Utilizing Word Space with Pointed and Un-pointed Letters for Arabic Text Watermarking

Reem A. Alotaibi
Computer Science Department
Faculty of Computing and Information Technology
King Abdulaziz University
Jeddah, Saudi Arabia
reem.a.safran@gmail.com

Lamiaa A. Elrefaei[1,2]
[1]*Computer Science Department*
Faculty of Computing and Information Technology
King Abdulaziz University, Jeddah, Saudi Arabia
[2]*Electrical Engineering Department*
Faculty of Engineering-Shoubra
Benha University, Cairo, Egypt
laelrefaei@kau.edu.sa, lamia.alrefaei@feng.bu.edu.eg

***Abstract*—The issue of protecting scientific and intellectual products and publications ownership constitutes an important deal within the domain of securing and protecting information. Digital watermarking is used to identify the ownership of the copyright and in authentication process. In this paper, an invisible blind watermarking method utilizing the word space in Arabic text is proposed. Since the pseudo-space is very small space used to separate two parts of the same word, it is inserted before normal space and after pointed letter to hold the binary bit 'one', and it is inserted before normal space and after un-pointed letter to hold binary bit 'zero'. The comparative results obtained by testing our proposed method with some of existing Arabic watermarking methods using four variable size text samples. The results show that our method has a higher capacity ratio and imperceptibility than other watermarking techniques.**

Keywords-Kashida, pseudo-space, capacity, imperceptibility

I. INTRODUCTION

Many issues have been raised in the past few years about how to protect the information from unauthorized usage through the internet. So, the need of an effective solution to maintain text privacy and security is a high demand. Digital watermarking is used to solve these problems. It embeds a signal called watermark into digital data (audio, images, video, and text) without destroying the data value to get the watermarked data. The watermark later detected or extracted to be used in many applications such as: copyright protection, data authentication, data hiding and covert communication [1].

The watermarking system involves two main processes: embedding of the watermark into the original data and extracting the watermark from watermarked data or attacked watermarked data. It can be described by a number of requirements [2,3]: robustness, capacity, imperceptibility and security. These requirements used in watermark embedding and extraction processes to be done in an effective manner. They vary depending on the watermarking applications and the hidden data types. The watermarking system robustness is measured by the ability to extract or recover the watermark after the watermarked data has been attacked. Capacity is defined as the maximum amount of embedded bits to be hidden in the cover data. Imperceptibility or invisibility is one of the most essential watermark requirements. The watermark embedding process should be done in a way that the watermark cannot be noticed by the human eye. The original data and the watermarked one have no difference between them in an imperceptible watermarking system. Security includes making the watermark information and its capacity well secured from unauthorized users. The attacker cannot detect the watermark without knowing the algorithms to embed or extract the watermark and the watermark itself. In some watermarking applications cryptographic keys are used in the watermark insertion and extraction processes to ensure security.

Arabic language contains many features that are exploited in the field of data hiding within its two branches: steganography and watermarking. Some of the Arabic scripts characteristics are: the existence of points above or under the letter, diacritics which are equivalent with the vowels in the English language and kashida which is the stretching character [4]. This paper is concerned in Arabic text watermarking since a little of researches are done in this area. A rich of Arabic text resources need to protect its copyrights and detect any modifications especially in sensitive text such as the Holly Quran. Arabic text watermarking method depending on the existing of spaces separate words and points in most of its letters is proposed. The embedding and extraction algorithms include insertion of pseudo-space before normal space according to the letter status. Pseudo-space is a non printing character used to separate two parts of the same word in Persian language. Utilizing each word space in the text and using of non printing character guarantee high insertion capacity and imperceptibility. Experiments are conducted to evaluate the performance of our proposed method and compare it with five existing Arabic text watermarking methods.

The rest of the paper is structured as follows: section II discusses the works related to Arabic text watermarking. The proposed method is explained in details in section III. The results of the proposed method are presented and compared with five of Arabic text watermarking methods in section IV and finally section V concludes the paper.

978-1-5090-0888-9/16 $31.00 © 2016 IEEE
DOI 10.1109/UKSim.2016.34

II. Related Work

Hiding the data in Arabic text is being interested recently. In 2006, the authors in [5] had created a new method to hide the data in Arabic and Persian texts. This method relies on shifting the points in the letters which contain points .It is used to hide secret bits of zeros and ones. It includes vertical movement of the point if the secret bit equals 1, else if the bit equals 0 nothing change. It provides high capacity where more than half of the Arabic language letters are pointed. Also, it is useful in printing documents. However, the secret information will destroy in retyping and this method requires special font. Persians in their writing use pseudo-space to make some words separated to contain two parts. The researchers in [6] suggest adding an ordinary space after pseudo-space to hide one and put nothing to denote zero.

Other methods proposed in the literature [7-9] for information hiding in Arabic text by utilizing the diacritical marks. Diacritics or (harakat) are intended to adjust the character appearance with marks showing the correct way to pronounce it. The need for diacritics is to reduce errors in pronunciation because there are similar words in writing, but differs in pronunciation and meaning. The original Arab reader can read the Arabic text fluently without diacritical marks which means that the using of these marks is optional. The authors in [7] use (fatha) to hide bit one and the remaining of the eight (harakat) to hide bit zero. The researchers in [8] try to increase the capacity as twice of capacity in [7] by using the omitted diacritics to hide data. If the diacritic is present, the secret bit equals 1, if the diacritic is absent or removed, the secret bit equals 0. These methods provide high capacity, but they may draw attentions where some diacritics are present and others not. Also, they require original text to extract the secret bits. The authors in [9] reverse the display of the diacritic (fatha) to be from left to right creating a new font family. They use the original (fatha) to hide bit 1 and the reverse one to hide bit 0. This method is robust against printing, but not in case of retyping.

Some suggested methods adopted in the field of Arabic text watermarking based on kashida. Kashida does not change the meaning, but it is attached to the letter to stretch it. Not all letters accept inserting kashida before or after them. Kashida based watermarking is firstly used in 2007, the authors in this work [10] proposed a novel method for steganography in Arabic text that use pointed letter with kashida to hide bit one, and un-pointed letter with kashida to hide bit zero. The research work in [11] aims to reduce the degree of extraction of the secret bits, where the authors identify the number and the placements of kashidas which are added within each word. To make the extraction process more difficult, the researchers in [12] add the kashida before specific letters where the kashida is unacceptable to attach after them. The last two methods used kashida in watermarking Arabic text, in 2014 the authors in [13] insert the kashidas before two different sets (set A, set B) based on the frequency of the Arabic letters. Set A includes the letters with higher frequency and set B includes the letters with lower frequency. The main advantage of Kashida based methods is to protect the hidden bits in printing, but they could be lost in retyping or using OCR.

III. Proposed Method

A new watermarking technique to hide a secret information in Arabic text is proposed. This method utilizes the dotting feature in Arabic script as most of Arabic letters have one point or more above or under the letter. The pointed letter refers to the letter which has points and un-pointed letter refers to the letter which has no points. Fig. 1 shows the classification of Arabic letters into pointed and un-pointed.

pointed letters	un-pointed letters
ش ز ذ خ ج ث ت ب	ص س ر د ح ا
ي ن ق ف غ ظ ض ة	و ه م ل ك ع ط

Figure 1. Arabic letters.

Pseudo-space is used in this method which is a small space used to separate the same word into two parts [6]. It is also called zero width non joiner (ZWNJ) which is a character does not appear in printing, but if it is placed between two letters they will be separated like this word: می‌خواهم. It is used in languages that does not take one shape of the letter such Persian language. The embedding and extraction algorithms of this method are discussed in the following subsections.

A. Embedding Algorithm

In the embedding algorithm, the watermark bits are embedded by checking each space between words and the letter before it. If the letter is pointed and the watermark bit is one, the pseudo-space is inserted. Else if the watermark bit is zero, no pseudo-space is inserted. If the letter is un-pointed and the watermark bit is zero, the pseudo-space is inserted. Else if the watermark bit is one, no pseudo-space is inserted. Fig. 2 shows an example where the colored letters indicate the existing of pseudo-space after it. Fig. 3 illustrates the flowchart of watermark embedding algorithm where OT, P, UP and W are referring to the original text, pointed letter set, un-pointed letter set and watermark respectively.

Watermarking bits	1001011
Original text	البر حسن الخلق والإثم ما حاك في نفسك
Output text	البر حسن الخلق والإثم ما حاك في نفسك 1 1 0 1 0 0 1

Figure 2. Example of the proposed method.

B. Extraction Algorithm

The extraction algorithm takes the watermarked text as an input and return the watermark is an output. The proposed

method is a blind watermarking method because the original text is not needed to extract or detect the watermark. To extract the watermark bits, the watermarked text is searched for each normal space which has a Unicode "U+0020" [14] to check if there is a pseudo-space (which has a Unicode "U+200C" [6]) before it or not. If the pseudo-space is founded after pointed letter, the watermark bit is set to one. But if it is founded after un-pointed letter, the watermark bit is set to zero. If there is no pseudo-space founded and the letter before the space is pointed, the watermark bit is set to zero. Else if the letter is un-pointed, the watermark bit is set to one. Fig. 4 shows the flowchart of the watermark extraction algorithm where WT, P and UP are referring to watermarked text, pointed letter set and un-pointed letter set respectively.

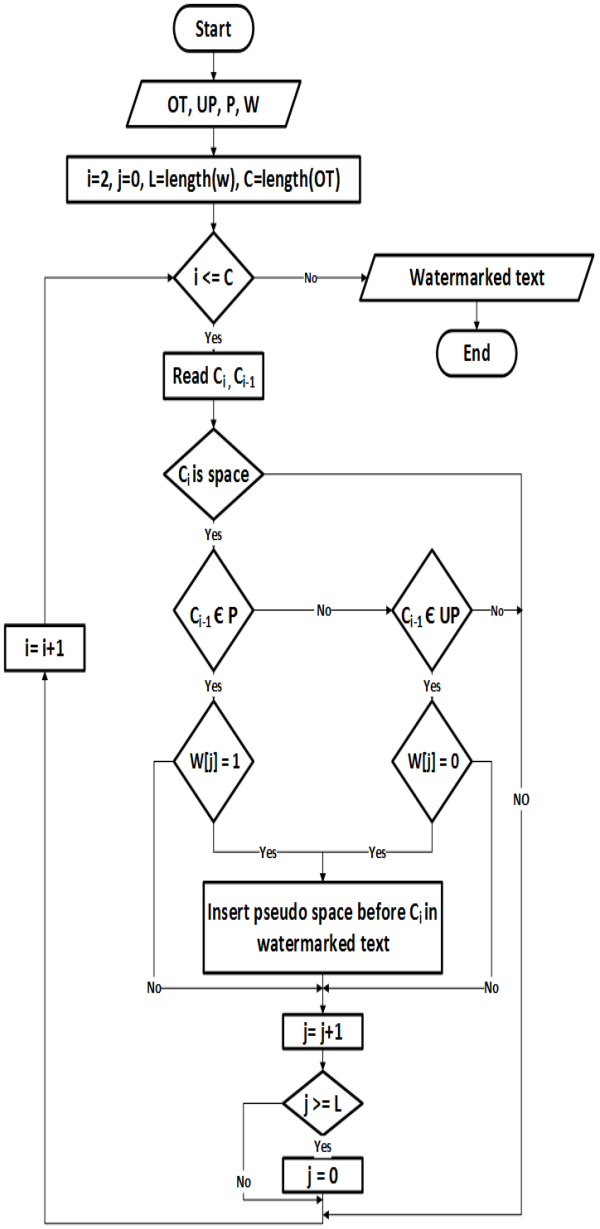

Figure 3. Flowchart of embedding algorithm in the proposed method.

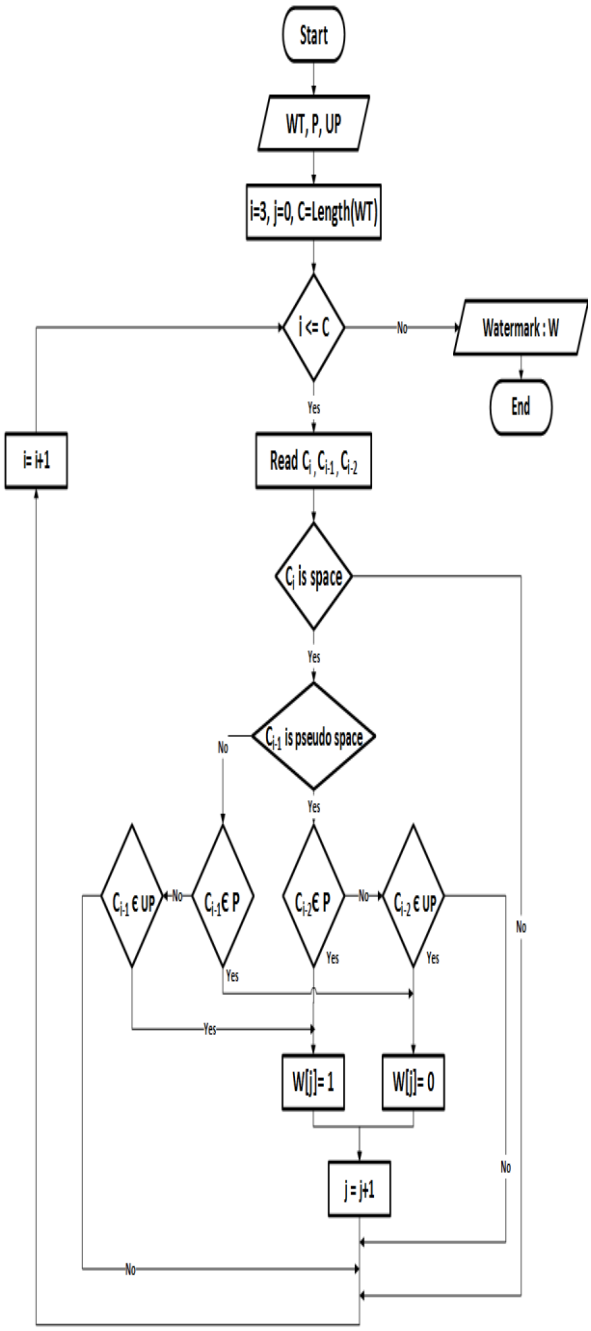

Figure 4. Flowchart of extraction algorithm in the proposed method.

IV. RESULTS AND DISCUSSION

The results of the proposed method are compared with five Arabic text watermarking methods: Diacritic based [7], Letter points and Kashida method [10], Enhanced Kashida Method [12], Enhanced Kashida Method A [13] and Enhanced Kashida Method B [13].

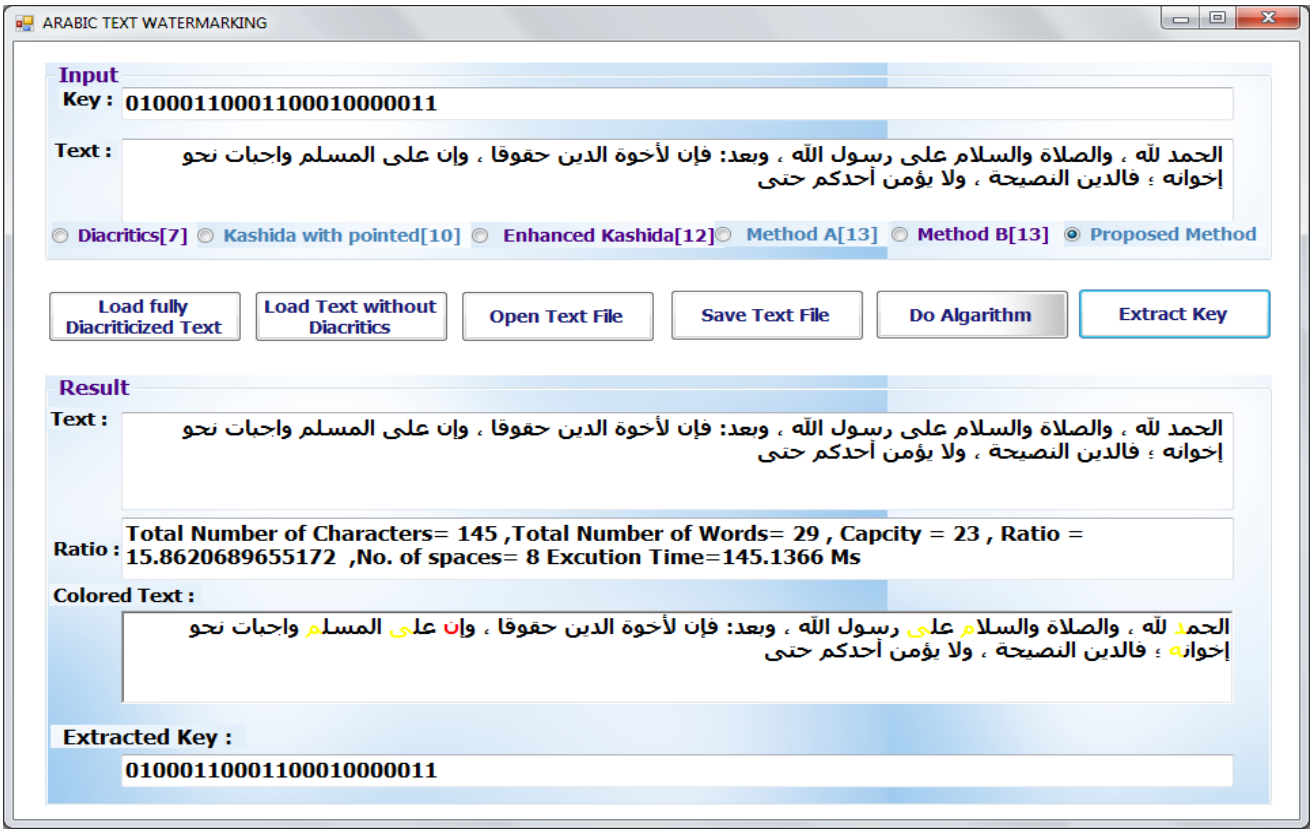

Figure 5. GUI used to test and compare the proposed watermarking method.

TABLE I. AN EXAMPLE OF A SAMPLED TEXT AFTER APPLYING THE WATERMARKING METHODS

Original text	الحمد لله ، والصلاة والسلام على رسول الله ، وبعد: فإن لأخوة الدين حقوقا ، وإن على المسلم واجبات نحو إخوانه ؛ فالدين النصيحة ، ولا يؤمن أحدكم حتى
Letter points and kashida [10]	الحمد لله ، والصلاة والسلام على رسول الله ، وبعد: فإن لأخوة الدين حقوقا ، وإن على المسلم واجبات نحو إخوانه ؛ فالدين النصيحة ، ولا يؤمن أحدكم حتى
Enhanced kashida method [12]	الحمد لله ، والصلاة والسلام على رسول الله ، وبعد: فإن لأخوة الدين حقوقا ، وإن على المسلم واجبات نحو إخوانه ؛ فالدين النصيحة ، ولا يؤمن أحدكم حتى
Method A [13]	الحمد لله ، والصلاة والسلام على رسول الله ، وبعد: فإن لأخوة الدين حقوقا ، وإن على المسلم واجبات نحو إخوانه ؛ فالدين النصيحة ، ولا يؤمن أحدكم حتى
Method B [13]	الحمد لله ، والصلاة والسلام على رسول الله ، وبعد: فإن لأخوة الدين حقوقا ، وإن على المسلم واجبات نحو إخوانه ؛ فالدين النصيحة ، ولا يؤمن أحدكم حتى
Proposed Method	الحمد لله ، والصلاة والسلام على رسول الله ، وبعد: فإن لأخوة الدين حقوقا ، وإن على المسلم واجبات نحو إخوانه ؛ فالدين النصيحة ، ولا يؤمن أحدكم حتى

The proposed method has implemented using C#.net programming language. Five of Arabic text watermarking methods which are mentioned above are implemented to be used in comparative results. Fig. 5 shows our GUI when selecting the proposed method. Colored letters indicate that pseudo-spaces is inserted after it. Yellow color is used for binary bit 0 and red color is used for binary bit 1. Table I shows the results of Kashida based watermarking methods and the proposed method applied on sample text using the same watermarking bits. The inserted kashidas are colored with red. Comparing the resulted watermarked text from the proposed method and other methods in Table I, it is clear that the proposed method has higher imperceptibility in which the human eye cannot distinguish between the original and the watermarked text.

Four different texts with different sizes are used to evaluate the capacity of the proposed method. The capacity is the total embedding bits in the text while the capacity ratio [12] computed as :

$$Capacity\ ratio = \frac{\text{Number of hidden bits}}{\text{Total number of characters}} \times 100 \quad (1)$$

Tables II-V show the comparative results that obtained by testing the Diacritic based [7], Letter points and Kashida method [10], Enhanced Kashida Method [12], Enhanced Kashida Method A [13], Enhanced Kashida Method B [13] and our proposed one using four different documents with variable lengths. The comparative results show that the number of pseudo-spaces inserted by the proposed method is lower than the number of inserted kashidas by Letter points and Kashida method [10], Enhanced Kashida Method A [13] and Enhanced Kashida Method B [13], but is higher than Enhanced Kashida Method [12]. These inserted pseudo-spaces, unlike Kashida, are not observed by the human eye and do not change the overall space between words. This means that our proposed method has higher imperceptibility.

In terms of capacity, the proposed method has a higher capacity than other methods except the work in [10]. Kashida method with pointed letters [10] which provide higher capacity, but the used kashidas could be easily detected.

TABLE II. COMPARISON RESULTS USING DOCUMENT1 (TOTAL CHARACTER=145, NO. OF WORDS=29)

	Capacity	Capacity ratio	No. of kashidas or spaces
Diacritic based [7]	0	0	-
Letter points and kashida method [10]	24	16.5	16
Enhanced kashida method [12]	6	4.13	5
Method A [13]	14	9.65	12
Method B [13]	16	11.03	19
Proposed Method	23	15.86	8

TABLE III. C OMPARISON RESULTS USING DOCUMENT2 (TOTAL CHARACTER=11215, NO. OF WORDS=2032

	Capacity	Capacity ratio	No. of kashidas or spaces
Diacritic based [7]	0	0	-
Letter points and kashida method [10]	2016	17.97	2016
Enhanced kashida method [12]	624	5.56	624
Method A [13]	1391	12.40	1391
Method B [13]	1619	14.43	1596
Proposed Method	1804	16.08	912

TABLE IV. CO MPARISON RESULTS USING DOCUMENT3 (TOTAL CHARACTER=87254, NO. OF WORDS=16094)

	Capacity	Capacity ratio	No. of kashidas or spaces
Diacritic based [7]	861	0.98	-
Letter points and kashida method [10]	13319	15.26	13319
Enhanced kashida method [12]	4431	5.07	6648
Method A [13]	10918	12.51	16379
Method B [13]	12679	14.53	8095
Proposed Method	13544	15.52	6062

TABLE V. CO MPARISON RESULTS USING DOCUMENT4 (TOTAL CHARACTER=489346, NO. OF WORDS=93565)

	Capacity	Capacity ratio	No. of kashidas or spaces
Diacritic based [7]	29971	6.12	-
Letter points and kashida method [10]	80632	16.47	80632
Enhanced kashida method [12]	27197	5.55	30695
Method A [13]	65338	13.35	73505
Method B [13]	74458	15.21	67189
Proposed Method	81373	16.62	39346

V. CONCLUSION

In this paper, a new Arabic text watermarking method is presented by utilizing each space between words with a pseudo-space. If the letter before the space is pointed and the watermark bit is one, the pseudo-space is inserted, else no pseudo-space is inserted. If the letter before the space is un-pointed and the watermark bit is zero, the pseudo-space is inserted else, no pseudo-space is inserted. The proposed method produces higher imperceptibility and capacity. The human eye cannot distinguish between the original text and the embedded one. It is also simple to implement and does not need complex computation.

For a future work, the letter after the space can be used in addition to one before the space in order to improve the capacity.

REFERENCES

[1] C.-S. Lu, *Multimedia Security: Steganography and Digital Watermarking Techniques for Protection of Intellectual Property: Steganography and Digital Watermarking Techniques for Protection of Intellectual Property*. United States: Igi Global, July. 2004.

[2] I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker, *Digital watermarking and steganography*, 2 ed. United States: Morgan Kaufmann, Nov. 2007.

[3] S. Stanković , I. Orović and E. Sejdić, *Multimedia signals and systems*. New York: Springer US, Sep. 2012.

[4] R. A. Alotaibi and L. A. Elrefaei, "Arabic Text Watermarking : A Review", *International Journal of Artificial Intelligence & Applications (IJAIA)* Vol. 6, No. 4, pp. 01-16, July. 2015.

[5] M. H. Shirali-Shahreza and M. Shirali-Shahreza, "A new approach to Persian/Arabic text steganography," in Computer and Information Science, 2006 and 2006 1st IEEE/ACIS International Workshop on Component-Based Software Engineering, Software Architecture and Reuse. ICIS-COMSAR 2006. 5th IEEE/ACIS International Conference on, July. 2006, pp. 310-315.

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

[7] M. A. Aabed, S. M. Awaideh, A.-R. M. Elshafei, and A. A. Gutub, "Arabic diacritics based steganography," in *Signal Processing and Communications, 2007. ICSPC 2007. IEEE International Conference on*, Nov. 2007, pp. 756-759.

[8] M. L. Bensaad and M. B. Yagoubi, "High capacity diacritics-based method for information hiding in Arabic text," in *Innovations in Information Technology (IIT), 2011 International Conference on*, April. 2011, pp. 433-436.

[9] A. Shah and M. S. Memon, "A novel text steganography technique to Arabic language using reverse fatah," *Pakistan Journal of Engineering Technology & Science (PJETS),* vol. 1, pp. 106-113, 2011.

[10] A. A.-A. Gutub, L. Ghouti, A. A. Amin, T. M. Alkharobi, and M. K. Ibrahim, "Utilizing Extension Character'Kashida'with Pointed Letters for Arabic Text Digital Watermarking," *Proceedings of International Conference on Security and Cryptography ,* in SECRYPT, July. 2007, pp. 329-332.

[11] F. Al-Haidari, A. Gutub, K. Al-Kahsah, and J. Hamodi, "Improving security and capacity for arabic text steganography using 'Kashida'extensions," in *Computer Systems and Applications, 2009. AICCSA 2009. IEEE/ACS International Conference on*, May. 2009, pp. 396-399.

[12] Y. M. Alginahi, M. N. Kabir, and O. Tayan, "An enhanced Kashida-based watermarking approach for Arabic text-documents," in *Electronics, Computer and Computation (ICECCO), 2013 International Conference on*, Nov. 2013, pp. 301-304.

[13] Y. M. Alginahi, M. N. Kabir, and O. Tayan, "An Enhanced Kashida-Based Watermarking Approach for Increased Protection in Arabic Text-Documents Based on Frequency Recurrence of Characters," *International Journal of Computer and Electrical Engineering,,* vol. 6, no. 5, pp. 381-392, Oct. 2014.

[14] Korpela, J. (2016). *Unicode spaces*. [online] IT and communication. Available at: https://www.cs.tut.fi/~jkorpela/chars/spaces.html [Accessed 7 Mar. 2016].