Steganography

Omar ROMDHANI
Omar.Romdhani@ensi-uma.tn
Ibrahim MAHDI
Ibrahim.Mahdi@ensi-uma.tn

July 2018

***Abstract*—This paper introduces two methods of LSB Steganography walking through the details of its implementations**

I. Introduction

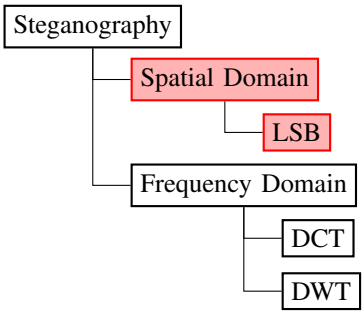
Steganography is the process of hiding private or secret data (a file, message, image, or video) within a carrier(message, image, or video) in an invisible manner[1].

The word steganography combines the Greek words steganos, meaning covered, and graphein meaning writing or drawing .

Application of Steganography varies from military, medical images[2], industrial applications ... to copyright and Intellectual Property Rights (IPR). By using loss-less (without loosing data) steganography techniques, messages can be sent and received securely[3].

At the beginning, Steganography was based on hiding data in images, but now it is safer to hide data in videos[4] due to the relative complexity of its structure compared to images.

There are various techniques available to implement steganography namely three well known techniques are the Least Significant Bit (LSB), the Discrete Cosine Transform (DCT) and the Discrete Wavelet Transform (DWT). There are two types of domains in which steganography is implemented i.e. spatial domain and frequency domain [5].



In spatial domain, processing is applied directly on the pixel values of the image whereas in frequency domain, pixel values are transformed and then processing is applied on the transformed coefficients. Steganography is used in covert communication. The secret image which is sent to the destination is embedded into the cover image.

In our project we have implemented two different LSB

methods (Spatial Domain).

II. Spatial Domain Steganography

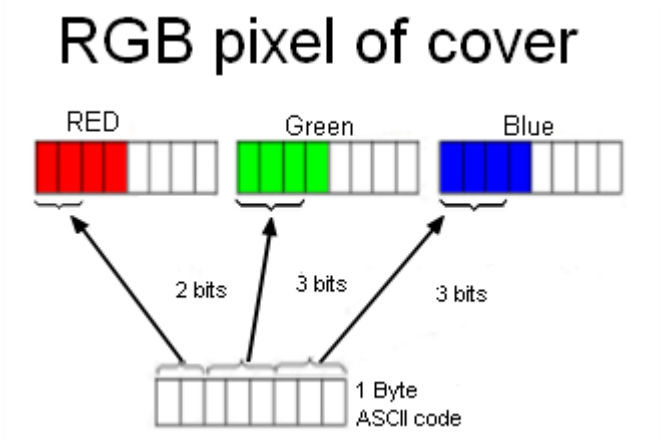
Least Significant Bit (LSB) technique is one of the simplest approaches for secure data transfer. In this technique, the least significant bits of binary sequences of the cover image are used to carry the secret message.

A. First method

In this method, a color image is considered as a cover and every character of the secret message will be hidden in a single pixel of the carrier image.

A character can be represented with its ASCII code using 8 bits.This Byte will be inserted in the least significant bits of RGB values (Red,Green,Blue) of a pixel.

E.g 2 bits are inserted in the 2 least significant bits of the Red, 3 bits are inserted in the 3 least significant bits of the Green and 3 bits are inserted in the 3 least significant bits of the Blue.



Encoding algorithm:

1) Read the cover image and text message which is to be hidden.
2) Add the special character '$\backslash 0$' to the end of the message to mark the end.
3) For each character of the text message do :
   a) Convert the ASCII code of the i^{th} character to its binary representation.
   b) Convert the R,G,B values of the i^{th} pixel to its binary representation.
   c) Substitute the 2 least significant bits of the Red with the first 2 bits of the ASCII code.
   d) Substitute the 3 least significant bits of the Green with the next 3 bits of the ASCII code.
   e) Substitute the 3 least significant bits of the Blue with the last 3 bits of the ASCII code.
4) Create the new image.
5) End.

Decoding algorithm:

1) Read the steganography image.
2) For each pixel of the image do :
   a) Convert the R,G,B values of the i^{th} pixel in binary.
   b) Retrieve the 2 least significant bits of the Red.
   c) Retrieve the 3 least significant bits of the Green.
   d) Retrieve the 3 least significant bits of the Blue.
   e) Combine these bits to retrieve the ASCII code of the character.
   f) if ASCII code is equal to zero then JUMP to 3).
   g) Save the character and JUMP to a).
3) Assemble the whole message.
4) End.

B. Second method

In this method we disperse further the text in the image instead of coding each character in a pixel we will code each character in 8 consecutive pixels : each character bits are inserted in the least significant bits of each pixel.

We write the bits of the message in the 1^{st} plan of bits of the Red color ... then 1^{st} plan of bits of the Green then the Blue When all the 1^{st} plans are used we move to 2^{nd} plan of the Red, the Green and the Blue

and so on until we use all the 3^{rd} plans.

Encoding algorithm:

1) Read the cover image and the secret text message.
2) Add the special character '$\backslash 0$' to the end of the message to mark the end.
3) Convert each character to its binary representation making a long sequence of bits (the binary representation of the first character then the second and so on).

4) For each bit of the bit sequence do:
   a) Colour = Red
   b) j = 1
   c) Start with the j^{th} bit of the ”Colour” value of each pixel (variable colour gets the values :Red then Green then Blue)
      i) Convert the ”Colour” value of the i^{th} pixel to its binary representation;
      ii) Substitute the least significant bit of the ”Colour” with the i^{th} bit of the bits sequence
      iii) If the j^{th} bits of the ”Colour” value of each pixel is used then :
         - if ”Colour” is Blue then JUMP to d)
         - else : ”Colour” = next(Colour) (Red then Green then Blue) , j = j + 1 and JUMP to c).
   d) if $j \leq 3$
      then :
      - ”Colour” = Red
      - j = j + 1
      - JUMP to c)
5) END.

Decoding algorithm:

1) Read the steganography image.
2) For each 8 consecutive pixels of the image do :
   a) Retrieve the least significant bits of the Red layer.
   b) Get the ASCII code of the hidden character.
   c) If the ASCII code is equal to Zero JUMP to 4)
   d) Save the character.
   e) If the end of the current plan is reached we move to the next plan of bits.{(Red,Green,Blue)First bit of each then (Red,Green,Blue)second bit of each the last plan}
   f) Repeat the process from a)
3) Assemble the whole message
4) END.

C. Implementation:

We have chosen Python3 as a language to implement our program.

Python is a high-level, interpreted and general-purpose dynamic programming language that focuses on code readability. It helps to do coding in fewer steps as compared to Java or C++. The Python is widely used in bigger organizations because of its multiple programming paradigms. They usually involve imperative and object-oriented functional programming. It has a comprehensive and large standard library that has automatic memory management and dynamic features.

In addition Python3 is a portable language you just need to install a python3 compiler and you will be able to run the program. You may run it on Windows , Unix or even Android; you just need to install the packages

In our program we have used several packages :

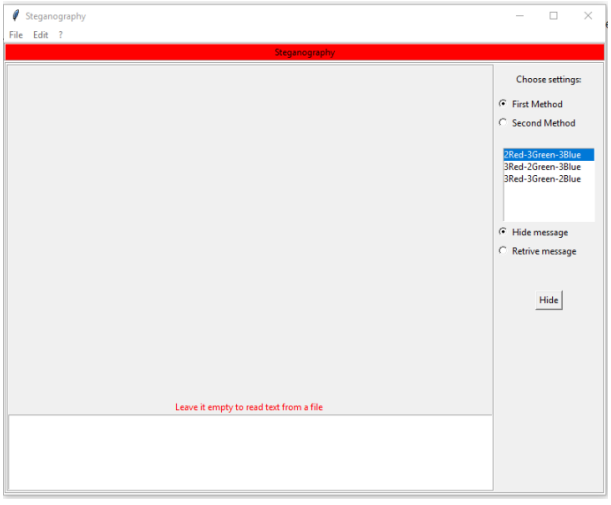
- tkinter (pre-installed)
- math (pre-installed)
- numpy ($3^{rd} - party$ package which you need to install it in order to execute our script)
- scipy ($3^{rd} - party$ package which you need to install it in order to execute our script)
- PIL($3^{rd} - party$ package :PIL or Pillow)

This is a video on : How to add packages to Python3

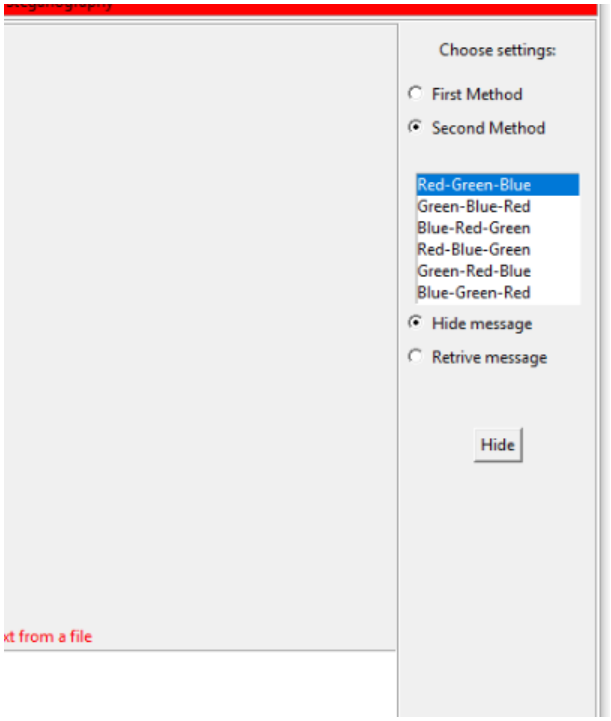
An executable was generated for easy use on Windows 32-bit or 64-bit.

D. How to use our program:

1) Run the executable(”GUI.exe”) or build the python script (after adding the packages Numpy,Scipy and Pillow)
2) This windows should Pop-up

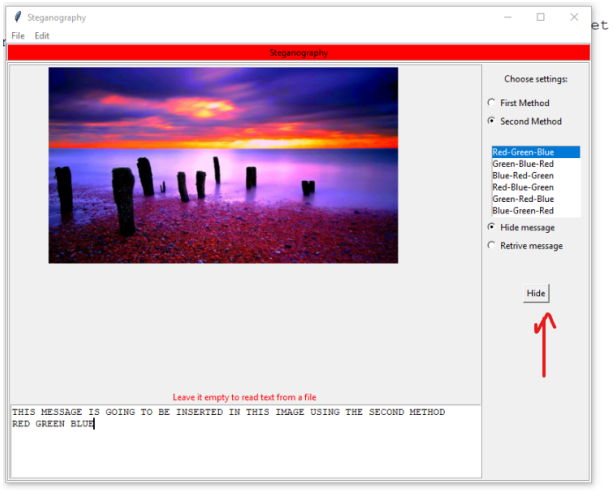


3) Now you may start by choosing which method you will use (1^{st} or 2^{nd}).
4) Choose whether you will hide or retrieve message .

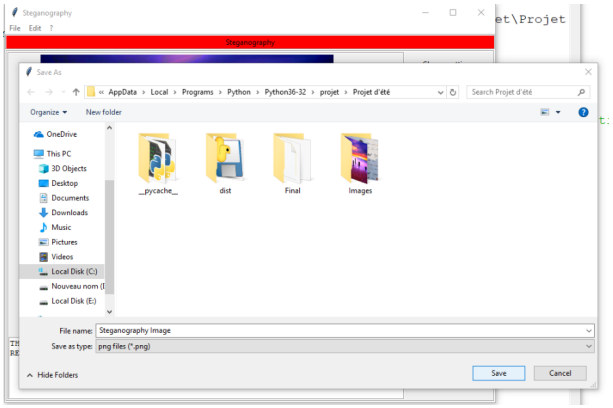


5) If you are going to hide information :
   a) Open cover image.

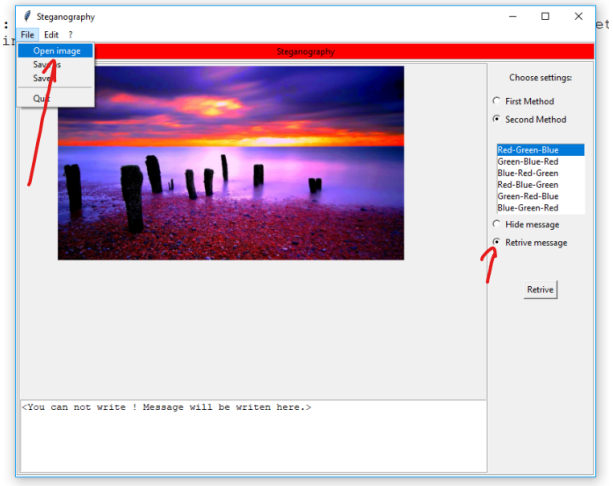
   b) Write message in the text box (or leave it empty to read from a file).
   c) Press the Hide button .



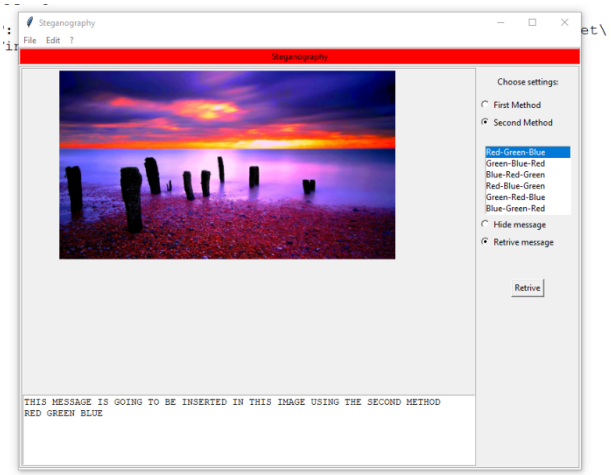
   d) Save the Steganography image with a new name or the same name to replace the old one.



6) If you are going to Retrieve information :
   a) Open the Steganography image.
   b) Select the right method of Steganography.

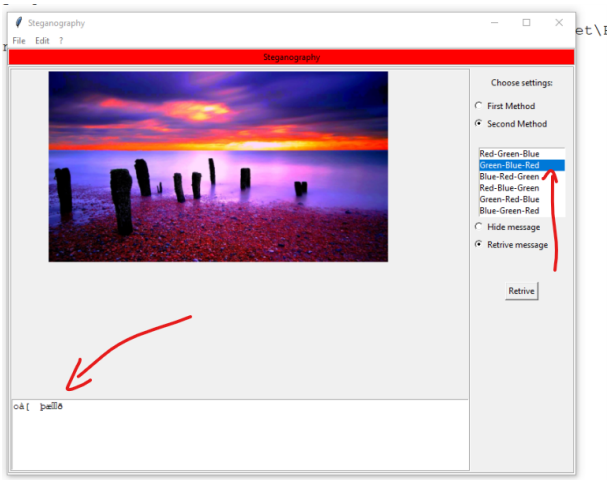


   c) Press the Retrieve button :



The message should be written in the text box.If the message is too long, it will be stored in a text file which you will choose its name.

If you choose a wrong way to retrieve the message, random and unreadable characters will be retrieved.



Please note:

Several menu-buttons are still under construction.

The Program may get as an input a ".bmp" or ".png" files, but as an output it creates a ".png" file.

III. Conclusion:

These two methods assure a data transmission without loosing any informations.

The first method can be implemented in 3 ways depending on the way of distributing the bits :

2 Red 3 Green 3 Blue or

3 Red 2 Green 3 Blue or

2 Red 3 Green 2 Blue.

The second method can be implemented in 6 ways depending on the order of color plans:

Red Green Blue, Red Blue Green,

Green Blue Red ,Green Red Blue ,

Blue Red Green and Blue Green Red.

Those 9 ways were implemented in our program.

Deficiencies

- The number of ways is not large, anyone can brute force trying the 9 ways and he will get the secret message.
- The size of text is limited :
  - 1^{st}method each character in a pixel so you can insert at maximum the number of pixels characters
  - 2^{st}method let N be the number of pixels of the image :
    you are allowed to insert $\frac{3 \times 3 \times N}{8}$ character in the image.
- The message will be destroyed if the 3 first plans of bit of every colour were changed.

Finalization There are many other methods of Steganograpy. If the message is needed partially a DCT or DWT would be a better alternative, since the information is distributed in the whole image and it could be always retrieved partially.

References

[1] M. M. Amin, M. Salleh, S. Ibrahim, M. R. Katmin, and M. Shamsuddin, "Information hiding using steganography," in *Telecommunication Technology, 2003. NCTT 2003 Proceedings. 4th National Conference on*. IEEE, 2003, pp. 21–25.

[2] S. Jiao and R. Goutte, "A secure transfer of identification information in medical images by steganocryptography," *International Journal of Communications, Network and System Sciences*, vol. 3, no. 10, p. 801, 2010.

[3] S. Katzenbeisser and F. Petitcolas, *Information hiding techniques for steganography and digital watermarking*. Artech house, 2000.

[4] D. Stanescu, M. Stratulat, B. Ciubotaru, D. Chiciudean, R. Cioarga, and M. Micea, "Embedding data in video stream using steganography," in *Applied Computational Intelligence and Informatics, 2007. SACI'07. 4th International Symposium on*. IEEE, 2007, pp. 241–244.

[5] M. Chen, R. Zhang, X. Niu, and Y. Yang, "Analysis of current steganography tools: classifications & features," in *Intelligent Information Hiding and Multimedia Signal Processing, 2006. IIH-MSP'06. International Conference on*. IEEE, 2006, pp. 384–387.