Learning Transformation Invariance for Pairwise Image Matching

Chen Xi

School of Computer Engineering

A thesis submitted to the Nanyang Technological University in fulfilment of the requirement for the degree of Doctor of Philosophy

2008

Abstract

Image matching is a fundamental problem in computer vision. In this thesis, we address the image matching problem as learning and classifying correspondences. More precisely, we formulate the image matching problem as: given a set of training pairs of images that implicitly captures the transformation(with both positive and negative classes), identify if a new pair of test images is matched via the transformation class. In this formulation, all the training data, as well as test data, are image pairs. The approach taken is to consider only relative visual content, rather than absolute visual content, so the learned image matching classifier could be applied to images of *totally different visual content* as compared to the training data. This is in contrast to appearance-based object detection methods, for which once the training process is completed, the classifiers may only be used to recognize objects of the same categories with the training images.

The same problem formulation is adopted throughout the thesis, but three different approaches are proposed to deal with three different classes of transformations. The first transformation class involves low distortion transformations. We address the image matching problem under this transformation class by learning a parametric feature distance measure to measure the dissimilarity between pairs of images. The method is based on optimizing the parameters of the distance measure in order to minimize correspondence classification errors on training data. Results on matching classification with a wide variety of image content show that the learnt feature distance measure clearly outperforms the standard measures of SSD, chamfer and Bhattacharyya histogram distances.

The second transformation class involves the heavy distortion, correspondence preserving transformations. We propose a framework that learns how image fea-

tures change through a class of unknown transformations, where these changes are encoded as probabilistic transformation descriptors. In situations where the transformation class is very large, the statistics of these descriptors, when considered as a whole, are insufficient to determine if two images match each other based on the features extracted. Instead, a method is proposed that automatically partitions the unknown transformation class into subsets in which the statistics of these descriptors become more discriminative. The extended matching process thus involves a maximum-likelihood estimation of the unknown subset labels. The results demonstrate how it is possible to successfully match an image to one which has undergone substantial transformation without the need for registration.

The third transformation class involves the heavy distortion, correspondence breaking transformations. This class of transformations can be extended to include non-structural relations such as the "transformation" between the bottom half and the top half of a face. The co-occurrence statistics between features across a pair of images are learned from the training set comprising matched and mismatched image pairs – these are expressed in the form of a cross-feature ratio table. The proposed method does not require feature-to-feature correspondences, but instead identifies and exploits feature co-occurrences that are able to provide discriminative result from the transformation. Experimental results indicate that this matching process not only clearly outperforms the classical methods of SSD and histogram matching, but are able to match images under extreme transformations.

Acknowledgements

I would like to express my gratitude to my supervisor, Dr. Cham Tat Jen, whose expertise, patience and understanding aided much to me through the graduation period. I wish to express my sincerest appreciation to him.

My study has been carried out in Centre for Multimedia and Network Technology (CeMNet) and Center for Graphics and Image Technology (CGIT). The members in the two labs are the most precious resource and I benefit from discussions with Dr. Qian Kemao, Dr. Gao Deyun, Song Peng, Feng Zhou, Pong Hon Keat, Pham Minh Tri, Jin Tao, Li Li, Yu Xiaozhou, Zhang Xiaozheng, Zhou Zhi, Wang Huiqing, Ye Minghua, Guo Jing, Wang Jinjun, Wu Jianhua, Li Xiangjun, Zhang Yuquan, Zhou Chen, Hu Yiqun, Liu Song, Zhou Zongzhao, Zhang Yu and Zhou Zixuan.

I would appreciate administrative members at CeMNet and CGIT: Dr. Chia Liang Tien, Chua Poo Hua, Oh Hwee May, Alan and Lee Poh, for their help to the convenient access to the utilities in the labs; and special thanks to Dr. Leung Karhang for providing the PIE dataset. I would also thank Nanyang Technological University for the Ph.D scholarship.

Finally, I wish to express my deepest gratitude to my parents and to my wife Jun Ling for their support and encouragement, without them this thesis would not exist.

Contents

Chapter 1

Introduction

1.1 Motivation

1.1.1 Introduction of Image Matching

Image matching is a fundamental problem in computer vision, with various applications such as image mosaicing, content-based image retrieval, object recognition and object tracking. Image matching is also referred to as image correspondence, since the image matching problem can be thought of as establishing the correspondences between a set of reference images and a set of target images. Although computer vision has been studied for over 40 years, image matching still remains challenging because of the various transformations relating the two corresponding images. The transformations could be caused by viewing direction changes, illumination changes, scale changes, occlusion and deformation, which could make the corresponding image pairs look quite different even from

a human's point of view sometimes, for example, the inverted reflection image in wavy water compared with the original image.

Appearance-based object detection methods [65, 27], which have been increasingly popular and extensively studied in recent years, can partly solve the difficulty caused by various transformations in image matching. In these methods classifiers are trained to recognize specific categories of objects, but they require a training dataset which is not only available but generalizes well to observed images. The training data typically includes a set of positive images which represent the appearance of objects under different transformations in the object category, and a set of negative images which do not belong to that object category. Once the training process is completed, the classifiers may only be used to recognize objects of the same categories with the training images.

So such appearance-based object detection methods can only be used for image matching when there are sufficient *a priori* known reference exemplars of the images to form a training set, and the observed images must be of the same visual content of the reference exemplars. This is a big limitation for the use of appearance-based object detection methods in general image matching problems, because there are a lot of image matching problems that involve matching two images for which the image content does not belong to *a priori* known object class. One example of such a problem is image mosaicing, where the visual content may not only be unknown *a priori* but may also be wildly varying from one instance to the next; thus traditional appearance-based classifiers cannot be applied. Another related scenario is in the online tracking of objects for which there may not be sufficient (or even any) prior exemplars for the use of appearance-based classifiers. For example, this may occur if a visual tracking system needs

to be operated quickly after hardware set up, and for which the camera angles are not significantly constrained. Nevertheless, there will likely be a strong appearance similarity between consecutive video frames that should lends itself to be exploited.

As such, the matching of image pairs have traditionally been carried out using simple metrics such as sum-of-squared differences (SSD) or histogram matching, with the built of the research focused on robustly estimating warp parameters that bring image pairs into alignment. However, finding the corresponding points for estimating warp parameters is difficult and vulnerable to noise, since the correspondence is largely based on local information. In addition, correctly finding the corresponding points is almost impossible when the distortion is severe and do not preserve image structure, such as the transformation shown in "Problem 3" in figure 1.1, where the transformation is created by a combination of multiple effect filters in Adobe Photoshop.

1.1.2 Problem Formulation

In this thesis, we will address the image matching problem as learning and classifying correspondences rather than learning appearance classifiers. More precisely, we formulate the image matching problem as:

> Given a set of training pairs of images that implicitly captures the transformation (with both positive and negative classes), identify if a new pair of test images is matched via the transformation class. The test images can have *totally different visual content* as compared to the training data.

We further divide the general image matching problem into three sub-problems according to the transformation classes they cope with, and they are illustrated in figure 1.1. The three transformation classes are:

Low-distortion transformations, which include minor changes such as those between consecutive video frames, or those caused by slight expression change or view point change.

Heavy-distortion, correspondence-preserving transformations, which include transformations where point correspondences are preserved, such as scale change, rotation or projective transformations.

Heavy-distortion, correspondence-breaking transformations, which include severe, unmodeled transformations either in shape or color. Image structures may not be preserved in such transformations. For example, the severe transformation shown in "Problem 3" in figure 1.1, which is created using multiple effect filters in Adobe Photoshop. This class of transformations can be extended to include non-structural relations, *e.g.* the "transformation" between the bottom half and the top half of a face.

In this formulation, all the training data, as well as test images, are image pairs. Therefore, it is possible that only the relative visual content, rather than absolute visual content being considered, so the learned image matching classifier could be applied to images of quite different visual content.

The approaches dealing with the three sub-problems will be outlined in Section 1.2, and they are fully described in Chapters 3, 4 and 5.

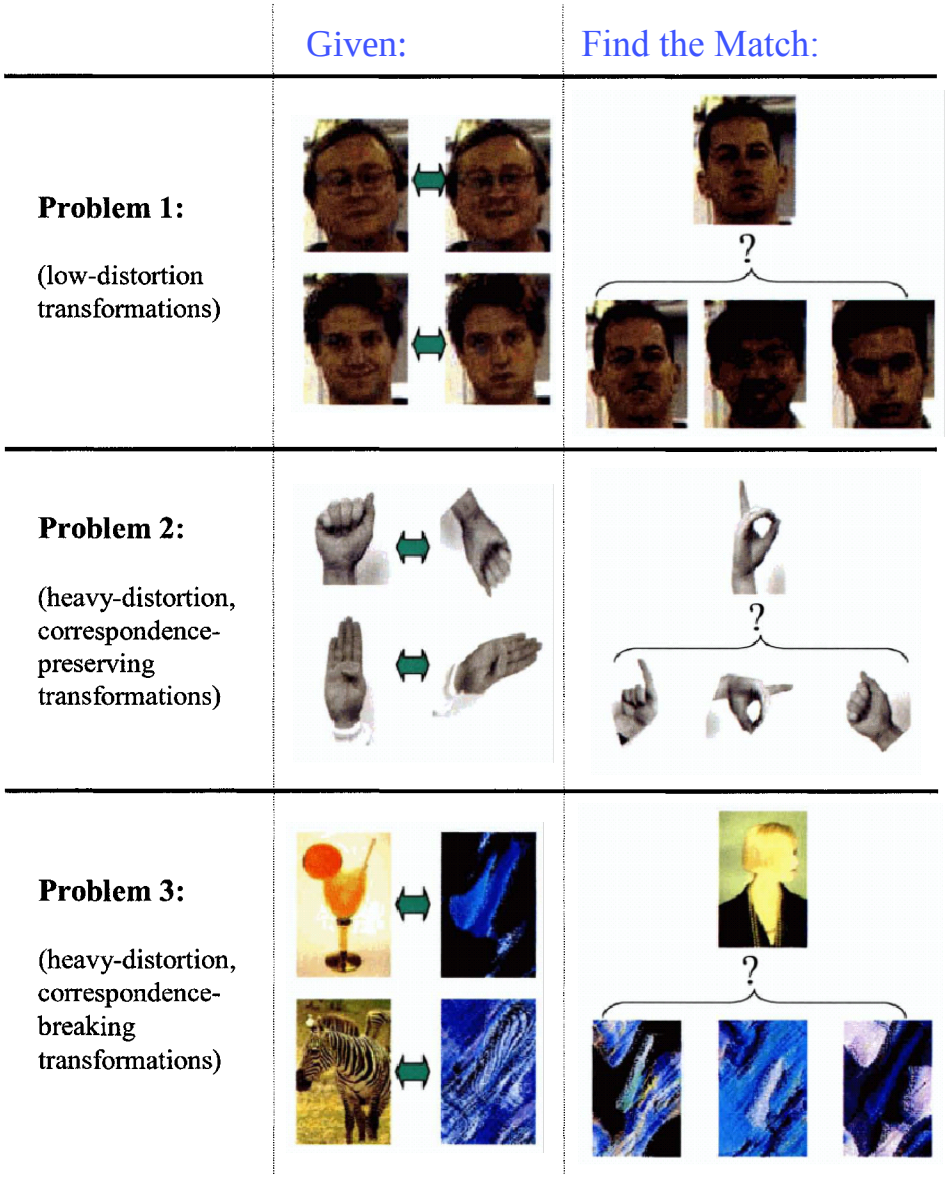


Figure 1.1: Given a set of training pairs of images that implicitly captures the transformation (with both positive and negative classes), identify if a new pair of test images is matched via the transformation class. The test images can have *totally different visual content* as compared to the training data. We divide the general image matching problem into three sub-problems according to the transformation classes they dealing with.

1.1.3 The Notion of Matching

Suppose we have an image A, which is to be compared to a second image B. Additionally, an arbitrarily complex transformation between A and B is allowed and denoted by $S_{\boldsymbol{\theta}}(\cdot)$, that is

$$B = S_{\boldsymbol{\theta}}(A)$$

where $\boldsymbol{\theta}$ is a vector consisting all the parameters of the transformation. The *match decision* problem of determining if A matches B may then be expressed as a binary classifier function $T(\cdot)$:

$$T(A, B) = T(A, S_{\boldsymbol{\theta}}(A)) \begin{cases} 1, & \text{if } A \text{ matches } B \\ -1, & \text{otherwise.} \end{cases} \tag{1.1}$$

An important point here is that the term "match" does *not* mean that A and B are identical. Matching here indicates that A and B are related with sufficient accuracy in the relevant transformation parameters $\boldsymbol{\theta}$ that the user is interested in. Images A and B may be considered to be "matched" despite large differences in transformation parameters that the user wants to ignore or be *invariant* to. This is illustrated in figure 1.2, where the transformation parameters are $[\theta_1, \cdots, \theta_m, \theta_{m+1}, \theta_N]$. $[\theta_1, \cdots, \theta_m]$ are the set of parameters to be recovered in matching, and their values determine whether the two images are matched or not. While $[\theta_{m+1}, \cdots, \theta_N]$ are the set of parameters to be ignored in matching. An example is that an image patch may be considered to be matched with a 90° rotated version if the user wants rotation invariance.

The notion of matching of two images is highly subject to human opinion, depending on the perception of individuals or the requirements of the problem domain. An example is the problem of matching human faces with expressions. If the problem is to identify the same person, then transformations caused by

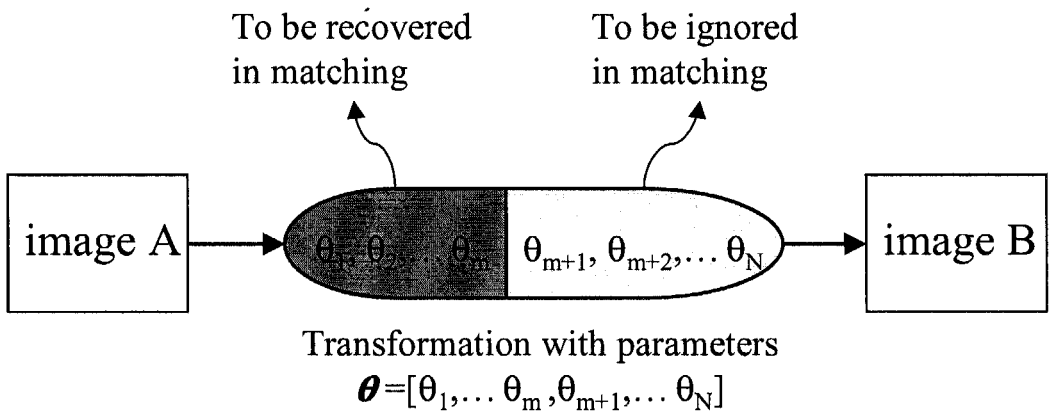


Figure 1.2: Image A is transformed to image B by the set of transformation parameters, some of them are to be recovered in matching, other are to be ignored.

expression changes should be ignored, while transformations caused by changes of identity should be recovered. In an alternative problem where the goal is to find similar expressions, then transformations caused by changes of identity should be ignored, while those caused by changes of expression should be recovered.

Thus any approach on image matching must therefore be context-specific. In our problem formulation, the "context" of the current matching problem is captured by the training image pairs.

1.2 Approaches to Solving Image Matching Problems

We propose three different approaches to deal with the three sub-problems, as is shown in figure 1.3. These approaches are discussed in sequence in the thesis. Firstly, we consider learning optimal distance measure for handling low-distortion

transformations, then an approach making use of feature correspondences for dealing with heavy-distortion, correspondence-preserving transformations is introduced, followed by another approach which uses feature co-occurrence to deal with heavy-distortion, correspondence-breaking transformations. They are described in more detail below.

Approaches:	Dealing with:	Discussed in:
Image matching via learning optimal distance measures	Problem 1 (low-distortion transformations)	*Chapter 3*
Image matching via feature correspondences	Problem 2 (heavy-distortion, correspondence-preserving transformations)	*Chapter 4*
Image matching via feature co-occurrence	Problem 3 (heavy-distortion, correspondence-breaking transformations)	*Chapter 5*

Figure 1.3: Three approaches are proposed to deal with the three sub-problems, and they are discussed sequentially in the thesis.

1.2.1 Learning Optimal Distance Measures

In Chapter 3, we discuss the image matching problem under low-distortion transformations. In the past, standard measures such as sum-of-squared pixel differences (SSD), chamfer distance [6], shuffle distance [62] and histogram measures including χ^2 distance, Bhattacharyya distance (as used in the mean-shift frame-

work [14]), *etc.* are often used to evaluate correspondences between images. These measures, being simple and intuitive, each have their advantage for different situations. However they are nevertheless *ad hoc* and may be sub-optimal in terms of accuracy.

Thus we propose an approach to learning a optimal distance measure in order to minimize the correspondence classification error on training data, which is based on optimizing the parameters of the distance measure. The learned distance measure may also apply to other images with very different visual content, because the learning process involves only relative, rather than absolute, visual content between image pairs.

1.2.2 Image Matching via Feature Correspondence

In Chapter 4, we deal with heavy-distortion, correspondence-preserving transformations for image matching. For a large group of transformations, the point-to-point correspondences can be preserved. These includes affine, projective and similarity transformations, which result from pure 3D camera rotations, or when viewing planar objects from different viewpoints.

We propose a novel framework for image matching under the correspondence-preserving group of transformations, which can learn how the features change under transformations, and the transformations are expressed in the form of Probabilistic Transformation Descriptors (PTD).

If the statistics of these descriptors are considered as a whole in situations where the transformation space is very large, they may not be sufficient to determine if two images match each other. Instead, a method is proposed which can automatically partition the unknown training transformation class into sub-classes

in which the statistics of these descriptors become more discriminative. The extended matching process thus involves a maximum-likelihood estimation of the unknown subset labels.

1.2.3 Image Matching via Feature Co-occurrence

In Chapter 5, we propose a novel method for the problem of matching a image to a similar that has been subjected to severe, correspondence-breaking transformations by using the co-occurrence statistics between features across a pair of images. The co-occurrence statistics are expressed in the form of a Cross-feature Ratio Table (CRT), which are learned from a training set comprising matched and mismatched image pairs. These statistics do not require feature-to-feature correspondences, but instead they enable identification and exploitation of co-occurrences that are able to provide discrimination for the transformation. This method is applicable to various kinds of transformations, including those destroying image structures and removing the possibility of establishing point-to-point correspondence.

1.3 Outline of Thesis

In Chapter 2, some basic concepts are reviewed, including image representation, image transformation and histogram distance measures. Chapter 3 discusses learning optimal distance measures for image correspondences, which is suited for small image distortions. Then Chapter 4 discusses image matching under correspondence-preserving transformations, where the Probabilistic Transformation Descriptors (PTD) is introduced. In Chapter 5, we will describe another

framework dealing with correspondence-breaking transformations based on Cross-feature Ratio Table (CRT). Chapter 6 concludes the thesis.

Chapter 2

Basic Concepts

This chapter provides a review of the basic concepts used in the thesis, as a bridge to readers unfamiliar with these topics. The basic concepts covered in this chapter include image representation, image transformations and histogram distance measures.

2.1 Image Representation

Image representation plays a substantial role in visual tasks such as image matching, object recognition and object detection, because it not only affects the efficiency of the underlying algorithm, but also sometimes being an inseparable part of the algorithm. Normally, images are represented as vectors or sets of features, which are described in detail below.

2.1.1 Vector Form

Most vision algorithms treat images as a 2D vector (matrix) of pixels, or 1D vector of pixels when the size of the image is known. The value of the pixels could be scalar for gray level images, or vector for color images. In vector form representation, the size and resolution of the input images have to be fixed in order for the comparison of two images. Figure 2.1 is an illustration of vector form representation of a gray-level image of size 4×4.

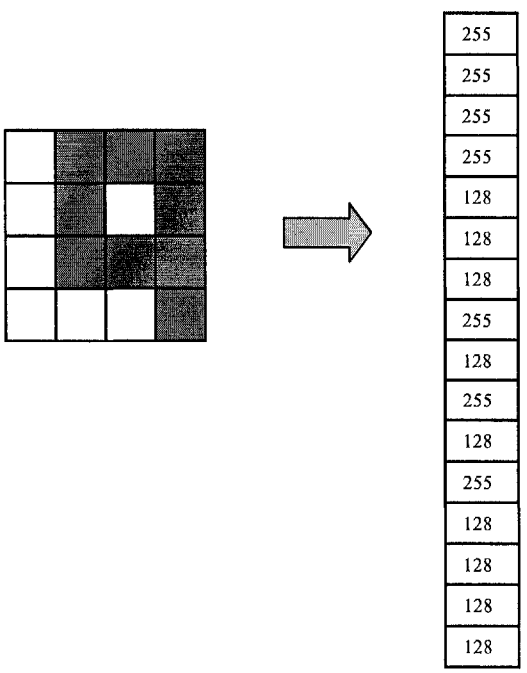


Figure 2.1: Illustration of the vector form representation of image. The image, as shown in the left, is a gray-level image of size 4×4. In the right is the 16 dimensional vector representation of the image, the values are the gray-levels of corresponding pixels. The ordering is top to bottom, left to right.

2.1.2 Set Form

The image representation used in the thesis is the set of features representation, or sometimes called in other name as distribution of features representation. The features may refer to descriptors or local image patches in different approaches. Figure 2.2 is an illustration of the set form representation of an image.

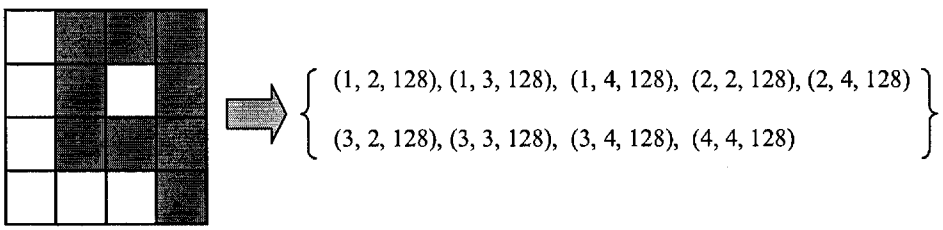


Figure 2.2: Illustration of the set form image representation. The image, as shown in the left, is a 4×4 gray-level image, with gray-level "255" (shown as white) indicating background . In the right, the image is represented by the set of foreground pixels, each pixel is represented by a 3 dimensional vector composed of the spacial position and the gray-level of that pixel.

Images represented as sets of features are recently highly popular in computer vision. One example is that an image may be represented as a set of SIFT [41] descriptors extracted at various key points, and each SIFT descriptor is a 128 dimensional vector. Other examples include representing an image as a set of detected oriented edge points, or a face image may be described as a set of patches representing different facial parts.

Sets of features representation can be expressed in the following three forms: raw feature set, signature and histogram. They are described sequentially below.

Raw Feature Set

Raw feature set in this thesis refers to a representation where the original features of an image is directly expressed in set form, as compared to reduced set-related representations such as signatures and histograms. Raw feature set representation is widely used in recent works on computer vision. For example, Jebara models images as bags of pixels or sets of vectors in [33], where for instance a gray-scale image is represented as a collection of (X, Y, I) pixel vectors. Similar image representations are also used in [67] and [37]. Grauman and Darrell represent images as sets of local features in [23], where the features are extracted from the interest points detected by Harris-Affine detector described in [42] or SIFT interest operator [41], and the low-dimensional gradient-based descriptor called PCA-SIFT [35] is used as the feature. Ullman in [63] represents images as collections of local image fragments of various sizes.

Suppose $F(x, y)$ is the feature extracted at position (x, y) in image I, which is a vector representing the local information at (x, y). The dimension of $F(x, y)$ may be different for different types of features. For instance, in the simplest case, the dimension of $F(x, y)$ is 1 if the feature is the pixel intensity, the dimension of $F(x, y)$ is 3 if the feature is the pixel color, or the dimension may be 128 if the feature is the SIFT descriptor as described in [41].

Although $F(x, y)$ is defined in the continuous domain, normally only samples located at a limited number of interest points are extracted to represent the image. We denote the set of K interest sample points as:

$$\mathcal{P} = \{(x_1, y_1), (x_2, y_2), ..., (x_k, y_k)\} \tag{2.1}$$

The image is then represented by the set of features vectors given by:

$$\mathcal{F} = \{\boldsymbol{f}_k \mid k = 1, \ldots, K\} \tag{2.2}$$

where $\boldsymbol{f}_k$ may defined as the following two forms:

- Definition (1)

$$\boldsymbol{f}_k = F(x_k, y_k) \tag{2.3}$$

- Definition (2)

$$\boldsymbol{f}_k = \left[\; x_k \; , \quad y_k \; , \quad F(x_k, y_k) \; \right]^T \tag{2.4}$$

In definition (1), $\boldsymbol{f}_k$ is simply the feature at the corresponding location. There is no exact information on location or spacial relationships of these features, so it might be possible that images with same set of features but in different space configurations be confused. However, this problem may be reduced by overlapping fragments at multiple scales so that the correct configuration is preferred, which is shown by Ullman *et al.* [63].

In definition (2), the spatial position of the feature is included with the feature value to form the final feature vector, which is a complete representation of the image with no loss in information. Definition (2) is used in chapter 3 and chapter 4. Two examples of feature vectors in the form of definition (2) are shown in figure 2.3, where the feature is pixel intensity for the left image, and directions of edge for the right image, positions of the features are concatenated to form the final feature vectors. The choice of the two definitions is based on the needs of the various methods.

Instead of considering $F(x, y)$ to be sampled with the same sampling function such that the resulting images may be treated as vectors in the same observation space, the set form representation makes it possible that the sampling functions are different. This is useful in a number of different situations, *e.g.*

- when images are at different resolutions, or

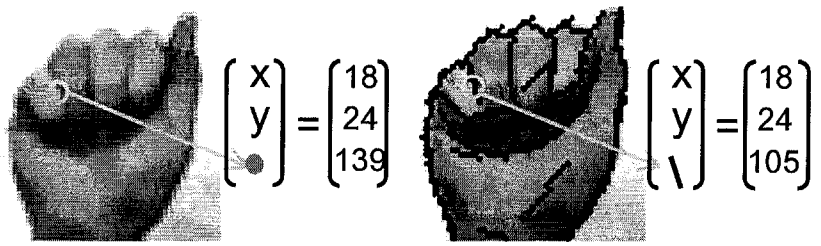


Figure 2.3: Images represented by distribution of feature vectors of definition(2), where position information is attached with the original feature. Left: image features are pixel intensities. Right: image features are directions of oriented edges.

- if the images are preprocessed by a segmentation algorithm giving rise to regions with different number of pixels, or
- if geometric features are used such as corners and edges (as we do later), or
- in a real-time algorithm where not all pixels can be processed and need to be randomly sampled.

Sometimes, the number of features in the feature set may be too large to be efficiently processed, and it may be advantageous to convert the raw feature set representation to compressed forms: signatures and histograms.

Histograms

Raw feature set representation may be easily converted to histogram through binning. Histogram representation applies to the situation when the proportion of different features provide much more explicit information about the image than a list of features. This is particularly useful when exact placement of features is less relevant. The feature space is partitioned into N bins with indices from 1 to N,

and the histogram is a N dimensional vector with real values denoting the number of features located in each bin. An illustration of histogram binning is shown in figure 2.4, where there are 9 bins indexed by 1 to 9. Histogram representation is widely used in computer vision, for example, Fei-Fei and Perona [39] represent the image of a scene by a histogram of local regions denoted as codewords obtained by unsupervised learning. Agrawal and Triggs [2] represent human body as a histogram of shape contexts [7]. Hadjidemetriou *et al.* [25] use multidimensional histograms of image intensities for object recognition.

The histogram form of feature representation can be expressed as:

$$\mathcal{F}_{hist} = [h_1, h_2, \cdots, h_N] \tag{2.5}$$

where h_i is the number of features in the ith bin.

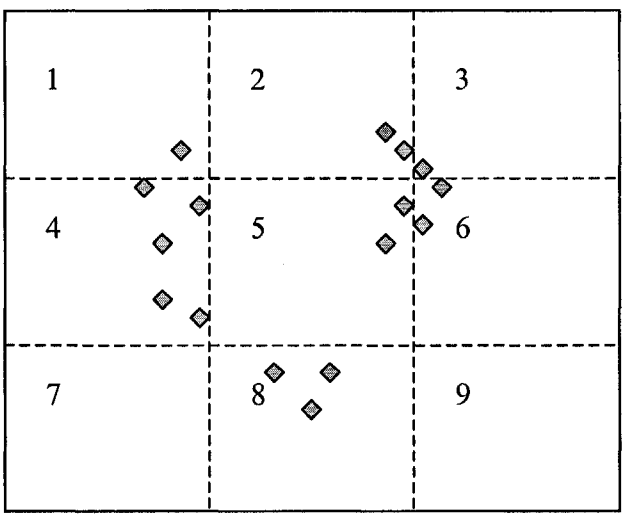


Figure 2.4: An illustration of histogram binning of a feature distribution. There are 9 bins separated by the doted lines, indexed from 1 to 9.

Histogram representation is used in both Chapter 4 and Chapter 5.

Signatures

In signatures, dominant clusters are extracted from the original distribution by a clustering algorithm such as vector quantization or k-means algorithm. A signature is a set of the main clusters of the original distribution, each represented as the central point of the cluster, together with a weight representing the size of the cluster, as illustrated in figure 2.5.

A signature with N clusters may be expressed as:

$$\mathcal{F}_{sig} = \{(\boldsymbol{f_n}, w_n) | \; n = 1, \cdots, N\} \tag{2.6}$$

where $\boldsymbol{f_n}$ and w_n are the center and weight of the nth cluster respectively. Signature representation is used by Rubner *et al.* [54] for measuring Earth Mover's Distance for image retrieval. In [53], Rubner and Tomasi use signatures to represent the full distribution of 24-dimensional texture features produced by an image. Lazebnik *et al.* [38] also use signatures to represent texture images by clustering the affine-invariant descriptors extracted from each image. The advantage of signatures is that they avoid the quantization and binning problems associated with histograms, especially in high dimensions. In addition, signatures are more efficient than histograms in terms of storage, especially when the histogram is finely quantized and there are a large population of zero bins.

2.2 Standard Image Transformation Models

The concept of image transformation in this thesis is quite general, which could be used to express any change between two images. There are a large number of standard image transformation models to express simple and well understood transformations, such as geometric and gray-scale transformations. However, these stan-

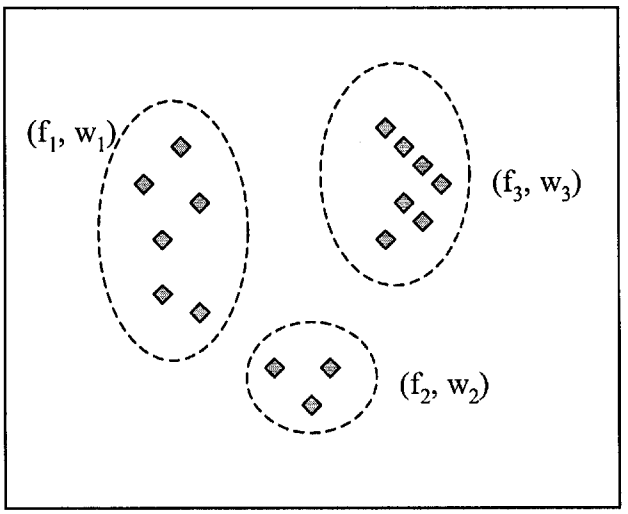


Figure 2.5: An illustration of a feature distribution represented as a signature. The signature is composed of the 3 clusters shown by the doted ellipses.

dard models are far from sufficient for modeling image changes in reality, such as those caused by non-rigid deformation and self-occlusion in cloth, 3D viewing point change in complex scenes, unknown illumination changes, *etc.* Nevertheless, the image matching approaches proposed in this thesis can cope with not only the standard image transformations, but also more complex image transformations. In this section, an introduction of some of the standard transformation models is provided, and in next section, the issues of real-world image transformations are discussed.

Generally, standard image transformation models can be divided into the following categories: 2D geometric transformations, gray-scale transformations, filters and special effects, and viewing point change of 3D rigid objects. They are described in the following.

2.2.1 2D Geometric Transformations

Generally, a geometric transformation can be defined as:

$$g(x', y') = g(a(x, y), b(x, y)) = f(x, y) \tag{2.7}$$

where $f(x, y)$ and $g(x, y)$ are the input and output images respectively. The functions $a(x, y)$ and $b(x, y)$ specify the spatial relationship between the two images. In this transformation, a point at position (x, y) in the input image is moved to (x', y') in the output image. Usually the input image is only sampled at integer positions, and we want to recover an output image also sampled at integer positions. But integer positions (x, y) in the input images will map to fractional (non-integer) positions in the output images, so gray-level interpolation is required to produce the output image which is only sampled at integer positions. The interpolation methods could be nearest neighbor or bilinear interpolation, but they are not discussed in this thesis, the details of which can be found in [10].

There are various kinds of widely used geometric transformations, they are introduced below.

Euclidean Transformations

Euclidean transformations are the most common transformation which include translation and rotation. The invariants in Euclidean transformation includes length, angles, ratios, parallelism, incidence and cross ratio. There are 3 Degrees of freedom(DOF) in Euclidean transformations.

For translation, we can express it in the 2D image plane as:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \tag{2.8}$$

where x_0 and y_0 are the translation displacements in x and y direction.

It's useful to use *homogeneous coordinates* [20, 44, 10] to represent geometric transformations, where the x–y plane is considered to be the $z = 1$ plane of a three-dimensional x, y, z space. Equation (2.8) can be expressed in homogeneous coordinates as:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{2.9}$$

The rotations can be expressed as:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{2.10}$$

where θ is the rotated angle about the image plane origin.

Generally, Euclidean transformations can be expressed in homogeneous coordinates as:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & t_1 \\ \sin\theta & \cos\theta & t_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{2.11}$$

Similarity Transformations

Similarity transformation is a superset of Euclidean transformation — it includes translation, rotation and uniform scaling. The invariants in similarity transformation includes angles, ratios, parallelism, incidence and cross ratio. There are 4 DOF in similarity transformations.

Similarity transformation can be expressed in homogeneous coordinates as:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & -b & t_1 \\ b & a & t_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{2.12}$$

Affine Transformations

Affine transformation is a superset of similarity transformation, it includes translation, rotation, scaling and shear. The invariants in affine transformation includes parallelism, incidence and cross ratio. There are 6 DOF in affine transformations. Orthographic projection can be represented by affine transformations.

Affine transformation can be expressed in homogeneous coordinates as:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{2.13}$$

Projective Transformations

Projective transformation is a superset of affine transformation, it includes translation, rotation, scaling, shear and perspective projection. The invariants in projective transformation includes incidence and cross ratio. There is 8 DOF for projective transformations.

Projective transformation can be expressed in homogeneous coordinates as:

$$\lambda \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \equiv \mathbf{T} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{2.14}$$

where $\mathbf{T}$ is non-singular $(det(\mathbf{T}) \neq 0)$.

The relationship between the above 4 geometric transformations are:

$$projective \supset affine \supset similarity \supset Euclidean$$

and examples of them are shown in figure 2.6.

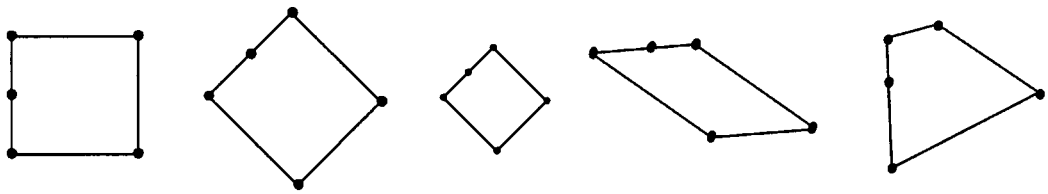


Figure 2.6: Examples of geometric transformations.

Polynomial Transformations

Polynomial transformations are one of the most widely used plane transformations, because they are simple to express but can show relatively complex distortions. They are expressed in the form of one pair of equations, a polynomial transformation of second order can be expressed as:

$$\begin{aligned} x' &= a_{20}x^2 + a_{02}y^2 + a_{11}xy + a_{10}x + a_{01}y + a_{00} \\ y' &= b_{20}x^2 + b_{02}y^2 + b_{11}xy + b_{10}x + b_{01}y + b_{00} \end{aligned} \tag{2.15}$$

Two examples of second order polynomial transformations are shown in figure 2.7.

Each of the above introduced geometric transformations can be uniquely determined by a specific number of point pairs, as shown in table 2.1.

original output 1 output 2

Figure 2.7: Examples of second order polynomial transformations.Left: original image, middle and right: two transformed images.

Euclidean	Similarity	Affine	Projective	Poly.(2nd)	Poly.(3rd)
2	2	3	4	6	10

Table 2.1: No. of corresponding point pairs needed to determine the transformations

2.2.2 Gray-Scale Transformations

Gray-scale transformation changes an image's gray-level in such a way that each output pixel's gray level depends only upon the corresponding input pixel's gray level. This contrasts with *filtering,* which will be introduced in the next section, where a neighborhood of input pixels determines the gray level of each output pixel. Gray-scale transformation can be expressed as:

$$B(x, y) = f(A(x, y)) \tag{2.16}$$

where $A(x, y)$ and $B(x, y)$ are the input and output images, and $f(\cdot)$ is the *gray-scale transformation function*, which specifies how to map input gray level to output gray level. Obviously the spacial relationships are not changed by gray-scale transformation. Readers can refer to [10] for more details on gray-scale transformations, where it is called point operations.

Gray-scale transformation are widely used in image processing to change the appearance of an image, in the following we introduce three applications of it.

Brightness Change

Brightness changes make the image appear lighter or darker. The simplest way to achieve this is through a linear gray-scale transformation function of (2.16) in the form below:

$$f(G_A) = \alpha G_A + \beta \tag{2.17}$$

where G_A is the gray level of the input pixel. If $\alpha > 1$ and $\beta > 0$, then the output image will appear lighter; if $0 < \alpha < 1$ and $\beta < 0$, then the output image will appear darker. Equation (2.17) also changes image contrast, when $\alpha > 1$, the contrast is increased, and when $0 < \alpha < 1$, the contrast is reduced. Other combinations of α and β, or nonlinear gray-scale transformation functions can also be used to adjust brightness of images.

Contrast Stretching

In some cases, the features of interest in an image only occupy a relatively narrow range of gray level. Contrast stretching can be used to expand the range so that the contrast of interested features may be improved. Contrast stretching can also be used to map the lightest and darkest pixels in the image to white and black, so that highlights appear lighter and shadows appear darker. The simplest way to do contrast stretching is by a linear gray-scale transformation function which can be expressed as:

$$f(G_A) = (G_A - c)\frac{255}{d - c} \tag{2.18}$$

where c and d are the lower and upper bound of interested gray levels, and the whole range of gray levels is assumed to be between 0 and 255.

Histogram Processing

The histogram of a digital image with gray levels $0 - L$ is a function $h(i) = n_i$, where i indicates the gray level, and n_i is the number of pixels with gray level i. The shape of the histogram provides useful information on how to enhance the appearance of the image. For example, if the pixel levels are largely distributed at one end of gray level, a corresponding gray-scale transformation function could be applied to flatten the histogram, increasing the contrast and dynamic range of the image. Unlike contrast stretching, histogram processing employs non-linear gray-scale transformation function to map between pixel gray levels in the input and output images.

Histogram processing includes two applications, which are histogram equalization and histogram matching (histogram specification). Histogram equalization improves contrast and dynamic range through the goal of obtaining a uniform histogram. The result of histogram equalization is similar to contrast stretching, but it offers the advantage of full automation, since histogram equalization automatically determines a gray-scale transformation function to produce a new image with a uniform histogram. The resultant "uniform" histogram is only true in the approximate sense due to the discrete nature of digital images. In reality, the resultant histogram will usually take a rather ragged appearance due to the finite number of available gray levels, so often good quality images may be degraded by histogram equalization.

Histogram equalization seeks to produce an output image that has a uniform

histogram; however sometimes we need to be able to specify the shape of the histogram that we wish the processed image to have, in order to highlight certain gray level ranges. This is possible via histogram matching (histogram specification). The detailed procedures of histogram equalization and histogram matching are rather sophisticated, which can be found in [10, 22, 9, 32].

2.2.3 Filters and Special Effects

In image filtering, the output pixel's gray level (or color) may depend upon a neighborhood of the input pixel's gray level (or color) or even the whole input image, so the filtering algorithms as well as the filtering effects can be much more complex. There are various kinds of filters, some can be applied for specific tasks such as noise reduction or sharpening, others are designed for artistic effects. In the following, we introduce a few categories of image filters according to what they do. Most of the filters have their correspondences in Adobe Photoshop 7 under the *Filter* menu.

Blurring

Image blurring is used for noise reduction, image smoothing or producing special effects such as motion blur. In general, linear filtering of an image A with a filter kernel K of size $m \times n$ can be carried out by convolution, which is expressed as:

$$B(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} K(s,t)A(x+s, y+t) \tag{2.19}$$

where $a = (m-1)/2$ and $b = (n-1)/2$. The kernel K is suitably chosen so that it is a low-pass filter for the purpose of blurring; such kernels include the widely used Gaussian kernel, or box kernel. Nonlinear filters, which are not implemented

by convolution, may also be applied for blurring. For example, the well known median filter is particularly effective for removing impulse noise (salt and pepper noise).

Sharpening

Sharpening is the opposite of blur, the objective of which is to highlight the fine details or abrupt changes in an image. Similar to blurring, sharpening can also be implemented by linear filters expressed in (2.19), but with high-pass kernels such as Laplacian, Sobel, Roberts, Laplacian of Gaussian (LoG) and Difference of Gaussian (DoG) kernels. The disadvantage of high-pass filters is that they tend to enhance high-frequency noise along with the image details of interest.

Spatial Distortions

Distortion filters distort images in various ways. Distortion filters are designed for the purpose of artistic effects, as the word "filter" usually implies, rather than for the purpose of image calibration or correspondence. A simple example of distortion filter is the twirling filter, which will twist an image around the center. One implementation of twirling filter could be expressed in the following way as proposed in [29]:

$$B(r, a) = A(r, a + r/3) \tag{2.20}$$

where $A(r, a)$ and $B(r, a)$ are the input and output images expressed in polar coordinates. The angle a of the output image is incremented with a third of the radius r compared with the input image, thus making the pixels swirl around the center of the image.

Another example of distortion filter is the ripple filter, which ripples an image

horizontally or vertically, and the wavelength and amplitude could be changed, the ripple shape can also be selected from sine, triangle or square. Still another example is the water filter which produces a water ripple effect on an image. See figure 2.8 for real image examples.

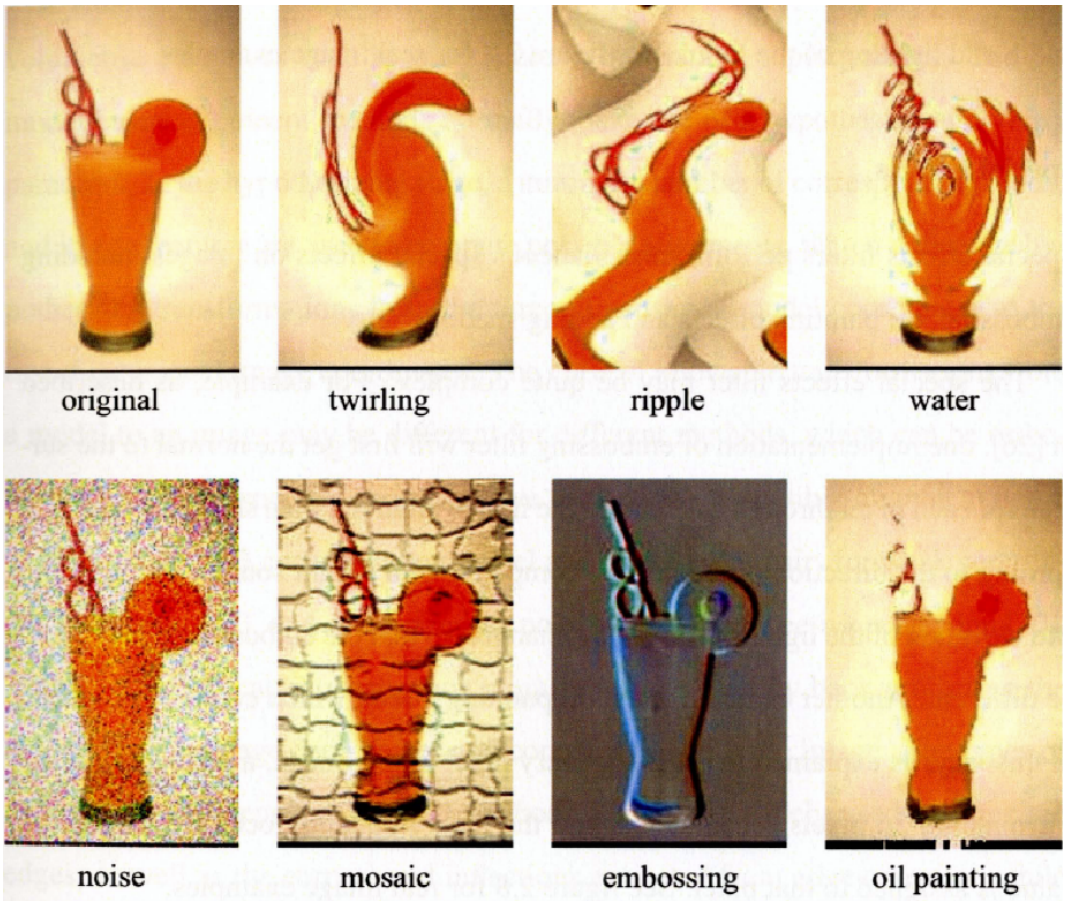


Figure 2.8: Examples of filters and special effects. The filter names are below their corresponding output images. The image in the upper left is the original image.

Textures

Texture filters generally produce some kind of abstract texture or pattern on the input image. One example is the noise filter, which adds random noise to an image, you can specify the amount of the noise, the distribution of the noise: uniform or Gaussian, and whether the noise is monochrome or colored. Another example is the mosaic filter, where you can specify the size of the tile and the width and lighting of the grout. See figure 2.8 for real image examples.

Special Effects

Special effects filters perform miscellaneous special effects on images including embossing, oil painting effects and adding shadows, *etc.*

The special effects filter may be quite complex. For example, as described in [26], one implementation of embossing filter will first get the normal to the surface for each pixel through calculating the image gradient, then shade the pixel according to the direction of the normal compared with a light source. The azimuth and elevation of the light source can be changed so that the embossing effect could be different. Another example is the oil painting effect, which could be produced in this way, as explained in [29]: for every pixel in the image, a histogram of the surrounding 36 pixels is calculated, and the most frequently occurred brightness value is assigned to that pixel. See figure 2.8 for real image examples.

2.2.4 Viewing Point Changes for 3D Rigid Objects

In all the above introduced image transformations, the output image is produced by some operations on the input image, thus when the transformation is fixed, the output image is uniquely determined by the appearance of the input image.

However for image transformation caused by viewing point changes for 3D rigid objects, the output image is not determined by the appearance of the input image, but the contents in the image.

Transformations caused by viewing point changes for 3D rigid objects are usually dealt with by methods that align 3D models to objects in images [40, 31]. Most of the 3D models are represented by polyhedral, polygonal meshes or volumetric primitives such as generalized cylinders and superquadrics. The 3D model based alignment methods normally make use of a hypothesize-and-verify paradigm. In the hypothesizing stage, a minimum number of corresponding model and image features are used to compute possible alignments, which define the hypothesized transformations from the three-dimensional model coordinates to the two-dimensional image coordinates. The class of allowable transformations from a model to an image may be different for different methods, which can be orthographic, weak perspective or perspective projection. A possible alignment could be determined by 3 corresponding model and image point pairs for weak perspective projection [31], or 4 corresponding point pairs for perspective projection [30]. Then in the verification stage, each alignment is verified by back-projecting the model into the image coordinates and comparing it with the image. The types of features are different for different methods. The most widely used features are edges, as well as the corners and inflections extracted from edge contours, while other features such gray value gradients [36] and image intensities [66, 50] are also used. In some works, invariant combinations of features rather than individual features are utilized for the alignment [19, 64].

Another way to handle viewing point changes of 3D rigid objects can be explained in the framework of aspect graphs [16, 49]. An aspect graph partitions

the view sphere into regions (aspects) where the same set of features are visible inside each region and the image structure is stable. Each aspect, or general view, is represented by a node in the aspect graph; whereas the edges of the aspect graph represent transitions between two neighboring general views, namely, the accidental views. A transition, or visual event, represents a singularity of the visual mapping, which is usually caused by the emergence or occlusion of some object parts. 3D object recognition based on aspect graphs is then carried out by comparing each aspect with the image of the object.

2.3 Image Transformations in Reality

While the previous sections indicate there are a large number of standard image transformation models, they do not capture the range of transformations that occur in reality.

Image transformations that occur in reality are much more complex then the standard models, as shown by the examples in figure 2.9. In figure 2.9 (a), the image transformation is caused by expression change, and exact modeling of the change requires a complex physical model of the human face. In figure 2.9 (b), the frames are extracted from a human walking sequence, where the posture of the person and the background are continuously changing. Thus in order to model the image transformation, not only would an articulated human body model be required together with clothing models, but background and lighting models need to be known as well. The third example is the 3D viewing point change of a complex scene, as shown in figure 2.9 (c). The exact 3D model of the scene must be known to deal with occlusions, but the model can be very hard to acquire due to the complexity of the surface (in this case the cluster of leaves).

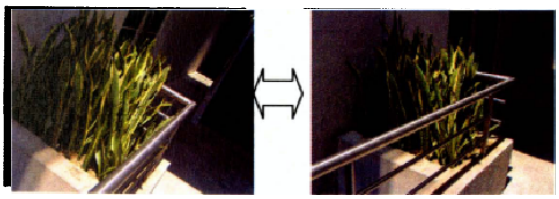
Image transformations in reality	Difficulties in modeling
 (a)	Expression model +lighting source
 (b)	Articulated figure model +varying background +varying lighting condition
 (c)	Complex and flexible 3D model of leaves and building +lighting source

Figure 2.9: The complexity of real world image transformations comes from the difficulty in modeling objects, 3D scenes, occlusions, and lighting.

The complexity of real world image transformations comes from the difficulty in modeling objects, 3D scenes, occlusions, and lighting. Thus exact modeling of such image transformations may require numerous parameters, and accurate estimation of the parameters may be difficult or even impossible through a limited number of images. Hence algorithms based on estimating complete transformation parameters are hard to extend to real situations. In contrast, the image matching approaches proposed in this thesis do not require an exact model of the image transformation.

2.4 Histogram Distance Measures

Histogram distance measures are used widely in the thesis, and a brief introduction of some well known histogram distance measures is given below. In this section, a histogram is represented by a vector $\boldsymbol{h}$ with components $h(i), i = 1 - N$. The histogram is normalized so that it is also suitable to represent a probability distribution:

$$\sum_{i=1}^{N} h(i) = 1 \tag{2.21}$$

Minkowski-Form Distance

Minkowski-form distance is defined based on the L^p norm:

$$d_{L_p}(\boldsymbol{h}, \boldsymbol{k}) = \left[\sum_{i=1}^{N} |h(i) - k(i)|^p \right]^{1/p} \tag{2.22}$$

where $\boldsymbol{h}$ and $\boldsymbol{k}$ are the two normalized histograms and $1 \leqslant p \leqslant \infty$. When $p = 1$, it is the L_1 norm, also called as city block or Manhattan distance; when $p = 2$, it is the L_2 norm, or Euclidean distance; when $p = \infty$, the L_∞ norm measures the maximal difference.

Chi-Square Distance

Chi-square distance is the most commonly used discrete measure to compare one dataset to a known distribution, or compare two unknown distributions. Suppose $\boldsymbol{h}$ is the observed distribution, and $\hat{\boldsymbol{h}}$ is the expected distribution, then the chi-square distance is given by:

$$d_{\chi^2}(\boldsymbol{h}, \hat{\boldsymbol{h}}) = \sum_{i=1}^{N} \frac{\left[h(i) - \hat{h}(i)\right]^2}{\hat{h}(i)} \tag{2.23}$$

which measures the likeliness of one distribution being drawn from another one.

Similarly, for the case of comparing two unknown distributions h and k, chi-square distance is expressed as:

$$d_{\chi^2}(\boldsymbol{h}, \boldsymbol{k}) = \sum_{i=1}^{N} \frac{(h(i) - m(i))^2}{m(i)} \tag{2.24}$$

where $m(i) = [h(i) + k(i)]/2$ is the mean of $h(i)$ and $k(i)$. The main disadvantage of chi-square distance is that singularities will arise whenever empty bins are compared. Moreover, it yields inaccurate results with low number of observations.

Bhattacharyya Distance

Bhattacharyya distance is defined as:

$$d_{Batt}(\boldsymbol{h}, \boldsymbol{k}) = \left[1 - \sum_{i=1}^{N} \sqrt{h(i) * k(i)}\right]^{1/2} \tag{2.25}$$

which has a straightforward geometric interpretation, because the term:

$$\sum_{i=1}^{N} \sqrt{h(i) * k(i)} \tag{2.26}$$

is the cosine of the angle between the unit vectors $[\sqrt{h(1)}, \sqrt{h(2)}, \cdots, \sqrt{h(N)}]^T$ and $[\sqrt{k(1)}, \sqrt{k(2)}, \cdots, \sqrt{k(N)}]^T$. Besides the geometric interpretation, Bhattacharyya distance also imposes a metric structure [15], and avoids the singularity

problem of the chi-square distance when comparing empty histogram bins [3]. More properties of Bhattacharyya distance can be found in [15, 34, 3].

Histogram Intersection Distance

Histogram intersection [61] distance is given by:

$$d(\boldsymbol{h}, \boldsymbol{k}) = 1 - \frac{\sum_{i=1}^{N} \min(h(i) - k(i))}{\sum_{i=1}^{N} h(i)} \tag{2.27}$$

The advantage of histogram intersection is that it can handle partial matches where the sizes of the two histograms (the sum over all bins) are different. When the sizes of the two histograms are the same, histogram intersection is equal to the L_1 norm, as shown in [61].

Kullback-Leibler Divergence and Jeffrey Divergence

Kullback-Leibler distance is perhaps the most frequently used information-theoretic distance measure, because it measures how inefficient on average it would be to encode one histogram using the other as the code-book . It is also called as relative entropy in information theory, and is closely related to cross entropy, information divergence and information for discrimination [18]. It is defined as:

$$d_{KL}(\boldsymbol{h}, \boldsymbol{k}) = \sum_{i=1}^{N} h(i) \log \frac{h(i)}{k(i)} \tag{2.28}$$

The disadvantage of K-L divergence is that it is non-symmetric and sensitive to histogram binning. The Jeffrey-divergence is the symmetric version of KL distance, which is numerically stable and robust with respect to noise and the size of histogram bins [52]. It is defined as:

$$d_{JD}(\boldsymbol{h}, \boldsymbol{k}) = \sum_{i=1}^{N} \left[h(i) \log \frac{h(i)}{m(i)} + k(i) \log \frac{k(i)}{m(i)} \right] \tag{2.29}$$

where $m(i) = [h(i) + k(i)]/2$.

Quadric-Form Distance

All the above introduced histogram distance measures account only for corresponding bins with the name index, and do not use information across bins. However, quadric-form distance can make use of cross-bin information, which was suggested in [46] for color based retrieval:

$$d_Q(\boldsymbol{h}, \boldsymbol{k}) = \sqrt{(\boldsymbol{h} - \boldsymbol{k})^T \boldsymbol{A} (\boldsymbol{h} - \boldsymbol{k})} \tag{2.30}$$

where $\boldsymbol{A}$ is a $N \times N$ similarity matrix with elements a_{ij} denoting the similarity between bins i and j. In the quadric-form distance measure, one bin in the first histogram can simultaneously correspond to multiple different bins in the second histogram. The quadratic distance provides better results than bin-by-bin only comparisons at the price of high computational costs. However, it could possibly lead to false positives as shown in [54, 60].

Earth Mover's Distance (EMD)

The Earth Mover's Distance (EMD) computes the distance between two distributions in some feature space, where the ground distance is given, which defines the distance between the basic features that are aggregated into the distributions. The EMD is then defined as the minimum amount of work needed to transfer one distribution to another one, and the work is based on the ground distance. Intuitively, given the two distributions, we can consider one as a mass of earth spread in space, and the other as a collection of holes in the same space. Then EMD computes the minimum amount of work needed to fill the holes with earth, and one unit of work corresponds to moving one unit of earth by a unit of ground distance. In computing the EMD, the distributions are represented by signatures, for which histograms can be considered as special cases.

The computation of EMD comes from a well-known *transportation problem*, for which efficient algorithms are available, such as the transportation-simplex method and interior-point algorithms. The advantage of EMD is that it works on distributions with a different number of bins, and it allows for partial matches in a very natural way. Moreover, when the ground distance conforms with perceptual similarity, the EMD matches perceptual similarity better than other measures, as shown in [54] for color- and texture-based image retrieval. The details of EMD can be found in [54].

Chapter 3

Learning Feature Distance Measures for Image Correspondences

This chapter deals with the image matching problem under low-distortion transformations, which is illustrated in figure 3.1. We propose a novel parametric distance measure, which can be optimized in order to minimize the correspondence classification error on training data. The learned distance measure may also apply to other images with very different visual content, because the learning process involves only relative, rather than absolute visual content between image pairs.

3.1 Context-specific Distance Measures

In the past, standard measures for evaluating correspondences are the sum-of-squared pixel differences (SSD), standard metrics in feature space (*e.g.* Euclidean is used in the SIFT framework [41]), the chamfer distance [6], the shuffle dis-

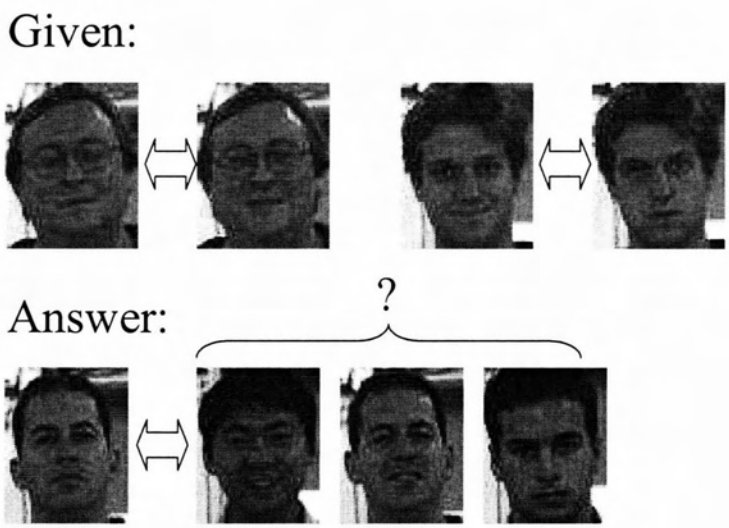


Figure 3.1: Classifying correspondences with different visual content. From a training set of paired images, the goal is to learn a match classifier that works on images with different visual content.

tance [62] and histogram measures such as χ^2 distance, Bhattacharyya distance (as used in the mean-shift framework [14]), Kullback-Leibler divergence, Earth Mover's Distance [54]. These measures, each has different properties, are used in different situations according to their performance. However, their performance may be sub-optimal in terms of accuracy because different features have different importance to distinguish between images in different applications, A solution is some kind of feature selection/aggregation, which can be achieved by optimizing the parameters of a generalized distance measure from a set of training images. This is the approach taken in this chapter.

The notion of similarity between two images or features is highly subject to *human* opinion, depending on the perception of individuals or the requirements of the domain. For example, in various problems that involve tracking an object, the notion of "similar" can range from similarity in the color histograms of the

tracked patches, to similarity of gray levels between exact corresponding pairs of pixels. There are no universally true quantitive distance measures of dissimilarity, except that all such distance measures should be zero when the images or features are absolutely identical.

Any proposed distance measures must therefore be context-specific and dependent on the spatial transformation model complexity. An ideal distance measure should be sensitive to the transformation parameters of interest and to the error tolerance on these estimates, but be fully invariant to those that the user wants to ignore. However, most vision algorithms in matching converge on standard measures such as the sum-of-squared pixel differences (SSD), chamfer distance, shuffle distance and histogram distances, which may not meet the proper contextual requirements for a suitable measure.

In this chapter, a framework for learning optimal distance measure will be described. In this framework, a parametric distance measure can be optimized in order to minimize the correspondence classification error on training data.

3.2 Match Decision by Distance Measure

It may be argued that having a binary match classifier is sufficient for the purpose of image matching. However, a real-valued distance measure is still very useful in many circumstances. A distance measure allows the ranking of different match hypotheses. Additionally, such a measure will also enable directed (*e.g.* gradient-based) searches that are based on minimizing this distance measure.

Suppose we have an image A, which is to be compared to a second image B, subject to a transformation $S_{\boldsymbol{\theta}}(\cdot)$ with parameters $\boldsymbol{\theta}$. The exact representation for A and B will be described in Section 3.3.1. Let the transformed version of B be

expressed as B' given by

$$B' = S_{\boldsymbol{\theta}}(B)$$

The *match decision* problem of determining if A matches B' may then be expressed as a binary classifier function $T_{\boldsymbol{\phi},t}(\cdot)$:

$$\begin{aligned} T_{\boldsymbol{\phi},t}(A, B, \boldsymbol{\theta}) &= T'_t\left(D_{\boldsymbol{\phi}}\left(A, S_{\boldsymbol{\theta}}(B)\right)\right) \\ &= \begin{cases} 1, & \text{if } D_{\boldsymbol{\phi}}(A, B') \leq t \\ -1, & \text{otherwise} \end{cases} \end{aligned} \tag{3.1}$$

where $D_{\boldsymbol{\phi}}(A, B')$ is a positive real-valued distance function that increases with a conceptual dissimilarity between images A and B', and is parameterized by $\boldsymbol{\phi}$; while $T'_t(\cdot)$ is the threshold function on the distance measure, with threshold t. The exact form of $D_{\boldsymbol{\phi}}(\cdot)$ will be described in Section 3.3.2.

An important point here is that the term "match" does *not* mean that A and B' are identical. Matching here indicates that A and B are related with sufficient accuracy in the relevant transformation parameters $\boldsymbol{\theta}$ that the user is interested in. A and B' may be considered to be "matched" despite large differences in transformation parameters that the user wants to ignore or be *invariant* to. An example is that an image patch may be considered to be matched with a 90° rotated version if the user wants rotation invariance.

3.2.1 Learning from Labeled Correspondences

As stated previously, the goal in this chapter is to learn a suitable feature distance measure from labeled training data that capture the contextual requirements for matching. In our context, this means learning the function $D_{\boldsymbol{\phi}}(\cdot)$ by discovering the optimal parameters $\boldsymbol{\phi}$.

Suppose that training data of correspondences is available comprising instances that are tuples (A, B, θ, l) where l is a label. There are two approaches to learning a distance measure from labeled correspondences:

- *Direct approach.* Here we can attempt to directly learn the distance measure $D_\phi(A, B')$ if the labels l are *real positive values* denoting the distance between A and B'. In this approach, ϕ may be optimized such that

$$\phi = \arg\min_{\phi} \left\{ \sum_{\text{training set}} (D_\phi(A, B') - l)^2 \right\} \tag{3.2}$$

  for all instances. However, this approach requires either that ground truth transformation parameters are available per training image to generate the distance labels (not always readily available), or that users manually specify a subjective distance label (noisy and error-prone).

- *Indirect Approach.* This approach involves indirectly learning a suitable distance measure through learning the match classifier function. This approach only requires that the labels l are binary class labels:

$$l_{A,B'} = \begin{cases} 1 & A \text{ and } B' \text{ match with sufficient accuracy} \\ -1 & A \text{ and } B' \text{ do not match} \end{cases} \tag{3.3}$$

  Such labels are much easier to specify, particularly if they have to labeled manually. Treating $T_{\phi,t}(\cdot)$ in the classification framework, we optimize ϕ and threshold t to order to *minimize the probability of classification error*:

$$(\phi, t) = \arg\min_{\phi,t} \{p(T_{\phi,t} = 1 | l = -1) + p(T_{\phi,t} = -1 | l = 1)\} \tag{3.4}$$

  This is the approach that is taken in this thesis.

By providing a substantially comprehensive training set of example pairs of matched images and mismatched images, the framework will learn the match classifier function $T_{\phi,t}(A, B, \theta)$, and thus the correct feature distance measure $D_{\phi}(A, B')$ and threshold t to use for matching. This distance measure will not be dependent on the *absolute* visual content of the images, but instead depend only on the *relative* visual content of pairs of images.

3.3 Parametric Family of Distance Measures for Feature Comparison

The form of distance measures is crucial in achieving a good result in the classification framework, but should also have smooth properties that make the distance measure usable in more general situations beyond matching. The form that is selected in this thesis has a further advantage that it subsumes other commonly used distance measures as degenerate special cases, as discussed later in Section 3.3.4.

3.3.1 Image Representation

The form of the distance measure is however tied to the image representation used, the image representation used in this chapter is the plain feature set representation described in Chapter 2, *i.e.*, an image is expressed in a discrete form as a set of features given by:

$$\mathcal{F} = \{\boldsymbol{f}_k \mid k = 1, \ldots, K\} \tag{3.5}$$

where

$$\boldsymbol{f}_k = \begin{bmatrix} x_{fk} & y_{fk} & F(x_{fk}, y_{fk}) \end{bmatrix}^T \tag{3.6}$$

Here (x_{fk}, x_{fk}) are sampled locations in F according to the spatial sampling function. Please refer to Section 2.1.2 for the meaning of the notations, and see figure 2.3 for an illustration.

Similar representations are also used in other works. For example, Kondor and Jebara [37] represent examples as sets of vectors, where an image can be seen as a collection of $(x, y, intensity)$ tuples, and a video sequence be seen as a collection of $(x, y, intensity, time)$ 4-tuples. Jebara used the same representation to model images as bags of pixels in [33]. Wolf and Shashua used a similar sets of vectors representation in combination with using the kernel trick in [67]. In our approach, while image representation is similar to those used in previous works cited above, the application of this representation is vastly different.

3.3.2 Distance Measure based on Feature Cross-Affinity

Given the image $\mathcal{F}$ and another image $\mathcal{G} = \{\boldsymbol{g}_j \mid j = 1, \ldots, J\}$ expressed in the form of sets of feature vectors, we now define a parametric distance measure to evaluate the similarity between them. The proposed distance measure is a generalization of the *Average Hausdorff Distance* [17, 5]:

$$D_{\phi}(\mathcal{F}, \mathcal{G}) = \frac{1}{|\mathcal{F}|} \sum_{\boldsymbol{f} \in \mathcal{F}} \min_{\boldsymbol{g} \in \mathcal{G}} h(\boldsymbol{f}, \boldsymbol{g}) + \frac{1}{|\mathcal{G}|} \sum_{\boldsymbol{g} \in \mathcal{G}} \min_{\boldsymbol{f} \in \mathcal{F}} h(\boldsymbol{g}, \boldsymbol{f}) \tag{3.7}$$

where $h(\boldsymbol{f}, \boldsymbol{g})$ is a Mahalanobis distance function comparing $\boldsymbol{f}$ to $\boldsymbol{g}$ given by

$$h(\boldsymbol{f}, \boldsymbol{g}) = \sqrt{(\boldsymbol{f} - \boldsymbol{g})^T \mathbf{C}^{-1} (\boldsymbol{f} - \boldsymbol{g})} \tag{3.8}$$

where $\mathbf{C}$ is a symmetric positive definite matrix (equivalent to the covariance matrix for a Mahalanobis distance). The matrix $\mathbf{C}$ is called the cross-affinity matrix and encapsulates the six degrees of freedom (*i.e.* ϕ) available in optimizing the

distance measure D for image matching. Mahalanobis distance is also used in machine learning on supervised distance metric learning, *e.g.* in [68].

Intuitively, this family of distance measures are used to compute a 'best' correspondence between two pixels in $\mathcal{F}$ and $\mathcal{G}$, where the measured feature differences are not limited to intensity differences of pixels with the same relative spatial position as in the case of straightforward SSD. Instead, some trade-off is allowed in pixel intensity differences and pixel position perturbation, when measuring the feature differences. The amounts of difference and trade-off are controlled by $\mathbf{C}$, and learned from data.

3.3.3 Analysis of the Cross-Affinity Matrix

If the cross-affinity matrix $\mathbf{C}$ is the identity matrix, the Mahalanobis distance reduces to the Euclidean distance. If the cross-affinity matrix is diagonal, then the resulting distance measure is normalized Euclidean distance:

$$h(\boldsymbol{f},\boldsymbol{g}) = \sqrt{(\boldsymbol{f}-\boldsymbol{g})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{f}-\boldsymbol{g})} \tag{3.9}$$

$$= \sqrt{\sum_{i=1}^{p}\frac{(f_i-g_i)^2}{\sigma_i^2}} \tag{3.10}$$

where σ_i is the ith diagonal element of diagonal matrix $\boldsymbol{\Sigma}$. The inverse of σ_i^2 can be seen as a weight for each summed term $(f_i - g_i)^2$, so the smaller σ_i is , the more $(f_i - g_i)$ affects the distance $h(\boldsymbol{f},\boldsymbol{g})$.

Generally, any matrix $\mathbf{C}$, which is symmetric positive definite, can be expressed in the following form by Principle Component Analysis (PCA) [18]:

$$\mathbf{C} = \mathbf{P}\boldsymbol{\Lambda}\mathbf{P}^T \tag{3.11}$$

with $\mathbf{P} = (e_1, ..., e_p)$ a p-dimensional square matrix whose columns are the normalized eigenvectors of $\mathbf{C}$ and $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, ..., \lambda_p)$ a p-dimensional diagonal

matrix whose diagonal elements are the eigenvalues of $\mathbf{C}$. Note that $\mathbf{P}$ is an orthogonal matrix.

Then (3.8) can be expressed as:

$$h(\boldsymbol{f},\boldsymbol{g}) = \sqrt{(\boldsymbol{f}-\boldsymbol{g})^T\mathbf{P}\boldsymbol{\Lambda}^{-1}\mathbf{P}^T(\boldsymbol{f}-\boldsymbol{g})} \quad (3.12)$$

$$= \sqrt{(\boldsymbol{u}-\boldsymbol{v})^T\boldsymbol{\Lambda}^{-1}(\boldsymbol{u}-\boldsymbol{v})} \quad (3.13)$$

where $\boldsymbol{u} = \mathbf{P}^T\boldsymbol{f}$, $\boldsymbol{v} = \mathbf{P}^T\boldsymbol{g}$. $\boldsymbol{u}$ (or $\boldsymbol{v}$) can be seen as a new feature vector formed by projecting $\boldsymbol{f}$ (or $\boldsymbol{g}$) on the set of orthogonal basis $(e_1, ..., e_p)$. Compared with (3.9), (3.13) can be looked as a normalized Euclidean distance based on the new feature vectors $\boldsymbol{u}$ and $\boldsymbol{v}$. The smaller λ_i is, the more $(u_i - v_i)$ affects the distance $h(\boldsymbol{f},\boldsymbol{g})$.

3.3.4 Comparison to Standard Distance Measures

The cross-affinity family of distance measures subsume commonly-used metrics as degenerate instances of $\mathbf{C}$ as shown below:

$$\mathbf{C}_{\text{SSD}} = \begin{bmatrix} 0^+ & 0 & 0 \\ 0 & 0^+ & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{C}_{\text{Chamfer}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0^+ \end{bmatrix}$$

$$\mathbf{C}_{\text{Shuffle}} = \begin{bmatrix} \infty & 0 & 0 \\ 0 & \infty & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3.14)$$

This may be considered intuitively. SSD enforces a strict correspondence and only pixels with the same spatial coordinates to be compared. Since $\mathbf{C}$ is the covariance matrix under a Mahalanobis interpretation, the positive zeros in the first two diagonal entries indicate that spatial coordinates are not allowed to vary

(zero variance) when scouring for the best pixel "correspondences" in computing the distance. Similarly, chamfer matching requires that edge pixels match to only edge pixels, hence the diagonal entry corresponding to feature value variance is zero; however it allows for spatial variation and thus the non-zero variances for spatial coordinates. The same analysis may be applied to the shuffle distance.

Having the family of cross-affinity distance measures subsume these commonly-used metrics is a strong benefit. If one of these standard measures is truly the optimal measure, learning the best cross-affinity distance measure will result learning this same optimal standard measure. Hence the cross-affinity distance measure is instantly guaranteed to outperform these standard measures, at least when applied to the training set.

3.4 Discriminative Analysis by Histogram Intersection

3.4.1 Definition of Histogram Intersection Distance

The misclassification rate here is the sum of probabilities of false positives and false negatives. Given a fixed ϕ and t, this probability is

$$p(\text{error}) = p(T_{\phi,t} = 1 | l = -1) + p(T_{\phi,t} = -1 | l = 1) \tag{3.15}$$

Assuming that an optimal threshold t is used, this error may also be expressed as the *intersection* of the distributions of matches and mismatches in the one-dimensional cross-affinity distance space (figure 3.2):

$$p(\text{error}) = \int_{-\infty}^{+\infty} \min \left\{ p(l = 1, D_\phi), p(l = -1, D_\phi) \right\} dD_\phi \tag{3.16}$$

The distributions of matches and mismatches in feature distance space can be approximated through the use of smoothed histograms.

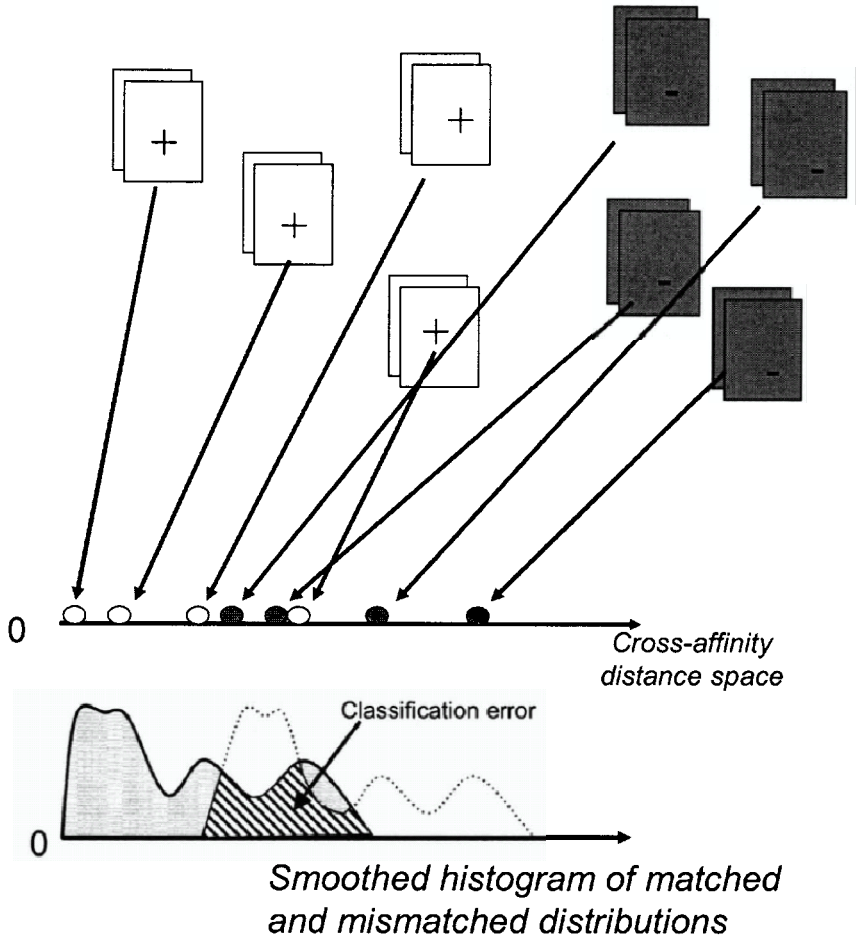


Figure 3.2: Smoothed histogram computation for matched and mismatched classes, and classification error in the form of histogram intersection.

Given the set of matched training instances $\{(A, B, \boldsymbol{\theta}, l) \mid l = 1\}$, we can compute a set of positive class distances $\{d_{Pi} = D_{\boldsymbol{\phi}}(A_i, B_i') \mid i = 1, \ldots, n\}$. The

smoothed histogram is defined as

$$\hat{f}_P(x) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x - d_{Pi}}{h}) \tag{3.17}$$

where x is the cross-affinity distance space ordinate. The function $K(\cdot)$ is a kernel function satisfying the condition

$$\int_{-\infty}^{+\infty} K(x)dx = 1 \tag{3.18}$$

In this thesis, the Gaussian kernel is used:

$$K(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} \tag{3.19}$$

The parameter h is the smoothing parameter (alternatively called the window width or bandwidth). An optimal h based on [57] is used here, the details for choosing h is described in the next section.

Similarly, for the negative distances $\{d_{Ni}, i = 1, \dots, m\}$, a smoothed histogram may also be obtained:

$$\hat{f}_N(x) = \frac{1}{mh}\sum_{i=1}^{m} K(\frac{x - d_{Ni}}{h}) \tag{3.20}$$

The smoothed histogram intersection distance is defined by

$$d_\cap(\hat{f}_P, \hat{f}_N) = 1 - \int_{-\infty}^{+\infty} min((\hat{f}_P(x), \hat{f}_N(x))dx \tag{3.21}$$

The optimal matrix $\mathbf{C}$ corresponds to the largest $d_\cap$ by this definition.

3.4.2 Choice of the Smoothing Parameter

The parameter h plays a important role in determining the value of the smoothed histogram intersection distance $d_\cap$. This can be explained by the following: when all the other conditions are fixed, a larger h, which means a larger extent of

smoothing, will lead to a larger intersection of the matched and mismatched distributions, therefore result in a small histogram intersection distance $d_\cap$, as expressed by (3.17), (3.20) and (3.21). In addition, the value of h should not be fixed during the process of optimization for $\mathbf{C}$, because in each iteration, a different $\mathbf{C}$ will produce different sets of positive and negative distances of the training class, therefore the value of h have to be dynamically adjusted for different histograms to best reveal the "real" distribution.

In this thesis, we adopt the method proposed by Silverman [57] to decide the values of h dynamically during the optimization process, which is briefly described below. Suppose $X_1, ...X_n$ to be a sample of n real observations from an i.i.d. continuous distribution f, and $\hat{f}$ be the estimated distribution. The kernel estimator with kernel K is defined by:

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x - X_i}{h}) \tag{3.22}$$

where h is the smoothing parameter (or window width) we hope to optimize to make $\hat{f}$ best reveal the real distribution f. The kernel function K satisfies the condition

$$\int_{-\infty}^{+\infty} K(x)dx = 1 \tag{3.23}$$

The *mean integrated square error* (MISE), which is the most widely used global measure of discrepancy between f and $\hat{f}$, is defined by

$$\mathbf{MISE}(\hat{f}) = E\int \{\hat{f}(x) - f(x)\}^2 dx \tag{3.24}$$

Then the optimal h, from the point of view of minimizing the MISE, is given by

$$h_{opt} = k_2^{-2/5}\{\int K(t)^2 dt\}^{1/5}\{\int f''(x)^2 dx\}^{-1/5} n^{-1/5} \tag{3.25}$$

where

$$k_2 = \int t^2 K(t)dt \tag{3.26}$$

The detailed steps leading to (3.25) are omitted here, see details in [57].

According to (3.25), the optimal smoothing parameter h_{opt} depends on the unknown real density f, which is not available. Whereas for automatic choice of smoothing parameter, standard distributions can be used as reference to assign a value to the term $\int f''(x)^2 dx$ in (3.25). In this thesis, the normal distribution is used as the reference distribution. Suppose a normal distribution with variance σ^2, and denoting ϕ to be the standard normal density, then:

$$\begin{aligned} \int f''(x)^2 dx &= \sigma^{-5} \int \phi''(x)^2 dx \\ &= \frac{3}{8}\pi^{-\frac{1}{2}}\sigma^{-5} \simeq 0.212\sigma^{-5} \end{aligned} \tag{3.27}$$

In addition, we use Gaussian kernel which is expressed as

$$K(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} \tag{3.28}$$

Then we combine (3.27) and (3.28) into (3.25) and get:

$$\begin{aligned} h_{opt} &= (4\pi)^{-\frac{1}{10}} \frac{3}{8} \pi^{-\frac{1}{2}} \sigma n^{-\frac{1}{5}} \\ &= 1.06\sigma n^{-\frac{1}{5}} \end{aligned} \tag{3.29}$$

where the standard deviation σ can be estimated from the samples, and n is the number of observations in the sample.

In this thesis, we use (3.29) to estimate the optimal smoothing parameter h for the positive class, and the assumption of using normal distribution as the reference distribution is reasonable from the experimental results. The same value of h is used for the negative class as well.

3.4.3 Minimization of the Histogram Intersection

The minimization of the histogram intersection distance is carried by manipulating the cross-affinity matrix $\mathbf{C}$ via the BFGS Quasi-Newton method [12]. Since this is a local gradient-based search, a number of different random initialization points were used in order to better find the global minimum.

There is a minor complication to updating the $\mathbf{C}$ matrix since the elements are not independent. In order to preserve the symmetric positive definiteness of $\mathbf{C}$, the Cholesky decomposition form [21, 45] is used instead

$$\mathbf{C} = \mathbf{U}^T\mathbf{U} \tag{3.30}$$

where $\mathbf{U}$ is an upper triangular matrix. Optimization is carried out with respect to the six free parameters in the $\mathbf{U}$ matrix given by:

$$\mathbf{U} = \begin{bmatrix} u_1 & u_2 & u_3 \\ 0 & u_4 & u_5 \\ 0 & 0 & u_6 \end{bmatrix} \tag{3.31}$$

The optimization of C is done with respect to the 6 non-zero elements $u_1, \ldots, u_6$. In our implementation, the Levenberg-Marquardt method [51] is used for optimizing these parameters.

3.5 Experiments

We conducted classification and tracking experiments on three sets of data. The first is the Triesch hand posture database, which is used as PETS2002 [1] posture data set. The second is the CMU pose, illumination and expression (PIE) database of human faces [58]. The third is a car sequence, which contains 50 frames of size

320×240. These images are from a camera in a trailing car, observing the car in front negotiating a slight bend.

3.5.1 Experiments on the Triesch Database

An edge-based classification task is performed on the Triesch database, with the intention of testing the ability of our method to distinguish between different postures performed by random people.

The Triesch hand posture database consists of 10 hand signs performed by 24 persons. For each person, the 10 postures were recorded in front of uniform light, uniform dark and complex background giving 720 images of which 2 were missing. The images are recorded in 8-bit gray-scale and are 128×128 pixels in size.

We selected pairs of images from the Triesch database to form a training set and a testing set. The training set includes images of the first 5 hand postures performed by all the 24 persons in front of uniform light background, and the test set includes images of the last 5 postures performed by all the 24 persons in front of uniform light background. All images in the training and testing sets are cropped hand images, so they are of different sizes, ranges from 45×47 to 95×117.

For either of the training and testing sets, the images are organized into 100 positive pairs and 300 negative pairs. Each positive pair contains two images of the same hand posture but from different persons, and each negative pair contains two images of different hand postures from random persons. The positive pairs are considered matched, and the negative pairs are considered mismatched. Our aim is to find the optimal Cross-Affinity Distance to best distinguish between the

positive and the negative pairs. Some sample image pairs from the training set are shown in figure 3.3. The 3 pairs in the top row are positive pairs, and the 3 pairs in the bottom are negative pairs. All the 5 hand postures in the training set can be found in figure 3.3.

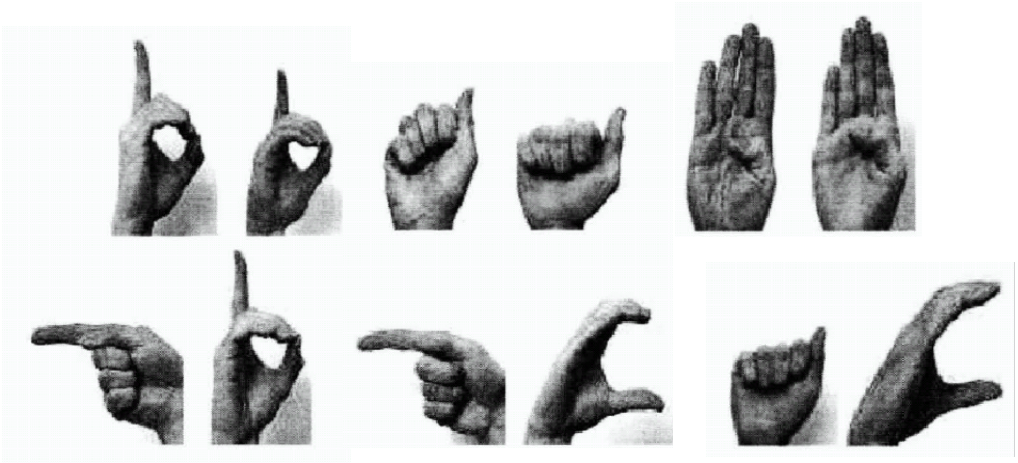


Figure 3.3: Positive (top row) and negative (bottom row) image pairs from cropped Triesch training set.

Oriented edge features are used in this experiment. These edge features are extracted by a set of 12 templates cover the directions from 0° to 180° with 15° step. The size of the templates is 7×7. The first seven templates are shown in the top row of figure 3.4. The edge extraction algorithm is simple normalized correlation with thresholding, and non-maximum suppression is used to reduce the number of extracted edges. Five extraction results are shown in the bottom row of figure 3.4, short lines representing extracted oriented edges are superimposed on the original images. These five images also show the 5 hand postures in the testing set. Then in the following experiment, each hand posture is represented by a set of oriented edge features with 3 properties: the x, y positions and the edge direction.

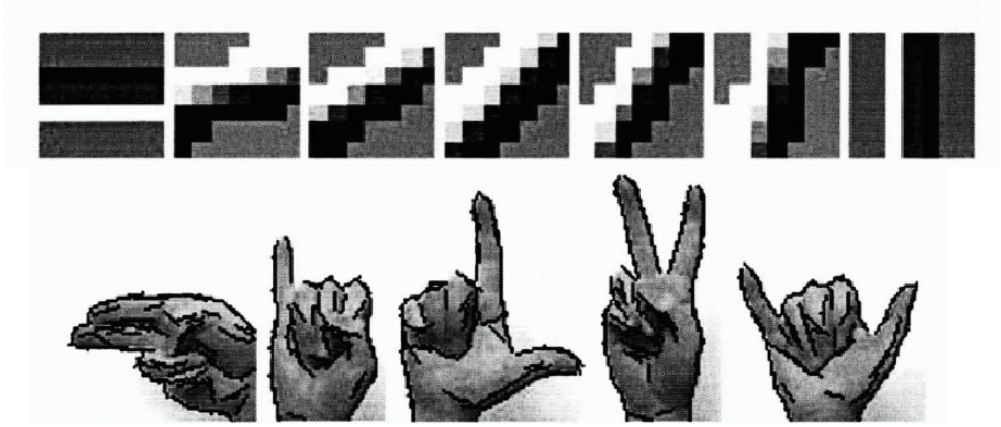


Figure 3.4: Edge templates and extracted edge features

The optimal cross-affinity matrix C found by the BFGS Quasi-Newton method is:

$$\mathbf{C} = \begin{bmatrix} 1.0000 & -0.64624 & 0.084404 \\ -0.64624 & 0.41792 & -0.054565 \\ 0.084404 & -0.054565 & 0.0071274 \end{bmatrix} \tag{3.32}$$

The corresponding minimum histogram intersection distances for the training set and test set are 0.127 and 0.124 respectively.

As explained in Section 3.3.3, the matrix C can be decomposed into the form of (3.11), where

$$\mathbf{P} = \begin{bmatrix} -0.040186 & 0.54460 & 0.83773 \\ 0.068016 & 0.83795 & -0.54149 \\ 0.99687 & -0.035219 & 0.070716 \end{bmatrix} \tag{3.33}$$

, and

$$\mathbf{\Lambda} = \begin{bmatrix} 1.9637 \times 10^{-6} & 0 & 0 \\ 0 & 2.0863 \times 10^{-4} & 0 \\ 0 & 0 & 1.4248 \end{bmatrix} \tag{3.34}$$

According to (3.12), the new features $\boldsymbol{u}$ (or $\boldsymbol{v}$) is formed by projecting $\boldsymbol{f}$ (or $\boldsymbol{g}$) on the set of orthogonal basis $(e_1, ..., e_p)$. In this experiment, the physical meaning

of the elements of $\boldsymbol{f}$ (or $\boldsymbol{g}$) is $(x, y, edge\_orientation)$, therefore we can express the physical meaning of $\boldsymbol{u}$ as:

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} -0.040186 \cdot x & + & 0.068016 \cdot y & + & 0.99 \cdot edge\_orientation \\ 0.54460 \cdot x & + & 0.83795 \cdot y & - & 0.035219 \cdot edge\_orientation \\ 0.83773 \cdot x & - & 0.54149 \cdot y & + & 0.070716 \cdot edge\_orientation \end{bmatrix} \tag{3.35}$$

$\boldsymbol{u}_1$ is mostly composed by the $edge\_orientation$ component, $\boldsymbol{u}_2$ and $\boldsymbol{u}_3$ are mostly composed by combination of x and y components. Since the first diagonal entry λ_1 in (3.34) is smallest, this means whether a corresponding feature point with the same edge orientation can be found in some variation of x and y components is the key to have a short distance. From this point, this distance measure is quite like Chamfer distance. But the handling of x and y components through $\boldsymbol{u}_2$ and $\boldsymbol{u}_3$ is different from Chamfer distance.

We also ran experiments on traditional Chamfer matching method to compare the results with our method. The result ROC curves for our method and chamfer matching is show in figure 3.5.

The distributions of the positive and negative class distances for our method and chamfer matching for the training set is shown in figure 3.6. Similar graphs for the test set are shown in figure 3.7. The Histogram Intersection Distance (HID) is also shown on top of each graph.

Among the 100 positive image pairs and 300 negative image pairs in the test set, the Cross-Affinity method resulted in 5 false negatives and 15 false positives, while for chamfer matching there were 11 false negatives and 33 false positives. The classification threshold in this instance was based on Equal Error Rate (EER). Some classification results of the test set are shown in figure 3.8. Correct classifications, *i.e.* classifying positive image pairs as matched or negative image pairs as

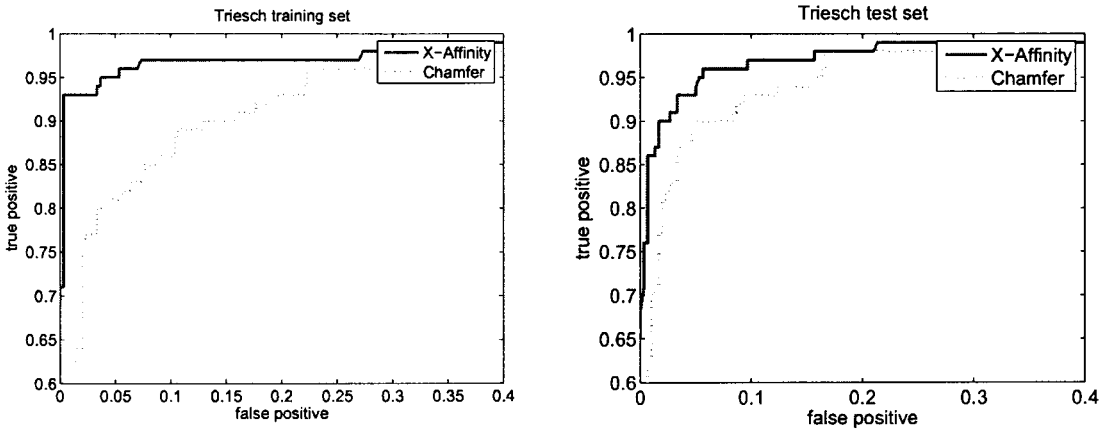


Figure 3.5: ROC curves for training(left)and test(right) sets of Triesch hand dataset.

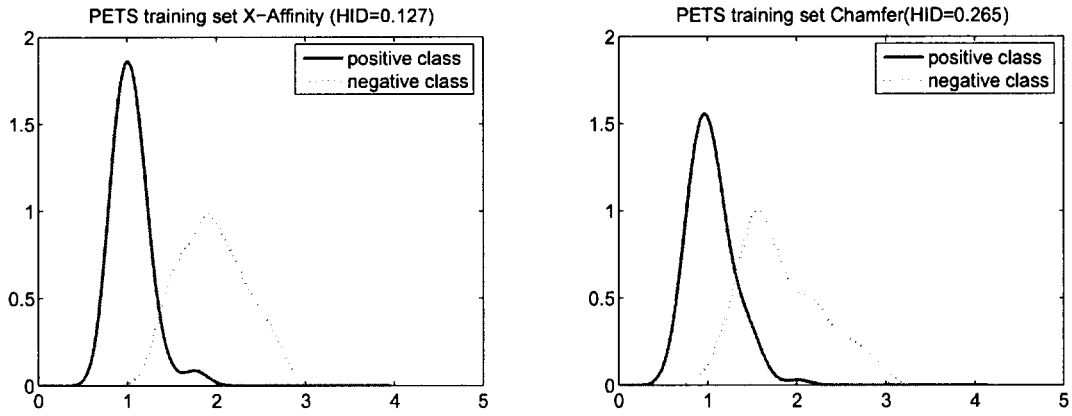


Figure 3.6: Distributions of Cross-Affinity(left) and chamfer(right) distances for the Triesch hand training set.

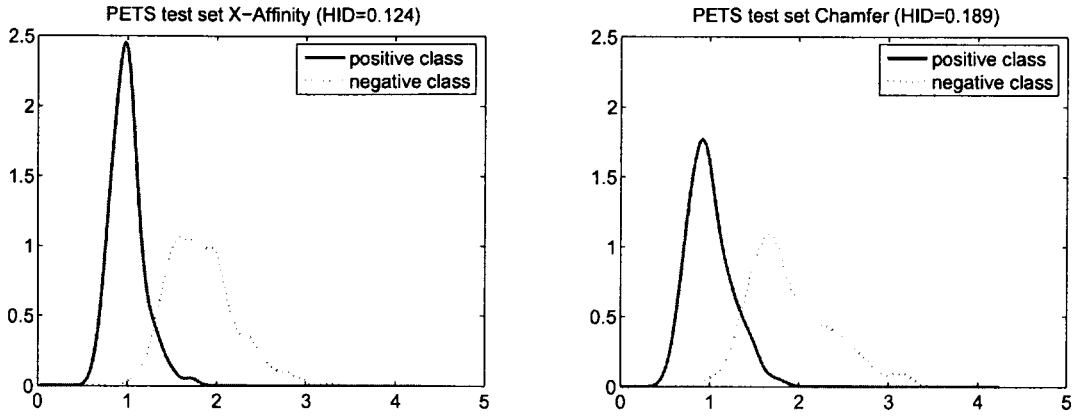


Figure 3.7: Distributions of Cross-Affinity(left) and chamfer(right) distances for the Triesch hand test set.

mismatched, are indicated by "$\surd$", while incorrect classifications are indicated by "$\times$". The correct classifications (ticks) by Cross-Affinity are shown in red color.

3.5.2 Experiments on the PIE Face Database

An intensity-based classification task is performed on the PIE database, in order to test the ability of our method to distinguish between different human faces in random expressions.

The CMU Pose, Illumination and Expression (PIE) database [58] contains 41368 face images of 68 people. The images are of different poses, illuminations and expressions. We chose images from the first 20 people to form the training set, and images from the next 20 people to form the test set. All the selected images are in the front view with the same illumination, but contain three different expressions – neutral, smile and blink. The images are cropped to just cover the face area, of size 53×67.

For both training and test sets, we constructed 50 positive image pairs as the

Figure 3.8: Sample classification results on Triesch test dataset. The top two rows are positive image pairs, while the bottom three rows are negative pairs. The classification results of Cross-Affinity and chamfer matching are shown below each image pair (chamfer matching results are shown in the parentheses). Correct classifications are indicated by "ticks", while incorrect classifications are indicated by "crosses". The correct classifications (ticks) by Cross-Affinity are emphasized in red color.

positive set and 100 negative images pairs as the negative set. Each positive pair contains 2 images from the same person but of different expressions, each negative pair contains 2 images from different persons of random expressions. Some sample image pairs for the training set is shown in figure 3.9; the six image pairs in the top two roes are positive pairs, and the six image pairs in the bottom two rows are negative pairs.

Figure 3.9: Sample image pairs from the PIE training set. The six image pairs in the top two rows are positive pairs, and the six image pairs in the bottom two rows are negative pairs.

The optimal cross-affinity matrix C found by the BFGS Quasi-Newton method

is:

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.054806 & 0 \\ 0 & 0 & 7.4977 \times 10^{-5} \end{bmatrix} \tag{3.36}$$

The corresponding minimum histogram intersection distances for the training set and test set are 0.0096 and 0.053 respectively.

The first and second diagonal entries (λ_1 and λ_2) of matrix $\mathbf{C}$ are much bigger than the thirds one (λ_3), this is like the $\mathbf{C}_{\text{Chamfer}}$ shown in (3.14). While as λ_1 is bigger than λ_2, when scouring for the best correspondence in computing the distance, the variation of x component can be bigger than y component. This is possibly because the changes between positive image pairs is mostly along the x direction, caused by opening and closing of eyes and mouthes.

We also ran experiments on SSD and histogram-based Bhattacharyya measure for comparison. The resulting ROC curves for our method, SSD and histogram measure are shown in figure 3.10. The left graph is for the training set, and the right graph is for the test set. The curve for Cross-Affinity does not appear in the left graph because it is superposed by the axes (perfect classification in this case).

The distributions of positive and negative class distances for training set is shown in figure 3.11, while the distributions for the test set is in figure 3.12.

For the 50 positive image pairs and 100 negative image pairs in the training set, the Cross-Affinity method has no error in classification, whereas the SSD method has 2 false negatives and 4 false positives. The six misclassified image pairs by SSD are shown in figure 3.13 for comparison. The classification threshold in this instance is based on Equal Error Rate (EER).

In addition, for the 50 positive image pairs and 100 negative image pairs in the test set, the Cross-Affinity method has 1 false negative and 2 false positives,

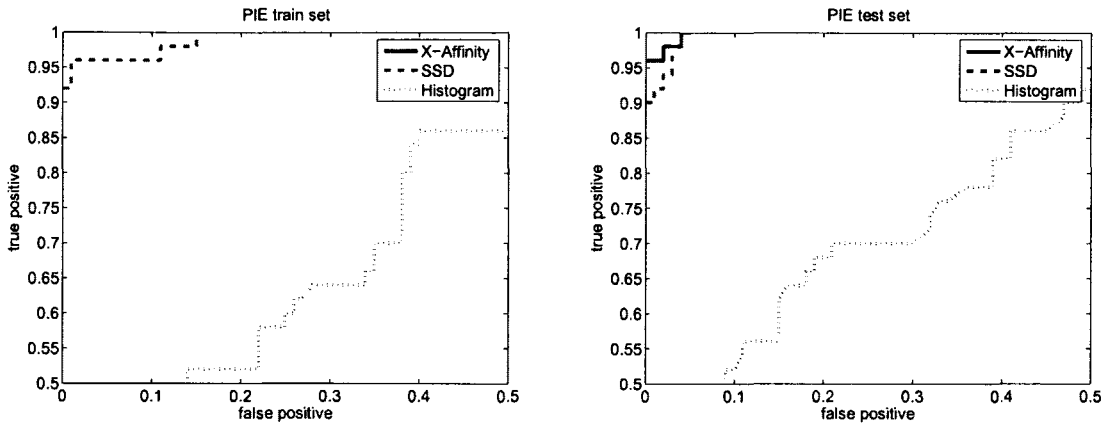


Figure 3.10: ROC curves for PIE training set (left) and test set (right) based on Cross-Affinity, SSD and Histogram distance. The curve for Cross-Affinity does not appear in the left graph because it is superposed by the axes (perfect classification).

whereas the SSD method has 1 false negative and 3 false positives. The misclassified image pairs for both methods are shown in figure 3.14.

3.5.3 Experiments on a Car Sequence

The car sequence contains 50 frames of size 320×240, showing images from a camera in a trailing car, observing the car in front negotiating a slight bend.

In the experiment, we investigated the classification performance of different metrics when applied to a dataset of smaller images that were manually cropped from the sequences mentioned above. The cropped images are of size 25x21.

The cropped dataset was divided into a training set and a test set. The training set contains the first 25 frames and the test set contains the last 25 frames. Each set has 25 images of the car: the positive class comprises images of the target region, while the 50 images in the negative class are randomly sampled from the

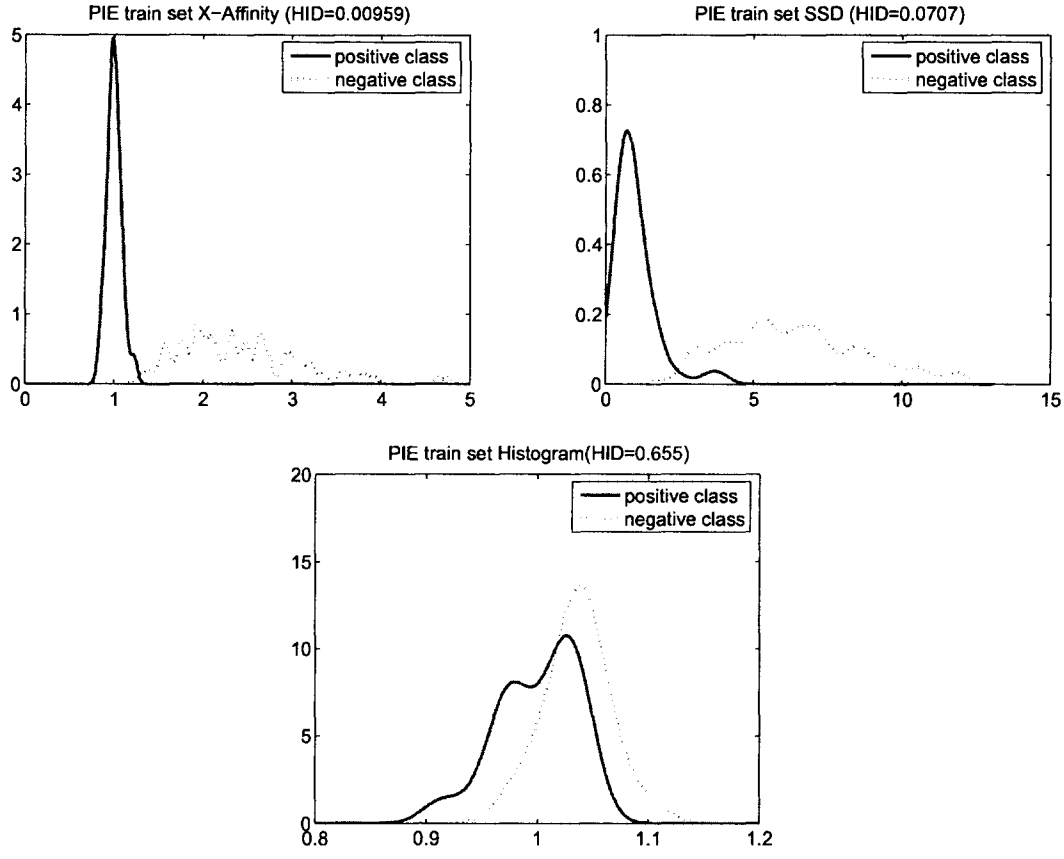


Figure 3.11: Distributions of Cross-Affinity(upper left), SSD(upper right) and Histogram(lower middle) for the PIE face training set.

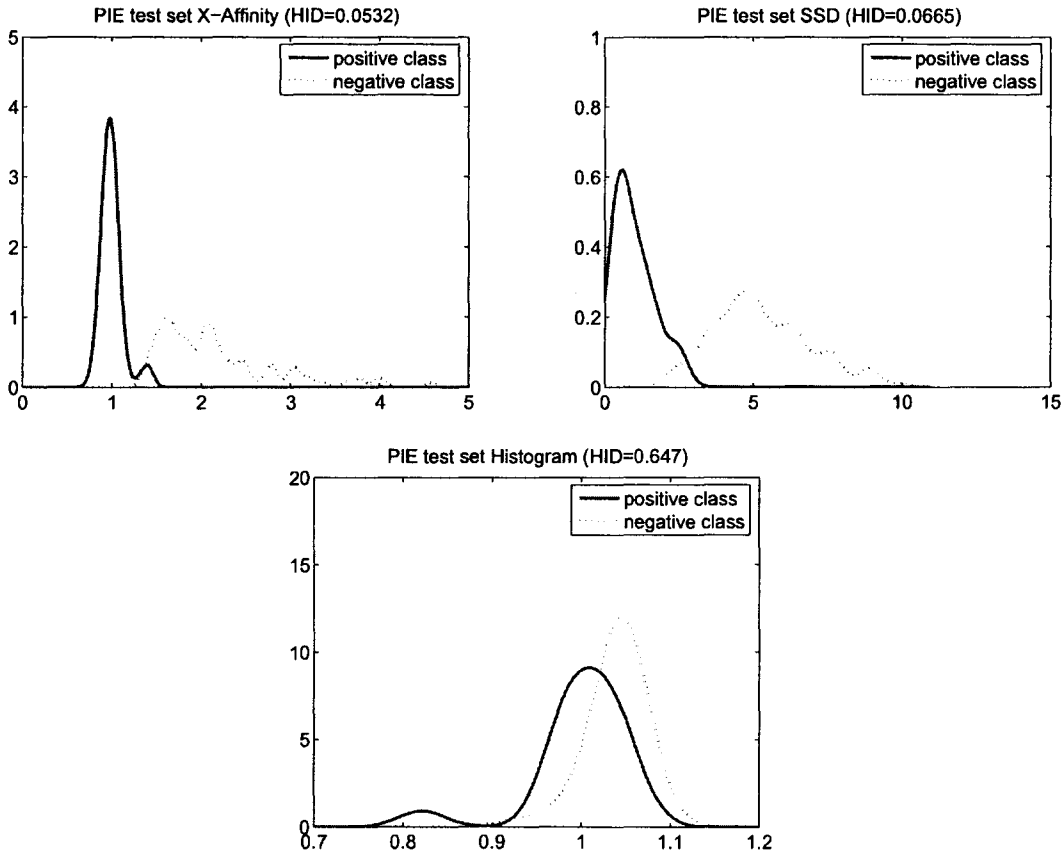


Figure 3.12: Distributions of Cross-Affinity(upper left), SSD(upper right) and Histogram(lower middle) for the PIE face test set.

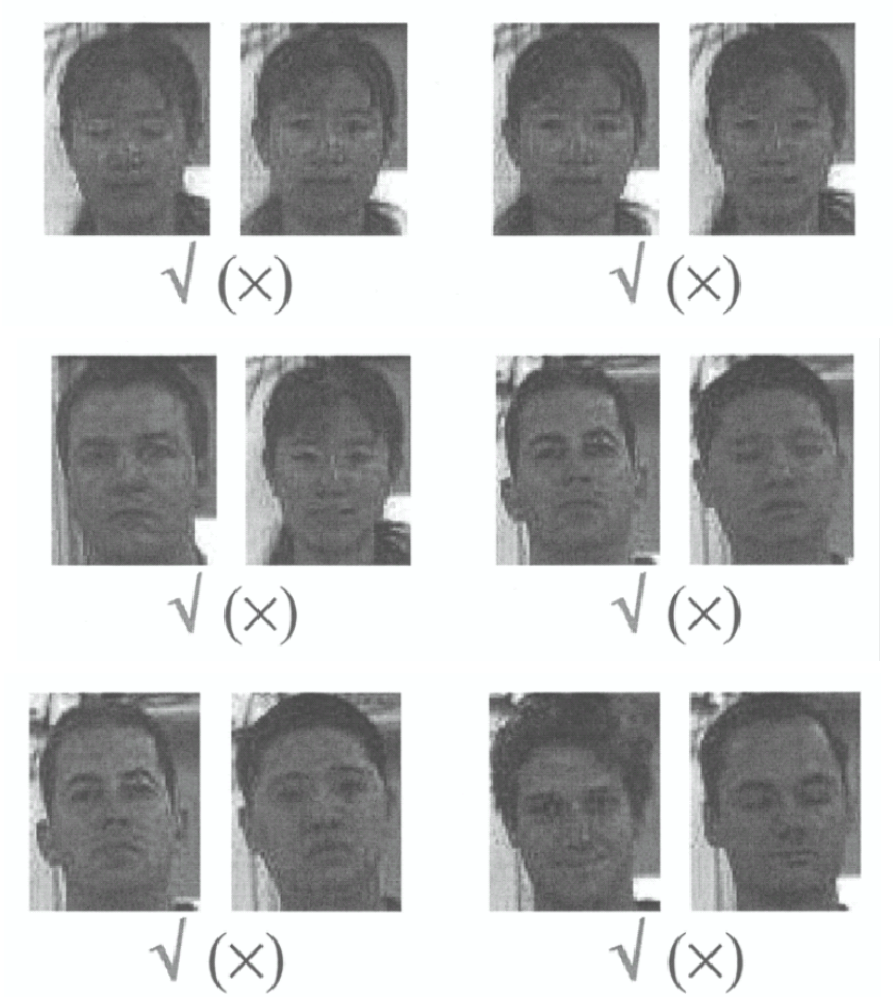


Figure 3.13: These 6 image pairs in the PIE training set are misclassified by SSD. Whereas all of them are correctly classified by Cross-Affinity method. The top row are positive image pairs, and the middle and bottom rows are negative pairs. The notations are the same as figure 3.8.

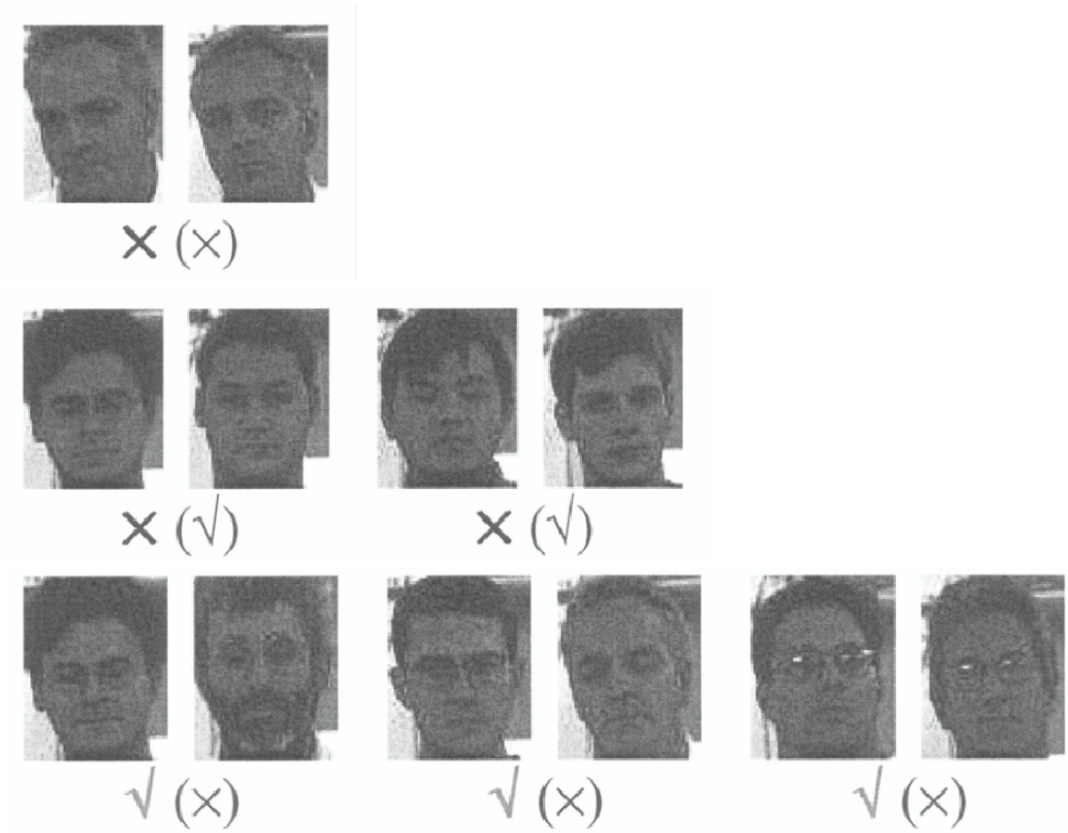


Figure 3.14: These figure shows all the misclassified image pairs in PIE test set, either by SSD or by Cross-Affinity method. The top row are positive image pairs, and the middle and bottom rows are negative pairs. The classification results of Cross-Affinity and SSD matching are shown below each image pair (SSD matching results are shown in the parentheses). The notations are the same as figure 3.8.

original frames away from the target regions.

Example images (in increasing frame order) from the positive class of the car dataset are shown in the top row of figure 3.15, while background images are shown in the second row. Notice that in the dataset, the target appearance changes systematically across the sequence, *e.g.*, the car in the sequence becomes increasingly smaller.

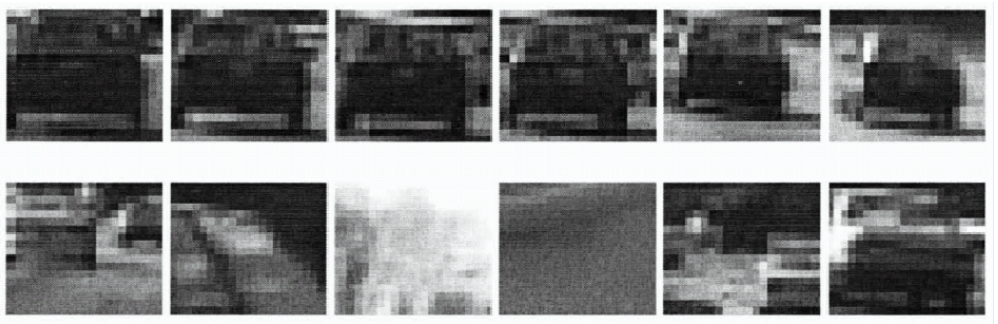


Figure 3.15: Positive and negative images from the car dataset.

Then in both cases for the training and test sets, there is a set of 625 (25x25) positive image pairs, which contains all the possible pairs between two positive images, as well as a negative image pairs set of size 1250 (25x50), containing all the possible pairs between a positive image and a negative image.

We carried out both pixel-based and edge-based experiments for this data set, the setting are the same with the previous two experiments respectively.

For pixel-based experiments, The optimal cross-affinity matrix $\mathbf{C}$ found by the BFGS Quasi-Newton method is:

$$\mathbf{C} = \begin{bmatrix} 1 & 0.024267 & 0.45957 \\ 0.024267 & 6.1463 \times 10^{-4} & 0.011115 \\ 0.45957 & 0.011115 & 0.21274 \end{bmatrix} \tag{3.37}$$

As explained in Section 3.3.3, the matrix $\mathbf{C}$ can be decomposed into the form

of (3.11), where

$$\mathbf{P} = \begin{bmatrix} 0.035731 & 0.41697 & 0.90822 \\ -0.99905 & 0.037633 & 0.022027 \\ -0.024994 & -0.90814 & 0.41792 \end{bmatrix} \tag{3.38}$$

, and

$$\mathbf{\Lambda} = \begin{bmatrix} 2.4786 \times 10^{-5} & 0 & 0 \\ 0 & 1.2692 \times 10^{-3} & 0 \\ 0 & 0 & 1.2121 \end{bmatrix} \tag{3.39}$$

According to (3.12), the new features $\boldsymbol{u}$ (or $\boldsymbol{v}$) is formed by projecting $\boldsymbol{f}$ (or $\boldsymbol{g}$) on the set of orthogonal basis $(e_1, ..., e_p)$. In this experiment, the physical meaning of the elements of $\boldsymbol{f}$ (or $\boldsymbol{g}$) is $(x, y, intensity)$, therefore we can express the physical meaning of $\boldsymbol{u}$ as:

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} 0.035731 \cdot x - 0.99905 \cdot y - 0.024994 \cdot intensity \\ 0.41697 \cdot x + 0.037663 \cdot y - 0.90814 \cdot intensity \\ 0.90822 \cdot x + 0.022027 \cdot y + 0.41792 \cdot intensity \end{bmatrix} \tag{3.40}$$

Here u_1 is mostly composed of y component, u_2 and u_3 are composed of combination of x and $intensity$ component. As λ_1 is smallest, u_1 has the smallest variation when scouring for the best correspondence in computing the distance.

The corresponding minimum histogram intersection distances for the training set and test set are 0.0060 and 0.0353 respectively. The ROC curves are shown in figure 3.16, where the comparison methods are SSD and gray level histogram distance. The distribution of positive and negative class distances for the training set is shown in figure 3.17, while those for the test set is shown in figure 3.18.

For the 625 positive image pairs and 1250 negative image pairs in the test set,

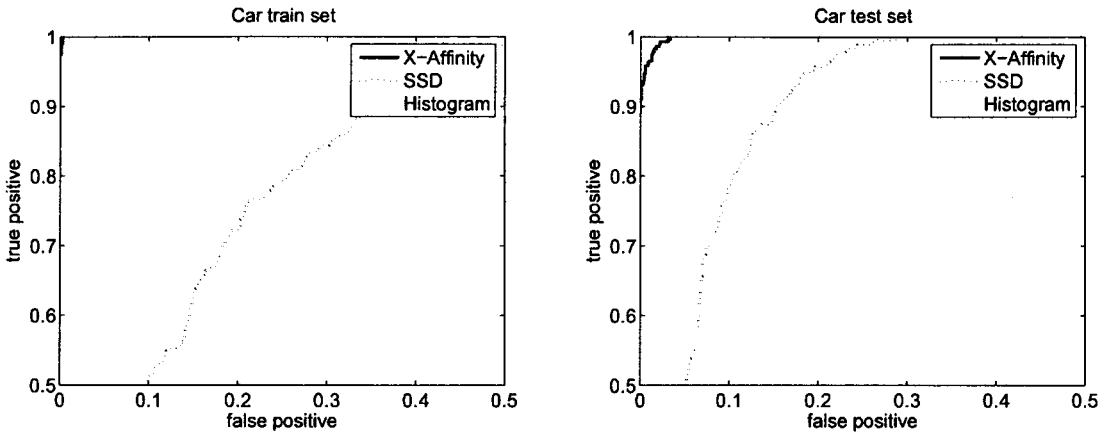


Figure 3.16: ROC curves for the car training set (left) and test set (right). The image feature is pixel intensity.

Cross-Affinity method has 10 false negative and 21 false positives, whereas SSD method has 78 false negative and 61 false positives. The misclassified image pairs for both methods are shown in figure 3.14. The classification threshold in this instance is based on Equal Error rate (EER).

For edge-based experiments, The optimal cross-affinity matrix $\mathbf{C}$ found by the BFGS Quasi-Newton method is:

$$\mathbf{C} = \begin{bmatrix} 0.04646 & 0.17264 & 0.097631 \\ 0.17264 & 1 & 0.5687 \\ 0.097631 & 0.5687 & 0.32383 \end{bmatrix} \tag{3.41}$$

As explained in Section 3.3.3, the matrix $\mathbf{C}$ can be decomposed into the form of (3.11), where

$$\mathbf{P} = \begin{bmatrix} 1.1597 \times 10^{-3} & -0.97669 & -0.21464 \\ -0.49462 & 0.18598 & -0.84898 \\ 0.86911 & 0.10715 & 0.48287 \end{bmatrix} \tag{3.42}$$

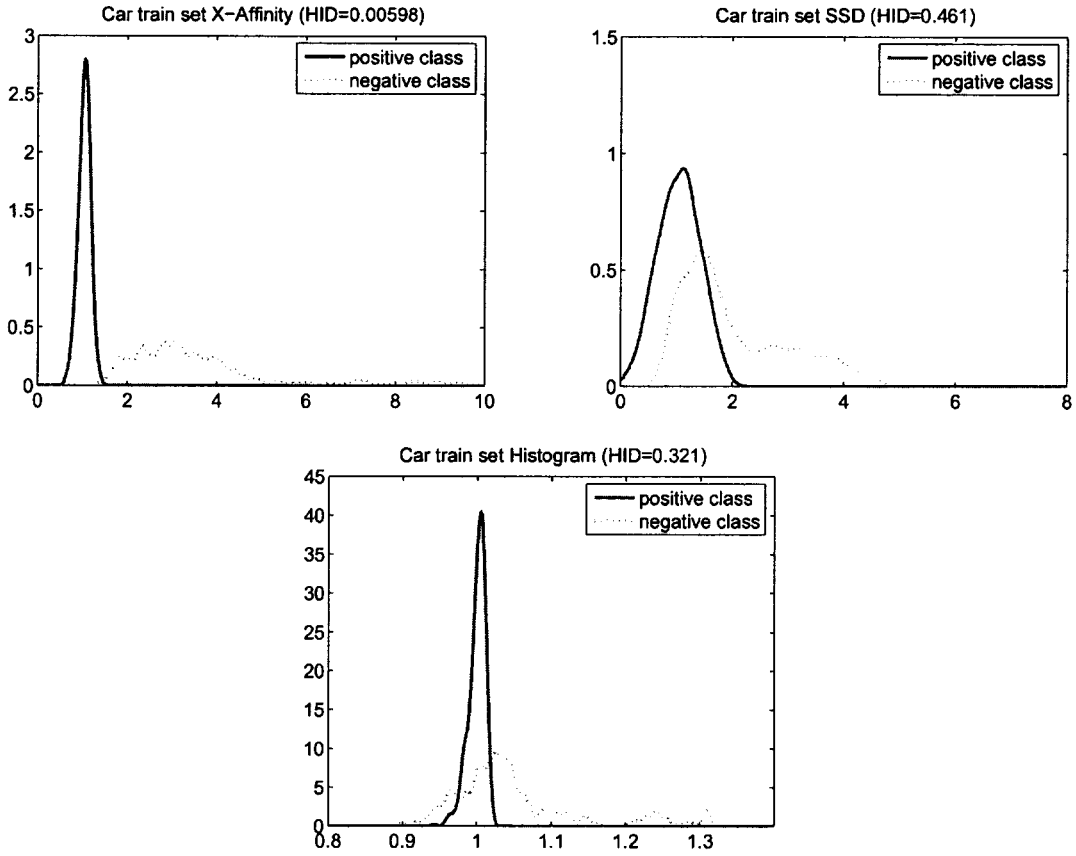


Figure 3.17: Distributions of Cross-Affinity(upper left), SSD(upper right) and Histogram(lower middle) distances for the car training set using pixel intensities

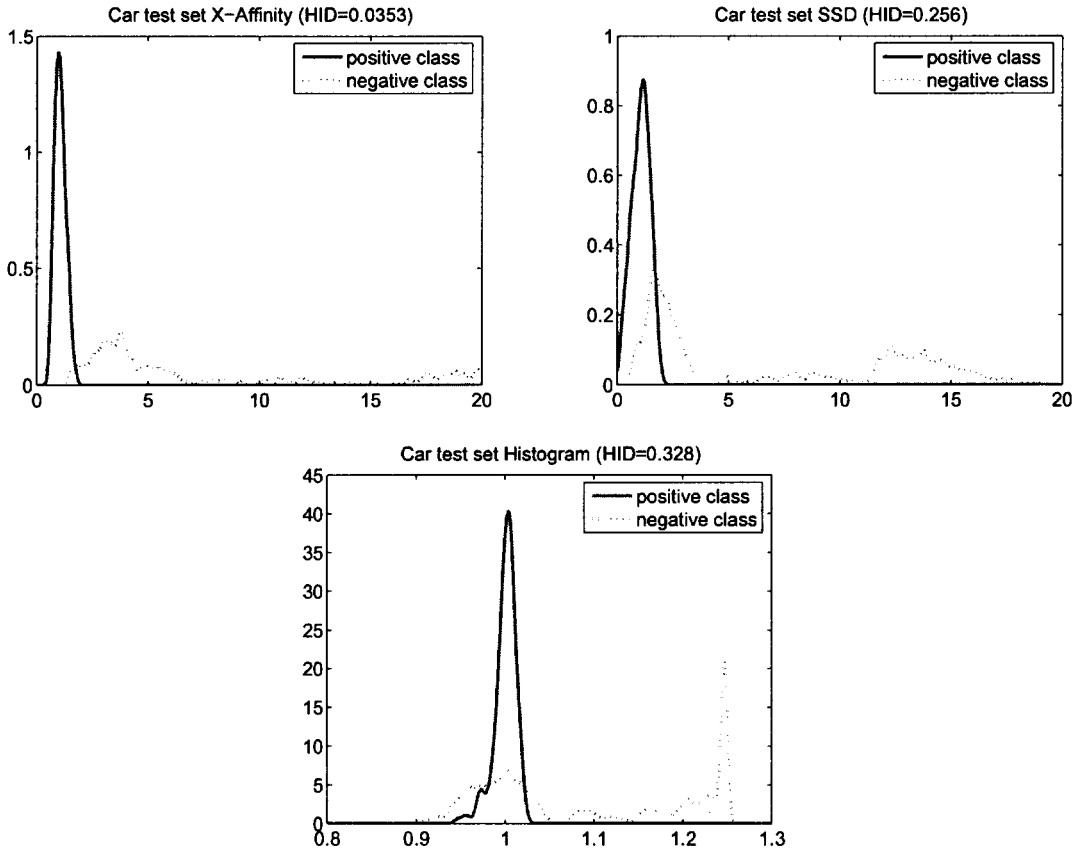


Figure 3.18: Distributions of Cross-Affinity(upper left), SSD(upper right) and Histogram(lower middle) distances for the car test set using pixel intensities

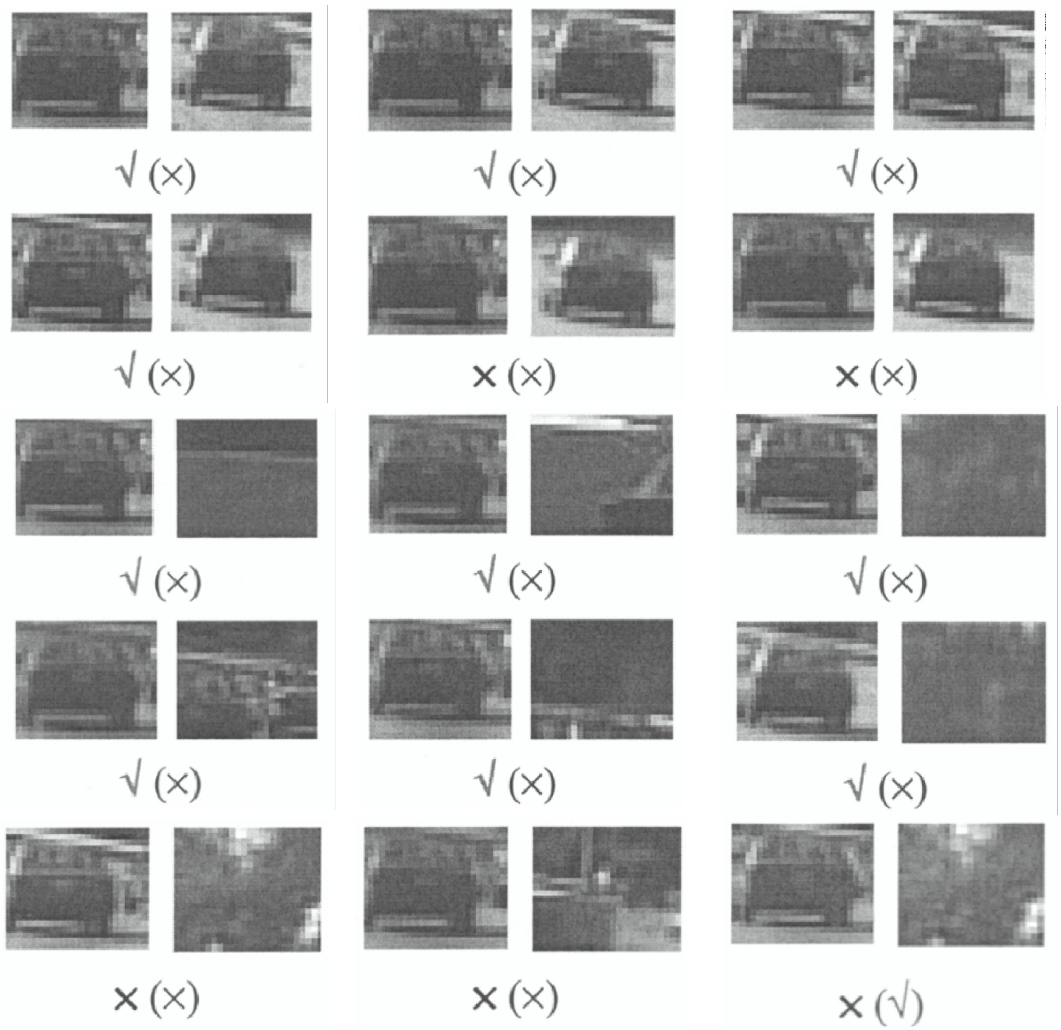


Figure 3.19: Sample classification results on the car test dataset. The top two rows are positive image pairs, while the bottom three rows are negative pairs. The classification results of Cross-Affinity and SSD are shown below each image pair (SSD matching results are shown in the parentheses). The notations are the same as figure 3.8.

, and

$$\mathbf{\Lambda} = \begin{bmatrix} 3.0943 \times 10^{-4} & 0 & 0 \\ 0 & 0.42102 & 0 \\ 0 & 0 & 1.3671 \end{bmatrix} \tag{3.43}$$

The physical meaning of the elements of $\boldsymbol{f}$ (or $\boldsymbol{g}$) is $(x, y, edge\_orientation)$, therefore we can express the physical meaning of $\boldsymbol{u}$ as:

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} 1.1597 \times 10^{-3} \cdot x & - & 0.49462 \cdot y & + & 0.86911 \cdot edge\_orientation \\ -0.97669 \cdot x & + & 0.18598 \cdot y & + & 0.10715 \cdot edge\_orientation \\ -0.21464 \cdot x & - & 0.84898 \cdot y & + & 0.48287 \cdot edge\_orientation \end{bmatrix} \tag{3.44}$$

Here u_1 is mostly composed of combination of y and $edge\_orientation$ components, while u_2 and u_3 are composed of all the three of x, y and $edge\_orientation$ components. As λ_1 is smallest, u_1 has the smallest variation when scouring for the best correspondence in computing the distance.

The corresponding minimum histogram intersection distances for the training set and test set are 0.0775 and 0.0113 respectively. The ROC curves are shown in figure 3.20, where the comparison method is chamfer matching. The distribution of positive and negative class distances for the training set is shown in figure 3.21. The distribution of positive and negative class distances for the test set is shown in figure 3.22.

We did further experiments on a tracking task. In this experiment, we investigated how well the optimized cross-affinity distance measure performs in a tracking task. The tracking procedure is as follows: the reference image is a small image patch located on the target in the first frame of the sequence. For each successive frame, the search window for the new target position is centered on the target position in the previous frame, using a window 2.5 times the size of

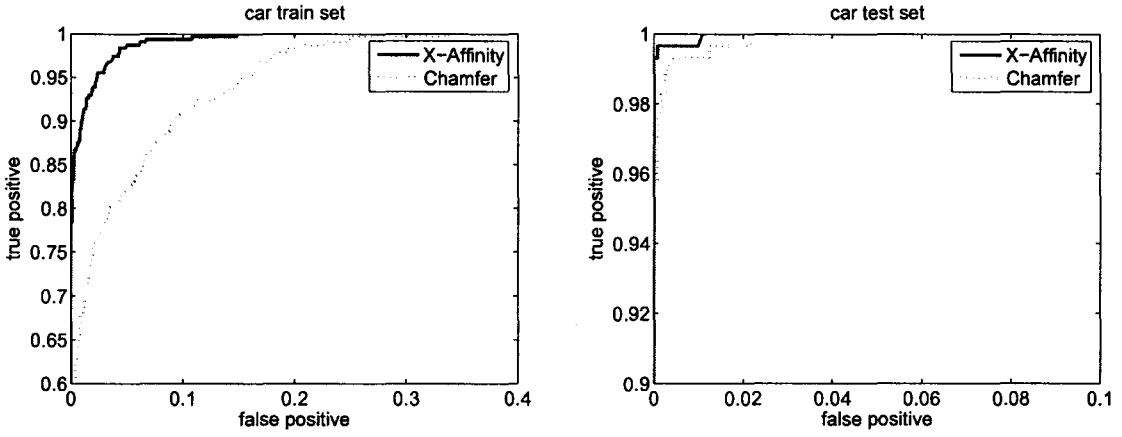


Figure 3.20: ROC curves for the car training (left) and test (right) sets using edge features.

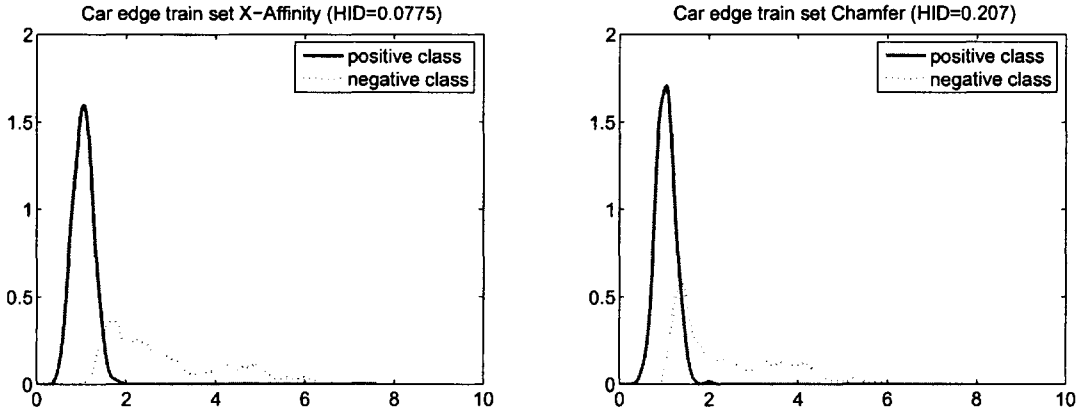


Figure 3.21: Distributions of Cross-Affinity(left) and chamfer(right) for training set using edge features

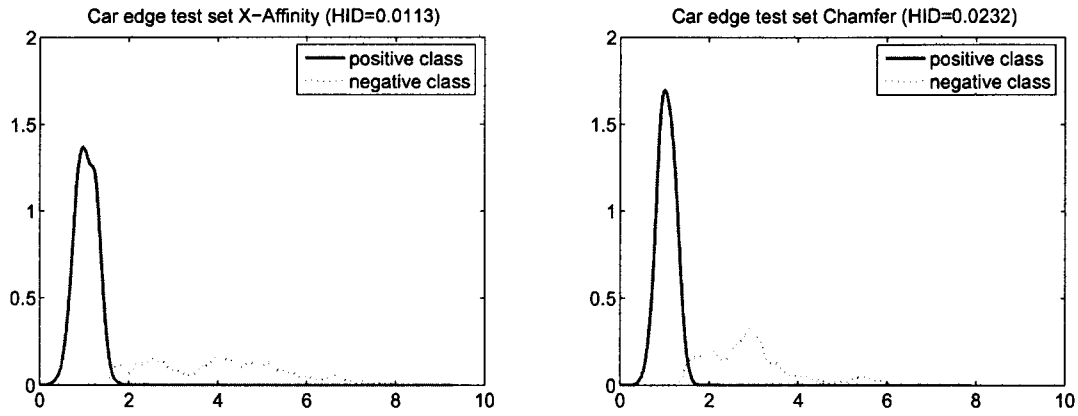


Figure 3.22: Distributions of Cross-Affinity(left) and chamfer(right) for test set using edge features

the reference image. The search window is scanned and compared to the reference image, and the position with the smallest matching distance to the reference image is selected as the new target position. The frames in which tracking fails catastrophically (as opposed to minor registration errors) are noted.

The reference image does not change throughout the tracking task, and the optimal measure computed from the previous section is used in the tracking task. No threshold is required as the position with the minimum distance is always selected as the target position. Additionally, no background modeling is done to aid the tracking.

Some sample tracking images by Cross-Affinity are shown in figure 3.23. The dark rectangles represent the target positions, while the light rectangles represent the search windows. The number of frame is shown below each image.

As before, the tracking task was performed using pixel-based SSD and histogram-based Bhattacharyya distance, and edge based chamfer distance for comparison. Table 3.1 shows the number of successive frames tracked successfully using the

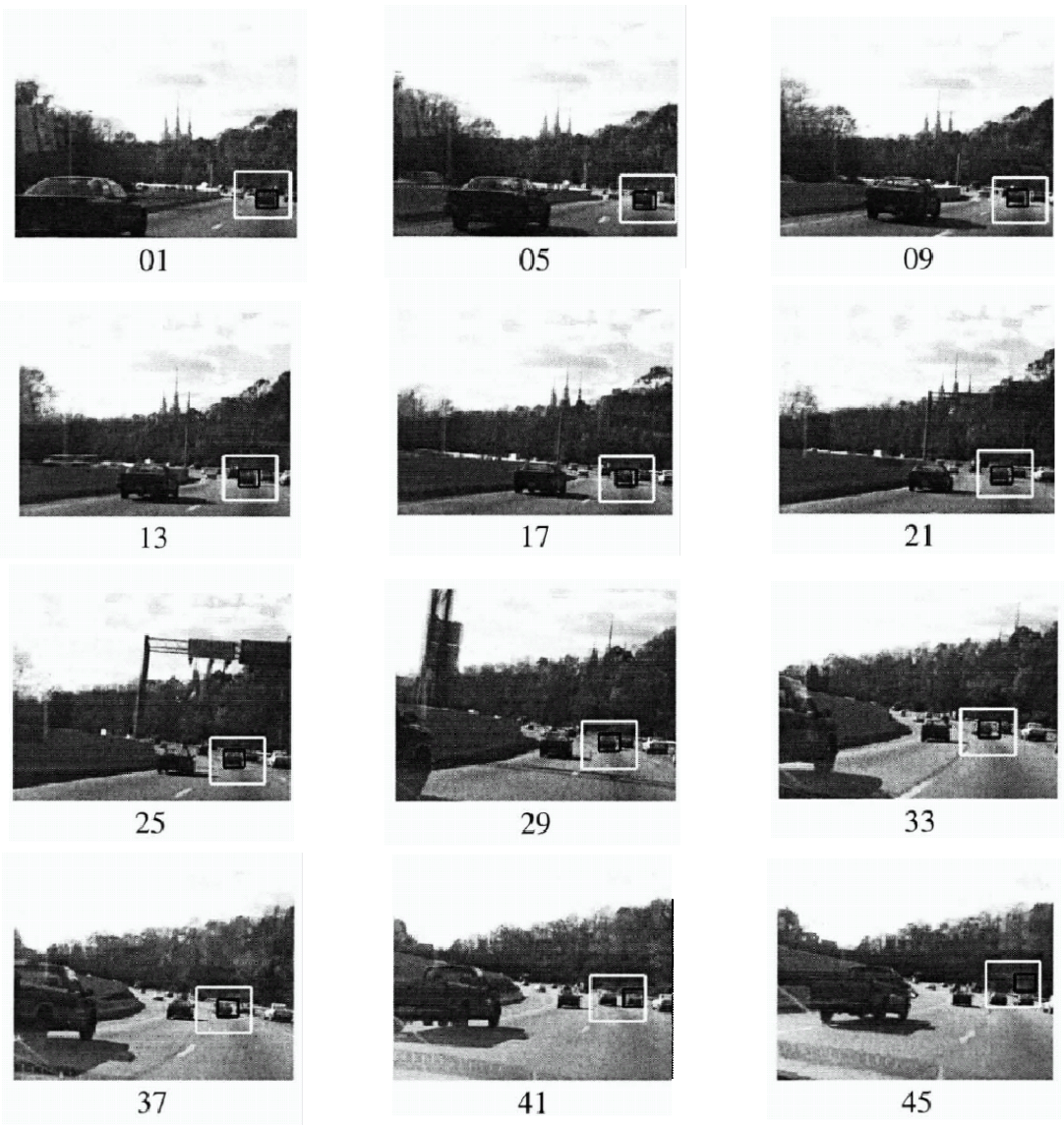


Figure 3.23: Some sample tracking images by Cross-Affinity. The dark rectangles represent the target positions, while the light (yellow) rectangles represent the search windows. The number of frame is shown below each image.

different distance measures. It is clear that the optimal cross-affinity measure leads to more robust tracking.

SSD	histogram	X-Affinity	X-Affinity(edge)	chamfer(edge)
13	13	41	20	16

Table 3.1: No. of consecutive frames tracked

3.6 Conclusions

In this chapter, we proposed a framework for learning the optimal feature distance measure to match pairs of images. It involves a parametric family of distance measures, called the cross-affinity distance measure, which is optimized to minimize image matching classification errors. Results on matching classification with a wide variety of image content show that the learnt feature distance measure clearly outperforms the standard measures of SSD, chamfer and Bhattacharyya histogram distances.

Chapter 4

Image Matching via Feature Correspondences

In this chapter, we deal with the image matching problem under heavy-distortion, correspondence-preserving transformations, which is illustrated in figure 4.1.

Transformations with correspondences is a large group of transformations where the point to point correspondences are preserved. They include scale change, rotation, affine and projective transformations, which are of substantial importance to computer vision because they are of the most frequently occurred transformations in vision problems, and it is challenging to cope with them. They also include any other image distortion where corresponding points can be found.

There are some related works that can deal with image matching under transformations with correspondences. One category of approaches make use of special interesting points that are detected in the images. The neighborhood of the interesting points are proved to be invariant to certain transformations, such as

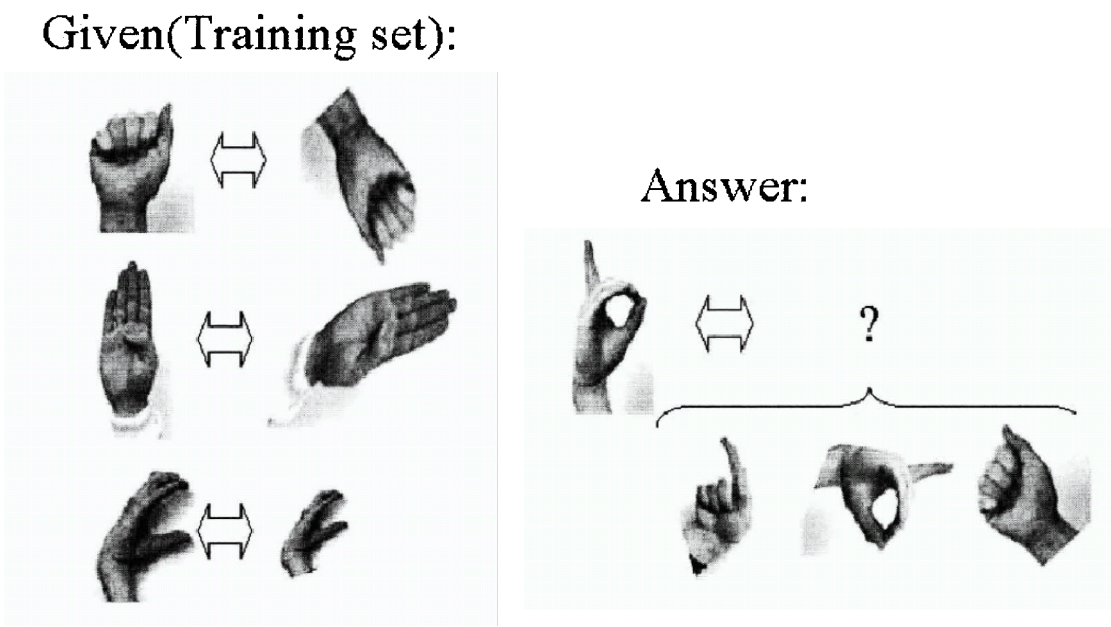


Figure 4.1: Matching under heavy-distortion, correspondence-preserving transformations . The problem addressed in this chapter is given training pairs of images that implicitly captures the transformation (with both positive and negative classes), identify if a new pair of test images is matched via the transformation class. The test images can have *totally different visual content* as compared to the training data.

SIFT [41] for scale and rotation transformation, the scale and affine invariant interesting point [42] for affine transformation. Such methods are efficient in identifying corresponding local image parts related by transformations, and the image matching problem could be solved through identifying multiple corresponding local parts. However, not all images are guaranteed to include sufficient number of interesting points for reliable recognition, and using interesting points alone sacrifices the information contained in smooth contours in the image.

Another category of approaches uses the framework of deformable shape matching [8, 7]. Generally, these approaches first solve the correspondence problem between the two images, then an aligning transformation, such as thin plate spline transform, is estimated by the correspondences. Finally, the matching distance is the sum of the extent of the transform and the residual difference. These approaches require accurate solving of the correspondences, usually through information of local image appearance. However if the transformation changes the appearance of the image substantially, it becomes very difficult to recover correspondences.

In this chapter, we describe a framework that learns how image features change through transformations with correspondences, where these changes are encoded as probabilistic transformation descriptors. In situations where the transformation class is very large, the statistics of these descriptors, when considered as a whole, are insufficient to determine if two images match each other based on the features extracted. Instead, a method is proposed that automatically partitions the unknown transformation classes into subsets in which the statistics of these descriptors become more discriminative. The extended matching process thus involves a maximum-likelihood estimation of the unknown subset labels. An ad-

vantage of this approach is that although it need feature correspondences to learn the probabilistic transformation descriptors in the training stage, there is no need to find corresponding features during the image matching stage.

4.1 Transformation Representation by Feature Correspondence

4.1.1 Image Representation

In this chapter, we represent images as raw feature sets. Formally, we have a feature type set:

$$\mathcal{F} = \{\phi_1, \phi_2, \phi_3, ..., \phi_n\} \tag{4.1}$$

where ϕ_i, $1 \leqslant i \leqslant n$, denotes the feature types that can be extracted from an image.

An image G is represented as the set of features extracted from it:

$$S_G = \{f_{g_1}, f_{g_2}, f_{g_3}, ..., f_{g_m}\} \tag{4.2}$$

where f_{g_i} denotes the feature extracted from position g_i, and the number of features is m. In this chapter, we only use the feature type parameter to describe the extracted feature, and ignore the position information of the feature. For example, if it is written $f_{g_2} = \phi_5$, this means the feature extracted at position g_2 is of type ϕ_5.

As there is no exact information on location or spatial relationships of these features, it might be possible that images with same set of features but in different space configurations be confused. However, this problem may be reduced

by overlapping fragments at multiple scales so that the correct configuration is preferred, as shown in [63].

4.1.2 Probabilistic Transformation Descriptor (PTD)

Let image G be transformed into image G' by a transformation T, and we assume each feature f_{g_i} in G has a corresponding feature $f_{g'_i}$ in G'. In other words, f_{g_i} is transformed into $f_{g'_i}$ by T. We call f_{g_i} as the source feature, $f_{g'_i}$ as the corresponding feature, and call these two features as a corresponding feature pair in later parts of this chapter, as illustrated in figure 4.2. Intuitively, different T may cause different changes between corresponding feature pairs. For example, one transformation T_1 could change a source feature of type ϕ_1 into the corresponding feature of type ϕ_2, but another transformation T_2 may change the same source feature of type ϕ_1, into a different corresponding feature of type ϕ_3. A key idea is that the mapping of one source feature type into other corresponding feature types may be used as a transformation descriptor.

Formally, we define a Probabilistic Transformation Descriptor (PTD) as an $n \times n$ table, which is denoted as $\mathbf{D}$, as shown in figure 4.3(a). The PTD for describing the transformation T is denoted as $\mathbf{D}\{T\}$. The entries of $\mathbf{D}\{T\}$ are expressed as:

$$\mathbf{D}\{T\}_{kl} = P(f_{b'} = \phi_l \mid T, f_b = \phi_k) \ , \ 1 \leqslant k, l \leqslant n \tag{4.3}$$

where f_b is the source feature, and $f_{b'}$ is the corresponding feature. Therefore, $\mathbf{D}\{T\}_{kl}$ is the probability of finding the corresponding feature f'_b is of type ϕ_l, given that the source feature f_b is of type ϕ_k, and the transformation is T. This is illustrated in figure 4.3(b).

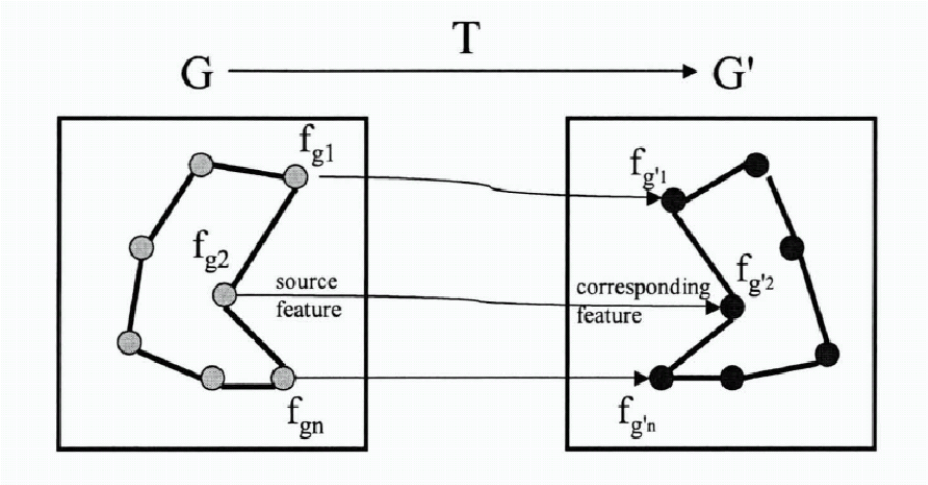
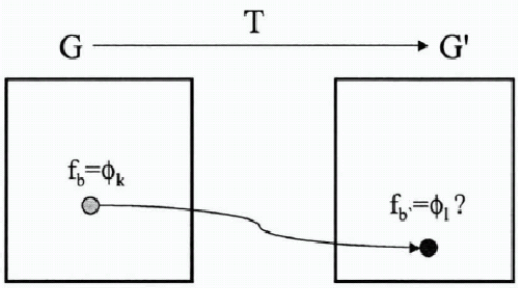


Figure 4.2: Illustration of corresponding feature pairs under transformation T. The transformation T transforms image G into G', and transforms source features into the corresponding features. Three of the corresponding feature pairs are shown connected by the lines.

PTD

D_{11}	D_{12}	D_{13}	...	D_{1n}
D_{21}	D_{22}	D_{23}	...	D_{2n}
D_{31}	D_{32}	D_{33}	...	D_{3n}
...	...	...	...	...
D_{n1}	D_{n2}	D_{n3}	...	D_{nn}

(a)



(b)

Figure 4.3: (a): A Probabilistic Transformation Descriptor (PTD) is an $n \times n$ table. (b): Each entry of the PTD, $D\{T\}_{kl}$ is the probability of finding the corresponding feature f'_b is of type ϕ_l, given that the source feature f_b is of type ϕ_k, and the transformation is T.

Suppose the parameters of transformation T can be denoted as a vector $\boldsymbol{\theta}_T$. If T is a specific transformation $\boldsymbol{\theta}_T = \boldsymbol{\theta}$, then the PTD for describing transformation T can be expressed as:

$$\mathbf{D}\{T\}_{kl} = P(f_{b'} = \phi_l \mid \boldsymbol{\theta}_T = \boldsymbol{\theta}, f_b = \phi_k) \ , \quad 1 \leq k, l \leq n \tag{4.4}$$

$\mathbf{D}\{T\}_{kl}$ is the probability of finding the corresponding feature f'_b is of type ϕ_l, given that the source feature f_b is of type ϕ_k.

More significantly, we can generalize T to not only represent a specific transformation, but a class of transformations, by considering the probability distribution over the parameters. In this case, $\boldsymbol{\theta}_T$ is a random variable, with a probability density function expressed as $P_{\boldsymbol{\theta}_T}(\cdot)$. The PTD for describing the transformation class T is expressed as:

$$\begin{aligned} & \mathbf{D}\{T\}_{kl} \\ = \ & P(f_{b'} = \phi_l \mid T, f_b = \phi_k) \\ = \ & \int P(f_{b'} = \phi_l \mid \boldsymbol{\theta}_T = \boldsymbol{\theta}, f_b = \phi_k) P_{\boldsymbol{\theta}_T}(\boldsymbol{\theta}_T = \boldsymbol{\theta}) d\boldsymbol{\theta} \ , \quad 1 \leq k, l \leq n \end{aligned} \tag{4.5}$$

Three example PTDs are shown in figure 4.4 (b), (c) and (d). The feature type set consists of 4 oriented edges features, as shown in figure 4.4(a). The transformation T is rotation counterclockwise of θ_T degree in image plane, θ_T is the transformation parameter. In figure 4.4(b), θ_T is a constant, while in (c) and (d), θ_T is a random variable.

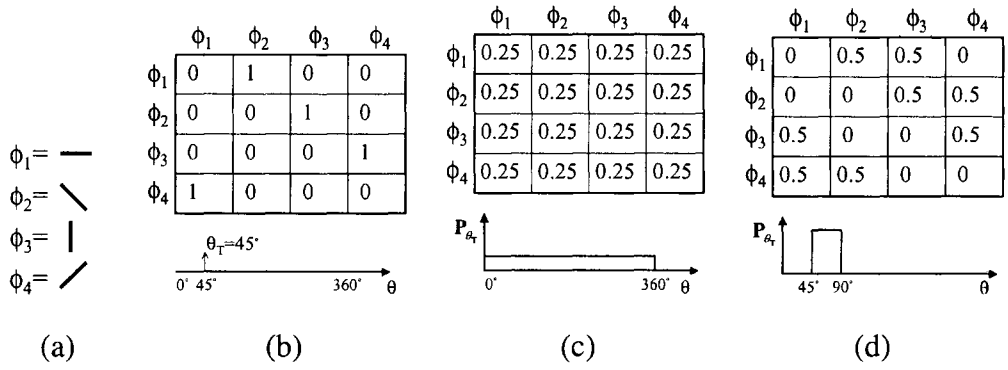


Figure 4.4: (a): The feature set is composed of 4 oriented edges. (b),(c),(d): PTDs describing transformations with different probabilistic distributions of parameters. The distributions are shown below each PTDs.

4.2 Approaches to Modeling Transformation Distributions by PTDs

A PTD can be used to predict how features will change under a class of transformations; however the predictive accuracy depends on the entropy of the transformation distribution. The smaller the entropy of the transformation distribution, the greater the predictive accuracy. This is because, as expressed by (4.5), the PTD for a transformation class is a marginal over the transformation parameters. For example, by considering figure 4.4(b), it is very certain that an image feature set $\{\phi_1, \phi_3\}$ will be changed into $\{\phi_2, \phi_4\}$, while in figure 4.4(d), the predictive accuracy is reduced. In the case of figure 4.4(c), the predictive accuracy goes to zero, because one feature can change into any other feature with equal probability.

The relations between PTDs and transformation distributions can be explained by figure 4.5, where each dot represents a specific parameter vector representing a single transformation, while a circle containing multiple dots represents a trans-

formation class whose range of parameters includes those dots. In the situation shown by figure 4.5(a), all parameter vectors are encapsulated within a single transformation distribution (the big circle) described by one PTD. The shortcoming of this PTD is the reduced accuracy of prediction. In contrast, figure 4.5(c) shows the situation where each individual transformations is described by its own PTD. In theory, these PTDs are most accurate in prediction, but it may be impossible to learn sufficient PTDs from training data to address our image matching problem. This is because while each PTD is a statistical description of its individual transformation, there is no generalization across different PTDs, resulting in insufficient modeling of the overall transformation distribution. If this approach is taken, an infeasibly large number of samples (pairs of training images) is needed to be accurate enough. This is akin to an overfitting problem where gaps across data samples are not adequately modeled. The situation depicted in figure 4.5(b) is a compromise between situations (a) and (c), where the whole range of transformations is partitioned into multiple smaller sub-ranges. Each transformation subclass with parameters distributed in one sub-range is depicted by its own PTD, and could be learned from a number of samples in that sub-range.

The examples in figure 4.4(c), (d) and (b) corresponds with the situations in figure 4.5(a), (b) and (c) respectively.

One limitation of PTDs is that the set of features should be relatively small rather than very large (*e.g.* several thousand features formed by VQ on image patches), since it is unlikely to have sufficient training data to populate a $N \times N$ table when N is very large.

In next section, a method is proposed to learn PTDs by automatically partitioning the overall transformation distribution into multiple smaller transforma-

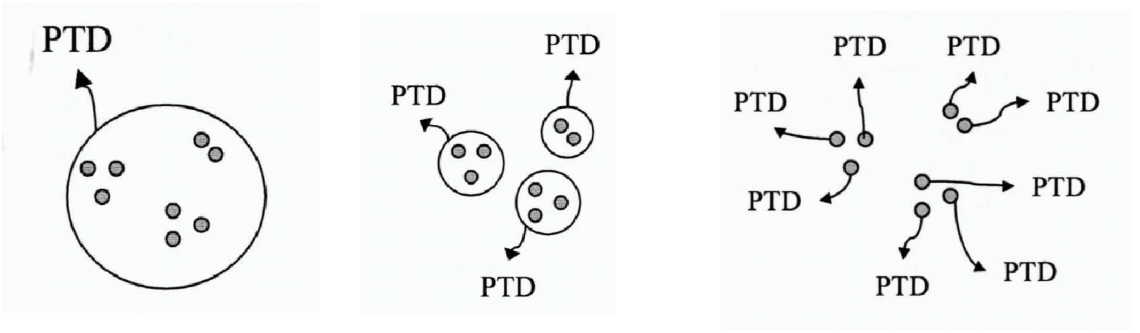


Figure 4.5: The relations between PTDs and ranges of transformation distributions. See text for explanation.

tion distributions.

4.3 Learning PTDs from Training Data

If an explicit transformation model is available, the different smaller transformation distributions may be defined by specifying sub-ranges on the transformation parameters. In this thesis, we avoid the use of an explicit transformation model, because we contend that real-world transformations in images are often too complex to model adequately, except with strong contextual knowledge and highly constrained scenarios. Instead, we implicitly define transformations through a training set composed of matched or mismatched image pairs with varying visual content. In this section, we will propose a novel strategy for transformation distribution partitioning.

4.3.1 Transformation Partitioning by Feature Change

The relationship of feature types across image transformations can be used as an approximate *index* to map to transformation distributions of different sub-ranges. For instance, if the change of feature ϕ_i into ϕ_j is a high probability characteristic of a set of transformations, then the observation of such a feature change leads to a high likelihood inference that the transformation belongs to this set.

We use the following strategy to learn PTDs from the training data. The training data is composed of many matched image pairs, where in each image pair, all the corresponding feature pairs are known. Suppose the number of matched image pairs in the training set is R, the transformation between the r-th ($1 \leq r \leq R$) image pair is T_r, and the parameter vector of T_r is denoted as $\boldsymbol{\theta}_r$, which may be unknown. Then the original training transformation distribution $P_{\boldsymbol{\theta}_T}(\cdot)$ can be expressed as:

$$P_{\boldsymbol{\theta}_T}(\boldsymbol{\theta}_T) = \begin{cases} \frac{1}{R} & \text{if } \boldsymbol{\theta}_T = \boldsymbol{\theta}_r \ , \ 1 \leq r \leq R \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

Suppose $(f_a, f_{a'})$ and $(f_b, f_{b'})$ to be two corresponding feature pairs that occur within a single image pair in the training set. Then we can reliably estimate the following set of conditional probabilities, if the size of training set is sufficiently large:

$$P(f_{b'} = \phi_l | f_a = \phi_i, f_{a'} = \phi_j, f_b = \phi_k) \ , \quad 1 \leq i, j, k, l \leq n \tag{4.7}$$

This is the probability of finding the corresponding feature f'_b is of type ϕ_l, given that the source feature f_b is of type ϕ_k, and further conditioned on having observed another corresponding feature pair $f_a = \phi_i$ and $f_{a'} = \phi_j$. This is illustrated in figure 4.6.

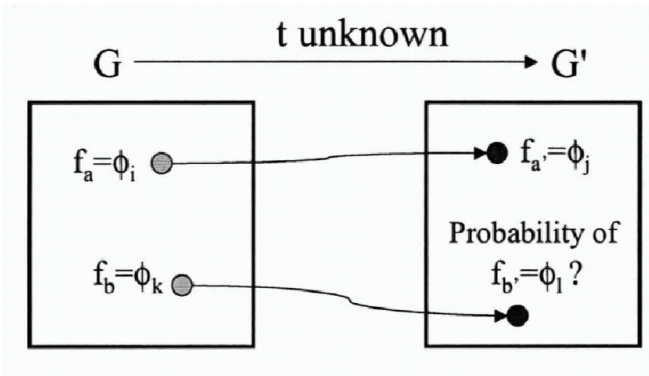


Figure 4.6: Illustration of the probability in (4.7)

These set of conditional probabilities can be derived by:

$$\begin{aligned} & P(f_{b'} = \phi_l | f_a = \phi_i, f_{a'} = \phi_j, f_b = \phi_k) \\ = & \frac{P(f_a = \phi_i, f_{a'} = \phi_j, f_b = \phi_k, f_{b'} = \phi_l)}{P(f_a = \phi_i, f_{a'} = \phi_j, f_b = \phi_k)} \\ = & \frac{P(f_a = \phi_i, f_{a'} = \phi_j, f_b = \phi_k, f_{b'} = \phi_l)}{\sum_{s=1}^{n} P(f_a = \phi_i, f_{a'} = \phi_j, f_b = \phi_k, f_{b'} = \phi_s)} \end{aligned} \tag{4.8}$$

where each joint probability above can easily be learned from the training data by definition.

For brevity of notation, we define:

$$C_{ij} \triangleq (f_a = \phi_i, f_{a'} = \phi_j) \ , \quad 1 \leq i, j \leq n \tag{4.9}$$

where C_{ij} denotes the combined condition $(f_a = \phi_i, f_{a'} = \phi_j)$ in (4.7). Then we

have:

$$
\begin{aligned}
& P(f_{b'} = \phi_l | f_a = \phi_i, f_{a'} = \phi_j, f_b = \phi_k) \\
= \; & P(f_{b'} = \phi_l | C_{ij}, f_b = \phi_k) \\
= \; & \sum_{r=1}^{R} P(f_{b'} = \phi_l | \boldsymbol{\theta}_T = \boldsymbol{\theta}_r, f_b = \phi_k) P(\boldsymbol{\theta}_T = \boldsymbol{\theta}_r | C_{ij}) && (4.10) \\
= \; & \sum_{r=1}^{R} P(f_{b'} = \phi_l | \boldsymbol{\theta}_T = \boldsymbol{\theta}_r, f_b = \phi_k) P_{\boldsymbol{\theta}_T}^{C_{ij}}(\boldsymbol{\theta}_T = \boldsymbol{\theta}_r) && (4.11) \\
= \; & P(f_{b'} = \phi_l | T_{C_{ij}}, f_b = \phi_k) && (4.12) \\
= \; & \mathbf{D}\{T_{C_{ij}}\}_{kl}, \; 1 \leq k, l \leq n && (4.13)
\end{aligned}
$$

where $T_{C_{ij}}$ denotes a sub-class of the overall transformation distribution subject to the existence of an observed condition C_{ij}, while $P_{\boldsymbol{\theta}_T}^{C_{ij}}(\boldsymbol{\theta}_T)$ is the conditional density of the transformation parameters upon having observed a corresponding feature pair $f_a = \phi_i$ and $f_{a'} = \phi_j$.

The relationship between the sub-range transformation distribution $P_{\boldsymbol{\theta}_T}^{C_{ij}}(\cdot)$, and the original training transformation distribution $P_{\boldsymbol{\theta}_T}(\cdot)$ is illustrated in figure 4.7. The idea is that $P_{\boldsymbol{\theta}_T}^{C_{ij}}(\cdot)$ will have smaller entropy and more well-defined peaks.

4.3.2 Analysis of the Transformation Partitioning

In the following, we show the relation between $\mathbf{D}\{T_{C_{ij}}\}$ and all the PTDs for the individual transformations in the training set $\mathbf{D}\{T_r\}$, $1 \leq r \leq R$

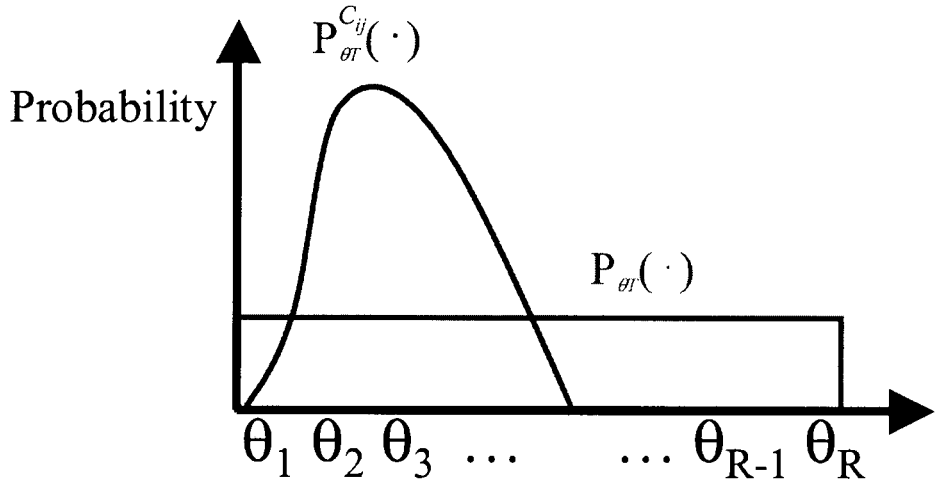


Figure 4.7: The relationship between the sub-range transformation distribution $P^{C_{ij}}_{\boldsymbol{\theta}_T}(\cdot)$ and the original training transformation distribution $P_{\boldsymbol{\theta}_T}(\cdot)$

In (4.10) we already have

$$\begin{aligned} & \mathbf{D}\{T_{C_{ij}}\}_{kl} \\ = \; & P(f_{b'} = \phi_l | T_{C_{ij}}, f_b = \phi_k) \\ = \; & \sum_{r=1}^{R} P(f_{b'} = \phi_l | \boldsymbol{\theta}_T = \boldsymbol{\theta}_r, f_b = \phi_k) P(\boldsymbol{\theta}_T = \boldsymbol{\theta}_r | C_{ij}) \end{aligned} \tag{4.14}$$

We further evaluate the term

$$\begin{aligned} & P(\boldsymbol{\theta}_T = \boldsymbol{\theta}_r | C_{ij}) \\ = \; & P(\boldsymbol{\theta}_T = \boldsymbol{\theta}_r | f_a = \phi_i, f_{a'} = \phi_j) \\ = \; & \frac{P(\boldsymbol{\theta}_T = \boldsymbol{\theta}_r, f_a = \phi_i, f_{a'} = \phi_j)}{P(f_a = \phi_i, f_{a'} = \phi_j)} \\ = \; & \frac{P(f_{a'} = \phi_j | \boldsymbol{\theta}_T = \boldsymbol{\theta}_r, f_a = \phi_i) P(\boldsymbol{\theta}_T = \boldsymbol{\theta}_r, f_a = \phi_i)}{P(f_{a'} = \phi_j | f_a = \phi_i) P(f_a = \phi_i)} \qquad (4.15) \\ = \; & \frac{P(f_{a'} = \phi_j | \boldsymbol{\theta}_T = \boldsymbol{\theta}_r, f_a = \phi_i) P(\boldsymbol{\theta}_T = \boldsymbol{\theta}_r) P(f_a = \phi_i)}{\sum_{s=1}^{R} P(f_{a'} = \phi_j | \boldsymbol{\theta}_T = \boldsymbol{\theta}_s, f_a = \phi_i) P(\boldsymbol{\theta}_T = \boldsymbol{\theta}_s) P(f_a = \phi_i)} \qquad (4.16) \\ = \; & \frac{P(f_{a'} = \phi_j | \boldsymbol{\theta}_T = \boldsymbol{\theta}_r, f_a = \phi_i)}{\sum_{s=1}^{R} P(f_{a'} = \phi_j | \boldsymbol{\theta}_T = \boldsymbol{\theta}_s, f_a = \phi_i)} \qquad (4.17) \end{aligned}$$

Here we are assuming $P(f_a = \phi_i|\boldsymbol{\theta}_T = \boldsymbol{\theta}_r) = P(f_a = \phi_i)$, because the probability of the type of the source feature does not depends on the transformation parameter. We also use the assumption from (4.6) that $P(\boldsymbol{\theta}_T = \boldsymbol{\theta}_i) = \frac{1}{R}$, $1 \leq i \leq R$.

Now combining (4.14) and (4.17), we get:

$$\mathbf{D}\{T_{C_{ij}}\}_{kl} \tag{4.18}$$

$$= \sum_{r=1}^{R} P(f_{b'} = \phi_l|\boldsymbol{\theta}_T = \boldsymbol{\theta}_r, f_b = \phi_k)\left(\frac{P(f_{a'} = \phi_j|\boldsymbol{\theta}_T = \boldsymbol{\theta}_r, f_a = \phi_i)}{\sum_{s=1}^{R} P(f_{a'} = \phi_j|\boldsymbol{\theta}_T = \boldsymbol{\theta}_s, f_a = \phi_i)}\right)$$

$$= \sum_{r=1}^{R} \mathbf{D}\{T_r\}_{kl}\left(\frac{\mathbf{D}\{T_r\}_{ij}}{\sum_{s=1}^{R} \mathbf{D}\{T_s\}_{ij}}\right) \tag{4.19}$$

where $\mathbf{D}\{T_r\}$ is the PTD for the transformation T_r expressed by the transformation parameter $\boldsymbol{\theta}_r$ such that

$$\mathbf{D}\{T_r\}_{kl} = P(f_{b'} = \phi_l|\boldsymbol{\theta}_T = \boldsymbol{\theta}_r, f_b = \phi_k) \tag{4.20}$$

Therefore

$$\mathbf{D}\{T_{C_{ij}}\} = \sum_{r=1}^{R} \mathbf{D}\{T_r\}\left(\frac{\mathbf{D}\{T_r\}_{ij}}{\sum_{s=1}^{R} \mathbf{D}\{T_s\}_{ij}}\right) \tag{4.21}$$

$$= \sum_{r=1}^{R} w_r \mathbf{D}\{T_r\} \tag{4.22}$$

where w_r is the term in parentheses in (4.21). We can see that $\mathbf{D}\{T_{C_{ij}}\}$ is a weighted sum of all $\mathbf{D}\{T_r\}$, and the weights are the relative probabilities that T_r will change feature ϕ_i into ϕ_j. If only a few of T_r can change ϕ_i into ϕ_j with high probability, then $\mathbf{D}\{T_{C_{ij}}\}$ is the weighted average of the few $\mathbf{D}\{T_r\}$.

The number of different conditions C_{ij}, $(1 \leq i, j \leq n)$ is n^2, so the number of PTDs learned is potentially very large. However, many different conditions C_{ij} may capture similar transformation classes and therefore lead to similar PTDs, because different feature changes may result from the same transformation. Take

for example the feature set in figure 4.4(a): the change from feature ϕ_1 into ϕ_2, and the change from feature ϕ_3 into ϕ_4 may result from an identical rotation, therefore in this particular case of rotation, $\mathbf{D}\{T_{C_{12}}\}$ and $\mathbf{D}\{T_{C_{34}}\}$ should be the same. So we can further classify the n^2 PTDs into $K(K \ll n^2)$ clusters, and use K PTDs to represent the overall transformation distribution. We denote the K representative PTDs as $\mathbf{D}\{T_{C_k}\}, k = 1$ to K in later parts.

4.4 Image Matching by Learned PTDs

With the K PTDs representing K different sub-range transformation distributions, we can match two images in a maximum likelihood scheme. Suppose S_G is the feature set of image G, the number of features is m, as shown in (4.2). Similarly, S_H is the features set of image H, the number of features in S_H is w:

$$S_H = \{f_{h_1}, f_{h_2}, f_{h_3}, ..., f_{h_w}\} \tag{4.23}$$

Suppose h is a random variable that takes the value of $h_i, 1 \leq i \leq w$, with equal probability, *i.e.*

$$P(h = h_i) = \frac{1}{w} \tag{4.24}$$

and f_h denotes the feature at position h. Here f_h is also a random variable that takes a value from the feature set $\mathcal{F}$. The distribution of f_h is :

$$\begin{aligned} & P(f_h = \phi_i) \\ = & \sum_{u=1}^{w} P(f_{h_u} = \phi_i) P(h = h_u) \\ = & \frac{1}{w} \sum_{u=1}^{w} \delta(f_{h_u} - \phi_i) \end{aligned} \tag{4.25}$$

where $\delta(.)$ is the Dirac's delta function, and $1 \leq i \leq n$. Note that $P(f_h)$ is identical to the normalized histogram of feature types in G.

Suppose S_Y is a virtual feature set transformed from S_G by the k-th sub-range transformation distribution described by PTD $\mathbf{D}\{T_{C_k}\}$, and the position y_u is the corresponding position of g_u:

$$S_Y = \{f_{y_1}, f_{y_2}, f_{y_3}, ..., f_{y_m}\} \tag{4.26}$$

then $P(f_{y_u} = \phi_i)$ can be evaluated by:

$$\begin{aligned} & P(f_{y_u} = \phi_i) \\ = \; & P(f_{b'} = \phi_i | \mathbf{D}\{T_{C_k}\}, b = g_u) \\ = \; & P(f_{b'} = \phi_i | \mathbf{D}\{T_{C_k}\}, f_b = f_{g_u}) \\ = \; & \mathbf{D}\{T_{C_k}\}_{si} \end{aligned} \tag{4.27}$$

for $1 \leq u \leq m$ and $1 \leq i \leq n$. Where s is an index, the feature type of f_{g_u} is ϕ_s. Suppose y is a random variable that takes the value of y_u, $i \leq u \leq m$, with equal probability, *i.e.*

$$P(y = y_u) = \frac{1}{m} \tag{4.28}$$

Then

$$\begin{aligned} P(f_y = \phi_i) &= \sum_{u=1}^{m} P(f_{y_u} = \phi_i) P(y = y_u) \\ &= \frac{1}{m} \sum_{u=1}^{m} P(f_{b'} = \phi_i | \mathbf{D}\{T_{C_k}\}, f_b = f_{g_u}) \end{aligned} \tag{4.29}$$

for $1 \leq i \leq n$. If H is really transformed from G by the sub-range transformation distribution described by $\mathbf{D}\{T_{C_k}\}$, then according to the model, the probability distribution $P(f_y)$ should be similar with $P(f_h)$, so the χ^2 distance between $P(f_y)$ and $P(f_h)$:

$$d_{GH}(\mathbf{D}\{T_{C_k}\}) = \chi^2(P(f_y), P(f_h)) \tag{4.30}$$

should be small. Thus $d_{GH}(\mathbf{D}\{T_{C_k}\})$ is a likelihood function for the event that H is transformed from G through C_k. The eventual distance between G and H is:

$$D_{GH} = \min_k d_{GH}(\mathbf{D}\{T_{C_k}\}) \tag{4.31}$$

If D_{GH} is less than a threshold, then G and H is classified as matched.

4.5 Experiments

We conduct image matching experiments Corel Stock Photo Library2 dataset, using 69 features learned from the training set. The transformation is synthesized projective transformation.

4.5.1 Construction of Synthesized Image Pairs

Corel Stock Photo Library2 includes 186 categories of photos, such as African Wildlife, Autumn in Maine and Freestyle Skiing, *etc.* Each category has 100 color images of size 768×512 or 512×768 , with different contents.

We randomly selected 20 categories for the training data, and another 20 categories for testing data. For each category, we have 10 images, so there are 200 training images and 200 test images. All the images are first resized to 25% of original size, *i.e.* 192×128 or 128×192 by bilinear interpolation, and changed to gray-scale images. Then the center part of size 96×96 is cut from each image as training and testing images, so that the size of all images in the experiment data set are the same. Some of the images in the experiment data set are shown in figure 4.8, column (a).

We constructed training and testing image pairs by synthesized projective transformation to simplify the task of finding corresponding features in the train-

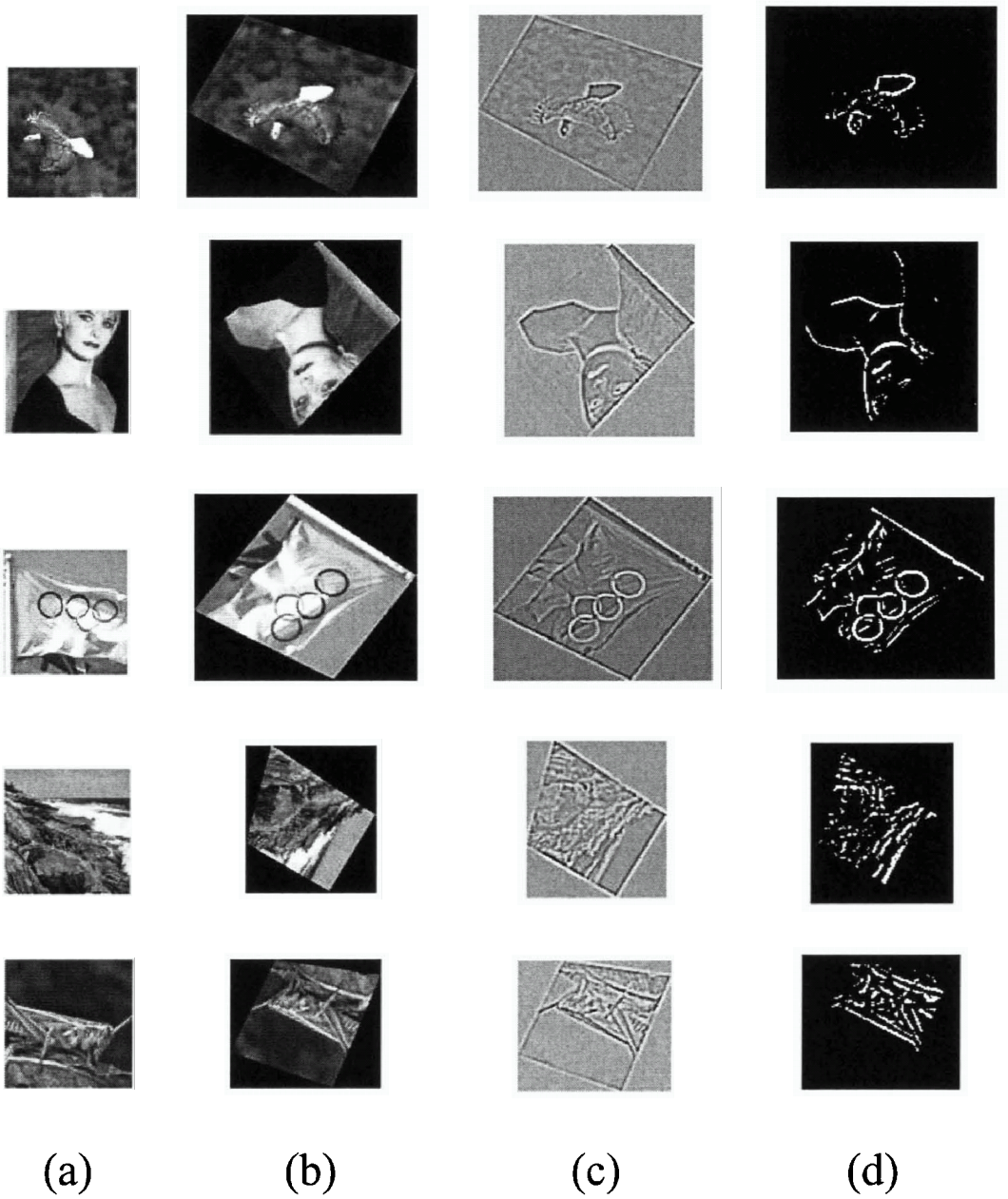


Figure 4.8: (a): sample images from training and testing dataset, they are gray-scale images of size 96×96. (b): The synthesized corresponding images by random projective transformations. The scales of the images are retained in the figure.

ing process. The image pairs are constructed from the 400 training and testing images. For each image, we first made a random projective transformation expressed by a set of random parameters, then applied the transformation to the image to make a corresponding image. The original image and the corresponding image form a matched image pair.

Each random projective transformation is actually defined by 4 corresponding point pairs. The 4 original points are (50,50), (50,-50), (-50,50) and (-50,-50). For each original point, the corresponding point is randomly selected within a $20\times$ 20 square area centered at the original point. After all 4 corresponding points are selected, the corresponding points are further randomly rotated within 0°–360°. The selected transformation is the projective transformation defined by the final 4 corresponding point pairs. The process of making random projective transformations is illustrated in figure 4.9.

For each original image, we created 10 matched image pairs by 20 random projective transformations. So for either the training set and the test set, we have 2000 positive (matched) image pairs. For each original image, we also made 20 unmatched image pairs, by randomly selecting 20 different original images and random transformations. So for either the training set and the test set, we have 4000 negative (mismatched) image pairs. Some example corresponding images are shown in figure 4.8, column (b). They are the corresponding images of those in column (a).

4.5.2 Feature Selection

The images features used in this experiment are 69 9×9 templates learned from the training data sets, as shown in figure 4.10. To reduce the number of features

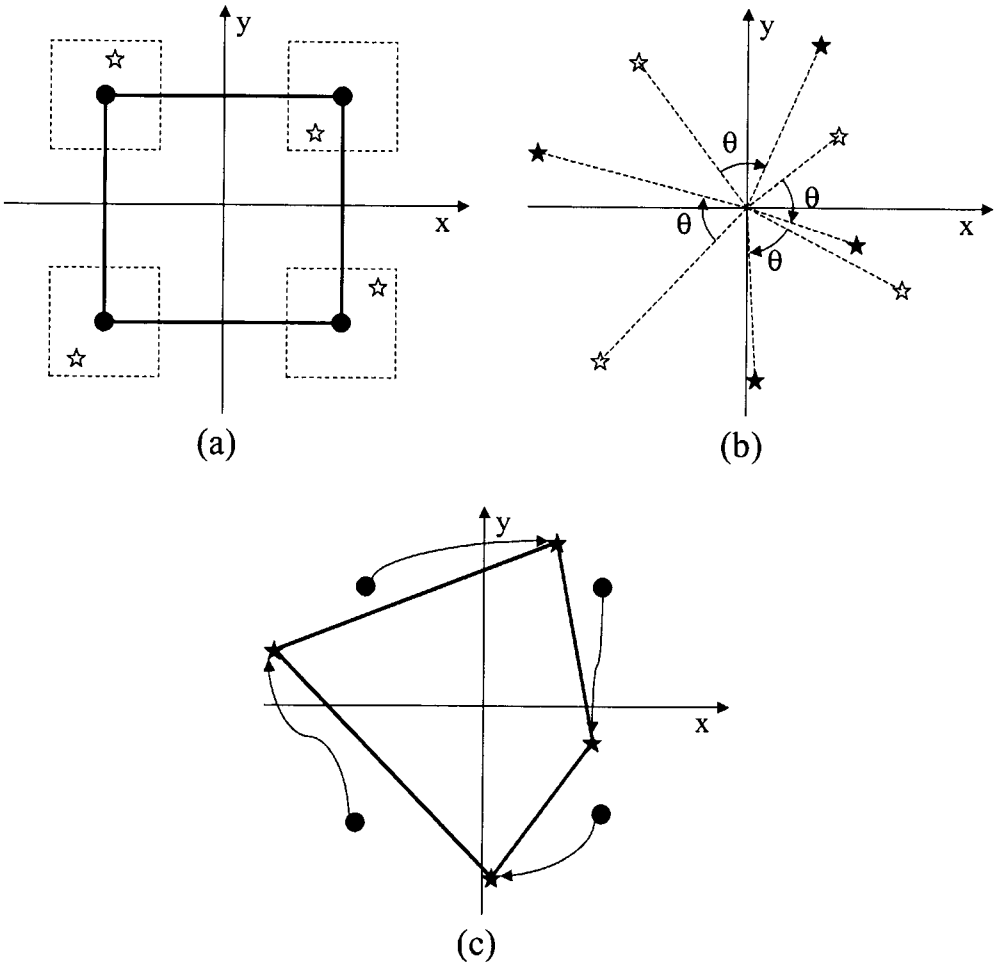


Figure 4.9: The process of making random projective transformations. (a): Four random corresponding points(the hollow stars) for four reference points (the round dots)are selected within each square area. (b): The corresponding points are further rotated for a random degree θ, to new positions(filled stars). (c): The projective transformation is defined by the 4 corresponding point pairs.

extracted in matching process, as well as for noise reduction, all the images are first filtered by a 5×5 LoG filter, with $\sigma = 1$, the results are shown in figure 4.8, column (c). Only responses higher than a certain threshold, which is set at 10 in the experiment, are considered interesting points, as shown in figure 4.8, column (d). All the 9×9 patches centered at these interesting points in the training image pairs are extracted and clustered by k-means algorithm with $K = 100$, the resulting 100 cluster centers (templates) are further refined so that if two templates are sufficiently similar, one of them is removed. Finally, there are 69 templates selected, which are showed in figure 4.10. In the image matching process, each interesting point indicates the location of a feature, the patch centered at the interesting points is compared with all the feature templates by Normalized Cross Correlation, and the feature type is decided by the closest feature template.

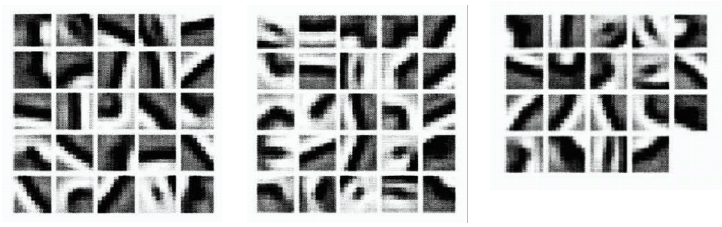


Figure 4.10: The 69 feature templates used in the experiment.

4.5.3 Learning and Matching Results

As there are 69 different features in the feature set, theoretically there are $69 \times 69 = 4761$ different PTDs that can be learned from the training process. However, most of the PTDs do not exist, or do not have sufficient samples to be learned because a large number of feature changes do not happen in the considered transformation class. For example, an oriented edge can change in direction as a result

of a projective transformation, but it will not transform into corners or circles. Hence the PTD expressing the condition "an edge changes into a corner" will not exist. Furthermore, a PTD needs a sufficient number of samples to be accurately learned; so the PTDs with relatively small number of samples are also ignored in the training process. In this experiment, only 207 PTDs with most samples are selected from the pool of 4761 PTDs, and the 207 PTDs are further clustered into 100 classes by K-means algorithm. The 100 cluster PTDs are used for the image matching task. We have also tried to use more number of PTDs, but the performance do not improve accordingly. The reason may be that most PTDs actually represent similar transformation distributions, *e.g.* the PTD expressing the condition "0° edge change into 45° edge" is the same as the PTD expressing "45° edge change into 90° edge". Some sample cluster PTDs among the 100 are shown in figure 4.11, where dark color represents high value.

The ROC curves of the matching result are shown in figure 4.12. Our method is indicated as PTD in the figure. For comparison, we also learned the PTD for the original training transformation distribution of the training data, as expressed in (4.6). We used this single PTD for image matching, the result is marked as "single PTD".

Our method is different to most existing related works on image matching under transformations with correspondences in the following two points. Firstly, our method does not require a explicit transformation model such as scale or affine transformation. Secondly, there is no need to solve correspondences between test image pairs. With the above two constraints, we only compare our method with "Chi Square", which is the χ^2 distance of the normalized feature type histograms.

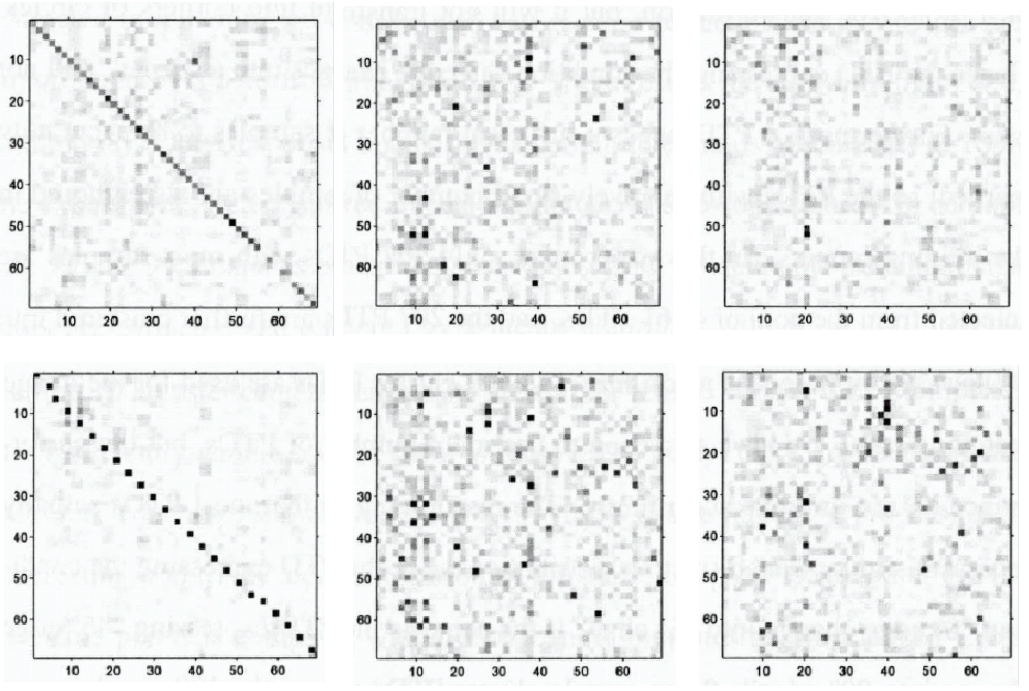


Figure 4.11: Six samples PTDs among the 100 PTDs used for image matching. Each PTD is a 69×69 table since there are 69 features in the feature set. The values of the elements are expressed by the grey level, where darker points corresponds to bigger value.

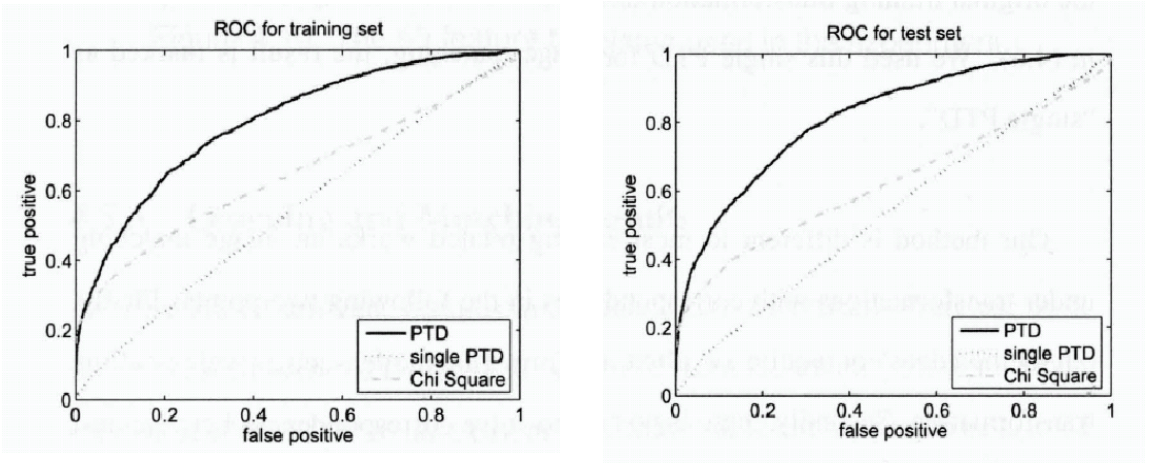


Figure 4.12: ROC curves

4.6 Conclusions

In this chapter we described a framework that copes with image matching problem under transformations with correspondences. It uses the statistics of feature changes under transformations, which is encoded in the Probabilistic Transformation Descriptor. The proposed method can automatically partition the original training transformation distribution into multiple sub-range transformation distribution, and the PTDs for the sub-range transformation distributions are more discriminant than the one for the original training transformation distribution. The results demonstrate how it is possible to successfully match an image to one which has undergone substantial transformation without the need for registration.

Chapter 5

Image Matching via Feature Co-occurrence

In this chapter, we will address the image matching problem under heavy-distortion, correspondence-breaking transformations, as illustrated in figure 5.1. This class of transformations can be extended to include non-structural relations, *e.g.* the "transformation" between the bottom half and the top half of a face, as illustrated in the bottom row of figure 5.2.

The task of *image matching*, as considered in the most generic context, involves determining whether a pair of images exhibit a special contextual relationship that distinguishes it from other random pairings of images. For example, human IQ quizzes regularly require the matching pairs of visual patterns that have undergone a particular spatial transformation, that is implicitly captured in other analogous pairs of *dissimilar* patterns provided as prior examples. In another example, children books and television programs often contain problems in which pairwise matching of visual images have to be performed based on more universally known contextual information, *e.g.* matching of two images that are parts of

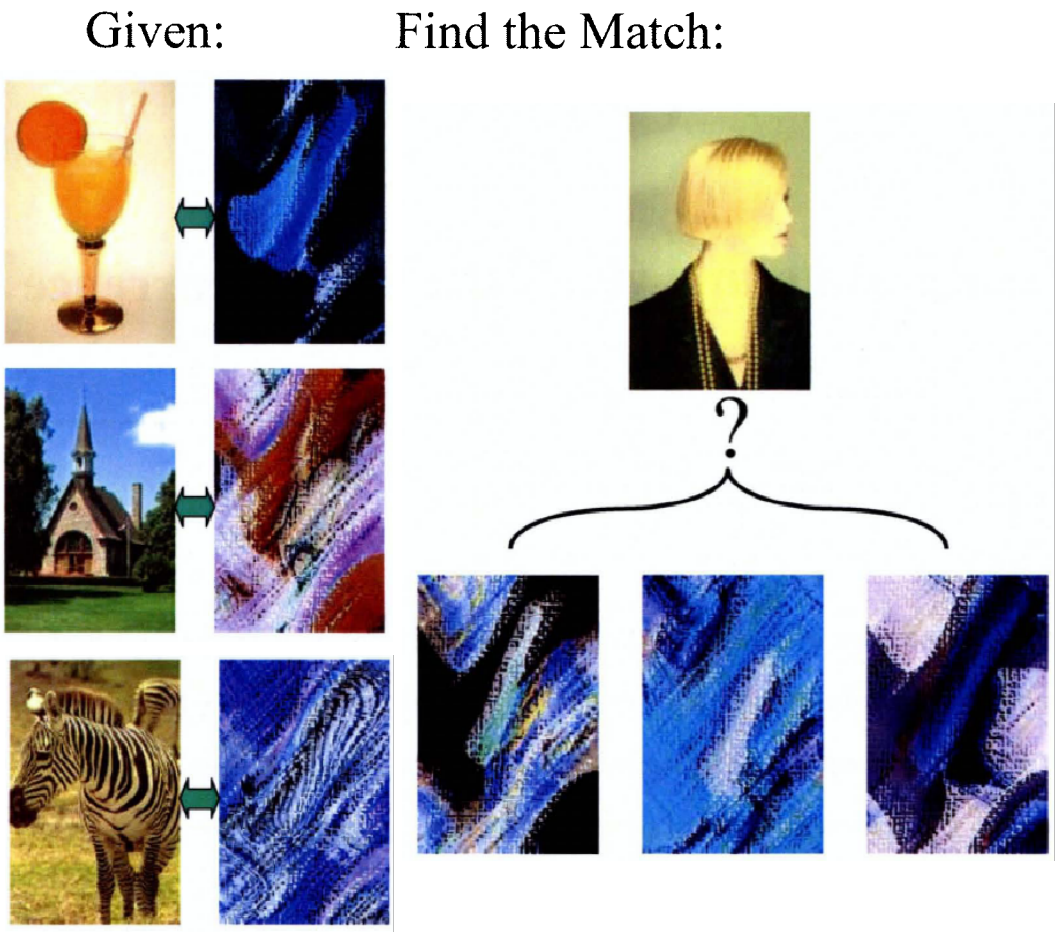


Figure 5.1: Matching under severe transformation. The problem addressed in this chapter is given training pairs of images that implicitly captures the transformation (with both positive and negative classes), identify if a new pair of test images is matched via the transformation class. The test images can have *totally different visual content* as compared to the training data.

the same object, as opposed to other entities.

In extending this generalized concept of image matching to machine vision, we want to highlight two key problems that are not considered in current methods:

- *Large differences of visual content between training data and actual/test data.* In image matching problems that may be cast into an object recognition framework, the goal is to recognize an object in the image, based on training data containing objects of a similar class. This presumes that the visual appearance or features of the object in the test image is adequately spanned by the training data. This will not be true of cases in figure 5.1.

- *Large differences of visual content between two (correctly) matched images.* Two images may have visual content that are entirely different from each other but nevertheless semantically related, possibly in a directional manner. For example in the bottom row of figure 5.2, there is an obvious relationship between the upper and bottom halves of a face, but the image structure is entirely different. Similarly, for the top row in figure 5.2, images are systematically related in the sense that the right image is the end product of filtering the left image through a fixed sequence of complex filters. However, in all instances, it is not only difficult to find pointwise correspondences, but correspondences do not even exist.

Most of existing methods do not address these problems (they were never intended to in the first place). For example, popular classification-based recognition methods depend on extensive training data. Nevertheless, there are numerous image matching problems that involve matching two images for which the visual content of the images do not belong to some prior known class. Such problems include image mosaicing, stereo correspondence and tracking with online acquired

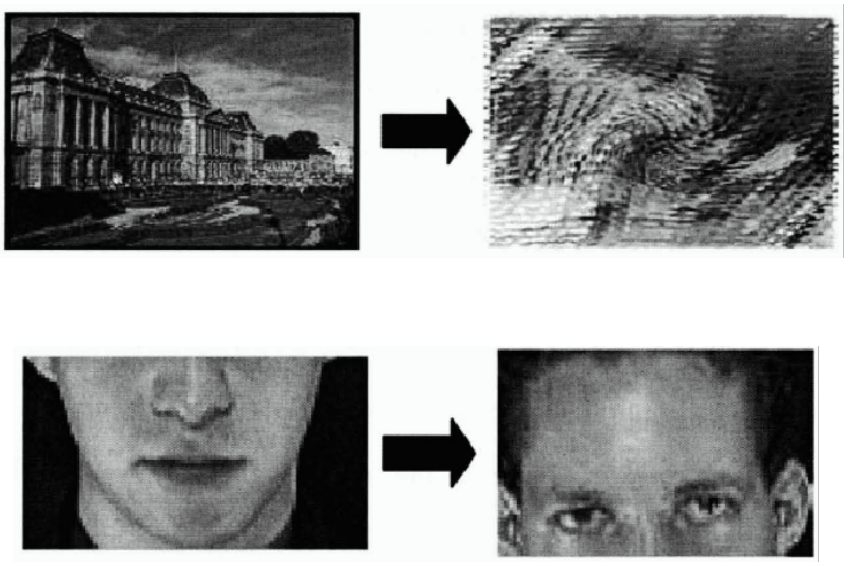


Figure 5.2: Large differences of visual content between "matched" images. Top row: this illustrates the result of the left original image undergoing a series of filters, resulting in the right image. Bottom row: Bottom half of a face is related to the top half. Current methods do not adequately deal with matching images with these forms of relation, particularly for test data which are also significantly biased as compared to the training data.

models. These problems do not readily admit the use of the classification approaches. As such, the matching of image pairs have traditionally been carried out using simple metrics such as sum-of-squared differences (SSD) or histogram matching, with the bulk of the research focused on robustly estimating warp parameters that bring image pairs into alignment.

A possible approach is to directly engineer solutions for each domain separately. For example, to solve the matching problem in affine transformation, one may attempt to use affine-invariant region detectors. The problem with such an approach is that explicit transformation models have to be assumed; further-

more, constraints on the range of transformations that are implicitly defined by the training data may be lost through this approach. More challenging would be the problem in figure 5.2, where no existing methods are able to address.

In this chapter, we make the following contributions:

- we articulate the problem of matching across severe, correspondence-breaking transformations as well as relations that do not preserve structure;
- provide a mechanism for matching test images which are visually very different from training data; and
- propose a unified framework for learning such transformations and relations directly from noisy training data, avoiding the need to engineer domain-specific models.

5.1 Related Work

In the past, standard measures for evaluating correspondences are the sum-of-squared pixel differences (SSD), standard metrics in feature space (*e.g.* Euclidean is used in the SIFT framework [41]), the chamfer distance [6], the shuffle distance [62], the cross-affinity family of distances [11] and histogram measures such as χ^2 distance, Bhattacharyya distance (as used in the mean-shift framework [14]), Kullback-Leibler divergence and Earth Mover's Distance [54]. These measures, while intuitive, fail when used for comparing images under heavy-distortion tranformations and non-similar relations as in figure 5.2.

Histograms of features has been successfully applied to image classification [23, 24], multimedia classification [43] and event recognition [69]. There are also methods that involve feature selection [4, 13]. However in most of these cases, the

requirement is that training data is available which has similar visual content (and will generate similar histograms) to the test cases. This requirement also applies to recent classifier-based methods for object recognition involving SVM [27, 65].

The image analogies work in Hertzmann *et al.* [28] is semi-related in that feature-wise relationships are involved. However, image analogies deal with image synthesis and require point-to-point correspondences, whereas this chapter deals image matching and the framework does not require correspondences, and even cater for matching between image structure so different that relevant correspondences may not exist.

5.2 Overview of Approach

At the heart of our proposed approach is the notion that images are composed of numerous small features as building blocks of the image. In spirit this is similar to existing approaches based on collections of sparse corner features, except that in our case, a fully dense and over-complete representation is used as discussed in section 5.4. Additionally, unlike these methods we do not attempt to characterize an image based on extracted features, since we are not interested in the images themselves *per se* – after all, the images in our test data is significantly different from those in the training data. Instead, our focus is on describing how features relate to other features *across* the relevant set of transformations or relations, where the set of relations is implicitly captured by training data provided.

A key assumption made in this chapter is that under a particular set of transformations or relations, certain features in one image are present in ***consistent ratios*** with other features in the second image, where these features may be of different types between the two images. While there is no exact basis for this

assumption, this is generally true for many transformations and relations. For example, consider a transformation comprising purely a 90° rotation followed by gray-scale inversion. Features corresponding to 45° dark edges will be present in equal quantity with −45° light edges across the transformation. Similarly, in a relation of upper faces to lower faces, there will be four eye-corner features for two mouth-corner features (note: this example is used illustratively and we do not explicitly search for such domain-specific features). Hence each new image, even with substantially different visual content from those previously observed, is simply "disassembled" into features. The frequency of each feature type (described later) may then be individually compared to frequencies of other feature types in the hypothesized matched image.

The main challenge of this approach is to determine which feature-to-feature ratios are systematically the result of the underlying transformation / relation, and which are attributed to incidental variations in image content, clutter and noise. Briefly, the process adopted in our framework is as follows:

1. Start with a positive training set comprising *pairs of matched images*.

2. For each feature type, measure the frequency of occurrence within each image.

3. Measure the frequency ratios between *all pairs* of feature types across each two matched images.

4. Quantize the frequency ratios and construct a normalized histogram of such ratios across all images in the positive training set.

5. Repeat with a negative training set comprising *pairs of mis-matched images*.

6. The differences between the histograms of the positive and negative test sets indicate the frequency ratios that are important in determining whether a test pair of images are matched.

The subsequent sections describe this process in more precise terms. Additionally, we also use the term "transformation" to describe both normal transformations as well as contextual pairwise relations such as two halves of a face.

5.3 Transformation Comparison

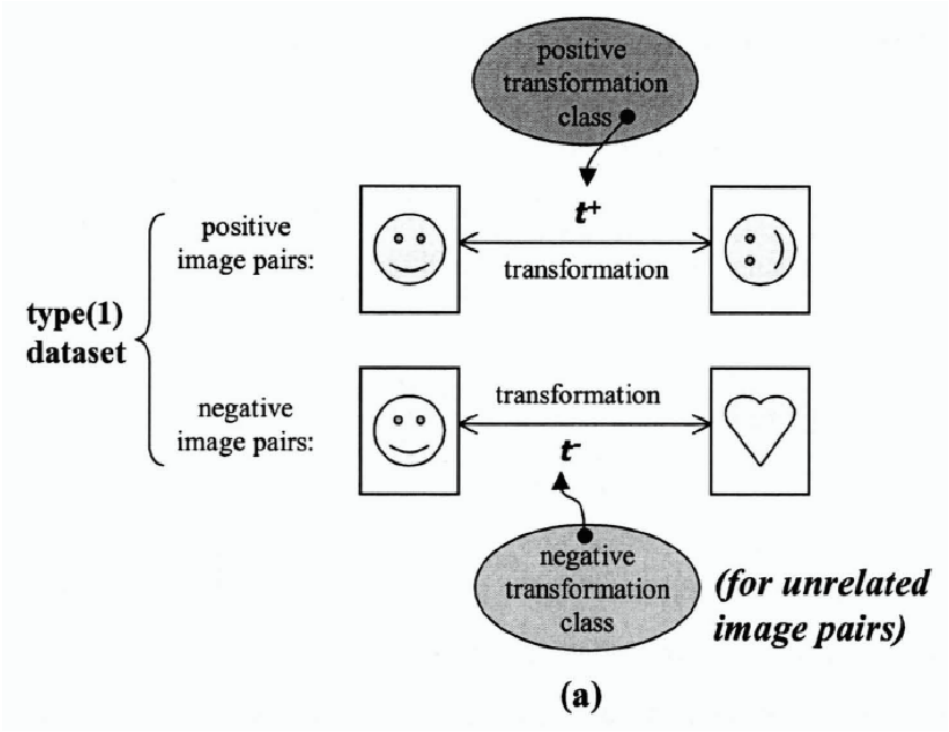
In this chapter, we consider each pair of images to fall into two classes:

- *Positive transformation class*, which expresses that the pair of images are satisfactorily matched under an implicitly-specified range of transformations and transformation parameters.
- *Negative transformation class*, where the pair of images do not match.

There are two differently constructed datasets:

- **Type (1) dataset**. The positive set is composed of image pairs related by transformation, and the negative set is composed of unrelated image pairs, possibly containing different visual content.
- **Type (2) dataset**. The positive set is composed of image pairs related by a subset of transformations, while negative set is composed by image pairs related by a different subset of transformations (*i.e.* the transformations in the positive set have a different range of parameters compared to those in the negative set).

These datasets are illustrated in figure 5.3.



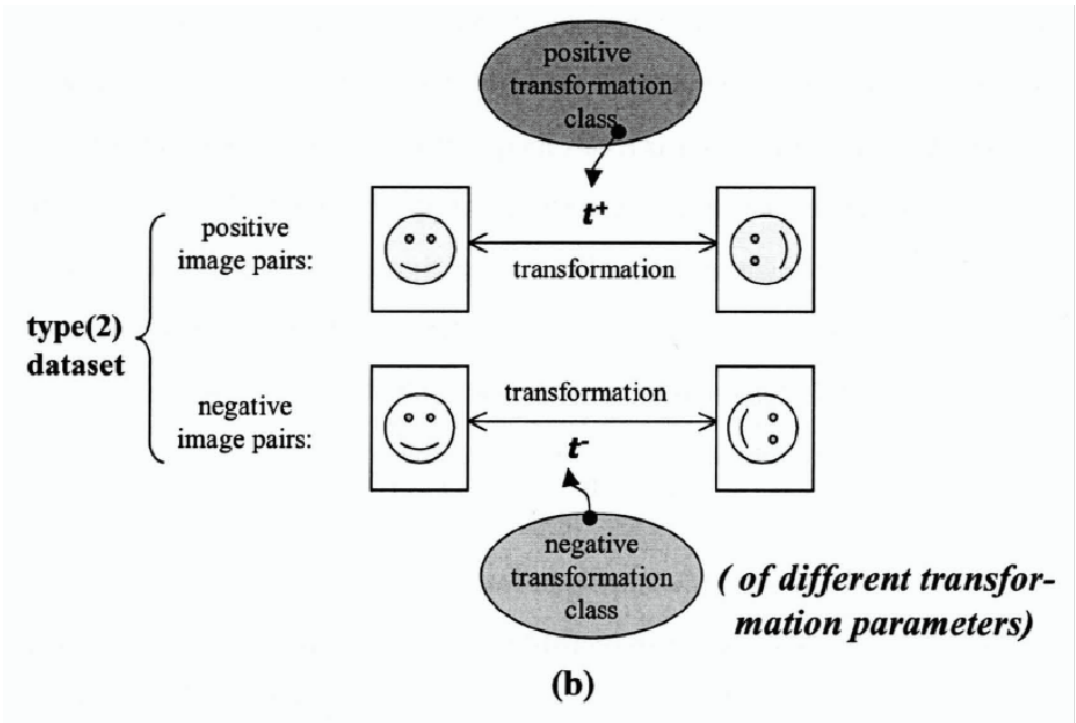


Figure 5.3: Illustration of two types of datasets. (a) for type(1) dataset; (b) for type(2) dataset.

5.4 Image Representation

In the framework of this chapter, images are represented as feature histograms, as described in Chapter 2, Section 2.1.2. However, as multiple features are used, the manipulation of the feature histograms is a little different, and they are described below.

A raw pixel image I is converted into a feature histogram:

$$I \mapsto \mathbf{h} = H(F(I)) \tag{5.1}$$

where h is an integer vector representing the histogram, $H(\cdot)$ is a histogramming operator and $F(\cdot)$ is a feature extraction operator.

However, instead of adopting the recent highly popular sparse representations (*e.g.* Grauman [23]), $F(\cdot)$ here involves extracting a fully-dense and over-complete feature set through convolution with a small filter bank. The size and type of filters in the filter bank can be varied according to different needs. For example, the filter bank may include Laplacian of Gaussian, first order derivatives of Gaussian and impulse filters. Each filter may be expressed as a histogram of quantized response strength, and the final image representation is the concatenation of the individual histograms. Similarly for the image with transformation:

$$\mathcal{T}(I) \mapsto \mathbf{h}' = H(F(\mathcal{T}(I))) \tag{5.2}$$

where $\mathcal{T}(\cdot)$ represent the unmodeled transformation.

In this way, each image may be considered as mapped to a fixed-length integer vector. In this representation, we consider each "feature type" of the image as the bin index of this histogram vector, and each "feature value" as the integer value of the associated bin.

5.5 Transformation Representation by Feature Co-Occurrence

While it is difficult to explicitly model complex real world transformations, these transformations may however be indirectly expressed through feature co-occurrence statistics that allow us to distinguish between different transformations. In this section, we will introduce the transformation representation by feature co-occurrence.

5.5.1 Feature Co-occurrence

Given two images I and I' which have been converted into histograms representations $\mathbf{h}$ and $\mathbf{h}'$ respectively, the problem remains on how to compare these images. In a classical approach, one might attempt to either learn $p(\mathbf{h}, \mathbf{h}'|T)$ where T is an label indicating if images are related via the transformation $I' = \mathcal{T}(I)$, or attempt to classify $(\mathbf{h}, \mathbf{h}')$ via SVMs. In either case, a large amount of training data is usually needed, and also presupposes that the test images appear similar to those in the training data. However, if the test images are significantly different, these approaches will generally perform poorly.

Instead, we adopt a component-based approach by assuming that the features of $h_1, h_2, \ldots, h_N$ in $\mathbf{h}$ *individually* represent abstract building blocks of images, and that new images are formed through the assembly of these features. This is analogous to the observation that words are the building blocks of documents. Indeed, word-based analysis has led to the use of measures in information-retrieval research such as tf-idf [55] that involves *term frequency* and *document frequency* in measuring the discriminative value of words for topical clustering of documents. Furthermore, text-based approaches have also been successfully applied

to image classification [48, 59, 47].

However in these previous approaches, these generalized "terms" are used to represent the *content* of the images, rather than the *transformation* which is what we are after. In order to extend the representation to represent transformations, we consider **pairwise ratios of cross-image features** as a measure of feature co-occurrence to describe the image transformation. In other words, given

$$\mathbf{h} = [h_1, h_2, \ldots, h_N] \quad \text{and} \quad \mathbf{h}' = [h'_1, h'_2, \ldots, h'_N], \tag{5.3}$$

the feature co-occurrence is expressed as the pairwise ratios of cross-image features:

$$r_{ij} = h_i / h'_j \qquad \text{for } 1 \leqslant i, j \leqslant N \tag{5.4}$$

5.5.2 Cross-feature Ratio Table (CRT)

Based on the feature co-occurrence, the image transformation may be represented in the form

$$\begin{aligned} \mathcal{T} \sim R &= \begin{bmatrix} h_1{:}h'_1, & \ldots, & h_1{:}h'_N \\ h_2{:}h'_1, & \ldots, & h_2{:}h'_N \\ & \cdots & \\ h_N{:}h'_1, & \ldots, & h_N{:}h'_N \end{bmatrix} \\ &= \begin{bmatrix} r_{11}, & \ldots, & r_{1N} \\ r_{21}, & \ldots, & r_{2N} \\ & \cdots & \\ r_{N1}, & \ldots, & r_{NN} \end{bmatrix} \end{aligned} \tag{5.5}$$

Here we can consider R as a $N \times N$ table, which we call the *Cross-feature Ratio Table* (CRT). The r_{ij} entries of the CRT effectively captures the co-occurrence

statistics of such pairs of features. An alternative interpretation is that these pairs of features are meta-terms, and the statistics capture the meta-term frequency within a pair of matched images. See figure 5.4 for a illustration of the process for creating the Cross-feature Ratio Table from a single pair of images.

The discriminative strength of different entries within the CRT are different. While some of the pairwise ratios are a direct consequence of the transformation, others may have resulted from incidental variations in image content, clutter and noise. In addition, if one of h_i or h'_i is zero or very small, then the corresponding cross-feature ratio can not be effectively captured, or too much vulnerable to noise.

Hence there is a need to determine the relative discriminative strength of the different ratios. This is covered in the following section.

5.6 Discriminative Analysis of Cross-Feature Ratios

The approach of this chapter to transformation comparison is to effectively determine which of the features the cross-feature ratios r_{ij} are useful in discriminating between the positive and negative transformation classes.

Suppose we are given positive and negative *image pairs* that form the training dataset. By computing the separate CRT's for these image pairs, we can compute the separate distributions for each of the CRT entries r_{ij} converted to fractional form:

$$H^+_{ij}(k) = p(\alpha(k) < r_{ij} \leqslant \alpha(k+1)|T) \tag{5.6}$$

$$H^-_{ij}(k) = p(\alpha(k) < r_{ij} \leqslant \alpha(k+1)|\neg T) \tag{5.7}$$

where $H^+_{ij}(k)$ and $H^-_{ij}(k)$ are normalized histogram (probability distribution) functions for the positive and negative set respectively, k is the index to a set of quan-

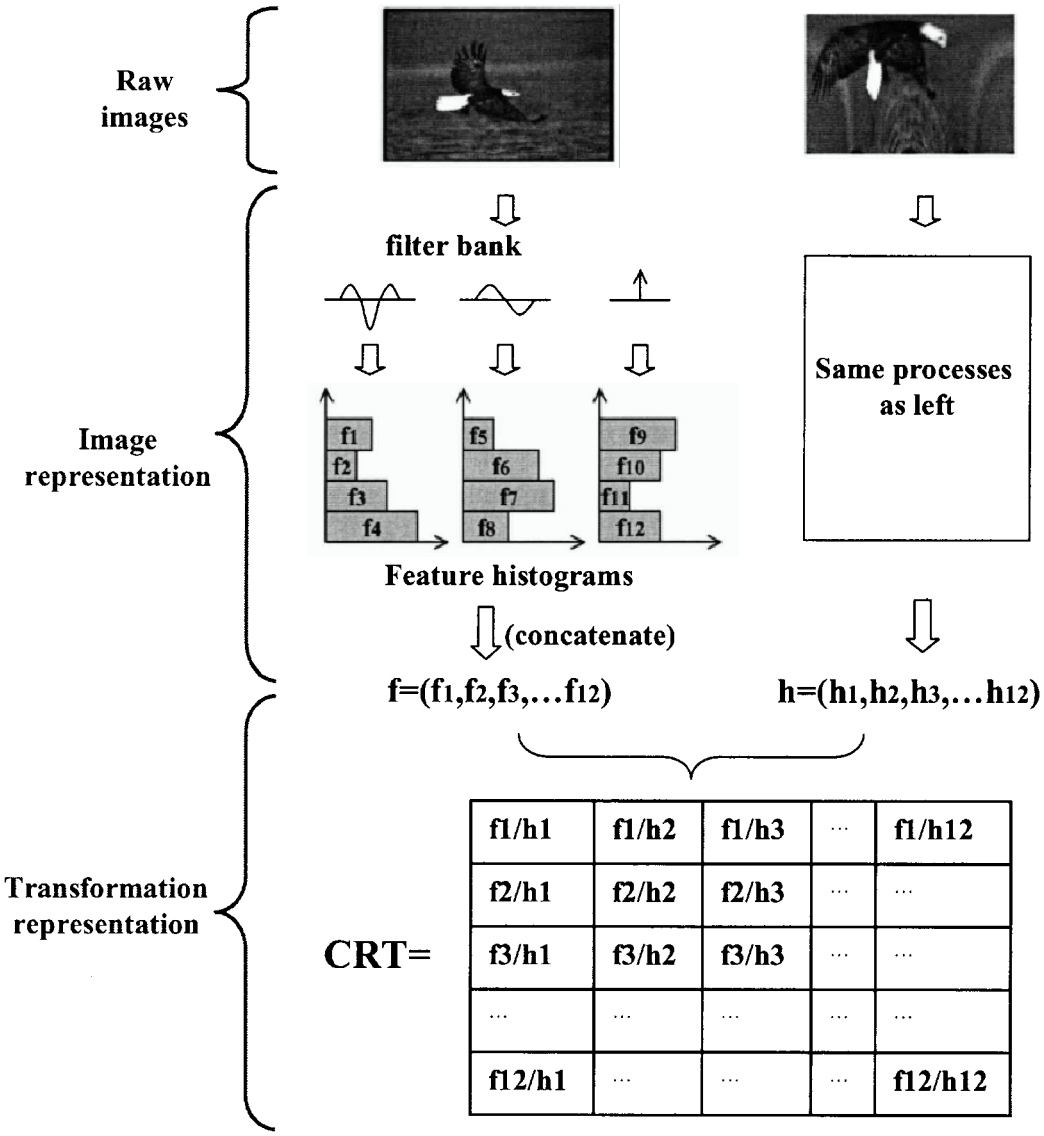


Figure 5.4: Overview of process for creating the Cross-feature Ratio Table for a single pair of images.

tized ratio bins, and $\alpha(k)$ provide the bin edges for these quantized ratios. Here, we ignore cross-feature response ratios of $0:0$ (*i.e.* no instances of either type of features exist in the pair of images) as these do not contribute in any fashion to discrimination.

The process of producing the CRT distributions H^+ and H^- are illustrated in figure 5.5.

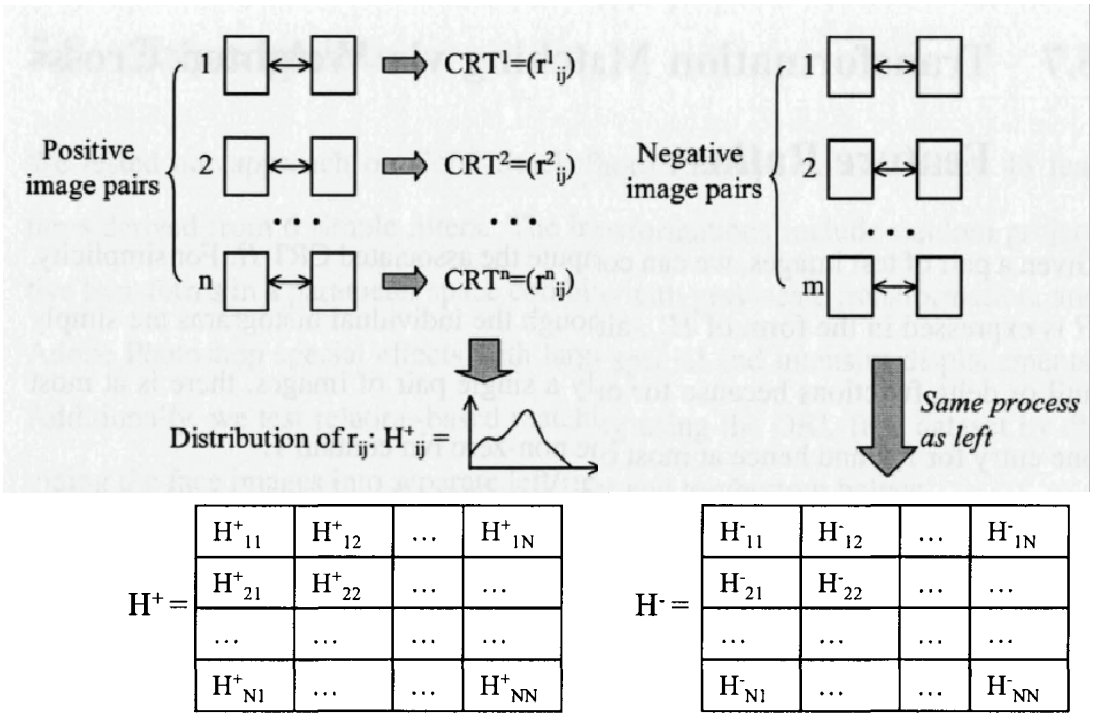


Figure 5.5: The process of forming CRT distributions H^+ and H^- from positive and negative training image pairs respectively.

The ideal scenario will be that some of the H_{ij}^+ and H_{ij}^- provide strong discrimination between cross-feature ratios from the positive and negative transformation classes. In order to evaluate this discrimination, we measure the χ^2 divergence to

estimate the relative discriminative power of the cross-feature ratios:

$$\begin{aligned} w_{ij} &= \chi^2(H_{ij}^+, H_{ij}^-) \\ &= \sum_k \frac{\left[H_{ij}^+(k) - H_{ij}^-(k)\right]^2}{H_{ij}^+(k) + H_{ij}^-(k)}, \text{ for } 1 \leq i, j \leq n \end{aligned} \tag{5.8}$$

The w_{ij} factors will be used to weight the cross-feature response ratios when computing similarity in the subsequent section.

5.7 Transformation Matching via Weighted Cross-Feature Ratios

Given a pair of test images, we can compute the associated CRT R. For simplicity, R is expressed in the form of H_{ij}^r, although the individual histograms are simply null or delta functions because for only a single pair of images, there is at most one entry for r_{ij}, and hence at most one non-zero bin contain 1.

The combined distance between the test pair CRT distribution H_{ij}^r and the positive transformation class CRT distribution H_{ij}^+ is defined as:

$$D^+ = \sum_{i,j} w_{ij} \| H_{ij}^r - H_{ij}^+ \|^2 \tag{5.9}$$

where $\| \cdot \|$ is the Euclidean norm, and the distributions H_{ij}^r and H_{ij}^+ are treated as vectors. Similarly, we can define a distance to the negative transformation class

$$D^- = \sum_{i,j} w_{ij} \| H_{ij}^r - H_{ij}^- \|^2 \tag{5.10}$$

The distances D^+ and D^- are used in deciding if the test image pair belongs to the positive or negative transformation class. The decision rule is given by

$$c = \text{sgn}(D^- - \lambda D^+) \tag{5.11}$$

where $\text{sgn}(x)$ is the sign function given by

$$\text{sgn}(x) = \begin{cases} 1, & \text{for } x > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{5.12}$$

λ is a constant parameter that is learnt from training. When $c = 1$, the image pair is computed as a match, while if $c = 0$ then the image pair is deemed mismatched.

5.8 Experiments

We tested our approach on Corel Stock Photo Library2 dataset, using 48 features derived from 6 simple filters. The transformations include random projective transforms in a parameter space combine with gray-scale transformation, and Adobe Photoshop special effects with large spacial and intensity displacements. Additionally, we test relation-based matching using the ORL face dataset by dividing the face images into separate left/right and top/bottom halves.

Note that the framework for training and classifying the matches are exactly the same across all the experiments. This is one of the key goals of our work – to provide *a unified framework for learning such relationships directly from training data, without the need to create domain-specific models that require manual intervention and possibly extensive engineering.*

5.8.1 Dataset Construction

Corel Stock Photo Library 2 includes 186 categories of photos, such as African Wildlife, Autumn in Maine and Freestyle Skiing, *etc.* Each category has 100 color images of size 768×512 or 512×768 , with quite different visual contents.

We randomly selected 20 categories as source for the training data, and another 20 categories for testing data. All the images are resized to 25% of original size by bilinear interpolation, and changed to gray-scale images.

We experimented with two differently constructed datasets: Type (1) and Type (2) datasets as described in section 5.3.

5.8.2 Filter Bank

The raw filtered features are derived from 6 simple filters: Delta, LoG and 4 oriented edge filters with 45° steps, each filter size is 3 by 3. The filter responses are divided into 8 ranges evenly distributed between -256 and 256, corresponding to 8 different features for each filter. One feature response is the number of responses of one filter fall in one range within the whole image. So totally there are 48 features used.

5.8.3 Results for Combined Geometric and Gray-Scale Transformations

The transformation used in this experiment is a combined geometric and gray-scale transformation. In this combined transformation, a gray-scale transformation will firstly be applied to the input image, followed by a geometric transformation. The gray-scale transformation involves adding Gaussian noise of standard variance 5 to the image, then decreasing the intensity of the image by 25 gray levels (the original images are in 255 gray-levels). The geometric transformation is composed of a random projective transformation component followed by a random rotation component. The random projective transformation is defined by 4 pairs of reference and corresponding points. The four reference points are located

at $(50, 50)$, $(50, -50)$, $(-50, 50)$ and $(-50, -50)$, and each corresponding point is randomly located within a 20 by 20 box centered at the reference point. The rotation component is a pure rotation around the image center of a random degree evenly distributed within a certain range. All the coordinates are in pixel units.

For the type (1) dataset, the geometric transformations are the random projective component followed by a random rotation between $0°$ and $180°$. For either the training set and testing set, we randomly chose 10 images from each of the 20 categories, cropped the 96×96 center part as reference images, resulting in 200 reference images. We applied 5 combined transformations (the geometric transformation parts are random, and therefore different from each other) on each reference image to get 5 corresponding images. In total there are 1000 matched image pairs in each (training and testing) positive data set. Furthermore, we made 2000 mismatched image pairs for each negative dataset, where each reference image has 10 transformed but mismatched corresponding images. Some of the positive image pairs in training set are shown in figure 5.6.

We also ran experiments on histogram-based Chi-Square distance measure for comparison. The ROC curves for the type (1) dataset are shown in figure 5.7.

We tested our approach on type (2) dataset where the reference images are the same as type (1) dataset. For positive image pairs, the geometric transformations are the random projective component followed by a random rotation of between $60°$ and $90°$; while for negative pairs, the random rotations are between $120°$ and $150°$. To each reference image we applied 5 combined transformations, resulting in 1000 image pairs for both positive and negative data sets. The ROC curves for type (2) dataset are shown in figure 5.8. The resultant χ^2 discriminative weights across CRT entries for type (1) and type (2) datasets are shown in figure 5.9.

Figure 5.6: Sample positive image pairs in the training set. The scales of the images are preserved. For each image pair, the left one is the input image, the right one is the transformed image. The transformation is a gray-scale transformation followed by a random geometric transformation.

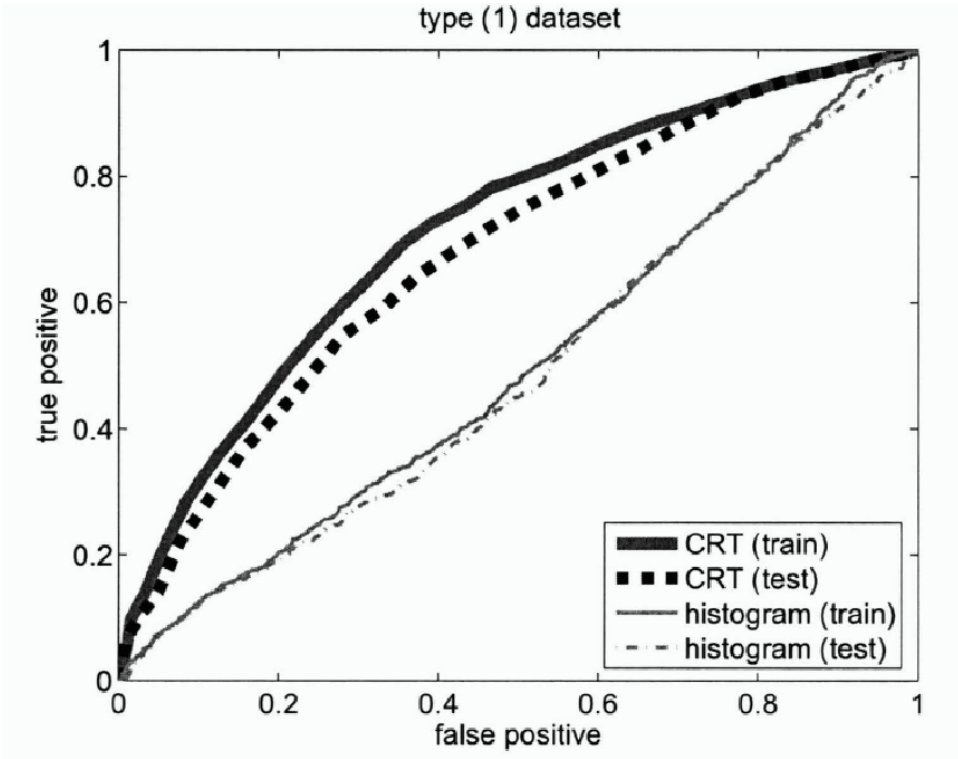


Figure 5.7: ROC curves of type (1) dataset (combined transformations).

Figure 5.7 and figure 5.8 show that the CRT-based approach outperforms the histogram-based method by a large extent. The classification ability of the histogram-based method is almost zero in the experiments as revealed by the near diagonal ROC curves. This is because histogram matching depends on brightness constancy in the transformation, which does not hold for the transformations in the datasets.

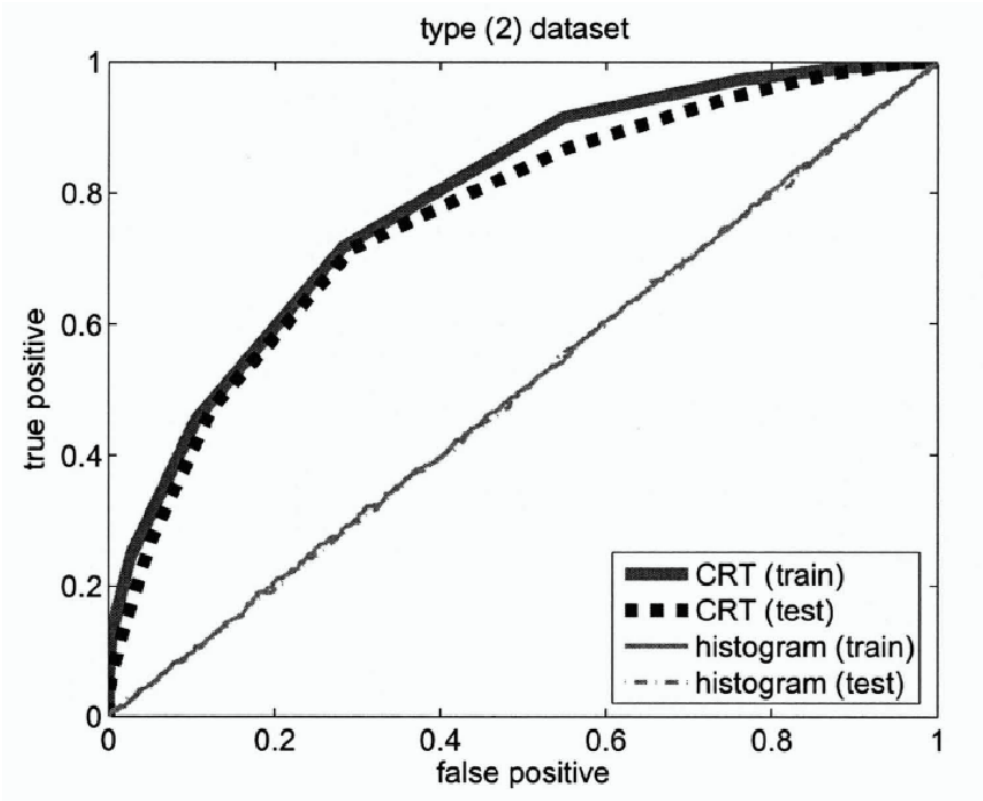
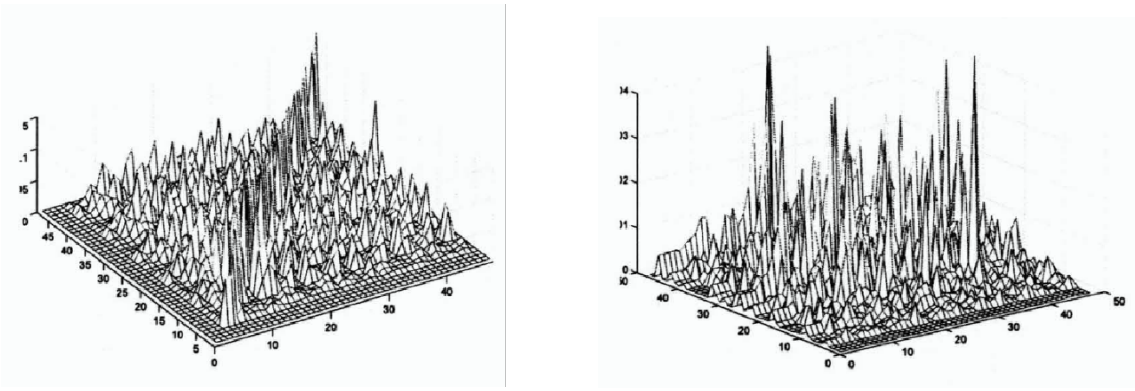


Figure 5.8: ROC curves of type (2) dataset (combined transformations).

5.8.4 Results for Adobe Photoshop Special Effects

We made use of two Adobe Photoshop effects to provide the image transformations in this experiment. The first effect is gray-level histogram equalization fol-



type (1) dataset type (2) dataset

Figure 5.9: χ^2 discrimination across CRT entries

lowed by the *Wave* distortion, while the second is intensity inversion followed by the *Rough Pastels* and the *Twirl* distortions. For both training and testing sets, 50 images are randomly chosen from each of 20 categories, so there are 1000 reference images in each set.

For both effects, we made a type (1) dataset, *i.e.* we applied the effect to all reference images through batch process, producing 1000 positive image pairs. We also make 2000 negative image pairs by selecting 2 mismatched transformed images for each reference image.

Some reference images and their corresponding transformed images by effect 1 and effect 2 are shown in figure 5.10. The left column are original images, the middle and right columns are transformed images by effect 1 and effect 2 respectively.

We also ran SSD and gray-level histogram-based Chi-Square distance measure for comparison. The ROC curves of type (1) dataset for effect 1 and 2 are shown in figure 5.11 and figure 5.12 respectively.

We also constructed type (2) dataset by the two Adobe Photoshop effects. The

Figure 5.10: Sample images of effect 1 and effect 2. Left column: reference images in training set; middle column: transformed images by effect 1; right column: transformed images by effect 2

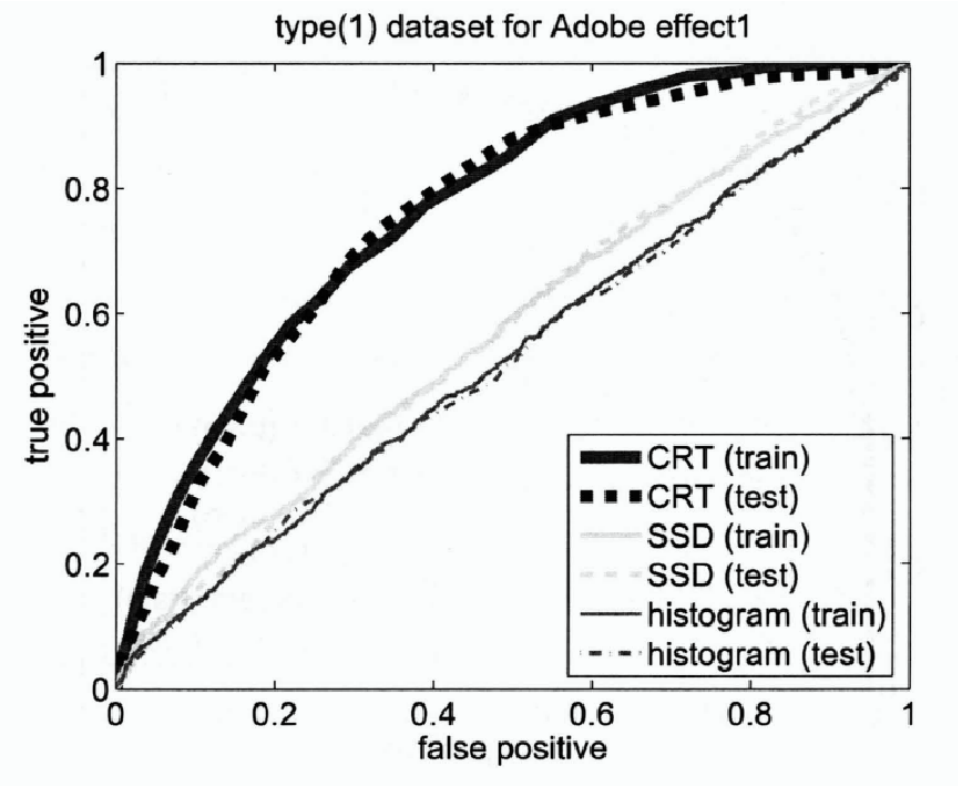


Figure 5.11: ROC curves for type (1) dataset (Adobe Photoshop effect1)

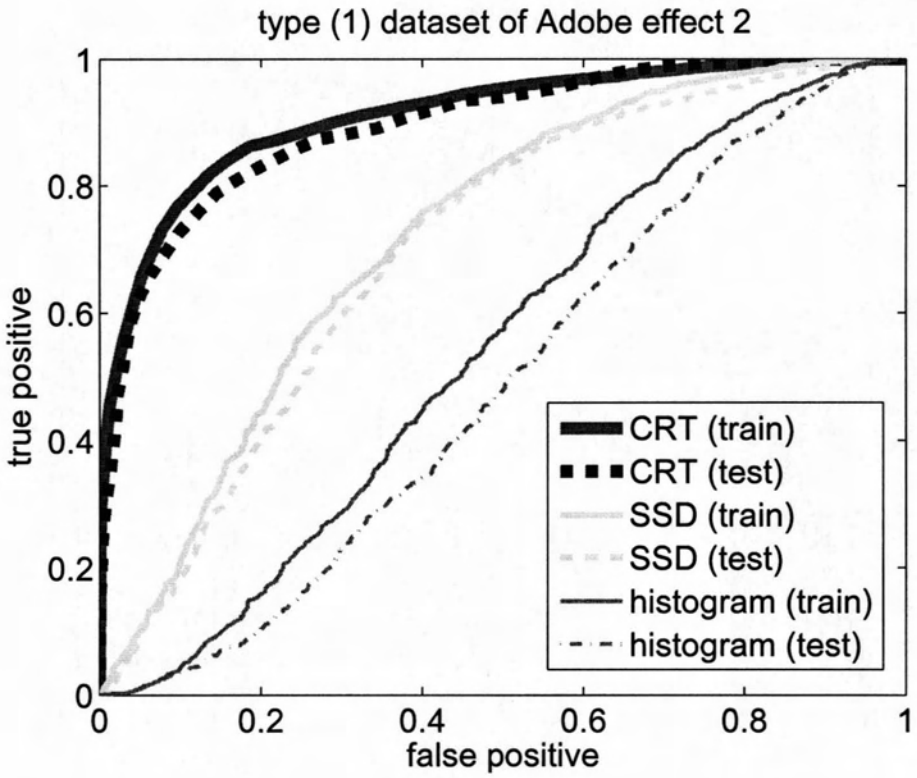


Figure 5.12: ROC curves for type (1) dataset (Adobe Photoshop effect2)

reference images are the same as those used in type (1) dataset, but this time the positive dataset includes 1000 matched image pairs by effect1, and the negative dataset includes 1000 matched images pairs by effect2, the reference images are the same. The ROC curves for type(2) dataset are shown in figure 5.13.

As shown in figure 5.11, the performances of both the SSD and histogram-based methods are both very poor. This is because the histogram-equalization operation in the transformation breaks the histogram-based methods, while the "Wave" distortion breaks the SSD-based methods. In figure 5.12, the SSD's performance improves because region-wise correspondences are partially retained by the "Rough Pastels" and "Twirl" distortions. In both experiments on type (1) dataset, the CRT-based approach outperforms SSD and histogram-based methods.

As shown in figure 5.13, the performance of the CRT-based approach on type (2) dataset is perfect. This is because the negative transformation class in type (1) dataset comprises unrelated image pairs, which is more complex than the negative transformation class in type (2) dataset comprising the same type of transformations except with different parameters, as illustrated in figure 5.3. There is good generalization from training data to test data for CRT-based approaches in all these experiments.

5.8.5 Results for Partial Face Matching

We made use of the ORL face dataset [56] as a basis to create multiple training and test sets for our partial-face matching experiments.

In the first experiment, we attempt to validate the method by testing if it was able to determine image pairs comprising complementary parts of face (*i.e.* parts that together form a complete face). A positive training set is created by taking

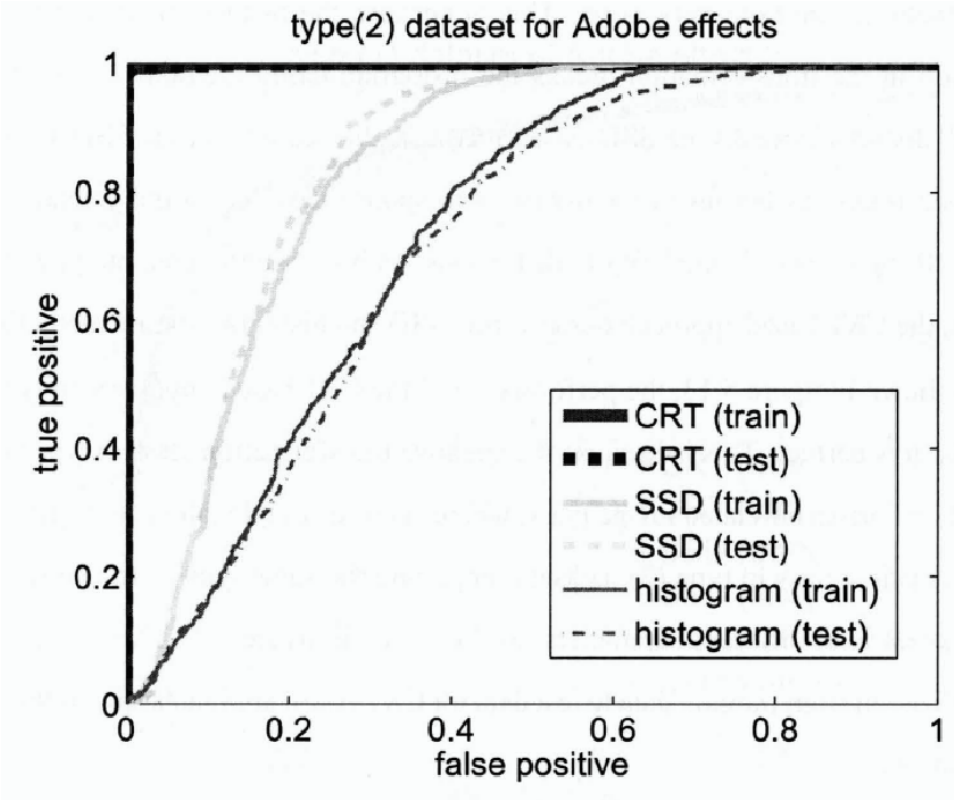


Figure 5.13: ROC curves for type (2) dataset (Adobe Photoshop effect1 for positive set and effect 2 for negative set).

images from half the individuals in the ORL dataset, and dividing the images into top and bottom halves as matched pairs. To compound the problem, half of these pairs are randomly selected and their relations reversed (*i.e.* half the pairs have top→bottom relation, while the other pairs have bottom→top relation). See figure 5.14. A negative training set is generated where for each image pair, one

Figure 5.14: Dataset for Partial Face Matching Experiment 1

image is half of the face, whereas the second image is either a copy of the first image, or a random selected image from the Corel database. The test sets are similarly constructed, *except that the individuals selected in the test sets do not appear in the training sets*. Note that the ORL dataset is not specially aligned, and therefore images in the top and bottom halves do not have aligned features, nor are all the features consistently present in the same half (*e.g.* tip of nose may appear in either half). See figure 5.15 for results.

While equivalent results may be achieved if half-face models are used in matching, the results indicate that our proposed generic framework is able to deal with the matching without domain-specific models.

In the second experiment, we attempt to evaluate if the method was capable of distinguishing if pairs of images comprising (a) left and right halves of a face, *be-*

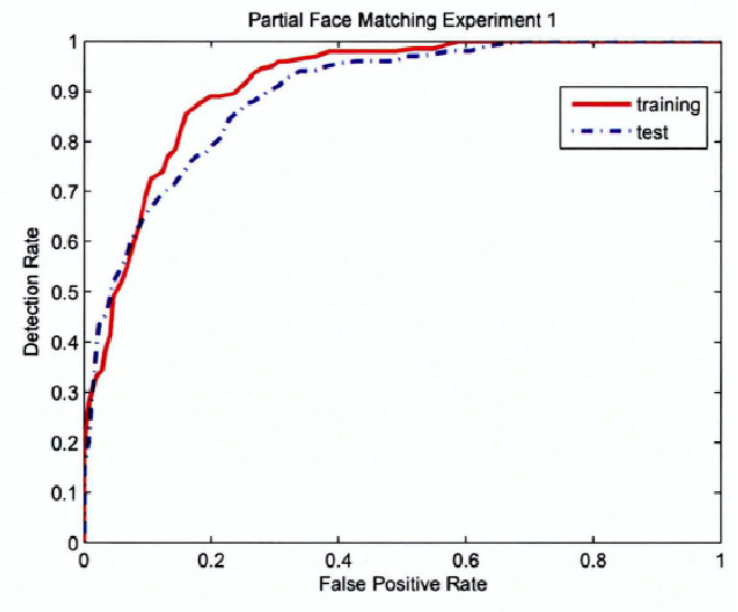


Figure 5.15: ROC Curve for Partial Face Matching Experiment 1

long to the same individual, and similarly for (b) top and bottom halves of a face. More significantly, the testing phase involves images of *individuals who were not used in the training phase*. As before, a positive training set is created by taking images from half the individuals in the ORL dataset, and dividing the images into two halves as matched pairs (experiment 2(a) involves left/right halves, while 2(b) involves top/bottom halves). The negative training set involved randomly swapping equivalent right (or bottom) halves of images such that pairs of images are not extracted from images of the same individual. The results are shown in figure 5.16.

The results of the final experiment are somewhat surprisingly good, even though the actual detection rates are only moderate. The experiment demonstrated that matching of lower faces and upper faces belonging to the same individuals can be carried out, *even though the individuals have not been encountered before in*

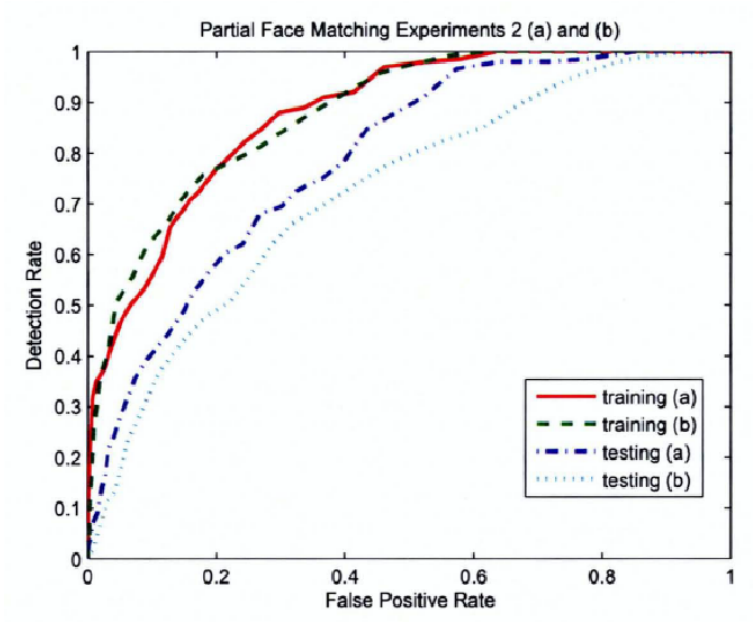


Figure 5.16: ROC Curves for Partial Face Matching Experiment 2

the training data. Moreover, the results were obtained using a generic framework, without the need for domain-specific face models and classifiers.

5.9 Conclusions

A novel approach for determining if a pair of images match each other under the effect of heavy-distortion transformation or non-structural relation. The co-occurrence statistics between features across a pair of images are learned from a training set comprising matched and mismatched image pairs – these are expressed in the form of a cross-feature ratio table. The proposed method does not require feature-to-feature correspondences, but instead identifies and exploits feature co-occurrences that are able to provide discriminative result from the transformation. The method not only allows for the matching of test image pairs that have

substantially different visual content as compared to those present in the training set, but also caters for transformations that do not preserve image structure. Experimental results indicate that this matching process not only clearly outperforms the classical methods of SSD and histogram matching, but are able to match images under extreme transformations and non-structural relations.

Chapter 6

Conclusions

6.1 Summary of Research

The central theme of this thesis is learning transformation invariance for pairwise image matching. Compared with appearance-based object detection methods, where the observed images must be of the same visual content of the training data, our problem formulation allows the test images to have *totally different visual content* as compared to the training data. This is achieved through the using of the training set composed of image pairs, which implicitly expresses the transformation by the difference between each pair of images, and the training process is based on the relative visual content between the image pairs rather than the absolute visual content of the images. So the training result can be applied to images with different visual content. The basic problem formulation is the same throughout the thesis, but three different approaches are proposed in three chapters to deal with three kinds of transformations.

In Chapter 3, we proposed a feature distance measure for image matching under low-distortion transformations. It involves a parametric family of distance measures called the cross-affinity distance measure, which is optimized to minimize image matching classification errors. The cross-affinity family of distance measures also subsume commonly-used metrics such as SSD and chamfer distance as degenerate instances. This is a strong benefit, because if one of these standard measures is truly the optimal measure, learning the best cross-affinity distance measure will result in learning this same optimal standard measure. Hence the cross-affinity distance measure is instantly guaranteed to outperform these standard measures, at least when applied to the training set. Results on matching classification with a wide variety of image content show that the learnt feature distance measure clearly outperforms the standard measures of SSD, chamfer and Bhattacharyya histogram distances.

Chapter 4 addressed the problem of image matching under heavy-distortion, correspondence-preserving transformations. Transformations with correspondences is a large group of transformations where the point to point correspondences are preserved, such as rotation, affine, projective transformations and any other image distortion where corresponding points can be found. With the help of correspondences, we can know how image features change under transformations, and they are encoded in the form of Probabilistic Transformation Descriptor (PTD). PTD models how the features change under image transformations, and test image pairs are compared via the model to determine if they match via the transformation or not. In situations where the transformation class is very large, the statistics of these descriptors, when considered as a whole, are insufficient to determine if two images match each other based. Instead, a method is proposed that automatically

partitions the unknown transformation class into sub-classes in which the statistics of these descriptors become more discriminative. The extended matching process thus involves a maximum-likelihood estimation of the unknown subset labels. Although this method needs feature correspondences in the training process to learn the PTDs, it does not need to find correspondences in the matching process.

Chapter 5 copes with image matching under heavy-distortion, correspondence-breaking transformations which may not preserve image structure. This class of transformations can be extended to include non-structural relations, *e.g.* the "transformation" between the bottom half and the top half of a face. As the transformation may be so severe that feature correspondences are unlikely to be found, the feature co-occurrence statistics between a pair of images, which are expressed in the form of a Cross-feature Ratio Table (CRT), are utilized to determine if images match each other. The CRT may be considered to be a representation of transformation learned through image pairs. The distribution of CRTs of training image pairs are captured in two tables: one for positive image pairs, the other for negative image pairs. The relative discriminative power (which is expressed as a set of weights) of cross-feature ratios, is estimated from these two CRT distributions. In the image matching stage, the CRT of the test image pairs is compared with the two reference CRT distributions by the set of weights to decide which transformation class it belongs to. Experimental results show that the CRT-based approach not only clearly outperforms the classical methods of SSD and histogram-based methods, but also are able to match images under extreme transformations.

6.2 Future Work

The future work based on this thesis include:

Extension of transformation distribution partitioning In Chapter 4, the relationship of feature types between one corresponding feature pair is used as an index to map to transformation distributions of different sub-ranges. One research direction in the future would be to extend to use more than one pair of corresponding features to index to transformation distributions of different sub-ranges. In this way, the range of the indexed transformation distributions will be smaller, and hence the predictive accuracy of the corresponding PTDs will increase. However, the number of sub-range distributions will increase exponentially, *e.g.*, when two pairs of corresponding features are used as index, the number of indexes will be n^4 (n is the number of feature types). This will bring efficiency problem in the learning stage, and the problem of how to select a limited number of PTDs to represent the original training transformation distribution needs to be investigated also.

Extension of the CRT-based approach. There are quite a few directions to extend the CRT-based approach introduced in Chapter 5. Firstly, the CRT-based method treats all transformations between the training image pairs as one transformation class, this is unlike the PTD-based method which partitions the training transformation distribution into many sub-range distributions. One direction of research in the future would be to partition the training transformation class into multiple sub-classes for the CRT-based approach. The advantage of partitioning is that the resulting transformation range for each partition is reduced, and therefore the CRT for each sub-class will be more accurate. Secondly, the image

features used in Chapter 5 are limited to a fully-dense and over-complete feature set through convolution with a small filter bank, one research area in the future would be using more complex and discriminative features such as image patches or SIFT features in the CRT-based approaches. Thirdly, the cross-feature ratio used in Chapter 5 may also be substituted by other measures, such as the absolute difference between features. One area for research in the future may be to compare different measures, and deciding which measure will be optimal for a certain class of transformations.

Extension to object class recognition The CRT-based method introduced in Chapter 5 exploits the co-occurrence statistics between image pairs, which does not require feature correspondences between the image pairs. Hence, CRT-based method has the ability to handle a very wide range of transformations, even those may not be considered as "transformations" in a traditional sense, *e.g.*, the image of an upper half face and the image of a lower half face may be considered as matched, the front view of a human and the back view of a human may also be considered as matched. The "transformations" between these kind of matched image pairs are related with the object class, which is unlike the traditional transformations that are separate from the object class. Therefore, one research direction in the future would be to extend the CRT-based method from image matching to object class recognition. Suppose different views of an object class are provided in the training data, then the CRT-based method would be invariant to the "transformations" between different views, thus obtain the ability of view-invariant object recognition.

Bibliography

[1] *Third IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS)*, Copenhagen, Denmark, 2002.

[2] A. Agarwal and B. Triggs. 3D human pose from silhouettes by relevance vector regression. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 882–888, Washington, DC, 2004.

[3] F. Aherne, N. Thacker, and P. Rockett. The Bhattacharyya metric as an absolute similarity measure for frequency coded data. *Kybernetika*, 34(4):363–368, 1998.

[4] S. Avidan. Joint feature-basis subset selection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 283–290, Washington, DC, 2004.

[5] A. Barla, F. Odone, and A. Verri. Hausdorff kernel for 3D object acquisition and detection. In *Proc. European Conference on Computer Vision*, pages 20–33, Copenhagen, Denmark, 2002.

[6] H. Barrow, J. Tenenbaum, R. Bolles, and H. Wolf. Parametric correspondence and chamfer matching: Two new techniques for image matching. In

International Joint Conferences on Artificial Intelligence, pages 659–663, Cambridge, MA, 1977.

[7] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, Apr 2002.

[8] A. C. Berg, T. L. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondences. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 26–33, San Diego, CA, 2005.

[9] R. Boyle and R. Thomas. *Computer Vision: A First Course*. Blackwell Scientific Publications, 1988.

[10] Kenneth R. Castleman. *Digital Image Processing*. Prentice Hall, 1996.

[11] X. Chen and T. J. Cham. Learning feature distance measures for image correspondences. In *CVPR*, volume 2, pages 560–567, 2005.

[12] Edwin K. P. Chong and Stanislaw H. Zak. *An Introduction to Optimization*. John Wiley and Sons, Inc, 2nd edition, 2001.

[13] R. T. Collins and Y. Liu. On-line selection of discriminative tracking features. In *Proc. International Conference on Computer Vision*, volume 1, pages 346–352, Nice, France, 2003.

[14] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 142–149, Hilton Head, South Carolina, 2000.

[15] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:564–577, 2003.

[16] C. M. Cyr and B. B. Kimia. 3D object recognition using shape similarity-based aspect graph. In *Proc. International Conference on Computer Vision*, volume 1, pages 254 – 261, Vancouver, BC, Canada, 2001.

[17] M. P. Dubuisson and A. K. Jain. A modified hausdorff distance for object matching. In *Proc. IAPR International Conference on Pattern Recognition*, volume 1, pages 566–568, Jerusalem, Israel, 1994.

[18] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley, 2nd edition, 2001.

[19] D. A. Forsyth, J. L. Mundy, A. Zisserman, C. Coelho, A. Heller, and C. Rothwell. Invariant descriptors for 3-D object recognition and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10):971–991, October 1991.

[20] D. A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, 2003.

[21] G. H. Golub and C. F. van Loan. *Matrix Computations*. The John Hopkins University Press, 2nd edition, 1989.

[22] R. Gonzalez and R. Woods. *Digital Image Processing,*. Addison-Wesley Publishing Company, 1992.

[23] K. Grauman and T. Darrell. Efficient image matching with distributions of

local invariant features. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 627–634, San Diego, CA, 2005.

[24] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *Proc. International Conference on Computer Vision*, volume 2, pages 1458–1465, Beijing, China, 2005.

[25] E. Hadjidemetriou, M. D. Grossberg, and S. K. Nayar. Multiresolution histograms and their use for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(7):831–847, July 2004.

[26] P. Heckbert. *Graphics Gems IV*. Academic Press, Boston, 1994.

[27] B. Heisele, P. Ho, and T. Poggio. Face recognition with support vector machines: Global versus component-based approach. In *Proc. International Conference on Computer Vision*, volume 2, pages 688–694, Vancouver, BC, Canada, 2001.

[28] A. Hertzmann, C. Jacobs, N. Oliver, B. Curless, and D. Salesin. Image analogies. In *SIGGRAPH*, Los Angeles, CA, 2001.

[29] G. J. Holzmann. *Beyond Photography: The Digital Darkroom*. Prentice-Hall, 1988.

[30] R. Horaud, B. Conio, O. Leboulleux, and B. Lacolle. An analytic solution for the perspective 4-point problem. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 500–507, San Diego, CA, 1989.

[31] D. P. Huttenlocher and S. Ullman. Recognizing solid objects by alignment with an image. *International Journal of Computer Vision*, 5(2):195–212, 1990.

[32] A. Jain. *Fundamentals of Digital Image Processing*. Prentice-Hall, 1986.

[33] T. Jebara. Images as bags of pixels. In *Proc. International Conference on Computer Vision*, volume 1, pages 265–272, Nice, France, 2003.

[34] T. Kailath. The divergence and Bhattacharyya distance measures in signal selection. *IEEE Transaction on Communication Technology*, 15:52–60, 1967.

[35] Y. Ke and R. Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 506–513, Washington, DC, 2004.

[36] H. Kollnig and H. H. Nagel. 3D pose estimation by directly matching polyhedral models to gray value gradients. *International Journal of Computer Vision*, 23(3):283–302, 1997.

[37] R. Kondor and T. Jebara. A kernel between sets of vectors. In *International Conference on Machine Learning*, Washington, DC, 2003.

[38] S. Lazebnik, C. Schmid, and J. Ponce. A sparse texture representation using local affine regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1265–1278, 2005.

[39] F. F. Li and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 524–531, San Diego, CA, 2005.

[40] D. G. Lowe. Fitting parameterized three-dimensional models to images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(5):441–450, May 1991.

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

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

[43] P. J. Moreno, P. Ho, and N. Vasconcelos. A kullback-leibler divergence based kernel for svm classification in multimedia applications. In *Conference in Neural Information Processing Systems*, 2003.

[44] J. L. Mundy and A. Zisserman. *Geometric Invariance in Computer Vision*. The MIT Press, 1992.

[45] J. C. Nash. *Compact Numerical Methods for Computers: Linear Algebra and Function Minimisation*. Taylor and Francis, 2nd edition, January 1990.

[46] W. Niblack, R. Barber, W. Equitz, E. H. Flickner, M. D.and Glasman, D. Petkovic, P. Yanker, C. Faloutsos, G. Taubin, , and Y. Heights. Querying images by content, using color, texture and shape. In *SPIE Conference on Storage and Retrieval for Image and Video Databases*, volume 1908, pages 173–187, 1993.

[47] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *CVPR*, New York, NY, 2006.

[48] S. Paek, C. L. Sable, V. Hatzivassiloglou, A. Jaimes, B. H. Schiffman, S. F. Chang, and K. R. McKeown. Integration of visual and text based approaches for the content labeling and classification of photographs. In *Proc. ACM SIGIR Workshop on Multimedia Indexing and Retrieval*, Berkeley, CA, 1999.

[49] S. Petitjean, J. Ponce, and D. Kriegman. Computing exact aspect graphs of curved objects:algebraic surfaces. *International Journal of Computer Vision*, 9(3):231–255, 1992.

[50] H. K. Pong and T. J. Cham. Alignment of 3D models to images using region-based mutual information and neighborhood extended gaussian images. In *Proc. Asian Conference on Computer Vision*, pages 60–69, Hyderabad, India, 2006.

[51] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 2nd edition, 1992.

[52] J. Puzicha, T. Hofmann, and J. Buhmann. Non-parametric similarity measures for unsupervised texture segmentation and image retrieval. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 267–272, San Juan, Puerto Rico, 1997.

[53] Y. Rubner and C. Tomasi. Texture-based image retrieval without segmentation. In *Proc. International Conference on Computer Vision*, volume 2, pages 1018 – 1024, Kerkyra (Corfu), Greece, 1999.

[54] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 2(40):99–121, 2000.

[55] G. Salton and M. Smith. On the application of syntactic methodologies in automatic text analysis. In *Proc. ACM SIGIR conference on Research and Development in Information Retrieval*, pages 137–150, 1989.

[56] F. Samara and A. Harter. Parameterisation of a stochastic model for human face identification. In *WACV*, Sarasota, FL, 1994.

[57] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, 1986.

[58] T. Sim, S. Baker, and M. Bsat. The CMU Pose, Illumination, and Expression database. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(12):1615–1618, 2003.

[59] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, Nice, France, 2003.

[60] M. Stricker and M. Orengo. Similarity of color images. In *SPIE Conference on Storage and Retrieval for Image and Video Databases*, volume 2420, pages 381–392, 1995.

[61] M. J. Swain and D. H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

[62] K. Toyama and A. Blake. Probabilistic tracking in a metric space. In *Proc. International Conference on Computer Vision*, volume 2, pages 50–57, Vancouver, BC, Canada, 2001.

[63] S. Ullman, M. Vidal-Naquet, and E. Sali. Visual features of intermediate complexity and their use in classification. *Nature Neuroscience*, 5(7):682–687, 2002.

[64] S. Vinther and R. Cipolla. Active 3D object recognition using 3D affine invariants. In *Proc. European Conference on Computer Vision*, volume 2, pages 15–24, Stockholm, Sweden, 1994.

[65] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 511–518, Kauai, Hawaii, 2001.

[66] P. Viola and W. M Wells. Alignment by maximization of mutual information. In *Proc. International Conference on Computer Vision*, pages 16–23, Cambridge, Massachusetts, 1995.

[67] L. Wolf and A. Shashua. Kernel principal angles for classification machines with applications to image sequence interpretation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 635–640, Madison, Wisconsin, 2003.

[68] E. Xing, A. Ng, M. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In *Advances in NIPS*, volume 15, 2003.

[69] L. Zelnik-Manor and M. Irani. Event-based analysis of video. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 123–130, Kauai, Hawaii, 2001.