View metadata, citation and similar papers at core.ac.uk
brought to you by CORE

Mathematical and Software Engineering Vol. 5, No. 2 (2019), 39-44.
Varɛpsilon Ltd, http://varepsilon.com

Image Steganography using Arduino Microcontroller

Hristo Paraskevov[1], **Aleksandar Stefanov**[2], **and Radostin Rafailov**[3]

[1]Department of Mathematics and Informatics, University of Shumen "Ep. Konstantin Preslavski", Bulgaria, h.paraskevov@shu.bg
[2]Department of Mathematics and Informatics, University of Shumen "Ep. Konstantin Preslavski", Bulgaria,
[3]Department of Mathematics and Informatics, University of Shumen "Ep. Konstantin Preslavski", Bulgaria

Abstract

This paper aims to investigate and develop an algorithm, based on the LSB method, for steganographic message hiding using a single-board Arduino microcontroller, with sequential scheme to embed the hidden message. It has been experimentally found that the changes that the algorithm makes to the carrier file are within normal limits and that the incompleteness is at a good level.

Keywords: steganography, image processing, arduino.

1 Introduction

A single-board computer is a working computer implemented on a printed circuit board that houses a microprocessor, RAM, I / O systems, and other modules required to operate it. Some are available as CPU and memory only cards, with the ability to add different extensions. Single-board computers are used by developers, in various fields, as demonstration systems or as part of industrial or embedded systems.

There are many advantages to using single board computers. They are lighter, compact, and use much less energy. All of these things make single-board computers suitable for developers who can use them to develop new applications, test, debug, hardware development, and more.[1]

2 Methods and Algorithms

Steganographic methods in the spatial area are most commonly used by stegoprograms because of the good hiding (inconspicuousness) of messages, high stegocapacity and easy implementation.

One of the best literary reviews for this steganography was made in [2]. This type

of steganography involves the methods of Least Significant Bit (LSB) [3,4,5] and BPCS (Bit Plane Complexity Segmentation) [6]. Direct methods embed information directly into bits of the container (for images in pixels). One of the earliest digital steganography methods with the idea of the LSB method was proposed in the early 1990s in [7]. The method for changing the lowest bit is the most common method among the methods for changing the spatial domain.
The youngest bit (LSB) in one image carries the least information. It is well known that human perceptions cannot sense a change in this lifestyle. In fact, the change in the LSB is a noise that can be used to embed a hidden message. In images where each pixel is encoded with a single byte, the size of the embedded information by this method can be up to 1/8 of the size of the container.
LSB methods can be divided into two main types - LSB replacement and LSB matching [3]. The first type directly replaces the youngest bits with the message bits. In the second type, bytes are selected in which the lowest bits of the container match the bits of the message. As a result, statistical and other characteristics do not change significantly in the container.
Similar to LSB bit exchange, block BPCS steganography hides secret data by block replacement. Each bit field in the image is segmented into identical pixel blocks (typically 8x8) that are classified into information and noise blocks [6].

The proposed embedding algorithm is as follows:

1. Enter the message that will be hidden in the BMP file.
2. The hide message is converted into binary sequence.
3. Check that the SD card is initialized and ready for use. If an SD card is not available an error message is displayed.
4. Check that the SD card contains a BMP file and if it does not, an appropriate missing message is displayed, otherwise it is loaded.
5. A second file of the same size and color range is created. (stego file).
6. Insert the hidden LSB message into the newly created file.
7. The steg file is written to the SD card and a message for successful program execution is displayed.

The proposed algorithm for retrieving a hidden message is in the following steps:

1. Check that the SD card is initialized and ready for use. If an SD card is not available an error message is displayed.
2. Check whether the stego file is in the SD card and if it is missing, the appropriate message is displayed, otherwise it is loaded.
3. The hidden message is retrieved using the LSB method.
4. The extracted hidden message from the stego file is saved in a text file.
5. The text file is saved on the SD card and a message is displayed for the successful execution of the program.

3 Results and Discussion

A software implementation of the proposed algorithm has been developed. For containers, we created a database of 100 BMP images downloaded from the

Internet up to 1KB in size. In addition, the preferred image processing database is available at: (http://sipi.usc.edu/database/database.php?volume=misc.). Hidden messages are text-based and vary from experiment to experiment.
When evaluating the results, two important indicators are considered: embedding efficiency and signal-to-noise ratio - PSNR.
Embedding efficiency E_e is the ratio of the size of the largest message that can be embedded in the container to the size of the container.

$$E_e = \frac{V_{mes.}}{V_c} \quad (1)$$

where:
V_{mes} – size of hidden message in KB or MB
V_c – size of the container in KB or MB.
PSNR calculates the peak signal-to-noise ratio, in decibels, between the two images. This ratio is often used to measure the quality of an original image and a modified image. The higher the PSNR values, the more noticeable the changes are.
To calculate the PSNR, first calculate the MSE using the following expression:

$$MSE = \frac{1}{mn}\sum_{0}^{m-1}\sum_{0}^{n-1}\|f(i,j) - g(i,j)\|^2 \quad (2)$$

where:
m, n - the dimension of the image
f (i, j) is a pixel of the original image
g (i, j) is the pixel of the modified image

$$PSNR = 20\log_{10}\left(\frac{MAX_f}{\sqrt{MSE}}\right) \quad (3)$$

where:
MAX_f - the maximum value used to identify a color.

Experiment 1:
This experiment used all database images and inserted text messages of different sizes, randomly generated through Lorem Ipsum. It aims to check the maximum length of hidden message characters due to limited memory on the single-board Arduino computer.
Experiment 1 Results:
During the experiments, it began with a gradual increase in the size of a five-character fixed text message and a graphic image size up to 255B. The reason for the small step is the limited internal memory on the single board computer. Until the maximum text size for the graphic image with the specified size is reached, the text is extracted without loss and error. Then, experiments with a twice as large image began. Upon reaching the length of the hidden message of 34 characters, retrieving the secret message became error-prone. Attempts were made with larger images than the base, but this did not change the final results.
It has been experimentally proved that in the proposed algorithm the maximum

length of the hidden message is 33 characters. This is due to limitations in available memory after compiling the program.

Experiment 2:
In this experiment, all database images were used and a text randomly generated through Lorem Ipsum was inserted. The experiment aims to check the values of the selected indicators for evaluating the results.

Visual analysis


Fig. 1.(a) Original


Fig. 1.(b) Stego file

After the visual evaluation, the histograms of the original images and stego files were also analyzed.

Histogram analysis

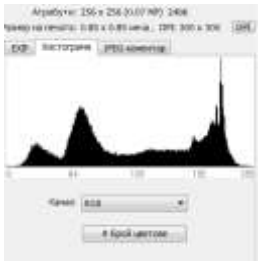
Fig. 2.(a) Original

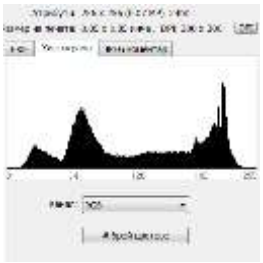
Fig. 2.(b) Stego file

Embedding efficiency

Table 1 presents the results obtained when examining the efficiency ratio. The table represents randomly selected images of different sizes, given in pixels. The size of the secret message increases to the maximum possible size found in Experiment 1. The table shows that due to the restrictions in the length of the secret message, the efficiency of the algorithm in larger images decreases.

PSNR analysis

PSNR research used 16x16 pixel base images and a 33 byte embedded message. Table 2 shows the experiment data of randomly selected images from the database.

4 Conclusion

Based on the research, analysis and experiments, the capabilities of single-board computers and their ability to implement steganographic software in them are presented. An insertion algorithm is presented that meets the requirements of the LSB method and meets its required quality, despite the established limitations of

the single-board computer, and an algorithm for extracting the lost message from a stego file without loss.

Table 1 Embedding efficiency E_e

Cover file size in pixels	The size of the secret message in bytes	Embedding efficiency
8x8	5	0,0260
8x8	15	0,0781
8x8	24	0,1250
16x16	25	0,0325
16x16	30	0,0390
16x16	33	0,0429
32x32	33	0,0107
32x32	33	0,0107

Table 2 PSNR values

Image	The size of the secret message in bytes	PSNR
Test1 (16x16)	33	60.78652822611028
Test2 (16x16)	33	60.45229067124079
Test3 (16x16)	33	60.08245500870909

The result of the algorithms applied to a single container with a sequential scheme for embedding the hidden message enables the implementation of new methods and ideas in order to improve and refine the steganographic algorithms applied in a single-board computer.
This could be accomplished by using a multiple container to embed the hidden message, i.e. hide a message in multiple images. The sequential embedding scheme of the hidden message could be replaced by the scattered one, using a pseudorandom number generator to distribute the message evenly in bytes of the entire container. It is also possible to integrate network steganography by using an internet module to a single-board computer, which will greatly increase the security of data transmission.

Acknowledgements
This work is partially supported by the Scientific Fund РД-08-96/01.02.2019.

References

[1] Ortmeyer, C. (2014) Then and now: a brief history of single board computers. Electron. Des. Uncovered, 6, 1-11.
[2] Chen, M., Zhang, R., Niu, X., & Yang, Y. (2006, December) Analysis of current steganography tools: classifications & features. In 2006 International Conference on Intelligent Information Hiding and Multimedia (pp. 384-387). IEEE.
[3] Koduri, N. (2011) Information security through image steganography using least significant bit algorithm. Master of Science in Information Security and Computer Forensics, University of East London.

[4] Johnson, N. F., & Jajodia, S. (1998) Exploring steganography: Seeing the unseen. Computer, 31(2), 26-34.
[5] Provos, N., & Honeyman, P. (2003) Hide and seek: An introduction to steganography. IEEE security & privacy, 1(3), 32-44.
[6] Siahaan, A. P. U. (2016) High Complexity Bit-Plane Security Enchancement in BPCS Steganography. International Journal of Computer Applications 148(3), 17-22.
[7] Petitcolas, F. A., Anderson, R. J., & Kuhn, M. G. (1999) Information hiding-a survey. Proceedings of the IEEE, 87(7), 1062-1078.

Copyright © 2019 Hristo Paraskevov, Aleksandar Stefanov and Radostin Rafailov. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.