International Journal of Recent Engineering Science
ISSN: 2349-7157/ https://doi.org/10.14445/23497157/IJRES-V12I3P107

Volume 12 Issue 3, 58-63, May-Jun 2025

Original Article

An Efficient Text Hiding Algorithm Based on ASCII Codes

Musah Yahaya[1], Edem Kwedzo Bankas[2], Salamudeen Alhassan[3]

[1,2]*Department of Computer Science, School of Computing and Information Sciences*
C. K. Tedam University of Technology and Applied Sciences, Navrongo, Ghana.
[3]*Department of Computer Science, University for Development Studies, Tamale, Ghana.*

[1]*Corresponding Author : musahyahaya@gmail.com*

Received: 01 April 2025 Revised: 08 May 2025 Accepted: 26 May 2025 Published: 16 June 2025

Abstract *- Steganography is a technique for concealing information within innocuous media to prevent detection. Unlike encryption, which secures content but reveals its presence, steganography obscures content and existence. This paper introduces a novel text-based steganographic technique using ASCII encoding and a dual-function approach for embedding and extraction. The proposed method utilizes a three-key embedding mechanism—initial position, stepping distance, and a constant—to generate a robust key space. Experimental evaluations demonstrate that the technique offers enhanced security, high embedding capacity, and improved performance in terms of time complexity and resource usage compared to existing approaches. These results indicate the method's suitability for secure and efficient covert communication.*

Keywords *- Steganography, Cryptography, ASCII, Embedding, Encryption, Decryption.*

1. Introduction

The discovery and application of steganographic methods in the current technological system are to protect and keep information and data safe from theft. Information handlers are concerned about data and information confidentiality, integrity, and authentication. The recent rise in technology has again rendered encryption for data protection fierce. Emerging computer software can break through most encryption keys used to protect information. Breaking through encrypted information or data has become easier for malicious computer users. Hence, data is most secure when adversaries are unaware of its existence [1].

Steganography and cryptography are two primary methodologies used to secure data. Steganography conceals a message within another to avoid arousing suspicion, ensuring that only the intended recipient can detect its presence. Steganography devises the use of the cover document to hide secret messages. This is done by sometimes compressing and encrypting the text of a secret message before embedding it into the cover document to increase security. On the other hand, cryptography is making a message incomprehensible to unauthorized individuals [2, 3].

Previous studies have introduced various steganographic techniques for data protection; however, many of these methods suffer from limitations such as low embedding capacity, limited key space, high processing time, and vulnerability to detection or deletion by standard word processing applications [5, 6, 7, 8, 9].

This paper proposes a robust steganographic algorithm designed to securely embed information within the cover text, ensuring resistance to detection and deletion by word processing software such as Microsoft Word. Unlike existing systems, the proposed method achieves higher embedding capacity while maintaining stealth and reliability. Central to the approach is two steganographic functions—embedding and extraction—executed using ASCII character values. These functions operate efficiently within natural language text, allowing for seamless integration into commonly used digital documents. The model incorporates cryptographic keys, including an initial position, stepping distance, and a constant, to enhance message confidentiality and safeguard against brute-force attacks. The increased key space and use of ASCII encoding improve security and contribute to the model's high embedding capacity, making it a viable solution for secure textual communication in sensitive environments.

2. Review of Related Works

Text steganography is one of the techniques employed to hide information and data by inserting it into a text file. According to [4], text steganography is the most challenging form due to the limited redundant information available in text compared to images or audio. [5] argued that techniques such as format modification, word substitution, or readable

text generation are insufficient for achieving strong and secure text steganography. This is because reformatting the text may remove the embedded information. Additionally, text can exist in various formats, such as HTML or PDF, and converting from one format to another may destroy the hidden data. This explains why earlier techniques faced significant challenges, highlighting the need for new principles to address the limitations of text steganography. Consequently, extensive research has been conducted, with various researchers proposing different approaches for concealing information using steganographic methods. However, most of these approaches focus on enhancing security, often through hybrid methods, without adequately considering the embedding capacity of their models.

Three novel approaches to text steganography were introduced by [6]: missing letter puzzles, wordlists, and hiding data within paragraphs. The author employs a one-time pad security system comprising four stages in the steganographic process: the encipher function, embedding/hide function, seek function, and decipher function. Among the three methods, HDPara has been identified as the most efficient, capable of concealing information without arousing suspicion. This technique uses the ciphertext's binary values for embedding, using the start and end letters of words in the cover text. Words in the cover text that share the same starting and ending letters are skipped. A binary '0' uses start letters, while a '1' is embedded using end letters. These characters collectively form the algorithm's stego file output.

Another research study by [7] proposed a new information-hiding scheme in text steganography, the WhiteSteg method. This approach is considered a hybrid technique, combining inter-word and inter-paragraph spacing. The model manipulates white spaces between words and paragraphs to embed hidden messages. It dynamically generates a suitable cover text for the user to conceal a secret message. The system selects the cover text with the most appropriate capacity based on the message length to be hidden.

[8] proposed a steganographic data-hiding technique based on a syntactic method, also known as linguistic steganography. This approach involves generating natural language text to conceal information. The model is both comprehensive and user-friendly, employing the principles of pure steganographic protocols. Secret messages are embedded in the cover text by manipulating the initial letters of words, either through bold formatting or color changes. Although the hidden messages can be effectively embedded and extracted, the visual alterations (bolding or coloring of initial characters) may arouse suspicion from eavesdroppers.

[9] designed a high-capacity text steganography scheme based on Huffman compression and color coding. This model has proven to be highly effective for concealing information. It incorporates three encryption principles: Move-to-Front (MTF) encoding, Huffman compression, and color coding. First, MTF is applied to encode the secret message and increase content correlation. Next, Huffman compression reduces the message size. Finally, color coding maps the secret data bits to designated colors. To embed the secret message into the cover text, the number of non-space characters in the cover text is counted, and an equivalent number of bits is extracted from the bit stream. These bits are used for color coding within the cover text, while the remaining bits are embedded into forwarded email addresses for transmission to the receiver.

As knowledge advances, there is a need to develop a comprehensive approach that addresses security and capacity challenges, thereby enhancing the core principles of steganography. The proposed model arises from the identification of weaknesses in existing approaches.

3. Proposed Model

Our comprehensive steganography method proposes a practical algorithm based on the ASCII code system for information embedding and extraction. This model demonstrates strong security features through a three-key system and offers high embedding capacity. The algorithm ensures that data can be securely hidden within the cover text without detection or removal by any software.

3.1. Proposed Steganographic Functions

To achieve the objectives of this project, the model introduces new steganographic functions for data embedding and extraction. These two functions ensure that information can be securely hidden within and accurately extracted from a cover text without detection. They include:

3.1.1. Embedding Function

The embedding function (Equation 1) conceals secret messages within a cover text. Given a cover text C, a secret message S and a constant K, the stego key E is derived using Equation 1.

$$E = (C_{(i+d)} - S_i) + K \quad (1)$$

Here, i and d represent the initial point and the embedding step distance, respectively, with the condition that i + d = l, where l is the length of S.

3.1.2. Extraction Function

The extraction function extracts secret messages from the cover text. Given the stego key E and cover text C and constant K, our proposed extraction function is defined in Equation 2.

$$S = \left(C_{(i+d)} - E_i\right) + K \quad (2)$$

3.1.3. Components of the Model

The operation of the proposed model is based on the following elements.

1. Cover Message (C): This is the host text used to carry the secret message. It appears harmless and inconspicuous, making it suitable for steganographic purposes. It is also referred to as the cover file.
2. Secret Message (S): This refers to the actual information to be concealed. Text, images, audio, video, or other data must remain hidden from unauthorized parties.
3. Stego File (E): The stego-object results from embedding the secret message into the cover message using a set of embedding keys.
4. Initial position (i): The starting index or location in the cover text where the embedding process begins.
5. Stepping Distance (d) defines the interval or gap between two consecutive embedding positions within the cover text.
6. Constant (K): A fixed numerical value introduced to enhance the security of the embedding keys
7. Length of Position of the Secret Message: This is determined within the cover text using the values of i, d, and K as part of the key generation or embedding logic.

3.1.4. Embedding Algorithm

The embedding algorithm conceals secret messages in cover text using the embedding function. This algorithm is defined as:

Input: C, S, K
Output: E

1. Read cover text C
2. Read secret message S
3. Read constant K
4. Read stepping/embedding distance d
5. For each character of S
   a. Obtain its ASCII
   b. Compute the next d
   c. Obtain the ASCII of the next character C at point $i + d$
   d. Compute E using Equation (1)
6. Transmit E

The flowchart for the proposed embedding algorithm is shown in Figure 1.

3.1.5. Extraction Algorithm

Our proposed extraction algorithm is defined below:

Input: C, E, K, d
Output: S

1. Read cover text C
2. Read secret message E
3. Read constant K
4. Read stepping/embedding distance d
5. For each value of E
   a. Compute the next d
   b. Obtain the ASCII of the next character C at point d
   c. Compute the ASCII S using Equation (2)
   d. Convert S to the character equivalent
6. Transmit S

The flowchart of the extraction algorithm is shown in Figure 2.

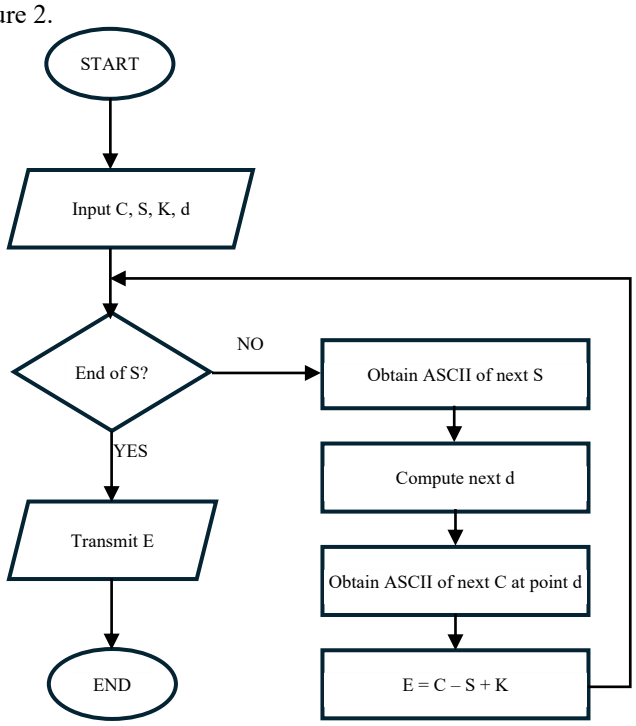

Fig. 1 Flowchart for embedding algorithm

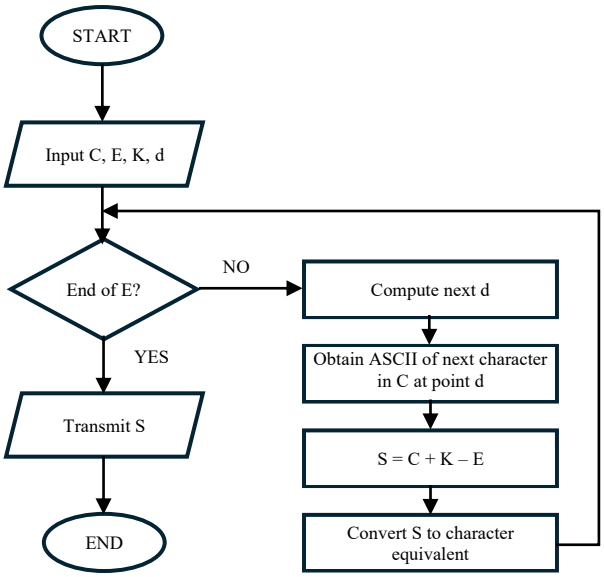

Fig. 2 Flowcharts for the proposed model for extraction

4. Results and Discussion

Various simulations were carried out to measure and compare the performance of the proposed algorithms against existing algorithms based on the following criteria:

- Security
- Embedding speed
- Extraction speed
- Coding efficiency
- Time complexity

4.1. Security

In terms of security, the proposed model focuses on two key aspects:

- Key sensitivity
- Keyspace

A robust steganographic scheme must generate distinctly different outputs for each unique key within the key space [11]. In other words, slight variations in the steganographic key should lead to entirely different results during the embedding and extraction processes.

To demonstrate this, the secret message *"Demonstration"* was embedded into the cover message *"The government embarks on free distribution of uniforms"* using minor variations in the steganographic keys. As shown in Table 1, the results confirm that the embedding algorithm is susceptible to key changes.

Table 1. Key sensitivity analysis of embedding algorithm

Embedding Keys	Stego keys
K = 3, d = 2, i = 2	36, 5, 12, 6, 2, -2, -81, -2, 3, -6, 9, -76, 7
K = 3, d =2, i =3	-33,13, 5, 2, -6, 4, -12, -13, 20, -81, 8, -6, -6

Furthermore, the recovered messages presented in Table 2 indicate that the model is susceptible to the extraction keys, thereby confirming the strong security capability of the proposed model.

Table 2. Illustration of extraction keys sensitivity

Extraction keys	Recovered message
K = 3, d = 2, i = 1	G-fbojÈjb{½b
K = 3, d = 2, i = 3	m\kfy¹gr)hµa
K = 3, d = 2, i = 4	Ftijo%ÁfkxÁa

4.2. Keyspace

A key space refers to all possible key values a given encryption or embedding algorithm allows. The key space must be at least 280 bits to resist brute-force attacks effectively. To ensure the security of the proposed model, three (3) keys are employed: the constant K, the stepping distance d, and the initial position i, which collectively provide a strong defense mechanism.

- Constant *K*: Assigned an integer range of {1,2,3,...,64}, giving it a binary representation of 2^{64}. This results in a key length of 64 bits and a key space 2^{64} = 18,446,744,073,709,551,616.
- Stepping Distance *d*: Defined within the integer range {1, 2, 3, 4… 10}, represented in binary as 2^{10} bits. His corresponds to a key length of 10 bits and a key space of 2^{10} = 1,024 bits. However, the value of d must not exceed the length of the cover text; therefore, the cover text must always be greater than or equal to d.
- The initial position (i) is also assigned an integer range of {1,2,3,...,10}, corresponding to a binary representation of 2^{10}. This gives it a key length of 10 bits and a key space of 2^{10} = 1,024.

The combined key space of the proposed algorithm is given by $K \times d \times i = 2^{64} \times 2^{10} \times 2^{10}$. This corresponds to a key length of 84 bits, sufficiently large to resist brute-force attacks.

4.3. Embedding and Extraction Times

The average embedding and extraction times were measured to evaluate the proposed algorithms' performance in terms of speed. Multiple simulations were conducted using secret messages of varying lengths, applying the proposed algorithm and the HDPara method introduced in [6] under identical conditions. The results of these simulations are summarized in Table 3. The proposed algorithm outperforms HDPara by 25 ms per character in terms of average embedding time. Similarly, it is 15 ms faster in extracting a single character than HDPara.

Additionally, the graphical representations in Figure 3 and Figure 4 illustrate the growth of time consumption for both embedding and extraction processes. These figures visually compare the performance of the proposed algorithm with the existing algorithm, highlighting the improved efficiency of the proposed method.

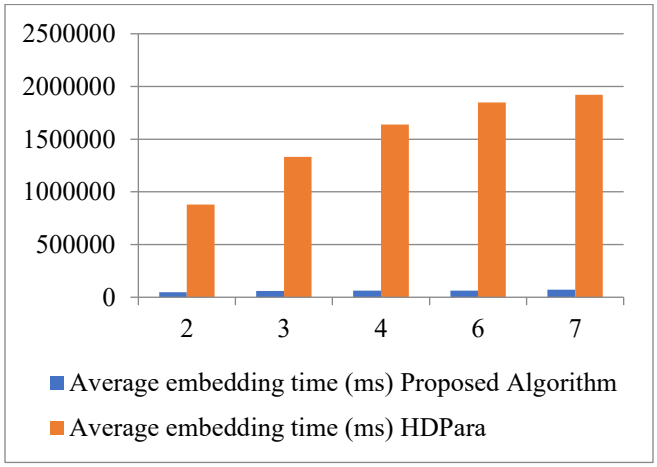

Fig. 3 Illustration of time consumption growth for the proposed and existing algorithms during the embedding process

Table 3. Average embedding and extraction times for various lengths of secret messages using a fixed length of cover text

Length of secret Message (number of characters)	Average embedding time (ms)		Average extraction time (ms)	
	Proposed Algorithm	HDPara	Proposed Algorithm	HDPara
2	47590	881030	178110	1523000
3	60340	1332360	181920	1603900
4	61419	1638780	244740	2394300
6	64339	1848800	254890	6298000
7	71640	1919700	670820	11041000

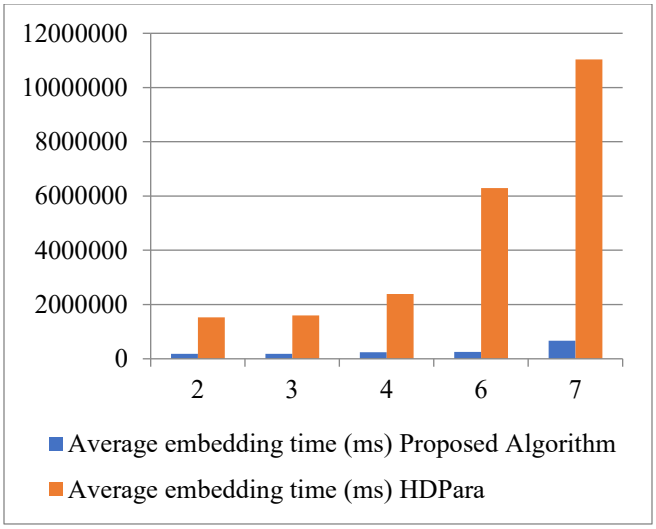

Fig. 4 Illustration of time consumption growth for the proposed and existing algorithms during the extraction process

4.4. Coding Efficiency

Code efficiency is closely tied to algorithmic performance and the runtime execution speed of the software. It plays a critical role in achieving high system performance by minimizing resource usage and execution time while maintaining minimal risk to the operating environment. HDPara employs nested loops in its implementation, resulting in increased execution time. In contrast, the proposed algorithm embeds characters directly onto the cover text on a character-by-character basis, significantly reducing the embedding time. HDPara, however, embeds each character by processing its binary equivalent, comprising 8 bits, thus increasing embedding time, as all bits are handled individually according to the algorithm's structure.

4.5. Time Complexity

The proposed algorithm utilizes a single loop that iterates through each character of the secret message, thereby performing a constant-time operation per character. Given that there are n characters in the secret message, the total time complexity of the loop is O(n). The space required to store the cover text (C), secret message (S), constant (K), stepping distance (d), and any temporary variables used during computation is fixed. As these variables do not scale with input size, the space complexity is considered O(1).

In contrast, the algorithms proposed in HDPara have a time complexity of $O(n^2)$ and a space complexity of O(n), indicating higher computational overhead. These comparisons demonstrate that the proposed algorithm is more efficient in time and space.

4.6. Summary Performance Comparison

The proposed model outperforms HDPara in key performance areas, including embedding time, extraction time, and space complexity. Specifically, the proposed model is 25 times faster in embedding and 15 times faster in extraction when compared to HDPara. Additionally, it demonstrates a lower space complexity, making it more efficient regarding memory usage. A summary of the performance comparison between the proposed model and HDPara is presented in Table 4.

Table 4. Comparison of HDPara and proposed model

Performance Parameter	Model	
	HDPara	Proposed model
Average embedding time	1524134.0	61065.6
Average Extraction time	4572040.0	306096.0
Time complexity	$O(n^2)$	O(n)
Space complexity	O(n)	O (1)
Key sensitivity	Sensitive	Sensitive

5. Conclusion

This paper has proposed an efficient steganographic scheme based on ASCII codes for concealing secret text messages. Two core functions, embedding and extraction, were developed to support the hiding and retrieval of information. The performance and security evaluations demonstrate the effectiveness of the proposed method. Specifically, the scheme:

- Successfully conceals secret text messages within cover files without detection or deletion by word processing applications.
- It provides a significantly higher embedding capacity compared to existing techniques.
- It employs a robust key combination mechanism capable of resisting brute force attacks.

- It achieves favorable time complexity and execution speeds, making it suitable for real-time applications.
- Exhibits high sensitivity to changes in steganographic keys, where even slight modifications to the key set yield completely different embedding and extraction results.

Recommendation

Based on this study's experimental and theoretical analyses, the proposed steganographic scheme demonstrates significant advantages over existing methods. With its improved embedding capacity, reduced execution times, robust key combination system, and high sensitivity to key alterations, the scheme offers enhanced security and performance. Consequently, it is highly recommended for adoption by individuals and organizations that require secure and covert communication of textual data. The scheme's ability to embed secret messages in cover files without detection by standard word processing applications further underscores its practical utility in confidential communications.

For future work, researchers are encouraged to explore the hardware implementation of the proposed technique. Such an implementation could further enhance the performance and broaden the method's applicability, particularly in real-time and embedded systems where speed and security are critical.

Funding Statement

This publication is a self-funded article and has not been funded by any organisation.

Acknowledgments

I extend my sincere gratitude to the Almighty Allah for His continuous protection, guidance, and the gift of good health throughout this research and the preparation of this publication.

I am also profoundly grateful to Author 2 and 3 for their unwavering support, patience, and invaluable guidance throughout my study. Their insightful feedback, constant encouragement, and thoughtful advice played a crucial role in helping me navigate the challenges encountered during this work. Their mentorship has been instrumental in the successful completion of this research.

References

[1] Robert Lockwood, and Kevin Curran, "Text Based Steganography," *International Journal of Information Privacy, Security and Integrity*, vol. 3, no. 2, pp. 134-153, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[2] Indradip Banerjee, Souvik Bhattacharyya, and Goutam Sanyal, "An Approach of Quantum Steganography through Special SSCE Code," *International Journal of Computer and Information Engineering*, 2011. [CrossRef] [Google Scholar] [Publisher Link]

[3] William August Kotas, "A Brief History of Cryptography," *Chancellor's Honors Program Projects*, pp. 1-68, 2000. [Google Scholar] [Publisher Link]

[4] Souvik Bhattacharyya, Indradip Banerjee, and Gautam Sanyal, "Design and Implementation of Secure Text Based Steganography Model," *Proceedings of the 2010 International Conference on Security & Management*, 2010. [Google Scholar]

[5] Prapti Sharma, and Shweta Goyal, "Steganography Techniques," *International Journal of Engineering Research & Technology*, vol. 2, no. 4, pp. 2425-2428, 2013. [Google Scholar] [Publisher Link]

[6] Naga Ranjith Kumar Kesa, "Steganography a Data Hiding Technique," *Culminating Projects in Information Assurance*, pp. 1-91, 2018. [Google Scholar] [Publisher Link]

[7] M. Agarwal, "Text Steganographic Approaches: A Comparison," *International Journal of Network Security & Its Applications*, vol. 5, no.1, pp. 91-106, 2013. [CrossRef] [Google Scholar] [Publisher Link]

[8] L.Y. Por, and B. Delina, "Information Hiding: A New Approach in Text Steganography," *7th WSEAS International Conference on Applied Computer & Applied Computational Science*, pp. 689-695, 2008. [Google Scholar]

[9] Aruna Malik, Geeta Sikka, and Harsh K. Verma, "A High Capacity Text Steganography Scheme Based on Huffman Compression and Color Coding," *Journal of Information & Optimization Sciences*, vol. 38, no. 5, pp. 647-664, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[10] Fabien A.P. Petitcolas, and Stefan Katzenbeisser, "Information Hiding Techniques for Steganography and Digital Watermaking," *EDPACS*, vol. 28, no. 6, pp. 1-2, 2000. [CrossRef] [Google Scholar] [Publisher Link]

[11] Riguang Lin, and Sheng Li, "An Image Encryption Scheme Based on Lorenz Hyperchaotic System and RSA Algorithm," *Security and Communication Networks*, vol. 2021, no. 1, pp. 1-18, 2021. [CrossRef] [Google Scholar] [Publisher Link]