

RGB Intensity Based Variable-Bits Image Steganography

Mohammad Tanvir Parvez and Adnan Abdul-Aziz Gutub

College of Computer Sciences & Engineering

King Fahd University of Petroleum & Minerals, Dhahran 31261, Saudi Arabia

{tparvez, gutub}@kfupm.edu.sa

Abstract

In this paper, we present a new algorithm for RGB image based steganography. Our algorithm introduces the concept of storing variable number of bits in each channel (R, G or B) of pixel based on the actual color values of that pixel: lower color component stores higher number of bits. Our algorithm offers very high capacity for cover media compared to other existing algorithms. We present experimental results showing the superiority of our algorithm. We also present comparative results with other similar algorithms in image based steganography.

Keywords: Steganography, RGB image, cryptography, information hiding.

1. Introduction

Steganography deals with embedding information in a given media (called cover media) without making any visible changes to it [1]. The goal is to hide an embedded file within the cover media such that the embedded file's existence is concealed. Image based steganography uses images as the cover media. Several methods have been proposed for image based steganography, LSB being the simplest one. Techniques involving bitmap images (RGB images) as cover media use single or multi-channel hiding, RNG or color cycle methods [2] etc. Gutub et al [3] describes the pixel indicator technique where one channel is used to locate the channel to store data.

In this paper, we propose a new technique for RGB image steganography, where color intensity (values of R-G-B) is used to decide the no of bits to store in each pixel. Channels containing lower color values can store higher no of data bits. The sequence of channels is selected randomly based on a shared key. Our technique ensures a minimum capacity (as

opposed to [3]) and can accommodate to store large amount of data. Experimental results show that our algorithm performs much better compared to the existing algorithms. Our algorithm can also be used to store fixed no of bits per channel, but can still offer very high capacity for cover media.

The rest of the paper is organized as follows. Section 2 deals with a brief discussion of the algorithm in [3]. Section 3 discusses our new algorithm. Section 4 presents the detailed experimental results and comparisons. Finally, section 5 is the conclusion.

2. Review

In this section, we review the pixel indicator technique in [3]. The pixel indicator technique uses the least two significant bits of one of the channels from Red, Green or Blue as an indicator for existence of data in the other two channels. The indicator channels are chosen in sequence, with R being the first. Table 1 shows the relation between the indicator bits and the amount of hidden data stored in the other channels.

Table 1: Relation between the indicator bits and the amount of hidden data.

Indicator bits	Channel 1	Channel 2
00	No hidden data	No hidden data
01	No hidden data	2 bits of hidden data
10	2 bits of hidden data	No hidden data
11	2 bits of hidden data	2 bits of hidden data

The disadvantage of the algorithm in [3] is that the capacity depends of the indicator bits and based on the cover image, the capacity can be very low. Also, the algorithm in [3] uses fixed no of bits per

channel (2 bits) to store data and the image may get distorted if more bits are used per channel.

The algorithm we propose in this paper, does not suffer from these problems. Our algorithm guarantees a minimum capacity for every cover image and the

number of bits stored in each channel varies depending on the intensity. This can give very high capacity for some of the cover images. In section 4, we present experimental results supporting our claim.

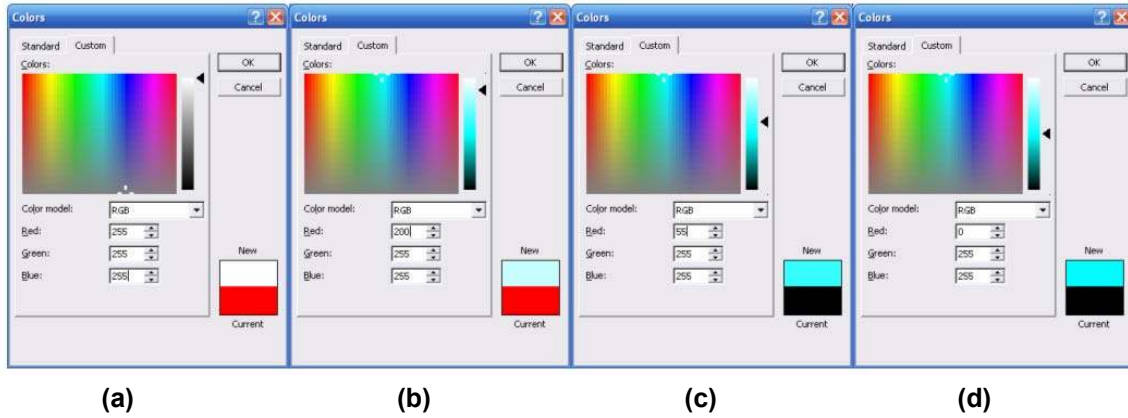


Figure 1: Effect in colors for changes in the 'Red' values.

3. The algorithm

In this section, we briefly outline our proposed algorithm for RGB image based steganography. The idea behind our algorithm is that, for 'insignificant' colors, significantly more bits can be changed per channel of an RGB image. Figure 1 shows one example. In (a) the color is WHITE (R=255, G=255, B=255). In (b), we only change the last 4 bits of R to zeros, resulting in a color where distortion can be noticed. In (c), the color component is same as (a), except that R = 55. In (c), we again set the least 4 bits of R to zeros, resulting in a color which seems to be same as (c).

Our idea is that, lower color-value of a channel has less effect on the overall color of the pixel than the higher value. Therefore, more bits can be changed in a channel having 'low' value than a channel with a 'high' value.

Therefore, we propose the following algorithm.

- Use one of the three channels as the indicator. The indicator sequence can be made random, based on a shared key between sender and receiver.
- Data is stored in one of the two channels other than the indicator. The channel, whose color value is lowest among the two channels other than the indicator, will store the data in its least significant bits.
- Instead of storing a fixed no of data-bits per channel, no of bits to be stored will depend on the color value of the channel. The lower the value, the higher the data-bits to be stored. Therefore a partition of the color-

values is needed. Through experimentations, we show that optimal partition may depend on the actual cover image used.

- To retrieve the data, we need to know which channel stores the data-bits. This is done by looking at the least significant bits of the two channels other than the indicator:
 - If the bits are same, then the channel following the indicator in cyclic order stores the data.
 - Otherwise, the channel which precedes the indicator in cyclic order stores the data.

Here, the cyclic order is assumed to be R-G-B-R-G-B and so on. The appropriate bits can be set while the data is stored.

Figure 2 shows the encoding (at sender's side) and decoding (at receiver's side) part of our algorithm in flow charts. Note that, it is assumed that a shared key and partition scheme is already agreed upon by the two parties.

Figure 3 demonstrates one example of storing data bits in a channel. In step – 1, the indicator channel (here G) is selected randomly. In step – 2, the data channel is chosen (R). In step – 3, no of data bits to store is determined from the current channel value and partition scheme. Step – 4 stores the data bits and step – 5 modifies the other channel (B) LSB, which is used while retrieving the data.

3.1. Partition schemes

In our algorithm, a partition scheme is defined as a monotonically decreasing sequence $[a_i]$, $i = 1$ to 8.

Assume that the color value of a channel is c . Then that channel with value c stores i no of data bits if $c \geq a_i$ and for all $j, j < i, c < a_j$. For correctness of the algorithm, we only use valid partitions schemes. We define a *valid* partition scheme as follows: let $[a_i]$ be a partition scheme where lower i bits of a_i is all 0. Let $[b_i]$, $i = 1$ to 8, be another sequence, where b_i is generated by setting the lower i bits of a_i all 1. If $a_i > b_{i+1}$, $i = 1$ to 7, then $[a_i]$ is a *valid* partition scheme. This simple condition ensures that the same no of data bits are read from a channel in the receiver's side as stored in the sender's side.

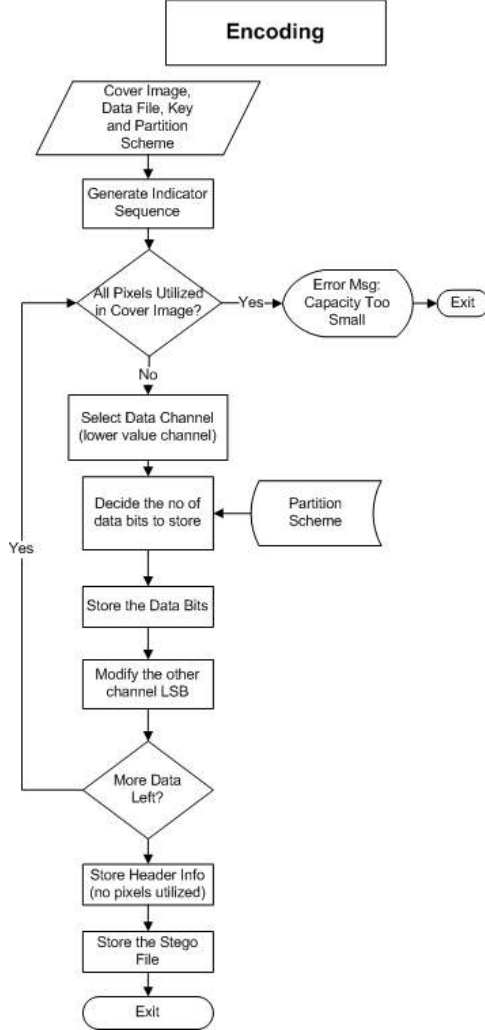


Figure 2a: Flow charts of the encoding part of our algorithm.

4. Experimentations

We implemented our algorithm in MATLAB 2007 and carried our experimentations with different cover images and data files exploring different values

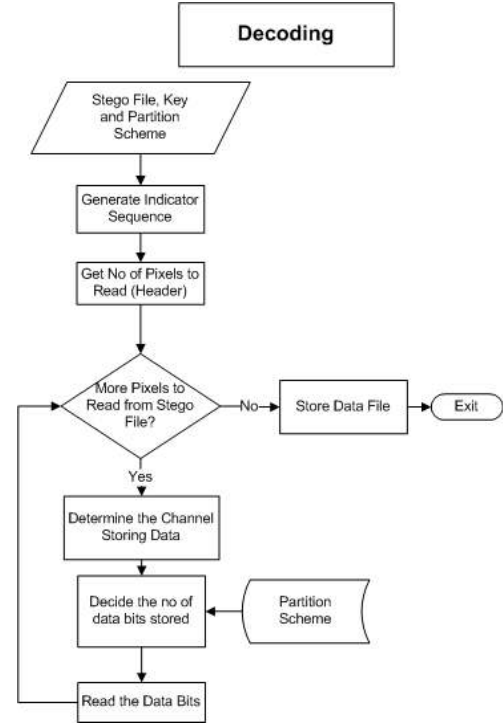


Figure 2b: Flow charts of the decoding part of our algorithm.

	R	G	B
	82	45	91
1	82	45	91
2	82	45	91
3	01010010	45	01011011
4	01011101	45	01011011
	93	45	91
5	01011101	45	01011010

Figure 3: An example of hiding data bits inside a channel.

for shared keys and partition schemes. Here, we only present the results obtained using the cover image and the data file as shown in Figure 4.

Figure 5 shows the stego files for 4 different partition schemes. The value of the key was 17 and

the indicator sequence was chosen randomly using

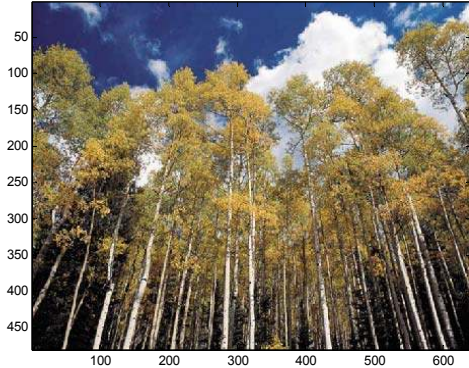


Image size: 640 X 480, Bit depth: 24
No of pixels = 307200
(a) Cover Image

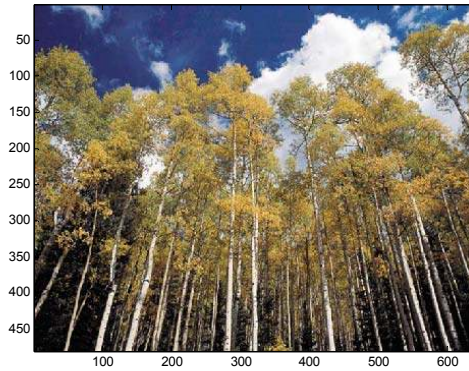


Image size: 150 X 117, Bit depth: 24
Data length = 150896 bits
(b) Data File (a bitmap)

Figure 4: Cover media and data file used in the experimentations.

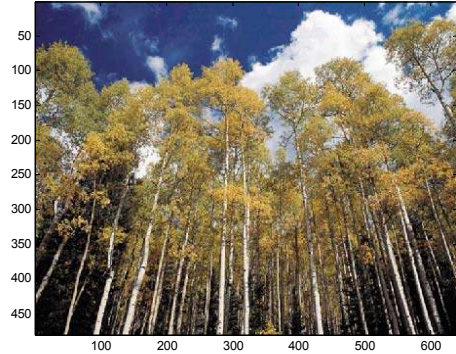


Pixels utilized in cover media: 50939
Partition Scheme: [256, 256, 0, 0, 0, 0, 0, 0]
(a) Constant 3 bits per channel

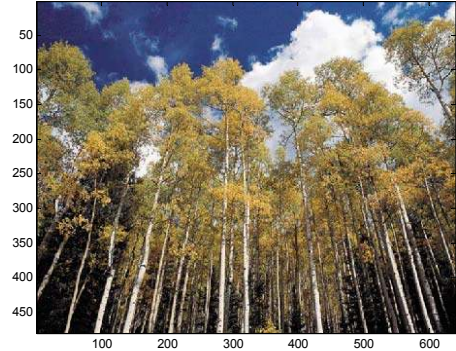


Pixels utilized in cover media: 41061
Partition Scheme: [256, 256, 96, 0, 0, 0, 0, 0]
(b) 3 to 4 bits per channel

this shared key.



Pixels utilized in cover media: 38364
Partition Scheme:[256, 256, 256, 0, 0, 0, 0, 0]
(c) Constant 4 bits per channel



Pixels utilized in cover media: 35791
Partition Scheme: [256, 256, 256, 48, 0, 0, 0, 0]
(d) 4 or 5 bits per channel

Figure 5: Stego files for 4 different partition schemes.

The same set of images were used to evaluate the algorithm in [3], the pixel indicator technique. Table 2 shows the comparative results of the two algorithms.

Table 2 clearly shows that, intensity based variable-bits algorithm clearly performs better than the pixel indicator algorithm in terms of capacity utilizations. For comparison purpose, we have modified the pixel indicator algorithm to store up to 3 or 4 data bits per channel. But this modification leads to visual distortions to the cover image.

Our algorithm ensures a minimum capacity for any cover image, whereas for pixel indicator method, the capacity may sometimes become very low. As an example, for the cover image and data file of Figure 6, our algorithm uses only 16.5% of the capacity of the cover image (for fixed 3 bits data per channel), whereas pixel indicator method cannot store the data file due to capacity shortage!

Table 2: Comparison of performance of two steganography algorithms.

Technique	No of data bits per channel (bits)	No of pixels of cover media utilized (pixels)	No of pixels of cover media utilized (percentage)
Intensity Based Variable-Bits	3	50939	16.58%
	3 or 4	41061	13.37%
	4	38364	12.49%
	4 or 5	35791	11.65%
Pixel Indicator	2 + 2	77578	25.25%
	3 + 3	59051	19.22%
	4 + 4	44687	14.55%

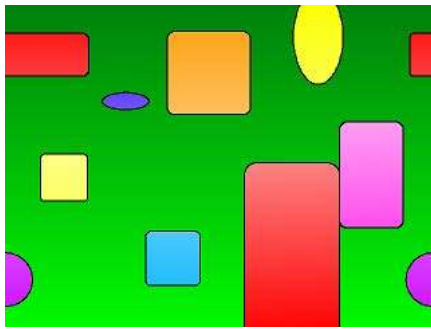


Image size: 640 X 480, Bit depth: 24
No of pixels = 307200

(a) Cover Image



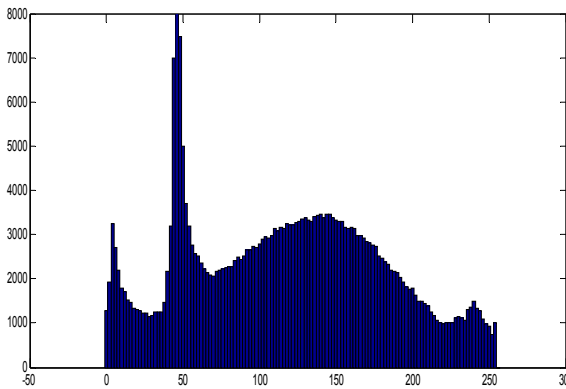
Image size: 150 X 117, Bit depth: 24
Data length = 150896 bits

(b) Data File (a bitmap)

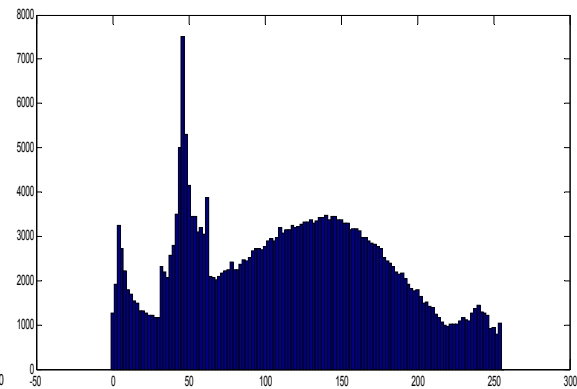
Figure 6: Capacity variations with different cover media.

Figure 7 – 9 show the color histogram plots for the cover and stego images for all three channels. The nice thing to note from the histograms is that, our algorithm preserves the general shapes of the

histograms. This feature of our algorithm makes it difficult to detect whether any data is hidden or not in the transmitted image.



(a) Cover image



(b) Stego image

Figure 7: Histogram of Red channel.

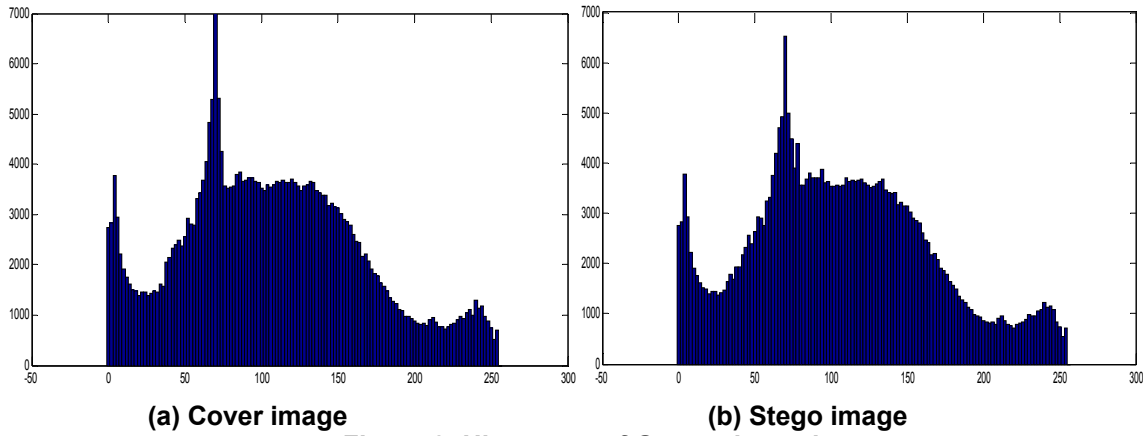


Figure 8: Histogram of Green channel.

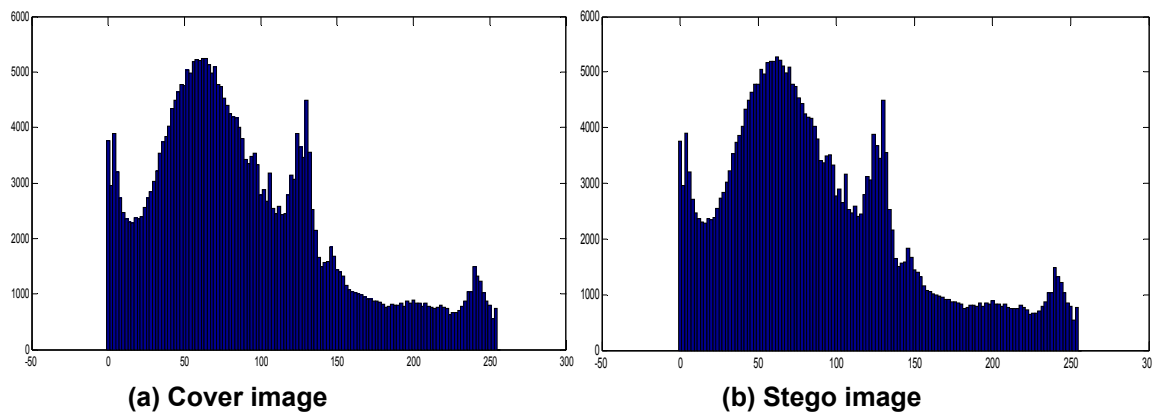


Figure 9: Histogram of Blue channel.

5. Conclusion and future works

In this paper, we introduce a new idea in image based steganography, where variable no bits can be stored in each channel. Our algorithm uses actual color of the channel to decide no of data bits to store. This approach leads to very high capacity with low visual distortions. Experimental results demonstrate that our algorithm performs better than other similar algorithms.

There are several ways to improve our variable-bits algorithm:

- Select the partition at run time, based on the cover media, rather than using the static (fixed) partition scheme for all cover images.
- Use color information of all three channels to determine the partition. This will lead to using different partition schemes for different parts of the image.

6. Acknowledgement

We would like to thank King Fahd University of Petroleum & Minerals (KFUPM) for supporting this work.

7. References

- [1] Provos, N., Honeyman, P, *Hide and seek: An introduction to steganography*, IEEE Security & Privacy Magazine 1 (2003) pp. 32-44
- [2] Karen Bailey, Kevin Curran, *An evaluation of image based steganography methods using visual inspection and automated detection techniques*, Multimedia Tools and Applications, Vol 30 , Issue 1 (2006) pp. 55-88
- [3] Adnan Gutub, Mahmoud Ankeer, Muhammad Abu-Ghalioun, Abdulrahman Shaheen, and Aleem Alvi, *Pixel indicator high capacity technique for RGB image based Steganography*, WoSPA 2008 – 5th IEEE International Workshop on Signal Processing and its Applications, University of Sharjah, Sharjah, U.A.E. 18 – 20 March 2008.