domain change has not received significant attention. Some recent efforts focus on semi-supervised DA [161, 6, 104]. However, with the ever-increasing availability of visual data from diverse acquiring devices such as a digital SLR camera or a webcam, and image collections from the internet, it is not always reasonable to assume the availability of labels in all

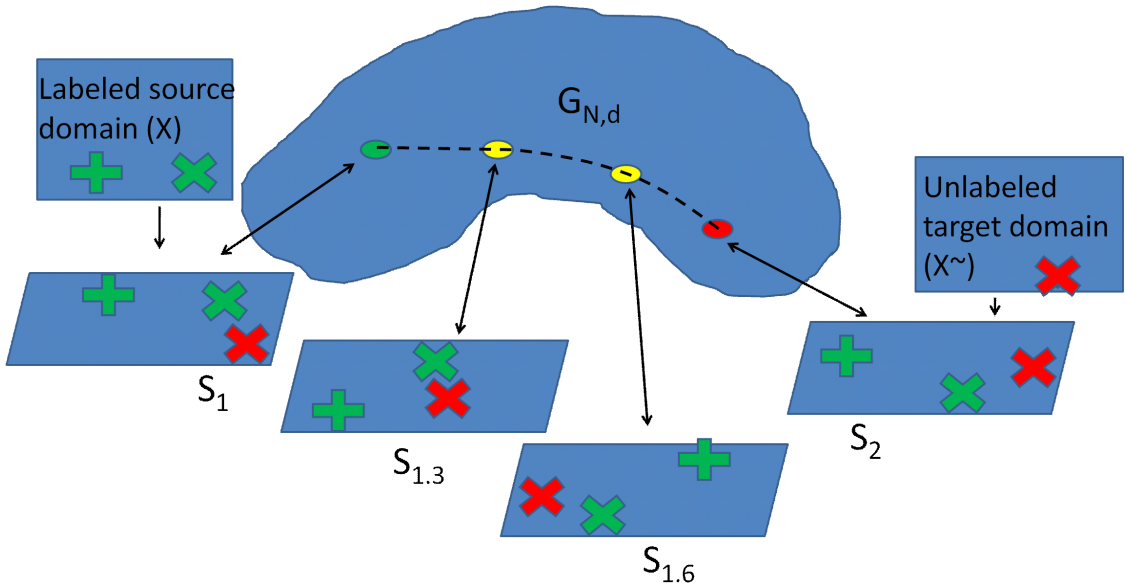


Figure 7.1: Say we are given labeled data X from source domain corresponding to two classes $+$ and $\times$, and unlabeled data $\tilde{X}$ from target domain belonging to class $\times$. Instead of assuming some relevant features or transformations between the domains, we characterize the domain shift between X and $\tilde{X}$ by drawing motivation from incremental learning. By viewing the generative subspaces S_1 and S_2 of the source and target as points on a Grassmann manifold $G_{N,d}$ (green and red dots respectively), we sample points along the geodesic between them (dashed lines) to obtain 'meaningful' intermediate subspaces (yellow dots). We then analyze projections of labeled $\times$, $+$ (green) and unlabeled $\times$ (red) onto these subspaces to perform classification.

domains. Specific example scenarios include, a robot trained on objects in indoor settings with a goal of recognizing them in outdoor unconstrained conditions, or when the user has few labeled data and lots of unlabeled data corresponding to same object categories, where one would want to generalize over all available data without requiring manual effort in labeling. Having said that, unsupervised DA is an inherently hard problem since we do not have any evidence on how the domain change has affected the object categories.

Contributions: Instead of assuming some information on the transformation or features across domains, we propose a data-driven unsupervised approach that is primarily motivated by incremental learning. Since humans adapt (better) between extreme domains if they 'gradually' walk through the path between the domains (e.g. [165, 38]), we propose:

- Representing the generative subspaces of same dimension obtained from X and $\tilde{X}$ as points on the Grassmann manifold, and sample points along the geodesic between the two to obtain intermediate subspace representations that are consistent with the underlying geometry of the space spanned by these subspaces;

- We then utilize the information that these subspaces convey on the labeled X, and learn a discriminative classifier to predict the labels of $\tilde{X}$. Furthermore, we illustrate the capability of our method in handling multiple source and target domains, and in accommodating labeled data in the target, if any.

Organization of the chapter: Section 7.1 reviews related work. Section 7.2

discusses the proposed method. Section 7.3 provides experimental details and comparisons with domain adaptation approaches for object recognition and natural language processing, and the chapter is concluded in Section 7.4. Figure 7.1 illustrates the motivation behind our approach.

7.1 Related Work

One of the earliest works on semi-supervised domain adaptation was performed by Daumé III and Marcu [50] where they model the data distribution corresponding to source and target domains to consist of a common (shared) component and a component that is specific to the individual domains. This was followed by methods that combine co-training and domain adaptation using labels from either domains [182], and semi-supervised variants of the EM algorithm [46], label propagation[204] and SVM [58]. More recently, co-regularization approaches that work on augmented feature space to jointly model source and target domains [49], and transfer component analysis that projects the two domains onto an reproducing kernel Hilbert space to preserve some properties of domain-specific data distributions [143] have been proposed. Under certain assumptions characterizing the domain shift, there have also been theoretical studies on the nature of classification error across new domains [23, 21]. Along similar lines, there have been efforts focusing on domain shift issues for 2D object recognition applications. For instance, Saenko et al [161] proposed a metric learning approach that could use labeled data for few categories from target domain to predict the domain change for unlabeled target categories.

Bergamo and Torresani [6] performed an empirical analysis of several variants of SVM for this problem. Lai and Fox [104] performed object recognition from 3D point clouds by generalizing the small amount of labeled training data onto the pool of weakly labeled data obtained from the internet.

Unsupervised DA, on the other hand, is a harder problem since we do not have any labeled correspondence between the domains to estimate a transformation between them. Differing from the set of many greedy (and clustering-type) solutions for this problem [172, 95, 34], Blitzer et al [28, 27] proposed a structural correspondence learning approach that selects some 'pivot' features that would occur 'frequently' in both domains. Ben-David et al [22] generalized the results of [28] by presenting a theoretical analysis on the feature representation functions that should be used to minimize domain divergence, as well as classification error, under certain domain shift assumptions. More insights along this line of work was provided by [26, 122]. Another related method by Wang and Mahadevan [191] pose this problem in terms of unsupervised manifold alignment, where the manifolds on which the source and target domain lie are aligned by preserving a notion of the 'neighborhood structure' of the data points. All these methods primarily focus on natural language processing. However in visual object recognition, where we have still have relatively less consensus on the basic representation to use for X and $\tilde{X}$, it is unclear how reasonable it is to make *subsequent* assumptions on the relevance of features extracted from X and $\tilde{X}$ [28] and the transformations induced on them [191].

7.2 Proposed Method

7.2.1 Motivation

Unlike existing methods that work with the information conveyed by the source and target domains *alone*, our methodology of addressing domain shift is inspired from incremental learning (that illustrates the benefits of adapting between extremes by gradually following the 'path' between them), and we attempt to identify 'potential' *intermediate domains* between source and target and learn the information they convey on domain change. In quest of these novel domains, (i) we assume that we are given a N-dimensional representation of data from X and $\tilde{X}$, which depends on the user/ application, rather than relying on the existence of pivot features across domains [28], and (ii) we learn the 'path' between these two domains by exploiting the geometry of their underlying space, without making any assumptions on the domain shifting transformation (as in [191]). We now state the problem formally.

7.2.2 Problem Description

Let $X = \{x_i\}_{i=1}^{N_1} \in \mathbb{R}^N$ denote the data from source domain pertaining to M categories or classes. Let $y_i \in \{1, 2, 3, ...M\}$ denote the label of x_i. We assume that the source domain is mostly labeled, i.e. $X = X_l \cup X_u$ where $X_l = \{x_{li}\}_{i=1}^{N_{l1}}$ has labels, say $\{y_{li}\}_{i=1}^{N_{l1}}$, and $X_u = \{x_{ui}\}_{i=1}^{N_{u1}}$ are unlabeled ($N_{l1} + N_{u1} = N_1$). We further assume that all categories have some labeled data. Let $\tilde{X} = \{\tilde{x}_i\}_{i=1}^{N_2} \in \mathbb{R}^N$ denote unlabeled data from the target domain corresponding to the same M categories. Since subspace models are highly prevalent in modeling data characteristics (e.g.

[184]), we work with generative subspaces[1] corresponding to the source and target domain. Let S_1 and S_2 denote generative subspaces of dimension[2] $N \times d$ obtained by performing principal component analysis (PCA) [184] on X and $\tilde{X}$ respectively, where $d < N$. We now address two questions: (i) How to obtain the $N \times d$ intermediate subspaces $S_t, t \in \mathbb{R}, 1 < t < 2$, and (ii) How to utilize the information conveyed by these subspaces on the labeled data X_l to estimate the identity of unlabeled $\tilde{X}$?

7.2.3 Generating Intermediate Subspaces

To obtain meaningful intermediate subspaces between S_1 and S_2, we require a set of tools that are consistent with the geometry of the space spanned by these $N \times d$ subspaces. The space of d-dimensional subspaces in $\mathbb{R}^N$ (containing the origin) can be identified with the Grassmann manifold $\mathbb{G}_{N,d}$. S_1 and S_2 are points on $\mathbb{G}_{N,d}$. Understanding the geometric properties of the Grassmann manifold have been the focus of works like [199, 60, 1], and these have been utilized in some vision problems with subspace constraints, e.g. [183, 84, 120]. A compilation of statistical analysis methods on this manifold can be found in [42]. Since a full-fledged explanation of these methods is beyond the scope of this chapter, we refer the interested readers to the papers mentioned above.

[1]Since we do not have labeled data from target domain, our initial start point will be generative subspaces that characterize the global nature of the domains, rather than discriminative ones.

[2]d refers to the number of eigenvectors of the PCA covariance matrix that have non-zero eigenvalues. We choose the value of d to be minimum of that of S_1 and S_2, and restrict its maximum value to be less than N to enable use of methods that'll be discussed soon. It is interesting to determine a better alternate way of doing this.

- Given two points S_1 and S_2 on the Grassmann manifold.
- Compute the $N \times N$ orthogonal completion Q of S_1.
- Compute the thin CS decomposition of $Q^T S_2$ given by
$$Q^T S_2 = \begin{pmatrix} X_C \\ Y_C \end{pmatrix} = \begin{pmatrix} V_1 & 0 \\ 0 & \tilde{V}_2 \end{pmatrix} \begin{pmatrix} \Gamma(1) \\ -\Sigma(1) \end{pmatrix} V^T$$
- Compute $\{\theta_i\}$ which are given by the arccos and arcsine of the diagonal elements of Γ and Σ respectively, i.e. $\gamma_i = \cos(\theta_i)$, $\sigma_i = \sin(\theta_i)$. Form the diagonal matrix Θ containing θ's as diagonal elements.
- Compute $A = \tilde{V}_2 \Theta V_1$.

Algorithm 6: Numerical computation of the velocity matrix: The inverse exponential map [79].

We now use some of these results pertaining to the geodesic paths, which are constant velocity curves on a manifold, to obtain intermediate subspaces. By viewing $\mathbb{G}_{N,d}$ as a quotient space of $SO(N)$, the geodesic path in $\mathbb{G}_{N,d}$ starting from S_1 is given by a one-parameter exponential flow [79]: $\Psi(t') = Q \exp(t'B)J$, where exp refers to the matrix exponential, and $Q \in SO(N)$ such that $Q^T S_1 = J$ and $J = \begin{bmatrix} I_d \\ 0_{N-d,d} \end{bmatrix}$. I_d is a $d \times d$ identity matrix, and B is a skew-symmetric, block-diagonal matrix of the form $B = \begin{pmatrix} 0 & A^T \\ -A & 0 \end{pmatrix}$, $A \in \mathbb{R}^{(N-d)\times d}$, where the superscript T denotes matrix transpose, and the sub-matrix A specifies the direction and the speed of geodesic flow. Now to obtain the geodesic flow between S_1 and S_2, we compute the direction matrix A such that the geodesic along that direction, while starting from S_1, reaches S_2 in unit time. Computing A is generally achieved

through the inverse exponential map (Algorithm 6). Once we have A, we can use the expression for $\Psi(t')$ to obtain intermediate subspaces between S_1 and S_2 by varying the value of t' between 0 and 1. This is generally performed using the exponential map (Algorithm 7). Let S' refer to the collection of subspaces $S_t, t \in \mathbb{R}, 1 \leq t \leq 2$, which includes S_1, S_2 and all intermediate subspaces. Let N' denote the total number of such subspaces.

- Given a point on the Grassmann manifold S_1 and a tangent vector $B = \begin{pmatrix} 0 & A^T \\ -A & 0 \end{pmatrix}$.
- Compute the $N \times N$ orthogonal completion Q of S_1.
- Compute the compact SVD of the direction matrix $A = \tilde{V_2}\Theta V_1$.
- Compute the diagonal matrices $\Gamma(t')$ and $\Sigma(t')$ such that $\gamma_i(t') = \cos(t'\theta_i)$ and $\sigma_i(t') = \sin(t'\theta_i)$, where θ's are the diagonal elements of Θ.
- Compute $\Psi(t') = Q \begin{pmatrix} V_1\Gamma(t') \\ -\tilde{V_2}\Sigma(t') \end{pmatrix}$, for various values of $t' \in [0, 1]$.

Algorithm 7: Algorithm for computing the exponential map, and sampling along the geodesic [79].

7.2.4 Performing Recognition Under Domain Shift

We now model the information conveyed by S' on X and $\tilde{X}$ to perform recognition across domain change. We basically approach this stage by projecting X and $\tilde{X}$ onto S', and looking for correlations between them (by using the labels available

from X). Let x'_{li} denote the $dN' \times 1$ vector formed by concatenating the projection of x_{li} onto all subspaces contained in S'. We now train a discriminative classifier $D(X'_l, Y'_l)$, where X'_l is the $dN' \times N_{l1}$ data matrix (with $x'_{li}, i = 1\ to\ N_{l1}$ forming the columns), and Y'_l is the corresponding $N_{l1} \times 1$ label vector (whose i^{th} row corresponds to y_{li}), and infer identity of $dN' \times 1$ vectors corresponding to projected target data $\tilde{x}'_i$. We use partial least squares[3] (PLS) [198] to construct D since dN' is generally several magnitudes higher than N_{l1}, in which case PLS provides flexibility in choosing the dimension of the final subspace unlike other discriminant analysis methods such as LDA [18].

7.2.5 Extensions

7.2.5.1 Semi-supervised Domain Adaptation

We now consider cases where there are some labels in the target domain. Let $\tilde{X} = \tilde{X}_l \cup \tilde{X}_u$ where $\tilde{X}_l = \{\tilde{x}_{li}\}_{i=1}^{N_{l2}}$ has labels, say $\{\tilde{y}_{li}\}_{i=1}^{N_{l2}}$, and $\tilde{X}_u = \{\tilde{x}_{ui}\}_{i=1}^{N_{u2}}$ is unlabeled ($N_{l2} + N_{u2} = N_2$). We now use a $dN' \times (N_{l1} + N_{l2})$ data matrix (whose columns correspond to the projections of labeled data from both domains onto S') and the corresponding $(N_{l1} + N_{l2}) \times 1$ label vector to build the classifier D, and infer the labels of $\tilde{x}_{ui}, i = 1\ to\ N_{u2}$.

[3] Alternately, one can choose any other method for the steps involving PCA, and PLS.

7.2.5.2 Adaptation Across Multiple Domains

There can also be scenarios where we have multiple domains in source and target [123, 57]. One way of dealing with k_1 source domains and k_2 target domains is to create generative subspaces $S_{11}, S_{12}, .., S_{1k_1}$ corresponding to the source, and $S_{21}, S_{22}, ..., S_{2k_2}$ for the target. From this we can compute the mean of source subspaces, say $\bar{S}_1$, and the mean for target $\bar{S}_2$. A popular method for defining the mean of points on a manifold was proposed by Karcher [97]. A technique to obtain the Karcher mean is given in Algorithm 8. We then create intermediate subspaces between $\bar{S}_1$ and $\bar{S}_2$, and learn the classifier D to infer target labels as before.

1. Given a set of k points $\{q_i\}$ on the manifold.
2. Let μ_0 be an initial estimate of the Karcher mean, usually obtained by picking one element of $\{q_i\}$ at random. Set j = 0.
3. For each $i = 1, .., k$, compute the inverse exponential map ν_i of q_i about the current estimate of the mean i.e. $\nu_i = \exp_{\mu_j}^{-1}(q_i)$.
4. Compute the average tangent vector $\bar{\nu} = \frac{1}{k}\sum_{i=1}^{k} \nu_i$.
5. If $\|\bar{\nu}\|$ is small, then stop. Else, move μ_j in the average tangent direction using $\mu_{j+1} = \exp_{\mu_j}(\epsilon\bar{\nu})$, where $\epsilon > 0$ is small step size, typically 0.5.
6. Set $j = j + 1$ and return to Step 3. Continue till μ_j does not change anymore or till maximum iterations are exceeded.

Algorithm 8: Algorithm to compute the sample Karcher mean [42].

(a)

Domain		Metric learning [161] (semi-supervised)		Ours	
Source	Target	asymm	symm	Unsupervised	Semi-supervised
webcam	dslr	0.25	0.27	0.19	0.37
dslr	webcam	0.30	0.31	0.26	0.36
amazon	webcam	0.48	0.44	0.39	0.57

(b)

Domain		Metric learning [161] (semi-supervised)		Ours	
Source	Target	asymm	symm	Un-supervised	Semi-supervised
webcam	dslr	0.53	0.49	0.42	0.59

Table 7.1: Comparison of classification performance with [161]. (a) with labels for all target domain categories. (b) with labels only for partial target categories. asymm and symm are two variants proposed by [161].

7.3 Experiments

We first compare our method with existing approaches for 2D object recognition [161, 6], and empirically demonstrate the benefits of creating intermediate domains. In this process, we also test the performance of the semi-supervised extension of our algorithm, and for cases with more than one source or target domains. Finally, we provide comparisons with unsupervised DA approaches on natural language processing tasks.

Figure 7.2: Sample retrieval results from our unsupervised method on the dataset of [161]. Left column: query image from target domain. Columns 2 to 6: Top 5 closest matches from the source domain. Source/ target combination for rows 1 to 4 are as follows: *dslr*/*amazon*, *webcam*/*dslr*, *amazon*/*webcam*, *dslr*/*webcam*.

7.3.1 Comparison with Metric Learning Approach [161]

We used the dataset of [161] that has 31 different object categories collected under three domain settings: images from *amazon*, *dslr camera*, and *webcam*. There are 4652 images in total, with the object types belonging to backpack, bike, notebook, stapler etc. The *amazon* domain has a average of 90 instances for each category, whereas $DSLR$ and *webcam* has roughly around 30 instances for a category. The domain shift is caused by several factors including change in resolution, pose, lighting etc.

We followed the protocol of [161] in extracting image features to represent the objects. We resized all images to 300×300 and converted them into grayscale. SURF features [16] were then extracted, with the blob response threshold set at 1000. The 64-dimensional SURF features were then collected from the images, and a codebook of size 800 was generated by k-means clustering on a random subset of *amazon* database (after vector quantization). Then the images from all domains are represented by a 800 bin histogram corresponding to the codebook. This forms our data representation for X and $\tilde{X}$, with $N = 800$. From this we learn the subspaces corresponding to source and target, and we chose the subspace dimension d to be the lower of the two (and less than N). The value of d was between 185 and 200 for different experiments on this dataset. We experimentally fixed the number of intermediate subspaces to 8 (i.e. $N' = 10$), and the PLS dimensions p to 30.

We report results on two experimental settings, (i) with labeled data available in both source and target domains - 3 labels per category in target for *amazon*/*webcam*/*dslr*,

and 8 per category in source domain for *webcam/dslr*, and 20 for *amazon*; and (ii) labeled data is available in both domains only for the first half of categories, whereas the last 16 categories has labels only in the source domain. For the first setting, we determine the identity of all unlabeled data from target domain, whereas for the second setting we determine the labels of unlabeled target data from the last 16 categories. For both experiments, we report the results of our method in unsupervised setting (where we do not use labels from target, even if available) and semi-supervised setting (where target labels are used) in Tables 7.1(a) and 7.1(b) respectively. The mean performance accuracy (number of correctly classified instances over total test data from target) is reported over 20 different trials corresponding to different labeled data across source and target domains. It can be seen that although our unsupervised adaptation results are slightly lower than that of [161] (which is reasonable since we throw away all correspondence information, while [161] uses them), our semi-supervised extension offers a better improvement. Also note that the result in Table 7.1(b) is better than the corresponding category of Table 7.1(a) since the former is a 16 way classification, while the later is a 31-way classification. Some retrieval results from our unsupervised approach are presented in Figure 7.2.

7.3.2 Comparison with Semi-supervised SVM's [6]

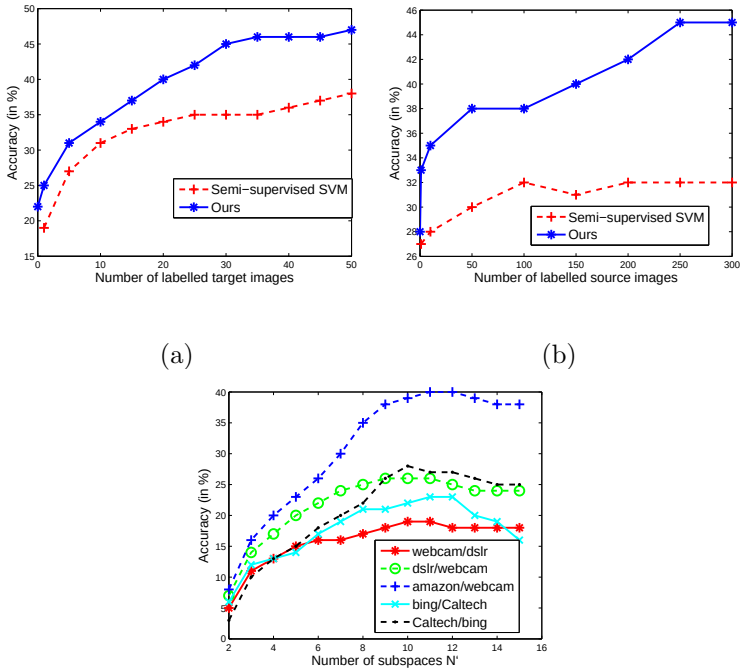
We then used the data of [6] that has two domains: the target domain with images from Caltech256 that has 256 object categories, and the source domain corresponding to the weakly labeled results of those categories obtained from *Bing*

image search. We used the classeme features to represent the images. Each image was represented by a 2625-dimensional binary vector, which models several semantic attributes of the image [6]. We followed the protocol of [6] and present results on classifying the unlabelled target data under two experimental settings, (i) by fixing the number of labeled samples from source domain and varying the labeled samples from target (starting from one), and (ii) doing the reverse by fixing the number of labeled target data, and varying the labeled samples from source. We also consider another operating point of no labeled data from target and source domain respectively (corresponding to the above two settings) to perform unsupervised DA. It can be seen from Figures 7.3(a) and 7.3(b) that our method has gives better performance overall, with the gain in accuracy improving with the number of labeled data. The performance is measured using the percentage of correctly classified unlabeled samples from the target, averaged across several trials on choosing different labeled samples.

7.3.3 Studying the information conveyed by intermediate subspaces, and multi-domain adaptation

We now empirically study the information we gain by creating the intermediate domains. We use the data of [161, 6] where we evaluate the performance of our algorithm (unsupervised case) across different values[4] of N' ranging from 2 to 15. The same experimental setup of Sec 7.3.1 and 7.3.2 was followed. $N' = 2$ denotes

[4] All these runs correspond to $p = 30$, which was empirically found to give the best performance.



(a) (b)

(c)

Figure 7.3: (a), (b): Performance comparison with [6]. (a) Number of labeled source data = 300. (b) Number of labeled target data = 10. Semi-supervised SVM refers to the top performing SVM variant proposed in [6]. *Please note that our method also has an unsupervised working point (at position 0 on the horizontal axis).* (c) Studying the effect of N' on data from [161, 6]. Naming pattern refers to source domain/ target domain. Accuracy for $N' > 2$ is more than that for $N' = 2$, which says that the intermediate subspces do provide some useful information.

no intermediate subspace, and we use the information conveyed by S_1 and S_2 alone. This provides a baseline for our method. As seen in Figure 7.3(c), all values of $N' > 2$ offers better performance than $N' = 2$. Although this result is data-dependent, we see that we gain some information from these new domains.

We then experimented with the data of [161] when there are multiple domains in source or target. We created six different possibilities, three cases with two source domain and one target domain, and the other three with two target domains and one source domain. The experimental setup outlined in Sec 7.3.1 was followed, where we consider the case with labels for all target categories. We provide the mean classification accuracy of our unsupervised and semi-supervised variants in Table 7.2. Although we do not have a baseline to compare with, one possible relation with the results in Table 7.1(a) is for the case where target domain is *webcam* and source domains contain *dslr* and *amazon*. It can be seen that the joint source adaptation results lie somewhere in between those single source domain cases.

7.3.4 Comparison with unsupervised approaches on non-visual domain data

We now compare our approach with other unsupervised DA approaches that are proposed for natural language processing tasks. We used the dataset of [27] that performs adaptation for *sentiment classification*. The dataset has product reviews from amazon.com for four different domains: books, DVD, electronics and kitchen appliances. Each review has a rating from 0 to 5, a reviewer name and location,

Domain		Ours	
Source	Target	Un-supervised	Semi-supervised
amazon, dslr	webcam	0.31	0.52
amazon, webcam	dslr	0.25	0.39
dslr, webcam	amazon	0.15	0.28
webcam	amazon, dslr	0.28	0.42
dslr	amazon, webcam	0.35	0.46
amazon	dslr, webcam	0.22	0.32

Table 7.2: Performance comparison across multiple domains in source or target, using the data from [161].

Domain		Method		
Target	Source	[28]	[27]	Ours
B	(D,E,K)	(.768,.754,.661)	(.797,.754,.686)	(.782,.763.742)
D	(B,E,K)	(.74,.743,.754)	(.758,.762,.769)	(.761,.758,.791)
E	(B,D,K)	(.775,.741,.837)	(.759,.741,.868)	(.812,.762,.876)
K	(B,D,E)	(.787,.794,.844)	(.789,.814,.859)	(.781,.820,.897)

Table 7.3: Performance comparison with some unsupervised DA approaches on language processing tasks [27]. Key: B-books, D-DVD, E-electronics, and K-kitchen appliances. Each row corresponds to a target domain, and three separate source domains.

review text, among others. Reviews with rating more than 3 were classified as positive, and those less than 3 were classified negative. The goal here is to see how the positive/ negative reviews learned from one domain, is applicable to another domain. We followed the experiment setup of [27], where the data representation for X and $\tilde{X}$ are unigram and bigram features extracted from the reviews. Each domain had 1000 positive and negative examples each, and the data for each domain was split into a training set (source domain) of 1600 instances and test set (target domain, with hidden labels) of 400 instances. We now report the classification accuracies with different settings of source and target domain in Table 7.3. We can see that our method performs better overall, even though we do not identify pivot features from the bigram/unigram data features (as done by the other two methods). This experiment also illustrates the utility of our method for domain adaptation across general, non-visual domains.

7.4 Discussion

We have proposed a data driven approach for unsupervised domain adaptation, by drawing motivation from incremental learning. Differing from the existing methods that make assumptions on transformations or feature distributions across domains, we illustrated the benefits of creating intermediate domains to account for the unknown domain shift. Despite creating these new domains using tools that respect the underlying geometry of data, we acknowledge the challenges posed by lack of correspondence across domains in explaining how well these domains correlate

with the ‘real’ domain change. In summary, although we do not consider to have solved the problem of unsupervised DA by any means, we have offered a principled alternate methodology that relaxes some assumptions made by existing methods.

Chapter 8

A Computationally Efficient Method for Contour-based Object Detection

Detecting objects in images using their contour information is a common problem in computer vision. It is generally used as a preprocessing step to localize potential regions pertaining to the object, before analyzing those regions using more detailed descriptors. Edges are perhaps the most informative low-level image features that give a good estimate of contours in images. Characterizing edges, hence, is an important aspect of this problem, and the traditional approaches have followed a two-step process: (i) building a set of shape primitives representing the object's contour under different deformations, say to detect a football, generate circles (or ellipses) of different scales (and orientations) to account for the distance (and the projective) effect of the imaging process; and then (ii) given a new image, looking for the object in it by first computing its edge map to get approximate information of the contours, and then obtaining a matching score for the shape primitives at different regions in the edge image. The regions with higher matching score imply a higher probability of presence of the object at that location.

Related Work: There are various contour-based shape matching algorithms proposed in the literature. In this work, we mainly focus on approaches that perform matching using the 'actual' contour information rather than computing a descriptor

based on the contour to handle shape deformations (e.g. [20]). In other words, given a set of contour primitives representing the object of interest, we are interested in finding regions in image containing the object. Methods in this category include the Hough transform ([52, 13, 100]) that maps the detection process from the image space onto an accumulator space spanned by all possible parameters of the object contour, where the points of local maxima correspond to the contour parameters of the object present in the edge image. On the other hand, correlation-based matching (e.g. [86, 66]) fits different shape contours to the edge image, and then compute the matching score by summing up the edge pixels underlying them. The regions with large correlation values potentially correspond to the object of interest. Since these methods are not robust to clutter, Chamfer matching-based algorithms (e.g. [14, 29]) together with distance transform have been popular in detecting occluded objects. However, they have some drawbacks, such as requiring much more training samples than the other two classes of approaches.

In all these methods, a critical step is the analysis of edge strength between a pair of points in the image. This information, computed between all possible point-pairs, is dependent on the number of intermediate points connecting the points of interest thereby making it computationally demanding. In this work, we address this problem by proposing a representation of the edge image using which the edge strengths can be computed in $O(1)$ operations, irrespective of the distance between the points. We approximate the object contour using line segments and compute the line integral image I_l, which is the cumulative sum of pixels of the edge image along different possible line orientations. This preprocessing step, motivated in

part by the integral image computations of a rectangular region [189], is done only once per image along different line orientations determined by the desired spatial resolution. I_l can then be used to analyze the edge strengths between any pair of contour points, across translations, rotations and scale variations of the object, in just $O(1)$ operations thereby resulting in huge computational savings. We motivate our work through a face detection application, by considering a correlation-based algorithm [129] that detects face contours using ellipses. We approximate the face contours using hexagons, and compare the computational savings obtained from our proposed approach in different stages of the matching algorithm. For the task of frontal face detection, across scale variations, we obtain a reduction in the overall computational complexity of [129] from quadratic to linear in time with respect to the number of contour primitives used for detection. We then improve the detection accuracy of [129] by analyzing regions pertaining to the hexagonal contour using a combination of three existing appearance-based descriptors. Specifically, we use the color information [89], histogram of oriented gradients [47], and eigenfaces [184] in a support vector machine framework [140] to obtain good detection results on the widely tested CMU+MIT dataset [166] on both frontal and profile faces.

Organization of the Chapter: We first motivate the need for a computationally efficient image representation for computing edge strengths between a pair of points through a face detection application in Section 8.1, and discuss its computational complexity in Section 8.1.1. We then present our proposed line integral image representation I_l and compare its computational efficiency across different stages of correlation-based matching in Section 8.2. The generalizability of I_l in detecting

contours of arbitrary objects, across different matching algorithms, is discussed in Section 8.3. Section 8.4 has details of experiments that study the computational efficiency of our method in detecting faces, and other arbitrary objects. We also discuss the face detection accuracy using the proposed fusion approach, by analyzing the regions pertaining to face contours with a combination of three appearance-based descriptors. The chapter is concluded in Section 8.5.

8.1 Baseline Face Detection Algorithm - A Brief Overview

We consider the problem of detecting faces using contour information to motivate the need for an efficient representation of edge image. We restrict our analysis to the task of frontal face detection using correlation-based contour matching. We first overview the baseline algorithm [129] and its computational stages in obtaining the edge map and matching contours.

The baseline face detection algorithm is a feature-based approach that employs an optimal step-edge operator to detect shapes. The core formulation of the algorithm is to detect 2D shapes by analyzing the intensity differentials across the object boundary. This method of detecting a object is in fact a natural extension of edge detection at the pixel level to that of global contour detection. The object boundary is assumed to be piecewise smooth, and the change in the intensity is modeled as a step function. In order to preserve the global step edge structure of the object under the presence of noise, an optimal smoothing filter $\hat{h}$ is first designed using a criterion that minimizes the sum of the noise power, and the mean square

error between the input signal and filter output. To explain this further, consider a step edge with amplitude (α)

$$X(t) = \begin{cases} 0 & t \leq 0 \\ \alpha & t > 0 \end{cases}$$

which is corrupted by additive white gaussian noise $N(t)$. Let the noise corrupted step-edge signal be represented by

$$\hat{X}(t) = X(t) + N(t)$$

We then apply the filter $\hat{h}$ to $\hat{X}(t)$ to obtain

$$\hat{Y}(t) = \hat{h} * \hat{X}(t) = \hat{h} * X(t) + \hat{h} * N(t)$$

where $*$ denotes convolution. Let $Y(t) = \hat{h} * X(t)$, $\hat{M}(t) = \hat{h} * N(t)$, and $\hat{E}(t) = X(t) - Y(t)$. Given this setup, we would like to obtain an optimal smoothing filter $\hat{h}$ that minimizes the squared sum $\hat{E}^2 + \hat{M}^2$, where $\hat{E}^2$ is the mean squared difference between the input signal and the filter output, and $\hat{M}^2$ is the mean squared sum of the output noise response. After some analytical operations, the optimal smoothing filter (8.1) is obtained as,

$$\hat{h}(t) = (d/2)\ {}^{*}exp(-d|t|) \tag{8.1}$$

The 1D smoothing operator is then extended to 2D by the following operation,

$$h(x, y) = \hat{h}((x^2 + y^2)^{1/2}) = (d/2)\ {}^{*}exp(-d|\sqrt{x^2 + y^2}|) \tag{8.2}$$

The optimal step-edge operator h' is then obtained by taking the piecewise derivative of the smoothing filter h following the simple relation between differentiation and

convolution.

$$(h * f)' = h' * f \tag{8.3}$$

where f is the image under interest. This filter h' turns out to be the derivative of double exponential (DODE) function, originally derived by [151]. While using this framework to detect faces present in f, the facial boundary is approximated as an ellipse. The DODE operator [151] is then applied across all possible 'hypothetical' elliptical contours in f to identify regions that might correspond to a face, as explained in the following sub-section.

8.1.1 Computational Complexity of the Baseline Algorithm

From a computational viewpoint, the baseline face detection algorithm can be viewed as a three-step process.

Step 1: The initial step is to compute the edge map of an image f of resolution $V_1^* V_2$, using the DODE filter of size, say $N^* N$. This essentially involves convolving the DODE filter at all image locations Z (where, $Z = V_1^* V_2$, is the number of pixels in f), which has a computational cost of

$$Z\ ^* N^2\ multiplications,\ Z\ ^*(N^2 - 1)\ additions \tag{8.4}$$

Step 2: The next step is to fit an ellipse over the computed edge map to find regions that might resemble a face. Let Z_1 (where, $Z_1 < Z$) denote the number of locations in the image where the ellipse can be placed. Then for every such location, say Z_1', ellipse fitting consists of three steps; (i) placing a hypothetical ellipse centered at Z_1', (ii) multiplying all points on the elliptical contour (say, M) with their counterparts

on the edge map, and (iii) summing up all the product values to get the overall response of the ellipse fit. This process is then repeated at all possible Z_1 locations of the image, which translates to

$$Z_1 \, {}^*M \; multiplications, \; Z_1 \, {}^*(M-1) \; additions \tag{8.5}$$

The response values obtained at all Z_1 locations are subject to a threshold, and those locations with response values more than the threshold signifies a higher likelihood of presence of a face.

Step 3: Then to detect faces of various sizes, the above process is repeated for different sized ellipses, thereby incurring similar computational cost, as (8.5), for each elliptical size. Specifically, if Z_i is the number of locations in the image where the ith ellipse can be placed, and if M_i is the number of points in its contour, then the cost of this process is given by

$$\sum_i (Z_i \, {}^*M_i) \; multiplications, \; \sum_i (Z_i \, {}^*(M_i - 1)) \; additions \tag{8.6}$$

Given the computational requirements of the baseline face detection algorithm in (8.4), (8.5), and (8.6), we now describe the proposed image representation in Section 8.2 and compare its computational savings with different stages of the corresponding baseline counterparts.

8.2 The Line Integral Image Representation

By approximating the object contour using line segments, we propose the line integral image I_l, represented by the cumulative sum of edge pixels along different

line orientations, as a preprocessing step to perform efficient computation of edge strengths between a pair of points. For detecting faces, we approximate the face contour using hexagons and perform detection by correlation-based matching. We first explain a simple strategy to perform efficient convolution for a class of symmetric filters, and then compare the construction of our image representation in stages that parallel the ones for the baseline algorithm discussed in previous section.

8.2.1 Speed-up 1

We first utilize the structure of the DODE filter to reduce the amount of computations in obtaining the edge map of the image. It can be easily seen that the DODE filter h', obtained by taking piecewise derivatives of h (8.2), is an odd-symmetric function. We use this property to reduce the computations required to perform convolution in the discrete domain. To begin with, consider a DODE filter of size N^*N as shown in Fig 8.1(i). Then while doing convolutions, instead of multiplying the entire N^*N filter coefficients with the underlying image values, we create a matrix K of size $(N/2)^*(N/2)$ containing the filter coefficients of the first quadrant alone. We then utilize the odd-symmetric property of the DODE filter by adding (and subtracting) the underlying image values (according to their quadrant position) and then multiplying them with the co-efficients stored in K. This essentially reduces the number of multiplications by substituting them with equivalent but less computationally intensive additions. The amount of computations required is given by (8.7), and it can be readily compared with its baseline algorithm counterpart

(8.4).

$$Z\ ^*(N^2/4)\ multiplications,\ Z\ ^*(3N^2/4 + N^2/4 - 1)\ additions \tag{8.7}$$

8.2.2 Speed-up 2

After obtaining the edge map of the image in a computationally efficient way, we experimented with replacing ellipses by hexagons to identify regions that probably contain faces. Hexagons, besides being a good approximation to ellipses, are more structured with the presence of six distinct vertices (labeled a-f, in Fig 8.1(ii)) that provide a good representation of the contour. The intuition behind this is: when fitted with hexagons, each point in the underlying edge map can lie on one of the three types of hexagonal edges that a vertex point can support namely, the rising edge (RE), the falling edge (FE), or the straight edge (SE) in Fig 8.1(ii). Hence the points on the hexagonal contour are more tractable than in the case of ellipses, where this kind of a structure is not present.

To illustrate this approach further, we shall consider a hexagon with a base size (say, L) for all its sides. We then pre-compute the edge strengths of all the Z image locations in three directions (RE, FE, SE), each of length L, by summing up their values in the edge image. Then to fit a hexagon at a particular image location, we first determine which points on the edge image would represent its vertices, and then sum their edge strengths (of the appropriate direction) to compute the overall response for the hexagonal fit. We then repeat this process at all possible Z_1 image locations. This way of computing the hexagonal response values requires the

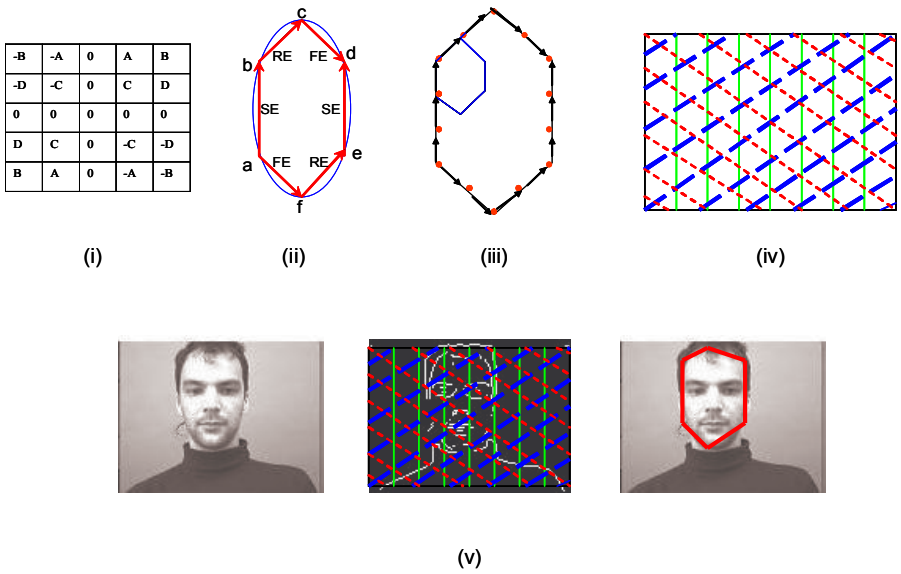


Figure 8.1: Speed-up methods: (i) A 5*5 DODE filter, (ii) Replacing ellipse by hexagon, (iii) Reusing hexagon values, (iv) The proposed line integral image I_l for three orientations of the hexagon - RE(Rising Edge - blue, long dashed lines in bold), FE(Falling Edge - red, dashed lines), SE(Straight Edge - green, solid lines). Each location on the lines denote the cumulative sum of pixels at that point, along the specified direction, (v) Pipeline for detecting frontal faces using the proposed method: Input image, preprocessing to compute I_l by overlaying three line orientations on the edge image, and detected face represented by a hexagon.

knowledge of only six vertex points at a given time, when compared with all the M contour points as in fitting an ellipse. This results in replacing the multiplications during the fitting stage with less computationally intensive additions, and does not add too much overhead to the number of additions in pre-computations either. Let the number of line segments needed to represent the contour be $\tilde{M}$, with D distinct line orientations (for a hexagonal representation of frontal faces, $\tilde{M} = 6$, $D = 3$). The computational cost of this method is given below,

$$\#Pre - computations : Z\ ^{*}(D(L-1))\ additions;$$

$$\#Computations\ for\ fitting : Z_1\ ^{*}(\tilde{M}-1)\ additions \tag{8.8}$$

It can be seen that (8.8) provides significant computational savings when compared to the complexity of the baseline algorithm given in (8.5).

8.2.3 Speed-up 3

Next, we fit different-sized hexagons to account for the possible variations in the size of faces present in images. We approach this stage by representing a larger sized hexagon, as an ordered collection of oriented base-size hexagonal vertices along its contour, as shown in Fig 8.1(iii). This representation enables us to re-use the edge strength values (of appropriate direction) of the vertex points computed for the base-size hexagon, rather than re-computing the entire contour edge strength every time as in the case of fitting ellipses. For instance, let the length of the side j ($j = 1\ to\ 6$) of the larger sized hexagon be represented by some integral multiple S

of the base-size hexagonal side L. Then, the computational cost for this speed-up approach during fitting is given by

$$\sum_i (Z_i \, {}^*((\sum_{j=1}^{\tilde{M}} S_{ij}) - 1)) \; additions \tag{8.9}$$

where the index i corresponds to the total number of hexagons used for fitting. When the hexagon reuse is not used, the number of computations that would result by sequential application of the algorithm until speed-up 2 is given by

$$\sum_i (Z_i \, {}^*(\tilde{M} - 1) + DZ \, {}^* \sum_k (S_k {}^* L - 1)) \; additions \tag{8.10}$$

where the index k refers to the number of different side lengths of the hexagon i. It can be easily seen that the computational cost after speed-up 3 (8.9), is significantly lower when compared to the cost without hexagonal reuse (8.10), and also with that of the corresponding baseline algorithm (8.6).

8.2.4 Speed-up 4: The Line Integral Image

Although the method discussed in Section 8.2.3 can be used for any polygon, one limitation is that the different-sized polygons must have their side lengths as some integral multiple S of the base side length L. To overcome this requirement, we propose an image representation I_l to characterize the edge strength of object contours represented by line segments, inspired in part, from the integral image representation for regions (f_{int}) introduced by Viola and Jones [189].

We first briefly review f_{int}. Given a $V_1^* V_2$ intensity image f, the integral image f_{int} at location (x, y) contains the sum of pixels above and to the left of

(x, y), inclusive:

$$f_{int}(x, y) = \sum_{x' \leq x, y' \leq y} f(x', y') \tag{8.11}$$

Using the pair of recurrences:

$$s_c(x, y) = s_c(x, y - 1) + f(x, y) \tag{8.12}$$

$$f_{int}(x, y) = f_{int}(x - 1, y) + s_c(x, y) \tag{8.13}$$

where $s_c(x, y)$ is the cumulative row sum, $s_c(x, -1) = 0$, and $f_{int}(-1, y) = 0$, the integral image f_{int} can be computed in one pass over the original image f. Such a representation can be readily used to compute the sum of the image values under any rectangular region, by doing just one round of computation over the entire image.

In our case, however, we are interested in the sum of edge pixels along the contour of an object. Towards this end, given a 2D rectangular lattice corresponding to the edge map of an image f and a set of D orientations corresponding to the sides of the polygon, we rotate the lattice into each of the D_i, $i = 1, ..., D$ orientation to compute the line integral image $I_l = \{s_{D_i}(x, y)\}_{i=1}^{D}$, $\forall (x, y) \in f$, which is the set of:

$$s_{D_i}(x, y) = s_{D_i}(x, y - 1) + f_{D_i}(x, y) \tag{8.14}$$

where $s_{D_i}(x, y)$ is the cumulative row sum of the edge image f_{D_i} at the pixel location (x, y) for the line orientation D_i. An illustration of I_l for a hexagon (with $D = 3$) is given in Figure 8.1(iv). With this set of s_{D_i}, $i = 1, ..., D$, in order to fit a polygon to an image, we first determine the set of $\tilde{M}$ vertices of the polygon and their corresponding orientations. Then for each pair of vertices, say (x_1, y_1) and (x_2, y_2), we compute the edge strength es between them *in just O(1) operations* using the

following formula,

$$es = abs(s_{D_j}(x_1, y_1) - s_{D_j}(x_2, y_2)) \tag{8.15}$$

where $s_{D_j}(x_i, y_i)$ is the cumulative row sum of the point (x_i, y_i) at the orientation D_j of the side connecting them. For a particular polygonal fit at the location Z'_1, es will be computed at all $\tilde{M}$ edges of the polygons to obtain the sum of edges along the contour.

For the case of detecting frontal faces using hexagons (where $\tilde{M}$=6), this translates into

$$\#Pre - computations : ZD\ additions;$$

$$\#Computations\ for\ fitting : \sum_i Z_i\ {}^*(2\tilde{M} - 1) additions \tag{8.16}$$

where Z_i refers to the number of location in which the ith hexagon needs to be fitted. This set of computations (8.16) is independent of the length of the polygonal side in the fitting stage, as opposed to speed-up 3 (8.9), thereby resulting in a considerable reduction in the number of additions. Further, it requires at most the same number of additions during pre-computation for the case when the base length $L = 2$, and reduced number of additions for any $L > 2$. A pipeline illustrating the proposed method is given in Figure 1(v).

The overall computational gains obtained from the strategies discussed in Sec 8.2.1 through Sec 8.2.4 can be visualized in Figure 8.2. The line integral representation (8.14) has translated a quadratic dependency in the relationship between the computational time with the number of ellipses (of the baseline algorithm), into a linear dependency. This significant gain in the computation time proves very handy

for real-time applications. Also, this representation can be readily extended to detect any general object contour approximated by line segments, as discussed in the next section.

8.3 Generalizability of the Line Integral Image Representation in Detecting Arbitrary Objects

Let us now analyze the information conveyed by (8.14) in more detail. The main claim behind this representation is to spend more resources in preprocessing by computing the line integrals, so that the computations during the fitting stage can be considerably reduced. We now qualitatively study the complexity of contour fitting process. Assume an arbitrary contour C' to be detected in an image f containing Z pixels. Let P' be the set of shape primitives representing C' under different deformations such as, translation, rotation, scaling and shear. To find the region corresponding to C' from the edge map of f, irrespective of the matching process like correlation or the Hough transform, one needs to estimate the possibility of each of primitives $P'_i \in P'$ at all possible locations in f. This basically involves computing the edge strengths corresponding to each P'_i with the underlying edge map of f. Let N_1 denote the total number of fitting operations needed for all shape primitives P' at all possible locations in f. This requires $O(N_1 x)$ computations, where x is a variable that can range from 1 up to the maximum number of intermediate points of the primitives P'. This is a computationally intensive process.

To circumvent this, our proposed line integral representation I_l (8.14) prepro-

cesses the edge image of f by computing the cumulative sum of pixels along different possible line orientations D. Since a line segment has two degrees of freedom, the intervals between two orientations in D can be chosen according to the desired spatial resolution. This one-time operation has a complexity of $O(ZD)$. Hence during fitting, we could accomplish the task of detecting *any* linearly approximated object contour in just $O(N_1)$ operations, without the need to re-compute the edge strengths corresponding to different P_i' all over again. This results in a tremendous decrease in computations, since ZD is generally several orders of magnitude lesser than N_1x. This is mainly because the preprocessing allows us to reuse the edge strength values, for instance, while fitting a straight line of same orientation but with different lengths.

Although one drawback of (8.14) is the assumption of a linear approximation of the contour C', the line integral representation can be extended to general polynomials. However, this might increase the number of preprocessing computations, and hence a balance needs to be established between the accuracy of contour representation and the computational efficiency. This tradeoff is dependent on the application. Sample examples of detecting arbitrary shapes using the representation in (8.14) are given in Figure 8.6.

8.4 Performance Analysis

In this section we discuss the details of experiments involving the computational requirements of the proposed representation I_l, and the face detection

accuracy obtained after analyzing regions pertaining to the hexagonal prior with appearance-based descriptors. We also show some examples of detecting arbitrary objects using I_l (8.14), under a linear approximation of their contour.

8.4.1 Computational Efficiency - Detecting the Facial Contour

On the effect of filter size N: We first experimentally evaluate the computational efficiency of the proposed speed-up algorithms presented in Section 8.2. We ran two experiments, one with the DODE filter of N=5 and the other with N=10. For both experiments we used ten different ellipses/hexagons to detect the facial contour, with the hexagonal sides being some integer multiple of the base side length L (for these two experiments, $L = 3$). This, although not necessary for speed-up 4, was done to compare all speed-up algorithms in a common benchmark. We repeated each experiment twenty times to compute the average time required for each of the speed-up method to perform detection. Standard image size of 240*320 was used, and the results are given in Figure 8.2. It can be seen that when the filter size N increases, the computational savings obtained from speed-up 1 (8.7) reduce and the amount of computations becomes almost equal to that of the baseline algorithm (8.4). This is because, although (8.7) reduces the number of multiplications, it results in an increase in the memory requirements to store the first quadrant coefficients of the filter in the matrix K. This speed-up step, though, is specific to the DODE filter [151] and the set of symmetric filters.

We then compare the remaining three speed-up methods that are generalizable

to any closed polygon of different side lengths and orientations. We notice that the speed-up methods 2 to 4 results in a substantial improvement in computational efficiency when compared with the baseline algorithm. Specifically, speed-up 4 (8.16) performs best by translating a quadratic dependency in the computational time with the number of shape primitives of the baseline algorithm to that of a linear dependency. This is mainly because speed-up 4 is not dependent on the length of the hexagonal sides, whereas speed-up 3 (8.9) requires more additions with since the number of vertex points M to represent the hexagonal contour increases with increasing side lengths. Speed-up 2 (8.10) performs poorly because it has to recompute the edge strengths of all the image points for different sizes of the shape primitive.

On the effect of varying base length L of the hexagon: Further, when increasing the length of the base side of an hexagon L, we see that speed-up 4 offers much better computational savings that speed-up 3. We did experiments using DODE filter with $N = 10$, and three different values for $L = 3, 5, 7$. This behavior is because speed-up 4 (8.16) does not depend on L during pre-computations, whereas speed-up 3 (8.9) does. This is in addition to the flexibility of speed-up 4 not requiring the hexagonal sides to be an integral multiple of L. Hence speed-up 4 (8.16), obtained from the line integral representation I_l (8.14), can be used as a preprocessing stage for any object detection algorithm, where the object's contour can be used as a prior for localization.

On the effect of varying image size f: We then experimented with changing the size of the input image f to observe variations in computational gains. As

explained by the equations in Section 8.2, the amount of computations is approximately a linear function of the image size (through the number of pixels in f, Z, and the number of possible locations for the contour fit, Z_1). We observed almost a linear dependency in the computations gains obtained from speed-up 4 with the increasing size of f. This is illustrated for three image sizes (240*320, 480*640, and 720*960) in Figure 8.2.

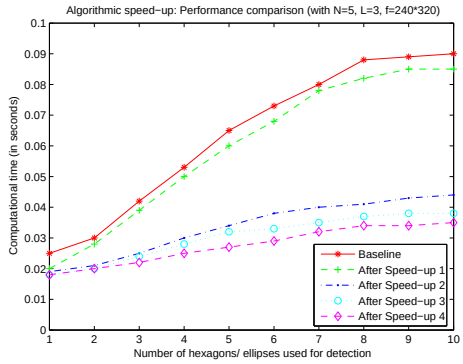
8.4.2 Face Detection Accuracy

Let $\{R_i\}_{i=1}^{N_S}$ denote the regions with high value of correlation when fitted with hexagonal primitives using (8.14). We now discuss the analysis of these regions with a combination of appearance-based descriptors.

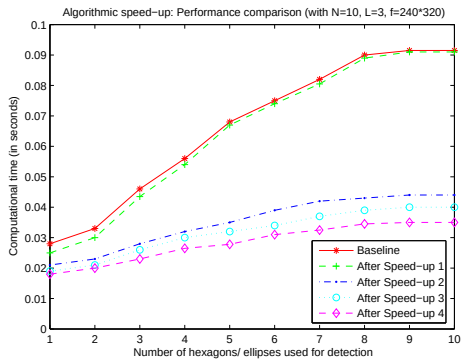
8.4.2.1 Feature Selection for Face Detection

We perform a face-adaptive post-processing on the intensity images of $\{R_i\}_{i=1}^{N_s}$ to verify if they correspond to a face. This is required because many objects other than the face can also have similar contours, for instance a football. Therefore we use a set of existing descriptors to characterize the appearance of faces and non-faces. Specifically, we compute three different cues - color [89], histogram of oriented gradients [47], and eigenfaces [184], and combine these feature channels using support vector machines [140] to identify which of the R_i pertain to a face. A block diagram of the proposed face detection system is given in Figure 8.3.

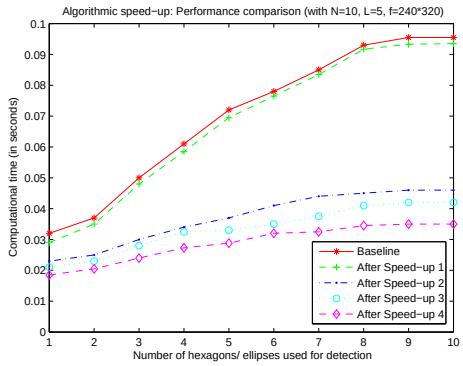
The motivation behind the choice of these descriptors is: (i) the human face



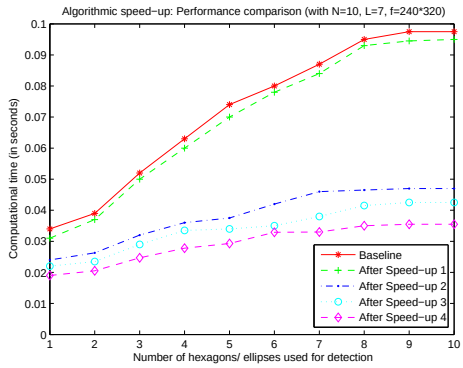
(a) L=3, N=5, f=240*320



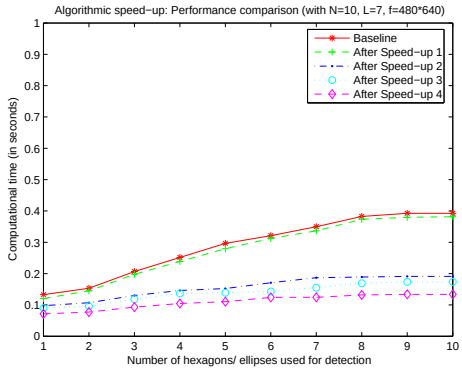
(b) L=3, N=10, f=240*320



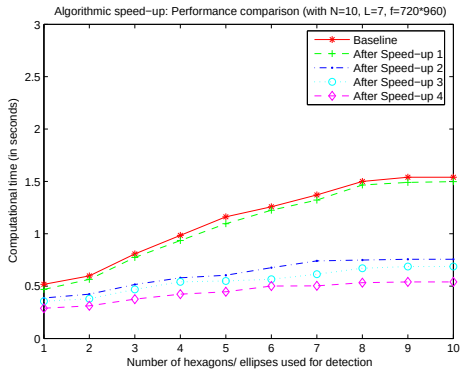
(c) L=5, N=10, f=240*320



(d) L=7, N=10, f=240*320



(e) L=7, N=10, f=480*640



(f) L=7, N=10, f=720*960

Figure 8.2: Comparison of the speed-up methods discussed in Section 8.2 by varying the hexagonal base length L, the filter size N, and the image size f. Speed-up 4 (8.16) results in substantial decrease in the computational requirements of the baseline algorithm [129], and is not affected much by the varying the parameters for polygon fitting. I_l, therefore, reduces the edge strength computations, without bringing a heavy overload from preprocessing. *Please note different y-axis scales for graphs in the last row.*

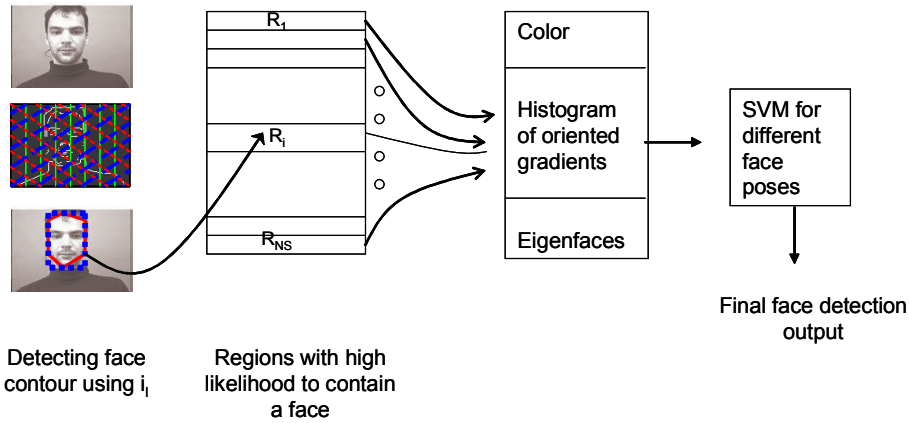


Figure 8.3: The proposed face detection system. Contour detection using I_l, followed by analyzing $R_i's$ by combining three appearance based descriptors using Support vector machines.

has a distinct color pattern which can be characterized by fitting Gaussian models for the color pattern of face regions and non-face regions; (ii) the histogram of oriented gradients captures the high interest areas in faces that have rich gradient information (such as, eyes, nose, and mouth), and provides reasonable robustness to lighting variations [40], and (iii) Eigenfaces captures the holistic appearance of the human face. These three feature channels represent a mix of global and local information about the face. We then assign the facial pose into three categories, one for frontal poses with the maximum in-plane and out-of-plane rotations lying between -20 degrees and +20 degrees, and two for profile views with rotations ranging between 20 degrees and 90 degrees, and the other from -20 degrees to -90 degrees. A separate SVM is trained for each of these cases with around 100 samples each for face and non-face class. We then normalize the test regions $\{R_i\}_{i=1}^{N_S}$, detected by constructing I_l with 10 equally spaced orientations D_i spanning the 0 to 180 degree

range, to a pre-specified size of 30*30 and concatenate their three appearance-based descriptions into a long vector to give as the input to the SVM. These regions are then classified as a face if at least one of the three SVM's gives high probability for the presence of a face.

8.4.2.2 Experimental Setting

We then tested this framework on the standard CMU+MIT face dataset [166] by first identifying regions corresponding to the hexagonal contour primitive, and then perform postprocessing on those regions. The first part of the CMU+MIT dataset (referred as dataset A in Fig 8.4) has 125 frontal face images with 483 labeled faces, the second part (dataset B) has 208 images containing 441 faces of both frontal and profile views. Since this dataset has only grayscale images, the color channel was not used for the post-processing discussed in Section 8.4.2.1. The three SVM'S described in Section 8.4.2.1, trained for different ranges of face pose, were applied on regions $\{R_i\}_{i=1}^{N_S}$ having high correlation-based matching scores with the shape primitives as determined in Section 8.2. The SVM results are then subjected to non-maximum suppression [134] to unify overlapping detection results to obtain the ROC curves given in Fig 8.4. These results are comparable with the existing approaches (e.g. [207], [166], [189], [157], [201]) as shown in Table 8.1. Only few operating points on the ROC are given in the table, since most of the existing approaches do not provide the full ROC.

Discussion: It is interesting to see that our algorithm performs almost equally

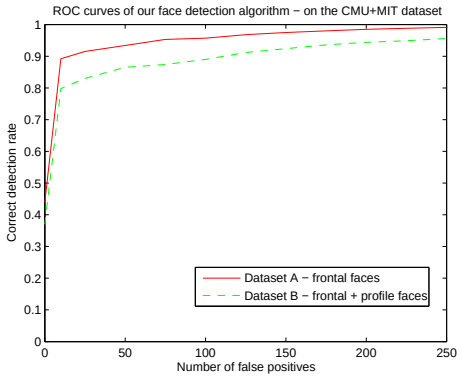


Figure 8.4: ROC curves of our face detection algorithm on the CMU+MIT dataset [166] on both frontal and profile faces.

Figure 8.5: Sample face detection results on the internally collected maritime face dataset

well on the dataset with profile views, given that we do postprocessing only on regions pertaining to shape primitives as determined in Section 8.2.4. This can be due to two reasons, (i) the DODE operator captures edge information very effectively, and (ii) the hexagonal fitting is relatively less-affected by changes in the face pose. This is an advantage of our approach. This, when coupled with algorithmic speed-up techniques (figure 8.2) that reduce computational complexity of the baseline face detection algorithm from quadratic to linear in time, is very useful for robust, real-time applications. The overall processing speed of our face detection algorithm is 30 frames per second on a standard 2 GHz processor. We present some sample face detection results in Fig 8.5 from an internally collected dataset that contains images taken at long distances.

8.4.3 Detecting Arbitrary Object Contours using the Line Integral Image Representation

We now present some examples of detecting arbitrary objects using I_l. We took some example images from the ETHZ dataset [65] that contains five different object categories under considerable clutter. We used the edge images provided in the dataset, and constructed a shape primitive through a linear approximation of the object contour. We then computed I_l from the edge image, along 10 orientations D_i equally spaced in the 0 to 180 degree interval. Using correlation-based matching, we present the detection results in Figure 8.6. Though only the external contours of the object have been used in the shape primitive, one can use the internal contours

Algorithm (with the correct detection rate in %)	Number of false positives			
	10	31	57	95
Ours	91.2%	92.6%	95.1%	95.8%
Viola Jones [189]	78.3%	85.2%	-	90.8%
Rowley et al [157]	83.2%	86.0%	-	89.2%
Wu et al [201]	90.1%	-	94.5%	95%

Algorithm (with the correct detection rate in %)	Number of false positives			
	8	12	34	91
Ours	79.8%	81.3%	85.2%	88.7%
Schneiderman et al [166]	-	75.2%	-	85.5%
Wu et al [201]	79.4%	-	84.8%	-

Table 8.1: Face detection - Experimental results on CMU+MIT dataset. (Top) Dataset A, (Bottom) Dataset B

if needed. The main point we would like to stress from this experiment is that I_l (8.14), in addition to *significantly* reducing the number of operations in computing the edge strengths, can be used for detecting *arbitrary* objects. Such a representation of edge strength values between all point-pairs can also be used with other matching algorithms for localizing objects.

We then analyze the computational requirements for this experiment by com-

paring correlation-based matching using I_l (8.14) with, correlation-based matching without I_l (8.6) and the generalized Hough transform [13]. We used a subset of five images (shown in Figure 8.6) obtained from the ETHZ dataset [65]. The shape primitives corresponding to all five objects were used for detection in the five images. The experiment was repeated ten times, and the mean and standard deviation of processing times for different matching methods is provided in Table 8.2. It can be seen that preprocessing using I_l (8.14) substantially reduces the computational complexity.

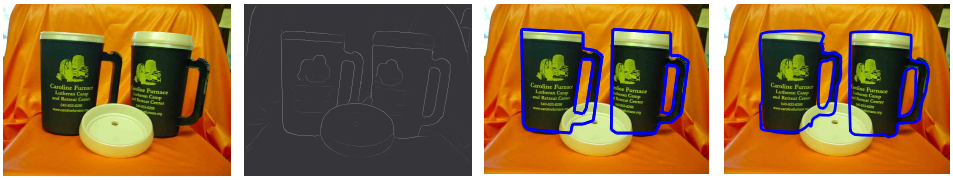
Algorithm	Computational time in seconds (mean±standard deviation)
Correlation-based matching (8.6)	0.51±0.067
Generalized Hough transform [13]	0.46±0.057
Using I_l (8.14)	0.115±0.021

Table 8.2: Computational requirements for different matching methods on the subset of five images from the ETHZ dataset [65].

8.5 Discussion

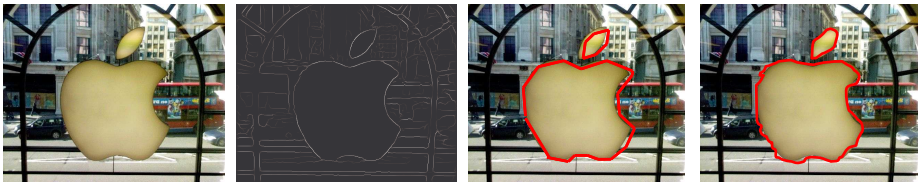
We have proposed an image representation, the line integral image, using which the edge strengths between any pair of points can be computed in just $O(1)$ operations. We showed the generalizability of this representation for efficient detection of arbitrary objects under a linear approximation of the object contour. Specifically, we illustrated its utility for contour-based face detection by approximating the face

contour using hexagons, and achieved a reduction in computational complexity from quadratic to linear in time with respect to the number of contour primitives used for detection. We then proposed a combination of three appearance-based features to analyze regions pertaining to the facial contour and obtained good face detection performance on standard datasets.



(a)

(b)



(c)

(d)

(e)

Figure 8.6: Detection results on ETHZ shape dataset [65]. Column orderings: (i) original image, (ii) edge image from the dataset, (iii) localization result (in blue/red) using I_l, and (iv) ground truth localization.

Chapter 9

Future Work

In this chapter we outline potential directions in which the problems addressed by this dissertation can be explored further.

9.1 Representing and matching non-planar shapes invariant to articulations

The method outlined in Chapter 2 compensates for non-planar articulations by 'explicitly' performing affine normalization of object parts. Another way to address this problem is to 'implicitly' generate an affine shape space [183] corresponding to each convex part, by sampling equal number of points on the contour of each part, and then forming the product manifold [60] of the affine subspaces corresponding to all parts of an articulating shape. Matching two shapes across articulations can then be seen as a problem of comparing product manifolds corresponding to those two shapes, which implicitly contain the set of all possible part-wise affine transformations of the shape. Besides the 'representation' aspect, the effects of articulations can also be addressed in the 'matching' stage. Given atleast a pair of shapes, with the corresponding descriptors generated from Chapter 2, it is interesting to see how the 'error' in shape matching correlates with the model assumptions for articulations. A particularly interesting case is to account for self-occlusions, where pixels

in two shapes that have been occluded can be thought of as imposing a structured noise in the matching error between the shapes. In other words, the manner in which the occluding pixels affect the shape descriptor of unoccluded pixels must be geometrically meaningful. This problem can in principle be formulated as a 'dirty paper coding' setting [44] where we have some prior on the noise in the channel (i.e. how pixel occlusions can affect the shape description on other points), and this can be used to explain shape matching errors due to self-occlusions.

9.2 Unconstrained face recognition using subspace representations

In Chapter 3 we showed the utility of subspace representations to match faces across arbitrary blur. To extend this line of work to recognize faces under unconstrained settings, one need to account for other facial variations such as pose, lighting etc. Linear subspace models for variations in pose [107], lighting [15], and registration [120] have been well studied in the literature. These data-driven/ model-driven subspaces when combined with the blur subspace can be thought of as a tensor (e.g. [186]) that intrinsically represents a face across all those variations. During recognition, a probe image can then be matched with tensors corresponding to gallery faces to determine its identity.

9.3 Alternate strategies for max-margin clustering

The max-margin clustering algorithm proposed in Chapter 6 has a complexity that is cubic in terms of the number of data points. There are atleast two possible

ways to perform efficient clustering. Firstly, the idea of core sets [12] can be used to examine a limited subset of points that need to be picked to obtain a good estimate of margin regions, rather than analyzing line intervals between all pair of points. The convergence bounds on identifying margins, atleast for a two cluster problem, are similar to those results established for a supervised two class problem [154, 75]. Secondly, instead of using the results derived in Chapter 6 to obtain a 'pair-wise similarity measure', there could be other ways of understanding the information between the projected points rather than computing just the location and distance of their projections.

9.4 Encoding transformation priors for unsupervised domain adaptation

In generating intermediate domains in Chapter 7 to understand the unknown domain shift, we use the geodesic between the two domains as a 'possible' path to traverse on. However, since we do not have correspondence between domains, and we have no knowledge about the physical transformation across domains, this path need not correspond to the 'actual' domain shift. In such cases, if we have some prior on the possible domain transformations such as, changes in pose, lighting, blur etc for a face recognition setting, we could use that knowledge to generate 'more meaningful' intermediate domains. The problem of choosing a path between the source and target domains then becomes a shortest path problem on the Grassmann manifold, with the domains as nodes and edges containing information on the 'cost'

of travelling between the two nodes. This cost would represent the effect of the domain shift between the two nodes, using the information conveyed by the prior on the physical domain shift.

Bibliography

[1] P.-A. Absil, R. Mahony, and R. Sepulchre. Riemannian geometry of Grassmann manifolds with a view on algorithmic computation. *Acta Applicandae Mathematicae*, 80:199–220, February 2004.

[2] A. Agrawal and Yi Xu. Coded exposure deblurring: Optimized codes for PSF estimation and invertibility. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2066–2073, Miami, FL, USA, June 2009.

[3] T. Ahonen, A. Hadid, and M. Pietikainen. Face recognition with local binary patterns. In *European Conference on Computer Vision*, pages 469–481, Prague, Czech Republic, May 2004.

[4] T. Ahonen, E. Rahtu, V. Ojansivu, and J. Heikkila. Recognition of blurred faces using local phase quantization. In *International Conference on Pattern Recognition*, pages 1–4, Tampa, FL, USA, December 2008.

[5] M. A. Aizerman, E. M. Braverman, and L. I. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, June 1964.

[6] B. Alessandro and T. Lorenzo. Exploiting weakly-labeled web images to improve object classification: A domain adaptation approach. In *Advances in Neural Information Processing Systems*, pages 181–189. Vancouver, Canada, December 2010.

[7] E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing multiclass to binary: a unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113–141, December 2000.

[8] M. Aly. Real time detection of lane markers in urban streets. In *IEEE Intelligent Vehicles Symposium*, pages 7–12, Eindhoven, Netherlands, June 2008.

[9] H. C. Andrews and B. R. Hunt. *Digital Image Restoration*. Prentice Hall Signal Processing Series, 1977.

[10] N. Apostoloff and A. Zelinsky. Robust vision based lane tracking using multiple cues and particle filtering. In *IEEE Intelligent Vehicles Symposium*, pages 558–563, Columbus, OH, USA, June 2003.

[11] A. Asuncion and D.J. Newman. UCI machine learning repository, 2010.

[12] M. Badoiu, S. Har-Peled, and P. Indyk. Approximate clustering via coresets. In *ACM Symposium on Theory of Computing*, pages 250–257, Montreal, Canada, May 2002.

[13] D. H. Ballard. Generalizing the Hough transform to detect arbitrary shapes. *Pattern recognition*, 13:111–122, January 1981.

[14] H. G. Barrow, J. M. Tenenbaum, R. C. Bolles, and H. C. Wolf. Parametric correspondence and chamfer matching: two new techniques for image matching. In *International Joint Conference on Artificial Intelligence*, pages 659–667, Cambridge, MA, USA, August 1977.

[15] R. Basri and D. Jacobs. Lambertian reflectance and linear subspaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:218–233, Feb 2003.

[16] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110:346–359, June 2008.

[17] E. Begelfor and M. Werman. Affine invariance revisited. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2087–2094, New York City, NY, USA, June 2006.

[18] P.N. Belhumeur, J.P. Hespanha, and D.J. Kriegman. Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:711 –720, July 1997.

[19] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15:1373–1396, June 2003.

[20] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:509–522, April 2002.

[21] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan. A theory of learning from different domains. *Machine Learning*, 79:151–175, May 2010.

[22] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems*, pages 137–145, Vancouver, Canada, December 2007.

[23] S. Ben-David, T. Luu, T. Lu, and D. Pal. Impossibility theorems for domain adaptation. In *International Conference on Artificial Intelligence and Statistics*, pages 129–136, Sardinia, Italy, May 2010.

[24] A. Ben-Hur, D. Horn, H. T. Siegelmann, and V. Vapnik. Support vector clustering. *Journal of Machine Learning Research*, 2:125–137, December 2001.

[25] M. Bertozzi and A. Broggi. GOLD: a parallel real-time stereo vision system for genericobstacle and lane detection. *IEEE Transactions on Image Processing*, 7:62–81, January 1998.

[26] J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman. Learning bounds for domain adaptation. In *Advances in Neural Information Processing Systems*, pages 129–136, Vancouver, Canada, December 2008.

[27] J. Blitzer, M. Dredze, and F. Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Annual Meeting of the Association for Computational Linguistics*, pages 440–447, Prague, Czech Republic, June 2007.

[28] J. Blitzer, R. McDonald, and F. Pereira. Domain adaptation with structural correspondence learning. In *Empirical Methods in Natural Language Processing*, pages 120–128, Sydney, Australia, July 2006.

[29] G. Borgefors. Hierarchical chamfer matching: A parametric edge matching algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10:849–865, June 1988.

[30] A. Borkar, M. Hayes, M.T. Smith, and S. Pankanti. A layered approach to robust lane detection at night. In *IEEE Workshop on Computational Intelligence in Vehicles and Vehicular Systems*, pages 51–57, Nashville, TN, USA, March 2009.

[31] A. Broggi, A. Cappalunga, C. Caraffi, S. Cattani, S. Ghidoni, P. Grisleri, P. P. Porta, M. Posterli, and P. Zani. Terramax vision at the urban challenge 2007. *IEEE Transactions on Intelligent Transportation Systems*, 11:194 –205, March 2010.

[32] A. M. Bronstein, M. M. Bronstein, A. M. Bruckstein, and R. Kimmel. Matching two-dimensional articulated shapes using generalized multidimensional scaling. In *Articulated Motion and Deformable Objects*, pages 48–57, Mallorca, Spain, July 2006.

[33] A. M. Bronstein, M. M. Bronstein, A. M. Bruckstein, and R. Kimmel. Partial similarity of objects, or how to compare a centaur to a horse. *International Journal on Computer Vision*, 84:163–183, August 2009.

[34] L. Bruzzone and M. Marconcini. Domain adaptation problems: A DASVM classification technique and a circular validation strategy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32:770 –787, May 2010.

[35] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, February 1998.

[36] J. F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:679–698, November 1986.

[37] A. Chakrabarti, T. Zickler, and W. Freeman. Analyzing spatially-varying blur. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2512–1519, San Francisco, CA, USA, June 2010.

[38] S. K. Chalup. Incremental learning in biological and machine learning systems. *International Journal of Neural Systems*, 12:447–466, June 2002.

[39] C. C. Chang and C. J. Lin. LIBSVM: a library for support vector machines, 2001.

[40] H.F. Chen, P.N. Belhumeur, and D.W. Jacobs. In search of illumination invariants. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 254 –261, Hilton Head, SC, USA, June 2000.

[41] H. Y. Cheng, B. S. Jeng, P. T. Tseng, and K. C. Fan. Lane detection with moving vehicles in the traffic scenes. *IEEE Transactions on Intelligent Transportation Systems*, 7:571–582, December 2006.

[42] Y. Chikuse. *Statistics on special manifolds*. Springer Verlag, 2003.

[43] J. M. Clanton, D. M. Bevly, and A. S. Hodel. A low-cost solution for an integrated multisensor lane departure warning system. *IEEE Transactions on Intelligent Transportation Systems*, 10:47–59, March 2009.

[44] M. Costa. Writing on a dirty paper. *IEEE Transactions on Information Theory*, 29:439–441, May 1983.

[45] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, December 2001.

[46] W. Dai, G. R. Xue, Q. Yang, and Y. Yu. Transferring naive Bayes classifiers for text classification. In *National Conference on Artificial Intelligence*, pages 540–543, Vancouver, Canada, July 2007.

[47] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 886–893, San Diego, CA, USA, May 2005.

[48] R. Danescu and S. Nedevschi. Probabilistic lane tracking in difficult road scenarios using stereovision. *IEEE Transactions on Intelligent Transportation Systems*, 10:272–282, June 2009.

[49] H. Daumé III, A. Kumar, and A. Saha. Co-regularization based semi-supervised domain adaptation. In *Advances in Neural Information Processing Systems*, pages 478–486, Vancouver, Canada, December 2010.

[50] H. Daumé III and D. Marcu. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26:101–126, May 2006.

[51] F. De la Torre and T. Kanade. Discriminative cluster analysis. In *International Conference on Machine Learning*, pages 241–248, Pennsylvania, PA, USA, June 2006.

[52] S. R. Deans. Hough transform from the Radon transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3:185–188, February 1981.

[53] A. Demiriz, K. P. Bennett, and J. Shawe-Taylor. Linear programming boosting via column generation. *Machine Learning*, 46:225–254, January 2002.

[54] E. D. Dickmanns and B. D. Mysliwetz. Recursive 3-D road and relative ego-state recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:199–213, February 1992.

[55] T.G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine learning*, 40:139–157, August 2000.

[56] S. K. Divvala, D. Hoiem, J. H. Hays, A. A. Efros, and M. Hebert. An empirical study of context in object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1271–1278, Miami, FL, USA, June 2009.

[57] M. Dredze and K. Crammer. Online methods for multi-domain learning and adaptation. In *Empirical Methods in Natural Language Processing*, pages 689–697, Honolulu, HI, USA, October 2008.

[58] L. Duan, I. W. Tsang, D. Xu, and T. S. Chua. Domain adaptation from multiple sources via auxiliary classifiers. In *International Conference on Machine Learning*, pages 289–296, Montreal, Canada, June 2009.

[59] R. O. Duda, P.E. Hart, and D.G. Stork. *Pattern classification*. Wiley, 2001.

[60] A. Edelman, T.A. Arias, and S.T Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal of Matrix Analysis and Application*, 20:303–353, April 1999.

[61] A. Elad and R. Kimmel. On bending invariant signatures for surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:1285–1295, October 2003.

[62] K. Etemad and R. Chellappa. Discriminant analysis for recognition of human face images. *Journal of the Optical Society of America A*, 14:1724–1733, August 1997.

[63] P.F. Felzenszwalb and J.D. Schwartz. Hierarchical matching of deformable shapes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Minneapolis, MN, USA, June 2007.

[64] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single photograph. *ACM Transactions on Graphics*, 25:787–794., March 2006.

[65] V. Ferrari, T. Tuytelaars, and L. Van Gool. Object detection by contour segment networks. In *European Conference on Computer Vision*, pages 14–28, Graz, Austria, May 2006.

[66] A. Fitzgibbon, M. Pilu, and R. B. Fisher. Direct least square fitting of ellipses. *IEEE Transactions on pattern analysis and machine intelligence*, 21:476–480, May 1999.

[67] J. Flusser, J. Boldys, and B. Zitová. Moment forms invariant to rotation and blur in arbitrary number of dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:234–246, February 2003.

[68] J. Flusser, J. Boldyš, and B. Zitová. Invariants to convolution in arbitrary dimensions. *Journal of Mathematical Imaging and Vision*, 13:101–113, February 2000.

[69] J. Flusser and T. Suk. Degraded image analysis: an invariant approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:590 –603, June 1998.

[70] J. Flusser, T. Suk, and S. Saic. Recognition of images degraded by linear motion blur without restoration. In *Proc. 7th TFCV on Theoretical Foundations of Computer Vision*, pages 37–51, Dagstuhl, Germany, March 1996.

[71] A. Franco and L. Nanni. Fusion of classifiers for illumination robust face recognition. *Expert Systems with Applications*, 36:8946–8954, May 2009.

[72] A.F. Frangi and J. Yang. Two-dimensional PCA: a new approach to appearance-based face representation and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:131–137, January 2004.

[73] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[74] Y. Freund. An adaptive version of the boost by majority algorithm. *Machine learning*, 43:293–318, June 2001.

[75] Y. Freund and R. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37:277–296, March 1999.

[76] Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55:119–139, August 1997.

[77] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *Annals of Statistics*, 38:337–374, April 2000.

[78] Keinosuke Fukunaga. *Introduction to statistical pattern recognition (2nd ed.)*. Academic Press Professional, Inc., 1990.

[79] K.A. Gallivan, A. Srivastava, X. Liu, and P. Van Dooren. Efficient algorithms for inferences on grassmann manifolds. In *IEEE Workshop on Statistical Signal Processing*, pages 315 – 318, St. Louis, Missouri, USA, September 2003.

[80] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:643–660, June 2001.

[81] R. Gopalan, T. Hong, M. Shneier, and R. Chellappa. Video-based Lane Detection using Boosting Principles. *Snowbird Learning*, 2009.

[82] R. Gopalan, P. Turaga, and R. Chellappa. Articulation-invariant representation of non-planar shapes. In *European Conference on Computer Vision*, pages 286–299, Heraklion, Greece, September 2010.

[83] R. Gross and V. Brajovic. An image preprocessing algorithm for illumination invariant face recognition. In *International Conference on Audio- and Video- Based Biometric Person Authentication*, pages 10–18, Guilford, UK, June 2003.

[84] J. Hamm and D. D. Lee. Grassmann discriminant analysis: A unifying view on subspace-based learning. In *International Conference on Machine Learning*, pages 376–383, Helsinki, Finland, July 2008.

[85] P. C. Hansen, J. G. Nagy, and D. P. O'Leary. *Deblurring Images: Matrices, Spectra, and Filtering (Fundamentals of Algorithms)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2006.

[86] R.M. Haralick and L.G. Shapiro. *Computer and robot vision*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1992.

[87] X. He, S. Yan, Y. Hu, P. Niyogi, and H. J. Zhang. Face recognition using laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:328–340, March 2005.

[88] D.D. Hoffman and W. Richards. Parts of recognition. *Cognition*, 18:65–96, December 1983.

[89] R. L. Hsu, M. Abdel-Mottaleb, and A. K. Jain. Face detection in color images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:696–706, May 2002.

[90] H. Hu and G. de Haan. Low cost robust blur estimator. In *International Conference on Image Processing*, pages 617–620, Atlanta, GA, USA, January 2006.

[91] M. Isard and A. Blake. Condensation:conditional density propagation for visual tracking. *International Journal of computer vision*, 29:5–28, August 1998.

[92] B. Jahne. Digital image processing. *Springer-Verlag*, 1995.

[93] A. K. Jain and R. C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

[94] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31:264–323, September 1999.

[95] N. Japkowicz and S. Stephen. The class imbalance problem: A systematic study. *Intelligent Data Analysis*, 6:429–449, October 2002.

[96] D. J. Kang and M. H. Jung. Road lane segmentation using dynamic programming for active safety vehicles. *Pattern Recognition Letters*, 24:3177–3185, December 2003.

[97] H. Karcher. Riemannian center of mass and mollifier smoothing. *Communications on Pure and Applied Mathematics*, 30:509–541, May 1977.

[98] A. Khotanzad and Y. H. Hong. Invariant image recognition by zernike moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:489–497, May 1990.

[99] Z. Kim. Robust lane detection and tracking in challenging scenarios. *IEEE Transactions on Intelligent Transportation Systems*, 9:16–26, March 2008.

[100] N. Kiryati, Y. Eldar, and A. M. Bruckstein. A probabilistic Hough transform. *Pattern Recognition*, 24:303–316, April 1991.

[101] C. Kreucher and S. Lakshmanan. LANA: a lane extraction algorithm that uses frequency domainfeatures. *IEEE Transactions on Robotics and Automation*, 15:343–350, April 1999.

[102] B. V. K. V. Kumar, M. Savvides, and C. Xie. Correlation pattern recognition for face recognition. *Proceedings of the IEEE*, 94:1963–1976, November 2006.

[103] M. Lades, J. C. Vorbruggen, J. Buhmann, J. Lange, C. v. .d. Malsburg, R. P. Wurtz, and W. Konen. Distortion invariant object recognition in dynamic link architecture. *IEEE Transactions on Computer*, 42:300–311, March 1993.

[104] K. Lai and D. Fox. Object recognition in 3D point clouds using Web data and domain adaptation. *The International Journal of Robotics Research*, 29:1019–1028, August 2010.

[105] L. J. Latecki, R. Lakämper, and T. Eckhardt. Shape descriptors for non-rigid shapes with a single closed contour. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 424–429, Hilton Head, SC, USA, June 2000.

[106] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278 –2324, November 1998.

[107] K-C. Lee, J. Ho, M-H. Yang, and D. Kriegman. Video-based face recognition using probabilistic appearance manifolds. In *IEEE Conf. On Computer Vision and Pattern Recognition*, pages 313–320, Madison, WI, USA, June 2003.

[108] K.C. Lee, J. Ho, and D. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:684–698, May 2005.

[109] T. S. Lee. Image representation using 2d gabor wavelets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(10):959 –971, October 1996.

[110] A. Levin, D. Lischinski, and Y. Weiss. A closed-form solution to natural image matting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:228–242, February 2008.

[111] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1964–1971., Miami, FL, USA, June 2009.

[112] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, December 2004.

[113] Q. Li, N. Zheng, and H. Cheng. Springrobot: A prototype autonomous vehicle and its algorithms for lane detection. *IEEE Transactions on Intelligent Transportation Systems*, 5:300–308, December 2004.

[114] J. M. Lien and N. M. Amato. Approximate convex decomposition of polygons. *Computational Geometry: Theory and Applications*, 35:100–123, January 2006.

[115] H. Ling and D. W. Jacobs. Shape classification using the inner-distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29:286–299, February 2007.

[116] H. Ling, S. Soatto, N. Ramanathan, and D. W. Jacobs. A study of face recognition as people age. In *IEEE International Conference on Computer Vision*, pages 1–8, Rio de Janeiro, Brazil, October 2007.

[117] A. Lingas. The power of non-rectilinear holes. In *Colloquium on Automata, Languages and Programming*, pages 369–383, Rouen, France, June 1982.

[118] Y. Liu, L. Si, and J. Carbonell. A new boosting algorithm using input-dependent regularizer. In *International Conference on Machine Learning*, pages 9–17, Washington D.C., USA, August 2003.

[119] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Internationa Journal on Computer Vision*, 60:91–110, November 2004.

[120] Y. M. Lui and J. R. Beveridge. Grassmann registration manifolds for face recognition. In *European Conference on Computer Vision*, pages 44–57, Marseille, France, October 2008.

[121] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, Berkeley, CA, USA, June 1967.

[122] Y. Mansour, M. Mohri, and A. Rostamizadeh. Domain adaptation: Learning bounds and algorithms. *Arxiv preprint arXiv:0902.3430*, 2009.

[123] Y. Mansour, M. Mohri, and A. Rostamizadeh. Domain adaptation with multiple sources. In *Advances in Neural Information Processing Systems*, pages 1041–1048. Vancouver, Canada, December 2009.

[124] D. Mateus, R. P. Horaud, D. Knossow, F. Cuzzolin, and E. Boyer. Articulated shape matching using Laplacian eigenfunctions and unsupervised point registration. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Anchorage, AK, USA, June 2008.

[125] J. C. McCall and M. M. Trivedi. Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation. *IEEE Transactions on Intelligent Transportation Systems*, 7:20–37, March 2006.

[126] A. K. McCallum, K. Nigam, J. Rennie, and K. Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3:127–163, July 2000.

[127] R. Meir and G. Rätsch. An introduction to boosting and leveraging. *Lecture Notes in Computer Science*, 2600:118–183, 2003.

[128] B. Moghaddam, T. Jebara, and A. Pentland. Bayesian face recognition. *Mitsubishi Electric Research Laboratories, TR2000-42*, 2002.

[129] H. Moon, R. Chellappa, and A. Rosenfeld. Optimal edge-based shape detection. *IEEE transactions on Image Processing*, 11:1209–1227, November 2002.

[130] K. R. Muller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12:181–201, March 2001.

[131] L. Nanni and A. Lumini. Wavelet decomposition tree selection for palm and face authentication. *Pattern Recognition Letters*, 28:343–353, January 2007.

[132] L. Nanni and D. Maio. Weighted sub-Gabor for face recognition. *Pattern Recognition Letters*, 28:487–492, February 2007.

[133] S. Nedevschi, R. Schmidt, T. Graf, R. Danescu, D. Frentiu, T. Marita, F. Oniga, and C. Pocol. 3D lane detection system based on stereovision. In *IEEE Conference on Intelligent Transportation Systems*, pages 161–166, Washington D.C., USA, October 2004.

[134] A. Neubeck and L. Van Gool. Efficient non-maximum suppression. In *International Conference on Pattern Recognition*, pages 850–855, Hong Kong, China, September 2006.

[135] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, pages 849–856, Vancouver, Canada, December 2001.

[136] M. Nishiyama, A. Hadid, H. Takeshima, J. Shotton, T. Kozakaya, and O. Yamaguchi. Facial deblur inference using subspace analysis for recognition of blurred faces. *Accepted for IEEE Transactions on Pattern Analysis and Machine Intelligence.*

[137] M. Nishiyama, H. Takeshima, J. Shotton, T. Kozakaya, and O. Yamaguchi. Facial deblur inference to improve recognition of blurred faces. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1115–1122, Miami, FL, USA, June 2009.

[138] V. Ojansivu and J. Heikkilä. A method for blur and affine invariant object recognition using phase-only bispectrum. In *International conference on Image Analysis and Recognition*, pages 527–536, Povoa de Varzim, Portugal, April 2008.

[139] M. Osadchy, DW Jacobs, and M. Lindenbaum. Surface dependent representations for illumination insensitive image comparison. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29:98–111, January 2007.

[140] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: an application to face detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 130–137, San Juan, Puerto Rico, USA, June 1997.

[141] Y. Otsuka, S. Muramatsu, H. Takenaga, Y. Kobayashi, and T. Monj. Multitype lane markers recognition using local edge direction. In *IEEE Intelligent Vehicle Symposium*, pages 604–609, Versailles, France, June 2002.

[142] S. E. Palmer. *Explorations in cognition.* Freeman, 1975.

[143] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22:199–210, February 2011.

[144] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22:1345 –1359, October 2010.

[145] C.P. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *International Conference on Computer Vision*, pages 555–562, Bombay, India, January 1998.

[146] J. Peng, L. Mukherjee, V. Singh, D. Schuurmans, and L. Xu. An efficient algorithm for maximal margin clustering. *Journal of Global Optimization*, pages 1–15, February 2011.

[147] P. J. Phillips. Support vector machines applied to face recognition. In *Advances in Neural Information Processing Systems*, pages 803–809, Denver, CO, USA, December 1998.

[148] P. J. Phillips, P. J. Flynn, T. Scruggs, K. W. Bowyer, Jin Chang, K. Hoffman, J. Marques, M. Jaesik, and W. Worek. Overview of the face recognition grand challenge. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 947–954, San Diego, CA, USA, June 2005.

[149] P. J. Phillips, Hyeonjoon Moon, S. A. Rizvi, and P. J. Rauss. The FERET evaluation methodology for face-recognition algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:1090–1104, October 2000.

[150] E. Rahtu, M. Salo, and J. Heikkila. A new convexity measure based on a probabilistic interpretation of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1501–1512, September 2006.

[151] K.R. Rao and J. Ben-Arie. Optimal edge detection using expansion matching and restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16:1169–1182, December 1994.

[152] C. Rasmussen. Combining laser range, color, and texture cues for autonomous road following. In *IEEE International Conference on Robotics and Automation*, pages 4320–4325, Washington D.C., USA, May 2002.

[153] G. Rätsch, T. Onoda, and K. R. Müller. Soft margins for AdaBoost. *Machine Learning*, 42:287–320, March 2001.

[154] F. Rosenblatt. *Principles of Neurodynamics*. New York: Spartan, 1962.

[155] P. L. Rosin. Shape partitioning by convexity. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 30:202–210, February 2000.

[156] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, December 2000.

[157] H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:23–38, January 1998.

[158] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, January 1992.

[159] J. Ruiz-del Solar and J. Quinteros. Illumination compensation and normalization in eigenspace-based face recognition: A comparative study of different pre-processing approaches. *Pattern Recognition Letters*, 29:1966–1979, July 2008.

[160] R. M. Rustamov. Laplace-Beltrami eigenfunctions for deformation invariant shape representation. In *Eurographics symposium on Geometry processing*, pages 225–233, Barcelona, Spain, July 2007.

[161] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *European Conference on Computer Vision*, pages 213–226, Heraklion, Greece, September 2010.

[162] S. Saitoh. Theory of reproducing kernels and its applications, Pitman Research Notes in Mathematics Series, 189, 1988.

[163] F. S. Samaria and A. C. Harter. Parameterisation of a stochastic model for human faceidentification. In *IEEE Workshop on Applications of Computer Vision*, pages 138–142, Sarasota, FL, USA, December 1994.

[164] M. Savvides, B. V. K. V. Kumar, and P. K. Khosla. Eigenphases vs. eigenfaces. In *International Conference on Pattern Recognition*, pages 810–813, Cambridge, UK, August 2004.

[165] J. C. Schlimmer and R. H. Granger. Incremental learning from noisy data. *Machine learning*, 1:317–354, March 1986.

[166] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 746–751, Hilton Head, SC, USA, June 2000.

[167] T. Schoenemann and D. Cremers. Matching non-rigidly deformable shapes across images: A globally optimal solution. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 23–29, Anchorage, AK, USA, June 2008.

[168] C. Schwarz, J. Teich, A. Vainshtein, E. Welzl, and B.L. Evans. Minimal enclosing parallelogram with application. In *Symposium on Computational geometry*, pages 34–35, Vancouver, Canada, July 1995.

[169] T. B. Sebastian, P. N. Klein, and B. B. Kimia. Recognition of Shapes by Editing Their Shock Graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:550–571, May 2004.

[170] L. G. Shapiro and R. M. Haralick. Decomposition of two-dimensional shapes by graph-theoretic clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1:10–20, January 1979.

[171] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888 –905, August 2000.

[172] H. Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90:227–244, October 2000.

[173] T. Sim, S. Baker, and M. Bsat. The cmu pose, illumination, and expression database. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:1615 – 1618, December 2003.

[174] I. Stainvas and N. Intrator. Blurred face recognition via a hybrid network architecture. In *International Conference on Pattern Recognition*, pages 805–808, Barcelona, Spain, September 2000.

[175] G. W. Stewart and Ji-Guang Sun. *Matrix Perturbation Theory.* Academic Press, June 1990.

[176] T. Suk and J. Flusser. Combined blur and affine moment invariants and their use in pattern recognition. *Pattern Recognition*, 36:2895 – 2907, December 2003.

[177] X. Tan and B. Triggs. Enhanced local texture feature sets for face recognition under difficult lighting conditions. *Lecture Notes in Computer Science*, 4778:168–182, 2007.

[178] R. Tapia-Espinoza and M. Torres-Torriti. A comparison of gradient versus color and texture analysis for lane detection and tracking. In *Latin American Robotics Symposium*, pages 1 –6, Valparaiso, Chile, October 2009.

[179] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-Posed Problems.* V. H. Winston & Sons, Washington, D.C.: John Wiley & Sons, New York,, 1977.

[180] A. Torralba. Contextual priming for object detection. *International Journal of Computer Vision*, 53:169–191, February 2003.

[181] Z. Tu and A. L. Yuille. Shape matching and recognition-using generative models and informative features. In *European Conference on Computer Vision*, pages 195–209, Prague, Czech Republic, May 2004.

[182] G. Tur. Co-adaptation: Adaptive co-training for semi-supervised learning. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 3721–3724, Taipei, Taiwan, April 2009.

[183] P. Turaga, A. Veeraraghavan, and R. Chellappa. Statistical analysis on Stiefel and Grassmann manifolds with applications in computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Anchorage, AK, USA, June 2008.

[184] M. Turk and A. Pentland. Face recognition using eigenfaces. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–591, Maui, HI, USA, June 1991.

[185] H. Valizadegan and R. Jin. Generalized maximum margin clustering and unsupervised kernel learning. In *Advances in Neural Information Processing Systems*, pages 1417–1424. Vancouver, Canada, December 2007.

[186] M. Vasilescu and D. Terzopoulos. Multilinear analysis of image ensembles: Tensorface. In *European Conference on Computer Vision*, pages 447–460, Copenhagen, Denmark, May 2002.

[187] R. C. Veltkamp and M. Hagedoorn. State of the Art in Shape Matching. *Principles of visual information retrieval*, pages 87–119, 2001.

[188] P. Viola and M. Jones. Fast and robust classification using asymmetric adaboost and a detector cascade. In *Advances in Neural Information Processing Systems*, pages 1311–1318, Vancouver, Canada, December 2002.

[189] P. Viola and M. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57:137–154, May 2002.

[190] L. L. Walker and J. Malik. Can convexity explain how humans segment objects into parts? *Journal of Vision*, 3:503, September 2003.

[191] C. Wang and S. Mahadevan. Manifold alignment without correspondence. In *International Joint Conference on Artificial Intelligence*, pages 1273–1278, Pasadena, CA, USA, July 2009.

[192] F. Wang, B. Zhao, and C. Zhang. Linear time maximum margin clustering. *IEEE Transactions on Neural Networks*, 21:319–332, February 2010.

[193] H. Wang, S. Z. Li, and Y. Wang. Face recognition under varying lighting conditions using self quotient image. In *IEEE International Conference on Automatic Face and Gesture Recognition*, pages 819–824, Seoul, Korea, May 2004.

[194] J. Wang and K. L. Chan. Shape evolution for rigid and nonrigid shape registration and recovery. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 164–171, Miami, FL, USA, June 2009.

[195] Y. Wang, E.K. Teoh, and D. Shen. Lane detection and tracking using B-Snake. *Image and Vision Computing*, 22:269–280, April 2004.

[196] M. K. Warmuth, K. Glocer, and G. Ratsch. Boosting algorithms for maximizing the soft margin. In *Advances in Neural Information Processing Systems*, pages 1585–1592, Vancouver, Canada, December 2008.

[197] M. K. Warmuth and J. Liao. Totally corrective boosting algorithms that maximize the margin. In *International Conference on Machine Learning*, pages 1001–1008, Pittsburgh, PA, USA, June 2006.

[198] H. Wold. *Partial Least Squares*, volume 6. Encyclopedia of Statistical Sciences, 1985.

[199] Y.-C. Wong. Differential geometry of Grassmann manifolds. *Proceedings of the National Academy of Science*, 57:589–594, March 1967.

[200] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31:210–227, February 2009.

[201] B. Wu, H. Ai, C. Huang, and S. Lao. Fast rotation invariant multi-view face detection based on real adaboost. In *IEEE International Conference on Automatic Face and Gesture Recognition*, pages 79–84, Seoul, Korea, May 2004.

[202] B. Wu and R. Nevatia. Detection and tracking of multiple, partially occluded humans by Bayesian combination of edgelet based part detectors. *International Journal of Computer Vision*, 75:247–266, November 2007.

[203] H. Wu, A. C. Sankaranarayanan, and R. Chellappa. Online empirical evaluation of tracking algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (PrePrints), 2009.

[204] D. Xing, W. Dai, G. R. Xue, and Y. Yu. Bridged refinement for transfer learning. In *Knowledge Discovery in Databases*, pages 324–335, Warsaw, Poland, July 2007.

[205] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. In *Advances in Neural Information Processing Systems*, pages 1537–1544, Vancouver, Canada, December 2005.

[206] M. H. Yang. Kernel eigenfaces vs. kernel fisherfaces: Face recognition using kernel methods. In *IEEE International Conference on Automatic Face and Gesture Recognition*, pages 215–220, Washington D.C., USA, May 2002.

[207] M. H. Yang, D. J. Kriegman, and N. Ahuja. Detecting faces in images: A survey. *IEEE Transactions on Pattern analysis and Machine intelligence*, 24:34–58, January 2002.

[208] X. Yang, X. Bai, L.J. Latecki, and Z. Tu. Improving shape retrieval by learning graph transduction. In *European Conference on Computer Vision*, pages 788–801, Marseille, France, October 2008.

[209] X. Yang, S. Köknar-Tezel, and L. J. Latecki. Locally constrained diffusion process on locally densified distance spaces with applications to shape retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 357–364, Miami, FL, USA, June 2009.

[210] J. Ye, Z. Zhao, and H. Liu. Adaptive distance metric learning for clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7, Minneapolis, MN, USA, June 2007.

[211] J. Ye, Z. Zhao, and M. Wu. Discriminative k-means for clustering. In *Advances in Neural Information Processing Systems*, pages 1649–1656. Vancouver, Canada, December 2008.

[212] L. Yuan, J. Sun, L. Quan, and H-Y. Shum. Progressive inter-scale and intra-scale non-blind image deconvolution. *ACM Transactions on Graphics*, 27:1–10, March 2008.

[213] C. T. Zahn and R. Z. Roskies. Fourier descriptors for plane closed curves. *IEEE Transactions on Computers*, 21:269 –281, March 1972.

[214] J. Zhang, R. Collins, and Y. Liu. Representation and matching of articulated shapes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 342–349, Washington D.C., USA, June 2004.

[215] K. Zhang, I. W. Tsang, and J. T. Kwok. Maximum margin clustering made practical. *IEEE Transactions on Neural Networks*, 20:583–596, April 2009.

[216] B. Zhao, J. Kwok, F. Wang, and C. Zhang. Unsupervised maximum margin feature selection with manifold regularization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 888–895, Miami, FL, USA, June 2009.

[217] B. Zhao, F. Wang, and C. Zhang. Efficient multiclass maximum margin clustering. In *International Conference on Machine Learning*, pages 1248–1255, Helsinki, Finland, July 2008.

[218] J. Zunic and P. L. Rosin. A new convexity measure for polygons. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:923–934, July 2004.