1-1-2009

Content-independent orientation detection with histogram of optimized local binary pattern

Nan Dong
Ryerson University

CONTENT-INDEPENDENT ORIENTATION DETECTION WITH HISTOGRAM OF OPTIMIZED LOCAL BINARY PATTERN

by

NAN DONG
B.Sc., University of Science and Technology of China,
China, 2006

A thesis
presented to Ryerson University
in partial fulfillment of the
requirement for the degree of
Master of Applied Science
in the Program of
Electrical and Computer Engineering.

Toronto, Ontario, Canada, 2009

Author's Declaration

I hereby declare that I am the sole author of this thesis.
I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

(Nan Dong)

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

(Nan Dong)

Borrower's Page

Ryerson University requires the signatures of all persons using or photocopying this thesis. Please sign below, and give address and date.

Name	Signature	Address	Date

ABSTRACT

Content-Independent Orientation Detection with
Histogram of Optimized Local Binary Pattern

©Nan Dong, 2009

Master of Applied Science
Electrical and Computer Engineering
Ryerson University

This thesis is primarily concerned with the introduction of a new approach to the general problem of automatic image orientation detection. Inspired by the local binary pattern(LBP), a luminance, rotation and scale invariant and content-independent algorithm is proposed, namely: *Histogram of Optimized Local Binary Pattern*(HOOPLBP).

Whilst the proposed approach is essentially generic, the core application considered in this study is human face orientation detection. To detect the face orientation, a general face model is trained using the HOOPLBP feature. The experiments show a very impressive result. Integrating this result with other face related techniques will facilitate some applications. To this end, this thesis propose a hybrid face detection system.

Specifically, the new system aims to detect both upright and tilted human faces in digital images. In the scheme, several face related algorithms are integrated to achieve different tasks in different stages. In addition, two modified systems are used in this thesis to detect faces in both grayscale images and color images.

The HOOPLBP is a new and robust method in automatic image orientation detection. It can be improved by other techniques and also can be used in many other fields. The future work is also included in the thesis.

Acknowledgment

I would like to acknowledge my heartfelt gratitude to my supervisor Dr. Ling Guan for his enlightening suggestions, vital encouragement and continuous support. It is his wisdom, patience and kindness that support and guide me to finish this work.

I would also like to thank Dr. Lei Zhang at Microsoft Research Asia. It is the conversation with him during his visit to Ryerson Multimedia Research Laboratory(RML) that direct and guide me to the work of this thesis.

A lot of thanks have to be given to all of the RML members. Their altruistic assistance and instruction help me a lot in my research. And their friendliness and humor make it an enjoyable and memorable time in this two-year study.

Much gratitude goes to my cousin's family, Uncle Xing's family, my girlfriend and my family. Their concerning and support make my life in toronto a decent time. In the end, I would like to thank my parents - my favourite father and mother. The achievement of this work and thesis would be impossible without their love, tolerance, encouragement and support.

I acknowledge the financial support provided by Natural Sciences and Engineering Research Council of Canada (NSERC) and Canada Research Chair Award. Without their support, this thesis would never have been possible.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Automatic Image Orientation Detection

THE rapid growth of digital imaging has led to an increase in image-related tasks such as enhancement, manipulation, compression, understanding, organization, and retrieval. Knowledge of the correct image orientation can be of great importance for these tasks. Automatic image orientation can drastically reduce the human effort otherwise needed to orient the images for viewing (either on a computer monitor, a handheld device, or a TV) or for organizing an album. In addition, many automatic algorithms for object recognition, scene classification, and content-based image retrieval either require a priori knowledge of the correct image orientation, or can perform significantly better if image orientation is known. For example, face detection algorithms [16] usually assume the image is in an upright orientation. Most sky detection algorithms are designed to take advantage of the fact that sky often appears as a blue region at the top of an image, with the exception of the clear blue sky detection method by Luo and Etz [1]. Semantic features are becoming increasingly important for content-based image retrieval and annotation [2, 13, 3]. For classification of images into indoor-outdoor [4], sunset, beach, field, fall foliage, mountain, and urban scenes [5], images are assumed to be in the upright orientation so that scene layout of prototypical scenes can be learned through training.

Automatic image orientation detection is a very difficult problem. Humans use object recognition and contextual information to identify the correct orientation of an image. Unfortunately, the state-of-the-art computer vision techniques still cannot infer the high-level knowledge abstraction

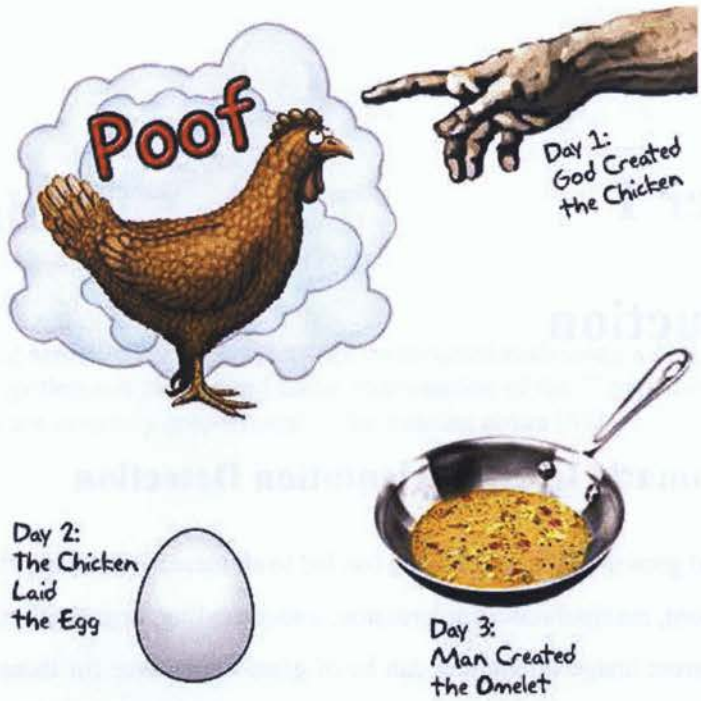


Figure 1.1: The dilemma: "chicken or the egg".

of the objects in the real world [24] . The alternative method is to exploit the low-level features (e.g., spatial color distributions, texture, etc.) from the images for orientation detection [25]. The problem between orientation detection and object recognition is like the dilemma - "Chicken or the egg", as shown in Fig. 1.1. On one hand, we need the correct orientation before pattern recognition. On the other hand, we also need the context information of the object for recognizing correct orientation. Fig. 1.2 illustrates the difficulty in image orientation estimation, where the true orientation cannot be detected unless you first recognize the object present in the image. Close-up images, low-contrast images, or images of uniform or homogeneous texture (e.g., sunset/sunrise and indoor images) pose additional problems for robust orientation estimation.

Figure 1.2: A difficult orientation detection problem [6]

1.2 Contributions

This thesis brings together new algorithms and insights to propose a content-independant orientation descriptor - Histogram of Optimized Local Binary Pattern(HOOPLBP). This method is demonstrated on and in part motivated by, the task of automatic digital image orientation detection. This task is basic but difficult in computer vision. There are few paper published in this field. Furthermore, a hybrid face detection system based on the HOOPLBP is proposed in this thesis. It can detect tilted faces in both grayscale and color images and the results outperform the previous work.

There are four main contributions in this thesis. We will introduce each of these ideas briefly below and then describe them in detail in subsequent chapters.

- The first contribution of this thesis is a new orientation descriptor called the Histogram of Optimized Local Binary Pattern(HOOPLBP). Motived in part by the local binary pattern(LBP), our method use the bit-wise shift times by which it achieves the minimum LBP value, instead of the minimum LBP value as the feature value. The feature makes automatic image orientation possible. Then the use of a histogram offers us a robust method to detect image orientation.

- The second contribution of the thesis is an optimization method for our proposed image orientation detection method. A large number of linear interpolations operations make the

method so time-consuming. The optimization converts the problem to the task of solving a quartic equation which can be easily done. This not only makes the algorithm more robust, but also increases the calculation precision.

- The third contribution of this thesis is to use the HOOPLBP on human face orientation detection. Human faces, of no matter what races or sex, have a set of similar features. Although they are easily observed by humans, it is difficult for computers. By training a general face model based on the HOOPLBP, our method can successfully detect the orientation of a human face.

- The fourth contribution of this thesis is to propose a hybrid face detection system. The state-of-the-art face detection systems use either a pyramid searching scheme or a candidate selection based scheme. One advantage of candidate selection based method is fast, which can eliminate most of the non-face regions. Our porposed hybrid face detection system integrates the candidate selection phase with our HOOPLBP operation to succseefully detect faces with rotation within the plane. Our system is the first that can detect faces with arbitrary-degree-rotation.

1.3 Outline of Thesis

The remainder of this thesis is organized in the following chapters:

Chapter 2: *Background*, provides a comprehensive study on the previous research in the area of automatic image orientation detection. The problem they solved and the features they used are presented here. The scale-invariant feature transform which is used for comparison with our proposed method is also presented in this chapter. A literature review of the face detection algorithm is given at last.

Chapter 3: *Histogram of Local Binary Pattern*, introduces our proposed automatic image orientation approach - HOOPLBP. The feature extraction and optimization are detailed in this chapter. The experiment results of HOOPLBP on digital images and human faces are also shown here.

Chapter 4: *Hybrid Face Detection Algorithm*, presents the proposed hybrid face detection algo-

rithm. A set of algorithms used in the system are shown in this chapter, including normalization, candidate selection, upright face detection.

Chapter 5: *Conclusions*, draws the concluding remarks and suggests the future research direction.

Chapter 2

Background

2.1 Introduction

THE purpose of this thesis is to propose a new and robust automatic image orientation detection algorithm and apply this method to a hybrid face detection system. In this chapter, we will review the literatures of existing image orientation detection algorithms and the state-of-the-art face detection algorithms.

The remaining of this chapter is organized as follows. First we will talk about object orientation from psychology perspective. Then a literature review of image orientation in specific fields is presented. In section 2.4, we will present the image orientation detection approaches in photo management. The algorithm and applications of scale-invariant feature transform is given in section 2.5. Finally, we will do a short review of the present face detection approaches and a summary is concluded at last.

2.2 A Psychophysical Study on the Perception of Orientation

While a small portion of the psychology literature involves human perception of orientation of gratings [7, 8], and recognition of rotated letters and digits [11, 12], most of the psychology literature involving orientation focuses on the interplay between orientation and recognition of objects, specifically the effect of in-plane rotation on the recognition of single object represented by line drawings.

A rigorous psychophysical study was conducted recently to investigate the perception of image orientation [9]. In this study, a collection of $1,000$ images (a mix of professional photos and consumer snapshots, intended to span reasonably well the photo space [10] in terms of, e.g., picture seasons, occasions, locations, indoor/outdoor, people/no people) was used. Each image was examined by at least five observers and shown at varying resolutions. At each resolution, observers were asked to indicate the image orientation, the level of confidence, and the cues they used to make the decision. Examples of these images are shown in 2.1.

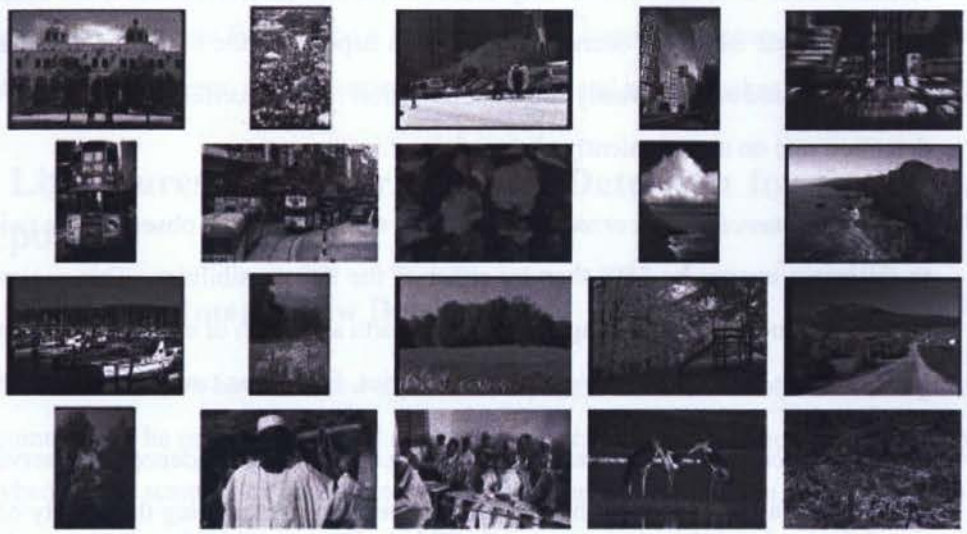


Figure 2.1: Examples of the photos used in the study

A number of insights are gained from this psychophysical study.

1. *Image resolution:* The study found that observer accuracy increases steadily with increasing resolution until what is referred to as Base / 4 (i.e. 256×384), at which point accuracy was 95.7%. Increasing to the next resolution level, 512×768, increased accuracy is less than a percentage point. A conclusion is that Base/4 is an adequate resolution for accurate orientation by human observers, and therefore, is probably a reasonably adequate resolution for automatic algorithms as well, especially considering the limitations of such algorithms in recognizing semantic objects.

2. *Upper bounds on accuracy:* It is safe to assume that an automatic orientation algorithm will not surpass the accuracy of an average human observer on an unconstrained set of images, given that inferring orientation is a task that humans are trained to do well. Humans are able to recognize thousands of objects and use high-level reasoning to deduce orientation. An automatic algorithm cannot rival that level of sophistication in the foreseeable future. Human performance, therefore, represents an upper bound on the accuracy that an algorithm can attain. We conclude that an upper bound for accuracy of an algorithm using all available semantic cues is about 96%. If only coarse semantics from thumbnails are used, the upper bound is about 84%. Of course, these bounds depend on the nature of the image set. An algorithm could achieve vastly different detection rates on different image sets, even 100% detection rate on a conveniently chosen dataset.

3. *Relative frequencies of incorrect answers:* The study found that observers are twice as likely to misorient images by 180° than by either of the 90° possibilities. This suggests that observers use cues that can distinguish between north and south or east and west, but are unable to distinguish between the remaining possibilities. Such cues could include horizon lines.

4. *Orientation confidence:* It was found that accuracy and confidence of observations were highly correlated, indicating that humans are very good at judging the quality of their decisions. This would be a very desirable characteristic of an automatic algorithm. When the confidence of the algorithm is low, the input image could be flagged and judged by a human, thus improving the overall accuracy of the system.

5. *Semantic cues:* Semantic cues are very important for image orientation. In this study, only 1.6% of images were oriented correctly without semantic cues. Some cues stood out as being very important, such as sky (used in 31% of the correct observations), and people (36.7% of the correct observations). Other important semantic cues include cloud, water, grass, and trees. In fact, a combination of sky, grass, or people were used in over 70% of the correct observations. These objects are all fairly well defined. We are in the process of developing more robust automatic algorithms for detecting these types of objects [9].

Other cues mentioned by observers are not as well defined, making detection of them by an automatic algorithm more difficult. Such cues include categories of objects, such as animals (all species), buildings (all types and styles), ground (dirt, carpet, tiles, etc.), furniture (all types), and vehicles (all types and models). Among them, it is possible and beneficial to develop automatic detectors for sub-categories of objects: the most promising cases include skyscrapers [15], passenger cars [16], paved road [14] and sand [13]. We do note that many of the published semantic object detectors actually use location cues (therefore explicitly assuming the correct image orientation). The least accurate semantic cues were flowers and snow. Text was found to be a low-payoff cue, because it occurs infrequently in typical photographic images, and the variety of languages and scripts makes it difficult to use.

2.3 Literatures about Orientation Detection for Specific Purposes

2.3.1 Document Image Skew Detection

Most of the related topics have focused on document page orientation detection. The digital image of a document may be rotated or skewed at an arbitrary angle because of how it was placed on the platen when it was scanned or because of a document feeder malfunction [17]. This results in a skewed image such as that shown in Fig. 2.2(b). This represents a skew of only 5 degrees. In fact, a skew of as little as 0.1 degrees may be apparent to a human observer. Thus, a desirable function in a digital photocopier is the automatic detection and correction of skew. Ideally, an input such as that shown in Fig. 2.2(b) would produce Fig. 2.2(a) as output. A skew detection algorithm is given the digital image of a document and it determines the angle (possibly zero degrees) by which it was skewed when it was digitized. It is assumed that there also exists a method for rotating the image to remove the skew.

A simple solution for skew detection is to determine the location of at least two corners of the original document [18] and compute the skew angle from them. However, this can be error-prone because of non-linear distortions that occur when pages are not flat on the platen [19]. Also, the entire scan surface may be obscured by the input document or the input itself has been produced

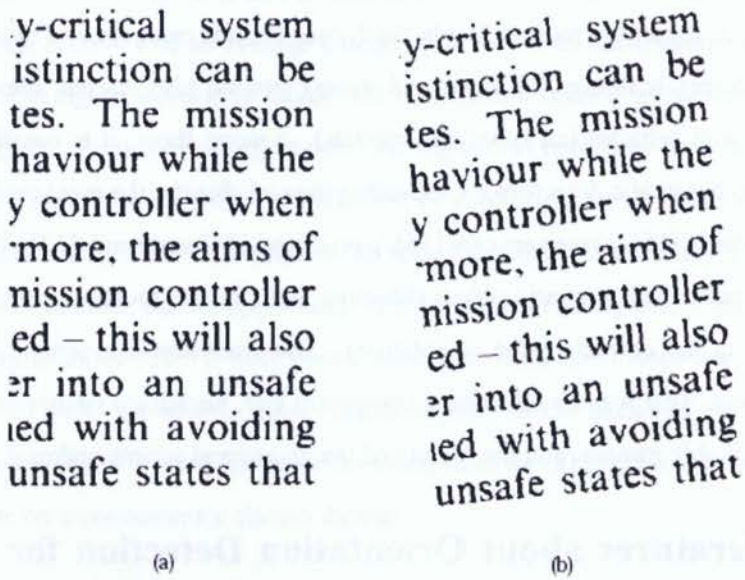


Figure 2.2: Original image in correct alignment (a) and skewed by 5 degrees (b) [17].

from a skewed original. In either case, deriving the skew angle from the corners or edges of the page is problematic. In almost every case, the algorithms that are discussed assume that an input document contains some amount of text. Features are often extracted from the text portion of the image that allow the skew to be calculated. This is done because the text is usually structured into lines that are co-linear and aligned with the horizontal (or vertical) axis of the page. Thus, detecting the skew of the text lines provides the skew of the document.

Techniques described in the scientific and patent literature often use the following four algorithms.

- Projection Profile Analysis
- Feature Point Distribution
- Hough Transform Analysis
- Orientation-sensitive Feature Analysis

The performance of most methods reported in the literature range up to a 0.1 degree accuracy. While it is arguable whether this fine resolution is needed in a digital copier application, at least a resolution of 0.2 to 0.3 degrees should be achieved. Only preliminary efforts have been conducted in comparative performance evaluation [20]. Further work in this area could help show the strengths and weaknesses of individual algorithms.

2.3.2 Medical Image Orientation Detection

The rapid rate of advancement in computer technology and image acquisition devices makes the hospitals a purely digital radiology department possible. Furthermore, The patient's digital medical images will be among the medical charts and records. Several modalities, such as Computed Tomography (CT), Ultra Sound, and Magnetic Resonance Imaging (MRI) can contribute to the images in the patient records. A technician could preview every image, but this wastes resources. It is only applicable to the daily caseload and would not include archival recall for comparisons against a patients past history. A routine that will automatically orient the medical images will avoid the necessity of having a technician preview every image.

As far as we know, few papers are published in doing auotmatic medical image orientation. In our reviews, only one about chest image was found [21]. The motivation is to automatically display the chest images in proper orientation to save radiologists' time. Linear regression is used on two othogonal profiles to determine the top of the image. The edge of the heart is found to make sure the image is not displayed as a mirror image. The algorithm was 90.4% successful on 115 chest images.

2.4 Image Orientation Detection in Photo Management

With the development of digital photography as well as inexpensive scanners, it is possible for us to store vacation and family photographs on our personal computers. This has created a need for developing image management systems that assist the user in storing, indexing, browsing, and retrieving images from a database. All image management systems require information about the true image orientation. When a user scans a picture, he expects the resulting image to be displayed

in its correct orientation, regardless of the orientation in which the photograph was placed on the scanner [23]. Thus, an image management system is expected to correctly orient the input images.

The number of literatures about automatic orientaion detection in photo management are also very few. Although the system is for specific purpose, we use a seperate section to review this topic, because the photos contain various content and only features which can deal with this situation can be used in the system. We will review the very few systems from three aspects : problem statement, feature extraction and classification systems.

2.4.1 Problem Statement

In all of the paper reviewed [6, 22, 23], the authors defined the correct orientation of an image as the orientation in which the scene, captured by the image, originally occurred. Because of the camera rotation while taking a picture or mis-placement of the photograph on a scanner, a digital image could be incorrectly oriented. However, most of the pictures are placed on a scanner with their boundaries aligned with those of the scanner plate. When this condition is not satisfied, that is, when a picture is placed on a scanner at a random orientation, we can always first de-skew this image based on detected outer edges of the picture, as mentioned in [23] . Therefore, in these work, the authors assume that the input image is restricted to only four possible rotations that are multiples of $90°$. That is, a digitized or scanned photograph can differ from its correct orientation by $0°$ (no rotation), $90°, 180°$, or $270°$. In conclusion, they represent the orientation detection problem as a four-class classification problem, with:

$$\omega_1 = 0°, \omega_2 = 90°, \omega_3 = 180°, \omega_4 = 270°, .$$

Fig. 2.3 shows the four possible orientations of an image.

2.4.2 Feature Extraction

Automatic image orientation detection is a very difficult problem. Humans use object recognition and contextual information to identify the correct orientation of an image. Unfortunately, the state-of-the-art computer vision techniques still cannot infer the high-level knowledge abstraction of the

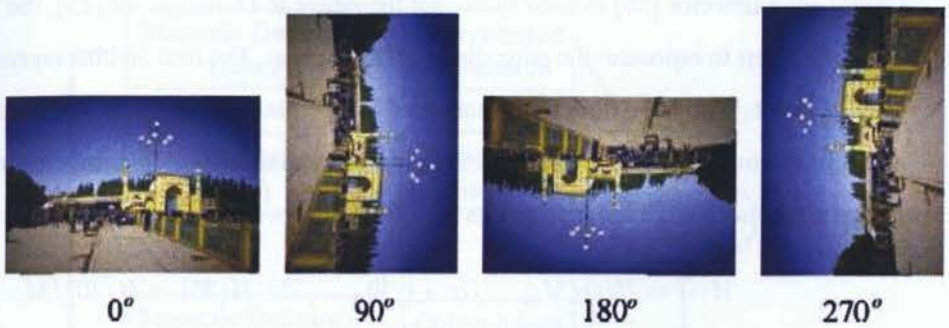


Figure 2.3: Four possible orientations of an image [23].

objects in the real world [24] . The alternative method is to exploit the low-level features from the images for orientation detection [25].

There are many potential features which can be used to represent an image. Different features have different abilities to detect the four possible orientations of the scanned image. Since global image features are not invariant to image rotation, local regional features are used for the classification in these work. An image is represented in terms of $N \times N$ blocks and the features are extracted from these local regions. The low level features used in the literatures are listed below:

- *Color Moments(CM):* It is shown in [26] that color moments (CM) of an image in the LUV color space are very simple yet very effective for color-based image analysis. In [23], the authors use the first order (mean color) and the second order moments (color variance) as the CM features to capture image chrominance information. Furthermore, the features were normalized to the same scale as follows:

$$y_i' = \frac{y_i - min_i}{max_i - min_i} \tag{2.1}$$

  where y_i represents the ith feature component of a feature vector y, min_i and max_i represent the range of values for the feature component over the training samples, and y_i' is the scaled feature component.

- *Edge Direction Histogram(EDH):* In [23], the authors utilize the edge direction histogram to characterize structural and texture information of an image, similar as that in [27]. The

Canny edge detector [28] is used to extract the edges in an image. In [23], the authors use a total of 37 bins to represent the edge direction histogram. The first 36 bins represent the count of edge points with edge directions quantized at 10° intervals. And the last bin represents the count of the number of pixels that do not contribute to an edge. To compensate for different image sizes, normalize the histograms as follows:

$$H(i) = H(i)/M_e, \quad for\ i \in [0, \ldots, 35]; H(36) = H(36)/M,$$

where $H(i)$ is the count in bin i of the edge direction histogram; M_e is the total number of edge points detected in the sub-block of an image; and M is the total number of pixels in the sub-block.

- *Semantic cues:* In [22], semantic cues are selected based on their correlation to image orientation, occurrence, and confidence of the corresponding detectors they can build. The cues chosen are : *face, blue sky, cloudy sky, ceiling/wall, and grass* in order to decrease usefulness, supported by the psychophysical study [9]. Other semantic cues, such as open water, building, furniture, cars, flowers, and text, incur diminishing returns and increasing difficulties for building the associated detector.

An integrated approach using low-level features and semantic cues is presented in [22]. Their system is shown in Fig. 2.4.

2.4.3 Classification Systems

1. Support Vector Machine

   Support vector machine [29] , which was introduced by Vapnik [30], [31], utilizes the structural risk minimization principle. It is primarily a dichotomy classifier. The optimization criterion is the width of the margin between the classes, i.e., the empty area around the decision boundary defined by the distance to the nearest training samples. These patterns, called the *support vectors*, finally define the classification function. The SVMs use optimization methods to maximize the gap between the classes. An SVM with a large margin separating two classes has a small VC dimension. A major advantage of SVM is its good

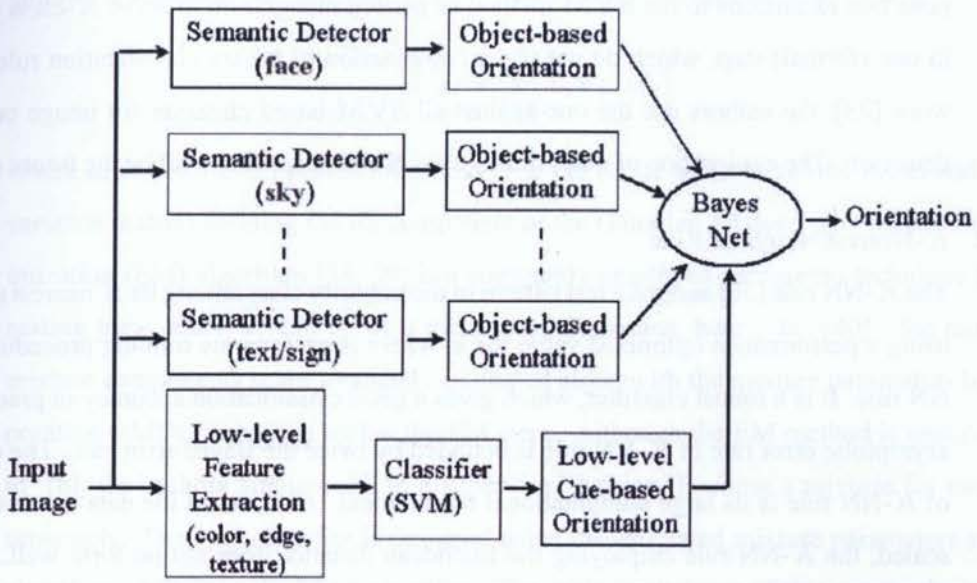


Figure 2.4: An integrated approach to image orientation detection using lowlevel and semantic cues [22].

generalization performance, which has also been demonstrated in some applications, such as object recognition [32] and face detection [33]. Since in this work, people aim at detecting the orientations of large varieties of images, good generalization over image variety is a desired property for our problem here. The computational complexity of the training procedure (a quadratic minimization problem) is one of the drawbacks of SVM. In [6], a number of classifiers were trained using different kernels (linear, polynomial, radial basis function, and sigmoid) for SVM. The best classification accuracy was achieved when a polynomial kernel function of degree 3 was used. Multi-class pattern recognition problems with SVMs (i.e., problems with more than two classes, where one has $K > 2$ classes) are typically solved by using voting scheme methods based on combining many binary classification decision functions [34] with either one-against-all or one-against-one schemes. In the one-against-all scheme, K classifiers are placed in parallel, one for each class. The Kth classifier constructs a hyper-plane between class K and the $K-1$ other classes. A new input can be classified by choosing the maximum applied to the outputs of K SVMs. Weston and Watkins [35] pro-

pose two extensions to the SVM method of pattern recognition to solve K-class problems in one (formal) step, which do not use a combination of binary classification rules. In the work [23], the authors use the one-against-all SVM-based classifier for image orientation detection. The exploration of other multi- class SVM classifiers will be the future direction.

2. K-Nearest Neighbor Rule

   The K-NN rule [36] assigns a test pattern to the majority class among its K nearest neighbors using a performance optimized value for k. There is no separate training procedure for K-NN rule. It is a robust classifier, which gives a good classification accuracy in practice. The asymptotic error rate of K-NN rule is bounded by twice the Bayes error rate. The drawback of K-NN rule is its large computational requirement. Also, when the data are not properly scaled, the K-NN rule employing the Euclidean distance does not perform well. So, data normalization is inevitable in most cases.

3. Hierarchical Discriminating Regression Tree

   The HDR algorithm [37] casts classification and regression problems into a unified regression framework. This unified view enables classification problems to use numeric information in the output space-distance metric among clustered class labels for coarse and fine classifications. Clustering is performed in both output space and input space at each internal node of the regression tree. Clustering in the output space provides virtual labels for computing clusters in the input space. Features in the input space are automatically derived from the clusters in the output space. These discriminating features span the subspace at each internal node of the tree. A hierarchical probability distribution model is applied to the resulting discriminating subspace at each internal node. To relax the per-class training sample requirement of traditional discriminant analysis techniques, a sample-size dependent negative-log-likelihood (NLL) is employed.

4. Mixture of Gaussian

   A mixture of Gaussians can be used to model a data set comprising of several distinct popu-

lations. A Gaussian mixture model $G(y)$, with K components can be written as

$$G(y) = \sum_{i=1}^{k} \omega_i * p(y\theta_i) \tag{2.2}$$

where ω_i is the mixing probabilities and θ_i is the set of parameters (the mean and the covariance matrix) defining the ith component of the Gaussian mixture. The expectation maximization (EM) algorithm [38, 39] is a commonly employed parametric technique for estimating parameters, ω_i and θ_i, of a mixture of Gaussians, here . In [40] , the number of mixture components is automatically estimated along with the mixture parameters by incorporating a MDL technique within the EM steps. Although the EM method is unsupervised, in [6], the authors employed it in a supervised fashion, learning a mixture for each class separately. The final classifier is designed using the estimated mixture parameters and then classifying based on the MAP classifier. The main limitations of this approach are that it doesnt scale well to large dimensional feature vectors and does not use the complete training data (each class is separately treated independent of the others) while estimating the mixture parameters.

2.5 Scale-invariant Feature Transform

Scale-invariant feature transform (SIFT) is an algorithm in computer vision to detect and describe local features in images. The algorithm was published by David Lowe in 1999 [41]. This approach transforms an image into a large collection of local feature vectors, each of which is invariant to image translation, scaling, and rotation, and partially invariant to illumination changes and affine or 3D projection. Following are the major stages of computation used to generate the set of image features [42]:

1. **Scale-space extrema detection:** The first stage of computation searches over all scales and image locations. It is implemented efficiently by using a difference-of-Gaussian function to identify potential interest points that are invariant to scale and orientation.
2. **Keypoint localization:** At each candidate location, a detailed model is fit to determine

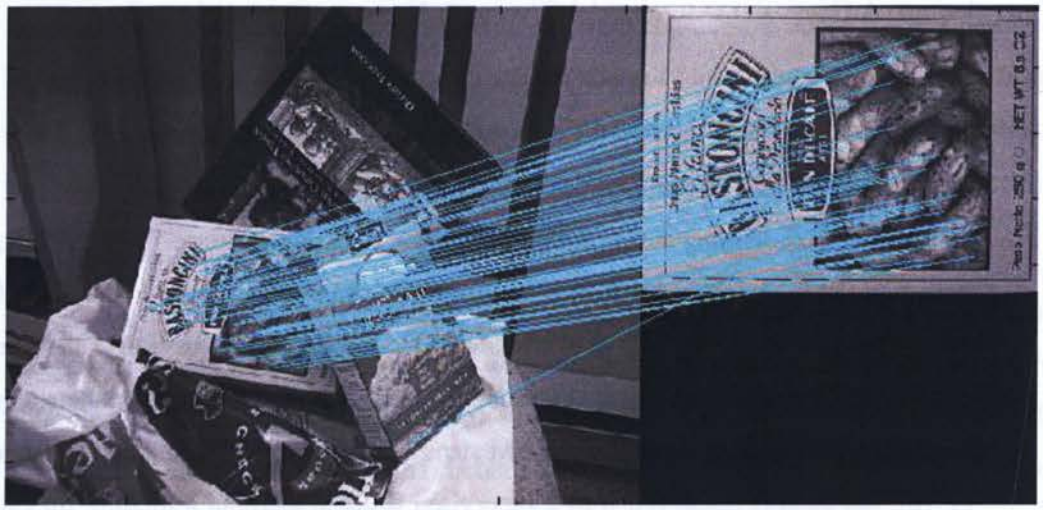


Figure 2.5: sift matching

location and scale. Keypoints are selected based on measures of their stability.

3. **Orientation assignment** One or more orientations are assigned to each keypoint location based on local image gradient directions. All future operations are performed on image data that has been transformed relative to the assigned orientation, scale, and location for each feature, thereby providing invariance to these transformations.

4. **Keypoint descriptor:** The local image gradients are measured at the selected scale in the region around each keypoint. These are transformed into a representation that allows for significant levels of local shape distortion and change in illumination.

For image matching, SIFT features are first extracted from a set of reference images and stored in a database. A new image is matched by individually comparing each feature from the new image to this previous database and finding candidate matching features based on Euclidean distance of their feature vectors, as shown in Fig. 2.5. The keypoint descriptors are highly distinctive, which allows a single feature to find its correct match with good probability in a large database of features.

The applications of SIFT includes object recognition [43], robot localization and mapping [44], panorama stiching, 3D scene modeling, recognition and tracking [45], human action recognition [46](more details can be found in [48]).

2.6 Literatures on Face Detection Algorithms

Face detection is one of the visual tasks which humans can do effortlessly. However, in computer vision, this task is not easy. A general statement of the problem can be defined as follows: Given a still or video image, detect and localize an unknown number (if any) of faces. The solution to the problem involves segmentation, extraction, and verification of faces and possibly facial features from an uncontrolled background. As a visual front-end processor, a face detection system should also be able to achieve the task regardless of illumination, orientation, and camera distance. Chellappa et al. [49] have conducted a detailed survey on face recognition research. In their survey, several issues, including segmentation and feature extraction, related to face recognition have been reviewed.

Early efforts in face detection have dated back as early as the beginning of the 1970s, where simple heuristic and anthropometric techniques [50] were used. These techniques are largely rigid due to various assumptions such as plain background, frontal face and typical passport photograph scenario. To these systems, any change of image conditions would mean fine-tuning, if not a complete redesign. Despite these problems, the growth of research interest remained stagnant until the 1990s [49], when practical face recognition and video coding systems started to become a reality. Over the past decade there has been a great deal of research interest spanning several important aspects of face detection. More robust segmentation schemes have been presented, particularly those using motion, color, and generalized information. The use of statistics and neural networks has also enabled faces to be detected from cluttered scenes at different distances from the camera. Additionally, there are numerous advances in the design of feature extractors such as the deformable templates and the active contours which can locate and track facial features accurately.

The face detection algorithms can be classified into four categories [51]:

- **Knowledge-based methods.** These rule-based methods encode human knowledge of what constitutes a typical face. Usually, the rules capture the relationships between facial features. These methods are designed mainly for face localization.

- **Feature invariant approaches.** These algorithms aim to find structural features that exist

even when the pose, viewpoint, or lighting conditions vary, and then use these features to locate faces. These methods are designed mainly for face localization.

- **Template matching methods.** Several standard patterns of a face are stored to describe the face as a whole or the facial features separately. The correlations between an input image and the stored patterns are computed for detection. These methods have been used for both face localization and detection.

- **Appearance-based methods.** In contrast to template matching, the models (or templates) are learned from a set of training images which should capture the representative variability of facial appearance. These learned models are then used for detection. These methods are designed mainly for face detection.

Fig. 2.6 summarizes algorithms and representative works for face detection in a single image within these four categories.

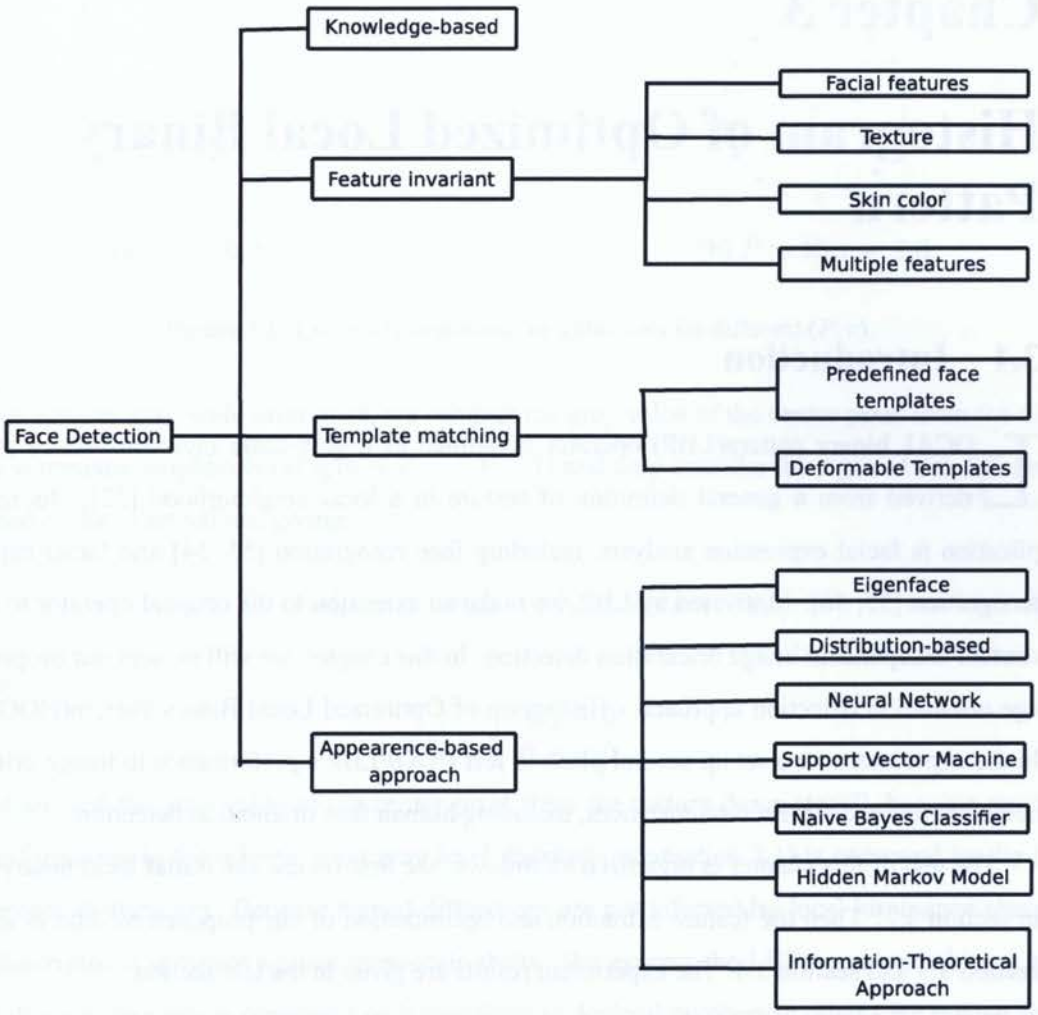


Figure 2.6: Categorization of Methods for Face Detection in a Single Image

Chapter 3

Histogram of Optimized Local Binary Pattern

3.1 Introduction

LOCAL binary pattern(LBP) operator is defined as a gray-scale invariant texture measure, derived from a general definition of texture in a local neighborhood [52]. Its main application is facial expression analysis, including face recognition [53, 54] and facial expression recognition [55, 56]. Motivated by LBP, we make an extension to the original operator to achieve content-independent image orientation detection. In this chapter, we will present our proposed image orientation detection approach - Histogram of Optimized Local Binary Pattern(HOOPLBP). In the experiments, we set up several plans to test HOOPLBP's performance in image orientation detection under different circumstances, including human face orientation detection.

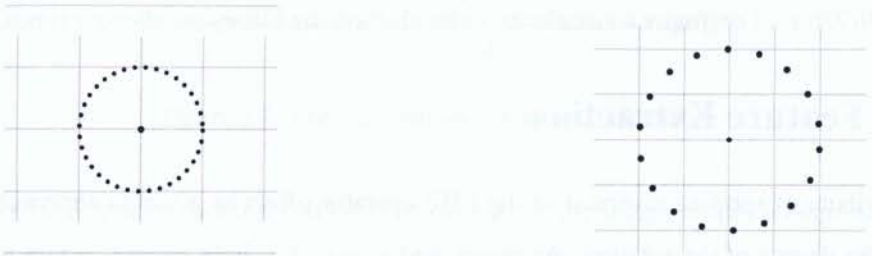
The rest of this chapter is organized as follows: we first review the initial local binary pattern in section 3.2. Then the feature extraction and optimization of our proposed method is shown in section 3.3 and section 3.4. The experiment results are given in the last section.

3.2 Gray Scale and Rotation Invariant Local Binary Patterns

The LBP operator was defined as texture descriptor. We first review the LBP operator by defining texture T in a local circle area centered on each pixel of the image:

$$T = t(g_c, g_0, \ldots, g_{p-1}), \tag{3.1}$$

where g_c is the gray value of the center pixel and $g_p(p = 0, \ldots, P-1)$ correspond to the gray values of P equally distributed pixels around the center pixel on a circle of radius $r(r > 0)$, shown in Fig. 3.1.



(a) $P = 36, r = 1.0$ (b) $P = 18, r = 2.0$

Figure 3.1: Circularly symmetric neighbor sets for different (P, r).

To achieve gray scale invariance, we subtract the gray value of the center pixel from the circularly symmetric neighborhood $g_p(p = 0, \ldots, P-1)$ and only consider the signs of the differences instead of the exact values, giving:

$$T = t(s(g_0 - g_c), s(g_1 - g_c), \ldots, s(g_{P-1} - g_c)), \tag{3.2}$$

where

$$s(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0. \end{cases}$$

We disregard the gray value of the center pixel from the texture descriptor T, because much of the information in the original joint gray level distribution(equation 3.1) is conveyed by the joint difference distribution. Because signed differences are not affected by local luminance changes, the descriptor is invariant against gray-scale shifts. We express the LBP operator in binary style for following processing compared with transfered to decimal numbers in other LBP related applications.

The LBP operator can form 2^p binary patterns, corresponding to the starting position of the P neighborhoods. When the image is rotated, the circle formed by the neighbor net of one pixel is correspondingly moved around the pixel. To remove the effect of the rotation, we can assign a

unique identifier to each rotation invariant local binary pattern:

$$LBP_{P,r} = min\{ROR(LBP_{P,r}, i)|i = 0, 1, \ldots, P-1\}, \tag{3.3}$$

where $ROR(x, i)$ perfomrs a circular bit-wise shift on the binary number x i times.

3.3 Feature Extraction

The attribute of rotation invariant of the LBP operator offers us a viable approach to record and detect the degree of the rotation. As mentioned above, the circle around certain pixel rotates the same degree as the whole Image rotates. To detect the degree, we record the sequence of the binary pattern which forms the minimum $LBP_{P,r}$ by recording the times of bit-wise shift i. For example, pattern 10011000 has the value of i as 5 if we perform circular left shifts and pattern 00010101 has the value of 0. We can deduce the degree of rotation with the following formula:

$$Deg = |i_1 - i_2| \times \frac{360}{P}, \tag{3.4}$$

where i_1 and i_2 denote the correspondent times of bit-wise shift of the original image and the rotated image. Hence, the difference of i of the same pixel in the original image and the rotated image denotes the degree of the rotation, and the precision of the degree is determined by the value P. The greater the value of P is, the higher the precision it achieves. If P equals 360, the precision of the degree will be $1°$ which may not be perceived by a human.

However it is not viable to locate the same pixel on two images. Furthermore, after rotation, the information of the neighbor set of the pixel may vary which will affect the result of the orientation detection. The attribute of rotation invariant of the LBP operator makes the changes of the LBP value over the whole image by the same amount which is caused by the rotation of the image. This is illustrated in Fig. 3.2.

In Fig. 3.2, where O is the center of the rotation, $\angle AOA'$ is the degree of the rotation. Point A denotes a pixel in the image and point B is a neighbor of pixel A. A' and B' denote the pixels after rotation respectively. We denote α as the angle between the line $\overline{AB}$ and axis x, β as the angle between the line $\overline{A'B'}$ and axis x. It can be seen that the difference($\angle Bo'B'$) between α and

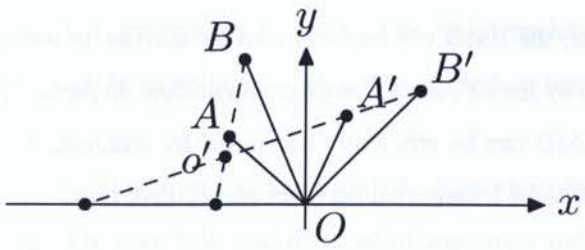


Figure 3.2: The explaination of rotational invariance

β is equal to the degree of rotation $\angle AOA'$. It means when the image rotates θ degrees around the center of O, the relative neighbor set of an arbitrary pixel A rotates the same degree θ around the center of point A.

Therefore instead of using one pixel, we can use the histogram computed over a region of image. As mentioned above, the rotation of image changes the LBP value over the whole image by the same amount. It means the degree of the rotation of image converts to the bit-wise shifts of the histogram, like $ROR(H, i|i = 0, 1, 2\ldots)$, where H denotes the histogram, i denotes the degree of rotation. Then we can deduce the degree of rotation with the following formula:

$$Deg = min\{d(H_1, ROR(H_2, i|i = 0, 1, 2\ldots))\} \tag{3.5}$$

where H_1, H_2 denote the histogram of the original image and the rotated image. $d(H_1, H_2)$ is the distance matrix used for the comparison of the histograms.

In our experiments, we use two different histogram comparison algorithms under different environments. The first is $chi - square$, high scores indicate good matches and low scores indicate bad matches, described as follows:

$$d_{chi-square}(H_1, H_2) = \sum_i \frac{(H_1(i) - H_2(i))^2}{H_1(i) + H_2(i)} \tag{3.6}$$

The other comparison method is the earth mover's distance(EMD), which is a mathematical measure of the distance between two distributions. Intuitively, given two distributions, one can be seen as a mass of earth properly spread in space, the other as a collection of holes in that same space. Then, the EMD measures the least amount of work needed to fill the holes with earth. Here, a unit of work corresponds to transporting a unit of earth by a unit of ground distance. If the

domain D is discrete, the EMD can be computed by solving an instance transportation problem, which can be solved by the so-called Hungarian algorithm. In particular, if D is a one-dimensional array of bins, the EMD can be efficiently computed by scanning the array and keeping track of how much dirt needs to be transported between consecutive bins.

3.4 Optimization

We need calculate numerous interpolations for each pixel to achieve high precision. If we want to get the precision of 1°, we need 360 calculations for each point. Therefore the process is time-consuming. In the following of the section, we give an optimization of the algorithm which saves us a great amount of time.

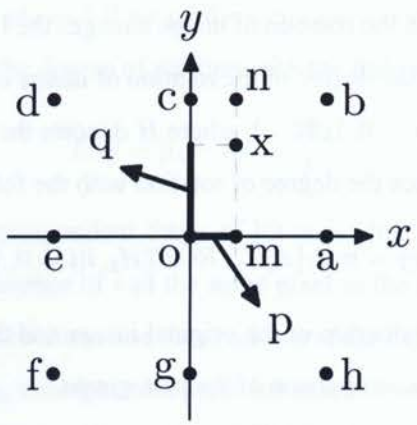


Figure 3.3: Linear interpolation in quadrant 1

The local binary pattern measures the signs of the difference between one pixel and its neighbor set regardless of the exact values of the difference. Because the neighbor sets of the pixel are calculated by linear interpolation, the values are monotonic in each one of the four quadrants. In Fig. 3.3, if the value of point a is less than that of point x, and the value of point c is bigger than that of point x, the value of the interpolated points will monotonic increasing along the perimeter of the circle from point a to point c. It means the sign of the difference will be 0 untill it meets the point whose interpolated value is the same as point x, then the sign will change to 1 and keeps

1 till the end point c. In short, it converts the numerous jobs of interpolation to finding a point on the circle whose value is the same as point x which definitely reduce the computation by a large amount.

We illustrate the mathematical relations and how to find the points in four quadrants in Fig. 3.3, Fig. 3.4, Fig. 3.5 and Fig. 3.6, each followed by a list of equations and constraints, containing two variables. Then the set of linear equations are converted to a quartic equation.

- **Quadrant 1(Fig. 3.3)**

$$\begin{aligned} p^2 + q^2 &= 1 \\ m &= x + (a - x)p \\ n &= c + (b - c)p \\ x &= m + (n - m)q \\ 0 &< p < 1 \\ 0 &< q < 1 \end{aligned} \tag{3.7}$$

where p, q are the ratio of the segments $\overline{ax}$ and $\overline{cx}$. m, n denote the temporary points on segments $\overline{ax}$ and $\overline{cx}$ after linear interpolation. By eliminating variables, the above equations can be converted to a quartic equation:

$$\begin{aligned} &(b - c - a + x)^2 p^4 + 2(b - c - a + x)(c - x)p^3 + \\ &[(c - x)^2 + (x - a)^2 - (b - c - a + x)^2]p^2 - \\ &2(b - c - a + x)(c - x)p - (c - x)^2 = 0 \end{aligned} \tag{3.8}$$

- **Quadrant 2(Fig. 3.4)**

$$\begin{aligned} p^2 + q^2 &= 1 \\ m &= x - (e - x)p \\ n &= c - (d - c)p \\ x &= m + (n - m)q \\ -1 &< p < 0 \\ 0 &< q < 1 \end{aligned} \tag{3.9}$$

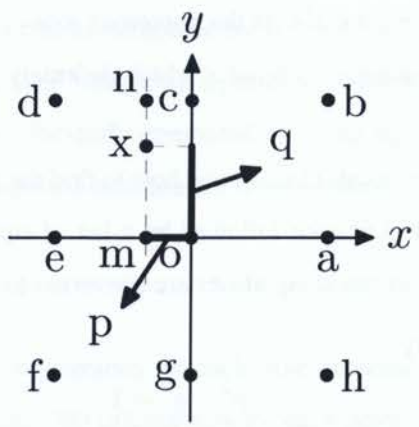


Figure 3.4: Linear interpolation in quadrant 2

where p, q are the ratio of the segments $\overline{ax}$ and $\overline{cx}$. m, n denote the temporary points on segments $\overline{ax}$ and $\overline{cx}$ after linear interpolation. By eliminating variables, the above equations can be converted to a quartic equation:

$$\begin{aligned}&(e+c-d-x)^2p^4+2(e+c-d-x)(c-x)p^3+\\&[(c-x)^2+(e-x)^2-(e+c-d-x)^2]p^2-\\&2(e+c-d-x)(c-x)p-(c-x)^2=0\end{aligned} \tag{3.10}$$

- **Quadrant 3(Fig. 3.5)**

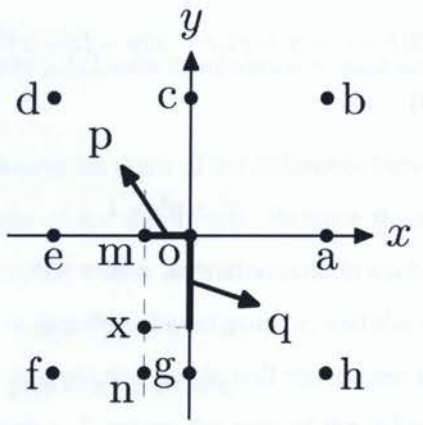


Figure 3.5: Linear interpolation in quadrant 3

$$
\begin{aligned}
p^2 + q^2 &= 1 \\
m &= x - (e - x)p \\
n &= g - (f - g)p \\
x &= m - (n - m)q \\
-1 &< p < 0 \\
-1 &< q < 0
\end{aligned}
\tag{3.11}
$$

where p, q are the ratio of the segments $\overline{ax}$ and $\overline{cx}$. m, n denote the temporary points on segments $\overline{ax}$ and $\overline{cx}$ after linear interpolation. By eliminating variables, the above equations can be converted to a quartic equation:

$$
\begin{aligned}
&(e + g - f - x)^2 p^4 + 2(e + g - f - x)(g - x)p^3 + \\
&[(g - x)^2 + (e - x)^2 - (e + g - f - x)^2]p^2 - \\
&2(e + g - f - x)(g - x)p - (g - x)^2 = 0
\end{aligned}
\tag{3.12}
$$

- **Quadrant 4(Fig. 3.6)**

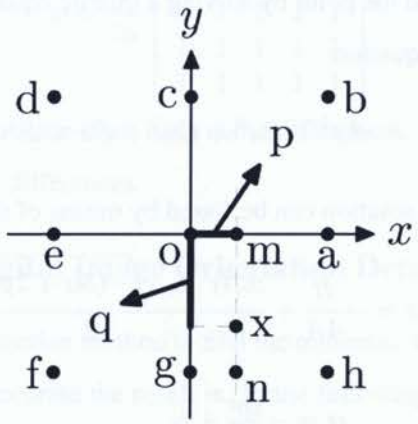


Figure 3.6: Linear interpolation in quadrant 4

$$\begin{aligned} p^2 + q^2 &= 1 \\ m &= x + (a - x)p \\ n &= g + (h - g)p \\ x &= m - (n - m)q \\ 0 &< p < 1 \\ -1 &< q < 0 \end{aligned} \tag{3.13}$$

where p, q are the ratio of the segments $\overline{ax}$ and $\overline{cx}$. m, n denote the temporary points on segments $\overline{ax}$ and $\overline{cx}$ after linear interpolation. By eliminating variables, the above equations can be converted to a quartic equation:

$$\begin{aligned} &(h - g - a + x)^2 p^4 + 2(h - g - a + x)(g - x)p^3 + \\ &[(a - x)^2 + (g - x)^2 - (h - g - a + x)^2]p^2 - \\ &2(h - g - a + x)(g - x)p - (g - x)^2 = 0 \end{aligned} \tag{3.14}$$

As we know, the quartic is the highest order polynomial equation that can be solved by radicals in the general case. So we find the point by solving a quartic equation and find the feasible solution. Given the general quartic equation:

$$Ax^4 + Bx^3 + Cx^2 + Dx + E = 0 \tag{3.15}$$

Using Ferrari's method, its solution can be found by means of the following formula:

$$x = -\frac{B}{4A} + \frac{\pm_s W \pm_t \sqrt{-(3\alpha + 2y \pm_s \frac{2\beta}{W})}}{2} \tag{3.16}$$

where

$$\begin{aligned} \alpha &= -\tfrac{3B^2}{8A^2} + \tfrac{C}{A} \\ \beta &= \tfrac{B^3}{8A^3} - \tfrac{BC}{2A^2} + \tfrac{D}{A} \\ \gamma &= -\tfrac{3B^4}{256A^4} + \tfrac{CB^2}{16A^3} - \tfrac{BD}{4A^2} + \tfrac{E}{A} \\ P &= -\tfrac{\alpha^2}{12} - \gamma \\ Q &= -\tfrac{\alpha^3}{108} + \alpha\gamma 3 - \tfrac{\beta^2}{8} \\ R &= -\tfrac{Q}{2} \pm \sqrt{\tfrac{Q^2}{4} + \tfrac{P^3}{27}} \end{aligned}$$

$U = \sqrt[3]{R}$(either sign of the square root will do)
$W = \sqrt{\alpha + 2y}$

The two $\pm_s$ must have the same sign, the $\pm_t$ is independent. To get all roots, compute x for $\pm_s, \pm_t = +, +$ and for $+, -$ and for $-, +$ and for $-, -$. This formula handles repeated roots without problem.

3.5 Experiment results

In this section, we will show the experiment results of our proposed image orientation detection methods. Originallly, the HOOPLBP is designed for digital images orientation detection. Further we apply it to human face orientation detection and achieve impressive results. We will present each experiment separately in the following sections. At first, HOOPLBP in digital image orientation detection is shown is section 3.5.1. Then application in face orientation detection is given in the next section.

Empirically, we convolve filter w with each image before applying HOOPLBP. w is an average filter:

$$w = \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

The pixel values after image rotation must have some differences. Applying an average filter can reduce the effect caused by the differences.

3.5.1 HOOPLBP in Digital Image Orientation Detection

For the experiments, we use bisection method to find the minimum range of proposed method. The bigger the range is, the more accurate the result is. In the following, we choose two different sets of images for testing. The first set is artificial images mainly from cartoons and computer games to reduce the effect of noises and fully show the ability of the method. The rotation of the images is achieved by the computer program. The second set of images are real photos. The rotation of the images is also performed by rotating the camera on purpose while taking the photos. The results

are shown in Fig. 3.7.

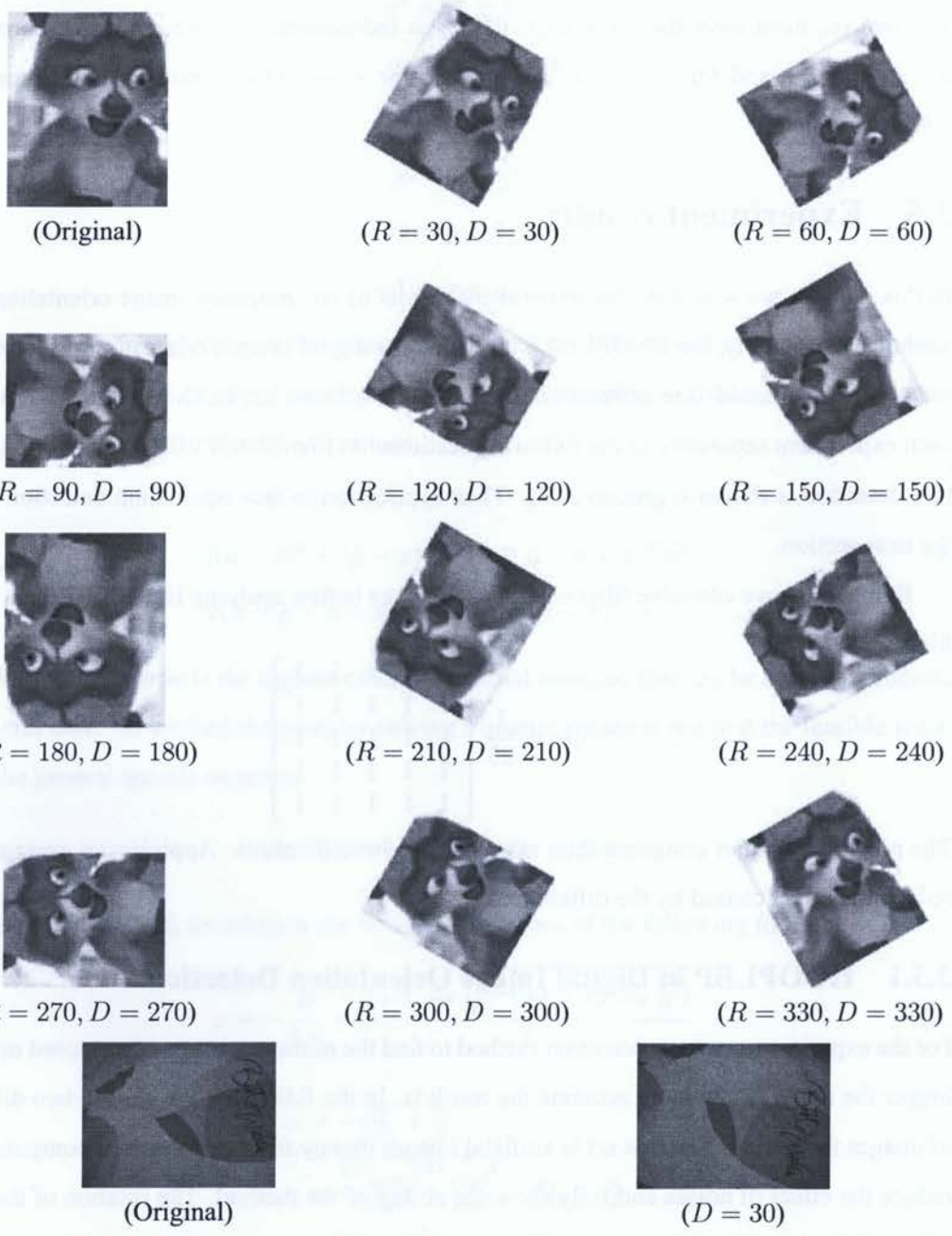
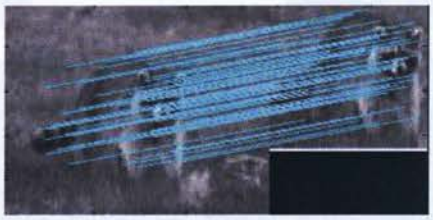


Figure 3.7: Experiments results. R and D denote the real rotation degree and detected rotation degree respectively.

The minimum range for optimum performance is 30°. Therefore, we rotated the artificial image every 30° and use $chi - square$ for histogram comparison. The degree of rotation detected by our approach is the same as which we rotated. For real photos, first we used noise reduction techniques. The result shows us the rotation degree of the camera is 30°, which is nearly the same as we estimated by human eyes. We use EMD for histogram comparison for real photos. EMD shows a better performance but is much slower.

In the next experiment, we show the differences between our proposed method and SIFT in orientation detection. SIFT is also a robust algorithm in orientation detection, as few as 3 SIFT features from an object are enough to compute its location and pose. Although it is invariant to image scale and rotation, it is restricted to image sources. It means the content of the images should be the same, because it is based on the keypoints detection and matching. The scale and rotation changes doesn't effect the content of the scene. This can be illustrated in Fig. 3.8. In Fig. 3.8(a), the two images have the same content but different scales and the matching is perfect. The difference of orientation is easily calculated as 0°. However in Fig. 3.8(b), the two images are different and there is no matching at all, although the two images are very similar.

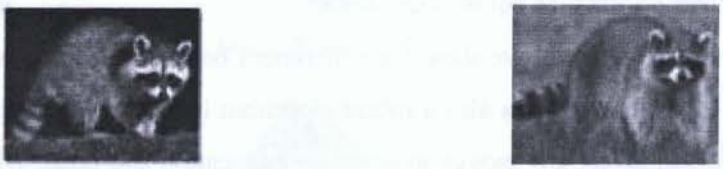


(a) SIFT features matching 1

(b) SIFT features matching 2

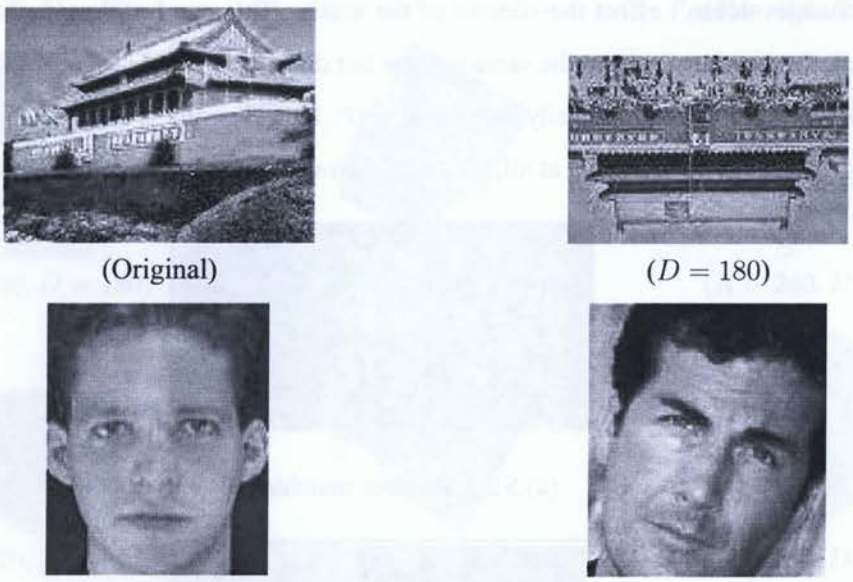
Figure 3.8: SIFT matching under different situations

Our proposed method is a content-independent approach compared with SIFT. It is based on the contrast of the nearby pixels and a region-based histogram. The advantage of the method in orientation detection is its effectiveness in similar images, as illustrated in Fig. 3.9. The degree of rotation detected by our method is $5°$ which is reasonable and acceptable. Fig. 3.10 shows some other examples using HOOPLBP in content-independant orientation detection.



(Original) ($D = 5$)

Figure 3.9: Experiment result. D denotes the detected degree by the HOOPLBP



(Original) ($D = 180$)

(Original) ($D = 24$)

Figure 3.10: Examples of content-independent orientation detection using the HOOPLBP. D denotes the detected degree by the HOOPLBP.

The algorithm proposed in section 2 was implemented under MATLAB. Using the optimization described above, the processing time is less than one second compared with several minutes of implementation without optimization.

3.5.2 HOOPLBP in Face Orientation Detection

Human faces, of no matter what races or sex, have a set of similar features. Although they are easily observed by humans, it is difficult for computers. In this section, by training a general face model based on the HOOPLBP, our method can successfully detect the orientation of a human face.

1. **Image Data and Experimental Setup**

   In our experiments, we use The Database of Faces (formerly 'the ORL Database of Faces') [57]. There are ten different images of each of 40 distinct subjects. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement). Based on the original form of the database, we rotated each image every 30 degrees. It means each single image now has 12 duplications with different orientations. The total size of our database is 4800. The samples of the database are shown in Fig. 3.11.

Figure 3.11: Samples of the database used in the experiment: one persone with different frontal position and 12 orientations.

We further divided the database into two categories for experiments: *category I* and *category II*. Category I includes eight images with 12 orientations of each subject for the first 30 subjects, totally 2880. Category II includes the rest of the database, which includes two images with 12 orientations of each subject for the first 30 subjects and all ten images with 12 orientations of each subject for the last 10 subjects, totally 1920.

2. **Experimental Plan**

Three separate experiments are planed to demonstrate the performance of HOOPLBP in face orientation detection.

(a) Use HOOPLBP to test the face orientation of the same image with different orientations. Compare the result with other content-based images.

(b) Use HOOPLBP to build one model for each orientation using database *category I* and compare the 12 models to test if the models have bit-wise shift relations.

(c) Use HOOPLBP to build one model from *category I* and use it for face orientation detection in *category II* to test the recognition rate.

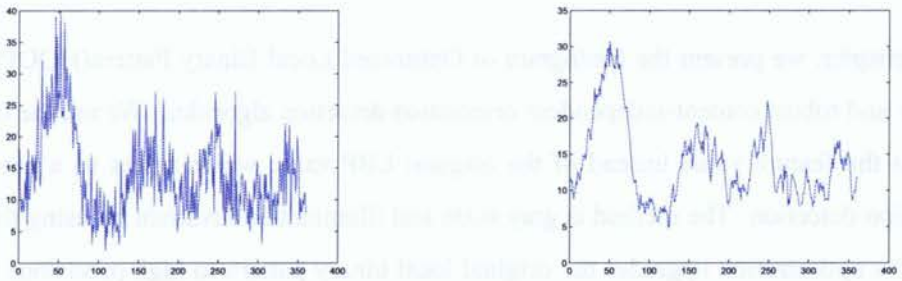
3. **Experimental Results**

In the rest of the section, we present the experimental results according to the experimental plan using the Database of Faces.

(a) In this experiment, we use bisection method to find the minimum range of orientation detection precision. The example images of human face and other content-based images are shown in Fig. 3.12. The range lies in around 30° with other content-based images, while the range decreases to less than 5° with human face images. Therefore, HOOPLBP shows better results in face images than others with the same preprocessing techniques and other environments. The results can be further improved by specific processing skills.

Figure 3.12: The examples of human face and other content-based images

(b) In this experiment, we use the mean of each orientation of *category I* for each model which are shown in Fig. 3.14. It means each model is built from 240 images. From the figure, we can observe the movements of the four waves, $30°/\ each\ time$, which shows their bit-wise shift relations. The orientation detection rate is 100% among these 12 models. Because we get the models from the means of different images, we can conclude that HOOPLBP is a content-independent orientation descriptor and works very well with human faces.



(a) (b)

Figure 3.13: The HOOPLBP of a normal single face and the feature after average filter

(c) From the above experiment, the 12 models have close bit-wise shift relations. Therefore we simply use the upright face model as a universal model for this experiment. The HOOPLBP of a normal single face and the histogram after average filter is shown

in Fig. 3.13. The orientation recognition rate with database *category II* is 66.67% with 80% upright face orientation recognition rate. We concern about the orientation recognition rate of upright face more, because they can be detected by present face detection approaches. We don't want them be mis-detected by our algorithm. By multiplying $\lambda(0 < \lambda < 1)$ with the $chi - square$ value around 0 degree, we can increase the orientation recognition rate of upright face without decrease the overall orientation detection rate a lot. When $\lambda = 0.9$, the orientation recognition rate over *category II* is 63.02% with 90.62% upright face orientation recognition rate.

The experiments show a very impressive results, especially the orientation recognition rate with 100% among the models. Considering that the images have differences in the head orientation, the lighting and the scaling and the face details are also various at beard, wrinkles, glasses and expressions, the orientation recognition rate is good enough. From Fig. 3.13(a), the HOOPLBP of a single face contains many high-frequency signals, but the ensemble shows clear indication and very similar with the above models.

3.6 Summary

In this chapter, we present the Histogram of Optimized Local Binary Pattern(HOOPLBP) which is a new and robust content-independent orientation detection algorithm. We use the bit-wise shift times as the feature value instead of the original LBP value which offers us a new method in orientation detection. The method is gray scale and illumination invariant by using the pixel contrast. The optimization upgrades the original local binary pattern to high precisions and reduces the run time by a large amount. The usage of histogram makes the proposed method content-independent, which makes the method different from other approaches. The experiments show that the method is robust and efficient. Moreover, we set up a set of experiments for the application of human face orientation detection and got very positive and impressive results. Based on LBP, HOOPLBP gets better results in human face images than other content-based images. The algorithm can successfully detect face orientations, rotated up to $\pm 180°$ in the image plane.

Without further pre-processing and training, the overall recognition rate can achieve 63.02% with upright face recognition rate to 90.62%. The future research will focus on learning a better model to increase the orientation detection rate.

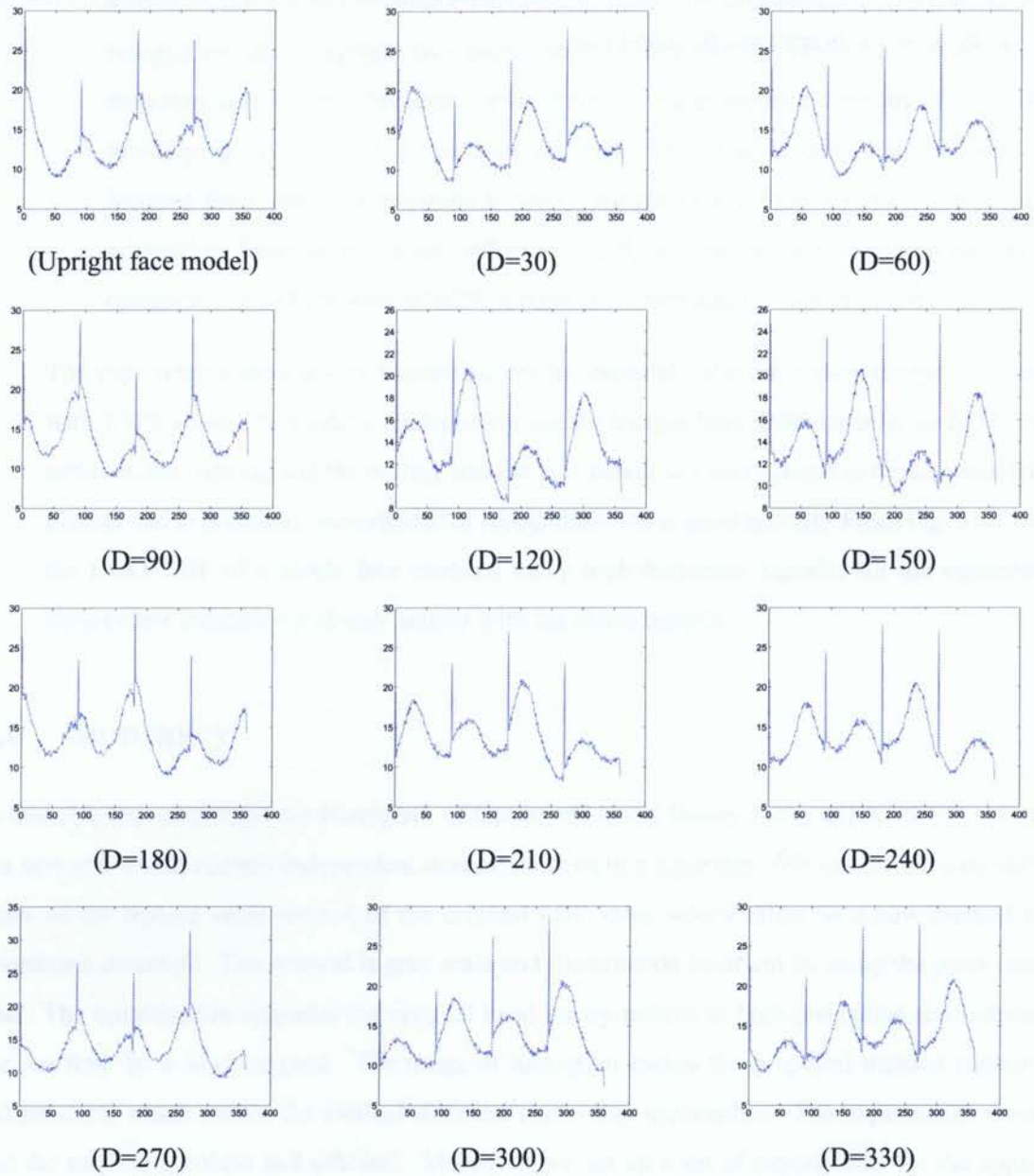


Figure 3.14: The HOOPLBP models for 12 orientations. *D* denotes the rotation degree between the model and the upright face model.

Chapter 4

Hybrid Face Detection Algorithm

4.1 Introduction

FACE detection is an important task in computer vision, especially in human-computer interaction. Most recent face detection approaches detect only upright human faces. Some reported that it can detect highly variable face patterns, rotated up to ± 20 degrees in image plane [58]. With our tests, the algorithm in [59] can detect faces within 15 degrees. However there are lots of images and photos with human faces rotated up to ± 180 degrees. [60] presents a face orientation detection method. However it detects the orientation after face detection. So a face orientation detection method before face detection is needed. In this chapter, we propose a hybrid face detection system by integrating the HOOPLBP and candidate selection method. The proposed system can successfully detect the faces within 360° rotation.

In the remaining of this chapter, we will present the details of our proposed system. In section 4.2, the diagram of our proposed system is shown. Then we implement the system by two different methods according to that if the input images are in grayscale or in color. We will discuss the methods of grayscale images in section 4.3, and the face detection system of color images is shown in section 4.4, followed by the summary of this chapter.

4.2 Proposed System

The state-of-the-art face detection system use either a pyramid searching scheme or a candidate selection based scheme. One advantage of candidate selection based method is fast, which can eliminate most of the non-face regions. In this thesis, we propose a hybrid face detection algorithm which integrates the candidate selection phase with our HOOPLBP operation to succseefully detect faces with rotation within the plane. Our system is the first that can detect faces with arbitrary-degree-rotation. The diagram of the system is shown in Fig. 4.1.

The conception of the system is as follows. By selecting the potential face regions, apply the face orientation detection method to these candidates. Rotate faces to its upright position, and detect the faces. An illustration of the process used in this thesis is given in Fig. 4.2.

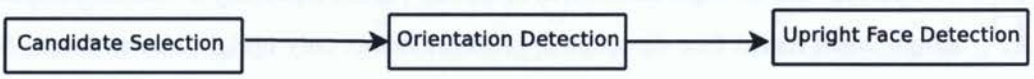


Figure 4.1: The diagram of the hybrid face detection system

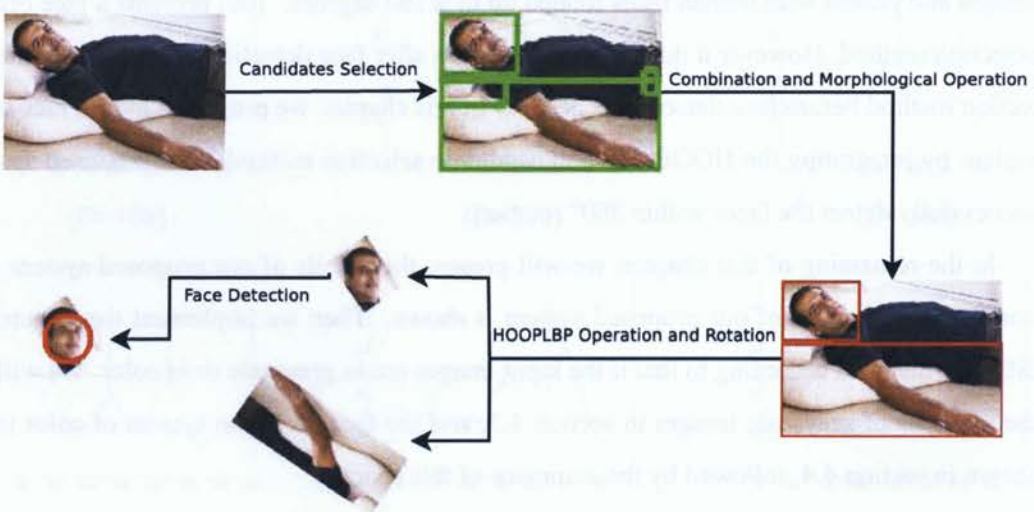


Figure 4.2: An illustration of the proposed hybrid face detection system

The first stage of the proposed scheme is locating the potential face regions. For color images, skin color segmentation is the most usable technology for candidate selection. However there

are also lots of gray images to be processed. Furthermore, most of the database for face related research are gray images. In such images, there are no color information so that we cannot apply skin color segmention for candidate selection. To achieve our proposed face detection system in gray images, we trained a cascade of classifiers*(Viola-Jones detector)*. The cascade of classifier algorithm was first introduced by Paul Viola [61] and improved by Rainer Lienhart [62] for the object detection, and implemented in OpenCV library. There are totally 20 stages in the cascade for face detection implemented in OpenCV. In this thesis, we use the first 15 stages to build up a set of candidates in an image to be further processed by the HOOPLBP. The number of stages for candidate selection is decided by trial and error, to achieve the optimum candidate selection rate. Less stages lead to too many candidates that will increase the further computational complexity. Otherwise, More stages will take the tilted human faces as negative and increase the false positive rate.

After candidate selection, the HOOPLBP is used for orientation detection. The details were discussed in chapter 3. The last phase of the system is upright face detection. The upright face detection algorithm can be chosen arbitrarily, because most of the state-of-the-art face detection algorithms perform well in upright face detection. In this thesis, we use the same *Viola-Jones detector* for both grayscale image processing and color image processing. It means we will apply the cascade of classifiers twice. At first time, we use the begining several stages of the cascade for candidate selection. At second time, all the stages are used for face detection.

4.3 Face Detection in Gray Image

Viola-Jones detector offers us a great tool for candidate selection in grayscale images. It uses a cascade of classifiers to achieve fast processing. The structure of the cascade reflects the fact that within any single image an overwhelming majority of sub-windows are negative. As such, the cascade attempts to reject as many negatives as possible at the earliest stage possible. In our system, we use the first several stages of the cascade of classifiers to eliminate most of the non-face regions, but preserve the potential face regions, either upright or tilted. In the rest of this section, we will introduce the *Viola-Jones detector* first.Then training our own detector using OpenCV library

is shown. The orientation detection phase and upright face detection phase are also included. We will also give experiment results after each subsection.

4.3.1 *Viola-Jones detector*

Haar Like Features

The simple features used are reminiscent of Haar basis functions which have been used by Papageorgiou et al. [63]. More specifically, the method uses three kinds of features. The value of a two-rectangle feature is the difference between the sum of the pixels within two rectangular regions. The regions have the same size and shape and are horizontally or vertically adjacent (see Fig. 4.3). A three-rectangle feature computes the sum within two outside rectangles subtracted from the sum in a center rectangle. Finally a four-rectangle feature computes the difference between diagonal pairs of rectangles.

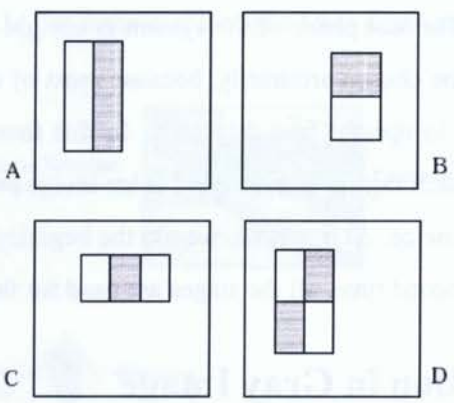


Figure 4.3: Example rectangle features shown relative to the enclosing detection window. The sum of the pixels which lie within the white rectangles are subtracted from the sum of pixels in the grey rectangles. Two-rectangle features are shown in (A) and (B). Figure (C) shows a three-rectangle feature, and (D) a four-rectangle feature.

Integral Image

Rectangle features can be computed very rapidly using an intermediate representation for the image which we call the integral image. The integral image at location x, y contains the sum of the pixels

above and to the left of x, y, inclusive:

$$ii(x,y) = \sum_{\{x' \leq x, y' \leq y\}} i(x',y') \tag{4.1}$$

where $ii(x,y)$ is the integral image and $i(x,y)$ is the original image (see Fig. 4.4). Using the following pair of recurrences:

$$s(x,y) = s(x,y-1) + i(x,y) \tag{4.2}$$

$$ii(x,y) = ii(x-1,y) + s(x,y) \tag{4.3}$$

(where $s(x,y)$ is the cumulative row sum, $s(x,-1) = 0$, and $ii(-1,y) = 0$) the integral image can be computed in one pass over the original image.

Using the integral image any rectangular sum can be computed in four array references (see Fig. 4.5). Clearly the difference between two rectangular sums can be computed in eight references. Since the two-rectangle features defined above involve adjacent rectangular sums they can be computed in six array references, eight in the case of the three-rectangle features, and nine for four-rectangle features.

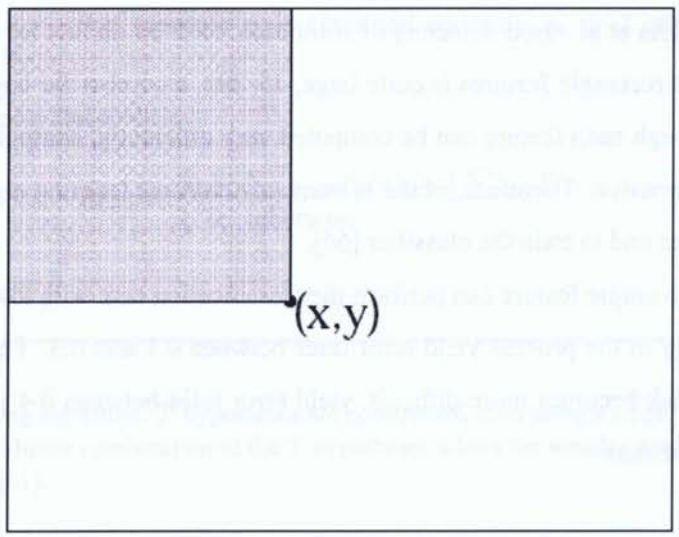


Figure 4.4: The value of the integral image at point (x,y) is the sum of all the pixels above and to the left.

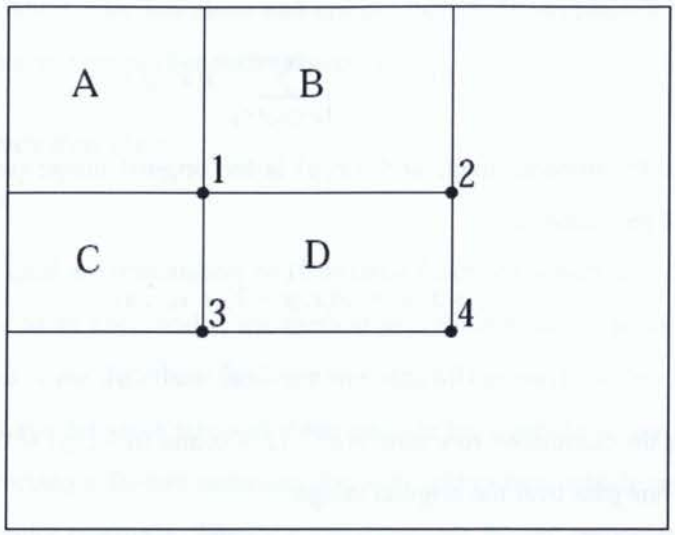


Figure 4.5: The sum of the pixels within rectangle D can be computed with four array references. The value of the integral image at location 1 is the sum of the pixels in rectangle A. The value at location 2 is $A+B$, at location 3 is $A+C$, and at location 4 is $A+B+C+D$. The sum within D can be computed as $4+1-(2+3)$.

Adaboost - Learning Classification Functions

In [61], Paul Viola et al. used detectors of resolution 24×24 on 384 by 288 pixel images. The exhaustive set of rectangle features is quite large, $45,396$, a number far larger than the number of pixels. Even though each feature can be computed very efficiently, computing the complete set is prohibitively expensive. Therefore, in the system an AdaBoost learning algorithm is used both to select the features and to train the classifier [64].

In practice no single feature can perform the classification task with low error. Features which are selected early in the process yield error rates between 0.1 and 0.3. Features selected in later rounds, as the task becomes more difficult, yield error rates between 0.4 and 0.5. Table 1 shows the learning algorithm.

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $\omega_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, \ldots, T$:
  1. Normalize the weights,
  $$\omega_{t,i} \leftarrow \frac{\omega t, i}{\sum_{j=1}^{n} \omega t, j}$$
  so that ω_t is a probability distribution.
  2. For each feature j, train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to ω_t, $\epsilon_j = \sum i\omega_i |h_j(x_i) - y_i|$.
  3. Choose the classifier h_t, with the lowest error ϵ_t.
  4. Update the weights:
  $$\omega_{t+1,i} = \omega t, i\beta t^{1-e_i}$$
  where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
- The final strong classifier is:
$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1} T\alpha_t \\ 0 & \text{otherwise} \end{cases}$$
where $\alpha_t = log\frac{1}{\beta_t}$

Table 4.1: The boosting algorithm. T hypotheses are constructed each using a single feature. The final hypothesis is a weighted linear combination of the T hypotheses where the weights are inversely proportional to the training errors [61].

Cascade

Cascade algorithm was designed for constructing a cascade of classifiers which achieves increased detection performance while radically reducing computation time. The key insight is that smaller, and therefore more efficient, boosted classifiers can be constructed which reject many of the negative sub-windows while detecting almost all positive instances. Simpler classifiers are used to reject the majority of sub-windows before more complex classifiers are called upon to achieve low false positive rates.

The structure of the cascade is that of a degenerate decision tree (see Fig. 4.6). A positive result from the first classifier triggers the evaluation of a second classifier which has also been adjusted to achieve very high detection rates. A positive result from the second classifier triggers a third classifier, and so on. A negative outcome at any point leads to the immediate rejection of the sub-window.

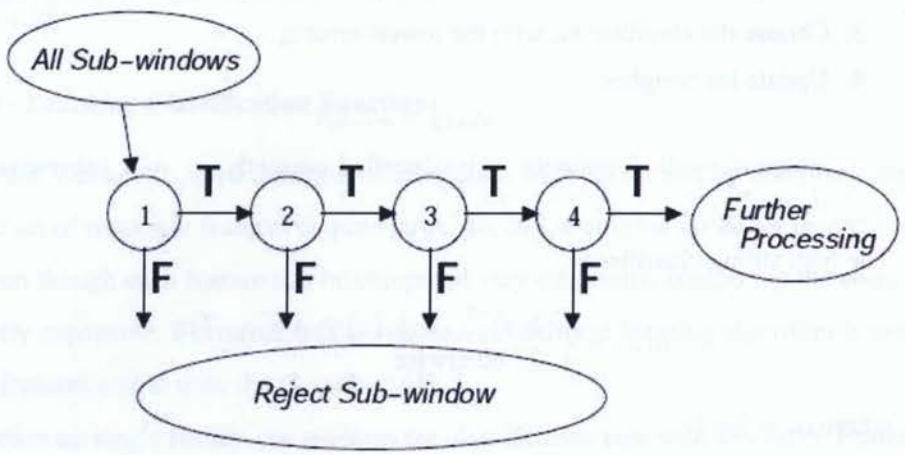


Figure 4.6: Schematic depiction of a the detection cascade. A series of classifiers are applied to every sub-window. The initial classifier eliminates a large number of negative examples with very little processing. Subsequent layers eliminate additional negatives but require additional computation. After several stages of processing the number of sub-windows have been reduced radically. Further processing can take any form such as additional stages of the cascade or an alternative detection system [61].

Each stage was trained using the Discrete Adaboost algorithm [65] . Discrete Adaboost is a powerful machine learning algorithm. It can learn a strong classifier based on a (large) set of

weak classifiers by re-weighting the training samples. Weak classifiers are only required to be slightly better than chance. At each round of boosting, the feature-based classifier is added that best classifies the weighted training samples. With increasing stage number, the number of weak classifiers, which are needed to achieve the desired detection rate and false positive rate, increases.

4.3.2 Training a Cascade Clssifier in OpenCV

Training the cascade classifier for candidate selection and upright face detection is complished by using OpenCV library. OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real time computer vision. It was originally developed by Intel, and now is supported by OpenCV community. It is free for commercial and research use under the open source BSD license. Th library is written in C and C++ and runs under Linux, Windows and Mac OS X. There is active development on interfaces for Python, Ruby, Matlab, and other languages.

One of OpenCVs goals is to provide a simple-to-use computer vision infrastructure that helps people build fairly sophisticated vision applications quickly. The OpenCV library contains over 500 functions that span many areas in vision, including factory product inspection, medical imaging, security, user interface, camera calibration, stereo vision, and robotics. Because computer vision and machine learning often go hand-in-hand, OpenCV also contains a full, general-purpose Machine Learning Library (MLL). This sublibrary is focused on statistical pattern recognition and clustering. The MLL is highly useful for the vision tasks that are at the core of OpenCVs mission, but it is general enough to be used for any machine learning problem.

The algorithm discussed in the previous section is commonly known as the *Viola-Jones detector*, which is implemented in OpenCV. OpenCV refers to this detector as the *Haar classifier* because it uses Haar features or, more precisely, Haar-like wavelets that consist of adding and subtracting rectangular image regions before thresholding the result. OpenCV ships with a set of pretrained object-recognition files, but the code also allows you to train and store new object models for the detector.

In this thesis, we used OpenCV library to train our own cascade of classifiers. The four steps of training a classifier are described as following (For more details, see the *haartraining* reference

manual supplied with OpenCV in the $opencv/apps/HaarTraining/doc$ directory.):

1. Gather a data set consisting of examples of the object you want to learn, also called positive samples. In our cases, the object we used are a set of face imgaes. Then these are stored in one or more directories indexed by a text file in the following format:

   $< path > /img\_name\_1\ count\_1\ x11\ y11\ w11\ h11\ x12\ y12 \ldots$

   $< path > /img\_name\_2\ count\_2\ x21\ y21\ w21\ h21\ x22\ y22 \ldots$

   ...

   Each of these lines contains the path (if any) and file name of the image containing the faces. This is followed by the count of how many faces are in that image and then a list of rectangles containing the faces. The format of the rectangles is the $x-$ and $y-coordinates$ of the upper left corner followed by the width and height in pixels.

   To make the classifier work well, a lot of high-quality data need to be gathered. High quality means that youve removed all unnecessary variance from the data. Therefore, in our experiment, we choose the **MIT CBCL FACE DATABASE**. It cantains $2,429$ faces for training. The images are 19×19 grayscale PGM format images, and all of them are high-quality images which are well aligned using eyes, the nose and mouth. The samples of the database are shown in Fig. 4.7. Because the positive samples we used contain only one face each that occupies the whole image, the index file in our experiment looks like this:

   $data/faces/face\_00001.pgm$ 1 0 0 19 19

   $data/faces/face\_00002.pgm$ 1 0 0 19 19

   ...

2. Use the utility application *createsamples* to build a vector output file of the positive samples. Using this file, you can repeat the training procedure below on many runs, trying different parameters while using the same vector output file. For example:

   createsamples -vec faces.vec -info faces.idx -w 30 -h 40

   This reads in the $faces.idx$ file described in step 1 and outputs a formatted training file,

Figure 4.7: The training images used in the experiment.

$faces.vec$. Then $createsamples$ extracts the positive samples from the images before normalizing and resizing them to the specified width and height (here, $30 - by - 40$). Note that $createsamples$ can also be used to synthesize data by applying geometric transformations, adding noise, altering colors, and so on. Th is procedure could be used (say) to learn a corporate logo, where you take just one image and put it through various distortions that might appear in real imagery. More details can be found in the OpenCV reference manual $haartraining$ located in */apps/ HaarTraining/doc/*.

In this thesis, we just used the $createsamples$ function to generate $.vec$ file to be used in the final step.

3. The Viola-Jones cascade is a binary classifier: It simply decides whether or not (yes or no) the object in an image is similar to the training set. Thus it needs a set of "no" samples as well, also called negative samples, so that the classifier can learn what does not look like our object. Any image that doesnt contain the object of interest can be turned into a negative sample. In this thesis, we collected the negative samples from the same **MIT CBCL**

Face Database, which are located in *train/non-face* directory. The samples of the non-faces images are shown in Fig. 4.8. Again we put the images into one or more directories and then make an index file consisting of a list of image filenames, one per line. To be specific, an image index file called *backgrounds.idx* was used in our experiment and contains the following path and filenames of image collections:

data/non-face/B1_00001.pgm

data/non-face/B2_00002.pgm

...

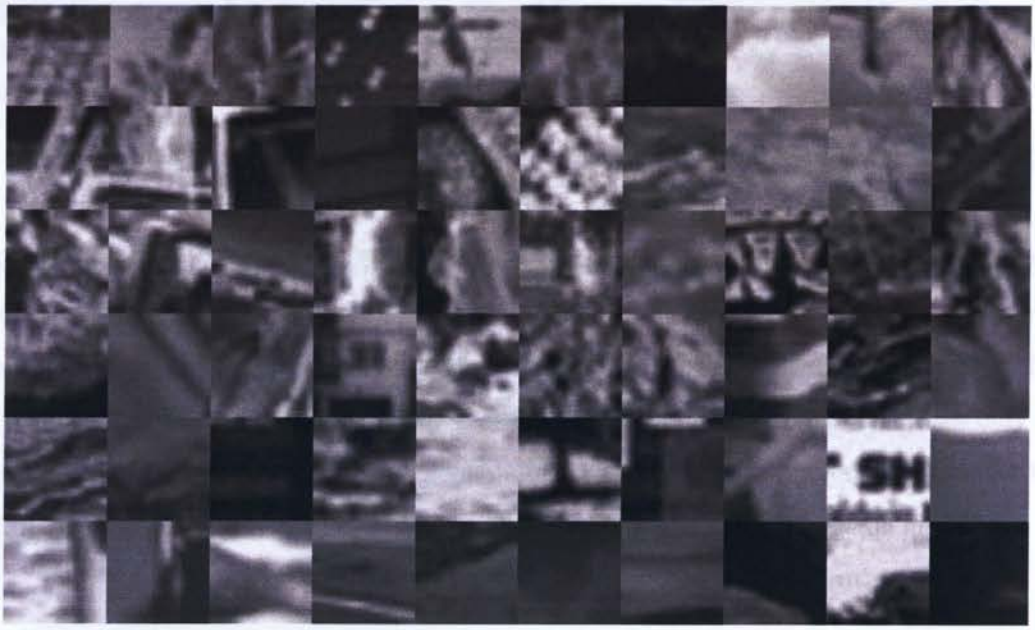


Figure 4.8: The non-faces images used for training in the experiment.

4. Training. Here's the training call which was used in the experiment:

   Haartraining /

   -data face_classifier /

   -vec faces.vec -w 19 -h 19 /

   -bg backgrounds.idx /

-nstages 15 /

-nsplits 1 /

[-nonsym] /

-minhitrate 0.998 /

-maxfalsealarm 0.5

In this call, the resulting classifier will be stored in *face_classifier.xml*. Here $faces.vec$ is the set of positive samples, and random images extracted from $backgrounds.idx$ will be used as negative samples. The cascade is set to have 15 stages, where every stage is trained to have a detection rate of 0.998 or higher.

In practice, a single face region can be detected as several candidates. It makes no sense to successfully detect the subregion orientation and will lead to final error of face detection. Toward this end it is useful to postprocess the detected skin regions in order to combine overlapping detections into a single detection. The set of detections are first partitioned into disjoint subsets. Two detections are in the same subset if their bounding regions overlap. Each partition yields a single final detection. The corners of the final bounding region are the average of the corners of all detections in the set. In some cases this postprocessing decreases the number of false positives since an overlapping subset of false positives is reduced to a single detection.

We show the results of candidate selection using the trained cascade of classifiers in Fig. 4.9 and Fig. 4.10

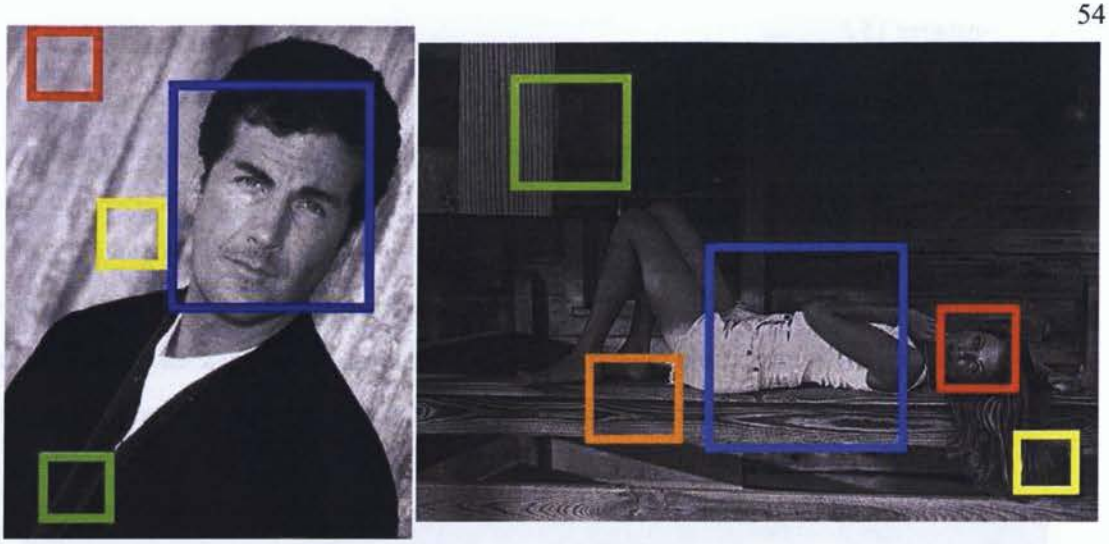


Figure 4.9: Candidate selection using the cascade of classifiers: part 1

Figure 4.10: Candidate selection using the cascade of classifiers: part 2

4.3.3 Rotation Detection and Upright Face Detection

We have several candidates from one image. Then the candidates are sent to process using the HOOPLBP. Campared with the general model we discussed in section 3.5.2, the orientation will be determined and the sub-region will be rotated by the degree. If the candidates contain a face,

it will be rotated to its upright position. Otherwise, non-face candidates will be rotated to a random orientation detected by the HOOPLBP. Examples of candidates before and after detection are shown in Fig. 4.11 and Fig. 4.12.

After rotating every candidates, they are forwarded to an existing upright face detection algorithm. In this thesis, we still use the *Viola-Jone detector* as the final detection method. It is implemented in OpenCV library, so that we suppose it have a good and stable performance. All of the 20 stages of the cascade of classifiers are used at this phase. If the detection result is positive, we map the position of the candidate back to the original input image. Otherwise, nothing will be done to the input image. Fig. 4.13 - Fig. 4.16 show the detection results by applying our proposed hybrid face detection algorithm for grayscale images.

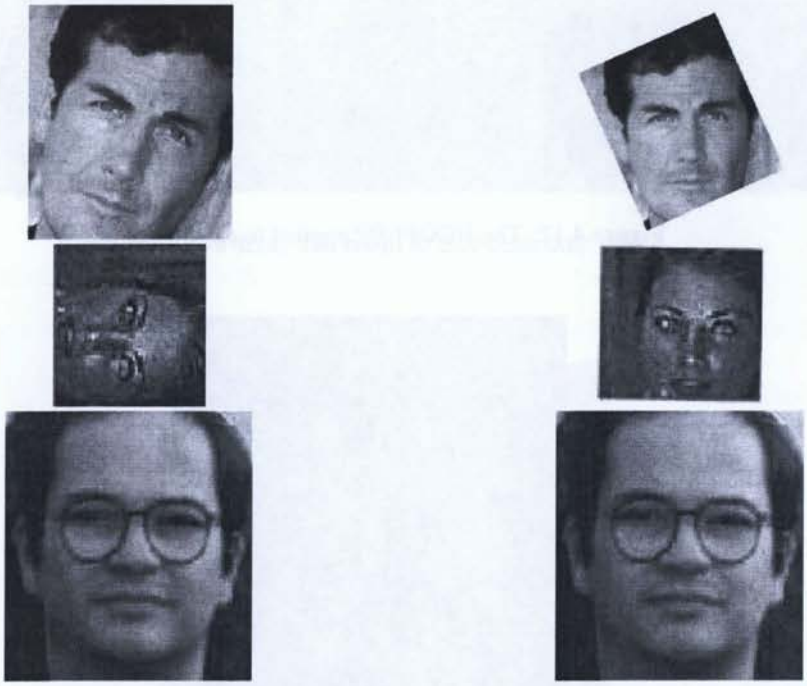


Figure 4.11: The HOOPLBP applied to candidates:part 1

Figure 4.12: The HOOPLBP applied to candidates:part 2

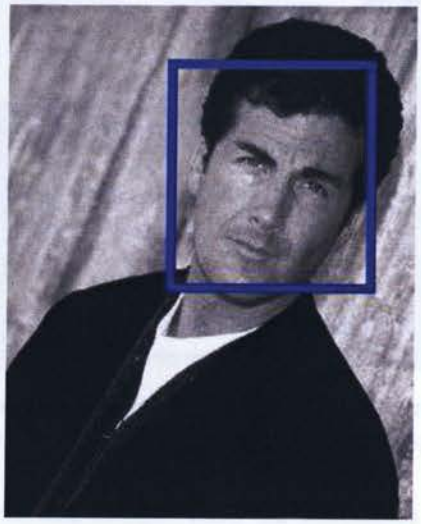


Figure 4.13: The result of face detection: part 1

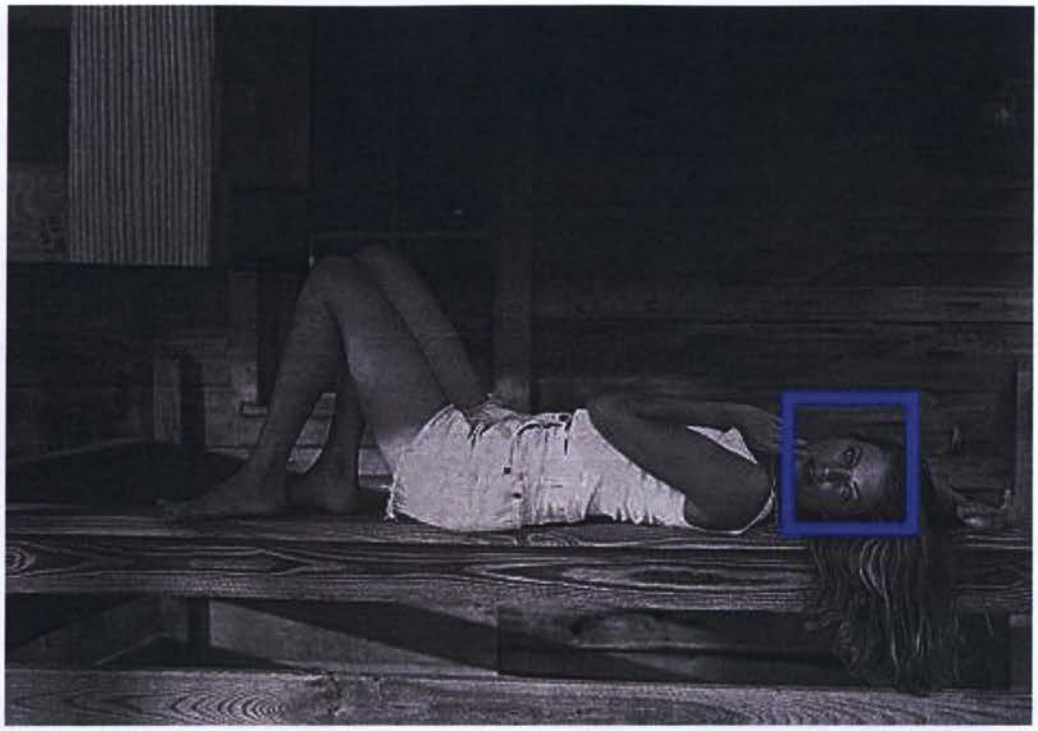


Figure 4.14: The result of face detection: part 2

Figure 4.15: The result of face detection: part 3

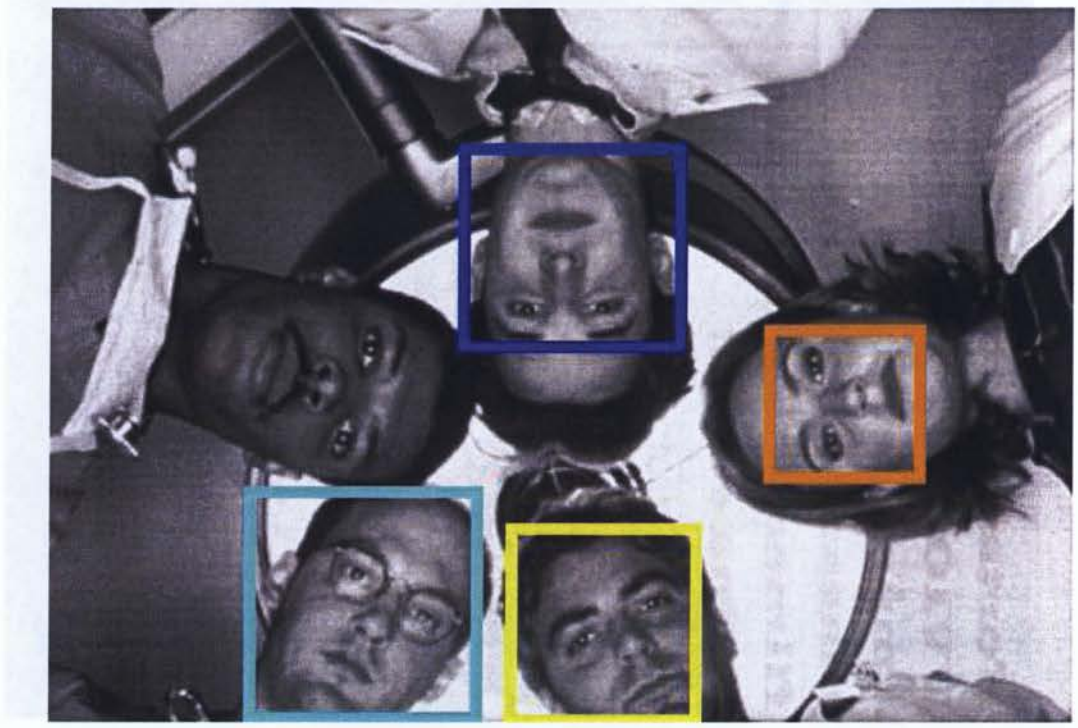


Figure 4.16: The result of face detection: part 4

4.4 Face Detection in Color Image

Color information is an efficient tool for identifying potential facial areas if the skin color model can be properly adapted for different lighting environments. However, such skin color models are not effective where the spectrum of the light source varies significantly. In other words, color appearance is often unstable due to changes in both background and foreground lighting. Though the color constancy problem has been addressed through the formulation of physics-based models [66], several approaches have been proposed to use skin color in varying lighting conditions. McKenna et al. presented an adaptive color mixture model to track faces under varying illumination conditions [67]. Instead of relying on a skin color model based on color constancy, they used a stochastic model to estimate an objects color distribution online and adapt to accommodate changes in the viewing and lighting conditions. Preliminary results show that their system can

track faces within a range of illumination conditions. However, this method cannot be applied to detect faces in a single image. In this thesis, we adopt the illumination compensation approach in [59]. The algorithm is used on images in grayscale. Therefore, we process the 3 channels of a color image separately. This illumination compensation procedure can count the effects of illumination variations, local shadowing and highlights in the original image, which may preserve the essential elements for further skin color segmentation.

In the next sections, we will present the illumination compensation methods, and skin color segmentation algorithm we used in this thesis. The experiment results are given at last.

4.4.1 Illumination Compensation

Illumination compensation consists of several stages, including gamma intensity correction (GIC), difference of Gaussian (DoG), local histogram matching (LHM) and local normal distribution (LND).

GIC corrects the overall brightness variation of the input image $s(x, y)$. This procedure compensates pixel values of an image, under unknown lighting conditions, by exponentiation to best match a canonically illuminated image $s_0(x, y)$, under the normal lighting condition. The GIC corrected image $s(x, y)$ is computed by transforming the input image over its position (x, y) pixel by pixel with an optimal Gamma coefficient γ^*

$$s'(x, y) = G(s(x, y); \gamma^*) \tag{4.4}$$

where $G(s(x, y); \gamma) = c \cdot s^{\frac{1}{\gamma}}(x, y)$, c is a gray stretch parameter, and γ^* can be computed as

$$\gamma^* = argmin_\gamma \sum_{x,y} [G(s(x, y); \gamma) - s_0(x, y)]^2 \tag{4.5}$$

GIC can enhance the local dynamic range of the face in dark or shadowed regions, compress in bright regions and at highlights, and compensates for global brightness changes of an image. In the implementation of GIC, γ^* is approximated using the golden section search with parabolic interpolation proposed in [68].

The intensity gradients such as shading effects are removed through a DoG filter, which is a popular method to obtain the resulting bandpass behavior for images. The selected values of

smaller or inner Gaussians are typically quite narrow so the detailed spatial information in high frequency is kept, while the outer ones might have more contents for low frequency.

We then apply LHM after GIC and DoG. To get the LHM transfer function, the histogram distribution of the input image and its local window are calculated first. The levels of the input image from previous processing is equalized by

$$s_k = T(r_k) = \sum_{j=0}^{k} \frac{n_j}{n}, \quad k = 0, 1, \ldots, L-1 \tag{4.6}$$

where n is the total number of pixels, n_j is the number of pixels with gray level r_j, and L is the number of discrete gray levels. The histogram distribution function $G(z)$ from the local window can be obtained by

$$G(z) = \nu_z = \sum_{0}^{z} p_z(z) \approx \sum_{i=0}^{z} \frac{n_i}{n} = s_k \tag{4.7}$$

where $p_z(z)$ represents the specified desirable PDF for output image in local window, and follow the transformation: $G(z) = T(r)$. The inverse transformation function $z = G^{-1}(s)$ is then applied to the levels obtained in Equation 4.6. The new, revised version of the original image consists of gray levels characterized by the specified density $p_z(z)$ which is then given by

$$z = G^{-1}(s) \quad or \quad z = G^{-1}[T(r)] \tag{4.8}$$

Finally, LND is applied on the resulting image by assuming the gray values drawn form a normal distribution. The output image $c(x, y)$ is normalized using (6)

$$c(x, y) = \frac{G^{-1}(s) - \mu_i}{\sigma_i} \tag{4.9}$$

where μ_i and σ_i are the mean and standard deviation of $G^{-1}[T(r)]$ over the whole image.

4.4.2 Candidate Selection

Skin chrominance information is used in skin color segmentation. In existing skin color segmentation methods, all visible colors are partitioned into two opposite groups: skin color versus nonskin color. Two color models have been evaluated and used. The YCbCr model is naturally related to MPEG and JPEG coding. The HSV (Hue, Saturation, Value) model is used mainly in computer

graphics and is considered by many to be more intuitive to use, closer to how an artist actually mixes colors. According to [69], the cluster of skin color is less compact in HSV space than in YCbCr. The projection onto the HS plane only is used by some authors like in [70] where skin color classification is performed by setting appropriate thresholds to Hue and Saturation.

In this thesis, we adopt a skin color segmentation method based on HSV color model. In a HSV color model, H ans S components contain the chromatic information and V represents the luminace information. HSV color model corresponds closely to the human intuition on color. Fig. 4.17 shows the model by an HSV color wheel(left) and the cone(right).

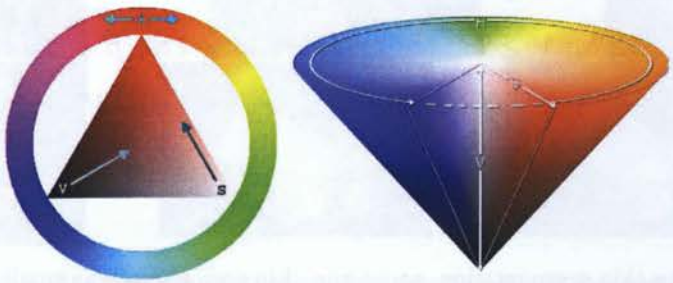


Figure 4.17: An HSV color wheel (left) allows the user to quickly select a multitude of colors. The conical representation (right) of the HSV model is well-suited to visualizing the entire HSV color space as a single object. Notice that the triangle in the left image corresponds to one face of the cone cross section in the right image [48].

In the HSV color wheel, the hue is represented by a circular region; a separate triangular region are used to represent saturation and value. Typically, the vertical axis of the triangle indicates saturation, while the horizontal axis corresponds to value. In the conical representation, the hue is depicted as a three-dimensional conical formation of the color wheel. The saturation is represented by the distance from the center of a circular cross-section of the cone, and the value is the distance from the pointed end of the cone.

To work under the HSV color space, we need to convert the original image from RGB color space to HSV color space by using the following formula [48]:

$$H = \{H_1 \; if \; B \leq G; \; 360° - H_1 \; if : B \; G\} \tag{4.10}$$

where

$$H_1 = cos^{-1}\{\frac{0.5[(R-G)+(R-B)]}{\sqrt{(R-G)(B-G)+(R-B)(G-B)}}\} \tag{4.11}$$

$$S = \frac{max(R,G,B) - min(R,G,B)}{max(R,G,B)} \tag{4.12}$$

$$V = \frac{max(R,G,B)}{255} \tag{4.13}$$

Then we directly estimated the shape of the skin color subspace in HSV. We use the planar envelop approximation method [70] to approximate the human skin color. In planar envelope method, a pixel is considered as a skin pixel if the color of the pixel satisfies the following conditions:

$$\begin{gathered} S \geq Th_s;\ V \geq Th_v;\ S \leq -H - 0.1V + 110;\ H \leq -0.4V + 75 \\ if\ (H \geq 0),\quad S \leq 0.08 \times (100 - V) \times H + 0.5V \\ else\quad S \leq 0.5H + 35 \end{gathered} \tag{4.14}$$

where Th_s and Th_v are set to 10 and 40 respectively.

After applying skin segmentation, some non-skin regions such as small isolated blobs and narrow belts are inevitably observed in the result as their color falls into the skin color space. Keeping these spurious skin regions will not only yield negative effects to the latter detection but also increase computational complexity to the orientation detection. Therefore, we apply morphological operation to implement the cleaning procedure. The closing operation is first performed to connect narrow gaps between skin regions, and then opening operation is applied to remove small isolated bulbs and seperate the regions connected by thin strips. Finally, we perform the filling operation to remove the black isolated holes. Finally, we combine the overlapping detections into a single detection.

The candidate selection and final face detection results for color images are shown in Fig. 4.18.

4.5 Summary

In this chapter, we proposed a hybrid face detection system. The system consists of 3 steps, which are candidate selection, face orientation detection and upright face detection algorithm. Skin color

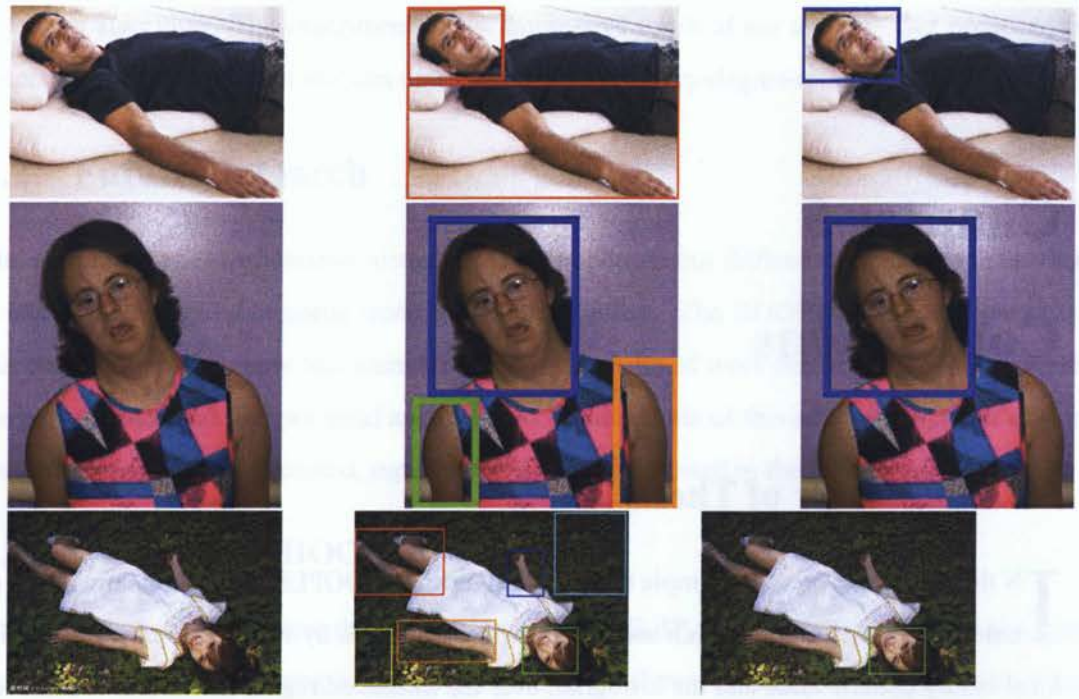


Figure 4.18: Candidate selection and face detection results for color images

segmentation is the most usable technique for potential facial region selection. However, lots of grayscale images make this method impossible because of the scarce of the imformation. Instead, we use the first 15 stages of the $Viola-Jonesdetector$ for candidate selection in grayscale images. And skin color segmentation technique is used for candidate selection in color images. Then each candidate is sent to our HOOPLBP operation to detect the right orientation if it contains a human face. We then use the same $Viola-Jonesdetector$ to detect the rotated candidates. If the result is positive, the position of the face is mapped back to the input image and a face is detected. The experiment results show the effectiveness of our proposed hybrid system.

Chapter 5

Conclusions

5.1 Summary of Thesis

IN this thesis, we present a simple and robust algorithm(HOOPLBP) for automatic image orientation detection. The approach uses the bit-wise shift times by which it achieves the minimum local binary pattern value and the histogram over the interested region as feature. It achieves image rotation by bit-wise shifting the histogram instead of really rotating images. An optimization methos is included in the algorithm. It saves a great amount of time by converting the calculationof a large number of linear interpolation to a solution of a few quartic equations.

The algorithm is effective in both artificial images and real photos. The minimum range for optimum performance is different according to different contents. In general, the range is $30°$ empirically. For human faces, the precision increases to $5°$. To be specific,the HOOPLBP has a very good effect on face orientation detection. The experiments show a very impressive results. The orientation detection rate is 100% among the 12 models. The orientation recognition rate for normal single face is 66.67% with 80% upright face orientation detection rate. Because we concern more about the detection rate of upright face, with the parameter λ, the detection rate is 63.02% over *category II* with 90.62% upright face orientation recognition rate.

Furthermore, we proposed a hybrid face detection system based on the HOOPLBP. We use a cascade of classifiers and skin color segmentation method separately for candidate selection in grayscale and color images. The HOOPLBP is used to detect the orientation of potential facial regions. Then the candidates are rotated by the detected degree, followed by an upright face

detection algorithm. The experiments show impressive result of our system. Our proposed face detection system is the first that can detect faces with arbitrary-degree-rotation.

5.2 Future Research

The automatic image orientation detection is an important but difficult task in computer vision. It is still a new field that needs more attention and effort. The HOOPLBP makes a progress in this field, but is still a new and immature algorithm. A lot of work needs to be done to improve the performance and find potential applications. In the remain of this section, a number of future research directions are discussed, especially what I am interested in the following research.

5.2.1 Variations in HOOPLBP

There are two ways to improve the performance of HOOPLBP. One is to apply more pre-processing techniques for the input images. The other is to use different patterns to integrate more nearby information.

In this thesis, we only filter the input images with an average filter w,

$$w = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

There are a lot of image processing algorithms that we can try on HOOPLBP, like kinds of filter, histogram normalization and color correction. We can even transform the input images to other presentations.

The neighbourhood pattern used in this thesis is the nearest neighbourhood. To be specific, it uses the nearest 8 points around one pixel. In the future, we can use the average pixel values as the nearest neighbour. The shape over which the average are calculated can change. We show two possible shapes in the Fig. 5.1.

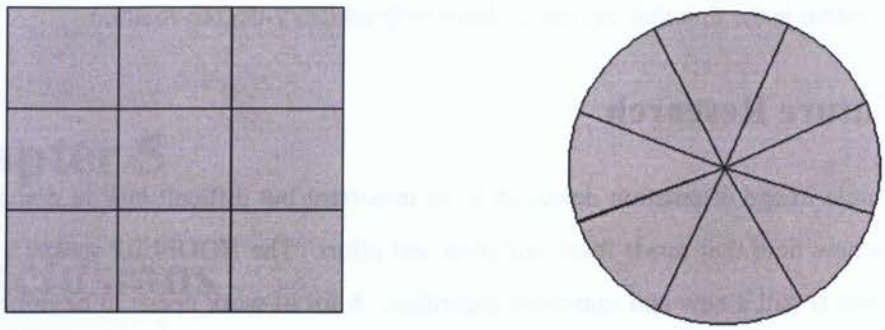


Figure 5.1: Two possible shapes which can be used for average.

5.2.2 Face Detection

The HOOPLBP has achieved impressive results in face orientation detection and the proposed hybrid face dtection system can detect faces within 360° rotation. The variant HOOPLBP mentioned above may improve the performace on face orientation detection as well.

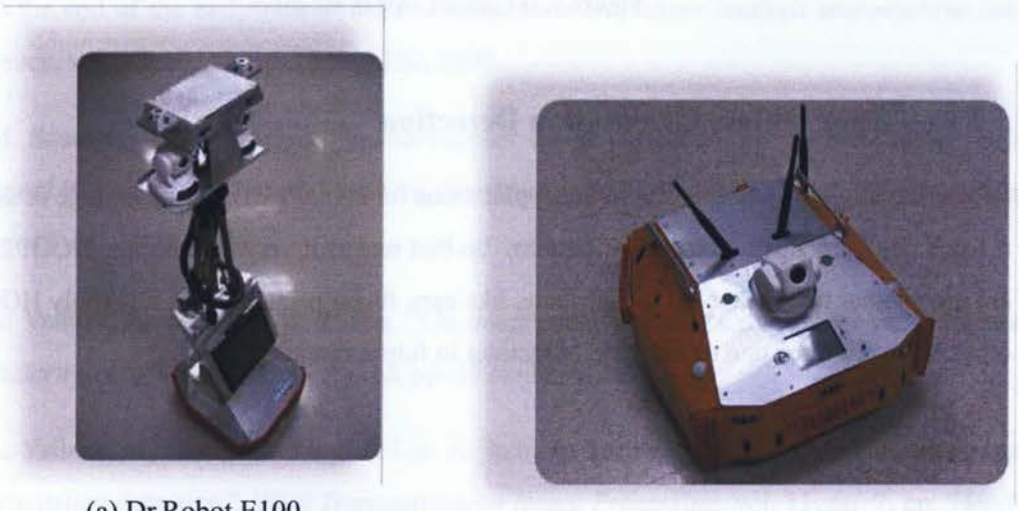
Two other key steps in our proposed hybrid system can be improved to increase the face detection rate. One is the candidate selection phase and the other is the classification method for face orentation detection. As we discussed, there are three aspects which affect face detection performace, candidate selection rate, face orientation detection rate and upright face detection rate. How to correctly and effectively select the candidate from grayscale and color images is an interesting problem.

Actually, the classification method used in this thesis for face orientation detection is very simple. A lot of other classification algorithm reviewed in section 2.4.3 can be used, and are expected to achieve better results. Genetic algorithm is a training method that may suit this research very well. In short, a well trained model or classification system will surelly improve the results on face orientation detection.

5.2.3 Application in Robots

It is irrefutable that robots and robotics will play a critical role in human society in the coming years. As the mechanical and electronic components required for reliable robotics becomes more practical, and as it becomes ever clearer that robots will become a part of our daily lives in the not too distant future, it is important now to focus on how humans will interact with these robots, and in turn how these robots will interact with us. For robots to truly integrate into human life and human society they need to be able to interact naturally with people and their surroundings. One of the key steps in the interaction is to achieve automatic human face detection. In robotics, real time algorithm is needed. How to integrate our proposed method with robots is a very specific task based on what robot system you work on.

The future research may focus on the robots that Ryerson Multimedia Lab has. The two Dr.Robot are E100 and I90. Both of the robots have similar platforms, and mainly differ by size and only a few component differences. The two robots are shown in Fig. 5.2(a) and Fig. 5.2(b). As



(a) Dr.Robot E100

(b) Dr.Robot I90

Figure 5.2: Dr.Robot in Ryerson Multimedia Lab

we saw, the Dr.Robot E100 has stereo cameras which may be popular in most robot systems. The stereo cameras will provide more information than monotonic camera. Fig. 5.3 shows the camera

used in the robot. How to use this additional information to accelerate the processing speed and integrate the stereo information to further improve our face detection system with rotation around the z axis are challenging topics.

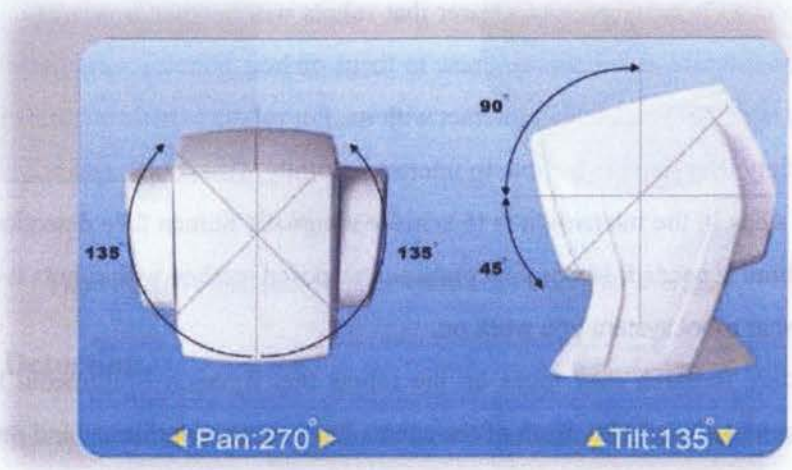


Figure 5.3: Camera used in the robot

5.2.4 Other Object Orientation Detection

Face orientation detection is one of the applications for HOOPLBP. The advantage of faces is that it has a similar shape and a similar pattern. So that makes it very suitable for HOOPLBP. There are many other things that has the attribute, like cars, finger print, houses.... Apply HOOPLBP to other objects will be one of the main objectives in future research.

Bibliography

[1] J. Luo and S. Etz, "A physical model-based approach to detecting sky in photographicimages," *IEEE Transactions on Image Processing*, vol. 11, no. 3, pp. 201–212, 2002.

[2] E. Saber, M. Tekalp, R. Eschbach, and K. Knox, "Automatic image annotation using adaptive color classification," *Graphical Models and Image Processing*, vol. 58, no. 2, pp. 115–126, 1996.

[3] A. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 12, pp. 1349–1380, 2000.

[4] M. Szummer and R. Picard, "Indoor-outdoor image classification," in *1998 IEEE International Workshop on Content-Based Access of Image and Video Database, 1998. Proceedings.*, pp. 42–51, 1998.

[5] A. Vailaya, A. Jain, and H. Zhang, "On image classification: City images vs. landscapes," *Pattern Recognition*, vol. 31, no. 12, pp. 1921–1935, 1998.

[6] A. Vailaya, H. Zhang, C. Yang, F. Liu, A. Jain, A. Technol, and P. Alto, "Automatic image orientation detection," *IEEE Transactions on Image Processing*, vol. 11, no. 7, pp. 746–755, 2002.

[7] S. Dakin, C. Williams, and R. Hess, "The interaction of first-and second-order cues to orientation," *Vision Research*, vol. 39, no. 17, pp. 2867–2884, 1999.

[8] I. Mareschal, M. Sceniak, and R. Shapley, "Contextual influences on orientation discrimination: binding local and global cues," *Vision Research*, vol. 41, no. 15, pp. 1915–1930, 2001.

[9] J. Luo, D. Crandall, A. Singhal, M. Boutell, and R. Gray, "Psychophysical study of image orientation perception," *Spatial vision*, vol. 16, no. 5, pp. 429–458, 2003.

[10] R. Segur, "Using photographic space to improve the evaluation of consumer cameras," in *Proceedings of IS&T Image Processing, Image Quality, Image Capture and Systems (PICS) Conference*. pp. 221–224, 2000.

[11] M. Corballis, N. Zbrodoff, L. Shetzer, and P. Butler, "Decisions about identity and orientation of rotated letters and digits." *Memory & Cognition*, vol. 6, no. 2, p. 98, 1978.

[12] P. JOLICOEUR, "Orientation congruency effects in visual search," *Resume*, vol. 16, pp. 351–364.

[13] M. Naphade and T. Huang, "A probabilistic framework for semantic indexing and retrieval in-video," in *2000 IEEE International Conference on Multimedia and Expo, 2000. ICME 2000*, vol. 1, 2000.

[14] N. Campbell, B. Thomas, and T. Troscianko, "Automatic segmentation and classification of outdoor images using neural networks." *International Journal of Neural Systems*, vol. 8, no. 1, p. 137, 1997.

[15] Q. Iqbal and J. Aggarwal, "Retrieval by classification of images containing large manmade objects using perceptual grouping," *Pattern recognition*, vol. 35, no. 7, pp. 1463–1479, 2002.

[16] H. Schneiderman and T. Kanade, "A statistical method for 3D object detection applied to faces and cars," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1. IEEE Computer Society; 1999, 2000.

[17] J. Hull, "Document image skew detection: Survey and annotated bibliography," *Document Analysis Systems II*, pp. 40–64, 1998.

[18] K. Sugishima and M. Yamada, "Multi-unit image processing system with central control," Jan. 10 1989, US Patent 4,797,706.

[19] S. Chen and R. Haralick, "An automatic algorithm for text skew estimation in document imagesusing recursive morphological transforms," in *IEEE International Conference Image Processing, 1994. Proceedings. ICIP-94.*, vol. 1, 1994.

[20] A. Bagdanov and J. Kanai, "Evaluation of document image skew estimation techniques," in *Proceedings of SPIE*, vol. 2660, p. 343, 1996.

[21] M. Evanoff and K. McNeill, "Automatically determining the orientation of chest images," in *Proceedings of SPIE*, vol. 3035, p. 299, 1997.

[22] J. Luo and M. Boutell, "Automatic image orientation detection via confidence-based integration of low-level and semantic cues," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 715–726, 2005.

[23] Y. Wang and H. Zhang, "Detecting image orientation based on low-level visual content," *Computer Vision and Image Understanding*, vol. 93, no. 3, pp. 328–346, 2004.

[24] L. Shapiro and G. Stockman, "Computer Vision, March 2000."

[25] R. Gonzalez and R. Woods, "Digital image processing. 2002," *ISBN: 0-201-18075-8.*

[26] W. Ma and H. Zhang, "Content-based image indexing and retrieval," *IEEE Multimedia Magazine*, vol. 1, no. 2, pp. 62–72, 1994.

[27] A. Jain and A. Vailaya, "Image retrieval using color and shape," *Pattern Recognition*, vol. 29, no. 8, pp. 1233–1244, 1996.

[28] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine intelligence*, pp. 679–698, 1986.

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

[30] V. Vapnik, "The nature of statistical learning theory. 2000,". Springer Verlag.

[31] ——, "Statistical learning theory. 1998," *NY Wiley*.

[32] V. Blanz, B. Scholkopf, H. Bulthoff, C. Burges, V. Vapnik, and T. Vetter, "Comparison of view-based object recognition algorithms using realistic 3D models," *Lecture Notes in Computer Science*, vol. 1112, pp. 251–256, 1996.

[33] E. Osuna, R. Freund, and F. Girosit, "Training support vector machines: an application to face detection," in *1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1997. Proceedings.*, pp. 130–136, 1997.

[34] B. Scholkopf, C. Burges, and V. Vapnik, "Extracting support data for a given task," in *Proceedings, First International Conference on Knowledge Discovery & Data Mining, Menlo Park*, pp. 252–257, 1995.

[35] J. Weston, "Multi-class support vector machines," *Proceedings of ESAN N99D*, 1998.

[36] R. Duda, P. Hart, and D. Stork, *Pattern classification*. Wiley New York, 2001.

[37] W. Hwang and J. Weng, "Hierarchical discriminant regression," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1277–1293, 2000.

[38] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 1–38, 1977.

[39] M. Figueiredo, J. Leitão, and A. Jain, "On fitting mixture models," *Lecture notes in computer science*, pp. 54–69, 1999.

[40] M. Figueiredo and A. Jain, "Unsupervised selection and estimation of finite mixture models," in *International Conference on Pattern Recognition*, vol. 15, pp. 87–90, 2000.

[41] D. Lowe, "Object recognition from local scale-invariant features," in *International Conference on Computer Vision*, vol. 2. Corfu, Greece, pp. 1150–1157, 1999.

[42] ——, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[43] J. Beis and D. Lowe, "Shape indexing using approximate nearest-neighbour search in high-dimensional spaces," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. pp. 1000–1006, 1997.

[44] S. Se, D. Lowe, and J. Little, "Vision-based mobile robot localization and mapping using scale-invariant features," in *In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, vol. 2, pp. 2051–2058, 2001.

[45] M. Brown and D. Lowe, "Recognising panoramas," in *Proceedings of the Ninth IEEE International Conference on Computer Vision*, p. 1218, 2003.

[46] I. Gordon and D. Lowe, "What and where: 3d object recognition with accurate pose," *Lecture Notes in Computer Science*, vol. 4170, p. 67, 2006.

[47] I. Laptev and T. Lindeberg, "Local descriptors for spatio-temporal recognition," *Lecture notes in computer science*, vol. 3667, p. 91, 2006.

[48] "Wikipedia, the free encyclopedia," *http://www.wikipedia.org*.

[49] R. Chellappa, C. Wilson, S. Sirohey *et al.*, "Human and machine recognition of faces: A survey," *Proceedings of the IEEE*, vol. 83, no. 5, pp. 705–740, 1995.

[50] T. Sakai, M. Nagao, and T. Kanade, "Computer analysis and classification of photographs of human faces," in *Proceedings of the First USA-Japan Computer Conference*, pp. 55–62, 1972.

[51] M. Yang, D. Kriegman, and N. Ahuja, "Detecting faces in images: A survey," *IEEE Transactions on Pattern analysis and Machine intelligence*, pp. 34–58, 2002.

[52] T. Ojala, M. Pietikainen, and T. Maenpaa, "Gray scale and rotation invariant texture classification with local binary patterns," *Lecture Notes in Computer Science*, vol. 1842, pp. 404–420, 2000.

[53] T. Ahonen, A. Hadid, and M. Pietikainen, "Face recognition with local binary patterns," *Lecture Notes in Computer Science*, pp. 469–481, 2004.

[54] ——, "Face description with local binary patterns: Application to face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 2037–2041, 2006.

[55] X. Feng, M. Pietikainen, and A. Hadid, "Facial expression recognition with local binary patterns and linear programming," *Pattern Recognition and Image Analysis*, vol. 15, no. 2, p. 546, 2005.

[56] C. Shan, S. Gong, and P. McOwan, "Robust facial expression recognition using local binary patterns." *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, vol. 2, pp. II-370-3, 2005.

[57] F. Samaria and A. Harter, "Parameterisation of a stochastic model for human faceidentification," in *Applications of Computer Vision, 1994., Proceedings of the Second IEEE Workshop on*, pp. 138–142, 1994.

[58] C. Garcia and M. Delakis, "Convolutional face finder: A neural architecture for fast and robust face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1408–1423, 2004.

[59] Y. Tie and L. Guan, "Automatic face detection in video sequences using local normalization and optimal adaptive correlation techniques," *Pattern Recognition*, vol. 42, no. 9, pp. 1859–1868, 2009.

[60] S. Han, G. Pan, and Z. Wu, "Human face orientation detection using power spectrum based measurements," in *Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings*, pp. 791–796, 2004

[61] P. Viola and M. Jones, "Robust real-time object detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2002.

[62] R. Lienhart and J. Maydt, "An extended set of haar-like features for rapid object detection," in *IEEE ICIP*, vol. 1, no. 2002, pp. 900–903, 2002.

[63] C. Papageorgiou, M. Oren, and T. Poggio, "A general framework for object detection," in *Computer Vision, 1998. Sixth International Conference on*, pp. 555–562, 1998.

[64] Y. Freund and R. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[65] ——, "Experiments with a new boosting algorithm,"*Machine Learning International Workshop Then Conference*. Citeseer, pp. 148–156, 1996.

[66] D. Forsyth, "A novel approach to colour constancy," in *Computer Vision., Second International Conference on*, pp. 9–18, 1998.

[67] S. McKenna, Y. Raja, and S. Gong, "Tracking colour objects using adaptive mixture models," *Image and Vision Computing*, vol. 17, no. 3-4, pp. 225–231, 1999.

[68] O. Arandjelovic and R. Cipolla, "An illumination invariant face recognition system for access control using video," *Proc. British Machine Vision Conference*, pp. 537–546, 2004.

[69] C. Garcia and G. Tziritas, "Face detection using quantized skin color regions merging and wavelet packet analysis," *IEEE Transactions on Multimedia*, vol. 1, no. 3, pp. 264–277, 1999.

[70] S. Tsekeridou and I. Pitas, "Facial feature extraction in frontal views using biometric analogies." *Proc. of EUSIPCO'98*, pp. 315–318, 1998.

Appendix A

List of Publications

The publications based on the work of this thesis are listed below:

Conference Paper

- Nan Dong, Ling Guan, "Content-free Image Orientation Detection Using Local Binary Pattern", accepted by IEEE International Workshop on Multimedia Signal Processing (MMSP), Rio de Janeiro, Brazil, October, 2009
- Nan Dong, Ling Guan, "Human Face Orientation Detection using Histogram of Optimized Local Binary Pattern", submitted to IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Dallas, Texas, U.S.A., March, 2010