

**IMAGE STEGANOGRAPHY**

**Submitted to**: Mrs. Saroj Bala

**Submitted by:** Ashish Gupta

2k16/MC/023

October 2018

**Certificate**

This is to certify that the report entitled, “**Image Steganography”** has been successfully completed by:-

**Ashish Gupta(2K16/MC/023),** student of Mathematics and Computing Branch, Delhi Technological University under the guidance and supervision of **Mrs. Saroj Bala** during the 5th Semester Session 2018-19.

­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Saroj Bala** **Ashish Gupta**

**Acknowledgement**

The success and final outcome of this project required a lot of guidance and assistance from many people and I am extremely privileged to have got this all along the completion of my project. All that I have done is only due to such supervision and assistance and I would not forget to thank them.

I would like to express my sincere gratitude to Mrs. Saroj Bala for giving me an opportunity to work on this project as well as guiding me to prepare this report without any errors. I am extremely thankful to her for providing such a great support and guidance, although she had busy schedule managing other affairs.

I would also like to thank *Intel Corporation and Willow Garage* the cofounders of OpenCV for their pioneering work in Computer Vision and free open source library that made this report possible. This report would not have been possible if not for their invaluable resource.

Finally, I would like to thank my friend Ishika Gupta for helping with the images in the report and laying out plan to complete this project on time.

**Table of Contents**

**Abstract 5**

**Introduction6**

**Data Hiding7**

**Image Steganography8**

**Theoretical Discussion9**

About LSB encryption**10**

Algorithm**11**

Embedding and Extraction Algorithms**12**

**Programming Implementation13**

**Future Research and Scope15**

**Conclusion16**

**Appendices17**

1. ASCII**20**
2. Images**21**

**Reference 23**

Abstract

This report deals with digital images, how they are represented in computers and *image steganography*, the process of hiding secret message with a larger size image in such a way that no one can know the presence or content of the hidden message. The report goes over theory of steganography, the algorithms of embeding and extraction of data and images. Finally the report goes over a python based implementation with an image. The report aims at introducing steganography as a medium to maintain *secret communication* between two parties.

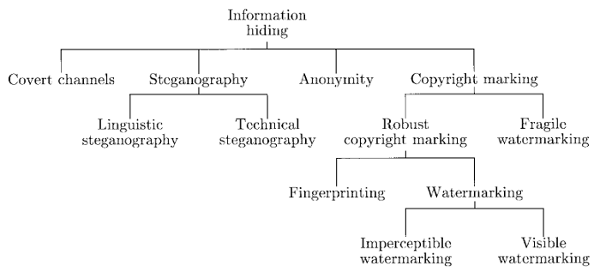
Introduction

The report is divided into 4 parts. In the first part we will discuss what is and the need for image steganography along with the popular methods out there. A discussion in made on which method to use when and why to use it. A look into other methods is also mentioned briefly.

The second part of the report deals with the LCB encryption of images. LCB stands for **least significant bits.** We show how it is the fastest form of encryption. All the mathematics and images involved are thoroughly explained using diagrams and images. The theory includes what are images? How are images similar to giant matrices and why computers are so effective with images? We will then seek out how to hide data into images. We will see how to make data invisible to the naked eyes and undetectable using binary, a heart of digital computing.

The third part deals with implementation using Python, a data science and image processing power house, which owes much of it’s powers into image processing from OpenCV. A brief discussion on OpenCV and python is made. We will see how the images are manipulated as matrices using simple loops and how the process ends up consuming computation power. We will encrypt entire **Harry Potter and the Sorcerer’s Stone** into one image

We end the discussion with the modern day problems into data hiding and how this technique of data hiding is already transforming millions of people’s lives. We will see how copyright laws are enforced easily using this technique and why medical imaging of brains owes its origins to this technique. Later at the end everything is summarized in conclusions and future research into the topic is highlighted along with various sources for study.

**Data Hiding**

Data Hiding is a very ancient art. It is the process of hiding a secret message within a larger one in such a way that someone cannot know the presence or contents of the hidden message.

* Caesar cipher.
* Egyptians used symbolic language in their pyramids.
* Coded Language.
* Writing with invisible ink.

With the dawn of the Digital World, now just the methods have changed, but the aim is still the same.

In Modern Times, Data Hiding is associated with digital forms such as cryptography, steganography, and watermarking.

1. **Cryptography** is obscuring the content of the message, but not the communication of the message.
2. **Steganography** which is Greek for “covered writing” is hiding the very communication of the message.
3. **Watermarking** attempts to add sufficient metadata to a message to establish ownership, provenance, source, etc.

A natural question to ask is why data hiding is so necessary. There are numerous reasons:

* Personal, Private Data.
* Sensitive Data.
* Confidential Data, Trade Secrets.
* To avoid Misuse of Data.
* Unintentional damage to data, human error, accidental deletion avoidance
* Monetary and law purposes.
* Hide Traces of a crime.
* As a backup.

**Image Steganography**

**Steganography** is also the process of hiding a secret message within a larger one in such a way that someone cannot know the presence or contents of the hidden message. Although related, Steganography is not to be confused with Encryption, which is the process of making a message unintelligible—Steganography attempts to hide the existence of communication.

The basic structure of Steganography is made up of three components: the “carrier”, the message, and the key. The carrier can be a painting, **a** **digital image**, an mp3, even a TCP/IP packet among other things. It is the object that will ‘carry’ the hidden message. A key is used to decode/decipher/discover the hidden message. This can be anything from a password, a pattern, a black-light, or even lemon juice.

Thus in Image Steganography following are the 3 components:

1. **Carrier:** A digital image.
2. **Message:** Any text file or dataset will do that is <1/4th of size of image.
3. **Key:** LSB encryption and decryption.

The properties of Image Steganography may be substituted with audio mp3’s, zip archives, and any other digital document format relatively easily. Hence this report becomes a ground work into future exploration. It is impossible to tell the difference between a **carrier image** and **normal image** with naked eyes.

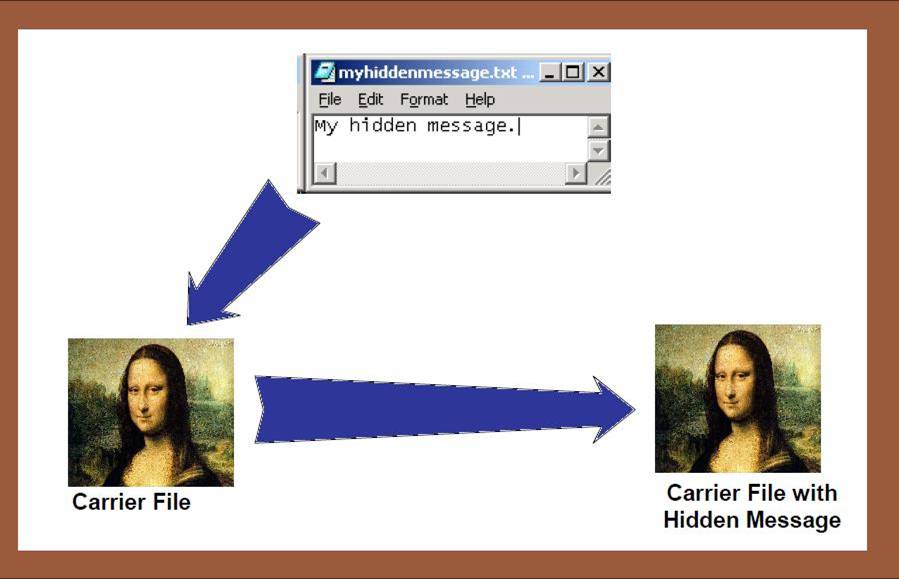


Image: depicts image steganography. Exactly same to eyes.

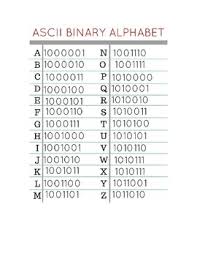
**Theoretical Discussion**

A digital image is described using a 2-D matrix of the color intestines at each grid point (i.e. pixel). Typically grey images use 8 bits(XXXX-XXXX), whereas color utilizes 24 bits to describe the color model, such as RGB model(8bits-8bits-8bits). The Steganography system which uses an image as the cover, there are several techniques to conceal information inside cover image. The spatial domain techniques manipulate the cover image pixel bit values to embed the secret information. The secret bits are written directly to the cover image pixel bytes. Consequently, the spatial domain techniques are simple and easy to implement. The Least Significant Bit (LSB) is one of the main techniques in spatial domain image The LSB is the lowest significant bit in the byte value of the image pixel. The LSB based image steganography embeds the secret in the least significant bits of pixel values of the cover image (CVR).

The concept of LSB Embedding is simple. It exploits the fact that **the level of precision in many image formats is far greater than that perceivable by average human vision.** Therefore, an altered image with slight variations in its colors will be indistinguishable from the original by a human being, just by looking at it. In conventional LSB technique, this requires eight bytes of pixels to store 2 bytes of secret data.

Here are the steps involved in the steganography. It is important to remember the simplicity of the model makes it very powerful to use.

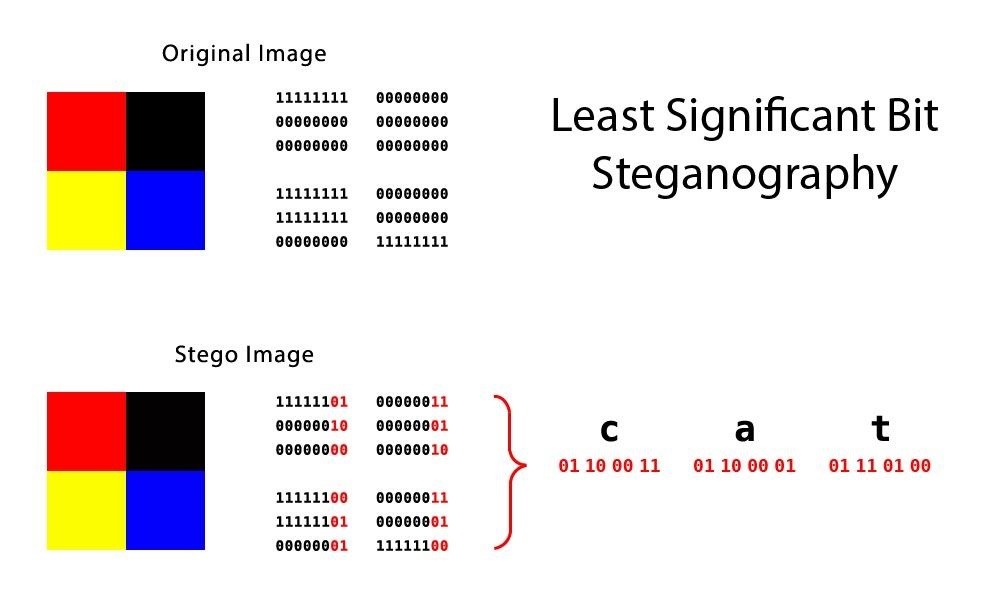
**Step 1: Convert the message to binary.** A message is converted to binary by using ascii table in Computer Science. A reference has been mentioned in the appendix 1 of this report. Here is a list of characters with ascii in binary:



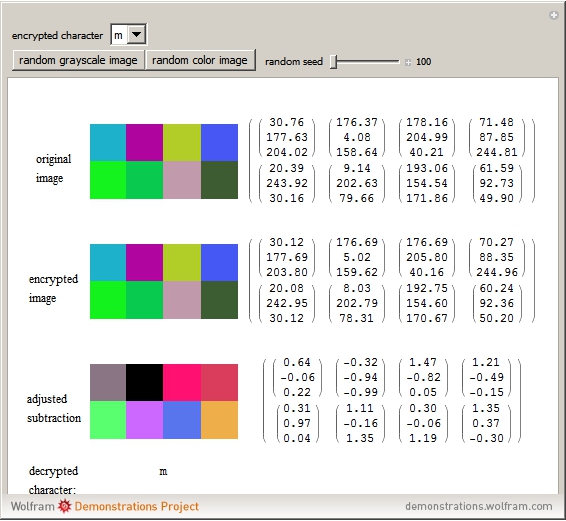
Hence the message “**This is hidden**” transforms to – **0101010001101000011010010111001100100000011010010111001100100000010010000110100101100100011001000110010101101110**

Note that a space has a separate binary number.

**Step 2: Encode the above message to image into the last 2 bits of each number.** So if the image is made of numbers, then those numbers range from 0-255 (only for images into computers.

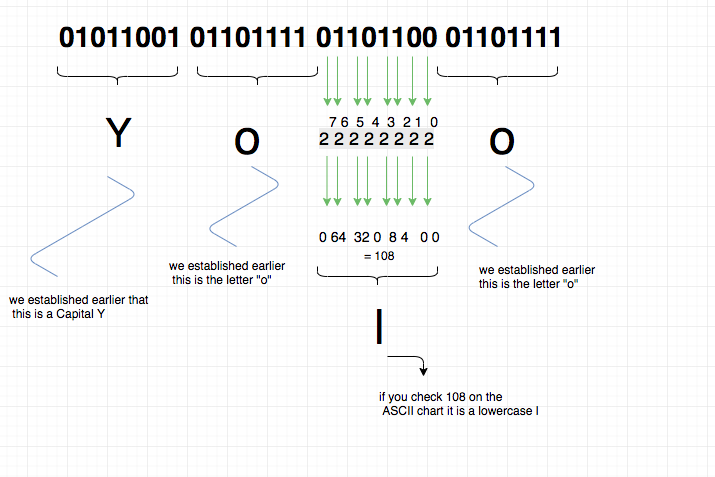


[Above] The red digits are the encrypted letters. See that 3 patterns are linked to 1 image due to 8bit-8bit-8bit nature of images.

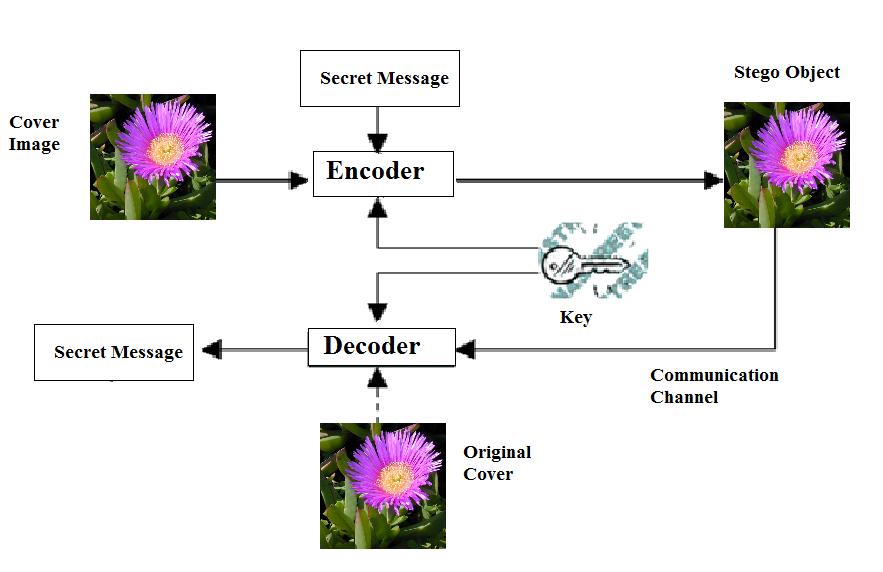


[Above] In implementation, numbers are altered to +/- 3 their values. Human eyes are not sensitive enough to notice change of +3 or -3 in this.

**Step 3: The image steganography is done.** Now the information can be reversed back from the image using the reverse technique.Take the last 2 bits of every number into image and convert it back to binary.

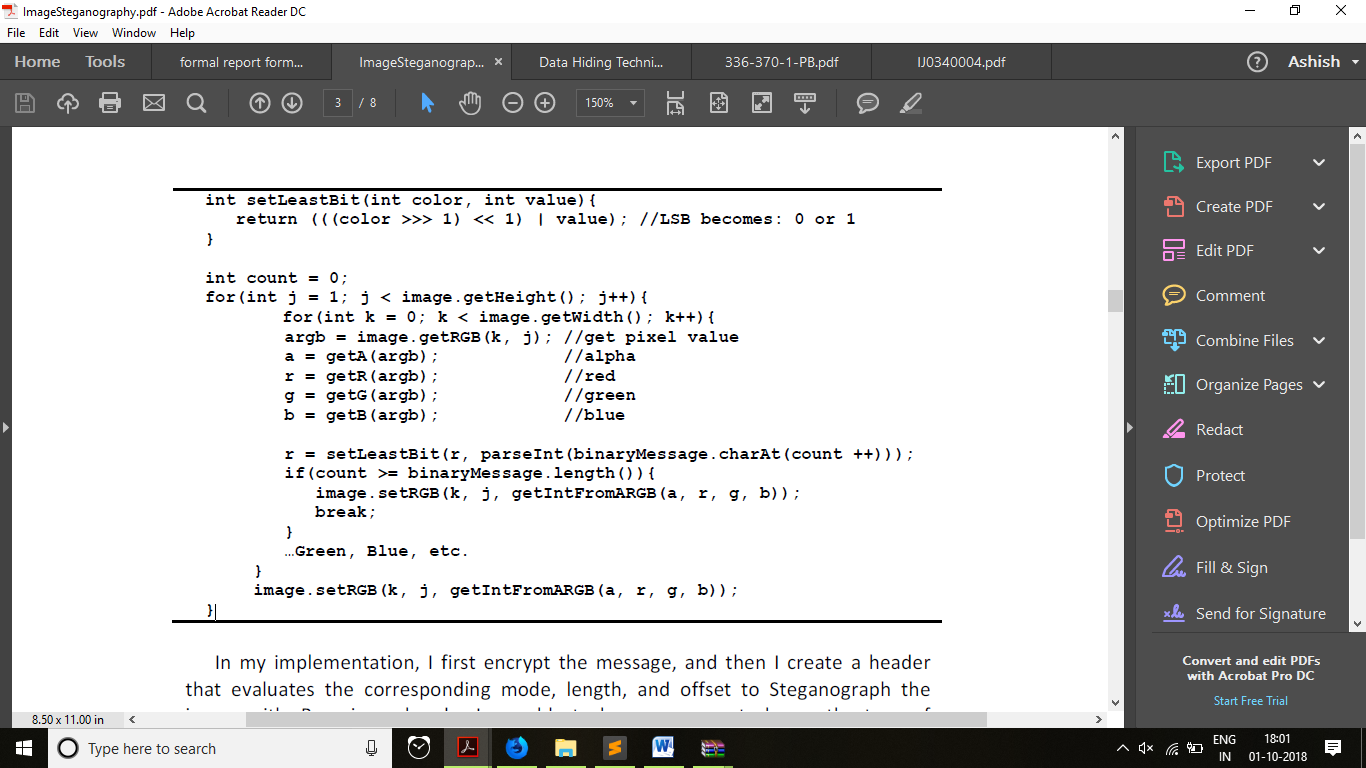


A simple demonstration of the conversion



[Above] To summarize the above information this is what it looks like.

# Algorithm:



**Embedding Algorithm (An explanation)**

In this process of encoding method, a random key is used to randomised the cover image and then hide the bits of a secret message into the least significant bit of the pixels within a cover image. The transmitting and receiving end share the stegano key and random-key. The random-key is usually used to seed a pseudo-random number generator to select pixel locations in an image for embedding the secret message.

**Inputs:** Cover image, stegano-key and the message

**Output:** stegano image

**Instructions**

1) Read character from text \_le that is to be hidden and convert the ASCII value of the character into equivalent binary value into an 8 bit integer array.

2) Read the RGB colour image(cover image) into which the message is to be embedded.

3) Read the last bit of red pixel.

4) Initialize the random key and randomly permute the pixels of cover image and reshape into a matrix.

5) Initialize the stegano-key and XOR with text to be hidden and give message.

6) Insert the bits of the secret message to the LSB of the Red plane's pixels.

7) Write the above pixel to Stegano Image File.

**Extraction of Hidden Message**

In this process of extraction, the process takes the key and then random key. These keys take out the points of the LSB where the secret message is randomly distributed. Decoding process searches the hidden bits of a secret message into the least significant bit of the pixels within a cover image using the random key. In decoding algorithm the random-key must match i.e. the random-key which was used in encoding should match because the random key sets the hiding points of the message in case of encoding. Then receiver can extract the embedded messages exactly using only the stegano key.

**Input:** Stegano image, stegano key, and random key.

**Output**: Secret message.

1) Open the Stegano image in read mode and from the image, read the RGB colour of each pixel.

2) Extract the red component of the host image.

3) Read the last bit of each pixel.

4) Initialize the random-key that gives the position of the message bits in the red pixel that are embedded randomly.

5) For decoding, select the pixels and Extract the LSB value of red pixels.

7) Read each of pixels then content of the array converts into decimal value that is actually ASCII value of hidden character.

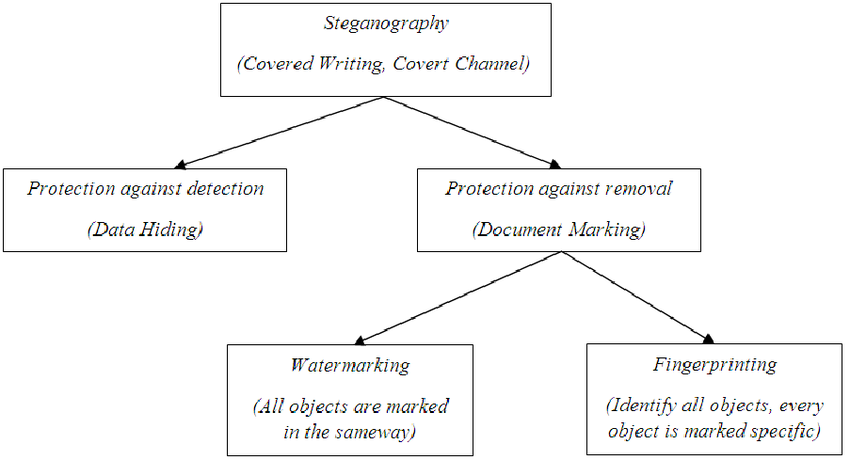
8) ASCII values got from above is XOR with stegano key and gives message, which we hide inside the cover image.

# Compression formats of images

According to Steganography, the secret message which is hidden may result in a distortion less image. At the same time this distortion will be perceptible to the naked eye. The quantity of information invisibly hidden in the image resulting in a distortion less image plays a pivotal role and this is decided by algorithm. The required characteristics are assesed while choosing a specific file format for Steganography as shown in Table

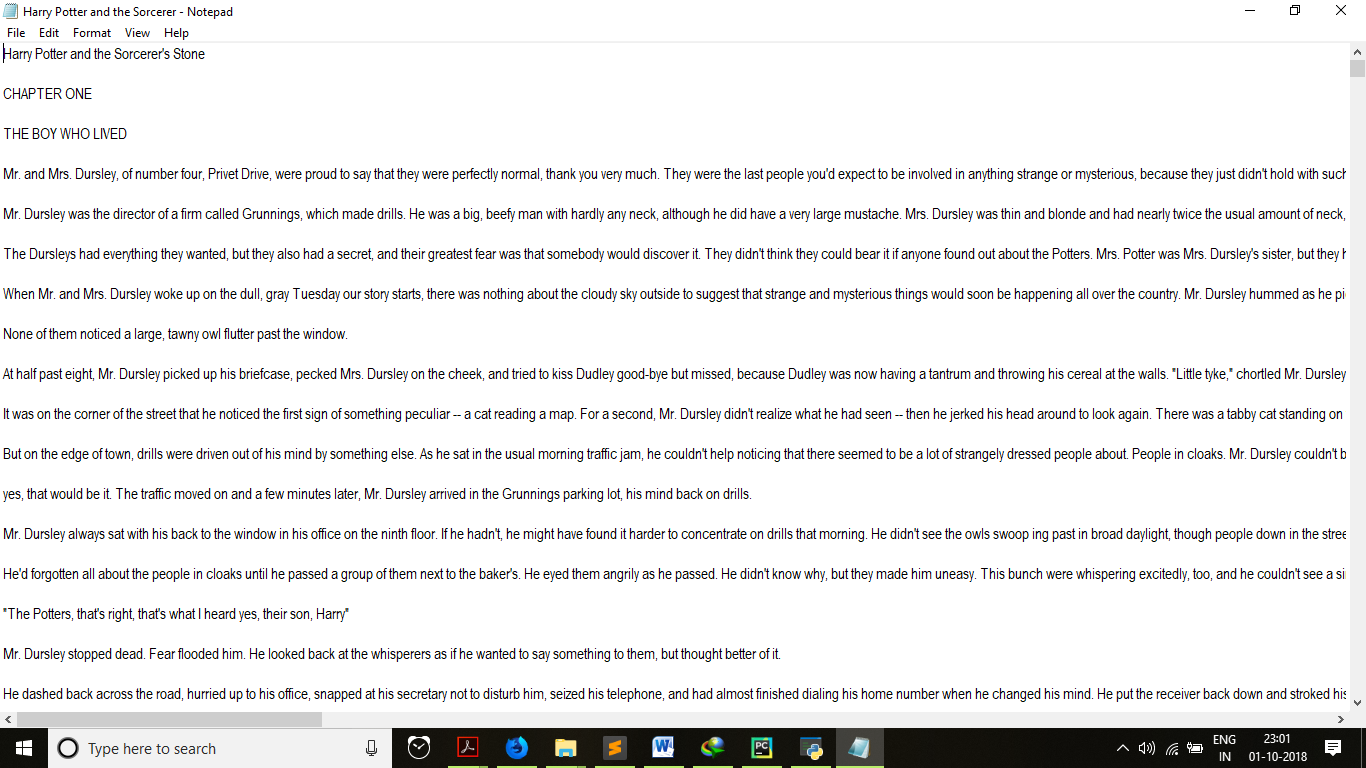
|  |  |  |  |
| --- | --- | --- | --- |
| **Comparison of LSB Method for Various Image File Formats Characteristic** | **LSB in**  **GIF** | **LSB in**  **PNG** | **LSB in**  **BMP** |
| Quantity of hidden data | Medium | Medium | High |
| Independent of file format | Low | High | Low |
| Steganalysis detection | Low | Low | Low |
| Image manipulation | Low | Low | Low |
| Percentage Distortion less resultant image | Medium | High | High |
| Embedding capacity | Medium | Medium | High |
| Invisibility | Medium | Medium | High |

**Some more applications of compression (Involves data hiding)**

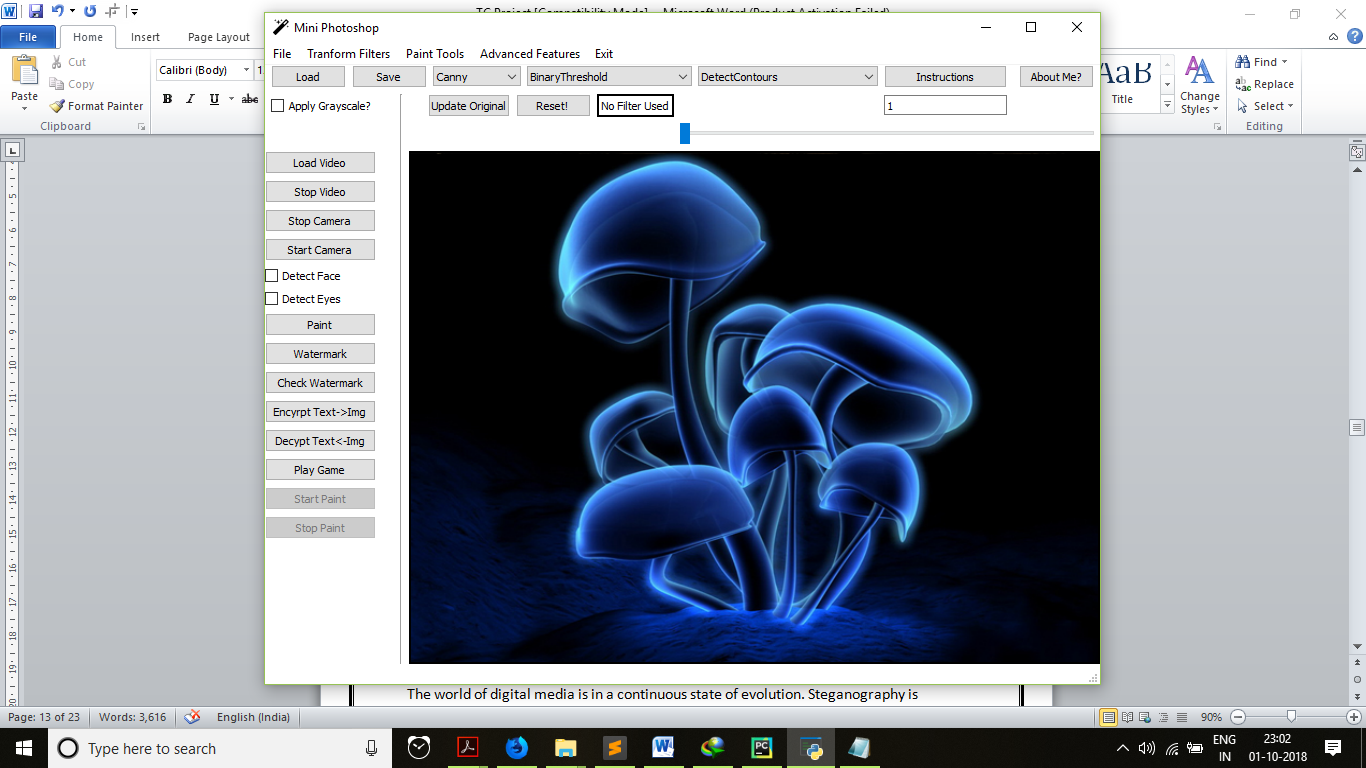


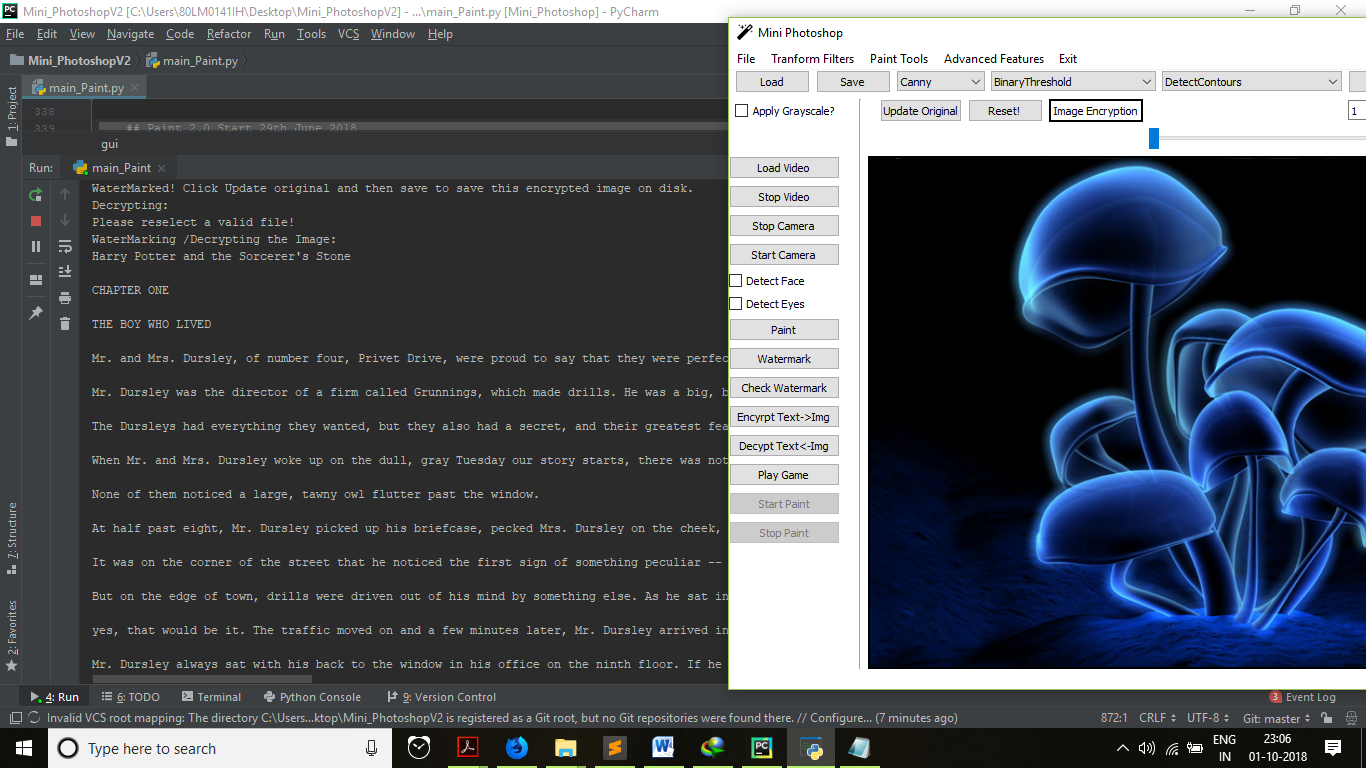
Programming Implementation

This implementation is a part of coding project undertaken while preparing this report. The code encodes entire text of **Harry Potter and the Sorcerer’s Stone** into one image.



The text and the software that was built along with image.





Generated output from the image! Python code.

**Future research and scope**

The world of digital media is in a continuous state of evolution. Steganography is regarded as technology that has major competitive applications. While a significant progress in the image steganography techniques has been achieved, still there is scope for the improvement as there is yet to be evolved a standard method and the proposed algorithms can be further enhanced. In this thesis, the study and analysis related to the image based steganography relating to LSB and DCT has been done.

The enhancements can be done by using soft computing techniques such as Neural based steganography, Fuzzy and Genetic algorithms based approaches. The future work can also take into considerations of the Quantum computation approaches which can extend the classical steganography for performance enhancement of the existing techniques. The existing transform and spatial domain based approaches can be enhanced with certain variations. The DCT and DWT techniques can also be enhanced by using randomization approach where the secret bits can be embedded randomly selected blocks. Additionally, improving the embedding capacity of these methods that can withstand severe compression can be considered. In the spatial domain enhancement can also be done.

The LSB based random embedding where the secret data is embedded only in red plane can be enhanced using two planes for embedding (Red, Green or blue plane) that will increase the embedding capacity and will also preserve the security. The embedding capacity can also be enhanced by using more LSBs and maintaining the statistical properties of the images.

Conclusion

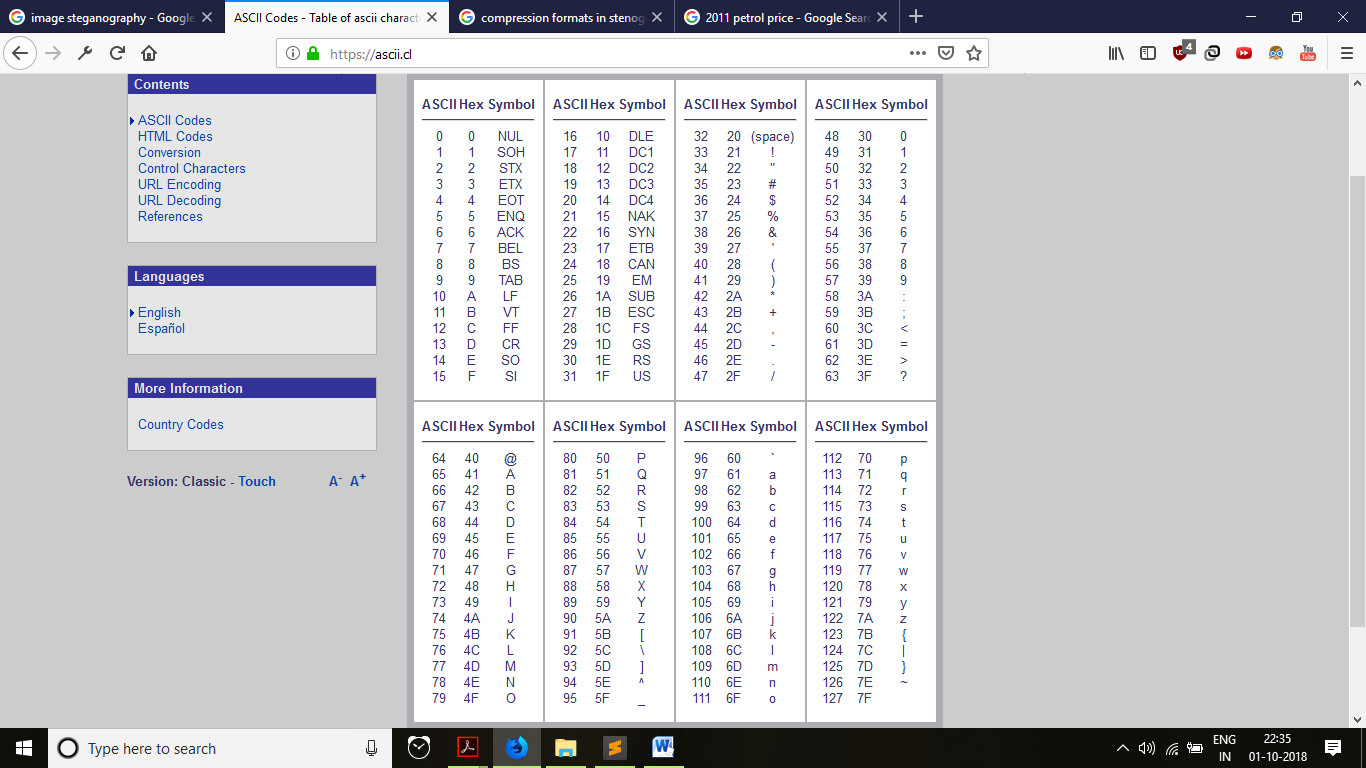
The proposed method is very useful technique for secure communication over the Internet. In the process of Steganography, the message which is hidden is invisible. An attempt has been made to implement encryption and decryption techniques on the data to be hidden into the carrier files, so that this will provide additional security to the data. The sender and receiver only know how to hide and unhide the data into the carrier files. No other intermediate person will even know that there is a second message inside the carrier file. The sender and receiver only know the commands to hide and unhide.

Since it utilizes lossless compression, LSB makes utilization of lossless image. To have the capacity to conceal secret information within this image, one requires a substantial cover medium. The advantage of LSB hiding is its simplicity. LSB embedding technique also allows high perceptual transparency. The data hiding capacity of LSB technique is high and more secure. Embedding secret information with Steganography technique decreases the probability of secret information being detected. LSB insertion method to image Steganography works effectively for 24 BMP, GIF and PNG image file formats. Using this embedding and extracting algorithms, one can extract the secret message exactly as original message without changing the cover image.

The future research in Image steganography is limitless. While some areas require secure transmission of images, others require storing potentially 2 or 3 times more data.

Appendix 1: ASCII

ASCII stands for American Standard Code for Information Interchange. Computers can only understand numbers, so an ASCII code is the numerical representation of a character such as 'a' or '@' or an action of some sort. ASCII was developed a long time ago and now the non-printing characters are rarely used for their original purpose. Below is the ASCII character table and this includes descriptions of the first 32 non-printing characters. ASCII was actually designed for use with teletypes and so the descriptions are somewhat obscure. If someone says they want your CV however in ASCII format, all this means is they want 'plain' text with no formatting such as tabs, bold or underscoring - the raw format that any computer can understand. This is usually so they can easily import the file into their own applications without issues. Notepad.exe creates ASCII text, or in MS Word you can save a file as 'text only'

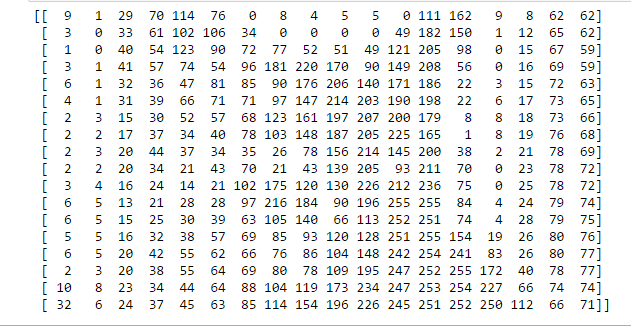


Appendix 2: Images

We all love to see beautiful images, but have you ever thought how do computers see an image? In this blog post, I will give an explanation of how images are stored in a computer.



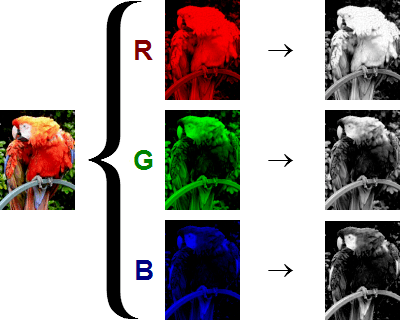
Consider the above image, a normal human can easily tell that, there is a cat in the image. But, can computers really see the cat? The answer is no, computers see a matrix of numbers (between 0 to 255). Broadly, we can classify images as grayscale images or color images. First of all, I will discuss grayscale images then color. Above image is a grayscale image means each pixel represents the brightness of a pixel. Know more about pixel. Let me first show you what computers see in the case of above image.



I've resized above image to 18 \* 18 in order to make it easy to understand. Unlike us, the computers see an image as a 2D matrix. You might have heard people saying that this image is of size 1800\*700 or 1300 \* 700. This size shows width and height of an image. In other words, if size is 1300 \* 700 then there are 1300 pixels horizontally and 700 vertically. That means there are total 910000 (1300\*700) pixels. If the image is the size of 700 \* 500 then the dimensionality of the matrix will be (700, 500). Here, each element (pixel) in the matrix represents the intensity of brightness in that pixel. Here, 0 represents black and 255 represents white color.

**Color Images**

In grayscale images, each pixel represents the intensity of only one color. In other words, it has one channel. Whereas in color images we have 3 channels RGB (red, green, blue). Standard digital camera will have 3 (RGB) channels.



As you can see in above image, the color image is composed of three channels red, green and blue. Now the question is, how do computers see this image? Again, the answer is they see the matrix. now the next question should be, how do we represent this image in the matrix since it has 3 channels, unlike grayscale images where we had only one channel. In this case, we will have 3D matrix. We have one matrix for one channel, but in this case, we'll have three matrices stacked onto each other, that are why it's 3D. Dimensionality of 700 \* 700 color image will be (700, 700, 3). Let's say, the first matrix represents red channel, then each element of that matrix represents an intensity of red color in that pixel, likewise in green and blue. In general, each pixel in color image has three numbers (0 to 255) associated with it. These numbers represent intensity of red, green and blue color in that particular pixel.

References

## Book​

|  |  |  |
| --- | --- | --- |
|  | **Zed Shawn, Learn Python the Hard Way: A Very Simple Introduction to the Terrifyingly Beautiful World of Computer, edition**3**(USA: Pearson).**  ​ Learning OpenCV, Book by Adrian Kaehler and Gary Bradski (New York: O Riely & Sons, 2008). |  |

## Report

|  |  |  |
| --- | --- | --- |
|  | IEEE International, IMAGE STEGANOGRAPHY USING LEAST SIGNIFICANT BIT WITH CRYPTOGRAPHY (Journal of Global Research in Computer Science, Volume 3, No. 3, March 2012). |  |

## Website

|  |  |
| --- | --- |
|  | ​Bassett, Vicki, Future of Image Processing shodhganga.inflibnet.ac.in/bitstream/10603/41637/10/10\_chapter%203.pdf  Nick Nabavian CPSC 350 Data Structures: Image Steganography  [nabav100@chapman.edu](mailto:nabav100@chapman.edu)  Slide Share Corporation:  https://www.slideshare.net/SreelekshmiSree1/image-steganography-using-lsb  https://www.slideshare.net/prashant3535/data-hiding-techniques?from\_action=save |

## Blog

|  |  |
| --- | --- |
|  | ​How do computers see images?  https://www.commonlounge.com/discussion/244616b76d3d40f88e8f12103a22743d  LSB techniques by Reshobha  http://shodhganga.inflibnet.ac.in/bitstream/10603/76441/16/16\_chapter%207.pdf |