e-Text Watermarking: Utilizing 'Kashida' Extensions in Arabic Language Electronic Writing

Adnan Abdul-Aziz Gutub
Computer Engineering Department
King Fahd University of Petroleum & Minerals, Dhahran 31261, Saudi Arabia
Email: gutub@kfupm.edu.sa

Fahd Al-Haidari, Khalid M. Al-Kahsah, Jamil Hamodi
King Fahd University of Petroleum & Minerals, Dhahran 31261, Saudi Arabia
Email: fahdhyd@kfupm.edu.sa, kkahsah@hotmail.com, jamil_hamodi@yahoo.com

***Abstract*— Digital watermarking is the process of embedding information into a digital signal. This work targets web applications that need to have hidden secure data within their Arabic e-text files. Many related watermarking techniques have been proposed as for text watermarking. However, most of them are suitable for English and cannot be generalized for different other languages such as Arabic. Arabic e-text watermarking is found having unique characteristics features that can be considered interestingly. In this paper, we are utilizing the extension Arabic character 'Kashida' to propose an improved method for Arabic e-text watermarking. We utilize all the extendable characters possibly fitted in words to represent some watermark bits. We embed bits within 'Kashida' characters in the cover text based on a secret key similar to classical cryptography. Our study showed that this watermarking scheme made the task of an attack much harder compared to previous similar and related methods. It also showed possibility to hide more secret data bits without degrading the security, which is believed to be attractive for web e-text data application such as preserving intellectual properties or copyright features.**

***Index Terms*—Arabic text, Cryptography, Feature coding, Information security, Steganography, Text watermarking.**

I. Introduction

Information hiding and data security consists of several areas involving different techniques needed by current applications of today's electronic world. This work focuses on watermarking, very related to steganography, concentrating mainly on hiding secret data bits in a cover media without being noticed. Our objective is to protect the secret data from detection by other parties. The application of this watermarking technique is to serve for copyright and intellectual protection purposes. The focus is on text watermarking by inserting secret data bits to define the owner of the media. This research objective can be extracted from Figure 1 (taken from [2]), which is separating watermarking from fingerprinting [1], both serving 'Protection against removal' under the main wide range security topic of steganography.

Manuscript received January 1, 2008; revised June 1, 2008; accepted July 1, 2008.

Correspondence Author: Adnan Gutub {gutub@kfupm.edu.sa}

Digital watermarking is the process of embedding information into a digital signal. The signal could be any type of media and if the signal is copied, then the information is also carried in the copy. The use of the word of watermarking is derived from the much older notion of placing a visible watermark on paper [3].

Watermarking can be divided into two categories: visible and invisible. In the visible watermarking, the embedded information into the media is visible, i.e. to identify the owner of the media. However, in the invisible watermarking, digital data is inserted into the cover media invisibly. Application to the invisible watermarking is the copyright protection systems, which are utilized to avoid unauthorized copying of digital media [3].

There are three main features to be considered for any watermarking system [4]:

- Watermark must be robust. It cannot be removed or destroyed without destroying the watermarked media.
- The original and watermarked media should be identical, or observed as impossible to tell apart.
- Unauthorized parties should not be able to find out the watermark or modify it.

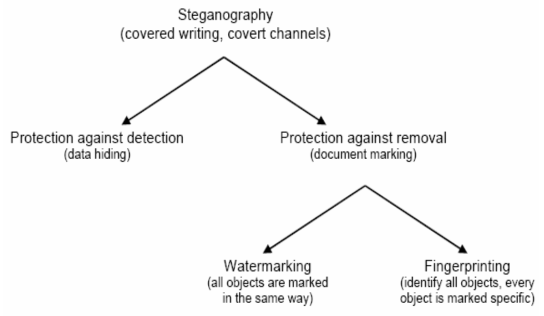

Steganography
(covered writing, covert channels)
Protection against detection
(data hiding)
Protection against removal
(document marking)
Watermarking
(all objects are marked in the same way)
Fingerprinting
(identify all objects, every object is marked specific)

Figure 1: Watermarking within steganography types

There are many techniques proposed for text watermarking. However, very few researches are found concerned with Arabic text watermarking. For example a method based on point-shifting is described in [5]. Another method used the extension character 'Kashida' inserted with pointed and un-pointed letters to hold secret bits '1' and '0', respectively, was proposed in [6].

We improve this Kashida method for Arabic text watermarking but based on a secret seed. The seed is used like a crypto key but to update the cover Arabic e-text before embedding the secret watermark information. In this method, we try to improve the security and capacity by utilizing more of the extension Kashida characters. This proposed approach is discussed in details in Section 3 after providing a brief literature review in Section 2. Section 4, later, shows our experimental work and the result achieved. We discuss and analyze the result in Section 5. Then, the work is closed with conclusion statement in Section 6.

II. Literature review

Several researchers contributed to digital text watermarking [9, 11, 13, 18, 21, 22]. The noticed commonly used techniques to hide watermarking information in most of them can be summarized as line-shift coding, word-shift coding, plus feature coding and inter-word space.

Three coding methods for discouraging unauthorized distribution have been performed by inserting a unique codeword to each document [9]. Several methods [11, 13], i.e. line-shifting, word-shifting and character modification, were embedding data within text image. These methods are robust; however, they could not directly protect embedded data.

Different methods for marking and decoding the mark from image documents have been described and compared in [12]. The researchers in [12] described architecture for distributing a large number of copies without making any load to the publisher to create and transmit the unique documents. This architecture allowed the publisher to define the identity of the document and receiver. It would be noticed if the document had been illegally redistributed; which is interesting since it is not compromising the privacy and work of normal legal individuals.

Embedding data in the 8-connected boundary of a character was a new method for data hiding in binary text proposed by [15]. This method could be applied to other types of binary images that contain connected components. However, this method is not robust to printing and scanning but it is useful for steganography and authentication applications.

Paper [17] shows an overview of recent development in binary document image and data hiding research. It presents interesting study on robustness and data hiding capacity for different techniques, utilized for discussion in our work according to its scope.

In [21], word-shift algorithm for text watermarking was proposed. The advantages of the proposed algorithm overcome the conventional word-shift algorithms from their concepts of the word and segment classification and of using the statistical distributions of inter-word spaces. In the conventional algorithms, individual lines or words hide a portion of the total watermarking information. The algorithm [21] had a global property in the sense that it hides some information into a segment class, where the segments are distributed over the entire document. Due to this segmentation, the algorithm is found tolerable to some kind of inaccuracy, i.e. segmentation errors, and reveals several advantages in terms of imperceptibility, robustness, and payload [21].

A method for watermarking e-text documents was presented in [22]. This method was similar in the word-shift and line-shift coding methods to [21], however, it did not need the original document to find out the watermarking data. It used spread-spectrum error correction techniques to handle the possible noise in the document. Also, this method worked with different associated types in the document by embedding watermarking data through increasing or decreasing the space between two lines or between words. They made this space variation depend on the watermarking bit value, i.e. '1' will add more space to the normal. In a similar fashion, detecting the watermarking bits can be done by measuring the spaces between words or lines and compare them with the normal spreading sequence [22].

In [26], a method for calculating the threshold between letters and word spaces has been introduced based on frequency distributions. Similarly, a spread spectrum technique for watermarking was proposed in [27] based on natural language text. Both methods of [26] and [27] were robust to additive, subtractive and distractive attacks.

Considering watermarking specifically designed for Arabic e-text, only few recent research papers are found in the literature [5, 6, 7, 28, 29, 30, 31, 32, 33, 34]. Method based on point-shifting and character coding are described in [5, 32, 33, 34]; however, these methods have some drawbacks in capacity and robustness and are font dependent. Other methods [28, 30] are based on diacritics related to Arabic e-text. Although diacritics can have high capacity for security purposes, they are considered rare in its usage. This made them easily observed and noticed which makes the security level reduced a lot. Other methods [6, 7, 29] are much related to ours using Kashida. In [6, 29], Kashida is used to hide data but with pointed and un-pointed letters. If a Kashida is to hold secret bits ‘1’ it should be attached to a pointed letter. Similarly, for hiding ‘0’ the Kashida is to be attached to an un-pointed letter. This method solves the problem of the negative robustness form [5, 32, 33, 34] because of its font independent features, and it solved the problem of the diacritics and their rare usage. However, attaching the Kashida with pointed letters limited the amount of data possible to be hidden and reduced the capacity of the system.

This work focused on Kashida usage but released it from pointed/un-pointed letters attachment similar to our introduced idea in [7]. We benefit from the well known fact that Kashida does not have any affect into the Arabic text content. It is meant to be for formatting purposes. In [7] research method, we suggested that whenever a letter cannot have Kashida or found without Kashida it doesn’t represent any secret bit, which is good in capacity but does not have any security key leaving the secrecy to the algorithm.

In this work, we related to our previous work in [7] but improved in the security. We propose using a secret key so that the task of an attack will be much harder similar in principle to classical cryptography. In the next section, we will detail our proposed approach of using Kashida with secret key for e-text Arabic watermarking.

III. Proposed Approach

The idea of the proposed approach is to use a secret key to generate random Kashida characters added to e-text words where secret message is then embedded in the words as watermarking code. The first addition of random Kashidas is for confusion purpose to insure security. Then, selected Kashidas are embedded based on the needed secret data to form the watermarking process to serve its applications.

The process secret key is the seed to generate a sequence of random values as follows: $C=\{c_0, c_1, c_2 \ldots c_n\}$, where $c_i \in [0,7]$. The values of C represents to the system the number of extendable characters in given words; i.e. those characters that can be extended without changing the style of the word. The secret key is the seed to generate C values considered to be given to the system as a randomly generated number or selected by the user. The key is to trigger any pseudorandom number generator, such as a linear feedback shift register to provide all C values. The secret key issue and C values generation is not in the scope of this work, where the assumption is made for C values to be as given information to start the e-text watermarking process. It may be worth mentioning the observation that the value of C range from *0* to *7* because of the possible number of Kashida characters that can be added within any Arabic e-text word. This range is based on our investigation in this filed that indicated the impossibility for Arabic words to handle the addition of more than *7* Kashidas.

As observed in the values of C above, they are given as $n+1$ random numbers, i.e. from c_0 to c_n. The value of n is the number of the words in the cover text. The overall objective of the process is to embed the watermarking data, W, in cover text, I, as observed in the block diagram in Figure 2.

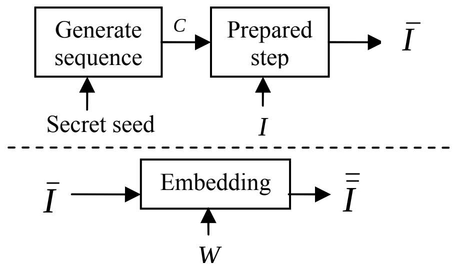


Figure 2: embedding process

To simplify the process explanation, after generating the sequence values, C, the cover text, I, will be updated based on that C sequence producing another updated cover text, say $\bar{I}$. This $\bar{I}$ is found to have Kashidas inserted in some words in the original cover text for security confusion purposes (as preparation phase). Algorithm 1 is used as this preparation phase to update the original cover text to generate $\bar{I}$. Then, second phase will update cover text $\bar{I}$ to embed the secret watermarking data, i.e. W, for generating the ending output file, $\bar{\bar{I}}$, as briefed in Algorithm 2. The watermarking embedding process is to insert some Kashida characters in words with a method to represent block of watermarked data limited by k as the block size representing the number of possible extendable letters in a word. However, the maximum number of Kashidas allowed to be inserted per word, represented by x, is different but should be less than k. In other words, the number of x Kashida characters to be inserted in a word should be less than the possible number k that can be inserted, which is given to increase the security of the system.

Note that the positions to insert Kashida characters in the words are based on a coding representation as in Figure 3. This figure, Figure 3, maps the value to the corresponding positions to insert Kashidas. In fact, Figure 3 is just an example to clarify such representation using maximum of two Kashidas in order to represent the same values. Consider to embed a block of three bits '101', which has the decimal value of 5, in a word having 3 extendable characters. The coding representation figures that as for value 5, Kashidas insertion should be at positions 1 and 3, as shown clearly in Figure 3.

Below are the outline of Algorithm 1 and Algorithm 2 which are making the main procedure for embedding watermarking data in cover text.

Algorithm 1:
Preparation Phase - Update the Cover Text

Input: secret seed, cover text, *I*.
Output: updated cover text, $\bar{I}$.

```
while data left to update do
    get next word from cover text, I
      $c_i$ ← next random(seed).
      n ←the number of possible extendable letters
        if $c_i$ <=n then
            insert Kashida into the word at $c_i$.
        end if
end while
```

Value	Positions	3	2	1
0	none			
1	1			x
2	2		x	
3	3	x		
4	1,2		x	x
5	1,3	x		x
6	2,3	x	x	

Figure 3: Example of coding representation

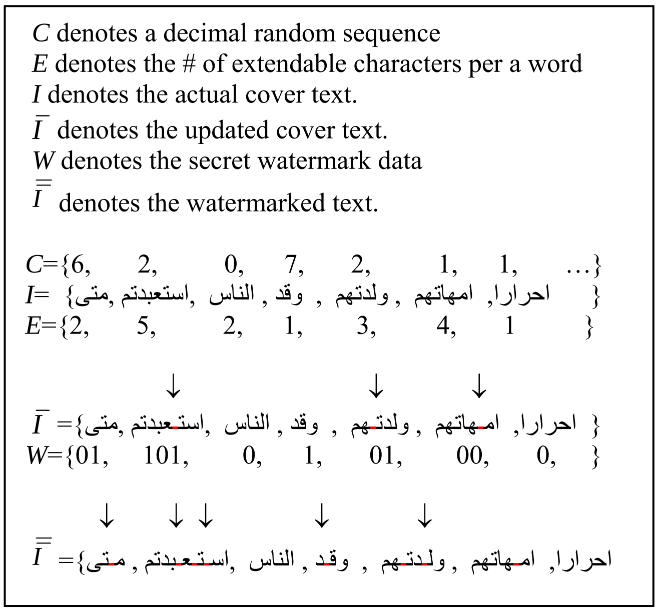
Algorithm 2:
Embedding Phase - Embedding Watermarking Bits

```
Input: updated cover text, watermark bits
Output: watermarked text

initialize x with maximum Kashidas used per word.

while data left to embed do
    get next word from cover text
        n ← the number of possible extendable letters.
        if n > 0 then
            s ← calculate the block size based on n and x.
            get the value of the next s-bit message block
            determine the positions that represent the value.
          insert Kashida into the word at positions.
          insert the word into watermarked text.
        end if
end while
```

A simple real example of embedding watermark data, *W*, in cover text, *I*, using the two algorithm phases is shown in Figure 4. Assume, the first random value in the sequence *C*, is $c_0=6$, which is generated randomly using the secret key (seed). However, the first word in the cover text, I_0, has only 2 extendable characters, $E=2$, thus no Kashida can be inserted in such word; i.e. the sequence value for such word (c_0) should be more than the number of extendable characters the word can hold. The second random value in sequence *C*, is $c_1=2$. The second word in the cover text, I_1, has $E=5$ extendable characters. Thus, since $E > C$, a camouflage (fooling) Kashida is to be inserted in such word at position 2 (i.e. based on c_1). This fooling Kashida is not to be used for any watermarking data hiding. Instead, it reduces the number of Kashidas possibly inserted in that word, which is considered capacity reduction assumed acceptable to increase security of the system. For example, to embed the secret watermarking block W_1 in the corresponding updated covertext block $\overline{I}_1$, we do not count the extendable character that was generated by the preparation phase. It will be understood as omitted. Thus, the embedding process deals with such word as having only 4 extendable characters instead of 5. This, if mapped as Figure 3, would represent '101' in the word with 4 extendable characters by inserting Kashidas at positions 1 and 2.In order to detect the watermarking information a blind technique is used as in Algorithm 3. Algorithm 3 extracts the watermarking data, *W*, using the same secret key (seed) to produce the same sequence value, *C*, used in the embedding method. Then, for each word in the watermarked file, $\overline{\overline{I}}$, it calculates the number of extendable characters in that word, k_i, and compares it to the corresponding sequence value C_i. In case of k_i having more value than C_i, a fooling (camouflage) Kashida is assumed at position C_i. The watermarking data bits, W_i, is to be obtained using positions of the remaining Kashidas. All W_i values can be obtained by searching the coding mapping representation of Figure 3 for values that matches the extracted positions.


C denotes a decimal random sequence
E denotes the # of extendable characters per a word
I denotes the actual cover text.
$\overline{I}$ denotes the updated cover text.
W denotes the secret watermark data
$\overline{\overline{I}}$ denotes the watermarked text.

C={6, 2, 0, 7, 2, 1, 1, …}
I= {متى, استعبدتم, الناس , وقد , ولدتهم , امهاتهم ,احرارا }
E={2, 5, 2, 1, 3, 4, 1 }

↓ ↓ ↓

$\overline{I}$ ={متى, استـعبدتم, الناس , وقد , ولدتـهم , امـهاتهم ,احرارا }
W={01, 101, 0, 1, 01, 00, 0, }

↓ ↓↓ ↓ ↓

$\overline{\overline{I}}$ ={مـتى , استـعـبدتم, الناس , وقـد , ولـدتـهم , امـهاتهم ,احرارا

Figure 4: Example of embedding process

Algorithm 3:
Extracting Watermarking Bits

```
Input: secret seed, watermarked text, I--.
Output: watermark
initialize k with maximum Kashidas used per word.

while word left to process do
    get next word from watermarked text
    $c_i$ ← next random(seed).
    n ← the number of possible extendable letters, omitting $c_i$.
    if n > 0 then
            s ← calculate the block size based on n and k.
            determine the positions of existed Kashidas.
            get the value represented by positions
            get the s-bits block representing the value.
            insert the s-bits block into watermark.
    end if
end while
```

IV. Experimental Results & Comparisons

Our proposed approach is implemented to embed secret watermarking data within e-text documents for web applications. To achieve this goal and test it, we developed a program using C# programming language. We have used the class 'RNGCryptoserviceProvider' in the .NET Framework to generate our cryptographic key to be used as our secret seed. We evaluated our proposed method using number of e-text cover files with different watermarking capacity sizes, i.e. 68.4kb, 128kb, 256kb and 512kb. Aside of capacity, we also considered the other two important watermarking issues of security and robustness in a general analytical manner.

It should be noted that when comparing the results of this work with others, unfairness situation is observed. This work is found different than all previous ones [5, 6, 7, 28, 29, 30, 31, 32, 33, 34]. However, other approaches results are reported to get an idea or a comparison thought between this work and others, i.e. some previous work averages are used without considering the details of the e-text watermarking or steganography approach.

The capacity can be evaluated by the watermarking capacity ratio vs. the e-text characters ratio, which can be computed by dividing the amount of hidden data bytes over the size of cover e-text media in bytes. Also, to compute the usable characters ratio, we count the number of usable characters per approach, as indicated before, independent from the scenario or the secret message to be embedded. We use *p* for the ratio of characters capable of baring a secret bit of a given level, and *q* for the ratio of characters capable of baring the opposite level. In the case of our approach, an extendable character may hide a secret bit of '0' or '1'. Thus, such characters may contribute to *p* and *q*, i.e. *p* equals *q*. Table 1 summarizes the results of the capacity ratio related to our proposed approach.

TABLE 1
Average capacity ratio for implementations of our proposed watermarking technique

Cover size (bytes)	**70,627**	**131,233**	**235,125**	**525,271**
Capacity Ratio of complete watermarking method	2.8	2.8	2.8	2.8
Capacity Ratio without Preparation phase	3.285	3.29	3.26	3.37
Usable Characters ratio	0.371	0.369	0.366	3.658

For the comparison purpose, a previous *diacritic* method is considered, which is also invented for similar Arabic e-text applications [28, 30]. The diacritic encoding/decoding program was developed under C++ that can accepts a HTML Arabic diacritized page for cover media, a file in any format to hide (.txt, .wav, .jpg … etc.), and produces a HTML page that has the previous cover media but containing the hidden file embedded within. The output file can be processed for the decoding program to retrieve the hidden file and save it in its previous and proper format. Table 2 lists the capacity results of this diacritic method.

TABLE 2
Diacritics Technique [30]

File type	**File size (Bytes)**	**Cover Size (Bytes)**	**Capacity (%)**
.txt	10,356	318,632	3.25
.wav	43,468	1,334,865	3.26
.jpg	23,796	717,135	3.32
.cpp	10,356	318,216	3.25
		Average	3.27

Furthermore, the work in [6] was also reported for comparison reasons. The technique is similar to ours in using Kashida but with pointed characters as follows; for every pointed character we insert a Kashida if we need to embed a '1', or don't insert a Kashida if we need to embed a '0'. The program was tested under various formats and results are reported in Table 3, as detailed in [30]. It produced a low average capacity of 1.22% compared to our new proposed approach in this work. This concludes that our approach here is higher in capacity than the previous one of [6], even though technically the work in [6] can be modified to increase capacity but with much greater loss of ambiguity as described in [30].

TABLE 3
Previous Kashida Technique [30]

File type	**File size (Bytes)**	**Cover Size (Bytes)**	**Capacity (%)**
.txt	4439	365,181	1.22
.html	4439	378,589	1.17
.cpp	10,127	799,577	1.27
.gif	188	15,112	1.24
		Average	1.22

Table 4 summarizes the capacity averages of the different methods, i.e. dots [5], Kashida [6], Diacritics [28, 30], and the new proposed method of this paper. The proposed work capacity average is found to be 2.8%, which outperforms the reported capacity ratio of the Dots method [5] and the previous Kashida method [6]. Note that on the other hand, our proposed method showed less capacity ratio than the reported average capacity of the Arabic e-text Diacritics stego technique [28]. Table 4 also considers comparing the average capacity of our proposed technique with and without the preparation phase (with and without Algorithm 1). This comparison shows a decrease in the capacity ratio of about 0.6% due to using some extendable characters in the cover text as camouflage extensions. This ratio is considered acceptable reduction in the capacity for increasing the security.

The proposed work also considered general comparison (capacity, robustness, and security) with different scenarios previously presented for the diacritics approach in [28], as shown in Table 5. The table, Table 5, listed the diacritics scenarios of text+softcopy, image+softcopy, and image+hardcopy. The image approach had two entries: one assuming a softcopy of the document image is distributed and the other one assuming a printed version is. The text approach is not, generally, robust to printing. However, it is capable of achieving arbitrarily high capacities. The file size might deteriorate the security level, however, if this approach is abused. The image approach is, to some extent, robust to printing. The softcopy version is only mentioned for completeness. It has a very low capacity. Its security is also vulnerable since text isn't usually sent in images. The hardcopy version of the image approach intents to achieve robustness with good security. The reader is referred to [28] for more elaboration and details.

TABLE 4
Reported Average Capacity ratio in the literature

Approach	Average Capacity
New Kashida proposed in this work - complete watermarking method	2.8
New Kashida proposed in this work without Preparation phase	3.3
Dots [5]	1.37
Kashida [6]	1.22
Diacritics [28, 30]	3.27

TABLE 5
Comparison between our new watermarking method and previous approaches in terms of capacity, robustness and security

Approach		Capacity	Robustness	Security
New proposed in this work		Moderate, better than Kashida with Doted letters but not as Diacritic High Capacity method	Robust to printing.	Visible used for Watermarking Applications
Diacritics [28]	Text + softcopy	High, up to infinity in 1st scenario.	Not robust to printing.	Invisible, in code.
	Image + softcopy	Very low, due to image overhead	Robust to printing.	Slightly visible.
	Image + hardcopy	Moderate, 1st scenario, block of 2	Robust to printing.	Slightly visible.

V. Disscussion & Security Analysis

The watermarking algorithm cannot be kept always secret since it will have to be revealed in court the first time it is used. Thus, security of our watermarking procedure cannot rely on the secrecy of the method; it should depend on the secrecy of the key used. Unauthorized users should be unable to extract the secret data without the key, i.e. we assume that the e-text does not reveal the hidden watermarking easily if the key is unknown. Any attacker needs reasonable amount of time to break the system, even if he/she knows that the host media contains watermarking Kashidas and is familiar with the exact e-text watermarking embedding method.

Although the proposed watermarking method is known, the security is increased because of inserting camouflage (fooling) Kashidas controlled by the key. These fooling Kashidas are embedded in the cover text before embedding the watermarking bits as in Algorithm-1 (preparation phase). Without running such preparation phase, the removal attack can easily remove all the Kashidas from the watermarked file. Then, the attack can insert Kashidas to represent his/her cheat watermarking using the watermarking algorithm which argued to be completely not secure.

The main application of our proposed approach can be to prove the ownership. If a removal attack tries to remove the existing Kashidas from the watermarked file, it is hard to recreate the cover text carrying the original camouflage Kashidas, as it is based on the sequence random values generated from the secret seed. Hence, such attack can not replace the existing ownership watermark by his/her cheating ownership.

Another security application related to our watermarking method is the ability to detect modifications in the watermarked file. This is used for authentication, where the objective is to detect modification of the data, to be achieved with so-called fragile watermarks. Fragile watermarks are watermarks that are used in authentication applications in order to detect modifications of the data rather than proofing the ownership changing. Our proposed approach can be used to satisfy this purpose, i.e. as fragile watermarking too. If an attack modifies the watermarking contents, the extracting algorithm that is based on secret sequence values will detect such modifications. It will be comparing the positions of the concealed Kashidas to the sequence values and discover the attack.

Some properties of the proposed approach and other remarks can be summarized as following:

- The proposed method is semi-robust. It can withstand printing and font changing, but not all OCR techniques nor retyping.
- It is fast and does not require large computational power.
- Simple and can be implemented manually if needed.
- Might raise suspicions since it is uncommon to send documents with too many Kashidas.

VI. Conclusion

The paper proposed a novel method of watermarking for e-text web applications related to Arabic language. This method can also fit for other similar Semitic languages, i.e. Urdu and Farsi. The main idea behind the proposed method is to utilize all the extendable characters in a word to represent some watermarking bits or fooling bits. The fooling bits are embedded in the cover text based on a secret key before the watermarking bits are embedded. These fooling Kashidas are added to make the task of an attack very hard providing security to the system.

The experimental results showed adequate capacity ratio compared to the previously proposed approaches for Arabic e-text watermarking in the literature. From the security prospective, our proposed method is mainly concerned about some watermarking removal attacks. Thus it can be used to serve different copyright applications in e-text watermarking, i.e. proofing the ownership of the e-text document.

Acknowledgments

Authors would like to thank King Fahd University of Petroleum & Minerals (KFUPM), Dhahran, Saudi Arabia, for supporting this research work. Special appreciation to all the 2006 and 2007 wonderful groups of graduate students of the course COE 509: Applied Cryptosystems - Techniques & Architectures for their valuable initiatives and positive cooperation.

References

[1] Yusnita Yusof and Othman Khalifa, “Digital Watermarking For Digital Images Using Wavelet Transform,” *IEEE International Conference on Telecommunications and Malaysia International Conference on Communications*, pp. 665-669, Penang, Malaysia 14-17 May 2007.

[2] Richard Popa, “An Analysis of Steganogrphic Techniques,” *Master Thesis submitted to Department of Computer Science and Software Engineering*, Faculty of Automatics and Computers, The Politehnica University of Timisoara, May 1998.

[3] Mohammad Tanvir Parvez and Adnan Gutub, “RGB Intensity Based Variable-Bits Image Steganography”, *APSCC 2008 – 3rd IEEE Asia-Pacific Services Computing Conference*, Yilan, Taiwan, 9-12 December 2008.

[4] M.A. Qadir, and I. Ahmad, “Digital Text Watermarking: Secure Content Delivery and Data Hiding in Digital Documents,” *IEEE 39th Annual International Carnahan Conference on Security Technology (CCST),* pp. 101-104, 11-14 Oct. 2005.

[5] M. Hassan Shirali-Shahreza, Mohammad Shirali-Shahreza, “A New Approach to Persian/Arabic Text Steganography,” *5th IEEE/ACIS International Conference on Computer and Information Science, (ICIS-COMSAR)*, pp. 310-315, July 2006.

[6] Adnan Gutub, Lahouari Ghouti, Alaaeldin Amin, Talal Alkharobi, and Mohammad K. Ibrahim, “Utilizing Extension Character ‘Kashida’ With Pointed Letters For Arabic Text Digital Watermarking,” *International Conference on Security and Cryptography – (SECRYPT)*, Barcelona, Spain, 28 – 31 July 2007.

[7] Fahd Al-Haidari, Adnan Gutub, Khalid Al-Kahsah, and Jameel Hamodi, "Improving Security and Capacity for Arabic Text Steganography Using 'Kashida' Extensions," *AICCSA-2009 - The 7th ACS/IEEE International Conference on Computer Systems and Applications*, Rabat, Morocco, 10-13 May 2009.

[8] S.H. Low, N.F. Maxemchuk, J.T. Brassil, Oapos, and L. Gorman, “Document Marking and Identification Using Both Line and Word Shifting,” *INFOCOM'95 -Fourteenth Annual Joint Conference of the IEEE Computer and Communications Societies on Bringing Information to People,* Vol. 2, pp. 853-860, Boston, MA, 2-6 April 1995.

[9] J.T. Brassil, S. Low, N.F. Maxemchuk, Oapos, and L. Gorman, “Electronic Marking and Identification Techniques to Discourage Document Copying,” *IEEE Journal on Selected Areas in Communications*, Vol. 13, No. 8, pp. 1495-1504, October 1995.

[10] Walter Bender, Daniel Gruhl, Norishige Morimoto, and Anthony Lu, “Techniques for Data Hiding,” *IBM Systems Journal*, Vol 35, pp. 313-336, Sept-Dec 1996.

[11] S.H. Low, N.F. Maxemchuk, A.M. Lapone, “Document Identification for Copyright Protection Using Centroid Detection,” *IEEE Transactions on Communications*, Vol. 46, No. 3, pp. 372-383, March 1998.

[12] J.T. Brassil, S. Low, and N.F. Maxemchuk, “Copyright Protection for the Electronic Distribution of Text Documents,” *Proceedings of the IEEE*, Vol. 87, No. 7, pp.1181-1196, July 1999.

[13] A.K. Bhattacharjya, and H. Ancin, “Data embedding in text for a copier system,” *Proceedings of the IEEE International Conference on Image Processing - ICIP 99*, Vol. 2, pp. 245-249, 1999.

[14] F. Hartung and M. Kutter, “Multimedia watermarking techniques,” *Proceedings of the IEEE*, Vol.87, No.7, pp.1079-1107, July 1999.

[15] Q.G. Mei, E.K. Wong and N.D. Memon, “Data Hiding in Binary Text Documents,” *Proceedings of the SPIE, Security and Watermarking of Multimedia Contents III*, Vol. 4314, pp. 369-375, San Jose, California, January 2001.

[16] Benjamín Baran, Santiago Gomez and Víctor Bogarin, “Steganographic Watermarking for Documents,” *34th Annual Hawaii International Conference on System Sciences (HICSS)*, Vol. 9, Outrigger Wailea Resort, Island of Maui, Hawaii, 3-6 January 2001.

[17] M. Chen, E.K. Wong, N. Memon, and S. Adams, “Recent Developments in Document Image Watermarking and Data hiding,” *Proceedings of SPIE Vol 4518: Multimedia Systems and Applications IV*, pp. 166-176, August 2001.

[18] D. Huang and H. Yan, “Interword distance changes represented by sine waves for watermarking text images,” *IEEE Transactions on Circuits and Systems for Video Technolo*, Vol. 11, No. 12, pp. 1237-1245, December 2001.

[19] M.S. Khankhalli, K.F. Hau, “Watermarking of Electronic Text Documents,” *Electronic Commerce Research,* Vol. 2, No. 1-2, pp. 169-187, January 2002.

[20] I.J. Cox , M.L. Miller, “The first 50 years of electronic watermarking,” *EURASIP Journal on Applied Signal Processing*, Vol. 2002, No. 2, pp.126-132, February 2002.

[21] Y. Kim, K. Moon, and I. Oh, “A Text Watermarking Algorithm based on Word Classification and Inter-word Space Statistics,” *Proceedings of the Seventh International Conference on Document Analysis and Recognition - ICDAR*, pp. 775-779, 3-6 August 2003.

[22] Adnan Alattar and Osama Alattar, “Watermarking electronic text documents containing justified paragraphs and irregular line spacing,” *Proceedings of SPIE Security, Steganography, and Watermarking of Multimedia Contents VI*, Vol. 5306, pp. 685-695, June 2004.

[23] M. Topkara, U. Topkara, M.J. Atallah, “Words are not enough: Sentence level natural language watermarking,” *Proceedings of 4th ACM Workshop on Content Protection and Security - MCPS’06,* Santa Barbara, CA, 27 October 2006.

[24] H.M. Meral, E. Sevinc, E. Unkar, B. Sankur, A.S. Ozsoy, T. Gungor, “Syntactic Tools for Text Watermarking,” *Proceedings of the SPIE International Conference on Security, Steganography, and Watermarking of Multimedia Contents,* San Jose, CA, 29 January – 1 February 2007.

[25] O. Vybornova and B. Macq, “Natural Language Watermarking and Robust Hashing Based on Presuppositional Analysis,” *IEEE International Conference on Information Reuse and Integration (IRI)*, pp. 177-182, 13-15 August 2007.

[26] C. Culnane, H. Treharne, A.T.S. Ho, “Improving Mutli-set Formatted Binary Text Watermarking Using Continuous Line Embedding,” *Second International Conference on Innovative Computing, Information and Control - ICICIC*, pp. 287-292, 5-7 September 2007.

[27] Jianlong Yang, Jianmin Wang, Chaokun Wang, and Deyi Li, “A Novel Scheme for Watermarking Natural Language Text,” *3rd International Conference on Intelligent Information Hiding and Multimedia Signal Processing - IIHMSP*, Vol. 2, pp. 481-484, 26-28 November 2007.

[28] Adnan Gutub, Yousef Elarian, Sameh Awaideh, and Aleem Alvi, "Arabic Text Steganography Using Multiple Diacritics," *5th IEEE International Workshop on Signal Processing & its Applications - WoSPA*, University of Sharjah, U.A.E. 18-20 March 2008.

[29] Adnan Gutub, and Manal Fattani, "A Novel Arabic Text Steganography Method Using Letter Points and Extensions", *Proceedings of World Academy of Science, Engineering and Technology*, Vol. 21, May 2007.

[30] Mohammed Aabed, Sameh Awaideh, Abdul-Rahman Elshafei, and Adnan Gutub, “Arabic Diacritics Based Steganography,” *IEEE International Conference on Signal Processing and Communications - ICSPC*, pp. 756-759, Dubai, UAE, 24-27 November 2007.

[31] Jibran Memon, Kamran Khowaja, and Hameedullah Kazi, "Evaluation of Steganography for Urdu/Arabic Text," *Journal of Theoretical and Applied Information Technology - JATIT*, Vol. 4, No. 3, pp. 232-237, 31 March 2008.

[32] M. Shirali-Shahreza, "Pseudo-space Persian/Arabic text steganography," *IEEE Symposium on Computers and Communications - ISCC*, pp. 864-868, 6-9 July 2008.

[33] M. Shirali-Shahreza, "A New Persian/Arabic Text Steganography Using “La” Word," *Proceedings of the International Joint Conference on Computer, Information, and Systems Sciences, and Engineering - CISSE*, Vol. 2, pp. 339-342, Bridgeport, CT, USA, 3-12 December 2007.

[34] M. Hassan Shirali-Shahreza, Mohammad Shirali-Shahreza, "Steganography in Persian and Arabic Unicode Texts Using Pseudo-Space and Pseudo-Connection Characters," *Journal of Theoretical and Applied Information Technology - JATIT*, Vol. 4, No. 8, pp. 682-687, August 2008.

Adnan Abdul-Aziz Gutub is an associate professor in Computer Engineering at King Fahd University of Petroleum and Minerals (KFUPM) in Dhahran, Saudi Arabia. He received his Ph.D. degree (2002) in Electrical & Computer Engineering from Oregon State University, USA. He has his BS in Electrical Engineering and MS in Computer Engineering both from KFUPM, Saudi Arabia. **Adnan's** research interests are in modeling, simulating, and synthesizing VLSI hardware for crypto and security computer arithmetic operations. He worked on designing efficient integrated circuits for the Montgomery inverse computation in different finite fields. He has some work in modeling architectures for RSA and elliptic curve crypto operations. His interest in computer security also involved steganography such as simple image based steganography and Arabic text steganography.

Adnan has been awarded the UK visiting internship for 2 months of summer 2005 and summer 2008, both sponsored by the British Council in Saudi Arabia. The 2005 summer research visit was at Brunel University to collaborate with the Bio-Inspired Intelligent System (BIIS) research group in a project to speed-up a scalable modular inversion hardware architecture. The 2008 visit was at University of Southampton with the Pervasive Systems Centre (PSC) for research related to advanced techniques for Arabic text steganography and data security.

Adnan Gutub filled many administrative academic positions in KFUPM; currently, he is chairing the computer engineering department (COE) at KFUPM in Dhahran, Saudi Arabia.

Fahd. Al Haidari received the B.S degree in Computer Science from University of Mousl, Iraq, in July 1999. He received the M.S in Computer Science from KFUPM, Saudi Arabia, in June 2007. Currently, He is a PhD student in Computer Science and Engineering at KFUPM, KSA.

Khalid Mohammed Al-Kahsah is a graduate student at the department of information and computer science at KFUPM. His research interests are software engineering, Arabic characters recognition, and cryptography. He received his B.Sc. in computer sciences form Basrah University, Iraq in 2001. He is working as a graduate assistant at the department of Computer Sciences and Mathematics, college of sciences, Ibb University, Yemen. He is on full scholarship to get master degree in computer sciences from KFUPM since 2005.

Jamil Hamodi received the B.S degree in Electrical & Computer Engineering from Sana’a University, Yemen, in 2000. Currently, He is a full time MS (Master of Science) student in Computer Networks at the department of Computer Engineering, KFUPM – Saudi Arabia.