Digital Watermarking of Text Document Images with Cloud Model

Liu Yi

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Computer Engineering

Prince of Songkla University

2011

Digital Watermarking of Text Document Images with Cloud Model

Liu Yi

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Engineering in Computer Engineering

Prince of Songkla University

2011

Thesis Title Digital Watermarking of Text Document Images with Cloud Model

Author Mr. Liu Yi

Major Program Computer Engineering

Major Advisor:

....

(Dr. Somchai Limsiroratana)

Co-advisor:

....

(Dr. Anant Choksuriwong)

Examining Committee:

....Chairperson

(Asst. Prof. Dr. Pornchai Phukpattaranont)

....

(Dr. Somchai Limsiroratana)

....

(Dr. Anant Choksuriwong)

....

(Dr. Chatchai Suppitaksakul)

The Graduate School, Prince of Songkla University, has approved this thesis as partial fulfillment of the requirements for the Master of Engineering Degree in Computer Engineering.

....

(Prof. Dr. Amornrat Phongdara)
Dean of Graduate School

Thesis Title: Digital Watermarking of Text Document Images with Cloud Model

Author Mr. Liu Yi

Major Program Computer Engineering

Academic Year 2011

ABSTRACT

With widespread use of internet and other communication technologies, it has become extremely easy to reproduce, communicate and distribute digital contents. Authentication and copyright protection of information contents has always been a concern in print media. Besides image, audio and video, the text is most extensively used medium travelling over the internet. Digital watermarking came up as a solution for copyright protection problem. In this thesis, we have proposed a new novel digital watermarking of text document images with cloud model. We have designed the suitable location for embedding watermark is chosen, by detection of incircle in the triangle of each three Harris Corners, and determined by cloud model generator to cloud droplets and corresponds to reference point at center of incircle of triangle. Unicode input is converted to binary bits and each bit transform into a cloud drop and embed inside the incircle. The design of watermarked embedded formula primary balances the contradiction between transparency and robustness. And the watermark can be extracted without referring the original image. Our scheme of hiding capacity is middle. For improved hiding capacity, they have been digital watermarking of text documents image using 2-D cloud model. Experimental results illustrate the capacity of cloud watermarking of text documents image with different font size and multilanguage (e.g.: English, Chinese and Thailand). The results of hiding capacity are also compared with 1-D cloud model and 2-D cloud model on text watermarking, however, the hiding capacity will be improved in the future.

Keywords: Digital Watermarking, Cloud Model, Text Document Images, Data Hiding

ACKNOWLEDGEMENT

First and foremost, I would like to express my gratitude to my principal supervisor, Dr. Somchai Limsiroratana, for his patience, encouragement, and professional instructions during my thesis writing. I am deeply grateful of his help in completion of this thesis. I would like to thank my associate supervisor, Dr. Anant Choksuriwong, for reviewing my work and offering their valuable comments and clever suggestions.

I want to thank Asst. Prof. Dr. Pornchai Phukpattaranont and Dr. Chatchail Suphapitaksakul for spending their valuable time and effort in saving on my committee.

I feel lot grateful all the teachers in Prince of Songkla University who once offered me valuable course and advice during my study. And all the staffs in Faculty of Engineering who have helped me.

I would like to express my gratitude to my family for their love and support throughout my life.

Liu Yi

CONTENTS

CONTENTS(CONT.)

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES (CONT.)

LIST OF ABBREVIATIONS

1-D	One Dimension
2-D	Two Dimension
AWGN	Additive White Gaussian Noise
CG	Cloud Generator
CLD	Cloud Drops
CLM	Cloud Model
DRM	Digital Rights Management
CPTWG	Copy Protection Technical Working Group
CSB	Cloud Significant Bit
DCT	Discrete Cosine Transform
DSHI	Data Structure of Hiding Information
DPI	Dot Per Inch
DRM	Digital Rights Management
DVD	Digital Versatile Disk
ECC	Error Correcting Code
ECRYPT	European Network of Excellence for Cryptology
GA	Genetic Algorithm
LNCS	Lecture Notes in Computer Science
NLP	Natural Language Processing
RHIL	Ratio of Hiding Information Length
RTI	Ratio of Threshold Inradius
SDMI	Secure Digital Music Imitative
PDF	Probability Density Function
PSNR	Peak Signal-to-Noise Ratio
WAVILA	Watermarking Virtual Lab

CHAPTER 1

INTRODUCTION

1.1 Importance of Research and Problem Statement

At the present, digital contents i.e. image, voice and video are worldwide use. Copyright protection is mainly application of watermarking because we want to allow user to see or listen contents. Compare to others, there are few research base on text document watermarking that harder than photo image watermarking because of low channel bandwidth for hiding data. For example, using blank spaces between two consecutive characters of word or the blank spaces between two consecutive lines, changing pixels value can be done with only two colors black/white (0/1) instead of full color and that should has minimum effect on human visual perception.

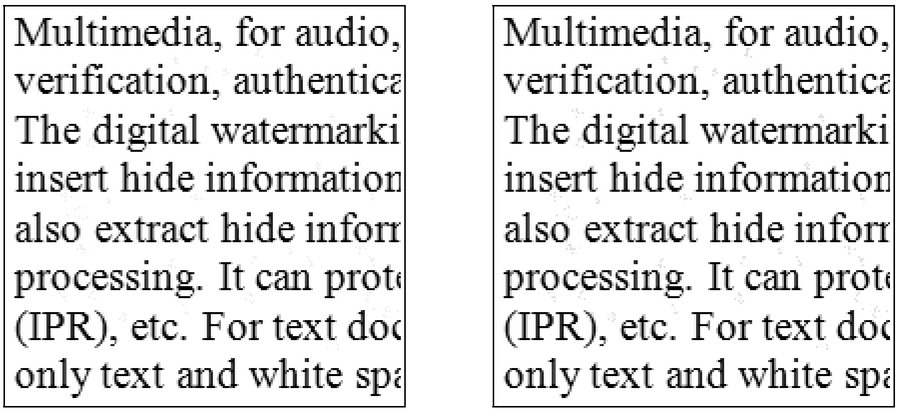
An earlier data hiding algorithm employing the line, word and character shifting or modifying the features of characters to embed the watermark in text document images are proposed in[1]-[6]. Several methods employ the visual distortion table to assess the "flippability" of a pixel [7], pairs of edge patterns [8], visual distortion table [9], and visual quality-preserving rules [10]. Data hiding in the real valued transform domain does not work for binary images due to the quantization errors introduced in the post-processing [11]. The morphological binary wavelet transform can be used to track the transitions in binary images by utilizing the detail coefficients [12], but robustness is middle. Generally speaking, the capacities of these methods are rather low and they are not target for authentication purposes. The watermarking effect of the denoise pattern-based data hiding method is good due to the denoising effects of the patterns, but the capacity is small for images of high resolutions. A further improvement on visual quality is made by embedding watermarks in the DC components of DCT domain [13].

As these methods have been described employing binary image in digital watermarking. Only few parts of methods based on text document are data hiding in the watermarking. In the corruptly digital society, data hiding in text document image have already been far from security. So we propose a new method for watermarking of text documents that "Embedded" location points of pixels values. The proposed method uses a cloud model technique to hiding information.

Now, cloud model technique can be applied to data mining [14], medical field [15] and wireless sensor networks [16].Up to now, R.D. Wang et al. [17] propose to the cloud watermarking of audio using their own audio feature. Y. Zhang et al. [18] the method of protecting relational databases copyright with cloud watermark for solving the invaded and pirating, but there is little correlative work on it and only on the numerical attributes and detection watermark needs the original relational databases.

In conclusion, before there are no people using cloud model technology for watermarking of text document image. In the plan, I just using cloud model technology deal with text document images and generated uncertainly data. Then, the processing of uncertainly data is embedded at the suitable location in the text document images. The suitable locations of pixel are determined by cloud drops and correspond to dereference point at center of incircle of triangle. A single cloud drops is no real meaning, However, the whole of all cloud drops(pixel drops) can have some meaning.

To simplify the coding tasks, the implementation is limited to gray level of text document image. The digital watermarking is based on 1-D cloud model. Figure 1.1(a)-1.1(b) shows digital watermarking in text document images by cloud model which image size 200x200, Font size 16, DPI is 1600, radius threshold is range from 4.12025 to 6.87803, s_f is 0.75, s_v is 16, 32. Total number of cloud drops is 312 and total number of cloud model is 36; the details and the effects will be explained in later chapters.



(a) s_V : 16 (b) s_V : 32

Figure 1.1: Cloud Watermarking in Text Document Image with Different Stego Value

1.2 Objectives

1. The new design for digital watermarking in text document images with cloud model and watermarked detection without the original text document image.

2. To validate digital watermarking of text document images with cloud model.

3. To study the optimal setup for digital watermarking system such as the s_f , s_v and parameters of cloud model.

1.3 Scope

1. Using Microsoft Visual C++ 2008 and Open Source Computer Vision (OpenCV) for implementation.

2. Test different language in the text document image and input hiding Multilanguage information (Such as support Unicode message) is performance of digital watermarking in the text document images by cloud model.

1.4 Tools

1. Testing is carried out on two Window XP/Window 7 machines, a notebook with Intel(R) Pentium(R) Dual T2330 CPU 1.60GHz, and RAM 1GB. A PC with Intel(R) Core(TM)2 Quad CPU 2.66GHz, and RAM 4GB.

2. HP Officejet 6500 Model E709C, HP Ink Cartridge 920(see Figure 1.2).

Figure 1.2: HP Officejet 6500

CHAPTER 2

LITERATURE REVIEW

This chapter presents background information on digital watermarking history. The principle and techniques of digital watermarking are discussed. Besides, the essential differences and advantages of existing watermarking techniques are explained. Moreover, the essential ingredients of text document watermarking are presented. The following section reviews a number of techniques proposed in the literature. Then different watermarking is given and those algorithms are implemented and evaluated. Finally, background of cloud model is explained.

2.1 What is Digital Watermarking?

2.1.1 Definition

Watermarking describes techniques which are used to convey information in a hidden manner by embedding the information into some innocent cover data. Typically, this information is required to be robust against intentional removal by malicious parties. In contrast to cryptography, where the existence but not the meaning of the information is known, watermarking aims to hide the existence of the information from any potential eavesdropper altogether. The general definitions of some common terms used in the area of watermarking are listed below.

Watermark: The information to be hidden. The term watermark also contains a hint that the hidden information is transparent like water.

Cover Media/Cover data: The media or data used for carrying the watermark. Sometimes the terms original media and host media are used to express it.

Watermarked Data: The media which contains the watermark

Embedding: The procedure used for inserting the watermark into the cover media.

Extraction: The procedure used for extracting the embedded watermark from the watermarked data.

Detection: The procedure used for detecting whether the given media containing particular watermark.

Watermarking: The method which is contain the embedding operator and the extraction or detection operator.

In the past decade, owing to the rapid-development of computer technology and the advance of the internet and the ubiquity of digital data, people had shifted their focus from traditional media to digital media (text, image, video or audio). It seems that digital watermarking is a good way to protect intellectual property from illegal copying. It provides a means of embedding a message in a piece of digital data without destroying its value. Digital Watermarking embeds a known message in a piece of digital data as a means of identifying the rightful owner of the data. As a result, watermarking techniques for digital data have been developed and have become popular.

2.1.2 Relationship of Watermarking to Steganography

Digital watermarking techniques derive form steganography, which means covered writing from the Greek words stegano or "covered" and graphos or "to write". Steganography is the science of communicating information while hiding the existence of the communication. The goal of steganography is to hide an information message inside harmless messages in such a way that it is not possible even to detect that there is a secret message present. In watermarking, for example, the important information is the digital media (e.g. image, voice etc.).The watermarked data are additional data for protecting the external data and to prove ownership. In Steganography, however, the external data are not very important. They are just a carrier of the important information.

On the other hand, watermarking is not like encryption. Watermarking does not restrict access to the data while encryption has the aim of marking messages unintelligible to any unauthorized persons who might intercept them. Once encrypted data is decrypted, the media is no longer protected. A watermark is designed to permanently reside in the host data. If the ownership of a digital watermark is in question, the information can be extracted to completely characterize the owner.

Steganography normally relates to point-to-point covert communication between two parties and a steganographic system is typically not required to be robust against intentional removal of the hidden message. Watermark, on the other hand, is usually a one-to-many communication and has the added notion the hidden message should be robust to attempts aimed at removing it. In the case of copyright protection, obviously the copyright information should resist any modification by pirated intending to remove it. Figure 2.1 shows how everything fits together.

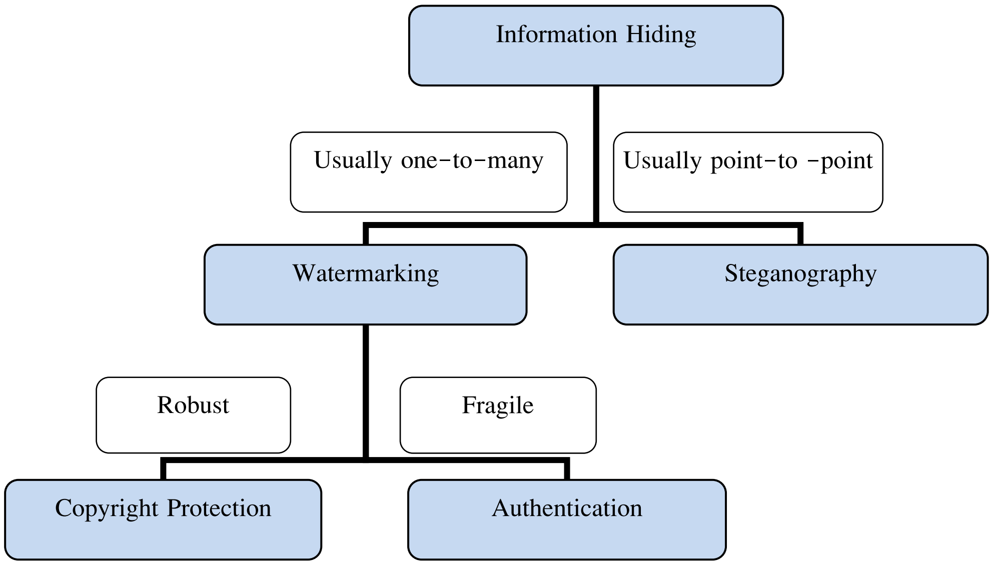


Figure 2.1: Relationship of Digital Watermarking, Steganography and Information Hiding.

Watermarking systems can be robust or fragile. Robust watermarks are required to resist any modifications which do not decrease the commercial value of the cover image. On the contrary, fragile watermarks are designed to fail when the cover image is modified. There is applied to several types of fragile watermarking. Some only allow us to detect if an image has been modified [19]; some allow us to calculate an approximate version of the original image in the modified regions [20]; while some allow us to invert the watermarking process and recover the original un-watermarked image if it is successfully authenticated. The last type is of particular interest to the medical community. Medical images cannot afford to be modified since this might cause misdiagnosis.

Fingerprinting refer to specific applications of watermarking, where the payload carries information such as the creator and the intended recipient of a piece of data. This is similar to the use of serial numbers to identify an individual copy of a product. Fingerprinting can be used to trace the origin of a piece of data if unauthorized copiers if it are found.

2.2 A Brief History

The Invention of digital watermarking can be ascribed Emil F. Hembrooke of the Muzak Corporation, who in 1954 filed a patent [21] describing a method for the embedding of inaudible codes into music signals with the objective of proving their ownership. However, it was not until the 1980s that the first discussions about digital watermarking can be traced back. The term digital watermarking appears to be first used in 1988 by Komatsu and Tominaga [22], even if the interest in digital watermarking technologies started to grow significantly from the middle of 1990s, how it is disclosed by the number of papers published on watermarking by IEEE [23]. This increasing attention was motivated by the beginning of Internet pervasiveness and by the wide spread of digital media distribution, which revealed the need of protection for intellectual property rights of digital contents and watermarking seemed to be a feasible option. Consequently, first digital watermarking techniques were mainly devoted to this aim, as originally conceived in [21]. However, the potential applications rapidly expanded to include a wide range of new applications, such as copy protection, metadata annotation, etc., and at the same time the watermarking methods were applied to an increasing variety of digital contents. All these applications were contained in the general definition of "information hiding" technologies.

At the end of 1990s, the great effort of information technology community was followed by the attention of several organizations and international conferences (LNCS International Workshop on Digital Watermarking, LNCS International Hiding Conference, LNCS Transactions on Data Hiding and Multimedia Security, and Multimedia and Security Workshop [24]).The Copy Protection Technical Working Group (CPTWG) [25] tested the adoption of watermarking technology in DVD to protect video contents. The European Union sponsored some projects [26] to test watermarking for broadcast monitoring. The Secure Digital Music Imitative (SDMI) [27] developed a system for protecting music based on watermarking. Some skeptical voices expressed serious concerns in opposition to the effectiveness of watermarking technologies in real-world application [28].This criticism brought to a deep reflection about watermarking with clarifying the digital watermarking was still an open topic and far from being a mature technology and identifying new challenges for information hiding. Some of these were focused in [29][30], where the scientific community foresaw relevant interest in watermarking security [31] and in the side information schemes [32]. In fact these were some of the topics of the Watermarking Virtual Lab (WAVILA) of the European Network of Excellence for Cryptology (ECRYPT) [33], a project funded by the European Community.

From them on, research activity on information hiding continues reach unmistakable results. Even if digital watermarking technologies have not successfully addressed some of the challenges in the Digital Rights Management (DRM) field, they have found market opportunities in other scenarios. For the moment, watermarking technologies are successfully exploited by some companies in a wide range of applications. Such as, Digimarc [34], TRedess [35], Datamark [36], MSI Copy Control [37] and Aquamobile [38] are some examples of companies with business models based on information hiding technologies.

Furthermore, the Digital Watermarking Alliance (DWA) [39] has been recently constituted by "a group of companies that share a common interest in furthering the adoption of digital watermarking and which are actively involved in commercialization of digital watermarking-based applications, systems and services". This renewed interest is justified by some reports [40] that foresee in the next years a rapid grow of identification technologies, such as digital watermarking, surpassing US $ 500 million worldwide by 2012.

2.3 Basic Watermarking Systems

Watermarking technique is a particular embodiment of multi-media security. Digital Watermark is defined as a digital signal or pattern inserted into a digital data, which can also be referred to copyright information. Watermarking is a key process in the protecting copyright ownership of electronic data, including image, videos, audio, etc.. The term watermarking comes from using the invisible ink to write secret messages [21]. The additional requirement for watermarking is robustness. Even if the existence of a watermark is known, such as the case in public watermarking schemes, it should be ideally impossible for an attacker to remove or destroy the embedded watermark without rendering the cover object unusable. Generally, watermark has three distinct properties imperceptible, inseparable from the work, and undergoes the same transformation as the work.

A simple of watermark system is shown in Figure 2.2. Normally, the company logo of signature will not be embedded into the digital multimedia directly; they will be firstly converted into watermark sequences, which have a noise-like structure and properties. Watermark is a design of the watermark signal W to be added in to the host signal. The watermark signal, apart from depending on the watermark information W', may also depend on a key K and the host data I into which it is embedded, shown in Equation 2.1.

$$W' = f(W, I, K) \tag{2.1}$$

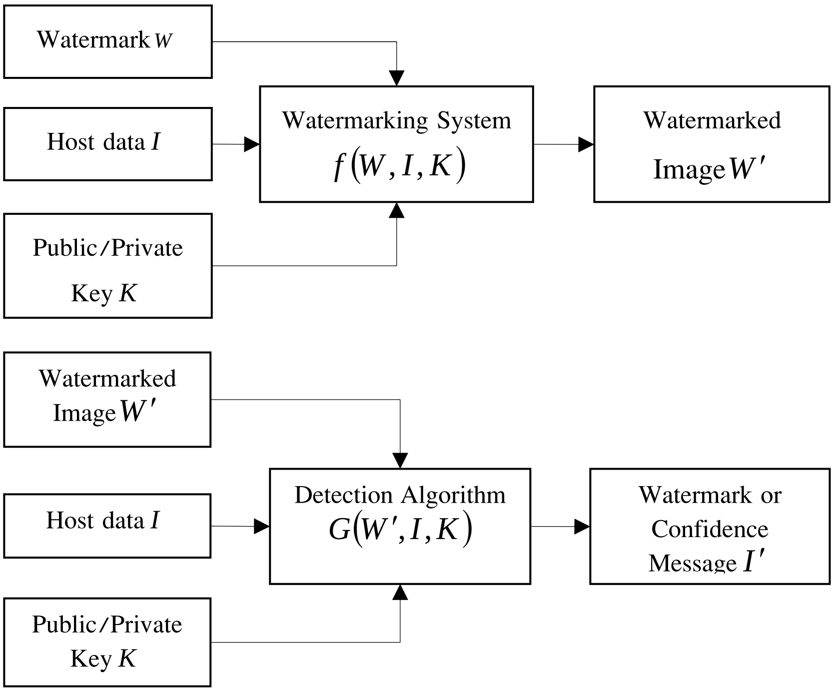


Figure 2.2: Watermarking Embedding and Detection Scenario

In watermarking algorithm, the host data I , the algorithm watermarks the image with a watermark W and output the watermarked image W' with the Equation 2.2.

$$I \oplus W \rightarrow W' \tag{2.2}$$

Verification algorithm is a design of the corresponding extraction method the recovers the watermark information from the sign mixture, perhaps with the help of the key and the original, shown in Equation 2.3.

$$I' = G(W', I, K) \tag{2.3}$$

Given the increasing availability of cheap yet high quality scanners, digital cameras, digital copiers, printers and mass storage media the use of document images in practical applications is becoming more widespread. However, the same technology that allows for creation, storage and processing of digital documents form, also means of mass copying and tampering of documents. Given the fact that digital documents need to be exchanged in printed format for many practical applications, any security mechanism for protecting digital documents would have to be compatible with the paper-based infrastructure. Consider for example the problem of authentication. Clearly an authentication tag embedded in the document should survive in the printing process. That means that the authentications tag should be embedded inside the document data rather than appended to the bit stream representing the document. The reason is that if the authentication tag is appended to the bit stream, a former could easily scan the document, remove the tag, and make changes to the scanned copy and then print the modified document.

Before we describe the different techniques that have been devised for data hiding, digital watermarking and steganography for document images, we briefly list different applications that would be enabled by such techniques.

2.4 Watermarking Applications

Most of the new research trends are technology driven. Digital watermarking is a good example of this fact. Depending on the specific application of a watermarking system, the actual requirement will vary. The emerging applications for data embedding have pushed the research in the field. At the beginning objective of watermarking was copyright protection. However, it has quickly evolved to other application as well. Here we review a few of the major applications of digital watermarking.

2.4.1 Copyright Protection

Copyright protection is one of the major forces which drive the research in watermarking. Data can now be distributed in digital format with ease due to the existent of the Internet. The objective here is to embed copyright information into the data so that the rightful owner of a piece of data can at least prove his/her ownership in case of a dispute. The watermarks in this scenario obviously require a high level of robustness and should resist attempts in removing them. Note that watermarks for copyright protection do not prevent people from copying the digital data, they simple exited as a means for owners to assets ownership over that digital data.

To assert ownership of a document, Alice can generate a watermarking signal using a secret private key, and embed it into the original document. She can then make the watermarked document publicly available. Later, when Bob contends the ownership of a copy derived from Alice's original, Alice can produce the unmarked original and also demonstrate the presence of her watermark in Bob's copy. Since Alice's original is unavailable to Bob, he cannot do the same provided Alice has embedded her watermark in the proper manner. For such a scheme to work, the watermark has to survive operations aimed at malicious removal. In addition, the watermark should be inserted in such a manner that it cannot be forged, as Alice would not want to be held accountable for a document that she does not own.

2.4.2 Copy Protection

In contrast to copyright protection, a copy protection mechanism actually prevents users form marking unauthorized copies of the digital data. This is difficult in open system like the Internet but it is possible to enforce copy protection in a controlled system like the DVD player. For example, every copy machine in an organization can include special software that looks for a watermark in documents that are copied. On finding a watermark the copier can refuse to create a copy of the document. In fact it is rumored that many modern currencies contain digital watermarks which when detected by a compliant copier will disallow copying of the currency. The watermark can also be used to control the number of copy generations permitted. For example a copier can insert a watermark in every copy it makes and then it would not allow further copying when presented a document that already contains a watermark.

2.4.3 Authentication

Fragile watermark is embedded so that if the data is manipulated in an unauthorized fashion, then the watermark will be destroyed to indicate that the data is not authentic. This type of application is important when a piece of audio is used as evidence in the court. Given the increasing availability of cheap yet high quality scanners, digital cameras, digital copiers and printers, the authenticity of documents has become difficult to ascertain. Especially troubling is the threat that is posed to conventional and well established document based mechanisms for identity authentication, like passports, birth certificates, immigration papers, driver's license and picture IDs. It is becoming increasingly easier for individuals or groups that engage in criminal or terrorist activities to forge documents using off-the-shelf equipment and limited resources.

Hence it is important to ensure that a given document was originated from a specific source and that it has not been changed, manipulated or falsified. This can be achieved by embedding a watermark in the document. Subsequently, when the document is checked, the watermark is extracted using a unique key associated with the source. And the integrity of the data is verified through the integrity of the extracted watermark. The watermark can also include information from the original document that can aid in undoing any modification and recovering the original. Clearly a watermark used for authentication purposes should not affect the quality of the document and should be resistant to forgeries. Robustness is not critical, as removal of the watermark renders the content inauthentic and hence is of no value.

2.4.4 Fingerprinting

Each customer is assigned a unique fingerprint with the purchased media. This technique is used to trace back the sourced of illegal copies. For example, if a customer made illegal copies then by reading the embedded fingerprint, the owner can be identified and punished. In applications where documents are to be electronically distributed over a network, the document owner would like to discourage unauthorized duplication and distribution by embedding a fingerprint in each copy of the data. If, at a later point in time, unauthorized copies of the document are found, then the origin of the copy can be determined by retrieving the fingerprint. In this application the watermark needs to be invisible and must also be invulnerable to deliberate attempts to forge, remove or invalidate. The watermark should also be resistant to collusion. That is, a group of k users with the same document but containing different fingerprints should not be able to collude and invalidate any fingerprint or create a copy without any fingerprint.

2.4.5 Metadata Binding.

Metadata information embedded in an image can serve many purposes. For example, a business can embed the Web site URL for a specific product in a picture that shows an advertisement for that product. The user holds the magazine photo in front of a low-cost CMOS camera that is integrated into a personal computer, cellular phone, or a personal digital assistant. The data are extracted from the low-quality picture and is used to take the browser to the designated Web site. For example, in the media bridge application, the information embedded in the document image needs to be extracted despite distortions incurred in the print and scan process. However, these distortions are just a part of a process and not caused by an active and malicious adversary.

The above list represents example applications where digital watermarks could potentially be of use. In addition, there are many other applications in digital rights management (DRM) and protection that can benefit from data hiding and watermarking technology. Examples include tracking the use of documents, automatic billing for viewing documents, and so forth. From the variety of potential applications exemplified above it is clear that a digital watermarking technique needs to satisfy a number of requirements. Since the specific requirements vary with the application, data hiding and watermarking techniques need to be designed within the context of the entire system in which they are to be employed. Each application imposes different requirements and would require different types of watermarking schemes.

Many applications require that the information embedded in a document be recovered despite accidental or malicious distortions they may undergo. Robustness to printing, scanning, photocopying, and facsimile transmission is an important consideration when hardcopy distributions of documents are involved. There are many applications where robust extraction of the embedded data is not required. Such embedding techniques are called fragile embedding techniques. For example, fragile embedding is used for authentication whereby any modification made to the document can be detected due to a change in the watermark itself or a change in the relationship between the content and the watermark. In the next section, we summarize recent developments in text document image watermarking techniques.

2.5 Watermarking Techniques for Text Documents

Digital watermarking is the process of embedding a unique digital watermark in a digital content to protect it from illegal copying and copying violation. The process of embedding and extraction a digital watermark to and from a digital text document which uniquely identifies the original copyright owner of that text is called digital text watermarking. It can be classified in the following categories: an image based approach, a syntactic approach, a semantic approach and a structural approach (The classification is summarized in Figure 2.3.). Description of each category and the work done accordingly are as follows:

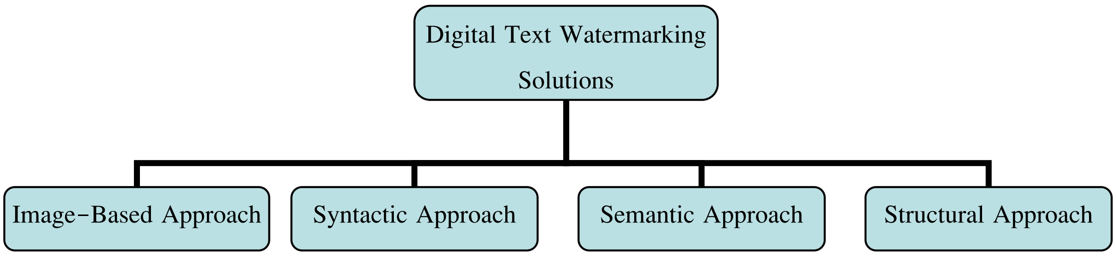


Figure 2.3: Digital Text Watermarking Solutions

2.5.1 Image-Based Techniques

Text document image is used to embed watermark in this approach. Text is difficult to watermark because of its simplicity, sensitiveness, and low capacity for watermark embedding. The initially attempts in text watermarking tried to treat text as image. Watermark was embedded in the layout and appearance of the text image.

The line-shift method is slightly shifted up or down each line according the value of a specific bit in the payload. It has low embedding capacity but the embedded data are robust to severe distortions introduced by processes. Decoding is achieved by comparing the distances between the bases of the lines which are normally uniformly spaced in the original document, or the distances between the centroids of the lines. The word-shift method, each line is first divided into groups of words. Arch group has a sufficient number of characters. Then, each even group is shifted to the left or the right according to the value of a specific bit in the payload. That method has better data embedding capacity but reduced robustness to printing, photocopying and scanning. And feature coding, certain text features are altered in a specific way to encode the zeros and ones of ones of the payloads. Watermark detection is achieved by comparing the original document with the watermarked document.

In J. Brassil and L. O'Gorman [45] has proposed to increase the data embedding capacity over the line and/or word shifting methods described above. This method is very sensitive to baseline skewing. Proper methods to deal with skewing require further research. S.H. Low et al. [1], [41]-[44] are applicable to documents that contain paragraphs of printed text. Data is embedded in text documents by shifting lines and words by a small amount. In N. Chotikakamthorn [46], character spacing is used as the basic mechanism to hide data. But improvement is needed for the method to be robust against severe document degradations. This method could be done by increasing the block size for embedding data bits, but this also decreases the data embedding capacity. Q. Mei et al. [8] proposed text watermarking algorithm using embedding the eight-connected boundary of a character. This method can be applied to general document images with connected components.

T. Amamo et al. [47] proposed extract local feature form text characters, and then made to the character features to embed data. This method could survive the distortions caused by print-and-scan processes. Its robustness to photocopying needs to be furthered investigated. In A.K. Bhattacharjya et al. [48] is presented to embed secret messages in the scanned grayscale image of a document. This method was claimed to be robust against printing and scanning. However, this method requires that the scanned grayscale image of a document be available. Matsui and Tanaka [49] were proposed to embed data in the run-lengths of facsimile images. Each run length of black pixels is shortened or lengthened by one pixel according to a sequence of signature bits and some pre-defined rules.

Z. Baharav et al. [50] proposed that two different dither matrices were used in the half-toning process to encode the watermark information and applied in the binary representation of the watermark. And H.-C.A. Wang [51] proposed that embeds data during the half-toning image and requires the original grayscale image. E. Koch et al. [52] proposed that used to form two halftone images and data hiding were embedded through the correlations between the two screens. M.S. Fu et al. [53]-[55] proposed to embed data at pseudo-random locations or select the best location without a set of candidate locations in half-tone image. Those algorithms require the original grayscale image.

M. Wu et al. [7] proposed that employ the visual distortion table to assess the “flippability” of a pixel that finds “suitable” location to hide the watermark data such that the visual distortion is low. Random shuffling is used to equalize the uneven embedding capacity of the image. It is done by random permutation of all pixels in the image after identifying the flappable pixels. In Y.C. Tseng et al. [5], an enhancement was made to the method proposed in H.K. Pan et al. [56] by imposing the constraint is adjacent to another bit that has the opposite value. It is improve the expense of sacrificing some data hiding capacity. And E. Koch et al. [52] have some robustness against noise if the difference between the thresholds for data bits 1 and 0 is sufficiently large, but this also decreases the quality of the marked document.

2.5.2 Syntactic Techniques

In this approach to the text watermarking, the researchers used the syntactic structure of the text to incorporate a watermark bit. Mikhail. J. Atallah et al. [57] proposed offer a natural language watermarking scheme using the syntactic structure of the text. The natural language processing (NLP) algorithm is used to analyze the syntax and semantic structure of the text, while marking changes to incorporate the watermark bits.

Hassan et al. [58] performed morphsyntactic alterations in text to watermark it. And Hassan et al. [59] also analyzed the available syntactic tools for text watermarking. Those algorithms are an effective approach, but the progress of research in NLP is very slow. So, the transformation applied using NLP algorithms most of the time, are not reversible.

2.5.3 Semantic Techniques

The semantic based text watermarking algorithms use the semantic structure of text to embed the watermark in text. Atallah et al. [60] proposed the semantic watermarking schemes in 2000. Peng Lu et al. [61] proposed pruning and grafting method based on text meaning representation string. Jonath et al. [62] and Robert [63] provide study and surveys of digital watermarking techniques for text image and video documents. Zhang et al. [64] explored the application text watermarking in digital reading, in which holders are compensate for any copyright violation.

2.5.4 Structural Techniques

Structural technique is the approach most recently used for watermarking text document. A text watermarking algorithm for copyright protection of text using occurrence of double letters in text to embed the watermark have recently been proposed [65]. Another zero-watermarking algorithm using prepositions and double letters has recently been proposed [66]. The Structural method is not applicable to all types of text documents and not robust under increasing volume of insertion and deletion attacks.

2.6 Comparison Of Text Watermarking

In the section 2.5, describe that these methods for text document Image watermarking. Robustness to printing, scanning, photocopying and transmission is an important consideration when hardcopy distributions of documents are involved. Of the methods described above, the line and word-shifting approaches describe in S.H Low [1], [41]-[44]. The method using intensity modulation of character parts [48] are reportedly robust to printing, scanning and photocopying operations. Three methods, however, have low data capacity.

The methods described in Y.C. Tseng et al. [5], M. Wu et al.[7], Q. Mei et al.[8], Matsui and Tanaka [49], H.-C.A. Wang [51], and M.S. Fu et al. [53]-[55] are not robust to printing, scanning and copying operations but they offer high data embedding capacity. These methods are useful in applications when documents are distributed in electronic form, when no printing, photocopying, and scanning of hardcopies are involved. The method in E. Koch et al. [52] also has high embedding capacity. It offers some amount of robustness if the two thresholds are chosen sufficiently apart, but this also decreases image quality.

Methods based on character feature modifications require reliable extraction of the features. For example, the method described that in T. Amamo et al. [47], the boundary modification method presented in Q. Mei et al. [8] traces the boundary of a character, which can always be reliably extracted in binary images. This method also provides direct and good image quality control. The method described in K. Matsui et al. [49] was originally developed for facsimile images, but could be applied to regular binary document images. The resulting image quality, however, may be reduced.

Other methods base on syntactic structure of in NLP algorithms. For example, the methods described that in Mikhail. J. Atallah et al. [57], Hassan et al. [58] and Hassan et al. [59]. Those algorithms are an effective approach, but the progress of research in NLP is very slow. The semantic based text watermarking algorithms use the semantic structure of text to embed the watermark in text. For example, the method described that in Atallah et al. [60],Peng Lu et al. [61] ,Jonath et al. [62] and Robert [63] , Zhang et al. [64]. And digital text document watermarking base on structural technique is used for watermarking text document [65] and Jalil et al. [66] described that zero-watermarking algorithm using prepositions and double letters. That method is not applicable to all types of text documents and not robust under increasing volume of insertion.

The comparison of the above methods shows that there is a tradeoff between embedding capacity and robustness. Data embedding capacity tends to decrease with increased robustness. We also observed that for a method to be robust, data must be embedded based on computing some statistics over a reasonably large set of pixels, preferably spread out over a large region, instead of based on the exact locations of some specific pixels. For example, in the line-shifting method, data are embedded by computing centroids position from a horizontal line of text pixels, whereas in the boundary modification method, data are embedded based on specific configurations of a few boundary pixel patterns.

In addition to robustness and capacity, another important characteristic of a data hiding technique is its security from a steganographic point of view. The block-based techniques and boundary technique presented may produce marked documents that are distinguishable if they introduce too many irregularities or artifacts. This needs to be further investigated. A similar comment applies to the techniques presented in the second section. In general, it appears that the development of steganography techniques for text or binary documents has not received enough attention in the research community and much work remains to be done in this area.

Table 2.1 summarizes the different methods in term of embedding techniques, robustness, advantages/disadvantages, data embedding capacity, and limitations. Robustness here refers to robustness to printing, photocopying, scanning and transmission.

Table 2.1: Complaint of Digital Watermarking Techniques for Text Document Image

Techniques	Robustness	Advantage(+)/disadvantage(-)	Capacity	Limitations
Line-shifting	High		Low	Formatted text only
Word-shifting	Medium		Low/ Medium	Formatted text only
Character-shift	medium	+Can be applied to languages with no clear-cut word boundaries	Low/ Medium	Formatted text only
Boundary modification	None	+ Can be applied to general binary images - Direct control on image quality	High	

Table 2.1: Complaint of Digital Watermarking Techniques for Text Document Image (CONT.)

Techniques	Robustness	Advantage(+)/disadvantage(-)	Capacity	Limitations
Modification of horizontal stroke widths	Medium		Low/ Medium	Languages rich in horizontal strokes only
Run-length modifications	None	- Image quality may be reduced	High	
Use two-dithering	None			Half-tone images only
Intensity modification of sub-character regain	Medium		Medium	Grayscale image of scanned document only
Embed data at pseudo-random locations	None		High	Half-tone images only
Modified error diffusion	None		High	Half-tone images only
Modified ordered dithering	None		High	Half-tone images only
Fixed partition Logical invariant	None	+ Embed multiple bits within each block -Use of a secret key	High	
Fixed partition: Percentage of 0/1 pixels	Low/ Medium	+ Can be applied to binary images in general -Image quality may be reduced	High	
Fixed partition: 0/1 pixels	None	+Can be applied to binary images in general	High	
Fixed partition: Connectivity-preserving	Medium	+Direct control on image quality -Loss some pixels hiding data	Low/ Medium	
Morphological wavelet transform domain	Medium		High	
Backward distortion minimization	Medium		High	

Syntactical approach	None	+Can be applied to color image −The NLP algorithm is very low	Medium	
Semantic structure	Low		Medium	Formatted text only
Base on structural approach	Low	-Applied to not all types of text images	Medium	

We have presented an overview and summary of recent developments in binary document image watermarking and data hiding research. Although there has been little work done on this topic until recent years, we are seeing a growing number of papers proposing a variety of new techniques and ideas. Research on binary document watermarking and data hiding is still not as mature as for color and grayscale images. More effort is needed to address this important topic. Future research should aim at finding methods that offer robustness to printing, scanning, and copying, yet provide good data embedding capacity. And be developed to evaluate quality images. So, we have been proposed that digital watermarking of text document image using cloud model for provide good data embedding capacity. In the following section, we present the detail of the cloud model background, definite and algorithm.

2.7 Cloud Model

During the history of artificial intelligence, the representation of knowledge holds a large part. But in real word, lots of phenomena are uncertain. The certain and inerratic phenomenon occurred after specifically condition, and existed only in a short and local range of time [67]. As a result of random, fuzziness, incompleteness and disagreement of description, the studies on uncertainty are partial in artificial intelligence [68].

The mathematic implements for handing uncertainty mostly are probability theory and fuzzy mathematics. Bayesian theory uses prior probability and samples data to compute and estimate unknown samples. The uncertain inference models by credence putted forward by Shortliff and evidence theory proposed by Dempster are representation of knowledge's random. In 1948, Shannon introduced entropy which appeared in energy information domain. Entropy can be used to describe the average uncertainty of an idiographic uncertain affair which appears in a case set.

The instrument of handing fuzziness is fuzzy theory [69]. The membership degree and membership function map a conception to a real number in the interval $[0,1]$. The numerical value between 0 and 1 can be used to express the membership degree of a conception. Methods such as fuzzy predicate, fuzzy rules and fuzzy framework will be a fuzzy treatment of precise knowledge.

Based on the above ideology, Profess D.Y. Li proposed cloud model theory to describe the uncertainty of knowledge [70] and is used to processing the uncertainty and providing a means of both qualitative and quantitative characterization of linguistic atoms. We first introduce the definition of membership function and member cloud, and numerical characteristic of cloud, then we have implemented the normal cloud generator and application of cloud model will be described in the following section.

2.7.1 Membership Function and Membership Cloud

In knowledge representation, it is more visual and more intuitive to describe knowledge using concept than using accurate mathematical description. The cloud model can also be seen as a transfer model between the quantitative and qualitative description of knowledge.

A. Membership Functions in Fuzzy Mathematics

Definition 1*: *Membership Function [68]. Let X denote an ordinarily set, $x \in \{x\}$ is called a domain. $\tilde{A}$ is the fuzzy subset on the domain X, which is defined: $\forall x \in \{x\}$, there always exists a numerical variable $\mu_{\tilde{A}}(x)$, which belongs to the interval $[0,1]$. The numerical variable $\mu_{\tilde{A}}(x)$ is called the element x membership degree on $\tilde{A}$. Then the mapping:

$$\mu_{\tilde{A}} : x \to [0,1], \forall x \in X, x \to \mu_{\tilde{A}}(x) \tag{2.4}$$

is called $\tilde{A}$'s membership function.

Membership function is the basis fuzzy math. The most common membership functions used in fuzzy mathematics are [71]

1) Line membership function:

$$\mu_{\tilde{A}}(x) = 1 - kx;$$

2) Γ (Gamma) Membership function:

$$\mu_{\tilde{A}}(x) = e^{-kx};$$

3) Convex membership function:

$$\mu_{\tilde{A}}(x) = 1 - ax^k;$$

4) Cauchy membership function:

$$\mu_{\tilde{A}}(x) = \frac{1}{1 + kx^2};$$

5) Mountain-shaped membership function:

$$\mu_{\tilde{A}}(x) = \frac{1}{2} - \frac{1}{2}\sin\left\{\left[\frac{\pi}{b-a}\right]\left[x - \frac{b-a}{2}\right]\right\};$$

6) Bell-shaped membership function:

$$\mu_{\tilde{A}}(x) = e^{\frac{-(x-a)^2}{2b^2}}.$$

The bell-shaped membership function is mostly used in daily life among them. Membership functions are the bridge between qualitative and quantitative description of a concrete concept. A conversion from qualitative to quantitative description can be drawn through the membership degree. But how to select a clear and concrete membership function is one of the complex issues in the study of fuzzy sets.

B. Entropy

Entropy is an important factor while studding uncertain intelligence. It first was introduced into Thermodynamics by Shannon [72] in 1948 to the area of Information Science. The entropy of information can be described as:

Definition 2: Information Entropy Assume a system X is formed by a series of cases X_i, where $X = \{X_i | i = 1,2,3,...n\}$, and the probability of X_i is $p(X_i)$. Then the information entropy can be defined:

$$H(X) = -\sum_{i=1}^{n} p(X_i)\log p(X_i) \tag{2.5}$$

Information entropy is used to describe the average uncertainty of appearance of X_i in cases set X. The larger entropy has been effect to the more uncertain for information. If the probability of X_i is equal to each other for case set X, when the information entropy reaches its maximum.

Entropy has widely usages after it was proposed such as in statistical physics and quantum physics areas appears the heat-entropy, electronic-entropy, spin-entropy etc.. Through the history of entropy we can see that the entropy is useful equipment for measuring the uncertainty. Based on its attributes, it was used in cloud model for measuring the granularity of a concept.

C. Membership Cloud and Normal Cloud

Based on the description of membership function in fuzzy theory and information entropy for information theory, the Membership cloud can be defined as follows:

Definition 3: Membership Cloud [68]. Let U be the set, $U = \{x\}$ as the universe of discourse represented by exact numerical values, and T a linguistic term associated with U . The membership degree of x in U to the linguistic term T , $u(x)$ is a random number with a stable tendency. And $u(x)$ takes the values in $[0,1]$ A compatibility cloud is a mapping the universe of discourse U to the unit interval $[0,1]$, that is:

$$u(x):U \to [0,1], \forall : x \in U, x \to u(x) \qquad (2.6)$$

The distribution of x on U is called Cloud, x is called a Cloud Droplet.

The cloud is from a series of cloud drops. In the process of the formation of clouds, a cloud droplet is a realization of qualitative concept through numeric measurement. The realization order between the cloud droplets is irrelevant. The random realization in Definition 3 is under the sense of probability, and the confirmation in Definition 3 is the membership degree of qualitative concept under fuzzy sense. As a single cloud droplet of little or no practical significance, just consider the whole clouds emerged by the characteristics of an individual rather than considering the specific characteristics of the cloud droplet.

On the probability of normal distribution is the most commonly used form of distribution. It is described by expectation (E_x) and variance (D). In fuzzy set theory, the bell-shape membership function $\mu_{\tilde{A}}(x) = e^{\frac{-(x-a)^2}{2b^2}}$ is also the most common membership function used in fuzzy sets.

Normal cloud combines the characteristics of two and then makes a further expansion. Normal cloud employs expectation, entropy (E_n) and hyper-entropy (H_e) to make the cloud generator, and generate the conversion model of qualitative description and quantitative description of the concrete concept.

Definition 4: Normal Cloud [68]. Let U denote a quantitative domain composed of precise numerical variables; T is a qualitative concept on U . If the quantitative value $x \in U$ is a random realization of qualitative concept T , $x \to NORM\left(E_X, E_n'^2\right)$, $E_n' \to NORM\left(E_n, H_e^2\right)$ and x's confirmation on T is

$$\mu = e^{-\frac{(x-E_x)^2}{2(E_n')^2}} \quad (2.7)$$

Then the distribution of x on U is called Normal Cloud.

The normal compatibility cloud characterizes the qualitative meaning of a linguistic atom with three digital characterizes:

$$CG(E_X, E_n, H_e)$$

where E_X , E_n and H_e are the expected value, entropy and deviation of the cloud respectively. As show in Figure 2.4, there is a 1-D Normal Cloud Which the expectation is 0, entropy is 3 and hyper-entropy is 0.03. This Cloud owns 10000 cloud droplets and the vertical direction describes the confirmation of the droplets.

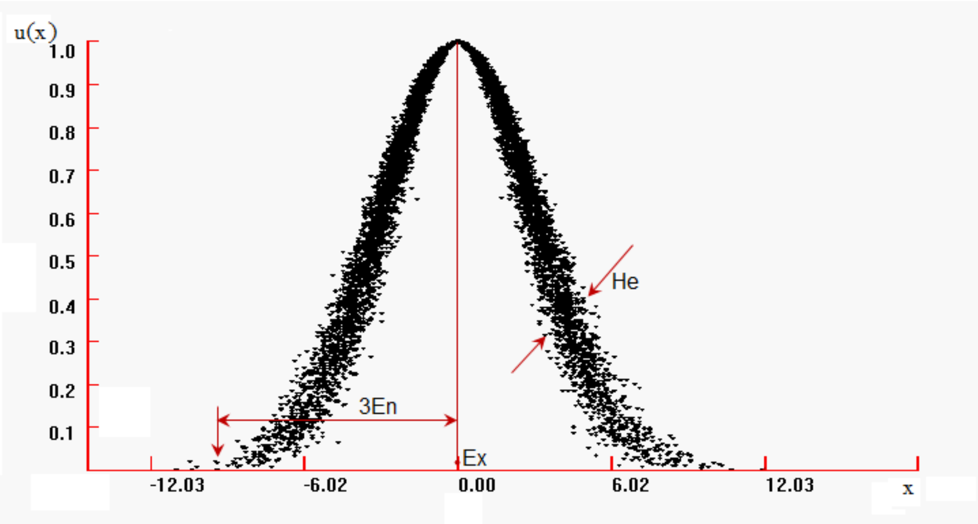


Figure 2.4: Cloud Model (1-D) and its Digital Characteristic

Definition 5: 2-D Cloud Model [68]. Following the above ***Definition 3*** ideas, we extend the linguistic cloud model to 2-Dimensional. Let U be the set $U = \{x, y\}$, as the universe of discourse represented by exact numerical values, and T a linguistic term associated with U . The membership degree of x in U to the linguistic term T , $u(x, y)$ is a random number with a stable tendency. And $u(x, y)$ takes the values in $[0,1]$, a 2-dimemsional compatibility cloud is a mapping form the 2-dimemsional universe of discourse U to the unit interval $[0,1]$, that is:

$$u(x, y) : U \to [0,1], \forall : (x, y) \in U, (x, y) \to u(x, y) \quad (2.8)$$

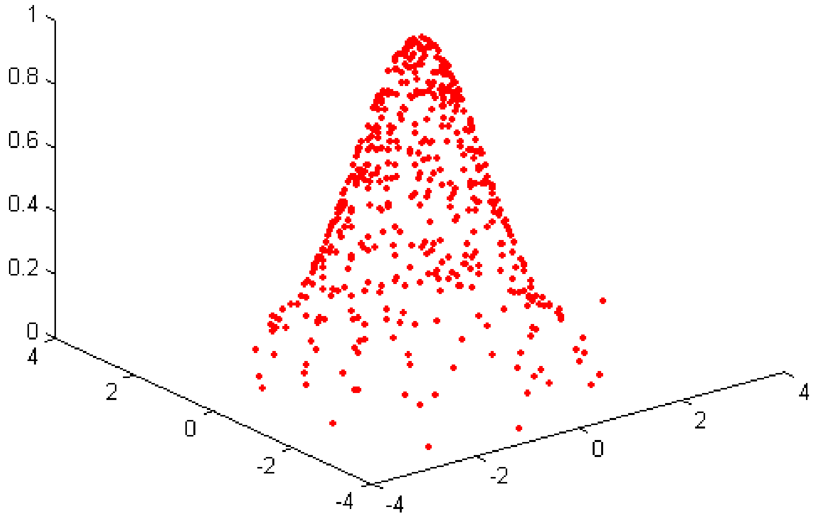


Figure 2.5: Cloud Model (2-D) and its Digital Characteristic

The concept of 2-D clouds is pictured as three-dimensional graphics. Figure 2.5 is 2-D cloud model with the expectation is $(0,0)$, entropy is $(1,1)$ and hyper-entropy is $(0.05,0.05)$. This Cloud owns 200 cloud droplets and the vertical direction describes the confirmation of the droplets. Suppose the two dimensions of a universe of discourse are independent, then the two-dimensional normal comparability cloud for a linguistic term in the universe is characterized with six digital characteristics:

$$CG(E_{xx}, E_{ex}, H_{ex}, E_{xy}, E_{ey}, H_{ey})$$

Where E_{xx} and E_{xy} are the expected values, E_{ex} and E_{ey} are entropies, H_{ex} and H_{ey} are the deviation in the two dimensions and respectively.

When the axes of the ellipse are not parallel to x and y axes respectively, we may add a digital characteristic θ to describe the cloud which is the angle between the corresponding axes. This cloud is referred to as rotated cloud, while the un-rotated cloud is considered to be standard cloud. If (x_i, y_i, u_i) are the drops of a standard 2-D cloud, then (x'_i, y'_i, μ_i) are the droplets of the rotated cloud, where (x'_i, y'_i) are computed as follows,

$$\begin{cases} x'_i = (x_i - E_{xx})\cos\theta - (y_i - E_{xy})\sin\theta + E_{xx} \\ y'_i = (x_i - E_{xx})\sin\theta + (y_i - E_{xy})\cos\theta + E_{xy} \end{cases} \quad (2.9)$$

2.7.2 Numerical Characteristic of Cloud

The cloud model has three numerical characteristics, including Expected value (E_x), Entropy (E_n) and Hyper-Entropy (H_e) [73], which integrates the fuzziness and randomness of spatial concepts in unified way. It is often pictured as two dimensional graphs with the universe of discourse represented as one dimensional.

The expected value (E_x) is center value of concept in the theory field, and it's the most representative value for qualitative concept. It reflects the cloud center of the cloud drops under this concept.

The entropy (E_n) is the measuring of the fuzziness of qualitative concept, reflects the numerical range which can be accepted by this concept in the theory field, and embodies the uncertain margin of the qualitative concept. The bigger the entropy is, the bigger numerical range can be accepted by the concept, and the fuzzier are the concepts. The entropy of a linguistic atom is defined by the bandwidth of the MEC (mathematical expected curve) of the normal compatibility cloud showing how many elements in the universe of discourse could be accepted to the linguistic atom. The MEC of the normal compatibility cloud to a linguistic atom is

$$MEC(u) = \exp\left[-\frac{(u - E_x)^2}{2E_n^{\ 2}}\right] \tag{2.10}$$

The hyper-entropy (H_e) is reflects the dispersion of the cloud drops. The bigger the Hyper-Entropy is the bigger of its dispersion and the randomness of degree of membership. It reflects cohesion in uncertainty of all point that represent relevant language value, that is, the cohesion degree of cloud drops. The value of the hyper-entropy can mirror the discrete degree and thickness of clouds.

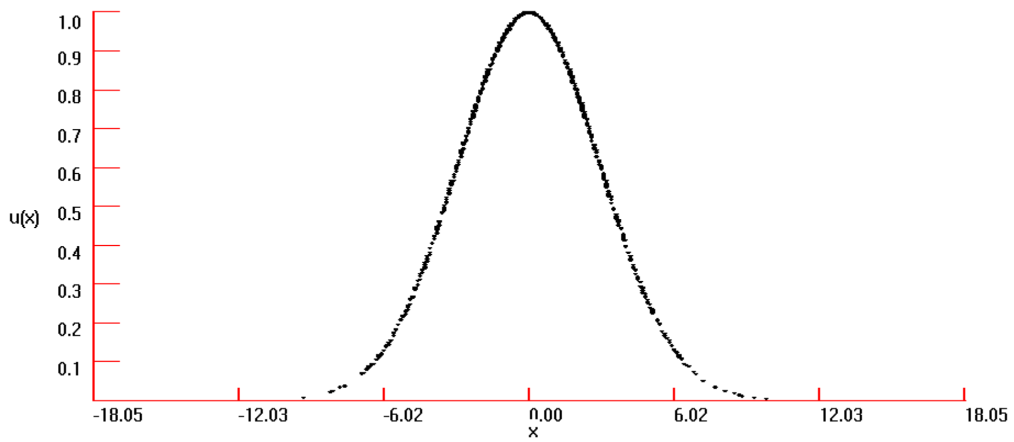


Figure 2.6: Cloud Model with $CG(0,3,0.00001)$ and Total Number of Cloud Drops are 1000

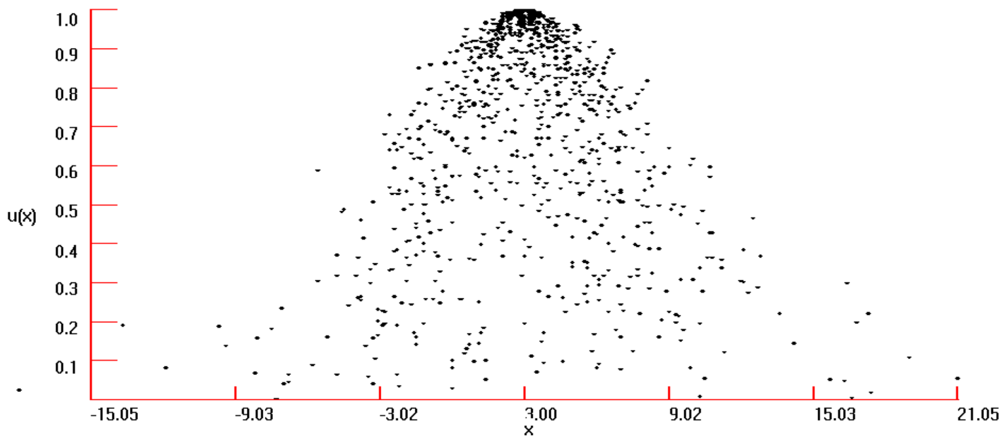


Figure 2.7: Cloud Model with $CG(3,3,2)$ and Total Number of Cloud Drops are1000

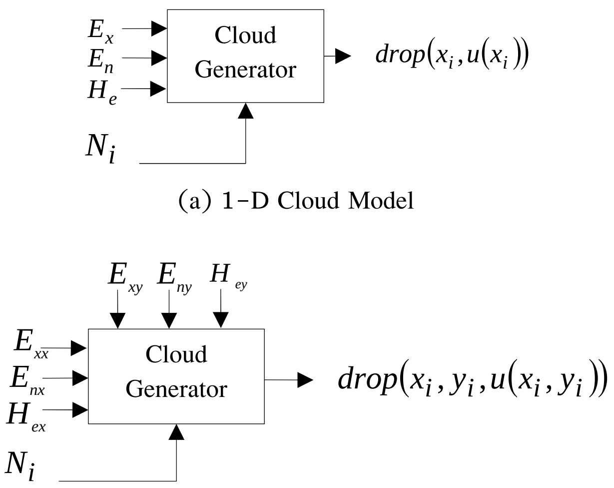
It should be noticed that the top, bottom, and waist of a cloud, however, do not need to be precisely defined at all. The three digital characteristics are good enough to represent a normal compatibility cloud. We have seen that the normal cloud is a useful transitional model between quality and quantity. If all cloud droplets would be much convergent if H_e is very small (see Figure 2.6) like to Normal distribution, cloud watermarking capacity is low and robustness is good. When E_n/H_e is very small, or H_e/E_n is very big, the cloud will behave as the shops of fog on the whole and the cloud is called fog (see Figure 2.7), cloud watermarking capacity is high but robustness is poor. If they have been occur to two situations in the digital watermarking, leading to data hiding capacity is low and poor visual effect. So, we must been avoid very small of H_e and E_n/H_e is very small for digital watermarking system, define as

$$H_e = E_e/\kappa \tag{2.11}$$

where κ is constant.

2.7.3 Normal Cloud Generator

The normal cloud is the most basic tool to express the language value, and a brand new model developed based on both the normal distribution and the bell shaped membership function. Cloud Generator (CG) may be an algorithm implemented in the computer.



(a) 1-D Cloud Model

(b) 2-D Cloud Model

Figure 2.8: Cloud Generator

Give three digital characteristics E_x , E_e and H_e the CG can produce as many drops of the cloud as you would like. All the drops obey the properties described above. Figure 2.8(a) show a 1-D cloud generator. Correspondingly, Figure 2.8(b) shows a 2-D cloud generator. The CG algorithm in details is:

Input: three parameters $CG(E_x, E_e, H_e)$ and the required number of Cloud drops N_{tcld} .

Output: N_i of cloud drops x_i and their certainty degree $u(x_i)$, $i = 0,1,2....,N_i - 1, N_i$.

Algorithm:

1) $En_i' = G(E_n, H_e^2)$, generating a m -dimensional normal random number En_i' with Expectation E_n, and Variance H_e^2 .

2) $x_i = G\left(E_x, En_i'\right)$, generating a m -dimensional normal random number x_i with expectation E_x, and variance En_i' .

3) Calculate $u(x_i) = \exp\left\{-\sum_{j=1}^{m}\left[\left(x_{i_j} - E_{x_j}\right)^2 \Big/ 2*\left(En_{i_j}'\right)^2\right]\right\}$, $(x_i, u(x_i))$ is a cloud droplet.

4) Repeat steps 1) to 3) until N_i cloud droplets are generated.

This algorithm is applicable to the one dimensional universal space situation. $Drop(x_i, u(x_i))$ describe that a cloud droplet where x_i describe an m -dimensional number and $u(x_i)$ is the confirmation of x_i.

Cloud droplets may be generated on conditions, Figure 2.9 shows the generator producing drops under a given numerical value u in the universe of discourse U; all the drops generated in Figure 2.9 have the same value u in the universe of discourse, and normal distributed membership degrees u_i.

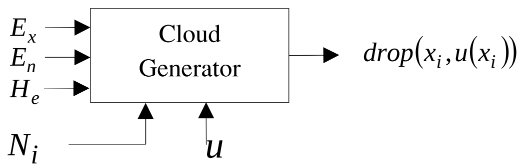
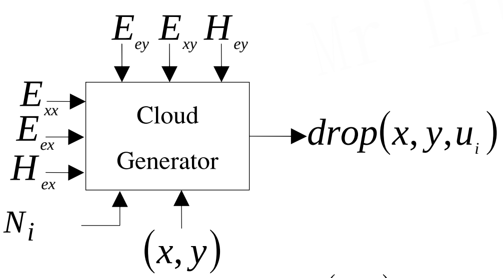
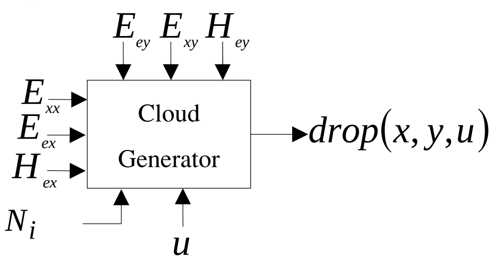


Figure 2.9: Cloud Generator on Condition of “u” (1-D)



(a) On the condition of “(x, y)”



(b) On the condition of “u”

Figure 2.10: Cloud Generator on Condition of “(x, y)” and “u” (2-D)

Similarly, 2-D cloud generator may produce cloud droplets on conditions. All the drops generated in Figure 2.10(a) have the same value (x, y), and normal distributed membership degrees u_i. Whereas all the drops generated in Figure 2.10(b) have the same membership degree, and normal distributed value (x_i, y_i) along the membership ellipse related to u.

During cloud generator processing, H_e is a constant. Usually people define as:

$$E_e / H_e \leq 10 \quad (2.12)$$

To make sure that the inaccuracy in this process is acceptable. With the same processing, the cloud model of unknown can be set up. So, we have known a constant of κ (see equation 2.11)

$$\kappa \geq 10 \quad (2.13)$$

2.7.4 Cloud Model Application

In real word, normal cloud attached lots of people's attention because there is a lot of ambiguity in the concept which uses the normal function to describe is the closet model of human thinking [74]. Cloud model unfolds a good character when it is used to deal with actual knowledge of uncertainty. The usage of similar cloud [75] in measuring and analyzing the uncertainty indicate that cloud model is an ideal tool for copying with the uncertainty of knowledge. Traditional genetic algorithm (GA) easily gets stuck at a local optimum and often has slow convergent speed. Cloud model based genetic algorithm was proposed in [76] which are based on both the idea of GA and the properties of random and stable tendency owned by a normal cloud. It can solve the drawback of traditional GA better. The idea of cloud can be applied to the space knowledge mining [14] [77]-[84] and image processing [15][85] have achieved better results.

2.8 Concept of Proposed Method

Generally speaking, the expected value should be *zero* or variance should not be zero while garneting normal random numbers. It is more adequate to employ these three numerical characteristics to reflect the uncertainty in common concepts, while the use of excessive numerical characters will contradict with the essence of the fact that people take qualitative thinking with natural languages.

According to calculation, we can neglect the contribution to the concept by the cloud drops out of the domain $[E_x - 3E_n, E_x + 3E_n]$. This is named "The $3E_n$ Rule" of the normal cloud. When the drops within $[E_x - E_n, E_x + E_n]$ take up 33.33% of the whole quantity and contribute 68.26% to the concept drops within $[E_x - 2E_n, E_x + 2E_n]$. The contribution to the concept by the cloud drops in different regions is illustrated in Figure 2.11.

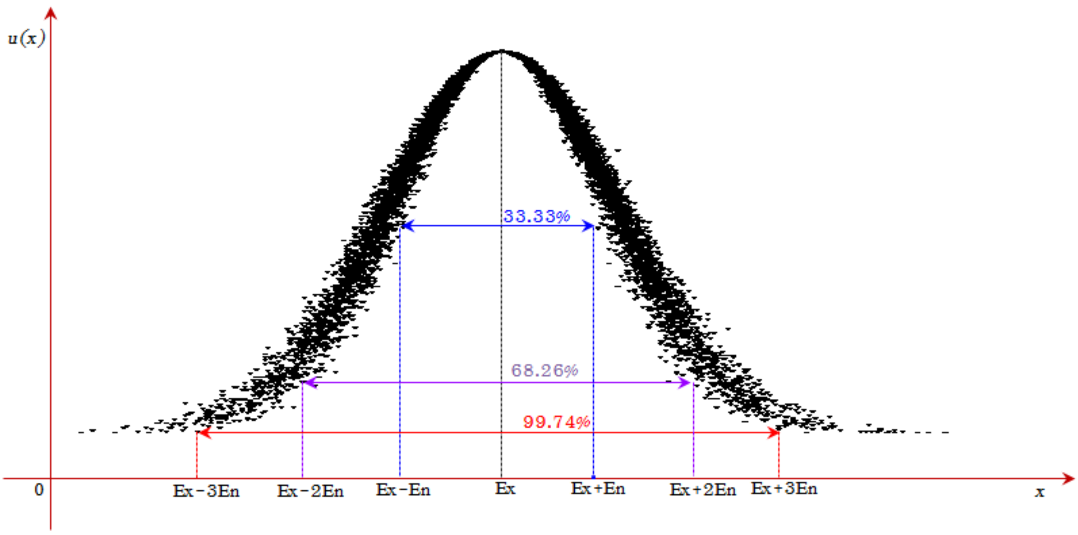


Figure 2.11: Contributions to the Qualitative Concept by Cloud Drops in Different Regions

According to the "The $3E_n$ Rule", there are 99.74% of cloud drops contained between the upper cloud curve and the lower cloud curve, so they have calculated to the value of E_n inradius of the incircle of triangle.

When rules are discovered for text document images databases, we can use the uncertainty reasoning mechanism in cloud theory for predictive digital watermarking. For read the text document image, I wrote the cloud drops base on cloud model technique in the text document images.

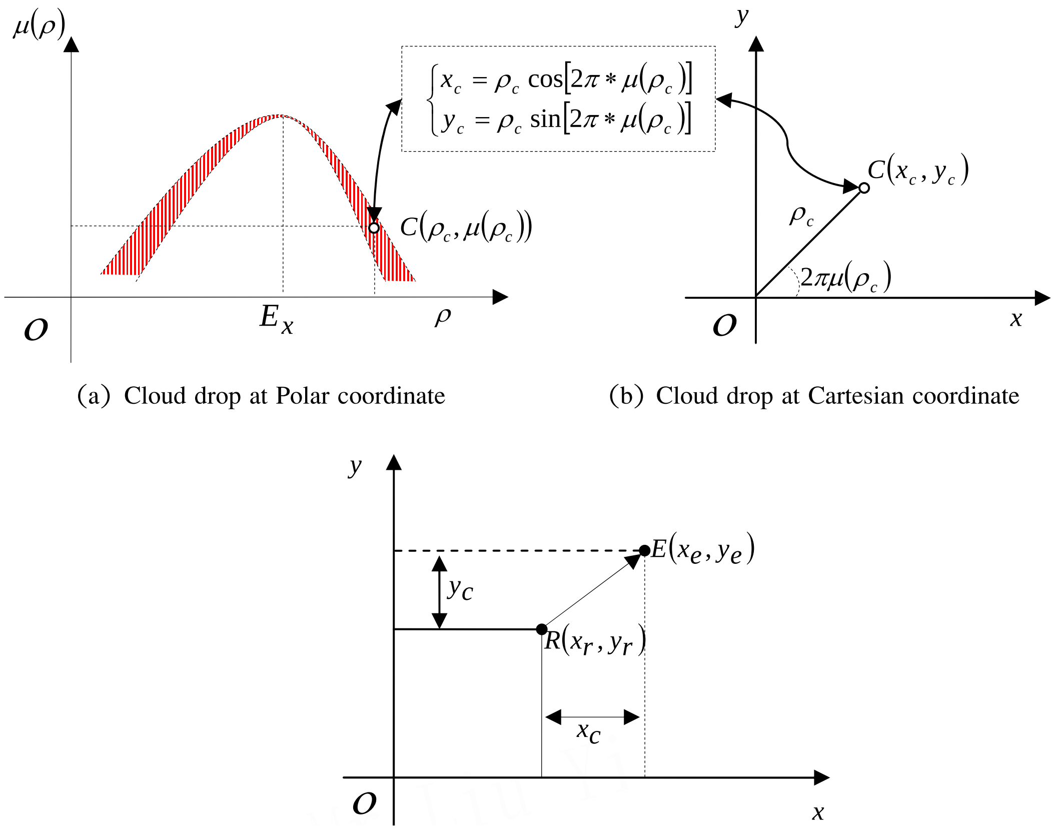
Figure2.12 show that cloud drops convert to text document image location point each 1-D cloud model, the $E(x_e, y_e)$ point call "embedding" pixel point. The $R(x_r, y_r)$ call named referent point. So they have been get as follow as:

$$\begin{cases} x_c = \rho_c \cos[2\pi * \mu(\rho_c)] \\ y_c = \rho_c \sin[2\pi * \mu(\rho_c)] \end{cases} \quad (3.13)$$

and

$$\begin{cases} x_e = x_r + x_c \\ y_e = y_r + y_c \end{cases} \quad (3.14)$$

Where $C(\rho_c, \mu(\rho_c))$ is cloud drop at Polar coordinate.



(a) Cloud drop at Polar coordinate

(b) Cloud drop at Cartesian coordinate

(c) Based on Referent point R generate to correspond to "watermarked" location point at Cartesian coordinate

Figure 2.12: Framework from Connect Cloud model (1-D) to Text document image

Similarly, the cloud droplets convert to text document location point each 2-D cloud model, according to *Equation (2.9)*, they have known that where embedding point in cover image. And Figure 2.13 shows the diagram of this load to dates, which are Inradius data and Referent point in the next chapter for in trials.

Cloud Droplets Location (In Appendix A1)
+ Load Original Image (text document image) + Load Inradius Data, Referent Point - Get Cloud Droplets Point

Figure 2.13: Get Cloud Drops Location

The following code segments are in the method CloudModel1DCloudDrops( ) and CloudModel2DCloudDrops( ), and the full code of the

1-D cloud drops location is in Appendix A1.1 and 2-D cloud drops is in Appendix A1.2.The foundation of the algorithm lies in the way random numbers are generated with normal distribution. The programing Microsoft Studio Visual C++ and OpenCV languages, there are functions to generate random numbers with uniform distribution in $[0,1]$. With the input $E_x = 3$, $E_n = 4$, $H_e = 0.5$ and $N_i = 10000$, the aforementioned algorithm produces the cloud graph of the joint distribution $\mu(x)$ illustrated in Figure 2.14.

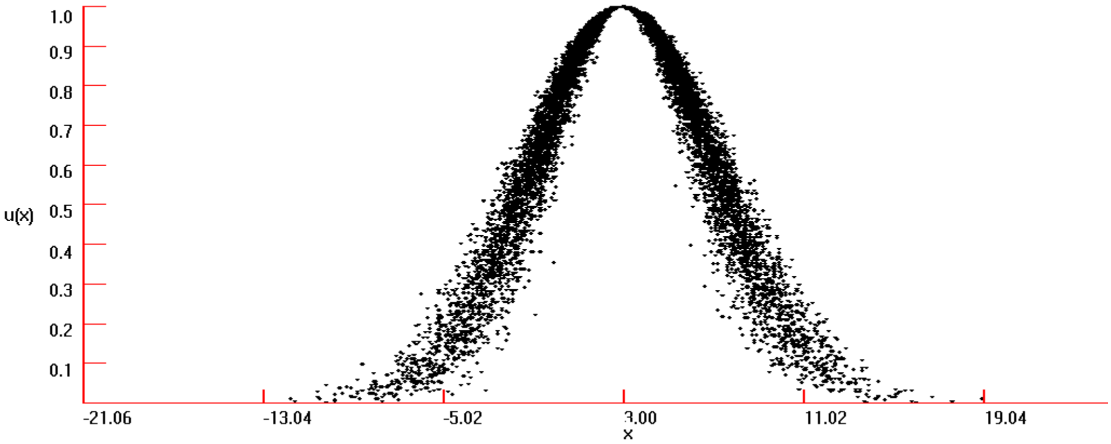


Figure 2.14: The Cloud Graph $\mu(x)$ generated by $CG(3,4,0.5)$

2.9 Summary

This chapter introduced the issues to digital watermarking concept, history, basic system, application and different text document image watermarking techniques, survey on current watermark techniques and cloud model scheme. Novel cloud watermarking of text document image scheme and watermarking preprocess are described in chapter 3 and the experimental results in Chapter 4 are followed. Finally, a conclusion would be given in chapter 5.

CHAPTER 3

NOVEL CLOUD WATERMARKING SCHEMES

In this chapter, we present the proposed innovative digital text document watermarking scheme. Embedding capacity and robustness are analysis and explain. In the following sections, the detail of preprocess cover image and this algorithm are described.

3.1 Preprocess

For searching of a suitable area for embedded watermark, we used Harris corner to detect feature point and making triangle mesh by Delaunay triangulation. Then we search for the triangles that have suitable size by threshold the radius of in-circle in the triangle. Figure 3.1 shows the preprocess information steps. After preprocess, we get or calculate the center point and radius of each selected triangle. Let's call In-radius and In-center Point.

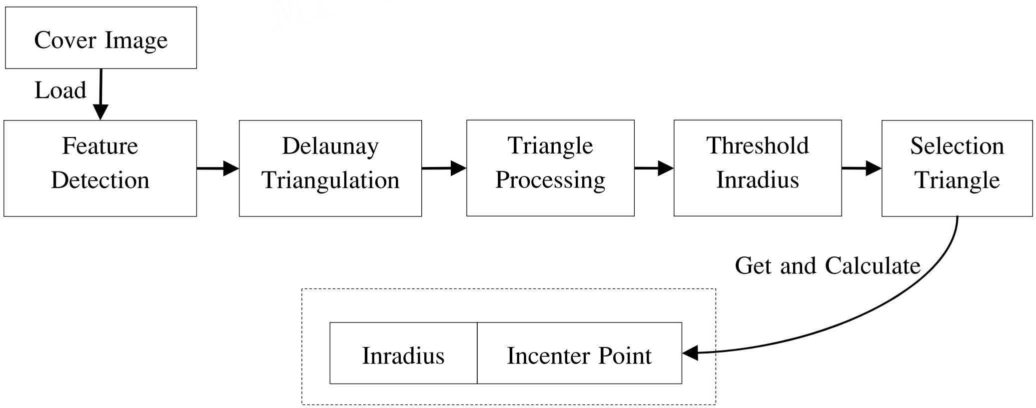


Figure 3.1: Preprocess Steps

3.1.1 Feature Detection by Harris Corner

In order to solve attack effect against rotation, scaling and translation, we use the Harris Corner Detection for adjacent corner points as three vertices of triangle. The Harris Corner Detector is an approach used within computer vision systems to extract certain kinds of features from an image. It is fast enough to work on computers. Also, it is popular because it is rotation, scale and illumination variation independent. However, the corner detector, which implemented in OpenCV is an improvement of the corner detector.

The corner detection frequently used in motion detection, image match, tracking, 3D modeling and object recognition.

The Harris Corner operator was developed by Harris et al. [86] in 1988 as a processing step to build interpretations of a robot's environment based on image sequences. They needed a method to match corresponding points in consecutive image frames, but were interested in tracking both corners and edges between frames. It is computed the locally averaged moment matrix computed from the image gradients and the combines the Eigen values of the moment matrix to compute a corner measure, form which maximum values indicated corners positions.

Consider any unit vector $(\Delta x, \Delta y) = (\cos\theta, \sin\theta)$.

Notice that

$$(\cos\theta, \sin\theta) M (\cos\theta, \sin\theta)^T = \sum_{(x,y)\in N, gd(x_0, y_o)} (\cos\theta, \sin\theta) \times \begin{bmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{bmatrix} \times \begin{bmatrix} \frac{\partial I}{\partial x} & \frac{\partial I}{\partial y} \end{bmatrix} \times \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix},$$

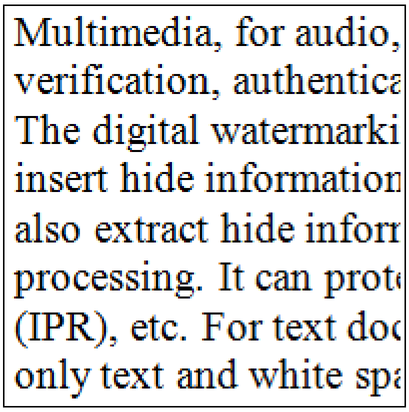
which is the sum of the square of the directional derivatives in direction θ, over the neighborhood of (x_0, y_0). For the intensities at a point (x_0, y_0) to be locally distinctive, the above quantity should be large. How can we quickly examine this condition for each (x_0, y_0). Since M is symmetric, it has two real eigenvalues and the two eigenvectors are orthogonal. By inspection, $(\Delta x, \Delta y) M (\Delta x, \Delta y)^T \geq 0$ for any $(\Delta x, \Delta y)$. Thus, the eigenvalues of M must be positive i.e. letting $(\Delta x, \Delta y)$ be an eigenvector with eigenvalue λ, we see $\lambda |(\Delta x, \Delta y)| \geq 0$. For $(\Delta x, \Delta y) M (\Delta x, \Delta y)^T$ to be much larger than zero for any unit length $(\Delta x, \Delta y)$, we require that both eigenvalues must be much greater than zero.

Note that computing the eigenvalues requires solving a quadratic equation, $a\lambda^2 + b\lambda + c = 0$, which in turn requires computing a square root. We will check every pixel (x_0, y_0) in an image to see if it is locally distinctive, and so it seems we need to compute a square root at each pixel. One trick to avoid computing λ_1, λ_2 explicitly was proposed by Harris and Stevens [86] is to take advantage of a basic fact from matrix algebra that the determinant of a matrix with eigenvalues λ_1 and λ_2 is the product $\lambda_1 \lambda_2$ and the trace is $\lambda_1 + \lambda_2$. The trick is to convert the conditions "both eigenvalues are large" into a condition on the determinant and trace of M, both of which can easily be computed from M i.e. no need to compute the λ's explicitly.

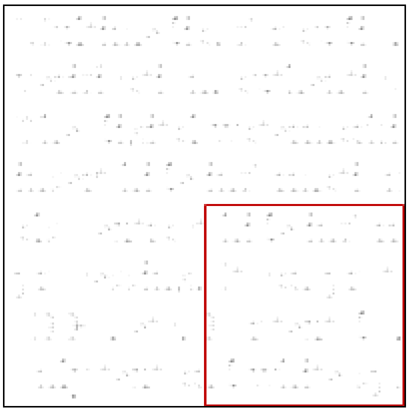
Harris and Stevens noticed that if k is small constant (say $k \approx 0.1$) then $\lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2$ is negative if one of the eigenvalues is 0, and this quantity is small if both of the eigenvalues are small. Otherwise this quantity is large. Thus the Harris corner detector or Harris interest point detector, as it's now known, looks for points in the image where the above expression takes a value above some given threshold.

Notice that if the determinant is near 0 but the trace is much different from zero, then one of the eigenvalues must be large and the other must be near zero. In this case, there must be strong image intensity gradients in the neighborhood of the point, but the gradients would be in only one direction. Thus, Harris and Stevens argued that using the trace and determinant of the second moment matrix could be used to detect "edges" as well as corners, where by "edges" they mean points whose neighborhood contains strong gradients that are all roughly in the same direction.

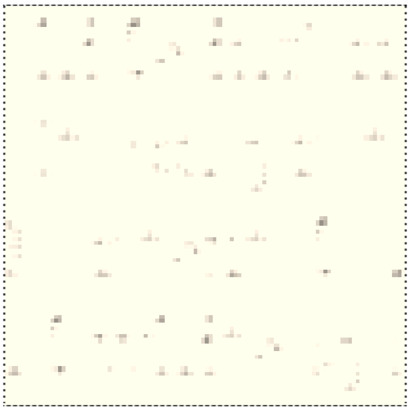
For text document image, we have been choice character feature point by Harris Corner Detection. Figure 3.2(a) shown in original text document image with size 200x200. Figure 3.2(b) shown in feature point result. But parts of feature point are weak and small. See Figure 3.2(c). It will effect to capacity of embedding data, so we want to remove those points. We have been using *cvThreshold( )* function in the OpenCV for threshold. Figure 3.2(d) shown in feature points by Harris Corner Detection with threshold. All the source code of Feature Point is Appendix A2.


Multimedia, for audio,
verification, authentica
The digital watermarki
insert hide information
also extract hide inforr
processing. It can prot
(IPR), etc. For text doc
only text and white spa

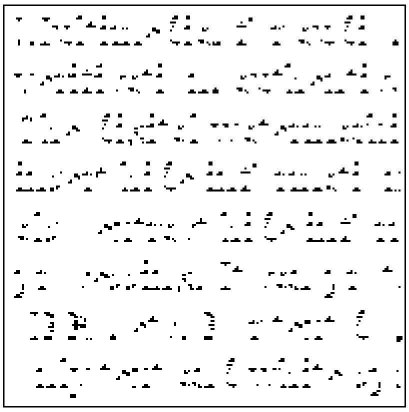
(a) Original Text Document Image



(b) Text Document Image with Feature Points



(c)Parts of Feature Point are Small



(d) Text Document Image with Feature Points

Figure 3.2: Text Document Image with Feature Point Processing

3.1.2 Generate Triangle Based on Detection Feature Points

The section 3.1.1 introduced that feature points detection by Harris corner. In this section, we group those feature points into a group of triangle by Delaunay triangulation. Because Delaunay triangulation technique is for connecting pint in a space into triangular groups such that the minimum angle of all the angles in the triangulation is a maximum. This means that Delaunay triangulation tries to avoid long skinny triangles when triangulating points.

In 1934, Delaunay [87] proved that the dual graph of the Voronoi diagram drawn with straight lines produces a planar triangulation of the Voronoi sites, now called the Delaunay triangulation. An example of the relationship between Voronoi regions and Delaunay triangulation in two dimensions is given in Figure 3.3. Similarly, we can obtain a triangulation for higher dimensions, for example in three dimensions if we connect all pairs of points sharing a common facet in the Voronoi diagram, the result is a set of tetrahedron filling the entire domain.

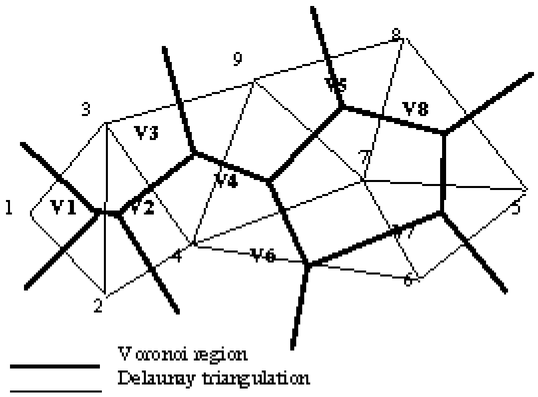


Figure 3.3: Voronoi regions and associated Delaunay triangulation

There are many Delaunay triangulation algorithms, some of which are surveyed and evaluated by Fortune [88] and Su and Drysdale [89]. Their results indicate a rough parity in speed among the incremental insertion algorithm of Lawson [90], the divide-and-conquer algorithm of Lee and Schachter [91], and the plane-sweep algorithm of Fortune [92]. However, the implementations they study were written by different peoples. The triangle is the first software tool which all three algorithms have been implemented with the same data structures and floating-point tests.

The algorithm can be described as follows: Let T_n be the Delaunay triangulation of a set n of points $V_n = \{P_i | i = 1,2,..,n\}$. By defining a simplex as any n -dimensional polygon and a convex hull as the domain to which these points belong to, is formalized as R_s the radius circumscribed to each simplex S of T and as Q_s the center of the n-dimensional circumscribed sphere. Now we insert a new point $P_n + 1$ in the convex hull of V_n and define $B = \{S | S \times T_n | d(P_n + 1, Q_S) < R_S\}$ where $d(p,Q)$ is the Euclidean distance between points P and Q. Now B is not empty as $P_n + 1$ lies in the convex hull of V_n and inside a simplex S_1 belonging to T_n, so at least S_1 belongs to B. The region C formed when B is removed from T is simply connected, contains $P_n + 1$, and $P_n + 1$ is visible from all the points that form the border of C. It is then possible to generate a triangulation of the set of points $V_n + 1 = V_n \{P_n + 1\}$ connecting $P_n + 1$ with all the points that form the border of C: this triangulation is a Delaunay triangulation.

For a given set of points in two dimensions, the Delaunay triangulation is univocally determined and therefore unique, but there are some cases when the triangulation is not unique as there exist different ways of connecting points and all lead to a valid triangulation. This degeneracy is quite common for regular distribution of points, for example in two dimensions when four points lie on a circle and the Voronoi vertexes are coincident. Figure 3.4 shows Delaunay triangulation algorithm steps, with $p_0, p_{1,....} p_{10}$ are feature points by corner Harris and Figure 3.5 show drawing triangle by Delaunay triangulation.

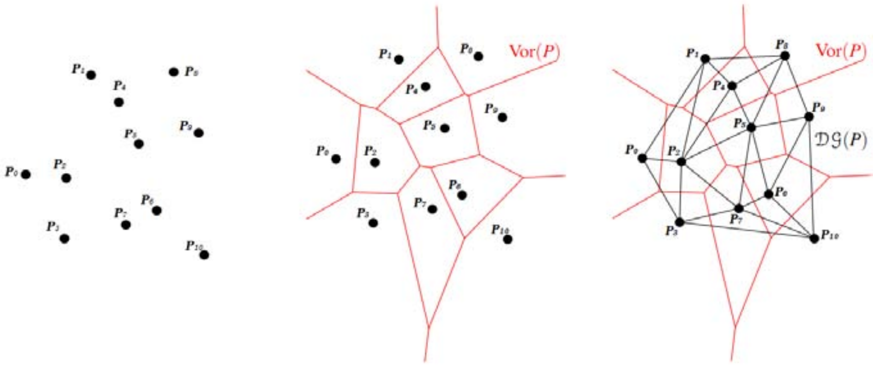


Figure 3.4: Delaunay Triangulation Algorithm Steps

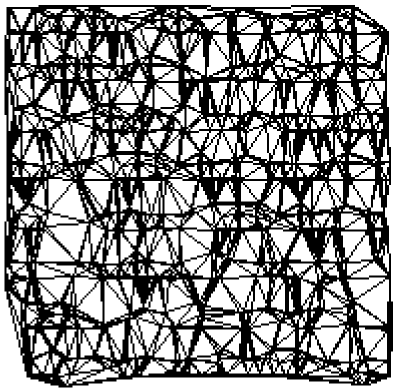


Figure 3.5: Drawing triangle by Delaunay Triangulation Algorithm

3.1.3 The Inradius and Incenter of the Incircle of a Triangle

The incircle of a triangle is that circle which just touches all three side of triangle. Figure 3.6 shows the incircle for a triangle. It is easy to see that the center of the incircle is at the point where the angle bisectors of the triangle meet. A known, but not well advertised theorem is that the inradius of the incircle of a Pythagorean triangle is an integer. After rediscovering this feature of Pythagorean triangle, we found it tucked away in a few number theory texts [93].

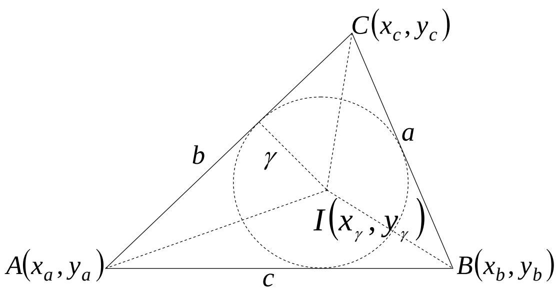


Figure 3.6: The incircle of triangle

Let a, b and c be the sides of a triangle and the three Cartesian vertices are located at $A(x_a, y_a)$, $B(x_b, y_b)$ and $C(x_c, y_c)$, then inradius γ of the incircle is given by

$$\gamma = \sqrt{\frac{(s-a)(s-b)(s-c)}{s}} \qquad (3.1)$$

And the Cartesian coordinates of the incenter $I(x_\gamma, y_\gamma)$ are given by

$$\begin{cases} x_\gamma = \dfrac{ax_a + bx_b + cx_c}{2s} \\ y_\gamma = \dfrac{ay_a + by_b + cy_c}{2s} \end{cases} \qquad (3.2)$$

Where $s = (a+b+c)/2$ is the semiperimeter.

Proof Equitation 3.1: Refer to Figure3.6. The area $(Area)$ of the triangle is the sum of the areas of the three interior triangles made by the angle bisectors. This sum is $Area = a\gamma/2 + b\gamma/2 + c\gamma/2 = (a+b+c)\gamma/2 = s\gamma$. Recalling Heron's Formula for the area of any triangle in terms of the semiperimeter $Area = \sqrt{s(s-a)(s-b)(s-c)}$. Immediately, this completes the proof.

Here we give the source code of function that the inradius and incenter of incircle of a triangle in Microsoft Studio Visual C++ and OpenCV.

```
//The Calucating Inradius Cloud Triangle
private: System::Void CloudTriangleInradius(CloudTriangle^c){
        double distance1 = 0.0;
        double distance2 = 0.0;
        double distance3 = 0.0;

        double perimeter = 0.0;
        double CInradius = 0.0;

        distance1 = sqrt((c->vtx1.X-c->vtx2.X)*(c->vtx1.X-c->vtx2.X)
                    +(c->vtx1.Y-c->vtx2.Y)*(c->vtx1.Y-c->vtx2.Y));
        distance2 = sqrt((c->vtx2.X-c->vtx3.X)*(c->vtx2.X-c->vtx3.X)
                    +(c->vtx2.Y-c->vtx3.Y)*(c->vtx2.Y-c->vtx3.Y));
        distance3 = sqrt((c->vtx3.X-c->vtx1.X)*(c->vtx3.X-c->vtx1.X)
                    +(c->vtx3.Y-c->vtx1.Y)*(c->vtx3.Y-c->vtx1.Y));
```

```
        perimeter = 0.5*(distance1+distance2+distance3);
        CInradius = sqrt((perimeter-distance1)*(perimeter-distance2)
                    *(perimeter-distance3)/perimeter);
 }
//The Calucating Incenter Cloud Triangle
private: System::Void CloudTriangleIncenter(PointF vtx1,PointF vtx2,PointF vtx3){
        double disa = 0.0;
        double disb = 0.0;
        double disc = 0.0;

        double disl = 0.0;
        Point Incenter;

        disa = sqrt((vtx2.X-vtx3.X)*(vtx2.X-vtx3.X)+(vtx2.Y-vtx3.Y)*(vtx2.Y-vtx3.Y));
        disb = sqrt((vtx1.X-vtx3.X)*(vtx1.X-vtx3.X)+(vtx1.Y-vtx3.Y)*(vtx1.Y-vtx3.Y));
        disc = sqrt((vtx2.X-vtx1.X)*(vtx2.X-vtx1.X)+(vtx2.Y-vtx1.Y)*(vtx2.Y-vtx1.Y));
        disl =disa+disb+disc;

        Incenter.X = int(vtx1.X*(disa/disl)+vtx2.X*(disb/disl)+vtx3.X*(disc/disl));
        Incenter.Y = int(vtx1.Y*(disa/disl)+vtx2.Y*(disb/disl)+vtx3.Y*(disc/disl));
}
```

3.1.4 Data Structure of Hiding Information(DSHI) in Each Cloud

The DSHI is a bit structure for alignment of cloud index, data length and data bits. The DSHI (see Figure3.7) is composed of three blocks. The first one that describes the index of cloud model, as the whole is a list of cloud model. The second block is for the bit size of data block in a cloud model. Then, the third block is a data block, which is a part of Unicode hide information bits.

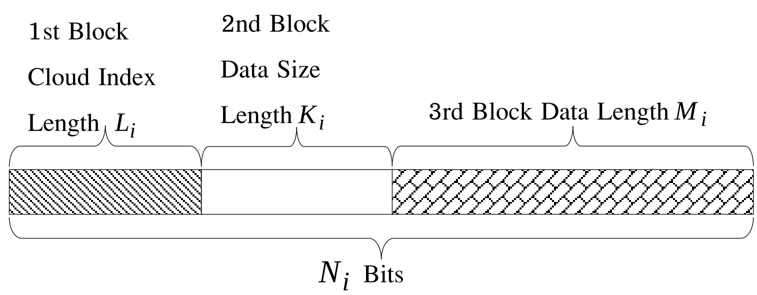


Figure 3.7: Framework of DSHI

The total number of DSHI is equal to total number of real cloud drops using find watermark location one cloud model, but usually different total number of cloud drops every cloud model. At least minimum size of DSHI is $L_i + 2$ bits. If they have been watermarking process a cloud model, the cloud index i th of cloud drops N_i more than 3bits, because of the minimum size of DSHI is 3 bits. The DSHI parameter of calculate:

$$L_i = \lceil \log_2 (i_{\max}) \rceil \tag{3.3}$$

$$N_i = L_i + K_i + M_i \tag{3.4}$$

$$K_i \geq \lceil \log_2 (M_i) \rceil \tag{3.5}$$

Then according Equations 3.3, 3.4 and 3.5, the Ratio of Hiding Information Length (RHIL) data Δ_i calculate as follow:

$$\Delta_i = \frac{M_i}{N_i} \times 100\% = \frac{N_i - \lceil \log_2 (i_{\max}) \rceil - K_i}{N_i} \times 100\% \tag{3.6}$$

In our experiments, Table 3.1, 3.2 and 3.3 show that RHIL obtains the text document image watermarking of capacity in different Language image(e.g.: English, Chinese and Thailand). Figure 3.8 is RHIL data with Multilanguage text document image with different of image size and font size. Horizontal ordinate represented ratio of hiding information length. Vertical ordinate represented is watermarking system average capacity. So, we have known RHIL Δ_i is equal to 30%~50% in the cloud watermarking of text document image.

Table 3.1: The Comparison of Capacity of Different RHIL in the English Text Image

		English Text Image Font Size:									Average Capacity(bits)
		12	14	16	18	20	22	24	26	28	
Ratio of Hiding Information Length (%)	10	133	196	235	340	300	296	356	473	525	317
	20	133	196	235	340	300	296	356	473	525	317
	30	133	196	235	340	300	296	356	473	525	317
	40	133	196	235	340	300	296	356	473	525	317
	50	122	151	223	340	300	296	356	473	525	310
	60	71	72	160	333	270	296	340	473	525	282
	70	10	24	68	225	206	261	220	400	501	213
	80	N*	N	15	88	55	127	97	189	248	117
	90	N	N	N	N	N	N	27	72	N	50

* N: Cloud watermarking is not performed embedding process.

Table 3.2: The Comparison of Capacity of Different RHIL in the Chinese Text Image

		Chinese Text Image Font Size:									Average Capacity(bits)
		12	14	16	18	20	22	24	26	28	
Ratio of Hiding Information Length (%)	10	184	447	471	359	287	485	473	556	625	432
	20	184	447	471	359	287	485	473	556	625	432
	30	184	447	471	359	287	485	473	556	625	432
	40	184	447	471	359	287	485	473	556	625	432
	50	159	447	471	359	287	485	473	556	625	429
	60	133	334	352	264	292	485	473	556	625	390
	70	33	232	200	137	230	381	330	434	518	277
	80	19	61	51	102	143	210	105	232	315	138
	90	N*	N	36	N	N	N	32	29	127	63

* N: Cloud watermarking is not performed embedding process

Table 3.3: The Comparison of Capacity of Different RHIL in the Thailand Text Image

		Thailand Text Image Font Size:									Average Capacity(bits)
		12	14	16	18	20	22	24	26	28	
Ratio of Hiding Information Length (%)	10	288	317	426	163	313	302	354	412	568	349
	20	288	317	426	163	313	302	354	412	568	349
	30	288	317	426	163	313	302	354	412	568	349
	40	288	317	426	163	313	302	354	412	568	349
	50	214	303	426	163	313	302	354	412	568	339
	60	109	162	358	162	204	282	354	412	568	290
	70	24	38	216	121	126	215	354	377	527	222
	80	N*	N	N	14	31	34	263	255	412	168
	90	N	N	N	N	N	N	28	87	110	75

* N: Cloud watermarking is not performed embedding process

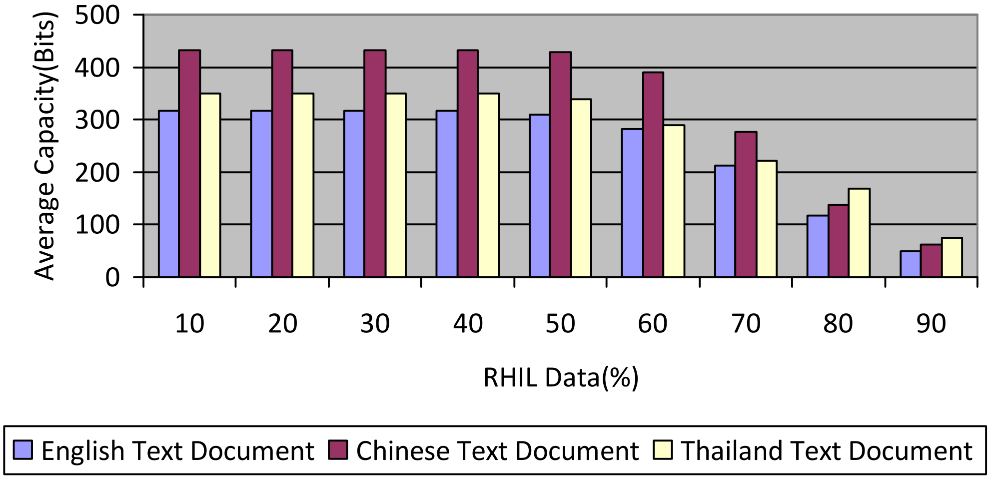


Figure 3.8: Ratio of Hiding Information Length Data

Here, we give the source code function that DSHI in Microsoft Studio Visual C++ and OpenCV.

```
//The Calacuation to HidingDataLength Size function
private: System::Int16 CheckHidingDataLength(int SizeDataLength){
        int HidingDataLength = 0;
        int SizeHideData = 0;
        int CheckHidingDataLength = 0;

        for(int n =SizeDataLength;n>0;n--){
             int DataMax1Length = int (ceil(log10(double(n))/log10(double(2))))+n;
             int DataMax2Length = int (ceil(log10(double(n+1))/log10(double(2))))+n+1;

             if((DataMax1Length<=SizeDataLength)&&(DataMax2Length>SizeDataLength)){
                  HidingDataLength = n;
                  SizeHideData =SizeDataLength-HidingDataLength;
                  CheckHidingDataLength =int(pow(double(2),double(SizeHideData)));

                  if(CheckHidingDataLength<=HidingDataLength){
                      HidingDataLength--;
                      SizeHideData++;
                      break;
                  }
              }
        }
        return(HidingDataLength);
   }
```

3.2 1-D Cloud Watermarking of Text Document Images Scheme

The new watermarking scheme we propose is based on cloud model. Figure 3.9 shows an overview of our watermarking embedding process and Figure 3.10 shows an overview of extraction processing. In our scheme, a text document image and hiding information are taken as the inputs, and the watermark is then embedded in each different parts which are locations of incircle of triangle, pixel value inside is corresponded to cloud droplet.

If applying a fixed parameters of cloud model(e.g.: E_x , E_n and H_e) to digital text document image watermarking lead to the problem of data hiding capacity and invisibility, our scheme employs independent watermarks for data hiding capacity high and invisible, by different input parameters of cloud model. The first parameter of cloud model E_x define equal to *Zero* and adding reference point for solving that different location points in the cloud model. The second parameter of cloud model E_n is come from inradius of incircle of a triangle and "The $3E_n$ Rule". And last parameter of H_e is see *equation 2.11, 2.12 and 2.13.* With these mechanisms, the proposed method is high data hiding capacity and robust against the attacks of pixel values, geometry attacks (i.e.: rotation, scaling and transition). This newly proposed scheme consists of three parts, including: watermark preprocess, watermark embedding and watermark detection. Detail is described in the following sections.

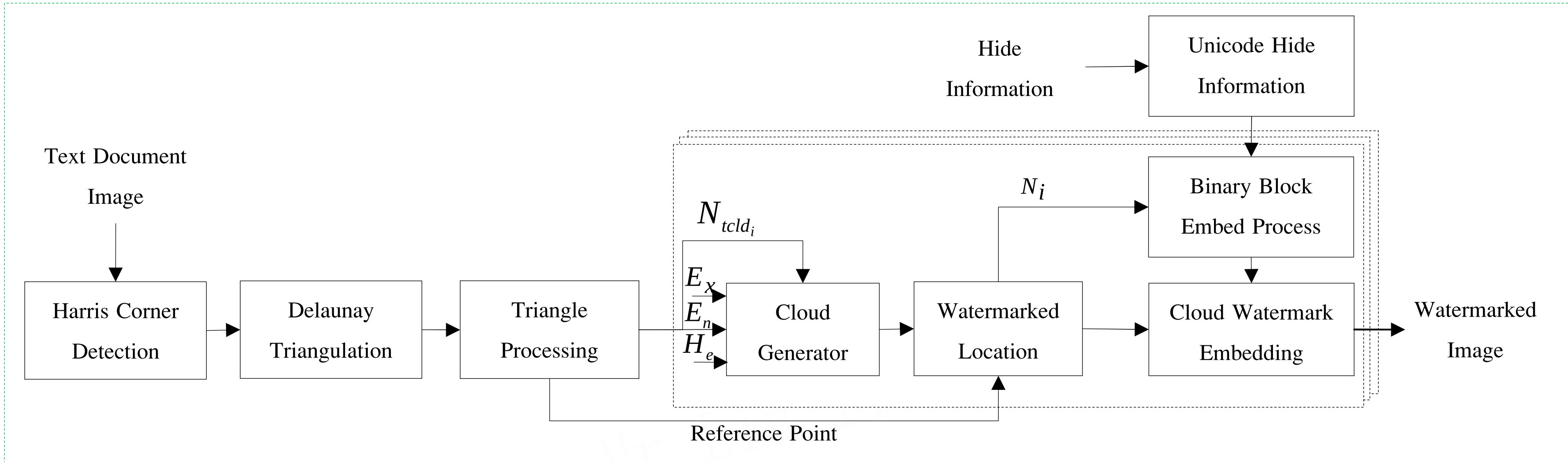


Figure 3.9: Overview of the cloud watermarking embedding process

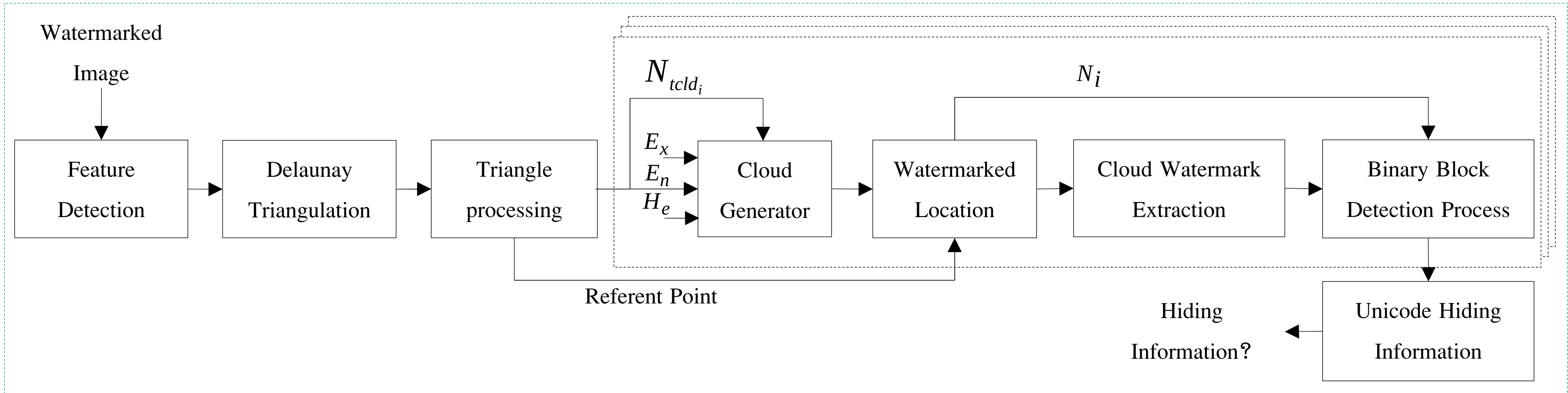


Figure 3.10: Overview of the cloud watermarking extraction process

3.2.1 Watermarking Embedding

We perform cloud model on the text document image. The cloud model generator to cloud drops, and then we adopt Equation 3.7 to embed each cloud drop on the text document image by changing pixel value at the cloud drop location with the condition in Equation 3.1, as follow [94]:

$$f_w(x,y)=\begin{cases} f(x,y)-f(x,y)\bmod S_v+S_f*S_v, \\ \qquad IF w=1, and (1-S_f)*S_v\le f(x,y)\bmod S_v; \\ [f(x,y)-(1-S_f)*Sv]-[f(x,y)-(1-S_f)*S_v]\bmod S_v+S_f*S_v, \\ \qquad IF w=1, and (1-S_f)*S_v> f(x,y)\bmod S_v; \\ f(x,y)-f(x,y)\bmod S_v+(1-S_f)*S_v, \\ \qquad IF w=0, and S_f*S_v\le f(x,y)\bmod S_v; \\ \left[f(x,y)-\frac{S_v}{2}\right]-\left[f(x,y)-\frac{S_v}{2}\right]\bmod S_v+(1-S_f)*S_v, \\ \qquad IF w=0, and S_f*S_v> f(x,y)\bmod S_v. \end{cases} \tag{3.7}$$

Where $f(x,y)$ denotes original pixel value at location (x,y) in original image. $f_w(x,y)$ denotes the pixel value at location (x,y) after embedded watermark bit w. S_f denotes stego factor and $S_f \in (0.5,1)$ .For the watermark to be robust, S_v controls the watermark embedding strength and should be as large as possible under the constraint of invisibility. In what follow, we call S_v the embedding strength. Note that the difference between $f_w(x,y)$ and $f(x,y)$ is between $-\frac{1}{2}S_v$ and $+\frac{1}{2}S_v$. Performing cloud model on the modified image, we obtain a watermarked image.

The cloud watermarking of text document images are embedded processing through the follows steps:

1) Load an image of text document as a cover image. Let the text document image size $m\times n$ denoted as $F=\{f(x,y), 1\le x\le m, 1\le y\le n\}$, where $f(x,y)=\{0,1,2,...,2^l-1\}$ represent pixel value at location (x,y) of the original text document image, l denotes bit size number of bits of gray image pixels.
2) Finding feature points by Harris corner (section 3.1.1). Next, generate triangle from the detected feature points by Delaunay triangle (section 3.1.2). Then, calculate length of inradius of each triangle (section 3.1.3), calculate maximum and minimum threshold of inradius for

choosing corresponds triangle processing. Choose the suitable triangles that have inradius between minimum and maximum threshold.

3) Then output inradius γ as input parameters of cloud model and incenter point as cloud model center point.
4) According to the "the $3E_n$ rule"(section 2.8), there are 99.7% of cloud drops contained between the upper cloud curve and the lower cloud curve. So they have calculated to the value of E_n in the inradius. The normal cloud generate algorithm is described in detail in the section 2.7.3 and 2.8.
5) The one dimension cloud model $CG(E_x, E_n, H_e)$ input three parameters and number of cloud drops: $E_x=0$, $E_n=\frac{\gamma_i}{3}$, $H_e=\frac{\gamma_i}{30}$, and number of cloud drops: $N_{tcld_i}=0.3\pi\gamma_i^2$. The E_x is set to zero, for position of cloud depend on center of triangle.
6) By cloud drops and incenter point, find the "embedding" pixels location in the text document image. Information size depends on number of cloud drops. The DSHI contain hide information. The DSHI is described in the section 3.1.4.
7) Using the "embedding" pixels location and DSHI bit data"1" or "0" for changing "embedding" pixels value according to Equation 3.7.
8) Process the next cloud model by following step 3) to 8).
9) When all processing of embedding cloud watermarking finished.

This algorithm is applied to one dimension cloud model. In our experiment, first, we have solving that threshold minimum and maximum in the cloud watermarking system. Second, how many parameters of S_v, S_f in the cloud watermarking processing, those parameters value is affected that data hiding capacity in the text document image. We have been analysis those in the next section (see section 3.4).

I wrote functions for Embedding process as shown in Figure 3.11.

Method	Process
OpenMenuItem_Click (......)	Get the original text document image with processing pixel value and location(See Appendix A4)
HarrisCornerButten(.....)	Get the Corner Harris point
	Corner Harris Point Closely Remove(See Appendix A2)
CloudTriangleDrawingTriangluation(......)	First of Drawing Triangle by Delaunay Triangle
	First Calculate to In-radius and In-center
	Remove Same Cloud Triangle
	Choice Triangle by Threshold Inradius
	Second of Drawing Triangle
	Second Calculate Inradius and Incenter(See Appendix A4)
CloudModel1DCloudDrops(......)	Cloud model Generator to Cloud Droplets(See Appendix A1.1)
	Remove to the cloud droplets and Descending order
	Cloud droplets base on referent point location
HideDataEmbededProcessing(......)	Change the Cloud droplets location of pixel value (See Appendix A4)

Figure 3.11: Function cloud watermarking Embedding Processing

There are 5 public methods in the function of watermarking embedding processing is shown in Figure 3.11. At first, I will use the method *OpenMenuItem_Click( )* to get the list which include the pixel value and location in the text document image. Then use the method find *HarrisCornerButten( )* by corner Harris point and corner Harris point closely remove, the next *CloudModel1DCloudDrops( )* to find the "embedding" pixel of point location, including generator cloud droplets ,remove the cloud droplets location and "embedding" pixel of point location based on referent point, the final result is the "embedding" pixel of point location. Then I will use the method *HideDataEmbededProcessing( )* to embedding data hiding to original text document images. The full code is in the Appendix A4 and Figure 3.12 shown cloud watermarked image.

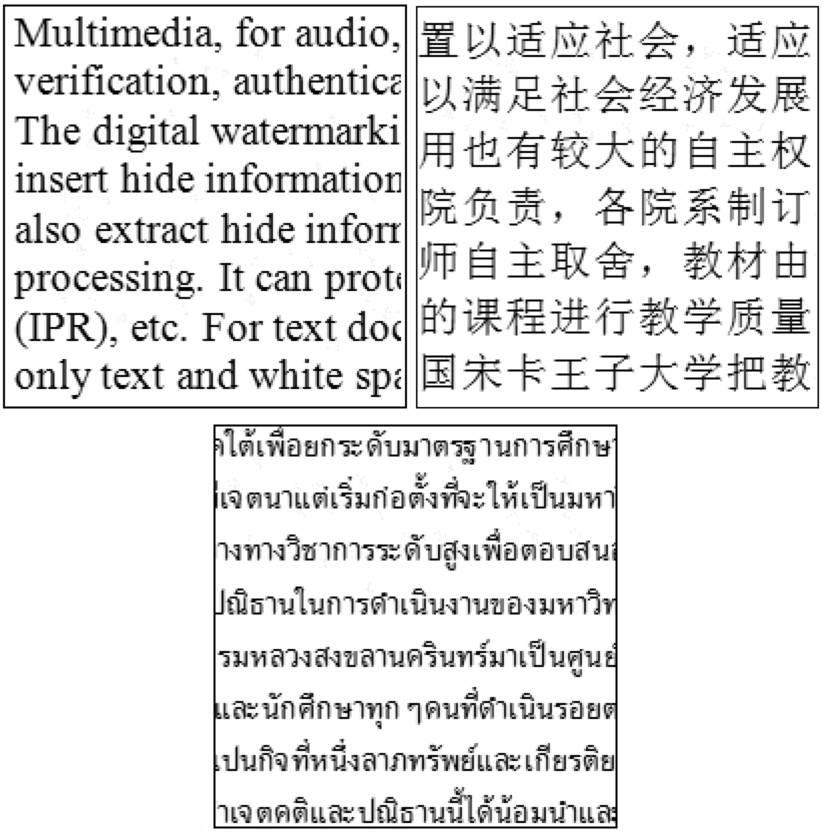


Figure 3.12: Cloud Watermarked Text Document Image

3.2.2 Watermarking Detection

Cloud Watermark detection is a process of detecting the inserted watermark to prove the ownership. In order to extract the watermark from the watermarked image, pixel value changes are detected form the cloud drops. Also, each pixel value is transformed to the cloud drops with Equation 3.8. Here, $f_w(x, y)$ is pixel value at location (x, y) in watermarked image, w denote a bit data of Unicode hiding information. Then the watermark is extracted with the following condition:

$$w = \begin{cases} 0, & IF f_w(x, y) \bmod S_v \le \dfrac{S_v}{2} \\ 1, & IF f_w(x, y) \bmod S_v > \dfrac{S_v}{2} \end{cases} \tag{3.8}$$

As an identical watermark is used for all “embedding” pixels location within incircle of a triangle in watermarked image. The cloud watermarking of text document images are extracted through the follows steps:

1) Load an image with text document as a watermarked image. Let the watermarked image size $m \times n$ denoted as $F = \{f(x, y), 1 \le x \le m, 1 \le y \le n\}$,

where $f(x,y)=\{0,1,2,...,2^l-1\}$ represent pixel value at location (x,y) of the watermarked image.

2) Finding feature point by Harris corner detection algorithm, and generate triangle by Delaunay triangulation algorithm for detection to location of pixels value in the watermarked image. Calculate minimum and maximum threshold of inradius.
3) By inradius threshold, choosing the suitable triangle.
4) Using cloud model generator to cloud drops in the every triangle. Then, we have been to ensure "Watermarked" pixel location.
5) The extraction information data of watermarked from watermarked image.
6) Check the block cloud index, data size length and data length. When DSHI block is OKAY each cloud, remove the block of cloud index and data size. Save to data each cloud.
7) Process the next cloud model.
8) Repeat steps 3). Then, combined all DSHIs to Unicode hide information message.

If all "embedding" pixel locations are found and all parts of the watermark are collected, the whole large hide information can be reconstructed. This reconstruction can be shown in Figure3.13.

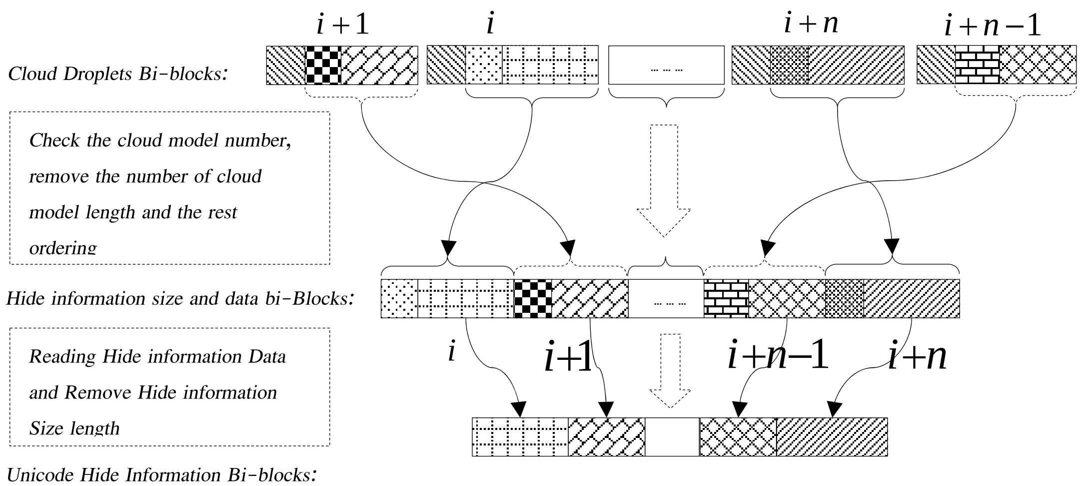


Figure 3.13: How to obtain Hiding Information extraction process

I wrote 6 functions for extraction hiding information data form cloud watermarked image in the extraction process. They have extraction processing purpose of obtain to data hiding information, In order data hiding information by pixel value image (see Equation 3.8).Then, they have obtain that data hiding information as shown in Figure 3.14. There are 6 functions for build watermarking extraction processing, it is shown the public methods it. And the full code of the function is in Appendix A5.

Method	Process
OpenMenuItem (……)	Get the original text document image with processing pixel value and location(See Appendix A4)
HarrisCornerButten(…..)	Get the Corner Harris point
	Corner Harris Point Closely Remove(See Appendix A2)
CloudTriangleDrawingTriangluation(……)	First of Drawing Triangle by Delaunay Triangle(See Appendix A3)
	First Calculate to Inradius and Incenter
	Remove to Cloud Triangle
	Choice Triangle by Threshold Inradius
	Second of Drawing Triangle
	Second Calculate Inradius and Incenter (See Appendix A5)
CloudModel1DCloudDrops(……)	Cloud model Generator to Cloud Droplets (See Appendix A1.1)
	Remove to the cloud droplets and Descending order
	Cloud droplets base on referent point location(See Appendix A5)
HideDataExtractionProcessing(……)	Get data hide information see Equation 3.8
	Check in corresponding cloud model number, Then read the hiding information size in the every cloud model
	Unicode information(See Appendix A5)

Figure 3.14: Function 1-D cloud watermarking extraction processing

3.3 2-D Cloud Watermarking of Text Document Image Scheme

In the previous section, a novel 1-D cloud watermarking scheme is proposed, which is designed based on many parameters, including visual quality, robustness and capacity analysis. In this section, we proposed cloud watermarking base on 2-D cloud model technique scheme can be improved capacity of data hiding information. Figure 3.15 shows the overall framework of the proposed scheme.

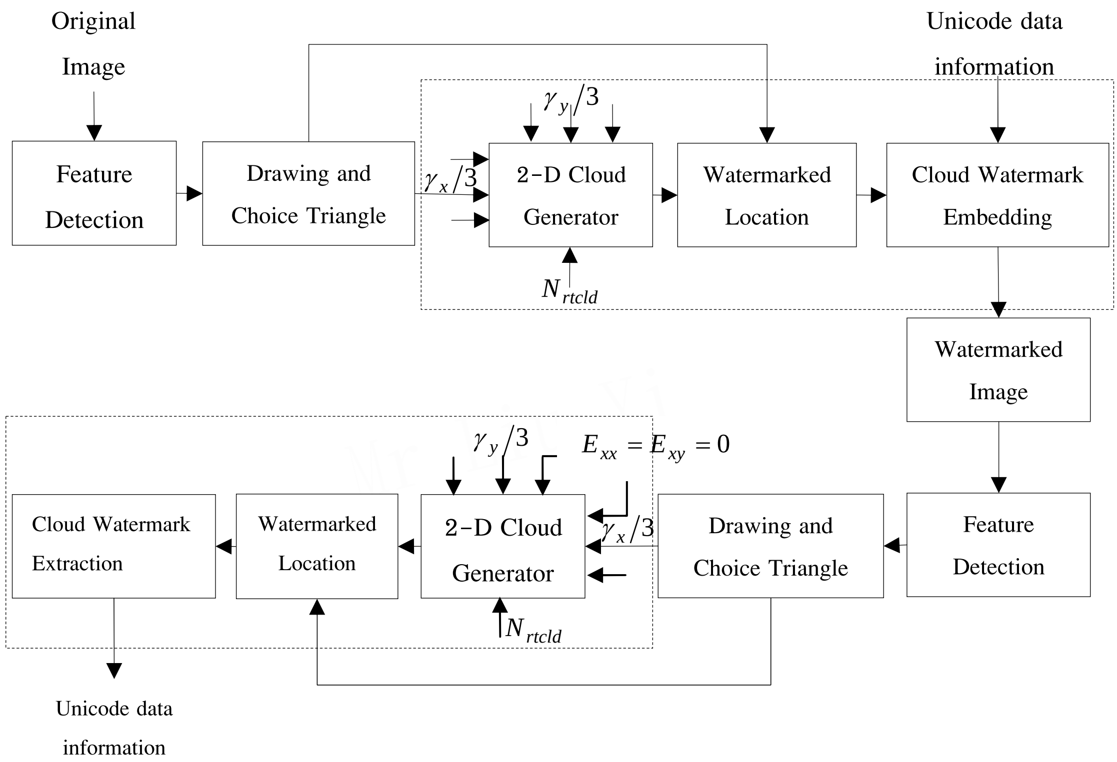


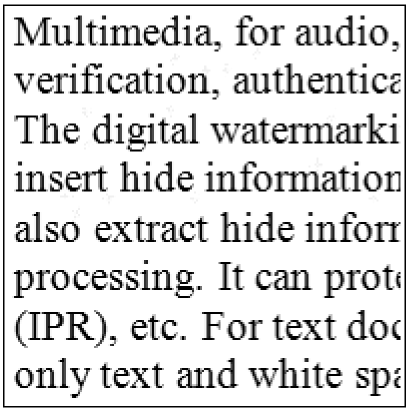
Figure 3.15: Framework of 2-D cloud watermarking of text document image system

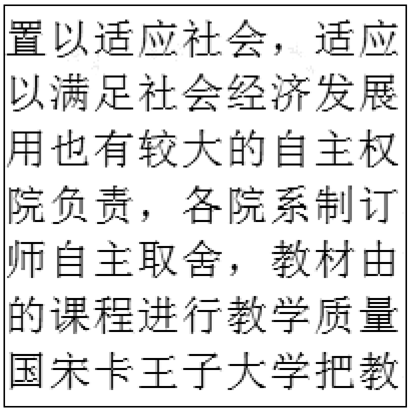
The 2-D cloud model technique detail has seen in the section 2.7.2 and section 2.7.3. And cloud droplets generation equation 2.9 section 2.7.2. I wrote functions for embedding process based on 2-D cloud model technique, as shown in Figure 3.16.

Embedding Processing (Appendix A4)
+ OpenMenuItem_Click (……)
+ HarrisCornerButten(…..)(See Appendix A2)
+ CloudTriangleDrawingTriangluation(……)(See Appendix A4)
+ CloudModel2DCloudDrops(……) (See Appendix A1.2)
+ HideDataEmbededProcessing(……)(See Appendix A4)

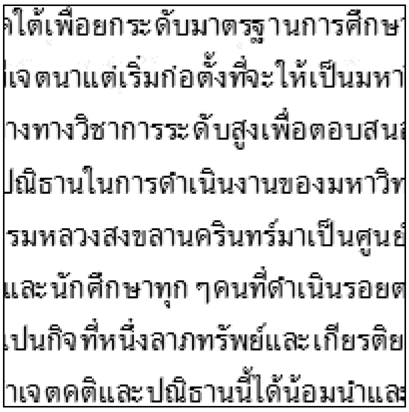
Figure 3.16: Function 2-D Cloud Watermarking Embedding Processing

The function *CloudModel2DCloudDrops(……)* describe the cloud which is the angle between the corresponding axes digital characteristic θ is *ZERO*. In other words, this cloud called un-rotated cloud or standard cloud. And Figure 3.17 shows 2-D cloud watermarking text document different images with size 200x200.


Multimedia, for audio,
verification, authentica
The digital watermarki
insert hide information
also extract hide inforr
processing. It can prot
(IPR), etc. For text doc
only text and white spa


置以适应社会，适应
以满足社会经济发展
用也有较大的自主权
院负责，各院系制订
师自主取舍，教材由
的课程进行教学质量
国宋卡王子大学把教

(a) Cloud Watermarked English Image (b) Cloud Watermarked Chinese Image


าได้เพื่อยกระดับมาตรฐานการศึกษ
เจตนาแต่เริ่มก่อตั้งที่จะให้เป็นมหา
างทางวิชาการระดับสูงเพื่อตอบสน
ปณิธานในการดำเนินงานของมหาวิท
รมหลวงสงขลานครินทร์มาเป็นศูนย์
และนักศึกษาทุก ๆคนที่ดำเนินรอยต
เป็นกิจที่หนึ่งลาภทรัพย์และเกียรติย
าเจตคติและปณิธานนี้ได้น้อมนำและ

(c)Cloud Watermarked Thailand Image

Figure 3.17: 2-D Cloud Watermarked of Different Language Text Document Image

3.4 Theoretical Analysis

In this section, Threshold of Inradius (T_γ) and analysis of S_V , S_f of the proposed watermarking scheme. The performance of the proposed algorithm is calculated. Let N_{tcld_i} be the total number of cloud droplets in the i th cloud, N_{rtcld_i} the total number of real cloud droplets in the text document, $n \times m$ be the size of the text document image and γ_i be inradius of incircle of the i th triangle, N_{tri} be choice the total number of suitable "embedding" triangles.

Size of Inradius(γ_i): see *Equation 3.1 in the Section 3.13.*

Size of cloud droplets (N_{tcld_i}): $0.3 \times \pi \times \gamma_i \times \gamma_i$.

Size of real cloud droplets for embedding pixel location (N_{rtcld_i}) less than N_{tcld_i} .

Size of hiding information binary blocks in the every cloud model: N_{rtcld_i} .

3.4.1 Analysis Threshold of Inradius of Incircle of Cloud Triangle

The selection of the threshold of inradius T_γ is important for the quality of the output watermarked image, watermarking robustness and capacity, and the hide information data in the extraction. We first calculate the average of the triangle in the cover image. Then we set the threshold of minimize inradius for watermarking capacity and robustness. And set the threshold of maximum inradius for capacity, robustness and quality visual.

To improve the performance and data hiding capacity, I have get the Threshold of minimum inradius length ($T_{r\min}$) and maximum inradius length ($T_{\gamma\max}$), ratio of threshold in inradius(RTI) function (Δ_T):

$$T_{\gamma\min} = \gamma_{ave} + \alpha \times \sigma_\gamma \tag{3.9}$$

$$T_{\gamma\max} = \gamma_{ave} + \beta \times \sigma_\gamma \tag{3.10}$$

$$\Delta_T = \frac{T_{\gamma\max} - T_{\gamma\min}}{\gamma_{\max} - \gamma_{\min}} \tag{3.11}$$

where γ_{ave} is expected value of inradius in the image, σ_γ is variance of inradius in the image, $\gamma_{\max}, \gamma_{\min}$ are inradius of maximum length and minimum length, respectably, and α , β are constant and α less than β in the image.

In our experiment, Table 3.4, 3.5 and 3.6 shown in RTI data in the Multilanguage (e.g.: English, Chinese and Thailand) image with size 200x200.

Table 3.4: RTI Data in the English Text Image

		English Text Image Font Size(Inch):									Average Capacity(bits)
		12	14	16	18	20	22	24	26	28	
Ratio of Threshold Inradius (%)	10	38	54	59	92	40	35	82	95	66	62
	20	44	79	85	135	72	99	181	196	269	129
	30	59	104	110	231	112	149	233	284	277	173
	40	92	114	133	264	189	222	273	300	340	214
	50	99	114	160	280	225	237	306	300	417	238
	60	112	140	199	311	251	272	329	345	441	267
	70	122	151	199	311	282	272	329	401	525	288
	80	122	151	210	340	300	296	356	433	525	304
	90	122	151	223	340	300	296	356	473	525	310

Table 3.5: RTI Data in the Chinese Text Image

		Chinese Text Image Font Size(Inch):									Average Capacity(bits)
		12	14	16	18	20	22	24	26	28	
Ratio of Threshold Inradius (%)	10	21	125	226	132	55	91	100	157	150	117
	20	30	189	382	222	86	211	210	290	297	213
	30	43	306	420	257	121	257	330	352	353	271
	40	58	378	435	257	137	306	400	436	437	316
	50	94	386	435	283	173	376	446	474	465	348
	60	137	400	435	283	203	376	446	495	529	367
	70	148	400	435	306	222	428	446	527	529	382
	80	159	447	435	359	265	485	446	527	625	416
	90	159	447	471	359	287	485	473	556	625	429

Table 3.6: RTI Data in the Thailand Text Image

		Thailand Text Image Font Size(Inch):									Average Capacity(bits)
		12	14	16	18	20	22	24	26	28	
Ratio of Threshold Inradius (%)	10	96	38	89	14	95	40	21	35	99	59
	20	123	113	117	32	134	62	63	139	156	104
	30	155	159	151	46	176	131	127	205	185	148
	40	190	204	219	61	207	198	185	267	291	202
	50	201	254	274	78	251	226	238	325	426	253
	60	201	276	338	98	269	288	305	325	458	284
	70	201	303	362	128	282	302	326	358	523	309
	80	214	303	426	139	297	302	326	358	568	326
	90	214	303	426	163	313	302	354	412	568	339

And compare of watermark system capacity of RTI data in Multilanguage text document image shown in Figure 3.18. So they have been RTI data equal to 90% in the digital watermarking system.

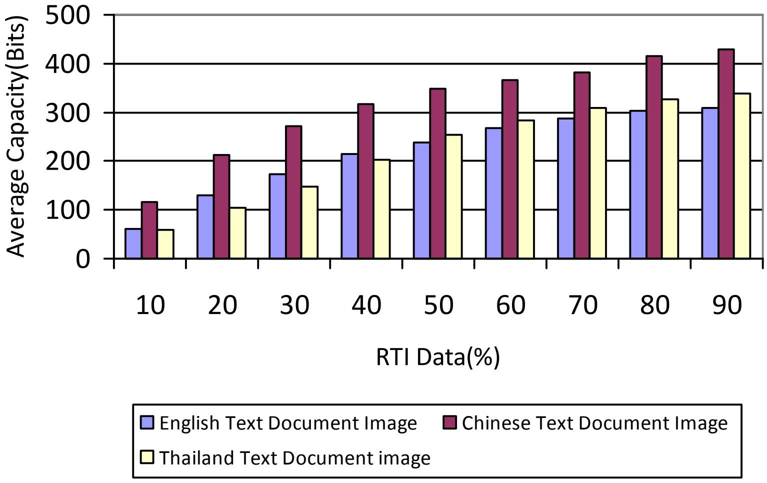


Figure 3.18: Ratio of Threshold Radius Data

In probability theory, we seek a function of a inradius variable to correspond to mean. If in N_{tri} trials the inradius variable takes values $\gamma_1, \gamma_2, \ldots \gamma_{N_{tri}-1}, N_{tri}$, then the mean of inradius value (γ_{ave}) is:

$$\gamma_{ave} = \frac{1}{N_{tri}}\left(\gamma_1 + \gamma_2 + \ldots + \gamma_{N_{tri}-1} + \gamma_{N_{tri}}\right)$$

$$= \frac{1}{N_{tri}}\sum_{i=1}^{N_{tri}}\gamma_i \quad (3.12)$$

And the variance of inradius(σ_γ) function:

$$\sigma_\gamma = \sqrt{\frac{1}{N_{tri}}\left\{(\gamma_1 - \gamma_{ave})^2 + (\gamma_2 - \gamma_{ave})^2 + \ldots + \left(\gamma_{N_{tri}-1} - \gamma_{ave}\right)^2 + \left(\gamma_{N_{tri}} - \gamma_{ave}\right)^2\right\}}$$

$$= \sqrt{\frac{1}{N_{tri}}\sum_{i=1}^{N_{tri}}(\gamma_i - \gamma_{ave})^2} \quad (3.13)$$

Then, according to least minimum of DSHI is $\lceil \log_2 N_{tri} \rceil + 2$ bit, so give the α minimum value is:

$$\gamma_{ave} + \alpha_{\min} \times \sigma_\gamma \geq \sqrt{\frac{\lceil \log_2 N_{tri} \rceil + 2}{\pi}} \Rightarrow \alpha_{\min} \geq \frac{1}{\sigma_\gamma}\left(\sqrt{\frac{\lceil \log_2 N_{tri} \rceil + 2}{\pi}} - \gamma_{ave}\right) \quad (3.14)$$

And the β maximum value is:

$$\gamma_{ave} + \beta_{\max} \times \sigma_{\gamma} \le \gamma_{\max} \Rightarrow \beta_{\max} \le \frac{1}{\sigma_{\gamma}}(\gamma_{\max} - \gamma_{ave}) \tag{3.15}$$

Finally, the constants of α , β using equation 3.14 and equation 3.15 in according to watermarking capacity and robustness in the digital cloud watermarking system. The final range of function is:

$$\frac{1}{\sigma_{\gamma}}\left(\sqrt{\frac{\lceil \log_2 N_{tri} \rceil + 2}{\pi}} - \gamma_{ave}\right) < \alpha < \beta < \frac{1}{\sigma_{\gamma}}(\gamma_{\max} - \gamma_{ave}) \tag{3.16}$$

I use the ratio of hiding information length (equation 3.6 in section 3.14) be calculate to constant α . As a parameter of the mean of inradius value (equation 3.12), the final function is:

$$\begin{aligned} \forall \quad \Delta &= \sum_{i=1}^{N_{tri}} \frac{M_i}{N_{rtcld_i}} \times 100\% \\ &\approx \frac{M}{0.3 \times \pi \times T_{\min}^{2}} \times 100\% \\ &= \frac{M}{0.3 \times \pi \times (\gamma_{ave} + \alpha \times \sigma_{\gamma})^2} \times 100\% \\ &= 1 - \frac{\lceil \log_2 N_{tri} \rceil + 1}{0.3 \times \pi \times (\gamma_{ave} + \alpha \times \sigma_{\gamma})^2} \end{aligned}$$

$$\therefore \alpha = \frac{1}{\sigma_{\gamma}}\left(\sqrt{\frac{\lceil \log_2 N_{tri} \rceil + 1}{0.3 \times \pi \times (1 - \Delta)}} - \gamma_{ave}\right) \quad , \ \gamma_{ave} > \lceil \log_2 N_{tri} \rceil + 2 \tag{3.17}$$

So, they have been get threshold of minimum inradius length ($T_{r\min}$) function is:

$$T_{\gamma\min} = \begin{cases} \sqrt{\dfrac{\lceil \log_2 N_{tri} \rceil + 2}{\pi}} + 1 & ,IF \quad \gamma_{ave} \le \lceil \log_2 N_{tri} \rceil + 2 \\ \sqrt{\dfrac{\lceil \log_2 N_{tri} \rceil + 1}{0.3 \times \pi \times (1 - \Delta)}} & ,IF \quad \gamma_{ave} > \lceil \log_2 N_{tri} \rceil + 2 \end{cases} \tag{3.18}$$

where N_{tri} is choice to number of "embedding" triangle in the original/watermarked image, γ_{ave} is the mean of inradius value (equation 3.12), Δ is the minimum ratio of hiding information length(equation 3.6 in section 3.14).

In order to improve that watermarking robustness, capacity and quality visual, I combine equation 3.11 to get the function, calculate to constant β . The final function is:

$$\forall \quad \Delta_T = \frac{T_{\gamma\max} - T_{\gamma\min}}{\gamma_{\max} - \gamma_{\min}}$$

$$= \frac{(\gamma_{ave} + \beta \times \sigma_\gamma) - (\gamma_{ave} + \alpha \times \sigma_\gamma)}{\gamma_{\max} - \gamma_{\min}}$$

$$= \frac{(\beta - \alpha) \times \sigma_\gamma}{\gamma_{\max} - \gamma_{\min}}$$

$$\therefore \quad \beta = \frac{\Delta_T \times (\gamma_{\max} - \gamma_{\min})}{\sigma_\gamma} + \alpha$$

$$= \frac{\Delta_T \times (\gamma_{\max} - \gamma_{\min})}{\sigma_\gamma} + \frac{1}{\sigma_\gamma}\left(\sqrt{\frac{\lceil \log_2 N_{tri} \rceil + 1}{0.3 \times \pi \times (1 - \Delta)}} - \gamma_{ave}\right)$$

$$= \frac{1}{\sigma_\gamma} \times \left(\Delta_T \gamma_{\max} - \Delta_T \gamma_{\min} - \gamma_{ave} + \sqrt{\frac{\lceil \log_2 N_{tri} \rceil + 1}{0.3 \times \pi \times (1 - \Delta)}}\right) \qquad (3.19)$$

Finally, they have been get the threshold of maximum inradius length ($T_{\gamma\max}$) function is:

$$T_{\gamma\max} = \begin{cases} \Delta_T \gamma_{\max} - \Delta_T \gamma_{\min} + \sqrt{\frac{\lceil \log_2 N_{tri} \rceil + 2}{\pi}} + 1 & IF \quad \gamma_{ave} \le \lceil \log_2 N_{tri} \rceil + 2 \\ \Delta_T \gamma_{\max} - \Delta_T \gamma_{\min} + \sqrt{\frac{\lceil \log_2 N_{tri} \rceil + 1}{0.3 \times \pi \times (1 - \Delta)}} & IF \quad \gamma_{ave} > \lceil \log_2 N_{tri} \rceil + 2 \end{cases} \qquad (3.20)$$

where N_{tri} is choice to number of "embedding" triangle in the original/watermarked image, γ_{ave} is the mean of inradius value (equation 3.12), Δ is the minimum ratio of hiding information length(equation 3.6 in section 3.14), Δ_T is the least ratio of threshold length in inradius(equation 3.11).

3.4.2 Analysis S_V and S_f in Cloud Watermarking System

In this section, we have been using equation 3.7(in section 3.2.2) and equation 3.8(in section 3.2.3) to give their rules in cloud watermarking system.

RULE:

1. All pixel value of original image is integer;
2. Pixel value Threshold maximum is 255;
3. $\frac{S_v}{2}$ is integer;
4. $S_v * S_f$ or $S_v * (1 - S_f)$ is integer;

And they have been giving Figure 3.19 shown general flow analysis S_v and S_f frameworks process in the cloud watermarking of text document image.

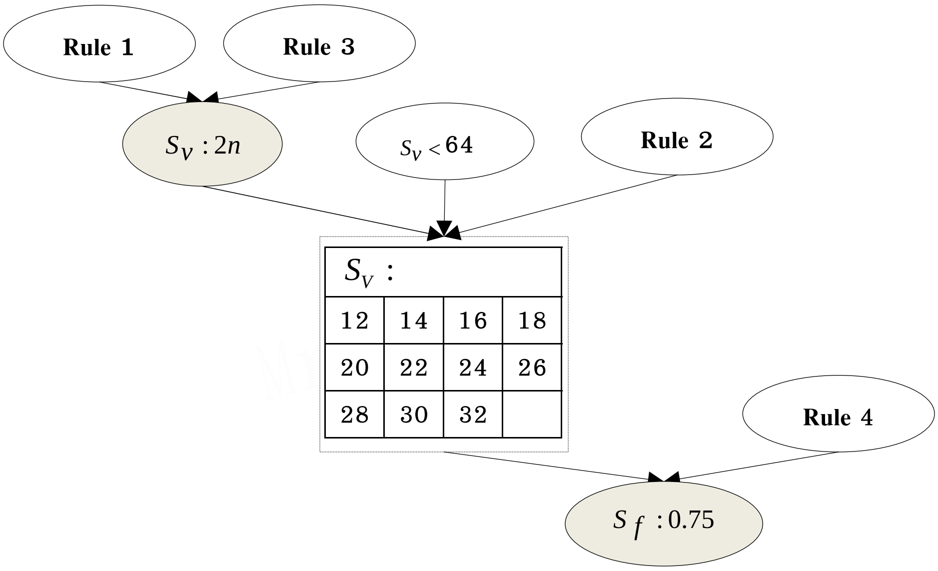
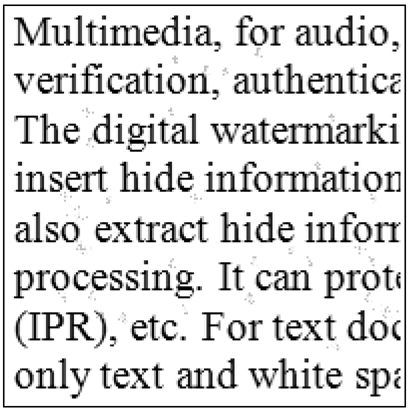
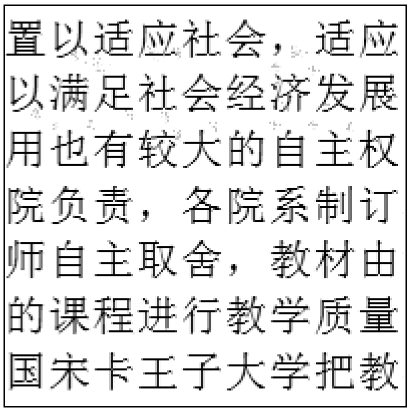


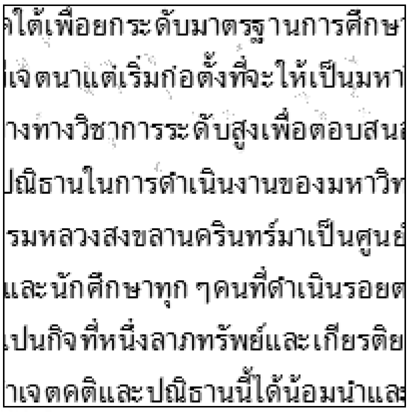
Figure 3.19: General Flow Analysis Sv and S_f Frameworks process

First, they have been digital cloud watermarking in the invisible watermarking but not visible watermarking. At the same time, in our experiment, they have been result $S_{v_{\max}} < 64$. If $S_{v_{\max}} \geq 64$, they have been working in the visible cloud watermarking. Figure 3.20 shown that cloud watermarked for English, Chinese and Thailand text document image with $S_{v_{\max}}$ equal to 64.





(a) Watermarked English Image (b) Watermarked Chinese Image



(c) Watermarked Thailand Image

Figure 3.20: Different language image cloud watermarked of text documents with S_V is 64

Then according to Rule3, S_V equal to $2n$, $n = 0,1,2,\ldots,31,32$. in the extraction processing. Let's Δ_{error} is the Error bi-block in the DSHI.

The pixel value threshold a function: $$T = S_V * \left\lfloor \frac{255}{S_V} \right\rfloor \qquad (3.21)$$

Lower Threshold Value: $$T_{\min} = S_V * \left\lfloor \frac{255}{S_V} \right\rfloor - \frac{S_V}{2}$$

$$= T - \frac{S_V}{2} \qquad (3.22)$$

Upper Threshold Value: $$T_{\max} = S_V * \left\lfloor \frac{255}{S_V} \right\rfloor + \frac{S_V}{2}$$

$$= T + \frac{S_V}{2} \qquad (3.23)$$

Then define as the finally function is:

$$\Delta_{error} = \begin{cases} \dfrac{T_{\max} - 255}{S_v/2} \times 100\%, & T_{\max} \geq 255 \\[2ex] \dfrac{S_v/2 + [T_{\max} - 255]}{Sv/2} \times 100\%, & T_{\max} < 255 \end{cases} \tag{3.24}$$

In the experimental results, listed in Table 3.7, describes that all data of stego value S_v with error bi-block in the invisible watermarking. These results were obtained processing the text document images with the suitable data of S_v. It results for improve data hiding capacity, visual quality and robustness in the digital watermarking system.

Table 3.7: All data of Stego Values with error bi-blocks in the DSHI

S_v	2	4	6	8	10	12	14	16
$T = S_v \times \lfloor 255/S_v \rfloor$	254	252	252	248	250	252	252	240
$T_{\min} = T - S_v/2$	253	250	249	244	245	246	245	232
$T_{\max} = T + S_v/2$	255	254	255	252	255	258	259	248
Δ_{error} (%)	0.00	50.00	0.00	25.00	0.00	50.00	57.14	12.50

S_v	18	20	22	24	26	28	30	32
$T = S_v \times \lfloor 255/S_v \rfloor$	252	240	242	240	234	252	240	224
$T_{\min} = T - S_v/2$	243	230	231	228	221	238	225	208
$T_{\max} = T + S_v/2$	261	250	253	252	247	266	255	240
Δ_{error} (%)	66.67	50.00	81.82	75.00	38.46	78.57	0.00	6.25

S_v	34	36	38	40	42	44	46	48
$T = S_v \times \lfloor 255/S_v \rfloor$	238	252	228	240	252	220	230	240
$T_{\min} = T - S_v/2$	221	234	209	220	231	198	207	216
$T_{\max} = T + S_v/2$	255	270	247	260	273	242	253	264
Δ_{error} (%)	0.00	83.33	57.89	25.00	85.71	40.90	91.30	37.50

Table 3.7: All data of Stego Values with error bi-blocks in the DSHI (CONT.)

S_v	50	52	54	56	58	60	62	64*
$T = S_v \times \left\lfloor {255}/{S_v} \right\rfloor$	250	208	216	224	232	240	248	192
$T_{\min} = T - {S_v}/{2}$	225	182	189	196	203	210	217	160
$T_{\max} = T + {S_v}/{2}$	275	234	243	252	261	270	279	224
Δ_{error} (%)	80.00	19.23	55.55	89.29	20.69	50.00	77.42	3.13

*When S_v =64, the cloud watermarking system is visible watermarking.

In order to data hiding capacity high, they have been stego value S_v more than 12, but if stego value S_v is small data, they have effect to robustness and S_v is big data, it lead to visual quality low, shown in Figure3.21.

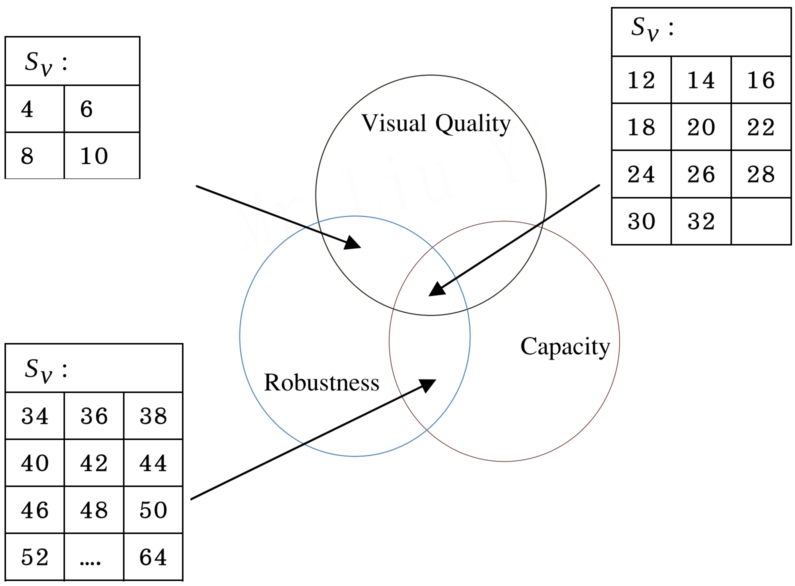


Figure 3.21: How much S_v in the cloud watermarking with S_f is 0.75

Then, they have been sure parameter of S_f equal to 0.75. Figure 3.22 shows how error bi-blocks of DSHI in the stego factor S_f. In our experiment, they have tested different text document image, the result that stego value 16 and 32, but they have two stego values for different Dots Per Inch (DPI) printer and robustness is different. We have been general stego value S_v equal to 16 in this thesis and 600dpi printer. If they have been improve watermarking robustness, the stego value S_v equal to 32 and 150 dpi printer.

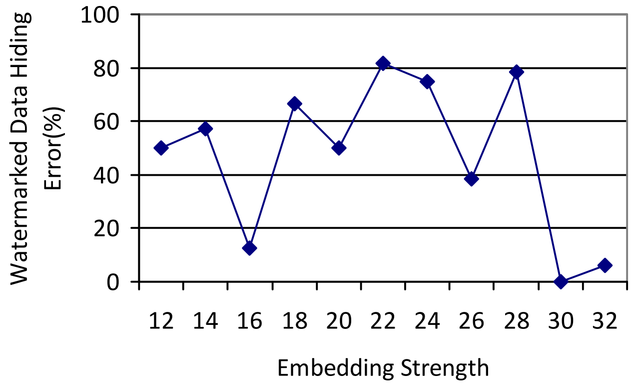


Figure 3.22: How error bi-blocks of DSHI when different S_V in the S_f 0.75

3.4.3 Capacity

In this section, we investigate the watermarking capacity based triangle masking effects. Capacity of the watermark is defined as how much information can be carried by the watermark when is embedded in a cover image. Each coefficient is considered as an independent random variable with its own noise distribution.

Here, let's assume $M_1, M_2, ..., M_{N_{tir}}$ be the changes of the cloud droplets due to watermarking and $S_1, S_2, ...S_{N_{tri}}$ be source coefficient pixel values in the watermarking. We define a masking function f . $E(XX^T) \leq f(S)$ where $X = [M_1, M_2, ..., M_{N_{tri}}]^T$ and $S = [S_1, S_2, ..., S_{N_{tri}}]^T$. In the receiver, consider $Y = S_W - S = X - Z$ where Z the noises are added to the coefficients during transmission. Then, the maximum capacity of these multi-variant symbols in Equation 3.25. We can assume X and Z are independent.

$$C = Max_{p(x):E(XX^T) \leq f(S)} I(X; Y) \tag{3.25}$$

$$= Max_{p(x)}[h(Y) - h(Y|X)] \tag{3.26}$$

$$= Max_{p(x)}[h(Y) - h(Z)] \tag{3.27}$$

where C represents data hiding information capacity. $p(.)$ represents any probability distribution, $I(.;.)$represents mutual information and $h(.)$ represents the differential entropy.

According to Theorem 9.6.5 in Elements of Information Theory [95], Y has zero mean and covariance $K = E(XX^T)$, the differential entropy of Y , i.e., $h(Y)$ satisfies the following

$$h(Y) \leq \frac{1}{2}\log(2\Pi \exp)^n |K| \tag{3.28}$$

with equality iff $Y \sim N(0,K)$ and $|.|$ is the absolute value of the determinant. Here, this theorem is valid no matter what the range of K . Therefore, from Equation3.25 and $\left|E(XX^T)+E(ZZ^T)\right| = \left|f(S)+E(ZZ^T)\right|$ we can see that

$$C=\frac{1}{2}\log(2\Pi\exp)^n\left|f(S)+E(XX^T)\right|-h(Z) \qquad (3.29)$$

This assumption means that embedded watermark values are mutually independent. Equation 3.29 is the watermarking capacity in a variant-state channel without specifying any type of noise. It is he capacity given a noise distribution. If we look at Equation 3.29 and Theorem 9.6.5 in [95] again, for all type of noise, we can find that C will be at least

$$C_{\min}=\frac{1}{2}\log(2\Pi\exp)^n\left|f(S)+E(ZZ^T)\right|-\frac{1}{2}\log(2\Pi\exp)^n\left|E(ZZ^T)\right| \qquad (3.30)$$

$$=\frac{1}{2}\left|f(S)+E(ZZ^T)^{-1}+I\right| \qquad (3.31)$$

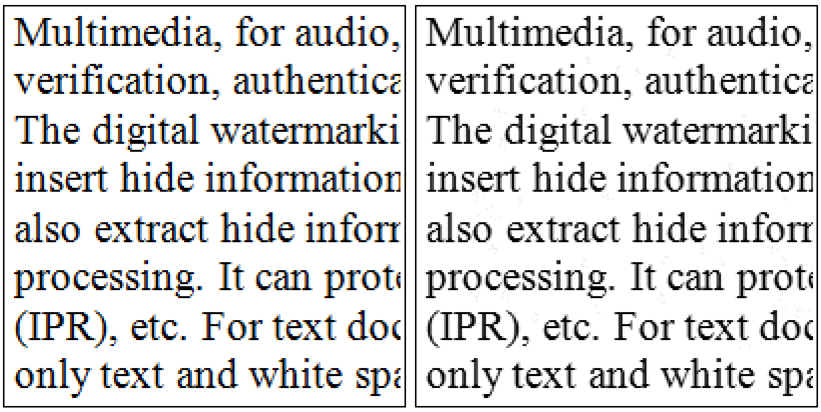
when noise is Gaussian distribution. If we assume that noise is also independent in samples, then the watermarking capacity will be:

$$C_{\min}=\sum_{i=1}^{n}\frac{1}{2}\log\left(1+\frac{P_i}{N_i}\right) \qquad (3.32)$$

$$=\sum_{i=1}^{n}\frac{1}{2}\log\left(1+\frac{P_i}{\sigma_n^2}\right) \qquad (3.33)$$

where P_i and N_i are the power constrains in the i th coefficient, respectively.

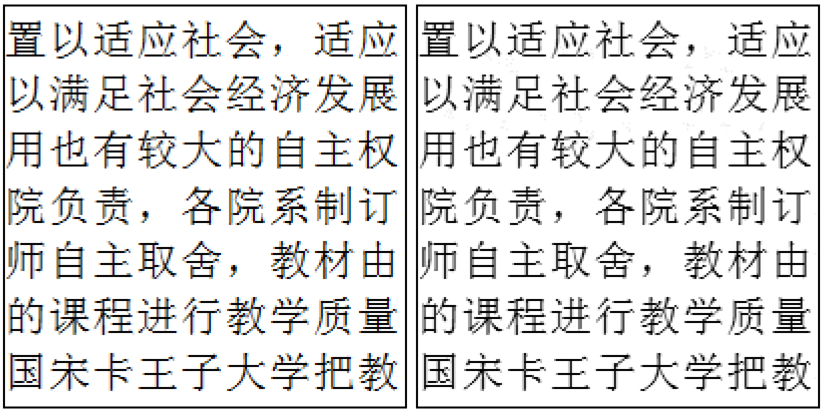
In our experiment, they have been tested watermarking of English, Chinese and Thailand Text document image, respectively. Figure 3.23 that watermarked capacity for English document image, Chinese document image and Thailand document image.



Multimedia, for audio,
verification, authentica
The digital watermarki
insert hide information
also extract hide inforr
processing. It can prote
(IPR), etc. For text doc
only text and white spa

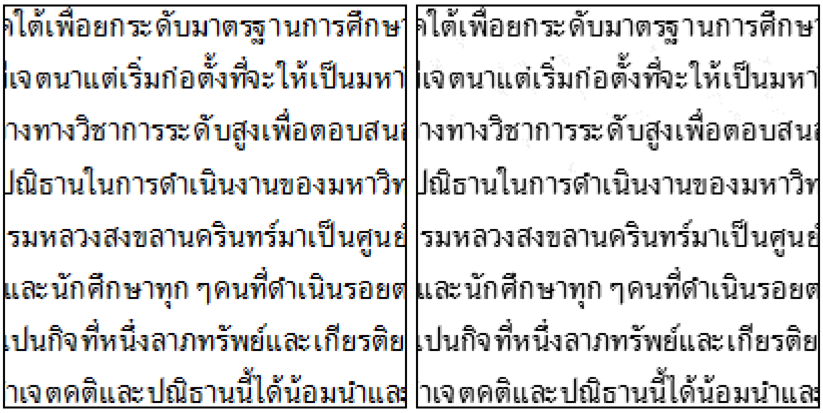
Multimedia, for audio,
verification, authentica
The digital watermarki
insert hide information
also extract hide inforr
processing. It can prote
(IPR), etc. For text doc
only text and white spa

(a) English Text Document Image (b) Watermarked English document image with Capacity data 312bits


置以适应社会，适应
以满足社会经济发展
用也有较大的自主权
院负责，各院系制订
师自主取舍，教材由
的课程进行教学质量
国宋卡王子大学把教

置以适应社会，适应
以满足社会经济发展
用也有较大的自主权
院负责，各院系制订
师自主取舍，教材由
的课程进行教学质量
国宋卡王子大学把教

(c) Chinese Text Document Image (d) Watermarked Chinese documents image with Capacity data 483 bits


ได้เพื่อยกระดับมาตรฐานการศึกษ
เจตนาแต่เริ่มก่อตั้งที่จะให้เป็นมหา
างทางวิชาการระดับสูงเพื่อตอบสน
ปณิธานในการดำเนินงานของมหาวิท
รมหลวงสงขลานครินทร์มาเป็นศูนย์
และนักศึกษาทุก ๆคนที่ดำเนินรอยต
ปนกิจที่หนึ่งลาภทรัพย์และเกียรติย
าเจตคติและปณิธานนี้ได้น้อมนำและ

ได้เพื่อยกระดับมาตรฐานการศึกษ
เจตนาแต่เริ่มก่อตั้งที่จะให้เป็นมหา
างทางวิชาการระดับสูงเพื่อตอบสน
ปณิธานในการดำเนินงานของมหาวิท
รมหลวงสงขลานครินทร์มาเป็นศูนย์
และนักศึกษาทุก ๆคนที่ดำเนินรอยต
ปนกิจที่หนึ่งลาภทรัพย์และเกียรติย
าเจตคติและปณิธานนี้ได้น้อมนำและ

(e) Thailand Text Documents Image (f) Watermarked Thailand Text Documents image with Capacity data 595 bits

Figure 3.23: How many capacity in the watermarked text document image

3.4.4 Robustness

In this section, the robustness of the 1-D cloud watermarking scheme is tested. For this purpose some text document images were watermarked. The detector output reveals the estimated embedded message as the one having the greatest correlation coefficient with the sample sequence extracted from the marked and possibly corrupted image.

Due to the constraint of imperceptibility, watermark embedding strength S_V and coding length, if error correcting coding (ECC) is applied, are conflicting with each other. Given a peak signal-to-noise ratio (PSNR), there is a trade-off between these two factors. For the unitary transform domain embedding algorithms using the following popular additive embedding equation:

$$f_{wi} = f_i + S_v * W_i, \qquad i = 1,2,...N_{rtcld} \tag{3.34}$$

we can derive the following inequality for the lower bound of PSNR of a marked image, T_{PSNR}.

$$T_{PSNR} \le 20\log_{10}\frac{PMN}{\sqrt{\sum_i (S_v \times W_i)^2}} \tag{3.35}$$

where the size of image is $M \times N$ and the maximum peak-to-peak signal swing is P. E.g. P is 255 for 8-bit images, S_v is the embedding strength, W_i's are watermark signals.it is clear that, give a lower bound of PSNR, T_{PSNR} , there exists a upper bound of embedding strength to ensure watermark imperceptibility.

Let X , Y and Z be random variables and defined as $X \in \left\{\frac{1}{4}, \frac{3}{4}\right\}$, $Y \in [0,1)$ and $Z \in \{f_{wi}\}, i = 1,2,...N_{rtcld}$ .and S_f is 0.75(Section 3.3.2). Then can be rewritten as

$$Z = (Y - X) * S \tag{3.36}$$

X and Y can be considered as independent, Then with equation3.7(section 3.2.2),we have equation 3.35 expressed as

$$T_{PSNR} \le 20\log_{10}\frac{PMN}{\sqrt{K * E(Z^2)}}$$

$$= 20\log_{10}\frac{PMN}{\sqrt{K * S_v^2 * \left[E\left(X^2\right) + E\left(Y^2\right) - 2E(X)E(Y)\right]}} \tag{3.37}$$

where K and S_v denote the length and strength of the watermark signal, respectively, $E(.)$ indicated the expectation operation.

Here, we discuss the problem by additive white Gaussian noise (AWGN).The detected watermarked signal can be modeled as follows:

$$r = q + \tau, v = r \bmod S_v \tag{3.38}$$

where q is a random variable representing embedded watermark, i.e., $q \in \left\{\frac{S_v}{4}, \frac{3Sv}{4}\right\}$; τ is the AWGN component, $\tau \sim N(0, \sigma^2)$; S_v is the embedding strength, r is the received signal, and v is decision variable. Then, the watermark bit is derived by comparing v with $\frac{S_v}{2}$.

When binary "0" is transmitted, the received signal is $r = q_0 + \tau = \frac{S_v}{4} + \tau$ .Similarly, when binary "1" is transmitted, the received signal is $r = q_1 + \tau = \frac{3S_v}{4} + \tau$. Here, the two conditional probability density functions (pdf) of r are

$$p(r|q_0)=\frac{1}{\sqrt{2\pi\sigma}}\exp\left\{-\frac{(r-S_V/4)^2}{2\sigma^2}\right\} \tag{3.39}$$

$$p(r|q_1)=\frac{1}{\sqrt{2\pi\sigma}}\exp\left\{-\frac{(r-3S_V/4)^2}{2\sigma^2}\right\} \tag{3.40}$$

It is the modulo S_V in equation 3.38 that makes error probability of watermarking different from the common communications. The received signal can be expressed as $r=kS_V+v$, where $k\in Z$. In fact, whenever r is in the intervals of $[(k+1/2)S_V,(k+1)Sv]$, the decision variable v will be greater than $S_V/2$, and consequently the decision is made in favor of $q=3S_V/4$. If $q=S_V/4$ was transmitted and the decision variable v was greater than or equal to $S_V/2$, false decision in watermark detection would occur. If $q=3S_V/4$ was transmitted and the decision variable v was less than $S_V/2$. False decion in watermark detection would also occur. We have

$$P\left(v\geq\frac{Sv}{2}\middle|q=\frac{S_V}{4}\right)=P\left(v<\frac{S_V}{2}\middle|q=\frac{3S_V}{4}\right) \tag{3.41}$$

So, the channel bit error rate is as follows.

$$\begin{aligned}P_b &= P(v\geq S_V/2|q=S_V/4)P(q=S_V/4)+P(v<S_V/2|q=3Sv/4)P(q=3S_V/4)\\ &=\frac{1}{\sqrt{2\pi\sigma}}\sum_{k=-\infty}^{\infty}\left[\int_{(k+0.5)S_V}^{(k+1)S_V}\exp\left(-\frac{(x-S_V/4)^2}{2*\sigma^2}\right)dx\right]\\ &=2*\sum_{k=0}^{\infty}\left[Q\left(\frac{(4k+1)S_V}{4\sigma}\right)-Q\left(\frac{(4k+3)S_V}{4\sigma}\right)\right]\end{aligned} \tag{3.42}$$

Obviously, the distribution of error regions here are different from that of general binary signals in AWGN, due to modulo operation in equation 3.38. Note that the Gaussian pdf has dropped closely to zero at three times standard deviation from the mean value. Hence, the above equation can be written as

$$P_b=2*\sum_{k=0}^{M}\left[Q\left(\frac{(4k+1)S_V}{4\sigma}\right)-Q\left(\frac{(4k+3)S_V}{4\sigma}\right)\right] \tag{3.43}$$

where M is a finite integer number.

Owing to the power constraint in digital watermarking, the energy per symbol E_c after using ECC with code ratio k/n satisfier the following equation [96]:

$$\frac{E_c}{N_0}=\left(\frac{k}{n}\right)\frac{E_b}{N_0} \tag{3.45}$$

Where N_0 is the variance of AWGN; E_b is the energy per bit before applying ECC. Thus, we use the below equation to calculate the ECC with coding ratio k/n :

$$P_b = 2 * \sum_{k=0}^{M} \left[Q\left(\frac{(4k+1)S_v}{4\sigma} \sqrt{\frac{k}{n}} \right) - Q\left(\frac{(4k+3)Sv}{4\sigma} \sqrt{\frac{k}{n}} \right) \right] \tag{3.46}$$

3.5 Summary

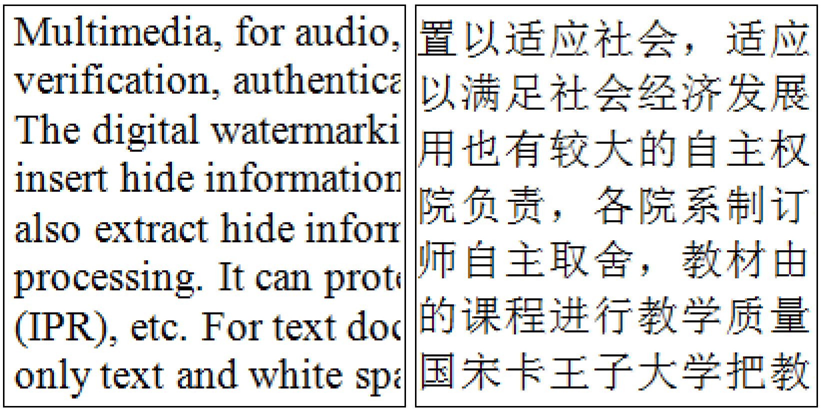
This chapter described that novel cloud watermarking of text document image scheme, including 1-D cloud watermarking scheme(section 3.2) and 2-D cloud watermarking scheme(section 3.3). And analysis data that threshold of Inradius (T_γ) (section 3.4.1), including minimum threshold of Inradius $T_{\gamma \min}$ see equation 3.18 and maximum threshold of Inradius $T_{\gamma \max}$ is see equation 3.20. And stego factor S_f is 3/4 (section 3.4.2), stego value/embedding strength S_v see Table 3.7 and Figure 3.22.

CHAPTER 4

EXPERIMENT RESULTS

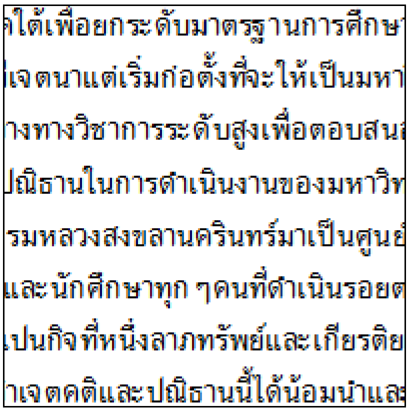
In Chapter 3, we developed two ways of digital watermarking, namely, cloud watermarking of text document images embedding processing and extraction processing in the current digital watermarking. In this chapter, we present in detail, the cloud watermarking of text document images experiment that we have conducted to test the performance of the different dimension cloud model implementation in the digital watermarking system. The motivation that drives the need for this experiment is the fact that the 1-D cloud model technique and 2-D cloud model (rotated cloud) technique produce watermarked image. The aim of this experiment is therefore: To visual quality and compare the performance of different dimension cloud model technique for information capacity. The chapter also explains in detail, the criteria for the selection of the "embedding" triangle by threshold of inradius, the process of generating of cloud droplets, and the analysis of the data obtained from the experiment.

In the following sections, we designed the image shown Figure 4.1 to be the original text document images used in the subjective. Three text documents have been used in the experiments, including one English, Chinese and Thailand Language text document image with size 200x200.


Multimedia, for audio,
verification, authentica
The digital watermarki
insert hide information
also extract hide inforr
processing. It can prote
(IPR), etc. For text doc
only text and white spa

置以适应社会，适应
以满足社会经济发展
用也有较大的自主权
院负责，各院系制订
师自主取舍，教材由
的课程进行教学质量
国宋卡王子大学把教

(a) English Original Image (b) Chinese Original Image


ใต้เพื่อยกระดับมาตรฐานการศึกษ
เจตนาแต่เริ่มก่อตั้งที่จะให้เป็นมหา
างทางวิชาการระดับสูงเพื่อตอบสน
ปณิธานในการดำเนินงานของมหาวิท
รมหลวงสงขลานครินทร์มาเป็นศูนย์
และนักศึกษาทุก ๆคนที่ดำเนินรอยต
เปนกิจที่หนึ่งลาภทรัพย์และเกียรติย
าเจตคติและปณิธานนี้ได้น้อมนำและ

(c) Thailand Original Image

Figure 4.1: Original Text Document Image with Different Language

It is important to design a watermarked image that based on 1-D cloud model technique and 2-D cloud model technique. Those cloud droplets generator are in section 2.7.3. The original image and watermarked images, which are printed on a piece of A4 quality paper using a HP Officejet 6500 Model E709C printer.

4.1 Experiment Cloud Watermarking System Parameters

The scope of this cloud watermarking of text documents image experiment has been to evaluate the increment in perceived using the cloud model technique proposed in Chapter 3. To this end, the goals of the experiments have been the following:

(1)The subjective evaluation of the number of cloud droplets is convergent and fog in the cloud model system. They have been constant of κ is 10 (See equation 2.13).

(2) The subjective evaluation of S_v controls the watermark embedding strength and should be as large as possible under the constraint of invisibility. It is visible digital watermarking processing in text document image, when embedding strength more or equal than 64. So, they have performed digital cloud watermarking system for text documents image with the embedding strength less than 64 and give other parameter of S_f . In our experiment, they have been parameter of S_v equal to 16 or 32. At the same time, the other parameter of Stego Factor $S_f = 3/4$ (Section 3.3.2).

(3)The subjective evaluation of choice "embedding" triangle is digital watermarking system improving capacity. They have been set of parameters: the RIHL data

(Δ) equal to 50 % (see Figure 3.8 in Section 3.14), the RTI data (Δ_T) equal to 90 %(see Figure 3.18 in Section 3.41).

4.2 Test on Capacity Cloud Watermarking of Text Document Image

We present our experimental results on the cloud watermarking of text documents image. The experiments are basically type: test on capacity of text document image with different Font size. In the following sections, we present the implementation detail of the proposed schemes with original image size 160x160 and the experimental results.

In this section, the capacity of cloud watermarking of text document image with different Font Size and Multilanguage is tested. This experiment is aimed at examining the capacity of text document image. Different Font Size and Multilanguage (e.g.: English, Chinese and Thailand) and watermark capacity results are shown in Tables 4.1, 4.2 and 4.3. Notice that in all the experiments, for the same original text document image size 160x160, the watermark capacity produced by the cloud model find adjust embedding location point are approximately equal to cloud droplets.

Table 4.1 Compare of Hiding Capacity of English Image different Font Size

		English Image No:										Average
		01	02	03	04	05	06	07	08	09	10	Capacity(bits)
English Image Font Size:	12	122	50	63	44	65	47	109	109	89	39	74
	14	151	88	146	155	101	135	136	77	101	104	119
	16	223	255	204	220	85	190	168	253	213	246	187
	18	340	326	356	331	304	324	231	290	243	274	301
	20	300	326	309	442	365	327	379	142	402	273	327
	22	296	292	392	321	420	373	426	418	339	312	359
	24	356	401	473	487	397	483	392	450	454	437	433
	26	473	502	432	487	397	342	452	323	92	432	393
	28	525	493	545	438	436	195	187	471	267	505	357

Table 4.2 Compare of Hiding Capacity of Chinese Image Font Size

		Chinese Image No:										Average
		01	02	03	04	05	06	07	08	09	10	Capacity(bits)
Chinese Image Font Size:	12	159	147	77	201	282	173	77	162	235	123	164
	14	447	646	232	312	208	395	409	491	222	405	377
	16	471	214	111	403	463	333	349	419	216	364	334
	18	359	282	273	293	379	231	228	172	146	256	262
	20	287	283	323	207	355	235	290	263	278	253	277
	22	485	190	219	125	415	286	246	356	371	409	310
	24	473	401	525	391	474	458	570	434	550	418	469
	26	556	546	487	564	491	478	485	558	499	558	522
	28	629	539	474	484	517	490	511	620	541	546	535

Table 4.3 Compare of Hiding Capacity of Thailand Image Font Size

		Thailand Image No:										Average
		01	02	03	04	05	06	07	08	09	10	Capacity(bits)
Thailand Image Font Size:	12	214	292	263	323	281	383	507	414	332	345	335
	14	303	307	322	442	425	280	277	271	418	379	342
	16	426	269	233	207	370	295	393	233	209	415	266
	18	163	63	444	434	479	342	374	369	351	383	340
	20	313	311	299	134	477	358	318	292	270	267	304
	22	302	351	549	301	439	349	419	307	222	450	369
	24	354	488	475	562	493	420	517	157	515	415	440
	26	412	501	374	384	262	504	430	490	453	442	425
	28	568	470	547	594	483	554	464	125	617	564	499

Table 4.1 shows the experimental results for the watermark capacity of cloud watermarking in the original text documents image with different font size and size 160x160. As it was to be expected from the results presented in Section 3.4.3 for watermark capacity, the capacity of data embedding information in the original English image has middle. Similar results have been obtained watermarked capacity in Chinese image shown in Table 4.2, the watermark capacity for cloud watermarking of text documents image with size 160x160 and font size 16 and Thailand image shown in table 4.3.

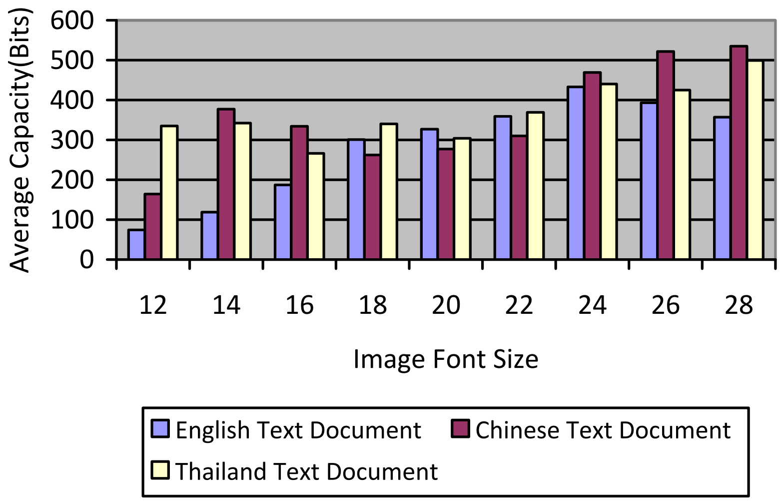
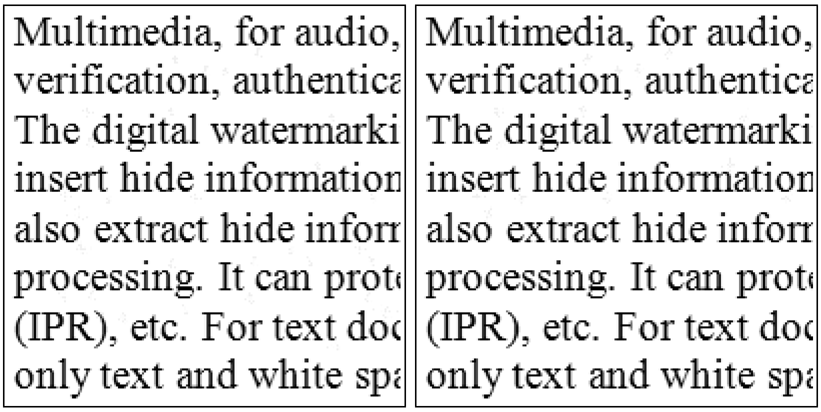


Figure 4.2: Watermark Capacity of English, Chinese and Thailand Text Document Image with Different Font Size

For the experiment, we find that the cloud watermarking of text documents image scheme is capacity improved by different Font size and Multilanguage, especially when the capacity of text document image with large Font size data is high. Due to the increase Font size, the capacity of cloud watermarking system is increased. Figure 4.2 show watermark capacity of English, Chinese, and Thailand Text Documents Image with different Font Size. However, the results of watermarked capacity of data hiding perform is meddle.

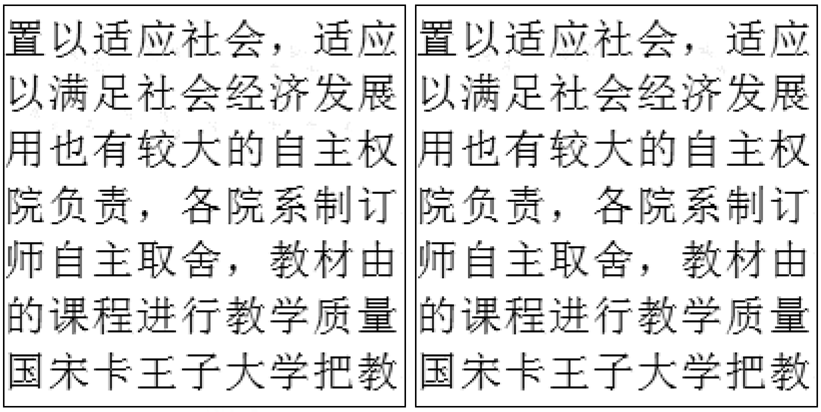
4.3 Comparison of Different Dimension Cloud Watermarking Capacity

For improving watermarked capacity, they have been digital watermarking of text documents image with 2-D cloud model techniques. A wide variety of images, including English, Chinese, and Thailand text images are used to test the capacities of using cloud model to partition the images. The results are shown in Figure 4.3. It can be seen from digital text watermarking by employing 1-D Cloud model scheme (e.g.: Figure 4.3(a) shows in watermarked English image, watermarked Chinese image in Figure 4.3(c) and Figure 4.3(e) shows watermarked Thailand image) and 2-D cloud model scheme (Figure 4.3(b) shows in watermarked English image, Chinese image Figure 4.3(d) and Thailand in Figure 4.3(f)). Experimentally, the use of cloud watermarking scheme with 1-D cloud model give the meddle capacity. Different dimension cloud model should be chosen for different application so that a good compromise capacity can be made.



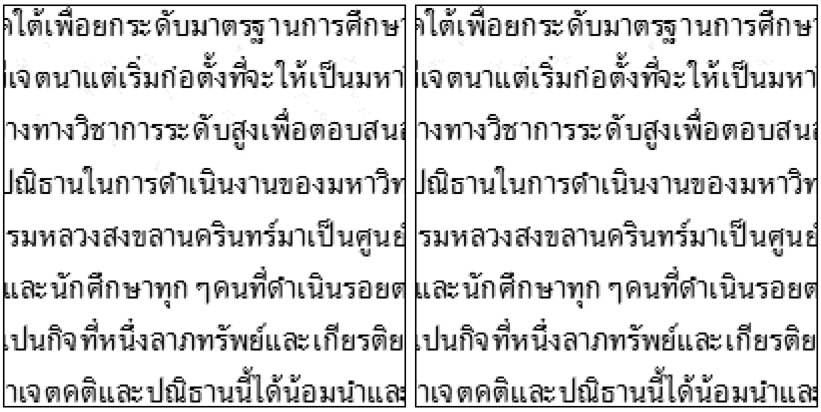
Multimedia, for audio,
verification, authentica
The digital watermarki
insert hide information
also extract hide inforr
processing. It can prot
(IPR), etc. For text doc
only text and white spa

(a) 1-D cloud Watermarked English Image (b) 2-D Cloud Watermarked English Image



置以适应社会，适应
以满足社会经济发展
用也有较大的自主权
院负责，各院系制订
师自主取舍，教材由
的课程进行教学质量
国宋卡王子大学把教

(c) 1-D Cloud Watermarked Chinese Image (d) 2-D Cloud Watermarked Chinese Image



ให้เพื่อยกระดับมาตรฐานการศึกษ
เจตนาแต่เริ่มก่อตั้งที่จะให้เป็นมหา
างทางวิชาการระดับสูงเพื่อตอบสน
ปณิธานในการดำเนินงานของมหาวิท
รมหลวงสงขลานครินทร์มาเป็นศูนย์
และนักศึกษาทุก ๆคนที่ดำเนินรอยต
ปนกิจที่หนึ่งลาภทรัพย์และเกียรติย
าเจตคติและปณิธานนี้ได้น้อมนำและ

(e) 1-D Cloud Watermarked Thailand Image (f) 2-D Cloud Watermarked Thailand Image

Figure 4.3: Cloud Watermarked Multilanguage Text Document Image

In our experiments, the performance of the proposed scheme is tested in ten text document images. Note that all of our test text images are of the font size 16 and different size (e.g.: 32x32, 64x64). Those data hiding capacity depend on cloud droplets each cloud model in the embed processing. Figure 4.4 compares the ratio of hiding capacity of 1-D and 2-D cloud methods for three language of text document image.

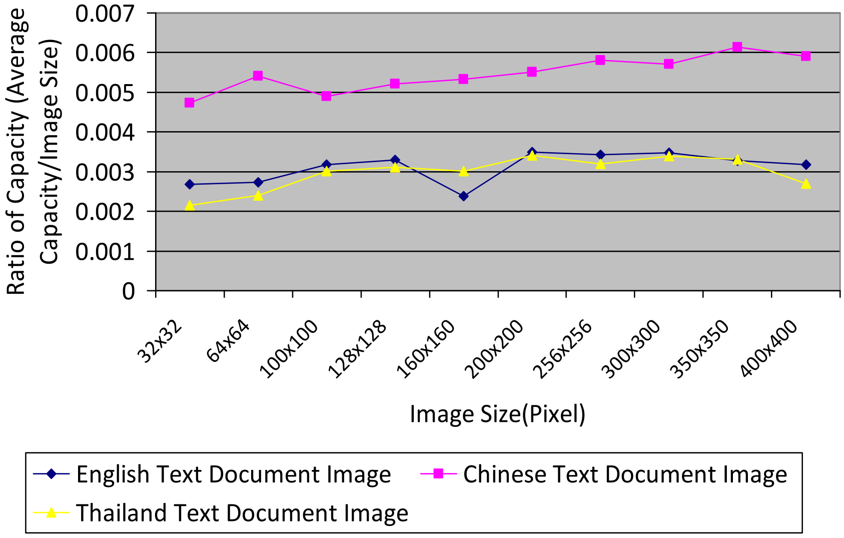


Figure 4.4: Compare of 1-D and 2-D cloud watermarking of Capacity

Since the content of cloud droplets is depend on triangle base on feature point in the. Thus the watermark capacity must be improved. And choose adjust to cloud model by inradius each triangle in the image.

CHAPTER 5

CONCLUSIONS

5.1 Conclusion

This thesis has provided a new method that cloud watermarking of text documents image with Cloud model, including 1-D Cloud Model method and 2-D Cloud model Method. And we have focused on the analysis of watermark capacity in the cloud watermarking system.

This thesis opened in Chapter 2 with an overview of digital watermarking techniques for text document, complaint of digital watermarking for text document (Table 2.1) that advantage and disadvantage, and cloud model technique, background and property in the Section 2.73 and 2.74.

The focus in Chapter 3, prepossessing data for obtain to parameter of E_n in the cloud model, including inradius (Section 3.13) and input information data strict in the every cloud droplets (Section 3.14).Thus, We proposed two methods of digital text watermarking system. The one method is 1-D cloud watermarking algorithm. This method has been hide information by cloud watermarking embed processing and obtain to information data during watermarking detection (Section 3.2.2 and 3.2.3 in the Chapter 3). At the same time, watermark capacity is meddle. For improvement watermark capacity, we have digital watermarking of text documents image by employing 2-D cloud model (Section 3.3 in the Chapter 3). And we can optimize the combination of the threshold of inradius for 2-D cloud watermarking of text document image.

In Chapter 4, the capacity of cloud watermarked of text document image schemes to test on capacity of text document image with different Font Size and Multilanguage, capacity of digital watermarking of text documents image with 1-D and 2-D cloud model and show its performance.

5.2 Future Work

Our proposed digital watermarking of text documents image with 1-D Cloud Model for copyright protection, but this method performs middle data hiding capacity. For improving capacity in the future, we will be digital watermarking of text documents image with 2-D cloud model and apply other technology such as compression, error control coding etc.

REFERENCES

[1] S.H. Low, N. Maxemchuk, J. Brassil and L. O'Gorman, "Document marking and identification using both line and word shifting", Proceedding of IEEE infocme'95,1995

[2] J. Zhao and E. Koch, "Embedding robust labels into images for copyright protection," in Proc. Int. Congr. Intellectual Property Rights for Specialized Information, Knowledge and New Technologies, Vienna, Austria, pp. 1-10, August 1995.

[3] J. K. Brassil, S. Low, N. F. Maxemchuk, and L. O'Gorman, "Electronic marking and identification techniques to discourage document copying," IEEE J. Select. Areas Common, vol. 13, no. 8, pp. 1495-1504, October 1995.

[4] T. Amamo and D. Misaki, "Feature calibration method for watermarking of document images," in Proc. Conf. Document Analysis and Recognition, Indian, pp. 91-94, 1999.

[5] Y.C. Tesng and H.K. Pan, "Data hiding in 2-color images," IEEE Trans. Computer., Vol. 51, No. 7, pp. 873-878, July 2002.

[6] Y.W. Kim and Il-Seok Oh, "Watermarking text document images using edge direction histograms", Pattern Recognition Letters, vol.25, no.11, pp.1243-1251, May 2004.

[7] M. Wu, E. Tang, and B. Liu, "Data hiding in digital binary image," in Proc. IEEE International Conference Multimedia and Expo., New York, pp. 393-396, 2002.

[8] Q. Mei, E. K. Wong, and N. Memon, "Data hiding in binary text documents", Proceeding of SPIE, vol. 4314, pp. 369-375, 2001.

[9] Min Wu, and Bede Liu, "Data Hiding in Binary Images for Authentication and Annotation", IEEE Trans. on Multimedia, vol. 6, no. 4, pp. 528-538, August 2004.

[10] H. Yang and Alex C. Kot, "Pattern-Based Date Hiding for Binary Images Authentication by Connectivity-Preserving", IEEE Trans. On Multimedia, vol. 9, no. 3, pp. 475-486, April 2007.

[11] Y. Liu, J. Mant, E. Wong and S. H. Low, "Marking and Detection of Text Documents Using Transform-domain Techniques", Proc. of SPIE, EI'99 Conf. on Security and Watermarking of Multimedia Contents, vol. 3657, pp. 317-328, San Jose, CA, 1999.

[12] H. Yang, A.C. Kot, S. Rahardja and X.D. Jiang, "High capacity data hiding for Binary Image in Morphological Wavelet Transform Domain", IEEE Trans. On Multimedia, pp.1239-1242, July 2007.

[13] H. Lu, X. Shi, Y.Q. Shi, A.C. Kot and L. Chen, "Watermark embedding DC components of DCT for binary images", in Proc. IEEE Workshop Multimedia signal Processing,pp.300-303,December 2002.

[14] H.L. Wang, M. Li, M.K. Cao and Y. Li, "Data Ming Application Based on Cloud Model in Spatial Decision Support System", CCCM 2008, pp.547-551, August 2008.

[15] Y.Q. Shi, and X.C. Yu, "Image Segmentation Algorithm Based on Cloud Model the Application of fMRI", ICICTA 2008, vol.2, pp.136-140, October 2008.

[16] R. Wang, W.G. Wan, X.L. Ma and Y.P. Li, "Cloud Model-based Control Strategy on Cluster Communication Coverage for Wireless Sensor Networks", the 2nd international Asia conference on CAR 2010,vol. 1, pp.307-310,April 2010.

[17] R.D. Wang and Y.Q. Xiong, "An Audio Aggregate watermark Based on Cloud Model", ICSP 2008, pp.2225-2228, December 2008.

[18] Y. Zhong, X.M. Niu and D.N. Zhao , "A Method of Protecting Relational Databases Copyright with Cloud Watermark", International Journal of Information Technology,vol.1,no.3,pp.112-116,2004.

[19] J. Fridrich, "Methods for Detecting changes in digital Image", In Processing of the 6th IEEE International workshop on intelligent signal processing and communication system, Melbourne, Australia.

[20] C. Rey and J-L. Dugelay , "Blind detection of malicious alterations on still images using robust watermarks", In Proc. Of IEEE, April, 2000.

[21] E.F. Hembrooke, "Identification of Sound and like Signals," 1961, United States Patent, 3004104.

[22] N. Komatsu and H. Tominaga, "Authentication system using concealed image intelematics," pp. 45-50, 1988.

[23] I.J. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker, *Digital Watermarking and Steganography, 2nd* Ed. (*The Morgan Kaufmann Series in Multimedia Information and Systems*), 2nd ed. Morgan Kaufmann, 11 2007.

[24] "ACM Multimedia and Security Workshop", http://wwwiti.cs.uni-magdeburg.de/iti_amsl/acm/acm_home.html.

[25] A.E. Bell, “The Dynamic Digital Disk”, Spectrum, IEEE, vol.36.no.10, pp.28-35, October 1999.

[26] F. Hartuang and M. Kutter, “Multimedia watermarking techniques”, proceeding of the IEEE, vol.87, no.7, pp.1079-1107, July 1999.

[27] “Secure Digital Music Initiative (SDMI)”, http://en.wikipedia.org/wiki/Secure_Digital_Music_Initiative.

[28] C. Herley, “Why watermarking is nonsense”, Signal Processing Magazine, IEEE, vol.19, no.5, pp10-11, September 2002.

[29] M. Barni, “What is the future for watermarking(part I)”,Signal Processing Magazine,IEEE,vol.20,no.5,pp.55-60,September 2003

[30] M. Barni, “What is the future for watermarking(part II)”,Signal Processing Magazine,IEEE,vol.20,no.6,pp.53-59,November 2003

[31] P.F. Luis, Pedro Comesana, R. Juna, T.P. Ramon and P.G. Fernado, “Watermarking security: a survey”, Transaction on DHMS I, LNCS 4300, pp.41-72, 2008.

[32] I.J. Cox, M.L. Miller and A.L. McKellips, “Watermarking as communication with side information”, Processing of the IEEE, vol.87, no.7, pp.1127-1141, July 1999.

[33] “ECRYPT WAVILA Watermarking Virtual Labe”, http://omen.cs.uni-magdeburg.de/ecrypt/.

[34] http://www.digimarc.com/.

[35] http://www.tredess.com/en/index.html.

[36] http://www.datamark.com.sg/.

[37] http://www.msicopycontrol.com/.

[38] http://www.aquamobile.es/.

[39] http://www.digitalwatermarkingalliance.org/.

[40] M. Kirstein, “Beyond traditional DRM: Moving to digital watermarking and fingerprinting in media monetization”, January 2006, available at http://www.multimediaintelligence.com.

[41] S.H. Low, A.M. Lapone, and N.F. Maxmchuk, “Document identification to discourage illicit copying”, IEEE Global Telecommunications Conference, vol.2, pp.1203-1208, 1995.

[42] 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,pp561-572, May 1998

[43] S.H. Low, N.F. Maxemchuk and A.M. Lapone, "Document identification for copyright protection using centroid detection" IEEE Trans. on Comm., vol. 46, no. 3, pp.372-383, March 1988.

[44] N.F. Maxemchuk and S.H. Low, "Marking text documents", Proceedings of IEEE Intl Conference on Image Processing, vol. 3, pp13, October 1997.

[45] J. Brassil and L. O'Gorman, "Watermarking document images with bounding box expansion", Procedding of 1st international workshop on information hiding, Newton Institue, Cambridge UK, pp.227-235, May 1996.

[46] N. Chotikakamthorn, "Document image data hiding technique using character spacing width sequence coding", Proc. IEEE International conference, Image Processing,vol.2 , pp.250-254, 1999.

[47] T. Amano and D. Misaki, "A feature calibration method for watermarking of document images", Proceedings of 5th International conference on Document Analysis and Recognition, pp.92-94, 1999.

[48] A.K. Bhattacharjya and H. Ancin, "Data embedding in text for a copier system", Proceedings of IEEE International Conference on Image Processing, vol.2, pp.245-249, 1999.

[49] K. Matsui and K. Tanaka, "Video-steganography: How to secretly embed a signature in a picture", Proceedings of IMA Intellectual Property Project, vol.1, pp.187-206, 1994.

[50] Z. Baharav and D. Shaked, "Watermarking of dither half-toned images", Proceeding of SPIE Security and Watermarking of Multimedia Contents, vol.1, pp.307-313, January 1999.

[51] H.-C.A. Wang, "Data hiding techniques for printed binary images", the international conference on Information Technology coding and computing, pp.55-59, April 2001.

[52] E. Koch and J. Zhao, "Embedding robust labels into images for copyright protection", proceeding international congress on intellectual property rights for specialized information, Knowledge and New Technologies, Vienna, August 1995.

[53] M.S. Fu and O.C. Au, "Data hiding for halftone images", proceeding of SPIE conference on Security and watermarking of Multimedia contents II, vol.3971, pp.228-236, January 2000.

[54] M.S. Fu and O.C. Au, “Data Hiding by smart pair toggling for halftone image”, proceeding of IEEE International conference Acoustics, speech and Signal Proceessing,vol.4, pp.2318-2321, June 2000.

[55] M.S. Fu and O.C. AU, “Improved halftone image data hiding with intensity selection”, Proceeding of IEEE International Symposium on Circuits and Systems, vol.5, pp.243-246, 2001.

[56] H.K. Pan, Y.Y. Chen and Y.C. Tseng, “A secure data hiding scheme for two-color images”, IEEE Symposium on Computer and Communications, pp.750-755, 2000.

[57] M.J. Atallah, C, McDonough, S. Nirenburg and V. Raskin, “Natural Language Processing for Information Assurance and Security: An Overview and Implementations”, Proceeding 9th ACM New Security Paradigms Workshop, pp. 51-65,September 2000,.

[58] H.M. Meral et al., “Natural Language Watermarking via Morph-syntactic Alterations”, Computer Speech and Language, pp.107-125, 1999.

[59] H.M. Meral et al., “Syntactic tools for text watermarking”, 19th SPIE Electronic image Conf. Security, Steganography and Watermarking of Multimedia Contents, Jan. 2007.

[60] M. Atallah and V. Raskin et al., “Natural Language Watermarking and Tamper proofing”, 5th information Hiding workshop LNCS 2578 October, 2002.

[61] Peng et al., “An Optimized natural language watermarking algorithm based on TMR”, on processing’s of 9th international Conference for Young Computer Scientists, 2009.

[62] K.S. Jonathan and B. Girod, “Digital Watermarking of Text, Image, and Video Documents”, Computer and Graphics, Elsevier Vol.22 no.6, pp.687-695,199.

[63] L. Robert and T. Shanmugapriya, “A Study on Digital Watermarking Techniques”, International Journal of Recent Trends in Engineering, vol.1, no.2, May, 1999.

[64] J. Zhang, Q. Li, C. Wang and J. Fang, “A novel application for text watermarking in digital reading”, in Proceedings of International Conference on AICI’09, Shanghai, China, pp.103-111,2009.

[65] Z. Jalil and A.M. Mirza, “An Ivisible Text watermarking Algorithm using Image Watermarking”, International conference on SCSS 2009, pp.147-152, 2009.

[66] Z. Jalil and A.M. Mirza et al., “A Novel Text Watermarking Algorithm Using Image Watermark”, International Journal of Computer Mathematics. vol. 7 no.3 pp.1255-1271, March 2011.

[67] D.Y. Li ,C.Y. Liu and Y. Du et al., "Artificial Intelligence with Uncertainty", Journal of Software,vol.15,no.11, pp.1-13,2004.

[68] D.Y. Li and Y. Du, "Artificial Intelligence with Uncertainty", National Defended Industry Press, Beijing 2005

[69] Y.M. Liu, "Fuzziness, other side of accuracy", Tsinghua University Press,Beijing 2001.

[70] D.Y. Li, H. J. Meng, X.M. Shi, "Membership Clouds and Membership Cloud Generators", Computer Research & Development, Vol. 32, pp.15—20, June 1995.

[71] X.H. Li and P.ZH. Wang , "Fuzzy Mathematics", National Defended Industry Press, Beijing 1994.

[72] C.E. Shannon, "A Mathemactical Theory of Communicaiont",ACM Sigmoble Mobile Computing and Communication Review,vol.2,no.1,pp.3-55,2001.

[73] D.Y. Li, D. Cheung and X. Shi, "Uncertainty Reasoning Based on Cloud Models in Controllers", Computer Math. Application, vol.3 pp.99-124, 1998.

[74] D.Y. Li and C.Y. Liu, "Study on the University of the Normal Cloud Model", Engineering Science, vol.6, pp.28-34, August 2004.

[75] Y. Zhang, D.N. Zhao and D.Y. Li, "The Similar Cloud and Measurement Method", Information and control, vol.33, pp.129-132, April 2004.

[76] C.H. Dai, Y.F. Zhu, W.R. Chen and J.H. Lin, "Cloud Model Based Genetic Algorithm and Its Application", ACTA electronic SINA. Vol.35, pp.141-1424, July 2007.

[77] Y. Deng and H.J. Wu, "Spatial Clustering Method Based on Cloud Model," The fourth International Conference on Fuzzy Systems and Knowledge Discovery, pp.272-276, December 2007.

[78] R. Xun, P. Nie, P. Lin and D. Chu, "Cloud Model based data attributes reduction for clustering", ICST, Australia, pp.33-36, January 2008.

[79] J.H. Fan and D.Y. Li "Mining classification Knowledge based on Cloud Models", Springerlink Verlag berlin Heidelberg, pp.317-326, 1999.

[80] D.Y. Li, D.D. Li and X. Shi, "Ming Association Rules with Linguistic cloud models," the National Nature Science Foundation of China, pp143-158, August 1998.

[81] H.J. Wang and Y. Deng, "Spatial clustering Method Based on cloud model and Data Field", ISICA 2007, Springer Verlag Beidelberg, pp420-427, 2007.

[82] D.Y. Li and W.W. Zhao, "Intrusion detection using cloud model", computer engineering and application, vol.39, no.26, pp.158-160. September 2003(Chinese).

[83] J.P. Wen and B.G. Cao, "A Modified Particle Swarm Optimize based on cloud model", Proceeding of the IEEE, International conference, pp.1238-1241, 2008.

[84] H.Y. Kang, Y.F. Li and S.P. Tang, "Application of cloud models in digital libraries", Proceeding of the 6th international conference on Machine Learning and Cybernetics Hong Kong, pp19-22, August 2007.

[85] K. Zhang, W.H. Zhang and J.S. Yuan, "Edge Detection of Images Based on Cloud Model Cellular Automata", Proceedings of the 27th Chinese Control Conference, pp.249-253, August 2008.

[86] C. Harris and M. Stephens, "A Combined corner and Edge Detector", Process Alvey Vision Conference, University Manchester, pp147-151, 1988.

[87] Delone B. N. Sur la sph´ere vide. Bul. Acad. Sci. URSS, and Class. Sci. Nat., pp.793-800, 1934.

[88] Steven Fortune, "Vornoi Diagrams and Diagrams and Delaunay Triangulations", Lecture Notes Series on Computing, World Scientific, Singapore, vol.1, pp.193-233, 1992.

[89] Peter Su and Robert L. Scot Drysdale, "A Comparison of Sequential Delaunay Triangulation Algorithms", Proceedings of the Eleventh Annual Symposium on Computational Geometry, pp.61-70, June 1995.

[90] C.L. Lawson. "Software for C Surface Interpolation. Mathematical Software III (John R. Rice, editor)", Academic Press, New York, pp.161-194, 1977.

[91] D.T. Lee and B.J. Schachter, "Two Algorithms for Constructing a Delaunay Triangulation", International Journal of Computer and Information Sciences, vol.9, no.3, pp.219-242, 1980.

[92] Steven Fortune, "A Sweepline Algorithm for Voron Diagrams", Algorithmica vol.2, no.2, pp.153-174, 1987.

[93] D.M. Burton, "Elementary Theory of Numbers", McGraw-Hill, New York, NY, pp.227-234, 1997.

[94] M. Tsai, K. Yuang and Y. Chen, "Join wavelet and spatial transformation for digital watermarking", IEEE Transaction on Consumer Electronics, vol.46, no.1, pp.241-245, 2000.

[95] T.M. Cover and J.A. Thomas, “Elements of Information Theory (Second Edition)”, John Wiley and Sones, Inc., Hoboken, New Jersey, pp.280-289, 1991.

[96] S.G. Hoggar , “Mathematics of Digital Images : Creation, Compression, Restoration, Recognition”, Cambridge University Press, pp.444-518, 2006.

APPENDIX A

Cloud Watermarking of Text Document Image Code

Appendix A1.1 1-D Cloud Model Code

```
/* 1-D Cloud Model Code
 * By Mr. Liu Yi
 * E-mail:gavinliuyi@hotmail.com/gavinliuyi@yahoo.com
 * Preprose original text document image ,then load in-radius data
 * and referents point(center of cloud model point).
 * See section 2.7.3 for details cloud model algrithm.
 */


//The 1-DIMNESINAL CLOUD DROPS Function
private: System::Void CloudModel1DCloudDrops(CloudTriangle^Tri,List<Point>^CloudDrops,
                         double Ex,double En,double He,int CloudNumber){
        double Enx;
        double TDX;
        double ESA;

        Point CPoint;
        Point EmCloudPoint;
        srand(CloudNumber);

        for(int i=0;i<=CloudNumber;i++){
             Enx = sqrt(-2*log(rand()*1./RAND_MAX))*cos(2*M_PI*rand()*1./RAND_MAX)*He +En;
             TDX = sqrt(-2*log(rand()*1./RAND_MAX))*cos(2*M_PI*rand()*1./RAND_MAX)*Enx +Ex;
             ESA = exp(-0.5*((TDX-Ex)*(TDX-Ex)/(Enx*Enx)));

             //Correspand a Point (x,y)1-D Cloud Model
             CPoint.X = int(TDX*cos(2*M_PI*ESA));
             CPoint.Y = int(TDX*sin(2*M_PI*ESA));

             EmCloudPoint.X = Tri->center.X +CPoint.X;
             EmCloudPoint.Y = Tri->center.Y +CPoint.Y;
             CloudDrops->Add(Point(EmCloudPoint.X,EmCloudPoint.Y));
        }
        RemoveSameCloudDrops(CloudDrops);
 }
```

Appendix A1.2 2-D Cloud Model Code

```
/* 2-D Cloud Model Code
 * By Mr. Liu Yi
 * E-mail:gavinliuyi@hotmail.com/gavinliuyi@yahoo.com
 * Preprose original text document image ,then load in-radius data
 * and referents point(center of cloud model point).
 * See section 2.7.3 for details cloud model algrithm.
 */

//The 2-DIMNESINAL CLOUD DROPS Function
private: System::Void CloudModel2DCloudDrops(CloudTriangle^Tri,List<Point>^CloudDrops,double
Ex,double En,double He,int CloudNumber){
        double Enx,Eny;
        double TDX;
        double TDY;
        double MESA;

        Point CPoint;
        Point EmCloudPoint;
        srand((unsigned)time(NULL));

        for(int i=0;i<=CloudNumber;i++){
         // Gencation Forword Cloud Model,Two-Dimnesinal cloud drops(TDX,TDY,ESA)
            Enx =sqrt(-2*log(rand()*1./RAND_MAX))*cos(2*M_PI*rand()*1./RAND_MAX)*He +En;
            Eny =sqrt(-2*log(rand()*1./RAND_MAX))*cos(2*M_PI*rand()*1./RAND_MAX)*He +En;
            TDX = sqrt(-2*log(rand()*1./RAND_MAX))*cos(2*M_PI*rand()*1./RAND_MAX)*Enx +Ex
            TDY = sqrt(-2*log(rand()*1./RAND_MAX))*cos(2*M_PI*rand()*1./RAND_MAX)*Eny +Ex;
            MESA = exp(-0.5*((TDX-Ex)*(TDX-Ex)/(Enx*Enx)+(TDY-Ex)*(TDY-Ex)/(Eny*Eny)));

            // 2-D Cloud Model
            CPoint.X =int(TDX);//cos0=1,Ex=0
            CPoint.Y =int(TDY);

            EmCloudPoint.X = Tri->center.X +CPoint.X;
            EmCloudPoint.Y = Tri->center.Y +CPoint.Y;
            CloudDrops->Add(Point(EmCloudPoint.X,EmCloudPoint.Y));
        }
        RemoveSameCloudDrops(CloudDrops);
        }
```

Appendix A2 Feature Detection by Harris Corner Code

```
/* Feacture Detection by Harris Corner Code
 * By Mr. Liu Yi
 * E-mail:gavinliuyi@hotmail.com/gavinliuyi@yahoo.com
 */

# include"cxcore.h"
# include"cv.h"

private: System::Void OpenImageButton_Click(System::Object^ sender, System::EventArgs^  e) {
        if(OpenFileDialog->ShowDialog() ==System::Windows::Forms::DialogResult::OK){
           MessageBox::Show(OpenFileDialog->FileName,"Original Image");
           Bitmap^bmp = gcnew Bitmap(OpenFileDialog->FileName);
           Bitmap^cbmp = gcnew Bitmap(bmp->Width,bmp->Height,
                                System::Drawing::Imaging::PixelFormat::Format24bppRgb);
           Graphics::FromImage(cbmp)->DrawImage(bmp,0,0,bmp->Width,bmp->Height);
           PictureBox->Image =cbmp;
        }
 }

private: System::Void HarrisCornerButton_Click(System::Object^sender, System::EventArgs^e) {
        if(PictureBox->Image == nullptr){
           System::Windows::Forms::DialogResult result = MessageBox::Show(this,
                                 L"No Input Image",L"Input Image ERROR",
                                 MessageBoxButtons::RetryCancel,MessageBoxIcon::Error);
           return;
        }
        Bitmap^ bitmap =(Bitmap^)PictureBox->Image;
        BitmapData^bitmapdata = bitmap->LockBits(System::Drawing::Rectangle(0,0,
                                       bitmap->Width,bitmap->Height),
                                       ImageLockMode::ReadWrite,bitmap->PixelFormat);

        IplImage *OriImage = cvCreateImageHeader(cvSize(bitmap->Width,
                                           bitmap->Height),IPL_DEPTH_8U,3);
        OriImage->imageData =(char*)(void*)bitmapdata->Scan0;
        OriImage->widthStep = bitmapdata->Stride;

        IplImage *GrayImage =cvCreateImage(cvSize(OriImage->width,
                                           OriImage->height),OriImage->depth,1);
        cvCvtColor(OriImage,GrayImage,CV_BGR2GRAY);
        IplImage *HarImage = cvCreateImage(cvSize(OriImage->width,
                                           OriImage->height),IPL_DEPTH_32F,1);
        IplImage *ProImage = cvCreateImage(cvSize(OriImage->width,
                                           OriImage->height),IPL_DEPTH_8U,1);

        cvCornerHarris(GrayImage,HarImage,3);

        double minVal =0.0;
        double maxVal =0.0;
        double scale,shift;
        double min =0.0;
        double max =255;
```

```
        cvMinMaxLoc(HarImage,&minVal,&maxVal,NULL,NULL,0);
        scale =(max-min)/(maxVal-minVal);
        shift =minVal*scale +min;

        double threshold =(maxVal-minVal)/3;
        cvThreshold(HarImage,HarImage,thrshold,255,CV_THRESH_BINARY);
        cvConvertScale(HarImage,ProImage,scale,shift);
        cvNot(ProImage,ProImage);
        bitmap->UnlockBits(bitmapdata);
        cvCvtColor(ProImage,OriImage,CV_GRAY2BGR);
        PictureBox->Invalidate();
}

private: System::Void SaveImageButton_Click(System::Object^ sender, System::EventArgs^ e) {
        if(PictureBox->Image == nullptr){
           MessageBox::Show(L"No Harris Corner Image");
           return;
        }
        else if(SaveFileDialog->ShowDialog() ==System::Windows::Forms::DialogResult::OK){
           PictureBox->Image ->Save(SaveFileDialog->FileName);
        }
}
```

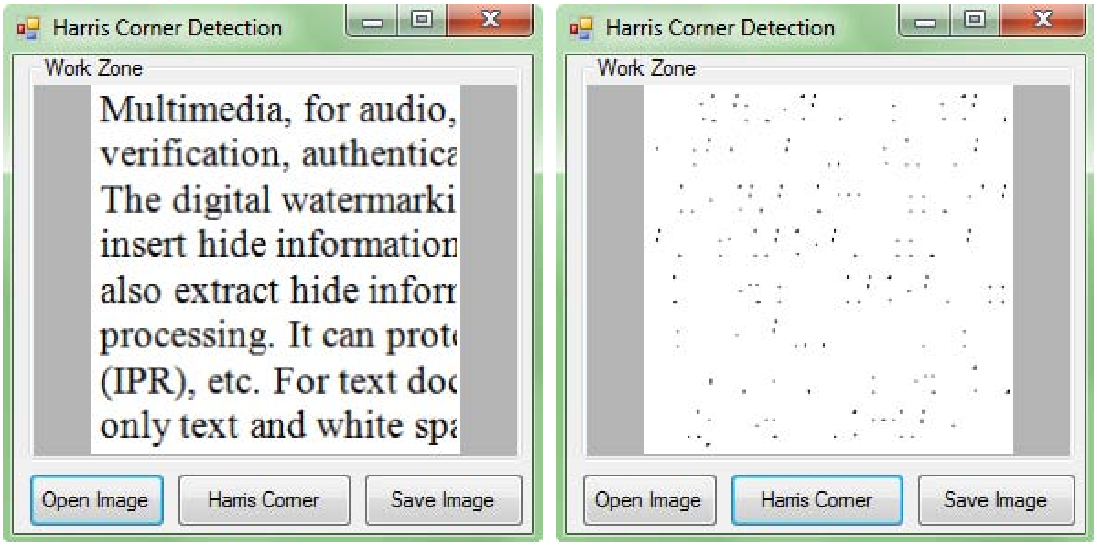


Figure A2.1: Input of Text document Image and output of Harris Corner Text document Image (Lift: Original Text Document Image, Right: Harris Corner Image with Threshold)

Appendix A3 Cloud Triangle Code

```
/* The Drawing Cloud Triangle Codes
 * By Mr. Liu Yi
 * E-mail:gavinliuyi@hotmail.com/gavinliuyi@yahoo.com
 */

private:System::Void DelaunayTriangleMenuItem_Click(System::Object^ sender,
                                                                System::EventArgs^e) {
        if(EmbedTab->Visible){
          if(EmbedPictureBox->Image == nullptr){
             System::Windows::Forms::DialogResult result = MessageBox::Show( this,
                                          L"No Input Image- Load Input Original Image",
                                          L"Input Original Image Error",
                                          MessageBoxButtons::RetryCancel,
                                          MessageBoxIcon::Error);
             return;
          }
        Bitmap^bitmap = (Bitmap^)EmbedPictureBox->Image;
        BitmapData^bmpdata = bitmap->LockBits(System::Drawing::Rectangle(0, 0,
                                          bitmap->Width,bitmap->Height),
                                          ImageLockMode::ReadWrite,bitmap->PixelFormat);
        IplImage *image =cvCreateImageHeader(cvSize(bitmap->Width,bitmap->Height),
                                          IPL_DEPTH_8U,3);
        image ->imageData = (char *)(void *)bmpdata ->Scan0;
        image ->widthStep = bmpdata ->Stride;
        CvSize size = cvGetSize(image);

        IplImage *grayimage = cvCreateImage (size,image->depth,1);
        cvCvtColor (image,grayimage,CV_RGB2GRAY);
        IplImage *curimage = cvCreateImage(size,IPL_DEPTH_32F,1);
        IplImage *harimage = cvCreateImage(size,IPL_DEPTH_8U,1);

        cvCornerHarris(grayimage,curimage,3);

        double minVal =0.0, maxVal = 0.0;
        double scale, shift;
        double min = 0,max = 255;

        cvMinMaxLoc(curimage,&minVal,&maxVal,NULL,NULL,0);
        scale = (max-min)/(maxVal-minVal);
        shift = minVal*scale + min;

        double threshold = (maxVal-minVal)/5;   //Set Threshold value
        cvThreshold(curimage,curimage,threshold,255,CV_THRESH_BINARY);

        List<Point>^AList = gcnew List<Point>( ); //Create and a new ArraryList

        for(int height =0; height <curimage->height;height++){
           for(int widtht = 0; widtht<curimage->width;widtht ++){
               CvScalar scalar = cvGet2D(curimage,height,widtht);
                 if(scalar.val[0] >threshold)
                    AList->Add(Point(widtht,height));
           }
```

```
}

CvMemStorage * storage = cvCreateMemStorage();
CvSubdiv2D* subdiv;     // the subdivision itself
subdiv = cvCreateSubdiv2D(CV_SEQ_KIND_SUBDIV2D,sizeof(*subdiv),
                          sizeof(CvSubdiv2DPoint),sizeof(CvQuadEdge2D),storage);
cvInitSubdivDelaunay2D( subdiv, cvRect(0,0,curimage->width,curimage->height));

for(int i=0; i< AList->Count; i++){
   CvPoint2D32f fp;
   fp = cvPointTo32f(cvPoint(AList[i].X, AList[i].Y));
   cvSubdivDelaunay2DInsert(subdiv,fp);
}

cvCalcSubdivVoronoi2D(subdiv);
CvSeqReader  reader;

int total = subdiv->edges->total;
int elem_size = subdiv->edges->elem_size;
int count=0;

cvStartReadSeq( (CvSeq*)(subdiv->edges), &reader, 0);
for(int i =0; i<total; i++){
    CvQuadEdge2D *edgeQ = (CvQuadEdge2D*)(reader.ptr);

    if(CV_IS_SET_ELEM(edgeQ)){
       CvSubdiv2DEdge edge =(CvSubdiv2DEdge)(edgeQ);
       CvSubdiv2DEdge t = edge ;
       int count =0;

       do{
         count++;
         t=cvSubdiv2DGetEdge(t,CV_NEXT_AROUND_LEFT);
       }while( t!=edge );

       CvPoint* buf =(CvPoint*)malloc(count*sizeof(buf[0]));
       t =edge;

       int j;
       CvSubdiv2DPoint * outer_vtx[3];

       for(j=0;j<count; j++){
         CvSubdiv2DPoint* pt = cvSubdiv2DEdgeOrg(t);
         outer_vtx[j] = pt;
         if(!pt||(pt->pt.x<0)||(pt->pt.y<0)||(pt->pt.x>size.width)
                   ||(pt->pt.y>size.height))
            break;
         buf[j] = cvPoint(cvRound(pt->pt.x),cvRound(pt->pt.y));
         t = cvSubdiv2DGetEdge(t,CV_NEXT_AROUND_LEFT);
        }

       if(j == count)
         cvPolyLine(curimage,&buf,&count,1,1,CV_RGB(0,0,255),1,CV_AA,0);
       free(buf);
     }
     CV_NEXT_SEQ_ELEM(elem_size,reader);
```

```
}
cvClearSubdivVoronoi2D(subdiv);
AList->Clear( );
cvConvertScale(curimage,harimage,scale,shift);

cvNot(harimage,harimage);
bitmap->UnlockBits(bmpdata);
cvCvtColor(harimage,image,CV_GRAY2RGB);

cvReleaseImageHeader(&image);
cvReleaseImage(&curimage);
cvReleaseImage(&harimage);

cvReleaseImage(&grayimage);
EmbedPictureBox->Invalidate();
}
```

Appendix A4 Digital Cloud Watermarking Embedding Process Code

```
/* Digital Cloud Watermarking embedded processs Codes
 * By Mr. Liu Yi
 * E-mail:gavinliuyi@hotmail.com/gavinliuyi@yahoo.com
 *
 */

private: System::Void OpenMenuItem_Click(System::Object^  sender, System::EventArgs^  e) {
           if(OpenOriFileDialog->ShowDialog() == System::Windows::Forms::DialogResult::OK){
              MessageBox::Show(OpenOriFileDialog->FileName,"Original Image");
              Bitmap^cbmp = gcnew Bitmap(OpenOriFileDialog->FileName);
              Bitmap^bmp = gcnew Bitmap(cbmp->Width,cbmp->Height,System::Drawing::Imaging
                                 ::PixelFormat::Format24bppRgb);
              Graphics::FromImage(bmp)->DrawImage(cbmp, 0, 0,cbmp->Width,cbmp->Height);
              EmbedPictureBox->Image = bmp;
           }
  }

private:System::Void SaveMenuItem_Click(System::Object^  sender, System::EventArgs^  e) {
        if(EmbedTab->Visible){
         //Save  the Watermarked Text Document Image
           if(EmbedPictureBox->Image == nullptr){
              MessageBox::Show(L"Not Watermarked Image");
              return;
            }
            else if (SaveWateFileDialog->ShowDialog ( ) == System::Windows::Forms::
                       DialogResult::OK){
                EmbedPictureBox->Image->Save(SaveWateFileDialog->FileName);
            }
        }
  }

private: System::Void ExitMenuItem_Click(System::Object^  sender, System::EventArgs^  e) {
        while(MessageBox::Show("Are you sure exit in CMDWatermarking?", "",
                 MessageBoxButtons::YesNo)==System::Windows::Forms::DialogResult::Yes){
              exit(0);
         }
 }

private: System::Void ClearInfobutton_Click(System::Object^  sender, System::EventArgs^ e) {
                         InfoTextBox->Enabled = true;
                         InfoTextBox->Text ="";
                         InfoTextBox->MaxLength = 200;
                 }

private: System::Void MsgHidebutton_Click(System::Object^  sender, System::EventArgs^  e) {
        if(String::IsNullOrEmpty(InfoTextBox->Text)){ //Check for null or empty string
             System::Windows::Forms::DialogResult result = MessageBox::Show( this,
                                   L"No Input - enter Hide Information data",L"Input Error",
                                   MessageBoxButtons::RetryCancel,MessageBoxIcon::Error);
             return;
        }

        String^FileDataName = InfoTextBox->Text;
```

```
        array<unsigned char>^ArrayFileDataName = System::Text::Encoding::Unicode
                                                ->GetBytes(FileDataName);
        String^Hexstring = "";

        for(int i = 0; i<ArrayFileDataName->Length; i++){
            for(int j =0; j<8;j++){
               Hexstring +=(ArrayFileDataName[i]&0x80)>0?"1":"0";
               ArrayFileDataName[i] <<=1;
            }
        }
        MessageBox::Show(String::Format("Datainfo:{0},{1}",Hexstring,Hexstring->Length));
 }

private: System::Void EmbedMenuItem_Click(System::Object^  sender, System::EventArgs^  e) {
           if(EmbedPictureBox->Image == nullptr){
              System::Windows::Forms::DialogResult result = MessageBox::Show( this,
                                 L"No Input Image- Load Input Original Image",
                                 L"Input Original Image Error",
                                 MessageBoxButtons::RetryCancel,MessageBoxIcon::Error);
                                  return;
                                  }

              Bitmap^bitmap = (Bitmap^)EmbedPictureBox->Image;
              BitmapData^bmpdata = bitmap->LockBits(System::Drawing::Rectangle(0, 0,
                 bitmap->Width,bitmap->Height),ImageLockMode::ReadWrite,bitmap->PixelFormat);

              IplImage *image =cvCreateImageHeader(cvSize(bitmap->Width,bitmap->Height),
                                 IPL_DEPTH_8U,3);
              image ->imageData = (char *)(void *)bmpdata ->Scan0;
              image ->widthStep = bmpdata ->Stride;

              CvSize size = cvGetSize(image);
              IplImage *grayimage = cvCreateImage (size,image->depth,1);
              cvCvtColor (image,grayimage,CV_BGR2GRAY);

              IplImage *curimage = cvCreateImage(size,IPL_DEPTH_32F,1);
              IplImage *harimage = cvCreateImage(size,IPL_DEPTH_8U,1);
              cvCornerHarris(grayimage,curimage,3);

              double minVal =0.0, maxVal = 0.0;
              double scale, shift;
              double min = 0,max = 255;

              cvMinMaxLoc(curimage,&minVal,&maxVal,NULL,NULL,0);
              scale = (max-min)/(maxVal-minVal);
              shift = minVal*scale + min;

              double threshold = (maxVal-minVal)/5;     //Set Threshold value
              cvThreshold(curimage,curimage,threshold,255,CV_THRESH_BINARY);

              List<Point>^AList = gcnew List<Point>( ); //Create and a new ArraryList
              for(int height =0; height <curimage->height;height++){
                 for(int widtht = 0; widtht<curimage->width;widtht ++){
                    CvScalar scalar;
                    scalar = cvGet2D(curimage,height,widtht);
                    if(scalar.val[0] >threshold){
```

```
                AList->Add(Point(widtht,height));
                }
            }
    }

List<CloudTriangle^>^cloudtriangle = gcnew List<CloudTriangle^>( );
CloudTriangleDrawingDelaunayTriangulation(AList,curimage,cloudtriangle);
cvConvertScale(curimage,harimage,scale,shift);
cvNot(harimage,harimage);
bitmap->UnlockBits(bmpdata);

RemoveSameCloudTriangle(cloudtriangle);//Remove the Same Cloud Triangle Function
int Thresholdmin =0;
int Thresholdmax =0;
Thresholdmin =System::Convert::ToInt16(TminTextBox->Text);
Thresholdmax =System::Convert::ToInt16(TmaxTextBox->Text);

for (int triangle =cloudtriangle->Count-1;triangle>= 0;triangle--){
        CloudTriangle^ c = cloudtriangle[triangle];
        double inradius =0.0;
        inradius =      CloudTriangleInradius(c);
        if((inradius<Thresholdmin)||(inradius>Thresholdmax))
                cloudtriangle->RemoveAt(triangle);
}

List <int>^BiStegoData = gcnew List<int>();
BiStegoDataCovertBinaryStegoData(BiStegoData);
int CheckBiCount = CheckBiStegoDataCount(cloudtriangle,BiStegoData->Count);

if(CheckBiCount ==0){
        List<Point>^CloudDrops = gcnew List<Point>( );
        int TempTriangle = cloudtriangle->Count;
        int IndexCloudNumber =(TempTriangle ==1)?1:int(ceil(log10(double(
                                TempTriangle))/log10(double(2))));
        int StartIndex = 0;
        int MaxDataLength = 0;
        int PreCDropsCount =0;
        int CoutStegoData =0; // Earch Next Blocks Start Position

        int IndexCout =0;
        int MarkTriangleCount =0;//Mark Using Hide data in the Triangle
        int MarkBloksCount=0;
        int MarkRealDropsCount =0;

        dataGridView->ColumnCount = 10;
        dataGridView->RowCount  = TempTriangle+3;

        for(int k =0; k<TempTriangle;k++){
                double Inradius = 0.0;
                CloudTriangle^ Tri =cloudtriangle[k];
                Inradius = CloudTriangleInradius(Tri);

                double Ex = 0.0;
                double En = Inradius/3;
                double He = Inradius/30;
                int CloudNumber = int(0.3*M_PI*Inradius*Inradius);
```

```
        DataGridViewTriangleCloudModelInformation(k,Tri,Inradius);
        CloudModel1DCloudDrops(Tri,CloudDrops,Ex,En,He,CloudNumber);
        #ifdef _2DCLOUDMODEL
        CloudModel2DCloudDrops(Tri,CloudDrops,Ex,En,He,CloudNumber);
        #endif

        for(int j =IndexCout;j<CloudDrops->Count-1;j++){
                CloudDropsBubbleSorting(CloudDrops,j,IndexCout);
                }
        IndexCout=CloudDrops->Count;
        //Preveiw Information Data
        int HidingDataLength=0;
        int PreHidingDataLength =0;
        int RealHidingDataLength =0;
        int TempCloudDrops = CloudDrops->Count-PreCDropsCount;

        int SizeDataLength = TempCloudDrops - IndexCloudNumber;
        dataGridView->Rows[k+1]->Cells[9]->Value =TempCloudDrops;
        PreHidingDataLength = CheckHidingDataLength(SizeDataLength
        int SizeData = SizeDataLength-PreHidingDataLength

        RealHidingDataLength =BiStegoData->Count-CoutStegoData;
        HidingDataLength =(RealHidingDataLength<PreHidingDataLength) ?
RealHidingDataLength : PreHidingDataLength;
        List<int>^DataBlocks = gcnew List<int>( );

        for(int IndexCNo=IndexCloudNumber-1;IndexCNo>=0;IndexCNo--){
                int Indexdata = k;
                DataBlocks->Add(Indexdata>>IndexCNo&1);
        }
        for(int SizeDataIndex =SizeData-1;SizeDataIndex>=0;
              SizeDataIndex--){
                int HidingDataLengthL = HidingDataLength;
                DataBlocks->Add(HidingDataLengthL>>SizeDataIndex&1);
        }

        MaxDataLength +=HidingDataLength;
        int ValidLength =0;

        if(MaxDataLength<=(BiStegoData->Count))
                ValidLength = MaxDataLength;
        else
                ValidLength =(BiStegoData->Count);

        for(int HidingData_Index=StartIndex;HidingData_Index<ValidLength;
               HidingData_Index++){
                DataBlocks->Add((BiStegoData[HidingData_Index]==1)?1:0);
                CoutStegoData++;
        }

        int HideTempCloudDrops=0;

        if((ValidLength-StartIndex)<=TempCloudDrops){
                HideTempCloudDrops = TempCloudDrops;
        }
```

```
                    else{
                            HideTempCloudDrops = TempCloudDrops-ValidLength+StartIndex;
                    }

                    if(StartIndex !=BiStegoData->Count){
                            int DataBlocksCount = 0;
                            int TotalTempTriangle = 0;
                            DataBlocksCount =DataBlocks->Count;
                            MarkRealDropsCount +=DataBlocksCount;

                            dataGridView->RowCount +=DataBlocksCount+3;
                            TotalTempTriangle =MarkBloksCount+TempTriangle;
                            DataGridViewCModelShownEIndex(TotalTempTriangle,k,DataBlocks
                    Count);//Create New Index In the DataGridView data
                            HideDataEmbededProcessing(IndexCloudNumber,SizeData,DataBloc
                    ks,grayimage,CloudDrops,TotalTempTriangle,PreCDropsCount);

                            MarkTriangleCount++;
                            MarkBloksCount +=DataBlocksCount+3;
                            }
                    StartIndex =CoutStegoData;//Mark Information data Start Index
                    Nunmber.
                    PreCDropsCount +=HideTempCloudDrops;
            }

            cvCvtColor(grayimage,image,CV_GRAY2BGR);
            ShowImageFucation(image->width,image-
        >height,TempTriangle,MarkTriangleCount,MarkRealDropsCount,CloudDrops);
            CloudDrops->Clear();
            MessageBox::Show(L"Embedding Processing is finished,please Think you!");
            }

            else if(CheckBiCount ==1){
            MessageBox::Show(this,L"Input StegoData Length Long",L"WARNING",

        MessageBoxButtons::RetryCancel,MessageBoxIcon::Error);
        }

        cvReleaseImageHeader(&image);
        cvReleaseImage(&curimage);
        cvReleaseImage(&harimage);
        cvReleaseImage(&grayimage);

        EmbedPictureBox->Invalidate();
        }
}

//The Information Convert to Binary funcation
private:System::Void BiStegoDataCovertBinaryStegoData(List<int>^BiStegoData){
        String^StegoData = InfoTextBox->Text;
        array<unsigned char>^ArrayStegoData = System::Text::Encoding::Unicode
                                        ->GetBytes(StegoData);
        for(int i = 0; i<ArrayStegoData->Length; i++){
            for(int j =0; j<8;j++){
                BiStegoData->Add((ArrayStegoData[i]&0x80)>0? 1 : 0);
                ArrayStegoData[i] <<=1;
```

```
            }
         }
}

//The Remove Same Cloud Triangle function
private: System::Void RemoveSameCloudTriangle(List<CloudTriangle^>^cloudtriangle){
        int CloudTCount =cloudtriangle->Count;
        if(CloudTCount>1){
           for(int Index1= cloudtriangle->Count-1; Index1>=0; Index1--){
                Point Centre1=cloudtriangle[Index1]->center;
                   for(int Index2=cloudtriangle->Count-2; Index2>=0;Index2--){
                        Point Centre2 = cloudtriangle[Index2]->center;
                        if((Index1 !=Index2)&&(Centre1==Centre2)){
                            cloudtriangle->RemoveAt(Index2);
                            Index1--;
                        }
                   }
           }
        }
   }

//The Remove Same Cloud Drops function
private: System::Void RemoveSameCloudDrops(List<Point>^CloudDrops){
        for(int i=CloudDrops->Count-1;i>=0;i--){
             Point Drops1=CloudDrops[i];
             for(int k=CloudDrops->Count-2;k>=0;k--){
                  Point Drops2 =CloudDrops[k];
                  if((i!=k)&&(Drops1==Drops2)){
                        CloudDrops->RemoveAt(k);
                        i--;
                  }
             }
        }
    }


//The Cloud Drops Bubble Sorting Funcation
private: System::Void CloudDropsBubbleSorting(List<Point>^CloudDrops,int j,int IndexCout){
        int i=j;
        if((CloudDrops[j].X>CloudDrops[j+1].X)||((CloudDrops[j].X==CloudDrops[j+1].X)
               &&(CloudDrops[j].Y>CloudDrops[j+1].Y))){
            Point temp;
            temp =CloudDrops[j];
            CloudDrops[j] = CloudDrops[j+1];
            CloudDrops[j+1] =temp;
        }

        if(i!=IndexCout){
           for(i =j;i>IndexCout;i--){
              if((CloudDrops[i].X<CloudDrops[i-1].X)||((CloudDrops[i].X==
                     CloudDrops[i-1].X)&&(CloudDrops[i].Y<CloudDrops[i-1].Y))){
                    Point tempi;
                    tempi = CloudDrops[i];
                    CloudDrops[i] = CloudDrops[i-1];
                    CloudDrops[i-1]=tempi;
              }
```

```
            }
        }
      }

//The Shown Image Information Setup Function
private: System::Void ShowImageFucation(int imagewidth,int imageheight,int TempTriangle,int
MarkTriangleCount,int MarkRealDropsCount,List<Point>^CloudDrops){
           ImageSizeWidthTextBox->Text=System::Convert::ToString(imagewidth);
           ImageSizeHeightTextBox->Text =System::Convert::ToString(imageheight);
           CloudModelCountTextBox->Text =System::Convert::ToString(TempTriangle);

           CloudModelRealCountTextBox->Text =System::Convert::ToString(MarkTriangleCount);
           RealCDropsTextBox->Text = System::Convert::ToString(MarkRealDropsCount);
           TotalCDropsTextBox->Text = System::Convert::ToString(CloudDrops->Count);
  }

// The Steganos Embedded processing Function
private: System::Void HideDataEmbededProcessing(int IndexCloudNumber,int SizeData,
                        List<int>^DataBlocks,IplImage*grayimage,List<Point>^CloudDrops,
                        int TempTriangle,int PreCDropsCount){
        for(int cl =0; cl<DataBlocks->Count; cl++){
          double OriPixelValue=0;
          double WaterPixelValue =0;
          int CloudIndex =cl+PreCDropsCount;
          int StegoValue = System::Convert::ToInt16(StegoValueTextBox->Text);

          double StrengthFactor = System::Convert::ToDouble(EmStrengthFactorTextBox->Text);
          CvScalar Embedpoint;
          Embedpoint = cvGet2D(grayimage,CloudDrops[CloudIndex].Y,CloudDrops[CloudIndex].X);
          OriPixelValue =Embedpoint.val[0];

          if(DataBlocks[cl] ==1){
             int PixelValue = (int)(Embedpoint.val[0]-(int)Embedpoint.val[0]%StegoValue
                                   +StrengthFactor*StegoValue);
             Embedpoint.val[0] =PixelValue;
             WaterPixelValue =PixelValue;
          }
          else{
            int PixelValue = (int)(Embedpoint.val[0]-(int)Embedpoint.val[0]%StegoValue
                         +(1-StrengthFactor)*StegoValue);
            Embedpoint.val[0] =PixelValue;
            WaterPixelValue =PixelValue;
          }
          cvSet2D(grayimage,CloudDrops[CloudIndex].Y,CloudDrops[CloudIndex].X,Embedpoint);
          DataGridViewCModelShownEData(IndexCloudNumber,SizeData,cl,TempTriangle,
                          CloudIndex,CloudDrops,OriPixelValue,WaterPixelValue,DataBlocks);
        }
 }


//The Sure Location Point function
private: System::Void LocationWatermarkedListWatermarkedPoint(CloudTriangle^VildCT,
                              List<Point>^ListWPoint,double Inradius,int StartWIndex){
        double Ex =0.0;
        double En =Inradius/3;
        double He = En/10;
```

```
        int CDrops =int(0.3*M_PI*Inradius*Inradius);

        CloudModel1DCloudDrops(VildCT,ListWPoint,Ex,En,He,CDrops);
        for(int k =StartWIndex; k<ListWPoint->Count-1;k++){
            double DistanceX = ListWPoint[k].X - VildCT ->center.X;
            double DistanceY = ListWPoint[k].Y - VildCT ->center.Y;
            double DistanceXY = sqrt(DistanceX*DistanceX + DistanceY*DistanceY);

            if((DistanceXY>Inradius)||(DistanceXY == Inradius))
                ListWPoint->RemoveAt(k);
        }

        for(int i =StartWIndex; i<ListWPoint->Count-1;i++){
            CloudDropsBubbleSorting(ListWPoint,i,StartWIndex);
        }
 }

private: System::Int16 SteganoBiDataSizeCalculate(List<int>^SteganoBiData, int
StegoDataSize){
        int Stego =0;
        int HideData =0;

        for(int k = StegoDataSize-1;k>=0;k--){
              HideData += int(pow(double(2),Stego))*SteganoBiData[k];
              Stego++;
        }
        return(HideData);
    }

private: System::Void SteganographydataShownStegoTextBox(List<int>^SteganoBiBlock){
        int Stegano =SteganoBiBlock->Count;
        String ^SteganoHex ="";

        for(int i =0;4*i<Stegano;i++){
            int SteganoEHex =0;
            int k =3;
            for(int j =0;j<4;j++){
               SteganoEHex +=SteganoBiBlock[4*i+j]*(int)pow(double(2),double(k));
               k--;
            }
            SteganoHex +=Convert::ToString(SteganoEHex,16);
        }
        MessageBox::Show(String::Format("HEX:{0},Length:{1}",SteganoHex,SteganoHex->Length));
    }

private: System::Void AboutMenuItem_Click(System::Object^  sender, System::EventArgs^  e) {
         MessageBox::Show(L"Copyright(C) 2008-2010 Mr.Liu Yi",L"CMDWatermarking",
         MessageBoxButtons::OK,MessageBoxIcon::Exclamation);
    }

//The DataGridView Save Button
private: System::Void DataGridViewSaveButton_Click(System::Object^  sender,
System::EventArgs^  e) {
        if(SaveDataGridViewFileDialog->ShowDialog ( ) == System::Windows::Forms
                                           ::DialogResult::OK){
            FileStream^ myStream =gcnew FileStream(SaveDataGridViewFileDialog
```

```
                                              ->FileName,FileMode::Create);
        StreamWriter^ sw=gcnew StreamWriter(myStream,System::Text::Encoding::
                                                GetEncoding(-0));
        if(dataGridView->RowCount !=0){
           for(int i =0;i<this->dataGridView->RowCount;i++){
               String^columnValue ="";
               for(int j=0;j<this->dataGridView->ColumnCount;j++){
                 if(j>0)
                     columnValue +="\t";
                 if(dataGridView->Rows[i]->Cells[j]->Value==nullptr)
                     columnValue +="";
                else
                   columnValue +=dataGridView->Rows[i]->Cells[j]->Value->ToString();
              }
              sw->WriteLine(columnValue);
          }
          myStream->Flush( );
          sw->Close( );
          myStream->Close( );

          MessageBox::Show("Save Successful For DataGridView that Triangle and
                             Steganography Information!","Polite Notice",
                             MessageBoxButtons::OK,MessageBoxIcon::Information);
      }
    }
  }

//The DataGridView Triangle Cloud Model Information Setup Fucation
private: System::Void DataGridViewTriangleCloudModelInformation(int k,CloudTriangle^Tri,
                                        double Inradius){
        array<String^>^Triangleheaders ={L"Index No.",L"Vertex1",L"Vertex2",L"Vertex3",
                     L"InCenter",L"InRadius,L"En",L"He",L"CloudNo.",L"Real CloudNo."};
        if(k==0){
           dataGridView->Rows[k]->Cells[0]->Value=Triangleheaders[0];
           dataGridView->Rows[k]->Cells[1]->Value=Triangleheaders[1];
           dataGridView->Rows[k]->Cells[2]->Value=Triangleheaders[2];

           dataGridView->Rows[k]->Cells[3]->Value=Triangleheaders[3];
           dataGridView->Rows[k]->Cells[4]->Value=Triangleheaders[4];
           dataGridView->Rows[k]->Cells[5]->Value=Triangleheaders[5];

           dataGridView->Rows[k]->Cells[6]->Value=Triangleheaders[6];
           dataGridView->Rows[k]->Cells[7]->Value=Triangleheaders[7];
           dataGridView->Rows[k]->Cells[8]->Value=Triangleheaders[8];
           dataGridView->Rows[k]->Cells[9]->Value=Triangleheaders[9];
        }
        dataGridView->Rows[k+1]->Cells[0]->Value =k+1;
        dataGridView->Rows[k+1]->Cells[1]->Value =Tri->vtx1;
        dataGridView->Rows[k+1]->Cells[2]->Value =Tri->vtx2;

        dataGridView->Rows[k+1]->Cells[3]->Value =Tri->vtx3;
        dataGridView->Rows[k+1]->Cells[4]->Value =Tri->center;
        dataGridView->Rows[k+1]->Cells[5]->Value =Inradius;

        dataGridView->Rows[k+1]->Cells[6]->Value =Inradius/3;
        dataGridView->Rows[k+1]->Cells[7]->Value =Inradius/30;
```

```
        dataGridView->Rows[k+1]->Cells[8]->Value =int(0.3*M_PI*Inradius*Inradius);

        dataGridViewHideData->Rows[k+1]->Cells[0]->Value =k+1;
        dataGridViewHideData->Rows[k+1]->Cells[1]->Value =Tri->vtx1;
        dataGridViewHideData->Rows[k+1]->Cells[2]->Value =Tri->vtx2;

        dataGridViewHideData->Rows[k+1]->Cells[3]->Value =Tri->vtx3;
        dataGridViewHideData->Rows[k+1]->Cells[4]->Value =Tri->center;
        dataGridViewHideData->Rows[k+1]->Cells[5]->Value = Inradius;

        dataGridViewHideData->Rows[k+1]->Cells[6]->Value = Inradius/3;
        dataGridViewHideData->Rows[k+1]->Cells[7]->Value = Inradius/30;
        dataGridViewHideData->Rows[k+1]->Cells[8]->Value = int(0.3*M_PI*Inradius*Inradius);
       }
    }

//The DataGridView Drawing Setup Function
private: System::Void DataGridViewCModelShownEIndex(int TempTriangle,int k,int
DataBlocksCount){
        array<String^>^headers ={L"Cloud Model No.:",L"Cloud Drops Total:",
                                 L"Index No.",L"Embed Location",L"OriPixel Value",
                                 L"EmbPixel Value",L"Block Bits"};
        dataGridView->Rows[TempTriangle+2]->Cells[0]->Value = headers[0];
        dataGridView->Rows[TempTriangle+2]->Cells[0]->Style->BackColor =Color::Pink;
        dataGridView->Rows[TempTriangle+2]->Cells[1]->Value = k+1;
        dataGridView->Rows[TempTriangle+2]->Cells[1]->Style->BackColor =Color::Pink;

        dataGridView->Rows[TempTriangle+2]->Cells[2]->Value = headers[1];
        dataGridView->Rows[TempTriangle+2]->Cells[2]->Style->BackColor =Color::Pink;
        dataGridView->Rows[TempTriangle+2]->Cells[3]->Value = DataBlocksCount;
        dataGridView->Rows[TempTriangle+2]->Cells[3]->Style->BackColor =Color::Pink;

        dataGridView->Rows[TempTriangle+3]->Cells[0]->Value = headers[2];
        dataGridView->Rows[TempTriangle+3]->Cells[1]->Value = headers[3];
        dataGridView->Rows[TempTriangle+3]->Cells[2]->Value = headers[4];
        dataGridView->Rows[TempTriangle+3]->Cells[3]->Value = headers[5];
        dataGridView->Rows[TempTriangle+3]->Cells[4]->Value = headers[6];
      }

//The DataGridView Cloud Model Shown Embedded Data Setup function
private: System::Void DataGridViewCModelShownEData(int IndexCloudNumber,int SizeData,
                        int cl,int TempTriangle,int CloudIndex,List<Point>^CloudDrops,
                        double OriPixelValue,double WaterPixelValue,List<int>^ DataBlocks){
        int SizeDataIndexCloudNumber =IndexCloudNumber+SizeData;
        //The Currently BlocksCell Identify the cell we have entered
        DataGridViewCell^BlocksCell =dataGridView->Rows[cl+TempTriangle+4]->Cells[4];
        //Set BlocksCell colors
        if(cl<IndexCloudNumber){
           for(int i =cl;i<IndexCloudNumber;i++){
             BlocksCell->Style ->BackColor=Color::Red;
           }
        }
        else{
          for(int k=cl;k<SizeDataIndexCloudNumber;k++){
             BlocksCell->Style->BackColor =Color::GreenYellow;
          }
```

```
        if(SizeDataIndexCloudNumber<=cl)
            BlocksCell->Style->BackColor =Color::Gray;
        }

        dataGridView->Rows[cl+TempTriangle+4]->Cells[0]->Value = cl+1;
        dataGridView->Rows[cl+TempTriangle+4]->Cells[1]->Value = CloudDrops[CloudIndex];
        dataGridView->Rows[cl+TempTriangle+4]->Cells[2]->Value = OriPixelValue;

        dataGridView->Rows[cl+TempTriangle+4]->Cells[3]->Value = WaterPixelValue;
        dataGridView->Rows[cl+TempTriangle+4]->Cells[4]->Value = DataBlocks[cl];
  }

//The DataGridViewHideData Database shown function
private: System::Void DataGridViewCModelShownExData(int TempWTriangle,int GridIndex,
                         int StartWIndex,int StartWANDTOIndex,List<Point>^ListWPoint,
                         double WatermarkPixelValue,int WCloudNumber,int PreHideData){
        int BlockValue =0;
        int StegoValue = System::Convert::ToInt16(StegoValueExTextBox->Text);
        double StrengthFactor = System::Convert::ToDouble(ExStrengthFactorTextBox->Text);
        int CheckDataHide =(int)WatermarkPixelValue%StegoValue;

        if( CheckDataHide==StrengthFactor*StegoValue)
                BlockValue =1;
        else if(CheckDataHide==(1-StrengthFactor)*StegoValue)
                BlockValue=0;

        dataGridViewHideData->Rows[GridIndex+TempWTriangle+StartWANDTOIndex+4]
                            ->Cells[0]->Value = GridIndex+1;
        dataGridViewHideData->Rows[GridIndex+TempWTriangle+StartWANDTOIndex+4]
                            ->Cells[1]->Value = ListWPoint[GridIndex+StartWIndex];
        dataGridViewHideData->Rows[GridIndex+TempWTriangle+StartWANDTOIndex+4]
                            ->Cells[2]->Value = WatermarkPixelValue;
        dataGridViewHideData->Rows[GridIndex+TempWTriangle+StartWANDTOIndex+4]
                            ->Cells[3]->Value = BlockValue;
        DataGridViewCell^BlocksCell =dataGridViewHideData->Rows[GridIndex+TempWTriangle
                            +StartWANDTOIndex+4]->Cells[3];

        if(GridIndex <WCloudNumber){
           for(int i =0;i<WCloudNumber;i++){
              BlocksCell->Style ->BackColor=Color::Red;
           }
        }
        else{
           for(int k =GridIndex;k<PreHideData;k++){
               BlocksCell->Style->BackColor =Color::GreenYellow;
           }
           if(PreHideData<=GridIndex)
              BlocksCell->Style->BackColor =Color::Gray;
        }
 }
```

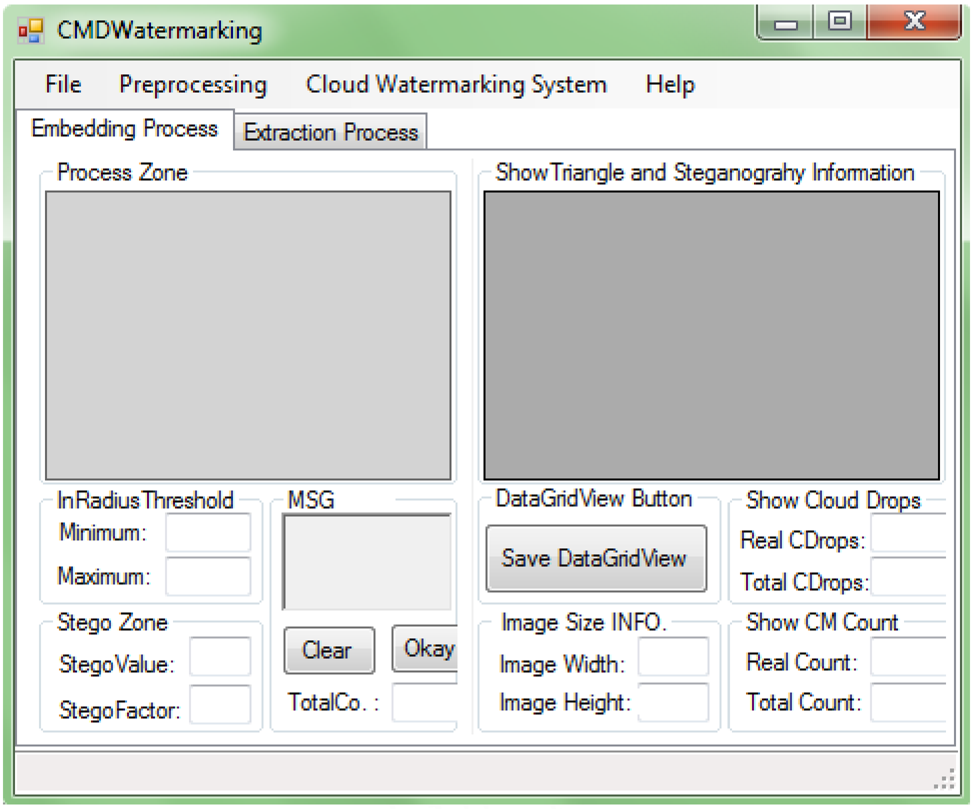


Figure A4.1 Cloud Watermarking Embedding Processing Window

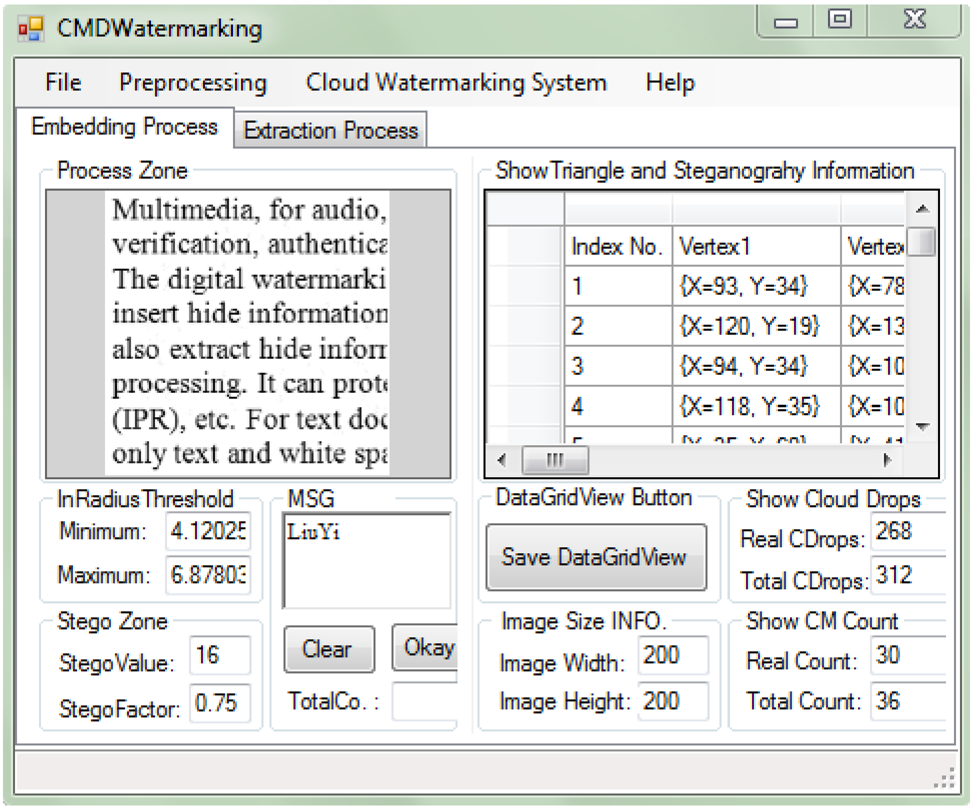


Figure A4.2 Output of Cloud Watermarking of Text document image Embedding Processing Example.

Appendix A5 Digtial Cloud Watermarking Extraction Process Code

```
/* Digital Cloud Watermarking extraction processs Codes
 * By Mr. Liu Yi
 * E-mail:gavinliuyi@hotmail.com/gavinliuyi@yahoo.com
 */

private: System::Void ExtractionMenuItem_Click(System::Object^  sender,
                                               System::EventArgs^  e) {
        if(ExtraTab->Visible){
           if(ExtraPictureBox->Image == nullptr){
                MessageBox::Show(L" Not Watermarked Image,Please Loading
                        Watermarked Image,Thanks you!");
                return;
           }

           Bitmap^WMarkbitmap = (Bitmap^)ExtraPictureBox->Image;//load a watermarked image
           Rectangle rect =Rectangle(0, 0, WMarkbitmap->Width,WMarkbitmap->Height);
           BitmapData^WMarkbmpdata = WMarkbitmap->LockBits(rect,ImageLockMode::ReadWrite,
                                                 WMarkbitmap->PixelFormat);
           IplImage *WMarkimage =cvCreateImageHeader(cvSize(WMarkbitmap->Width,
                                                WMarkbitmap->Height),IPL_DEPTH_8U,3);

           WMarkimage ->imageData = (char *)(void *)WMarkbmpdata ->Scan0;
           WMarkimage ->widthStep = WMarkbmpdata ->Stride;
           IplImage *GrayWMarkimage = cvCreateImage (cvGetSize(WMarkimage),
                                                WMarkimage->depth,1);
           cvCvtColor (WMarkimage,GrayWMarkimage,CV_BGR2GRAY);
           IplImage *CornerFImage = cvCreateImage(cvGetSize(WMarkimage),IPL_DEPTH_32F,1);
           IplImage *CornerUImage = cvCreateImage(cvGetSize(WMarkimage),IPL_DEPTH_8U,1);
           cvCornerHarris(GrayWMarkimage,CornerFImage,3);

           double MinVal = 0.0;
           double MaxVal = 0.0;
           double CornerScale = 0.0;
           double CornerShift = 0.0;

           double min = 0.0;
           double max = 255.0;
           double CornerThreshold = 0.0;

           cvMinMaxLoc(CornerFImage,&MinVal,&MaxVal,NULL,NULL,0);
           CornerThreshold = (MaxVal-MinVal)/5;
           CornerScale = (max-min)/(MaxVal-MinVal);
           CornerShift = MinVal*CornerScale + min;

           cvThreshold(CornerFImage,CornerFImage,CornerThreshold,255,CV_THRESH_BINARY);
           List<Point>^CornerPoint = gcnew List<Point>( ); //Create and a new ArrayList
           List<CloudTriangle^>^ExCloudtriangle = gcnew List<CloudTriangle^>( );

           for(int Horizontal =0; Horizontal <CornerFImage->height;Horizontal++){
               for(int Vertical = 0; Vertical<CornerFImage->width;Vertical++){
                     CvScalar Scalar_FImage;
                     Scalar_FImage = cvGet2D(CornerFImage,Horizontal,Vertical);
```

```
            if(Scalar_FImage.val[0] >CornerThreshold)
                  CornerPoint->Add(Point(Vertical,Horizontal));
      }
}

CloudTriangleDrawingDelaunayTriangulation(CornerPoint,CornerFImage,
                       ExCloudtriangle);//Drawing the Cloud Triangle Function
cvConvertScale(CornerFImage,CornerUImage,CornerScale,CornerShift);
cvNot(CornerUImage,CornerUImage);
WMarkbitmap->UnlockBits(WMarkbmpdata);
RemoveSameCloudTriangle(ExCloudtriangle);

for (int TriThreslod =ExCloudtriangle->Count-1;TriThreslod>= 0;TriThreslod--){
     CloudTriangle^ TriC = ExCloudtriangle[TriThreslod];

     double inradius = CloudTriangleInradius(TriC);
     double Thresholdmin =System::Convert::ToInt16(ThresholdMinTextBox->Text);
     double Thresholdmax =System::Convert::ToInt16(ThresholdMaxTextBox->Text);
     if((inradius<Thresholdmin)||(inradius>Thresholdmax))
         ExCloudtriangle->RemoveAt(TriThreslod);
}

int TempWTriangle = ExCloudtriangle->Count;
int WCloudNumber = (TempWTriangle ==0)?1:
                    int(ceil(log10(double(TempWTriangle))/log10(double(2))));
int StartWIndex =0;
int StartWANDTOIndex =0;

int MarkTriangleCount=0;// Mark Triangle Count
int StegoValue = System::Convert::ToInt16(StegoValueExTextBox->Text);
double StrengthExFactor = System::Convert::ToDouble(ExStrengthFactorTextBox
                           ->Text);

List<Point>^ListWPoint = gcnew List<Point>( );
List<int>^SteganoBiBlock = gcnew List<int>( );
dataGridViewHideData->ColumnCount= 9;
dataGridViewHideData->RowCount =TempWTriangle+3;

for(int IndexCount =0;IndexCount<ExCloudtriangle->Count;IndexCount++){
     int HideBiData =0; // Hide Information Data Binary Bits Count
     int HideBiDataSize =0; // Check in Hide Infromation Count Size
     double Inradius =0;

     CloudTriangle^ VildCT =ExCloudtriangle[IndexCount];
     Inradius =CloudTriangleInradius(VildCT);
     DataGridViewTriangleCloudModelInformation(IndexCount,VildCT,Inradius);
     LocationWatermarkedListWatermarkedPoint(VildCT,ListWPoint,Inradius,
                   StartWIndex);//The ListWPoint add Location Watmermarked Point

     int CloudDropsFCG =ListWPoint->Count-StartWIndex;
     int HideBiDataANDSize =CloudDropsFCG-WCloudNumber;
     int PreHideData=CheckHidingDataLength(HideBiDataANDSize);

     List<int>^SteganoBiData = gcnew List<int> ( );
     HideDataExtractionProcessing(HideBiDataANDSize,GrayWMarkimage,StartWIndex,
                        WCloudNumber,StegoValue,StrengthExFactor,  SteganoBiData);
```

```
  int HideDataSizeCheck = HideBiDataANDSize-PreHideData;
  int RealPreHideData = SteganoBiDataSizeCalculate(SteganoBiData,
                        HideDataSizeCheck);
  while(RealPreHideData>PreHideData){
          PreHideData++;
  }

  HideBiData = PreHideData;

  int SteganBlockCount =SteganoBiData->Count-HideBiData;
  int StegoBlockSize =SteganoBiDataSizeCalculate(SteganoBiData,
                    SteganBlockCount);
  for(int k =0;k<StegoBlockSize;k++){
       SteganoBiBlock->Add(SteganoBiData[k+SteganBlockCount]);
  }

  if(StegoBlockSize!=0){
     int RowsTriangle = TempWTriangle+StartWANDTOIndex;
     int IndexNoANDHideBiDataSize = CloudDropsFCG-HideBiData;

     dataGridViewHideData->RowCount +=CloudDropsFCG+3;
     DataGridViewCModelShownEIndex(RowsTriangle,IndexCount,CloudDropsFCG);

     for(int GridIndex=0;GridIndex<CloudDropsFCG;GridIndex++){
        CvScalar GridPoint =cvGet2D(GrayWMarkimage,
                            ListWPoint[GridIndex+StartWIndex].Y,
                            ListWPoint[GridIndex+StartWIndex].X);
        int CloudIndex = GridIndex+StartWIndex;
        double WatermarkPixelValue =GridPoint.val[0];

        DataGridViewCModelShownExData(TempWTriangle,GridIndex,StartWIndex,
                               StartWANDTOIndex,ListWPoint,
                               WatermarkPixelValue,WCloudNumber,
                               IndexNoANDHideBiDataSize);
      }

  MarkTriangleCount++;
  }
  StartWIndex = ListWPoint->Count;
  StartWANDTOIndex = StartWIndex+3*(IndexCount+1);
}

SteganographydataShownStegoTextBox(SteganoBiBlock);

ShowImageFucation(GrayWMarkimage->width,GrayWMarkimage->height,
                  ExCloudtriangle->Count,MarkTriangleCount,
                  SteganoBiBlock->Count,ListWPoint);
cvCvtColor (GrayWMarkimage,WMarkimage,CV_GRAY2BGR);

ListWPoint->Clear( );
SteganoBiBlock->Clear( );

cvReleaseImageHeader(&WMarkimage);
cvReleaseImage(&GrayWMarkimage);
cvReleaseImage(&CornerFImage);
```

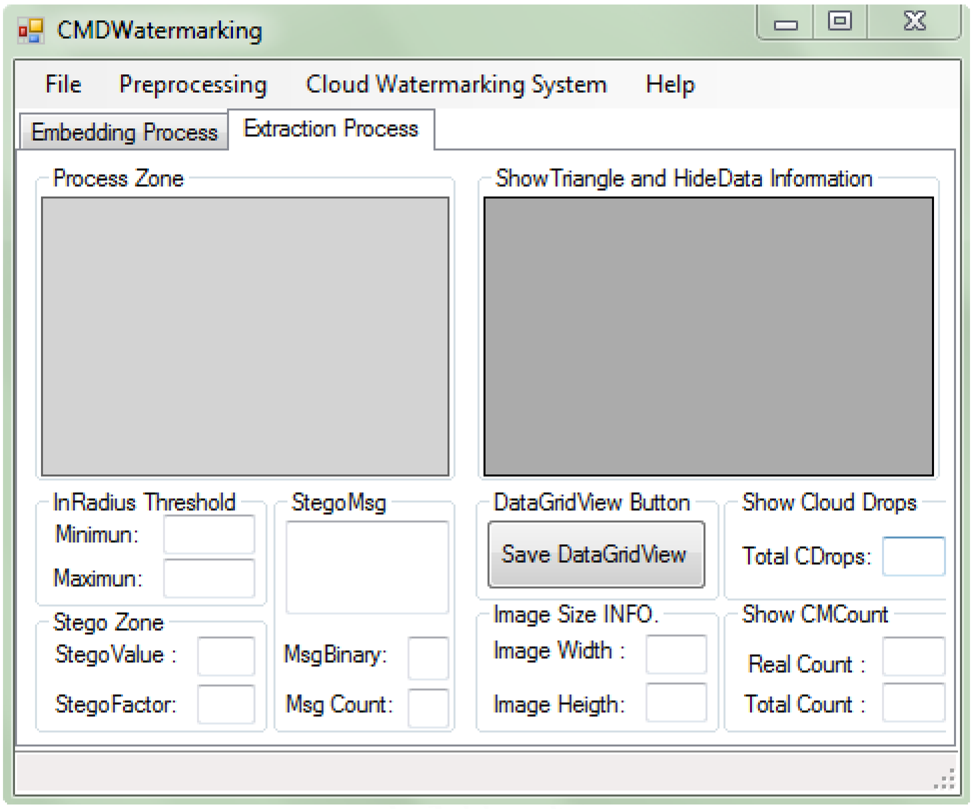
```
                cvReleaseImage(&CornerUImage);

                MessageBox::Show(L"Extraction Processing is finished,please Think you!");
                ExtraPictureBox->Invalidate();
          }
          else
            return;
     }

//The Remove Same Cloud Triangle function
private: System::Void RemoveSameCloudTriangle(List<CloudTriangle^>^cloudtriangle){
          int CloudTCount =cloudtriangle->Count;
          if(CloudTCount>1){
             for(int Index1= cloudtriangle->Count-1; Index1>=0; Index1--){
                  Point Centre1=cloudtriangle[Index1]->center;
                     for(int Index2=cloudtriangle->Count-2; Index2>=0;Index2--){
                          Point Centre2 = cloudtriangle[Index2]->center;
                          if((Index1 !=Index2)&&(Centre1==Centre2)){
                              cloudtriangle->RemoveAt(Index2);
                              Index1--;
                          }
                     }
             }
          }
     }

//The Remove Same Cloud Drops function
private: System::Void RemoveSameCloudDrops(List<Point>^CloudDrops){
          for(int i=CloudDrops->Count-1;i>=0;i--){
               Point Drops1=CloudDrops[i];
               for(int k=CloudDrops->Count-2;k>=0;k--){
                    Point Drops2 =CloudDrops[k];
                    if((i!=k)&&(Drops1==Drops2)){
                          CloudDrops->RemoveAt(k);
                          i--;
                    }
               }
          }
      }

//The Cloud Drops Bubble Sorting Funcation
private: System::Void CloudDropsBubbleSorting(List<Point>^CloudDrops,int j,int IndexCout){
          int i=j;
          if((CloudDrops[j].X>CloudDrops[j+1].X)||((CloudDrops[j].X==CloudDrops[j+1].X)
                 &&(CloudDrops[j].Y>CloudDrops[j+1].Y))){
              Point temp;
              temp =CloudDrops[j];
              CloudDrops[j] = CloudDrops[j+1];
              CloudDrops[j+1] =temp;
          }

          if(i!=IndexCout){
             for(i =j;i>IndexCout;i--){
                if((CloudDrops[i].X<CloudDrops[i-1].X)||((CloudDrops[i].X==
                       CloudDrops[i-1].X)&&(CloudDrops[i].Y<CloudDrops[i-1].Y))){
                      Point tempi;
```

```
                    tempi = CloudDrops[i];
                    CloudDrops[i] = CloudDrops[i-1];
                    CloudDrops[i-1]=tempi;
                }
              }
          }
        }

private: System::Void ShowImageFucation(int imagewidth,int imageheight,int TempTriangle,int
MarkTriangleCount,int MarkRealDropsCount,List<Point>^CloudDrops){
           WaterImageSizeWidthTextBox->Text = System::Convert::ToString(imagewidth);
           WaterImageSizeHeigthTextBox->Text = System::Convert::ToString(imageheight);
           RealCMCountWaterTextBox->Text = System::Convert::ToString(MarkTriangleCount);

           TotalCMCountWaterTextBox->Text = System::Convert::ToString(TempTriangle);
           TotalCDWaterTextBox->Text = System::Convert::ToString(CloudDrops->Count);
  }

private: System::Void HideDataEmbededProcessing(int IndexCloudNumber,int SizeData,
                        List<int>^DataBlocks,IplImage*grayimage,List<Point>^CloudDrops,
                        int TempTriangle,int PreCDropsCount){
        for(int cl =0; cl<DataBlocks->Count; cl++){
          double OriPixelValue=0;
          double WaterPixelValue =0;
          int CloudIndex =cl+PreCDropsCount;
          int StegoValue = System::Convert::ToInt16(StegoValueTextBox->Text);

          double StrengthFactor = System::Convert::ToDouble(EmStrengthFactorTextBox->Text);
          CvScalar Embedpoint;
          Embedpoint = cvGet2D(grayimage,CloudDrops[CloudIndex].Y,CloudDrops[CloudIndex].X);
          OriPixelValue =Embedpoint.val[0];

          if(DataBlocks[cl] ==1){
             int PixelValue = (int)(Embedpoint.val[0]-(int)Embedpoint.val[0]%StegoValue
                                  +StrengthFactor*StegoValue);
             Embedpoint.val[0] =PixelValue;
             WaterPixelValue =PixelValue;
          }
          else{
            int PixelValue = (int)(Embedpoint.val[0]-(int)Embedpoint.val[0]%StegoValue
                           +(1-StrengthFactor)*StegoValue);
            Embedpoint.val[0] =PixelValue;
            WaterPixelValue =PixelValue;
          }
          cvSet2D(grayimage,CloudDrops[CloudIndex].Y,CloudDrops[CloudIndex].X,Embedpoint);
          DataGridViewCModelShownEData(IndexCloudNumber,SizeData,cl,TempTriangle,
                          CloudIndex,CloudDrops,OriPixelValue,WaterPixelValue,DataBlocks);
        }
 }

private: System::Void HideDataExtractionProcessing(int HideBiDataANDSize,
                        IplImage * GrayWMarkimage, int StartWIndex, int WCloudNumber,
                        double StegoValue,double StrengthExFactor, List<int>^SteganoBiData){
        for(int HideDataSizeStart =0;HideDataSizeStart<HideBiDataANDSize;
                                                  HideDataSizeStart++){
            CvScalar HideDateSizePoint=cvGet2D(GrayWMarkimage,
```

```
                                    ListWPoint[StartWIndex+WCloudNumber+HideDataSizeStart].Y,
                                    ListWPoint[StartWIndex+WCloudNumber+HideDataSizeStart].X);
                int BlockBinary =0;
                int HideBiDataSize =(int)HideDateSizePoint.val[0]%StegoValue;

                if(HideBiDataSize==StrengthExFactor*StegoValue)
                    BlockBinary =1;
                else if(HideBiDataSize ==(1-StrengthExFactor)*StegoValue)
                    BlockBinary =0;
                SteganoBiData->Add(BlockBinary);
            }

    }

private: System::Void DataGridViewTriangleCloudModelInformation(int k,CloudTriangle^Tri,
                                                    double Inradius){
        array<String^>^Triangleheaders ={L"Index No.",L"Vertex1",L"Vertex2",L"Vertex3",
                     L"InCenter",L"InRadius,L"En",L"He",L"CloudNo.",L"Real CloudNo."};
        if(k==0){
               dataGridViewHideData->Rows[k]->Cells[0]->Value =Triangleheaders[0];
               dataGridViewHideData->Rows[k]->Cells[1]->Value =Triangleheaders[1];
               dataGridViewHideData->Rows[k]->Cells[2]->Value =Triangleheaders[2];

               dataGridViewHideData->Rows[k]->Cells[3]->Value =Triangleheaders[3];
               dataGridViewHideData->Rows[k]->Cells[4]->Value =Triangleheaders[4];
               dataGridViewHideData->Rows[k]->Cells[5]->Value =Triangleheaders[5];

               dataGridViewHideData->Rows[k]->Cells[6]->Value =Triangleheaders[6];
               dataGridViewHideData->Rows[k]->Cells[7]->Value =Triangleheaders[7];
               dataGridViewHideData->Rows[k]->Cells[8]->Value =Triangleheaders[8];
        }

        dataGridViewHideData->Rows[k+1]->Cells[0]->Value =k+1;
        dataGridViewHideData->Rows[k+1]->Cells[1]->Value =Tri->vtx1;
        dataGridViewHideData->Rows[k+1]->Cells[2]->Value =Tri->vtx2;
        dataGridViewHideData->Rows[k+1]->Cells[3]->Value =Tri->vtx3;

        dataGridViewHideData->Rows[k+1]->Cells[4]->Value =Tri->center;
        dataGridViewHideData->Rows[k+1]->Cells[5]->Value = Inradius;
        dataGridViewHideData->Rows[k+1]->Cells[6]->Value = Inradius/3;
        dataGridViewHideData->Rows[k+1]->Cells[7]->Value = Inradius/30;
        dataGridViewHideData->Rows[k+1]->Cells[8]->Value = int(0.3*M_PI*Inradius*Inradius);
    }

private: System::Void DataGridViewCModelShownEIndex(int TempTriangle,int k,int
DataBlocksCount){
        array<String^>^headers ={L"Cloud Model No.:",L"Cloud Drops Total:",
                                 L"Index No.",L"Embed Location",L"OriPixel Value",
                                 L"EmbPixel Value",L"Block Bits"};
           dataGridViewHideData->Rows[TempTriangle+2]->Cells[0]->Value = headers[0];
           dataGridViewHideData->Rows[TempTriangle+2]->Cells[0]->Style->BackColor
                                                               =Color::Pink;
           dataGridViewHideData->Rows[TempTriangle+2]->Cells[1]->Value = k+1;
           dataGridViewHideData->Rows[TempTriangle+2]->Cells[1]->Style->BackColor
                                                                =Color::Pink;
              dataGridViewHideData->Rows[TempTriangle+2]->Cells[2]->Value = headers[1];
```

```
            dataGridViewHideData->Rows[TempTriangle+2]->Cells[2]->Style->BackColor
                                                              =Color::Pink;

            dataGridViewHideData->Rows[TempTriangle+2]->Cells[3]->Value = DataBlocksCount;
            dataGridViewHideData->Rows[TempTriangle+2]->Cells[3]->Style->BackColor
                                                              =Color::Pink;
            dataGridViewHideData->Rows[TempTriangle+3]->Cells[0]->Value = headers[2];
            dataGridViewHideData->Rows[TempTriangle+3]->Cells[1]->Value = headers[3];
            dataGridViewHideData->Rows[TempTriangle+3]->Cells[2]->Value = headers[5];
            dataGridViewHideData->Rows[TempTriangle+3]->Cells[3]->Value = headers[6];
       }

//The DataGridView Cloud Model Shown Embedded Data Setup function
private: System::Void DataGridViewCModelShownEData(int IndexCloudNumber,int SizeData,
                        int cl,int TempTriangle,int CloudIndex,List<Point>^CloudDrops,
                        double OriPixelValue,double WaterPixelValue,List<int>^ DataBlocks){
        int SizeDataIndexCloudNumber =IndexCloudNumber+SizeData;
        //The Currently BlocksCell Identify the cell we have entered
        DataGridViewCell^BlocksCell =dataGridView->Rows[cl+TempTriangle+4]->Cells[4];
        //Set BlocksCell colors
        if(cl<IndexCloudNumber){
           for(int i =cl;i<IndexCloudNumber;i++){
             BlocksCell->Style ->BackColor=Color::Red;
           }
        }
        else{
          for(int k=cl;k<SizeDataIndexCloudNumber;k++){
             BlocksCell->Style->BackColor =Color::GreenYellow;
          }
        if(SizeDataIndexCloudNumber<=cl)
            BlocksCell->Style->BackColor =Color::Gray;
        }

        dataGridView->Rows[cl+TempTriangle+4]->Cells[0]->Value = cl+1;
        dataGridView->Rows[cl+TempTriangle+4]->Cells[1]->Value = CloudDrops[CloudIndex];
        dataGridView->Rows[cl+TempTriangle+4]->Cells[2]->Value = OriPixelValue;

        dataGridView->Rows[cl+TempTriangle+4]->Cells[3]->Value = WaterPixelValue;
        dataGridView->Rows[cl+TempTriangle+4]->Cells[4]->Value = DataBlocks[cl];
  }

//The DataGridViewHideData Database shown function
private: System::Void DataGridViewCModelShownExData(int TempWTriangle,int GridIndex,
                         int StartWIndex,int StartWANDTOIndex,List<Point>^ListWPoint,
                         double WatermarkPixelValue,int WCloudNumber,int PreHideData){
        int BlockValue =0;
        int StegoValue = System::Convert::ToInt16(StegoValueExTextBox->Text);
        double StrengthFactor = System::Convert::ToDouble(ExStrengthFactorTextBox->Text);
        int CheckDataHide =(int)WatermarkPixelValue%StegoValue;

        if( CheckDataHide==StrengthFactor*StegoValue)
                BlockValue =1;
        else if(CheckDataHide==(1-StrengthFactor)*StegoValue)
                BlockValue=0;
        dataGridViewHideData->Rows[GridIndex+TempWTriangle+StartWANDTOIndex+4]
                           ->Cells[0]->Value = GridIndex+1;
```

```
        dataGridViewHideData->Rows[GridIndex+TempWTriangle+StartWANDTOIndex+4]
                              ->Cells[1]->Value = ListWPoint[GridIndex+StartWIndex];
        dataGridViewHideData->Rows[GridIndex+TempWTriangle+StartWANDTOIndex+4]
                              ->Cells[2]->Value = WatermarkPixelValue;
        dataGridViewHideData->Rows[GridIndex+TempWTriangle+StartWANDTOIndex+4]
                              ->Cells[3]->Value = BlockValue;
        DataGridViewCell^BlocksCell =dataGridViewHideData->Rows[GridIndex+TempWTriangle
                              +StartWANDTOIndex+4]->Cells[3];
        if(GridIndex <WCloudNumber){
           for(int i =0;i<WCloudNumber;i++){
              BlocksCell->Style ->BackColor=Color::Red;
           }
        }
        else{
           for(int k =GridIndex;k<PreHideData;k++){
               BlocksCell->Style->BackColor =Color::GreenYellow;
           }
           if(PreHideData<=GridIndex)
              BlocksCell->Style->BackColor =Color::Gray;
        }
 }

private: System::Void DataGridViewHideDataSaveButton_Click(System::Object^  sender,
                      System::EventArgs^  e)  {
        if(SaveDataGridViewHideDataFileDialog->ShowDialog ( ) ==
                                    System::Windows::Forms::DialogResult::OK){
           FileStream^ myStream =gcnew FileStream(SaveDataGridViewHideDataFileDialog
                                          ->FileName,FileMode::Create);
           StreamWriter^ sw=gcnew  StreamWriter(myStream,System::Text::Encoding
                           ::GetEncoding(-0));
          if(dataGridViewHideData->RowCount !=0){
             for(int i =0;i<this->dataGridViewHideData->RowCount;i++){
                 String^ColumnValue ="";
                 for(int j=0;j<this->dataGridViewHideData->ColumnCount;j++){
                     if(j>0)
                          ColumnValue +="\t";
                     if(dataGridViewHideData->Rows[i]->Cells[j]->Value==nullptr)
                          ColumnValue +="";
                     else
                       ColumnValue +=dataGridViewHideData->Rows[i]->Cells[j]
                                     ->Value->ToString();
                }
                sw->WriteLine(ColumnValue);
           }
           myStream->Flush( );
           sw->Close( );
           myStream->Close( );
           MessageBox::Show("Save Successful for Extraction DataGridViewHideData!",
                         "Polite Notice",MessageBoxButtons::OK,MessageBoxIcon::Information);
           }
        }
    }
```



FigureA 5.1: Cloud Watermarking Embedding Processing Window

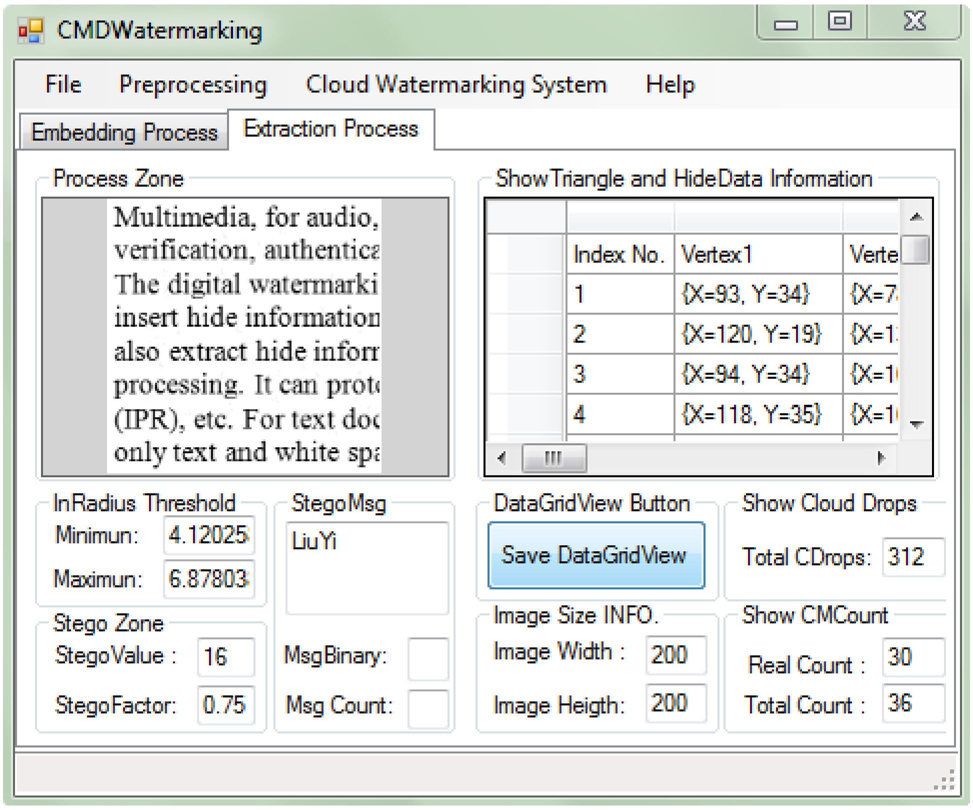
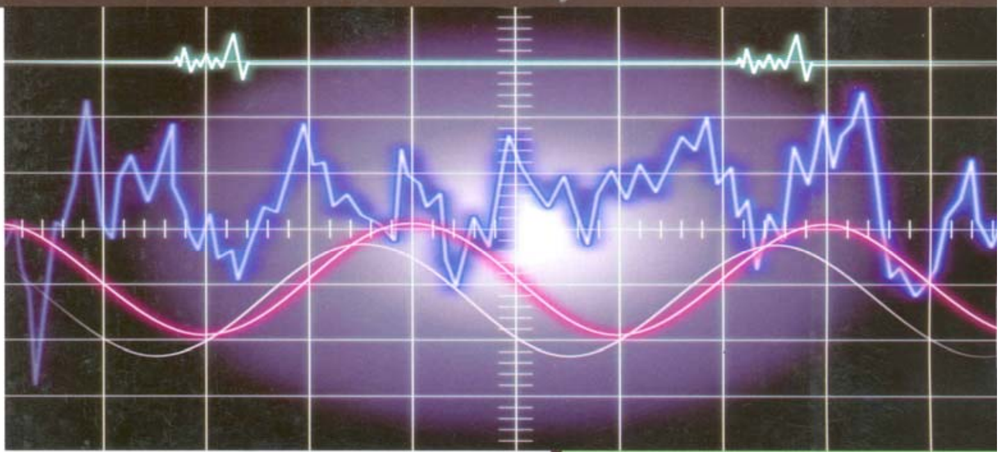


Figure A5.2: Output of cloud watermarking of text document images extraction processing example

APPENDIX B

Published Paper

2011 4th International Congress on Image and Signal Processing



CISP 2011

15-17 October 2011
Shanghai, China

Editors:
Peihua Qiu
Yong Xiang
Yongsheng Ding
Demin Li
Lipo Wang

東華大學 IEEE EMB

Volume 2

2011 4th International Congress on Image and Signal Processing

Data Hiding in Text Document Images by Cloud Model

Yi Liu
Dept.of Computer Engineering, Faculty of Engineering
Prince of Songkla University
Hat Yai, Songkhla, Thailand

Somchai Limsiroratana and Anant Choksuriwong
Dept. of Computer Engineering, Faculty of Engineering
Prince of Songkla University
Hat Yai, Songkhla Thailand

***Abstract* – In this paper, we proposed the new design for data hiding in text document images by cloud model. The design of watermark embedded formula primly balances the contradiction between transparence and robustness. The suitable location of pixel is determined by cloud model generator and corresponds to reference point at center of in-circle of triangle. Hence, the watermark can be extracted without referring to the original image. The suitable locations of pixel are limit inside in-circle of triangle, which are three vertexes of Harrier corner points. Experimental results using text document image include Chinese and English are provided to indicate its practical applicability.**

Keywords - digital watermarking; cloud model; text; document images; data hiding

I. INTRODUCTION

During the past decade, it has been see an explosion in the use of digital Multimedia, for audio, image and video, which required owner verification, authentication, copy protection and legacy enhancement. The digital watermarking is a technology for these requirements, to insert hide information in to cover image in embedding processing, and also extract hide information from watermarked image in extraction processing. It can protected author copyright/ intellectual property rights (IPR), etc..

For text documents, data hiding is very hard because there are only text and white space. If we apply signals on white space, it is very easy to visible. Many researches applied signal on text in binary image are proposed [1-8]. However, the capacities of these methods are rather low or middle.

One early research on data hiding in binary image of text document images is to hiding information by line-shift, word-shift [1] and character-shift [2]. In which each line, word or character is selected to carry watermark is slightly moved correspond to the value of watermark bit. The extraction process is achieved by comparing the distances between the references. These methods are limit to uniformly distance of line, word or character. Hence, the original document is no need for the detection process. The capacity of these methods is low.

Other research on data hiding in binary images is to hiding information by how to find suitable locations [3],[7].In which suitable locations is be watermark by modifying the stroke character or find suitable location to embedding watermarking. The detection requires an decorate extraction of character and detection watermarking with need the original image. These methods are limited to the scanned grayscale document images and the visual distortion is low.

So, we present method for text document image which use data hiding in text document images by cloud model, similar to find suitable location to embedding watermarking, however, unlike to those methods. Then the design for watermark embedded formula primly balances the contradiction between transparence and robustness. Our method pretends to protect the original document form unauthorized changes from third persons and blind detection watermark.

Cloud model presented by academician Prof. De-Yi Li provides a kind of new thought for retrieval of uncertain data. Now, cloud model is major address to data mining [9], medical field [10], and wireless sensor networks [11]. Up to now, R.D.Wang et al.[12] propose to the watermark cloud drops by using their own audio features cloud model combine together, but limit to audio watermarking and human experience to set parameters of cloud model. And Y. Zhang et al.[13] the method of protecting relational databases copyright with cloud watermarking is urgent to solve the problem of relational databases copyright in the need of databases security, and has detestability and robustness. But also is only on the numerical attributes and cloud watermark detection needs the original relational databases.

In this paper, we propose focuses on research and apply for cloud model in watermarked image. Then we have been embedding data hiding processing and extracting hiding information. The rest of this paper is organized as follows, in section II, we briefly introduce the cloud model, radius of the in-circle of a triangle and their properties, and bi-blocks. The proposed embedding and extracting schemes are described in section III, the experimental results are described in section IV, followed by a conclusion in section V.

II. CLOUD MODEL, RADIUS OF THE IN-CIRCLE OF A TRIANGLE AND BI-BLOCKS

A. Normal Cloud

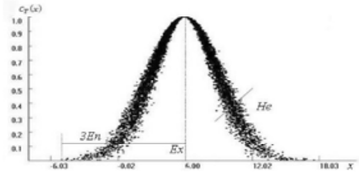


Figure 2.1 Cloud model and its digital characteristic with $E_x = 6, E_n = 3, H_e = 0.3$.

Cloud model is an uncertainty model used to realize the transition between quality and quantity [14].Let U is a universal set described by precise numbers, and T be the qualitative concept related to U . If there is a number $x \in U$, which randomly realizes the concept T , and the certainty degree of x for T , name is $C_T(x)$ which the uncertain

978-1-4244-9305-0/11/$26.00 ©2011 IEEE

number with the certain tendency is. The number feature of normal cloud is signified with the expected value (E_x), entropy (E_n) and hyper entropy (H_e).An example of three numerical characteristics of linguistic term denotes in Fig. 2.1, with cloud drops are 10000.

Expected value is the mathematical expectation of the cloud drop distributed in the universal set. The entropy is the uncertainty measurement of the qualitative concept, which determined by both the randomness and randomness. The hyper-entropy is the uncertainty measurement of the entropy. All cloud drops would be much convergent if H_e is very small, or relative to E_n is very big, the cloud will behave as the shops of fog on the whole, and the cloud is fog. As a single cloud drops of little or no practical significance, only the whole shape of all cloud drops can have some meaning.

B. Normal Cloud Generator

The normal cloud is the most basic tool to express the language value and can be generated by cloud digital character three parameters (E_x, E_n, H_e) .The algorithm of the normal Cloud Generator (CG) (See Fig. 2.2) is as follows:

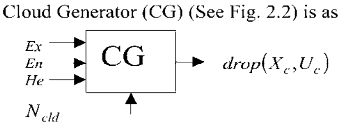


Figure 2.2 Cloud Generator

Input: three parameters (E_x, E_n, H_e) and the required number of Cloud drops (N_{cld}).

Output: N_{cld} of cloud drops X_c and their certainty degree U_c , $c = 0,1,2....N_{cld}$.

1) $En_c^{'} = G(E_n, H_e^{2})$, generating a normal random number $En_c^{'}$ with expectation E_n , and variance H_e^{2} .
2) $X_c = G(E_x, E_{n_c}^{'})$, generating a normal random number X_c with expectation E_x , and variance $En_c^{'}$.
3) Calculate $U_c = \exp\left\{-(X_c - E_x)^2 / 2*(E_{n_c}^{'})^2\right\}$, (U_c, X_c) is a cloud drop.
4) Repeat steps 1) to 3) until N_{cld} cloud drops are generated.

This algorithm is applicable to the one dimensional universal space situation. Generally speaking, the variance should not be zero while generating normal random numbers If $H_e = 0$, as a result X_c will become a normal distribution. If $H_e = 0$, $E_n = 0$, the generated X_c will be a constant E_x and $U_c \equiv 1$ .By this means, we can see that certainty is the special case of the uncertainty. In this paper, we have give cloud model three parameters E_x equal to *Zero*. And He equal to E_n/α , α is constant. So, how to obtain to E_n is very importation in cloud model and center point of cloud model.

C. Radius of the In-circle of a Triangle

For solving to two problems, which are E_n and cloud model center point, we address to Radius of the In-circle of a triangle.Visaing the Delaunay triangulation algorithm drawing the triangle. Then, they have to begin by finding the radius of the in-circle in any triangle. The center of the in-circle of a triangle is located at the intersection of the angle bisectors of the triangle (see Fig. 2.3). Given the side lengths of the triangle, it is possible to determine the radius of the in-circle.

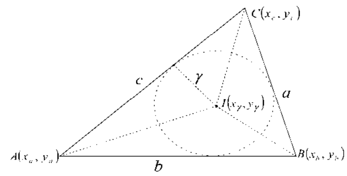


Figure 2.3 Radius of a triangle

Let a, b and c be the sides of a triangle and the three Cartesian vertices are located at $A(x_a, y_a)$, $B(x_b, y_b)$ and $C(x_c, y_c)$, Then radius γ of the in-circle is given by

$$\gamma = \sqrt{(s-a)(s-b)(s-c)/s} \qquad (1)$$

and the Cartesian coordinates of the in-center $I(x_\gamma, y_\gamma)$ are given by

$$\begin{cases} x_\gamma = (ax_a + bx_b + cx_c)/2s \\ y_\gamma = (ay_a + by_b + cy_c)/2s \end{cases} \qquad (2)$$

where $s = (a+b+c)/2$ is the semi-perimeter.

This radius of the in-circle of a triangle is ensured to embed/extract watermark zone and chosen to radius threshold, to avoid a small/larger triangle embedded processing. Center point of cloud model is the Cartesian coordinates of the in-center point.

D. Bi-Blocks

We have been bi-block in the unicode hide information, which composed '0'and'1' form message binary and prefix identify data in the block. Bi-block includes in cloud number, hide information size and hide information (See Fig. 2.4).

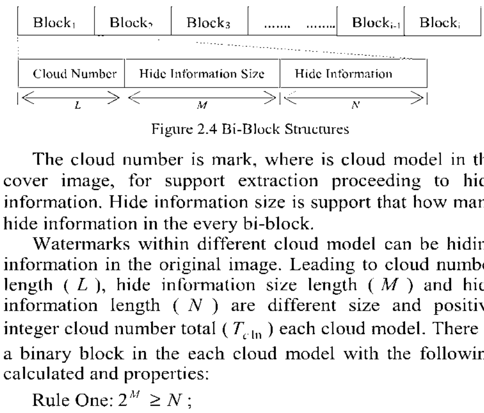


Figure 2.4 Bi-Block Structures

The cloud number is mark, where is cloud model in the cover image, for support extraction proceeding to hide information. Hide information size is support that how many hide information in the every bi-block.

Watermarks within different cloud model can be hiding information in the original image. Leading to cloud number length (L), hide information size length (M) and hide information length (N) are different size and positive integer cloud number total ($T_{c\ln}$) each cloud model. There is a binary block in the each cloud model with the following calculated and properties:

Rule One: $2^M \geq N$;

Rule Two: $N_{cld} = L + M + N$;

Cloud Number: $L \geq \log_2(T_{c\ln})$;

Hide Information Size:

$$2^M + M + L \geq N_{cld}, M = 1,2...N_{cld} - L.$$

These bi-blocks are generator to in the embed process and reading to extraction process unicode hide information. For example, consider a bi-block that includes the following bit string: 011111001001 with T_{cln} : 3 and N_{cld} : 12.

Cloud Number: $L \geq \log_2 3 \leftrightarrow L = 2$;

Hide Information Size: $2^M + M + 2 \geq 12 \leftrightarrow M = 3$

The bi-block can be segmented as follow:

01 111 1001001.

The first digital watermark carries the first portion (01) of the cloud model number. The second digital watermark carries the second portion (111) of hide information Size and the third digital watermark carries the third portion (1001001) of partition the unicode hide information. All three watermarks must be successfully decoded to recover the complete bi-block.

III. THE PROPOSED WATERMARKING SCHEME

A. Watermark Embedded procedure

The basic idea is to embed watermark into cover image so that the cover image becomes watermarked and it is also to make sure that the watermark is secure in the watermark image and should be recovered from the watermarked image later on. Fig. 3.1 displays the block diagram of the proposed embedding algorithm. The detailed embedding processes are as follows:

1) Let the original text document image size NxM dented as $\{f(x,y), 1 \leq x \leq N, 1 \leq y \leq M\}$, where (x,y) represent pixel coordinate of the original text document image. The Harris corner detection algorithm [15] is used find the corner point.
2) Then the three corner points, which are closely to corner points, the delaunay triangulation algorithm [16] is used drawing triangle, then, eqs.(1),(2) Calculate to radius γ and in-center $I(x_\gamma, y_\gamma)$. According to radius length, we have been sure radius minimum and maximum threshold, for select to triangle adjust embedded pixel location in the cloud model. And N_{tri} is total number of triangles.
3) In probability theory, the normal distribution properties, between standard deviation and confidence interval about 99.7% are within three standard deviations. So, they have been sure *En*, which is cloud model digital character one of three major parameters.
4) According to in-circle in each triangle of calculate to cloud drops total N_{cld} and the cloud model parameters *En* equal to $\gamma / 3$, then, generator to cloud drops and processing that base on reference point have ensure "suitable" pixel location. So, they have been known embedding pixel coordinate in the embedding process. At the same time, input unicode hide information covert to binary bit.
5) Then, according to bi-blocks partition correspond to bi-blocks each cloud model. The data hiding information of watermarking for each image pixel is change pixel value in the invisible watermark embedding.

$$f_w(x,y) = \begin{cases} f(x,y) - f(x,y) \bmod s_v + s_f * s_v, \\ \quad if \quad w=1, and \; (1-s_f)*s_v \leq f(x,y) \bmod s_v. \\ \\ [f(x,y) - (1-s_f)*s_v] - [f(x,y) - (1-s_f)*s_v] \\ \bmod s_v + s_f * s_v, \\ \quad if \quad w=1, and \; (1-s_f)*s_v > f(x,y) \bmod s_v. \\ \\ f(x,y) - f(x,y) \bmod s_v + (1-s_f)*s_v, \\ \quad if \quad w=0, and \; s_f * s_v \leq f(x,y) \bmod s_v. \\ \\ [f(x,y) - 0.5*s_v] - [f(x,y) - 0.5*s_v] \bmod s_v \\ + (1-s_f)*s_v, \\ \quad if \quad w=0, and \; s_f * s_v > f(x,y) \bmod s_v. \end{cases}$$

Where $f_w(x,y), f(x,y)$ denotes original image and watermarking image location (x,y) pixel value, s_f is stego factor and $0.5 < s_f < 1$, s_v is stego value and should be maximized under the constraint of invisibility. And the difference $f_w(x,y)$ and $f(x,y)$ is between $-0.5*s_v$ and $+0.5*s_v$. When $w=1$, $f(x,y) \bmod s_v = s_f * s_v$; When $w=0$, $f(x,y) \bmod s_v = (1-s_f)*s_v$.

6) Process the next cloud model.
7) Repeat steps 3) to 6) until N_{tri} watermarking processing.

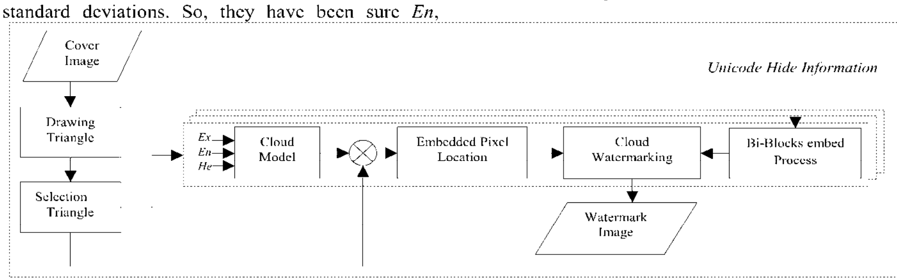


Figure 3.1 Block diagram for watermark embedding process

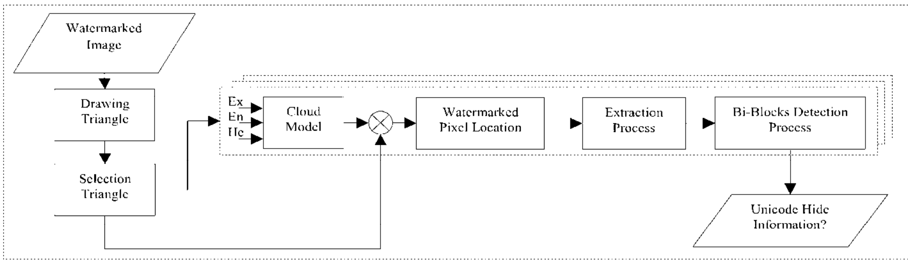


Figure 3.2 Block diagram for watermarking extraction scheme

It should be noted that the "suitable" pixel location are those cloud drops that are selected and used in the data embedding process. In order to guarantee the correct extraction processing of watermark, we have to ensure that the suitable location of embedded pixel coordinate in the watermark embedding process.

B. *Watermark Extraction procedure*

The extraction scheme is to extract the watermark for the watermarked image to further prove the ownership. The watermarking extraction part is illustrated in Fig. 3.2.

1) The corner detection algorithm is used to detect the corner point used for the cloud model location during the embedding process. Then the delaunay triangulation algorithm generated the triangle for detection to location to pixels value in the watermarked image.
2) Then, eqs.(1),(2) Calculate to radius γ and in-center $I(x_\gamma, y_\gamma)$. According to radius γ in the each triangle, we have to minimum and maximum threshold for select to triangle. And N_{tri} is total number of triangles.
3) Using cloud model generator to cloud drops in the every triangle. Then, we have been to ensure "Watermarked" pixel location.
4) The extraction information data of watermarked for watermarked image.
$$w = \begin{cases} 1, & if \quad f_w(x,y) \bmod s_v > 0.5 * s_v \\ 0, & if \quad f_w(x,y) \bmod s_v \le 0.5 * s_v \end{cases}$$, Where $f_w(x,y)$ is coordinate (x,y) pixel value in watermarked image.
5) Chick cloud number, hide information size and hide information. When bi-block is OKAY each cloud model, remove the bi-block of cloud number length and bi-block of hide information size length. Save to hide information in the cloud model.
6) Process the next cloud model.
7) Repeat steps 3) to 6) until N_{tri} .Then all of hide bi-block using to Unicode hide information conversation to hide information Message.

IV. EXPERIMENTAL RESULTS

The proposed method has been implemented and several tests are conducted. They have been known that data hiding processing two parameters are s_f equal to 0.75, and s_v equal to 16, 32.Two text documents image have been used in the experiments, including on Chinese and English text documents.

Comparison of the different language text document has made, the results can be seen on Fig. 4.1 are text documents image by cloud model. It can be observed from the results that in general from $s_v = 16$ or 32, the visual quality of the text documents image is different visual distortion.

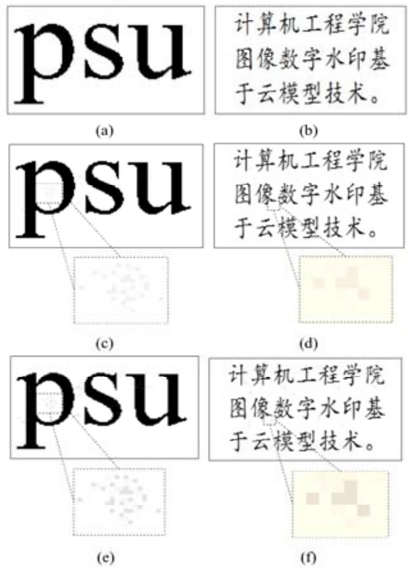
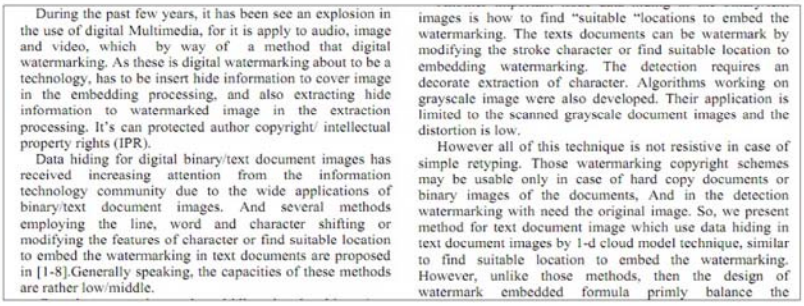


Figure 4.1 Comparison of different text document by cloud model: (a) the original English logo image which is of size147x87.(c) s_v =16,and(e) s_v =32, 188 bits are embedded by the proposed approach watermarked image with 10 number of cloud models and radius threshold minimum is 5,maximum is 20.(b) the original Chinese Image, which is of size 236x108, (d) s_v =16, and(f) s_v =32, 111 bits are embedded by the proposed approach watermarked image with 13 number of cloud models and radius threshold minimum is 5, maximum is 10.

Extensive experiment is conducted to test the large watermarked image, the result show in Fig. 4.2, which is hided 1583bits by the propose algorithm.



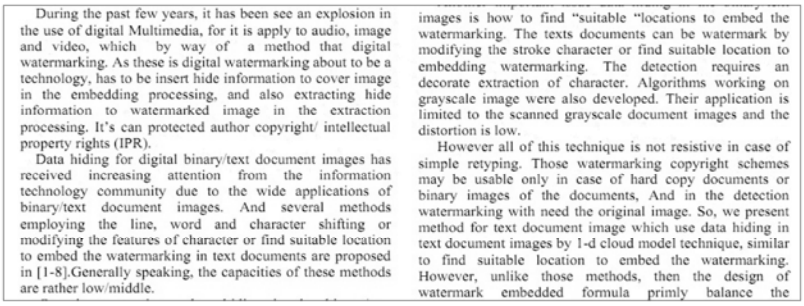
During the past few years, it has been see an explosion in the use of digital Multimedia, for it is apply to audio, image and video, which by way of a method that digital watermarking. As these is digital watermarking about to be a technology, has to be insert hide information to cover image in the embedding processing, and also extracting hide information to watermarked image in the extraction processing. It's can protected author copyright/ intellectual property rights (IPR).

Data hiding for digital binary/text document images has received increasing attention from the information technology community due to the wide applications of binary/text document images. And several methods employing the line, word and character shifting or modifying the features of character or find suitable location to embed the watermarking in text documents are proposed in [1-8].Generally speaking, the capacities of these methods are rather low/middle.

images is how to find "suitable "locations to embed the watermarking. The texts documents can be watermark by modifying the stroke character or find suitable location to embedding watermarking. The detection requires an decorate extraction of character. Algorithms working on grayscale image were also developed. Their application is limited to the scanned grayscale document images and the distortion is low.

However all of this technique is not resistive in case of simple retyping. Those watermarking copyright schemes may be usable only in case of hard copy documents or binary images of the documents, And in the detection watermarking with need the original image. So, we present method for text document image which use data hiding in text document images by 1-d cloud model technique, similar to find suitable location to embed the watermarking. However, unlike those methods, then the design of watermark embedded formula primly balance the

(a)



During the past few years, it has been see an explosion in the use of digital Multimedia, for it is apply to audio, image and video, which by way of a method that digital watermarking. As these is digital watermarking about to be a technology, has to be insert hide information to cover image in the embedding processing, and also extracting hide information to watermarked image in the extraction processing. It's can protected author copyright/ intellectual property rights (IPR).

Data hiding for digital binary/text document images has received increasing attention from the information technology community due to the wide applications of binary/text document images. And several methods employing the line, word and character shifting or modifying the features of character or find suitable location to embed the watermarking in text documents are proposed in [1-8].Generally speaking, the capacities of these methods are rather low/middle.

images is how to find "suitable "locations to embed the watermarking. The texts documents can be watermark by modifying the stroke character or find suitable location to embedding watermarking. The detection requires an decorate extraction of character. Algorithms working on grayscale image were also developed. Their application is limited to the scanned grayscale document images and the distortion is low.

However all of this technique is not resistive in case of simple retyping. Those watermarking copyright schemes may be usable only in case of hard copy documents or binary images of the documents, And in the detection watermarking with need the original image. So, we present method for text document image which use data hiding in text document images by 1-d cloud model technique, similar to find suitable location to embed the watermarking. However, unlike those methods, then the design of watermark embedded formula primly balance the

(b)

Figure 4.2 Test large watermark image result (a) the original image of size 874x350.(b) Hide 1583 bits by propose algorithm, with 199 number of cloud models and radius minimum is 5, maximum is 10.

V. CONCLUSIONS

In this paper, a novel data hiding in the text document images by cloud model is proposed. The radius of the in-circle of a triangle threshold range from 5 to 10 is chosen in ordering to increase the data hiding capacity, and $s_v = 16$ is the best that give low visual distortion.

The experiment results show that our method performs middle data hiding capacity. We may apply other technology such as compression, error control coding, etc. to improve the performance in the future.

REFERENCES

[1] S.H. Low, N. Maxemchuk, J.T. Brassil and L. O'Gorman, "Document marking and identification using both line and word shifting", Proceeding of IEEE infocme'95, Vol. 2,pp.853-860,April 1995.

[2] Nopporn Chotikakamthorn, "Document image data hiding techniques using character spacing width sequence coding", in Proc.1999 Int.Conf. Image processing, vol.2, pp250-254, Oct. 24-28, 1999.

[3] Q. Mei, E.K. Wong and N. Memon, "Data Hiding in Binary Text Documents", Proceedings of SPIE, vol.4314, pp.369-375, 2001.

[4] Y. Liu, J. Mant, E. Wong and S.H. Low, "Making and Detection of Text documents using Transform-Domain Techniques", Processing of SPIE, vol.3657,pp.300-303,Dec.2002.

[5] Min Wu and Bede Liu, "Data Hiding in Binary Images for Authentication and Annotation", IEEE Trans. on Multimedia, vol.6, no.4, pp.528-538, August 2004.

[6] HuiJuan Yang and Alex C. Kot, "Binary Image Authentication with Tampering Localization By Embedding Cryptographic Signature and Block Identifier", IEEE Signal Processing Letters, vol.13, no.12, pp.741-744, Dec.2006.

[7] HuiJuan Yang and Alex C. Kot, "Pattern-Based Data Hiding for Binary Images Authentication by Connectivity-Preserving", IEEE Trans. On Multimedia, vol.9, no.3, pp.475-486, April 2007.

[8] HuiJuan Yang and Alex C. Kot, "Backward-Forward Distortion Minimization for Binary Images Data Hiding", ISCAS 2008, pp.404-407, May 2008.

[9] Hong-li Wang, Li Meng , Mu-kun Cao and Yan Li, "Data Mining Application Based on Cloud Model in Spatial Decision Support System", CCCM 2008,pp.547-551,Aug.2008

[10] Yunqiang Shi, Xianchuan Yu, "Image Segmentation Algorithm Based on Cloud Model the Application of fMRI", ICICTA 2008, vol.2, pp.136-140, Oct.2008.

[11] R. Wang, W. G. Wan, X.. L. Ma, and Y.P. Li "Cloud Model-based Control Strategy on Cluster Communication Coverage for Wireless Sencor Networks", the 2nd International Asia conference on CAR 2010,Vol.1,pp.307-310,April 2010

[12] R.D.Wang and Yi-qun Xiong, "An Audio Aggregate watermark Based on Cloud Model", ICSP 2008 Proceedings, pp.2225-2228. Dec.2008

[13] Y. Zhang, X. Niu and D. Zhao, "A Method of protecting Relational Databases Copyright with Cloud Watermark", Intentional of information Technology, pp. 112-116. 2004.

[14] De-yi Li, D. Cheung, Xuemei Shi and V.NG, "Uncertainty Reasoning Based on Cloud Models in Controllers", Computers Math. Application Vol.35, No.3, pp.99-123, 1998.

[15] C. Harris and M. Stephen, "A combined corner and edge detector", In Proceedings of the 4th Alvey Visin Conference, pp.147-151, 1988.

[16] B. Delaunay, "Sur la sphere vide", Izveria Akademii Naukk SSSR, Otdelenie Matematicheskikh I Estestvenny kh Nauk 7, pp.793-800, 1934.

VITAE

Name Mr. Liu Yi

Student ID 5110120136

Educational Attainment

Degree	Name of Institution	Year of Graduation
Bachelor of Engineering	Jiangxi University of Science and Technology	2007

List Publication and Proceedings

[1] Yi Liu, Somchai Limsiroratana and Anant Choksuriwong , "Data Hiding in Text Document Image By Cloud Model", The 4th International Congress on Image and Signal Processing (CISP'11), Vol.2 pp.1039-1043, Shanghai, P.R. China,15-17 October, 2011.