The Dissertation Committee for Vishal Monga
certifies that this is the approved version of the following dissertation:

PERCEPTUALLY BASED METHODS FOR ROBUST IMAGE HASHING

Committee:

Brian L. Evans, Supervisor

Ross Baldick

Wilson S. Geisler

John E. Gilbert

Joydeep Ghosh

Sriram Vishwanath

PERCEPTUALLY BASED METHODS FOR ROBUST IMAGE HASHING

by

Vishal Monga, B.Tech.; M.S.E.E.

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

The University of Texas at Austin

August 2005

This thesis is dedicated to my mother and the greatest influence on my life, Late Mrs. Sushil Monga

Acknowledgments

I would like to begin by thanking my parents, albeit I understand any amount of gratitude shown to them is woefully inadequate. My father's unconditional support is largely the reason that this PhD is completed in United States. No words are sufficient to describe my late mother's contribution to my life. I owe every bit of my existence to her. This thesis is dedicated to her memory.

I have been lucky to receive tremendous affection from several members in my extended family. Their support and encouragement has been instrumental in my overcoming several hurdles in life. I am particularly grateful to my fiance Nimisha and her parents who have kept exemplary patience while I completed my thesis. I am indeed blessed to have them in my life.

I am indebted to my advisor, Prof. Brian Evans. Brian has influenced not only my graduate studies, but my whole life. He has instilled in me by example, a strong sense of discipline and integrity, for which I am eternally grateful. Brian is a deeply committed researcher, teacher, and advisor. Observing him for four years has helped me define my own research goals.

The most precious gift I received during my graduate studies at UT is a friend by the name of Moin. He is and will always remain my best buddy. Moin has encouraged me to realize my potential, and become more practical. I am also enormously appreciative of his patience in dealing with my absent-mindedness, in particular the two cases when I lost my passport. Next on the list is Prabhat who introduced me to life outside research, and the importance of optimism.

I would like to thank my committee members, Prof. Baldick, Prof. Geisler, Prof. Gilbert, Prof. Ghosh, and Prof. Vishwanath (in alphabetical order). I am honored to have them serve on my committee. Prof. Baldick teaches a great optimization course, which I believe, influences the research of many graduate students in signal processing and communications. My initiation into research was brought about by taking Prof. Geisler's Vision Systems class, which also influenced much of my later research in human visual system (HVS) modeling. He is a truly brilliant instructor and I am honored to have him as a co-author on the first paper I wrote.

Prof. Gilbert, Prof. de Veciana and Prof. Cline's classes at UT Austin introduced me to the beauty and strength of mathematics. All of these gentlemen also had a major philosophical impact on my research.

Although, I did not get the chance to take one of Dr. Ghosh's classes, his personal research has inspired me. My growing interest in data mining and its connections with signal processing, information theory and linear algebra, is borne out of discussions with him and his research group. I would particularly like to thank Arindam, who I treat as a benchmark. Arindam has heavily influenced my approach to problem solving, and I am truly honored to have authored a couple of papers with him on my dissertation topic. I am also privileged to claim him as a friend. Several other students in Dr. Ghosh's group namely Srujana, Sreangsu, and Suju have given me valuable insights on several problems.

I am thankful to students in Embedded Signal Processing Laboratory (ESPL), and the Wireless Networking and Communications Group (WNCG) at large who provide a very pleasant environment for quality work to flourish. Among the students in Dr. Bovik's lab, I have had stimulating discussions with Raghu and Umesh. I will sorely miss having them around.

Finally, I would like to thank several friends outside of UT Austin who have helped enrich my graduate studies experience. This includes Niranjan Damera-Venkata at HP Labs; Raja Bala, Gaurav Sharma and Shen-ge Wang at Xerox Research, and Kivanc Mihcak at Microsoft Research. I am especially thankful to Kivanc for several brianstorming sessions on statistical signal processing and its relationship to media hashing.

Vishal Monga
August, 2005

PERCEPTUALLY BASED METHODS FOR ROBUST IMAGE HASHING

Publication No. _____

Vishal Monga, PhD

The University of Texas at Austin, 2005

Supervisor: Brian L. Evans

Hash functions are frequently called message digest functions. Their purpose is to extract a short binary string from a large digital message. A key feature of conventional cryptographic (and other) hashing algorithms such as message digest 5 (MD5) and secure hash algorithm 1 (SHA-1) is that they are extremely sensitive to the message; i.e., changing even one bit of the input message will change the output dramatically. However, multimedia data such as digital images undergo various manipulations such as compression and enhancement. An image hash function should instead take into account the changes in the visual domain and produce hash values based on the image's visual appearance. Such a function would facilitate comparisons and searches in large image databases. Other applications of a perceptual hash lie in content authentication and watermarking.

This dissertation proposes a unifying framework for multimedia signal hashing. The problem of media hashing is divided into two stages. The first stage extracts media-dependent intermediate features that are robust under incidental modifications while

being different for perceptually distinct media with high probability. The second stage performs a media-independent clustering of these features to produce a final hash.

This dissertation focuses on feature extraction from natural images such that the extracted features are largely invariant under perceptually insignificant modifications to the image (i.e. robust). An iterative geometry preserving feature detection algorithm is developed based on an explicit modeling of the human visual system via end-stopped wavelets. For the second stage, I show that the decision version of the feature clustering problem is NP-complete. Then, for any perceptually significant feature extractor, I develop polynomial time clustering algorithms based on a greedy heuristic.

Existing algorithms for image/media hashing exclusively employ either cryptographic or signal processing methods. A pure signal processing approach achieves robustness to perceptually insignificant distortions but compromises security which is desirable in applications for multimedia protection. Likewise pure cryptographic techniques while secure, completely ignore the requirement of being robust to incidental modifications of the media. The primary contribution of this dissertation is a joint signal processing and cryptography approach to building robust as well as secure image hashing algorithms. The ideas proposed in this dissertation can also be applied to other problems in multimedia security, e.g. watermarking and data hiding.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 The Need for Image Hashing

Due to the popularity of digital technology, more and more digital images are being created and stored every day. This introduces a problem for managing large image databases. One cannot determine if an image already exists in a database without exhaustively searching through all the entries. Further complication arises from the fact that two images that appear identical to the human eye may have different digital representations, which makes it difficult to compare a pair of images, e.g. an original image and its compressed version, an image stored using distinct transforms, or an image enhanced via common signal processing operations. This has spurred interest in developing algorithms to generate suitable image identifiers, or *image hash* functions. One possible option to derive content-dependent short binary strings from the image is the use of conventional cryptographic hashes such as message digest 5 (MD5) and secure hash algorithm 1 (SHA-1) [3]. However, the problem associated with these is that they are extremely sensitive to the message being hashed; i.e., changing even one bit in the input changes the output dramatically. Instead, these identifiers must necessarily take into account the changes in the visual domain and capture the essential perceptual attributes of the image. For this reason, such an image identifier is termed as a *perceptual image hash*.

Further need for such image descriptors arises for the purpose of *integrity verification*.

Because of the easy-to-copy nature of digital media, digital data can be tampered with and hence there exits a need to verify the content of the media to ensure its authenticity. In the literature, the methods used for media verification can be classified into two categories: digital signature-based [1], [2], [4], [5], [6], [7], [8], [9] and watermark-based [10], [11], [12], [13], [14], [15]. A digital signature is a set of features extracted from the media that sufficiently represents the content of the original media. Watermarking, on the other hand, is a media authentication/protection technique that embeds invisible (or inaudible) information into the media. For content authentication, the embedded watermark can be extracted and used for verification purposes.

The major difference between a watermark and a digital signature is that the embedding process of the former requires the content of the media to change. However, for content authentication, both the watermark-based approach and the digital signature-based approach are expected to be sensitive to any malicious modification of the media while being *robust* to incidental modifications such as JPEG compression (with compression ratios that do not result in significant loss of perceptual quality) or image enhancement. Fig. 1.1 illustrates this requirement with a practical example. Fig. 1.1 (b) shows the original *tiff* image of a former US President and the first lady. The JPEG compressed (quality factor (QF) = 40) version of the same image is shown in Fig. 1.1 (a). Fig. 1.1 (c) then shows a tampered version of the image in Fig. 1.1 (a) in which a malicious change is made to the First Lady'S face. It is desired then that the signatures (or hashes) extracted from Fig. 1.1 (a) and (b) agree whereas those for Fig. 1.1 (b) and (c) be significantly different. In practice, extracting content descriptors (or image features) that can guarantee the detection of all malicious changes has proved infeasible. To a large extent, my research has hence focused on developing randomized algorithms for media hashing (Chapters 3 and 4) that significantly enhance security against maliciously generated inputs.


(a) JPEG Compressed Image


(b) Original Image


(c) Tampered Image

Figure 1.1: Example illustrating the requirements of a hash in a content authentication scenario. The hash values from images in (a) and (b) are required to agree, while being different from the one extracted from the image in (c).

Other applications of perceptual image hashing have recently been conceived for content dependent key generation and synchronization in video watermarking [16, 17].

1.2 Review of Related Work and Open Issues

This section reviews the current research in content-dependent digital signature/hash extraction from images. Open research issues are subsequently summarized.

1.2.1 Image Statistics Based Approaches

The fundamental premise underlying these approaches is:

> *There exists a certain class of statistics of the image that are largely invariant under small (visually insignificant) perturbations to the image.*

In one of the earliest approaches, Schneider [4] *et al.* use intensity histograms of image blocks for authentication. The verification process involves computing the Euclidean distance between the histogram of the original and the candidate image to be verified. The sum of all such distances over the image is used as a measure of image authenticity. This approach requires storage of public key encrypted histograms which can be considerably

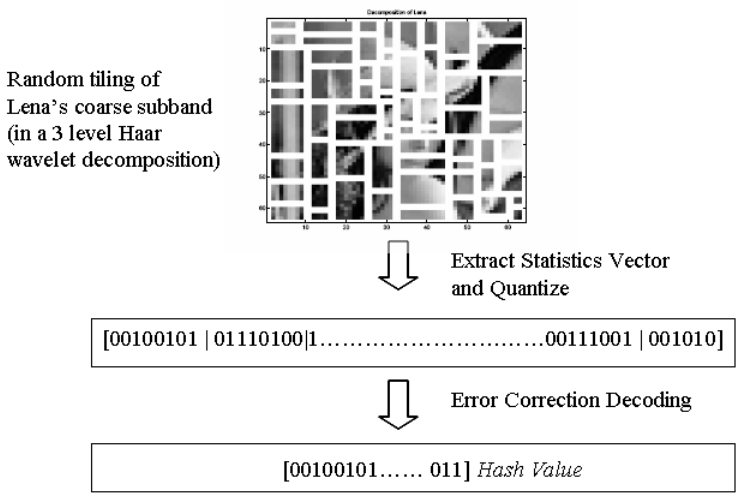


Figure 1.2: Illustration of the hash algorithm by Venkatesan *et al.* [1]

large. The most significant drawback of their method is that it is easy to modify an image without altering its histogram (e.g. permuting pixels within a block). This makes the scheme less secure. Kailasnathan *et al.* [5] extract a signature that is based on intensity statistics such as mean, variance and other higher order moments of image blocks. While simple in concept, their scheme has drawbacks similar to that in [4].

In [1] Venkatesan *et al.* develop an image hash based on an image statistics vector extracted from the various sub-bands in a wavelet decomposition of the image. They observe that statistics such as averages of coarse sub-bands and variances of other (fine detail) sub-bands stay invariant under a large class of content-preserving modifications to the image. The algorithm is randomized by first dividing each sub-band into random regions by using a secret key and then extracting statistics as before from each region. The quantized statistics are then input to the decoding stage of a Reed-Muller error-correcting code [18] to generate the final hash value. Fig. 1.2 illustrates this scheme. Although statistics of wavelet coefficients have been found to be far more robust than

intensity statistics, they do not necessarily capture content changes[1] well, particularly those that are maliciously generated.

1.2.2 Preserving Coarse Image Representations

In [8] the authors propose a robust hash based on preserving selected discrete cosine transform (DCT) coefficients. Their method is based on the observation that large changes to low frequency DCT coefficients of the image are needed to change the appearance of the image significantly. To randomize the procedure (or dependent on a key) the authors in [8] first generate several smooth and zero-mean random patterns $P^{(i)}$, $i = 1, 2, ..., N$. Considering the DCT block B from the image and the pattern as vectors, the image I is projected on each pattern and its absolute value is compared with the threshold Th to obtain N bits $b_i, i = 1, 2, ..., N$

$$if \ \ |B.P^{(i)}| < Th \ \ b_i = 0 \tag{1.1}$$

$$if \ \ |B.P^{(i)}| \geq Th \ \ b_i = 1 \tag{1.2}$$

Since the patterns have a zero mean, the projections do not depend on the mean gray-value of the block and only depend on the variations within the block itself. The hash extracted via this method is fairly robust to JPEG compression, uniform noise addition, and standard linear sharpening and blurring filters. However, the method is very sensitive to even small global geometric transformations, such as rotation and scaling, and local ones, such as random bending or shearing.

[1] A content change here signifies a perceptually meaningful perturbation to the image, e.g. adding/removing an object, significant change in texture, and morphing a face. In general, a perceptual hash should be sensitive to both incidental as well as malicious content changes. A major challenge in secure image hashing is to develop algorithms that can detect (with high probability) malicious tampering of image data.

Mihcak and Venkatesan [2] develop another image hashing algorithm by using an iterative approach to binarize the DC subband (lowest resolution wavelet coefficients) of a 3-level Haar wavelet decomposition of the image. The key observation in their work is that the significant geometric features of an image are preserved under small perturbations to the image. Their hash algorithm is summarized below in Fig. 1.3.

The DC subband, i.e. *coarse detail*, carries low resolution wavelet coefficients that represent crude image features. A thresholding of these coefficients is hence used to form the hash. This is similar to the approach in [8] in which DCT coefficients were used instead. The LSI filtering (Step 4) introduces blurred regions to gain robustness against small modifications. Most significantly, the iterative nature of the algorithm repeatedly emphasizes (or strengthens) geometrically "strong" components while eliminating "weaker" ones.

The aforementioned approaches implicitly make the simplifying assumption that most robust attributes of an image's visual appearance are captured by low spatial frequency or equivalently low spatial resolution coefficients in a DCT/discrete wavelet transform (DWT) version of the image. While DCT/DWT have proven to be quite effective for conventional image processing applications, it is still an open question as to which mappings (if any) from DCT/DWT coefficients preserve essential image information for perceptual image hashing.

1.2.3 Relation Based Approaches

Relation-based approaches are also based on forming suitable content identifiers based on a transform domain (DCT/DWT) representation of the image. However, unlike the methods in Section 1.2.2, relation-based approaches do not preserve certain transform

1. Find the Discrete Wavelet Transform (DWT) of image I up to level L. Let I_A be the resulting DC subband.

2. Perform the following *thresholding* operation on I_A to produce the binary map $\mathbf{M}$

$$M(i,j) = \begin{cases} 1 & \text{if } I_A(i,j) \geq T, \\ 0 & \text{otherwise} \end{cases}$$

   where T is a threshold that is adaptively chosen.

3. Let $\mathbf{M}_1 = \mathbf{M}$, $ctr = 1$

4. Apply 2-D linear shift invariant (LSI) filtering on $\mathbf{M}_1$ via filter f to obtain $\mathbf{M}_2$

5. Apply a thresholding on $\mathbf{M}_2$ as in step 2. Let $\mathbf{M}_3$ be the binary output

6. If $ctr \geq C$, terminate the iteration and go to Step 7. Else, find the Hamming distance $D_H(\mathbf{M}_3, \mathbf{M}_1)$; if it is less than ρ (a user-defined value), then terminate the iteration and go step 7 else, set $\mathbf{M}_1 = \mathbf{M}_3$ and go to step 3

7. $H(I) = \mathbf{M}_3$

Figure 1.3: The hash algorithm by Mihcak *et al.* [2] based on preserving low resolution wavelet coefficients. $H(I)$ denotes the final hash value.

coefficients but look to identify (approximately) invariant relationships between those coefficients.

A typical relation-based technique for image authentication tolerating JPEG compres-

sion has been reported by Lin and Chang [6], [7]. They extract a digital signature by using the invariant relationship between any two DCT coefficients, which are at the same position of two different 8×8 blocks. Let $\mathbf{F}_p$, $\mathbf{F}_q$ denote DCT coefficients of two arbitrary non-overlapping blocks of an image, at the same position. Similarly, let $\tilde{\mathbf{F}}_p$ and $\tilde{\mathbf{F}}_q$ denote the corresponding DCT coefficients of the JPEG compressed version of the image. Then, define $\Delta\mathbf{F}_{p,q} = \mathbf{F}_p - \mathbf{F}_q$ and $\Delta\tilde{\mathbf{F}}_{p,q} = \tilde{\mathbf{F}}_p - \tilde{\mathbf{F}}_q$. Lin and Chang [7] identify the following properties must hold true:

1. if $\Delta\mathbf{F}_{pq} > 0$ then $\Delta\tilde{\mathbf{F}}_{pq} > 0$
2. else if $\Delta\mathbf{F}_{pq} < 0$ then $\Delta\tilde{\mathbf{F}}_{pq} < 0$
3. else if $\Delta\mathbf{F}_{pq} = 0$ then $\Delta\tilde{\mathbf{F}}_{pq} = 0$

The aforementioned differences, i.e. $\Delta\mathbf{F}_{pq}{}'s$ are computed (for randomly selected DCT coefficients from the original image) and stored as the digital signature. The authentication procedure then involves deriving a new signature by computing the same differences from a given query image and comparing with the pre-computed signature to determine if the query image is authentic. This scheme, by virtue of its design, is very robust to JPEG compression, i.e. the same signature results even after the image is JPEG compressed. However, it still remains vulnerable to several other perceptually insignificant modifications, e.g. where the statistical nature of distortion is different from the blur caused by compression.

Recently, Lu *et al.* [9] have proposed a "structural digital signature" for image authentication. They observe that in a sub-band wavelet decomposition, a parent and child node are uncorrelated, but they are statistically dependent. In particular, they observe that the difference of the magnitude of wavelet coefficients at consecutive scales (i.e. a parent

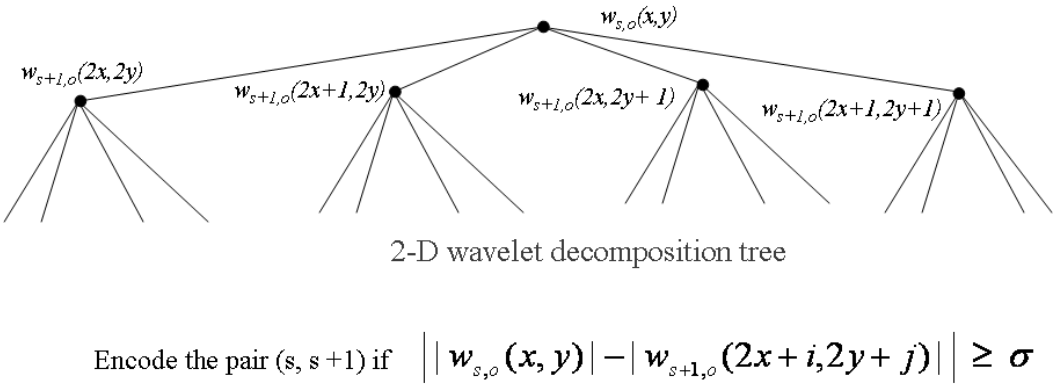


Figure 1.4: Structural digital signature by Lu *et al.* $w_{s,o}(x,y)$ represents a wavelet coefficient at scale s, orientation o and position (x,y). σ denotes a positive constant.

node and its four child nodes) remains largely preserved for several content-preserving manipulations. Identifying such parent-child pairs and subsequently encoding the pairs form their robust digital signature. Qualitatively hence, their approach is similar to that of Lin and Chang [7], except that the invariant relationship is identified between wavelet coefficients instead of DCT coefficients. Fig. 1.4 illustrates the invariant underlying their scheme.

1.2.4 Open Research Issues

Feature points have long been used in computer vision for the purpose of object recognition and classification. Feature point detectors are attractive for their inherent sensitivity to content changing manipulations. Current approaches based on feature points [19, 20], however, have limited utility in perceptual hashing applications since they are sensitive to several perceptually insignificant modifications as well. A robust image hashing algorithm based on visually significant feature points remains elusive.

Several geometric manipulations (e.g. large rotation and scaling) do not change the image's appearance. When comparing two images, one of which has suffered a large geometric attack, by using one of the existing techniques, the two images will have very different hash values. This is because the content descriptors, e.g. coarse wavelet/DCT coefficients, do not have a geometrically invariant representation. A good feature point detector can also yield a representation of image content that is naturally robust to local and geometric distortions.

Section 1.1 identifies two major objectives of perceptual image hashing. First is resilience against non intentional or perceptually insignificant modifications to the image, known as *perceptual robustness* (or simply robustness) of the hash. Second is the ability to survive intentional attacks (generated by a malicious adversary), referred to as hash security. It has further been identified [3] that the security properties of a hash are intimately related to the randomization scheme employed in the design of the hash algorithm.

Table 1.1 provides a comparative summary of the image hashing algorithms surveyed in this chapter. It can be seen from Table 1.1 that algorithms that achieve good robustness typically compromise security. Further, existing methods do not facilitate a trade-off between the two aforementioned objectives. Another very important question that remains to be answered is the (minimum) length of the hash required to successfully achieve a desired level of robustness.

Finally, several researchers have identified randomization as an important ingredient for secure hashing. A theoretical analysis of randomized media hashing algorithms, and the quantitative relationship of randomization parameter(s) with hash security, however, has not yet been reported in literature.

Image Hashing Algorithm	*Robustness*	*Security*	*Remarks*
Cryptographic hashes			
MD5, SHA-1	Poor	Good	No trade-offs possible
Statistics Based			
Schneider *et al.* [4]	Poor	Poor	–
Kailasanathan *et al.* [5]	Poor	Poor	–
Venketasan *et al.* [1]	Fair	Fair	Trade-offs hard to achieve
Coarse Representations			
Fridrich *et al.* [8]	Fair	Poor	Sensitive to small geometric changes
Mihcak *et al.* [2]	Good	Poor	Trade-offs hard to achieve
Relation Based			
Lin *et al.* [7]	Fair	Poor	–
Lu *et al.* [9]	Fair	Fair	Sensitive to small geometric changes

Table 1.1: A comparison of the image hashing algorithms surveyed in this chapter. Note the trade-off between hash robustness and security/randomization.

1.3 Contributions and Outline of Dissertation

The following are contributions to the theory, algorithms, and design of perceptually based robust image hashing schemes included in this dissertation, which are described in [21], [22], [23] [24], [25].

1. I develop a novel unifying framework for perceptual media hashing that uses a media-dependent feature extractor followed by media-independent clustering of vectors in the feature space. I introduce quantitative definitions for the goals of media hashing algorithms which encompass requirements of perceptual robustness as well

as hash security.

2. I develop an iterative image feature extraction algorithm based on an explicit modeling of the human visual system (HVS) via end-stopped wavelets [26]. Within the feature extractor, I enable trade-offs between perceptual robustness, fragility, and randomization of the hash that previously proposed schemes did not address.

3. I develop a novel cost function for feature vector compression and show that the decision version of the feature clustering problem is NP-complete. Then, for any perceptually significant feature extractor, I develop polynomial time clustering algorithms based on a *greedy heuristic*. The proposed algorithm automatically determines the final hash length required to satisfy a specified distortion. Unlike existing methods for hash compression [1], [27] that are limited to binary/Euclidean vectors, the proposed clustering is applicable to feature vectors in any metric space.

4. I develop novel randomized clustering algorithms for secure media hashing. I quantify the relationship of randomization with hash security. I quantitatively as well as qualitatively establish the virtues of randomization in compensating for the practical limitations of feature detectors.

5. Based on the feature extractor in step 2, I develop a digital signature based scheme for image authentication under geometric attacks. I generalize the well known Hausdorff distance [28] and bring out its efficacy in capturing visual changes in image content. The new distance includes several earlier Hausdorff measures [28], [29] as special cases.

Chapter 2 presents a unifying framework for perceptual media hashing. First, the desired properties of a perceptual image hash are formally defined. Trade-offs between

these properties are identified. Next, a novel two-stage framework is introduced for perceptual hashing, which could be extended to other media besides images, e.g. audio, and documents.

Chapter 3 presents a novel solution to the first stage of the image hashing problem using visually robust feature points. Previous work on robust feature detection from natural images is reviewed. An iterative feature extraction algorithm is then developed that can employ a variety of feature detectors. The proposed feature detector finds low-level robust image features based on an explicit modeling of the human visual system. Within the feature detector, trade-offs between perceptual robustness and fragility to visually distinct images, are facilitated.

Chapter 4 designs media independent clustering algorithms for feature vector compression. Limitations of traditional compression approaches for the hashing application are discussed and subsequently a novel cost function is proposed for feature vector compression. It is shown that the decision version of the underlying feature clustering problem is NP-complete. For any perceptually significant feature extractor, polynomial time clustering algorithms are developed based on a *greedy heuristic*. The number of clusters (or equivalently the length of the final hash) is determined naturally as an outcome of the proposed clustering. Randomized algorithms are then developed for secure media hashing. The proposed algorithms (deterministic as well as randomized) allow clustering of vectors in any metric space with a meaningful notion of distance on image features.

Chapter 5 develops a framework for image authentication under geometric attacks using the feature extractor in Chapter 3. A generalized Hausdorff distance measure is developed to compare features from two different images. The new distance accounts for occasional feature detector failure and is shown to more accurately capture visual

changes in image content.

Chapter 6 concludes the dissertation by summarizing the contributions and provides suggestions for future work.

Chapter 2

A unifying framework for media hashing

2.1 Introduction

This chapter presents a unifying framework for perceptual media hashing. It also develops a formal (quantitative) description of the desired properties of a perceptual image hash. The key objective of this chapter is to highlight the fundamental challenges in perceptual image hashing that solutions developed in the subsequent chapters will address.

Section 2.2 defines the desired properties of a perceptual image hash. Trade-offs between these properties are described. Section 2.3 then introduces a two-stage unifying framework for media hashing. The framework consists of a perceptually meaningful media dependent feature extractor followed by a media independent clustering of vectors in the feature space. Section 2.4 summarizes the ideas discussed in this chapter.

2.2 Perceptual Image Hashing: Statement of Goals

In view of the discussion in Chapter 1, I will now quantify the desired properties of a perceptual image hash.

Let $\mathcal{I}$ denote a set of images (e.g., all natural images of a particular size) with finite

cardinality. Also, let $\mathcal{K}$ denote the space of *secret* keys[1]. Our hash function then takes two inputs, an image $I \in \mathcal{I}$ and a *secret* key $K \in \mathcal{K}$, to produce a q-bit binary hash value $h = H(I, K)$. Let $I_{ident} \in \mathcal{I}$ denote an image such that I_{ident} looks the same as I. Likewise, an image in $\mathcal{I}$ that is perceptually distinct from I will be denoted by I_{diff}. Let θ_1, θ_2 satisfy $0 < \theta_1, \theta_2 < 1$. Then, three desirable properties of a *perceptual* hash are identified as follows:

1. **Perceptual robustness**:
   $Probability(H(I, K) = H(I_{ident}, K)) \geq 1 - \theta_1$, for a given θ_1

2. **Fragility to visually distinct images**:
   $Probability(H(I, K) \neq H(I_{diff}, K)) \geq 1 - \theta_2$, for a given θ_2

3. **Unpredictability of the hash**:
   $Probability(H(I, K) = v) \approx \frac{1}{2^q}, \ \ \forall\ v \in \{0, 1\}^q$

Let $\mathcal{Q} = \{H(I, K) \mid I \in \mathcal{I},\ K \in \mathcal{K}\}$, i.e., the set of all possible realizations of the hash algorithm on the product space $\mathcal{I} \times \mathcal{K}$. Also, for a fixed $I_0 \in \mathcal{I}$ define $\mathcal{O} = \{H(I_0, K) \mid K \in \mathcal{K}\}$. That is, for a fixed image, $\mathcal{O}$ is the set of all possible realizations of the hash algorithm over the key space $\mathcal{K}$.

Note that the probability measure in the first two properties is defined over the set $\mathcal{Q}$. For example, property 1 requires that for any pair of "perceptually identical" images in $\mathcal{I}$ and any $K \in \mathcal{K}$, the hash values must be identical with high probability. The probability

[1]The key space in general can be constructed in several ways. A necessary but not sufficient condition for secure hashing is that the key space should be large enough to preclude exhaustive search. For this paper, unless specified otherwise, I will assume the key space to be the Hamming space of 32-bit binary strings.

measure in the third property, however, is defined on $\mathcal{O}$. That is, the third property requires that as the secret key is varied over $\mathcal{K}$ for a fixed input image, the output hash value must be approximately uniformly distributed among all possible q-bit outputs.

Remark: The three desired properties as laid out above are those of an "ideal" hash algorithm. Whether or not such hash algorithms can even be constructed (esp. in a computationally feasible time) remains an outstanding open problem in media hashing. I therefore do not claim to achieve these properties for arbitrarily low values of θ_1, θ_2 and q, but instead provide heuristic solutions that achieve these goals with high probability.

Further, the three desirable hash properties conflict with one another. The first property amounts to robustness under small perturbations, whereas the second one requires the minimization of collision probabilities for perceptually distinct inputs. There is clearly a trade-off here. For example, if very crude features were used, then they would be hard to change (i.e., robust), but it is likely that one is going to encounter collision of perceptually different images. Likewise for perfect randomization, a uniform distribution on the output hash values (over the key space) would be needed, which in general, would deter achieving the first property. From a security viewpoint, the second and third properties are very important; i.e., it must be extremely difficult for an adversary to manipulate the content of an image and yet obtain the same hash value. It is desirable for the hash algorithm to achieve these (conflicting) properties to some extent and/or facilitate trade-offs.

2.3 Hashing Framework

I partition the problem of deriving an image hash into two steps, as illustrated in Fig. 2.1. The first step extracts a feature vector from the image, whereas the second

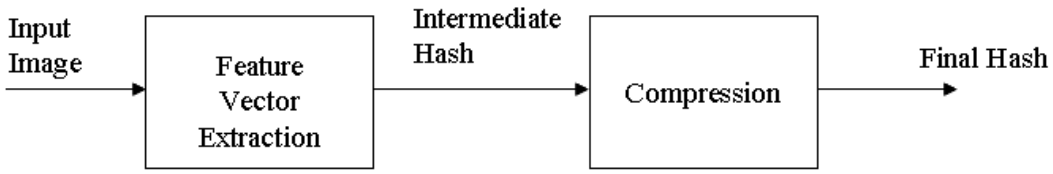


Figure 2.1: Block diagram of the hash function.

stage compresses this feature vector to a final hash value. In the feature extraction step, the two-dimensional image is mapped to a one-dimensional feature vector. This feature vector must capture the perceptual qualities of the image. That is, two images that appear identical to the human visual system should have feature vectors that are close in some distance metric. Likewise, two images that are clearly distinct in appearance must have feature vectors that differ by a large distance. For the rest of the dissertation, I will refer to this visually robust feature vector (or its quantized version) as the "intermediate hash". My proposed approach to extracting visually robust image features is detailed later in Chapter 3.

The second step then compresses this intermediate hash vector to a final hash value. This will involve clustering between the intermediate hash vector of an input source (image) and the intermediate hash vectors of its perceptually identical versions. In Chapter 4, I develop a clustering algorithm based on the distribution of intermediate hash vectors to address this problem.

There are are two major motivations for such a partitioning of image hashing algorithm(s). First, quite naturally image hashing lends into being modeled as a clustering problem. In particular, we are attempting to cluster visually indistinguishable images into the same cell (or map to the same hash value). Second, the proposed framework allows for a somewhat media independent approach; i.e. if a common solution to stage

2 is developed, then it may be used to compress/cluster feature vectors independent of the media that the features were derived from. Hence, the proposed framework is a unified one for media hashing. With that in view, Chapter 4 develops a family of generic clustering algorithms that can be applied to features from an arbitrary media. This dissertation, however, focuses exclusively on robust feature extraction techniques from natural images.

2.4 Conclusion

This chapter introduced formal mathematical definitions for the desirable properties of a perceptual image hash. A unified framework for hashing within which these properties would be targeted was subsequently presented. Two important observations were made: 1) there is an inherent trade-off between the desired properties of a perceptual hash and hashing algorithms must facilitate these, and 2) the cryptographic secret key plays an important role in randomizing the hash and enabling security against maliciously generated inputs (image pairs). In the next chapter, I will present the design of a visually robust image feature extractor that has characteristics as desired in stage 1 of the proposed hashing framework.

Chapter 3

Feature Extraction

3.1 Introduction

This chapter proposes a paradigm for deriving intermediate hash (or feature) vectors from images using visually significant feature points. The feature points should be largely invariant under perceptually insignificant distortions. To satisfy this, I propose an iterative feature extractor to obtain significant geometry preserving feature points. Based on an underlying robust feature extraction algorithm, I develop explicit randomized feature extraction techniques to enhance hash security.

End-stopped wavelet kernels that capture essential and robust attributes of human perception are described in Section 3.2. Section 3.3 then proposes a feature detector based on constructing visually significant end-stopped wavelets [26]. Section 3.4 presents a probabilistic quantization approach to binarize image feature vectors that enhances robustness to perceptually insignificant perturbations, and at the same time, introduces randomness. Iterative algorithms (both deterministic and randomized) that construct intermediate hash vectors are described in Section 3.5. Experimental results demonstrating perceptual robustness, sensitivity to content changes, and receiver operating characteristic (ROC) analysis across 1000 different images are reported in Section 3.6. Finally, Section 3.7 summarizes the key ideas introduced in this chapter.

3.2 End-Stopped Wavelets

Psychovisual studies have identified the presence of certain cells, called hypercomplex or end-stopped cells, in the primary visual cortex [30]. For real-world scenes, these cells respond strongly to extremely robust image features such as corner like stimuli and points of high curvature [26], [31]. The term end-stopped comes from the strong sensitivity of these cells to end-points of linear structures. Bhattacherjee *et al.* [26] construct "end-stopped" wavelets to capture this behavior. The construction of the wavelet kernel (or basis function) combines two operations. First, linear structures having a certain orientation are selected. These linear structures are then processed to detect line-ends (corners) and/or high curvature points.

Morlet wavelets can be used to detect linear structures having a specific orientation. In the spatial domain, the two dimensional (2-D) Morlet wavelet is given by [32]

$$\psi_M(\mathbf{x}) = (e^{j\mathbf{k}_0.\mathbf{x}} - e^{-\frac{1}{2}|\mathbf{k}_0|^2})(e^{-\frac{1}{2}|\mathbf{x}|^2}) \tag{3.1}$$

where $\mathbf{x} = (x, y)$ represents 2-D spatial coordinates, and $\mathbf{k}_0 = (k_0, k_1)$ is the *wave-vector* of the mother wavelet, which determines scale-resolving power and angular-resolving power of the wavelet [32]. The frequency domain representation, $\psi_M(\mathbf{k})$, of a Morlet wavelet is

$$\hat{\psi}_M(\mathbf{k}) = (e^{-\frac{1}{2}|\mathbf{k}-k_0|^2} - e^{-\frac{1}{2}|\mathbf{k}_0|^2})(e^{-\frac{1}{2}|\mathbf{k}|^2}) \tag{3.2}$$

Here, $\mathbf{k}$ represents the 2-D frequency variable (u, v). The Morlet function is similar to the Gabor function, but with an extra correction term $e^{-\frac{1}{2}(|\mathbf{k}_0|^2+|\mathbf{x}|^2)}$ to make it an admissible wavelet [33]. The orientation of the wave-vector determines the orientation tuning of the filter. A Morlet wavelet detects linear structures oriented perpendicular to the orientation of the wavelet.

In two dimensions, the end points of linear structures can be detected by applying the first-derivative of Gaussian (FDoG) filter in the direction parallel to the orientation of structures in question. The first filtering stage detects lines having a specific orientation and the second filtering stage detects end-points of such lines. These two stages can be combined into a single filter to form an "end-stopped" wavelet [26]. An example of an end-stopped wavelet and its 2-D Fourier transform follow:

$$\psi_E(x,y) = \frac{1}{4} y e^{-\left(\frac{x^2+y^2}{4} + \frac{k_0}{4}(k_0 - 2jx)\right)} \tag{3.3}$$

$$\hat{\psi}_E(u,v) = 2\pi \left(e^{-\frac{(u-k_0)^2+(v)^2}{2}}\right)\left(jv e^{-\frac{u^2+v^2}{2}}\right) \tag{3.4}$$

Eqn. (3.4) shows $\hat{\psi}_E$ as a product of two factors. The first factor is a Morlet wavelet oriented along the $u-$axis. The second factor is a FDoG operator applied along the frequency-axis v, i.e. in the direction perpendicular to the Morlet wavelet. Hence, this wavelet detects line ends and high curvature points in the vertical direction. Fig. 3.1 illustrates the behavior of the end-stopped wavelet as in (3.3)-(3.4). Fig. 3.1 (a) shows a synthetic image with L-shaped region surrounded by a black background. Fig. 3.1 (b) shows the raw response of the vertically oriented Morlet wavelet at scale $i = 2$. Note that this wavelet responds only to the vertical edges in the input. The response of the end-stopped wavelet is shown in Fig. 3.1 (c) also at scale $i = 2$. The responses are strongest at end-points of vertical structures and negligibly small elsewhere. The local maxima of these responses in general correspond to corner-like stimuli and high curvature points in images.

(a) Synthetic L-shaped image (b) Response of a Morlet wavelet, orientation = 0^o (c) Response of the end-stopped wavelet

Figure 3.1: Behavior of the end-stopped wavelet on a synthetic image: note the strong response to endpoints and corners.

3.3 Proposed Feature Detection Method

The proposed approach to feature detection computes a wavelet transform based on an *end-stopped* wavelet obtained by applying the FDoG operator to the Morlet wavelet:

$$\psi_E(x, y, \theta) = (FDoG)\ o\ (\psi_M(x, y, \theta)) \tag{3.5}$$

Orientation tuning is given by $\theta = \tan^{-1}(\frac{k_1}{k_0})$. Let the orientation range $[0, \pi]$ be discretized into M intervals and the scale parameter α be sampled exponentially as α^i, $i \in Z$. This results in the wavelet family

$$\left(\psi_E(\alpha^i(x, y, \theta_k)\right), \alpha \in \mathcal{R},\ i \in Z \tag{3.6}$$

where $\theta_k = (k\pi)/M$, k = 0,..., M-1. The wavelet transform is

$$W_i(x, y, \theta) = \int f(x_1, y_1)\psi_E{}^* \left(\alpha^i(x - x_1, y - y_1), \theta\right) dx_1 dy_1 \tag{3.9}$$

The sampling parameter α is chosen to be 2.

1. Compute the wavelet transform in (3.9) at a suitably chosen scale i for several different orientations. The coarsest scale ($i = 1$) is not selected as it is too sensitive to global variations. The finer the scale, the more sensitive it is to distortions such as quantization noise. I choose $i = 3$.

2. Locations (x, y) in the image that are identified as candidate feature points satisfy

$$W_i(x, y, \theta) = \max_{(x',y') \in N_{(x,y)}} |W_i(x', y', \theta)| \tag{3.7}$$

   where $N_{(x,y)}$ represents the local neighborhood of (x, y) within which the search is conducted.

3. From the candidate points selected in step 2, qualify a location as a final feature point if

$$\max_{\theta} W_i(x, y, \theta) > T \tag{3.8}$$

   where T is a user-defined threshold.

Figure 3.2: Feature detection method that preserves significant image geometry feature points of an image.

Fig. 3.2 describes the proposed feature detection method. Step 1 computes the wavelet transform in (3.9) for each image location. Step 2 identifies significant features by looking for local maxima of the magnitude of the wavelet coefficients in a preselected neighborhood. I chose a circular neighborhood to avoid increasing detector anisotropy. Step 3 applies thresholding to eliminates spurious local maxima in featureless regions of the image.

The method in Fig. 3.2 has two free parameters: integer scale i and real threshold T. The threshold T is adapted to select a fixed number (user defined parameter P) of feature points from the image. An image feature vector is formed by collecting the magnitudes of the wavelet coefficients at the selected feature points. The length P feature vector is labeled as $\mathbf{f}$.

3.4 Probabilistic Quantization

Once the feature vector is obtained, the next step is then to obtain a binary string from the same that would form the intermediate hash. Previous approaches [4], [19] use public-key encryption methods on image features to arrive at a digital (binary) signature. Such a signature would be very sensitive to small perturbations in the extracted features (here, the magnitude of the wavelet coefficients). I observe that under perceptually insignificant distortions to the image, although the actual magnitudes of the wavelet coefficients associated with the feature points may change, the "distribution" of the magnitudes of the wavelet coefficients is still preserved.

In order to maintain robustness, I propose a quantization scheme based on the probability distribution of the features extracted from the image. In particular, I use the normalized histogram of the feature vector $\mathbf{f}$ as an estimate of its distribution. The normalized histogram appears to be largely insensitive to attacks that do not cause significant perceptual changes. In addition, a randomization rule [34] is also specified which adds unpredictability to the quantizer ouput.

Let L be the number of quantization levels, $\mathbf{f}_q$ denote the quantized version of $\mathbf{f}$, and $\mathbf{f}(k)$ and $\mathbf{f}_q(k)$ denote the k^{th} elements of $\mathbf{f}$ and $\mathbf{f}_q$, respectively. The binary string obtained from the quantized feature vector $\mathbf{f}_q$ is hence of length $P\lceil \log_2(L) \rceil$ bits. If quantization

were deterministic, then the quantization rule would be given by

$$l_{i-1} \leq \mathbf{f}(k) < l_i, \quad \mathbf{f}_q(k) = i \tag{3.10}$$

where $[l_{i-1}, l_i)$ is the i^th quantization bin. Note, the quantized values are chosen to be i, $1 \leq i \leq L$. This is because unlike traditional quantization for compression, there is no constraint on the quantization levels for the hashing problem. These may hence be designed for convenience as long as the notion of "closeness" is preserved. Here, we design quantization bins $[l_{i-1}, l_i)$ such that

$$\int_{l_{i-1}}^{l_i} p_f(x)dx = \frac{1}{L}, \quad 1 \leq i \leq L \tag{3.11}$$

where $p_f(x)$ is the estimated distribution of $\mathbf{f}$. This ensures that the quantization levels are selected according to the distribution of image features. In each interval $[l_{i-1}, l_i)$, I obtain center points C_i with respect to the distribution, given by

$$\int_{C_i}^{l_i} p_f(x)dx = \int_{l_{i-1}}^{C_i} p_f(x)dx = \frac{1}{2L} \tag{3.12}$$

Then, I find *deviations* P_i, Q_i about C_i where $l_{i-1} \leq P_i \leq C_i$ and $C_i \leq Q_i \leq l_i$, such that

$$\frac{\int_{C_i}^{Q_i} p_f(x)dx}{\int_{C_i}^{l_i} p_f(x)dx} = \frac{\int_{P_i}^{C_i} p_f(x)dx}{\int_{l_{i-1}}^{C_i} p_f(x)dx}, \quad 1 \leq i \leq L \tag{3.13}$$

P_i, Q_i are hence symmetric around C_i with respect to the distribution $p_f(x)$. By virtue of the design of C_i's in (3.12), the denominators in (3.13) are both equal to $\frac{1}{2L}$ and hence only the numerators need to be computed. The probabilistic quantization rule is then completely given by

$$P_i < \mathbf{f}(k) < Q_i, \quad \mathbf{f_q}(k) = \begin{cases} i & \textit{with probability } \dfrac{\int_{P_i}^{\mathbf{f}(k)} p_f(x)dx}{\int_{P_i}^{Q_i} p_f(x)dx} \\ i-1 & \textit{with probability } \dfrac{\int_{\mathbf{f}(k)}^{Q_i} p_f(x)dx}{\int_{P_i}^{Q_i} p_f(x)dx} \end{cases}$$

$$l_{i-1} \leq \mathbf{f}(k) \leq P_i, \quad \mathbf{f}_q(k) = i - 1 \quad \textit{with probability} \quad 1 \tag{3.14}$$

and

$$Q_i \leq \mathbf{f}(k) \leq l_i, \quad \mathbf{f}_q(k) = i \quad \textit{with probability} \quad 1 \tag{3.15}$$

The output of the quantizer is deterministic except in the interval (P_i, Q_i) Note, if $\mathbf{f}(k) = C_i$ for some i, k, then the assignment to levels i or $i-1$ takes place with equal probability, i.e. 0.5. The quantizer output in other words is completely randomized. On the other hand, as $\mathbf{f}(k)$ approaches P_i or Q_i the quantization decision becomes almost deterministic.

In the next section, I present iterative algorithms that employ the feature detector in Section 3.3, and the quantization scheme described in this section to construct binary intermediate hash vectors.

3.5 Intermediate Hash Algorithms

3.5.1 Deterministic Intermediate Hash Algorithm

The intermediate hash function for image I is represented as $\mathbf{h}(I)$ and $D_H(\cdot, \cdot)$ denotes the normalized Hamming distance between its arguments (binary strings).

Mihcak *et al.* [2] observe that primary geometric features of the image are largely invariant under small perturbations to the image. They propose an iterative filtering scheme that minimizes the presence of "geometrically weak components" and enhances "geometrically strong components" by means of *region growing*. I adapt the algorithm in [2] to lock onto a set of feature-points that are largely preserved under perceptually insignificant distortions to the image. The *stopping criterion* for the proposed iterative algorithm is achieving a *fixed point* for the binary string obtained on quantizing the vector of feature points $\mathbf{f}$.

1. Get parameters MaxIter, ρ and P, and set count = 1

2. Use the feature detector in Fig. 3.2 to obtain a length P vector $\mathbf{f}$.

3. Quantize $\mathbf{f}$ according to the rule given by (3.10) and (3.11) (i.e. deterministic quantization) to obtain a binary string $\mathbf{b}_f^1$

4. (Perform order-statistics filtering) Let $I_{os} = OS(I; p, q, r)$. For a 2-D input X, $Y = OS(X; p, q, r)$ where $\forall i, j$, $Y(i, j)$ is equal to the r^{th} element of the sorted set of $X(i', j')$, where $i' \in \{i-p, i-p+1, ..., i+p\}$ and $j' \in \{j-q, j-q+1, ..., j+q\}$. Note, for $r = (2p+1)(2q+1)/2$ this is same as median filtering.

5. Perform low-pass linear shift invariant filtering on I_{os} to obtain I_{lp}.

6. Repeat steps (2) and (3) with I_{lp} to obtain $\mathbf{b}_f^2$

7. If (count = maxIter) go to step 8.
   else if $D_H(\mathbf{b}_f^1, \mathbf{b}_f^2) < \rho$ go to step 8.
   else set $I = I_{lp}$ and go to step 2.

8. Set $\mathbf{h}(I) = \mathbf{b}_f^2$

Figure 3.3: Deterministic intermediate hash algorithm

Fig. 3.3 describes the proposed intermediate hash algorithm. Step 4 eliminates isolated significant components. Step 5 preserves the "geometrically strong" components by low-pass filtering (which introduces blurred regions). The success of the deterministic

algorithm relies upon the *self-correcting* nature of the iterative algorithm as well as the robustness of the feature detector. The above iterative algorithm is fairly general in that any feature detector that extracts visually robust image features may be used.

3.5.2 Randomized Intermediate Hash Algorithm

Randomizing the hash output is desirable not only for security against inputs designed by an adversary (malicious attacks), but also for scalability, i.e. the ability to work with large data sets while keeping the collision probability for distinct inputs in check. The algorithm as presented in Fig. 3.3 does not make use of a secret key and hence there is no randomness involved.

In this section, I will construct randomized intermediate hash algorithms using a secret key K, which is used as the seed to a pseudo-random number generator for the randomization steps in the algorithm. For this reason, I now denote the intermediate hash vector as $\mathbf{h}(I, K)$, i.e. function of both the image and the secret key. I present a scheme that employs a *random partitioning* of the image to introduce unpredictability in the hash values. A step-by-step description is given in Fig. 3.4.

Qualitatively, the randomized intermediate hash algorithm enhances the security of the hash by employing the deterministic iterative algorithm[1] on randomly chosen regions or sub-images. As long as these sub-images are sufficiently unpredictable (i.e. they differ significantly as the secret key is varied), then the resulting intermediate hashes are also different with high probability. Examples of random partitioning of the *lena* image using Algorithm 2, are shown in Fig. 3.5. In each case, i.e. Figs. 3.5 (a), (b), and (c), a different secret key was used.

[1]This would now use a probabilistic quantizer.

1. (Random Partitioning) Divide the image into N (overlapping) random regions. In general, this can be done in several ways. The main criterion is that a different partitioning should be obtained (with high probability) as the secret key is varied. In our implementation, we divide the image into overlapping circular/elliptical regions with randomly selected radii. Label, these N regions as C_i, $i = 1, 2, ..., N$.

2. (Rectangularization) Approximate each C_i by a rectangle using a $waterfilling$ [35] like approach. Label the resulting random rectangles (consistent with the labels in Step 1) as R_i, $i = 1, 2, ..., N$.

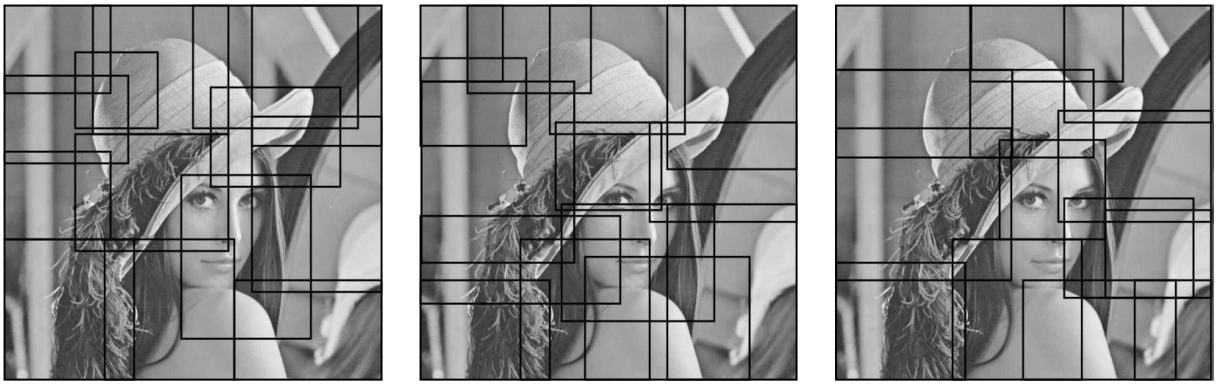
3. (Feature Extraction) Apply Algorithm 1 on all R_i, and denote the binary string extracted from each R_i as $\mathbf{b}_i$. Concatenate all $\mathbf{b}_i$'s into a single binary vector $\mathbf{b}$ of length B bits.

4. (Randomized Subspace Projection) Let $A < B$ be the desired length of $\mathbf{h}(I, K)$. Randomly choose distinct indices $i_1, i_2, ..., i_A$ such that each $i_m \in [1, B], m = 1, 2, ..., A$.

5. The intermediate hash $\mathbf{h}(I, K) = \{\mathbf{b}(i_1), \mathbf{b}(i_2), ..., \mathbf{b}(i_A)\}$

Figure 3.4: Randomized intermediate hash algorithm

The approach of dividing the image into random rectangles for constructing hashes was first proposed by Venkatesan *et al.* in [1]. However, their algorithm is based on image statistics. In the proposed framework, by applying the feature point detector to



(a) Secret key K_1 (b) Secret key K_2 (c) Secret key K_3

Figure 3.5: Examples of random partitioning of the *lena* image into $N = 13$ rectangles. The random regions vary significantly based on the secret key.

these semi-global rectangles, an additional advantage is obtained in capturing any local tampering of image data (results presented later in Section 3.6.2). These rectangles in Fig. 3.5 are deliberately chosen to be overlapping to further reduce vulnerability to malicious tampering. Finally, the randomized sub-space projection step adds even more unpredictability to the intermediate hash. Trade-offs among randomization, fragility and perceptual robustness are analyzed later in Section 3.6.3.

3.6 Results

I compare the binary intermediate hash vectors obtained from two different images for closeness in (normalized) Hamming distance. Recall from Section 2.2 that $(I, I_{ident}) \in \mathcal{I}$ denote a pair of perceptually identical images, and likewise $(I, I_{diff}) \in \mathcal{I}$ represent perceptually distinct images. Then, I require

$$D_H(\mathbf{h}(I), \mathbf{h}(I_{ident})) < \epsilon \tag{3.16}$$

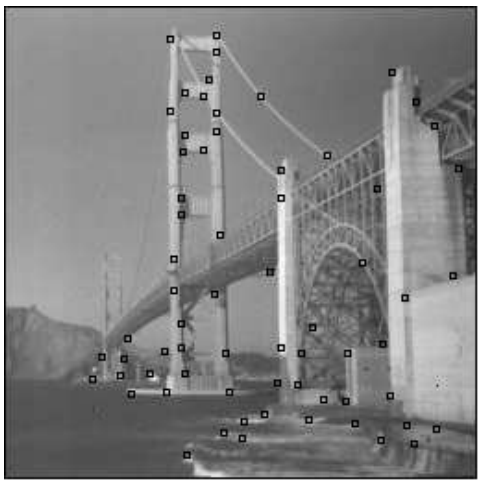
$$D_H(\mathbf{h}(I), \mathbf{h}(I_{diff})) > \delta \tag{3.17}$$

where the natural constraints $0 < \epsilon < \delta$ apply. For results presented in Sections 3.6.1 and 3.6.2, the following parameters were chosen for Algorithm 1: a circular (search) neighborhood of 3 pixels was used in the feature detector, $P = 64$ features were extracted, the order statistics filtering was $OS(3, 3, 4)$ and a zero-phase 2-D FIR low-pass filter of size 5×5 designed using McClellan transformations [36] was employed. For Algorithm 2, the same parameters were used except that the image was partitioned into $N = 32$ random regions. For this choice of parameters, I experimentally determine $\epsilon = 0.2$ and $\delta = 0.3$. A more elaborate discussion of how to choose the best ϵ and δ will be given in Section 3.6.4. All input images were resized to 512×512 using bicubic interpolation [37]. For color images, both Algorithm 1 and 2 were applied to the luminance plane since it contains most of the geometric information.

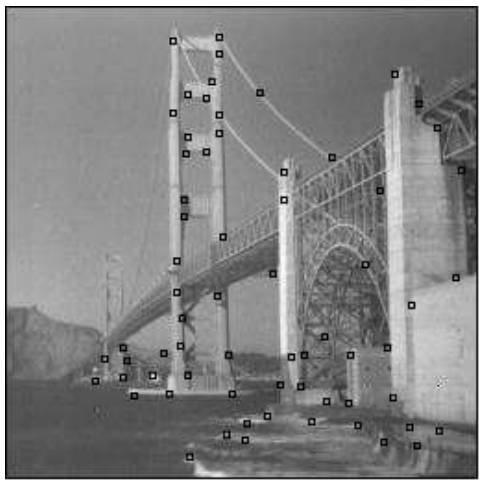
3.6.1 Robustness Under Perceptually Insignificant Modifications

Figs. 3.6 (a)-(d) show four perceptually identical images. The extracted feature points at algorithm convergence are overlayed on the images. The original *bridge* image is shown in Fig. 3.6(a). Figs. 3.6(b), (c), and (d), respectively, are the image in (a) attacked by JPEG compression with quality factor (QF) of 20, rotation of 2^o with scaling, and the Stirmark local geometric attack [38]. It can be seen that the features extracted from these images are largely invariant under these attacks.

Table 3.1 then tabulates the quantitative deviation as the normalized Hamming distance between the intermediate hash values of the original and manipulated images for various perceptually insignificant distortions. The distorted images were generated using the Stirmark benchmark software [38].

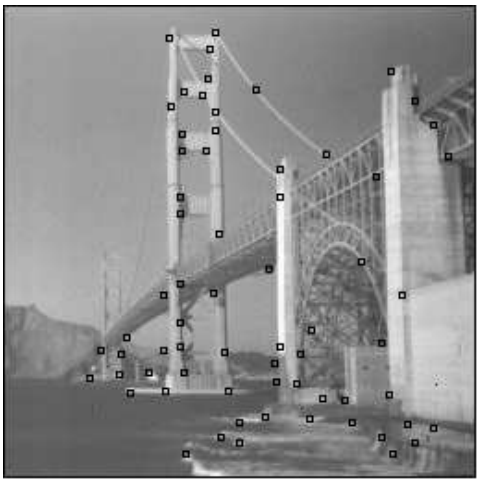


(a) Original Image



(b) JPEG, QF = 10

(c) 2° rotation and scaling



(d) Stirmark local geometric attack

Figure 3.6: Original/attacked images with feature points at algorithm convergence. Feature points overlaid on images.

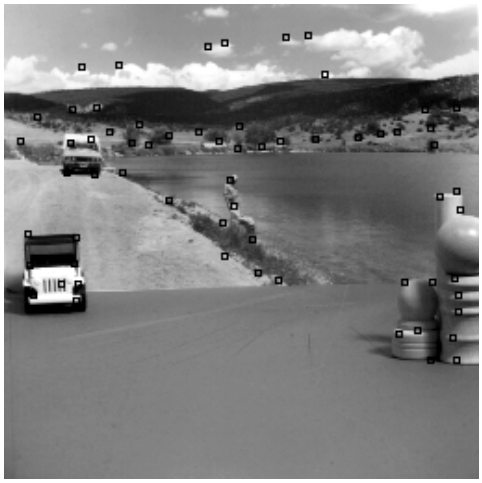
The results in Table 3.1 reveal that the deviation is less than 0.2 except for large rotation (greater than 5°) and cropping (more than 20%).

Attack	*Lena*	*Bridge*	*Peppers*
JPEG, QF = 10	0.04	0.04	0.06
AWGN, $\sigma = 20$	0.04	0.03	0.02
Contrast enhancement	0.00	0.06	0.04
Gaussian smoothing	0.01	0.03	0.05
Median filter (3×3)	0.02	0.03	0.07
Scaling by 60%	0.02	0.04	0.05
Shearing by 5%	0.08	0.14	0.10
Rotation by 3^o	0.13	0.15	0.15
Rotation by 5^o	0.18	0.20	0.19
Cropping by 10%	0.12	0.13	0.15
Cropping by 20%	0.21	0.22	0.24
Random bending	0.15	0.17	0.14
Local geometric attack	0.12	0.02	0.13

Table 3.1: Normalized Hamming distance between intermediate hash values of original and attacked (perceptually identical) images.

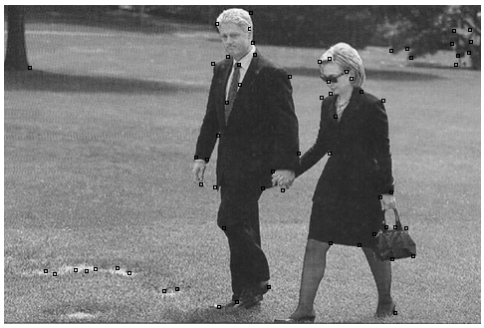
3.6.2 Fragility to Content Changes

The essence of the proposed feature point based hashing scheme lies in projecting the image onto a visually meaningful wavelet basis, and then retaining the strongest coefficients to form the content descriptor (or hash). The particular choice of basis functions, i.e. end-stopped type exponential kernels, yield strong responses in parts of the image where the significant image geometry lies. It is this very characteristic that makes the proposed scheme attractive for detecting content changing image manipulations. In



(a) Original *toys* image

(b) Tampered *toys* image



(c) Original *clinton* image

(d) Tampered *clinton* image

Figure 3.7: Content changing attacks and feature extractor response. Feature points overlayed on the images.

particular, I observe that a visually meaningful content change is effected by making a significant change to the image geometry.

Fig. 3.7 shows two examples of malicious content changing manipulation of image data and the response of the proposed feature extractor to those manipulations. Fig. 3.7 (a) shows the original *toys* image. Fig. 3.7 (b) shows a tampered version of the image in Fig. 3.7 (a), where the tampering is being brought about by addition of a "toy bus". In Fig. 3.7 (d), an example of malicious tampering is shown where the face of the lady in Fig.

3.7 (c) has been replaced by a different face from an altogether different image.

Comparing Figs. 3.7 (a) and (b), and Figs. 3.7 (c) and (d), it may be seen that several extracted features do not match. This observation is natural because the proposed algorithm is based on extracting the P *strongest* geometric features from the image. In particular, in Fig. 3.7 (d), tampering of the lady's face is easily detected because most differences from Fig. 3.7 (c) are seen in that region. Quantitatively, this translates into a large distance between the intermediate hash vectors.

With complete knowledge of the iterative feature extraction algorithm, it may still be possible for a malicious adversary to generate inputs (pairs of images) that defeat the proposed intermediate hash algorithm, e.g. tamper content in a manner such that the resulting features/intermediate hashes are still close. This, however, is much harder to achieve, when the randomized intermediate hash algorithm (Algorithm 2) was used.

I also tested under several other content changing attacks including object insertion and removal, addition of excessive noise, alteration of the position of image elements, tampering with facial features, and alteration of a significant image characteristic such as texture and structure. In all cases, the detection was accurate. That is, the normalized Hamming distance between the image and its attacked version was found to be greater than 0.3. Table 3.2 shows the normalized Hamming distance between intermediate hash values of original and maliciously tampered images for many different content changing attacks. Algorithm 2 with $N = 32$ was used for these results.

3.6.3 Performance Trade-Offs

A large search neighborhood implies that the maxima of wavelet responses are taken over a larger set and hence the feature points are more robust to small perceptually

Attack	*Lena*	*Clinton*	*Barbara*
Object Addition	0.43	0.42	0.46
Object Removal	0.47	0.44	0.52
Excessive Noise Addition	0.53	0.45	0.38
Face Morphing	0.50	0.44	0.34

Table 3.2: Normalized Hamming Distance between intermediate hash values of original and attacked images via content changing manipulations

insignificant perturbations. Likewise, consider selecting the feature points so that $T_1 < \max_\theta W_i(x, y, \theta) < T_2$. Note the feature detection scheme as described in Fig. 3.2 implicitly assumes T_2 to be infinity. If T_1 and T_2 are chosen to be large enough, then the resulting feature points are very robust, i.e. retained in several attacked versions of the image. Similarly, if the two thresholds are chosen to be very low, then the resulting features tend to be easily removed by several perceptually insignificant modifications. The thresholds and the size of the search neighborhood facilitate a perceptual robustness vs. fragility trade-off.

When the number of random partitions N is one, and a deterministic quantization rule is employed in Section 3.4, Algorithms 1 and 2 are the same. If N is very large, then the random regions shrink to an extent that they do not contain significant chunks of geometrically strong components and hence the resulting features are not robust. The parameter N facilitates a randomness vs. perceptual robustness trade-off.

Recall from Section 3.4 that the output of the quantization scheme for binarizing the feature vector is completely deterministic except for the interval (P_i, Q_i). In general, more than one choice of the pair (P_i, Q_i) may satisfy (3.13). Trivial solutions to (3.13)

are (a) $P_i = Q_i = C_i$ and (b) $P_i = l_{i-1}$, $Q_i = l_i$. While (a) corresponds to the case when there is no randomness involved, the choice in (b) entails that the output of the quantizer is always decided by a randomization rule. In general, the greater the value of $\frac{\int_{C_i}^{Q_i} p_f(x)dx}{\int_{C_i}^{l_i} p_f(x)dx}$, the larger the amount of unpredictability in the output. This is a desired property to minimize collision probability. However, this also increases the chance that slight modifications to the image result in different hashes. A trade-off is hence facilitated between perceptual robustness and randomization.

3.6.4 Statistical Analysis

In this section, I present a detailed statistical comparison of our proposed feature-point scheme for hashing against methods based on preserving coarse image representations. In particular, I compare the performance of the proposed intermediate hash based on the end-stopped wavelet transform against the discrete wavelet transform (DWT) and the discrete cosine transform (DCT).

Let $\mathcal{U}$ denote the family of perceptually insignificant attacks on an image $I \in \mathcal{I}$, and let $U \in \mathcal{U}$ be a specific attack. Likewise, let $\mathcal{V}$ represent the family of content changing attacks on I, and let $V \in \mathcal{V}$ be a specific content changing attack. Then, I define the following terms:

Probability of False Positive:

$$P_{fP}(\epsilon) = Probability(D_H(\mathbf{h}(I), \mathbf{h}(V(I))) < \epsilon \tag{3.18}$$

Probability of False Negative:

$$P_{fN}(\delta) = Probability(D_H(\mathbf{h}(I), \mathbf{h}(U(I))) > \delta \tag{3.19}$$

To simplify the presentation, I construct two representative attacks:

- **A strong perceptually insignificant attack** in $\mathcal{U}$: A composite attack was constructed for this purpose. The complete attack (in order) is described as: (1) JPEG compression with QF = 20, (2) 3^o rotation and rescaling to the original size, (3) 10% cropping from the edges, and (4) Additive White Gaussian Noise (AWGN) with $\sigma = 10$ (image pixel values range in 0-255). Fig. 3.8 (a) through (e) show the original and modified *house* images at each stage of this attack.

- **A content changing attack** in $\mathcal{V}$: The content changing attack consisted of maliciously replacing (a randomly selected) region of the image by an alternate unrelated image. An example of this attack for the *lena* image is shown in Fig. 3.9.

For fixed ϵ and δ, the probabilities in (3.18) and (3.19) are computed by applying the aforementioned attacks to a natural image database of 1000 images and recording the failure cases. As ϵ and δ are varied, $P_{fP}(\epsilon)$ and $P_{fN}(\delta)$ describe an ROC (receiver operating characteristic) curve.

All images were resized to 512×512 prior to applying the hash algorithms. For the results to follow, the proposed intermediate hash was formed as described in Section 4.5.1 by retaining the P strongest features. The intermediate hash/feature vector in the DWT based scheme was formed by retaining the lowest resolution sub-band in an M-level DWT decomposition. In the DCT scheme, correspondingly, a certain percentage of the total DCT coefficients were retained. These coefficients would in general belong to a low frequency band (but not including DC, since it is too sensitive to scaling and/or contrast changes).

Fig. 3.10 shows ROC curves for the three schemes for extracting intermediate features of images: preserving low-frequency DCT coefficients, low resolution wavelet coefficients, and the proposed scheme based on end-stopped kernels. Each point on these curves

represents a (P_{fP}, P_{fN}) pair computed as in (3.18) and (3.19) for a fixed ϵ and δ. I used $\delta = \frac{3}{2}\epsilon$ in all cases. For the ROC curves in Fig. 3.10, I varied ϵ in the range $[0.1, 0.3]$. A typical application, e.g. image authentication or indexing, will operate at a point on this curve.

To ensure a fair comparison among the three methods, I consider two cases for each hashing method. For the DWT, ROC curves are shown when a 6-level DWT and 5-level DWT transform were applied. A 6-level DWT on a 512×512 image implies that 64 transform coefficients are retained. In a 5-level DWT, 256 coefficients are retained. Similarly, for the DCT-based scheme two different curves are shown in Fig. 3.10, respectively, corresponding to 64 and 256 low-frequency DCT coefficients. For the proposed intermediate hash, ROC curves corresponding to $P = 64$ and $P = 100$ are shown.

In Fig. 3.10, both the false positive as well as the false negative probabilities are much lower for the proposed intermediate hash algorithm. Predictably, as the number of coefficients in the intermediate hash is increased for either scheme, a lower false positive probability (i.e. fewer collisions of perceptually distinct images) is obtained at the expense of increasing the false negative probability. Recall from Section 3.6.3 that this trade-off can be facilitated in deriving the proposed intermediate hash even with a fixed number of coefficients — an option that the DWT/DCT does not have.

In Fig. 3.10, with $P = 64$ features, the proposed algorithm based on end-stopped kernels vastly outperforms the DCT as well as DWT based intermediate hashes[2] in achieving lower false positive probabilities, even as a much larger number of coefficients is used for

[2]All the wavelet transforms in the MATLAB wavelet toolbox version 7.0 were tested. The results shown here are for the discrete Meyer wavelet "dmey" which gave the best results among all DWT families in the toolbox.

them.

3.7 Conclusion

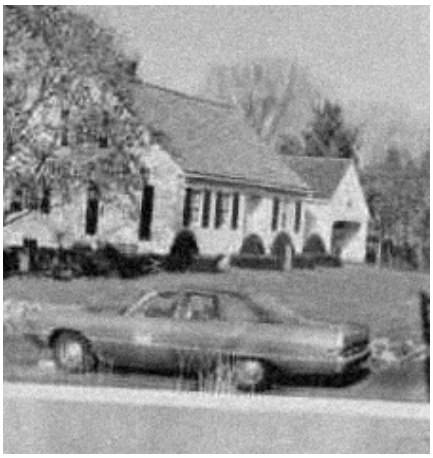
This chapter develops a general framework for constructing intermediate hash vectors from images via visually significant feature points. An iterative feature extraction algorithm based on preserving significant image geometry is proposed. Several robust feature detectors may be used within the iterative algorithm. Parameters in the proposed feature detector enable trade-offs between robustness and fragility of the hash, which are otherwise hard to achieve with traditional DCT/DWT based approaches. I develop both deterministic and randomized algorithms to prevent against guessing and forgery. ROC analysis is performed to demonstrate the statistical advantages of the proposed algorithm over existing schemes based on preserving coarse image representations. The next chapter addresses the problem of compressing the intermediate hash (or feature) vector derived in this chapter to a final hash value.

(a) Original *house* image

(b) JPEG, QF = 20

(c) 3^o rotation and scaling of (b)

(d) Image in (c) cropped 10% on the sides and rescaled to original size



(e) Final attacked image: AWGN attack on the image in (d)

Figure 3.8: Representative perceptually insignificant attack on the *house* image: images after each stage of the attack.

(a) Original Lena Image (b) Tampered lena image

Figure 3.9: Example of the representative content changing attack on the *lena* image: 15% of the image area is being corrupted.

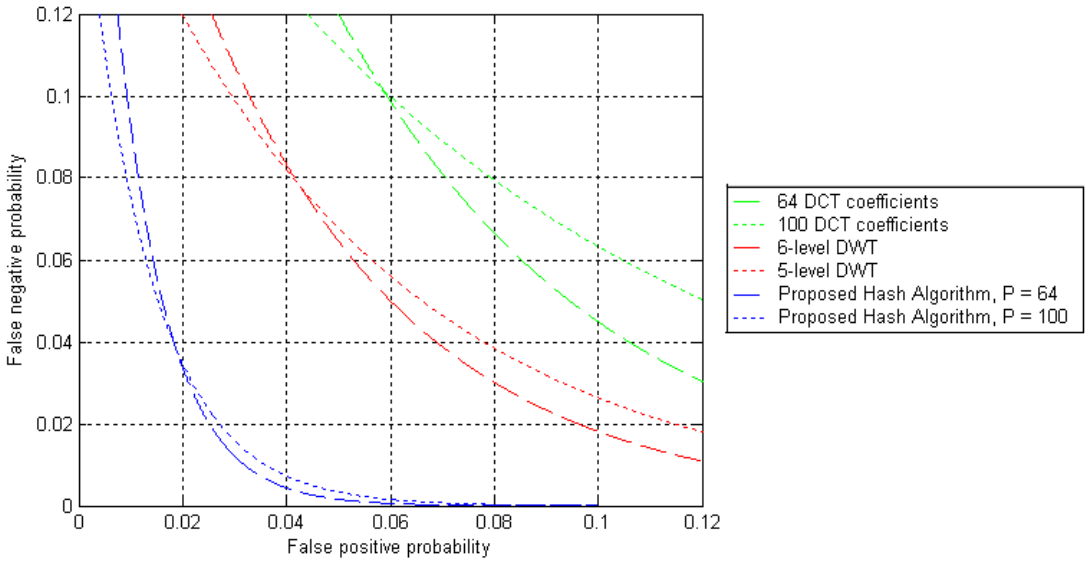


Figure 3.10: ROC curves for hash algorithms based on three approaches: DCT transform, DWT transform, and proposed intermediate hash based on end-stopped kernels. Note that error probabilities are significantly lower for the proposed scheme.

Chapter 4

Clustering Algorithms for Feature Vector Compression

4.1 Introduction

In this chapter, I develop clustering algorithms to compress the feature vector (or intermediate hash) derived in Chapter 3. I prove that the decision version of the underlying clustering problem is NP complete. Then, for any perceptually significant feature extractor, I propose a polynomial-time heuristic clustering algorithm that automatically determines the final hash length needed to satisfy a specified distortion. Based on the proposed algorithm, I develop two variations to facilitate perceptual robustness vs. fragility trade-offs. Finally, I develop randomized clustering algorithms for the purposes of secure image hashing.

Section 4.2 formally defines the problem for the feature vector compression step of the two-step hash function. For this second step, Section 4.3 brings out the limitations of traditional vector quantization (VQ) based compression approaches. Section 4.4 then proposes a new cost function for feature vector (or intermediate hash) compression for the perceptual hashing application. Section 4.5 presents heuristic clustering algorithms for minimizing the cost function defined in Section 4.4. I first present a deterministic algorithm in Section 4.5.1 that attempts to retain the perceptual significance of the hash

as best as possible. Next, a randomized clustering is proposed (based on a secret key) in Section 4.5.2 for the purposes of secure hashing. Experimental results are presented in Sections 4.6.1 through 4.6.3. In Section 4.6.1, I compare with traditional VQ as well as error correction decoding approaches [1] to show the efficacy of the proposed clustering algorithm(s) for perceptual hash compression. Section 4.6.2 presents a statistical analysis of the algorithm using precision-recall (or receiver operating characteristic (ROC)) curves. Section 4.6.3 then presents results that demonstrate security properties of the randomized clustering algorithm. Section 4.7 concludes the chapter by summarizing the central ideas governing the proposed clustering algorithm(s).

4.2 Problem Statement

I first establish notation that will be used throughout this chapter. Let V denote the metric space of intermediate hash vectors extracted at stage 1 of the hash algorithm. Let $\mathcal{L} \subseteq V$ represent a finite set of vectors $\{l_i\}_{i=1}^{n}$ on which the clustering/compression algorithm is applied. Let $D : V \times V \rightarrow \mathcal{R}_+$ be the distance *metric* defined on the product space. Finally, let $C : \mathcal{L} \rightarrow \{1, 2, ..., k\}$ denote the clustering map. Note in a typical application, $k << n$, re-emphasizing the fact that the clustering as well the overall hash is a many-to-one mapping.

Our goal is to have all images that are visually indistinguishable map to the same hash value with high probability. In that sense an image hash function is similar to a *vector quantization* (VQ) or *clustering* scheme. We are attempting to cluster images whose intermediate hash vectors are close in a metric into the same cell. In particular, it is desired that with high probability

$$if\ D(l_i, l_j) < \epsilon \text{ then } C(l_i) = C(l_j) \tag{4.1}$$

$$if\ D(l_i, l_j) > \delta \text{ then } C(l_i) \neq C(l_j) \tag{4.2}$$

where $0 < \epsilon < \delta$. Let l_i, l_j denote random vectors in $\mathcal{L}$ (following the distribution of the intermediate hash) and let $C(l_i)$, $C(l_j)$ represent the clusters to which these vectors map after applying the clustering algorithm.

4.3 Conventional VQ based Compression Approaches

The goal of the compression step as discussed above is to achieve a clustering of the intermediate hash vectors of an image I and the intermediate hash vectors of images that are identical in appearance to I with high probability. In that respect, it is useful to think of perceptually insignificant modifications or attacks on an image as "distortions" to the image. We may then look to compress the intermediate hash vectors while tolerating a specified distortion. The design problem for a vector quantization or compression scheme that minimizes an average distortion is to obtain a K partioning of the space V by designing codevectors $\{c_k\}_{k=0}^{K-1}$ in V such that

$$\sum_{k=0}^{K-1} \sum_{l \in S_k} P(l) D(l, c_k) < \epsilon \tag{4.3}$$

Here, $P(l)$ denotes the probability of occurrence of vector l and S_k denotes the k^{th} cluster. Average distance minimization is a well known problem in the VQ literature and many algorithms [39], [40], [41] have been proposed to solve it.

However, an average distance type cost function as in (4.3) is not inherently well suited for the hashing application. First, while the design of the codebook in (4.3) ensures that the average distortion is less than ϵ, there is no guarantee that perceptually distinct vectors, i.e. intermediate hash vectors that are separated by more than δ, indeed map to different clusters. In some applications, such as image authentication where the goal

is to detect content changes, such guarantees may indeed be required because mapping perceptually distinct vectors to the same final hash value would be extremely undesirable. More generally, the nature of the cost function in (4.3) does not allow trade-offs between desired properties (4.1) and (4.2) of the hash algorithm.

Secondly, the cost in (4.3) increases linearly as a function of the distance between the intermediate hash vector(s) and the codebook vector(s). Intuitively though, it is desirable to penalize some errors much more strongly than others, e.g. if vectors really close are not clustered together, or if vectors very far apart are compressed to the same final hash value. A linear cost function does not reflect this behavior.

Based on these observations, I propose a new cost function for the perceptual hashing application that does not suffer from the limitations of average distance measures.

4.4 Formulation of the Cost Function

In this section, I formulate the cost function to be minimized by the proposed clustering algorithm. First, I analyze several fundamental properties of the requirements in (4.1), (4.2), and the intermediate hash.

An error is encountered when either (4.1) and/or (4.2) is not satisfied for any pair of vectors $(l_i,\, l_j)$. The requirement in (4.1) is actually impossible to guarantee for every input pair. Intuitively then, we must ensure that errors occur for vectors that are less likely or that the clustering must necessarily be dictated by the probability mass function of the vectors in $\mathcal{L}$.

I now describe the construction of our clustering cost function. Let $\mathbf{P} : \mathcal{L} \times \mathcal{L} \rightarrow [0, 1]$

be the joint distribution matrix of intermediate hash pairs

$$\mathbf{P} = \begin{bmatrix} p(1,1) & p(1,2) & \cdots & p(1,n) \\ p(2,1) & p(2,2) & \cdots & p(2,n) \\ \vdots & \vdots & \ddots & \vdots \\ p(n,1) & p(n,2) & \cdots & p(n,n) \end{bmatrix} \tag{4.4}$$

where $\mathbf{P}_{ij} = p(i,j) = p(i)p(j)$. Here, $p(i)$, $p(j)$ respectively denote the probability of occurrence of vectors l_i, l_j and n is the number of vectors in $\mathcal{L}$..

To estimate the probability measure introduced above, I employ a statistical model on the intermediate hash/image feature vectors. The fundamental underlying principle is to define rectangular blocks (or sub-images) in an image as a real two-dimensional homogenous Markov random field (MRF) $X(m_1, m_2)$ on a finite lattice $(m_1, m_2) \in L \subset Z^2$. The basis for connecting such a statistical definition to perception is the hypothesis first stated by Julesz [42] and reformulated by several other authors, e.g. [43], [44] : there exists a set of functions $\phi_k(X), k = 1, 2, \ldots N$ such that samples drawn from any two MRFs that are equal in expectation over this set are visually indistinguishable.

In particular, I employ a universal parametric statistical model for natural images developed by Portilla and Simoncelli [45] that works with a complex overcomplete wavelet representation of the image. Recall the image features that I extract in Chapter 3 are indeed based on such a representation. The Markov statistical descriptors, i.e. ϕ_ks, are then based on pairs of wavelet coefficients at adjacent spatial locations, orientations and scales. In particular, we measure the expected product of the raw coefficient pairs (i.e., correlation), and the expected product of their magnitudes.

There is no inherent structure to the probability mass functions associated with these random fields (except the Markov property due to spatial correlation in images). A

mathematically attractive choice is a maximum entropy density [35] of the form

$$\mathcal{P}(\vec{x}) \propto \prod_k e^{-\lambda_k \phi_k(\vec{x})} \tag{4.5}$$

where $\vec{x} \in \mathcal{R}^{|L|}$ corresponds to a vectorized sub-image, and λ_ks are the Lagrange multipliers. The maximum entropy density is optimal in the sense that it does not introduce any new constraints on the MRF beyond those of perceptual equivalence under expected values of ϕ_ks. The density in (4.5) is defined on MRFs that are portions of natural images. Since features are functions of MRFs a probability density is in turn induced on the feature vectors.

My choice of a statistical model vs. using an empirical distribution on the extracted image features is based on the robustness of model parameters as more samples (images) are added. By the weak law of large numbers, it can be argued that the model parameters become nearly invariant once a sufficiently large sample set is considered (I worked with a set of roughly 2500 natural images [46]). More details on the model parameters and the typical distributions on image feature vectors may be found in [45].

Next, I define $\mathbf{C}_1$ as the joint cost matrix for violating (4.1), i.e. the cost paid if $D(l_i, l_j) < \epsilon$, yet $C(l_i) \neq C(l_j)$. In particular, $\forall\ i, j = 1, 2, ..., n$

$$c_1(i,j) = \begin{cases} \Gamma^{-\alpha D(l_i,l_j)} & \text{if } D(l_i,l_j) < \epsilon,\ C(l_i) \neq C(l_j) \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

where $\alpha > 0$ and $\Gamma > 1$ are algorithm parameters. This construction follows intuitively because the cost for violating (4.1) must be greater for smaller distances, i.e. if the vectors are really close and not clustered together.

Similarly, $\mathbf{C}_2$ is defined as the joint cost matrix for violating (4.2)

$$c_2(i,j) = \begin{cases} \Gamma^{\alpha D(l_i,l_j)} & \text{if } D(l_i,l_j) > \delta,\ C(l_i) = C(l_j) \\ 0 & \text{otherwise} \end{cases} \tag{4.7}$$

In this case however, the cost is an increasing function of the distance between (l_i, l_j). This is also natural as we would like to increase the penalty if vectors far apart (and hence perceptually distinct) are clustered together. An exponential cost as opposed to linear in an average distance VQ ensures that errors associated with large distances are penalized severely. To maintain lucidity, I specify the same parameters i.e., Γ and α in (4.6) and (4.7). This, however, is not a constraint. In general, these parameters may be separately chosen (optimized empirically) for both (4.6) and (4.7).

Further, let matrices $\mathbf{S}_1$ and $\mathbf{S}_2$ be defined as

$$s_1(i,j) = \begin{cases} \Gamma^{-\alpha D(l_i,l_j)} & \text{if } D(l_i,l_j) < \epsilon \\ 0 & \text{otherwise} \end{cases} \tag{4.8}$$

$$s_2(i,j) = \begin{cases} \Gamma^{\alpha D(l_i,l_j)} & \text{if } D(l_i,l_j) > \delta \\ 0 & \text{otherwise} \end{cases} \tag{4.9}$$

Note, that $\mathbf{S}_1$ is different from $\mathbf{C}_1$ in the sense that the entries of $\mathbf{S}_1$ include the cost for all possible errors that can be committed, while $\mathbf{C}_1$ is the cost matrix for the errors actually made by the clustering algorithm. The same holds for $\mathbf{S}_2$ and $\mathbf{C}_2$. Then, I normalize the entries in $\mathbf{C}_1$ and $\mathbf{C}_2$ to define normalized cost matrices $\tilde{\mathbf{C}}_1$ and $\tilde{\mathbf{C}}_2$ such that

$$\tilde{c}_1(i,j) = \frac{c_1(i,j)}{\sum_i \sum_j s_1(i,j)} \tag{4.10}$$

$$\tilde{c}_2(i,j) = \frac{c_2(i,j)}{\sum_i \sum_j s_2(i,j)} \tag{4.11}$$

This normalization ensures that $\tilde{c}_1(i,j), \tilde{c}_2(i,j) \in [0,1]$.

Finally, the total cost function is defined as

$$P_{err} = E[\tilde{\mathbf{C}}_1 + \tilde{\mathbf{C}}_2] \tag{4.12}$$

The expectation is taken over the joint distribution of (l_i, l_j); i.e., (4.12) may be rewritten as

$$P_{err} = \sum_i \sum_j p(i)p(j) \left(\tilde{c}_1(i,j) + \tilde{c}_2(i,j)\right) \tag{4.13}$$

At this point it is worth re-emphasizing that the distance function $D(l_i, l_j)$ can be any function of l_i and l_j that satisfies metric properties, i.e. non-negativity, symmetry and triangle inequality. In particular, I am not restricting $D(\cdot, \cdot)$ to any class of functions other than requiring it to be a metric. In practice, the choice of $D(\cdot, \cdot)$ is motivated by the nature of features extracted in Stage 1 of the hash algorithm.

The two additive terms in (4.12), $E[\tilde{\mathbf{C}}_1]$ and $E[\tilde{\mathbf{C}}_2]$ quantify the errors resulting from violating (4.1) and (4.2), respectively. In particular, $E[\tilde{\mathbf{C}}_1]$ can be interpreted as the expected cost of violating (4.1). Similarly, $E[\tilde{\mathbf{C}}_2]$ signifies the expected cost incurred by violating (4.2). It is this structure of the cost function in (4.12) that our proposed clustering algorithm exploits to facilitate trade-offs between goals (4.1) and (4.2) of the hash algorithm. Note in the special case that $\alpha = 0$, $E[\mathbf{C}_1]$ and $E[\mathbf{C}_2]$ represent the total probability of violating (4.1) and (4.2), respectively.

Indyk *et al.* [47], [48] have addressed a problem similar to the one I present in Section 4.2. They introduce the notion of *locally sensitive hashing* (LSH) [47] and use it to develop sublinear time algorithms for the *approximate nearest neighbor search* (NNS) problem [49] in high dimensional spaces. The key idea in their work is to use hash functions [50], [51] such that the probability of collision is much higher for vectors that are close to each other than for those that are far apart. However, while they prove the existence of certain parametrized LSH families [47], they do not concern themselves with the problem of codebook design for specific cost functions. Instead, their work focuses on developing fast algorithms for the NNS problem based on the availability of such hash

codebooks. My objective here is to develop a clustering algorithm or equivalently design a codebook to minimize the cost function in (4.12) that is well suited for the perceptual image (or media) hashing application.

4.5 Proposed Clustering Algorithms

Finding the optimum clustering that would achieve a global minimum for the cost function in (4.12) is a hard problem. The decision version of the problem "for a fixed number of clusters k, is there a clustering with a cost less than a constant?" is NP-complete. I sketch a proof of NP completeness in Appendix A. Hardness results for the search version, that actually finds the minimum cost solution, can be similarly shown. I present a polynomial-time *greedy heuristic* for solving the problem.

4.5.1 Deterministic Clustering

For the following discussion, vectors in $\mathcal{L}$ will be referred to as "data points". Fig. 4.1 describes the basic clustering algorithm. A visualization of the same is shown in Fig. 4.2. The data points in the input space are covered to a large extent by hyperspheres (clusters) of radius $\frac{\epsilon}{2}$. For each pair of points $(l_i, l_j) \in S_k$ and cluster center l^k, we have

$$D(l_i, l_j) \leq D(l_i, l^k) + D(l^k, l_j) \tag{4.14}$$

This is true because $D(\cdot, \cdot)$ defines a metric. By virtue of Steps 3 and 5 of the basic clustering algorithm, $D(l_i, l^k) < \frac{\epsilon}{2}$, $D(l^k, l_j) < \frac{\epsilon}{2}$ and hence $D(l_i, l_j) < \epsilon$. The algorithm therefore attempts to cluster data points within ϵ of each other and in addition the cluster centers are chosen based on the strength of their probability mass function. This ensures that "perceptually close" data points are clustered together with a very high likelihood. At this stage, we make the following observations about the basic clustering algorithm:

1: Obtain user defined parameters ϵ and δ. Set the number of clusters $k = 1$.

2: Select the data point associated with the highest probability mass, and label it l^1

3: Make the first cluster by including all data points l_j such that $D(l^1, l_j) < \frac{\epsilon}{2}$

4: $k = k + 1$. Select the highest probability data point l^k among the unclustered points such that $\min_{S \in \mathcal{C}} D(l^k, S) \geq \frac{3}{2}\epsilon$ where S is any cluster and $\mathcal{C}$ denotes the set of clusters formed up to this step of the algorithm. $D(l^k, S)$ is calculated using the notion of distance from a set given by $D(x, S) = \min_{y \in S} D(x, y)$

5: Form the k^{th} cluster S_k by including all unclustered data points l_j such that $D(l^k, l_j) < \frac{\epsilon}{2}$

6: Repeat steps 4–5 until no more clusters can be formed.

Figure 4.1: Basic clustering algorithm.

- The minimum distance between any two members of two different clusters has a lower bound of ϵ and hence there are no errors from violating (4.1), which is guaranteed by Step 4 of the basic clustering algorithm.
- Within each cluster the maximum distance between any two points is at most ϵ, and because $0 < \epsilon < \delta$, there are no violations of (4.2).
- The data points that are left unclustered are less than $\frac{3}{2}\epsilon$ from any member of each of the clusters.

For perceptual robustness, i.e. achieving (4.1), we would like to minimize $E[\tilde{\mathbf{C}}_1]$. Likewise, in order to maintain fragility to visually distinct inputs, we would like $E[\tilde{\mathbf{C}}_2]$ to be as small as possible (ideally zero). Exclusive minimization of one would compromise the

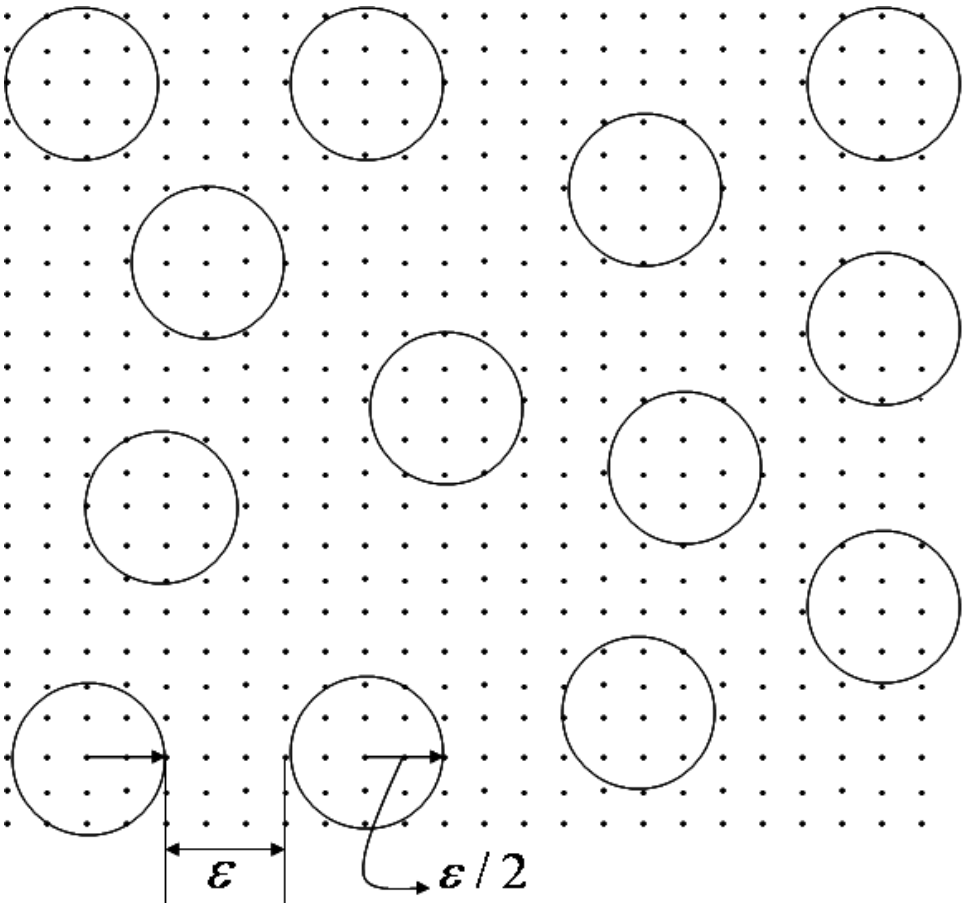


Figure 4.2: Visualization of the Basic clustering algorithm given by Fig. 4.1

other. Next, I present two different approaches to handle the unclustered data points so that trade-offs may be facilitated between achieving properties (4.1) and (4.2).

4.5.1.1 *Approach 1*

Fig. 4.3 describes Approach 1 for handling the unclustered data points. Step 2 of the algorithm in Fig. 4.3 looks for the set of clusters $\mathcal{S}_\delta$, such that every point in each of the clusters is less than δ away from the unclustered data point l^* under consideration.

1: Given the k clusters formed by running the basic clustering algorithm, select the data point l^* among the unclustered points that has the highest probability mass

2: For each existing cluster S_i, $i = 1, 2, ...k$, compute $d_i = \max_{x \in S_i} D(l^*, x)$

Let $\mathcal{S}_\delta = \{S_i \text{ such that } d_i \leq \delta\}$

3: IF $\mathcal{S}_\delta = \phi$ THEN $k = k + 1$ and $S^k = l^*$ is a cluster of its own

ELSE for each $S_i \in \mathcal{S}_\delta$ define $F(S_i) = \sum_{l \in \bar{S}_i} p(l)p(l^*)c_1(l, l^*)$

where $\bar{S}_i$ denotes the complement of S_i; i.e., all clusters in $\mathcal{S}_\delta$ except S_i. Then, l^* is assigned to the cluster $S^* = \arg\min_{S_i} F(S_i)$

4: Repeat steps 1–3.

Figure 4.3: Approach 1 clusters remaining data points such that $E[\tilde{\mathbf{C}}_2] = 0$ where $\tilde{\mathbf{C}}_2$ is defined by (4.11).

Step 3 then computes the minimum cost cluster to which to assign l^*. In essence, this approach tries to minimize the cost in (4.12) conditioned on the fact that there are no errors from violating (4.2). This could be useful in authentication applications in which mapping perceptually distinct inputs to the same hash may be extremely undesirable.

4.5.1.2 *Approach 2*

Approach 1 clusters the remaining data points to ensure that $E[\tilde{\mathbf{C}}_2] = 0$. The goal in Approach 2 is to effectively trade-off the minimization of $E[\tilde{\mathbf{C}}_1]$ at the expense of increasing $E[\tilde{\mathbf{C}}_2]$ via a tuning parameter[1] β (see Fig. 4.4). This can be readily observed by

[1] $\beta \in [\frac{1}{2}, 1]$ as opposed to $[0, 1]$. This is because values of $\beta \in [0, \frac{1}{2})$ do not lead to meaningful

1: Given the k clusters formed by running the basic clustering algorithm, select the data point l^* among the unclustered points that has the highest probability mass

2: For each existing cluster S_i, $i = 1, 2, ...k$, define

$F(S_i) = \beta \sum_{l \in \bar{S}_i} p(l)p(l^*)c_1(l, l^*) + (1 - \beta) \sum_{l \in S_i} p(l)p(l^*)c_2(l, l^*)$

where $\beta \in [\frac{1}{2}, 1]$, and $\bar{S}_i$ denotes the complement of S_i. Then, l^* is assigned to the cluster $S^* = \arg\min_{S_i} F(S_i)$. Analogous to Approach 1, this includes the case that l^* is a cluster by itself; in that case, k is incremented.

3: Repeat steps 1–2.

Figure 4.4: Approach 2 enables trade-offs between goals (4.1) and (4.2) by varying the real-valued parameter β.

considering extreme values of β. For $\beta = \frac{1}{2}$ a joint minimization is performed. The other extreme $\beta = 1$ corresponds to the case when the unclustered data points are assigned, so as to exclusively minimize $E[\tilde{\mathbf{C}}_1]$. For $\delta \geq \frac{5}{2}\epsilon$, Approaches 1 and 2 coincide because all of the unclustered points are then necessarily within δ of the existing clusters. Finally, note that a meaningful dual of Approach 1 does not exist. This is because requiring $E[\tilde{\mathbf{C}}_1] = 0$ leads to the trivial solution that all data points are collected in one big cluster.

In traditional VQ based compression approaches, the number of codebook vectors or the *rate* of the vector quantizer [39] is decided in advance and an optimization is carried out to select the best codebook vectors. In our algorithm, the length of the hash (given

clusterings. For example, $\beta = 0$ ignores the minimization of $E[\tilde{\mathbf{C}}_1]$ which is the primary objective of the algorithm.

by $\lceil \log_2(k) \rceil$ bits) is determined adaptively for a given ϵ, δ and source distribution. Note, however, that I do not claim for this to be the minimum possible number of clusters that achieves a particular value of the cost function in (4.12). Nevertheless, the length of the hash in bits (or alternatively the number of clusters) as determined by our proposed clustering is enough so that the perceptual significance of the hash is not compromised.

Remark: Note that another difference from compression applications is the fact that compression entails the design of reconstruction values as well (in addition to quantization bins/clusters or Vornoi regions). In the hashing application, however, these may be chosen for convenience (e.g., a straightforward enumeration using $\lceil \log_2(k) \rceil$ bits for k clusters) as long as the notion of closeness is preserved.

4.5.2 Randomized Clustering

The clustering algorithm as presented in the previous subsection is a perfectly deterministic map; i.e. a particular input intermediate hash vector always maps to the same output hash value. I now present a randomization scheme to enhance the security properties of our hash algorithm and minimize its vulnerability to malicious inputs generated by an adversary.

Recall that the heuristic employed in the deterministic algorithm (for both Approaches 1 and 2) was to select the vector or data point with the highest probability mass among the candidate unclustered data points as the cluster center. In other words, the data point that has the highest probability mass is selected as the cluster center with probability equal to one. The randomization rule that I propose modifies this heuristic to select cluster centers in a probabilistic manner. That is, there is a non-zero probability of selecting each candidate unclustered data point as the cluster center. This probability

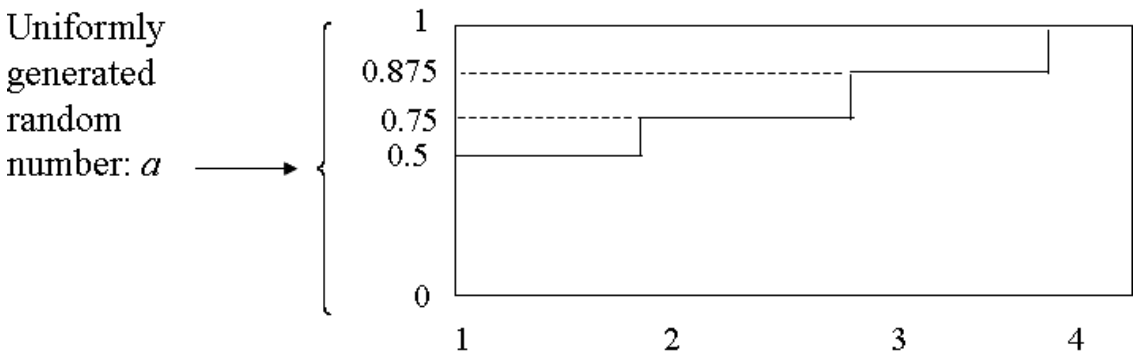


Figure 4.5: Example selection of data points as cluster centers in a probabilistic sense

in turn is determined as a function of the original probability mass associated with the data points.

Consider the clustering algorithm with $m \geq 0$ clusters already formed and $i < n$ points clustered. Let $\mathcal{X} \subset \mathcal{L}$ denote the set of unclustered data points that can be chosen as cluster centers. Note that $|\mathcal{X}|$ is not necessarily $n - i$. As described in the basic clustering algorithm (Fig. 4.1) the set $\mathcal{X}$ consists of all data points $l \in \mathcal{L}$ such that $\min_{S \in \mathcal{C}} D(l, S) \geq \frac{3}{2}\epsilon$ where S is any cluster and $\mathcal{C}$ denotes the set of clusters formed prior to this step of the algorithm. When no more cluster centers can be identified in this manner, the set $\mathcal{X}$ indeed consists of all unclustered data points.

Then, a probability measure on the elements of $\mathcal{X}$ may be defined as

$$\pi_i^{(s)} = \frac{(p_i)^s}{\sum_{j \in \mathcal{X}} (p_j)^s} \tag{4.15}$$

where $s \in \mathcal{R}^+$ is an algorithm parameter and p_i denotes the probability mass associated with data point $l_i \in \mathcal{X}$. The data point $l_i \in \mathcal{X}$ is then chosen as a cluster center with a probability equal to $\pi_i^{(s)}$ [52].

Example: A hypothetical example is presented in Fig. 4.5. In the example, the set $\mathcal{X}$ consists of four data points $\{l_1, l_2, l_3, l_4\}$ with probability mass values of 0.4, 0.2, 0.1

and 0.1, respectively. The normalized probabilities $\{\pi_i^{(s)}\}_{i=1}^4$ using $s = 1$ are given by $\pi_1^{(1)} = 0.5$, $\pi_2^{(1)} = 0.25$, $\pi_3^{(1)} = 0.125$, and $\pi_4^{(1)} = 0.125$. A secret key K_1 is used to serve as a seed to a pseudorandom number generator that generates a uniformly distributed number a in [0,1] which in turn is used to select one of the data points as the cluster center. Note that the probability that $a \in [0, 0.5]$ is 0.5 and hence the data point l_1 is selected with a probability of 0.5. In general, any data point l_i is selected with probability $\pi_i^{(s)}$. This is indeed the classical approach of sampling from a distribution.

The randomization scheme can be summarized by considering extreme values of s. Note

$$\lim_{s\to\infty} \pi_i^{(s)} = \begin{cases} 1 & \text{for the highest probability data point} \\ 0 & \text{for all other } l_i \in \mathcal{X} \end{cases}$$

In other words, $s \to \infty$ corresponds to the deterministic clustering algorithm. Similarly, the other extreme, i.e. $s = 0$, implies that $\pi_i^{(0)}$ is a uniform distribution or that any data point in $\mathcal{X}$ is selected as a cluster center with the same probability equal to $\frac{1}{|\mathcal{X}|}$. To enhance security, the parameter s may also be generated in random fashion using a second secret key K_2.

In general, in the absence of the secret keys K_1 and K_2, it is not possible to determine the mapping achieved by the randomized clustering algorithm. I demonstrate the hardness of generating malicious inputs by means of experimental results in Section 4.6.3.

4.6 Experimental Results

As in Chapter 3, the intermediate hash (or feature) vector extracted from an image I will be referred to as $\mathbf{h}(I)$. Recall further, it was determined that

$$D(\mathbf{h}(I), \mathbf{h}(I_{ident})) < 0.2 \tag{4.16}$$

$$D(\mathbf{h}(I), \mathbf{h}(I_{diff})) > 0.3 \tag{4.17}$$

In our clustering framework, the two equations above yield $\epsilon = 0.2$ and $\delta = 0.3$.

4.6.1 Deterministic Clustering Results

4.6.1.1 *Comparison with Error Correction Decoding and Conventional VQ*

In the following experiments, I extract a binary intermediate hash vector of length $L = 240$ bits from the image. V is, therefore, the Hamming space of dimension L. Further, for this case $\mathcal{L} = V$ and hence the total number of vectors to be clustered i.e., $n = 2^{240}$. Because of space and complexity constraints it is clearly impractical to apply the clustering algorithm to that large of a data set. Hence, I take the approach commonly employed in space constrained VQ problems [39], i.e. divide the intermediate hash vector into segments of length $M = \frac{L}{m}$ (where m is an integer) and apply the clustering on each segment separately. The resulting binary strings are concatenated to form the final hash. A similar approach for an irreversible compression of binary hash values was used by Venkatesan *et al.* in [1]. They employ error control decoding using Reed-Muller codes [18]. In particular, they break the hash vector to be compressed into segments of length as close as possible to the length of codevectors in a Reed-Muller error correcting code. Decoding is then performed by mapping the segments of the hash vector to the nearest codeword using the exponential pseudo norm [1].

Tables 4.1, 4.2, and 4.3, respectively, show values of the cost function in (4.12) by compressing the intermediate hash vector using 1) the proposed clustering scheme, 2) error control decoding scheme as described in [1] and 3) an average distance VQ approach [39]. The results in Table 4.1 were generated by using Approach 2 with $\beta = \frac{1}{2}$. For the error control decoding scheme, (8,4), (16,5) and (16,11) Reed-Muller codes were

M	$E[\tilde{\mathbf{C}}_1]$	$E[\tilde{\mathbf{C}}_2]$	*Final Hash Length*
8	1.86×10^{-5}	2.372×10^{-7}	102 bits
16	1.219×10^{-7}	5.70×10^{-9}	54 bits

Table 4.1: Compression of intermediate hash vectors using the proposed clustering. M is the segment length in bits. $\tilde{\mathbf{C}}_1$ and $\tilde{\mathbf{C}}_2$ are defined in (4.10) and (4.11), respectively. $E[\tilde{\mathbf{C}}_1]$ and $E[\tilde{\mathbf{C}}_2]$ represent the measures of violating desirable hash properties in (4.1) and (4.2), respectively.

used. Our proposed clustering algorithm as well as the average distance VQ compression were also employed on segments of the same length to yield a meaningful comparison[2]. Note that VQ compression [39] based on descent methods that gradually improve the codebook by iteratively computing centroids cannot be applied here since the vectors to be compressed are themselves binary (i.e., codebook vector components cannot assume values between 0 and 1). For the results in Table 4.3, the binary VQ compression based on "soft-centroids" proposed by Franti *et al.* [41] was used.

The results in Tables 4.1, 4.2 and 4.3 clearly reveal that the values for the expected cost of violating (4.1) and (4.2), i.e. $E[\tilde{\mathbf{C}}_1]$ and $E[\tilde{\mathbf{C}}_2]$, are orders of magnitude lower when using our clustering algorithm (even as better compression is achieved for the proposed clustering). Hence, I show that the codebook as obtained from using error correcting codes and/or conventional VQ based compression approaches does not fare as well for

[2]For an average distance VQ the rate or the number of codebook vectors is to be decided in advance. This number was decided upon by determining first the number of clusters (or equivalently the hash length in bits) that result from the application of our proposed clustering and then using a rate slightly higher than that for the average distance VQ. This ensures a fair comparison across the two methods.

M	$E[\tilde{\mathbf{C}}_1]$	$E[\tilde{\mathbf{C}}_2]$	*Final Hash Length*
8	1.526×10^{-3}	5.55×10^{-4}	120 bits
16	9.535×10^{-2}	6.127×10^{-3}	75 bits
16	5.96×10^{-4}	3.65×10^{-5}	165 bits

Table 4.2: Compression of intermediate hash vectors using error control decoding. M is the segment length in bits. $\tilde{\mathbf{C}}_1$ and $\tilde{\mathbf{C}}_2$ are defined in (4.10) and (4.11), respectively. $E[\tilde{\mathbf{C}}_1]$ and $E[\tilde{\mathbf{C}}_2]$ represent the measures of violating desirable hash properties in (4.1) and (4.2), respectively.

M	$E[\tilde{\mathbf{C}}_1]$	$E[\tilde{\mathbf{C}}_2]$	*Final Hash Length*
8	1.44×10^{-3}	5.88×10^{-4}	120 bits
16	3.65×10^{-4}	7.77×10^{-5}	60 bits

Table 4.3: Compression of intermediate hash vectors using a conventional average distance VQ. M is the segment length in bits. $\tilde{\mathbf{C}}_1$ and $\tilde{\mathbf{C}}_2$ are defined in (4.10) and (4.11), respectively. $E[\tilde{\mathbf{C}}_1]$ and $E[\tilde{\mathbf{C}}_2]$ represent the measures of violating desirable hash properties in (4.1) and (4.2), respectively.

perceptual hash compression.

Remark: The proposed clustering algorithm can be used to compress feature vectors as long as the distance measure defined on the product space $V \times V$ satisfies metric properties. For example, if the features were to be real valued, the number of data points n or equivalently the set $\mathcal{L}$ should be chosen large enough to sufficiently represent source (feature vector) statistics. A codebook can then be derived from the set $\mathcal{L}$ using the proposed clustering and feature vectors can be mapped to the nearest vector in the

codebook based on a minimum cost decoding rule [39].

4.6.1.2 *Perceptual Robustness vs. Fragility Trade-offs*

Table 4.4 compares the value of the cost function in (4.12) for the two different clustering approaches. For Approach 2 (rows 2 and 3 of Table 4.4) the value of $E[\tilde{\mathbf{C}}_1]$ is lower than that for Approach 1. In particular, it can be shown that (via our clustering algorithm) the lowest value of the cost function is obtained using Approach 2 with $\beta = \frac{1}{2}$. Trade-offs are facilitated in favor of (4.1) by minimizing $E[\tilde{\mathbf{C}}_1]$ using Approach 2 with $\beta \in (\frac{1}{2}, 1]$ and in favor of (4.2) by employing Approach 1. For these results, the clustering algorithm was applied to segments of length $M = 20$ bits.

Clustering Algorithm	$E[\tilde{\mathbf{C}}_1]$	$E[\tilde{\mathbf{C}}_2]$
Approach 1	7.64×10^{-8}	0
Approach 2, $\beta = \frac{1}{2}$	7.43×10^{-9}	7.464×10^{-10}
Approach 2, $\beta = 1$	7.17×10^{-9}	4.87×10^{-9}

Table 4.4: Cost function values using Approaches 1 and 2 with trade-offs numerically quantified.

4.6.1.3 *Validating the Perceptual Significance*

I applied the two-stage hash algorithm (using Approach 2 with $\beta = \frac{1}{2}$) on a natural image database of 100 images [46]. The final hash length obtained was 46 bits. For each image, 20 perceptually identical images were generated using the Stirmark software [38], [53]. The attacks implemented on the images included JPEG compression with quality factors varying from 10 to 80, adding white Gaussian noise (AWGN), enhancing contrast,

non-linear (e.g., median) filtering, scaling and random shearing, and small rotation and cropping. The resulting hash values for the original image and its perceptually identical versions were the same in over 95% cases.

I also compared hash values for all possible pairings of the 100 distinct images (4950 pairs). One collision case was observed[3]. For all other cases the hash values (on a pairwise basis) were very far off. In general, the performance of our hash function is limited by the robustness of the feature detector.

For the same set of images, using an average distance VQ for feature vector compression resulted in about a 70% success rate of mapping perceptually identical versions to the same hash value. In addition, 40 collision cases (same hash value for perceptually distinct images) were observed.

4.6.2 Precision Recall or ROC Analysis

I now present a detailed statistical comparison of our proposed clustering with the average distance VQ and error correcting deccoding using precision-recall (or ROC) curves [54].

The precision-recall terminology comes from document retrieval, where precision quantifies (for a given query) how many of the returned documents are correct. Recall, on the other hand, measures how many correct documents were returned. Fig. 4.6 illustrates this scenario. In this case, recall can be improved by simply returning as large a set as possible. This, however, will heavily compromise the precision of the search.

[3]The results for the randomized clustering algorithm by appropriately choosing s (detailed in Section 4.6.3) were very similar to the ones reported here. In particular, the same trend was observed over several different choices of the secret key K_1.

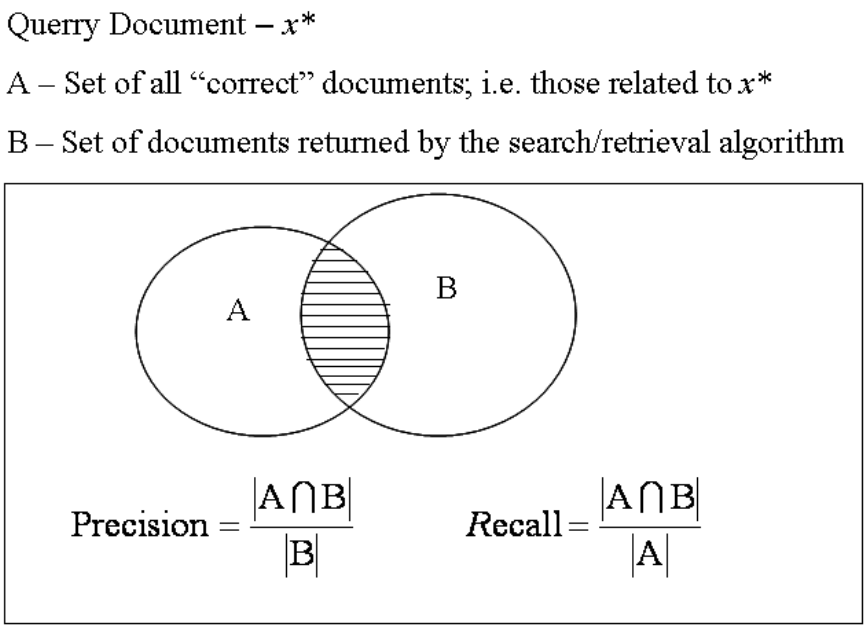


Figure 4.6: Illustration of Precision and Recall in a document retrieval scenario

A precision-recall curve illustrates this trade-off and provides valuable insight especially for problems in which absolute maximization of precision and/or recall is possible only via trivial solutions. For our problem in Section 4.2, I employ the notion of pairwise precision [54] in the following manner

$$Prec_\epsilon = \frac{|X_S \cap X_A|}{|X_A|} \tag{4.18}$$

where $X_S = \{(l_i, l_j) \mid D(l_i, l_j) < \epsilon\}$ is the set of all pairs that should be in the same cluster. X_A then denotes the set of pairs that a given algorithm A puts in the same cluster.

Similarly, pairwise recall is defined as

$$Rec_\epsilon = \frac{|X_S \cap X_A|}{|X_S|} \tag{4.19}$$

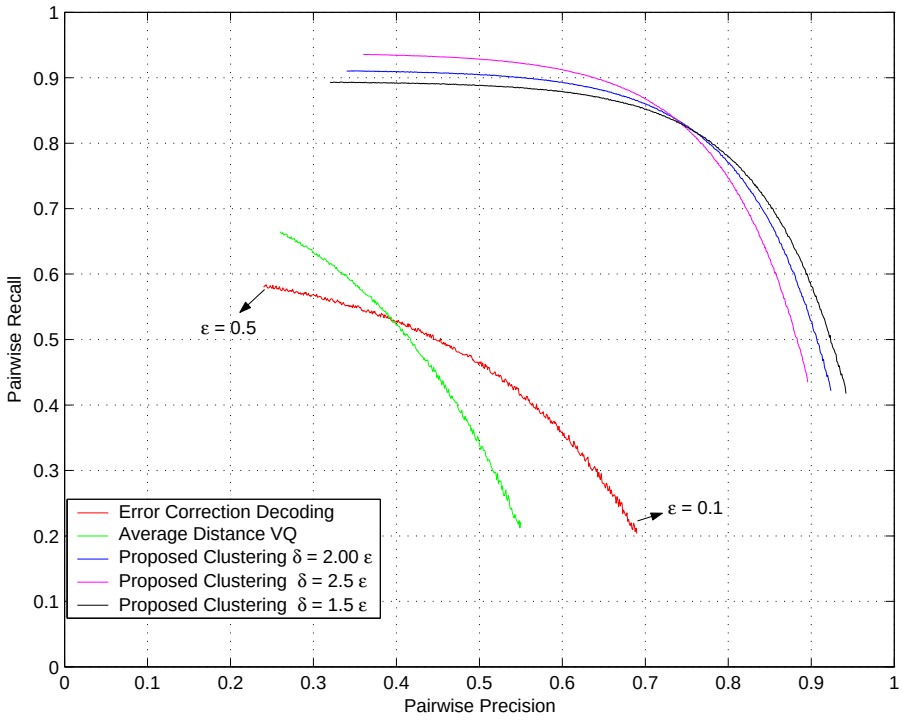


Figure 4.7: Precision-recall curves for three compression approaches: traditional VQ, error correction decoding, and proposed clustering. Each curve results from varying $\epsilon \in [0.1, 0.5]$, with the leftmost point corresponding to $\epsilon = 0.5$.

Clearly, $0 \leq Prec_\epsilon \leq 1$, $0 \leq Rec_\epsilon \leq 1$ (recall may trivially be made 1 by putting all vectors in the same cluster). Fig. 4.7 shows an analysis via precision-recall curves, of three algorithms: 1) average distance VQ, 2) error correction decoding (ECD), and 3) the proposed clustering. Each point on the curve(s) in Fig. 4.7 is a precision-recall pair for a particular value of ϵ, i.e. the precision and recall values computed using (4.18) and (4.19) when the algorithm is run for that ϵ. As indicated in Fig. 4.7, for each curve ϵ was varied in the range $[0.1, 0.5]$.

Comparing the precison-recall curves for the average distance VQ and ECD, it may be observed that the average distance VQ affords a better recall rate at the cost of

loosing precision which is higher for ECD. This explains partially the higher number of collisions in the hash values for perceptually distinct images using the average distance VQ. Note that both the precision as well as recall values are much higher using our proposed clustering algorithm[4].

Note also that there are three different curves for our proposed clustering algorithm. These correspond to different choices of δ (as a function of ϵ) in our algorithm. The average distance VQ and ECD do not have a δ parameter; hence, I present results of the proposed clustering for different δ to ensure a fair comparison between the three schemes. This also provides insight on how δ may be chosen for a given ϵ (which is typically determined empirically from the feature space) to attain greater flexibility in the precision-recall trade-offs.

4.6.3 Security Experiments

An important observation underlying the need for randomization is the fact that feature extraction is seldom perfect. That is by means of thorough analysis it may be possible for an adversary to manipulate image content and yet generate vectors over the feature space that are close. The goal of randomization is hence to make the job of defeating the hash algorithm significantly harder.

A malicious adversary may try to accomplish the same in one of two ways:

1. The adversary may try to generate perceptually identical inputs for which the hash

[4]The precision-recall values plotted in Fig. 4.7 are based on a simple counting of the cardinalities of the sets X_S, X_A, etc. That is there is no weighting by the probability mass of features. In practice, the weighted precision-recall are both pretty close to 1 by using our proposed clustering as illustrated by the results in Section 4.6.1.3.

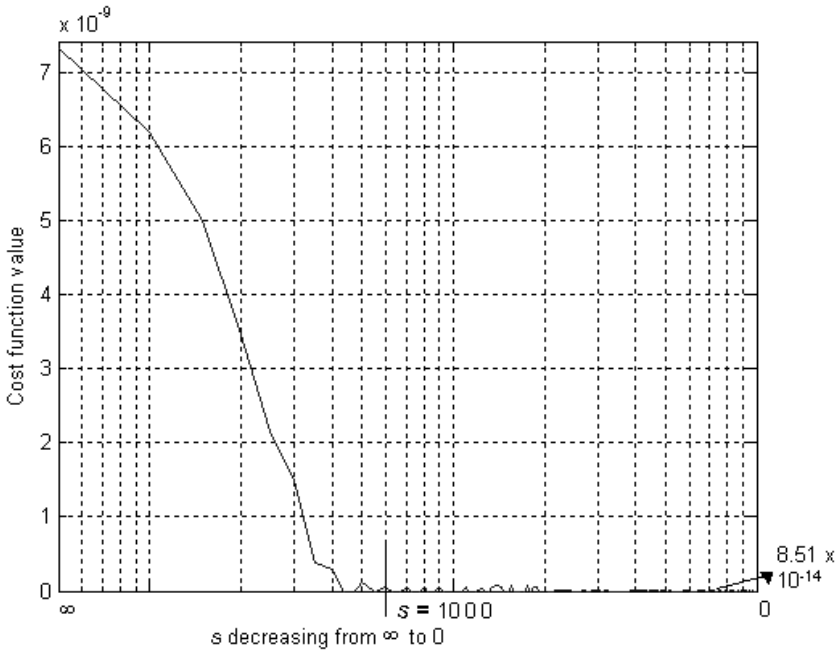


Figure 4.8: Clustering cost function computed over the set E. E is the set of intermediate hash vector pairs over which the deterministic clustering makes errors and s is the randomization parameter.

algorithm generates different hash values, or

2. The adversary may attempt to tamper with the content so as to cause significant perceptual changes such that the hash algorithm generates the same hash value.

I assume here that the adversary has complete knowledge of the intermediate hash (or feature) vector extraction as well as the deterministic clustering algorithm for intermediate hash vector compression. Hence, the adversary is capable of analyzing the algorithm and would attempt to generate inputs over the set $E \subset U$, where U represents the set of all possible pairs of intermediate hash vectors and E is the set of intermediate hash vector pairs over which the deterministic clustering algorithm makes errors.

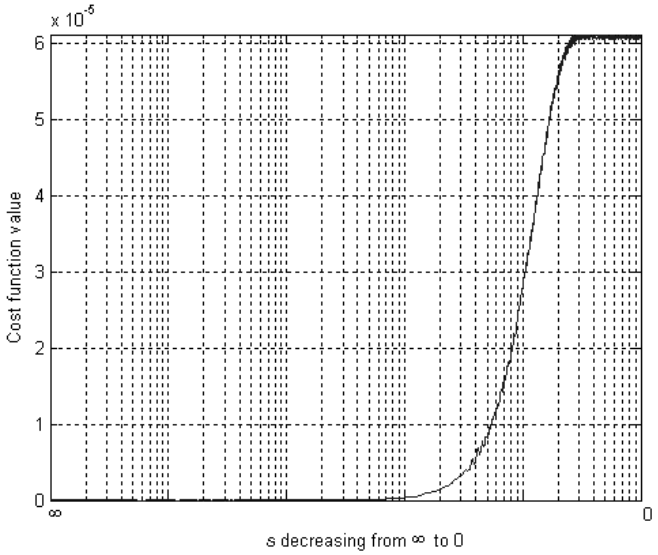
4.6.3.1 *Security Via Randomization*

For the results presented next, the randomized clustering algorithm in Section 4.5.2 was employed with Approach 2 and $\beta = \frac{1}{2}$. Fig. 4.8 shows a plot of the cost in (4.12) computed over the set E against values of s decreasing from ∞ to 0. It can be seen that the cost decreases with s (although not monotonically) and is reduced by orders of magnitude for values of $s < 1000$. Decreasing s is tantamount to increasing randomness. Hence, the plot in Fig. 4.8 reveals that as randomness is increased beyond a certain level, the adversary meets with very little success by generating input intermediate hash pairs over the set E.

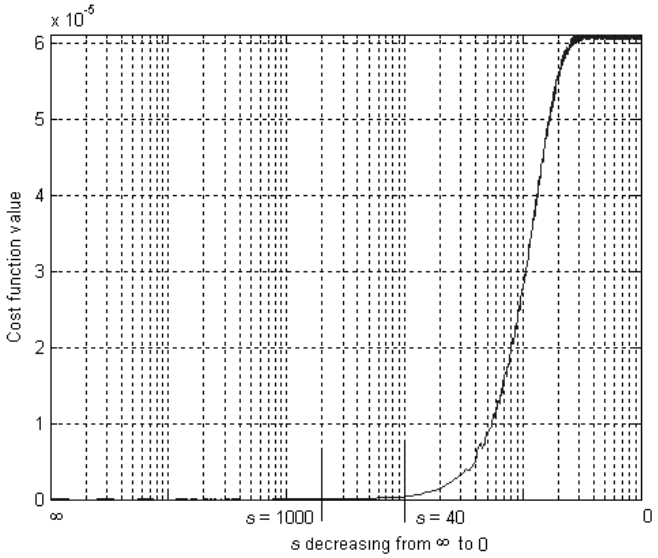
4.6.3.2 *Randomness vs. Perceptual Significance Trade-offs*

Let $\bar{E}$ denote the complement set of E, i.e. the set of all intermediate hash vector pairs over which no errors are made by the deterministic clustering algorithm. Fig. 4.9 (a) then shows the plot of the clustering cost function against decreasing s as before. In this case, the cost increases with decreasing s (again not monotonically). As $s \rightarrow \infty$, the cost is zero since the deterministic clustering algorithm makes no errors over the set $\bar{E}$.

Fig. 4.9 (b) shows a sum of the cost in the two plots in Figs. 4.8 and 4.9 (a). This plot therefore shows the total cost computed over the set U as a function of s. Figs. 4.10 (a) and 4.10 (b), respectively, show the same cost function plots as in Figs. 4.9 (a) and 4.9 (b) but with the $y-$axis in log-scale. As s approaches 0, the value of the cost is increased significantly over the cost incurred by the deterministic algorithm. The cost achieved by the deterministic algorithm is the value of the cost function in Fig. 4.9 (b) (or Fig. 4.10 (b)) as $s \rightarrow \infty$ and equal to 7.43×10^{-9}. At $s = 0$, the total cost is 6.12×10^{-5}. This increase is intuitive as complete randomness (i.e. $s = 0$) would affect the perceptual



(a)

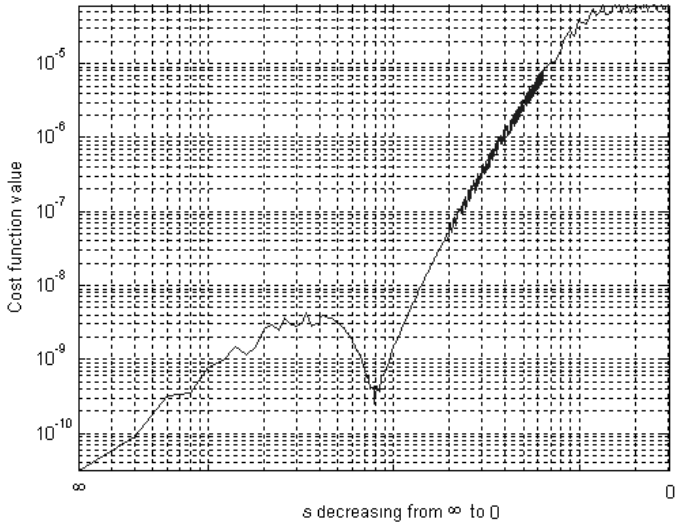


(b)

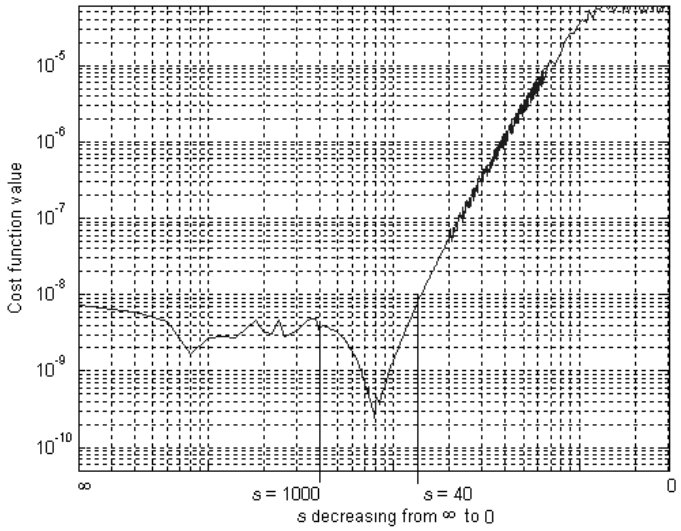
Figure 4.9: (a) Clustering cost function over the set $\bar{E}$. $\bar{E}$ denotes the complement set of E, and (b) Clustering cost function over the complete set U of intermediate hash pairs. $U = E \cup \bar{E}$. s is the randomization parameter.

qualities of the hash.

It is of interest to observe the values of the cost function in Fig. 4.10 (b) for $40 < s <$



(a)



(b)

Figure 4.10: (a) Clustering cost function over the set $\bar{E}$ with the vertical axis on a log scale to show more detail of Fig. 4.9 (a), and (b) Clustering cost function over the complete set U with the vertical axis on a log scale to show more detail of Fig. 4.9 (b).

1000. This region is zoomed into and plotted in Fig. 4.11. It can be observed from Fig. 4.11 that the total cost is of the order of the cost incurred by the deterministic algorithm.

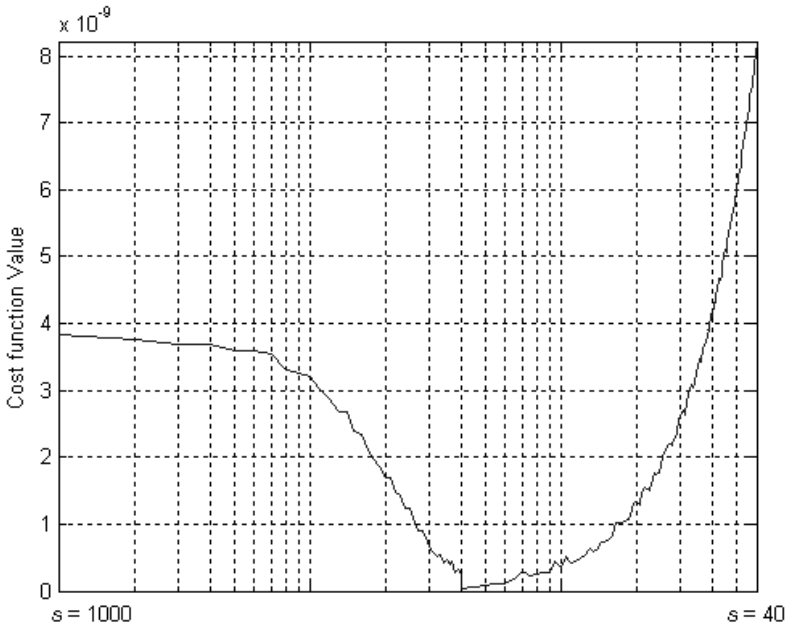


Figure 4.11: Clustering cost function over the set U of intermediate hash pairs in the region $40 < s < 1000$

Further, from Fig. 4.8, the cost over the set E for $s < 1000$ decreases to the extent that the adversary cannot gain anything by generating input pairs on this set. By choosing a value of s in this range, we can largely retain the perceptual qualities of the hash and also reduce the vulnerability of the hash algorithm to malicious inputs generated by the adversary.

4.6.3.3 *Distribution of Final Hash Values*

Finally, I evaluate our success in meeting the third desired property of the hash, i.e. the closeness to uniform distribution. I employ the widely used Kullback Leibler (KL)

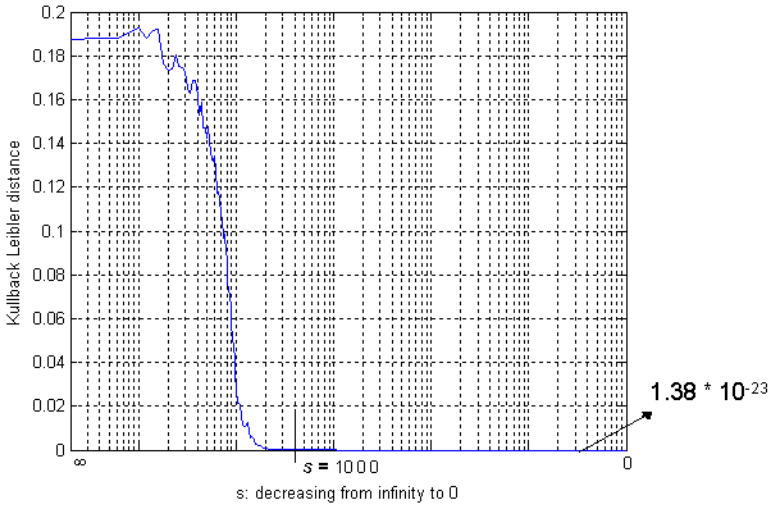


Figure 4.12: Kullback-Leibler distance of the hash distribution measured with the uniform distribution as the reference. Here s is the randomization parameter.

distance [35] given by

$$D(h||u) = -\sum_{x \in C} h(x) \log \frac{h(x)}{u(x)} \tag{4.20}$$

where $C = \{x : h(x) > 0\}$ represents the support set of $h(x)$. Here $h(x)$ denotes the distribution of hash values generated by our algorithm and $u(x)$ denotes the uniform distribution over the set C. The set C was obtained by generating the hash values for a given image used in our experiments over the key space (of K_1).

Fig. 4.12 shows the plot of the KL measure against values of s decreasing from ∞ to 0. Even as $s \to \infty$ this value is pretty low (≈ 0.2) and for $s < 1000$, i.e. the desired range for secure hashing, a near uniform distribution is achieved. Very similar results were observed for all of the 100 images in our experiments.

4.7 Conclusion

This chapter presents *greedy heuristic* based clustering algorithms for compression of intermediate image features. A novel cost function consisting of two additive exponential terms was developed. Such a cost better addresses the goals of perceptual hashing as opposed to traditional average distance type distortion measures.

Hardness results were derived, and the underlying clustering problem was shown to be NP-complete. The proposed solution to the clustering problem then proceeds by assigning "more likely" and close feature vectors to the same cell. A basic clustering was developed first that makes clusters without incurring any cost. For the remaining unclustered vectors, two approaches were presented that facilitate robustness vs. fragility trade-offs. The proposed clustering outperforms known compression techniques of traditional VQ and error correction decoding, for perceptual hash compression. The heuristic in the deterministic clustering algorithms was modified to develop a randomized clustering algorithm. The proposed randomization scheme was shown to significantly enhance security while largely retaining the robustness of the hash.

The proposed algorithms have two mathematically attractive properties: 1.) the number of clusters (or equivalently the length of the hash) is automatically determined, and 2.) the clustering can be applied to vectors in any metric space, i.e. no assumptions on the topology of the space are made. I believe these two properties will make the proposed algorithms valuable in hashing applications for other media, and more generally in data compression and/or dimensionality reduction.

Chapter 5

Image Authentication Under Geometric Attacks

5.1 Introduction

This chapter exploits the invariance properties of the feature extractor developed in Chapter 3 to develop an image authentication scheme that survives geometric attacks. Note that the image hashing algorithms presented in Chapters 3 and 4, and others reported in the existing literature, would fail to authenticate content under severe geometric manipulations such as large rotation and translation. I develop a generalized *Hausdorff* distance measure to compare features from two images. A search strategy is further employed to match features under a well-defined model of the geometric distortion. The use of the novel *Hausdorff* distance is crucial to the robustness of the scheme, and accounts for feature detector failure or occlusion, which previously proposed methods do not address.

Section 5.2 brings out the limitations of current approaches for geometric authentication based on image watermarking. A digital signature or feature based scheme for image authentication under geometric attacks is then proposed in Section 5.3. Within the scheme, I model the geometric distortion via an affine transformation, which in turn is estimated using object matching algorithms [55]. In Section 5.3.1, I propose a generalized robust *Hausdorff* distance for comparing image features. The proposed distance encompasses several other known *Hausdorff* measures as special cases. Section 5.4 shows

experimental results, that verify the capability of the proposed scheme to withstand both global and local geometric distortions, as long as they are perceptually insignificant. Section 5.5 concludes the chapter by summarizing the contributions.

5.2 Limitations of Geometrically Invariant Watermarking

Recall from Section 1.1, watermarking is the process of embedding information in an image (or media), which can later be retrieved for authentication purposes. In robust authentication scenarios, the watermark is required to be retained in the image under a set of allowable distortions on the image. These distortions as described before, are characterized as being "perceptually insignificant".

An important subset of allowable distortions on an image is geometric manipulations. These can further be decomposed into two classes: global transformations such as scaling, rotations and translations, and local transformations such as random bending and shearing (e.g. the StirMark attack). One major drawback of classical watermarking [10, 11, 14, 15] as well as digital signature schemes [4, 7, 19, 20] is the lack of robustness to geometric distortions. For this reason, significant attention has been devoted in recent years towards developing geometrically invariant watermarking schemes. This includes periodic insertion of the mark [56, 57, 58], template insertion [59], mark embedding in geometrically invariant domains [60, 61], and content based watermarking schemes that extract image feature points [62, 63, 64].

Watermarking schemes based on periodic insertion [56, 57, 58] introduce redundancy in the mark embedding process, e.g. doing a periodic tiling of the image and embedding the same (but randomly generated) watermark in each tile. This redundancy can be used to localize the position of the mark and improve the watermark detection phase.

Template based schemes [59] embed a well defined geometric pattern in an image, which can be easily detected after the image is rotated, scaled, and translated. It is also possible to first transform the image to a geometrically invariant domain, e.g. the Fourier-Mellin transform, and then embed the watermark in this domain [60, 61]. A common shortcoming of the methods in [56]-[61] is that they are not robust to local geometric transformations. Further, schemes based on embedding in geometrically invariant domains are very vulnerable to common signal processing operations, such as compression and enhancement.

While the methods in [62] - [64] exhibit robustness to both global and local distortions, they implicitly make very strong assumptions of the feature point detector. In other words, feature points from the watermarked original image and a candidate image are required to exactly match (under a model of the geometric distortion) for the mark to be successfully detected. In practice, under arbitrary geometric distortions, such an assumption often proves too optimistic. Also, feature detection is seldom perfect. Feature points that are detected in the original copy may not be present in the version that has undergone a (perceptually insignificant) geometric transformation.

The limitations of aforementioned approaches forms the motivation for the authentication scheme I develop in this chapter. Further, my proposed scheme is signature (and not watermark) based. To the best of my knowledge there are no known digital signature based schemes for robust authentication under geometric attacks.

5.3 Proposed Scheme for Image Authentication

The proposed image authentication scheme is illustrated in Fig. 5.3. The set of feature points $\mathbf{N}$ extracted from a candidate image (using the feature extractor described in

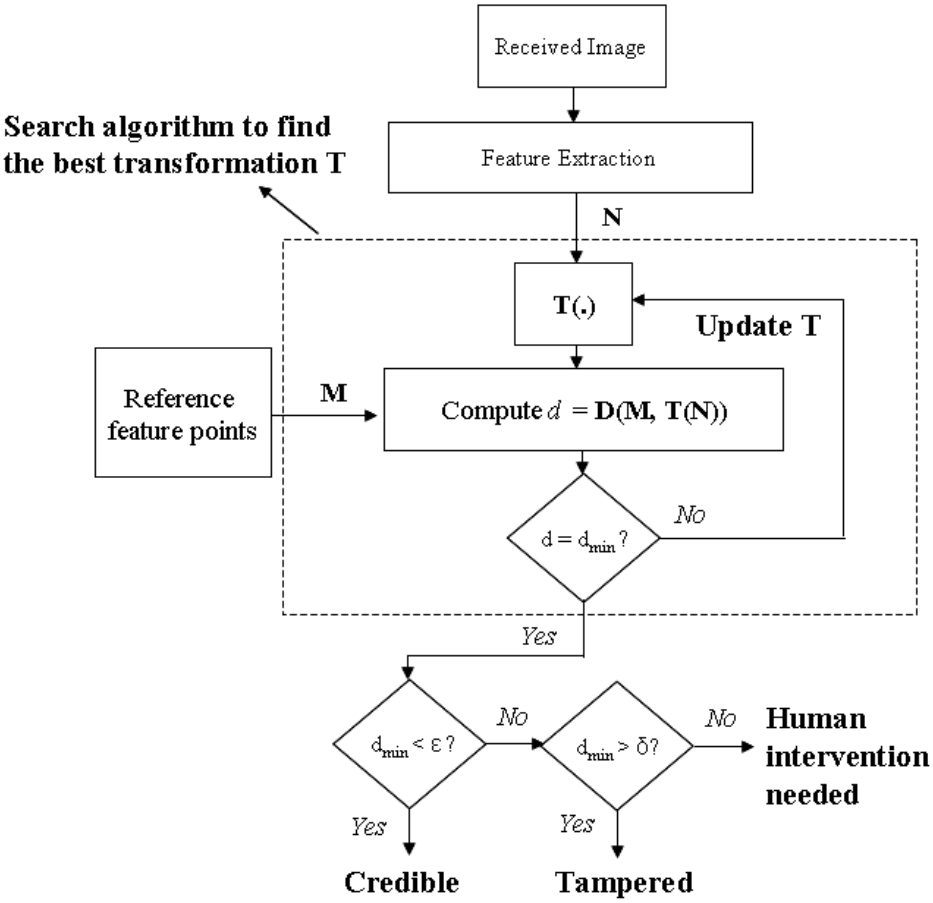


Figure 5.1: Flow chart of the image authentication scheme

Chapter 3) is transformed by a suitable model $\mathbf{T}$, of the geometric distortion. The transformed set of points is then compared against the (pre-computed) set of feature points $\mathbf{M}$ from a reference image using a robust distance measure $\mathbf{D}(\cdot, \cdot)$. The transformation $\mathbf{T}$ is updated using an intelligent search strategy until a local minima of the distance function is reached. Based on the value of this minimum distance, we declare the image to be credible or tampered. Next, I detail the particular choice of various components in the proposed authentication framework.

5.3.1 Distortion Modeling

I model the geometric distortion on the feature points via an affine transformation $\mathbf{T}$ such that

$$\mathbf{T}(\mathbf{x}) = \mathbf{y} = \mathbf{R}\mathbf{x} + \mathbf{t} \tag{5.1}$$

where $\mathbf{x} = (x_1, x_2)$, $\mathbf{y} = (y_1, y_2)$, $\mathbf{R}$ is a 2×2 matrix and $\mathbf{t}$ denotes a 2×1 vector. Using an affine transform permits an exact modeling of distortions such as rotation, scaling, translation, and shearing effects. Also, under a robust distance measure several other geometric distortions are well approximated via the affine transform.

5.3.2 Robust Distance Measure on Image Features

5.3.2.1 Hausdorff Distance

Given two finite point sets $\mathbf{M} = \{m_1, ..., m_p\}$ and $\mathbf{N} = \{n_1, ..., n_q\}$, the Hausdorff distance is defined as

$$H(\mathbf{M}, \mathbf{N}) = \max(h(\mathbf{M}, \mathbf{N}), h(\mathbf{N}, \mathbf{M})) \tag{5.2}$$

where

$$h(\mathbf{M}, \mathbf{N}) = \max_{m \in \mathbf{M}} \min_{n \in \mathbf{N}} \| m - n \| \tag{5.3}$$

and $\| \cdot \|$ is the underlying norm on the points of $\mathbf{M}$ and $\mathbf{N}$. The function $h(\mathbf{M}, \mathbf{N})$ is called the *directed* Hausdorff distance from $\mathbf{M}$ to $\mathbf{N}$. $h(\mathbf{M}, \mathbf{N})$ in effect ranks each point of $\mathbf{M}$ based on its distance to the nearest point of $\mathbf{N}$ and then uses the largest ranked such point as the distance. The Hausdorff distance $H(\mathbf{M}, \mathbf{N})$ is the maximum of $h(\mathbf{M}, \mathbf{N})$ and $h(\mathbf{N}, \mathbf{M})$. Thus it measures the degree of mismatch between any two shapes described by the sets $\mathbf{M}$ and $\mathbf{N}$. The choice of Hausdorff distance is based on

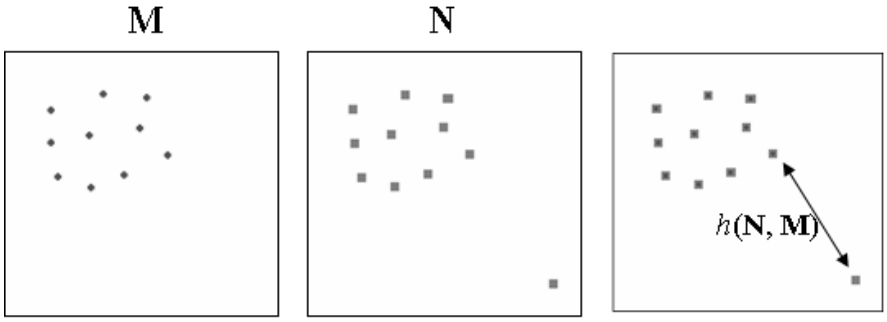


Figure 5.2: The directed Hausdorff distance is large just because of a single outlier

its relative insensitivity to perturbations in feature points, and robustness to occasional feature detector failure or occlusion [28].

The function $H(\mathbf{M}, \mathbf{N})$ can be trivially computed in time $O(pq)$ for two point sets of size p and q, respectively, and this can be improved to $O((p+q)\log(p+q))$ [65].

5.3.2.2 Modifying the Hausdorff Distance

The original Hausdorff distance in (5.2) is of limited utility in a robust authentication application because of its sensitivity to outliers. This is illustrated in Fig. 5.2. Therefore, I develop a generalized directed distance given by

$$h_g(\mathbf{M}, \mathbf{N}) = \sum_{i=1..|\mathbf{M}|} \alpha_i \min_{n \in \mathbf{N}} \parallel m_i - n \parallel, \;\; where \;\; \sum_i \alpha_i = 1 \tag{5.4}$$

The generalized Hausdorff distance $H_g(\mathbf{M}, \mathbf{N})$ is the maximum of $h_g(\mathbf{M}, \mathbf{N})$ and $h_g(\mathbf{N}, \mathbf{M})$.

Note this distance is generalized[1] because for the case that only one of the α_i's is equal to one (corresponding to $m_i \in \mathbf{M}$ that is farthest away from the closest point in $\mathbf{N}$) and

[1]The α_i's in (5.4) were empirically chosen.

rest are zero, (5.4) reduces to the directed Hausdorff distance in (5.3). Also, if each of the $\alpha_i = \frac{1}{|\mathbf{M}|}$ then this reduces to an average Hausdorff distance proposed by Jain *et al.* [29].

5.3.3 Authentication Procedure

After extracting the feature point set $\mathbf{N}$ from a received image, I find the affine transformation $\mathbf{T}^*$ that best approximates the geometric distortion. That is,

$$\mathbf{T}^* = \arg\min_{\mathbf{T}} H_g(\mathbf{M}, \mathbf{T}o\mathbf{N}) \tag{5.5}$$

The search strategy to find $\mathbf{T}^*$ is based on a divide and conquer rule and is detailed in [55].

Finally, $H_g(\mathbf{M}, \mathbf{T}^*o\mathbf{N})$ is compared against predefined thresholds ϵ and δ (where $0 < \epsilon < \delta$) to determine the credibility of image content. Note that to be able to fix ϵ and δ, we need a normalized distance (between zero and a constant). However, there is no natural way to normalize the distance in this case. For this reason, we normalize the data sets $\mathbf{M}$ and $\mathbf{N}$, i.e. recompute their coordinates such that the mean is zero and variance is set to unity. Then, I determine empirically $\epsilon = 0.15$ and $\delta = 0.2$.

5.4 Experimental Results

5.4.1 Robustness under perceptually insignificant geometric manipulations

Fig. 5.3 (a) shows the original *bridge* image with the extracted feature points overlayed. Three modified versions of this image under both global and local geometric distortions are shown in Figs. 5.3 (b) though (d). From a visual inspection of Figs. 5.3 (a)-(d) it can be ascertained that the features largely follow the geometric transformation on the image.

This validates the capability of the feature detector to successfully capture information about the geometric distortion on the image. For each of the distorted images, Fig. 5.3 also shows, an estimate of the geometric transformation as determined by the authentication procedure, and the final generalized Hausdorff distance between image features under this estimated transformation. Table 5.1 then tabulates this distance for three different images across several different (allowable) geometric distortions. The distorted images were generated using the Stirmark benchmark software [38]. The deviation is less than 0.15 except for very large cropping (more than 25%).

Visual as well as quantitative results for some more images, and attacks are reported in Appendix B.

Attack	*Lena*	*Bridge*	*Peppers*
JPEG, QF = 10	0.0857	0.1112	0.105
Scaling by 50%	0.0000	0.0020	0.1110
Rotation by 25^o	0.0030	0.1277	0.0078
Random Bending	0.0345	0.0244	0.0866
Print and Scan	0.0905	0.1244	0.1091
Cropping by 10%	0.0833	0.0025	0.1117
Cropping by 25%	0.2414	0.2207	0.2766

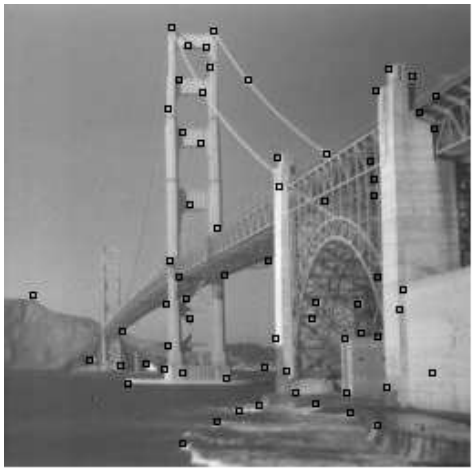
Table 5.1: Generalized Hausdorff distance ($H_g(\mathbf{M}, \mathbf{T}^* o \mathbf{N})$) between features of original and distorted images.

5.4.2 Security Via Randomization

I propose to enhance algorithm security by using a *randomized subspace projection* scheme. In particular, I first extract a large feature set $A = \{a_1, ..., a_Q\}$, and then (pseudo) randomly project it to a much smaller feature space spanned by the set $B = \{b_1, ..., b_P\}, where P < Q$, which is finally used in image comparisons. This is accomplished via using a secret key K to seed a cryptographically secure random number generator. This ensures that with high probability, the features that are extracted will not be the same unless the secret key is available. In practice, this significantly reduces the vulnerability to attacks by an adversary who attempts to generate malicious inputs (images) that defeat the authentication scheme.

5.5 Conclusion

This chapter introduces a framework for image authentication under geometric attacks using visually significant feature points. Geometric distortions are modeled via an affine transformation, and an intelligent search strategy is employed to find the best matching transformation. The key component of the scheme that enables robustness to geometric distortions is the use of a generalized *Hausdorff* distance to match geometric structures. Experimental results show that such a distance more accurately captures visual changes in image content, and also compensates for occasional failure of the feature detector. Finally, a randomized feature extraction scheme was presented to enhance security against maliciously generated geometric attacks.

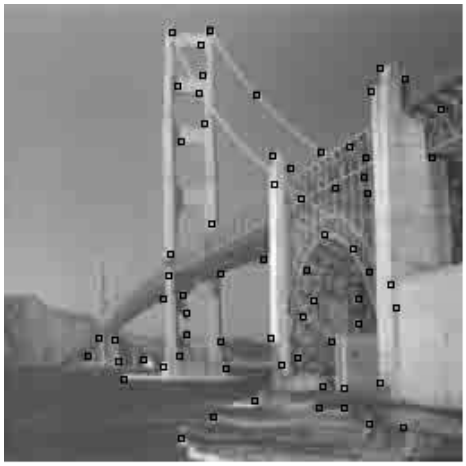


(a) Original image

$$\mathbf{R} = \begin{pmatrix} 0.9141 & 0.4258 \\ -0.4287 & 0.8984 \end{pmatrix} \quad \mathbf{t} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$H_g\left(\mathbf{M}, \mathbf{T}^* o \mathbf{N}\right) = 0.1277$$

(b) 25^0 rotation



$$\mathbf{R} = \begin{pmatrix} 0.9961 & 0 \\ 0 & 0.9961 \end{pmatrix} \quad \mathbf{t} = \begin{pmatrix} 0 \\ -1 \end{pmatrix}$$

$$H_g\left(\mathbf{M}, \mathbf{T}^* o \mathbf{N}\right) = 0.1112$$

(c) JPEG, QF = 10

$$\mathbf{R} = \begin{pmatrix} 1.0000 & 0.0703 \\ -0.0117 & 1.0000 \end{pmatrix} \quad \mathbf{t} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$H_g\left(\mathbf{M}, \mathbf{T}^* o \mathbf{N}\right) = 0.0961$$

(d) Random bending

Figure 5.3: Examples of geometrically distorted images. Feature points are overlayed.

Chapter 6

Conclusion

The problem of multimedia (e.g. image and audio) signal hashing has assumed a lot of importance over the last few years. Such hashes are required to be *perceptual* in nature; i.e. they should represent the content of the underlying media object. Applications such as database search impose the requirement of *robustness*; i.e. the hash should be invariant under perceptually insignificant (or incidental) modifications to the media. This facilitates searching images and audio clips in large media databases. An example scenario would be locating a query media file that is "perceptually" the same[1] as other media files in the database, but has a very different digital representation, e.g. a compressed or non-compressed image stored in a different format. Further, multimedia protection applications require the hashing algorithm to be secure. This is tantamount to requiring the hash to survive (intentional) attacks of guessing and forgery.

This dissertation develops new mathematical techniques for the design, analysis, and evaluation of perceptual image hash functions. Here, I summarize the contributions of this dissertation and suggest opportunities for future work.

[1]The meaning of perceptually the same depends on the underlying media. For example, for images, it means identical in visual appearance.

6.1 Summary of Contributions

Chapter 2 proposes a novel unifying framework for media hashing. The two-stage framework comprises of a media dependent feature extractor followed by media independent clustering of vectors in the feature space. I develop quantitative definitions for the desired properties of a perceptual image hash functions. The primary contribution of these definitions is to provide a conceptual benchmark for the evaluation of media hashing algorithms.

Chapter 3 develops a feature extraction scheme for images based on an explicit modeling of the human visual system (HVS) via end-stopped wavelets. Iterative feature extraction procedures are presented based on preserving significant image geometry. I show that the extracted features have favorable robustness properties for applications in image identification and hashing. In addition, the proposed technique outperforms existing approaches for the detection of content changing image manipulations, i.e. significantly enhances security. I quantify trade-offs between robustness, fragility, and security of the features via algorithm parameters.

Chapter 4 proposes clustering algorithms for compressing the features extracted in stage 1 of the two-step hash framework to a final hash value. I propose a novel cost function for feature vector compression and show that the decision version of the underlying clustering problem is NP-complete. I then present polynomial-time clustering algorithms based on a greedy heuristic. The proposed clustering is seen to vastly outperform traditional vector quantization (VQ) based compression and error correction decoding approaches for perceptual hash compression. Finally, I develop randomized clustering algorithms for the purposes of secure image hashing. Several researchers [1], [2] have identified randomization as essential for secure hashing. However, to the best of

my knowledge, this dissertation is the first to present a theoretical analysis of randomized media hashing algorithms and quantify the relationship of randomization with hash security.

Table 6.1 provides a comparison of the proposed hash algorithm against several existing image hashing paradigms reviewed earlier in Section 1.2. The perceptual image hash developed in this dissertation has desirable robustness as well as security properties. This is unlike previous methods, which typically compromise one at the cost of another. This advantage is a natural result of the joint cryptographic-signal processing approach that I adopt in the design of the proposed hash algorithm(s).

Chapter 5 addresses the problem of image authentication surviving geometric attacks. Previous solutions developed for the same were all watermark based. I develop a passive or signature based scheme based on the feature extraction scheme developed in Chapter 3. I model the geometric distortion on the image as an affine transformation, and employ object matching algorithms [55] to find the best matching transformation. To compare features from two images, I generalize the well known *Hausdorff* distance. The new distance significantly enhances the robustness of the scheme and accounts for feature detector failure, which previously proposed methods did not address.

6.2 Future Research

- *Pseudo-random signal representations:* It is useful to think of the binary string extracted via the randomized hash algorithm as a pseudo-random signal representation scheme for images; i.e. a different representation, each sufficient to characterize the image content, is obtained (with high probability) as the secret key is varied. Future work could explore alternate pseudo-random signal representations for im-

Image Hashing Algorithm	*Robustness*	*Security*	*Remarks*
Cryptographic hashes			
MD5, SHA-1	Poor	Good	No trade-off possible
Statistics Based			
Schneider *et al.* [4]	Poor	Poor	–
Kailasanathan *et al.* [5]	Poor	Poor	–
Venketasan *et al.* [1]	Fair	Fair	Trade-off hard to achieve
Coarse Representations			
Fridrich *et al.* [8]	Fair	Poor	Sensitive to small geometric changes
Mihcak *et al.* [2]	Good	Poor	Trade-off hard to achieve
Relation Based			
Lin *et al.* [7]	Fair	Poor	–
Lu *et al.* [9]	Fair	Fair	Sensitive to small geometric changes
Proposed Algorithm			
Monga *et al.* [24]	Good	Good	Trade-off facilitated

Table 6.1: Comparison of the image hashing algorithm developed in this dissertation against other methods in the literature. The proposed hash algorithm possesses desirable robustness as well as security and allows for a trade-off via hash algorithm parameters.

age identification and hashing. In particular, the goal of secure image hashing can be understood as developing the pseudo-random image representation that leaks the minimum amount of information about the image.

- *Rate-distortion analysis of hashing:* In this dissertation, I provide a heuristic solution to the finding the length of the hash required to sufficiently represent a media

set. The problem of determining the minimum hash length so as to meet a given distortion measure is similar to an information theoretic rate-distortion problem. In particular, for image hashing, given $0 < \theta < 1$, $\epsilon > 0$, and a visually meaningful notion of distance on images $\mathbf{D}(\cdot, \cdot)$; the problem is to find the minimum hash length such that

$$Pr(H(I) = H(I_{ident})) > 1 - \theta, \;\; if \;\; \mathbf{D}(I, I_{ident}) < \epsilon \tag{6.1}$$

where (I, I_{ident}) represent a pair of perceptually identical images in some class of images $\mathcal{I}$.

- *Alternate clustering algorithms with performance guarantees:* I developed heuristic clustering algorithms for compressing intermediate features of images. Although the proposed clustering vastly outperforms traditional compression approaches such as average distance VQ and error correction decoding, it does not come with any performance guarantees. This means, that the particular value of the objective/cost function achieved by the proposed clustering, is neither a local minima nor guaranteed to be within a constant of the global minimum. Designing clustering algorithms with performance guarantees is especially valuable from the viewpoint of hash scalability. Hierarchical clustering approaches may then be used to generate provably optimal[2] clusterings for $k + 1$ clusters, given the optimal clustering for k clusters is known.

- *Efficient implementation of image hashing algorithms:* In the proposed hash algorithm, there are several opportunities for speeding up the computation by employing parallel and/or distributed processing. For example, in the randomized inter-

[2]not necessarily a global optima

mediate hash algorithm, feature extraction from each random region can proceed independently. From a practical point of view, fast computation of the hash is very desirable. Further, it is not unreasonable to imagine the availability of generous computing resources, particularly for security applications, e.g. matching fingerprint images in secure databases. Efficient architectures for the implementation of media hashing algorithms is in general, a wide-open topic. Specific techniques for computational speed up will depend on the underlying media (e.g. images, audio etc.) and the specifics of the hash algorithm.

- *Hashing of other media:* Another possible future direction is in audio hashing, or more generally perceptual hashing of other media. Since the second step is (approximately) media independent[3], an appropriate feature detector may be applied in the first step to make the framework applicable to other media data sets.

- *Game-theoretic security analysis:* Finally, a very interesting direction for future research is to analyze the secure media hashing problem formally in a game theoretic setting, and draw comparison with watermarking games [66]. Note that with watermarking, the first move belongs to the embedding algorithm which is tied to a particular watermark insertion strategy that an attacker can subsequently try to remove. From a game theoretic point of view, hashing may infact be stronger than watermarking, since hashing algorithms can be adapted to attacks after these occur and without the need to modify and re-release deployed images.

[3]By approximately media independent, it is implied that the notion of distance on features extracted from the media and the probability measure induced in the feature space, are determined by the underlying media.

Appendix A - Proof of NP-completeness

In this section, we prove that a decision version of the clustering problem that asks if it is possible to have a k-clustering such that the cost function in (4.13) is below a certain constant is NP-complete. We achieve this by a reduction (details skipped for brevity) from the decision version of the k-way weighted graph-cut problem [67].

Proof. (Sketch) Let $G = (V, W(E))$ be a weighted graph where V is the set of vertices, E is the set of edges and $W(E)$ denote the weights on the edges. It is useful to think of V as the set of points to be clustered, and the weight $W(e_{ij})$ on the edge e_{ij} between v_i and v_j as the distance between the points v_i and v_j. The k-way weighted graph-cut problem asks if there is a subset $C \subseteq E$ of edges with $\sum_{e \in C} W(e) \leq K_0$, where K_0 is a constant, such that the graph $G' = (V, W(E \setminus C))$ has k pairwise disjoint subgraphs. We sketch a log-space reduction to the clustering problem in (4.13) for a fixed k. We construct a graph $\tilde{G} = (V, \tilde{W})$ from G as follows: Consider each possible vertex pair (v_i, v_j) with $i, j = 1, \ldots, n$. Denote $w_{ij} = W(e_{ij})$. If $w_{ij} < \epsilon$, $\tilde{w}_{ij} = K_1 c_1(i, j)$, where $c_1(i, j)$ is defined in (4.6) with $D(l_i, l_j) = w_{ij}$, and K_1 is a positive constant. If $w_{ij} > \delta$, then $\tilde{w}_{ij} = -K_2 c_2(i, j)$, where $c_2(i, j)$ is as defined in (4.7) with $D(l_i, l_j) = w_{ij}$, and K_2 is a positive constant. For $\epsilon \leq w_{ij} \leq \delta$, $\tilde{w}_{ij} = 0$. Consider the same k-way graph-cut problem on $\tilde{G}$. Let $\tilde{C}$ be a subset of the edges. For edges in $\tilde{C}$ with positive w_{ij}, the sum of the weights, say S_1, directly correspond to the sum of the $c_1(i, j)$ terms in (4.13). For edges in $\tilde{C}$ with negative weights, the sum of the weights, say S_2 is negative. Let $-N, N > 0$, denote the sum of all negative weights in $\tilde{W}$. Now, $N + S_2$ is the sum of the weights in $\tilde{W} \setminus \tilde{C}$, that exactly corresponds to the sum of the $c_2(i, j)$ terms in (4.13). Hence, $N + S_1 + S_2$ corresponds to the cost function in (4.13) up to an additive constant, when the $p(i)$ is uniform. Note that only constant number of indices of the vertices,

which need $O(\log n)$ space, must be maintained to complete the reduction. Hence, the k-way weighted graph-cut reduces to the clustering problem in log-space.

Appendix B - Authentication surviving geometric attacks: more examples

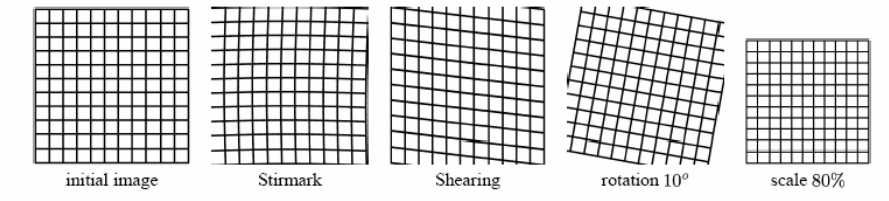
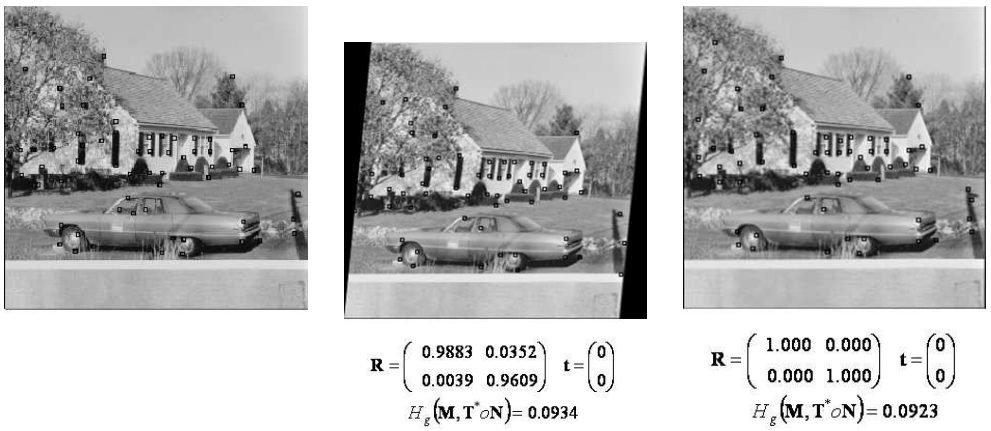
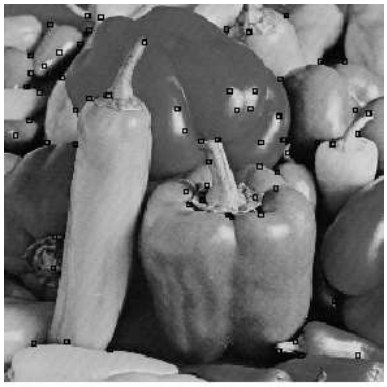


Figure 6.1: Representation of various geometric distortions applied to a grid.

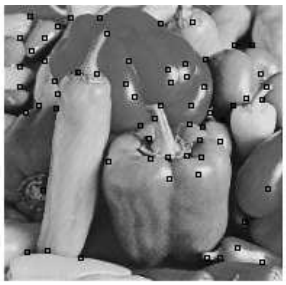


(a) Original *house* image (b) Random bending (c) Stirmark local geometric attack

Figure 6.2: Examples of geometrically distorted images. Feature points are overlayed.



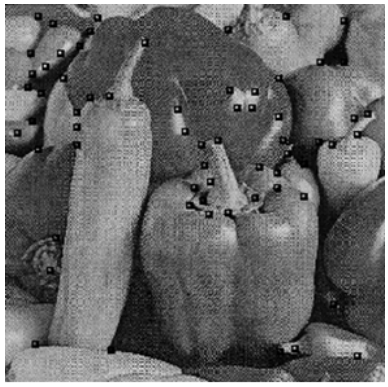
(a) Original *peppers* image



$$\mathbf{R} = \begin{pmatrix} 1.3359 & 0.0000 \\ 0.0000 & 1.3320 \end{pmatrix} \quad \mathbf{t} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$H_g\left(\mathbf{M}, \mathbf{T}^* o \mathbf{N}\right) = 0.1110$$

(b) Scaling by 75%



$$\mathbf{R} = \begin{pmatrix} 1.0000 & 0.0039 \\ -0.0039 & 1.0000 \end{pmatrix} \quad \mathbf{t} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$H_g\left(\mathbf{M}, \mathbf{T}^* o \mathbf{N}\right) = 0.1091$$

(c) Print-scan geometric distortion

Figure 6.3: Examples of geometrically distorted images. Feature points are overlayed.

Appendix C - Summary of notation

1. $\mathcal{I}$: Class of images of a particular size.
2. (I, I_{ident}): pair of perceptually identical images in $\mathcal{I}$.
3. (I, I_{diff}): pair of perceptually distinct images in $\mathcal{I}$.
4. $\mathcal{K}$: key space, K: a particular *secret* key in $\mathcal{K}$.
5. $\mathbf{h}(I)$: intermediate hash vector obtained from the image I at stage 1 of the hashing framework using the deterministic intermediate hash algorithm in Fig. 3.3.
6. $\mathbf{h}(I, K)$: intermediate hash vector obtained from the image I at stage 1 of the hashing framework using the randomized intermediate hash algorithm in Fig. 3.4.
7. $H(I, K)$: final hash value computed using the randomized two-stage hash algorithm. The intermediate hash extraction and/or the clustering stages could be randomized.
8. $\psi_M(x, y)$: basis function of the Morlet wavelet.
9. $\psi_M(x, y, \theta)$: basis function of the End-stopped wavelet.
10. $W_i(x, y, \theta)$: end-stopped wavelet transform coefficient of image I computed at scale i and orientation θ.
11. $D_H(\cdot, \cdot)$: normalized Hamming distance.
12. $D(\cdot, \cdot)$: distance metric applicable to image feature/intermediate hash vectors.

13. $P_{fP}(\epsilon)$: probability of false positive, i.e. intermediate hash vectors separated by less than ϵ for visually distinct images.

14. $P_{fN}(\delta)$: probability of false negative, i.e. intermediate hash vectors separated by more than δ $(0 < \epsilon < \delta)$ for visually identical images.

15. $E[\tilde{\mathbf{C}}_1]$: clustering cost incurred by violating (4.1).

16. $E[\tilde{\mathbf{C}}_2]$: clustering cost incurred by violating (4.2).

17. $Prec_\epsilon$: precision ratio of any scheme used for feature vector compression as given by (4.18).

18. Rec_δ: precision ratio of any scheme used for feature vector compression as given by (4.19).

19. $h(\mathbf{M}, \mathbf{N})$: directed Hausdorff distance between finite point sets $\mathbf{M}$ and $\mathbf{N}$ as given by (5.3).

20. $H(\mathbf{M}, \mathbf{N})$: Hausdorff distance between finite point sets $\mathbf{M}$ and $\mathbf{N}$ as given by (5.2).

21. $h_g(\mathbf{M}, \mathbf{N})$: generalized directed Hausdorff distance between finite point sets $\mathbf{M}$ and $\mathbf{N}$ as given by (5.4), corresponding generalized Hausdorff distance denoted by $H_g(\mathbf{M}, \mathbf{N})$.

Bibliography

[1] R. Venkatesan, S. M. Koon, M. H. Jakubowski, and P. Moulin, "Robust image hashing," *Proc. IEEE Conf. on Image Processing*, vol. 3, pp. 664–666, Sept. 2000.

[2] K. Mihcak and R. Venkatesan, "New iterative geometric techniques for robust image hashing," *Proc. ACM Workshop on Security and Privacy in Digital Rights Management*, pp. 13–21, Nov. 2001.

[3] A. Menezes, V. Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1998.

[4] M. Schneider and S. F. Chang, "A robust content based digital signature for image authentication," *Proc. IEEE Conf. on Image Processing*, vol. 3, pp. 227–230, Sept. 1996.

[5] C. Kailasanathan and R. S. Naini, "Image authentication surviving acceptable modifications using statistical measures and k-mean segmentation," *IEEE-EURASIP Work. Nonlinear Sig. and Image Processing*, vol. 1, June 2001.

[6] C. Y. Lin and S. F. Chang, "Generating robust digital signature for image/video authentication," *Proc. ACM Multimedia and Security Workshop*, Sept. 1998.

[7] C. Y. Lin and S. F. Chang, "A robust image authentication system distingushing JPEG compression from malicious manipulation," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 11, pp. 153–168, Feb. 2001.

[8] J. Fridrich and M. Goljan, "Robust hash functions for digital watermarking," *Proc. IEEE Int. Conf. on Information Technology: Coding and Computing*, pp. 178–183, Mar. 2000.

[9] C.-S. Lu and H.-Y. M. Liao, "Structural digital signature for image authentication," *IEEE Trans. on Multimedia*, vol. 5, pp. 161–173, June 2003.

[10] I. J. Cox, J. Kilian, F. T. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Trans. on Image Processing*, vol. 6, pp. 243–246, Dec. 1996.

[11] E. T. Lin and E. J. Delp, "A review of fragile image watermarks," *Proc. ACM Multimedia and Security Workshop*, vol. 1, pp. 25–29, Oct. 1999.

[12] M. M. Yeung and F. Mintzer, "An invisible watermarking scheme for image verification," *Proc. IEEE Conf. on Image Processing*, vol. 1, pp. 680–683, Oct. 1997.

[13] M. Wu and B. Liu, "Watermarking for image authentication," *Proc. IEEE Conf. on Image Processing*, vol. 2, pp. 437–441, Oct. 1998.

[14] R. B. Wolfgang and E. J. Delp, "Fragile watermarking using the VW2D watermark," *Proc. SPIE/IS&T Int. Conf. Security and Watermarking of Multimedia Contents*, pp. 204–213, Jan. 1999.

[15] L. Xie and G. R. Arce, "A class of authentication digital watermarks for secure multimedia communication," *IEEE Trans. on Image Processing*, vol. 10, pp. 1754–1764, Nov. 2001.

[16] G. L. Friedman, "The trustworthy digital camera: restoring credibility to the photographic image," *IEEE Trans. on Consumer Electronics*, vol. 39, pp. 905–910, Nov.

1993.

[17] M. K. Mihcak and R. Venkatesan, "Video watermarking using image hashing," *Microsoft Research Technical Report*, Jan. 2001.

[18] R. E. Blahut, *Theory and Practice of Error Control Codes.* Addison-Wesley Publishing Company, 1983.

[19] S. Bhatacherjee and M. Kutter, "Compression tolerant image authentication," *Proc. IEEE Conf. on Image Processing*, vol. 1, pp. 435–439, 1998.

[20] J. Dittman, A. Steinmetz, and R. Steinmetz, "Content based digital signature for motion picture authentication and content-fragile watermarking," *Proc. IEEE Int. Conf. on Multimedia Computing and Systems*, vol. 2, pp. 209–213, 1999.

[21] V. Monga and B. L. Evans, "Robust perceptual image hashing using feature points," *Proc. IEEE Conf. on Image Processing*, vol. 1, pp. 677–680, Oct. 2004.

[22] V. Monga and B. L. Evans, "Perceptual hashing via image feature points: Performance evaluation and trade-offs," *IEEE Trans. on Image Processing*, submitted, 2005.

[23] V. Monga, A. Banerjee, and B. L. Evans, "Clustering algorithms for perceptual image hashing," *Proc. IEEE Digital Sig. Processing Workshop*, pp. 283–287, Aug. 2004.

[24] V. Monga, A. Banerjee, and B. L. Evans, "A clustering based approach to perceptual image hashing," *IEEE Trans. on Signal Processing*, accepted with minor revisions.

[25] V. Monga and B. L. Evans, "Image authentication under geometric attacks via structure matching," *IEEE Int. Conf. Multimedia and Expo*, accepted, 2005.

[26] S. Bhatacherjee and P. Vandergheynst, "End-stopped wavelets for detection low-level features," *Proc. SPIE, Wavelet Applications in Signal and Image Processing VII*, pp. 732–741, Jan. 1999.

[27] M. Johnson and K. Ramachandran, "Dither-based secure image hashing using distributed coding," *Proc. IEEE Conf. on Image Processing*, vol. 3, pp. 14–17, Sept. 2003.

[28] W. J. Rucklidge, "Efficient computation of the minimum Hausdorff distance for visual recognition," *PhD Thesis, Cornell University*, 1995.

[29] M. P. Dubuisson and A. K. Jain, "A modified Hausdorff distance for object matching," *Proc. IEEE Int. Conf. on Pattern Recognition*, pp. 566–568, Sept. 1994.

[30] D. H. Hubel and T. N. Wiesel, "Receptive fields and functional architecture in two nonstriate visual areas of the cat," *J. Neurophysiology*, pp. 229–289, 1965.

[31] A. Dobbins, S. W. Zucker, and M. S. Cynader, "End-stopping and curvature," *Vision Research*, pp. 1371–1387, 1989.

[32] J.-P. Antoine and R. Murenzi, "Two-dimensional directional wavelets and the scale-angle representation," *Signal Processing*, pp. 259–281, 1996.

[33] S. Mallat, *A Wavelet Tour of Signal Processing.* Academic Press, 1999.

[34] R. Motwani and P. Raghavan, *Randomized Algorithms.* Cambridge University Press, 1996.

[35] T. M. Cover and J. A. Thomas, *Elements of Information Theory.* Wiley Interscience, 1998.

[36] D. E. Dudgeon and R. M. Mersereau, *Multidimensional Digital Signal Processing*. Prentice-Hall, 1984.

[37] G. Sharma, *Digital Color Imaging Handbook*. CRC Press, 2002.

[38] "Fair evaluation procedures for watermarking systems ." http://www.petitcolas.net/fabien/watermarking/stirmark, 2000.

[39] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Kluwer Academic, 1991.

[40] X. Wu, "Adaptive binary vector quantization using Hamming codes," *Proc. IEEE Conf. on Image Processing*, vol. 3, pp. 93–96, Oct. 1995.

[41] P. Franti and T. Kaukoranta, "Binary vector quantizer design using soft-centroids," *Signal Processing: Image Communication*, vol. 14, pp. 677–681, Sept. 1999.

[42] B. Julesz, "Visual pattern discrimination," *IEEE Trans. on Information Theory*, vol. 8, pp. 84–92, Feb. 1962.

[43] J. I. Yellott, "Images, statistics and textures: Implications of triple correlation uniqueness for texture statistics and the Julesz conjecture," *Journal of Optical Society of America*, vol. 10, pp. 777–793, Oct. 1993.

[44] S. Zhu, Y. N. Wu, and D. Mudford, "Filters, random fields and maximum entropy (frame) - towards the unified theory for texture modeling," *ACM Int. Journal of Computer Vision*, vol. 27, pp. 107–126, Mar. 1998.

[45] J. Portilla and E. P. Simoncelli, "A parametric texture model based on joint statistics of complex wavelet coefficients," *Kluwer Int. Journal of Computer Vision*, vol. 40, pp. 49–71, Jan. 2000.

[46] "The USC-SIPI image database." http://sipi.usc.edu/database/, 2004.

[47] P. Indyk and R. Motwani, "Approximate nearest neighbor: towards removing the curse of dimensionality," *Proc. ACM Symp. Comput. Geometry*, pp. 604–613, May 1998.

[48] P. Indyk, *High-dimensional Computational Geometry.* PhD Thesis, Stanford University, 2001.

[49] J. E. Goodman and J. O'Rourke, *Handbook of Discrete and Computational Geometry.* CRC Press, 1997.

[50] M. L. Fredman, J. Komlos, and E. Szemeredi, "Storing a sparse table with O(1) worst case access time," *Journal of the ACM*, vol. 31, pp. 538–544, June 1984.

[51] M. L. Friedman, J. L. Bentley, and R. A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Trans. on Mathematical Software*, pp. 209–226, Sept. 1977.

[52] A. Papoulis and S. U. Pillai, *Probability, Random Variables and Stochastic Processes.* McGraw Hill College Series, 2000.

[53] F. A. P. Petitcolas and R. J. Anderson, "Evaluation of copyright marking systems," *Proc. IEEE Int. Conf. on Multimedia Systems*, pp. 574–579, June 1999.

[54] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning.* Springer-Verlag, 2001.

[55] W. J. Rucklidge, "Locating objects using the Hausdorff distance," *IEEE Int. Conf. on Computer Vision*, pp. 457–464, 1995.

[56] M. Kutter, "Watermarking resistant to translation, rotation and scaling," *Proc. SPIE Multimedia Systems and Applications*, vol. 3528, pp. 423–431, Nov. 1998.

[57] T. Kalker, G. Depovere, J. Haitsma, and M. Maes, "A video watermarking system for broadcast monitoring," *Proc. SPIE Symp. on Electronic Imaging*, pp. 103–112, Jan. 1998.

[58] D. Delanay and B. Macq, "Generalized 2-D cyclic patterns for secret watermark generation," *Proc. IEEE Conf. on Image Processing*, pp. 77–80, Sept. 2000.

[59] S. Pereira and T. Pun, "Fast robust template matching for affine resistant watermarking," *Proc. Int. Workshop on Information Hiding*, vol. 1768, pp. 200–210, 1999.

[60] J. K. O. Ruanaidh and T. Pun, "Rotation, scale and translation invariant spread spectrum image watermarking," *Signal Processing: Image Comm.*, vol. 66, pp. 303–317, May 1998.

[61] C. Y. Lin, M. Wu, A. B. J, M. L. Miller, I. Cox, and Y. M. Lui, "Rotation, scale, and translation resilient public watermarking for images," *IEEE Trans. on Image Processing*, vol. 10, pp. 767–782, May 2001.

[62] Q. Sun, J. Wu, and R. Deng, "Recovering modified watermarked image with reference to original image," *Proc. SPIE Symp. on Electronic Imaging*, pp. 415–424, Jan. 1999.

[63] Z. Duric and N. F. Johnson, "Recovering watermarks from images," *Information and Software Engineering Technical Report*, Apr. 1999.

[64] P. Bas, J. M. Chassery, and B. Macq, "Geometrically invariant watermarking using feature points," *IEEE Trans. on Image Processing*, vol. 11, pp. 1014 – 1028, Sept.

2002.

[65] B. B. H. Alt and J. Blomer, "Measuring the resemblance of polygonal shapes," *Proc. ACM Symp. Comput. Geometry*, June 1991.

[66] P. Moulin and K. Mihcak, "The parallel-Gaussian watermarking game," *IEEE Trans. on Information Theory*, vol. 50, pp. 272–289, Feb. 2004.

[67] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman & Co., 1979.

Vita

Vishal Monga received his B.Tech degree in electrical engineering from the Indian Institute of Technology (IIT), Guwahati in May 2001 and his M.S.E.E. degree from The University of Texas, Austin in May 2003. During the summers of 2003 and 2004, he was a summer intern at Xerox Labs in Webster, NY, where he worked on non-separable color transformations and multidimensional interpolation. In summer 2005, he is a research intern at Microsoft Research in Redmond, WA. Mr. Monga received the IS&T Raymond Davis scholarship in 2004, a Texas Telecommunications Consortium (TxTec) Graduate Fellowship from The University of Texas for the year 2002-2003, and the President's Silver Medal in 2001 at IIT Guwahati. He is a member of IEEE, SPIE and IS&T.

Permanent address: J-220, LIC Colony
Paschim Vihar, Delhi, 110087
INDIA

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.