Payload Capacity Scheme for Quran Text Watermarking Based on Vowels with Kashida

Alkhafaji, Ali A.R. ; Sjarif, Nilam Nur Amir ; Shahidan, M.A ; Azmi, Nurulhuda Firdaus Mohd ; Sarkan, Haslina Md ; Chuprat, Suriayati ; Khalaf, Osamah Ibrahim ; Al-Khannaq, E.N.M.

DOI
10.32604/cmc.2021.015803

Publication date
2021

Document Version
Final published version

Published in
CMC Computer, Materials and Continua

Citation (APA)
Alkhafaji, A. A. R., Sjarif, N. N. A., Shahidan, M. A., Azmi, N. F. M., Sarkan, H. M., Chuprat, S., Khalaf, O. I., & Al-Khannaq, E. N. M. (2021). Payload Capacity Scheme for Quran Text Watermarking Based on Vowels with Kashida. *CMC Computer, Materials and Continua*, *67*(3), 3865-3885. https://doi.org/10.32604/cmc.2021.015803

Computers, Materials & Continua
DOI:10.32604/cmc.2021.015803
Article

Tech Science Press

Payload Capacity Scheme for Quran Text Watermarking Based on Vowels with Kashida

Ali A.R. Alkhafaji[1,*], Nilam Nur Amir Sjarif[1], M.A Shahidan[1], Nurulhuda Firdaus Mohd Azmi[1], Haslina Md Sarkan[1], Suriayati Chuprat[1], Osamah Ibrahim Khalaf[2] and Ehab Nabiel Al-Khanak[3]

[1]Razak Faculty of Technology and Informatics, University Teknologi Malaysia, Kuala Lumpur, 54000, Malaysia
[2]Al-Nahrain University, Al-Nahrain Nanorenewable Energy Research Centre, Baghdad, Iraq
[3]Department of Transport and Planning, Faculty of Civil Engineering and Geosciences (CiTG), Delft University of Technology, Delft, Netherlands
[*]Corresponding Author: Ali A R. Alkhafaji. Email: aliraheem1983@gmail.com
Received: 07 December 2020; Accepted: 12 January 2021

Abstract: The most sensitive Arabic text available online is the digital Holy Quran. This sacred Islamic religious book is recited by all Muslims world-wide including non-Arabs as part of their worship needs. Thus, it should be protected from any kind of tampering to keep its invaluable meaning intact.

Different characteristics of Arabic letters like the vowels (أ. و. ي), Kashida (extended letters), and other symbols in the Holy Quran must be secured from alterations. The cover text of the Quran and its watermarked text are different due to the low values of the Peak Signal to Noise Ratio (PSNR) and Embedding Ratio (ER). A watermarking technique with enhanced attributes must, therefore, be designed for the Quran's text using Arabic vowels with kashida. The gap addressed by this paper is to improve the security of Arabic text in the Holy Quran by using vowels with kashida. The purpose of this paper is to enhance the Quran text watermarking scheme based on a reversing technique. The methodology consists of four phases: The first phase is a pre-processing followed by the second phase-the embedding process phase—which will hide the data after the vowels. That is, if the secret bit is "1", then the kashida is inserted; however, the kashida is not inserted if the bit is "0". The third phase is the extraction process and the last phase is to evaluate the performance of the proposed scheme by using PSNR (for the imperceptibility) and ER (for the capacity). The experimental results show that the proposed method of imperceptibility insertion is also optimized with the help of a reversing algorithm. The proposed strategy obtains a 90.5% capacity. Furthermore, the proposed algorithm attained 66.1% which is referred to as imperceptibility.

Keywords: Quran text watermarking; Arabic text; capacity; imperceptibility; Kashida; vowels; reversing technique

1 Introduction

A digital watermark is a digital signal or sample that can also be seen as a sort of digital signature inserted right into digital data for the protection of copyright ownership of digital statistics, including text, image, video, audio, etc; [1]. Two main processes are involved in the watermarking system: (1) incorporating the watermark into the original data and (2) extracting the watermark from watermarked data or attacking watermarked data. Capacity, imperceptibility, robustness and protection are the requirements for watermarking. Capacity is the total number of hidden bits in an object. Imperceptibility is used to measure the difference between the original and watermarked object by noting any addition to the original object. Robustness is the ability to remove or detect the watermark after the watermarked object has been attacked. Finally, protection is when the watermark is difficult to remove without the watermarked item being lost [2].

Text watermarking has become important due to developments in the field of information hiding. Protection of such information is necessary and useful as the copying of digital media and preserving it unchanged is not easy. Nevertheless, the watermark is considered the best and most perfect application to host secret text, identify the ownership, acquire authentication and keep documents secure. Additionally, many researchers aim to use such watermarking applications so much so that it has become an important and necessary field of study [3].

There are many methods for embedding secret data in text for different languages each according to an individual language's features. As the Quran is written in Arabic, one needs to know the features of the language in order to hide secret data within it. In this regard, of the many tasks watermarking can perform, a few of these are used in the "kashida" letter extension (one of the Arabic language's features) to embed diacritics, a secret message, the order of the words in statement and the conjunctions of the Arabic language as well. Even hyper (the mixing of two or more features) are used to hide a secret message in Quran's text. Each of these methods has the advantages and disadvantages, for this reason, it needs to come up with a new algorithm in a view of overcoming all the disadvantages of existing methods [4].

The contributions of this paper is hoped to improve the embedding capacity and imperceptibility by proposing that the Quran text watermarking method be used in preserving the original text and meaning of the Quran which is based on vowels with three different types of characters (ي,و,أ) in the Quran's text (which is the most frequently used in the Arabic language) and the redundant letters in the word. The proposed method uses the reversing technique which was chosen to be added to the kashida after each vowels to get an elevated imperceptibility. The existence of these vowels is used to hide bit "1", in kashida and the absence of the kashida is used to hide bit "0".

Arabic is used as the principal language in all Arab countries of the Middle East and Northern Africa. It is considered the world's 5th most influential language. It is central to other languages in the Muslim world such as Farsi (Persian), Urdu, Sindhi and Pashto. Some minority languages in China such as Uighur, Kazakh, and Kirghiz are all written using a modified Arabic script. The Arabic alphabet is actually an abjad which includes 28 characters and has many unique characteristics. For example, Arabic script is always written from right to left, there are no capital letters in this language, and each letter in Arabic takes different forms according to its location in the word. In addition, Arabic has many characters which may have one, two or three dots placed above or below some letters. Each Arabic word contains more than one character that are linked together. This connection feature is useful in terms of data hiding. Furthermore,

Arabic contains 15 characters, five of which are multi-point characters, unlike English, which does not contain multiple characters. Meanwhile, each word often has some special marks called "Harakaat" [5]. Keeping in mind, the Arabic text has something called diacritics added above or below the characters in order to form the vowel sounds. There are eight Arabic text diacritics (Harakaat) as shown in Fig. 1.

Fatha	ـَ	Kasrah	ـِ
Dhammah	ـُ	Sukkon	ـْ
Shaddah	ـّ	Tanween Fath	ـً
Tanween Kasr	ـٍ	Tanween Dham	ـٌ

Figure 1: Main Arabic diacritics [5]

These diacritics are represented by several digitally inserted characters as separate letters found inside the computer's 0 location. The use of the Arabic grammatical markings in the Arabic language is mentioned according to the standards and practices of modern Arabic writing. Therefore, it is necessary for the Quran and most religious texts [6].

Moreover, the "Kashida" which is used between Arabic characters has a well-known extension character [7]. Most of the Arabic letters have been written with "Kashida" based on their position in the word. It is utilized as a form of text justified for some cursive scripts similar to Arabic. White-space justification expands the spaces between words. Hence, it will increase the length of the text line. Along these lines, Kashida justification is achieved by elongating the character at certain chosen points [8,9]. Vowels in Arabic can be defined as animated sounds which help to determine word pronunciation, and writing them is the same as normal but their placement gives them different pronunciations. Choosing these letters supposedly helps find the positions for embedding the secret data as shown above in Fig. 2.

Letters	Frequency	Letters	Frequency	Letters	Frequency	Letters	Frequency
ā (ا)	600	n (ن)	221	k (ك)	112	ḏ (ذ)	35
l (ل)	437	r (ر)	155	d (د)	92	ṣ (ص)	32
m (م)	320	ʿ (ع)	131	s (س)	91	ḫ (خ)	20
h (ه)	273	f (ف)	122	q (ق)	63	ṯ (ث)	17
(*) ū+w (و)	262	t (ت)	120	ḥ (ح)	57	ṭ (ط)	15
(*) ī+y (ي)	252	b (ب)	112	ǧ (ج)	46	ḡ (غ)	15
						ẓ (ظ)	8

Figure 2: Frequency of vowel letters [6]

A watermarking process based on kashida with a dotting property has been presented in [10]. This approach is achieved via inserting the kashida before or after letters containing points to indicate bit 1. Meanwhile, inserting the kashida before or after letters that do not contain points given indicating a bit 0. The kashida is integrated into a particular pattern in this approach.

Therefore, changing the insertion process has been implemented to increase security. Regarding another mechanism which has been proposed in [11], one kashida is inserted after any letter can hold it to represent a bit 0, and two consecutive kashidas are inserted to represent a bit 1. Kashida which is mentioned above used to describe whether inserting 1 or 0. Quran text watermarking using vowel characters with the absence of Kashida in the word means 0 otherwise, mean 1. The Kashida (or extension of the letter) does not change the meaning of the word in the Arabic language. Hence, the presence or absence of a Kashida will not affect the meaning of the text, rather just gives the text an informal style.

2 Related Work

A state-of-the-art review of the recent Arabic text watermarking was studied and summarized in this part. Generally, text watermarking is one of the most challenging kinds of watermarking methods. Text watermarking tries to exploit the characteristics of the letters or the entire text in a certain language to hide information. Text watermarking comes with its own difficulties when compared to the watermarking of images or audio as the text usually has a relative lack of redundant information that makes it difficult for hiding data. However, texts differ from one language to another. For instance, the Arabic language has specific characteristics in the context of writing which makes text watermarking much more applicable as compared to other languages [11–13].

The Arabic language uses special characters such as kashida, diacritics, special letters and vowels, not to mention special shapes of certain letters to give the context a higher level of clarity as well as making the shape of the context much more artistic. It should be mentioned that the Arabic language has 6 types of artistic writing which can be recolonized from the shape of the letters. These writings were gradually developed through antiquity and are used even until today. For instance, the kashida is a method to lengthen some of the letter in Arabic text and connect it to other letters on the right side (Arabic script is written from right to the left), while some letters have accepted the insertion of the kashida after it [9–14]. The Quran was written in a distinct Arabic language and has unique characteristics [11–15]. In addition to diacritics, the Quranic text includes "tajweed" symbols–symbols used in the recitation rules that direct readers to correctly recite the Quran. These tajweed symbols are not used in Arabic literature and are, therefore, unique to the Holy Quran [10–16].

Many studies have tried to use watermarking in Arabic texts or in the text of the Quran. For example, [17–19] used the diacritic "Fatha" to hide information. This method utilizes the usual form of the Fatha in order to use code 1 while using the revised form of fatha to use code 0. The method is very brilliant in terms of capacity because it does not need a big computational ability and may be achieved manually; however, it has a lack of imperceptibility and it is not appropriate to be added to the text, as shown in Fig. 3.

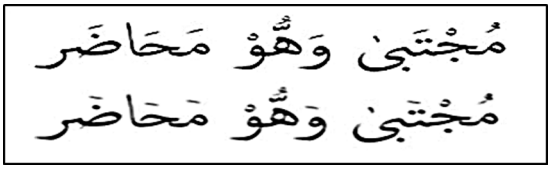

Figure 3: Information hiding in cover text with reverse Fatha [17]

The use of diacritics in Arabic text is necessary for those involved in the text analysis. However, diacritics-based watermarking is seen as a critical issue if the original text has been dispossessed of the diacritics in the detection process. Therefore, being able to perform diacritics-based watermarking at 100% depends on the number of diacritical marks in the text. There are 22 letters in the Arabic language that are usually connected to form a single word. Sometimes a single letter in the Arabic language can be written in four different shapes according to the location of the letter in a certain word as shown above in Fig. 4.

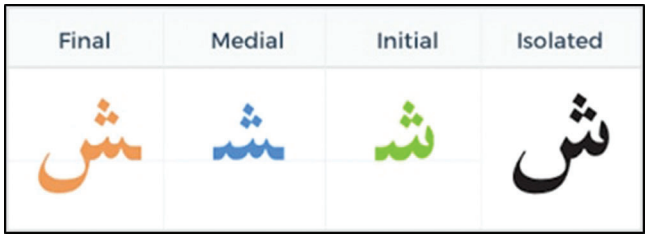

Figure 4: Letter shapes according to its position in the word [19]

A Unicode method suggesting a unique number for each letter in the Arabic language for programing purposes has been developed by [20]. Many relevant works have been implemented by utilizing the Unicode for programming. For example, a strategy based on the Unicode method has been achieved in view of hiding the secret information into Arabic cover text [21]. A secret data 0 and 1 are embedded in a run-length encoding format. However, this approach is restricted to hiding data using isolated letters. Moreover, the approach was suitable when the text appears in its non-diacritics form while it comes up short when there are a diacritics included within the text.

Conversely, six out of the 28 Arabic letters are categorized as special letters that do not connect with other letters when these letters fall at the beginning or in the middle of the word. These special letters are ((ا) *alif*, (د) *daal*, (ذ) *thaal*, (ر) *raa*, (ز) *zaa*, and (و) *waaw*). Moreover, these letters can be utilized as a word divider. For instance, in the Arabic word (الرسالات), the word is not connected due to the (ا) *alif* appearing at the beginning of the word. Likewise, (ر) *raa* is not connected to the succeeding letter and (ت) *taa* appears as an isolated letter at the end of the word as there is an (ا) *alif* fall before it that is one of the six letters defined early [22].

Some scholars have used the kashida and dotted letters in the Arabic language for coding. An approach based on coding a secret bit 'one' in the doted letter followed by Kashida while coding a secret bit '0' with the un-pointed letters followed by the kashida has been presented in [23] It should be noted that an extended letter with the kashida does not have any impact on the writing content as depicted in Fig. 5.

Dividing the Arabic letters into two sets based on the occurrences of these letters has been presented in [24]. The potential of detecting the kashida in the text is considered to be the weakness of Kashida-based strategies. In addition, the letter of the word capacity to be extended which makes the kashida can be inserted only in such a certain place in the word.

Hiding information in Arabic text via using sun and moon letters in the Arabic language has been proposed in [25]. This approach has been implemented by utilizing four scenarios. The first scenario, kashida is inserted after the sun letter given hiding the secret bits 00. The second is performed via inserting two kashida after the sun letter in order to hide the secret bits 11. Regarding the third scenario, Kashida is inserted after the moon letter for hiding the secret bits

01. The fourth scenario or characters is achieved by inserting two Kashida to the moon letter for hiding the secret bits 1, 0. This method tries to increase the probability of hiding the secret bits in any letter via using the moon and sun letters concept as shown in Fig. 6. Nonetheless, it is very important to maintain the imperceptibility aspect while improving the capacity.

Secret bits	110010
Cover-text	من حسن اسلام المرء تركه ما لا يعنيه
Steganographic text	مـــن حسن اسلام المرء تـــركـــه مـــالا يـــعنـــيه ↑ ↑ ↑ ↑ ↑ ↑ 1 1 0 0 1 0

Figure 5: Adding extensions after letters [23]

Moon letter				Sun letter			
1	أ	8	خ	1	ت	8	ش
2	ب	9	ف	2	ث	9	ص
3	غ	10	ع	3	د	10	ض
4	ح	11	ق	4	ذ	11	ط
5	ج	12	ي	5	ر	12	ظ
6	ك	13	م	6	ز	13	ن
7	و	14	هـ	7	س	14	ل

Figure 6: Moon and Sun letters [25]

Increasing the capacity of kashida by adding white spaces as an extra redundant cover has been proposed in [26]. This method has proved to be more obvious in terms of capacity without security deterioration. Besides, this method can help users for replacing information within the text and making secure connections at the same time. In contrast, the kashida cannot be added at the beginning or end of words which add a disadvantage to this method. The Kashida can only be added in order to join the characters in words [27].

The Holy Quran consists of 30 parts (juz') which includes 114 *surah* (chapters). Some surah are as short as three *ayahs* (*vs.*) and some can reach up to 286 *ayahs* [28]. An approach based on converting Uthmani text into plain text has been presented in [28]. This method uses the Tuned BM algorithm (BMT) exact with XOR operation pattern matching algorithm to verify the substituted Uthmani verse with a given database of plain Quran style. Experimental results show that the proposed approach is useful and effective in authenticating multi-style texts of the Quran with 87.1% accuracy with a low capacity.

The Holy Quran consists of 30 parts (juz') which includes 114 *surah* (chapters). Some surah are as short as three *ayahs* (*vs.*) and some can reach up to 286 *ayahs* [28]. An approach based on converting Uthmani text into plain text has been presented in [29]. This method uses the

Tuned BM algorithm (BMT) exact with XOR operation pattern matching algorithm to verify the substituted Uthmani verse with a given database of plain Quran style. Experimental results show that the proposed approach is useful and effective in authenticating multi-style texts of the Quran with 87.1% accuracy with a low capacity.

Many studies have suggested the used of diacritics or kashida instead of vowel characters for hiding information [11–29] Redundant letters in the word or inverses are also used to hide secret bits in Quran watermarking. However, these methods show a weakness in terms of authentication, capacity and imperceptibility. Therefore, the Arabic vowels أ, و, ي are proposed for hiding information in the Holy Quran since it is written in the Arabic language as illustrated in Fig. 7.

Arabic vowel letters
أ (a)
و (u +w)
ي (I +y)

Figure 7: Arabic vowel letters [11]

Unfortunately, there are no existing studies in which the Arabic vowels أ, و, ي are implemented to investigate the imperceptibility and capacity for different styles of Quran texts. Hence, the focus of this paper is to propose an approach that can solve the imperceptibility and capacity issues of Quran texts by using either a kashida or a non-kashida after each vowels.

A reverse process or technique means changes to the opposite process. In this context, reversing can be accomplished by writing down the sequence of the process and starting backwards from the end and working your way to the beginning of the text as is necessary in the reverse process. In the watermarking technique, there is a process called embedding when hiding or inserting the data within the cover media in the sender's part and the other part (the receiver's part) tries to reverse the process to get the hidden data [29].

A multilevel histogram technique which is modified for the implementation of vector maps reversible watermarking strategy has been presented in [30]. The embedding process of this method relies on the vector map features construction. This method retains the original information accuracy with no impact on the normal map information. Moreover, a lossless carrier map for information recovery is obtained after the watermark extraction process. Regarding these lines, this mechanism can be considered to be most applicable in many areas like vector map integrity authentication and converting communications. In this method, the embedded distortion is controlled efficiently with a bit large watermarking capacity. The main advantage of this algorithm is that it can obtained better embedding performance as compared with other relevant works based on the vector map reversible watermarking techniques. However, the drawback of this algorithm is that the low redundant vector information as compared with the rest information as well as small embedding capacity. A binary up-down counter and gray code counters are employed as address generators [31]. With a concern to still decrease the switching activity, a new address generator has been developed and executed via utilizing a bit reversal technique along with a modulo counter and grey code converter as demonstrated above in Fig. 8.

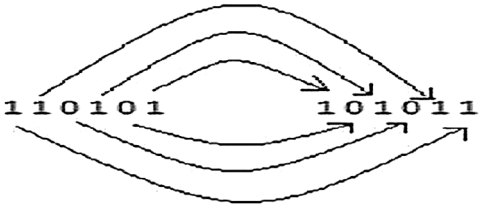

Figure 8: Reversing technique [29]

A novel reversible data hiding technique in encrypted images is presented by [32]. The reversing of some pixels is estimated before encryption procedures so that the additional data can be embedded can help in estimating errors. Hence, instead of embedding data in encrypted images directly, a benchmark encryption algorithm is applied to the rest pixels of the image and a special encryption scheme based on reversing is designed to encrypt the estimating errors.

3 Proposed Watermarking Method

Watermarking is the procedure of hiding data into certain media like that found in literature including image, audio, video, and text. Most techniques presented in watermarking utilize images, audio, videos as the cover media. However, the text watermarking technique is interesting using a new algorithm to hide the secret bits within Quranic text. A novel algorithm suggests including four main phases. First, the pre-processing phase that is responsible for preparing the hosting media (such as the Quranic text in the case) and secret bits (data that was hidden in the text). The second phase is the embedding of the secret bits within the Quran text. The third stage involves the extraction of the data including the attack process. The final phase was the performance evaluation of the scheme in terms of various measures. The details of these phases are discussed as follows:

3.1 Preprocessing of the Quran Text Watermarking

In the proposed watermarking scheme, data pre-processing is performed on the secret watermark and hosting media. The ASCII code is represented in the computer as a hexadecimal system that starts from 0600 and ends with 06FF. In the Quran text, the hexadecimal is used as a location scheme with the base of 16 to describe the numbers. Compared to the traditional method of expressing numbers with ten symbols, it uses sixteen distinct symbols. Most commonly, "0" to "nine" symbols reflect the "0" to "nine" whereas an "A" to "F" to reflect the values from "ten" to "fifteen". Each character has its hexadecimal number even the characteristics of the language such as kashida and diacritics. When inserting one kashida the statement is increased by one more letter with a representation of a hexadecimal number. In this regard, Quranic text is changed to the ASCII code before embedding so that it can analyze the letters of the Quran into the ASCII as shown in Tab. 1.

In addition to the Quran text preparation, the secret bit is also converted into ASCII code and then into the binary code of zero (0) and one (1). To increase the capacity of the secret data embedded, the compression was performed with the secret data. After this process, the preprocessing phase is complete, the Quran text is covered with a secret bit and is now ready for the next phase-embedding.

Table 1: Letters of some Quran text with the corresponding hexadecimal code

	مي ح ر ا ل	ن م ح ر ا ل	ه ل ل ا	م س ب
Arabic letters	م ي ح ر ا ل ن م ح ر ا ل ه ل ل ا م س ب			
Hexadecimal	0645 064A 062D 0631 0644 0627 0646 0645 062D 0631 0644 0627 0647 0644 0644 0627 0645 0633 0628			

3.2 Embedding Process of Quran Text Watermarking

This phase includes three processes which are locating or positioning the letters, adding the kashida to achieve the embedding, and checking the match of the secret bits with the bits of the Quran text for embedding decision as shown in Fig. 9. The first process is necessary for the imperceptibility that measures the robustness immunity of the text watermarking scheme against any attempts that causes the removal or degradation of the watermark. The PSNR value is the measure or evaluation criteria of the imperceptibility. The high imperceptibility means the scheme is more robust against attack; otherwise, it reflects the weakness of the scheme.

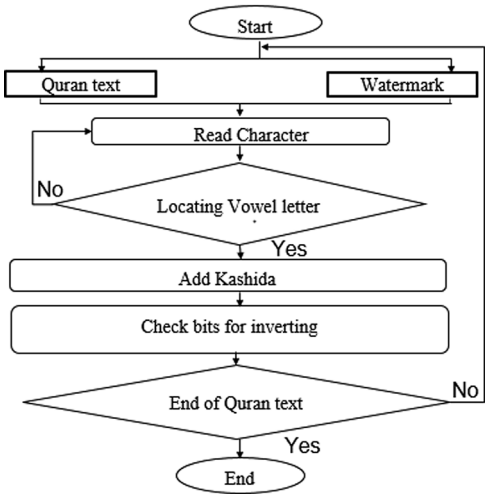

Figure 9: Embedding algorithm processes

Each of the abovementioned processes depends on the previous one in the sequence. For example, the addition of the Kashida cannot happen unless the right position is determined and this phase involves the following three sub-processes:

3.2.1 Locating Vowel Letters

The Quran text consists of the statements called *ayat* (*vs.*), where each *ayat* is composed of several words and each word is made of letters. There are three letters that act as vowels (ا, و, ي), and they are the most frequently used letters in the Arabic language. This is why these letters were chosen., They can be determined by comparing the ASCII code of each letter as shown in Fig. 10.

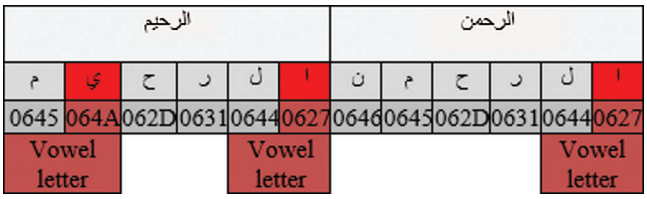

Figure 10: Locating the vowel letters in the Arabic text

3.2.2 Add Kashida

The extension letter or Kashida occurs when the bit needed to be hidden or embedded equals one (1). When a secret bit is equal to 0 then there is no Kashida. The addition of the kashida to the word does not change the meaning of the word. The mechanism of adding Kashida is to extend (by Kashida) the same vowels if possible or if next letter to it represents (1) bit of secret bits. In the same context, the absence of kashida in these positions represents 0 of the secret bit as shown in Figs. 11 and 12.

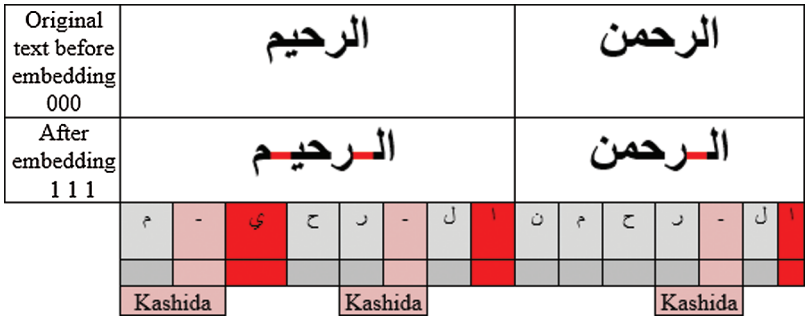

Figure 11: Add the Kishida

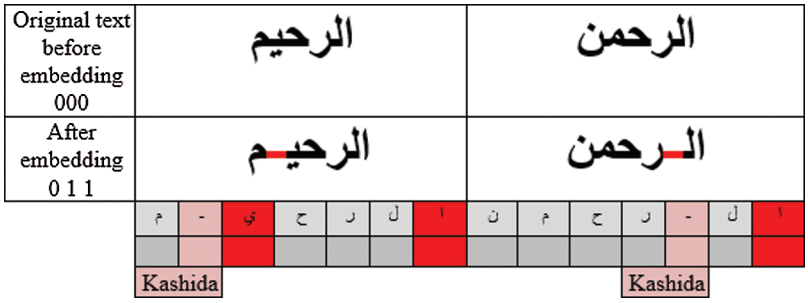

Figure 12: Add or remain of the word Kashida in different bit

Embedding involves three sub-processes: first, the embedding of bit (1) was done after the vowels wherein Kashida was inserted to represent a bit (1) similar to the vowels in the second word. Second, in the last vowels, the Kashida was inserted into the vowel itself (meaning the letter ي) because the Kashida cannot be inserted to show its value at the end of the word. Third, a bit (0) represented the absence of kashida. However, if the receiver could detect the absence of

kashida in this position then it was considered as bit (0). In case the vowels were located at the end of the word then the inclusion of kashida was ignored.

3.2.3 Checking Bits for Inverting

Increasing the complexity of the techniques to make it more difficult to solve, the hidden serial bits must be manipulated before embedding. In this regard, bits of 0's are considered without affecting the words or verses. However, the expected redundancy of the embedding bits should be avoided. To proceed from the reversing, the most similar secret bits with cover text bits must be considered.

As displayed in Fig. 13, comparing a number of bits (experimentally chosen) in a secret bit was considered with the corresponding bits in the original text (with a specified position in advance). If half of the secret bits match. then the secret bits are reversed or kept the same otherwise. The main purpose of reversing the secret bits was to reduce the modification of the text as much as possible. More changes in the Quran's text may be noticeable or recognized by an intruder, wherein the imperceptibility measure can determine the extent of such changes in the hosting media and its difference compared with the original media. Following this method, it is possible to achieve higher security because opening such a technique is almost impossible unless the secret key is found. Increasing the authenticity is one of the main objectives of the watermarking issue. Thus, it can be concluded that the first step in the embedding of the proposed scheme is to locate the appropriate position under certain conditions before checking the similarity of the secret bits with the original text. Finally, add kashida according to (1) or (0).

Figure 13: Strategy of the bit reversing

The main purpose of performing changes or using a reverse process is to reduce the modification of the text as much as possible. More changes in Quran's text may be noticeable or recognizable by the intruder. Keep in mind that the imperceptibility is the measurement of how much the host media is changed and how much of the difference between the change and the original media is noticeable. By using this method, more security can be obtained because of the robustness of the proposed algorithm making it almost very difficult unless finding out the key. In addition, increasing authenticity is one of the objectives of watermarking text. Regarding Fig. 13, once can conclude that the first step of the embedding process of the proposed method is locating an appropriate position under such a certain condition, checking the similarity of secret bits with the original text, and finally adding Kashida according to 1 or 0. Fig. 14 illustrates the Embedding Algorithm of the proposed strategy.

1. **Stage one:** decode secret bit from watermark
2. **For** all binary image watermark **Do**
3. **Extract** secret bit as
4. **If** a[watermark] =255 **then** secret bit =0
5. **Else** Secret bit=1;
6. **Store** vector of secret bits
7. **Update** watermark as secret bits
8. **Stage two:** open original text for reading
9. **Repeat** for all text file
10. **Check** vowel letter
11. **If** vowel letter **then** check condition
12. **Store** position in vector
13. **Loop** for embedding
14. **If** secret bit =1 **then** embed Kashida
15. **Else** no Kashida
16. **Close** loop
17. **Create** watermarked key
18. **Update** original text as watermarked text

Figure 14: Embedding algorithm of algorithm

3.3 Extracting Process of Quran Text Watermarking

The extraction process is necessary to remove the hidden data from the watermarked text. This process is exactly the reverse of the embedding stage. First, the receiver needs a watermarked Quran text. In addition to the original hosting media, the secret bit is needed due to it is containing the most important information for extraction. Without the secret bit, the receiver cannot extract any information and then everything is for naught. The extracting procedure of the proposed approach is accomplished by looking for the vowels in order to extract the watermarked text from it. It can follow the color of each process as shown in Fig. 15.

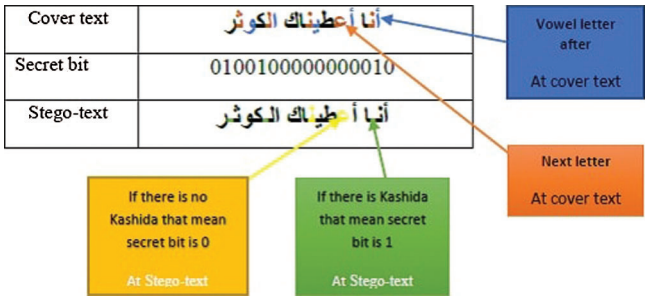

Figure 15: Extracting the secret bits

This process is exactly the reverse of the embedding procedure. Fig. 16 shows the exact sub-processes within extraction. Therefore, read one verse text inside the watermarked text (included the key) then pointing the located of vowels so it can check the extracting process. If the difference is more than half then have to reverse the secret bits, then assign with the next vowels. If the condition true, then continue the following bit for reversing. Thereafter the embedding process,

the secret bit has to check the text concerning the original cover. If the text is the same as the original cover it means that the condition is verified.

Otherwise, the extracting algorithm of the proposed approach starts to read again. Fig. 17 illustrates the extracting Algorithm of the proposed strategy.

1. **Stage one:** collect data from watermarked key
2. **Open** watermarked key
3. **Extract** vectors from watermarked key
4. **Update** data inside it
5. **Stage two:** getting watermarked text
6. **Open** text file for reading
7. **Apply** strategy of data in the watermarked key
8. **Get** vector
9. **Repeat** for all watermarked text
10. **Check** vowel letter
11. **If** vowel letter **and** condition, **then** extract according condition
12. **Loop** for extracting
13. **If** Kashida **then** secret bit =1
14. **Else** secret bit = 0
15. **Store** secret vector
16. **Close** loop
17. **Update** watermarked text as original text
18. **Stage three:** Reconstruct logo
19. **Get** secret bit vector
20. **Loop** for all vector
21. **If** secret bit =1 **then** set pixel value= 0 (black)
22. **Else** set pixel value = 1 (white)
23. **Update** pixels as logo

Figure 16: Extracting algorithm of the proposed strategy

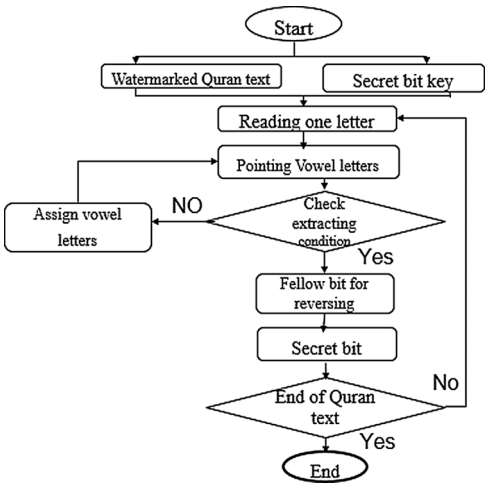

Figure 17: Extracting process of the proposed method

3.4 Experimental Evaluation and Results

The implementation of the proposed algorithms has been achieved via using MATLAB software. In order to evaluate the performance of the proposed approach, it has been considered the standard and authentic version of Quran datasets from tanzil.net [33] respectively. These standards have been used in previous research. This dataset was further verified by experts. Moreover, the

standard Quran datasets which have been utilized in this work are selected based on the payload capacity. The performance evaluation of this work is performed through using four different payload capacity (Al-Kursi verse 4 Kelo bits (KB), Surah Al-i-Imran 16 KB, Surah An-Nisa and Surah Ya-sin 21.3 KB, Surah Al-Baqara and Surah Al-Kahf 23.2 KB). Fig. 18 shows the system interface of the proposed strategy.

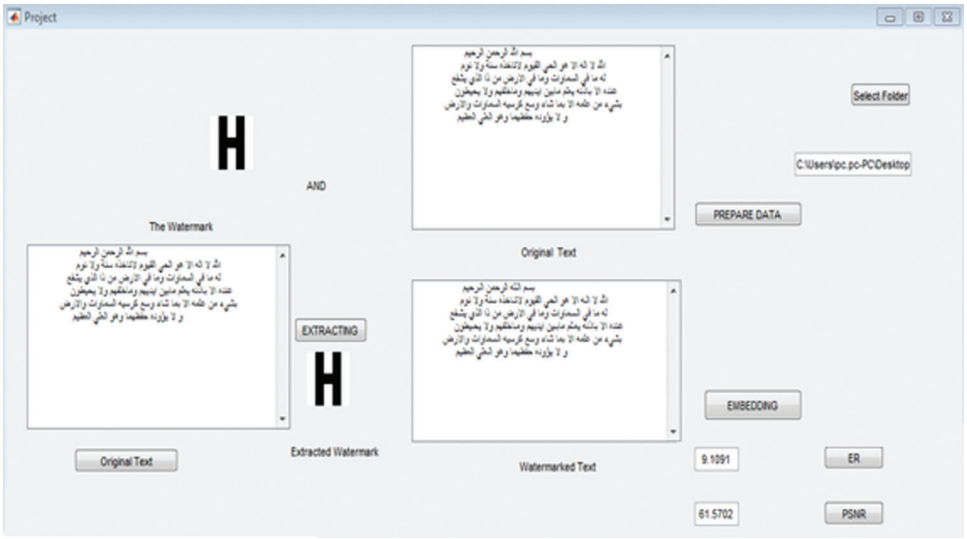

Figure 18: The proposed system interface

The main aim of the research is to improve the watermarking in terms of capacity. Capacity is determined by the embedding ratio and the efficiency ratio features. Embedding ratio refers to the amount of secret data embedded concerning the available places in the text file. On the other hand, the embedding ratio is considered the total satisfying conditions bit not the total size of the text file. Therefore, embedding ratio (ER) is important criteria in watermarking and steganography system. Embedding ratio can be defined according to Eq. (1).

$$\mathrm{ER} = \frac{\text{Total letters of the cover text\_Letters of the embedded Watermarked}}{\text{Total letters of the cover text}} \tag{1}$$

Following the above equation, it's clear that the main factor that affects the error rate is the total letters of the cover file. A big file size produces a less error rate. Regarding this fact, the embedding ratio will decrease so the percentage which is added to the file size is naturally lower as well. In this regard, it is important to mention the inverse method which is implemented to decrease the embedded watermarked letters for keeping the numerator high as much as possible ((Total letter of cover text (letters of the embedded watermarked)) in Eq. (1) which leads to an increase the embedding ratio (increase proportionally). Therefore, the percentage of embedding ratio provides a good perception of how much secret data can be carried by a certain media. In this type of text watermarking the main limitation is due to the monotone embedding methods which does not indicate any extra gain with others. Hence, it is necessary to find some method for scamming the intruder. To overcome the disadvantages of the existing methods the proposed scheme used a novel embedding process. Fig. 18 illustrates the results of the embedding ratio (ER) based on reversing technique for different payload capacity and a 25 bits size.

In addition, applying the reversing technique will help the embedding ratio due to decrease the distortion of the cover text which can help on the other side to embed more amount of the secret bits. However, the embedding ration results show an exponential impact as noticed in Fig. 18. Thus, respect of performing two certain amounts of payload capacity of 16 KB and 23.2 KB which are unlike the impact among the payload capacity of 4KB and 16KB. This behavior proves the effective performance of the proposed method.

In light of view the benchmarking of the proposed method, a significant criterion has been achieved based on embedding ration for portraying the power of the proposed strategy as well as make it easy to compare with the state of art relevant works. Along these lines, the comparative analysis has been accomplished in order to come up with the improvement of the proposed algorithm and riding of the drawback outcomes of the relevant methods as well. Tab. 2 shows the benchmarking of the proposed method which has been compared with the state of art relevant works.

Table 2: Comparison of the capacity evaluation

Methods	Capacity evaluation (ER) (%)
[34]	61.6
[35]	73.4
Proposed method	90.58

In accordance to the results which is illustrated in Tab. 2, the average embedding ratio for the first state of artwork [34] is considered well because it was developed based on the concept of adding kashida after any letter. Hence, this method does not rely on a certain characteristic possessed by any letter in the cover text. The results for the average embedding ratio are 61.16%, respectively. The first features which are the moon and sun letters allow secret bits to be hidden in any letter as all Arabic words will contain either moon or sun letters. Besides, this method allows two secret bits to be hidden in a letter. Moreover, the results of other relevant work are automatically generating a steganographic text based on Markov chain model and Huffman coding [34]. It can automatically generate fluent text carrier in terms of secret information which need to be embedded. This proposed model can have relatively high concealment, while its embedding rate can still achieve 73.4%. This model can even achieve a higher embedding rate.

This proves that the proposed model can achieve relatively high concealment and hiding capacity at the same time by adjusting the average bit number of each word which has been embedded. The proposed method using vowels that can add a kashida after each these letters (أ, و, ي) in terms of secret bits which need to be embedded. The proposed method produces higher results compared to the other previse method in terms of the chosen features. The proposed method can have active high results by using Huffman algorithm, while its embedding radio 90.58%, which are using vowels to embed a Kashida after each of these letters in Quran text to active high capacity.

On the other hand, the distribution of the embedded secret bit to cover Quran text reflects the Peak Signal to Noise Ratio (PSNR) value, while the PSNR detect the frequency of the bits within cover Quran text. When the bits become heterogeneous or in other words more chaotic, then the PSNR will get high value. It has been proposed the reversing technique for embedding

procedures. High PSNR reflects good Quran text quality so all previous methods try to increase the PSNR [35].

Subjective methods depend on the observation of humans and judgment which means without using any reference criteria. PSNR used to measure the quality of Quran text after embedding and defined according to Eq. (2).

$$PSNR = 10.log_{10}(\frac{MAX_I^2}{MSE}) \tag{2}$$

With,

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2 \tag{3}$$

MAX is the maximum possible letters value of the text; m, n is the range of the whole text and number of *vs.*; I, K., are original and noisy text. The PSNR value is based on Mean Square Error (MSE) which has an adverse impact. The parameters of PSNR allow normalizing the Eq. (2.1), for all methods and text types.

In light of view the proposed system which has been implemented via utilizing the reversing technique, a high impact has been carried out especially when the cover text size is big enough. Therefore, the file size 16 KB obtains a PSNR of 64.28 dB without accomplishing the reversing mechanism of the embedding procedures. On the other side, through using the same cover text size and hiding the same secret bits with the aid of the reversing technique, the PSNR is equal to 69.21 dB as demonstrated in Fig. 19.

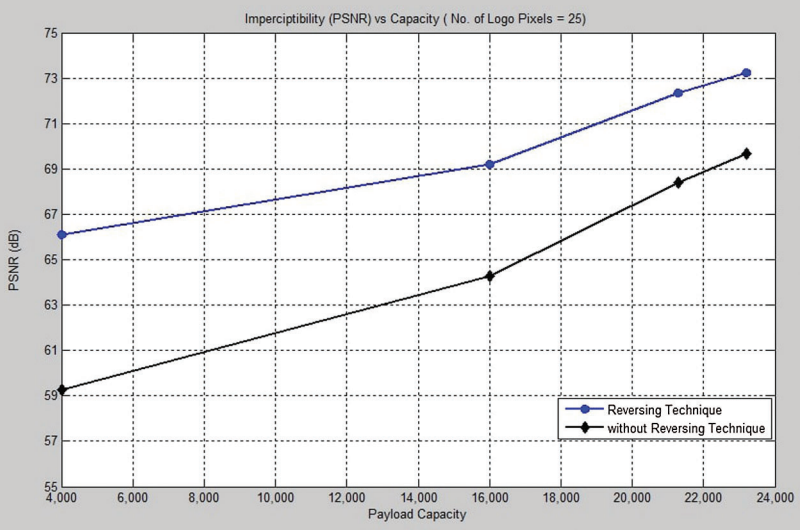

Figure 19: The imperceptibility evaluation using reversing technique

Regarding Fig. 19, it's clear to notice the difference between embedding with or without reversing technique. This Figure illustrates different payload capacity with a fixed secret logo which is 25 pixels. In the beginning, the difference about 7 dB in PSNR this gab due to the hosting size

of file text is few and reversing plays an important role in this case. The rest cases are different because of big file size allows using random function among such satisfying conditions, when finding more vowels which are ready to host the secret bit then it has to choose these letters randomly. In addition, increasing the file size will gradually increase the PSNR as well. It can recognize at the same Figure the steps or jumping from one limit capacity to other inconsistent steps but the last position not too much different with previous one this due to the available space for hosting secret bits. This is meaning that the availability of more secret bits will lead to making free space untapped.

One of the inverse relationships is the relation between the number of pixels in the logo and the imperceptibility. This inverse relationship will appear clearly at the time of increasing the number of logo pixels which is affecting on the distortion of the image and imperceptibility will be decreased slightly. Fig. 20 investigates the PSNR of the proposed strategy with respect to changing the number of logo pixels according to different payload capacity. Regarding Fig. 20, the four different capacity for each number of logo pixels investigates the satisfactory results in term of benchmarking. Hence, the amount of secret bits control the PSNR due to the increase of the secret bits inside the hosting text which will lead to increase the distortion and vice versa (inverse relationship).

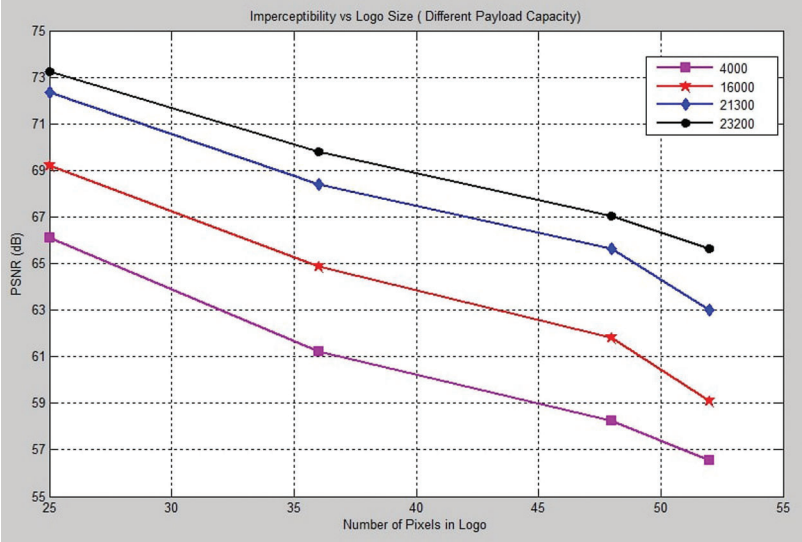

Figure 20: Imperceptibility *vs.* logo size for different payload capacity

According to the results which have been investigated in Tab. 3, the greater the number of the hidden bits, the less transparency, but with very few differences. For example, Logo 52 is higher than logo 25 because less coefficients are modified. The imperceptibility of the proposed vowels strategy is about 71 dB for the case of 4 Kb payload capacity which is considered to be a high PSNR value.

Regarding the results of the proposed method, it has been achieved a good imperceptibility (PSNR) compared with the existing state of art methods. The promising results have been accomplished due to eliminating the disadvantages and constraints of the existing methods through implementing the vowels and the reversing technique as well. In terms of the proposed strategy

benchmarking, Tab. 4 shows the comparative analysis of existing methods with the achieved proposed method outcomes.

Table 3: Summarizes the PSNR results to the logo size and with different payload capacity

Watermark/byres	Size of text file/kb	Imperceptibility PSNR/db	Embedding Ratio/ER/%
25	4	66.11	9.09
36	16	69.21	9.11
48	21.3	70.33	9.21
52	23.2	71.24	9.24

Table 4: Benchmarking of the proposed method with the relevant state of artworks

Methods	Imperceptibility and Security Evaluation (PSNR)
[36]	62.46 dB
[37]	70.88 dB
Proposed Method	72.33dB

Tab. 4, can illustrate the throughput of the proposed scheme while the comparison should be on the same criteria in case of amount of hosting file (cover file) and payload embedding (capacity) so we tried to a marriage of these criteria to get the same foundation of comparison. Big gap between the capacity of payload and size of cover text give different results in term of both ER and PSNR as shown in Tab. 3.

4 Conclusion

This paper proposes a new approach for improving the embedding capacity and impressibility. The proposed strategy has been achieved via implementing the text watermarking method based on vowels with three different types of characters (ا,و,ي) in Quran text (which is the most frequently used in the Arabic language) and redundant of these letters in the word. In this work, using the reversing technique were chosen to be added to the kashida after each vowels. The existence of these vowels is used to hide bit "1", in kashida and the absence of Kashida is used to hide bit "0". The experimental results show that the proposed approach achieves an embedding ration about 90.05%. Therefore, the promising results obtain a high impressibility (PSNR) about 72.33 dB and outperform the previous state of artwork in terms of performance evaluation.

The future direction of this work will be focusing on the strengthening of the verification phase by working on the boundaries of the proposed approach and expanding it to resolve more complex patterns. There are still a lot of issues that need to be addressed. First, the availability of digital Quran in different patterns is another pressing search problem. It would be interesting to expand the proposed approach to validate the other patterns as well. Second, it will be interesting to work on improving security and evaluating its accuracy in large data sets. Furthermore, our immediate goal is to make this web-based system as much as the public and extend the platform based on mobile users.

Acknowledgement: The authors would like to thank those who contributed to the article and who support them from Universiti Teknologi Malaysia (UTM) for their educational.

Funding Statement: The authors extend them appreciation to the Ministry of Higher Education (MOHE) and Universiti Teknologi Malaysia (UTM) for their educational and financial support. This work is conducted at Razak Faculty of Technology and Informatics, under cyber physical systems research group and funded by MOHE (FRGS: R.K130000.7856.5F026), Received by Nilam Nur Amir Sjarif.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] A. Alwan, M. Shahidan, N. Amir, M. Hashim and M. S. Mohd, "A review and open issues of diverse text watermarking techniques in spatial domain," *Journal of Theoretical and Applied Information Technology*, vol. 96, pp. 5819–5840, 2018.

[2] M. Hashim, M. S. Rahim and A. A. Alwan, "A review and open issues of multifarious image steganography techniques in spatial domain," *Journal of Theoretical & Applied Information Technology*, vol. 96, no. 4, pp. 1–22, 2018.

[3] S. M. Al-Nofaie and A. A. Gutub, "Utilizing pseudo-spaces to improve Arabic text steganography for multimedia data communications," *Multimedia Tools and Applications*, vol. 79, no. 2, pp. 19–67, 2020.

[4] M. M. Hashim, M. S. Rahim, F. A. Johi, M. S. Taha, A. A. Alwan *et al.,* "An extensive analysis and conduct comparative based on statistical attach of LSB substitution and LSB matching," *International Journal of Engineering & Technology*, vol. 7, no. 4, pp. 4008–4023, 2018.

[5] M. Hashim, M. S. Rahim and A. A. Alwan, "A review and open issues of multifarious image steganography techniques in spatial domain," *Journal of Theoretical & Applied Information Technology*, vol. 96, no. 4, pp. 1–22, 2018.

[6] S. Hakak, A. Kamsin, P. Shivakumara, M. Y. Idris, G. A. Gilkar *et al.,* "A new split based searching for exact pattern matching for natural texts," *PLoS One*, vol. 13, no. 7, pp. 1–13, 2018.

[7] A. Taha, A. S. Hammad and M. M. Selim, "A high capacity algorithm for information hiding in Arabic text," *Journal of King Saud University-Computer and Information Sciences*, vol. 32, no. 6, pp. 658–665, 2018.

[8] S. A. Salloum, A. Q. Al-Hamad, M. Al-Emran and K. Shaalan, "A survey of Arabic text mining," in *Intelligent Natural Language Processing: Trends and Applications*. Cham: Springer, pp. 417–431, 2018.

[9] Q. Dupont, "Social and technical opportunities and risks of cryptocurrencies and blockchains," *Committee on Science, Technology, and Law, National Academies of Sciences Engineering, and Medicin*, vol. 9, pp. 1–7, 2018.

[10] A. Odeh and K. Elleithy, "Steganography in Arabic text using zero width and kashidha letters," *International Journal of Computer Science & Information Technology*, vol. 4, no. 3, pp. 1–11, 2012.

[11] L. Tan, K. Hu, X. Zhou, R. Chen and W. Jiang, "Print-scan invariant text image watermarking for hardcopy document authentication," *Multimedia Tools and Applications*, vol. 78, no. 10, pp. 13189–13211, 2019.

[12] S. k. Prasad , J. Rachna, O. I. Khalaf and D. N. Le, "Map matching algorithm: Real time location tracking for smart security application," *Telecommunications and Radio Engineering (English translation of Elektrosvyaz and Radiotekhnika)*, vol. 79, no. 13, pp. 1189–1203, 2020.

[13] A. A. A. Gutub and W. Al-Alwani, "Improved method of Arabic text steganography using the extension 'Kashida' character," *Bahria University Journal of Information & Communication Technologies*, vol. 3, no. 1, pp. 1–10, 2010.

[14] Y. M. Alginahi, N. K. Muhammad and O. Tayan, "An enhanced kashida-based watermarking approach for Arabic text-documents," in *Int. Conf. on Electronics, Computer and Computation*, Ankara, Turkey, IEEE, pp. 301–304, 2013.

[15] M. H. Shirali-Shahreza and M. Shirali-Shahreza, "Arabic/Persian text steganography utilizing similar letters with different codes," *Arabian Journal for Science and Engineering*, vol. 35, no. 1, pp. 1–10, 2010.

[16] A. Ditta, C. Yongquan, M. Azeem, K. G. Rana, H. Yu *et al.,* "Information hiding Arabic text steganography by using unicode characters to hide secret data," *International Journal of Electronic Security and Digital Forensics*, vol. 10, no. 1, pp. 61–78, 2018.

[17] A. A. Mohamed, "An improved algorithm for information hiding based on features of Arabic text: A unicode approach," *Egyptian Informatics Journal*, vol. 15, no. 2, pp. 79–87, 2014.

[18] Y. M. Alginahi, N. K. Muhammad 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. 1–12, 2014.

[19] A. A. Shaker, F. Ridzuan and S. A. Pitchay, "Text steganography using extensions kashida based on the moon and sun letters concept," *International Journal of Advanced Computer Science and Application*, vol. 8, no. 8, pp. 286–290, 2017.

[20] S. Al-Nofaie, M. Fattani and A. Gutub, "Capacity improved Arabic text steganography technique utilizing 'Kashida' with whitespaces," in *3rd Int. Conf. on Mathematical Sciences and Computer Engineering*, Kuala Lumpur, Malaysia, pp. 38–44, 2016.

[21] H. El-Fiqi, E. Petraki and H. A. Abbass, "Network motifs for translator stylometry identification," *PLoS One*, vol. 14, no. 2, pp. 1–33, 2019.

[22] J. Dong, T. Jiang, C. Li, A. Anwar and Y. Yang, "A compromise Arabic-Kazakh coded character processing method based on the opentype font format," *Computer Standards & Interfaces*, vol. 55, no. 1, pp. 1–7, 2018.

[23] S. Al-Nofaie, A. Gutub and M. Al-Ghamdi, "Enhancing Arabic text steganography for personal usage utilizing pseudo-spaces," *Journal of King Saud University-Computer and Information Sciences*, vol. 3, no. 1, pp. 1–12, 2019.

[24] A. A. Alwan, M. S. Abdulah and N. N. Sjarif, "A survey on combined various data hiding techniques," *Open International Journal of Informatics*, vol. 7, no. 2, pp. 31–44, 2019.

[25] A. R. A. Alkhafaji, N. N. Sjarif and M. A. Shahidan, "A review of comparative spatial domain techniques of steganography and watermarking," *Journal of Technology Reports of Kansai University*, vol. 62, no. 5, pp. 1–17, 2020.

[26] Y. M. Alginahi, N. K. Muhammad and O. Tayan, "An enhanced Kashida-based watermarking approach for Arabic text-documents," in *Int. Conf. on Electronics, Computer and Computation*, Antara, Turkey, IEEE, pp. 1–14, 2013.

[27] O. Tayan, N. K. Muhammad and Y. M. Alginahi, "A hybrid digital-signature and zero-watermarking approach for authentication and protection of sensitive electronic documents," *Scientific World Journal*, vol. 2014, no. 1, pp. 1–15, 2014.

[28] A. R. A. Alkhafaji, N. N. Sjarif and M. A. Shahidan, "Digital text watermarking techniques classification and open research challenges: A review," *Journal of Technology Reports of Kansai University*, vol. 62, no. 5, pp. 1–22, 2020.

[29] M. S. Memon and A. Shah, "A novel text steganography technique to Arabic language using reverse Fat5Th5Ta," *Pakistan Journal of Engineering, Technology & Science*, vol. 1, no. 2, pp. 1–8, 2015.

[30] E. N. Alkhanak, S. R. Khan, A. Verbraeck and H. V. Lint, "A conceptual framework supporting pattern design selection for scientific workflow applications in cloud computing," in *Closer*, vol. 10, pp. 229–236, 2020.

[31] S. Vennelakanti and S. Saravanan, "Design and analysis of low power memory built in self-test architecture for soc based design," *Indian Journal of Science and Technology*, vol. 8, no. 14, pp. 1–14, 2015.

[32] S. Hakak, A. Kamsin, P. Shivakumara and M. Y. Idris, "Partition-based pattern matching approach for efficient retrieval of Arabic text," *Malaysian Journal of Computer Science*, vol. 31, no. 3, pp. 200–209, 2018.

[33] H. Z. Zadeh, "Tanzil Quran Navigator," 2013. [Online]. Available: http://tanzil.net/#1:1.

[34] Z. Yang, Y. Jin, Y. Huang, Y. Zhang, H. Li *et al.,* "Automatically generate steganographic text based on Markov model and Huffman coding," *arXiv preprint arXiv*, vol. 1811.04720, pp. 1–10, 2018.

[35] H. J. Shiu, B. S. Lin, B. S. Lin, P. Y. Huang, C. H. Huang *et al.,* "Data hiding on social media communications using text steganography," in *Int. Conf. on Risks and Security of Internet and Systems*, Cham. Springer, pp. 217–224, 2017.

[36] R. Gupta and V. Sharma, "A vision on text steganography with proper investigation report to identify the associated problem," *International Journal of Computer Trends and Technology*, vol. 54, no. 1, pp. 35–39, 2017.

[37] N. A. S. Al-maweri, W. A. Adnan, A. R. Ramli, K. Samsudin, S. M. S. A. A. Rahman *et al.,* "Robust digital text watermarking algorithm based on unicode extended characters," *Indian Journal of Science and Technology*, vol. 9, no. 48, pp. 1–14, 2016.