Compound Exemplar based Object Class Detection and Beyond with VARIS System

BY

KAI MA
M.S., Kansas State University, 2007
B.E., Shandong University of Technology, 2005

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Chicago, 2014

Chicago, Illinois

Defense Committee:

Jezekiel Ben-Arie, Chair and Advisor
Rashid Ansari
Daniel Graupe
Milos Zefran
Barbara Di Eugenio, Computer Science

To myself:

Be as smart as you can, but remember that it is always better to be wise than to be smart.

A smart man learns from the mistake he makes and never makes the same one again.

But a wise man learns from the smart man about how to avoid the mistake altogether.

ACKNOWLEDGMENTS

Since the long battle is almost to the end, it is a great pleasure to look over the past and remember all the people who have helped and supported me along this journey.

I would like to first express my gratitude to Professor Jezekiel Ben-Arie, who are not just my study mentor but also a true friend. He shows me how to think the problem out of the box, solve the problem not just correctly but also with creativity. Beyond the PhD study, he also shares precious experience with me, such as politics, religions, art and even stock market.

I would also like to thank my defense committee members, Professor Rashid Ansari, Professor Barbara Di Eugenio, Professor Daniel Graupe and Professor Milos Zefran, who help me review the thesis and provide valuable feedback. I am grateful for their thoughtful and detailed comments.

I would not have completed this journey without my parents, Hongmin Ma and Chunling Ma. They support every decision I have made during the study and encourage me all the time even though they are far far away in China. I would also like to thank my girlfriend, Shu Wang, who always takes care of me and my pet dog Boxer. To Shu, thank you for tolerating my bad habits.

And last but not least, to all my friends who love me, thank you for your support.

KM

TABLE OF CONTENTS

CHAPTER			PAGE
1	**INTRODUCTION**		1
	1.1	The Goal	2
	1.2	The Challenges	2
	1.3	Overview of Our Approach	3
	1.4	Outline of the Thesis	6
2	**STATE OF THE ART**		7
	2.1	Image Features	7
	2.1.1	Sparse Local Representation	8
	2.1.2	Dense Representation of Image Regions	11
	2.2	Classification Methods	14
	2.2.1	Discriminative Approaches	16
	2.2.2	Generative Approaches	17
	2.3	Part-based Approach	18
	2.4	Exemplar-based Approach	19
	2.5	Our Approach - VARIS	19
3	**FEATURE INDEXING AND PART DEFINITION**		21
	3.1	Feature Extraction	21
	3.2	Feature Indexing	23
	3.3	Feature Importance Weighting	29
4	**SEQUENCING AND EXEMPLAR COMPOUNDING**		31
	4.1	Dynamic Programming	31
	4.2	1-D Sequencing with Dynamic Programming	32
	4.3	2-D Sequencing of VARIS	33
	4.4	Exemplar Compounding	35
5	**FACE DETECTION AND POSE RECOGNITION**		39
	5.1	Data Preparations	39
	5.2	Pose Recognition	40
	5.3	Face Detection	43
6	**GENERAL OBJECT DETECTION**		50
	6.1	Data Preparations	50
	6.2	Training Stage	51
	6.3	Testing Stage and Results	51

TABLE OF CONTENTS (Continued)

CHAPTER **PAGE**

7 IMPROVED INDEXING STAGE WITH RANDOM FOREST 58
7.1 Modified Random Forest Framework 59
7.2 Training Procedure 61
7.3 Testing Procedure 63
7.4 Vote Weighting 64
7.5 Experiment and Results 65

8 BEYOND 2D OBJECT DETECTION - 3D HUMAN POSE AND SHAPE ESTIMATION 69
8.1 Human Body Representation in 3D 70
8.2 Data Preparation 71
8.3 Parametric Deformable Model 74
8.3.1 Pose Deformation 75
8.3.2 Shape Deformation 77
8.4 Partial Data Completion with PDM 77

9 CONCLUSION AND FUTURE WORK 81

CITED LITERATURE 83

VITA 90

LIST OF TABLES

TABLE		PAGE
I	Pose recognition comparison between VARIS with compounding and ESVM.	43
II	Comparisons of average detection precision on VOC2007 data set. ESVM and DPM are implemented with authors' default settings. VARISEC stands for VARIS with exemplar compounding. The highest scores are highlighted in bold.	52
III	Performance comparisons. The first 4 rows show the performance of our competitors. MRFGC in the fifth row is our basic proposal without incremental learning. MRFGC+IL shows the performance of our extended system trained with extra hard examples from the validation set.	66

LIST OF FIGURES

FIGURE		PAGE
1	Challenge of object detection. All the images are from the car category defined by human. Note the large variations of appearance, pose, scale, illumination changes and occlusion.	4
2	Flow chart of VARIS. Top: Validation stage. New exemplars are iteratively added to the initial training dataset until no further improvement is achieved; Bottom: Testing stage. Similar exemplars are selected by applying VARIS on the final dataset and new compound exemplars with higher similarity are assembled with parts from different exemplars.	5
3	Interest points extracted by the DoG operator. Radius of circles stands for the scale of interest points and the bars inside of circles show the dominate orientations.	9
4	SIFT interest point descriptor. The extracted image patches are sampled into $8x8$ pixel cells. $2x2$ such cells are combined into one histogram vector.	10
5	12 different Haar-like wavelet filters used in Viola-Jones face detector.	12
6	Example of integral image (also known as summed area table) used in Viola-Jones face detector. The value at any point $I(x, y)$ in the integral image is the sum of all pixels above and to the left. To calculate the rectangle area $ABCD$, it is just simple as $I(A) + I(C) - I(B) - I(D)$.	14
7	Diagram of HOG feature extraction.	15
8	Example of linear support vector machine(SVM) learning. The goal is to find the maximum margin that seperates two classes.	17
9	Cascade of Adaboost classifier. Each layer is a strong classifier with high detection rate and moderate false alarm rate. Detection windows classified as faces by the current layer will be tested on next layer and rest will be rejected as non-faces. After the whole cascade, the system will still maintain a good detection rate while having a very low false alarm rate.	18
10	A kd-tree and the associated spatial subdivision.	24
11	Priority search of a KD-tree. The red dot is the query point and its cell 3 has its true nearest neighbor. The search goes down the tree and retrieve the first node as the neighbor candidate (marked as 1 in this case). However, this point is often not the closest neighbor of the query point. A priority search proceeds in order through other nodes of the tree in order of their distance from the query point. In this image, it would be through node 2 to 5. The search is bounded by a hyper-sphere of radius r. When there are no more cells within this radius, the search terminates.	26

LIST OF FIGURES (Continued)

FIGURE		PAGE
12	Examples of feature indexing. (a) A test image with overlapping HOG feature grid. The eye in red square and the mouth in blue square are two indexed patches; (b) 10 nearest neighbors of the indexed eye. The eyes in green squares (6 and 10) are from non-frontal face classes; (c) 10 nearest neighbors of the indexed mouth. The patches in green squares (5, 8 and 9) are from non-frontal face classes.	28
13	Spatial restrictions of 1D dynamic programming.$a0$ to $b1$ is a violation of the first restriction. $a5$ to $b3$ is a violation of the second restriction.	33
14	2D sequencing in the geometry verification stage. Left image shows the horizontal sequencing at image patch level between input and one exemplar. Right image shows the vertical sequencing for multiple rows of the input image and the exemplar. Both sequencing approaches are subject to two spatial constraints mentioned in Figure 13. The blue solid lines are the optimal selections maximizing the similarities among all the mapping relationships that are labeled with gray dash lines.	35
15	Example of a compound exemplar. Left: test image; Middle: the compound exemplar that is assembled with parts from different exemplars of the same pose class. It has a higher similarity with the test image than each of the exemplars on the right; Right: original exemplars with partial similarities to the input.	36
16	Diagram of 2D sequencing and compounding procedures. In the first phase, the 1D sequencing searches for the optimized exemplar rows matching to each input row. In the second phase all input and exemplar rows are assembled into columns. Next, 1D sequencing finds the optimal matching of the input column to each exemplar column. Compounding takes the best rows from different exemplars and creates a new exemplar that has a higher similarity.	38
17	Examples of face exemplars. The final training data set includes human faces of different genders, ages, races, poses and appearance.	41
18	Pose recognition results. The confusion matrix shows the correct and incorrect pose estimations of VARIS with compounding on the FERET face dataset.	42
19	Computational efficiency. Computation time and recognition rates are measured under different settings of k. The time is the summation of indexing and sequencing steps for all exemplars.	44
20	Discrete ROC curve. VARIS+CE is VARIS with compound exemplars.	46
21	Continuous ROC curve. VARIS+CE is VARIS with compound exemplars.	47
22	Detection results. The ground truth is labeled by red ellipses and our detection is labeled by squares. First row shows the correct detections where both face locations and face poses are correctly estimated. Second row shows the cases in which VARIS doesn't work well. Faces labeled only by red ellipses are the missed cases. The magenta texts label the faces that are correctly detected but the poses are not correct. The magenta text and square denote the false detection cases.	48

LIST OF FIGURES (Continued)

FIGURE		PAGE
23	Detection comparisons between VARIS and other approaches. First row shows results of VARIS. Second row shows results of ESVM algorithm. Third row shows results of LSVM algorithm.	49
24	Detection results of airplane, bird, bicycle and bottle classes. Red rectangle labels the ground truth and yellow rectangle labels detections of VARIS plus compound exemplar. If there are classes in the same image, the second class will be labeled in blue.	54
25	Detection results of bus, car, cat, dog, cow and horse classes. Red rectangle labels the ground truth and yellow rectangle labels detections of VARIS plus compound exemplar. If there are classes in the same image, the second class will be labeled in blue.	55
26	Detection results of sheep, sofa, tv(monitor), plant and train classes. Red rectangle labels the ground truth and yellow rectangle labels detections of VARIS plus compound exemplar. If there are classes in the same image, the second class will be labeled in blue.	56
27	Cases that our framework fails on.	57
28	Different patch representations in the leaf node between the standard random forest and the proposed modified random forest. Left shows that the input patch travels through the tree and retrieve a continuous probability distribution of the relative object center. Right shows that, in our proposal, the same input patch reaches the leaf node and retrieves different exemplar patches that are visually similar. Note that this discrete patch representation enables the direct modification on the leaf node, which is the basis of the incremental learning.	60
29	Detection results on different object classes. The detected objects are labeled by yellow rectangles and the smaller images are the corresponding exemplars that are visually similar to the detections.	67
30	Additional detection results from the extended system. The top row of each sub-image shows the new exemplars from the validation set. The bottom rows show the new detection results which were missed by the basic version of our proposal. It is obvious that the new added information is helpful to enhance system performance.	68
31	Left: Configuration of Microsoft Kinect device. Right: Depth image captured by Kinect. Different colors label different depth information.	70
32	3D representation of human body.	72
33	Mesh instances in different poses generated from Poser.	72
34	Mesh instances in different body shapes generated from Poser.	73
35	Template mesh model with 16 joint landmarks. Different body parts are shown in different colors. Joint landmarks are shown in white squares.	75
36	PDM pose deformation. Left figure shows a driving pose. Middle figure shows the template pose. Right figure shows a walking pose.	76

LIST OF FIGURES (Continued)

FIGURE		PAGE
37	PDM shape deformation along the direction of first principal component. Left figure shows an extreme case of an underweight person. Middle figure shows the template mesh. Right figure shows an extreme case of an overweight person.	78
38	Partial view completion flow chart and result.	80

LIST OF ABBREVIATIONS

CE	Compound Exemplar
DoG	Difference of Gaussian
ESVM	Exemplar-SVM
DP	Dynamic Programming
DPM	Deformable Part Model
DTW	Dynamic Time Warping
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
HOG	Histogram of Oriented Gradient
ICP	Iterative Closest Point
ISM	Implicit Shape Model
NN	Nearest Neighbor
PASCAL	Pattern Analysis, Statistical Modeling and Computational Learning
PCA	Principal Component Analysis
PDM	Parameterized Deformable Model
RISq	Recognition by Indexing and Sequencing

LIST OF ABBREVIATIONS (Continued)

RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
SCAPE	Shape Completion and Animation of PEople
SIFT	Scale Invariant Feature Transform
SOA	State-Of-the-Art
SURF	Speeded Up Robust Feature
SVM	Support Vector Machine
UIC	University of Illinois at Chicago
VARIS	Vector Array Recognition by Indexing and Sequencing
VOC	Visual Object Classes

SUMMARY

Determining the location and scale of a particular object class in a 2D image is usually referred as object detection in the computer vision area. Object detection is a well studied topic and many successful algorithms have been proposed during the last two decades. However, recent experimental surveys reveal that the performance of the state-of-the-art detection systems still have low performance on images in unconstrained environment. The major reasons are due to high intra-class variation and object self-occlusion.

In this thesis, I will present a novel exemplar-based object detection framework that outperforms the state-of-the-art systems in terms of accuracy. The proposed method, Vector Array Recognition by Indexing and Sequencing (VARIS) (10), is designed to fulfill two requirements in object detection: Generalization and Reliability. The foundation of VARIS is to dynamically assemble an object exemplar that maximizes the similarity to the input image. Experimental results show that VARIS achieves better results than its competitors even with a very compact training dataset. Meanwhile, the computational speed is significantly increased with the help of a modified random forest module, which allows the full system to run in real time on standard images.

Beyond 2D object detection topic, I also explored the 3D computer vision domain. I designed and implemented a novel framework that estimated human body shape and pose simultaneously, which is named as parametric deformable model (PDM). PDM demonstrates the ability to recover the true human body pose and shape even by given a noisy and occluded 3D depth image as the input. PDM

also brings many potential applications, such as better body joints estimation. Once the joint locations are determined, we can extend the 1D VARIS system to recognize the human activity.

CHAPTER 1

INTRODUCTION

In our present lives, computers has been used everywhere. They follows our instructions and perform repetitive, delicate and complex tasks more accurately and more efficiently than humans. To gain more benefits from computers, it is natural to extend their functionality to perform more intelligent tasks, such as analysis of images, videos and speech, logical inference and decision determination - in brief the high-level tasks that we humans perform subconsciously hundreds of times every day with so much ease that we do not usually even realize that we are performing them. Take the example of the human visual system. Our daily lives are filled with thousands of objects ranging from man made classes like cars, bicycles, buildings, tables, chairs to natural ones like sheep, cows, dogs, cats and humans. Any given class has a huge intra-class variation. For example, car can be used to denote many four wheeled vehicles, including various sub categories like sedan, hunchback, station wagon or SUV. The exact type, color and viewpoint of a car is irrelevant to the decision that an object is a car. Similarly, we are able to detect people under widely varied conditions - irrespective of the color or kind of clothing, pose, appearance, partial occlusions, illumination or background clutter. Computers are currently far behind humans in performing such analysis and inference.

Thus one goal of researchers working in computer vision has been to grant computers the ability to see - visual analysis and interpretation of images or videos. One of the primary tasks is the detection of different classes of objects in images and videos. Such a capability would have many applications, for example, in human computer interaction, robotics, automatic analysis of personal or commercial digital

media content, automated manufacturing processes, smart autonomous vehicles and public security.

1.1 The Goal

In the computer vision domain, the basic goal of object detection is to search the given image and determine whether there exist any pre-defined objects and, if present, return the location and scale of each detected object instancea. Many object detection methods fulfill with this requirement. Faced with a new task, one simply carves up the solution space into classes (e.g., cars, people, animals), assigns class labels to training examples and applies one of the many popular machine learning tools to obtain a classifier. This report, however, targets the problem of object detection in a different way. We would not tell ”What is it?” but rather ”What is it like”. When faced with a new object, we try to associate it with the most similar objects in the memory. The stored objects, in turn, provide the meta-data, such as object pose, appearance, associated actions and so on, that is needed to better interpret the object for human understanding. This object association idea is similar to human visual cognition.

1.2 The Challenges

The foremost difficulty in building a robust object detector is the amount of variation in images and videos. Several factors contribute to this:

- The image formation process suppresses 3-D depth information and creates dependencies on viewpoint such that even with a small change in the object's position or orientation, the appearance of object class changes considerably. A related issue is the large variation in scales under which an object can be viewed. An object detector must handle the issues of viewpoint and scale changes and provide invariance to them.

- Most natural object classes have large within-class variations. For example, for humans both appearance and pose change considerably between images and differences in clothing create further changes. A robust detector must try to achieve independence of these variations.
- Background clutter is common and varies from image to image. Examples are images taken in natural settings, outdoor scenes in cities and indoor environments. The detector must be capable of distinguishing object class from complex background regions.
- Partial occlusions create further difficulties because only part of the object is visible for processing.

Figure 1 provides some examples illustrating these difficulties for car detection. Note the amount of variations in the images.

1.3 Overview of Our Approach

We propose a novel method, Vector Array Recognition by Indexing and Sequencing (VARIS) (10; 54; 53) in order to tackle the problem of multi-view multi-class object detection. VARIS combines two popular frameworks in computer vision, the part-based approach and the exemplar-based method. The basic idea of VARIS is to dynamically assemble an object exemplar that maximizes the similarity to the input. To overcome the limitation of the original training dataset, we introduce a novel mechanism called exemplar compounding that allows parts from different exemplars of the same class to be optimally selected and assembled into a new exemplar. Based on a much smaller training set, VARIS achieves better results, in most cases, than the current state-of-the-art object detection systems with a reasonable computation cost.

VARIS is designed to fulfill two major, sometimes conflicting, requirements of object detection:

Figure 1: Challenge of object detection. All the images are from the car category defined by human. Note the large variations of appearance, pose, scale, illumination changes and occlusion.

Generalization and *Reliability*. Generalization requires recognizing a very large number of different appearances. For this reason, the desired system should have a flexible representation that can tolerate large geometrical variations. We collect a set of face exemplars and represent each exemplar by a tessellation of small image patches. Geometrical flexibility is achieved by the ability of VARIS to dynamically modify the location of patches. Reliability is maintained by the indexing and sequencing steps in VARIS. The indexing step retrieves the stored exemplar patches most similar to each input patch. The sequencing step finds the optimal matching of the indexed patches to the input while preserving the mutual topology.

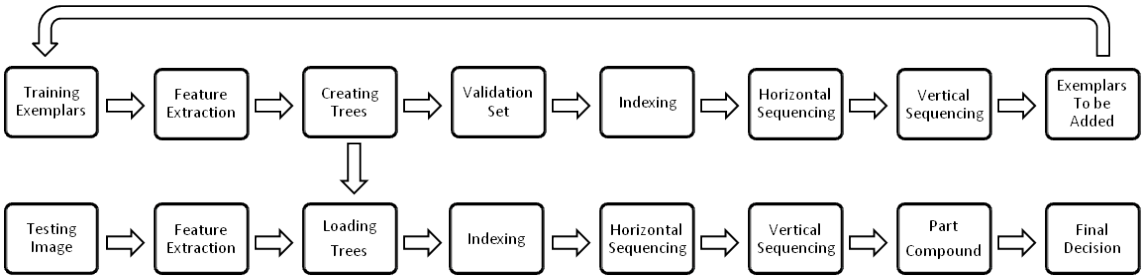

Figure 2: Flow chart of VARIS. Top: Validation stage. New exemplars are iteratively added to the initial training dataset until no further improvement is achieved; Bottom: Testing stage. Similar exemplars are selected by applying VARIS on the final dataset and new compound exemplars with higher similarity are assembled with parts from different exemplars.

The basic idea of VARIS is to maximize the similarity between the input and a stored exemplar which is composed of the optimal combination of classified feature patches. The optimization is constrained by the requirements of preserving the topology of both input and exemplars. In this work, we define a non-parametric part representation, which enables a dynamical assembly of local exemplar features. Moreover, parts from different exemplars of the same class can be optimally selected and assembled into a new compound exemplar that has the highest similarity with the input. The general flow chart of VARIS with compounding is shown in Figure 2. Based on a much smaller training set, our system achieves better results than the current state-of-the-art object detection systems (31) (57) with a reasonable computational cost. In contrast to other methods, VARIS with compounding is also less sensitive to geometrical distortion, noise and partial occlusion.

1.4 Outline of the Thesis

In this chapter, we first introduced the importance of object detection in computer vision area and the major obstacles. Then I briefly described our novel system that is designed for solving the multi-view multi-class object detection problem. The rest of the thesis is structured as follows.

- Chapter 2 summarizes relevant previous work on multi-view multi-class object class detection.
- Chapter 3 describes the feature indexing step of VARIS.
- Chapter 4 describes the details of sequencing and exemplar compounding.
- Chapter 5 displays the experimental results of face detection and face pose recognition as well as the comparisons with other approaches.
- Chapter 6 demonstrates an extended experimentation in detecting general objects.
- Chapter 7 shows an extension work of VARIS, which increase the recognition accuracy as well as the speed.
- Chapter 8 introduces my work, Parameterized Deformable Model (PDM), on 3D human body pose and shape estimation. It demonstrates the mathematics foundation and the full details of implementation.
- Chapter 9 summarizes the work I have done and also mentions some future work.

CHAPTER 2

STATE OF THE ART

Object detection in images and videos has received a lot of attention in the computer vision and pattern recognition communities in recent years. The two key issues for object detection are: what kind of features to extract, and which learning algorithm to apply. This chapter reviews successful techniques used in object detection and localization area. Most of the cited works can be classified according to:

- The image descriptors or feature vectors that they use
- The detection framework that is built over these descriptors

We review the relevant work according to these categories. Section 2.1 covers the different image feature sets that have been used. Section 2.2 provides an overview of the classification approaches.

2.1 Image Features

The image feature set should include the most relevant features for object detection or classification while providing invariance to changes in illumination, differences in viewpoint and deformations in object shapes. To achieve such invariances, one often uses more advanced local image descriptors instead of directly applying raw image information. Such descriptors can be based on points, blobs, intensities, gradient, color, texture, or combinations of several of all of these. The final descriptors need to characterize the image sufficiently well for the detection and classification task at hand. We divide the various approaches into two broad categories: sparse representations based on points, image fragments or part detectors; and dense representation using image intensities or gradients.

2.1.1 Sparse Local Representation

The use of salient local points or regions for object detection has a long history (72; 75; 88; 50; 1; 34; 27; 51; 66; 48; 59). These approaches extract local image features at a sparse set of salient image points - usually called points of interest or key points. The final detectors are then based on feature vectors computed from these key point descriptors. The hypothesis is that key point detectors select stable and more reliable image regions, which are especially informative about local image content. The overall detector performance thus depends on the reliability, accuracy and repeatability with which these key points can be found for the given object class and the informativeness of the points chosen. Commonly used key point detectors include Harris (41), Laplacian (49), difference of Gaussian (DoG) (51) and scale invariant Harris-Laplace (60). Some point detectors such as DoG or Harris-laplace also provide additional local scale and/or dominant orientation information. One advantage of sparse key point based approaches is the compactness of the representation: there are fewer key-point descriptors than image pixels, so the latter stages of the classification process are speeded up. However note that most key point detectors are designed to fire repeatedly on particular objects and may have limitations when generalizing to object classes or categories, *i.e.* they may not be repeatable for general object classes.

Regarding the computation of feature vectors or descriptors over the local image regions surrounding the key points, many approaches have been tried. Currently the most popular approaches are image gradient based descriptors such as the Scale Invariant Feature Transformation (SIFT) (51) and its variants, PCA-SIFT (45) and SURF (7). As one of the most popular local feature descriptors, SIFT uses the local scale and dominant orientation given by the DoG detector to vote into orientation histograms with

Figure 3: Interest points extracted by the DoG operator. Radius of circles stands for the scale of interest points and the bars inside of circles show the dominate orientations.

weighting based on gradient magnitudes. Figure 3 shows an image that displays interest points detected

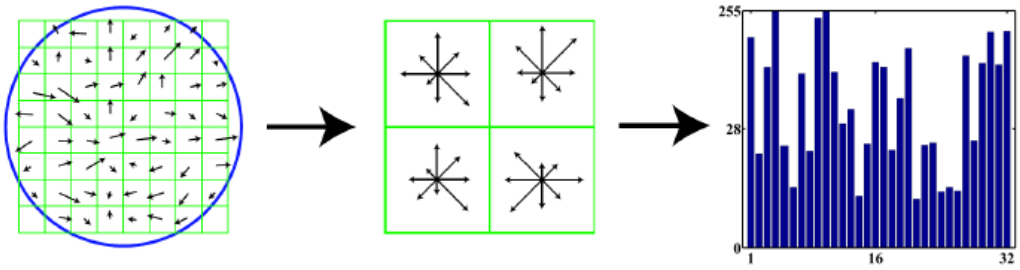

Figure 4: SIFT interest point descriptor. The extracted image patches are sampled into $8x8$ pixel cells. $2x2$ such cells are combined into one histogram vector.

by the DoG detector. As shown in (51), "the particular gradient magnitudes m and local orientations ϕ for each pixel $I(x, y)$ in the image are calculated by simple pixel differences according to"

$$\begin{aligned} m &= \sqrt{(I(x+1,y) - I(x-1,y))^2 + (I(x,y+1) - I(x,y-1))^2} \\ \phi &= \arctan\left((I(x,y+1) - I(x,y-1))/(I(x+1,y) - I(x-1,y))\right) \end{aligned} \tag{2.1}$$

It thus computes scale and rotation invariant feature vectors. The scale information is also used to define an appropriate smoothing scale when computing image gradients. For each detected key-point, SIFT first normalizes all the weighted gradients in the circular region according to the dominant orientation. Then the circular region is further divided into 4×4 patches without any overlapping. Next SIFT calculates the histogram of oriented gradients within the patches and applies smoothing in order to avoid sudden changes of orientation. The size of the histogram vector is limit to 8 for compactness. At the end, a $4 \times 4 \times 8 = 128$ dimensional feature vector is generated for each key-point. Figure 4 illustrates the SIFT descriptor for a 2×2 window.

2.1.2 Dense Representation of Image Regions

Another approach to extract image features is to densely sample the entire image or detection window into a high-dimensional descriptor vector that can be used for discriminative image classification or labeling window as object or non-object. Typically the representation is based on image intensities, gradients or higher order differential operators.

Regions and Fragments Based on Image Intensity

One of the primary works using simple image intensities is the ”eigenfaces” approach of Sirovitch and Kirby (79) and Turk and Pentland (83), where the pixels of fixed-resolution face images are rearranged to form large feature vector and Principle Component Analysis (PCA) is used to characterize the main variations of the ensemble of face vectors. Another work using intensity images is the face detection system of Rowley *et al.*(70) who locally correct the lighting of the images by performing histogram equalization before passing them to a neural network classifier (15) for face/non-face detections.

Wavelet Based Detectors

Some well known approaches to object detection are described in Papageorgiou and Poggio (68), Mohan *et al.*(61), Viola and Jones (86). These approaches use dense encoding of image regions based on operators similar to Haar wavelets. Figure 5 shows some examples of Haar wavelet features.

Papageorgious and Poggio (68) use absolute values of Haar wavelet coefficients at different orientations and scales as their local descriptors. Images are mapped from pixel space to an over-complete dictionary of Haar wavelets that is rich enough to describe patterns. Horizontal, vertical and diagonal wavelets are used. To obtain an over-complete basis, the wavelets are computed with overlapping sup-

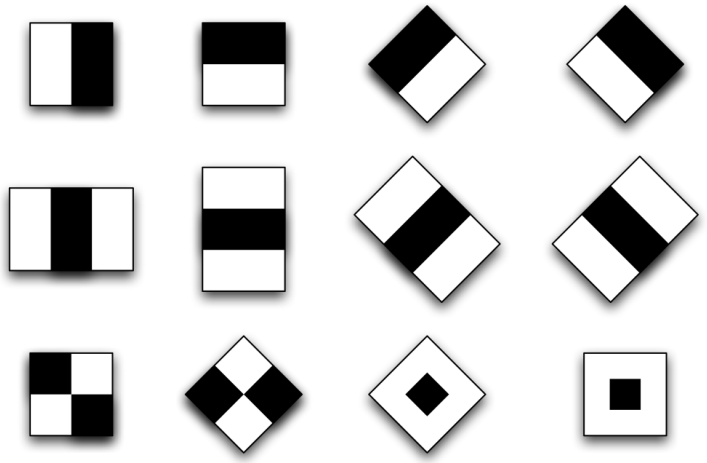

Figure 5: 12 different Haar-like wavelet filters used in Viola-Jones face detector.

ports. Haar wavelets can be calculated efficiently, while still remaining rich enough to encode visually significant patterns, and the use of over-completeness provides a reasonable degree of translation invariance. The descriptor vectors are used in a kernelized Support Vector Machine (SVM) framework, so the final decision criterion is a sum of weighted kernel distances from selected training examples.

Viola and Jones applied the integral image for rapid computation of Haar-like rectangle features. With the integral image, they can efficiently compute the sum of values in a rectangle subset of a grid. The integral image is constructed as follows:

$$\mathcal{I}(x,y) = \sum_{x' \leq x, y' \leq y} i(x', y') \tag{2.2}$$

where $\mathcal{I}(x, y)$ is the integral image at pixel location (x, y) and $i(x', y')$ is the pixel in the original image. Since The sum of pixels in rectangle region $ABCD$ in Figure 6 can be calculated with 3 summations, the computation of the Haar feature with intergral image turns out to be extremely efficient, as shown:

$$\sum_{(x,y)\in ABCD} i(x, y) = \mathcal{I}(D) + \mathcal{I}(A) - \mathcal{I}(B) - \mathcal{I}(C) \tag{2.3}$$

Edge and Gradient Based Detectors

Image edges and gradient filters have also been used for object detection. A popular approach is the pedestrian detection system of Gavrila and Philomin (39), who propose to extract edge images and match them to a set of learned exemplars using chamfer distance. Dalal and Triggs (25) present a human classification scheme that uses SIFT-inspired features, called histograms of oriented gradients (HOGs). "An HOG feature divides the region into k orientation bins (in most cases, $k = 9$). Instead of computing the ratio between two bins, they define four different cells that divide the rectangular feature. The resulting feature is a 36D vector containing the summed magnitude of each pixel cells. An overview of HOG feature extraction is shown in Figure 7.

The original work lists several advantages of using HOG feature. "The use of orientation histograms over image gradients allows HOGs to capture local contour information, *i.e.* the edge or gradient structure, that is very characteristic of local shape. In conjunction with the spatial quantization into cells, it allows them to capture the most relevant information with controllable precision and invariance." Translations and rotations make little difference as long as they are much smaller than the local spatial or orientation bin size. Gamma normalization and local contrast normalization contribute another key

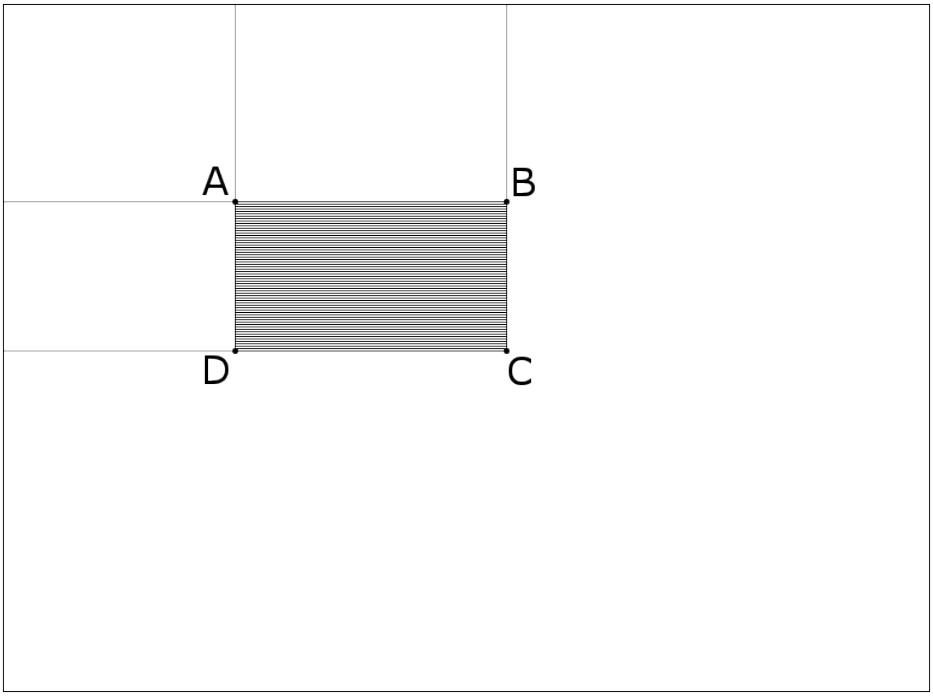

Figure 6: Example of integral image (also known as summed area table) used in Viola-Jones face detector. The value at any point $I(x, y)$ in the integral image is the sum of all pixels above and to the left. To calculate the rectangle area $ABCD$, it is just simple as $I(A) + I(C) - I(B) - I(D)$.

component: illumination invariance. These steps ensure that as little information as possible is lost during the encoding process.

2.2 Classification Methods

Classification methods can be divided into discriminative approaches such as Support Vector Machines, and generative ones such as graphical models, where each method can be further divided into holistic approach and part-based approach, depending on how to use the extracted features.

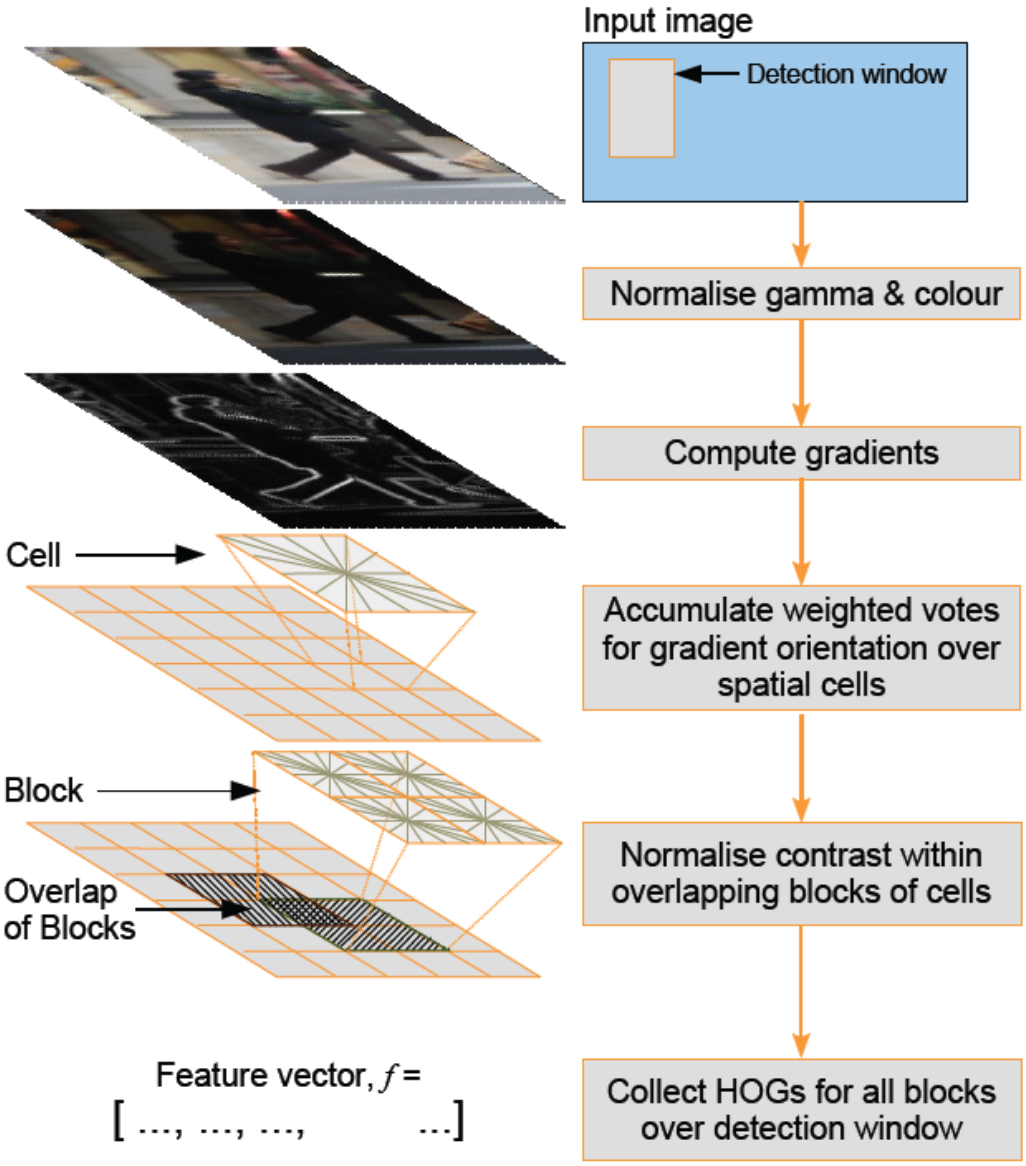

Figure 7: Diagram of HOG feature extraction.

2.2.1 Discriminative Approaches

Discriminative models are trained explicitly to differentiate the foreground objects from their background. Machine learning techniques such as SVM (85) and boosting (71) have become popular as classifiers for object detection owing to their ability to automatically select relevant descriptors or features from large feature sets, their superior performance and their relative ease of use.

Support Vector Machine Classifiers

SVM have been widely used for object detection for the past decade (25; 17; 31; 57). "Given a training data set, SVM tries to find a separating hyperplane that maximizes the margin between the object class and non-object class in either the input feature space or a kernelized version of this. The location of this hyperplane is determined by a subset of the data points, known as support vectors." New examples are then mapped into the hyperspace and predicted to belong to a category based on which side of the gap they fall on. Figure 8 shows a simple linear-separable two-class case of SVM learning.

Cascaded AdaBoost

AdaBoost combines a collection of weak classifiers to form a stronger one. In vision, it is used to particularly to build cascades of pattern rejecters (see Figure 9), with at each level of the cascade choosing the feature most relevant for its rejection task. Although AdaBoost cascades are slow to train, owing to their elective feature encoding they offer significant improvement in the run-time of the final detectors. Viola and Jones (86) use AdaBoost to train cascades of weak classifiers for face detection, using spatial and temporal difference-of-rectangle based descriptors. The seminal work has made face detection specifically feasible in real world applications such as digital cameras and photo organization software.

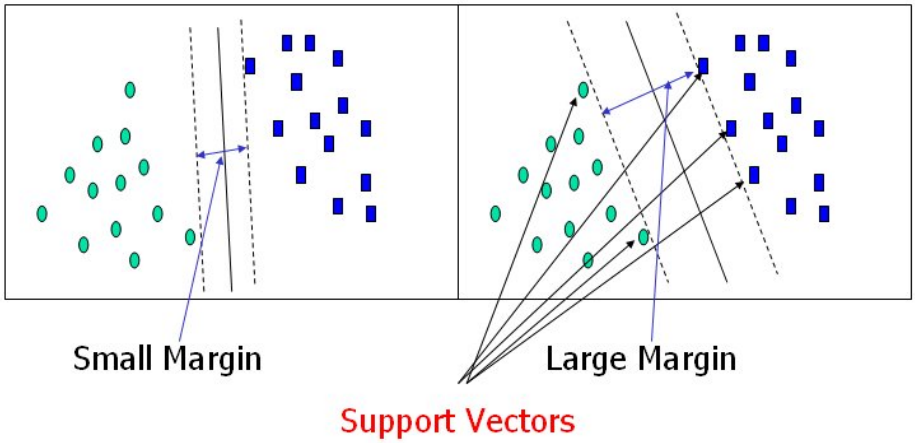

Figure 8: Example of linear support vector machine(SVM) learning. The goal is to find the maximum margin that seperates two classes.

2.2.2 Generative Approaches

Generative learning is another common way to learn the information of object classes. For a given data population, generative learning attempts to generate a parameterized statistical model to represent the data. Several generative models has been widely used in computer vision applications due to the simplicity structure, such as Principal Component Analysis (PCA) (43), the Gaussian Mixture Model (GMM) (26), and the naive bayes (47). However, these models oftern have poor performance due to several drawbacks. First of all, shape and appearance information of objects are statistically represented in generative models based on certain simple assumptions of data distribution. However, the underlying complexity of real data usually cannot be represented by those simple assumptions. Moreover, only

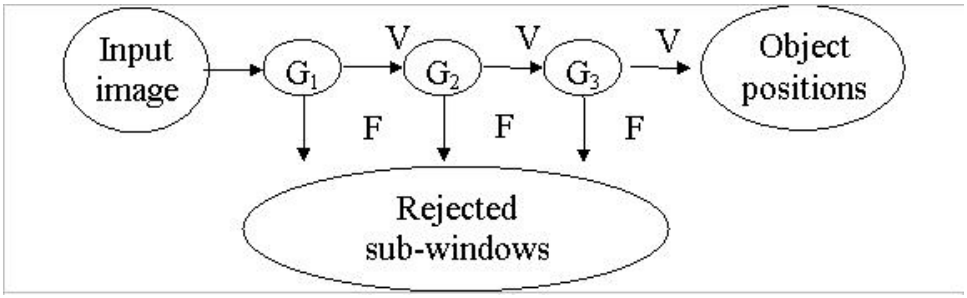

Figure 9: Cascade of Adaboost classifier. Each layer is a strong classifier with high detection rate and moderate false alarm rate. Detection windows classified as faces by the current layer will be tested on next layer and rest will be rejected as non-faces. After the whole cascade, the system will still maintain a good detection rate while having a very low false alarm rate.

positive examples are modeled in generative learning framework. As a result, the learned models could have difficulty in discriminating between positive and negative cases.

2.3 Part-based Approach

There is a significant body of work on part-based approaches. Seminal work of Felzenszwalb *et al.* (31) is based on mixtures of deformable part models (DPM). Ott and Everingham (67) extended the DPM framework by sharing parts among mixture components as well as object classes. Poselet framework proposed by Bourdev and Malik (17) demonstrated a state-of-the-art performance of people detection with rich part annotations.

Compared to holistic approaches, part-based approaches have many advantages in terms of face detection: Parts of the face are less sensitive than the whole face to the difference in appearance; Parts also have better tolerance to lighting variations and partial occlusion. Another inspiring work was proposed by Thomas *et al.*(82). Their system consists of a strong part-based object detector, referred to

as Implicit Shape Model, and a pose recognition system that enables vote from the same part to serve multiple poses. While achieving better performance than the holistic methods (25) and the constellation approaches (34), part-based discriminative approaches suffer from the data set limitations.

2.4 Exemplar-based Approach

Exemplar-based approaches (22; 56) are recently getting more attention. Instead of training sophisticated classifiers, these approaches simply compute similarities between an input image and pre-stored exemplars. This type of non-parametric techniques has demonstrated the ability to solve various computer vision problems. However, the performance in object detection tasks has been outperformed by the discriminative approaches. In our opinion, the decline in performance is due to the poor generalization ability of exemplar-based approaches, which is caused by the large intra-class variations and insufficient number of exemplars. To overcome the above problems, Malisiewicz and *et al.* (57) proposed a method that combines the exemplar-based approach with a discriminative framework. They trained each individual SVM classifier with one object exemplar versus millions of background images. This conceptually simple idea achieved promising results on VOC challenge. However, the computation complexity grows enormously due to the large number of exemplars that have to be examined during the testing stage.

2.5 Our Approach - VARIS

Compared with other systems, the proposed VARIS system combines two previously mentioned frameworks, the part-based approach and the exemplar-based approach and takes advantages from both. First of all, our system is part-based and has a flexible representation that allows for large geometrical

variations limited only by topological constraints. Such flexible representation makes our system very robust to occlusion. Second of all, our system measures similarities between input image and pre-stored exemplars. The performance of the system can be easily extended by adding more exemplars. Beyond these two major advantages, our system fills two requirements of object detection: Generalization and Reliability. The experimental results show that VARIS achieves better performance than most of State-of-the-Art systems in terms of detection rate versus false alarm rate.

CHAPTER 3

FEATURE INDEXING AND PART DEFINITION

VARIS inherits the characteristics of patch-based approaches. It segments exemplar images into a tessellation of small patches and represents the training data set as a combination of feature banks. Indexing one patch in the feature banks retrieves similar patches that are previously stored. Therefore, object class is determined by comparing the similarity between the input image and pre-stored exemplars. Moreover, we define the object part as an array of low level image patches and this definition has more flexibility than other approaches in obtaining global similarities by taking advantage of VARIS and compounding.

3.1 Feature Extraction

Inspired by the substantial properties of local rotation-invariant image features (51), many modern part-based approaches apply the interest point detector as the way to discover major object parts. However, the interest point detector always prefers the salient features such as eyes, nose and mouth while ignoring other less informative ones like weak edges. Therefore, if one or few parts are not detected due to blur or occlusion, the system performance drops drastically. On the other hand, the holistic detection system proposed by Dalal and Triggs (25) shows superior results with densely sampled features. Later Felzenszwalb *et al.*proposed a part-based approach (31) with similar representation and achieved the current state-of-the-art performance in general object detection. In this aspect, we apply the same strategy of densely sampling each training image into a normalized Histogram of Oriented Gradient (HOG)

grid map.

We extract a 31-dimensional histogram of oriented gradients (HOG) features to represent each patch area. It includes 18-dimensional contrast sensitive features, 9-dimensional contrast insensitive features and 4-dimensional texture features. Let $\theta(x, y)$ and $r(x, y)$ be the orientation and magnitude of the intensity gradient at a pixel (x, y) in an image. We compute gradients using the simple difference filters $[-1, 0, +1]$. Similar to (31), "we use the color channel with the largest gradient magnitude to define θ and r at each pixel. The gradient orientation at each pixel is discretized into one of the p values using either a contrast sensitive (B_1) or insensitive (B_2):

$$B_1(x, y) = \text{round } (\frac{p\theta(x, y)}{2\pi}) \mod p \tag{3.1}$$

$$B_2(x, y) = \text{round } (\frac{p\theta(x, y)}{\pi}) \mod p \tag{3.2}$$

The pixel-level feature grid map specifics a sparse histogram of gradient magnitudes at each pixel. Let $b \in 0, \ldots, p-1$ range over orientation bins. The feature vector at (x, y) is

$$F(x, y)_b = \begin{cases} r(x, y) & \text{if } b = B(x, y) \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

We can think of F as an oriented edge map with p orientation channels. For each pixel we select a channel by discretizing the gradient orientation. The gradient magnitude can be seen as a measure of edge strength. Let F be a pixel-level feature map for a $w \times h$ image. We define a dense grid of rectangular "cells" and aggregate pixel-level features to obtain a cell-based feature map C, with feature

vectors $C(i, j)$ for $0 \leq i \leq [(w-1)/k]$ and $0 \leq j \leq [(h-1)/k]$. This aggregation provides some invariance to small deformations and reduces the size of a feature map. Rather than mapping each pixel to a unique cell we use the "soft binning" approach where each pixel contributes to the feature vectors in the four cells around it using bilinear interpolation."

During the testing stage, features on the input image are indexed to find similar features from pre-stored exemplars. To build a better correspondence between the input image and exemplars, we set up a middle layer that dynamically assembles features into a part representation. Each object part $\mathbb{A}$ is defined as a vector array consisting of all HOG cells on the same row or column of the same exemplar. Rows are defined as object parts only when the height-width ratio of the exemplar is above 1. If the ratio is less than 1, columns are defined as parts instead. For simplicity reasons, in the following sections, we only consider the former case. Although we use part definition here, our indexing elements are still the HOG features.

3.2 Feature Indexing

Nearest neighbor (NN) search is a fundamental approach to establish the correspondences of local points by measuring their metric distances. Given a data point $x_i \in \mathbb{R}^k$, where k is the number of dimensions, NN search returns with a point that is closer to the query point than any other points in the data set. A well-known variant of NN search is the k-nearest neighbor search (k-NN). In computer vision area, the (k-NN) search has been widely applied to classify objects based on closest neighbors from the training examples. Since our main purpose in the indexing stage is to find similar features for the query feature, a k-NN search algorithm will satisfy the requirement.

We apply a kd-tree based technique to index the input feature vectors. Initially proposed by Bentley

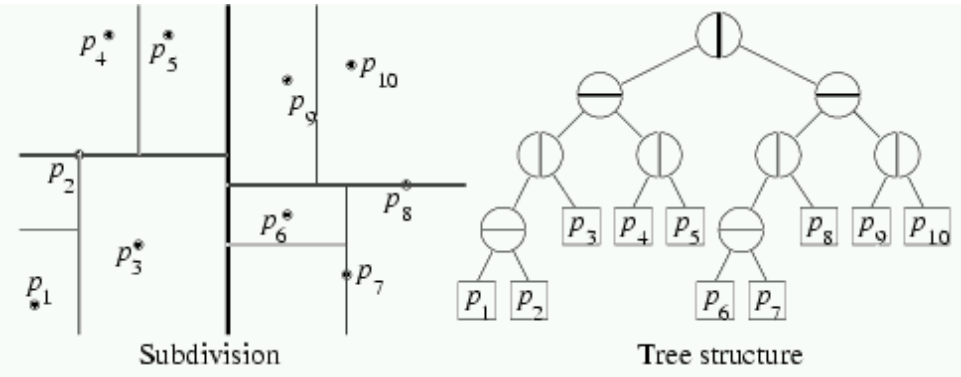

Figure 10: A kd-tree and the associated spatial subdivision.

(12), the kd-tree has been widely adopted to build a balanced binary tree data structure for nearest neighbor indexing. As stated in (78), "the elements stored in the KD-tree are high-dimensional vectors in R^k. At the first level of the tree, the data is split into two halves by a hyperplane orthogonal to a chosen dimension at a threshold value. Generally, this split is made at median in the dimension with the greatest variance in the data set. By comparing the query vector with the partitioning value, it is easy to determine to which half of the data the query vector belongs. Each of the two halves of the data is then recursively split in the same way to create a fully balanced binary tree. At the bottom of the tree, each tree node corresponds to a single point in the data set." The total height of the tree will be $\log_2 N$ where N is the number of points in the data set. Figure 10 shows a simple example of a built kd-tree.

(78) also states that "given a query vector, it requires $log_2 N$ comparisons down the tree and leads to a single leaf node. The data point associated with this first node is the first candidate for the nearest neighbor. It is useful to remark that each node in the tree corresponds to a cell in R^k. And a search with

a query point lying anywhere in a given leaf cell will lead to the same leaf node. The first candidate will not necessarily be the nearest neighbor to the query vector. It must be followed by a process of backtracking, or iterative search, in which other cells are searched for better candidates." kd-tree has shown efficient and accurate search ability in low dimensions. However, the efficiency vanishes when the data dimensions increase because the backtrack in high dimensions becomes computationally expensive. Therefore, we implement the priority kd-tree (8) for feature indexing, which keeps a high probability of finding the true nearest neighbors while the backtracking is limited. The search terminates when there are no more cells within the distance defined by the best point found so far (see Figure 11). There are many other nearest-neighbor search algorithms that have similar indexing functionality, and a thorough study can be found in (46).

Tree construction: Our initial training data set contains face exemplars in different poses. We manually separate the exemplars into subclasses according to their pose, and keep 10 to 15 exemplars for each subclass. HOG features in the exemplars are defined as $\mathbf{v}_{p,q}^{m,n}$, where p and q are the row and column indices of that feature, and m and n denote the subclass and exemplar indices. We have previously defined the part $\mathbb{A}$ as a one-dimensional array of feature vectors:

$$\mathbb{A}^{m,n}(p) = \{\mathbf{v}_{p,q}^{m,n} | q = 1, 2, \cdots, Q\} \tag{3.4}$$

We construct the kd-tree $T(p)$ with $\mathbb{A}(p)$ from all exemplars, and in total we get P kd-trees, where P is the number of total row arrays in one exemplar. Since exemplars are randomly selected under different

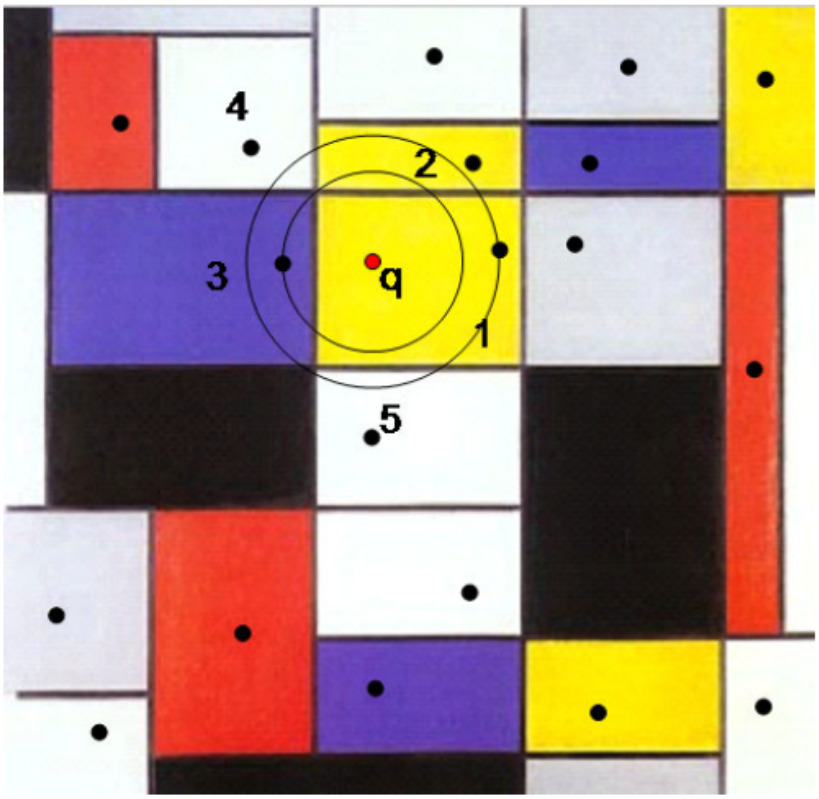

Figure 11: Priority search of a KD-tree. The red dot is the query point and its cell 3 has its true nearest neighbor. The search goes down the tree and retrieve the first node as the neighbor candidate (marked as 1 in this case). However, this point is often not the closest neighbor of the query point. A priority search proceeds in order through other nodes of the tree in order of their distance from the query point. In this image, it would be through node 2 to 5. The search is bounded by a hyper-sphere of radius r. When there are no more cells within this radius, the search terminates.

conditions, the parts are not well aligned. To better cope with this spatial variation, each kd-tree also includes adjacent row arrays. Then we redefine $T(p)$ as:

$$T(p) = \{\mathbb{A}^{m,n}(b) | m = 1, \cdots, M; n = 1, \cdots, N; \\ b = p - l, \cdots, p, \cdots, p + l\} \tag{3.5}$$

where $p > l > 0$ and $p + l < P$. l defines how much vertical shift that the part can tolerate.

Tree indexing: For each training exemplar, we create a matrix to preserve the indexing information. Given an input image I, we extract the HOG features first and then do a k nearest-neighbor search for each $\mathbf{x}_{i,j}$ within $T(i)$. The search returns with k nearest neighbors according to the Euclidean distance. If vector $\mathbf{v}$ is selected as $\mathbf{x}_{i,j}$'s neighbor, we add one entry to the corresponding element of the matrix with the indexing information: the row and column indexes, i and j, and the distance value. To abide by the principle of maximum aggregated similarity, we convert each distance value to a similarity score with the Gaussian function:

$$S(\mathbf{x}, \mathbf{v}) = exp(-\frac{1}{2\sigma^2} d(\mathbf{x}, \mathbf{v})) \tag{3.6}$$

where $d(\mathbf{x}, \mathbf{v}) = ||\mathbf{x} - \mathbf{v}||^2$ and σ is estimated in the validation step. We would like to comment here that aggregating similarity scores is much more effective than minimizing accumulated distances (36).

Our results show that HOG features are highly discriminative and most of the salient ones have neighbors falling into the same class as the input (see Figure 12). For the negative images, neighbors of the indexed feature vectors are more uniformly distributed over all subclasses. After the indexing stage, each element of the matrix may have zero, one or multiple entries depending on the similarities

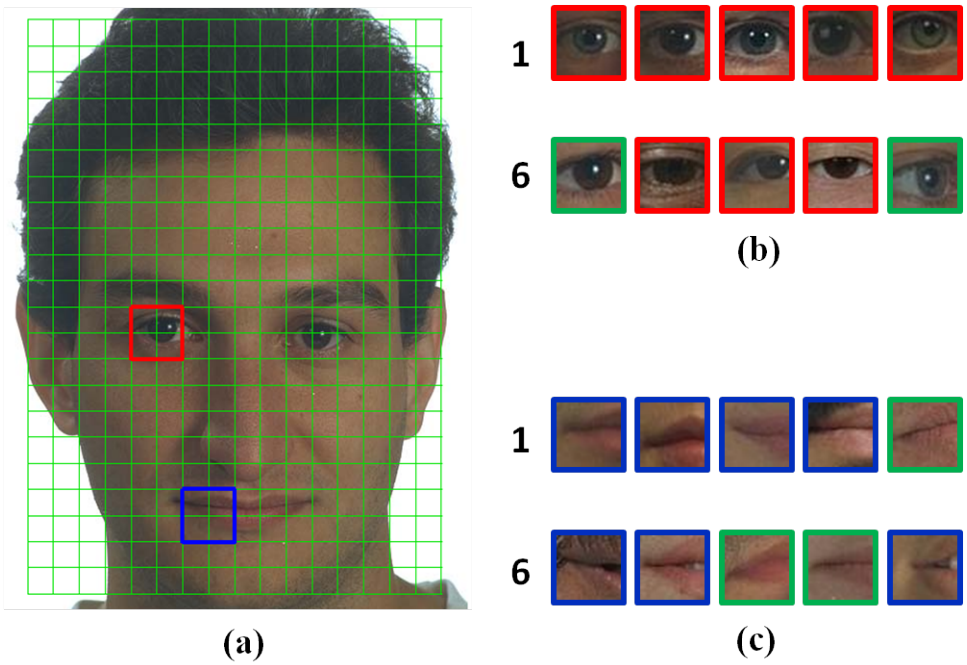

Figure 12: Examples of feature indexing. (a) A test image with overlapping HOG feature grid. The eye in red square and the mouth in blue square are two indexed patches; (b) 10 nearest neighbors of the indexed eye. The eyes in green squares (6 and 10) are from non-frontal face classes; (c) 10 nearest neighbors of the indexed mouth. The patches in green squares (5, 8 and 9) are from non-frontal face classes.

of the corresponding feature vectors. To further utilize the similarity information, a 2D sequencing step is applied to select the optimal feature set $\mathbb{W}$ from the non-empty elements. More details about the selection will be revealed in section 4.

3.3 Feature Importance Weighting

In each object class, some of the features can be more informative than others thus more discriminative for representing the object class. For example, eyes and nose are more salient features of human faces than cheek and forehead. Recognizing the salient features will have a better chance to correctly predict the object class. Thus, we should put different weights on different features. This assumption holds in particular for dense sampling of features, which tends to produce better performance than interest points in many classification tasks (73; 63; 84).

Since our approach only checks the similarities between the input image and positive exemplars, the max-margin framework (55; 57) cannot be adopted here to learn the feature weights. In addition, VARIS dynamically selects the optimal feature subset during the testing stage, therefore a traditional learning framework for the part-based approach (19; 37) is not feasible either. To emphasize discriminative features in an exemplar, we simply use the power of the gradient magnitude to represent the feature importance. Because our training data set has a limited number of exemplars, we can manually remove high power features that are outside the face, to make the weighting even more precise. During the experiments, we observe that the false positive cases emerge when many non-salient features are indexed with high-similarity neighbors, which also contribute enough similarity scores to the final sum-

mation. To alleviate this effect, we set a threshold t_w that turns the similarity scores of those low power features to negative scores. As a result, equation 3.6 is modified to:

$$S(\mathbf{x}_{i,j}, \mathbf{v}_{p,q}^{m,n}) = \begin{cases} G_{i,j} \times G_{p,q}^{m,n} \times exp(-\frac{1}{2\sigma^2} d(\mathbf{x}_{i,j}, \mathbf{v}_{p,q}^{m,n})) \\ \qquad \text{if } G_{p,q}^{m,n} \geq t_w \\ \\ -G_{i,j} \times G_{p,q}^{m,n} \times d(\mathbf{x}_{i,j}, \mathbf{v}_{p,q}^{m,n}) \\ \qquad \text{if } G_{p,q}^{m,n} < t_w \end{cases} \tag{3.7}$$

where $G_{p,q}^{m,n}$ is the power of gradient magnitude for each $\mathbf{v}_{p,q}^{m,n}$ and $G_{i,j}$ is the corresponding power for each $x_{i,j}$. Therefore, the salient features contribute positive scores to the total similarity while the non-salient features, on the other hand, contribute negative scores or 0 in the best case. This negative weighting mechanism significantly reduces the false alarm rate of the detection.

CHAPTER 4

SEQUENCING AND EXEMPLAR COMPOUNDING

In the previous indexing step, we create a similarity comparison mechanism at the local feature level. Each input patch finds k neighbors by indexing within the kd-tree. To achieve the detection goal, we need to organize the indexing information and optimize the similarity at the global level. Since previous work (13; 31) have demonstrated that geometric information plays an important role in part/patch based object detection approaches, the global similarity optimization in our approach is constrained by the requirement of preserving the topology of both input and exemplar images. In this chapter, we introduce a novel 2D sequencing approach based on dynamic programming that searches for optimal global similarity. Furthermore, we invent another novel mechanism called exemplar compounding that further optimize the similarity between input and multiple exemplar images.

4.1 Dynamic Programming

Our 2D sequencing approach is based on dynamic programming(DP) (9). As stated in (33), "dynamic programming is a powerful general technique for developing efficient discrete optimization algorithms. In computer vision it has been used to solve a variety of problems including curve detection (2; 40; 62), contour completion (77), stereo matching (5; 65), and deformable object matching (3; 6; 23; 30; 32).The basic idea of dynamic programming is to decompose a problem into a set of subproblems. An important aspect of dynamic programming is that the solution of a single subproblem is often used multiple times, for solving several larger subproblems. Thus in a dynamic programming

algorithm we need to "cache" solutions to avoid recomputing them later on. this is what makes dynamic programming algorithms different from classical recursive methods."

4.2 1-D Sequencing with Dynamic Programming

Let $A = (a_0, \ldots, a_{n-1})$ and $B = (b_0, \ldots, b_{m-1})$ be two sequences of elements. We try to find a set of correspondences between A and B that maintains the natural ordering of the elements. In the computer vision area, this is an important problem that arises both in stereo matching (5; 65) and in matching deformable curves (6).

Intuitively we want to consider correspondences between A and B with following spatial restrictions,

- A strict one-to-one matching is applied. In particular, if a_i has multiple matched b_j and one b_j has been selected to be in the sequence, then other corresponding b_j cannot be included.
- Any two matchings cannot lie across each other. For example, if an input vector a_i is selected to match b_j, then next a_{i+1} can only match to b_{j+t}, where $t \geq 1$.

Figure 13 demonstrates those restrictions. Similar approaches are commonly used in biology for comparing DNA sequences. It has also been used in speech recognition (35) and human activity recognition (11), where the features can be represented by 1-D array of multi-dimensional vectors.

The purpose of sequencing step is to find the optimal correspondence between feature sets A and B. The brute-force method would be to generate all the possible correspondences and then find the maximum or minimum, depending on the problem. But if A has n elements in it we are looking at a search space of size 2^n. Oftentimes n is huge making a brute-force method computationally infeasible.

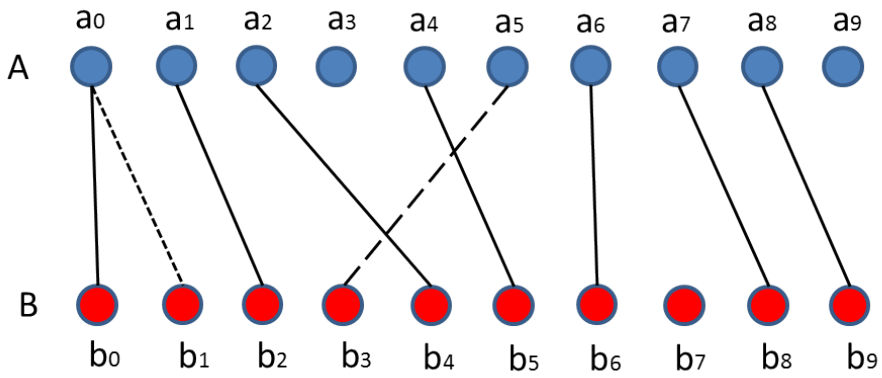

Figure 13: Spatial restrictions of 1D dynamic programming.$a0$ to $b1$ is a violation of the first restriction. $a5$ to $b3$ is a violation of the second restriction.

The dynamic programming approach breaks the whole sequencing problem into different subproblems. Once the solution to the given subproblem has been computed, it is stored or memorized. The next time the same solution is needed, it is simply looked up. Because we only need to traverse the memory once, the run-time is reduced to $O(n)$ from $O(n^2)$.

4.3 2-D Sequencing of VARIS

In the previous step, each input vector $\mathbf{x}$ finds k neighbors by being indexed within a kd-tree. The mapping information is recorded in the data matrices. Since one input feature vector $\mathbf{x}$ could match multiple feature vectors $\mathbf{v}$ in the same exemplar and only one such matching is allowed, we want to find

an optimal subset $\mathbb{W} \subseteq \mathbb{V}$ that maximizes the similarity with the input features $\mathbb{X}$ for each exemplar's feature set $\mathbb{V}$.

$$\hat{\mathbb{W}} = \arg\max_{\mathbb{W}} \sum_{\mathbf{v} \in \mathbb{W}} S(\mathbf{x}, \mathbf{v}) \tag{4.1}$$

where $S(\bullet)$ is the similarity function in equation 3.7. The optimization is constrained by the requirements of preserving the topology of both input image and exemplars.

An unsophisticated way of finding the optimal set is to examine all the combinations and choose the one with the highest value, but this is impractical due to the large computational cost, where he time required of the brute-force method can be as high as the number L of possible part placements to the power of the number of parts P, *i.e.* $O(L^P)$. In this report, we implement a 2-D sequencing approach to search for the optimal set $\mathbb{W}$, where the cost can be reduced to $O(PL)$. Since we may have negative similarity scores from the previous step, the traditional dynamic programming that maximizes the total score is no longer valid. We make a tradeoff between the similarity and the topology, where the system takes the negative scores into account and maximizes the feature similarity as well as the topology:

$$\hat{\mathbb{W}} = \arg\max_{\mathbb{W}} \sum_{\mathbf{v} \in \mathbb{W}} S(\mathbf{x}, \mathbf{v}) + \sum_{\mathbf{v} \notin \mathbb{W}} L(\mathbf{v}) \tag{4.2}$$

where $L(\mathbf{v}) = \alpha \times G_v$ is a loss function that penalizes the final similarity score if $\mathbf{v}$ is not selected in the sequence.

The approach is 2-D because the same sequencing search runs recursively in horizontal and vertical directions. We first implement the horizontal dynamic programming searching for an optimal sequence of features that best represent object rows (parts). After the horizontal search, each row of input image

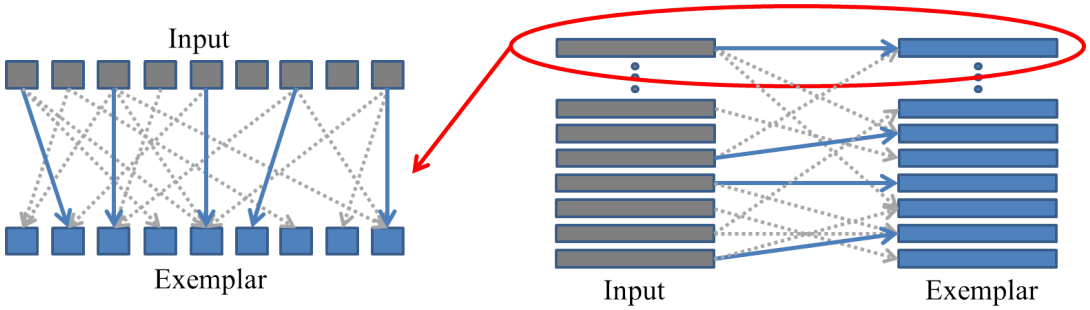

Figure 14: 2D sequencing in the geometry verification stage. Left image shows the horizontal sequencing at image patch level between input and one exemplar. Right image shows the vertical sequencing for multiple rows of the input image and the exemplar. Both sequencing approaches are subject to two spatial constraints mentioned in Figure 13. The blue solid lines are the optimal selections maximizing the similarities among all the mapping relationships that are labeled with gray dash lines.

has multiple matching rows in each exemplar. Then we proceed the vertical search to select the optimal sequence of the rows (See Figure 14). The advantage of the 2D sequencing approach is that the geometrical flexibility is achieved by dynamically assembling local features during the testing phase. Those deformable templates have large tolerance to the spatial variation and occlusion.

4.4 Exemplar Compounding

One of the most successful object detection approaches is the deformable part-based model (DPM) proposed by Felzenzwalb *et al.*(31). The DPM method partitions the object model into a set of local parts, which can appear in different image regions with soft spatial constraints. Moreover, DPM learns Different detectors for different aspects of an object, such as frontal/profile viewpoint. By using the idea of mixture models, DPM has large invariance to object intra-class appearance differences. While the mixture model used by the DPM has been shown state-of-the-art performance in object detection,

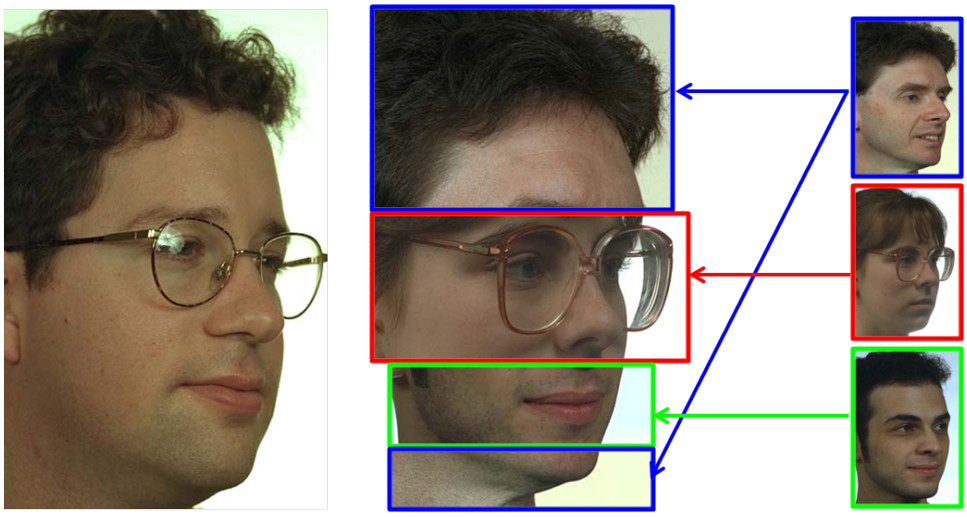

Figure 15: Example of a compound exemplar. Left: test image; Middle: the compound exemplar that is assembled with parts from different exemplars of the same pose class. It has a higher similarity with the test image than each of the exemplars on the right; Right: original exemplars with partial similarities to the input.

further improving accuracy by increasing the number of mixture components is not straightforward. Since the number of images in the training data set is fixed, the amount of training data for each model decreases linearly with the number of mixture models. Therefore, larger number of mixture components results in poor generalization.

Similar to other part-based approaches, VARIS has a larger degree of tolerance to spatial variations due to the fact that the 2-D sequencing step dynamically assembles exemplar parts and allows parts to move inside the exemplar subject to geometrical constraints. Furthermore, to better utilize the information of the training data set, we propose a novel compounding procedure (54; 53). The compounding procedure allows VARIS to have an assembled exemplar with higher similarity while keeping a low

number of exemplars in the training data set (see Figure 15). This idea is inspired by the observation during our experiment that many exemplars are partially similar to the input image but none of them has a distinguishing score that can be positively recognized. Although the compounding approach increases the final similarities of the true positives, the scores of negative inputs are also raised. Therefore, we set a restriction that exemplars could offer parts to be compounded only when their similarity scores pass a pre-defined threshold t_c, where t_c is usually set to a value two times smaller than the final detection threshold. The diagram in Figure 16 illustrates the full 2D sequencing and the compounding procedures.

After the 2-D sequencing step, the similarity scores of several exemplars that belong to the same object class may pass the threshold t_c. We call them exemplar candidates. The compounding procedure searches for an optimal combination of parts from different candidates where the new similarity score is maximized while the geometrical constraints are still maintained. In our current research, the compounding procedure only takes effect among exemplars that belong to the same sub-category, *e.g.* the same pose. Although compounding parts of exemplars in different pose may result a better performance, it increases the computational complexity as well as the chance of getting more false alarms.

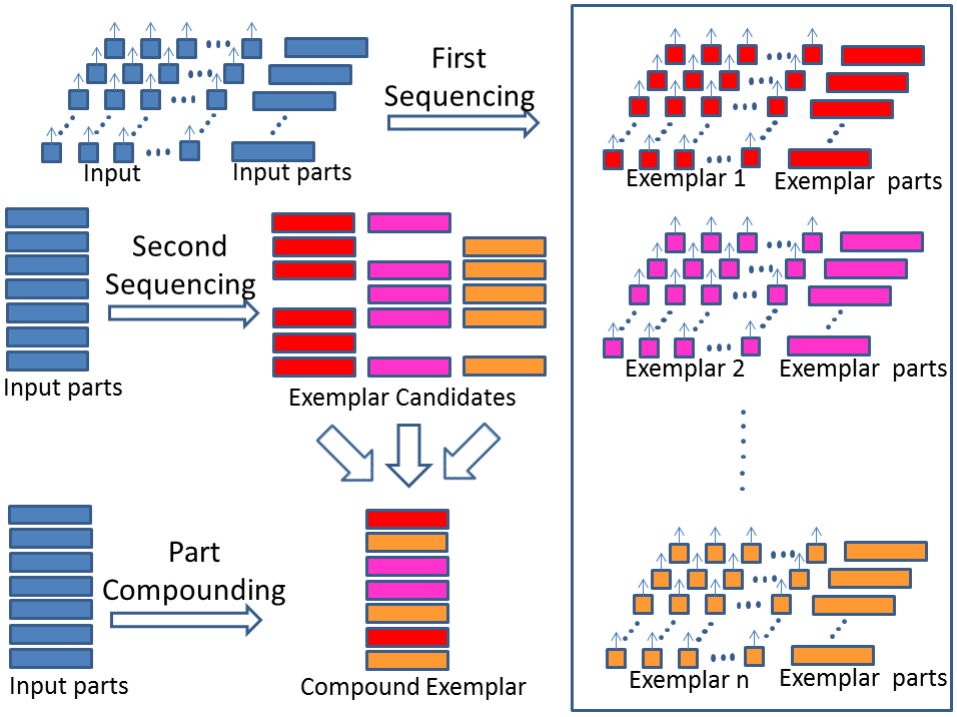

Figure 16: Diagram of 2D sequencing and compounding procedures. In the first phase, the 1D sequencing searches for the optimized exemplar rows matching to each input row. In the second phase all input and exemplar rows are assembled into columns. Next, 1D sequencing finds the optimal matching of the input column to each exemplar column. Compounding takes the best rows from different exemplars and creates a new exemplar that has a higher similarity.

CHAPTER 5

FACE DETECTION AND POSE RECOGNITION

Face detection is one of the fundamental techniques that enables human-computer interaction. It is also the initialization step to all other algorithms related to facial analysis, including face recognition, expression recognition, gender and age recognition, head pose tracking and so on. While face detection appears to be a trivial task for humans, it turns to be a challenging task for computers. In this chapter, we demonstrate the experiments of multi-view face detection implemented by our VARIS system with exemplar compounding. We also compare our results with several current state of the art methodologies.

5.1 Data Preparations

In the face detection experiment, the training images are from two separate sources, the color FERET data set (69) and the FDDB data set (42). We evaluate our system with the FERET data set for pose recognition first, and then conduct experiments with the FDDB data set for face detection. We also compare our results with several baselines. All the results shown in this section are averaged over 10 runs on a 2.4GHz Intel i7 processor with 8GB memory. The major part of the algorithm is written in Matlab $2010b$ without extra optimization and run in a single thread.

The initial exemplars of the training data set are selected from the FERET data set. The full data set consists of 7810 single face images with face poses spanning from -90° to 90° in yaw. It also provides ground truth of pose angle and locations of eyes, nose and mouth for each face. Based on ground truth, we use different rectangle bounding-boxes to extract face exemplars. We manually define 9 subclasses to

represent 9 different yaw angles, which are $0°$, $\pm 20°$, $\pm 45°$, $\pm 70°$ and $\pm 90°$ and fill each subclass with 10 to 15 face exemplars. Since this initial data set is insufficient to cover all face poses, we iteratively run the algorithm on a validation data set to fill up the missing cases. The validation data set is selected from the FDDB data set.

The FDDB data set includes 2845 real life images with 5171 faces (labeled with ellipses as ground truth) originally partitioned into 10 fixed folds. In our experiments, each time we use one of the folds as the validation set and average the results of 10-fold cross validations. New exemplars that improve system performance are added to the final training set. To avoid the overfitting problem and reduce the computational cost, we limit the total number of exemplars stored in the final training set to 200. If the number is exceeded, we replace the least used exemplar with a new one from the rest of the current validation fold and continue the iteration until no further improvement is obtained. After the validation stage, we have 200 exemplars in about 18 poses. Figure 17 shows some face examples of the final training set.

5.2 Pose Recognition

To evaluate the performance of our system, we first run the pose recognition test on the FERET data set. All the training exemplars are normalized to 96×72 pixels and then represented by 31-dimensional HOG features as in (31). The pose of the winning subclass is compared against ground truth and marked as a correct recognition if the deviation is within $10°$. Figure 18 shows the confusion matrix of the results. The average pose recognition rate is around 94%.

We also evaluate the ESVM approach proposed by (57) with the FERET data set. All images in one fold (about 520 faces) of the FDDB data set are selected as the positive training exemplars. To make

Figure 17: Examples of face exemplars. The final training data set includes human faces of different genders, ages, races, poses and appearance.

a fair comparison, we add the non-duplicated exemplars of our final training set to ESVM's training set. Then we use a second fold as the validation set to calibrate ESVM. The negative data is carefully selected from non-face images. Table 1 shows that VARIS with compounding completely defeats the ESVM approach in the discrete pose recognition test.

In the pose recognition experiment, we are also concerned with the effect of the parameter k on the system's recognition rate and the computational speed, where k determines the number of returned nearest neighbors for each input feature vector. Since the indexing step of VARIS is highly efficient

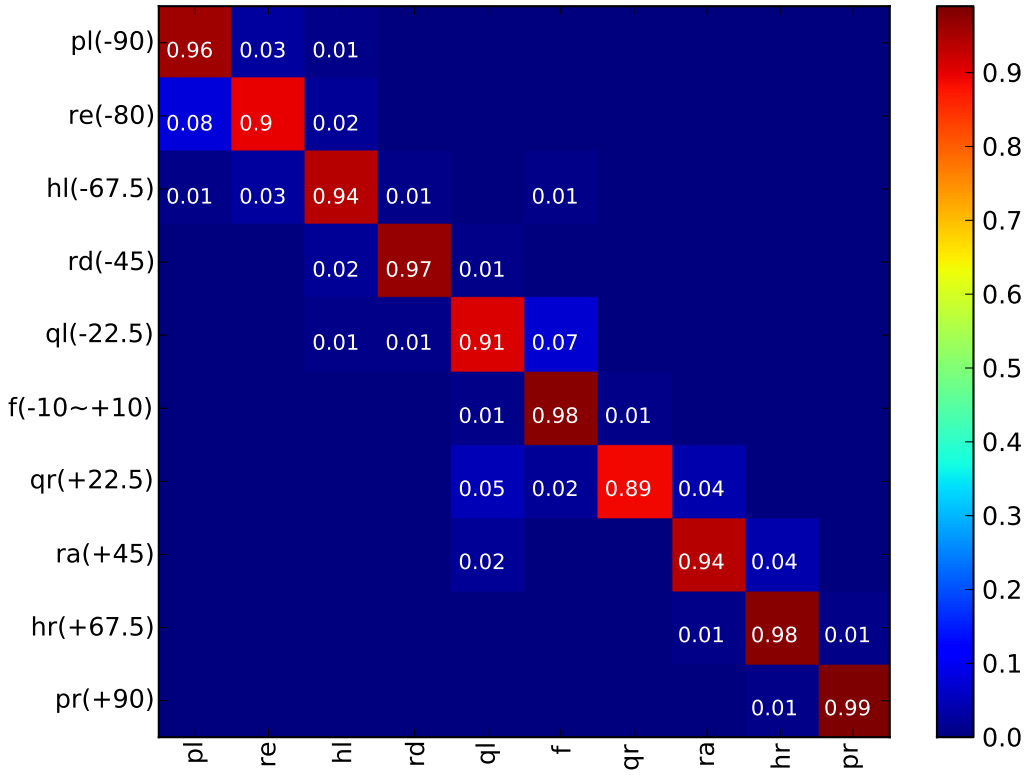

Figure 18: Pose recognition results. The confusion matrix shows the correct and incorrect pose estimations of VARIS with compounding on the FERET face dataset.

and it only takes a few milliseconds to look up the neighbors, most of the computation time is spent on the sequencing step. A larger value of k might increase the probability of correct recognition but will definitely increase the computational complexity. Figure 19 shows the recognition rates versus the computational time at different values of k.

Method	-90°	-80°	-67.5°	-45°	-22.5°	$0^\circ \& \pm 10^\circ$	$+22.5^\circ$	$+45^\circ$	$+67.5^\circ$	$+90^\circ$
VARIS	96%	90%	94%	97%	91%	98%	89%	94%	98%	99%
ESVM	87%	80%	84%	92%	84%	93%	81%	83%	87%	91%

TABLE I: Pose recognition comparison between VARIS with compounding and ESVM.

5.3 Face Detection

In the multi-view face detection experiment, we test our system on the FDDB data set. We use the same fold of the data set that is used in pose recognition experiment and separate the rest as testing images. During the testing stage, a 96×72 sliding window with 25% overlapping is applied to scan the input image in scale-space for possible face locations. A 320×240 testing image returns with 800 detection windows in 10 scales. Since VARIS has large tolerance to spatial variations, the number of scanning windows is significantly reduced. In the end, the standard non-maxima-suppression approach is applied to remove overlapping detections. We set $\sigma = 0.02$ (Eq. 3 & 4), $t_w = 0.1$ (Eq. 4), $\alpha = -0.25$ (Eq. 5) and $t_c = 0.2$ for all exemplars. We also set k equal to 220 for a higher computational efficiency. To further improve the detection speed, we prune exemplars that have few matching features after the indexing stage. Some detection results are displayed in Figure 22 and 23.

To validate the performance of VARIS with exemplar compounding, we compare it with the ESVM approach and the LSVM (31) approach in the context of multi-view face detection. The LSVM is trained with thousands of positive images from different sources, including the training data set of VARIS, and a negative data set. The ESVM uses the same data set from the pose recognition section. We implement these two systems both in their authors' default configurations. We also add a few results from some

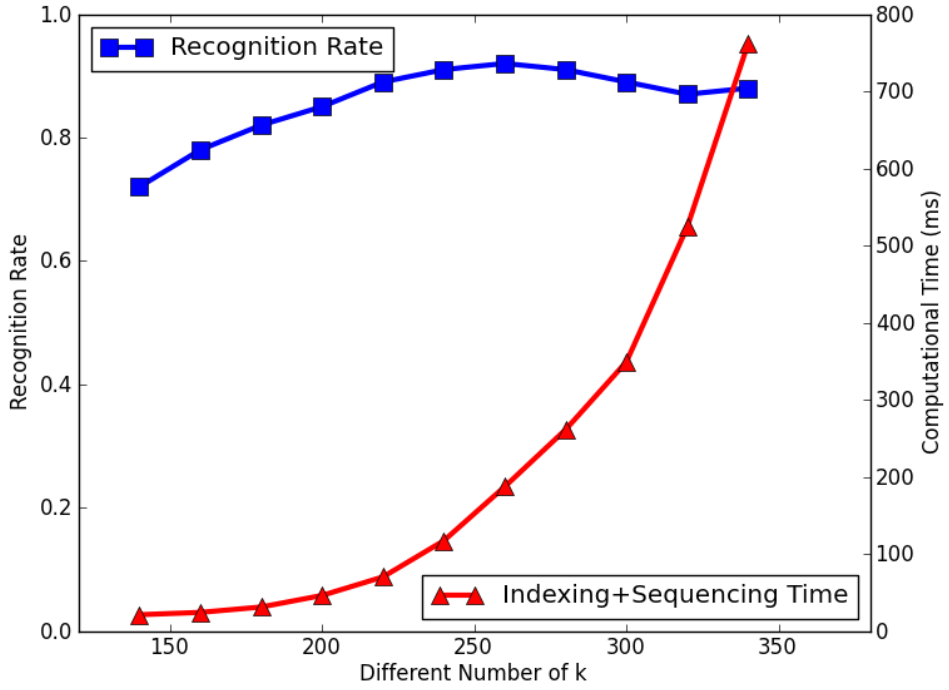

Figure 19: Computational efficiency. Computation time and recognition rates are measured under different settings of k. The time is the summation of indexing and sequencing steps for all exemplars.

previous work as baselines[1], which includes Viola-Jones face detector (44), Mikolajczyk *et al.*'s face detector (60) and Subburaman and Marcel's face detector (81).

We adopt the same evaluation criterion as in (42) that represents the degree of matching between a

[1]The results have been released with the FDDB data set and are available at vis-www.cs.umass.edu/fddb/results.html

detection bounding-box (bb_i) and ground truth (gt_j) by using the ratio of intersected regions to joined regions:

$$M(bb_i, gt_j) = \frac{region(bb_i) \bigcap region(gt_j)}{region(bb_i) \bigcup region(gt_j)} \tag{5.1}$$

Based on our detection results and ground truth, two Receiver Operating Characteristic (ROC) curves are generated by the FDDB evaluation toolkit: Discrete Score (DS) and Continuous Score (CS). The DS considers the estimated bounding-box as a correct detection if the overlapping is more than 50% with the ground truth, otherwise a false positive detection is granted. The CS directly draws the curve according to the real-value result The estimated bounding-box is considered as a correct detection if the overlapping is more than 50%, otherwise a false positive detection is granted. Figure 20 and 21 shows the ROCs of the approaches mentioned above as well as two different versions of VARIS, one with compound exemplars and the other without. The curves show that VARIS achieves better detection results than the compared approaches. We also note that the exemplar compounding scheme is instrumental to VARIS in achieving high detection rates.

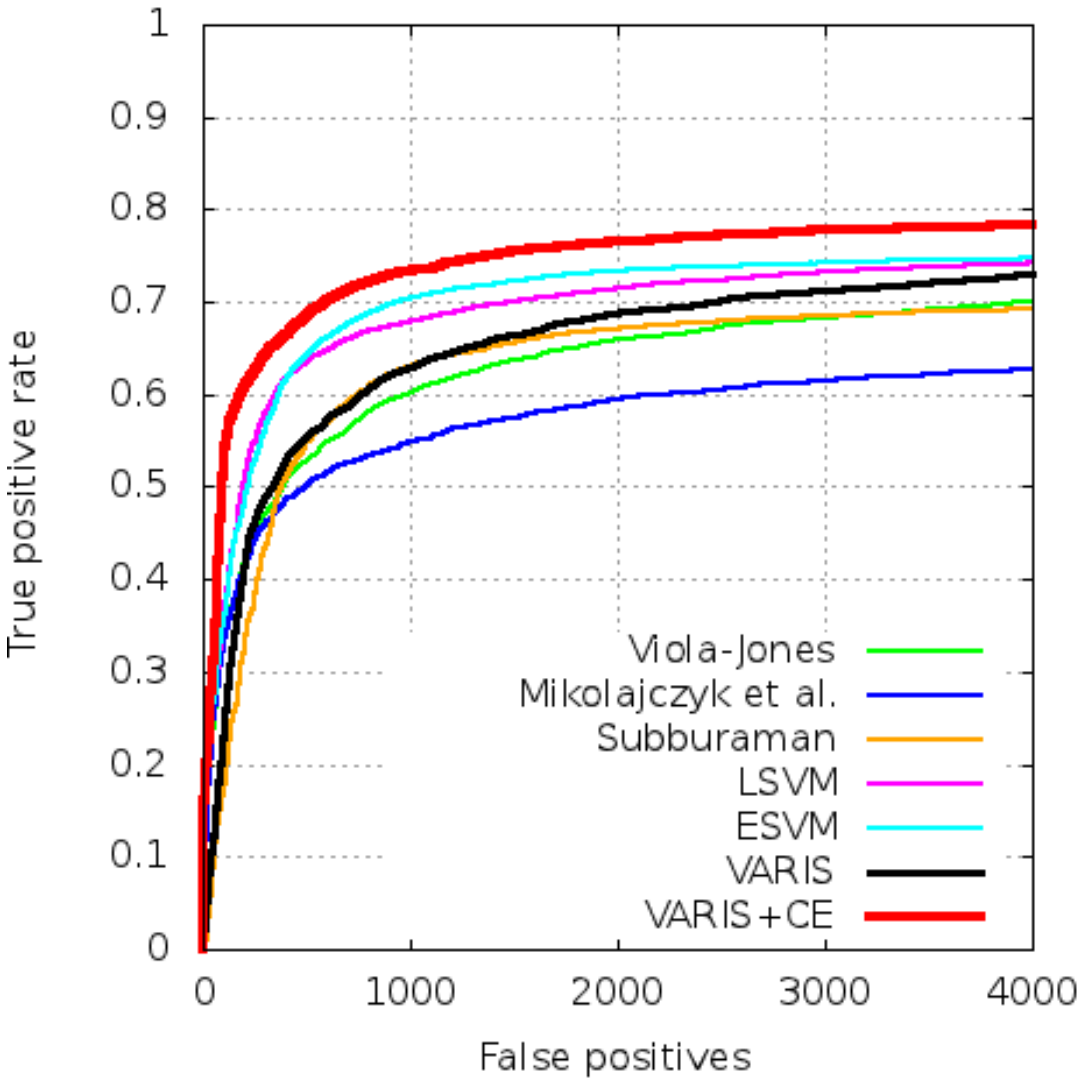

Figure 20: Discrete ROC curve. VARIS+CE is VARIS with compound exemplars.

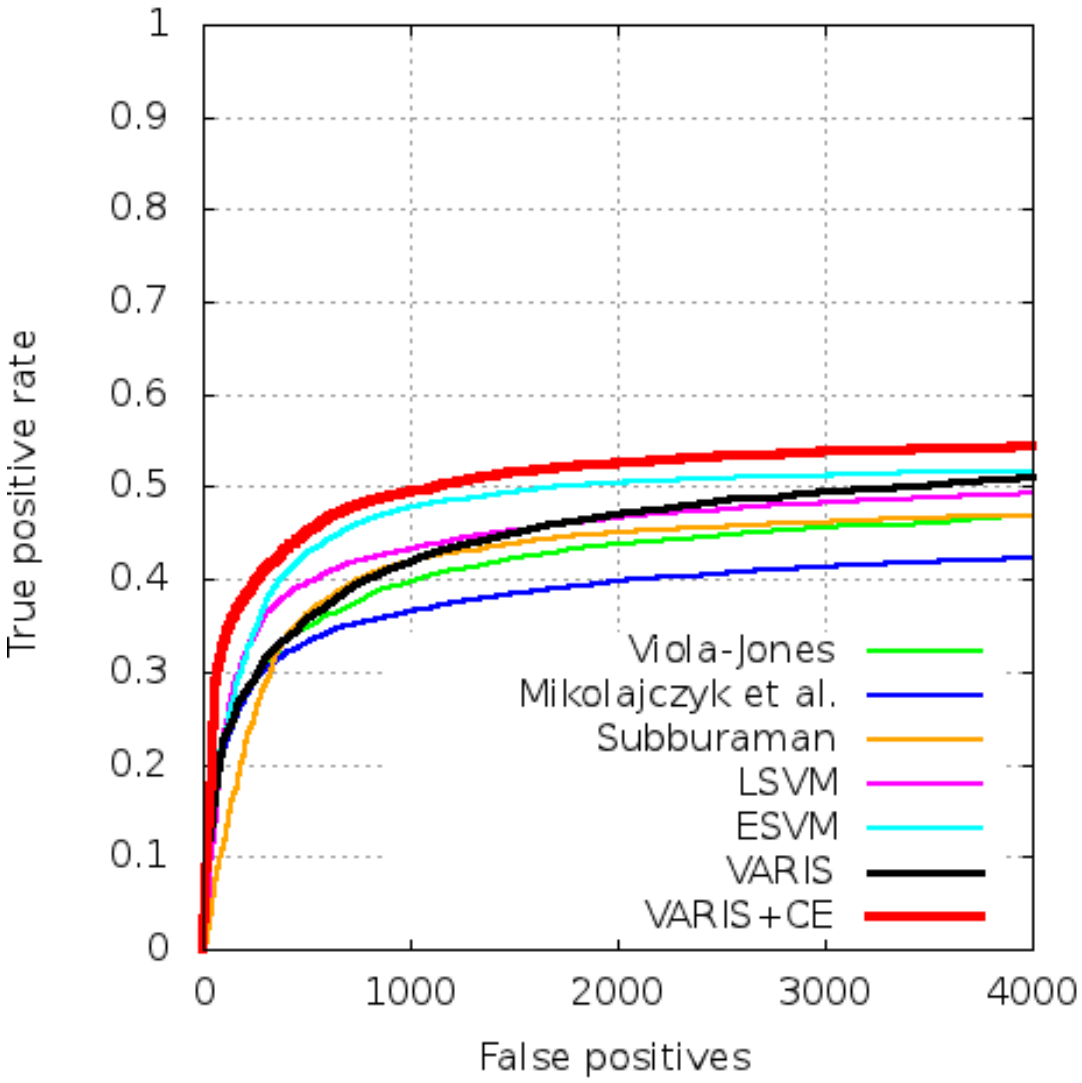

Figure 21: Continuous ROC curve. VARIS+CE is VARIS with compound exemplars.

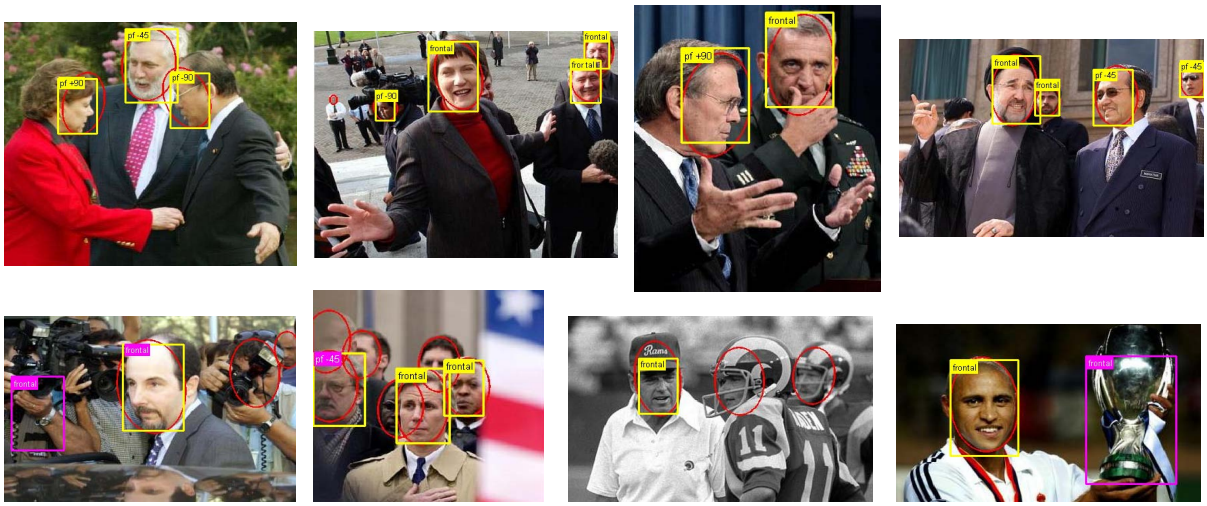

Figure 22: Detection results. The ground truth is labeled by red ellipses and our detection is labeled by squares. First row shows the correct detections where both face locations and face poses are correctly estimated. Second row shows the cases in which VARIS doesn't work well. Faces labeled only by red ellipses are the missed cases. The magenta texts label the faces that are correctly detected but the poses are not correct. The magenta text and square denote the false detection cases.

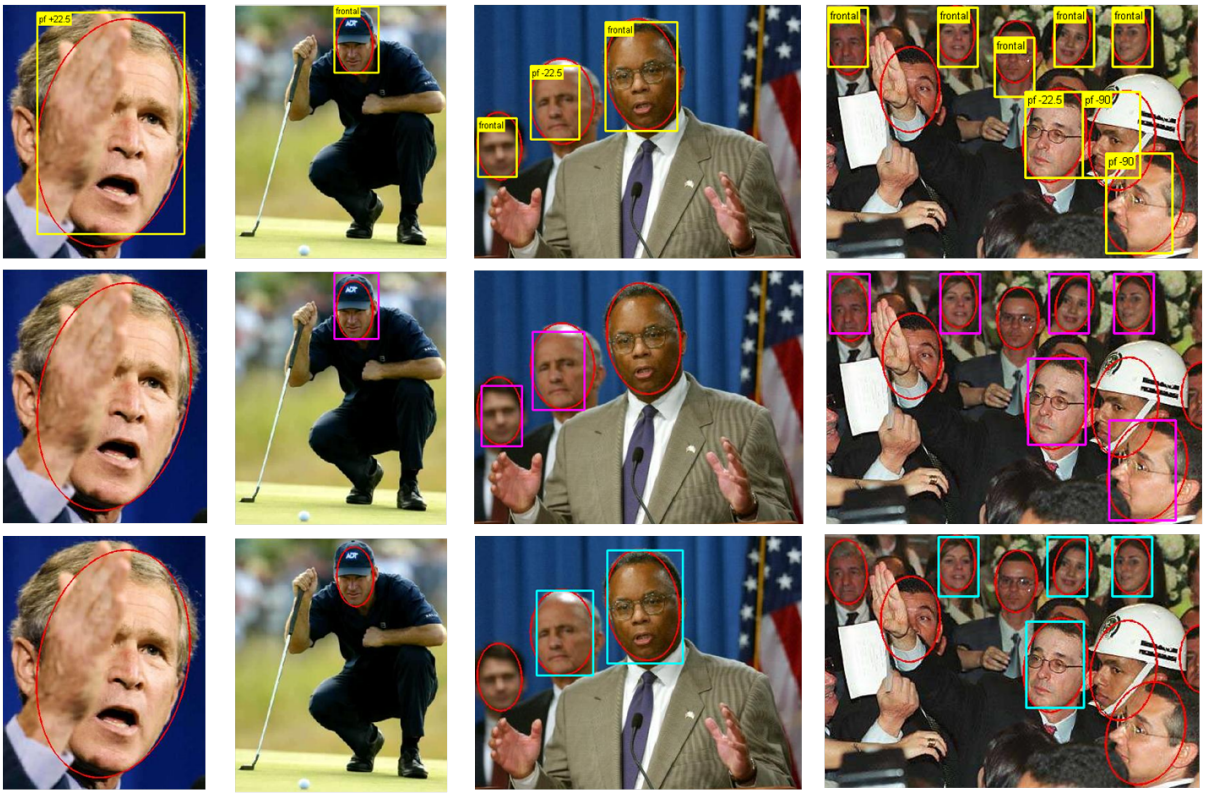

Figure 23: Detection comparisons between VARIS and other approaches. First row shows results of VARIS. Second row shows results of ESVM algorithm. Third row shows results of LSVM algorithm.

CHAPTER 6

GENERAL OBJECT DETECTION

Computer vision approaches currently cannot reliably detect, segment and parse objects in cluttered images, especially for those deformable object classes. Here the parsing means the ability to recognize object parts that is useful for further applications. Although there have been many approaches that demonstrate some degrees of success, none of them have reached the performance level of human recognition system. Fortunately, various methods are proposed by the community everyday and their performance show steady progress. In this chapter, we extend our VARIS plus exemplar compounding system to general object detection area. The experiments confirm that our system has better performance than the current state-of-the-art discriminative methods.

6.1 Data Preparations

For performance evaluation, we use the PSACAL (Pattern Analysis, Statistical Modeling and Computational Learning) Visual Object Classes (VOC) challenge 2007 data set (28). "The goal of the challenge is to detect objects from a number of visual object classes in realistic scenes. It is fundamentally a supervised learning problem in that a training set of labeled images is provided. The training data provided consists of a set of images; each image has an annotation file giving a bounding box and object class label for each object in one of the 20 classes present in the image. The full data set includes 9963 images containing 24640 annotated objects. It is separated into three sub sets, the training (`train`), evaluation (`val`) and testing (`test`) set."

In our experiment, the `train` set is the data source where the exemplars are extracted. The `val` set is used to validate the system performance during the exemplar selection step. The `test` set is for evaluating the final performance. Here we follow the same precision/recall criteria of 50% overlapping area with ground truth. We also compare our system performance with DPM (31) and ESVM (57).

6.2 Training Stage

We randomly select 150 object exemplars from the `train` set at the beginning for most of the object classes and 200 for those difficult classes such as birds, plants and so on. We iteratively validate the system performance on the `val` set, replacing the least used exemplar of each class with a new exemplar from the rest of the `train` set. The new exemplars are kept in the tree structures only if the performance is improved. The iteration continues until no further improvement is achieved.

Different from discriminative frameworks that learn model parameters from the training data set, our approach in the training stage tries to solve an optimization problem of choosing a small portion of data set that could effectively represent the object class. The ideal representation set should have object exemplars that cover the common features of the object class and enlarge the detail variance as much as possible. Our experiments confirm that the final training set obtained by the learning process consists of exemplars with a large appearance variance; meanwhile a great amount of exemplars are removed from the final training set because similar ones has been selected in the set.

6.3 Testing Stage and Results

We use the same parameter configuration as in (31) to retrieve the HOG pyramid for the input image. For the exemplars, the HOG features are extracted only at the root scale. Images in the `train` set are divided into 10 sub-classes according to their height-width ratios. We set $l = 2$ (Eq.3.5), $\sigma = 0.05$

TABLE II: Comparisons of average detection precision on VOC2007 data set. ESVM and DPM are implemented with authors' default settings. VARISEC stands for VARIS with exemplar compounding. The highest scores are highlighted in bold.

	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbik	pers	plant	sheep	sofa	train	tv
ESVM	.208	.401	.089	.105	.102	.347	.406	.097	.106	.146	.024	**.122**	.371	.320	.184	**.102**	.152	.120	.295	.303
DPM	.287	.551	.006	.145	**.265**	.397	.502	.163	**.165**	.166	.245	.050	.452	.383	.362	.090	**.174**	.228	.341	.384
VARIS	.197	.425	.002	.131	.009	.279	.347	.088	.007	.126	.136	.048	.314	.337	.208	.071	.102	.153	.318	.284
VARISEC	**.337**	**.599**	**.112**	**.176**	.104	**.415**	**.539**	**.184**	.149	**.174**	**.298**	.091	**.472**	**.400**	**.403**	.083	.163	**.238**	**.406**	**.425**

(Eq.3.7), $t_w = 0.1$ (Eq.3.7), $\alpha = -0.25$ (Eq. 4.2) and $t_c = 0.3$ for all exemplars. For each indexing feature vector, we retrieve 250 nearest neighbors. The sliding-window is used to scan the input image in multi-scales. The compounding procedure is activated only on exemplars that belong to the same class and subclass. We use the same non-maxima-suppression approach to remove duplicate detections.

Quantitative detection results and comparisons are listed in Table II. We evaluate the VARIS system with and without compounding procedure separately. VARIS without compounding achieves an average precision (AP) score of 0.18 over 20 classes. The compounding procedure increases system AP score from 0.18 to 0.29, which outperforms DPM (0.26) and ESVM (0.20). The performance is significantly improved by the compounding procedure for most deformable object classes, especially for aeros, birds and persons. In the comparison with DPM and ESVM, our approach wins in 15 out of the 20 categories. However, the AP scores of some classes are still relatively low due to the large appearance variations of object instances. Fortunately, our method can alleviate this situation simply by importing more exemplars into the training set. Moreover, our system is fairly efficient even without parallel computation and extra optimization. It is 3 times slower than the DPM approach and 2 times faster than ESVM. Some detection results of 20 VOC classes are shown in Figure 24, 25, and 26. In figure 27, we also show

results where our system fails. In general, the failure is caused by either the object is too small to detect or the object is heavily occluded. There are also some mis-classification cases shown in Figure 27, such as a jumping human is detected as a bird, horses are detected as cows or flying birds are detected as airplanes. Since we only use gradient cue that captures the shape information from objects, the system cannot differentiate two objects with similar contours. To deal with these hard cases, we can create new binary classifiers that can be turned on only we both classes have close similarity scores.

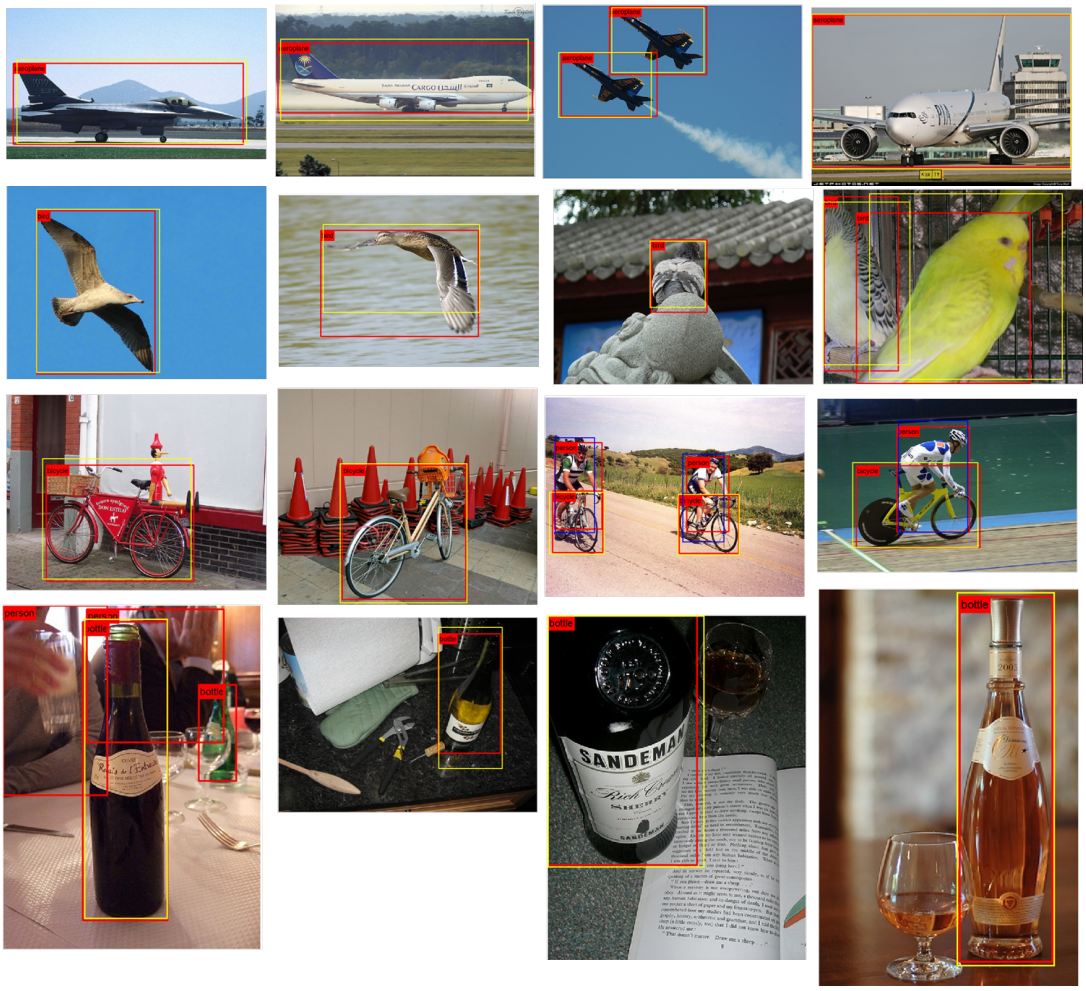

Figure 24: Detection results of airplane, bird, bicycle and bottle classes. Red rectangle labels the ground truth and yellow rectangle labels detections of VARIS plus compound exemplar. If there are classes in the same image, the second class will be labeled in blue.

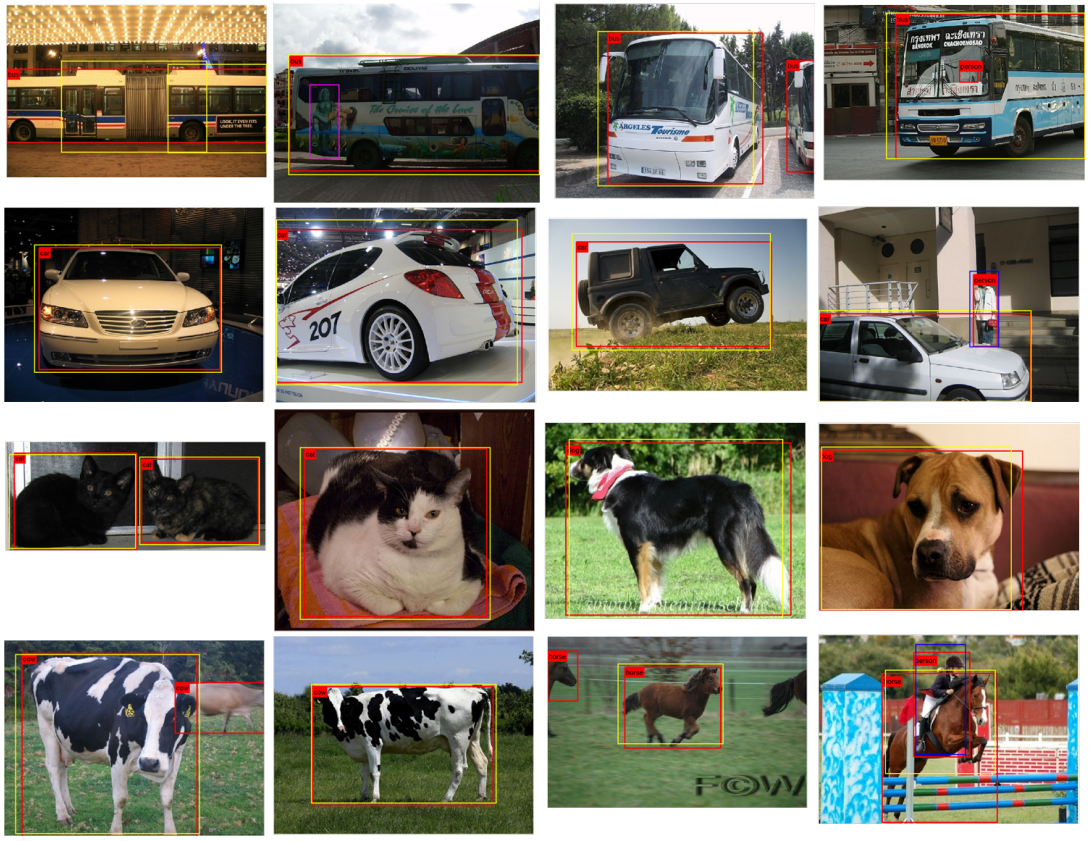

Figure 25: Detection results of bus, car, cat, dog, cow and horse classes. Red rectangle labels the ground truth and yellow rectangle labels detections of VARIS plus compound exemplar. If there are classes in the same image, the second class will be labeled in blue.

Figure 26: Detection results of sheep, sofa, tv(monitor), plant and train classes. Red rectangle labels the ground truth and yellow rectangle labels detections of VARIS plus compound exemplar. If there are classes in the same image, the second class will be labeled in blue.

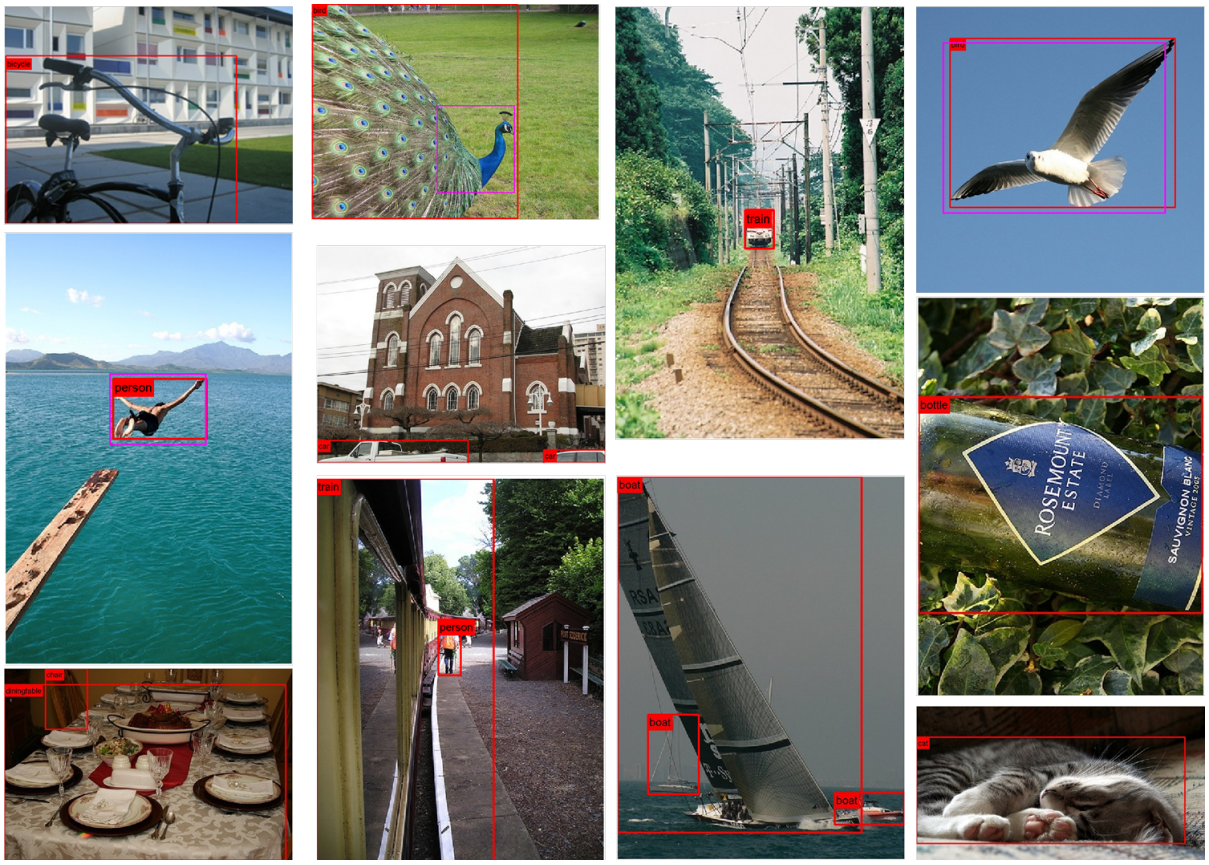

Figure 27: Cases that our framework fails on.

CHAPTER 7

IMPROVED INDEXING STAGE WITH RANDOM FOREST

In the previous work, we built a novel framework that addresses the multi-view multi-class object detection problem. The proposed VARIS system is based on maximizing the similarity between the input image and a compounded exemplar. The experimental results showed that the VARIS outperforms the current state-of-the-art object detection methods in terms of precision. However, the major issue of the VARIS framework is the high computational cost. The average time spent on processing a standard 640 by 480 image is about one minute.

After fully analyzing each component of VARIS, we found out the major reason of high computational cost is due to the processing on too many background regions. Since we apply the sliding window mechanism to scan the original image in different locations and scales, the total number of input to VARIS could be over one thousand. Among all the potential regions, maybe only a few of them are the true object locations and all the others are belong to background. Although VARIS achieves high precision on the real object locations, it seems that VARIS lacks of ability to distinguish the background regions in the early stage. More precisely, the indexing procedure usually does not reject background patches but treat them as neighbors with lower votes than the real object patches. The final decision is made after the compounding procedure and rejects regions with low overall similarity scores. If we can roughly remove some of the background regions out of scope early in the indexing stage, the system speed should get a significant boost. Therefore, we bring random forest into the VARIS framework and replace the old kd-tree module. Random forest is a machine learning method designed for classification

and regression purpose. Here we use it to determine the small patches belong to foreground or background.

Random forest framework is getting more attention recently in computer vision area, especially in image classification (16), segmentation (76), object detection and pose estimation (29; 38) and so on. It has demonstrated the ability to handle large amounts of training data efficiently while reducing the tendency of generating overfitting models. Although achieving high detection rate on rigid object classes such as face, car and etc., the performance of standard random forest based learning framework on non-rigid object classes is still poor. One explanation could be that the Gaussian-like models are applied to simulate the probability distribution of the object centers, which results with an average model that doesn't cover many intra-class variations. In deformable or articulated object classes, parts with large geometry variations are treated as rare cases and contribute only low voting. Another problem of the standard random forest is that the parametric information stored in the grown trees is hard to extend for the rare cases. In other words, the system doesn't have the ability for incremental learning.

In the following sections, we will go through the details of the modified random forest module and show how to embed it into our current VARIS system. We also demonstrate that the new module improves the precision as well as the speed.

7.1 Modified Random Forest Framework

Originally introduced in (18), random forest assembles a set of randomized binary trees to perform classification or regression function. Each tree is built from the root node with all training data, and then iteratively adds split nodes that optimally separate the data into two portions according to pre-defined rules. Nodes at the bottom of the tree are the leaf nodes that store the statistical information of all

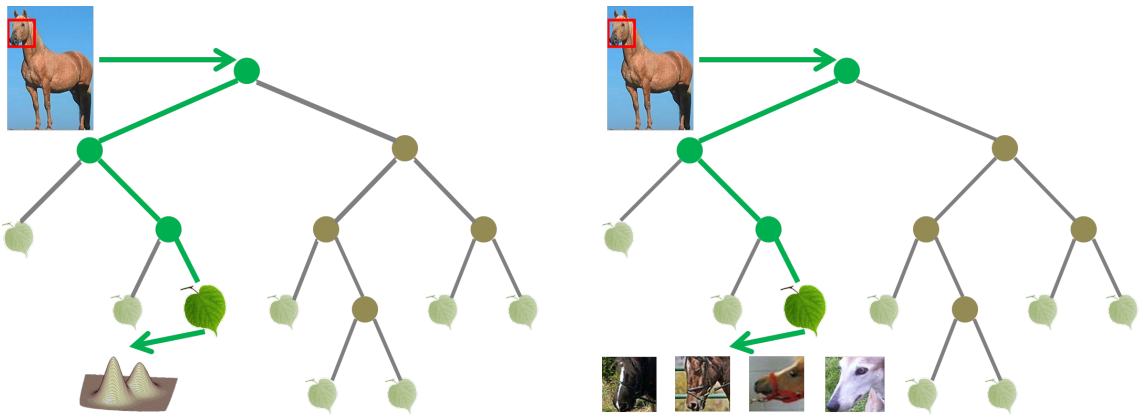

Figure 28: Different patch representations in the leaf node between the standard random forest and the proposed modified random forest. Left shows that the input patch travels through the tree and retrieve a continuous probability distribution of the relative object center. Right shows that, in our proposal, the same input patch reaches the leaf node and retrieves different exemplar patches that are visually similar. Note that this discrete patch representation enables the direct modification on the leaf node, which is the basis of the incremental learning.

the data that arrived. Depending on the task, the stored information can be either the class probability $p(C|L)$ or distributions of the estimated parameters, e.g. the object centers (Fig. 28 Left).

Our major modification of the general random forest framework is a non-parametrically representation of the positive patches reached at each leaf node. More precisely, in each leaf node, we store the original locations of the reached patches in the exemplar images and the ID number of the correspondent exemplars. At test stage, the input patch passes through the tree and reaches one of the leaf nodes. Then all the positive patches stored in this leaf will be considered as the k nearest neighbors to the input (Fig. 28 Right). By this way, we associate the input image with the exemplars in the local patch level. Different from other nearest neighbor approaches that try to learn distance metric functions, the random

forest is more efficient in searching for the optimal nearest neighbors due to the binary tree structure and discriminatively trained split nodes.

7.2 Training Procedure

Initially we prepare a large dataset for the training step, including images from the positive classes and the negative class (the background). For the positive classes, we extract local patches $P_i^+ = \{I_i, C_i, E_i, \boldsymbol{d_i}\}$ within the bounding box, where

I_i is the feature representation of the local patch,

C_i is the class label,

E_i is the exemplar label in class C_i,

$\boldsymbol{d_i}$ is a two dimensional vector representing the top-left

corner location of patch P_i.

For the negative class, on the other hand, we extract local patches densely all over the image and in different scales. The negative patches P_i^- have only feature representation attribute and all other fields are set to pseudo values. Training procedure for each tree $T_i \in T$ starts at the root node with a randomly selected set of local patches $p \subset P$. Then the tree T_i grows recursively in the way:

1. Randomly generate a set of split functions f_Φ for the current node, where each function is defined by the parameter $\{\phi \in \Phi | \phi = \{\boldsymbol{p}, \boldsymbol{q}, f, \tau\}\}$. The output of the split test is defined as:

$$f_\phi(Pi) = \begin{cases} 0 & \text{if } I^f(\boldsymbol{p}) - I^f(\boldsymbol{q}) < \tau \\ 1 & \text{otherwise} \end{cases} \tag{7.1}$$

2. Separate the patch set p_{node} into two subsets p_L and p_R for each split function $\phi \in \Phi$.

$$p_L(\phi) = \{p_i \in p_{node} | f_\phi(p_i) = 0\} \tag{7.2}$$

$$p_R(\phi) = \{p_i \in p_{node} | f_\phi(p_i) = 1\} \tag{7.3}$$

3. The split function that maximizes the gain function $g(x)$ (Eq. 4) is saved for the current node.

$$g(\phi, p_{node}) = \mathcal{H}(p_{node}) - \sum_{s \in \{L,R\}} \frac{|p_s(\phi)|}{|p_{node}|} \mathcal{H}(p_s(\phi)) \tag{7.4}$$

where $\mathcal{H}(\mathcal{P})$ is the entropy measurement (Eq. 5) of the set $\mathcal{P}$ that reflects the class uncertainty after the classification.

$$\mathcal{H}(\mathcal{P}) = -\sum_c p(c|\mathcal{P}) log(p(c|\mathcal{P})) \tag{7.5}$$

4. Repeat steps 1 to 3 for p_L and p_R until some stopping criteria are met; otherwise, create the leaf node and save all the patch information.

Two criteria are defined to regularize the tree growth. The first one is the max depth. Once the tree grows to the max depth, we create the leaf node with all the patches reaching to this level. The other criterion is the minimum number of patches in each leaf. If $|p_{node}|$ is smaller than a pre-defined number ϵ, a leaf node is created.

As mentioned earlier, our approach is different from the regular random forest in the way of organizing the leaf nodes. Instead of modeling the distribution of patch relative locations with Gaussian

functions, we specifically store each patch's location, the ID of its originated exemplar and the class label. All the patches are treated equally as the neighbors of the input patch that reached at this leaf during the testing procedure. The discrete representation of patch locations creates explicit mapping relationship between each input patch and exemplar patches, which can also be utilized for the meta-data transfer. Furthermore, the discrete representation is suitable for incremental learning. By adding patches to the leaf node or replacing useless patches with new ones, the system can be easily tuned to integrate with the new information. The only problem seems to be that only a small portion of the exemplar patches are saved at the leaf nodes due to the randomly selected training subset for each tree. However, our experiment demonstrates that the forest composed of more than 10 trees usually provides a good generalization.

7.3 Testing Procedure

In the testing scheme, image patches are densely extracted from the input image and represented by N-dimensional feature vectors. Each image patch travels through the trained forest, retrieves the nearest neighbors and casts weighted votes. After this, the input image is associated with object class exemplars in terms of the similarities in the local feature space. Since each input patch can vote for multiple exemplar patches and each exemplar patch can be voted multiple times by different input patches, it again creates a many-to-many mapping relationship between input patches and exemplar patches. Therefore we apply the same 2D sequencing and compounding procedures explained in the previous chapter to get the final detection results.

Compared to the kd-tree indexing scheme, our modified random forest framework is able to distinguish a patch belong to foreground class or background class. Due to such discriminative power, the

number of returned neighbors in the background region is significantly reduced. Therefore, we can skip the time-consuming sequencing procedure on those classified background regions, which improves the detection speed of the whole system.

7.4 Vote Weighting

Like many other codebook approaches, we assign different weights to the local exemplar patches. Compared to the voting schemes that distribute equal weights to local patches (24; 74), approaches with learnt weights usually demonstrate better performance. By assigning higher weights to the patches that have more salient information about the class, the margin between different classes is increased. However, instead of learning millions of weights for all local exemplar patches (37), our approach assigns the same weight to the patches in the same leaf as long as they belong to the same class. Our intuition is that the random forest framework keeps the discrimination power of local patches by splitting them into different leaf nodes, and the importance of each patch cluster can be simply represented by the leaf purity defined in Eq. 6,

$$w_{c,e} = \frac{|P^+|}{|P_{node}|} \cdot \frac{|P^+_{c,e}|}{|P^+_c|} \tag{7.6}$$

where $|P^+|$ is the total number of positive patches in the leaf node, $|P_{node}|$ is the total number of patches in the leaf node, $|P^+_{c,e}|$ is the number of different exemplars from class c and $|P^+_c|$ is the total number of patches from class c. In other word, the weight $w_{c,e}$ is

- highest when the patches are all from the same positive class but appear in different exemplars.
- lower when the patches are all from the same positive class and some patches are from the same exemplar.

- even lower when some of the patches are from the background images.
- lowest when all the patches are from the background images.

7.5 Experiment and Results

For evaluating the performance of the improved system, we apply the same PASCAL VOC 2007 dataset (28) as our training, validation and test sets. Since we do not consider the case of patch sharing among different object classes in this work, we simply train the system in the way of binary classification and repeat the same procedure for all the classes. We extract object exemplars according to the original labeling information and divide them into several different subgroups based on their aspect ratios. For the exemplars in the same subgroup, we crop and resize them to the same size with maximum width and height of 100 pixels, and then flip them in the horizontal direction to cope with view variations. The image patch is defined as a 16×16 pixel square and same 32-dimensional features defined in (38) are generated.

We initially train each decision tree of random forest with 100 randomly selected exemplars from each object class. Since our system is capable of doing incremental learning, the initial exemplars can be replaced by other exemplars that have better representation of the class. 400 patches are extracted from each exemplar as well as the background images. In total, we have $160,000$ ($400 \times 2 \times 200$) patches for a single tree with a maximum depth of 25. ϵ is set to 50 for each leaf node, which specifically defines the maximal number of nearest neighbors for each input patch. Overall, we trained 20 trees for each object class.

In the test stage, the input image is densely sampled where each patch retrieves the corresponding neighbors. Then the sliding window mechanism is applied to search for the potential object locations

TABLE III: Performance comparisons. The first 4 rows show the performance of our competitors. MRFGC in the fifth row is our basic proposal without incremental learning. MRFGC+IL shows the performance of our extended system trained with extra hard examples from the validation set.

	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbik	pers	plant	sheep	sofa	train	tv	
ESVM (57)	.208	.401	**.089**	.105	.102	.347	.406	.097	.106	.146	.024	.122	.371	.320	.184	.102	.152	.120	.295	.303	.2
DPM (31)	.287	.551	.006	.145	**.265**	.397	**.502**	.163	.165	.166	.245	.050	.452	**.383**	.362	.090	.174	.228	.341	.384	.268
HF (38)	.238	**.554**	.007	.102	.202	**.459**	.501	.14	.181	. 137	.143	.014	.229	.294	.265	**.12**	.171	**.282**	.324	.376	.225
VARIS (54)	.197	.425	.002	.131	.009	.279	.347	.088	.007	.126	.136	.048	.314	.337	.208	.071	.102	.153	.318	.284	.179
MRF-VARIS	.226	.473	.011	.097	.198	.405	.492	.142	.12	.156	.207	.039	.387	.241	.259	.103	.163	.199	.347	.309	.223
MRF-VARISEC	**.305**	.548	.052	**.151**	.223	.423	.501	**.187**	**.203**	**.261**	**.249**	**.182**	**.524**	.378	**.384**	.117	**.208**	.271	**.362**	**.432**	**.298**

with the 2D sequencing approach mentioned in chapter 4. To handle the scale variations, we also resize the input image with different ratios $\{\delta = \sqrt{2}^k | k = \{-4, -2, \ldots, 2, 4\}\}$ and apply the same approach. All the results are back-projected to the original image, where the hypothesis is drawn. During the experiment, we notice that the sequencing approach in pixel-by-pixel level achieves the best result, but it suffers a high computation cost due to the large numbers of mapping relationships. Therefore, we combine the mapping information of patches within a 8×8 pixel region (referred as a cell) both in the input image and exemplars, and then apply the sequencing approach at the cell level.

We also compare the results of improved VARIS system with previously mentioned SOA algorithms as well as standard VARIS. All the algorithms including ours are tested on the 20-category PASCAL VOC 2007 object detection challenge and the performance comparison is shown in Table III. The proposed VARIS system with random 100 exemplars for each tree of each category outperforms the ESVM and the VARIS system but is worse than the Hough forest and DPM algorithms in some object classes. Figure 29 shows several detection results. To improve the final results as well as demonstrate the ability of incremental learning, we conduct another experiment where the hard object instances in the validation

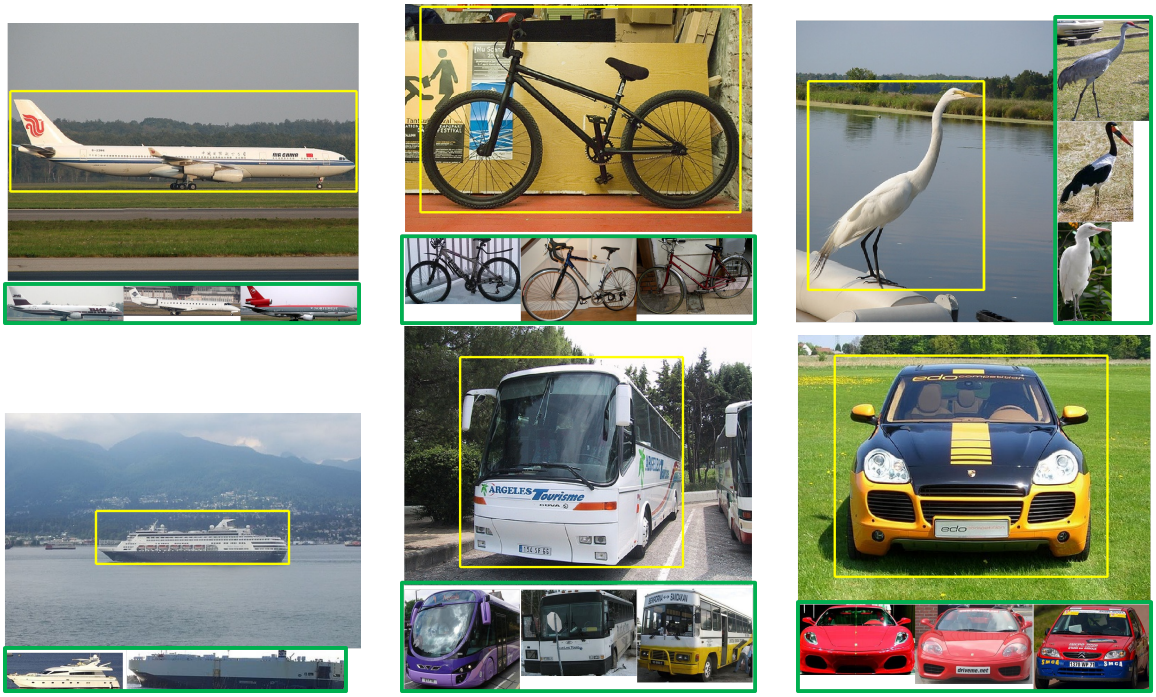

Figure 29: Detection results on different object classes. The detected objects are labeled by yellow rectangles and the smaller images are the corresponding exemplars that are visually similar to the detections.

set are added to the system. Since the images in the validation and test sets are different, the expanded system does not have the overfitting issue. The performance of the extended system is also shown in Table III, which achieves the best AP. Figure 30 shows several cases where the new information is helpful for the generalization in detection.

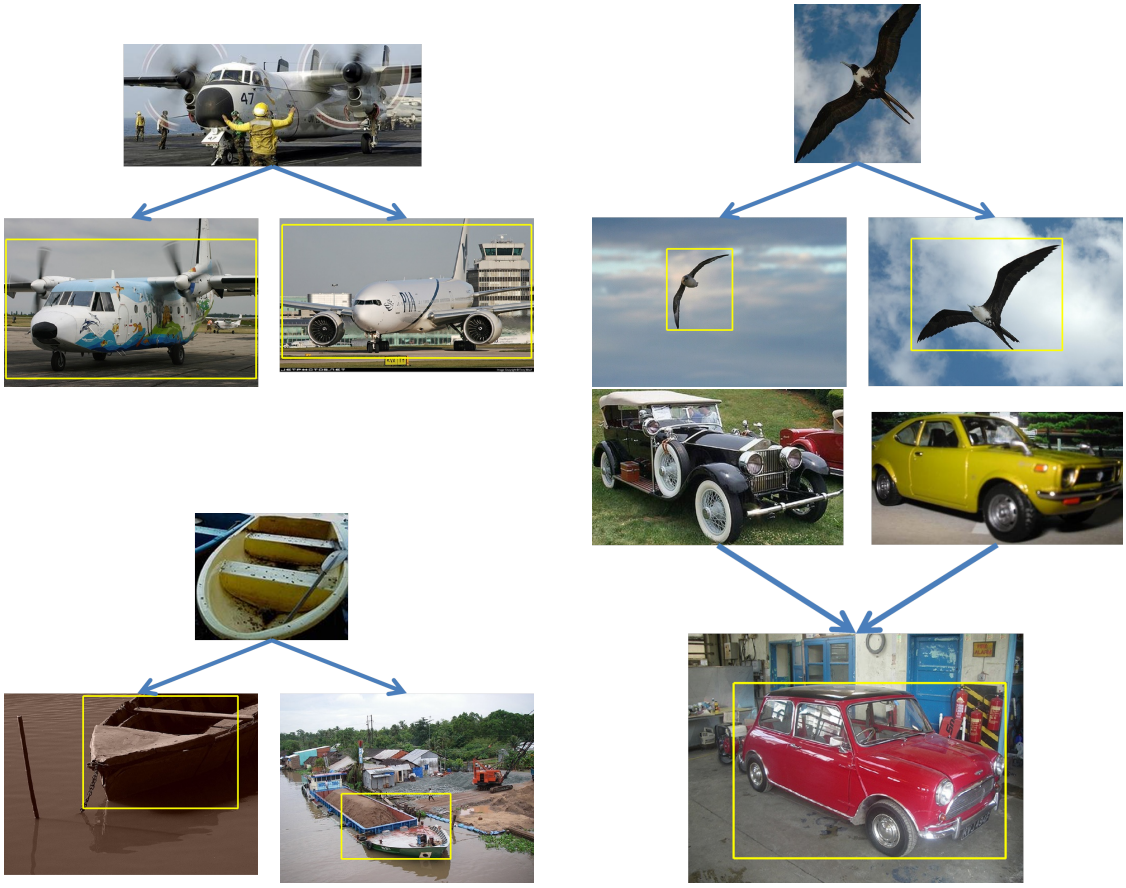

Figure 30: Additional detection results from the extended system. The top row of each sub-image shows the new exemplars from the validation set. The bottom rows show the new detection results which were missed by the basic version of our proposal. It is obvious that the new added information is helpful to enhance system performance.

CHAPTER 8

BEYOND 2D OBJECT DETECTION - 3D HUMAN POSE AND SHAPE ESTIMATION

Beyond 2D object detection problem, I recently explore the 3D human activity recognition problem with our VARIS system. Recognizing human activities brings many new applications such as human computer interaction, video surveillance safety, elder people nursing and so on. Recent real-time depth sensing system, e.g. Microsoft Kinect, (Fig. 31 Left) sparks a revolution in this area. It captures a depth-map stream (Fig. 31 Right) at 30 fps and estimates predefined body joints that constitute a wired human skeleton map of the user. With the extracted joints information, subsequent algorithm can be applied to recognize the human activity.

Although the depth camera in general offers better 3D information than those estimated from monocular sensors, simply using such 3D information to estimate joint positions for human action is not plausible. One reason is due to intrinsic and extrinsic properties of the Kinect depth sensor, the raw input depth images have some inevitable defects, such as noise and occlusion that can affect the accuracy of the joint estimation process. Moreover, the skeleton map does not include the shape information, which is also considered to be one of substantial properties of the human body. Therefore, we need a method that can model human pose and shape simultaneously with better accuracy before the activity recognition.

Among the work of human body modeling, SCAPE (4) has been widely applied in different applications. Instead of learning a complex function for many correlated pose and shape parameters, SCAPE

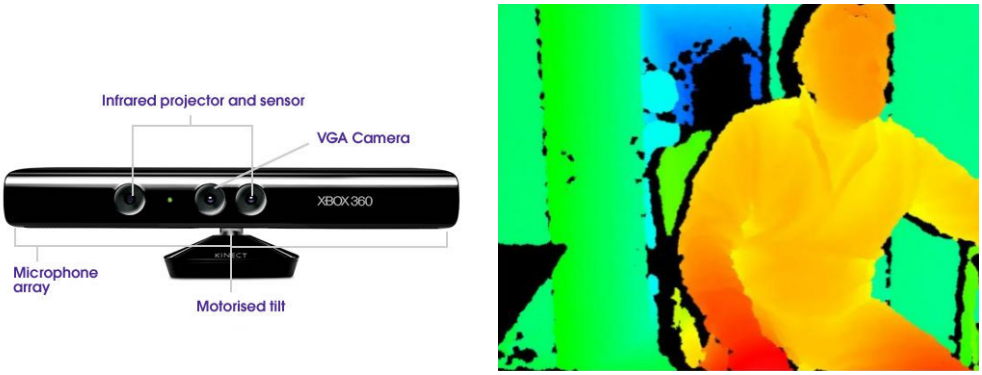

Figure 31: Left: Configuration of Microsoft Kinect device. Right: Depth image captured by Kinect. Different colors label different depth information.

decouples the problem into two separate optimization processes. It first learns a pose deformation model from one person with different poses, and then learns a shape deformation model from different person with one pose.

We implement the basic SCAPE as the baseline of our project. And then modify the optimization equations to fit our situation. To train the pose and shape deformation model separately as done in the original SCAPE work, we prepare two separate 3D mesh dataset with 200 mesh instances in each. The trained deformation models allow pose and shape deformations to be combined in one unified framework. We name our work as parametric deformable model (PDM).

8.1 Human Body Representation in 3D

In computer graphics, 3D objects are represented by three-dimensional surfaces X. There are two popular ways to discretize the surface, the points cloud and the polygon mesh. A point cloud is a

collection of sensor readings about the object surface, where each element in the cloud represents the 3D coordinate of a surface point (Fig 32). The point cloud is denoted as $P^X = (V^X)$. Point cloud is a common representation of the sensor data from the most recent 3D acquisition devices, such as Kinect. However, it discards the information about the continuous surface, especially the topology of the surface connectivity.

Another common representation of object surface is to use polygon mesh. A polygon mesh is a collection of vertices and edges that defines the shape of the object. It is denoted as $M^X = (V^X, P^X)$, where the $V^X = (x_1, \ldots, x_k)$ represents the vertices and $P^X = (p_1, \ldots, p_k)$ includes the vertex indices that composed of the current polygon. In our experiment, we always use triangles as the polygons in the mesh. Therefore, each triangle p_k has three vertices $(p_{1,k}, p_{2,k}, p_{3,k})$ and three edges $(v_{1,k}, v_{2,k}, v_{3,k})$.

Figure 32 shows examples of a human body surface, the point cloud representation of the same surface and the polygon mesh representation of the same surface.

8.2 Data Preparation

As previously mentioned, the training process of PDM requires a large number of 3D mesh data. Building such dataset with real human model is one plausible solution but also expensive and time-consuming. It requires a high-precision laser scanner that captures the person from different view angles. Then a registration algorithm needs to be applied to construct the full body model with those partial views. Last but not least, filling holes, removing noises and smoothing the surfaces for each 3D mesh model also require tremendous human efforts.

Compared to the real human model, synthetic human model from 3D commercial software is much simpler and faster to generate. In our case we use a 3D animation software called Poser (80). Poser

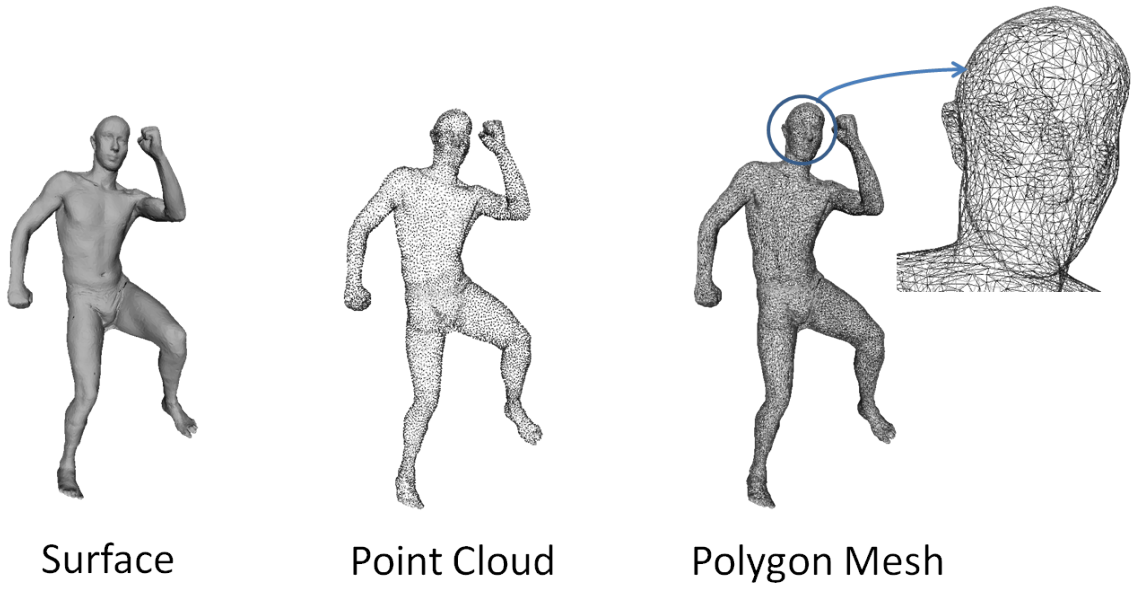

Figure 32: 3D representation of human body.

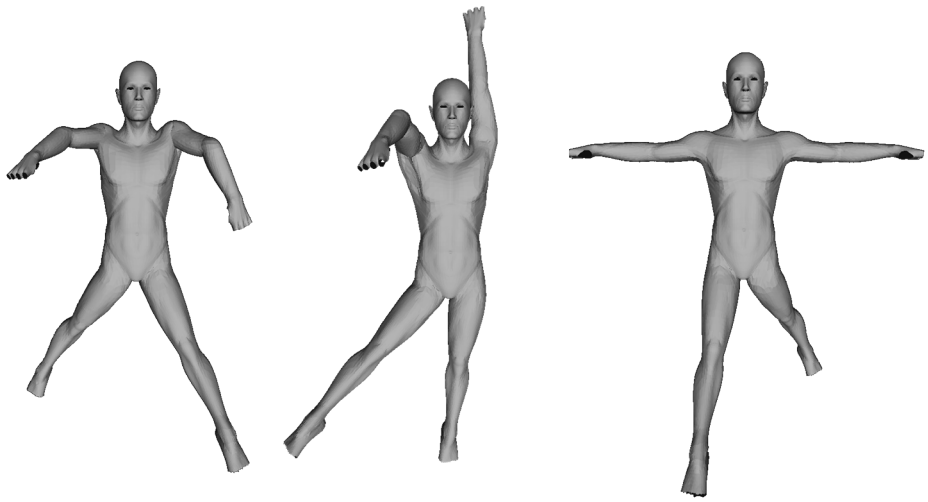

Figure 33: Mesh instances in different poses generated from Poser.

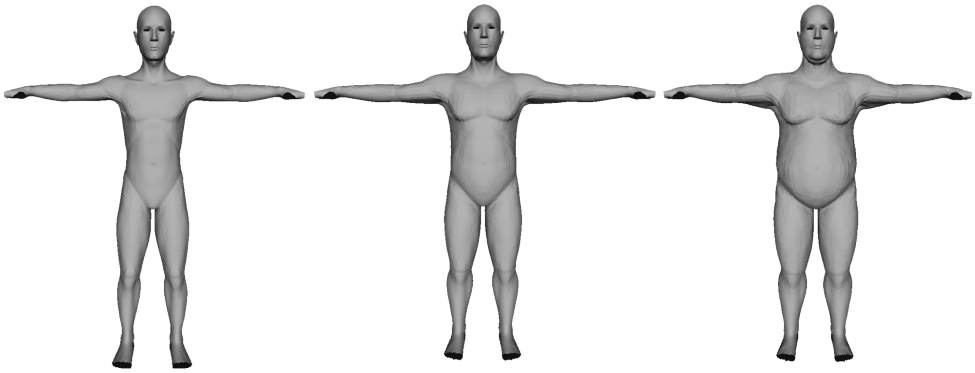

Figure 34: Mesh instances in different body shapes generated from Poser.

is a 3D rendering software that can be used for posing and animation of 3D polygon mesh of human. It comes with a library of pre-built human models with large freedom of joint and shape parameters. Together with programming interface of Python, it can automatically generate thousands of 3D human mesh instances with different poses and shapes in an hour (Figure 33 & Figure 34). Another advantage of using Poser is that all the mesh instances originated from the same mesh template are fully labeled and registered. For example, the number of vertices in each mesh instance is same as the template mesh; the vertex order keeps the same no matter what kind of pose the mesh forms. Therefore, Poser is a better solution for us to prepare the training dataset. Besides the advantages, however, there are two small issues of Poser we have to mention in this report. First, the data generation is fully unsupervised only with some initial user input, so it could generate some poses that are unrealistic to real person. So we need to examine all the generated meshes and manually remove the unrealistic ones. Second, Poser itself does not simulate the gravity effect on human body. So the person with standing pose has the same body shape as the one in lying down pose.

In the original SCAPE work, the pose deformation model is trained with different mesh instances of the same human subject in varies poses, while the shape deformation model is trained with mesh instances from many subjects in a neutral pose. Recent work (21) has pointed out that such decoupling of shape and pose deformation has a major problem. For example, if the human subject is male for the pose training data, then the pose and shape estimation for a given female subject will not be accurate. However, in our experiment, we have a strong prior that the gender of the test subject is known, therefore such decoupling will not be a problem. We train two separate pose deformation models (male and female) and apply only one of them during the test according to the gender prior.

Among all the training mesh data, we select a person with a regular body size and a neutral pose as the template mesh. We divide the full body into 17 different body parts. We also manually select 16 points on the template mesh as the joint landmarks. The template mesh (male) and the joints landmarks are shown in Figure 38.

8.3 Parametric Deformable Model

The PDM divides full human body deformation into two separate deformations, pose and shape deformation, where the pose deformation can be further divided into rigid and non-rigid deformations. As stated in the second chapter, the human body is composed of a set of polygon meshes. Therefore, the difference between template mesh model and input mesh can be represented by the deformations of each individual triangle mesh. We denote each triangle in the template mesh model as p_k and the two edges of the triangle as $\hat{v}_{k,j}, j = 2, 3$. We can write:

$$v^i_{k,j} = R^i_{l[k]} S^i_k Q^i_k \hat{v}_{k,j} \tag{8.1}$$

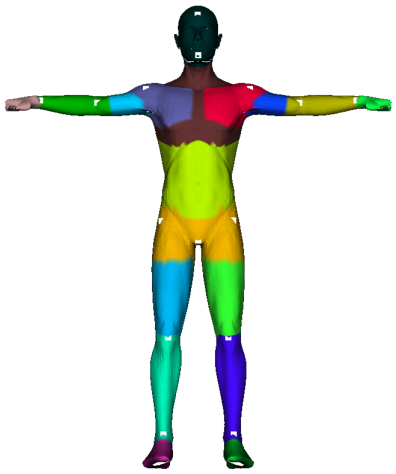

Figure 35: Template mesh model with 16 joint landmarks. Different body parts are shown in different colors. Joint landmarks are shown in white squares.

where $R_{idx[k]}$ is the rigid rotation matrix that is the same to all the triangles belong to the same body part. S_k^i is the shape deformation matrix and Q_k^i is the pose deformation matrix.

8.3.1 Pose Deformation

To learn the pose deformation model, we follow SCAPE that learn a regression function for each triangle p_k, which estimate the pose deformation matrix Q as a function of the twists of its two nearest joints $\triangle r_{l[k]}^i$,

$$q_{k,l[m]}^i = a_{k,l[m]}^T \cdot \begin{bmatrix} \triangle r_{l[k]}^i \\ 1 \end{bmatrix} \tag{8.2}$$

In the above equation, $\triangle r$ can be calculated from the rigid rotation matrix R. However, the non-rigid deformation matrix Q for each triangle is unknown. We estimate those matrices by solving an

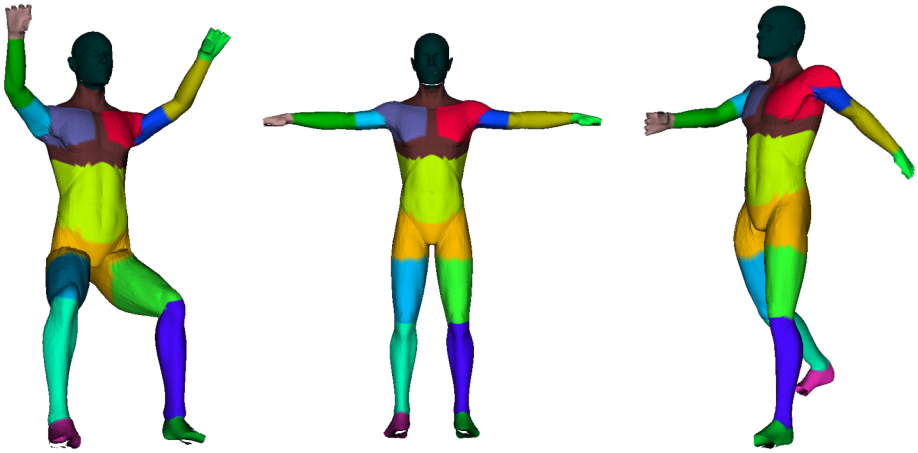

Figure 36: PDM pose deformation. Left figure shows a driving pose. Middle figure shows the template pose. Right figure shows a walking pose.

optimization problem that minimizes the distance between the deformed template mesh and the training mesh data with a smoothness constraint,

$$\underset{\{Q_1^i \cdots Q_P^i\}}{\operatorname{argmin}} \sum_k \sum_{j=2,3} ||R_k^i Q_k^i \hat{v}_{k,j} - v_{k,j}^i||^2 + w_s \sum_{k_1,k_2 adj} I(l_{k_1} = l_{k_2}) \cdot ||Q_{k_1}^i - Q_{k_2}^i||^2 \tag{8.3}$$

where the regularization part intend to have similar deformations in adjacent triangles that belong to the same rigid body part.

We trained the PDM pose deformation model with 200 mesh instances. After the training, we are able to manipulate the template mesh model to form different body poses by initialing the rigid rotation matrix R with different values. Figure 36 shows results of such pose deformation.

8.3.2 Shape Deformation

To learn the shape deformation model, we again follow the original SCAPE work that implements PCA to represent shape deformation matrices as a linear combination of a few eigen-spaces,

$$S^i = F_{U,\mu}(\beta^i) = \overline{U\beta^i + \mu} \tag{8.4}$$

Similar to the pose estimation, the shape deformation matrix S for each triangle is unknown. Therefore, we use the same technique to estimate the matrix S,

$$\underset{S^i}{\operatorname{argmin}} \sum_k \sum_{j=2,3} ||R_k^i S_k^i Q_k^i \hat{v}_{k,j} - v_{k,j}^i||^2 + w_s \sum_{k_1,k_2 adj} ||S_{k_1}^i - S_{k_2}^i||^2 \tag{8.5}$$

where the regularization part chooses similar shape deformations in adjacent triangles.

We trained the PDM shape deformation model with 200 mesh instances in different body shapes, from tall to short, from underweight to overweight, from strong to slim and so on. Once the model learns the PCA parameters, we can also manipulate the template mesh to form different body shapes by giving a initial vector of β. Figure 37 shows results of such shape deformation.

8.4 Partial Data Completion with PDM

In the previous two sections, we have discussed the details about how to learn a pose deformation model as well as a shape deformation model. A direct application of the trained PDM is to the task of partial data completion. By given a Kinect point cloud data and a few joint landmarks, the PDM

Figure 37: PDM shape deformation along the direction of first principal component. Left figure shows an extreme case of an underweight person. Middle figure shows the template mesh. Right figure shows an extreme case of an overweight person.

should generate a realistic full 3D mesh output that is consistent with the partial data by minimizing the objective function:

$$\underset{\{Q_1^i \cdots Q_P^i\}}{\operatorname{argmin}} \sum_k \sum_{j=2,3} ||R_k F_{U,\mu}(\beta)\Gamma_{a_k}(\triangle r_{l[k]})\hat{v}_{k,j} - (y_{j,k} - y_{1,k})||^2 + w_Z \sum_{l=1}^{L} w_l ||y_l - z_l||^2 \qquad (8.6)$$

The first part of the function defines the mesh output to be consistent with the learned PDM model and the second part regulates the optimization to find the set that best fits to the input point cloud. w_Z is a weighting term that balances the importance of the two parts.

One of the major differences between PDM and SCAPE is that we add another weight term w_l in (Equation 8.6). In the original work of SCAPE, the input data is from a laser scan, so each point can be equally weighted. In our work, however, the input is from Kinect depth sensor, where the data accuracy

is affected by actual distance between the sensor and each sensored point. So we create a noise model to simulate such effects, which generates different value of w_l for each registered point. More details will be discussed in the next chapter.

(Equation 8.6) has three parameter sets (R, Y and β) to be optimized simultaneously. It forms a standard non-linear and non-convex optimization problem. To avoid the possibility of converging to a sub-optimal solution, we follows SCAPE that uses an iterative process to optimize the three parameters. We treat the three sets of parameters separately, optimizing only one of them each time while keeping the other two fixed:

- Optimize R with S and Y fixed, then update $\triangle R$ and Q accordingly.
- Optimize Y with R and S fixed.
- Optimize S with R, Q and Y fixed.

In (Equation 8.6) the second part requires finding a correspondence between the input point cloud and landmarks on 3D mesh model. In our experiment, we first estimate an initial R based on the given joint landmarks. Then we proceed the pre-mentioned optimization several times to get an estimated mesh model M_{curr}, where only the joints landmarks are used to find the correspondence. Next we run a registration algorithm based on Iterative Closest Point (ICP)(20)(14) and get the full registration between the input point cloud and the current 3D mesh model M_{curr}. We also remove the correspondences between pairs that have larger distance than a pre-defined threshold. The left correspondences are used to estimate the new R. This optimization-registration step runs iteratively until converging. Figure 38 shows the full procedure of partial data completion with PDM algorithm and the result.

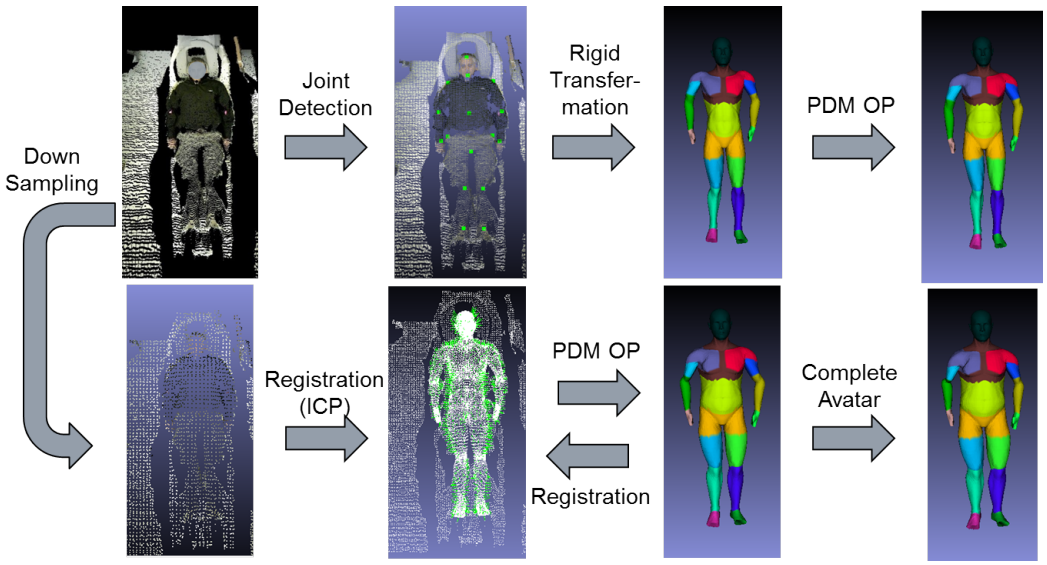

Figure 38: Partial view completion flow chart and result.

CHAPTER 9

CONCLUSION AND FUTURE WORK

Our research work, VARIS system with exemplar compounding, brings together new algorithms and insight to construct a framework for robust multi-view multi-class object detection. It was initially motivated by the requirement of multi-view face detection, and later extended to general object detection. The framework is based on maximizing the aggregate similarity scores between the input image and an optimally assembled exemplar. The framework inherits the advantages of part-based framework and exemplar-based framework, which enhance the system tolerance to pose variation and occlusion simultaneously. Meanwhile, different from other binary classifier-based approach, our data-driven-based framework returns more information about the detected object that is similar to the corresponding exemplar, such as pose, scale, similarity and so on. The experiments show that our work outperforms current state-of-the-art approaches in both face detection and general object detection domains. Moreover, with the new implementation of the modified random forest procedure, the detection accuracy and speed are further improved.

Beside the 2D object detection tasks, I also explore the 3D pose and shape estimation of human body during my Ph.D. study. Initially we want to extend our VARIS system to human activity recognition with 3D depth data. However, we found out the joint estimation process offered by default Kinect program does not work well, especially when the target human object stay close to the wall or lying on the bed. To improve the joint estimation performance in such cases, I develop an algorithm called parameterized deformable model (PDM) that estimate human body pose and shape simultaneously. In

return, it gives high accurate human pose information as well as the shape information.

In the future, I will combine and extend the two pre-mentioned algorithms, VARIS and PDM, to achieve the activity recognition purpose. As stated in (87), ”activities are spatio-temporal patterns. There are two important issues in activity recognition: the presentation of suitable features and the modeling of dynamical patterns.” Since the 3D joint position is helpful for activity recognition, I will first implement PDM on the depth image to estimate the 3D joint locations. Then I will apply 1D VARIS on the spatio-temporal joint feature to recognize the human activity. To demonstrate the efficiency of the proposed system, I will also compare the results with several standard approaches used in activity recognition, such as dynamic temporal warping(DTW)(64), Hidden Markov Model(HMM)(52) and Recurrent Neural Network(RNN)(58).

CITED LITERATURE

1. Agarwal, S. and Roth, D.: Learning a sparse representation for object detection. In European Conference on Computer Vision (ECCV), pages 113–130, 2002.

2. Amini, A., Weymouth, T., and Jain, R.: Using dynamic programming for solving variational problems in vision. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1990.

3. Amit, Y. and Kong, A.: Graphical templates for model registration. IEEE Transactions on Pattern Analysis and Machine Intelligence, 18(3):225–236, March 1996.

4. Anguelov, D., Srinivasan, P., Koller, D., Thrun, S., Rodgers, J., and Davis, J.: SCAPE: shape completion and animation of people. ACM Trans. Graph, 24:408–416, 2005.

5. Baker, H. H. and Binford, T. O.: Depth from edge and intensity based stereo. In Proceedings of the 7th international joint conference on Artificial intelligence - Volume 2, IJCAI'81, pages 631–636, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.

6. Basri, R., Costa, L., Geiger, D., and Jacobs, D.: Determining the similarity of deformable shapes. Vision Research, 38:135–143, 1995.

7. Bay, H., Tuytelaars, T., and Gool, L. V.: Surf: Speeded up robust features. In European Conference on Computer Vision (ECCV), pages 404–417, 2006.

8. Beis, J. S. and Lowe, D. G.: Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1997.

9. Bellman, R.: Dynamic Programming. 1957.

10. Ben-Arie, J.: Method for recognition of multidimensional patterns. In US Patent Application 13/573,231, 2012.

11. Ben-Arie, J., Pandit, P., and Rajaram, S.: View-based human activity recognition by indexing and sequencing. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2001.

CITED LITERATURE (Continued)

12. Bentley, J. L.: Multidimensional binary search trees used for associative searching. Communications of the ACM, 1975.

13. Bergtholdt, M., Kappes, J., Schmidt, S., and Schnörr, C.: A Study of Parts-Based Object Class Detection Using Complete Graphs. International Journal of Computer Vision, 87(1):93–117–117, March 2010.

14. Besl, P. and McKay, N. D.: A method for registration of 3-d shapes. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 14(2):239–256, 1992.

15. Bishop, C. M.: Neural Networks for Pattern Recognition. Oxford University Press, USA, 1 edition, January 1996.

16. Bosch, A., Zisserman, A., and Muoz, X.: Image classification using random forests and ferns. In ICCV, 2007.

17. Bourdev, L. D. and Malik, J.: Poselets: Body part detectors trained using 3d human pose annotations. In International Conference on Computer Vision (ICCV), 2009.

18. Breiman, L.: Random forests. Mach. Learn., 45(1):5–32, October 2001.

19. Cai, H., Yan, F., and Mikolajczyk, K.: Learning weights for codebook in image classification and retrieval. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010.

20. Chen, Y. and Medioni, G.: Object modeling by registration of multiple range images. In Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on, pages 2724–2729 vol.3, 1991.

21. Chen, Y., Liu, Z., and Zhang, Z.: Tensor-based human body modeling. In Computer Vision and Pattern Recognition, 2013 IEEE International Conference on, 2013.

22. Chum, O. and Zisserman, A.: An exemplar model for learning object classes. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2007.

23. Coughlan, J., Yuille, A., English, C., and Snow, D.: Efficient deformable template detection and localization without user initialization. 2000.

24. Csurka, G., Dance, C. R., Fan, L., Willamowski, J., and Bray, C.: Visual categorization with bags of keypoints. In ECCV, 2004.

CITED LITERATURE (Continued)

25. Dalal, N. and Triggs, B.: Histograms of oriented gradients for human detection. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2005.

26. Dempster, A. P., Laird, N. M., and Rubin, D. B.: Maximum likelihood from incomplete data via the em algorithm. JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B, 39(1):1–38, 1977.

27. Dorkó, G. and Schmid, C.: Selection of scale-invariant parts for object class recognition. In International Conference on Computer Vision (ICCV), pages 634–, Washington, DC, USA, 2003. IEEE Computer Society.

28. Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A.: The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results.

29. Fanelli, G., Gall, J., and Gool, L. V.: Real time head pose estimation with random regression forests. In CVPR, 2011.

30. Felzenszwalb, P. F.: Representation and detection of deformable shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(2):208–220, February 2005.

31. Felzenszwalb, P. F., Girshick, R. B., McAllester, D. A., and Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2010.

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

33. Felzenszwalb, P. and Zabih, R.: Dynamic programming and graph algorithms in computer vision. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 33(4):721–740, April 2011.

34. Fergus, R., Perona, P., and Zisserman, A.: Object class recognition by unsupervised scale-invariant learning. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 264–271, 2003.

35. Franzini, S. and Ben-Arie, J.: Speech Recognition by Indexing and Sequencing. In Proceedings of the International Conference of Soft Computing and Pattern Recognition, 2010.

36. Franzini, S. and Ben-Arie, J.: Speech Recognition by Indexing and Sequencing. International Journal of Computer Information Systems and Industrial Management Applications, 2012.

CITED LITERATURE (Continued)

37. Frome, A., Sha, F., Singer, Y., and Malik, J.: Learning globally-consistent local distance functions for shape-based image retrieval and classification. In International Conference on Computer Vision (ICCV), 2007.

38. Gall, J. and Lempitsky, V.: Class-specific hough forests for object detection. In CVPR, 2009.

39. Gavrila, D. M. and Philomin, V.: Real-time object detection for smart vehicles. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1(c):87–93 vol.1, 1999.

40. Geiger, D., Gupta, A., Costa, L. A., and Vlontzos, J.: Dynamic Programming for Detecting, Tracking and Matching Deformable Contours. IEEE Transactions on Pattern Analysis and Machine Intelligence, 17(3):294–302, 1995.

41. Harris, C. and Stephens, M.: A Combined Corner and Edge Detection. In Proceedings of The Fourth Alvey Vision Conference, pages 147–151, 1988.

42. Jain, V. and Learned-Miller, E.: Fddb: A benchmark for face detection in unconstrained settings. Technical Report UM-CS-2010-009, University of Massachusetts, Amherst, 2010.

43. Jolliffe, T. I.: Principle Component Analysis. Springer, 2002.

44. Jones, M. J. and Viola, P.: Fast multi-view face detection. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2003.

45. Ke, Y. and Sukthankar, R.: Pca-sift: a more distinctive representation for local image descriptors. In Proceedings of the 2004 IEEE computer society conference on Computer vision and pattern recognition, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 506–513, Washington, DC, USA, 2004. IEEE Computer Society.

46. Kumar, N., Zhang, L., and Nayar, S. K.: What is a good nearest neighbors algorithm for finding similar patches in images? In European Conference on Computer Vision (ECCV), 2008.

47. Langley, P., Iba, W., and Thompson, K.: An analysis of bayesian classifiers. In IN PROCEEDINGS OF THE TENTH NATIONAL CONFERENCE ON ARTI CIAL INTELLIGENCE, pages 223–228. MIT Press, 1992.

48. Leibe, B., Seemann, E., and Schiele, B.: Pedestrian detection in crowded scenes. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01, IEEE Conference on Computer

CITED LITERATURE (Continued)

Vision and Pattern Recognition (CVPR), pages 878–885, Washington, DC, USA, 2005. IEEE Computer Society.

49. Lindeberg, T.: Feature detection with automatic scale selection. International Journal of Computer Vision, 30:79–116, 1998.

50. Lowe, D. G.: Local feature view clustering for 3d object recognition. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 682–688. Springer, 2001.

51. Lowe, D. G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 60(2):91–110, November 2004.

52. Lv, F. and Nevatia, R.: Recognition and Segmentation of 3-D Human Action Using HMM and Multi-class AdaBoost. In ECCV'06: Proceedings of the 9th European Conference on Computer Vision, 2006.

53. Ma, K. and Ben-Arie, J.: Multi-view multi-class object detection via exemplar compounding. In International Conferrence on Pattern Recognition (ICPR), 2012.

54. Ma, K. and Ben-Arie, J.: Vector array based multi-view face detection with compounded exemplars. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012.

55. Maji, S. and Berg, A.: Max-margin additive classifiers for detection. In International Conference on Computer Vision (ICCV), 2009.

56. Malisiewicz, T. and Efros, A. A.: Recognition by association via learning per-exemplar distances. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2008.

57. Malisiewicz, T., Gupta, A., and Efros, A. A.: Ensemble of exemplar-svms for object detection and beyond. In International Conference on Computer Vision (ICCV), 2011.

58. Martens, J. and Sutskever, I.: Learning Recurrent Neural Networks with Hessian-Free Optimization. In International Conference on Machine Learning (ICML), 2011.

59. Mikolajczyk, K., Leibe, B., and Schiele, B.: Local features for object class recognition. In International Conference on Computer Vision (ICCV), pages 1792–1799, 2005.

60. Mikolajczyk, K. and Schmid, C.: Scale and affine invariant interest point detectors. International Journal of Computer Vision, 60(1):63–86, 2004.

CITED LITERATURE (Continued)

61. Mohan, A., Papageorgiou, C., and Poggio, T.: Example-based object detection in images by components. IEEE Transactions on Pattern Analysis and Machine Intelligence, 23:349–361, 2001.

62. Montanari, U.: On the optimal detection of curves in noisy pictures. Communications of the ACM, 14(5):335–345, May 1971.

63. Moosmann, F., Nowak, E., and Jurie, F.: Randomized clustering forests for image classification. IEEE Transactions on Pattern Analysis and Machine Intelligence, 30(9):1632–1646, September 2008.

64. Müller, M. and Röder, T.: Motion templates for automatic classification and retrieval of motion capture data. In Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation, 2006.

65. Ohta, Y. and Kanade, T.: Stereo by Intra- and Inter-Scanline Search Using Dynamic Programming. IEEE Transactions on Pattern Analysis and Machine Intelligence, 7:139–154, 1985.

66. Opelt, A., Fussenegger, M., Pinz, A., and Auer, P.: Weak hypotheses and boosting for generic object detection and recognition. In European Conference on Computer Vision (ECCV), pages 71–84, 2004.

67. Ott, P. and Everingham, M.: Shared parts for deformable part-based models. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011.

68. Papageorgiou, C. and Poggio, T.: A trainable system for object detection. International Journal of Computer Vision, 38(1):15–33, June 2000.

69. Phillips, P. J., Wechsler, H., Huang, J., and Rauss, P.: The FERET database and evaluation procedure for face recognition algorithms. Image and Vision Computing, 1998.

70. Rowley, H., Baluja, S., and Kanade, T.: Neural network-based face detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(1):23 –38, jan 1998.

71. Schapire, R. E.: The boosting approach to machine learning: An overview.

72. Schiele, B. and Crowley, J. L.: Object recognition using multidimensional receptive field histograms and its robustness to view point changes. In European Conference on Computer Vision (ECCV), 1996.

CITED LITERATURE (Continued)

73. Schmid, C.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2169–2178, 2006.

74. Schmid, C.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In CVPR, 2006.

75. Schmid, C. and Mohr, R.: Local grayvalue invariants for image retrieval. IEEE Transactions on Pattern Analysis and Machine Intelligence, 19:530–535, 1997.

76. Schroff, F., Criminisi, A., and Zisserman, A.: Object class segmentation using random forests. In BMVC, 2008.

77. Shashua, A. and Ullman, S.: Structural Saliency: The Detection of Globally Salient Structures Using a Locally Connected Network. In International Conference on Computer Vision (ICCV), number AIM-1061, 1988.

78. Silpa-Anan, C. and Hartley, R.: Optimised kd-trees for fast image descriptor matching. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2008.

79. Sirovich, L. and Kirby, M.: Low-dimensional procedure for the characterization of human faces. Journal of the Optical Society of America, 4(3):519–524, Mar 1987.

80. Software, S.: Poser pro 2013.

81. Subburaman, V. B. and Marcel, S.: Fast bounding box estimation based face detection. In ECCV, Workshop on Face Detection: Where we are, and what next?, 2010.

82. Thomas, A., Ferrari, V., Leibe, B., Tuytelaars, T., Schiele, B., and Gool, L. J. V.: Towards multi-view object class detection. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2006.

83. Turk, M. and Pentland, A.: Face recognition using eigenfaces. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 586 –591, Jun 1991.

84. Tuytelaars, T.: Vector quantizing feature space with a regular lattice. In International Conference on Computer Vision (ICCV), 2007.

85. Vapnik, V. N.: The nature of statistical learning theory. New York, NY, USA, Springer-Verlag New York, Inc., 1995.

86. Viola, P. and Jones, M.: Rapid object detection using a boosted cascade of simple features. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 511–518, 2001.

87. Wang, J., Liu, Z., Wu, Y., and Yuan, J.: Mining actionlet ensemble for action recognition with depth cameras. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012.

88. Weber, M., Welling, M., and Perona, P.: Unsupervised learning of models for recognition. In European Conference on Computer Vision (ECCV), pages 18–32, 2000.

VITA

NAME: Kai Ma

EDUCATION:

B. E., Electrical Engineering, Shandong University of Technology, China, 2005

M.S., Electrical and Computer Engineering, Kansan State University, Manhattan, Kansas, 2007

Ph.D., Electrical and Computer Engineering, University of Illinois at Chicago, Chicago, Illinois, 2014

EXPERIENCE:

Research Scientist, Siemens Corporate Research, Princeton, New Jersey, 2013 – Present

Computer Vision Intern, Siemens Corporate Research, Princeton, New Jersey, May 2013 - Sep 2013

Computer Vision Intern, Immersive Labs, New York City, New York, May 2012 - Apr 2013

PROFESSIONAL MEMBERSHIP:

Member, Institute of Electrical and Electronics Engineers (IEEE)
Member, International Association for Pattern Recognition (IAPR)

PUBLICATIONS:

Kai Ma, Jezekiel Ben-Arie. Multi-view Multi-class Object Detection via Exemplar Compounding. International Conference on Pattern Recognition (ICPR), 2012 (Oral)

Kai Ma, Jezekiel Ben-Arie. Vector Array based Multi-view Face Detection with Compounded Exemplars. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012. (Poster)

Barbara Di Eugenio, Milos Zefran, Jezekiel Ben-Arie, Mark Foreman, Lin Chen, Simone Franzini, Shankaranand Jagadeesan, Maria Javaild and **Kai Ma**. Towards Effective Communication with Robotic Assistants for the Elderly: Integrating Speech, Vision and Haptics. Dialog with Robots, AAAI 2010 Fall Symposium, Arlington, VA, USA, 2010