Steganography for Font-dependant Binary Documents

Zeng Jiancheng

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of
Master of Engineering

2004

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research and has not been submitted for a higher degree to any other University or Institution.

....

Date Zeng Jiancheng

To my parents and my brother.

To my loved Cao Ke.

Acknowledgments

First of all, I would like to express my sincere gratitude to my supervisor, Prof. Kot Chichung, Alex, for his precious time, patient guidance, invaluable experience, helpful instructions on this research, and persistent encouragement through the tough time of my study towards the Master of Engineering degree. His knowledge, wisdom, kindness and spirit have benefited me for these years, and will benefit me for my whole life.

I am very grateful to Dr. Liu Jun for guiding me, giving me valuable and detailed instructions on my research, and for his continuous encouragement and help in writing this thesis. I want to thank Dr. Chen Jianping and Yang Huijuan for helping me implement the system and review the thesis.

Taking this opportunity, I would like to thank my fellow colleagues and graduate students Lu Haiping, Cheng Jun and Cao Hong for their help in my research work and thesis writing. I also want to thank the technicians of Workstation Resource Lab, for their kind and sincere assistance.

Finally, I want to thank my loved parents for teaching me the importance of hard work and perseverance, and for their love and encouragement. I would like

to thank my brother Zeng Jian Guang and my dear girlfriend Cao Ke, for their love, care and support. Without them I could never accomplish this thesis. I would also like to thank all my friends to give me the strength to go on.

Thanks again to all people who have helped me in one way or another.

Table of Contents

Summary

In recent years, digital technology has been developing at a tremendous fast speed. Due to the advantages of digital format, the use of digital media (audio, video and images) has been growing rapidly. At the same time, copyright protection problem is becoming a greater concern. Steganography/digital watermarking is thus proposed for these security-related issues. It embeds ownership information into the media content to prevent illegal reproduction and redistribution. Recently, the area of document authentication has gained more and more research interest. In this thesis, we discuss the issues on binary image steganography/watermarking for a secured document authentication system.

Compared to gray-level image watermarking, binary document watermarking is much more challenging. The existing algorithms for binary document images either have low embedding capacity, or can cause high distortion in human visual perception. Moreover, most of them are non-reversible algorithms, which limits the application scope of the watermarking system. Being aware of this situation, we propose a fragile watermarking algorithm for binary document images in spatial domain. The proposed algorithm tries to achieve a reasonably high embedding capacity while maintaining high quality for the watermarked image. Two

novel methods are proposed in this thesis, parametric-based method and pixel-based method. In the first method, we use a set of outline equations to describe an alphabet. Data are embedded in the document by adjusting some parameters of the outline equations. In the second method, some feature points of the letters are used for embedding data to simplify the algorithm. In both approaches, the original document images can be reconstructed during the watermark extraction process, which means that both of them are reversible watermarking methods. The experimental results indicate that the proposed algorithm has satisfactory performance in terms of visual quality of the watermarked images. The embedding capacity is reasonably high and sufficient for most document authentication systems.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

With the rapid development of Electronic World (*E-World*), the use of digital media (audio, video and images) has been growing at a tremendous fast speed. Obviously media in digital-format has many advantages: it can be perfectly copied with no loss of quality; it is faster to transmit; it reduces physical storage required; it is easier and more convenient to archive. All these features of digital media make it cost effective. While from a different point of view, these advantages have their shortcomings. It is also faster, easier and more convenient for pirates to make copies and to distribute with perfect quality, *while for analog media the copies are degraded versions of the originals* [1]. Apart from the copyright issue, it is also much easier for people to modify information or even to make forged documents in digital domain. With the increasing use of digital media, these security-related issues are becoming greater concerns.

Digital watermarking is thus proposed for these security issues. It embeds ownership information (*watermark*) into media content to prevent, or more precisely, to discourage illegal reproduction/redistribution. It can also be used to monitor data transmission in order to control royalty payments (*data monitoring*), or simply track the distribution to localize the data for marketing and sales purposes (*data tracking*). At the same time, it is strongly required not to deteriorate the media significantly. Subsequently the watermark data need to be detected/extracted to make an assertion about the media. For copyright protection applications, those algorithms are required to be robust enough to resist different attacks, like image cropping/rotation, etc. Unfortunately, to avoid all attacks for copyright application is an uphill task that may never be successful.

Authentication, also called tamper proofing, is another important application of watermarking. Recently, this area has gained more and more research interest. Instead of proving the ownership for copyright protection, authentication is to tell whether a medium is true or false and the creator of the medium can be identified. Sometimes the algorithms for this kind of applications are called *steganography/data hiding*.

There are some minor differences between the terms of digital watermarking and steganography/data hiding, as described below:

- **Digital Watermarking:** The broad definition of this term refers to this whole area, including steganography/data hiding. For the narrow definition, it only refers to those algorithms for copyright protection purpose (*robust watermarking*).

- **Steganography** & **Data Hiding:** These two terms usually have the same meaning, referring to those algorithms for authentication purpose which emphasis on the secure/hidden information, while robustness is usually not a concern. Therefore those algorithms are usually called *fragile watermarking*.

In this thesis, we will use these terms interchangeably. Our main focus is on steganography for binary document images (digital/electronic documents). There are a great number of algorithms/techniques proposed for image watermarking. However, most of them concern color/grayscale images, and only a few are proposed for binary images.

It is agreed by many researchers that binary image watermarking is a much more challenging task to be addressed. For binary images, the pixel values are either black ("0") or white ("1"). In order to embed information, it needs to flip pixels from white to black or vice versa. If this process is not done properly, the changes can be easily noticeable. This means that the distortion could be very high for the watermarked image. That's why techniques proposed for color/grayscale images usually fail for binary images.

Moreover, most of the existing binary watermarking methods are non-reversible. That is, the original image cannot be recovered on the watermark extraction side, which limits the application scope of the watermarking system. In addition to the considerations about the watermark extraction process, great attention should be given to watermarking reversibility.

In view of the fact that most electronic documents are created in plain text

without noise, we study features of noise-free binary document images (with plain text only), and propose a novel way to embed information like biometrics data with much higher capacity for authentication, while at the same time keeping the distortion low in human visual perception.

1.2 Objectives

The objectives of this work are to study the features of English letters (alphabets) and use these features to develop a watermarking algorithm for noise-free binary document images in spatial domain with minimum distortion and reasonably high embedding capacity. Moreover, the proposed watermarking algorithm need to be reversible. The concepts and techniques developed in this work can be also applied to other languages with some modifications. The performance results are to be compared with the existing methods, and a secured document authentication system is to be implemented for actual application based on the proposed watermarking algorithm.

1.3 Major Contributions of the Thesis

This thesis presents a detailed analysis of a number of novel watermarking algorithms for binary document images. In the thesis, the descriptions and specifications of TrueType font and METAFONT systems are studied and discussed on how to design/draw nice-looking characters. A process is implemented to obtain a set of outline equations from image in order to describe letterforms.

These equations are used to draw characters with different shapes by adjusting the parameters of the outline equations. The features of letters are then studied in detail. Based on these features, a new parametric watermarking algorithm is proposed for binary document images. Later a pixel-based watermarking approach is also proposed and implemented to simplify the algorithm. These two approaches are both reversible watermarking methods since the original images can be reconstructed on the watermark extraction side. The performance of these two approaches are compared with other watermarking methods. One main purpose of the proposed watermarking algorithm is to keep the distortion low in human visual perception.

1.4 Organization of the Thesis

In this chapter, a brief introduction to the thesis is provided including the motivation, objectives and the main contributions. The rest of this thesis is organized as follows:

- Chapter 2 presents a brief review on watermarking algorithms. A common watermarking system is introduced, and classifications of watermarking techniques are discussed. Some assumptions of our proposed algorithm are then described.

- Chapter 3 presents the proposed binary document watermarking algorithm and discusses each function component in detail. The representation and description of the TrueType font and METAFONT systems are studied and

a procedure is described to obtain outline equations of an alphabet step by step. To gain low distortion for watermarked images, the features of letters are studied. Several methods are used to represent outline equations with compared experimental results.

- Chapter 4 proposes a simpler pixel-based reversible watermarking algorithm based on the parametric approach mentioned in previous chapter. The experimental results are shown and some discussions are made.

- Chapter 5 compares the proposed algorithm with the existing methods. It also discusses the relative merits and limitations of the proposed algorithm.

- In Chapter 6, conclusions are drawn based on the proposed algorithm. Recommendations for future research work are also presented.

Chapter 2

Literature Review on Digital Watermarking

2.1 Overview

In this chapter, the security issues in the E-World are introduced, and principles of the digital watermarking and common watermarking schemes are presented. The existing watermarking techniques are then classified and discussed. Some digital watermarking algorithms on binary document images are presented and discussed. Finally some assumptions of the proposed reversible digital watermarking algorithm are described.

2.2 Security Issues in E-World

As stated in [2], *Human eyes are inherently digital, made from individual rod and cone cells.* According to this biological fact, most objects in the real world can be perfectly represented in digital format, like audio, video and images. Digital technology and E-World concepts have been developing rapidly these years. With the increasing use of digital document images, their security-related issues are becoming greater concerns. During the last decade (early papers on digital watermarking date back to 1994, and some go even as far back as the 1950s), digital watermarking, and more generally, data hiding, has been one of the most active research fields in the signal/image processing. Recently, data hiding for document authentication has gained more research interest. Although the field is very young, watermarking is becoming a more mature discipline [3].

For digital watermarking, the common argument is that it is not as secure as *cryptography.* In fact, the basic purpose of digital watermarking is not to achieve absolute security, but to increase the cost of breaking the system. Compared to cryptography, the key advantage of using digital watermarking is the difficulty to tell whether a medium is watermarked or not. This serves as deterrence for others to make illegal copies or forge documents.

In the E-World, security services consist of six components [4] as shown in Figure 2.1.

- **Confidentiality:** Requires that the information be read accessible only for authorized parties. It protects information from passive attacks, while the

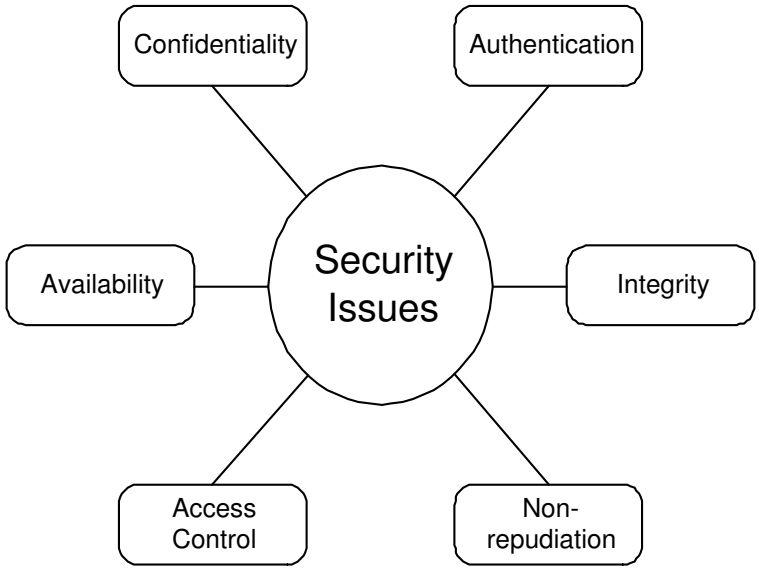


Figure 2.1: Security Issues in the E-World.

opposite is *information disclosure*. This issue can be fulfilled with satisfactory by cryptography. Cryptography is a mature technique, consisting of encryption and decryption processes. Triple Data Encryption Standard (3DES) and RSA are two widely used cryptography algorithms. The encryption key length can be from 56 bits to even 2048 bits. It will take thousands of years even using millions of powerful workstations for *Brute Force attack* to be successful.

- **Authentication:** Requires that the origin of a message be correctly identified, with an assurance for the true identity of the sender or the source. The opposite word can be *forged*. The identity cannot be drawn from the document itself. Extra information must be used to prove its identity. This issue can hopefully be solved by digital watermarking. The basic concept of digital watermarking is to insert/hide information (watermark) in the other

information (host). Later the watermark can be used to prove the identity of the host information.

- **Integrity:** Requires that information can only be modified by authorized parties. Modification includes writing, changing, changing status, deleting, creating, and the delaying or replaying of transmitted messages. This issue is solved by applying the concept of *data signature/fingerprint/digest*. Message Digest algorithm (MD5) and Secure Hash Algorithm (SHA) are two common used techniques. For a given message in any length, the output is in fixed length, which is called hashing/digest value. Like DNA sequences, it is extremely difficult to generate the same hash value mathematically for two different inputs.

- **Nonrepudiation:** Requires that neither the sender nor the receiver of a message be able to deny the transmission. The receiver can prove that the message was in fact sent by the alleged sender. Similarly, the sender can also prove that the message was in fact received by the alleged receiver. This problem can be seen as a two-way authentication issue. One solution is for the sender to ask for an acknowledgment from the receiver.

- **Access Control:** Requires that access to information resources be controlled by or for the target system.

- **Availability:** Requires that computer system assets be available to authorized parties when needed.

In order to build a secured system in the E-World, the above security issues (con-

fidentiality, authentication, integrity and nonrepudiation) need to be considered. Access control and availability have higher requirements on physical actions and are not as closely related to the research topic on data hiding. More works are trying to combine cryptography, hashing values and digital watermarking together to achieve higher system security.

2.3 Digital Watermarking Algorithms

Digital watermarking is firstly proposed for copyright protection purpose. It inserts/hides ownership information in the carrier/host media, while at the same time requiring not to deteriorate the carrier information significantly. Later the embedded data need to be detected or extracted to make an assertion about the host information. Most of the existing watermark detection schemes use some kind of correlation detectors to verify the presence of the watermark with some confidence measure.

2.3.1 Common Watermarking System

A digital watermarking system consists of two schemes: watermark *embedding* scheme at the sender side and watermark *extraction* scheme at the receiver end. The typical schemes are presented and explained below.

- **Common Watermark Embedding Scheme**

Figure 2.2 shows the common watermark embedding scheme [5]. The inputs consist of watermark, original image and an optional public/private key. The original image is the host (also called the cover data), which is to be protected or to be authenticated later. The watermark can be a number (e.g., Identity Card No.), text string (e.g., company name), an image (e.g., company logo), or a combination of the above. A secret or public key can be used to further ensure data security. Under a cryptographic key scheme, if they do not have the correct key, the watermark could not be recovered by unauthorized parties even if the scrambled data is available. In combination with a secret or a public key, the watermarking techniques are usually referred to as *secret* or *public watermarking* [6, 7].

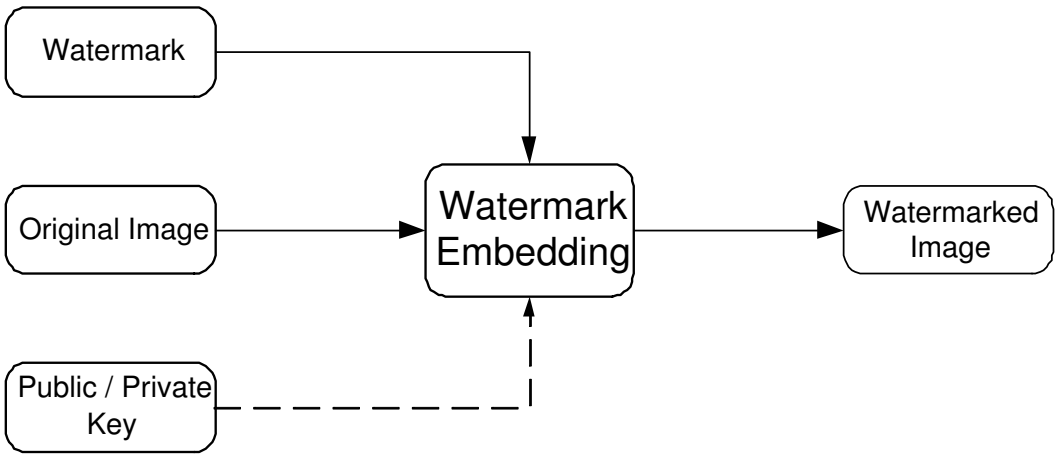


Figure 2.2: Common Digital Watermark Embedding Scheme.

The watermark is then embedded into the original image by using watermarking algorithms, and the watermarked image is sent to the receiver.

- **Common Watermark Extraction Scheme**

At the receiver end, the common watermark extraction process [5] is depicted in Figure 2.3. Inputs to the watermarking scheme are the suspect watermarked image, the optional secret or public key, and the optional original image or watermark, depending on *blind* or *non-blind* watermarking techniques used. If one technique does not require the original image for watermark extraction, it is usually called blind watermarking (blind detection is preferred). Otherwise, it is a non-blind watermarking technique. Non-blind watermarking algorithms are usually used for applications that require high robustness to resist attacks, while data authentication systems usually require blind watermarking. The output of the watermark recovery process, the retrieved data, can be the recovered watermark, or the recovered original image (for reversible watermarking techniques), or some confidence measure indicating how likely it is for the given watermark at the input to be present in the data under inspection.

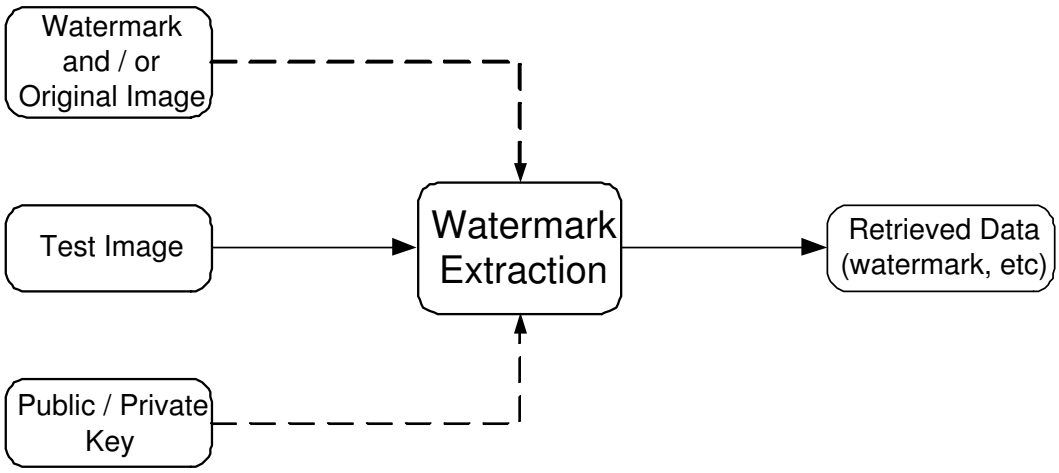


Figure 2.3: Common Digital Watermark Extraction Scheme.

2.3.2 Classifications of Watermarking Algorithms

A great deal of research efforts has been focused on image watermarking in recent years, and digital watermarking technology has evolved very quickly [8, 9, 10]. We do not give an exhaustive review of this whole broad area, and readers can refer to [11, 12, 13] for more extensive literature reviews. This section mainly presents several types of watermarking techniques on binary images.

The digital watermarking techniques can be classified based on many different categories. Learning from Kundur's classification [1], we classify the watermarking techniques as shown in Figure 2.4.

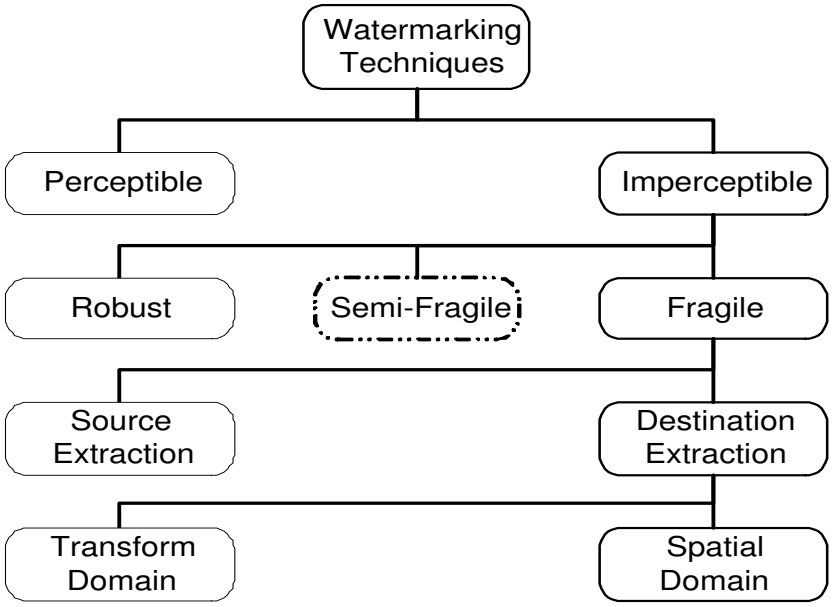


Figure 2.4: The Classification of Digital Watermarking Techniques.

- **Perceptible vs. Imperceptible**

  Perceptible (visible) digital watermarking creates noticeable changes (like ownership logo) in the host signal when added, but does not severely impede

the host signal from communicating the original message [10, 14, 15, 16]. Since the marking can be somewhat obtrusive, perceptible watermarking is not as popular as the imperceptible one. Most digital watermarking algorithms are imperceptible [8, 17]. Unless stated clearly, digital watermarking usually refers to imperceptible techniques. The class of imperceptible watermarking may be further divided into two subclasses: robust and fragile.

- **Robust vs. Fragile**

  The classification of robust and fragile watermarking is based on their different application purposes. Robust watermarking is trying to prove *an image belongs to somebody, not others*, while fragile watermarking is to identify *this image is true and original, not forged or modified*. Robust watermarking is mainly to protect intellectual property [18, 19, 20, 21]. For this purpose, it is designed to withstand arbitrarily malicious attacks. An attack is defined as any watermarked signal modification which can affect the reliability of the extracted watermark. There are many attacks since after the proposal of a new watermarking technique, some respective attack techniques will come up soon trying to break it. Image scaling, bending, cropping, and lossy compression are typical attacks [22].

  For robust digital watermarking, how robust is robust enough? Same as no absolute security for cryptography, there is no absolute robustness for digital watermarking. As mentioned before, robust watermarking is to *discourage* illegal copies, or to increase pirates' cost to do so. Benchmark tools, like StirMark and Checkmark, are published as generic tools for simple robustness testing of image watermarking algorithms [23, 24].

For fragile watermarking techniques [25, 26, 27, 28, 29, 30], it is to make sure whenever there is any unwanted transformation of the watermarked signal, the extracted watermark alters to indicate that tampering process exists. In other words, fragile watermarking is to detect any unauthorized modification for the purpose of image authentication, thus it is usually called *authentication* watermarking.

In recent years, some algorithms called semi-fragile watermarking [31, 32, 33, 34, 35, 36] are proposed, which can be seen as a new subclass between robust watermarking and fragile watermarking. The main difference between semi-fragile and fragile watermarking is that semi-fragile image authentication allows some acceptable manipulations, such as lossy compression. Such systems can tolerate incidental distortions but reject malicious modifications.

Fragile watermarking methods can be further divided into two subclasses: source extraction and destination extraction techniques.

- **Source Extraction vs. Destination Extraction**

  Source extraction techniques [16, 37] require that the original host signal be available to recover the watermark, which is also called non-blind watermarking as mentioned before. They are usually used for the applications like copyright protection that requires high robustness to signal distortions. Obviously, source extraction techniques are not suitable for image authentication, since for such applications usually there is no original image available at the receiver side.

Destination extraction schemes only require a key for watermark extraction without the original host image, therefore they are also called blind watermarking, or more precisely, blind detection. It can be further divided into two groups: spatial domain and transform domain techniques.

- **Spatial Domain vs. Transform Domain**

  Embedding domain is possibly the most common criteria used to classify watermarking algorithms. It includes spatial domain and transform (*frequency*) domain. The basic principles of most watermarking methods are to apply small, pseudo-random changes to the selected coefficients in these two domains. Generally speaking, transform domain algorithms (DCT, DWT, Hadamard transform, etc) [16, 38] are more robust and secure than the spatial domain approaches [17, 18, 19, 20, 25]. Transform domain methods are usually applied to multi-resolution (grayscale and color) images [39, 40], and could cause higher distortion for binary images [41]. Spatial domain algorithms usually cater for low distortion of watermarked images, which is especially important for binary images.

- **Reversible vs. Non-reversible**

  Besides the categories mentioned above, sometimes reversible and non-reversible are also used for watermarking classification. For the existing algorithms, most of them are non-reversible. As the watermark modifies pixel values, it also prevents strict integrity control. To solve the problem, the watermarking process has to be reversible. In addition to the considerations about the watermark extraction process, great attention should be

given to watermarking reversibility. Since the original image can be reconstructed/recovered/restored, reversible watermarking [42, 43, 44, 45, 46, 47, 48] is also called lossless/invertible watermarking, which is usually used for authentication system.

Reversibility also enlarges the scope of watermarking system. For instance, it is desired for image obtained at great cost, or in applications where the quality desired for the rendering is still unknown. Medical imaging, prepress industry, image archival systems, precious artworks, military images, and remotely sensed images are all candidates for lossless processing, and in particular for lossless watermarking [42].

2.3.3 Previous Work for Binary Document Images

As mentioned before, most proposed watermarking techniques concern grayscale or color images, while much less work has been reported for document images because of their binary nature. The difficulty lies in the fact that changing pixel values in a binary document could introduce irregularities that are very visually noticeable. Recently, there are more papers addressing this area [49, 50, 51, 52]. An overview on the recent development of document image watermarking is given in [53].

In this section, we will give a brief survey of the binary document watermarking algorithms in the literature. According to the embedding methods, they can be roughly divided into three groups: *relative space modification*, *character feature modification* and *block-based watermarking methods*. All these algorithms

are fragile and will fail for Optical Character Recognition (OCR) attack.

⋆ Relative Space Modification

Document images usually have many spaces, like spaces between paragraphs, between text lines, between words, and between characters. Space pattern is one distinct and unique characteristic of a text document, which can be used for data embedding. The techniques about line-shifting, word-shifting and character-shifting are discussed below:

- **Line/Word Shifting**

  In these methods, the text lines are shifted upper or lower (words are shifted to the left or right) by an indiscernible amount to embed data. These simple but valuable concepts are firstly proposed by Low et al. [54, 55, 56, 57, 58] for copyright protection. Brassil et al. stated in [18] that formatted documents with justified text typically use variable spacing between words to distribute white space in a visually pleasing fashion. Readers accept a wide variation in text setting within a line. It means that human eyes are not sensitive to the space modifications between words. Therefore, the spaces between words can be used to embed data. Correlation detection and centroid detection are applied to the vertical and horizontal profiles to extract the watermark. These algorithms are robust for scanning and photocopying processes which introduce noise.

  Huang and Yan [59] suggested that space patterning of text documents are useful for watermarking, and proposed an approach to modify the interword

spaces of different text lines slightly. After the modification, the average interword space changes form the sampling points of a sine wave, which is the watermark information. Both non-blind and blind watermarking algorithms are discussed.

Liu et al. [60] proposed a combined approach to mark a text document by line or word shifting suggested by Low et al. [54], and to detect the watermark in the frequency domain by using the algorithm suggested by Cox et al. [61]. The watermark is computed in the frequency domain after the spatial embedding through shifting.

- **Character Shifting**

  Character shifting is the extension of word-shift coding, using spaces between characters (including spaces between words) to embed data.

  Chotikakamthorn [62] proposed a method to embed data by adjusting widths of a few consecutive character spaces on the same text line according to a predefined rule. The advantage of this algorithm over word shifting is that it can be applied to written languages that do not have spaces with sufficiently large width for word boundaries, such as Chinese, Japanese and Thai languages.

  Yang et al. [17] proposed an approach that makes use of the integrated inter character and word spaces for watermark embedding. A new set of rules is defined to modify this technique: an overlapping component of size three is utilized, whereby the relationship of the left and right spaces of the character are employed for embedding to further increase the watermarking

capacity.

On the whole, line/word shifting methods achieve high visual quality. It is very difficult for human eyes to detect such kinds of space modifications. However, their embedding capacity is very limited. Compared to the line-shift coding, the word-shift coding method has higher embedding capacity but lower robustness. Compared to line/word shifting, character shifting methods further increase the watermarking capacity, but at the same time further decrease the robustness and visual quality of the watermarked images in human visual perception.

Since the word/character spacing in the original document is not uniform, detecting a word/character displacement requires knowledge of the original word/character spacing. Hence, the word/character position in the unmarked document must be known in order to extract the hidden information. In order to achieve blind watermark extraction, special rules must be used in the watermark embedding process.

⋆ Character Feature Modification

- **Character Height Modification**

  Brassil et al. [18, 19, 20, 63] proposed watermarking algorithms by choosing some text features to embed data. A small number of pixels can be added to the endlines of characters with ascenders or descenders. The upward, vertical endlines (the top of letters ‘b’, ‘d’, ‘h’ etc) are used in this method to embed data. However, human eyes are quite sensitive to these alterations,

thus the visual quality of the watermarked image is low.

- **Character Stroke Width Modification**

  Amamo and Misaki [64] proposed a feature calibration method for document watermarking, in which two sets of partitions are symmetrically arranged, and the difference between the average feature values of the two sets is extracted. The feature used in the paper is the average width of the horizontal strokes of characters. Two operations,"make fat" and "make thin", are defined to modify (increase and decrease the width of) the horizontal strokes in target partitions to embed information.

- **Character 8-connected Boundary Modification**

  Mei et al. [65] proposed a method for data hiding in binary text documents by embedding data in the 8-connected boundary of a character. They have identified a fixed set of pairs (28 pairs in the paper) of five-pixel long boundary patterns for embedding data. Two patterns in each pair are dual of each other and flipping the pixel value of one pattern at the center position would result in the other. This property allows easy detection of the embedded data without referring to the original document and without using any special enforcing techniques, which means blind watermark detection. Two patterns in each pair are used to embed one bit of data. The embedding capacity is much higher than the previous mentioned algorithms with reasonably good visual quality.

⋆ Block-based Watermarking Methods

- **Score-based Watermarking Method**

  Wu et al. [66] proposed a method to embed data in general binary images. In a block of size 3×3 pixels, they determine whether the center pixel can be flipped by examining this pixel and its 8 neighbors to establish a score indicating how noticeable it will be if the pixel is flipped, considering the change in smoothness and connectivity. The smoothness is measured by horizontal, vertical and diagonal transitions, while the connectivity is measured by the number of black and white clusters in the block. Pixels with lower scores are suitable candidates for flipping. Data is embedded by flipping the pixel in a block with the lowest score to enforce the total number of black pixels in the block to be odd (embed bit '1') or even (embed bit'0'). Shuffling, a random permutation of all pixels in the image, is done after the flippable pixels have been identified based on the scores to handle the uneven embedding capacity due to uneven distribution of flippable pixels. A look-up table containing the scores is built for use in flipping. The embedding capacity is quite high with reasonably good visual quality of the watermarked image. On the other hand, such rule-based approach becomes very complicated to consider a larger neighborhood.

- **DRDM Watermarking Method**

  Lu et al. [67, 68] proposed a novel objective distortion measure called Distance-Reciprocal Distortion Measure (DRDM), which uses a weight matrix to measure the distortion. They claimed that DRDM has much better

correlation with human visual perception than PSNR (peak signal-to-noise ratio) when applied to binary document images. Based on this DRDM method and learning from the score-based watermarking method, they proposed a data-hiding method in binary document images. The distortion due to flipping is calculated online and able to take the effect of a large area of neighbor pixels into accounts. Data is embedded by enforcing the odd-even features of non-uniform blocks and the 2-D shifting is employed to provide security against tampering. It has a simple form and it is straightforward to calculate for a window of any size. The watermarking capacity is very high, and the watermarked image has good visual quality.

2.4 The Proposed Algorithm and Assumptions

From the literatures reviewed, there are only a few papers for binary document watermarking. For text documents, human eyes are quite sensitive to usual modifications used for gray/color images. Therefore, the embedding capacities are usually low in order to keep low distortion of the watermarked images. Moreover, most of the existing document watermarking methods are non-reversible. The scope of watermarking system is limited if the original document cannot be recovered on the watermark extraction side. Being aware of this situation, we propose a fragile watermarking algorithm for binary documents with higher capacity, and both the parametric approach and pixel-based approach are invertible. Several assumptions of the proposed algorithm are discussed in the following part:

- **Applicable images**

  Only binary document images mainly with pure plain text are considered in this thesis. Binary images like halftone images, binary comic images and binary maps are to be segmented out for the plain text before data embedding.

- **Noise**

  Many watermarking algorithms and processes introduce noise, e.g., the Analog to Digital Conversion and Digital to Analog Conversion (ADC & DAC). In this thesis, we only consider the whole scheme totally in the digital domain with TrueType font being used. Although there may be distortion in the conversion process for word document to bitmap document, random noise is not an issue here.

- **Robustness**

  As mentioned before, our proposed schemes are fragile watermarking methods for document authentication purpose. Therefore, robustness is not our concern, and no robustness test will be performed.

- **The watermark**

  In the proposed watermarking algorithm, a randomly generated bit stream of '0's and '1's is used as the watermark for the experiments. Similar results will be obtained when we use meaningful watermarks, such as names, numbers and/or company logos.

- **Security concerns for application**

  Instead of implementing a complete secured system, we mainly focus on the

embedding and the extraction of the hiding data. Some security concerns are mentioned, but no detailed treatment is provided.

- **Simulation Environment**

  In our experiments, the simulation results are obtained from the computer with 2.4GHz CPU (Pentium 4) and 512MB memory. The computer is running Windows XP operation system, and MATLAB 6.5 is used for the experimental simulation.

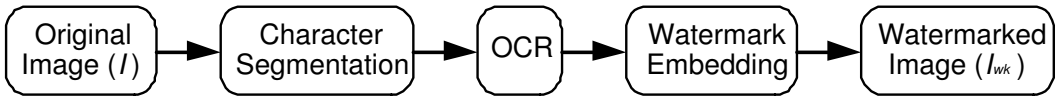
Chapter 3

Character Based Data-hiding Algorithm

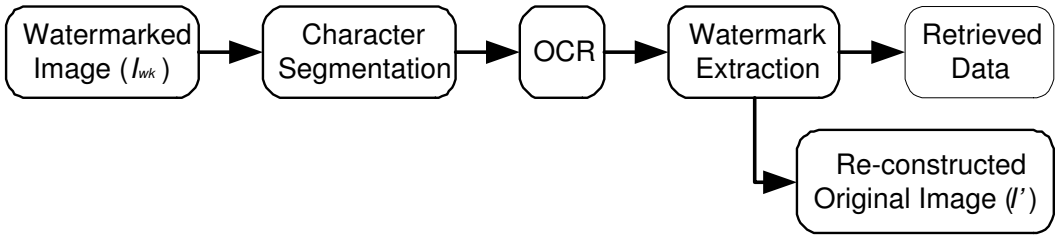
This chapter presents a character based data hiding scheme for binary document images. Each part of the data-hiding process is studied and discussed. Some experimental results are shown as well. The fundamentals of TrueType font and METAFONT systems are studied on how to design/draw nice-looking characters. Based on these fundamentals, a set of outline equations is derived to describe a given letter image. The parameters in these outline equations are studied. By adjusting these parameters, new letterforms with different sizes and different shapes are produced and compared. Data can be embedded into the document image by modifying the letter shapes/features to achieve high visual quality in human visual perception. During the watermark detection process, the original image can be reconstructed by using the reference letters.

3.1 Proposed Binary Document Watermarking Algorithm

The block diagram of the proposed watermark embedding scheme is shown in Figure 3.1(a).



(a) Proposed Watermark Embedding Scheme



(b) Proposed Watermark Extraction Scheme

Figure 3.1: Proposed Watermark Embedding/Extraction Schemes.

First of all, given a text document with standard fonts, a convertor is needed to convert the document to binary image. The basic idea of the proposed algorithm is to embed information by modifying the letter shapes/features. In order to do this, we need to know what characters (alphabets for English document) they are, and their exact positions in the document image. Therefore, *image segmentation* and *optical character recognition* (OCR) are applied to the original image as the preprocessing steps. The document image is segmented to separated letters using existing segmentation method, and these separated letters are then recognized

by OCR technique. The position of each character and its respective ASCII code are recorded to a profile for data embedding process.

For watermark embedding process, the first appeared letters in the document are set as the reference. For example, the first recognized 'A' is the reference for all the following 'A's, the first 'B' is the reference for the rest 'B's, and so on. Reference letters cannot be modified. They will be used to reconstruct the original image during the watermark extraction process. Therefore, our proposal is a reversible watermarking method.

For non-reference letters, the features are modified to embed data. For example, assume each alphabet 'A' can hide 8-bit data. To embed data "10000001" in 'A', only the first and eighth features are modified while the others remain unchanged. The details of data embedding techniques will be discussed later.

Once the whole watermark embedding process is finished, the watermarked document image is sent to the receiver. The extraction is just the reverse process of the embedding scheme, as shown in Figure 3.1(b).

To retrieve the embedded data, we proceed as follows:

- Step 1: Segment the image to get separated letters.
- Step 2: Recognize the separated letters by OCR, and record the positions and ASCII codes of the letters to a profile.
- Step 3: Set the first appeared letters as the reference.
- Step 4: Extract the embedded data by checking the features from the non-

reference letters. For example, we check the eight features of a recognized 'A'. If the first and eighth features are modified while the others remain unchanged, then the detected data is "10000001". In this way, the embedded data is retrieved.

- Step 5: Replace the non-reference letters with their respective reference. For example, replace all 'A's with the reference 'A', replace all 'B's with the reference 'B', and so on. In this way, the original image is reconstructed. That is to say, our proposed watermarking algorithm is *reversible*.

The block functions of the binary document watermarking schemes are discussed in detail below with the experimental results.

3.1.1 Sample Image

Without loss of the generality, we only study the alphabets in UPPERCASE. Since there is no benchmarking image available, we create one for our experiments. In Microsoft Word XP, we choose one of the most popular setting for documents: font type of *Times New Roman* with font size of 12 point using *JUSTIFY* text alignment. This word document is then printed to a pdf file using AdobePS Acrobat Distiller. Using Adobe Acrobat 5.0.5, it is converted to a grayscale image (tiff format) at the resolution of 300 (dpi) dots per inch. Using Microsoft Paint, this tiff image is binarized and saved as a bitmap file as shown in Figure 3.2.

After removing some white margins, the whole image has the size of 1820 $\times$ 2650 pixels. To save space in presentation, a small portion of the image with size

A WATERMARK IS A SIGNAL ADDED TO DIGITAL DATA (AUDIO, VIDEO AND STILL IMAGES) THAT CAN BE DETECTED OR EXTRACTED LATER TO MAKE AN ASSERTION ABOUT THE DATA. IN THIS PAPER, WE WILL BE CONCERNED ONLY WITH STILL IMAGES. EMBEDDING A WATERMARK IN AN IMAGE MEANS INSERTING INFORMATION IN THE IMAGE, SUCH THAT THE IMAGE QUALITY DOES NOT DETERIORATE SIGNIFICANTLY.

A WATERMARK CAN BE VISIBLE OR INVISIBLE. A VISIBLE WATERMARK TYPICALLY CONTAINS A VISUAL MESSAGE OR A COMPANY LOGO THAT INDICATES THE OWNERSHIP OF THE IMAGE. AN INVISIBLE WATERMARKED IMAGE IS VISUALLY VERY SIMILAR TO THE ORIGINAL IMAGE. THE EXISTENCE OF SUCH A WATERMARK CAN BE DETERMINED ONLY THROUGH A WATERMARK EXTRACTION OR DETECTION ALGORITHM. INVISIBLE WATERMARK TECHNIQUES CAN BE CLASSIFIED AS EITHER “ROBUST” OR “FRAGILE”.

ROBUST WATERMARKS ARE DESIGNED TO BE HARD TO REMOVE AND TO RESIST COMMON IMAGE-MANIPULATION PROCEDURES. THEY ARE USEFUL FOR COPYRIGHT AND OWNERSHIP ASSERTION PURPOSES.

FRAGILE WATERMARKS (OR AUTHENTICATION WATERMARKS) ARE EASILY CORRUPTED BY ANY IMAGE-PROCESSING PROCEDURE. WATERMARKS FOR CHECKING IMAGE INTEGRITY CAN BE FRAGILE BECAUSE IF THE WATERMARK IS REMOVED, THE WATERMARK DETECTION ALGORITHM WILL REPORT THE CORRUPTION OF THE IMAGE.

HALFTONING IS A PROCESS TO CONVERT A GRAYSCALE IMAGE G INTO CORRESPONDING BINARY IMAGE B, SUCH THAT B LOOKS LIKE G WHEN VIEWED FROM A DISTANCE. CLASSIC HALFTONE METHODS INCLUDE ORDERED DITHERING, ERROR DIFFUSION [ULICHNEY, 1987] AND DOT DIFFUSION [KNUTH, 1987]. HALFTONE IMAGES APPEAR ROUTINELY IN BOOKS, MAGAZINES, NEWSPAPERS, PRINTER OUTPUTS AND FAX DOCUMENTS. HALFTONE IMAGES CAN BE DISPERSED-DOT OR CLUSTERED-DOT. USUALLY, DISPERSED-DOT HAS A BETTER VISUAL QUALITY, BUT SOME DEVICES CANNOT REPRODUCE TOO-FINELY-DISPERSED DOTS (LIKE LASER PRINTER) AND THEY MUST USE CLUSTERED-DOT HALFTONING.

IN THE LITERATURE, THERE ARE MANY AUTHENTICATION WATERMARKING TECHNIQUES FOR CONTINUOUS-TONE IMAGES (GRAYSCALE AND COLOR) [ZHAO, 1995; YEUNG, 1997; WONG, 1997; WONG, 2002]. HOWEVER, IT SEEMS TO BE ALMOST IMPOSSIBLE TO DESIGN A REALLY SECURE AUTHENTICATION WATERMARKING WITHOUT MAKING USE OF THE SOLID CRYPTOGRAPHY THEORY AND TECHNIQUES. INDEED, THOSE WATERMARKING TECHNIQUES THAT WERE NOT FOUNDED IN CRYPTOGRAPHY THEORY [ZHAO, 1995; YEUNG, 1997] OR THOSE THAT APPLIED INCORRECTLY CRYPTOGRAPHIC TECHNIQUES [WONG, 1997;

Figure 3.2: Sample Image of Size 1820 × 2650 pixels.

of 512×512 pixels is used, as shown in Figure 3.3.

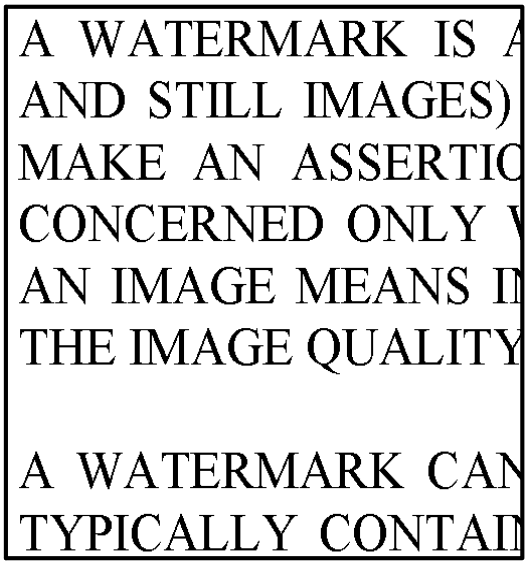


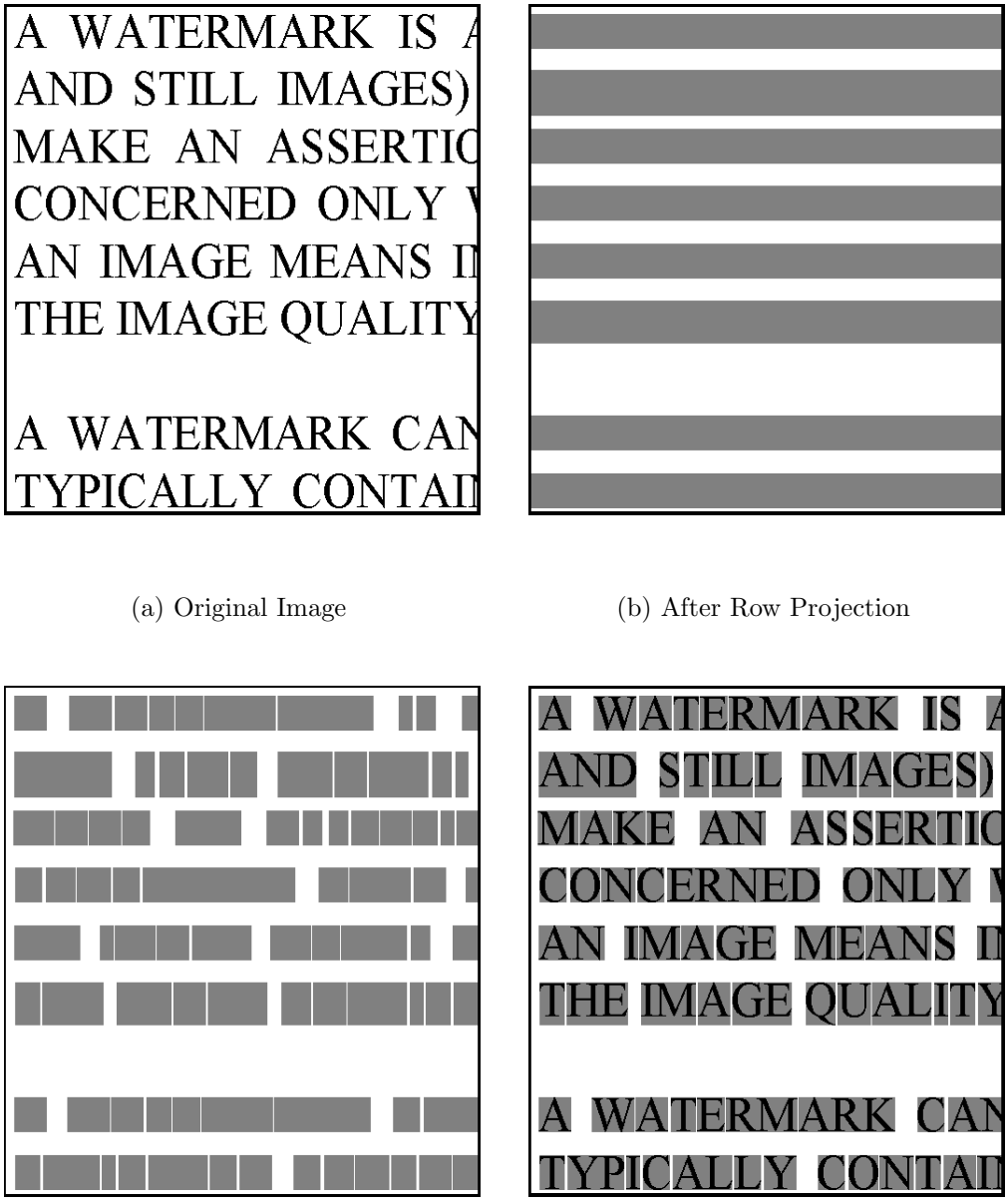
Figure 3.3: Sample Image of Size 512×512 pixels.

3.1.2 Character Segmentation

Segmentation is commonly used for digital image processing. It basically divides the whole image into different categories for the next process: image classification. A large number of techniques have been proposed in literatures [69, 70]. Readers may refer to [71] for more information on image segmentation.

Our proposed watermarking algorithm is to embed information by modifying the letter shapes/features. In order to do this, we need to segment the image into separated letters. Since our test image is the machine-printed document without noise, the segmentation task is simple. Two techniques are used in our experiments: *RXYC technique* [70, 72] and *Contour Tracing technique* [73].

RXYC technique, i.e. recursive X-Y cuts or recursive projection profile cuts



(a) Original Image

(b) After Row Projection

(c) After Column Projection

(d) Character Segmentation

Figure 3.4: Character Segmentation using RXYC Technique.

method, is clearly shown in Figure 3.4. Firstly, the original image (shown in Figure 3.4(a)) is scanned row by row. If there is one black pixel at this row, the whole row is marked as black. After doing this row projection, text lines are

clustered out from the whole image, as shown in Figure 3.4(b).

Secondly, the original image is scanned column by column. For each text line in Figure 3.4(b), if there is no black pixel in Figure 3.4(a), the whole column of this text line is marked as white. Figure 3.4(c) shows the final segmentation result. Mapping Figure 3.4(c) to 3.4(a), it can be seen that most letters are segmented to black rectangles (*bounding boxes*), as shown in Figure 3.4(d). However, some letters are still connected. For example in word "WATERMARK", letters "RM" and "ARK" are connected, which can be seen clearly in Figure 3.5. By using this RXYC algorithm, some characters are segmented wrongly. For example, the word "AN" has two separated letters, but they are segmented as one component, as shown clearly in Figure 3.6. The letters "GE" in word "IMAGE" are also marked wrongly. This is due to the limitation of the column projection.

Figure 3.5: Example of Connected Characters.

As shown in Table 3.1, there are 1,923 letters. 1,140 of them are successfully segmented and the other 732 letters are treated as connected components. Those connected components will not be used to embed data. If we can separate these letters, more data can be embedded and therefore the watermarking capacity will be increased.

This segmentation can also be done by using the so-called contour tracing technique. For each letter, the top, bottom, left and right values can be found

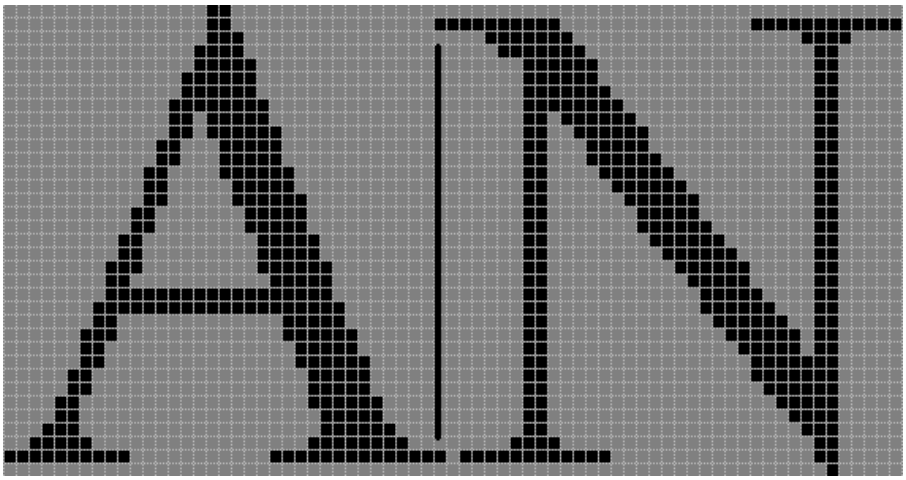


Figure 3.6: Example of Segmentation Error.

Table 3.1: The Statistics of the Test Document.

Type	Number
Paragraph	6
Text Line	41
Word	351
Letter	1923
Space Character	346
Segmented Character	
RXYC	1140
Contour Tracing	1,421
Simulation Time	MATLAB Environment
RXYC	4.844 seconds
Contour Tracing	184.783 seconds

to form the bounding box. By using this method, "AN" and the letters "GE" in words "IMAGE" can be separated. There are 1,421 segmented letters, 281 letters more than using RXYC technique. On the other hand, contour tracing

technique is very tedious and timing-consuming compared to RXYC method. It takes 184.783 seconds for MATLAB simulation, which is much slower than 4.844 seconds of RXYC method.

3.1.3 Optical Character Recognition

Character recognition has been the subject of intensive research during the last decades. It provides a solution for processing large volumes of data automatically. Many OCR algorithms are proposed [74, 75], and there are some commercial OCR products in the market, like FineReader from ABBYY Software House, CuneiForm from Cognitive Enterprises, WinOCR from Electronic Document Technology, etc. In Microsoft Office XP, the feature of handwritten character recognition is also available.

Nowadays, the accuracy of these software is very high even for handwritten characters, and we expect that the accuracy will be 100% for E-documents without noise. We use FineReader 7.0 Professional Edition (Trial version) from ABBYY to recognize the sample image. All the characters are recognized as expected ones. This recognition process is so fast than only takes around 1 second in the test machine mentioned in Chapter 2.

In our experiments, we simulate this OCR function by comparing the characters with their reference letters. We compare a letter with the reference 'A', for example, provided that they have same font size. If the percentage of similarity is higher than the threshold (say 95%), then we can confirm that this letter is 'A'. Since we only consider noise-free binary document, same letters (e.g., all 'A's)

are all identical. The process takes around 329 seconds (more than 5 minutes) in MATLAB environment.

For each bounding box of the segmented characters, the coordinates of top-left corner, height and width are recorded. The result of OCR (ASCII code) is also recorded in a profile as well. The embedding side does not need to send the profile to the receiver side. During the data extraction process, the profile can be created again dynamically. The format of the profile is shown in Table 3.2 (only list 10 records), where the top-left corner of the document image has the coordinate of (1,1).

Table 3.2: Profile of Pre-processing.

index	top	left	height	width	ASCII value	character
1	11	12	34	35	65	A
2	12	71	34	46	87	W
3	11	120	34	35	65	A
4	12	157	33	27	84	T
5	12	185	33	30	69	E
6	12	425	33	15	73	I
7	11	444	35	21	83	S
8	11	493	34	35	65	A
9	11	557	35	21	83	S
10	12	582	33	15	73	I

3.2 Parametric Watermarking Approach

In order to embed data, we need to have a set of outline equations to represent letters. After that, we can modify the letterforms to embed data by adjusting some parameters of the outline equations.

3.2.1 TrueType Font System

TrueType fonts, the most widely used font system, are created by Microsoft for simpler and easier usage of fonts. There are hundreds of TrueType fonts available in all Windows and Mac environments. A TrueType font file contains the information for displaying/printing a smooth font at any size [76].

A TrueType font's glyph shapes are described by their outlines that consist of a series of contours (one or more contours) [77]. These contours are composed of straight lines and curves, where straight lines are defined by two consecutive points, and curves are defined by a series of points describing second-order Bezier-splines. The TrueType Bezier-spline format uses two types of points to define curves, those that are *on* the curve and those that are not on (*off*) the curve.

Any combination of on and off curve points is acceptable when defining a curve, as shown in Figure 3.7. The points that make up a curve must be numbered in consecutive order. It makes a difference whether the order is increasing or decreasing in determining the fill pattern of the shapes that make up the glyph. The direction of the curves has to be such that, if the curve is followed in the direction of increasing point numbers, the black space (the filled area) will always

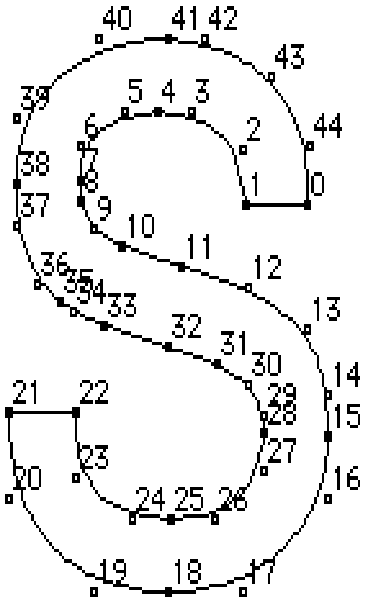


Figure 3.7: A Glyph Description Consisting of *on* and *off* Curve Points [77].

be to the right.

In a TrueType font file, point locations are described in font units, or *FUnits*. An FUnit is the smallest measurable unit in the *em square*, an imaginary square that is used to size and align glyphs [77]. The dimensions of the em square typically are those of the full body height of a font plus some extra spacing to prevent lines of text from colliding when typeset without extra leading.

Figure 3.8 describes the process to display glyphs on raster devices. First, the outline stored in the font file is scaled to the requested size. Once scaled, the outline points are no longer recorded in the FUnits, but have become device-specific pixel coordinates. Next, the instructions associated with this glyph are carried out by the interpreter. The result is a grid-fitted outline for the requested

glyph. This outline is then scan-converted to produce a bitmap that can be rendered on the target device [77].

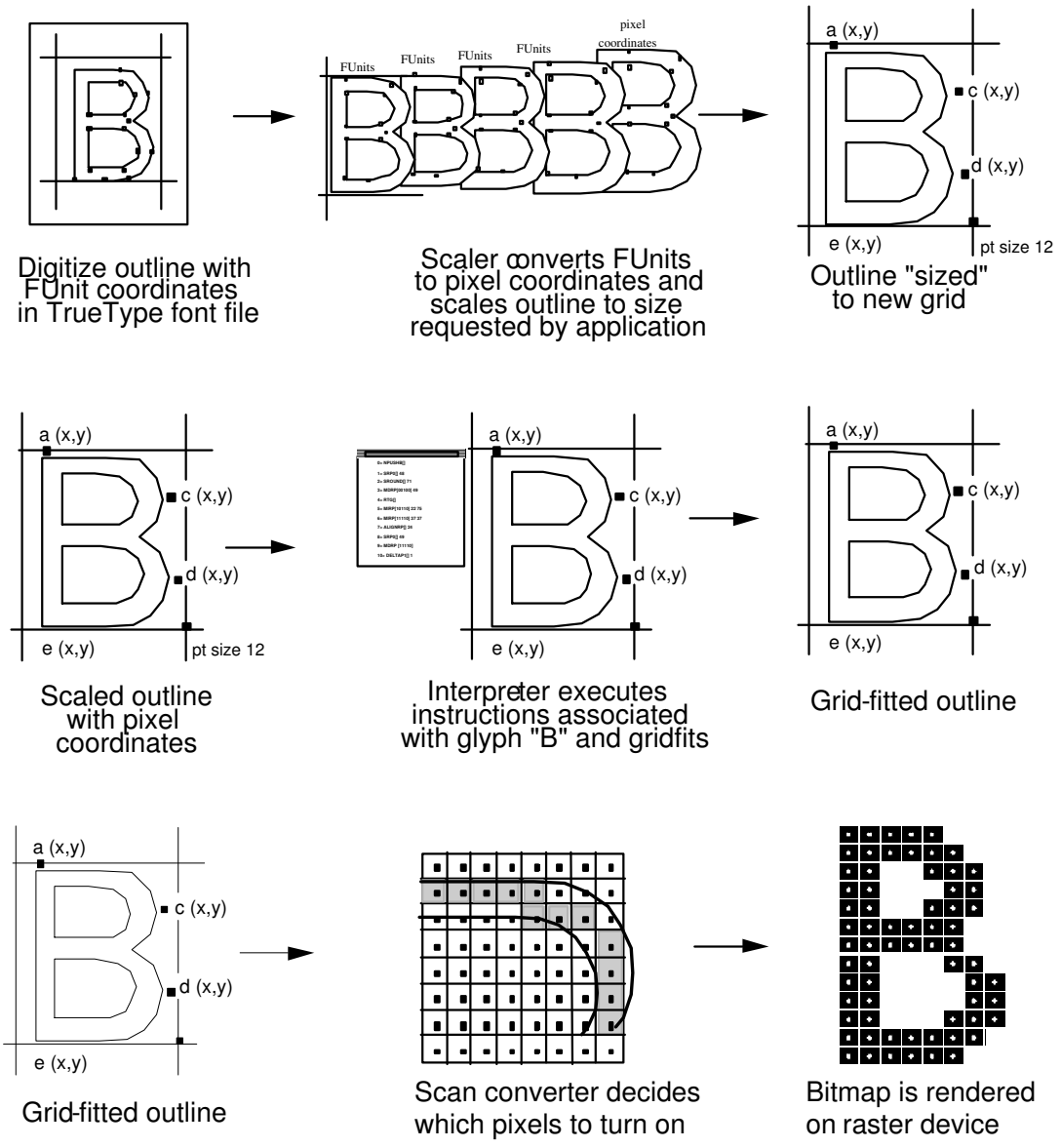


Figure 3.8: TrueType Font Specifications: Digitizing A Design [77].

3.2.2 METAFONT System

Introduced by Knuth [78], the METAFONT system of typeface design is quite different from the previous approaches. It describes the motion of the center of a "pen" or "eraser" instead of describing the boundary/outlines of each character. It is possible to generate many different alphabets from the same general specifications. A "meta-font" is a schematic description of how to draw a family of fonts. The rules of a metafont will thereby define many different individual fonts, depending on the setting of the parameters. The task of the METAFONT is to obtain satisfactory typefaces with parametric variations.

METAFONT system has three unusual characteristics: [2]

- Characteristic 1

  METAFONT understands a special language for drawing shapes with simulated pens that have thicknesses. The actual edge of the curve may be quite complex and difficult to describe, but the pen motion is quite simple.

- Characteristic 2

  The METAFONT language also encourages the construction of designs with explicit parameters, so that a large family of shapes can be described, rather than a single shape. METAFONT is a high-level description that transcends any of the individual fonts it describes.

- Characteristic 3

  In order to support characteristics 1 and 2, METAFONT descriptions of letterforms are given as programs. METAFONT programs are quite different

> from ordinary computer programs because they are largely "declaration" rather than "imperative". In other words, they state relationships that are supposed to hold; they do not tell the computer how to satisfy those conditions. Our eyes interact with the shapes in complicated ways. It is best to let different parts of a design interact, rather than to specify them independently.

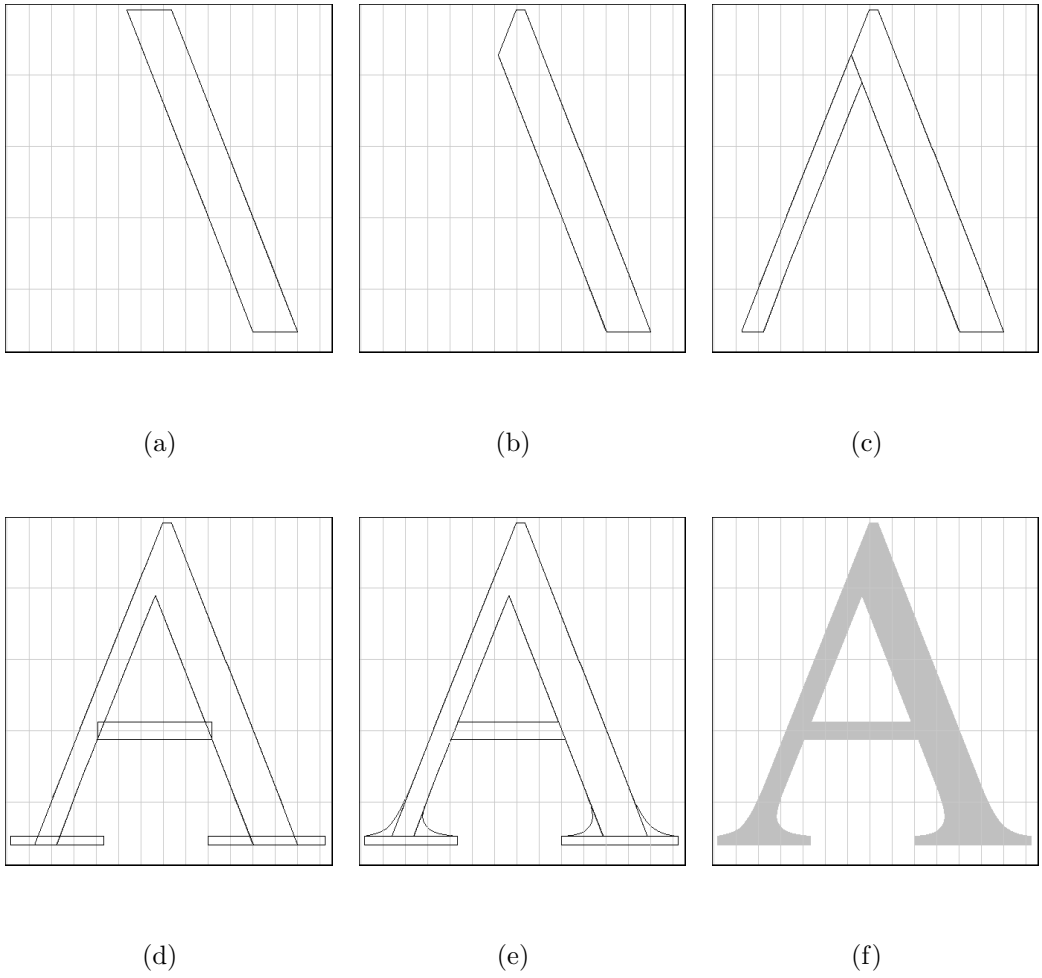
There are 62 parameters in the METAFONT programs [2, 78]. By changing these parameters, it is possible to generate an extensive array of fonts, known as the Computer Modern family of typefaces. These parameters can be divided naturally into several groups:

- Parameters for height of the characters, like height of lowercase letters (with or without ascenders and descenders), height of uppercase letters, etc.
- Parameters for width of the characters.
- Parameters for finer adjustment.
- Parameters for the darkness or "weight" of the characters.

Many of the 62 parameters depend on each other in subtle ways. Thousands of variations that are reasonably near these "standard" values can be expected to provide satisfactory results. Of course, the parameter values must satisfy certain minimum conditions, or disaster will almost certainly ensure. The program will fail at extremely low resolution (fewer than 100dpi) [2].

For example, the METAFONT program has about 20 parameters for letter 'S' [2], telling how big a hairline pen is, how wide it should be when drawing

straight or curved stem lines, and specifying the sizes and proportions of ascenders and descenders, the M-width, the length of serifs, and so forth. By changing these parameters, we obtain infinitely many different styles of types, yet all of them are related and they seem to blend harmoniously with each other.



(a) (b) (c)

(d) (e) (f)

Figure 3.9: Steps to Draw Times New Roman 'A'.

The steps of creating 'A' (Times New Roman) can be briefly shown in Figure 3.9. Figure 3.9(a) draws the first (right) stroke; Figure 3.9(b) removes a left corner of this right stroke; Figure 3.9(c) draws the left stroke; Figure 3.9(d) adds the center stroke and the two horizontal short strokes (called serifs); In Figure 3.9(e)

four curves are used to make the connection smooth; Figure 3.9(f) shows the final shape of character 'A'.

Therefore, to draw this letter 'A' by using METAFONT system, the parameters needed is shown in Figure 3.10. The right stroke is defined by parameters 1 and 2. 'l' and 'r' are the left and right widths of the stroke. Parameters 3 and 4 define the left stroke. These two strokes are intersected at the position 0. The center stroke is defined by parameters 5 and 6. (a1 a2), (b1 b2), (c1 c2) and (d1 d2) are the four curves to make smooth connection of the serifs.

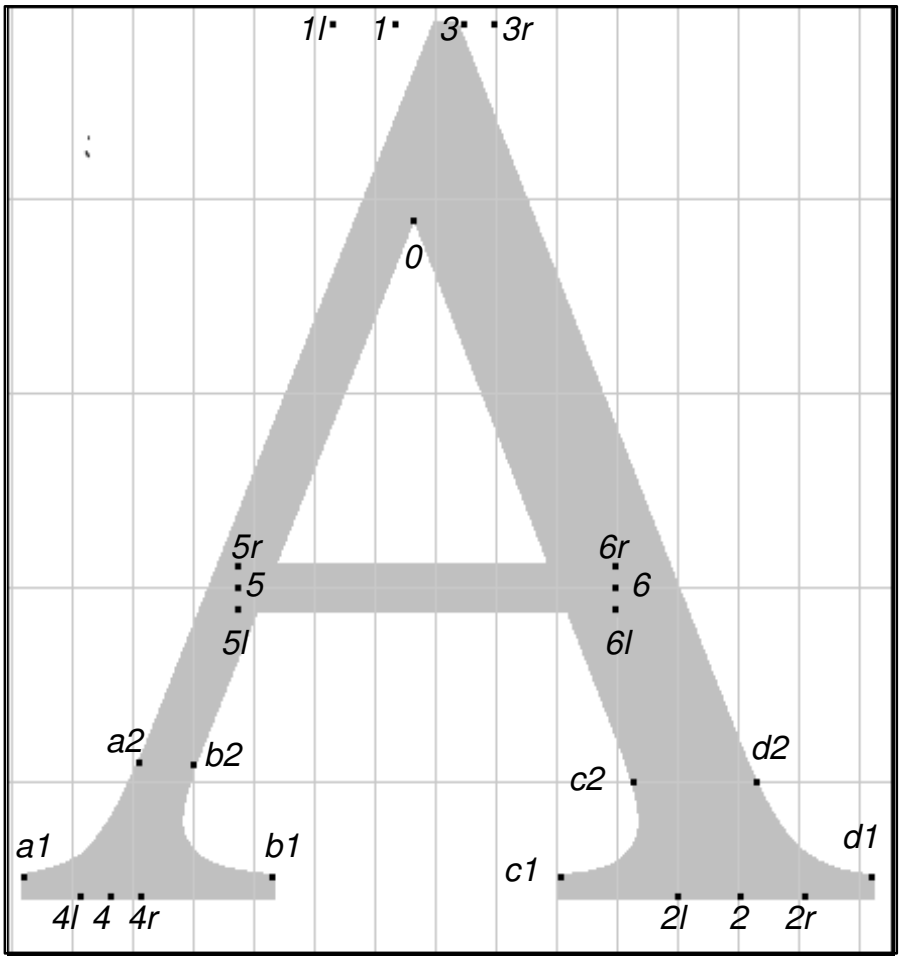


Figure 3.10: Parameters to Draw Times New Roman 'A'.

3.2.3 Outline Equations for Alphabet 'A'

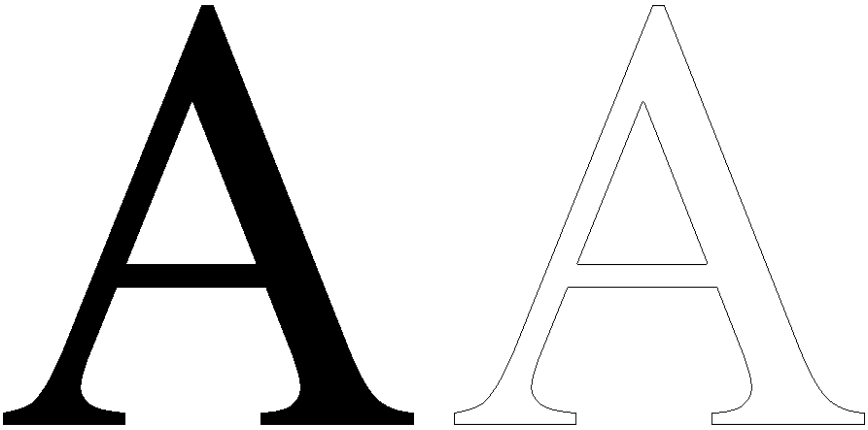
Cabrelli et al. [79] proposed a method to represent binary images through the description of the boundaries of the objects that define such images. One main purpose of this method is to reconstruct the image at any size without losing any quality.

Combining the principles of TrueType font and METAFONT systems, it is possible to describe the outline of a letter or symbol analytically by using a set of equations. By adjusting some parameters of the outline equations, we can create new shapes for the letter. Information can be embedded by replacing the original letterform with a new one. In this way, the embedding capacity can be increased and the watermarked document will have low HVS distortion, the main purpose of our proposal.

In order to get the precise outline equations, it is necessary to use a big letter image, as shown in Figure 3.11(a). It is an uppercase Times New Roman 'A', with font size of 72 points high (typographical units), and it is scanned with 600dpi resolution. The result image has the size of 423×407 pixels. For this given letter image, the outline equations are obtained as follows:

Boundary Extraction

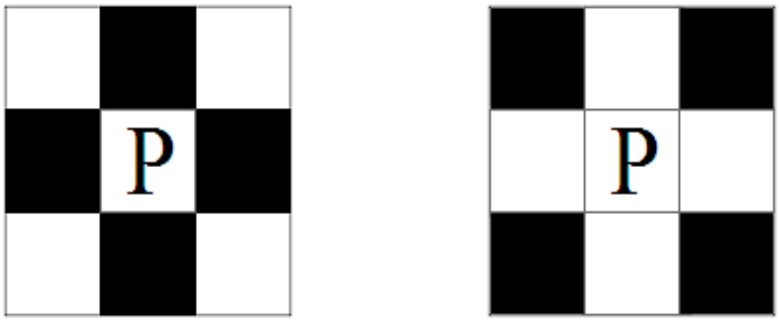
Firstly, we extract the boundary of the letter image. The idea of this algorithm is to recognize the interior of the objects and "paint" it white. Therefore after this process, the remaining black points belong to the boundary of the objects.



(a) Original Image (b) Contour Image

Figure 3.11: Contour of the Original Image (423 × 407 pixels).

The definition of four nearest neighbors (4NN) and four furthest neighbors (4FN) pixels are used, as shown in Figure 3.12. For a pixel P, if all the 4NN pixels are black, it is marked as white. After this process, the boundary extracted is shown in Figure 3.11(b).



(a) 4NN of pixel P (b) 4FN of pixel P

Figure 3.12: The 4NN or 4FN of the Pixel P.

Contour Following

Generally the boundary consists of one or more disjoint closed curves, called as contours. For example in Figure 3.11(b), there are two contours for alphabet 'A': the inner triangle and the remainder.

Contour following is to treat these boundary points as points of a parametric plane curve $(x(n), y(n))$. In other words, we have to provide the boundary points with a natural order. For each boundary point, we will find the next and the preceding points. This step is necessary for the following processes. When performing the contour following (or boundary points ordering), there must be rules to follow.

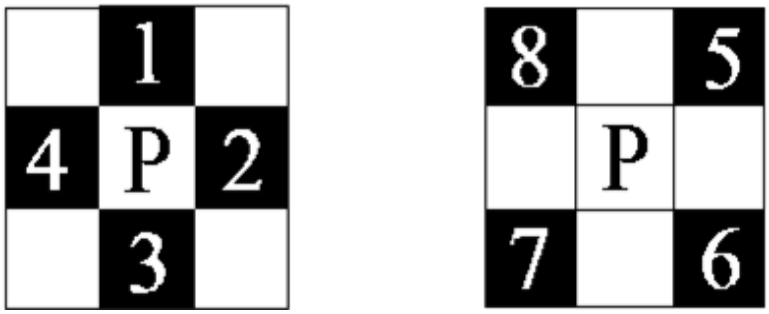


Figure 3.13: Priority Assignment for Contour following.

We search the 4NN points according to the priority, as shown in Figure 3.13. If there are no 4NN points available, the 4FN ones will be searched instead. For this letter 'A', there are 1,621 points for the outer boundary, and 448 points for the inner triangle contour.

Contour Segmentation

After the contour following, these curves need to be segmented. The *singular points* are to be found where the direction of the curve changes abruptly. At those singular points the left and right vector derivatives have different directions. For example, in Figure 3.14, there are 15 singular points for alphabet 'A' (represented by the small circles), and this indicate 15 segments.

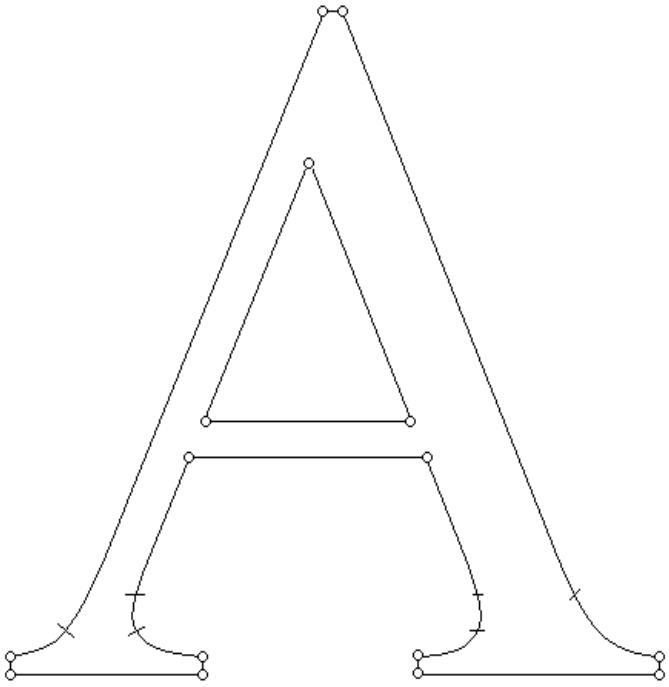


Figure 3.14: The Singular Points and the Subsectors.

Pattern Classification

After the contour segmentation, we need to classify the segments between any two singular points (15 segments here). These segments can be straight lines or curves. For curves, Bezier functions [79] are used to describe these points. If the

average subsector error is too large, we simply split the subsector in half and take different Bezier curves. After the subsector classification, the final result is shown in Figure 3.14, indicated by the short lines.

There are 12 straight lines and 6 curves for the outer contour, and 3 straight lines for the inner boundary component. Hence, even for this simple alphabet 'A', 12+6+3 = 21 equations are required to describe the outline.

Curve Fitting

Curve fitting is to find the mathematical equations for the segments (sectors and subsectors) found from pattern classification. One important fact is: if two curves join in a sector, they have to join smoothly. In other words, the left and right vector derivatives must have the same direction at the joint point.

⋆ Straight Lines

Straight lines are much simpler to be described compared to curves. A line can be represented as $x = c_0 + c_1 y$, with the X-Y coordinate is shown in Figure 3.15.

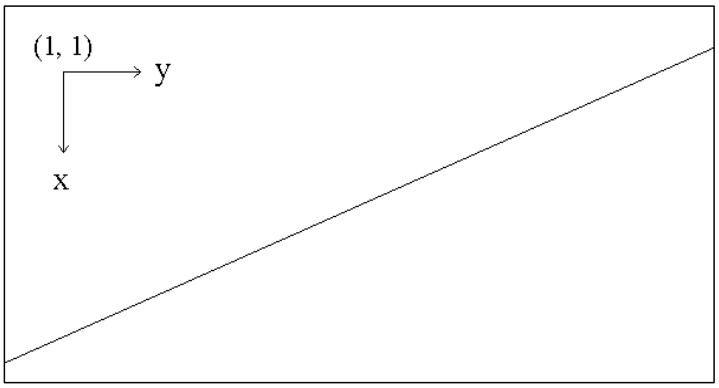


Figure 3.15: A Straight-line Segment.

We use two methods in our experiments to represent a straight line, and we will compare the outcome.

- **Method 1: Two points define a line.**

Mathematically, a line can be defined by 2 points. We use the starting and ending points to draw a line. One hidden assumption of this method is that, it is straight in an analog domain. In Figure 3.16, the line can be seen clearly. The simulation result is shown in Table 3.3. Using two points, $(x_1, y_1) = (284, 1)$ and $(x_2, y_2) = (34, 600)$, we obtain the line equation $x = c_0 + c_1 y = 284.4174 - 0.4174y$. The difference of these two lines is around 16.33%, as shown in Figure 3.17.

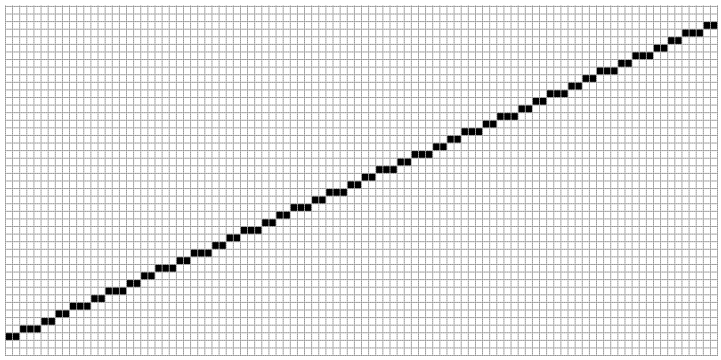
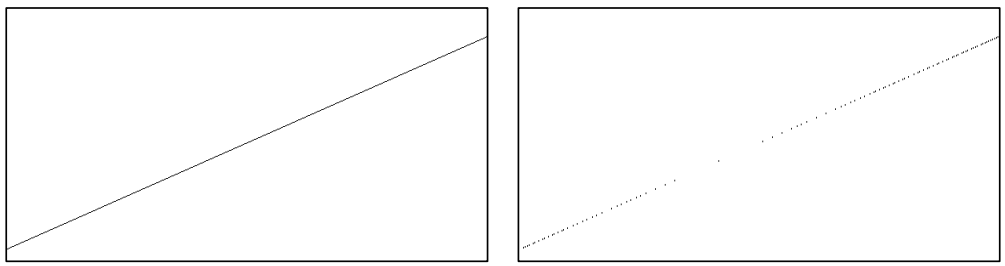


Figure 3.16: The Close View of the Straight-line Segment.



(a) Reconstructed Line (b) Difference Image

Figure 3.17: The Result of Two-point Line Fitting.

Table 3.3: The Result of Two-point Line Fitting.

No. of Black pixels	$N = 600$
Starting point	(284, 1)
Ending point	(34, 600)
$x = c_0 + c_1 y$ $y = 1, 2, 3, ..., 600$	$c_0 = 284.4174$ $c_1 = -0.4174$
No. of different pixels	98
% of error	98/600 = 16.33 %

- **Method 2: Minimum-Square-Error line fitting**

In the second method, all the pixels are used to find the best fitting line having minimum square error (MSE). More precisely, given a set of points (x_i, y_i), $i = 1, 2, ...n$, in the plane, we seek two numbers c_0 and c_1 such that the error function

$$\sum_{i=1}^{n} [(c_0 + c_1 x_i) - y_i]^2 \tag{3.1}$$

is minimum. By using the pseudo-inverse of a matrix, this problem can be elegantly solved. The coefficients of this equation u can be obtained as

$$u = (A^T A)^{-1} A^T b \tag{3.2}$$

where

$$A = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ . & \\ . & \\ . & \\ 1 & x_n \end{bmatrix}, \; u = \begin{bmatrix} c_0 \\ c_1 \end{bmatrix}, \; b = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_n \end{bmatrix}$$

For the detailed mathematical proof, please refer to [80]. The assumption of this solution is that the columns of A are linearly independent to each other. That is, we cannot apply this method to horizontal and vertical lines, for x(n) and y(n).

The simulation result is shown in Table 3.4, from which we find that the calculated line matches the original one perfectly with 0% error. Comparing Table 3.3 (16.33% error) and Table 3.4, it can be seen clearly that MSE line fitting method is much more accurate than two-point line fitting method. In our demonstration, we only apply the two-point line fitting method for the horizontal and vertical lines, and use MSE line fitting method for the others.

Table 3.4: The Result of MSE Line Fitting.

No. of Black pixels	$N = 600$
Starting point	(284, 1)
Ending point	(34, 600)
$x(n) = a_x n + b_x$ $x(n) = 0.6861n - 251.0471$	$a_x = 0.6861$ $b_x = -251.0471$
$y(n) = a_y n + b_y$ $y(n) = -n + 600$	$a_y = -1$ $b_y = 600$
$n = i/N$	$i = 1, 2, ..., N$
No. of different pixels	0
% of error	0%

⋆ Curves

The pseudo-inverse approach for MSE line fitting can be easily extended to find the MSE polynomial that fits to a set of points. One example is shown in Figure 3.18.

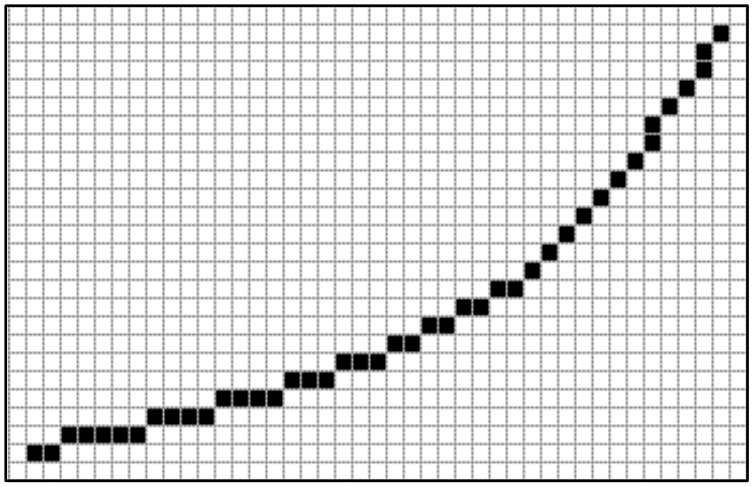


Figure 3.18: Example of a Curve.

For a curve of d order:

$$y = c_0 + c_1 x + c_2 x^2 + ... + c_d x^d \tag{3.3}$$

let u ($u = (A^T A)^{-1} A^T b$) be the coefficients which will make minimum square error. Now

$$A = \begin{bmatrix} 1 & x_1 & x_1^2 & .... & x_1^d \\ 1 & x_2 & x_2^2 & .... & x_2^d \\ . & & & & \\ . & & & & \\ . & & & & \\ 1 & x_n & x_n^2 & .... & x_n^d \end{bmatrix}, \; u = \begin{bmatrix} c_0 \\ c_1 \\ . \\ . \\ . \\ c_n \end{bmatrix}, \; b = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_n \end{bmatrix}$$

Obviously, the higher order polynomial equation we use, the more accurate the curve fitting will be. To prove this, we compare the performance of 3rd-order and 5th-order polynomial equations as described below.

$$x(n) = a_0 + a_1 n + a_2 n^2 + a_3 n^3 \tag{3.4}$$

$$y(n) = b_0 + b_1 n + b_2 n^2 + b_3 n^3$$

$$x(n) = a_0 + a_1 n + a_2 n^2 + a_3 n^3 + a_4 n^4 + a_5 n^5 \tag{3.5}$$

$$y(n) = b_0 + b_1 n + b_2 n^2 + b_3 n^3 + b_4 n^4 + b_5 n^5$$

Their respective coefficients are shown as follows:

$$u_{x\_3rd} = \begin{bmatrix} -0.0223 \\ -9.1438 \\ 6.8072 \\ -21.5944 \end{bmatrix} \quad u_{y\_3rd} = \begin{bmatrix} -0.6219 \\ 38.2009 \\ 14.2668 \\ -11.5289 \end{bmatrix}$$

$$u_{x\_5th} = \begin{bmatrix} -0.4383 \\ 3.1083 \\ -87.8819 \\ 255.5246 \\ -336.8622 \\ 143.6934 \end{bmatrix} \quad u_{y\_5th} = \begin{bmatrix} -0.8759 \\ 39.8090 \\ 24.8857 \\ -79.8146 \\ 110.9462 \\ -55.3365 \end{bmatrix}$$

The line fitting error is 6.93% (3-bit error) for the 3rd-order polynomial and only 2.33% (1-bit error) for the 5th-order one, as shown in Table 3.5.

Table 3.5: The Result of MSE Curve Fitting.

No. of Black pixels	$N = 43$
Starting point	(25, 2)
Ending point	(2, 42)
n = i/N	i = 1, 2, ... N
Using 3rd order of polynomial	
No. of different pixels	3
% of error	3/43 = 6.98 %
Using 5th order of polynomial	
No. of different pixels	1
% of error	1/43 = 2.33 %

After fitting all the segments (lines and curves), the total set of outline equations describing the alphabet 'A' is complete. The final set of equations to describe 'A' is shown in Table 3.6 with total 21 equations (15 lines and 6 curves).

In *Start Point* column of the table, '——' means that this segment (line/curve) starts from the end of the previous one. In *coefficients* column, if no coefficient is available, two-point line fitting method is used to draw this line by using the starting and ending points. These lines are either horizontal or vertical. For curves, the coefficients are shown in Table 3.7 using 5th-order polynomial equations.

Table 3.6: Outline Equations of Alphabet 'A'.

No.	Start Point (x, y)	End Point (x, y)	Line/ Curve	Coefficients $[c_0, c_1, c_2, c_3, c_4, c_5]$
1	(1, 205)	(1, 217)	Line	———
2	———	(358, 368)	Line	c_x=[0, 358] c_y =[216.3193, 150.1989]
3	———	(396, 423)	Curve	c_{x1}, c_{y1}
4	———	(407, 423)	Line	———
5	———	(407, 266)	Line	———
6	———	(396, 266)	Line	———
7	———	(380, 305)	Curve	c_{x2}, c_{y2}
8	———	(358, 305)	Curve	c_{x3}, c_{y3}
9	———	(274, 271)	Line	c_x =[358, -84] c_y =[305.5978, -34.2424]
10	———	(274, 118)	Line	———
11	———	(358, 83)	Line	c_x =[273, 85] c_y =[117.7193, -35.9336]
12	———	(380, 83)	Curve	c_{x4}, c_{y4}
13	———	(396, 126)	Curve	c_{x5}, c_{y5}
14	———	(407, 126)	Line	———
15	———	(407, 1)	Line	———
16	———	(396, 1)	Line	———
17	———	(358, 51)	Curve	c_{x6}, c_{y6}
18	———	(1, 205)	Line	c_x =[358, -356] c_y =[52.5912, 152.1541]
19	(94, 195)	(252, 261)	Line	c_x =[93, 158] c_y =[195.5504, 65.9375]
20	———	(252, 127)	Line	———
21	———	(94, 195)	Line	c_x =[252, -157] c_y =[126.8589, 67.2804]

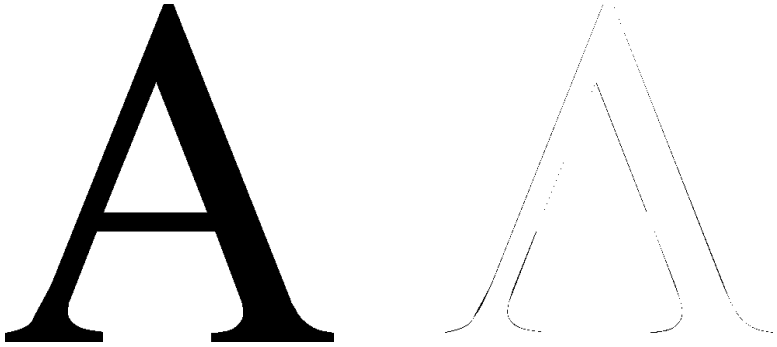
Table 3.7: Curve Coefficients of the Outline Equations.

c_{x1}	[-0.1963, 42.9248, 128.5022, -273.8607, 169.4549, -28.3268]
c_{y1}	[-0.8455, 35.2895, -17.9128, 194.4019, -269.2354, 113.6435]
c_{x2}	[-0.5312, 14.7085, -113.8482, 281.7765, -311.1720, 112.6853]
c_{y2}	[-0.7115, 34.4764, 44.0667, -121.9778, 146.4109, -63.1879]
c_{x3}	[1, -23, 0, 0, 0, 0]
c_{y3}	[0.1037, -3.6895, 79.8928, -230.1261, 244.1597, -90.9144]
c_{x4}	[-1, 23.0000, 0, 0, 0, 0]
c_{y4}	[-0.3787, -8.6506, 42.3409, -142.6974, 190.0383, -81.5009]
c_{x5}	[-2.0095, 68.9086, -132.1289, 145.1902, -84.8131, 21.1411]
c_{y5}	[-0.8298, 28.3232, 79.8354, -168.5586, 161.9831, -57.8929]
c_{x6}	[1.3621, -38.0443, 167.7920, -414.3184, 330.8198, -85.1975]
c_{y6}	[-1.3091, 67.7489, -70.0902, 218.6197, -276.2796, 111.8677]

Reconstructured Letter Shapes

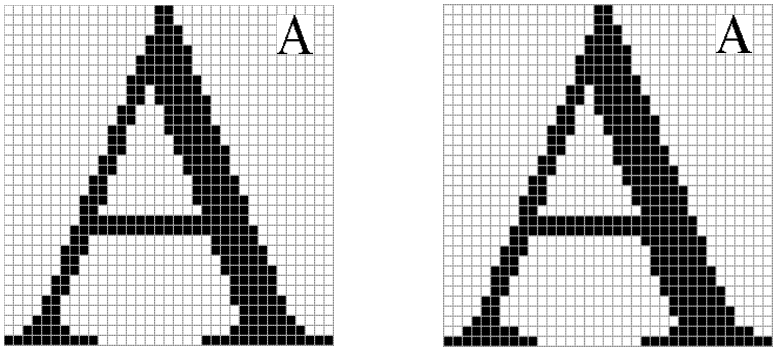
We use the outline equations to reproduce the alphabet 'A' with 407-pixel height and 423-pixel width, as shown in Figure 3.19(a). Compared to the original image (40,613 black pixels), the reconstructed one has 497 different pixels, as shown in Figure 3.19(b). The difference rate is 497/40,613 = 1.2237%.

The letterform in small size (34-pixel width) are drawn in Figure 3.20(b). We choose three independent parameters to be adjusted for data embedding, which means that each 'A' can host 3 bits of information. In Figure 3.20(c), the left serif of 'A' is 1-pixel longer than the one in Figure 3.20(b). In Figure 3.20(d), the right serif is 1-pixel longer. In Figure 3.20(e), the image is 1-pixel shorter. It seems that human eyes are not sensitive to these modifications, and their zoom-out letter images are quite similar.

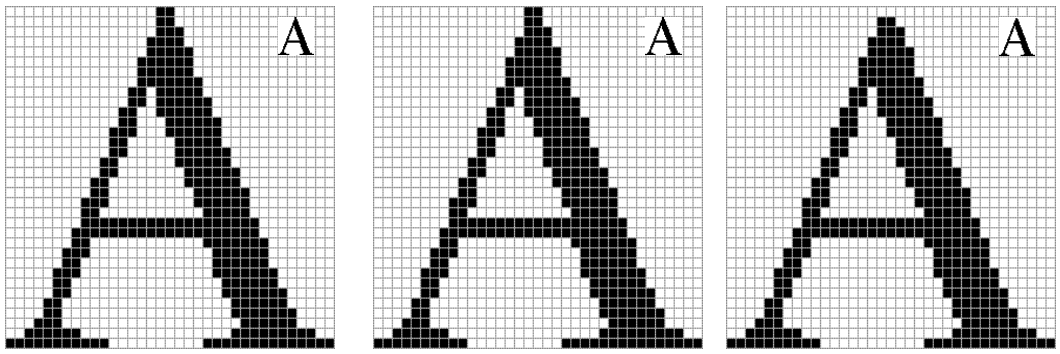


(a) Drawn Image (b) The Difference

Figure 3.19: The Result Image (423 × 407 pixels) using Outline Equations.



(a) Original Image (b) Drawn Image



(c) Left Serif (d) Right Serif (e) Height

Figure 3.20: The Result Images (35 × 34 pixels) using Outline Equations.

Figure 3.21 shows the watermarked image. It can be seen that the visual quality is quite good.

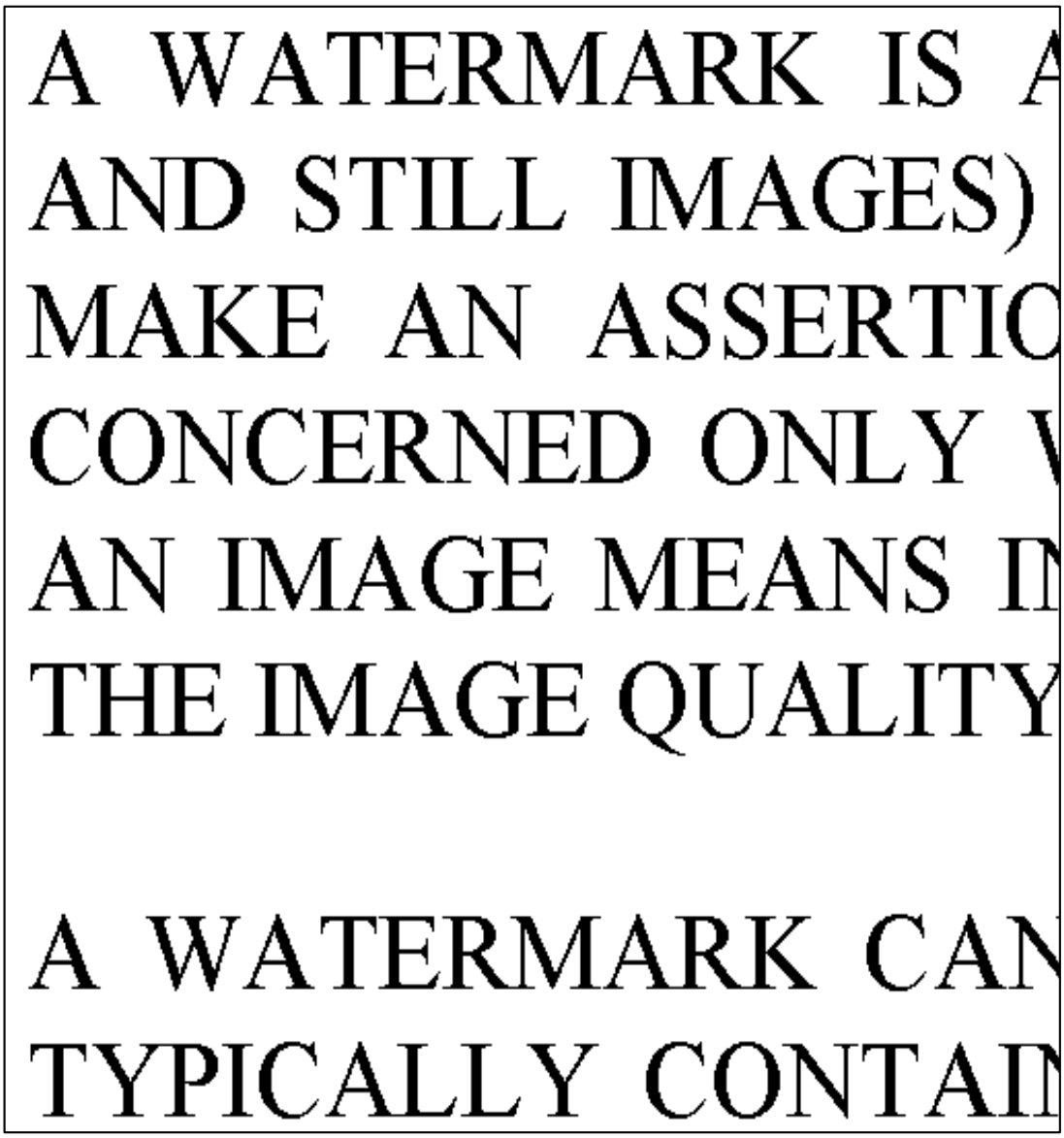


Figure 3.21: Watermarked Image of Parametric-based Approach (512 × 512 pixels).

In this image, only alphabet 'A's are used to embed data. For the whole document, there are 176 'A's, in which 84 are not connected to others characters. The

first 'A' is set as the reference, and no modification on it is allowed. A piece of 3-bit information can be embedded in each of all the other 'A's. Therefore the total embedding capacity is $83 \times 3 = 249$ bits. For the complete system (all 26 alphabets are used to embed data), if we assume the average embedding rate is 3 bits per letter, the watermarking capacity will be much higher. There are 1,421 segmented letters, among which 26 are set as references. Therefore there are $1,421 - 26 = 1,395$ letters that can be used to embed data. The total capacity is $1,395 \times 3 = 4,185$ bits.

3.3 Summary

In this chapter, the block functions of the watermarking schemes are studied in detail. RXYC technique is implemented to segment the document image into separated letters. It is simple and effective for machine-printed document without noise, while some letters are segmented wrongly as connected component. In order to increase the watermarking capacity, contour tracing technique is used. The product FineReader 7.0 is applied to recognize the segmented letters. For the test document image, the recognition result is perfect with 100% accuracy.

The fundamentals of TrueType font and METAFONT systems are studied, and we demonstrate how to produce accurate outline equations for alphabet 'A'. Using these outline equations, alphabet 'A's with different sizes and shapes are produced. Three parameters of the outline equations are chosen to make similar letter shapes with the original one. By replacing the original letters with these created letters, data can be embedded. The parametric-based approach has high

watermarking capacity, while the visual quality is quite good in human visual perception. During the watermark extraction process, the non-reference letters are replaced with their respective reference. Therefore the original image can be reconstructed, and our proposed watermarking algorithm is reversible.

Chapter 4

Pixel-based Watermarking Approach

In this chapter, the features of alphabet 'A' are studied. The features are normally located at the edges of the letter. It is noted that the change (flipping of these feature pixels) will have very little distortion to the letter. Using these features, data can be embedded into the document image to achieve high visual quality. A lower resolution (150dpi) document image is also studied. Experimental results are presented and discussed. Finally, the vertical word/character-shift coding method is proposed and tested.

4.1 Features of Alphabet 'A'

As mentioned before, the human visual system does remarkably well at identifying text set off with respect to the logical baseline. In order not to break this rule,

we keep the outer boundary of each letter unchanged. That is, we only choose those feature points which will not affect the width or height of the letter. Eight feature points are chosen for alphabet 'A' to embed data, which are numbered from left to right and from bottom to top as shown in Figure 4.1.

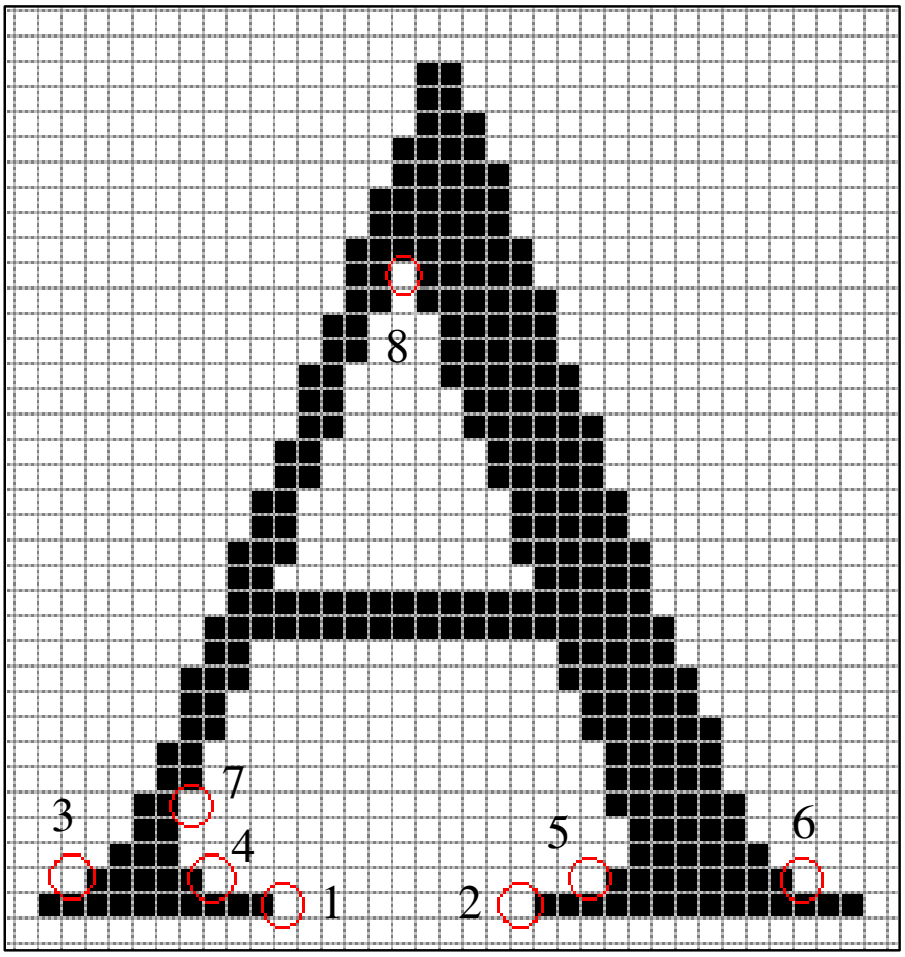
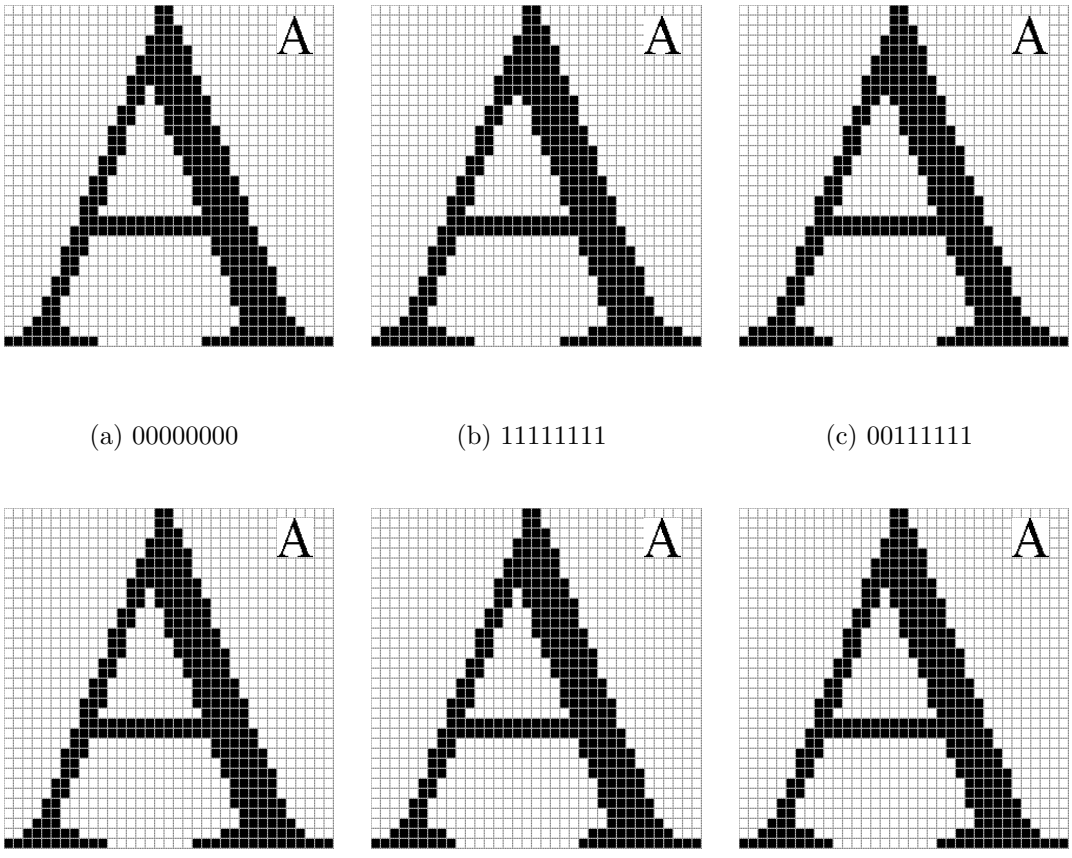


Figure 4.1: Feature Points for Alphabet 'A' (35 × 34 pixels).

Figure 4.2(a) shows the original image of alphabet 'A'. In the Figure 4.2(b), all the eight feature points are flipped. Figure 4.2(c) flipped all the feature points except the first and second features. The small 'A' in each image is the zoom-out version.

In order to embed data, we can choose to flip black pixels to white or vice versa. For example at feature point 1 in Figure 4.1, we can choose to increase or decrease the horizontal 10-pixel long segment by one pixel. In Figure 4.2(d), the feature points 1, 2, 4 and 5 are all flipped from white to black; while in Figure 4.2(e), the feature points 4 and 5 are flipped from white to black, and the feature points 1 and 2 are changed from black to white. Compared to Figure 4.2(e), it can be seen that the distortion in Figure 4.2(d) is much lower in human visual perception. Therefore, we decide to flip all the feature points from white to black.



(a) 00000000 (b) 11111111 (c) 00111111

(d) 11011000 (e) 11011000 special (f) 00011000

Figure 4.2: Result Images after Features Changed (35×34 pixels).

In Figure 4.1, the feature points 4, 5, 6 and 7 are less noticeable, since human eyes are less sensitive to the modifications in curves than in straight lines. Some are more noticeable, like feature 8. Some combinations of feature points may cause less distortion than changing single feature point. For example, if feature 1 and feature 4 are both flipped at the same time, it will be less noticeable than only flipping the 4th feature point, as shown in Figure 4.2(c). It can be seen that it is difficult to differentiate these six small letters in Figure 4.2 by human eyes.

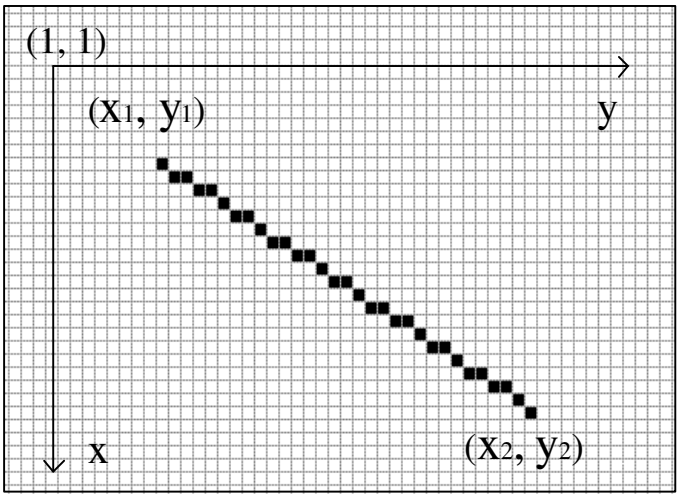


Figure 4.3: System's Coordinate.

Using the coordinate system in Figure 4.3, the eight feature points for alphabet 'A' are recorded in the Table 4.1, where (x_1, y_1) is the starting point of the segment, (x_2, y_2) is the ending point, and α is the control factor indicating the point of the pixel to be flipped in the segment. The relationship of these variables is shown as the following equation:

$$(x, y) = \alpha(x_1, y_1) + (1 - \alpha)(x_2, y_2) \tag{4.1}$$

Table 4.1: Feature Points of Alphabet 'A'.

Feature No.	Start Point (x_1, y_1)	End Point (x_2, y_2)	Segment Length	Control Factor (α)
1	(34, 1)	(34, 10)	10	-1/9
2	(34, 22)	(34, 35)	14	14/13
3	(33, 3)	(33, 7)	5	5/4
4	(33, 3)	(33, 7)	5	-1/4
5	(33, 25)	(33, 32)	8	8/7
7	(25, 7)	(29, 7)	5	-1/4
6	(33, 25)	(33, 32)	8	-1/7
8	(4, 16)	(8, 16)	5	-1/4

or

$$x = \alpha x_1 + (1 - \alpha)x_2$$

$$y = \alpha y_1 + (1 - \alpha)y_2$$

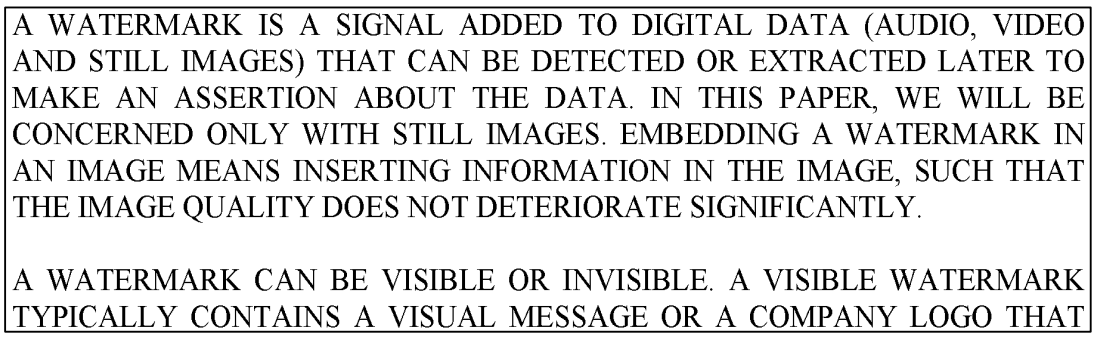
For example, the first feature point (34, 11) can be calculated from,

$$x = (-1/9) \times 34 + (1 - (-1/9)) \times 34 = 34$$

$$y = (-1/9) \times 1 + (1 - (-1/9)) \times 10 = 11$$

For the watermark embedding process, in the document image as shown in Figure 4.4, the first 'A' is set as the reference without modification. Reference letters are used to recover the original document image at the watermark extraction process. For example, to embed a piece of 16-bit data "10000001,01010000", the first 8 bits are embedded in the first 'A' of word "WATERMARK", and the next 8 bits are embedded in the next separated 'A' (after word "IS"). To embed "10000001",

only the first and eighth feature points are flipped while the others remain unchanged. For the next 8-bit data "01010000", only the second and fourth features are modified. From Figure 4.4, it can be seen that the visual quality is quite good. Even in the big image shown in Figure 4.5, it is difficult for naked eyes to find the differences between 'A's.


A WATERMARK IS A SIGNAL ADDED TO DIGITAL DATA (AUDIO, VIDEO AND STILL IMAGES) THAT CAN BE DETECTED OR EXTRACTED LATER TO MAKE AN ASSERTION ABOUT THE DATA. IN THIS PAPER, WE WILL BE CONCERNED ONLY WITH STILL IMAGES. EMBEDDING A WATERMARK IN AN IMAGE MEANS INSERTING INFORMATION IN THE IMAGE, SUCH THAT THE IMAGE QUALITY DOES NOT DETERIORATE SIGNIFICANTLY.

A WATERMARK CAN BE VISIBLE OR INVISIBLE. A VISIBLE WATERMARK TYPICALLY CONTAINS A VISUAL MESSAGE OR A COMPANY LOGO THAT

Figure 4.4: Watermarked Image of Pixel-based Approach (1820 × 512 pixels).

The watermark extraction process is similar to the one for the parametric watermarking approach. The first appeared letter 'A' is the reference. For the remaining 'A's, their features are checked to extract the embedded data. For example, if the first and eighth features are modified while the others remain unchanged, the detected data is "10000001". Similarly, another 8 bits "0101000" can be extracted from the next 'A'. In this way, the embedded data is retrieved. After that, the original document image can be reconstructed by replacing the non-reference letters with their respective reference. Therefore, it is also a reversible watermarking method.

As mentioned before, there are 176 'A's for the whole document, in which 84 are not connected to other characters. One 'A' is set as the reference. For other

'A's, each can host 8 bits. The total embedding capacity is $83 \times 8 = 664$ bits. For the complete system (all the 26 alphabets are used to embed data), there are 1,421 segmented letters, among which 26 are set as reference letters. If we assume other letters can also embed 8 bits each, the capacity will be $(1,421 - 26) \times 8 = 11,160$ bits. This embedding capacity is sufficient for many applications.

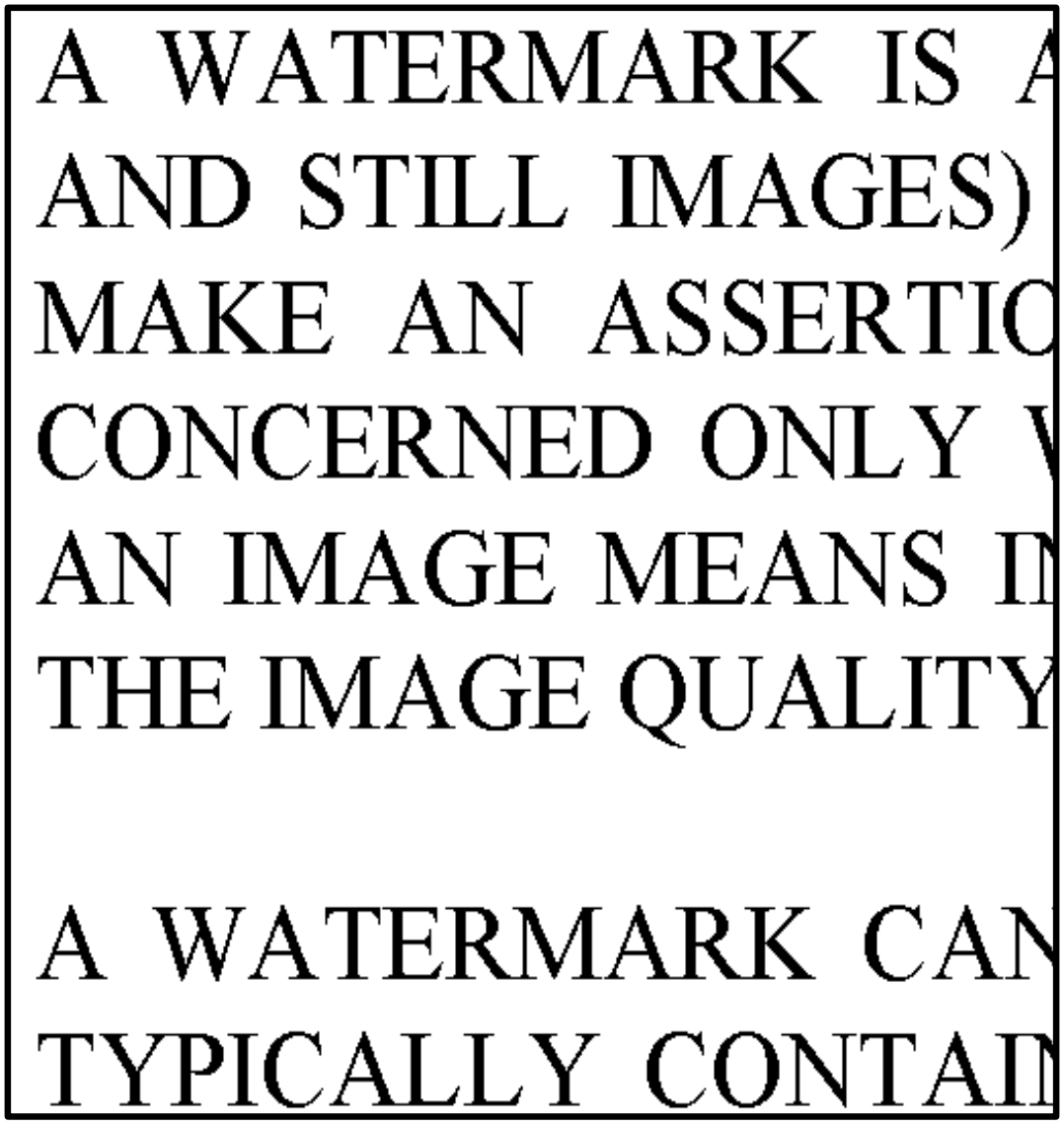


Figure 4.5: Watermarked Image of Pixel-based Approach (512×512 pixels).

4.2 Pixel-based Approach for Lower Resolution Document

If we decrease the resolution of the document image, the letter features available for embedding data will be reduced. Therefore, the watermarking capacity tends to lower, and the visual quality turns to poorer as well. The document image with 150dpi resolution is studied in our experiment. Figure 4.6 shows the alphabet 'A' with 150dpi resolution, having 18-pixel width and 17-pixel height. In order to keep the visual quality for human visual perception satisfactory, we only choose five feature points (instead of 8) from each 'A', numbered from left to right and from bottom to top.

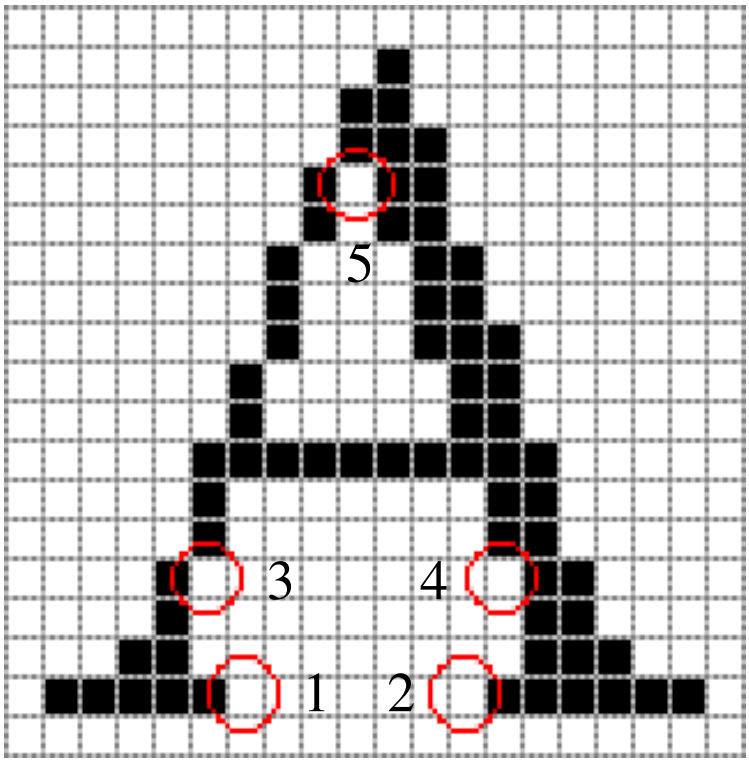
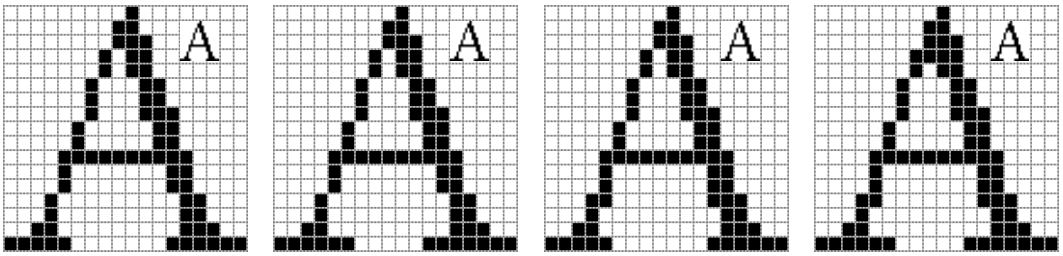


Figure 4.6: Features for Alphabet 'A' with 150dpi Resolution (18×17 pixels).

Figure 4.7 shows the letterforms with some feature points modified. Figure 4.7(a) shows the original letter. In Figure 4.7(b), the first and second feature points are flipped. Figure 4.7(c) shows the letter image with feature points 3 and 4 changed. In Figure 4.7(d), all the five features are modified. It can be seen that the four smaller letters look quite similar and it is difficult for human eyes to notice the difference.



(a) 00000 (b) 11000 (c) 00110 (d) 11111

Figure 4.7: Letterforms after Features Modification (18 × 17 pixels).

The resultant document image with data embedded to the separated 'A's is shown in Figure 4.8. Compared to the document at 300dpi resolution, there are more separated letters available for embedding. For example, in word "WATERMARK", letters "ARK" are no longer connected together, which can be seen clearly in Figure 4.9. There are now 104 separated 'A's available for embedding data, 20 more than the one in the document with 300dpi resolution. Therefore, the capacity is $(104 - 1) \times 5 = 515$ bits, which is not as low as expected.

For the watermarked image, it is still difficult to find the differences between the 'A's. But the visual quality of the document image is much poorer than the one with 300dpi resolution. The letter strokes/edges are not very smooth. The

resolution of 300dpi is more suitable to be used for electronic documents and for the data hiding.

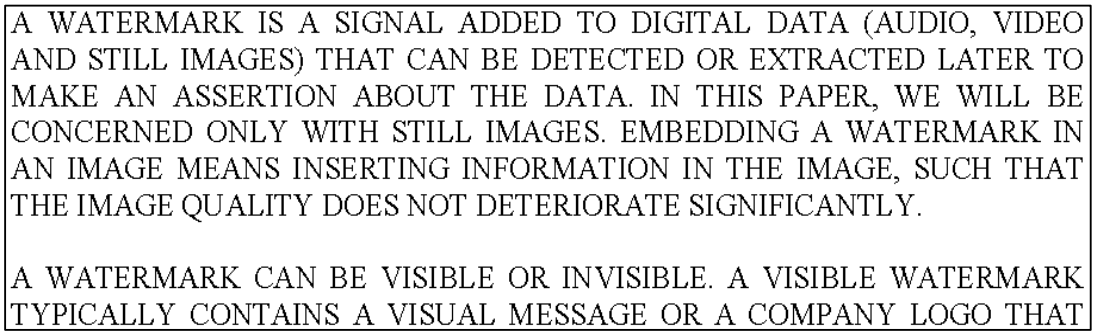


Figure 4.8: Watermarked Image with 150dpi Resolution (910 × 256 pixels).

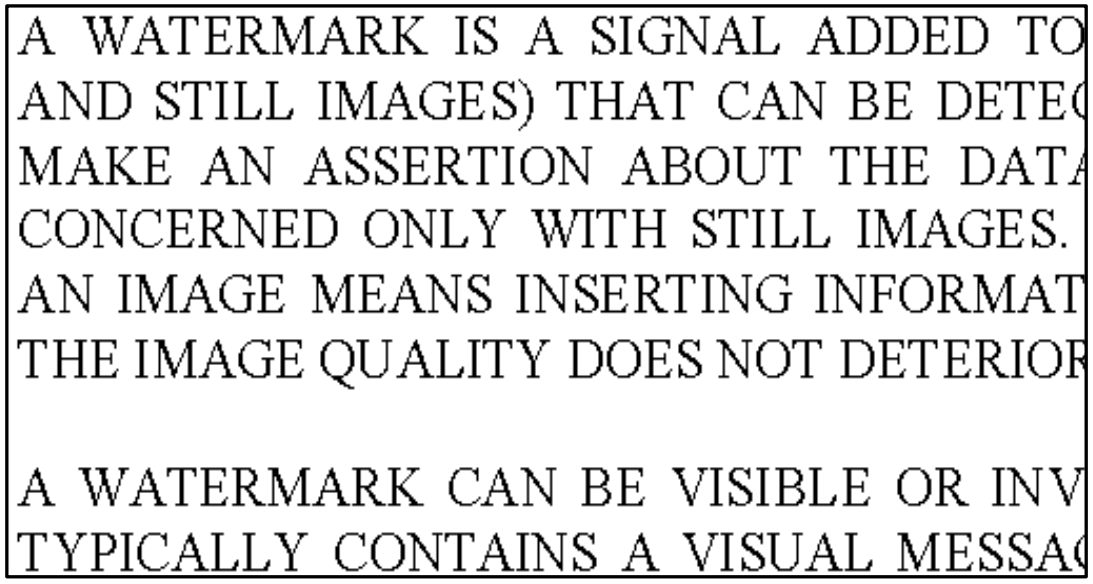


Figure 4.9: Enlarged Watermarked Image with 150dpi Resolution (512 × 256 pixels).

4.3 Vertical Word/Character-shift Coding

For the proposed document image watermarking algorithms like word-shift coding and character-shift coding [17, 18, 19, 20], the relative horizontal spaces between words/characters are modified to embed data while the *vertical* positions are not touched. The reason is simple as mentioned before that the human visual system does remarkably well at identifying text set off with respect to the logical baseline [8]. For document images, the logical baseline is horizontal. We want to test the vertical word/character-shift coding while trying not to break this horizontal baseline for text, and study the HVS distortion of the result image.

Firstly, we need to check the logical baseline feature of characters (average bottom level of the characters in the same word/text line). For font type of Times New Roman with font size of 12, the characters are converted with 300-dpi resolution, and their heights are shown in Table 4.2.

Table 4.2: The Alignment of Upper-case Characters.

Groups	Character Height	Characters
Group 1	33	BDEFHIKLMPRTXYZ
(16 letters)	34	A
Group 2	34	NJUVW
(9 letters)	35	CGOS
Group 3	43	Q
(1 letter)		

According to characters' baseline, the 26 upper-case characters can be divided into three groups as shown in Table 4.2. The heights of the characters varies

from 33 to 43 pixels, where most characters have heights of 33/34 pixels (21 letters), only 4 letters have 35-pixel heights and 1 letter have 43-pixel height. If we set the baseline of the characters in Group 1 as the reference, the baseline is 1 pixel lower for Group 2 characters, and 9 pixels lower for 'Q'.

Some example characters from the three groups are shown in Figure 4.10. It can be seen clearly that there is 1-pixel difference at the bottom between 'A' and 'J'. For 'Q', the bottom is much lower due to its tail.

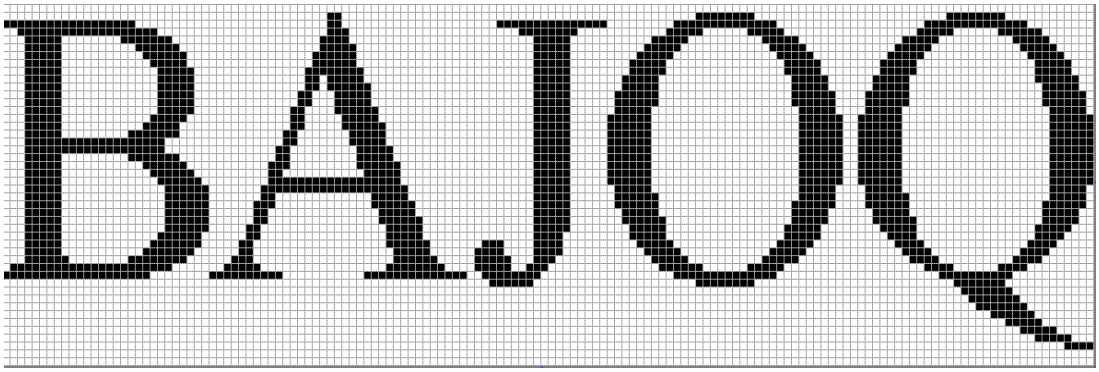
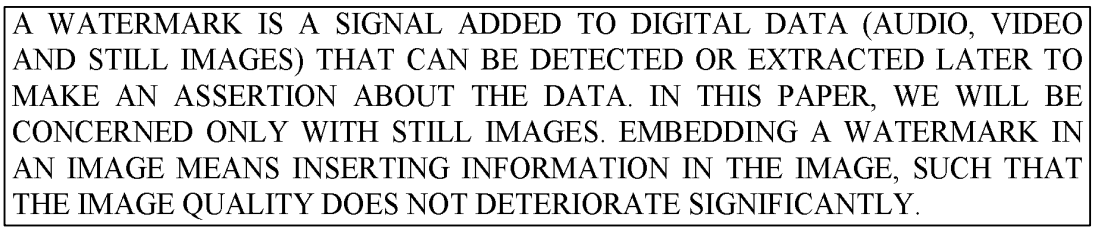


Figure 4.10: Upper-case Characters Alignment.

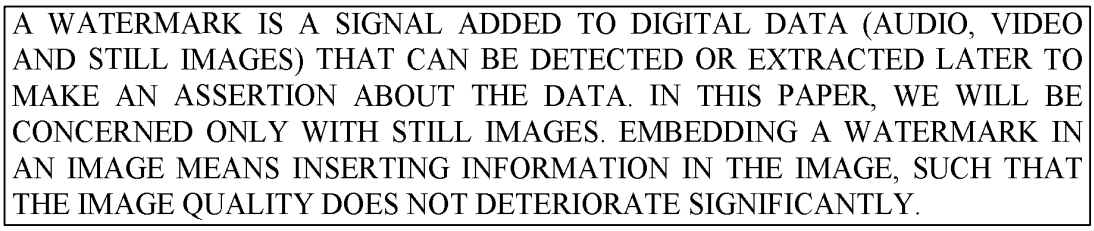
Since the heights of characters are not the same and their bottoms are not aligned, it is possible to move some characters up or down by 1 pixel, while still keeping the logical baseline for text and therefore maintaining the low distortion for human visual system. We want to see this vertical word/character shifting effects.

One example of vertical word/character-shift coding is shown in Figure 4.11. Figure 4.11(a) is the original image. In Figure 4.11(b), the second and third text lines are modified to show the word-shifting effect. The words "STILL", "THAT", "BE", "OR" and "LATER" in the second line, and "ASSERTION",

"THE", "IN", "PAPER" and "WILL" in the third line are shifted up by 1 pixel.



(a) Original Image



(b) Watermarked Image

(c) Example of Vertical Character Shifting

Figure 4.11: Vertical Upper-case Word-shift Coding.

In Figure 4.11(b), the word "WATERMARK" in the first line shows the character shifting effect. The characters 'A', 'E', 'M' and 'R' are shifted up by 1-pixel. The zoom-in image is shown in Figure 4.11(c). For lower-case characters, similar results are shown in Figure 4.12 and Figure 4.13.

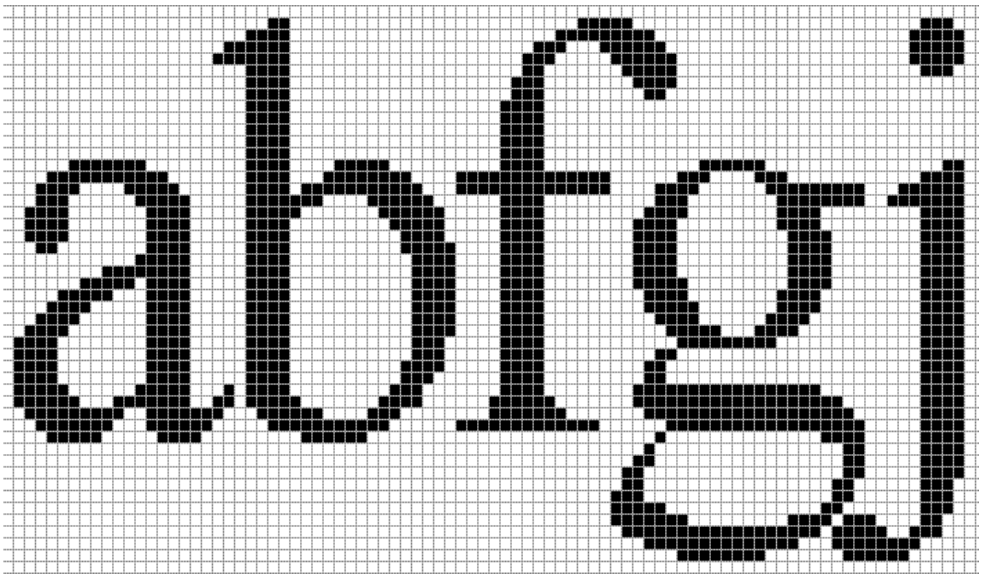


Figure 4.12: Lower-case Characters Alignment.

It can be seen that for vertical word shifting (Figure 4.11(b) and 4.13(b)), the modification is still difficult to be detected by human eyes; while it is much more obvious for the changes of vertical character shifting (Figure 4.11(c) and 4.13(c)). The watermarked image is somehow ugly. It is not comfortable to read this kind of document. For example, originally for the word "WATERMARK", there is 1-pixel difference at the bottom between 'W' and 'A'. The tops and bottoms of the characters form two horizontal logical baselines. After shifting up 'A' by 1 pixel, the difference is now 2 pixels, and the baselines are broken as shown in Figure 4.11(c). In this case, it will be very obvious for human eyes' perception. In order to apply vertical word/character-shift coding to embed data, further study needs to be done.

A watermark is a signal added to digital data (audio, video and still images) that can be detected or extracted later to make an assertion about the data. In this paper, we will be concerned only with still images. Embedding a watermark in an image means inserting information in the image, such that the image quality does not deteriorate significantly.

(a) Original Image

A watermark is a signal added to digital data (audio, video and still images) that can be detected or extracted later to make an assertion about the data. In this paper, we will be concerned only with still images. Embedding a watermark in an image means inserting information in the image, such that the image quality does not deteriorate significantly.

(b) Watermarked Image

watermark

(c) Example of Vertical Character (lower-case) Shifting

Figure 4.13: Vertical Lower-case Word-shift Coding.

4.4 Summary

In this chapter, a pixel-based reversible watermarking approach is proposed. The outer boundary of each letter are unchanged to keep high visual quality of the watermarked image. The 8 features of alphabet 'A' (300dpi) are studied, and the effects of changing these features are compared. It is difficult for naked eyes to find the differences between the modified 'A's. The first appeared letters in the document are chosen as the reference, which are to be used to reconstruct the original image during the watermark extraction process. Non-reference letters

are used to embed data. The embedding capacity is very high and sufficient for many applications. The visual quality is very good. For a lower resolution (150dpi) document image, fewer features for each letter (5 for 'A') are used to embed data. While at the same time, there are more separated letters available for embedding data. Thus the watermarking capacity is reduced but not as low as expected. The distortion of the watermarked image is still low.

We have tested the vertical word/character-shift coding, and study the HVS distortion of the result image. For vertical word shifting, the modification is difficult to be detected by human eyes. For vertical character shifting, the distortion is high. In order to apply vertical word/character-shift coding to embed data, further study needs to be done.

Chapter 5

Experimental Results and Discussions

In this chapter, the experimental results for parametric-based approach and pixel-based approach are presented. The performance comparisons (in terms of visual quality and embedding capacity) among several watermarking methods for binary document images are studied and discussed. A secured document authentication system using our proposed reversible watermarking algorithm is presented with showing some experimental results.

5.1 Performance Comparisons

Comparisons are made between our proposed methods and some other algorithms for binary document images: Line-shift coding, Word-shift coding, Character-shift coding (two methods are used for the comparisons) and DRDM method.

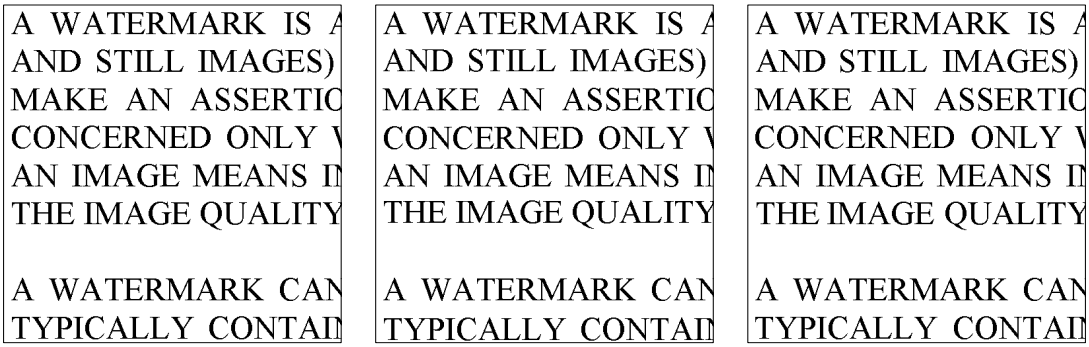
The watermarking capacity (for 2650 × 1820-pixel test image) comparisons are shown in Table 5.1 in ascending order, and the visual quality performance results are shown in Figure 5.1. Table 5.2 shows the statistics of the whole test document.

Table 5.1: The Performance Result of Watermarking Methods.

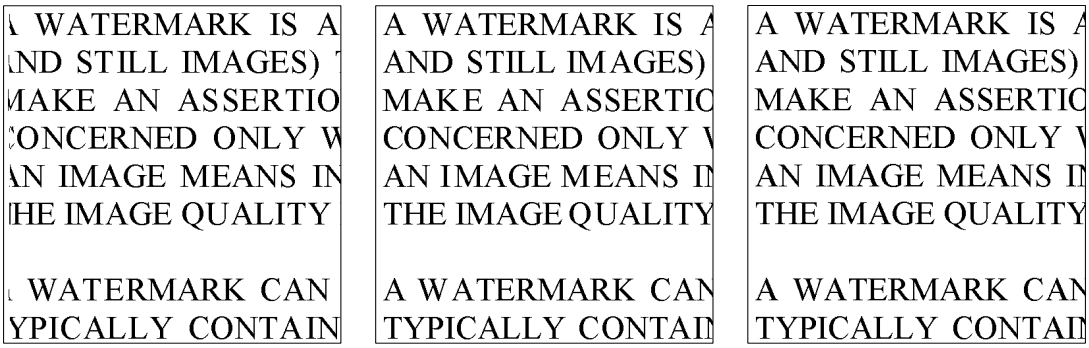
Watermarking Methods	Capacity (bits)	Visual Quality	Reversible Watermarking
Line-shift	20	High	No
Word-shift	351	High	No
Character-shift			
By Yang et al.	739	Medium	No
By Chotikakamthorn et al.	984	Medium	No
Parametric Approach	4,185/249('A')	High	Yes
Pixel-based Approach	11,160/664('A')	High	Yes
DRDM (16 × 16 block)	12,150	Low	No
(8 × 8 block)	32,173	Low	No

Table 5.2: The Statistics of the Sample Document.

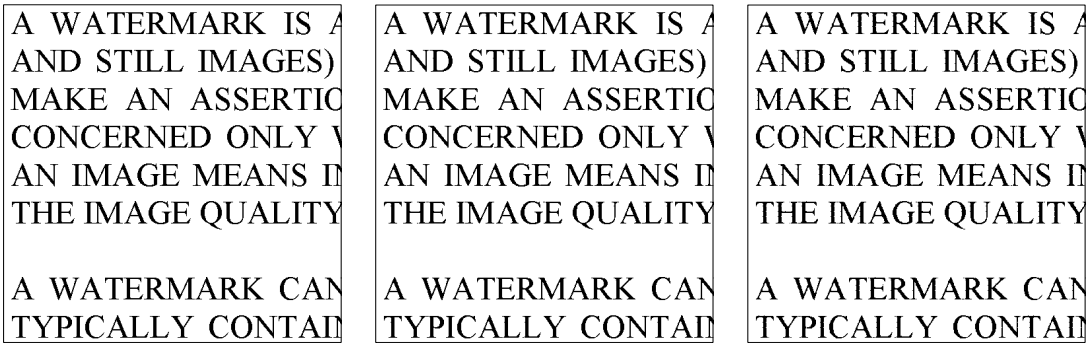
Type	Number
Paragraph	6
Text Line	41
Word	351
Letter	1923
Space Character	346
Segmented Character	1,421



(a) Original Image (b) Line-Shifting (c) Word-Shifting



(d) Character-Shifting 1 (e) Character-Shifting 2 (f) Parametric Approach



(g) Pixel-based Approach (h) DRDM(16×16 block) (i) DRDM(8×8 block)

Figure 5.1: Watermarked Images (512×512 pixels) Comparison.

5.1.1 Line-shift Coding

Line-shift coding method has a very low embedding capacity and low HVS distortion. As shown in Table 5.2, there are 41 lines in the whole document. Only the even-number lines are shifted: line 1, 3, 5... are not moved; line 2, 4, 6... are moved up or down. For this test document image with 300dpi resolution, it is possible to shift the line 2 pixels up or down while maintain the high visual quality. If there is only 1-pixel shifting for each line, it can only host 1 bit per line, and the total capacity is 20 bits. If it is allowed to shift 2 pixels, then 2-bit data can be embedded to each line, and the total capacity is 40 bits. For such a low embedding capacity, it is not suitable for most applications.

In Figure 5.1(b), line 2 is shifted up by 1 pixel, and line 4 is shifted 1 pixel down; line 6 is moved up by 2 pixels, and line 8 is 2 pixels down. Compared with the original image as shown in Figure 5.1(a), it can be seen that human eyes are not sensitive to those changes, therefore the visual quality is high.

5.1.2 Word-shift Coding

There are 351 words in the whole document. Using the method proposed by Brassil et al. [18], if 1 bit is embedded to each word, the embedding capacity will be 351 bits. This capacity is much higher than the line-shift coding method, but it is still not high enough to embed the 1024-bit encrypted hashing value in many applications. The visual quality of the watermarked image is very good as shown in Figure 5.1(c). The shortcoming of this method is that, since the word

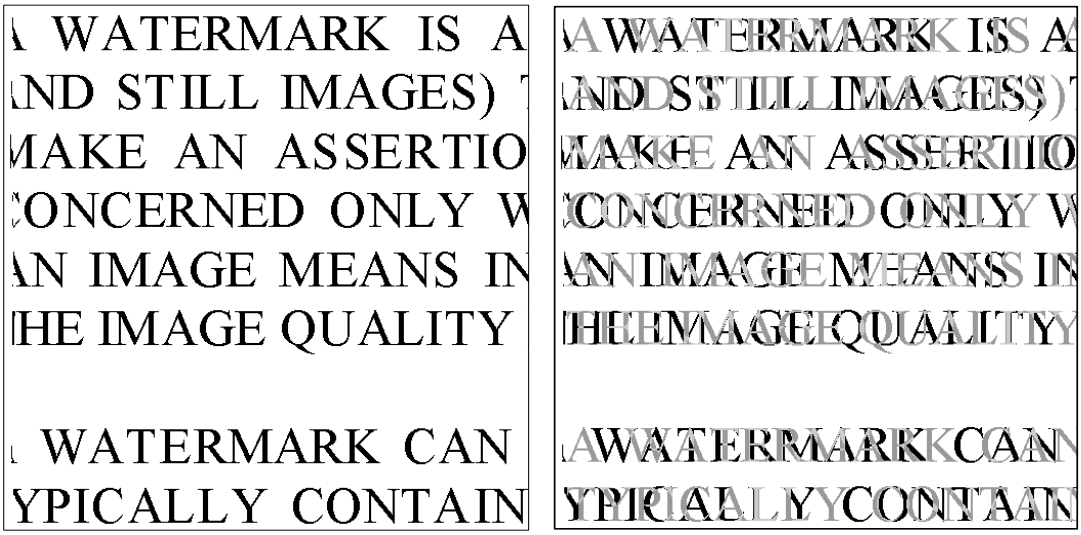
spacing in the original document is not uniform, detecting a word displacement requires knowledge of the original word spacing. Hence, the word positions in the unmarked document must be known in order to extract the hidden information.

5.1.3 Character-shift Coding

Two methods are used in our experiment: Chotikakamthorn's method [62] and Yang's method [17]. Their capacities for the test document image are 984 bits and 739 bits respectively, around 3 times as high as the capacity of word-shift coding method. Compared to the line/word-shift coding, the visual quality of character shifting is lower, as shown more clearly in Figure 5.2.

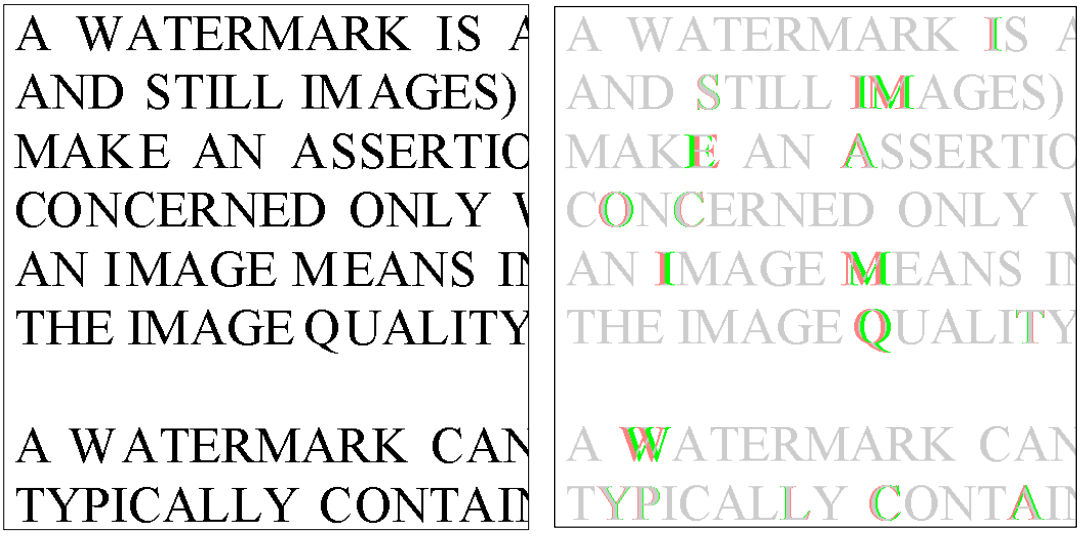
The result by using Chotikakamthorn's method is shown in Figure 5.2(a) and its difference image is shown in Figure 5.2(b). In the difference image, the original letters are in the lighter gray color and the darker ones are shifted letters. It can be seen from the results that the letters are all shifted a lot. If a document does not have enough margin, some letters are shifted outside.

The enlarged watermarked image by Yang's method is shown in Figure 5.2(c), and Figure 5.2(d) shows the difference image. It can be seen that some characters are shifted 1 pixel to the left or right, like 'S' in word "STILL" at line 2. Some characters are shifted by 2 or more pixels. For example, in the word "WATERMARK" at the next last line, 'W' is shifted 5 pixels to the left. The word "IMAGE" both appears in the fifth and sixth lines. Comparing these two words carefully, we can see that in the word at the fifth line, the space between 'I' and 'M' is much wider than the one at the sixth line.



(a) Chotikakamthorn's Method

(b) Difference Image



(c) Yang's Method

(d) Difference Image

Figure 5.2: Performance of Character-shift Coding (512 × 512 pixels).

5.1.4 Parametric Approach and Pixel-based Approach

The results of the Parametric and pixel-based watermarking approaches are shown in Figure 5.1(f) and Figure 5.1(g). It can be seen that the visual quality of these

two watermarked images are very good. In this image, only alphabet 'A's are used to embed data. For the whole document, there are 176 'A's, in which 84 are not connected to other characters. Set one 'A' as the reference with no modification on it. For parametric approach, each 'A' can host 3-bit data, therefore the total embedding capacity is $83 \times 3 = 249$ bits. For the complete system, if we assume that the average embedding rate is 3 bits per letter, the watermarking capacity will be much higher. There are 1,421 segmented letters, among which 26 are set as references. Therefore there are $1,421 - 26 = 1,395$ letters that can be used to embed data. The total capacity will be $1,395 \times 3 = 4,185$ bits.

For pixel-based approach, each 'A' can host 8-bit data. The total embedding capacity is $83 \times 8 = 664$ bits. For the complete system, if we assume each other letters can also host 8 bits, the capacity will be $1,395 \times 8 = 11,160$ bits. For these two approaches, their capacities are sufficient for most document authentication applications hosting 1,024-bit encrypted hash value.

Using the reference letters, the original document images of the proposed approaches can be reconstructed. Therefore, both of the parametric and pixel-based approaches are reversible watermarking methods while the others are non-reversible. Reversibility enlarges the application scope of the watermarking system.

5.1.5 DRDM Watermarking

Lu et al. [67, 68] proposed this algorithm to calculate the DRDM value for each pixel in the whole block and chose the best pixel to embed data. If we use the

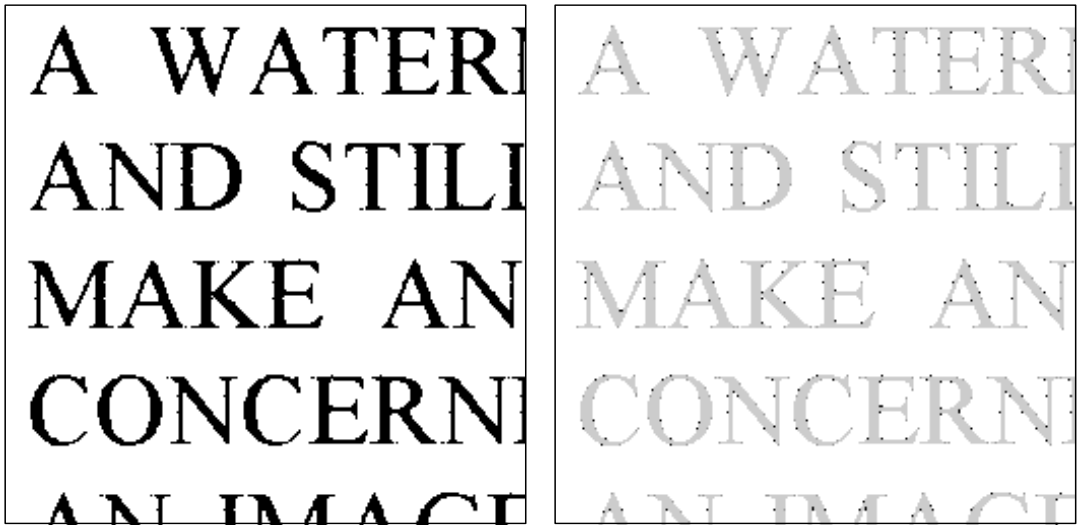
block of size 8×8 pixels, 32,173 bits can be embedded in the test image, and the result image is shown in Figure 5.1(i). Figure 5.3(a) shows the enlarged watermarked image and the difference image is shown in Figure 5.3(b). The dots in the difference image indicate the positions where pixels are flipped. There are 2 types of dots: the darker dot means that this pixel is flipped from white to black, and the lighter one indicates that the pixel is flipped from black to white.

The visual quality of the image is still not very good, as shown in Figure 5.3(a). For example, in the word 'WATERMARK' at the first line, for the vertical stem of the character 'E', there are 3 black dots on the left edge and 2 white dots on the right edge. The serif of 'T' is also broken. This kind of pixel flipping is quite noticeable, which should be avoided to maintain low HVS distortion.

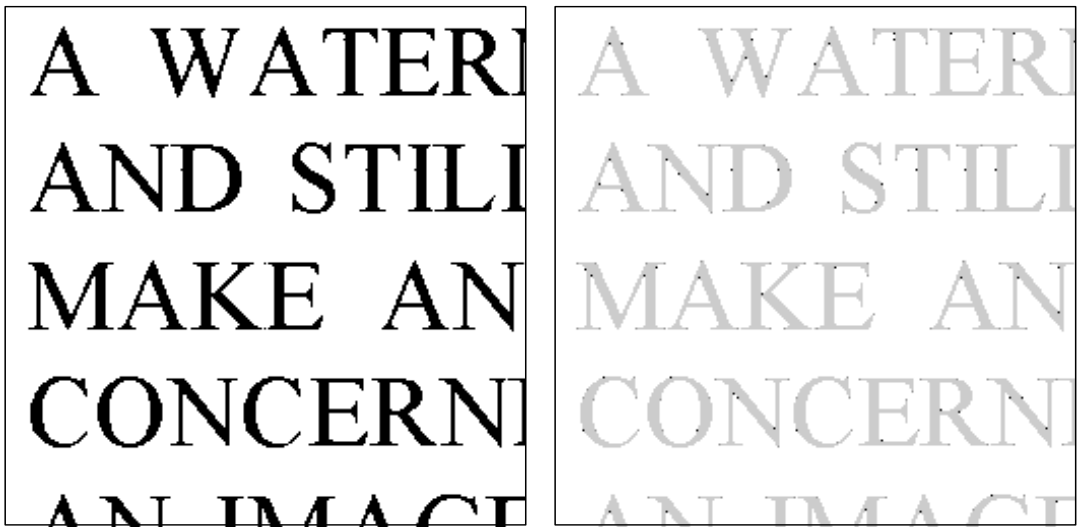
There is a trade-off between the visual quality and embedding capacity. The visual quality can be improved by decreasing the capacity. We now choose the block size of 16×16 pixels to do the watermarking, and the embedding capacity is decreased to 12,150 bits, which is still very high. Compared Figure 5.1(h) to Figure 5.1(i), it can be seen that the visual quality for the watermarked image is much higher than the one using 8×8-pixel blocks. As shown more clearly in Figure 5.3(c) and Figure 5.3(d), for the same character 'E' in the word"WATERMARK", there is no obvious changes on the stem. But in "STILL" at the second line, there are still noticeable flipped pixels on the stems of 'T', 'I' and 'L'.

This DRDM method is not specified for document images, and it can be used for many other binary images like halftone images. Therefore, it is reasonable

that the visual quality is not as good as the other methods which are only for binary document images.



(a) Watermarked Image (8×8 Block). (b) Difference Image (8×8 Block).



(c) Watermarked Image (16×16 Block). (d) Difference Image (16×16 Block).

Figure 5.3: Performance of DRDM Watermarking (256 $\times$256 pixels).

5.2 Discussions

According to [67, 68], there is still no objective measure which is suitable enough for binary images, especially for binary text documents. In many situations, objective measure does not agree with the subjective one. Therefore, subjective measure (human visual perception) is used in this thesis. For the visual quality of the watermarked images, line-shift coding, word-shift coding and our proposed methods achieve high scores. It is difficult for human eyes to detect the modifications in these images. Character-shift coding methods have acceptable visual quality, but they introduce some changes obvious for human eyes. The performance of DRDM methods is acceptable but not as high as that of other methods. The visual quality can be improved by decreasing the embedding capacity (using bigger block size to embed data). Compared with other methods , the visual quality of the proposed methods is higher. The reasons are shown below:

- Methods [65, 66] are for general binary images; our proposed methods are only consider text documents.
- Methods [65, 66] consider noisy images; our proposed methods are for noise-free image.
- Methods [65, 66] dynamically choose pixels to embed data; our proposed methods statically choose best positions of each character.

Therefore, methods [65, 66] have wider applications while our proposed methods have better visual quality.

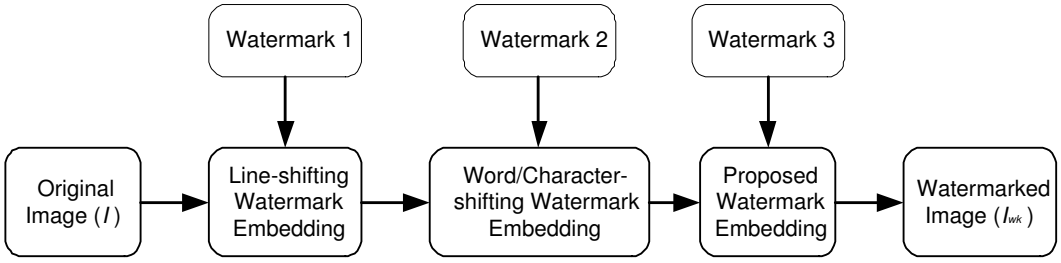
From the embedding capacity point of view, the performance of line-shift and word-shift coding methods are very low. The character-shift coding methods achieve medium score, but their capacity is not high enough for our authentication application. DRDM method has the highest capacity among these algorithms. Parametric and Pixel-based approaches have embedding capacities resides between the character-shift coding and DRDM methods.

Line-shifting, word-shifting and character-shifting methods are simple and easy to be implemented. Their embedding and detection processes are very fast. DRDM method has high computation load. For our proposed algorithms, OCR process is involved which makes the system complicated. Parametric approach needs character segmentation, boundary extraction, contour following and contour segmentation, whose computational complexity is rather high. The pixel-based approach is thus proposed to reduce the complexity. It does not need contour following and contour segmentation. The algorithms are limited to those documents with known font type and font size. The feature points of each characters must be available at the embedding and detection processes, which limits the application of the proposed algorithm. A codebook may be used to solve this problem.

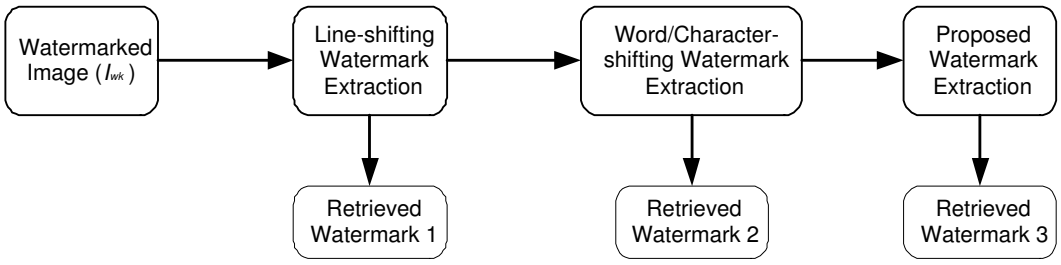
On the whole, these binary document watermarking algorithms have their own advantages and disadvantages. It depends on the requirements of real applications to decide which watermarking algorithm is more suitable.

In some cases, more than one watermarking algorithms can be used. As shown in Figure 5.4, line-shifting, word/character-shifting techniques and the proposed

watermarking methods can be used together. Watermarks 1, 2 and 3 can be embedded and extracted separately, and the total watermarking capacity is the sum of the three capacities. The condition is that each of these methods does not affect the others, otherwise there will be problems in watermark extraction process. For example, the proposed methods and the DRDM method cannot be used together, since the DRDM method may change the letter features which affect the detection rate of the proposed methods.



(a) Combined Watermark Embedding Scheme



(b) Combined Watermark Extraction Scheme

Figure 5.4: Combined Watermark Embedding/Extraction Schemes.

5.3 Secured Authentication System

Based on our proposed reversible watermarking algorithm, we present a secured document authentication system, which consists of two schemes: the secured watermark embedding scheme and the secured watermark authentication scheme.

5.3.1 Secured Watermark Embedding Scheme

For grayscale images, cryptography-based authentication watermarking schemes [6, 7, 22], the least significant bits (LSBs) of all pixels are cleared first, and then the hash value is calculated. For binary images, Kim et al. [25] proposed using a generator with a known seed to generate a set of pseudo-random locations L within the image for a secured authentication watermarking. All the pixels belonging to L are cleared first to generate the hash value.

For these authentication systems, the host image can be divided into two parts: the first part is used to generate the hash value; the second part is to embed watermark (usually hash value). By doing so, the hash value can only be used to authenticate the first part of the image instead of the whole content. Therefore, reversible watermarking methods should be used to solve this problem. The hash value is calculated based on the whole contents of the document, and is then embedded into the image. At the receiver end, the original image can be reconstructed to generate the hash value again. In this way, the authentication of the whole document can be ensured.

The watermark embedding scheme of the secured authentication system is

shown in Figure 5.5. Hash function (MD5 algorithm) is used to generate a message digest for authentication purpose, which is then protected by the encryption function (3DES algorithm).

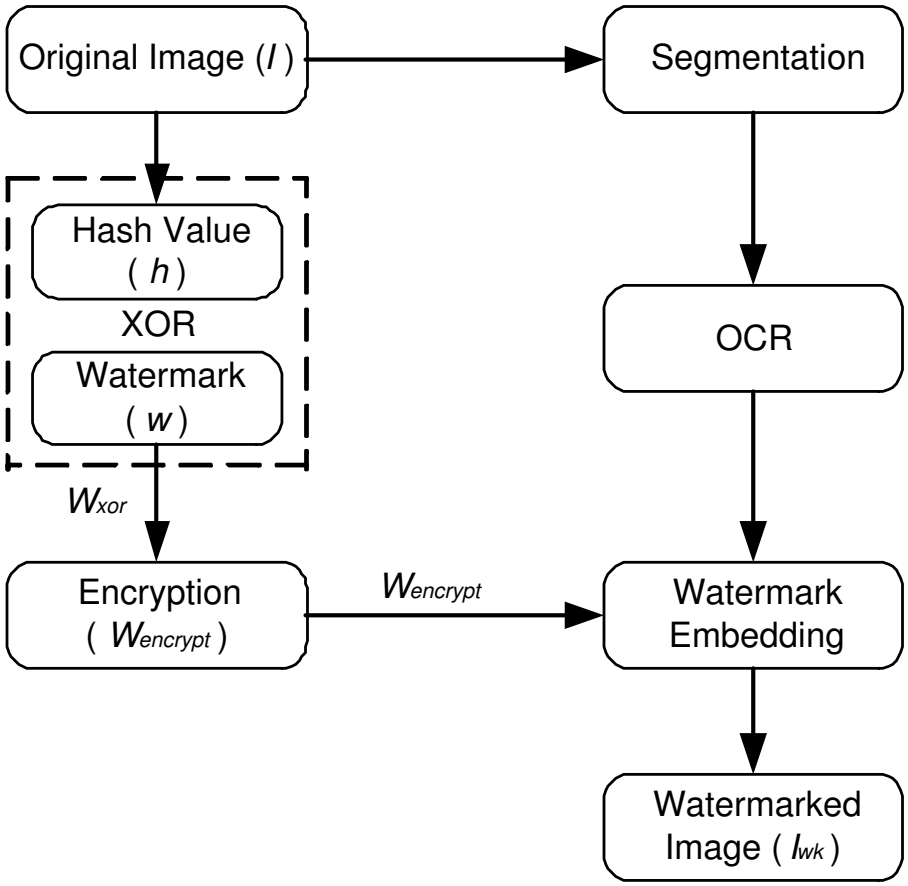


Figure 5.5: Secured Watermark Embedding Scheme.

Steps of the secured watermark embedding scheme are as follows:

- Step 1: $h = H(I)$. To calculate hash value (h) of the original image I, where H is the hash function (MD5 algorithm is used in our experiment).
- Step 2: $W_{xor} = w \oplus h$. To apply *exclusive or (XOR)* operation to the watermark w and hash value h.

- Step 3: $W_{encrypt} = E(W_{xor}, k)$. To protect W_{xor} with encryption function E (3DES algorithm) and a secret key.
- Step 4: Embed $W_{encrypt}$ into the original image by using proposed methods.

5.3.2 Secured Watermark Authentication Scheme

At the receiver side, the test image is to be authenticated. The secured watermark authentication scheme is shown in Figure 5.6.

To authenticate the document, we proceed as follows:

- Step 1: Extract the embedded data $W_{retrieve}$.
- Step 2: Reconstruct the original document image (I') by replacing the non-reference letters with their respective reference letters.
- Step 3: $h' = H(I')$. Re-generate the message digest (h') by using the same hash function $H(I')$.
- Step 4: $W'_{xor} = w \oplus h'$. To apply XOR operation again for h' and the applied watermark w.
- Step 5: $W'_{encrypt} = E(W'_{xor}, k)$. To encrypt W'_{xor} using the same encryption function and the same secret key (k). $W'_{encrypt}$ is the theoretical data that should be embedded.
- Step 6: Compare $W'_{encrypt}$ and $W_{retrieve}$. If they are identical, the suspect image is authenticated, and it is a true copy. Otherwise, if they not same, it means that the image is forged, or it has been modified.

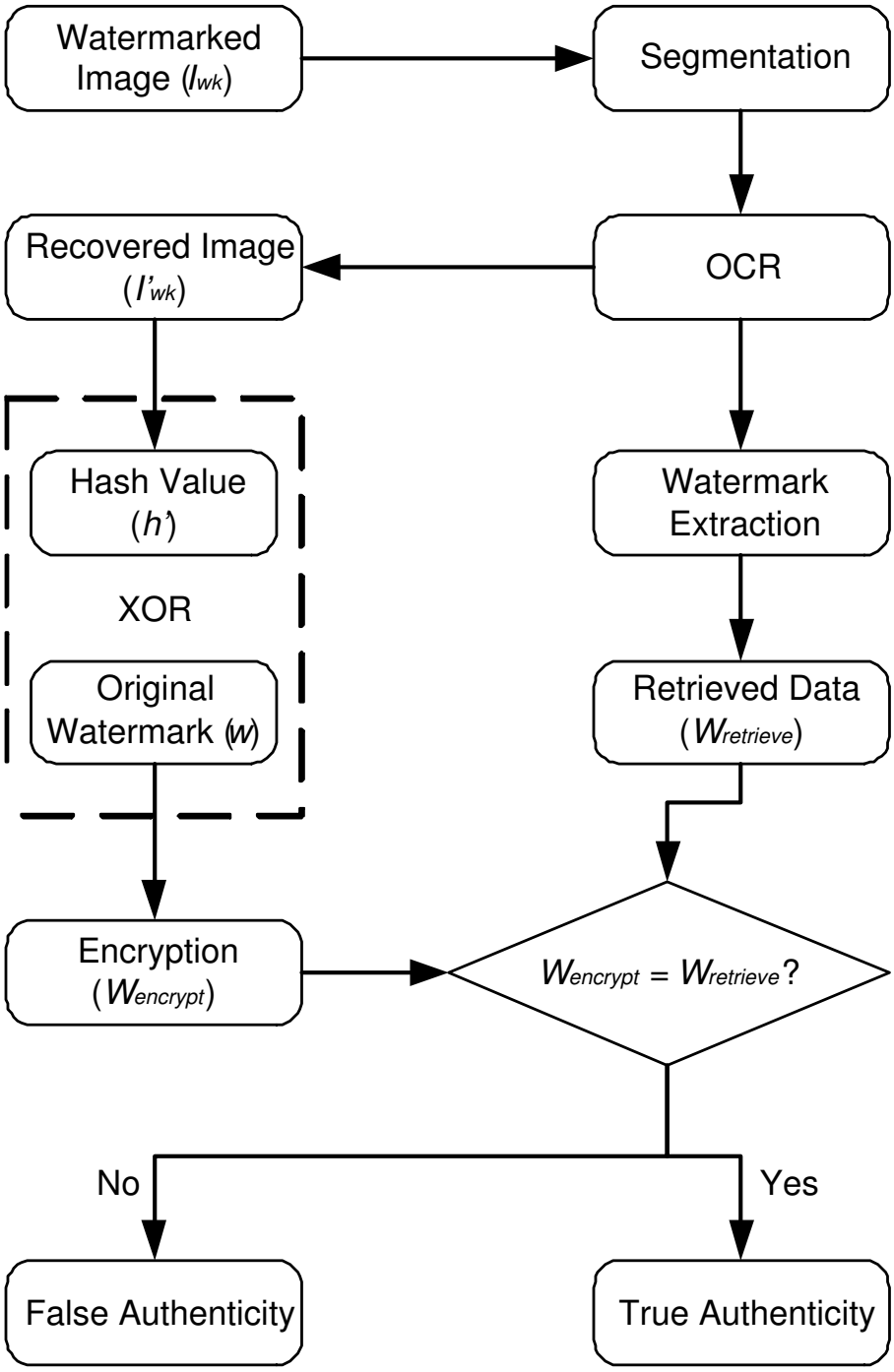


Figure 5.6: Secured Watermark Authentication Scheme.

5.3.3 Authentication Functions

For authentication systems, there must be an authenticator: a value to be used to authenticate a message. Authentication function algorithms have much in common with techniques used in encryption, but to a different end. One difference is that message digest algorithms need not to be reversible, as it must for decryption. According to William Stallings [4], these functions may be grouped into three classes :

- Message encryption

  The ciphertext of the entire message serves as its authenticator. Obviously the size is too big to be embedded for our system.

- Cryptographic checksum

  A fixed-length value is produced by a secret key and a public function of the message. This fixed-length value serves as the authenticator. The checksum also called message authentication code (MAC), which is appended to the message.

- Hash function

  A public function that maps a message of any length into a fixed-length hash value, which serves as the authenticator. h = H(M), where M is a variable-length message, and H(M) is the fixed-length hash value, also called a message digest.

In this authentication system, we choose 128-bit MD5 algorithm, the most commonly used message digest algorithm. For more information on MD5, please

refer to [81]. The pixel values ('0's and '1's) of the full-size image ($1820 \times 2650 = 4823000$) form a 4823000-bit long data stream. MD5 takes this stream as the input, and computes the message digest. This computation takes only a few milliseconds in C environment. If we randomly flip one pixel of the original image, the new message digest is very different from the original one, as shown in Table 5.3.

Table 5.3: The Results of Hash Function (MD5).

Input (Original Image)	Message Digest (HEX)
4823000-bit data stream	0c9f6c2191502ef7302c266b90507736
1-bit difference to above stream	054af7345dff7578d20744255c7bf29e

Because the hash function itself is not considered to be secret, some means (encryption function) is required to protect the hash value.

5.3.4 Encryption Function

It is agreed by many that it is not secure enough by using watermarking algorithm only. Recently many proposed techniques [25] combine watermarking and cryptography to achieve more secured systems.

Cryptography techniques can be *symmetric* or *asymmetric* [4], as shown below:

- Symmetric Cryptography

  The same algorithm with the same key is used for encryption and decryption. It is also called *conventional cryptography*. International data encryp-

tion algorithm (IDEA) and the data encryption standard (DES) are two examples. The triple DES (3DES) algorithm just repeats three time of the DES encryption process.

- Asymmetric Cryptography

  Same algorithm is used for encryption and decryption, but with a pair of keys: one for encryption and the other for decryption. For the two keys, one is called *public* key, and the other is *private* key. It is also called *public-key cryptography* (e.g., RSA algorithm).

We use triple DES algorithm in our experiment. The watermark length need to be limited since the capacity is usually not high for binary document images. In this case, we randomly generate 128-bit stream w as the information data. Then we do the Exclusive OR (XOR) operation of h and w ($W_{xor} = h \oplus w$), and the result W_{xor} is then encrypted and inserted into the host image. W_{xor} has the size of 128 bytes, or 1024 bits. The encryption results are shown in Table 5.4.

Table 5.4: The Results of Symmetric Encryption (3DES).

Message Digest h	0c9f6c2191502ef7302c266b90507736
Encryption Key k	"Authenticate Key"
128-bit watermark w	2923be84e16cd6ae529049f1f1bbe9eb
XOR Result W_{xor}	25bcd2a5703cf85962bc6f9a61eb9edd

5.4 Summary

In this chapter, the experimental results for parametric-based approach and pixel-based approach are presented. The comparisons among several watermarking methods for binary document images indicate satisfactory performance of our proposed algorithm in terms of visual quality and embedding capacity. The proposed algorithm can be used together with some other watermarking algorithms to further increase the embedding capacity. Moreover, for the proposed algorithm, the original document image can be reconstructed during the watermark extraction process (reversible watermarking), which enlarges the application scope of the document/watermarking system.

A secured document authentication system using our proposed reversible watermarking algorithm is presented. For many existing authentication systems, only a part of the image instead of the whole content is protected/authenticated. This problem can be solved by using reversible watermarking algorithm. MD5 algorithm (128-bit hash value) is used for authenticity purpose and 3DES technique is applied to encrypt the hash value (1024 bits after encryption). Some experimental results of MD5 and 3DES are shown as well.

Chapter 6

Conclusions and Recommendations

6.1 Conclusions

The technology of the digital watermarking has been developing rapidly, and the area for binary document authentication has gained more and more research interest. In this thesis, we proposed a fragile reversible watermarking algorithm for binary document images in spatial domain.

Compared to gray-level image watermarking, binary document image watermarking is a much more challenging task. The existing algorithms for document images either have low embedding capacity, or can cause high distortion in human visual perception. Moreover, most of them are non-reversible watermarking algorithms, which limits the application scope of the document/watermarking system.

Being aware of this situation, we proposed a fragile reversible watermarking algorithm with good visual quality and reasonably high embedding capacity. We studied how TrueType font and METAFONT systems design/draw nice-looking letters. Based on these fundamentals, a set of equations is used to describe letter outlines. By adjusting some parameters of these outline equations, we are able to produce the letter in many different shapes, which are used to replace the original alphabets for embedding data. In order for the watermarked image to have low distortion for HVS perception, we kept the widths and heights of alphabets unchanged. A pixel-based watermarking approach was also proposed and implemented to simplify the algorithm. Both of the proposed approaches are reversible watermarking methods. The comparisons with some other methods for binary document watermarking indicate satisfactory performance of the proposed methods.

From the above studies, we may conclude that some problems have been solved for binary document watermarking and for reversible watermarking. However, in order to use the proposed methods in real applications, some other scenarios need to be considered.

6.2 Recommendations for Further Research

Recommendations to further enhance the current watermarking methods, together with the future research directions in binary document watermarking and in reversible watermarking, are briefly discussed below.

- TrueType font and METAFONT systems need to be further studied to find better ways to describe the alphabets and control their shapes. It seems that METAFONT system is more suitable to describe alphabets in parameters. It may be possible to add the watermarking function to some LaTeX compiler. Some information can be embedded into the pdf/image file after file compiling.

- For alphabet 'A', the outline equations can be further studied and improved. The quantization and smoothing techniques can be applied to further decrease the HVS distortion for document images. For image segmentation process, the connected letters may also be used by new segmentation techniques to further increase the embedding capacity.

- More attention needs to be paid on the curves in the outline equations to further increase the embedding capacity and decrease the HVS distortion. Human eyes are less sensitive to the modifications in curves than in straight lines. However, the sizes of letters are usually small in document images. Therefore, it will be difficult to detect the modifications in curves due to the quantization error.

- In this thesis, we only studied the alphabet 'A' in font type of Times New Roman and font size of 12 points. More letters (uppercase, lowercase, and other font types/sizes) need to be studied and implemented to complete this watermarking algorithm.

- To increase the watermarking capacity, vertical character-shift coding method for document images need to be further studied and implemented. Special

rules must be applied to shift different characters vertically to keep the HVS distortion low.

- For secured document authentication systems, further research needs to be done on reversible watermarking algorithms. Semi-fragile watermarking is another possible research direction for binary document images.

- It is valuable to further study the human visual system for binary documents, and find more suitable distortion measure for binary document watermarking.

Bibliography

[1] D. Kundur, "Multiresolution digital watermarking: Algorithms and implications for multimedia signals," Ph.D. dissertation, University of Toronto, Canada, Aug. 1999. [Online]. Available: http://ee.tamu.edu/∼deepa/pdf/phd99.pdf

[2] D. E. Knuth, "Digital typography," Stanford, Calif. : CSLI Publications, c1999.

[3] Signal Processing Forum, "What is the future for watermarking? (Part 1)," *IEEE Signal Processing Mag.*, pp. 55-59, Sept. 2003.

[4] W. Stallings, "Network and internetwork security: principles and practice," Prentice-Hall Inc., 1995.

[5] J. Shen, "Development OF Digital Rights Management System for Satellite Images," M.Eng. dissertation, Nanyang Technological University, Singapore, 2003.

[6] P. W. Wong, "A public key watermark for image verification and authentication," *IEEE Int. Conf. Image Processing*, vol. 1, pp. 455-459, 1998.

[7] P. S. L. M. Barreto, H. Y. Kim and V. Rijmen, "Toward a secure public-key blockwise fragile authentication watermarking," *IEEE Proc. Vision, Image and Signal Processing*, vol. 149, no. 2, pp. 57-62, 2002.

[8] S. Katzenbeisser and A. P. F. Petitcolas, "Information hiding techniques for steganography and digital watermarking," Artech House, 2000.

[9] C. Podilchuk and W. Zeng, "Image-adaptive watermarking using visual models," *IEEE J. Select. Areas Commun*, vol. 16, pp. 525-539, May 1998.

[10] R. B. Wolfgang, C. I. Podilchuk, and E. J. Delp, "Perceptual watermarks for digital images," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1108-1126, July 1999.

[11] N. Memon and P. W. Wong,"Protecting digital media content," *Commun. of the ACM*, vol. 41, pp. 35-43, July 1998.

[12] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding - a survey", *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1062-1078, July 1999.

[13] F. Hartung and M. Kutter, "Multimedia watermarking techniques," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1079-1107, July 1999.

[14] G. W. Braudaway, K. A. Magerlein, and F. Mintzer, "Protecting publicly available images with a visible image watermark", *Proc. SPIE, Optical Security and Counterfeit Deterrence Techniques (R. L. van Renesse, ed.)*, vol. 2659, pp. 126-133, Feb. 1996.

[15] M. S. Kankanhalli, Rajmohan and K. R. Ramakrishnan, "Adaptive visible watermarking of images," *IEEE Proc. of ICMCS'99, Florence, Italy*, vol. 1, pp. 568-573, June 1999.

[16] S. P. Mohanty, K. R. Ramakrishnan, and M. S. Kankanhalli, "An adaptive DCT domain visible watermarking technique for protection of publicly available images," *ICMPS'2000*, pp. 195-198, 2000. [Online]. Available: http://www.csee.usf.edu/ smohanty/research/ConfPapers/ ICMPS2000PaperMohanty.pdf

[17] H. Yang and A. C. Kot, "Text document authentication by integrating inter character and word spaces watermarking," ICME'2004, Taiwan, June 2004.

[18] J. T. Brassil, S. Low, N. Maxemchuk, and L. O'Gorman, "Watermarking document images with bounding box expansion," *Information hiding: first international workshop, Lecture Notes in CS, Springer*, vol. 1174, pp. 227-236, 1996.

[19] J. T. Brassil, S. Low, N. F. Maxemchuk, and L. O'Gorman, "Electronic marking and identification techniques to discourage document copying", *IEEE Journal on Selected Areas in Commun.* vol. 13, no. 8, Oct. 1995.

[20] J. T. Brassil, S. Low, and N. F. Maxemchuk, 'Copyright protection for the Electronic Distribution of Text Documents', *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1181-1196, July 1999.

[21] J. Zhao and E. Koch, "Embedding robust labels into images for copyright protection," *Proc. International Congress on Intellectual Property Rights for*

Specialized Information, Knowledge and New Technologies, pp. 242-251, Aug. 1995.

[22] M. Holliman, and N. Memon, "Counterfeiting attacks on oblivious block-wise independent invisible watermarking schemes," *IEEE Trans. Image Processing*, vol. 9, no. 3, pp. 432-441, 2000.

[23] F. A. P. Petitcolas, R. J. Anderson and M. G. Kuhn. "Attacks on copyright marking systems," *Information hiding, second international workshop, IH98*, Portland, Oregon, U.S.A., LNCS 1525, Springer-Verlag, ISBN 3-540-65386-4, pp. 219-239, Apr. 1998,

[24] F. A. P. Petitcolas. "Watermarking schemes evaluation," *IEEE. Signal Processing*, vol. 17, no. 5, pp. 58-64, Sept. 2000.

[25] H. Y. Kim and A. Afif, "Secure authentication watermarking for binary images," *IEEE, SIBGRAPI'03*, pp. 199-206, Oct. 2003.

[26] Y. Yuan and C. T. Li, "Fragile watermarking scheme exploiting non-deterministic block-wise dependency," *Proc. IEEE, ICPR'04*, Cambridge, UK, Aug. 2004. [Online]. Available:
http://www.dcs.warwick.ac.uk/people/research/Yinyin.Yuan
/ICPR04Fragile.pdf

[27] E. T. Lin and E. J. Delp, "A review of fragile image watermarks," *ACM Multimedia '99, Multimedia Contents*, Orlando, FL, pp. 25-29, Oct. 1999. [Online]. Available:
http://shay.ecn.purdue.edu/ linet/Papers/ACM-1999.PDF

[28] H. Lu, A. C. Kot and J. Cheng, "Secure data hiding in binary document images for authentication," *Proceedings of the 2003 IEEE International Symposium on Circuits and Systems (ISCAS 2003)*, Bangkok, Thailand, vol. 3, pp. 806-809, May 2003 (invited paper).

[29] H. Lu, A. C. Kot and R. Susanto, "Binary Image Watermarking through Biased Binarization," in Proceedings of the 2003 IEEE International Conference on Multimedia & Expo (ICME 2003), Baltimore, Maryland, July 2003.

[30] H. Lu, "Bi-level image watermarking and distortion measure," M.Eng. dissertation, Nanyang Technological University, Singapore, July 2003.

[31] S. H. El-Din and M. Moniri, "Fragile and semi-fragile image authentication based on image self-similarity El-Din," *IEEE, ICIP'2002*, vol. 2, 22-25 pp. II-897 - II-900, Sept. 2002.

[32] Zhe-Ming Lu, Dian-Guo Xu, Chun-He Liu, Sheng-He Sun, "Semi-fragile image watermarking method based on index constrained vector quantisation," *Electronics Letters*, vol. 39, issue: 1, pp. 35-36, Jan. 2003.

[33] Qibin Sun, Shih-Fu Chang, K. Maeno and M. Suto, "A new semi-fragile image authentication framework combining ECC and PKI infrastructures," *IEEE, ISCAS'2002*, vol.2, pp. II-440 - II-443, May 2002.

[34] Qibin Sun, and Shih-Fu Chang, "Semi-fragile image authentication using generic wavelet domain features and ECC," *IEEE, ICIP'2002*, vol. 2, pp. II-901 - II-904, Sept. 2002.

[35] H. H. Ko and S. J. Park, "Semi-fragile watermarking for telltale tamper proofing and authenticating," *ITC-CSCC* 2002. [Online]. Available: http://www.kmutt.ac.th/itc2002/CD/pdf/17_07_45/WP2_OD/2.pdf

[36] Y. Zhao, "Dual domain semi-fragile watermarking for image authentication", Master dissertation, University of Toronto, Canada, 2003. [Online]. Available: http://ee.tamu.edu/ deepa/theses/zhao03.pdf

[37] B. Tao, and B. Dickinson, "Adaptive watermarking in the DCT domain," *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, vol. 4, pp. 2985-2988, 1997.

[38] Y. Wang, J. F. Doherty and R. E. V. Dyck, "A wavelet-based watermarking algorithm for ownership verification of digital images," *IEEE Trans.on Image Processing*, vol. 11, issue. 2, pp. 77-88, Feb. 2002.

[39] M. Barni, F. Bartolini and A. Piva, "Multichannel watermarking of color images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, issue: 3, pp. 142-156, Mar. 2002.

[40] H. Guo, N. D. Georganas, "Multi-resolution image watermarking scheme in the spectrum domain," *IEEE CCECE 2002*. Canada, vol. 2, pp. 873-878, May 2002.

[41] H. Lu, X. Shi, Y. Q. Shi, A. C. Kot and L. Chen, "Watermark embedding in DC components of DCT for binary images," *Proceedings of the IEEE International Workshop on Multimedia Signal Processing (MMSP 2002)*, US Virgin Islands, pp. 300-303, Dec. 2002.

[42] C. De Vleeschouwer, J. F. Delaigle, and B. Macq, "Circular interpretation of bijective transformations in lossless watermarking for media asset management," *IEEE Transactions on Multimedia*, Vol. 5, No. 1, pp. 97-105, Mar. 2003.

[43] C. De Vleeschouwer, J. F. Delaigle, and B. Macq, "Circular interpretation of histogram for reversible watermarking," *Proc. IEEE 4thWorkshop Multimedia Signal Processing*, pp. 345-350, Oct. 2001.

[44] A. M. Alattar, "Reversible watermark using the difference expansion of a generalized integer transform," *IEEE Transactions on Image Processing*, Vol. 13, No. 8, pp. 1147-1156, Aug. 2004.

[45] A. M. Alattar, "Reversible watermark using the difference expansion of quads," *ICASSP*, pp. III-377–III-380, 2004.

[46] D. M. Thodi, and J. J. Rodriguez, "Reversible watermarking by prediction-error expansion," *IEEE Southwest Symp. on Image Analysis and Interpretation*, pp. 21-25, Mar 28-30, 2004.

[47] J. Fridrich, J. Goljan, R. Du, "Invertible Authentication," *Proceedings of SPIE, Security and Watermarking of Multimedia Content*, San Jose, Jan. 2001.

[48] J. Tian, "High capacity reversible data embedding and content authentication," *ICASSP*, Vol. 3, pp. III-517–III-520. Apr. 2003.

[49] Y. C. Tseng, Y. Y. Chen and H. K. Pan, "A secure data hiding scheme for binary images," *IEEE Transactions on Communications*, vol. 50, no. 8, pp. 1227-1231, Aug. 2002.

[50] Y. C. Tseng and H. K. Pan, "Data hiding in 2-color images," *IEEE Transactions on Computers*, vol. 51, no.7, pp. 873-880, July 2002.

[51] Y. C. Tseng and H. K. Pan,"Secure and invisible data hiding in 2-color images,"*Proc. IEEE INFOCOM 2001*, vol. 2, pp. 887-896, 2001.

[52] H. K. Pan, Y. Y. Chen, and Y. C. Tseng, "Secure and invisible data hiding in 2-color images,"*Proc. the 5th IEEE Symposium on Computers and Communications*, pp. 750-755, 2000.

[53] M. Chen, E. K. Wong, N. Memon, and S. Adams, "Recent developments in document image watermarking and data hiding," *Proc. SPIE Conference on Multimedia Systems and Applications IV*, vol. 4518, pp. 166-176, Aug. 2001.

[54] S. H. Low, N. F. Maxemchuk, and A. M. Lapone,"Document identification for copyright protection using centroid detection,"*IEEE Transactions on Communications*, vol. 46, no. 3, pp. 372-383, Mar. 1998.

[55] S. H. Low and N. F. Maxemchuk, "Performance comparison of two text marking methods," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 4, pp. 561-572, May 1998.

[56] N. F. Maxemchuk and S. H. Low, "Marking text documents," *Proc. IEEE International Conference on Image Processing*, vol. 3, pp. 26-29, Oct. 1997.

[57] S. H. Low, N. F. Maxemchuk, J. T. Brassil, and L. OGorman, "Document marking and identification using both line and word shifting," *Proc. IEEE INFOCOM'95*, vol. 2, pp. 853-860, Apr. 1995.

[58] S. H. Low, A. M. Lapone, and N. F. Maxmchuk, "Document identification to discourage illicit copying," *Proc. IEEE Global Telecommunications Conference*, vol. 2, pp. 1203-1208, Nov. 1995.

[59] D. Huang and H. Yan, "Interword distance changes represented by sine waves for watermarking text images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 12, pp. 1237-1245, Dec. 2001.

[60] Y. Liu, J. Mant, E. Wong, and S. H. Low, "Marking and detection of text documents using transform-domain techniques," *Proc. SPIE Conference on Security and Watermarking of Multimedia Contents III*, vol. 3657, pp. 317-328, Jan. 1999.

[61] I. J. Cox, J. Kilian, F. T. Leighton and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Transactions on Image Processing*, vol. 6, no.l2, Dec. 1997.

[62] N. Chotikakamthorn,"Document image data hiding techniques using character spacing width sequence coding," *Proc. IEEE International Conference on Image Processing*, vol. 32, pp. 250-254, 1999.

[63] J. T. Brassil, S. H. Low, N. F. Maxemchuk, and L. OGorman, "Marking text features of document images to deter illicit dissemination," *in Proc. the*

12th IAPR International Conference on Pattern Recognition, vol. 2, pp. 315-319, Oct. 1994.

[64] T. Amamo and D. Misaki,"A Feature calibration method for watermarking of document images," *Proc. the 5th International Conference on Document Analysis and Recognition*, pp. 91-94, Sept. 1999.

[65] Q. G. Mei, E. K. Wong, and N. D. Memon, "Data hiding in binary text documents," *Proc. of SPIE Conference on Security and Watermarking of Multimedia Contents III*, vol. 4314, pp. 369-375, Aug. 2001.

[66] M. Wu, E. Tang and B. Liu, "Data hiding in digital binary image," *Proc. IEEE International Conf. on Multimedia and Expo*, vol. 1, pp. 393-396, 2000.

[67] H. Lu, A. C. Kot and Y. Q. Shi, "Distance-reciprocal distortion measure for binary document images," *IEEE Signal Processing Letters*, vol. 11, no. 2, pp. 228-231, Feb. 2004.

[68] H. Lu, J. Wang, A. C. Kot and Y. Q. Shi, "An objective distortion measure for binary document images based on human visual perception," *Proceedings of the Sixteenth International Conference on Pattern Recognition (ICPR 2002)*, Quebec City, vol. 4, pp. 239-242, Aug. 2002.

[69] N. R. Pal and S. K. Pal, "A review on image segmentation techniques," *Pattern Recognition*, vol. 26, no. 9, pp. 1277-1294, 1993.

[70] D. Wang and S. N. Srihari, "Classification of newspaper image blocks using texture analysis," *Computer Vision, Graphics, and Image Processing 47*, pp. 327-352, 1989.

[71] M. Singh, "Image segmentation-quantitative evaluation", Department of Computer Science, University of Exeter, UK. [Online]. Available: http://www.dcs.ex.ac.uk/research/pann/pecva/segment/list.htm

[72] G. Nagy, S. Seth, and M. Viswanathan, "A prototype document image analysis system for technical journals," *Computer*, vol. 25, no. 7, pp. 10-22, 1992.

[73] A. K. Jain, "Fundamentals of digital image processing," Prentice-Hall, Inc. Cliffs, New Jersey. pp. 358, c1989.

[74] M. Maragoudakis, E. Kavallieratou, N. Fakotakis and G. Kokkinakis: "An effective stochastic estimation of handwritten character segmentation bounds", *ISAP 2003: Competitive Environment, Renewable Energy, Distributed Generation*. Lemnos, Greece, 2003.

[75] E. Kavallieratou, H. Antoniades, N. Fakotakis and G. Kokkinakis: "Extraction and recognition of handwritten alphanumeric characters from application forms", *DSP'97, 13th International Conference on Digital Signal Processing*, Santorini, Greece, vol. 2, pp. 695-698, July 1997.

[76] "Glossary of windows terms," [online]. Available: http://www.computertips.com/glossarywin.htm

[77] "TrueType 1.0 Font Files: Technical Specification Revision 1.66," Aug. 1995.

[78] D. E. Knuth, "Computer modern typefaces," Addison-Wesley Pub. Co., 1986.

[79] C. A. Cabrelli and U. M. Molter, "Automatic representation of binary images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 12, pp. 1190-1196, Dec. 1990.

[80] R. O. Duda and P. E. Hart, "Pattern classification and scene analysis," A Wiley-Interscience Publication, New York; Chichester; Brisbane; Toronto; Singapore.

[81] J. Walker, "MD5 command line message digest utility", Apr. 2003. [online]. Available: http://www.fourmilab.ch/md5/