A Novel Technique of Information Hiding of a Digital Image using LSB Algorithm

Md. Jashim Uddin, Khandaker Takdir Ahmed, S.M. Abdur Rahim, Md. Abdul Kadir, Md. Abdullah-al-Imran

Abstract— Steganography is the art of hiding information in a cover medium in such a method that the existence of any communication itself is undetectable. It can be applied in open systems such as the internet. We present an LSB Algorithm approach text stenography aimed at increasing the robustness and capacity of hidden data. The used cover text is a set of random numbers. First, the secret text is encrypted and then converted into its ASCII form. The genetic algorithm is then applied to the cover text obtained from a set of randomly generated numbers to embed the secret message into the text data. The generated cover text is dependent on the length of the secret message. Once optimal results have been reached the embedding process begins to produce a steganography text. To get the original secret message, an extraction algorithm is applied. The method shows that the proposed approach satisfies security, robustness, and hiding capacity requirements. By applying this algorithm we have developed an application that would help users to effectively hide the data.

Index Terms— Steganography, cryptography, secret key, LSB algorithm, embedding, extraction, Message hiding.

1 Introduction

Steganography is a technique that establishes a covered information channel in point-to-point connections by embedding or hiding data inside a cover medium in such a way that the existence of any communication itself is undetectable[1]. Steganography technologies are a very important part of the future of Internet security and privacy on open systems such as the Internet because important data can be hidden inside a medium so that only the parties intended to get the message knows that a secret message exists. The mostly used medium includes an image, video, audio, or text [2].

Image Steganography: Straight message insertion may encode every bit of information in the image or selectively embed the message in "noisy" areas that draw less attention, to hide information—those areas where there is a great commitment of natural color variation. Text Steganography: Is a method of using written natural language to conceal a secret message. Hiding information in the form of plain text can be done in many different ways. One of the techniques this type of steganography includes the modification of the layout of a text, rules like using every character, or the altering of the amount of white space after lines or between words [2],[3]. Video Steganography: Video files are generally a gathering of images and sounds, so most of the presented techniques on images and audio can be applied to video files.

The advantages of video are the large amount of data that can be hidden inside and the fact that it is a moving stream of images and sounds. Therefore, any small but noticeable distort things might go by unobserved by humans because of the continuous flow of information.

The advantage of image steganography over other media like video, audio, and text is its smaller memory occupation and simpler communication. It is the most secure way to transmit confidential information. In this, we can transmit our secret message without detecting. That means we can send and receive our secret message but no one will detect it. Many intelligence organization uses this technique for transmitting its secret message. For this reason, I have chosen this image steganography technique [4].

With the worldwide need for secure data transmission over the internet and the failures on most of the currently used algorithms, there was a need to look at the capabilities of LSB Algorithm as applied to image steganography. This is because the LSB algorithm approach tends to focus on the long-hidden messages, higher text quality, and increase in the robustness. LSB algorithm helps to keep hide text or information when transmitted [5]. This project aimed to develop a tool by use of LSB algorithm technique on the cover image in order to investigate how to increase the overall rate of hidden data without affecting the quality of the cover image and with a reduced probability of message detection.

2 Methodology

This project used the objector use case approach which was proposed by I. Jacobson (1994). This involved identification of functional requirements from which use case artifacts were developed, after which, the dynamic and static behavior of the system were analyzed and modeled. The modeling of static behaviors was done through the identification of objects and classes which were represented using Unified Modeling Language (UML) diagrams. The dynamic aspects of the sys-

- *Md. Jashim Uddin is an Assistant Professor of the Dept. of Information and Communication Technology at Islamic University, Kushtia, Bangladesh. Ph: +8801716555094, E-mail: jashim.iu@gmail.com.*
- *Khandaker Takdir Ahmed is an Assistant Professor of the Dept. of Information and Communication Technology at Islamic University, Kushtia, Bangladesh. Ph: +88001719252243, E-mail: takdir.iu0607@gmail.com.*
- *S.M. Abdur Rahim is an Assistant Professor of the Dept. of Electrical and Electronic Engineering at Islamic University, Kushtia, Bangladesh. Ph: +88001715544555, Email; smrahimiu@gmail.com*
- *Md. Abdul Kadir pursuing a B.Sc. degree program in Information & Communication Technology at Islamic University, Kushtia, Bangladesh. Ph: +8801733996189, E-mail: abdulkadir.ice.iu@gmail.com.*
- *Md. Abdullah-Al-Imran pursuing a B.Sc. degree program in Information & Communication Technology at Islamic University, Kushtia, Bangladesh. Ph: +8801794517851, E- mail: imran136658@gmail.com.*

International Journal of Scientific & Engineering Research, Volume 11, Issue 6, June-2020
ISSN 2229-5518

tem were modeled using sequence, interactive, state diagrams, and collaboration diagrams. Thereafter, implementation was done, where the algorithm was implemented using the LSB algorithm approach.

2.1 System Analysis

This method is one of the main techniques and the primary idea to replace the LSB of the image with the bits of the message or to hide text files without degrading the quality of the property or resolution of the image. The LSB-based method becomes more challenging when we replace a few LSB bits of the cover object. Because it is very hard to differentiate between the cover- object and Steganography object [6]. For security reasons, we use the concept of steganography to transfer data through the internet safely and faster to the destination. This is all about making an application by using of Least Significant Bit (LSB) algorithm for hiding the data into an image. We are going to implement this through the Microsoft .NET framework using C# as a programming language [7].

2.2 Steganography Conceptual framework

Figure 1 below demonstrates the conceptual framework for image steganography process.

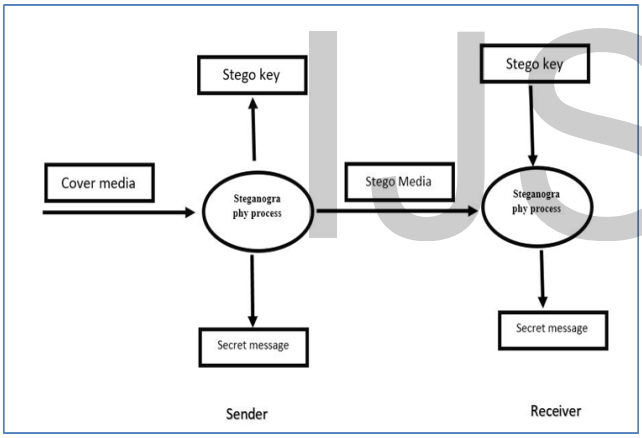

Fig. 1. A conceptual framework for image steganography process.

2.3 Required Elements

Image Steganography needs three elements which are as follows: 1. Cover Medium – It is an image in which we hide our information. 2. The Secret Message – It is the information that we hide inside our image. 3. The Stego-Key – This is the key that may or may not be needed to hide our information. It depends upon the algorithm.

2.4 Least Significant Bit (LSB) Algorithm

Least Significant Bits (LSB) is a very basic approach to embed your information in the cover image. This is the main steganography technique that replaces the bits of the message directly into the least significant bit plane of 3/13 Binary Representation Of TEXT the cover image. Changing the least significant bit does not mean changing the property of the cover image. It looks like both the original image and steganography-image because the amplitude of the changes is quite small. To implement this method, we require a proper cover image with a high resolution to hide our secret message inside an image. Because this technique uses the last bit of each pixel in the image. It is very important to use a lossless compression format because, in the transformations of a lossy compression algorithm, we lost the hidden messages. We can use a bit of each of the red, green, and blue color components by using a 24-bit color image, so we can store 3 bits in each pixel. For example, suppose our message is "TEXT" [7],[8]. The binary representation of each character of the message is given below. We consider the below grid as 3 pixels of a 24-bit color image, using 9 bytes of memory:

T	0	1	1	0	0	0	1	1
E	0	1	1	0	0	0	1	1
X	0	1	1	0	0	0	1	1
T	0	1	1	0	0	0	1	1

Fig. 2. The binary representation of TEXT

Below the binary representation of an image in which we are going to hide our message. We convert every single pixel into binary form which is an 8-bit combination. Now what we are going to do is going to take every single bit from the information grid and replace it with the last bit of each pixel of the cover image [9].

01011010	00101011	10101011	10101010	11101011	11010100	01000111	11111001
01011010	10101101	10010111	10101111	10101011	10100111	01010110	01011011
10110111	11111011	00101011	10010101	10101000	01010100	10101010	11010101
10100100	01011000	11011010	01010101	01001001	10110000	01000010	01010100

Fig. 3. Image Bits Before

Now you can see, the last bit of every pixel is replaced by each bit of information in the below image. Every blue and the red color number represents each bit of our message. Blue colored number means that message's bit is as same as the pixel's bit and blue color means both bits are different [10].

01011010	00101011	10101011	10101010	11101010	11010100	01000111	11111000
01011010	10101101	10010111	10101110	10101011	10100111	01010111	01011011
10110110	11111011	00101011	10010100	10101001	01010101	10101010	11010101
10100100	01011001	11011011	01010100	01001000	10110000	01000011	01010100

Fig. 4. Image Bits After

2.5 Image Steganography process using LSB algorithm

In this process hiding information keeps invisible. No one cannot see the secret message. Look at the following image. Can you see the difference? No. Because of the message in invisible. Though the second image contains a text or information.

International Journal of Scientific & Engineering Research, Volume 11, Issue 6, June-2020
ISSN 2229-5518

Fig. 5. Image before Hiding Message and after hiding a Message

Image Steganography is one of the most powerful techniques to transfer data securely with the help of images. We can easily spread an image over the WWW or in newsgroups with a secret message. The Least Significant Bit (LSB) is a very common method to hide a message inside an image [10].

Just for understanding, we take a very small image like 15 × 15 sizes. Now, look at the image. Is there any difference? Yes. The second image contains the message. That's why there is some change. The marked point containing the message. It is not the drawbacks of this technique. It is an example just for understanding. Because we don't use like the small image for transforming our confidential information [11].

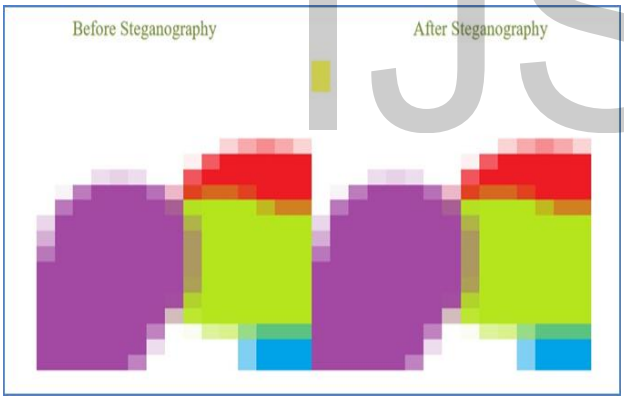

Fig. 6. A sample 15 × 15 size image

3 Proposed Algorithm

The algorithm that we have proposed in this system is an extension of the original LSB which is quite vulnerable. Instead of hiding the information in the least significant bits of the RGB components of a pixel, we in this algorithm would be hiding data as follows:

Let the information to be hidden in the word "ABC" ASCII code of A= 65 and the corresponding binary is 01000001. ASCII code of B= 66 and its binary is 01000010. ASCII code of C= 67 and its binary is 01000011[12].

Let the first pixel's RGB component be: -

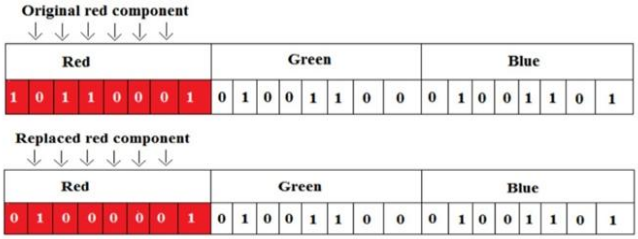

Fig. 6. Red component is replaced with the binary of 65 i.e. A

Let the second pixel's RGB component be: -

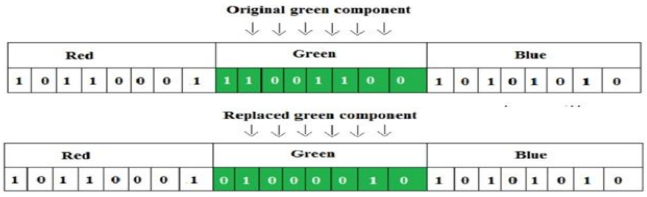

Fig. 7. The green component of the second pixel is replaced with the binary of 66 i.e. B

Let the Third pixel's RGB component be: -

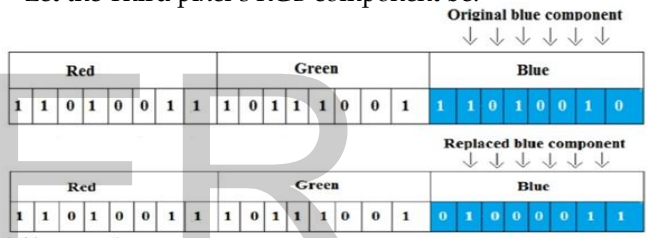

Fig. 8. The blue component of the third pixel is replaced with the binary of 67 i.e. C

The resulting steganography image that we are obtaining after the algorithm completes its execution, is distorted and is easy to detect, that some kind of alteration has been done to the image. So, to enhance the security of the secret message we would be covering the resulting steganography image with a new cover image, this is the first level of security. By just looking at the resulting image no one could be able to understand that something is hidden inside it. The new cover image can be the same or different from the original [12],[13]. To increase the storage capacity of the image, a compression algorithm has been used, we know that each element of an RGB pixel is represented with 8 bits. So, the maximum compression would be 8 bits/pixels, and the minimum would be 1 bit/pixel. The proposed steganographic algorithm comprises two techniques they are data hiding technique and data retrieving technique. The data hiding technique as the name suggests is used to hide secret information and key in the cover image, while the data retrieving method is used to retrieve the key and the hidden secret message from the steganography image. Therefore data is protected in the image without revealing to an unauthorized party [14].

International Journal of Scientific & Engineering Research, Volume 11, Issue 6, June-2020
ISSN 2229-5518

3.1 Proposed Algorithm embedding technique.

Inputs: - Text file, cover image 1, cover image 2, and secret key.
Output: - Stego image.
Step 1. Select a text file, convert it into binary form, and calculate the number of bits in it.
Step 2. Select a carrier image (cover image 1) for hiding purpose, find the number of pixels, convert it into RGB image, and calls the compression function.
Step 3. If bits calculated are compatible with the image resolution, then
Start sub iteration 1
Replace the red component of the first pixel with the first character.
Replace the green component of the second pixel with the second character.
Replace the blue component of the third pixel with the third character.
And repeat iterations until pixels get exhausted.

Stop sub iteration 1
Else

Repeat sub iteration 1
Finds the necessary compression ratio and perform sub iteration 2.

Sub iteration 2
Replace necessary bits as defined by the compression ratio in the immediate component of each pixel. Store the information about bits embedded in a binary address file.

Stop sub iteration 2
Step 4. Provide a security key as encryption completes.
Step 5. Select 2nd cover image to hide the distorted steganography image.
END

3.2 The proposed Algorithm extraction technique

Input: - Stego image and the secret key.
Output: - Secret text file.
Step 1. Start
Step 2. Browse the steganography image.
Step 3. Choose the folder in which you want to extract the hidden text file.
Step 4. Provide the necessary security key.
Step 5. Convert the binary file into a human-readable form.
Step 6. Stop

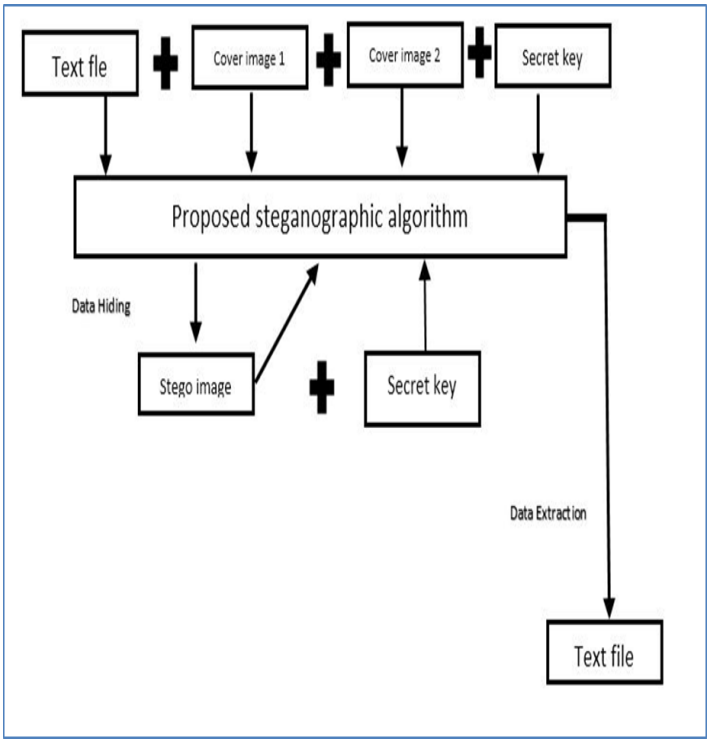

Fig. 9. The basic structure of the proposed algorithm

4. Working Process and result discussion

4.1 Import Image

Let's come to the coding part. First, we take a button for indicating an image from our directory as an input. So that we can hide our message in the image. In this method, we take a png type image as an input to hide the message. When you click on this button, a pop up will come to indicate your image. When the image is successfully loaded, we calculate the maximum length of a message that can be preserved inside an image. We save that length inside the variable [15].

4.2 Hiding Your Text:

We use this method to encode our message inside the image. After importing an image, we write our information in the text box. If you do not import an image or left the text box as blank, it will show you a message like "Please Select the Image to encode" or "Please Enter the Text to be hidden". When you write your information into the textbox, it converts your message into binary form and calculates it's the length. If your text's length is greater than the value of the text size variable, it will show you a message. Suppose, the capacity of carrying information your image is 60kb and you want to save a 70kb size information inside the cover image. Then the information will be like "KB". After checking the information size, we take a pixel from an image and find the R, G, B's value from it. After retrieving R, G, B's value, we take the first word from the information and divide every letter from that word and convert it into binary form. Take the binary number and replace it with the value of Blue pixel's value. Replacing every bit from the message, we save the message length in the last pixel of the image [16].

International Journal of Scientific & Engineering Research, Volume 11, Issue 6, June-2020
ISSN 2229-5518

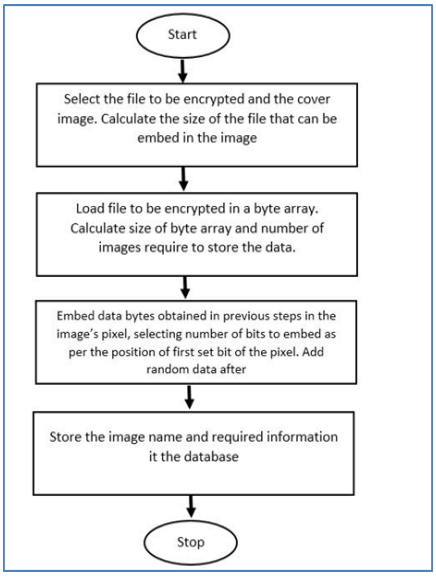

Fig. 10. Message Hiding Flowchart

4.3 Extracting Message from Image

We use this method to extract your information from the cover image. In order to extract, you have to select an image which contains information inside. This method will check whether we select a cover image or not to perform an extraction. If you do not indicate any image, it will show a message like "Please Select the Image to decode". When we indicate a cover image, we get the message's length from the last pixel of the image. Then run a loop from the image's height to its width. We print every letter from the last bit of every pixel and print the information until the loop reaches the message's length. In any case, if we indicate an image that does not contain any secret information then it will print a message like "No data hidden in the image". After successfully extracting our information from the cover image, we print this message in a textbox. As a result, the information will be like "Image Decoded Successfully Check the decoded information in the Decoded TextBox" [17].

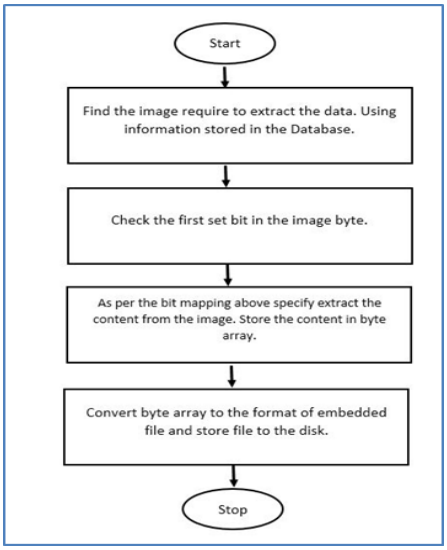

Fig. 11. Message extracting Flowchart

4.4 Interface of Application

Below is the interface of my application. The yellow box is a picture box. I have also added another feature. As a result, you can hide your information from a text file as well.

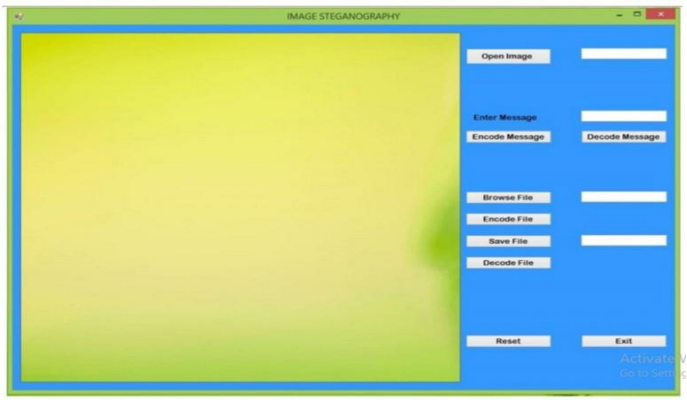

Fig. 12. Home page of the application interface

4.5 Importing Image

First of all, you have to indicate a cover image by clicking on the open image button. When you take a cover image, it will be shown in this picture box and the path of your image will show in the first text box. See the image as follows-

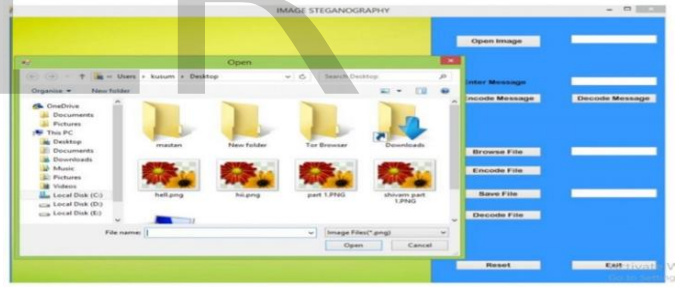

Fig. 13. Importing Image interface page

4.6 Write Your Message

When you select your cover image, you can write your message in the second text box.

Fig. 14. Writing message -1(a)

When you write your message in the textbox then click on

International Journal of Scientific & Engineering Research, Volume 11, Issue 6, June-2020
ISSN 2229-5518

the encode information button. As a result, a dialogue box will open to preserve your image. See the image below- Now write a name for your cover image and click on the Save button. Finally, your information is saved.

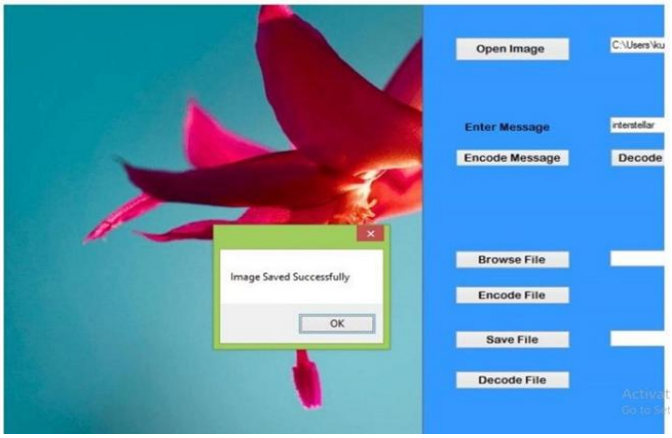

Fig. 15. Writing message(Image saved Successfully notification) -1(b)

4.7 Extract Your Information

In order to extract, you have to click on the 'open image' button. As a result, a dialogue box will open and you have to select a cover image that has your secret message inside. When you click on the "Decode Message" button, it will print your information inside the textbox.

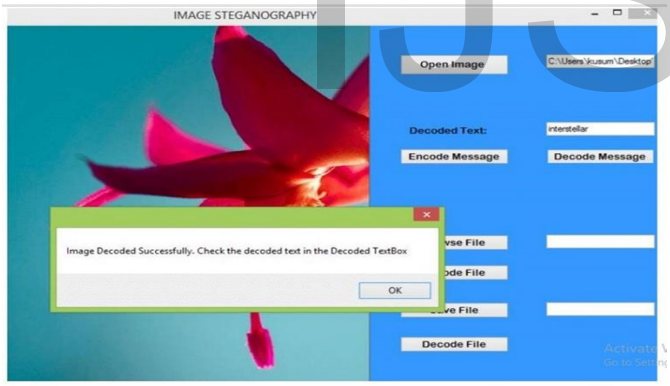

Fig. 16. Extracting Information from image

5 Conclusion

In the present era, data transfer is rapidly growing. Security is a very important issue. This application creates a steganography image in which personal information is embedded. The proposed method provides higher security and can protect the information from steganography attacks. The image resolution doesn't change much and is negligible when we embed the information into the image. The major limitation of this method is designed for bitmap images(.png). It takes only png images as a carrier file, and the compression depends on the text's size and the carrier image size. The future Work on this application will be to develop the types of images like .jpg etc. for encoding and decoding purposes. You can also make an android application for the same to make the method more handy and portable.

ACKNOWLEDGMENT

This work was supported by Dept. of ICT, Islamic University, Kushtia-7003, Bangladesh.

REFERENCES

[1] H. Wu, H. Wang, C. Tsai and C. Wang, "Reversible image steganographic scheme via predictive coding", 1(2010), ISSN: 01419382, 35-43.

[2] J, Corporation, Steganography, "http://www.webopedia.com/TERM/S/steganography.html", 2005.

[3] B. Dunbar, "A detailed look at Steganographic Techniques and their use in an Open-Systems Environment", Sans Institute, 1(2002).

[4] C. Christian, "An Information-Theoretic Model for Steganography, Proceedings of 2nd Workshop on Information Hiding", MIT Laboratory for Computer Science. 1998.

[5] N Ghoshal, J K Mandal, "A steganographic scheme for colour image authentication (SSCIA)", Recent Trends in Information Technology ICRTIT 2011 International Conference on (2011), 826 831.

[6] M. D. Swanson, B. Zhu and A. H. Tewfik, "Robust Data Hiding for Images", IEEE Digital Signal Processing Workshop, University of Minnesota, September 1996 (37-40).

[7] N. Johnson, "Digital Watermarking and Steganography: Fundamentals and Techniques", The Computer Journal. (2009).

[8] N. Johnson, "Survey of Steganography Software", Technical Report, January 2002.

[9] W, Peter, "Disappearing Cryptography: Information Hiding: Steganography & Watermarking (second edition)", San Francisco: Morgan Kaufmann. 3(1992) 192-213.

[10] S. K. Muttoo, Sushil Kumar ,"Data Hiding in JPEG Images", Department of Computer Science, University of Delhi.

[11] M. Chen, N. Memon, E.K. Wong, "Data hiding in document images", in: H. Nemati (Ed.). Premier Reference Source–Information Security and Ethics: Concepts, Methodologies, Tools and Applications, New York: Information Science Reference, 2008, pp. 438-450.

[12] D.C. Lou, J.L. Liu, H.K. Tso, "Evolution of information – hiding technology", in H. Nemati (Ed.), Premier Reference Source–Information Security and Ethics: Concepts, Methodologies, Tools and Applications, New York: Information Science Reference, 2008, pp. 438-450.

[13] E. Cole, "Hiding in Plain Sight: Steganography and the Art of Covert Communication", Indianapolis: Wiley Publishing, 2003.

[14] H. Nemati (Ed), "Premier Reference Source–Information Security and Ethics: Concepts, Methodologies, Tools and Applications", New York: Information Science Reference, 2008, pp. 554-569.

[15] M. Warkentin, M.B. Schmidt, E. Bekkering, "Steganography and steganalysis, Premier reference Source–Intellectual Property Protection for Multimedia Information technology", Chapter XIX, 2008, pp. 374-380.

[16] N.N. El-Emam,"Hiding a large amount of data with high security using steganography algorithm", Journal of Computer Science (2007)223-232 .

[17] P.Y. Chen, W.E. Wu, "A modifed side match scheme for image steganography", International Journal of Applied Science & Engineering 7 (2009) 53-60.

[18] C.C. Chang, H.W. Tseng, "A steganographic method for digital image using side match", Pattern Recognition Letters 25 (2004) 1431-1437.

[19] P.C. Wu, W.H. Tsai, "A steganographic method for images by pixel-value differencing", Pattern Recognition Letters 24 (2003) 1613-1626.